Design and Development Three-Tier J2EE application

“Personal Tutor System”

Mr Nutawoot Wootinunsulasit
BSC in Computer Science 2004
Abstract

This project involves the design and implementation of the personal tutor reporting and recording system. The software application will be designed to capture the conversation record between personal tutor and students and provide appropriate tool to assist their work.

The technologies which will be used for implementation on this project are based on Java Enterprise Edition platform. Therefore, it should give the reader a very clear picture how each of these technologies components operate. The database is also designed in an optimally normalise form and will be implemented using MySQL.

The Data Protection Act must be also influence the security aspect of the design because the software will be used to store student information and must be kept secure at all time.

Alternative design and technique are investigated on this project in order to produce the best design that would be easier to maintain and extensible for further development. The user interfaces of this software are designed base on HCI principle so that the software is usable and accessible by anybody.
Acknowledgements

I would like to thank Dr Alwyn Barry for agreeing to supervise my project and I would to thank you all my friends and family for giving such a great support.
# Table of Contents

1 **Introduction**....................................................................................................................1  
1.1 Title ................................................................................................................................1  
1.2 Description .......................................................................................................................1  
1.3 Motivation for project ......................................................................................................2  
1.4 Orientation Summary .....................................................................................................2  

2 **Literature Review** ..........................................................................................................3  
2.1 Introduction ....................................................................................................................3  
2.2 Software Development process .......................................................................................3  
2.3 Requirement Engineering process ...................................................................................6  
2.4 Software Architecture ....................................................................................................8  
2.5 Security Authentication ..................................................................................................9  
2.6 Security Authorization ...................................................................................................9  
2.7 Choosing the right technology ........................................................................................10  
2.8 Relational Database Design ............................................................................................16  
2.9 Choosing the right database ............................................................................................17  
2.10 Data Protection Issue ....................................................................................................19  
2.11 Conclusion ...................................................................................................................21  

3 **Requirement Analysis of Personal Tutor System** .........................................................22  
3.1 Introduction ....................................................................................................................22  
3.2 Requirement Elicitation ................................................................................................22  
3.3 Functional Requirements ..............................................................................................25  
3.4 Non-Functional Requirements ......................................................................................31  
3.5 Conclusion .....................................................................................................................32  

4 **Enterprise Java Beans** .....................................................................................................33  
4.1 Introduction ....................................................................................................................33  
4.2 The enterprise bean component .....................................................................................33  
4.3 Conclusion .....................................................................................................................36  

5 **Database Design** ............................................................................................................37  
5.1 Introduction ....................................................................................................................37  
5.2 Entity ................................................................................................................................37  
5.3 Relationship ...................................................................................................................39  
5.4 Normalisation ................................................................................................................41  
5.5 Implementation .............................................................................................................41  
5.6 Conclusion .....................................................................................................................41  
5.7 XML and Database .......................................................................................................42  
5.8 Conclusion .....................................................................................................................44
1 Introduction

1.1 Title
The development a scalable three-tier Personal Tutor Reporting and Recording system using J2EE

1.2 Description
At present, the useful way of gathering information of the students such as personal information during their academic years is by recording meetings in a paper-based format and putting them together into a student files. These student files are kept at the department and can only be accessed by some academic staff at the University. For example, information like a conversation note between a student and his/her personal tutor which is private and personal for this particular student will also be recorded in a piece of paper and will eventually go into a student file. However, the process of organising and putting this information into a student file is not straightforward because it requires the information to be passed from one person to the other before eventually entering the student file. This confidential information such as the above has to be passed from the personal tutor to other academic staff such as the Director of Study, to review the student problem and take any further actions in order to help the students solve the problem. Once this action has been taken and the problem is solved then the information can eventually be stored into a student file. This method of reviewing information is not convenient and often not very effective. The Director of Studies would write down the problem of a student that has been emailed by Personal Tutors into the Director of Study’s notebook in order to review and provide a set of actions. Once these actions have been completed then the Director of Studies may email it back to personal tutor in order to take any further actions or putting them back in the student files and so on. These problems make me suggest that it would be important to have a system to help recording and organising this student information and also providing the capability to access and modify the data securely and effectively. Therefore, I am proposing to develop a Personal Tutor Reporting and Recording system as my final project work. This is a software development project and I shall adopt a software process—a coherent set of activities that I shall follow in order to deliver a software product. I will be using a software development tools such as Use Cases, UML notation during the software development process.

As the system would be developed and deployed using Enterprise Java Beans sever-side components such as entity, session and message-driven beans, I need to learn these fundamental concepts as a first step before doing the design and implementation of the system. One of the aspects that I also have to investigate further is a resource management within EJB server itself, which is a fundamental part of deploying application using EJB servers. The resource management covers concepts such as instance pooling, concurrency, transactions, persistence, JNDI as a directory look up and security mechanisms that support authentication, access control and secure communication through Secure Socket Layer(SSL) protocol – and some mechanism for authentication such as via LDAP, Table Database so on. Although EJB applications themselves are portable, the manner in which we install and run EJB products varies widely from one vendor to another so I have decided to use JBOSS as an application server to deploy the personal tutor system. This system will be developed as a web based application it should then integrate with a database. So I
will consider using either using MySQL-Max version 4.0 or PostGreSQL, which are open source relational database management system (DBMS). These DBMS supports transaction mechanisms such as roll back and commit. It also easy to install and set up and also supports SQL and JDBC drivers. However, any database that is intended for server use will work with for example ORACLE or HypersonicSQL. Since most effective J2EE applications are written using container managed persistence (CMP), most applications will not need to write a single line of SQL. Unless we plan on optimising our application by tweaking the EJB container (through custom deployment descriptors), or plan on writing all database access code ourselves, we can use any Database or change to commercial database in future.

This web based application need to run on a number of web browsers such as Internet Explorer, Netscape, Mozilla and so on. This application may also run on different platforms and this could present a problem within the design of the web-tier interface. I will consider using an off-the-shelf JSP based web application framework call the Apache Struts Framework that should cope with our cross platform design constraints and also allow us to organise our code in MVC manner and reuse EJB components. Not only choosing the framework for our application is important, I will also consider investigating and researching ways in which users may interact with the system for example, how users would enter information from the screen, are they using a drop down menu?, how users generate a set of actions?, and so on. The interesting point here is that we may introduce an XML mark-up language to describe the information that the users entering into the system and allow certain parts of the information to be seen or accessed by other users. I will need to investigate further on this method. I will have to apply some principle of computer human interaction that I am going to study in Semester 2 in order to design an effective GUI for web-based application.

1.3 Motivation for the project
This project does not represent a detailed research investigation a particular subject area or have direct relevant relevance to a particular academia unit. Instead it allows students to explore and implement complex real world problem by using some real world scenarios and involving tradeoffs between different technologies. Moreover, Enterprise Java Beans, JBOSS and Web application technologies are the key for getting job important technologies when applying for jobs after graduate graduation, and especially for IT industrial therefore, this would be a useful topic to get hands-on experience from the project.

1.4 Orientation Summary
The report begins with a literature review which explores the structure and technologies of the system that will be implemented and discusses the problems. Because this system is new and has never been implemented before all the case studies were only included to highlight those possible issues that may or may not causes problems in the actual trading environment, which the system is designed to support.

The software process used to gather the system requirements is detailed within the next chapter as a result a set of data flow models is then generated. There will be also a discussion regarding the various available technologies which the system could be implemented. The next chapter details the development of the database design and the discussion that follows which must be taken into account is to deliver an effective user interface. The following chapter then details the basic functionality that will be achieved by the system. And finally we reflect upon details on of this project.
2 Literature Review

2.1 Introduction
My first action for doing my software development project is to undertake a literature search to establish how other people have tackled the same or similar problems, find out whether there have been any kinds of similar software and source code available around which I can study from it, what success they have achieved, what kind of technologies they were using and so on. Therefore, after complete writing this report I hope to achieve the following the aims of the literature search were:-

- To identify methods of software development process and requirements engineering processes;
- To identify important technologies and relevant issues that will be used for the project;

2.2 Software Development process
The first task that I would must be considered before writing software is to find out the problem and what sort of activities that the software will help clients to solve their problems and what goals they wish to achieve their goal. Therefore, to produce such software that meets our customer requirements we must have a set of activities to follow. This set of activities is called the “software development process”. Although there are many different software processes, there have share common activities that we should meet. These are:

1. **Software specification** – we should identify the functionality and constraints of the software.
2. **Software design and implementation** – to develop the software that meets software specification.
3. **Software validation** – The software should be validated that ensure that it does what customer want.
4. **Software evolution** – to develop the software that evolve to meet changing customer needs.

As I describe above there are many different software processes for us to follow and for many systems, of course unfortunately, there is no single software process that is appropriate in all situations used. This is also true and applies of my project. The most commonly used models are the waterfall model, evolutionary model, and spiral model.

2.2.1 The ‘waterfall’ model
The model involves the cascade from one phase to another, this model is known as the ‘waterfall model or software life cycle which illustrated in Figure 2.1. The fundamental development activities are:

1. **Requirement analysis and definition** – to find out system’s services, constraints and goals by consultation with system users. This requirement can be defined in detail and serve as a system specification.
2. **System and software design** – partitioning the requirement of hardware and software to be used for the system. It establishes overall the system architecture. Software design involves identifying software system abstractions and their relationships.
3. *Implementation and unit testing*—During this stage, the software design is realised as a set of programs and should involve unit testing to ensure that it meets its specification.

4. *Integration and system testing*—The individual set of programs are integrated and tested as a complete system to ensure that the software requirements have been met. After testing, the software system is delivered to the customer.

5. *Operation and maintenance*—Correcting errors which were not discovered in earlier stages of the life cycle, improving the implementation of system units and enhancing the system services as new requirements are discovered.

2.2.2 *The ‘evolutionary development’ model*

This model involves development is based on the idea of developing an initial implementation, exposing the implementation to the users, getting feedback and refining this through many versions until the system meets the customer requirements. Rather than have separate specification, development and validation activities, these are carried out concurrently with rapid feedback across these activities. This is illustrated in Figure 2.2.
There are two types of evolutionary development:

1. **Exploratory development** – the objective is to work with the customer to explore their requirements and deliver a final system. The development starts with the part that is well understood and adding evolves by adding new features as they are proposed by the customer.

2. **Throw-away prototyping** – objectives are to understand the customer’s requirements and hence develop a better requirement for the system. The prototype is developed to experiment with those parts of the customer requirements which are poorly understood.

### 2.2.3 The ‘Spiral’ model

Boehm (1998) proposed the spiral model (see Figure 2.3 the Spiral Model). The process is represented as a spiral. Each loop in the spiral represents a phase of the software process. The major difference between spiral model and other processes is a clear consideration of risk in the spiral model.

![Spiral model of the software process](image)

### 2.2.4 Incremental development

The incremental approach to development was suggested by Mills as a means of reducing rework in the development process and giving customers some opportunities to delay a decision on the full requirements of the system until they had have used a prototype. The incremental model involves the customer specifying an outline of the services they desire. Each increment delivered to the customer contains a sub-set of the desired functionality. In other words, the highest priority services are delivered first to the customer.

### 2.2.5 Comparison of the water fall, evolutionary, spiral and incremental models

The water fall model is appropriate if a full set of requirements can be established before the implementation phase. This is very important because any change of the new requirements would mean the process has to revisit each phase which is costly and time consuming, and therefore, it would not be suitable for this project.
The evolutionary approach is also similar, in that the full set of requirements has to be established. However, it is more effective than the waterfall model in producing systems which meet the immediate needs of customers. The prototype from each version can be developed during the software development. So the advantage of software that is developed based on the evolutionary approach is that the specification can be developed incrementally. However, this is again also time consuming and expensive in some cases.

The spiral and incremental model on the other hand does not require a full set of requirements to be established before the implementation phase. It is actually common for additional requirements to be identified by the customer once the system is in use. For the spiral model, this is very useful as the prototype is produced after each development phase can be used to resolve any uncertainties in the requirements and therefore reduce risk. However, mapping additional customer requirements onto appropriately sized increments for development is often a complex process and also during each iteration a thorough risk assessment must be carried out, which is time consuming and expensive.

For the incremental model, once the process is completed and delivered, customers can put it into services that mean that this allows them to take early delivery of part of the system functionality. They can experiment with the system which helps them clarify their requirements for later increments and for later version of the current increment.

**2.2.6 Findings**
The incremental model was selected for the development of the personal tutor system, and the most highest priority needed requirement needed by the customer will be implemented first and will be delivered as the first incremental.

**2.3 Requirement Engineering Process**
Part of the software development process is to do with gathering a set of requirements from the user; this is where the requirements engineering comes in. It is a process that involves all the activities required to create and maintain a system requirements document. There are four generic, high-level requirements engineering process activities. These are:-

1. *Feasibility study* – a short study of the system. The results of the feasibility study should be a report which recommends whether or not it is worth carrying on with the requirements engineering and system development process.

2. *Requirement elicitation and analysis* – this activity involves technical software development staff work with customers to find out about the application domain, what services the system should provide, the hardware constraints and so on. This activity also involves interacting with number of stakeholders who have directly or indirectly influence on the system requirements. Some
techniques for requirements elicitation and analysis may be used to carry out those tasks.

The process of requirement elicitation and analysis activities are:

2.1 Domain understanding Analysts must develop their understanding of application domain. For example, if a system for a Personal Tutor is required, the analyst must find out how personal tutor or director of studies will use the system.

2.2 Requirements collection this process is to interact with stakeholders in the system to discover their requirements. Analyst will develops understanding of their domain further.

2.3 Classification This activity takes the instructed collection of requirements and organises them into coherent clusters.

2.4 Conflict resolution when multiple stakeholders are involved requirements will conflict. This activity is concerned with finding and resolving these conflicts.

2.5 Prioritisation In any set of requirements some will be more important than other. This stage involves interaction with stakeholders to discover the most important requirements.

2.6 Requirement checking The requirements are checked to discover if they are complete, consistent and in accordance with what stakeholders really want from the system.

3. Requirement validation – concerned with showing that the requirements actually define the system which the customer wants.

4. Requirement management– to cope with the change of the requirement once the end-users have experience of a system.

2.3.1 Findings
In order to obtain the requirements for the system scenarios, Use-cases and UML diagram will be used. For a small system like this one it would be inappropriate to conduct the feasibility study because it will be time consuming. Requirement elicitation will be done at the very first stage by interviewing customers and develop a small set of use cases. For the validation of the requirement there will be a requirement document produced for completeness and correctness to be check by both developer and customer.

2.4 Software Architectures
Software architectures are concerned with decomposing a system into a set of interacting sub-systems. The software architecture then shows how these elements interact to form the final system. The architecture requirement for this project is to develop the three-tier architecture therefore; I would like to outline this architecture in more detail. (See Figure 2.4)

Figure 2.4 Three Tier Architectures
The first tier provides the user system interface or user services. These services include session handling, display information. The middle tier runs upon an application server like Tomcat, JBOSS, and SQL server and so on. The middle tier also provides the functional process logic; transaction management and process management also handle by the middle tier. The third tier provides database management regarded on databases vendors.

2.5 Security Authentication
A web application is expected to provide valuable services to its users, Therefore there will be a way to restrict access to a limited set of users, To do this the server needs to be able to authenticate the client – in other words, to make sure that the user is who they say they are. Let explore more on authentication methods available for Servlets technology. They are the following:-
- HTTP Basic
- HTTP Digest
- Form based
- HTTPS Client

We will cover the most commonly used methods are HTTP Basic and Form based authentication

2.5.1 HTTP Basic
The HTTP protocol provides the Basic authentication mechanism based on the username/password model. When a user requests access to a protected resource, the server responds by popping up a dialog box for them to enter the user information, which is sent back to the server in plain text. If the submitted username and password match the data in the server’s database then access is granted to that user.

2.5.2 Form based
The advantage of form-based authentication is that it allows us to write custom login and error pages using HTML or JSP. The pages we create are responsible for transmitting the username and password to the web container this is much more flexible than Basic authentication where this is all handled by the browser.

2.5.3 HTTPS Client Authentication
HTTPS client authentication is based on Public Key Certificates. This is a very strong authentication mechanism. It uses HTTP over SSL (HTTPS), in which the server and, optionally, the client authenticate each other with Public Key Certificates. Secure Sockets Layer (SSL) provides data encryption, server authentication, message integrity, and optional client authentication for a TCP/IP connection. We can think of a public key certificate as the digital equivalent of a passport. It is issued by a trusted organization, which is called a certificate authority (CA), and provides identification for the bearer. If you specify client-certificate authentication, the Web server will
authenticate the client using an *X.509 certificate*, a public key certificate that conforms to a standard that is defined by X.509 Public Key Infrastructure (PKI).

2.5.4 **Digest Authentication**
Digest authentication is similar to Basic authentication, but the only difference is that username and password are transmitted back to the server in an encrypted form.

2.5.5 **Findings**
Username and Password will be used to authenticate users of the system. The Form based mechanism will be chosen as because we can customise the login page using HTML or JSP. Even though the username and password are not encrypted during the transmission, this is the trade off for the performance constraints and assumed that the user will access the application within the protected environments.
In future, however we may be switch to HTTP basic or form-based authentication over SSL by configure the server.

2.6 **Security Authorization**
J2EE promotes resource authorization using roles – a user can have a number of different roles, for example in a personal tutor system personal tutor might have the role *tutor*, while director of studies might have the roles *tutor* and *director*. This means that the director of studies can do everything that the personal tutor can do, but additionally can perform operations that require the role director.
Most java-based web servers allow the developer to specify a custom realm for performing authentication and authorization. A very simple security realm could just use a text file, or and XML file, containing the security credentials and access control lists, which is read by the server when it starts up. However, there is a drawback for this approach as you have to restart the server before the newly added users can start using the system. Most of the modern servers let the users specify alternative realms like LDAP, JDBC sources.

2.6.1 **Findings**
Since security realm mechanism can be support by most of Java application sever for example Tomcat, JBOSS and so on. Luckily, Tomcat and JBOSS provides a realm based on JDBC database access.
This realm uses two database tables for performing authentication and authorization: the first table needs to store the username and passwords, and the second defines the roles each user has been assigned. This may seem a bit complex, and it is- the JDBC realm is designed to be very flexible and can be configured in many different ways.

2.7 **Choosing the Right Technology**
The technology that will use for developing and implementing the system should meet our need for the following:

- Standards-based technology
- Support for rapid development
- Extensibility
- Proven web application development paradigm
- Good quality tool support
- Vendor-neutral and low-cost implementation
Proper division of roles during application design and development: keeping Java development separate from development of the look of the finished web pages

Taking all these factors into consideration, there are number of the technologies I would like to explore.

2.7.1 **Java 2, Enterprise Edition (J2EE) Technologies**

As the project will be developed using J2EE technology, it is very important to gain the understanding of its concepts once we have done that. I hope we can apply the knowledge to solve our problems using all the combination technologies which available from J2EE specification. The specification for the Java 2 Enterprise (J2EE) defines a platform for developing web-enabled applications that includes Enterprise Java Beans (EJBs), Servlets and Java Server Pages (JSP). Therefore, J2EE products are application servers that provide a complete implementation of the EJB, Servlets, and JSP technologies shown in Figure 2.5. In addition, the J2EE outline how these technologies work together to provide a complete solution. So I would like to introduce concept of servlets, JSP and EJB.

![Figure 2.5 Web Tier and J2EE Applications](http://www.java.sun.com)

Copyright ©2004

2.7.2 **Servlets**

The servlet specification defines a server-side component model that can be implemented by web server or container like Tomcat. Servlets are written using the Java Servlet API for generating web pages dynamically. Servlets are developed in the same fashion as enterprise beans; they are Java classes that extend a base component class and have a deployment descriptor. Once a servlet is developed and packaged in a JAR file, it can be deployed in a web sever. When a servlet is deployed, it job is to handle requests for a specific web page or to assist other servlets in handling page requests. Servlets are similar to session beans because they both perform a service and can directly access backend resource (e.g. databases) through JDBC, but they do not represent persistent data.

2.7.3 **Java Server Page**
JavaServer Pages is an extension of the servlet component model that simplifies the process of generating HTML dynamically. JSP allows developers to embed Java code directly into an HTML page as a scripting language. In J2EE, the Java code in JSP page can access the JNDI ENC, just like the code in servlet. In fact JSP pages are translated and compiled into Java servlets, which are then run in a web server just like any other servlet. You can also use JSP to generate XML documents dynamically.

2.7.4 The Enterprise Java Beans
Enterprise Java Beans server-side components come in three fundamentally different types: entity, session, and message-driven beans. Both session and entity beans are RMI-based server-side components that are accessed using distributed object protocols. The message-driven bean, which is new to EJB 2.0, is an asynchronous server-side component that responds to JMS asynchronous messages.

Entity beans can be thought of modelling business concepts that can be expressed as nouns. For example, an entity bean might represent a Student, a set of problems within a problem category, and so on. In other words, entity beans model real-world objects; these objects are usually persistent records in some kind of database.

Session beans are an extension of the client application and are responsible for managing processes or tasks. For example, session beans can manage particular kinds of activities like generating student reports. They have a lot to do with the relationship between different enterprise beans.

Message-driven beans are responsible for coordinating tasks involving other session and entity beans. A message-driven bean subscribes to or listens for specific asynchronous messages to which it responds by processing the message and managing the actions other beans take in response to those messages. It would be very hard to cover all the details of Enterprise JavaBeans as it involves lots of technical aspects, however, this should give the reader the guideline of what components are available for J2EE technology.

2.7.5 The Struts Framework
Struts (org.apache.struts) is an open-source web application framework, based on the MVC design pattern, and built using the Servlet and JSP APIs that can be used for building complex web applications. It allows us to decouple the business logic, control logic, and presentation code of an application, making it more reusable and maintainable. The Struts framework is a part of the Jakarta Project managed by the Apache Software Foundation. Struts can be downloaded from http://jakarta.apache.org/struts.

The Struts framework provides the following services.

- A servlet that acts as the controller in the MVC paradigm.
- JSP tag libraries for bean management, HTML and JavaScript generation, template handling and flow control in a JSP.
- A framework for internationalising messages. Internationalising your web application means making sure that any user messages that occur when the
web application is being used are written in the preferred spoken language for the user. This means that you need to create an application resources file containing the messages for each language.

- An implementation of JDBC to define data sources and a database connection pool.
- A generic error and exception handling mechanism, which involves retrieving error messages from an application resources file.
- XML parsing.
- Files upload utilities.
- Logging utilities.

**The Struts Architecture**

Struts is built using the MVC paradigm (See the figure 2.6) below depicts the high-level architecture for the framework.

We will now review each of the elements of this diagram.

The presentation tier (the view) of a Struts-based application is built using the **Struts tag libraries** (taglibs). Requests from the client are passed to the **ActionServlet**, which acts as the controller. In an application that uses Struts, all the requests that need to go through the framework pass through the **ActionServlet**. This **ActionServlet** passes data from the request into an **ActionForm** JavaBean.

An **ActionForm** is a JavaBean that represents the input data from a form view component. These forms are generated by JSPs using the Struts html tag library. The bean is populated with request parameters by the **ActionServlet**, which may also ask the **ActionForm** to validate the form data that the user submitted.

The **ActionServlet** is configured by defining a set of **ActionMappings**. An **ActionMapping** is an object that maps the URL in a request to components provided by the application developer to process the request. Configuration of the **ActionServlet** and **ActionMappings** are performed in XML configuration files.
The application-specific components used to process the request are known as *Action* classes; they represent the *model* in the MVC architecture. It may be used to validate the information entered by the users, and if an application error occurs during the information processing the *Action* class can create instance of error objects that are then stored in the HTTP *request* object. If the logic within the *Action* class is been executed successfully, the class passes an *ActionForward* object, representing the JSP needed to render the response, to the controller. These *ActionForwards* come in two kinds: forwards specific to a particular *Action* class or a global forward (any *Action* class can pass these *ActionForwards* to the controller).

The description above may seem to technical and confuses but implementing the MVC design using Struts for the web application is pretty natural (See Figure 2.7 Provides a graphical illustration of how this works.).

![Figure 2.7 A MVC architecture diagram for a web application](image)

Figure 2.7 shows the MVC pattern implemented for a Web application. Processing flows as follows.

1. The client browser issues an HTTP request to the application. So, user browser represents the view created using HTML, JSP, and Struts tag and so on.
2. The controller component (*Action* class) received the request. It then decides on how to proceed with processing based on the business rules encoded in it. This process normally involves interacting with Model component (Java Beans EJBs), Databases, and generating View component (*Action Form* class).
3. The model components perform the actual interaction with the persistent data stores or remote systems that they manage interaction with.
4. Based on the results of processing and the data returned from the Model components, the Controller determines the View component that should be used to render the user display. Data is prepared for the View object to render.
5. The View component chosen renders the HTTP response to be sent to the user.
To be more specific about our components involves for building web application using struts these sections provide information about Model, View and Controller component design in general.

2.7.5.1 **Model Components**
Model components generally represent a business object and access back-end databases system. For example, a system in which an EJB server managed the back-end its model components would manage access to the EJBs. In some designs, the model component themselves may be EJBs. This model can be write in such a way that provide a programming interface that hides the details of some back-end data store. So this would give us a great flexibility to change model in the future without touching other components. Professor David Parnas publish his influential in Decomposing Systems into Modules, “for the journal Communications of the Association for Computing Machinery. In it, he wrote:

“We have tried to demonstrate by these examples that it is almost always incorrect to begin the decomposition of a system into modules on the basis of a flowchart. We propose instead that one begins with a list of difficult design decision or decisions which are likely to change. Each module is then designed to hide such a decision from the others”.

Using these ideas, Struts Model components hide implementation details for interacting with back-end database and remote systems.

2.7.5.2 **View Components**
View components are associated with the response to the users. In struts view components are the JSP files using combination for struts tag to render HTML to be sent to the users.

2.7.5.3 **Controller Components**
Controller components direct all the action. All the action must be passed through the controller component. For example, whenever the user submits a form, it is the controller that controller that process it. The Controller component also collects the data from the Model components and feed it to the view component so it will use it to display.

In struts, a Controller component performs several primary tasks as following:

- Validate that the data entered by the user was valid
- Makes decisions about which model components need to be accessed or updated and manages these activities
- Collects the data that the View component will need for display
- Makes decisions about how to recover when errors occur in processing the request or response
- One of the most important tasks is deciding which View component should be displayed to the user.

2.7.6 **JBOSS Application server**
JBoss is advanced middleware with a full J2EE based. JBOSS as a highly flexible service oriented architecture on which to build their own products. The figure 2.8 illustrated the features of JBOSS.

- Full microkernel approach based on Java Management eXtensions (JMX)
- Fully hot-deployable and cyclable service layer with Service Archive format (SAR)
- Fully automated and net based installation with hot-deploy of applications
- Full J2EE 1.3 support (EJB, JCA, JSP, JMX, HTTP etc)
- Full security implementation and JAAS integration
- Full clustering of any java objects (EJB, HTTP, POJO)
- Ground breaking Aspect Oriented Programming (AOP).
- Fully supported by the JBoss Group

Figure 2.8
JBoss features

**JBoss Architecture Overview**
Micro kernels, services and aspects

1. *Microkernel* layer. JBoss first and foremost is a micro kernel based server. The micro kernel is small in footprint. We use our Java Management extensions (JMX) implementation as our first layer. It provides a lightweight component model with full hot-deploy on networks and advanced class loading features and lifecycle management.

2. *Service Oriented Architecture* On top of the micro kernel layer, JBOSS provides a series of Services. Each service is a neatly package and fully hot-deployable unit known as a 'Service ARchive' (SAR). The services can be a transaction monitor or a messaging service, or a mail service, or a database, or a security service or a connection pool. All of JBOSS SOA functionality is packaged in SARs. In fact you can remove the webserv of JBoss, and the EJB container by simply removing the corresponding SAR. Many ISV's and IT shops build their own services and package.
them as SARs, you can deploy any service you build into JBoss that way. It is extremely easy to extend JBOSS.

3. Aspect Layer The aspect layer (AOP) is embodied in interceptors since the 2.x days. Interceptors enable the system to transparently add the behavior provided by the services into any objects. This is how JBoss's EJB container is built, as a pre-packaged set of interceptors. You can add and remove to these, effectively extending J2EE both client and server side since the 3.x series. In the 4.x series, the focus is on simplifying the programming by letting you weave in these aspects easily with tag-driven development, a bit like in C#.

4. Application Layer The application layer is the ‘user-land’ of JBOSS. This is where your applications live. Whether they leverage the services directly (programmatically) or you use the AOP layer and the tag-driven aspects to add behaviour to your objects, your applications can greatly leverage the vast infrastructure present in JBoss. JBoss is the future of application development. JBoss today has features such as “no compilation” that make it the server of choice for advanced developers.

(Source reference: http://www.jboss.org)

2.8 Relational Database Design
A relational database is a collection of tables. A table contains a number of rows all contain information regarding the same type of item. For example, the personal tutor system will use a relational database to store the student’s information. Therefore the database must be designed so that it allows rapid and efficient access to data.

2.8.1 Database Normalisation
Normalization is a method for increasing the quality of database design. The goal of normalization is identical to that of data modelling- a high-fidelity design. It usually involves dividing a database into two or more tables and defining relationships between the various tables. Normalising a database increases the integrity of the data by minimising redundancy and reducing the occurrences of data anomalies and data inconsistencies but this does not mean it can give you a correct design of your database.

However, normalising a database can affect the retrieval rate of data. Imagine that it would be faster to retrieve a data from a single database in an un-normalised design than retrieve a data from multiple tables normalised design because SQL query will involve join multiple tables. Therefore, there is a trade off for doing normalisation but this is very rare to happen that not to normalise the design database since the performance of the database server are increasing and can cope with this problem. Moreover, other method can be used for example we can create a view of the table to make it a single table from normalise database so that we also query from a single table this would help us increase the performance plus having a high-fidelity design of the database.

Codd first developed the process of normalisation (Codd 1972) and it was further progressd by Codd(1974) and Fagan(1977 & 1979).
“A relation is in **first normal form** if and only if all columns are single valued. A relation is in **second normal form** if and only if it is in first normal form, and all non-key columns are dependent on the key… A relation is in **third normal form** if and only if it is in third normal form and has no transitive dependencies… A relation is in **Boyce-Codd normal form** if and only if every determinant is a candidate key...A relation is in **fourth normal form** if it is Boyce-Codd normal form and all multivalued dependencies on the relation are functional dependencies.. A relation is in **fifth normal form** if and only if every join dependency of the relation is a consequence of the candidate keys of the relation.” (Watson 2002)

According to the various sources the fourth and fifth normal forms are not popular this is due to the complexity and only suitable for some particular problems. Nevertheless, for general problems using up third normal form would be sufficient.

### 2.9 Choosing the right database

As Structured Query Language or SQL for short is a statement that we sent to an RDBMS in order to get it to retrieve data. Therefore we have to find the appropriate RDBMS to use and support our SQL. MySQL has a popular feature that we need for the project and also freely available under GNU Public License.

#### 2.9.1 MySQL database server

The MySQL database server is the world's most popular open source database. Its architecture makes it extremely fast and easy to customize. Extensive reuse of code within the software and a minimalistic approach to producing functionally-rich features has resulted in a database management system unmatched in speed, compactness, stability and ease of deployment. The unique separation of the core server from the storage engine makes it possible to run with strict transaction control or with ultra-fast transaction less disk access, whichever is most appropriate for the situation.

The MySQL database server is available without a license fee under the GNU General Public License (GPL). Commercial non GPL licenses are available for users who prefer not to be restricted by the terms of the GPL.

There are four versions of the database server available:

- **MySQL Standard** includes the standard MySQL storage engines and the InnoDB storage engine. InnoDB is a transaction-safe, ACID-compliant storage engine with commit, rollback, crash recovery and row-level locking capabilities. This version is for users who want the high-performance MySQL database with full transaction support. MySQL Standard is licensed under the GPL. MySQL Pro is the commercially-licensed version of the server with the same feature-set.

- **MySQL Max** is for the user who wants early access to new features. This version includes the standard MySQL storage engines, the InnoDB storage engine, and other extras like the Berkeley database (BDB) storage engine, and support for splitting tables across multiple files to avoid operating system file size limitations. In future releases, MySQL Max will include more cutting-edge features.

- **MySQL Pro** is the commercially licensed version of the MySQL Standard database server, including InnoDB support.
MySQL Classic only includes the standard MySQL storage engines, differing from MySQL Pro and MySQL Standard only by the omission of the InnoDB storage engine. It is only available under a commercial license.

Features

ANSI SQL syntax support
The MySQL database server supports a broad subset of the ANSI SQL 99 syntax, along with extra extensions such as the REPLACE statement and the LIMIT clause for SELECT and DELETE. Alternative syntaxes from other database systems are also supported, to make porting applications easier. Our coverage of the complete ANSI SQL 99 syntax is expanding, with version 4.0 introducing support for the UNION statement.

Cross-platform support
We provide optimised binaries for a wide range of platforms, including Linux, Microsoft Windows, FreeBSD, Sun Solaris, IBM's AIX, Mac OS X, HP-UX, AIX, QNX, Novell NetWare, SCO OpenUnix, SGI Irix, and Dec OSF. You can connect to a MySQL database server from all of the major platforms, using nearly any programming language, with our standard thread safe client library or one of the products in our Connector family of database drivers.

Independent storage engines
MySQL database server's unique independent storage engines let you choose the type of database storage that is most appropriate for your particular needs. If you need row-level locking and transaction support, you can use the InnoDB storage engine. If your application doesn't require transactions, you can use the MyISAM storage engine for maximum performance.

Transactions
Using the InnoDB or Berkeley DB (BDB) storage engines, the MySQL database server supports transactions. The InnoDB storage engine also supports foreign key constraints.

Flexible security system, including SSL support
The MySQL database server has an advanced permissions and security system, including support for SSL transport-layer encryption. As of version 4.0, the security system also allows you to limit server resources on a per-user basis.

Query caching
Version 4.0 of the server includes a new query cache, which can significantly increase the performance of commonly-issued queries, without requiring any special programming. Performance can be increased by over 200% in typical usage.

Replication
Using database replication, you can have many "slave" servers running off a single "master" server for robustness and speed.

Full-text indexing and searching
Full-text indexes allow you to search fields containing arbitrary text for specific words and phrases, including relevance rankings. With version 4.0, we've expanded the fulltext search to include exact phrase matching and Boolean search operators, which allows for even more fine-grained control over your search results.

**Embedded database library**
Using the embedded database library (libmysqld), you can include the full power of the MySQL database server into applications and electronics devices, without your end-user having any awareness of the underlying database. The embedded MySQL database is ideal for use behind the scenes in Internet appliances, public kiosks, turn-key hardware/software combination units, high performance Internet servers, self-contained databases distributed on CD-ROM, and more possibilities just waiting for you to invent them.

(Source reference: [http://www.mysql.com](http://www.mysql.com))

### 2.10 Data Protection issues
As the software project will be recorded the data of students therefore, we need to be aware of the data protection issues. The information below should give us a guideline of what are the issues that we have to be concerned. For further information please visit [http://www.dataprotection.gov.uk](http://www.dataprotection.gov.uk)

#### 2.10.1 Subject access – education records in England
The Data Protection Act 1998 came into force on 1 March 2000. It gives all individuals who are the subject of personal data ("data subjects") a general right of access to the personal data which relates to them. These rights are known as "subject access rights". Requests for access to records and for other information about those records are known as "subject access requests." Personal data may take the form of computerised or, in some cases, paper records.

The Act also sets out specific rights for school students in relation to educational records held within the state education system whether these are held in computerised or paper form. Educational records are the official records for which head teachers are responsible. The rights of students lie alongside the rights of parents to obtain copies of the educational records relating to their children. These are set out in separate education regulations The Education (Pupil Information) (England) Regulations 2000. This leaflet explains the rights of both students and parents in relation to official educational records. (The leaflet does not set out to explain the general right of access to personal data not forming part of the official record, for instance records held by individual teachers for their own use, or records held by independent schools. This is described in the leaflet, "Using the law to protect your information," also available from the Data Protection Commissioner.)

**Pupil Rights**
The Data Protection Act gives all school students, regardless of age, the right of access to their school pupil records. Requests to see or receive copies of records should be made in writing to head teachers.

In addition to the right to be given a copy of the educational record, students are entitled to be given a description of the personal data which makes up the record, together with details of the purposes for which the data are processed, the sources of
the data (if known) and the individuals or organisations to which the data may have been disclosed.

Periods of up to 15 school days are allowed in which to respond to a subject access request. (The equivalent period for other types of record is up to 40 days.) If asked to provide a hard copy of the record, a fee may be charged according to the number of pages. (See below for the scale of charges.) Students may be asked for information to verify their identity if is necessary, for instance in the case of former pupils who may not be currently known to the school. They may also be asked for information necessary to locate the data held about them. For instance a student may be asked to supply the dates between which he or she attended the school.

While in principle students have a right of access to the whole of their educational records, in exceptional cases some information may be withheld. The main exemptions are for information which might cause harm to the physical or mental health of the student or a third party, information which may identify third parties (for example other pupils, although not teachers), and information which forms part of some court reports. Information may also be withheld if in that particular case it would hinder the prevention and detection of crime or the prosecution or apprehension of offenders to provide it.

If students are incapable of understanding or exercising their own rights under the Data Protection Act, (for instance because they are too young), parents can, of course, make subject access requests on their behalf.

If a request for information under the Act is refused or ignored, the matter can be referred to the Data Protection Commissioner or an application for disclosure can be made to a court.

Parents rights

In addition to the subject access right which can be exercised by pupils or by parents acting on behalf of pupils, parents have their own independent right of access to the official educational records of their children under the separate education regulations referred to in footnote 1. In essence the information to which parents are entitled and the exemptions are the same as for pupils although there is no parental right of access to information which does not form part of the official record. Requests to see or receive copies of the educational records of their children should be made in writing to head teachers. If asked to supply a hard copy of the record, a fee covering the cost of supplying the information, may be charged. This is set by the governing body. A parent seeking access to an education record does not, however, have a right of redress under the Data Protection Act unless he or she is acting on behalf of their child. If a parent is not given a copy of his or her child's records, in the first instance he or she should contact the governing body and, after as a last resort, the courts.

Because parents have an independent right of access to pupil records, the students themselves have no right to prevent their parents from obtaining a copy of their school records.

Subject Access Fees

<table>
<thead>
<tr>
<th>No of Pages</th>
<th>Maximum Fee</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-19</td>
<td>£1</td>
</tr>
<tr>
<td>20-29</td>
<td>£2</td>
</tr>
<tr>
<td>30-39</td>
<td>£3</td>
</tr>
<tr>
<td>100-149</td>
<td>£10</td>
</tr>
<tr>
<td>150-199</td>
<td>£15</td>
</tr>
<tr>
<td>200-249</td>
<td>£20</td>
</tr>
</tbody>
</table>
Further Information
Further information about the Data Protection Act can be obtained from the Commissioner’s web site (www.dataprotection.gov.uk) or can be requested from our the Information Line (01625 545745).
(Source reference http://www.dataprotection.gov.uk )

2.11 Conclusion

The software development project would be to develop a scalable three-tier Personal Tutor Reporting and Recording system using J2EE. Therefore, I considered using the combination of enterprise beans namely Entity beans and Session beans but not message driven bean because it does not include as the part of our software requirements. Entity beans would be used to represent our real-world objects in our application and handling major tasks for accessing MySQL database. Session beans would take care of the work flow of the program and I will also use stateless session bean to call other session beans and entity beans to do a specific tasks. JSP and Servlets will be used to create a web-user interface and display to the users.

Struts framework will be used as the web application framework and the interesting part would be how to integrate Sturts components together with EJBs and run smoothly within JBOSS.

The software will also be deployed in JBOSS server which free open source and contain all the features which we need. That’s why we choose to use it. Alternatively, we may also use J2EE server from Sun as well however J2EE from Sun need more space to install and run. A new version 3.x of JBOSS provide both Tomcat server and EJB server and support JSP these really give a great flexibility for the application deployment.

As the software will be recorded students information therefore, the data protection issues would have influence to the software design and implementations for example, how the software should allow users to access their data as securely as possible.

3 Requirement analysis of Personal tutor System

3.1 Introduction

This chapter details the process uses for gathering the requirements. The main discussions of these requirements are focusing on users’ requirements and architecture of the systems. A set of scenarios and basic data flow models are then generated using the requirements gathered.

3.2 Requirements Elicitation
This process involves system developer work with clients who in this case a supervisor to find out about the application domain and what functionality the system should provide. There are also some other stakeholder who may be involve directly or indirectly or even influence the systems.

3.2.1 Elicitation Process
The first step would be interviewed clients and recorded our conversation between myself, system developer and clients. This step can also be broken down into different sessions. For example, the first session client would be asked to answer some simple series questions that would let them explain the basic ideas or what functionality that client really has in their mind and also this is the good chance for system developer to develop the first understanding of the system. Once the first interview session has completed and the system developer go away and had sometime to think about and try to understand in order to develop more further understanding about the application domain then system developer come back and do a second interview session. The second interview sessions would involve systems developer asking clients more specific question related to the system. Identifying some constraints and other related issues such as technologies, environmental issues and so on that would cause the problems during the development process and sometime the system developer even suggests alternative solution to tackle the problem. The technique which I normally used during talking to the clients is called Use Cases. Use cases provide a good way of getting an overall picture of what is happening or plan to happen in the new system.

This approach is based on the activities are the following:

3.2.1.1 Find Actor and Use Cases
Actors represent the roles that people perform in using the system and actor also include other system such databases system, SAMIS systems and so on. It is easier to start with by asking clients these questions:

- “Who are the people will use this personal tutor system.”
- “What are the other systems that personal tutor system will interact with?”

Example 3.1 Here is a short excerpt from an interview transcript with my supervisor (Dr Barry who is having the idea of building the system and he is very technical person). Lek is a final year student and plays the role of system analysts for his project.

---
LEK: Who are the people will use the system?
Dr Barry: The people who will use the system essentially going to be personal tutor and director of studies who will interview their own students in order to track their progress and helping system to solve their problems during the year and the system could also be extended to other people in the university for example AWARE, Learning Support, Financial office, Career service and Medical Centre and so on. These other people would probably have their own system but the personal tutor system would give them additional information which
allows them to access where it permitted. For example, there is a blind student who wants to get access for the electronic copy of book and we try to help them to get it but at the end of the day this student also went to see the Learning support and if we would have seen his note for the Learning support and the Learning support would have seen his note for us then we would have been realised that we both Learning support and us doing the same thing and potentially we would have to hand it in for one or another to do it. So, in this case Learning support may be have their own expertise and be able to help the student bra bra bra. But in essentially, the system has to decide on the note of the student and what can be reviewed and this is where the security aspect comes in. Well, but the system can be go wider than that.

Dr Barry: There will be also the administrative roles who will main the information of students and staffs and so on.

For the conversation above there are four main actors who are directly interacting with the systems. They are personal tutor, and director of studies as these two are directly performing basic tasks like recording information and setting the basic security. And other people like Aware, Learning support and so on as these people will gain the access of the record. However, the other people like Learning support may have their own system that should be directly or indirectly linked to the personal tutor system in order to access and be able to review information which has been recorded in the personal tutor system and finally, the administrator who will maintain the system and also registration users to the system -this is a separate tasks from the other three and also registration user to the system. Another major key aspect that Dr Barry mentions is also the reviewing and security issue for the users to access information and various part of information that who be will specify what and who can be reviewed. For example, the note recorded between a personal tutor and a blind student that part of it can be opened to be reviewed by Learning support. Therefore, if we look at the actions of the system in the example there are four distinct use cases.

- Recording the interview conversation of students.
- Specify security access for the particular problem conversation.
- Access information from their own system.
- Maintaining the system and record of staffs.

These are show in Figure 3.1. Note that for each uses cases above we have not broken it down into the level of details yet as because we try to group sequences of actions together into units that are meaningful in the context of the business.
3.2.1.2  **Prioritize Use Cases**
The next step is to assign prioritise the use cases. The purpose of this is to ensure that
the important use cases are developed in early iterations of the software development
process. Because this is not a real client project work selecting a priority use case to
develop may have to consider the work load and how difficult each use cases can may
be turned it into a working part of the program.

3.2.1.3  **Developing each Use Case (Starting with the Priority ones)**
The purpose of the activities is to produce the detailed specification of regarding to
the behaviour of each use cases. This activities may also result in new use cases
coming to light this is true because the system analyst gain more understanding of the
requirements

3.2.1.4  **Structure the Use Case Model**
This is finally activities that will added to the use case diagram through the use of
generalization and also extending relationships of other use cases and grouping of use
cases in packages.

For most cases where clients are non technical producing use case diagrams using pen
and paper and walking them through the system and explaining step-by-step would be
appropriated appropriate and will really help the system developer to analyse the
requirements.
The point of doing a second interview and have some holding a conversation would is to really help both clients to express what they want or even what they do not want from the system and the system developer to develop more a greater understanding of the system and resolve any ambiguity in the requirement issues given by the client from the first interview session. In addition, Using requirements analysis techniques such as use cases also very useful and save time.

3.2.2 Problems Encountered
The staff can be very busy sometime, so to get them interview can be very difficult. This is also very hard particularly difficult when system analysts are trying to develop a new system. Some of the staffs were not forthcoming with the finer details and some of them had never heard of it before. Using scenarios and asking sensible questions to a range number of staffs would help the system developer gaining an overview of the process in involved.

3.3 Functional Requirements
Functional requirements for the system are described as the functionality or services that the system is expected to provide. The descriptions below outline the functionality that the personal tutor system should provide.

3.3.1 Personal Tutor requirements
Personal Tutors are responsible for interviewing students especially when they come up with some problems that need help. Personal tutors can use the system to record detail of student’s problems during their conversation. Personal tutor can also want to use system to plan and keeping track of series of actions in order to help student solving their problems. In some problem cases personal tutor may feel that he/she can not deal with the problems themselves therefore, it would be appropriate for them to forward those problems to several of appropriate people in order to help students solve their problems effectively. These people are Director of studies, Learning support staffs, Career offices staffs, AWARE, Medical centre and so on. This action of forwarding student information in order to be reviewed by various people must be recorded by the system.

Because of the information during the interview between student and personal tutor should be kept in confidential at all time unless students allow their personal tutor to send their information to be reviewed. Therefore, the personal tutor should be able to identify part of the information which he/she want to send it to reviewed by whom and allow them to access only those parts of the students information. This is where the level of security of access issues comes in to play a very important part of the system in term of developer perspective. Therefore, the system must not allow users who do not have their right to review the information to see it. This level of security must be specified by personal tutor during enter information and of course allowing access must be approved by the students during their interview.

3.3.1.1 Adding the record detail of problems of students during their conversation. The system should allow user to enter the title, date of
problem that occurs, provide some sensible way of identifying and describing the problem.

3.3.1.2 **Editing the existing record of students’ problems for accuracy.** Only the author of the record should have the right to do this operation. Any changes that have been made will be recorded on the system database and will also be notified to others relevant users who own the problems.

3.3.1.3 **Log in/off to enter/leave the system.**

3.3.1.4 **Record number of regular meeting between personal tutor and students.** The system should provide some way of monitoring the number of times that student come to see their personal tutor.

3.3.1.5 **Record multiple problems at the same time.** For each meeting the student may have more than one problem that need to be recorded therefore the system should provide appropriate view for the user to record multiple problems.

3.3.1.6 **The system should allow personal tutor to add some notes/comment on each problem that he/she enter into the system.** This note can also be reviewed by the student in some circumstances due to the data protection however, in normal condition it will only allow the student to see them in the appropriate context.

3.3.1.7 **Provide the generic categories of problem for personal tutor to select.** The system will provide the general list of categories such as personal problem, academic problem, physiological problem, exam problem and so on. This list of generic problem are maintained by system administrator.

3.3.1.8 **Allow personal tutor to build their own categories of problems.** Because of the limitation of generic list of category of the system, personal tutor allow entering their own description to describe the nature of the problem.

3.3.1.9 **Specify the granularity level of reviewing access on each of the student record.** Within one particular problem it can be declare as public where everybody has the right to see it. On the other hand, it can be also be declared in confidential this is where some part or even everything which can be accessed and reviewed by various kind of people again this permission depends on the situation decided by personal tutor. This is also a security issue that the system must take care of. The author who in this case, personal tutor may for example have to specify on each part of the body of the problem in order to allow various people to access it. However, the system should provide the default which is the person who allow to review information is the person who taking to the student when the information is being recorded in time.

3.3.1.10 **Specify time interval to allow people to review information of students send by personal tutor.** In some events some issues should be kept open a
period of time. For example, personal tutor may want to send their students to see learning support and part of the problem can be opened for one week for learning support to be reviewed.

3.3.1.11 **Identify the links of relevant problems associate to the current problem record.** For each meeting there may be more than one problem that needs to be recorded into the system. These problems sometime related to each other or even associated to series of previous problem. Therefore the system should also provide categories list of problems that associated with the current that being record.

3.3.1.12 **Display an appropriate view and suitable tools for personal tutor.** The system should only display an appropriate view for personal tutor.

3.3.1.13 **For each recorded problem personal tutor should be able to select an appropriate list of actions in order to solve the problem.**

The lists of actions are the following:

- Ask the students to fill in some forms and collecting some evidence to back up their situation.
- Send email to other lectures for example asking for the extension of the coursework and so on. The main body of email again should be filled in by personal tutor.
- Pass the problem onto the director of studies
- Go to see people such as learning support, career services, AWARE and medical centre and so on.

3.3.1.14 **Allow personal tutor to create their own description of actions.** This can be done by providing free form of entering free text. It can be something for example title of the action, the reason for doing it, description and so on. Once the new action has been created it should be available for user to choose from the drop down list menu.

3.3.1.15 **Should be able to generate set of reports for each meeting** specify all the problems that students have and list of actions which they should be carried out in order to solve their problems. This will help both students and personal tutors to keep track of which actions have been completed and which actions have not.

3.3.1.16 **Provide and appropriate view to identify various actions for students to carry out.** For example there may be a tick box for personal tutor to tick off any actions that have been completed by students.

3.3.1.17 **Allow the personal tutor to build their agenda.** The agenda can be for example list of things that personal tutor ask student to do but they have not complete it yet. So the system will allow them to send may be notifying email to students and so on.

3.3.1.18 **The personal tutor should also be able to attach files & other electronic copy to a record** (as should other users).
3.3.1.19 Provide print out function to print the report for student to review.

3.3.2 **Director of Studies requirement requirements**
Director of studies can perform all the tasks that allow for personal tutor plus their own additional tasks

3.2.2.1 **Director of studies is also personal tutor themselves therefore they should be able to perform all the tasks given for personal tutor.** Because director of studies also have number of students assign to him personally as well as other students who need to see him, therefore director of studies need to perform personal tutor task for his own students plus other additional tasks for other students.

3.2.2.2 **Director of studies is also personal tutor themselves therefore they should be able to perform all tasks given for personal tutor.**

3.2.2.3 **The system should provide additional actions that only be able suitable for director of study.** These actions are something that personal tutor can not deal with themselves for example, if student want to withdraw from the course then they will have to come to see director of studies and so on.

3.2.2.4 **Allow the director of studies to add their own comments.** Director of studies may want to add some comment regarding on some of the students problem or create a new comment for themselves. This comment must be recorded within the system and can be specified to view by various people.

3.2.2.5 **All the director of studies to edit their own comments and their own part of record.** The change will be flagged and notify to other users.

3.2.2.6 **All the director of studies to delete their own comments.** The change will be flagged and notify to other users.

3.2.2.7 **Allow the director of studies to specify their own comments to be reviews by number of people**

3.2.2.8 **The system should display list of students who need help sort by date, their urgent and who has send the actions and so on.** This list of students is coming from various sources for example it comes from personal tutor forward number of students or it comes from learning support people because the student went to see them and discuss about some problem.

3.2.2.9 **Attach some files or electronic copy of document associate with student record.**

3.3.3 **Students requirement requirements**
3.3.3.1 The system should allow students to access and review their information.

3.3.3.2 The system should also display list of actions that student have to follow in order to solve their problems.

3.3.3.3 The system must only display the information that belongs to students and they have their own right to see it. Some information that is entered by personal tutor or director of studied or learning support people can not be shown to students. This accessibility should be specified when those people enter the data.

3.3.3.4 If the student invokes the Data Protection act, then _all_ information will need to be revealed, but it would normally be through a printed report of all information and not an electronic report.

3.3.3.5 The functionality to produce the printed report should be present, however. The capabilities of different services will depend on the service. For example, AWARE and Learning Support would have similar capabilities to those of a Personal Tutor, whereas the Student Money Service or Careers would only record details of a meeting and would not tend to refer people on to see others.

3.3.4 Services people requirements

Learning support, Career office, AWARE and so on is considered to be services people. These people can access the system and use the system. Because of these people can offer their expertise to help the students it does not mean they are internal staffs within the department. Therefore, the system should provide the functionality suitable for them. Students can go to see services people directly even before they see their personal tutor or director of studies. Services people will then record the information between them. Services people can forward the record of student to various numbers of people. There are between services people themselves, between services people and personal tutor and between services people and director of studies. In other word the record information that services people enter can be specify the level of access in the same way as personal tutor and director of studies can perform their operation. The levels of access again need to be agreed by students during the interview. Services people may want to add their own notes and allow various users to view it.

The system should allow personal tutor do the following:

3.3.4.1 Adding the record detail of problems of students during their conversation.

3.3.4.2 Allow external people to specify which part of the problem can be reviewed by whom. This is because external people may want to send some part of the information to other users to review a specific problem related to student. However, this can be done if and only if agree by the student.
3.3.4.3 **Editing the existing record of students’ problems for accuracy.** Only the author of the record allows editing their record. After making changes, the system will notify other users who own the problems.

3.3.4.4 **Deleting the existing record of students.** Only the author of the record should have their right to delete the information. Any changes that have been made will be recorded into the system database and will also be notified to other relevant users who own the problems.

3.3.4.5 **Log in/off to enter/leave the system.**

3.3.4.6 **Display appropriate tools suitable for services people.** This can be free text form allowing them to record their advice for helping students. Again, this advice could also be useful for other users to get access therefore; services people should specify who have the right to see the data.

3.3.4.7 **For each recorded problem external people should be able to select an appropriate list of actions in order to solve the problem.** Because external people are experts in their fields, therefore, each of them should get the different list of actions. However, there should be some common actions that the system should provide.

3.3.5 **Administrator requirements**

3.3.5.1 The system should provide the functionality for administrator to maintain the system.

3.3.5.2 Administrator can do roll forward the record of students for example to roll first year students to second year students when students get to the end of their year.

3.3.5.3 The system should provide archive for previous interview record.

3.3.5.4 Some students may be suspended or repeat their year or even withdraw from their courses. The system should make sure how to deal with these issues when rolling the student record.

3.3.5.5 The system may receive the Excel spreadsheet from external systems (i.e. SAMIS) and need to extract certain information which will be used for maintenance the record of students within the system.

3.3.5.6 The admin also has to enrol the people with particular responsibility for example given them Director of study roles, Personal tutor roles, which students should be assign to which tutor and so on.

3.3.5.7 The system should allow administrators to give people with the right user name and password.

3.3.5.8 The system should provide a hook up functionality to communicate with some external system.
3.3.5.9 Some staffs include personal tutors can leave and they has to be reassigned a student with a new personal tutor. So the system also need to keep track on the report that previous personal tutor wrote and allow the new assign personal tutor to.

3.3.5.10 The system should allow admin to maintain the data integrity within the system. For example, pick up the duplicate action phrases and replace them with some meaningful phrases.

Learning support, Career office, AWARE and so on is considered to be services people. These people can access the system and use the system. Because of these people can offer their

3.4 **Non-Functional Requirements**
Non-functional requirements are the requirements which are not directly concerned with the specific functions delivered by the system. Non-functional requirements are often related to emergent properties, for example, reliability. In personal tutor system the non-functional requirements for the systems should be concern on the following area:

3.4.1 **Security requirements.**
- All personal data must be kept secure at all time due to the Data Protection Act (DPA 1998).
- The user and password should be used for authenticate and authorization the server before accessing the system.
- All passwords must be encrypted before transmission.

3.4.2 **Usability requirements.**
- The users should not spend time more that 10 minutes to train how to use the system.
- The system should provide a meaningful feedback to users so that users are aware of any changing state occur by the software.
- The personal tutor system is a web-based application therefore; it should be able to access it from any platform via web browser.
- A range of input can be entered by the users therefore, the system must be handling how to recover and prevent any serious errors.

3.4.3 **Performance requirements.**
- The system should be able to handle multiple request data simultaneously. As there will be some users who will access the same record at the same time even most of it would be read only. The system should find way to handle this.
- The single data should be allowed to display to multiple users at the same time.
- The system must be scalable and be ready to deploy in different server.
- The system should not crash during use if this disaster mean to happened then all the data must be store in the database before everything is shutting down.

3.4.4 **Look and Feel requirements.**
- The colour combination must be suitable for all users include colour-blind users.
• All text and icon must be readable and meaningful for the users.
• The menu and drop down list must be used to provide the grouping logical function for the users

3.4.5 **Legal Requirements**
• The system must follow all the rules state by the Data Protection Act (DPA 1998).

3.5 **Conclusion**
Requirement analysis is the most important part of this project because the development process can be done or plan any further without this part. As can be seen from the above the different techniques are put together such as interviewing users, drawing different use cases and scenarios in order to gathering requirements. As a result the set of functional and non-functional requirement have been produced successfully.

So, the next big step of after this would be an implementation but because of the complexity of the platform choice and language selection plus the number of workload and coursework to carried out during the final year. Most of the requirement will not be able to implement and the complete software will not be able to deliver to the users in time unfortunately. Nevertheless, in order to gain an experience on software development project and explore the variety of different technologies most significant basic requirement on this project will be implemented and in order to see how all the component technologies are put together to provide a simple but functional application.

So the next big things to come are the following chapter. The consideration of selecting the right enterprise Java beans since there are so many types and each of them consider used best on its own way but when put all of them together they are not as good as it should be. Database design is also consider as the heart of this project without the solid RDBMS and good selection of SQL queries join different tables can have an impact on the performance. Finally, the case study for the prototype of personal tutor application which will demonstrate the software development skills which follow the principle concept of software engineering and Object oriented programming.

4 **Enterprise Java Bean**
4.1 Introduction
Enterprise Java Beans has revolutionized the way we develop our software application today. It combines server-side components with distributed object technologies and asynchronous messaging to greatly simplify the task of application development. Enterprise Java Bean is portable, scalable for business systems and Java is platform independent. However, Enterprise Java Beans is an implementation independent. For example it can be deployed in any application servers that implement EJB specification such as JBOSS, BEA’s Weblogic and so on. It can also access many vendors of relation databases (DB2, ORACLE, SQLServer or even open source MySQL).

So, in this chapter it would be too ambitious and impossible to cover all the aspect of enterprise java bean but my aim is to outline the different type of them and how each of them can be combined for the project. So let start with explain the enterprise bean component and explain each of them in more detail and finally let see where can they fit into the personal tutor project.

4.2 The Enterprise Bean Component
Enterprise Java Beans come in three fundamentally different types: entity, session, and message-driven beans. Both entity and session bean are RMI-based server-side component but the message-driven bean is an asynchronous server side component that responds to JMS asynchronous messages. So the following are brief description of each component.

4.2.1 Entity Beans
Entity beans are primarily used as an interface to a database. It allows the developer to focus on the database data object level but not to worry about the low level management of the database. The actual JDBC calls as well as the code are taken care of by the entity bean and the EJB container.

Entity beans are not intended to contain a great deal of implementing business logic because they are tight integration with the database instead it uses it for model our business concepts that can be expressed as nouns. In other word, entities bean represent data in the database, so any changes to an entity bean result in changes in the database and that way representing data as entity beans can make development easier and cost-effective. So to be more precise how this is entity is going to work let consider the following statement. When a new EJB is created a new record must be inserting into the database and a bean instance must be associated with that data. As the EJB is used and its state changes, these changes must be synchronized with the data in the database so in other word a bean instance persists with the databases.

There are two basic types of entity beans namely container-managed persistence beans and bean-managed persistence. Both of them are distinguished by how they manage persistence:

4.2.1.1 Container-managed persistence (CMP)
This type of bean, the container knows how a bean instance’s persistence. This has to specify the relationship fields in the deployment descriptor which will map to the
database. So the container then knows how to take care of inserting, removing, updating the database. For entity beans to model real-world database modelling concept, they must be capable of forming the complex relationships with each other. This is also true for CMP to specify these relationships with in its deployment descriptor which can be complicated and require solid experience of J2EE. The relationship can be something simple such as two entities forming one-to-one unidirectional relationship to a large complex like more than 10 entities forming many-to-many and other relationships and so on.

The most interesting part of CMP that I found is that how complicate the container can handle cascade insert, delete and remove the entity bean’s data from the database. Deleting bean’s data, of course has an impact on the relationships that entity bean has with other entity bean. So again the cascade relationship can also be specify within the deployment descriptor and this is very hard but interesting. So nothing more to say about the cascade delete except it is a powerful tool, but it’s also dangerous and should be handle with cares.

So far, Container-managed persistence has been changed so dramatically and not compatible with EJB 1.1 (an older version) therefore, selecting the application server for the development is important as the vendor should support both older and new version of EJB. For this project there is no problem but if you imagine a big company with the legacy system there could be a big investment on this. Anyway, even thought the CMP is complex to implement and configure they can map and scalable and can be finer in granularity on the database.

4.2.1.2 Bean-Managed Persistence (BMP)
BMP gives developer more flexibility of writing the entities beans. Unlike CMP the developer must explicitly know what type of database and how the bean class’s fields map to that database. The deployment descriptor can be used only specify what database required for this entity bean to connect. Some people may prefer CMP because they can configure their entity mean and take advantage of container to manage their beans however; this is only work for one database at a time. Imagine that you have to run the program to do transaction over different databases then does it CMP has not provided with the functionality to do that. Instead BMP allows you. In other word BMP is more benefit if you need to use data from a combination of different databases or other resources such as legacy system.

The hard part of using BMP is again more work required to define a bean. The developer must understand the structure of the databases, the logic of the bean to create, update and remove data. Moreover, because the beans are tight to the database so any changes to the databases then the result of updating the definition of the beans. This sounds like a hard work but again it is a trade off between flexibility and complexity.

4.2.1.3 Conclusion
The complexity of the personal tutor software is how to model entities and how to capture the data recording between the student and personal tutor moreover each
record has a security constraints associated with it. Therefore, the developer must truly understand the structure of the database and write the bean to map the structure of the database and in order to accommodate a complex and unusual set of data. I would rather consider implement BMP type of entities beans for this project or at least use it as the first stage to tackle this problem.

4.2.2 **Session Beans**

"Without you I do not know what to do", sorry to mention this on this report but it helps me to describe the Sessions beans. Session bean are mostly used for the business logic and a bit for data access. Unlike the Entity beans which provide an interface for the developer to create, insert, update and share data in the database and entity beans also allow the developer to reuse the code and cut down the development costs. For example, on the project you can reuse the concept of Student entity bean without having to redefine them in the business logic.

Session bean on the other hand describes the work flow or logic of your program which will be fitted in as your business method for example, if you want to register the users then Session beans describe the concept for this to say how to collaborate with other beans in order to complete this register tasks. Not only you can use Session beans to describe the logic of you program Session bean itself can also be used to access the record and modify the record in database directly. However, if you wish to do that you can and you may end up writing a thick Session beans which can do everything and do not represent any shared data in the database in other word it is a bad design. Having mention that bad design or dirty trick always does the job for example if you want to update or access multiple record from multiple database in a single transaction then consider to wrap it up within a single session bean can be a good idea and easy to maintain but this is case by case basis.

“As a rule of thumb, an entity bean is developed to provide a safe and consistence interface too a set of shared data that defines a concept. This data may update quickly. Session bean access data that spans concepts, is not shared, and is usually read-only.” (Enterprise JavaBeans O’Reilly).

Session beans are divided into two basic types: stateless and stateful.

4.2.2.1 **Stateless Session beans**

Stateless session bean is a collection of procedures that when you invoke a method on a stateless it executes its method and returns the result. Once its given tasks is completed it ready to serve other tasks and do not care what other requests have gone before or might follow. The advantage of stateless is that it being relatively lightweight and gives flexibility to developer to structure its tasks and because the stateless session is not dedicated to one client therefore, it can be reused for other clients.

The way I see it is that stateless session beans are the most versatile of all the EJB form describe above you can almost do anything with them especially when you are writing up a web application and taking advantage of web contain session management control so that you do not need to worry which bean are serving which client.

4.2.2.2 **Stateful Session beans**
Stateful session beans are similar to stateless beans except that they remain stateful with the user’s session in the EJB server. So, that ideally it serves a single client at the time and you can predict what going on when the method of Stateful session bean invoke. This is may sounds alright to you because the bean main conversation state between itself and client so that you can do all lots of things with it for example you may want to hold the information of your data between requests. However, this is comes at a price. The price is that EJB container has to dedicate resources to maintain the stateful session bean until the session on the EJB container ends. If there are many users using them at the same time then maintaining the pool of stateful session beans can take up a lot of resources.

One situation in which stateful session bean can be useful is when each task may rely on some information gathered or changed by a previous operation for example when you fill in the fields on a GUI client you are creating conversation state. Pressing a button executes an operation that might fill in more fields, based on the information you entered previously. If this situation occur to you and you need the EJB container to manage transaction, use stateful session beans – just keep them small as small as possible and remove them as soon as possible is a good tip for using it.

4.2.2.3 Message-Driven Beans
The latest breed of EJBs is called message-driven beans. Message-driven beans essentially wrap a JMS (Java Message Service) queue and allow the full power of JMS to be used. I have not explored much in this area because it does not relevant for the project but from my past experience using Message-Driven Beans can be useful in the same place that using JMS. For example, When you want to transfer data to back end messaging system such as MQSeries the result of this is may be to update information of your database and return the response back to client. You may use Message-Driven Bean can to also leave the response asynchronously so that using message-driven beans, you can originate the transaction and look for the response later or pass the response to other process.

4.3 Conclusion
The personal tutor system is a web-base application and I would rather take advantage of web-container management to manage its session rather than using EJB severs therefore, I choose to use stateless session beans to do most of the tasks for the business method for the system. This is not only easy and powerful to implement but also it has an impact on the performance of the application. Entity Bean is also using to model all the entity model of the database this is for one main reason is to reuse and shared data across the system. The next chapter is talking about the database design and where the entity fit in.

Still there are lots to mention and explore in this areas, J2EE is such a huge area and lots details detail has been left out to discover and difficult to be master. So, writing good EJBs need experience and carefully design if you want to learn more about it please visit http://www.java.sun.com.

5 Database Design

5.1 Introduction
This chapter details the development of the database design. Most of the entities are arise from the requirements capture during the interview of users. (See Appendix A for more details).

5.2 **Entity**

Our entities will be modelled to mirror real-world relationships and will be designed to support processing business transactions and managerial decision making. So by creating these entities they should help us to understand the structure and meaning of our data. Each entity contains number of attribute which describe an entity and contain the entity’ data we want to store. Each entity must also have a unique identifier called primary key which have to obey entity integrity rule that primary of each entity can not null. These entities for personal tutor system are the following:

5.2.1 **Student**

The student information need to be kept secure on the database therefore, it has to have its own entity. At this state we are not quite sure what field would be appropriate to describe them but at least the following must be described for the table fields, Username, Password, Firstname, Lastname, year, Assigned_Personal_Tutor_Id and Assigned_Director_of_studies.

5.2.2 **Personal Tutor**

The personal tutor information must also be kept to as its own entity. The personal tutor is described as the follow Username, Password, Firstname, Lastname and department.

5.2.3 **Director of studies**

The director of studies information must also be kept to as its own entity. The director of studies is described as follow Username, Password, Firstname, Lastname, and department.

**Note:** Personal tutor and the director of studies are having the same fields this is because they can be the same person for example director of studies also play the role of personal tutor and role of director of studies at the same time. But for the future use separate them into different tables can be easier to maintain.

5.2.4 **Problems**

The problem is kept as its own entity and this is what we have to record. Many problems are belong to particular student however many problem are recorded by many users for example, personal tutor, director of studies, can record problems. Each problem can also belong to particular category. For each problem they can be also associated with other problems. So, the problem is described as follow Problem_ID, Recorded_By, Roles, Access_Level, Title, Description, Category_ID, Student_ID, Problem_Related_ID and Created_Date.

5.2.5 **Sections**

Each problem is broken down into many different sections which user can enter information and specify the level of security that who should be able to see its problem. Nothing interesting about this section accept for foreign key that link it back
to the problem tables. So section describes as follows Sections_ID, Problem_ID and Description.

5.2.6 Sections_user_access
This table is specifying which users and which problem and which sections they are allow to access. This table will be used for checking authentication when users want to view the problem or other related problem. This table will be updated automatically according to users who record the problem and specify authorisation for other users to access. The default is the person who records the problem and the student who own his problem. Section_Users_Access is describing as follows Section_User_Access_ID, Section_ID, Problem_ID, UserName, Roles, Description, Review From and Review_To.

5.2.7 Actions
Each section will have one or more actions associate with it therefore for a particular problem and particular section there will be one or more actions associated to it. An action can be either for Student or User who record the problem and other services that may involve or participate for the action of the student. So they can be described as follows Action_ID, Problem_ID, Sections_ID, Category_ID, Title, Description, UserName, Related_To_Student, Action_For, Created_Date and Service_ID.

5.2.8 Category
This table record the category detail own by users and students. Because users who record the problem are allowed to create their own category for both problem and actions therefore, we have to record them separately and specifically for each user. So they can describe as follow. Category_ID, Title, Description, Category_For, UserName, Related_To_Student, Created_Date

5.2.9 Users
This table will be used by Application Server to authenticate the access of the users described as Username and Password

5.2.10 UserRoles
This table will be used by Application Server to authenticate the access of the user described as Username and userRoles

5.2.11 Services
This table record all the services that will provide to the users to select. This described as Servies_ID, Title, Description and Created_Date.

5.2.12 Next_Services_ID, Next_Actions_ID, Next_Problems_ID, Next_Sections_ID, Next_Sections_User_Access_ID
These tables are used to store the primary key so that it will be used for retrieving the unique primary key for services, category, problems, actions, sections_users_access and section tables. Therefore, we can make sure that every insert record into our database is unique.

5.3 Relationship
Each entity in the relational database is related and the reason why we do not want to store our problem in a single big table because the following reasons:
- **Insert anomalies** for example, if we create a single problem table without any other related table then what if we want to create a student record then it means that we have to create the problem as well. This is basically not true for the real world because student may or may not have any problem during the year.

- **Delete anomalies** so the same reason above apply when we try to delete one problem from the problem table then we end up delete the record of the student.

- **Update anomalies** so if we have a single big table update piece of data mean change of every rows so that it is not efficiency and increase the redundancy so if we create a link then we only have to update only one place.

In our database design each entity are related with a pair of primary key and foreign key. Our relationship also obey to the entity integrity and referential integrity rule which sate that “No component of the primary key of a relation may not be null” and “A database must not contain any unmatched foreign key values” (Watson).

Using concept describe above can represent our database model of the personal tutor system as the following. (See Figure 5.1)

The relation shown above may seem complicated but it is actually not I will explain step-by-step

### 5.3.1 The relationship of users and userroles
The relation between users and user role is from one-to-many relationship. A single user can have one or more roles associated with them. These two database tables will
be also used by JBOSS server deploy descriptor for authenticate access when users are log in.

5.3.2 The relationship of Director of studies, Students and Personal tutor and users and userroles
All username and password in director of studies, Students and Personal tutor tables will have to be inserting into users and userroles tables. The main reason for this is because JBOSS sever needs it for authenticate and authorise the users. In future, other external users tables can be created separately and store part of their information namely username and password so that they can gain access to the personal tutor system.

5.3.3 The relationship of problem and category
There is one-to-many relation between problem and category so that you can have many problems belong to one category.

5.3.4 The relationship of actions and category
There is one-to-many relation between action and category so that you can have many actions belong to one category.

5.3.5 The relationship of problem and sections
There is one-to-many relation between problem and sections since a problem consists of many sections with belong to a problem.

5.3.6 The relationship of problem and actions
There is one-to-many relation between sections and actions since each section can have its own action associated to it and again in turn one or more actions belong to particular sections which one or more sections belong to specific problems. That’s why there are many PF (Foreign key) in actions table.

5.3.7 The relationship of actions and services
The relationship between actions and services is one-to-many since each action will have one or more services involve..

5.3.8 The relationship of sections and sections_users_access
There is one-to-many relation between sections and sections_users_access. For a particular section there will have one or more users’ access to it. So instead of allowing the user to access the whole problem we break it down into different sections and each of sections specify the security issue associate to it.

5.3.9 The relationship of users and sections_users_access
There is one-to-many relation between users and sections_users_access. For a particular user they can access many sections. So that if you look back at 5.4.7 then the users entity and sections entity are forming many-to-many relationship.

5.3.10 The relationship of problem and related problem
There is one-to-one recursive relationship between problem entities itself since one or more problems can be related to other problems. However, this relationship does not present in figure 5.1 because of software that used to generate this diagram does not support recursive relationship.
5.4 **Normalisation**
Normalization is a method for increasing the quality of database design. I am using the work by Codd to identified first (1NF), second (2NF) and third (3NF).

5.4.1 **First normal form (1NF)**
A relationship is in first normal form if and only if all columns are single-valued so that in our personal tutor database every row within each table has the same number of columns. So that it is in 1NF.

5.4.2 **Second normal form (2NF)**
A relationship is in second normal form if and only if the database is in first normal form and all non-key columns are depend on the component of primary key. So, within our personal tutor database there is no composite primary key so that each table has it own unique primary key so that we meet 2NF.

5.4.3 **Third normal form (3NF)**
A relationship is in third normal form if and only if it is in the second normal form and has no transitive dependencies. I think the personal tutor system also meet 3NF because there are no transitive key dependencies.

5.5 **Implementation**
A full list of the SQL commands used to create the database can be founded in Appendix A.

5.6 **Conclusion**
A full design of the personal database has been designed and to capture all entities that would represent our real world problem as much as possible. This database has also been design in such a way that easy to query using simple SQL form from the web application where possible. Doing so would increase the performance of our web application when we are creating and modifying our data. See Appendix B for SQL use for personal tutor system.

One limitation of relational database that I found here is that once the database has been design it is quite hard to change and to be honest to you good solid properly design database can be quite hard to get it right. I must admit that also the first design of personal tutor system may have to change over and over due to the requirement change and the developer gain more understanding of the requirement in the future.

Therefore, making change of this can be difficult not only in term of migrating old data from existing database to the new one but also the developer have to redefine the definition of entity bean to correspond to this change. Moreover, our personal tutor database has been design to capture what is called hierarchical data, where there are number of relationship associated with other relationships. Therefore, inserting and querying these data require the developer to write a number of SQL statements that use the using join operation to interact with number of different tables.

So that is it and this is the way it goes for relational database and how we structure and modelling our data today using our knowledge of modelling real world entities and SQL query and using Normalisation rules.
5.7 XML and Database
As I mentioned in section 5.6 of the limitation using relational database to capture the hierarchical data was that you are ending up having multiple tables and you also have to use SQL join operation to get the data in and out. More than that the structure of your data is quite rigid otherwise you will not be able to put thing back and forth on the database and this is fine for normal data but the story is not quite the same for the hierarchical data structure.

Taking this project as an example supposing users came along and say they want to be able to specify what action and who can be response for that action in order to help the student do such and such and so on. Then what would you do at this stage so you may probably create a new table call “external user table” and add some fields to describe it and insert some primary key and foreign key to be able to links with other tables and so on and of course a new optimise SQL to query your new data and so on. So this is and it is actually happened in the real world the hardest thing of all is to interact with your back-end database. Especially if you are writing J2EE application you have to redefine your entity bean to correspond to your database table.

One method to overcome this limitation is to integrate XML with your existing database and use it to describe our record of student problem data until the final requirement has been met. XML could take care of the structure of your extension requirement for student data and keep the original design within the relational database. If anything has to change then we can add a new node into our XML file. (See figure 5.2)

As can be shown from above a new user has been added between actions tag that would represent who will be the users response to this action and so on. One of the difficulties of working with both XML and a traditional relational database is that the way we structuring our data. For Relational database we are dealing with rows and columns but with XML we are dealing with very hierarchical information just like present in Figure 5.2 therefore manipulate this XML structure require great deal of DOM, SAX, JDOM, XSLT and XPATH programming.

```
... ...
<category_problem>Academic Problem</category_problem>
<sections>
  <section id="1">
    <description>Lek has problem with his exam and need help</description>
    <actions>
      <action id="1" sectionid="1" problemid="1" studentid="ce9nw">
        Go to see Learning support
      </action>
    </actions>
    <users>
      <user>
        <username>cs0pt</username>
        <roles>personal tutor</roles>
      </user>
    </users>
  </section>
</sections>
... ...
```
Another concept of capturing our data in XML format also related to something called **denormalization**, in which we do the opposite from normalisation so instead of putting data across separate tables we capture everything in one XML format. Therefore when you want to query data you do it in one place in and XML data. So that this again dirty trick that will make your database faster. In relational database world this is the idea of creating a virtual table using VIEW command.

Even though XML help us to describe and structure our data to store an XML can be difficult unless we end out using XML database however, the technology of XML database is quite new and not very mature yet and people do not want to throw away their solid rock relational database therefore, perhaps we could store all of the information in one table in the relational database; see figure 5.3

<table>
<thead>
<tr>
<th>Problem_id</th>
<th>Category_id</th>
<th>Problem_XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>XML Document</td>
</tr>
</tbody>
</table>

Figure 5.3 Storing XML

So, for entire new requirements from the users can be kept in a chunk of XML format and since we have one XML document representing each extend problem, we can get all the information for an entire application from one field using the SQL below.

```sql
SELECT Problem_XML FROM problems where problem_id = '1'
```

And still we can keep our existing tables in our existing relational database and for the new extend requirements related to our problems data we keep it in an XML format.

**5.8 Conclusion**

The method mention on the section 5.7 works best when your database is a staging database or a temporary holding place for data. Moreover, this is also for adding your capability into existing application and to make a great deal out of extensibility of XML.
You may also notice here that the data is not normalising at all as we can store in a large chunk. We are also eliminating joins and complex SQL statements for retrieving those data. However, it does make it harder to write other applications that might want to use parts of that data, since those parts aren’t directly exposed. So that I am sure you are all agree that you will have to move those XML data into a relational database some days once the real requirement has been established by using some object to read XML from the staging database in to your back-end for posterity. And for this personal tutor system this could be the way forward.

My last word would be keeping the existing relational database because it provide you with great the structure of modelling your data and SQL statement can also be optimised to suite your performance. The security of RDBMS is still matured than other XML databases out there and for other reasons thought big solid relational database company also putting load of their resources to integrate the XML capability into their existing product. So for us make use or integrate the use of XML for extensibility and keep our solid of database design in relational database is good way to do

6 Interface Design

6.1 Introduction
This chapter examines the various considerations that have been taken into account for the personal tutor system interface.

6.2 Log in screen
Log in screen See figure 6.1 this is nothing much exciting because all we present here is to use the text box for user to enter his/her own user id and password.

However, what interesting is something behind the scene when the user name and password are get validate by the JBOSS server so to do that you got to configure login-conf.xml which can be found on your JBOSS_HOME directory. See figure 6.2

```
<application-policy name="personaltutorsystem">
  <authentication>
    <login-module flag="required" code="org.jboss.security.auth.spi.UsersRolesLoginModule"/>-->
    <login-module code="org.jboss.security.auth.spi.DatabaseServerLoginModule" flag="required">
      <module-option name="dsJndiName">java:/DefaultDS</module-option>
      <module-option name="principalsQuery">select passwd from Users where username=?</module-option>
      <module-option name="rolesQuery">select userRoles, 'Roles' from UserRoles where username=?</module-option>
    </login-module>
  </authentication>
</application-policy>
```

This page needs to have a HTML form that posts to the URL j_security_check; the form should have two fields for entering the username and password, called j_username and j_password. When the users try to access secure resources, the web container will send the login page to the use before access for the requested resource is granted. Once the user is successfully authenticated, the server checks whether the user has the required role for accessing the resource.

6.3 Welcome page
This welcome screen (see figure 6.3) is obviously the first place they can users will start when exploring this web application so and the users who will be using this
application would be presented with the list of their tutees or list of students he/she have to help.

On the top of the screen, the user can also see their username display, who is logged on to the system—obviously, it will be the same personal of using this. However, the reason to put that in is to show that the system is response to only that user at that time. To make it easier for users to access it, a link for displaying information of particular student is displayed for the user to select.

6.4 Student view page

Once the user clicks on the student name that they want to help, then for that particular student, the system will make a list of all the problems related to the selected students.

6.4.1 Cognition

Students the system will make a list of all the problems related to the selected students.

Student view page present the interesting issue of cognitions in the Human short-term memory concept. So that we are not expecting all the tutors to remember all the
problem or keep track of what is really going on in the past related to their students therefore, the list of the problems are displayed. The information presentation at this state also crucial, the system limited not to present too much of information only display title, date and time and level of access for the users to read. Simply because what will happened is that the user will sit next to the student to investigate his problem so he can always refer back to the student and have better face-to-face communication with the students.

The screen also contain large amount of white space that will make it clean looking interface and help users to focus on the important issue. The table is used as well as columns and rows to group together related information present to the users.

All functionality contain with the system are labelled with the links so that the users can access via the hyperlinks. Icons and Buttons are not used within the system because they are boring and making thing more complicate for the look and feel for example, some people may feel that they have one meaning to them but other will not feel the same but for the links it certainly tell them that they will get some where safe.

6.5 View student problem page

View student See figure 6.5 page is pretty much using the HCI principle describe for section 6.4

![Image](Image)

**Figure 6.5 View Student problem page**

Each sections are grouped with the heading for example for you have heading title say problem title and below that is the problem detail that user fill in and so on. The related problem sections is interesting again we use hyperlink that will take you too another pages and that mean user can explore other problem quickly too.

6.6 Add student problem page
Adding or recording the information of the student is a heart of this software system therefore the great time and effort are putting together to design this user interface to get it right.

The form has been used to capture the record of problem information and the most common form object such as drop down menu, check box and radio box are used. The drop down menu on the right hand corner allow users to select problem category from the drop down list if there can not find the one they like they can also enter the own favourite category on the text box right next to the drop down men. This is great because uses can automatically create their own problem category on the fly and safe time. And because these create category is related to specific user the system will customise this and there will be entered in the drop down menu automatically so next
time when the users are log in and record the problem with the same student all those category they previously created will be presented in the drop down menu.

All in one, because the system display the form in one whole page so that the user can see everything rather that have to move back and forward like other systems and to have to remembering what they have filled before and normally it create more errors for it. All text and buttons are labelled and using clear fonts and cool colour so make it more look and feel friendly to the users. Checkbox is provided where relevant for example on the problem related sections the users can check just like they tick of the list of their shopping list so this makes it more physiologically naturally to the way users do things. Text area of each section is broken down so users can fill in the information and each text area is word wrap so they can be in a very nice format.

6.7 Error Recovery
Errors should be catch and users love to make errors therefore, the system is designed to help users to prevent the user to enter the right information into our system. The displays of the messages are carefully chosen and using positively worded and user oriented. (See Figure 6.7)

![Error page](image)

Negative wording can make your system resent really badly and make user feel embarrass and defensive. The message are also do not use too formal or technical as the user would not understand the computer jargon. Instead it using simple English and telling them what they did wrong and therefore rectify their behaviour. The systems will not make any noise because it will annoy and embarrass the user themselves and students who will be aware of their information being record incorrectly.
6.8 Conclusion
An interface of the personal tutor system has been designed for the usability in mind the knowledge of human computer interaction has been put together such as navigation, cognition, short-term memory of user and the system to be usable and interactive at the same time. The system also taking care of error recovers in order to help user make less mistake as much as possible.

For the technical side aspect every single page of this system are define using purely HTML and CSS mark up therefore it is easier to read and maintain and can be browse able on any platform that can access to the internet. Every page also save as .jsp file which allow us to embedded java programming language which some developer found it useful to make any change or interact with the back end database or even insert some applet if we have to. So the point I try to make here is we have can have separate presentation layer from business layer and as a result this look and feel can be easily to maintain.

Another point that I will think of in the future of this web design to design this application for disables people to use it as well. The web accessibility is increasing more and more important and lots of sensitive issues which developer should concern when they design their application. This idea comes from HCI lecture on the web accessibility issues I found it very interesting and really open a new ideas or another IT area to investigate. I also have seen some of staffs and colleagues who are disables and because of personal tutor system will be used by these users so the web accessibility will have to be taken into the consideration. For more information of web accessibility please visit. [http://www.w3.org/TR/WCAG20](http://www.w3.org/TR/WCAG20)
7 Implementation

7.1 Introduction
This chapter describes the personal tutor system web application. The personal tutor system has only one client: It is a web client used by personal tutor, director of studies and students and for the future there will number of other users such as Learning Support, Medical Centre, AWARE and so on. The web client access and modify the information of the students which will be stored in a database through enterprise beans. The Personal tutor application has still does not meet the customer requirements yet and should not be considered as the deliver product to the customer yet this is due to the complexity of the requirement itself and the time constraints. This application then instead develop to demonstrate how all the component of J2EE are put together to provide a simple but functionality application and how to develop and design an n-Tier Application. By doing this approach will have numerous advantages for the developer to maintain and develop further functionality for the users in the future.

7.2 Requirement Analysis
The full set of requirement analysis can be found on the Chapter 3, but in this section part of it will be identified where the operation that the system’s user can perform as they interact with the system. The capture all these requirements we are using use case model which will be shown below. (See figure 7.1)

![Use Case Diagram](image.png)

Figure 7.1 Use Case diagram

case model which will be shown below. (See figure 7.1) from this use case diagram it demonstrate that they will be three basic users who will interact with the systems.
7.2.1 **Personal Tutor & Director of studies**
Personal tutor can perform the following tasks

- Authentication for before they can access and adding data Log in/off
- Add category
- Add problem
- View Report for students
- Add services
- Specify security level of access and grant which and who allow the access the detail of record.

7.2.2 **Students**
Personal tutor can perform the following tasks

- Log in/off
- View Report for student

7.3 **Mock up screens**
When users access the personal tutor application they will be first present with a log in page (See section 6 for all the mock up screens) that require their users and password for security requirement. All the web pages and resources on the system are protected by configure the web.xml file therefore; users are required to log in first. Once the users log in and the systems authenticate it the system will take the user to the view students’ page. These will be presented the user with the list of their tutee. At this stage the users can select the link of the student that they want to view. The system then will take the user to the view information page. This view information page will list all problems that related to this chosen students so that the user can either choose to see the existing problem that they recorded in the past or can choose to add a new problem. If they choose to view the existing record they can click at the hyperlink title of the problem and the system will display the page where they can see the information of existing record. But if user want to add a new the problem then they can click at a add new problem hyperlink and a adding new page will be displayed.

7.4 **Choosing the right technology**
The important point after doing the requirement analysis is to selecting the technology to be used and as we know we will have to use enterprise Java Beans and JBOSS application sever will also consider the following:

- **Standards-based technology** using JAVA as a main language for develop web application is standard not only it will run on any platform the solution can be easily be founded on the internet because everyone using it and can answer your question. [www.java.sun.com](http://www.java.sun.com) is a good example.
- **Support for rapid development** I think this is pretty obvious for Java.
- **Extensibility** using JAVA you can extend core component business object and integrate web services into your system.
- **Good quality tool support** All tools for Developing Web application are available out there for example NetBeans IDE, JBOSS server, MySQL database. Tomcat and Struts from Apache.
- **Vendor-neutral and low-cost implementation** You can get all these for free.

Taking all these consideration then we are on the right track and ready to get yourself for some fun.
7.5 Application Architecture
There is worth spending more time on the architecture of the system. Even though you can not complete the whole application yourself but with a good solid model of your design you can partitioning the your application and shared it with other developers that we safe lost of time and cost in the long run and will be easier for maintaining the whole system. Therefore, the important factors that we need to consider in formulating architecture for the system include:

- **Proper partitioning** of the application with modular, component based design so that each module can be simple and not interact to other modules in a complicate way so in other word we try to decouple it.
- **Maintainability** this will come when you did a proper partitioning
- **Separating presentation from content**, the module can work on its own even without having knowledge about the presentation one way of doing that it via the interface and passing parameters along therefore, if in the future we want to have a new look and feel then it will be easier to change.

Keep this picture in mind and apply to personal tutor system we will arrive with the MVC paradigm describe as the following:

**Database access logic component:** This candidate will be entity beans and we will use Bean Manage Persistence (BMP) type of entity bean the reasons is simple because the simplicity over CMP and flexibility for changing a new database in future. So for each tables corresponding in our database there will be an each entity beans. They are the following:
- ActionBean
- CategoryBean
- DirectorofstudiesBean
- SectionsBean
- StudentsBean
- ProblemsBean
- SectionsUsersAccessBean
- ServicesBeans
- PersonaltutorBean

The idea of using number of different entity beans is to map into the structure of the database and allow the business logic component to reuse these entities bean as they can share the information across.

**Business logic component:** This candidate will be Stateless Session Beans because they are fast and easy to implement and they are not tight to one particular client so that they can increase the performance of the personal tutor database application. These session bean will provide the workflow of the business logic of the program and also maintaining the various entities beans above. However, some of the database access method will be also added in the session for fast operation where it suitable. They are the following:
- AccesBOControllerBean
- StudentsBOControllerBean
- ProblemscontrollerBean
**Dependent Value component:** This candidate is simple Java classes custom serializable object. Dependent value classes are useful for packaging data and moving across between entity beans and remote clients which could be session beans or even web client from the entity abstract persistence model. By doing that will allow us to change our entities beans without impacting existing clients. You see this is why we are separating them into layers. What goes in dependent value class is nothing fancies than setters and getter methods they are the following:

- ProblemStudentDetails
- ProblemDetail
- AccessTitle

**Model for web component:** This candidate is a Java Class that will wrap up our Session beans and will be use it for integrating our Enterprises beans with the web components part. Therefore, all web components need to know in order to communicate is how to and what parameter they can pass it too this model component and all the rest will be take care by our business objects. So that we are simply decouple this application.

**Web component:** This candidate is our last candidates to render the user interface we are using number of JSP pages and Strut template combine them together. Some of the StrutForm and StrutAction classes from Struts framework could also go here and I choose to take advantage of it by using there struts configuration XML file to handle all the requests from the users and forward it to other pages. Rather than having to learn all the html struts tags which can be quite complicated and time consuming so I leave the validation part for the business logic and whatever errors occurs can be throw back to Model web component and it will display it back to the users.
Figure 7.2 gives a high-level view of how the components interact. The figure 7.2 show how each component for Personal Tutor System are lay out in the J2EE server. The entity beans on the right hand side are mapping and interact with the back-end database. The Session beans which are acted like a client for entities beans and maintain the relationship between all the entity beans. The web container includes number of JSP pages and Action Classes which use to communicate with the Session Bean via the business layer object. The details of Enterprise beans and Web Client are discussed in the next section.
7.6 **Session Beans**
The personal tutor application has three session beans: AccessBOControllerBean, StudentsBOControllerBean and ProblemControllerBean. These session beans provide a client’s view of the application’s business logic. Behind the scene hidden from the clients are the sever-side routines that implement the business logic, access databases, manage relationships, and perform error checking.

7.6.1 **ProblemControllerBean**
The business methods of the ProblemControllerBean session bean perform tasks that fall into the following categories: creating ProblemsBean entity beans, add problem for the students by coordinating and managing relationship with other entities beans. The following methods create entity beans:
- createProblems

This methods of the ProblemControllerBean session bean call the create methods of the ProblemsBean entity bean. The createProblems method throws an IllegalUserTypeException and ProblemNotFoundException if the given parameter types are wrong and the same problem already exists. For createProblems to verify whether the problems has been already existing or not it invokes the findByPrimaryKey method of ProblemsBean entity bean. If the result of this verification returns empty string of problem id then the createProblems throws a ProblemNotFoundException.

The following method is adding a new problem:
- addnewproblem

This is one of the most complicated methods for the personal tutor system. All the entities beans on this system have related and forming a relationship such as ProblemsBean entityBean and SectionsBean have a one-to-many relationship and SectionsBean and ActionsBean have one-to-many relationship and so on. Because the entity beans use bean-managed persistence, so that all the relation are managed by using the combination of SQL statement and sub methods call.
For example, adding a new given problem addnewproblem method requires a single parameter which is ProblemDetail a dependent value component object. ProblemDetail object encapsulate all the details of a new problem that required by addnewmethod. Once the addnewproblem method received ProblemDetail object it will then extract all information by calling a set of getters methods of ProblemDetails. Once all the parameters need to be inserted into the database are set the addnewproblem methods start verifying for example whether sections already exists, category already exists and so on. Again the verifying method is invoked by calling findbyXXX within our entities beans. Then to create a relationship and forming the model for our new problem, the addnewproblem will then insert a row into various tables which corresponded of our entities beans. Again inserting or creating data on the tables will be handled by our related entity beans.
7.6.1 **StudentsBOControllerBean**
The business methods of the StudentsBOControllerBean session bean perform tasks that fall into the following categories: creating StudentsBean entity beans, Listing the problem of students and managing students account.

The following methods create entity beans:
- createStudents

This methods of the StudentBOControllerBean session bean call the create methods of the StudentBean entity bean. The createStudents method throws an IllegalUserTypeException and StudentNotFoundException if the given parameter types are wrong and the same student already exists. For createStudents to verify whether the student has been already existing or not it invokes the findByPrimaryKey method of StudentsBean entity bean. If the result of this verification returns empty string of student id then the createStudents throws a StudentNotFoundException.

The following method is adding a new problem:
- ListStudentProblem

This is a business methods call by the client to list all the problem title and some of the details and return it to the users. Each of the problem detail that return will result in create a new ProblemStudentsDetails independent object and add it into the ArrayList therefore, all client have to do is to extract the information from this independent object.

The following method is adding a new problem:
- reassignedDirectorofstudies
- reassignedPersonaltutor
- rolledYearForward

7.6.1 **AccessBOControllerBean**
The business methods of the AccessBOControllerBean session bean perform all tasks that related to security issues for this personal tutor system. The business methods of the AccessBOControllerBean session bean perform tasks that fall into the following categories: authorising the user for viewing the student problem record, get problem for the students by coordinating and managing relationship with other entities beans.

The following methods authorising the users:
- authoriseuser

This authoriseuser method invokes when the user want to view the particular problem associated with particular users. Because each problem can have many sections related to it and each sections is defined a security constraints associated with number of users who allow seeing this record within that sections. Therefore what authoriseuser simply does is to query and check all the reference in the sections_users_access table and using join SQL statement to query across multiple table and validate data return from it. If it match to the definition rules for allowing users to access then it will return true otherwise return false than mean users are not allow to access the chosen record.
The following methods get the problem related to student:

- **ViewStudent**

Once the authoriseuser method returns true value then ViewStudent method invokes. This is nothing complicated just another SQL query to get the data out from the database. All the security hard work is supposed to be handling by authoriseuser method. And for usual trick AccessTitle dependent object will be returned from ViewStudent method to capture the model of our problem record detail. AccessTitle object is carefully designed to model to reflect our problem structure which contains title and sections and actions and so on. AccessTitle object then will be used to move the data across between Sessions Client to Web Clients.

### 7.7 Entity Beans

Entities bean are used for corresponding to our table in personal tutor database. The personal tutor system has the following entities beans.

- ActionBean
- CategoryBean
- DirectorofstudiesBean
- SectionsBean
- StudentsBean
- ProblemsBean
- SectionsUsersAccessBean
- ServicesBeans
- PersonaltutorBean

The purpose of these beans is to provide an object view of these database tables: actions, students, problems, sections, sections_users_access, services and personal_tutor. For each column in a table, the corresponding entity bean has instance variable. The Bean-Managed-Persistence (BMP) type is used for these entities because there are simpler to write and gives more flexibility over use of number different databases the details of that can be found on the ejb-jar.xml file on the Appedendix B. Our BMP entities contain the SQL statements that access the tables. For example the create method of StudentsBean entity bean calls the SQL INSERT command.

Entity beans do not validate method parameters except for ejbCreate( ) all the validation will be handle by Session Beans.

### 7.8 Helper Classes

The EJB JAR files include several helper classes that are used by the enterprise beans. The source code for these classes is in com.proj.util directory. Table 7.3 briefly describes the helper classes.
Table 7.3 briefly describes the helper classes.

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProblemStudentDetails</td>
<td>Encapsulates the state of a ProblemsBeans entity bean used by StudentsBOController sessions bean. This class contains getters/setters methods</td>
</tr>
<tr>
<td>ProblemDetail</td>
<td>Encapsulates the state of a ProblemsBeans entity bean used by ProblemsController sessions bean. This class contains getters/setters methods</td>
</tr>
<tr>
<td>AccessTitle</td>
<td>Encapsulates the state of a All entity bean used by AccessBOController sessions bean. This class contains getters/setters methods</td>
</tr>
<tr>
<td>CodedNames</td>
<td>Define the strings for the lookup method (For example: java:comp/env.ejb/Students</td>
</tr>
<tr>
<td>DBHelper</td>
<td>Provides methods that generate the next primary keys</td>
</tr>
<tr>
<td>EJBGetter</td>
<td>Has methods that locate and return home interface (for example, getStudentsBOControllerHome)</td>
</tr>
</tbody>
</table>

### 7.9 Database Tables

A database table of personal tutor system may be categorised by its purpose: representing business entities and holding the next primary key.

#### 7.9.1 Tables representing business entities

All these tables are explaining in the database design sections for each tables present on this database there will be an entity bean corresponds to it.

#### 7.9.2 Tables that hold the next primary key

These tables have the following names:

- next_problems_id
- next_category_id
- next_sections_id
- next_sections_users_access_id
- next_actions_id
- next_services_id

Each of these tables has a single column name id. The value of is id is the next primary key that is passed to the create method of an entity bean.
7.10 Protecting the Enterprise Beans

One of number of reasons than makes J2EE application is so secure is you can protect an enterprise bean by specifying the security roles that can access its methods. In the personal tutor system there are for example two roles personal tutor and students are defined and each of them can perform different operations.

A personal tutor role allows to perform add new problem operation function by calling addnewproblem method but student can not perform this in order to do that you have to do the following:

- Declare method permission this involve configure the jobs-ejb.xml for example
  ```xml
  <assembly-descriptor>
    <security-role>
      <role-name>Personal Tutor</role-name>
    </security-role>
    <method-permission>
      <role-name>Personal Tutor</role-name>
      <method>
        <ejb-name>ProblemsControllerEJB</ejb-name>
        <method-intf>Remote</method-intf>
        <method-name>addnewproblem</method-name>
        <method-params>
          <method-param>com.proj.util.ProblemDetail</method-param>
        </method-params>
      </method>
    </method-permission>
  </assembly-descriptor>
  ```

- Using programmatic Security in the EJB Tier
  Programatic Security of EJB Tier consists of the getCallerPrincipal and the isCallerInRole methods. For example you can determine whether enterprise’s bean caller belongs to which roles by invoke the following.

  ```java
  boolean result = context.isCallerInRole("Personal Tutor");
  ```

So, that in future the developer can implement more sophisticated method for administrators’ user roles which would have roles for manage all the students, personal tutor and director of studies accounts and take advantage of resources protected of EJB and define the methods to perform various tasks as describe above.
7.11 **Web Client**
In the personal tutor system application, the web client is used by personal tutor, director of studies, student and number of staffs in order to record the problem of students. See table 7.4 for more detail.

<table>
<thead>
<tr>
<th>Function</th>
<th>JSP Pages</th>
<th>JavaBeans Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homepage</td>
<td>home.jsp and homeContent.jsp</td>
<td></td>
</tr>
<tr>
<td>Log on or off the application</td>
<td>login.jsp and loginContent.jsp</td>
<td></td>
</tr>
<tr>
<td>List student for tutor</td>
<td>viewStudent.jsp and viewStudentContent.jsp</td>
<td>viewStudentAction</td>
</tr>
<tr>
<td>View problem</td>
<td>viewproblem.jsp and viewProblemContent.jsp</td>
<td></td>
</tr>
<tr>
<td>Display students</td>
<td>display.jsp and displayContent.jsp</td>
<td>studentAction</td>
</tr>
<tr>
<td>Add new problem</td>
<td>addproblem.jsp and addproblemContent.jsp</td>
<td>validateaddproblem.jsp</td>
</tr>
<tr>
<td>Template</td>
<td>template.jsp</td>
<td></td>
</tr>
</tbody>
</table>

7.12 **Design Strategies**
The main job of JSP pages in the personal tutor system is presentation. We are also using struts template tag mechanism to maintain a common look across all the JSP pages. Therefore, if we want to change the look and feel later on we know where to change. The JSP pages use enterprise beans to handle interaction with the database. Some of the enterprise Java beans are embedded within the scripts for the fast interaction however, this should have been changed because it will have make JSP pages more difficult to maintain. Struts Action Classes has been use for JavaBeans Components acted as a front end to a database or as a façade to the interface provided by and enterprise java bean. For more detail please look at the source code in the CD-ROM.

7.13 **Security**
For personal tutor system the main security aspect are the following:

7.13.1 **Authentication**
Form based Authentication has been used in the personal tutor system but this is only partial secure *In form-based authentication, the content of the user dialog box is sent as plain text, therefore, the target server is not authenticated. Basic authentication sends user names and passwords over the Internet as text that is uuencoded, but not encrypted. This form of authentication, which uses Base64 encoding, can expose your user names and passwords unless all connections are over SSL. If someone can*
intercept the transmission, the username and password information can easily be decoded (java.sun.com)

Since we are installing the software in a secure environment therefore this is acceptable however it would have been more appropriate if we could change it. The authentication to be used within the web application is specified in the web.xml deployment descriptor file and the snippet from the web application deployment descriptor shows below:

```xml
<login-config>
  <auth-method>FORM</auth-method>
  <form-login-config>
    <form-login-page>/login.jsp</form-login-page>
    <form-error-page>/login.jsp</form-error-page>
  </form-login-config>
</login-config>
```

7.13.2 Authorization
For J2EE web application within web container, the resources may be secured by defining the security constraints in the deployment descriptor, as show below:

```xml
<security-constraint>
  <web-resource-collection>
    <web-resource-name>/viewstudentAction.do</web-resource-name>
    <url-pattern>/viewstudentAction.do</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>PersonalTutor</role-name>
  </auth-constraint>
</security-constraint>
```

This state that the web resource collection accessed by the request URL viewstudentAction.do can be accessed only by users having the role PersonalTutor. What is interesting about this is you can just configure it straightway in web.xml files. If you remember previous section of protecting resources of EJB it is similar but that can be done by programmatic security. So that’s why J2EE platform is so secure.

7.13.3 Security Realms
Security realms come into the picture when you define as a logical group of users, roles and access control lists. In personal tutor system the access control and the roles are defined within the web application deployment descriptor, but mapping the physical users and groups to these is done using the web-container in JBOSS.

The basic tasks for the JBOSS web-container needs to perform where user’s accesses a secure resource are:
- Check whether the user is already authenticated.
- If not, prompt for the credentials
- Check whether the credentials supplied exist in the security realm the container is using
- Check whether the authenticated user belongs to a group authorized to access the requested resource
The realm for JBOSS support also based on JDBC database access and use two database tables for performing authentication and authorization: the first table needs to store the usernames and passwords, and the second defines the roles for each user has been assigned. To specify that we want to use this realm, we add an entry to login-conf.xml file of JBOSS_HOME directory.

The example of this file is shown below.

```xml
<application-policy name="personaltutorsystem">
  <authentication>
    <login-module flag="required" code="org.jboss.security.auth.spi.UsersRolesLoginModule"/>
    <login-module code="org.jboss.security.auth.spi.DatabaseServerLoginModule" flag="required">
      <module-option name="dsJndiName">java:/DefaultDS</module-option>
      <module-option name="principalsQuery">select passwd from Users where username=?</module-option>
      <module-option name="rolesQuery">select userRoles, 'Roles' from UserRoles where username=?</module-option>
    </login-module>
  </authentication>
</application-policy>
```

7.14 Conclusion

I am so glad that to come to this far. There are also lot more details to cover for more information please look at the source code in the CD-ROM. This chapter should illustrate that how we can put together all the J2EE technologies and product in a simple application but functional. The application almost develop pretty more follow the principle of Software Engineering concepts and use the standard UML notation like use cases to help the developer come up with the software analysis. The heart of this application may not be the functionality but a multi-tiered approach architecture design which have really stand out and give numerous advantages for future development. Some of the advantages are the following:

- Reduce complexity for developers. For example the web developers who are writing HTML and JSP page do not have to know about EJBs. All they need to know is how to interface with the business layer (Sesssion Beans) via passing to an interface.
- A new database can be plug-in or change anytime you in future BMP is great flexible to use over number of different database vendor. Only change JDBC plug in and you are back to the business.
- More scalable than traditional client-sever application like the old 2 tier version. The data objects could be shared and sever more clients and when the application grows, extra server can be added to the environment, to share the load for Business Objects and Data Services Object which is not easy for 2-tier client-server.
- Increasing reusability which is obvious and Data Object like Entity beans can be shared and reused across number of application.
- Greater security because the user never access the data directly, always via a business tier. This can help to prevent data from being corrupted inadvertently or deliberately. However, it is not still suitable protection against a skilled and malicious programmer.
Some of the security aspect of web application also has mentioned and this is hopefully encouraging the developer in what to be aware of if they appear to extend the functionality of this personal tutor system application.

8 Critical Appraisal

8.1 Introduction
This chapter contains reflection on the degree in which the completed the project met with the expectations. Therefore, there will be a discussion of the lesson lean from the project, strengths and weaknesses of the personal system. Further suggestions of what can be improved or should have done better if the project were to be run again also discussed.

8.2 Problem Description
The aim of this project was to design, develop and implement a consultations system called “Personal Tutor Reporting and Recording System” which would be used to record interview information between personal tutor and students. The Personal Tutor System would also have to be scalable and secureable and using J2EE technologies. Because this project is not for commercial used and targeted only for supporting academic staffs and students at the university therefore all the software that were used on this project was fully open source and free of charge. During this project there were so many problems to tackle therefore the literature review was chosen to be the first tasks to carry out. The literature review explore in the number of different areas that should be useful and relevant for solving the project. There can be categories as the follow:
- Software Engineering Process
- Choosing the right technology and Architecture design
- Data Protection issues
Software Engineering Process explains how to structure our software development process and gather all the users’ requirements for the project and also provide some guideline that we must not ignore. In the mean time, we were also pay attention on our technologies choice for implementing the project. Because the main requirement of the software would be implemented with J2EE technologies therefore a lots of afford have been put in learning this technology right from the start of the project. Our last concern was the data protection issue that stated that all the information of students must be kept secured and protected at all time and that requirement had been influence on the design of the software.

8.3 The Resulting System
The resulting of this project was a three-tier system contain a MySQL database as a back end, a collection of entity beans and Session beans for the business tier layer and a collection of web pages written using a combination of HTML and JSP pages. The web-tier also using the Struts Framework to in order to take advantage of action class controller, template library tags and Struts configuration file which will keep control
when users make a request for a web page. The application was also designed in such a way that it easy to maintain and extend for the further requirements.

8.3.1 Strengths

Some of the basic requirements had been gathered using the appropriate methods and elicitation techniques. Set of uses cases and scenarios had been drawn so that it allows the developer to construct a set of requirement and identify a various solution to solve the problems.

This is not another web database application project likes other projects in the past and because of the problems that we try to solve here were more complex and hard to capture than a normal simple database web application so that why it really makes this project stands out and interesting. The relational database were designed in such a way that to captured a hierarchical data which involve having multiple tables forming complex relationship such as one-to-many, many-to-many and recursive relationship as a result of this there would have quite a number of primary key and foreign keys that needed to be carefully identified and handle in order to links all these tables together. All the tables on the database also had to be normalised and obey all the integrity rules which will make it easier to insert, delete and update the record. At every stage that the design decision had been made the developer also be aware of the performances for accessing and manipulation the data which must come first and especially for this particular type of database where there must be involved query multiple tables to get the data using join SQL statements. Therefore, all the SQL statement were carefully identified to be optimised and efficiency as much as possible. This is because we are knowing that the limitation of the relational database and how we can capture the data is rigid and the number of tables will always expand as soon as requirement changes or extend and the only tool that could save us time and money is our optimise SQL statements.

The further strengthen of this project is also based on the fact that we were using J2EE technologies to solve the problem. J2EE technologies offered a multi-tier distribution application model so that each components can be located on different severs, J2EE also offered a reuse component as can be seen on the project that all the concept of entities beans can be reused by the Sessions beans as our business objects. The J2EE enterprise also offered the granularity security model that we can use in our application either use it as programmatic security on the EJB-Tier or just add on entry of the web deployment descriptor (web.xml) on the Web-Tier which both of this gives us the flexibility for us the make our application more secure.

Personal Tutor System is developed follows the principle of the J2EE Architecture design and as a result of that the system result was in three-tier application because they are distributed over three different locations: Client machine (i.e. Web Browser), JBOSS server machine and the MySQL database back ends. Each layer is carefully designed as already mentioned on this report. The back-ends database using MySQL version 4 which is an open source relational database and support transactional
management. JBOSS is chosen to be a dedicated J2EE server as it is rock solid and support what is called “hot deploy” so it allows us to redeploy all the EJB jar files and WAR files without having to restart our server and it is also free. All the Entities beans, Sessions Beans and Java Components for the web-tier are also running on the EJB container and Web Container respectively provided by JBOSS. Finally, the Client machine which using web browser to access the system can be almost anywhere which can access to the internet. So, by developing three-tier application it makes our system more scalable and easier to maintain.

In future, any further change of a new requirements the system can be easily extended for example if the new business requirement changes then the developer can rewrite a new definition of Session Beans and reuse the components of entities beans and so on. Not only this give you the extensibility of the application but also all the headache tasks of the implementation can also be separated for example you make have a web developer to concentrate on the look and feel of your application and EJB developer concentrate on the server-side business objects and other developer concentrate on configure on deployment and integrate all the components and so on.

8.3.2 Weaknesses
Due to the learning curve of J2EE technologies and work load during the year so some of the functionality requirements could not be developed in the schedule plan and for this reason it a major weakness of this project. As a result of this failure it may lead to unsatisfied and meeting expectation of the customers which should not happen in the real world business and for career.

However, the simple application demonstrates a substantial software development skills and how to put all these J2EE components together and deliver the simple but functional application which consider matching up with the basic requirement of customers.

Another major weakness of this project was that the proper testing has not been carried out this is due to time constraint. The system could have been further improved by redesigned the system based on the findings of testing and this would have been ensured that the product fitted the requirement as closely as possible.

Even though J2EE technologies offers load of strength on scalability and portability for the application but the most drawback of J2EE that is adding the complexity on to your application. Imagine if you are a new entrant for Java and hit the J2EE project where will you start JMS, JSP, Servlets or an EJB. There are hundred of technologies involves on this platform so that it is nightmare to get it right. The concept of deployment descriptor JAR, WAR and EAR is also confusing and takes sometime to figure out what you should do. However, having mentioned all that J2EE still lead us to the better design it really depends on us what to put in and combine them to make it works for task by tasks basic.

The last point that I also wanted to make on the weaknesses of the system is a security issues because there was not a clear detail of how the security issues should actually be implemented apart from encrypted password during the transmission however, this could be done at least in two ways: First you can configure the JBOSS server to encrypt the password during the authentication process that uses HTTP over SSL. Once this process have been authenticated and users are authorized then the security could be handle by the business objects to access the data by placing the combination
of SQL statements to get access to the data. Second, you can also design which business objects should be performed by which role’s clients this could be done by configure EJB-Tier deployment descriptor.

8.3.3 Additional Functionality
It is likely that this project will have to evolve in changing on the new requirements or the next developers who also want to developer further functionality may consider the following areas:

Database System:
I have no doubt that the database back end will change often but if there was the case that more expensive database will be replace MySQL then changing on the JDBC driver and configure on deployment descriptor can switch to the new one easily. Because of the BMP entities beans has been used for the system to define and map all the corresponds tables on this database it will allow you to have a separate entity beans working on the separate databases therefore, you may want to use the XML database to sit along side with the relational database. The XML database is still not mature on the security side but this is not the problem if you want to store any temporary data and require fast retrieval. Because XML database does give you a flexibility to store your XML document on its own format and provide query tools such as XPATH and XQUERY to query and retrieve your XML data which is more hierarchal rather than using join SQL statements to query the data which is more rigid structure.
Furthermore, if you would have to embed a student report mechanism on to your system then you may also using XSL to create a PDF files, Word Document file and so on.

Business Logic:
This project only implements basic functionality that illustrate how the J2EE component are putting together and had not met the expectation of the client yet therefore more business functionalities are required to implement for this system. These are Emailing facilities, linking to SAMIS, Administrator function, Generating Printed Report. However, because the system is well designed all these tasks can be done by adding more roles into the Session Beans for example, administrator functions can be added into StudentBOController beans.

Web Accessibility:
To be truly accessible by anybody then the redesign of the web interface have to taken into consideration. Web Accessibility is the way forward for the web designer to pay attention to when designing a web application. Because all the functionality can be extend via the business tier and can be kept totally separate from the presentation layer. This then mean it allows the web developer to concentrate on a web accessibility issue rather than having to worry on the business logic side and how to interact with the database and so on. So it use give the web interface or usable and accessible by anybody.
8.3.4 Final reflection

If I were to repeat the project again I would create more realistic work schedule. The project plan that I was created contains large number of estimated time especially for the implementation and testing parts. However, I did not expect the concept of learning and understanding enterprise java bean would take me too long but obviously my assumption was wrong. This lesson had taught me how to prioritise my tasks more efficiently.

Despite the failure of the implementing the product that user wants, this project has taught me well of the concept of J2EE technologies and choosing the right technologies for the right tasks which should be more important for my future career and also allowed me to apply my software engineering skills for the project and participate on the discussion with my supervisor during my work.

If the time constraints and the complexity of J2EE were not be and issue I would like to develop more further testing using some of open source testing frameworks such as JUnit and Cactus which was the standard tools for testing EJBs. This would have also lead to carry out a heuristic evaluation as well.
9 Reference


Appendix A Structure of the database

9.1 Student
The student information need to be kept in secure on the database therefore, it has to have its own entity. At this state we are not quite sure what field would be appropriate to describe them but at least the following must be described for the table fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username(PK) (VARCHAR)</td>
<td>Primary key for each student so each student has its own unique username</td>
</tr>
<tr>
<td>Password (VARCHAR)</td>
<td>Each student requires password to access information this must be kept secure and use for authenticate and authorize access. This should be the same as BUCS</td>
</tr>
<tr>
<td>Firstname (VARCHAR)</td>
<td>Student has first name</td>
</tr>
<tr>
<td>Lastname (VARCHAR)</td>
<td>Student has last name</td>
</tr>
<tr>
<td>Year (VARCHAR)</td>
<td>Student has it year and this is can be rolled forward every year until the end of student course</td>
</tr>
<tr>
<td>Assigned_Personal_Tutor_Id (FK) (VARCHAR)</td>
<td>Each student will be assigned to a personal tutor or has got his own personal tutor. This can be also change during the academic year for some reason but mostly is stay the same during his years at university.</td>
</tr>
<tr>
<td>Assigned_Director of Studies (FK) (VARCHAR)</td>
<td>All student within his department will be assigned by a single director of studies. Again this can be changed but most of time it is only one director of studies</td>
</tr>
</tbody>
</table>

9.2 Personal Tutor
The personal tutor information must also be kept to as its own entity. The personal tutor is described as the following.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username (PK) (VARCHAR)</td>
<td>Primary key for each personal tutor</td>
</tr>
<tr>
<td>Password (VARCHAR)</td>
<td>Each personal tutor require password to access information and this should be the same as BUCS</td>
</tr>
<tr>
<td>Firstname (VARCHAR)</td>
<td>Personal tutor has first name</td>
</tr>
<tr>
<td>Lastname (VARCHAR)</td>
<td>Personal tutor has last name</td>
</tr>
<tr>
<td>Department (VARCHAR)</td>
<td>Each personal tutor works on its own department</td>
</tr>
</tbody>
</table>
9.3 Director of Studies
The director of studies information must also be kept to as its own entity. The director of studies is described as the following.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username (PK) (VARCHAR)</td>
<td>Primary key for each director of studies</td>
</tr>
<tr>
<td>Password (VARCHAR)</td>
<td>Each director of studies require password to access information and this should be the same as BUCS</td>
</tr>
<tr>
<td>Firstname (VARCHAR)</td>
<td>Director of studies has first name</td>
</tr>
<tr>
<td>Lastname (VARCHAR)</td>
<td>Director of studies has last name</td>
</tr>
<tr>
<td>Department (VARCHAR)</td>
<td>Each director of studies works on its own department</td>
</tr>
</tbody>
</table>

9.4 Problems
The problem is kept as its own entity and this is what we have to record. Many problems are belong to particular student however many problem are recorded by many users for example, personal tutor, director of studies, can record problems.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem_ID (PK) (VARCHAR)</td>
<td>Primary key for each problem</td>
</tr>
<tr>
<td>Recorded_By (FK) (VARCHAR)</td>
<td>Username of user who record this problem</td>
</tr>
<tr>
<td>Roles (FK) (VARCHAR)</td>
<td>Roles of users</td>
</tr>
<tr>
<td>Access Level (VARCHAR)</td>
<td>Each problem describe the level of access</td>
</tr>
<tr>
<td>Title (VARCHAR)</td>
<td>Each problem has it own title.</td>
</tr>
<tr>
<td>Description (VARCHAR)</td>
<td>Each problem contain description</td>
</tr>
<tr>
<td>Category_ID (FK) (VARCHAR)</td>
<td>Each problem having its own category</td>
</tr>
<tr>
<td>Student_ID (FK) (VARCHAR)</td>
<td>Each problem belong to particular student</td>
</tr>
<tr>
<td>Problem_Related_ID (FK)</td>
<td>Problem are related (Recursive fields)</td>
</tr>
<tr>
<td>Created_Date (VARCHAR)</td>
<td>Date of recording this problem</td>
</tr>
</tbody>
</table>
9.5 Sections
Each problem is broken down into many different sections which user can enter information and specify the level of security that who should be able to see its problem.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sections_ID (PK) (VARCHAR)</td>
<td>Primary key for each section</td>
</tr>
<tr>
<td>Problem_ID (FK) (VARCHAR)</td>
<td>This section belong to this problem particular problem ID</td>
</tr>
<tr>
<td>Description (VARCHAR)</td>
<td>Detail of the problem</td>
</tr>
</tbody>
</table>

9.6 Section Users Access
This table is specifying which users and which problem and which sections they are allow to access.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section_User_Access_ID (PK) (VARCHAR)</td>
<td>Primary key for each section_user_access table</td>
</tr>
<tr>
<td>Sections_ID (FK) (VARCHAR)</td>
<td>This section ID user can see</td>
</tr>
<tr>
<td>Problem_ID (FK) (VARCHAR)</td>
<td>For a particular problem</td>
</tr>
<tr>
<td>UserName (VARCHAR)</td>
<td>User who can access this problem, default is author and student who own its problem.</td>
</tr>
<tr>
<td>Roles (VARCHAR)</td>
<td>Roles of users such as personal tutor, director of studies and so on.</td>
</tr>
<tr>
<td>Description (VARCHAR)</td>
<td>Description for users or notes</td>
</tr>
<tr>
<td>Review From (VARCHAR)</td>
<td>This section can also be opened from</td>
</tr>
<tr>
<td>Review To (VARCHAR)</td>
<td>This section can also be expired until this date</td>
</tr>
</tbody>
</table>

9.7 Actions
Each section will have one or more actions associate with it therefore for a particular problem and particular section there will be one or more actions associated to it. An action can be either for Student or User who record the problem and other services
that may involve or participate for the action of the student. So they can be described as follows

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action_ID (PK) (VARCHAR)</td>
<td>Primary key for each action</td>
</tr>
<tr>
<td>Problem_ID (FK) (VARCHAR)</td>
<td>Reference key for problem that action belong</td>
</tr>
<tr>
<td>Section_ID (FK) (VARCHAR)</td>
<td>Reference key for section that hold action</td>
</tr>
<tr>
<td>Category_ID (FK) (VARCHAR)</td>
<td>The category that hold action belong</td>
</tr>
<tr>
<td>Title (VARCHAR)</td>
<td>Roles of users such as personal tutor, director of studies and so on.</td>
</tr>
<tr>
<td>Description (VARCHAR)</td>
<td>Description for action or notes</td>
</tr>
<tr>
<td>UserName (FK) (VARCHAR)</td>
<td>User who create this action</td>
</tr>
<tr>
<td>Related_To_Student (FK) (VARCHAR)</td>
<td>Student who carried out action</td>
</tr>
<tr>
<td>Action_For (VARCHAR)</td>
<td>Specify whether this action for student or Author</td>
</tr>
<tr>
<td>Created_Date (VARCHAR)</td>
<td>The date when this action create</td>
</tr>
<tr>
<td>Service_ID (FK) (VARCHAR)</td>
<td>Service id who will participant for this action.</td>
</tr>
</tbody>
</table>

9.8 Category
This table record the category detail own by users and students. Because users who record the problem are allowed to create their own category for both problem and actions therefore, we have to record them separately and specifically for each user.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category_ID (PK) (VARCHAR)</td>
<td>Primary key for each category</td>
</tr>
<tr>
<td>Title (VARCHAR)</td>
<td>Title of the category</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>Description (VARCHAR)</td>
<td>Description detail of each category</td>
</tr>
<tr>
<td>Category_For (VARCHAR)</td>
<td>Either Category for Problem or Action</td>
</tr>
<tr>
<td>UserName (FK) (VARCHAR)</td>
<td>Users who create category</td>
</tr>
<tr>
<td>Related_To_Student (FK) (VARCHAR)</td>
<td>Student who own this category</td>
</tr>
<tr>
<td>Created_Date (VARCHAR)</td>
<td>The date when the category is created</td>
</tr>
</tbody>
</table>

### 9.9 Users
This table will be used by Application Server to authenticate the access of the users.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username (PK) (VARCHAR)</td>
<td>Primary key for users</td>
</tr>
<tr>
<td>Password (VARCHAR)</td>
<td>password</td>
</tr>
</tbody>
</table>

### 9.10 Userroles
This table will be used by Application Server to authenticate the access of the users.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username (PK) (VARCHAR)</td>
<td>Primary key for users</td>
</tr>
<tr>
<td>UserRoles (VARCHAR)</td>
<td>Roles for each user</td>
</tr>
</tbody>
</table>

### 9.11 Services
This table record the service that will provide for the users

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Services_ID (PK) (VARCHAR)</td>
<td>Primary key for services</td>
</tr>
<tr>
<td>Title (VARCHAR)</td>
<td>Name of the services</td>
</tr>
<tr>
<td>Description (VARCHAR)</td>
<td>Description describe the services</td>
</tr>
<tr>
<td>Created_Date (VARCHAR)</td>
<td>Date of creating this services</td>
</tr>
</tbody>
</table>