6.0 Introduction

In this chapter, the implementation of the StuMOP tool will be discussed. This chapter starts by documenting the hardware and software requirements followed by database design, software architecture, process flow and testing. The software architecture section describes the coding of StuMOP.

6.1 Hardware and Software

Implementing the chosen design requires decision on important issues such as the type of technology, tools and languages used for creating the system and suitable hardware to support these. The five main issues resolved were:

i) Decision to use on open source technology was made as it is freely accessible and modifiable so that the application can be maintained easily and its lifespan maximized throughout operating system upgrades, and so on.

ii) Apache web server was decided to be used as a web server for the StuMOP tool; it is freely available and is arguably the most popular web server in use on the Internet today. It has been ported to most operating systems, and is highly stable. As a result of its popularity the bugs in each release are found very quickly and immediately resolved. It is fast and reliable. Thomas (2007) states that Apache web server, which hosts more than 70% of the websites worldwide, fully support the JSP technology under the Jakarta Tomcat project. Tomcat is a servlet container with a JSP environment. A servlet container is a runtime shell that manages and invokes servlets. Tomcat can be used as either a stand-alone container (mainly for development and debugging) or as an out-of-process add-on to an existing
web server (Apache). JSP technology follows the model of separating programming logic from page design through the use of components that are called from the page itself and provide developers an easier and faster alternative to creating web applications using CGI scripts. JSP is being developed through an industry-wide initiative led by Sun Microsystems and is designed to be both platform and server independent.

iii) MySql was decided to be used as a database server. It is freely available high speed database driver that is highly popular, secure and fast. It communicates easily with JAVA. MySql-Front is a graphical user interface to MySql that helps to ease the management the database.

iv) JavaServer Pages (JSP) technology was decided to be used as it is well on its way to becoming the most excellent Java technology for building applications that serve dynamic web content. Seshadri (1999) states that JSP brings the "write once, run anywhere" paradigm to interactive Web pages; and it allows the use Java as a server-side scripting language, which provides inherent scalability and support for complex operations. The biggest advantage of using JSP is that it helps effectively separate presentation from content. It is compatible with the server chosen, and capable of interacting with the database server chosen.

v) This StuMOP tool will be designed to run on PC’s (mouse, monitor, etc) that are connected to the Intranet using the TCP/IP protocols. The researcher did not use any additional device. The minimum requirements to best view this web-based tutorial would be:

- 256 MB RAM
- 800 x 600 pixels, 15” monitor
- Internet Explorer 6.0
6.2 Database

The relational database was designed to ensure that all possible requirements could be satisfied. All the tables are in third normal form; represents the “data sources” in JSP model 1 architecture shown in Figure 6.1. Table structures are listed and explained below are free from insertion, updation and deletion anomalies:

Table 6.1: Admindetail

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adminid (PK)</td>
<td>varchar(10)</td>
</tr>
<tr>
<td>adminpw</td>
<td>varchar(10)</td>
</tr>
<tr>
<td>firstName</td>
<td>varchar(25)</td>
</tr>
<tr>
<td>lastName</td>
<td>varchar(25)</td>
</tr>
<tr>
<td>Dob</td>
<td>date</td>
</tr>
<tr>
<td>Address</td>
<td>Text</td>
</tr>
<tr>
<td>Country</td>
<td>varchar(15)</td>
</tr>
<tr>
<td>Email</td>
<td>varchar(20)</td>
</tr>
</tbody>
</table>

The above table stores the detail of lecturers. A unique adminid is used as the primary key.

Table 6.2: userdetail

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>userid (PK)</td>
<td>char(20)</td>
</tr>
<tr>
<td>Userpw</td>
<td>varchar(10)</td>
</tr>
<tr>
<td>firstName</td>
<td>varchar(25)</td>
</tr>
<tr>
<td>lastName</td>
<td>varchar(25)</td>
</tr>
<tr>
<td>classid</td>
<td>Varchar(15)</td>
</tr>
<tr>
<td>dob</td>
<td>date</td>
</tr>
<tr>
<td>address</td>
<td>Text</td>
</tr>
</tbody>
</table>
userdetail table stores the detail of students. A unique userid is used as the primary key.
Password is encoded and stored in password field userpw.

Table 6.3: classdetail

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>adminid</td>
<td>Varchar(15)</td>
</tr>
<tr>
<td>classid (PK)</td>
<td>Varchar(15)</td>
</tr>
<tr>
<td>year</td>
<td>Int(4)</td>
</tr>
<tr>
<td>status</td>
<td>Varchar(15)</td>
</tr>
</tbody>
</table>

Classdetail table stores the class detail of all lecturers. A lecturer can have many classes, but a class can have only one lecturer. Lecturers can inactivate a class upon its completion. Status field flags an active or inactive class.

Table 6.4: lecturerNameChapter1

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>questionid (PK)</td>
<td>int(10)unsigned</td>
</tr>
<tr>
<td>subject</td>
<td>varchar(10)</td>
</tr>
<tr>
<td>question</td>
<td>Text</td>
</tr>
<tr>
<td>answer1</td>
<td>Text</td>
</tr>
<tr>
<td>answer2</td>
<td>Text</td>
</tr>
<tr>
<td>answer3</td>
<td>Text</td>
</tr>
<tr>
<td>answer4</td>
<td>Text</td>
</tr>
<tr>
<td>correctAnswer</td>
<td>Text</td>
</tr>
<tr>
<td>adminid</td>
<td>Varchar(15)</td>
</tr>
<tr>
<td>chapterNo (FK)</td>
<td>int(2)</td>
</tr>
</tbody>
</table>
This table is automatically created when a lecturer creates a quiz for new concept; it stores the questions for that concept. A table similar to this is created for each concept. The same table structures are used for multiple choices and fill in the blank questions. Where no data are stored in attributes answer2, answer3 and answer4. The foreign key chapterNo relates the concept to the question.

### Table 6.5: subject

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>char(20)</td>
</tr>
<tr>
<td>topic</td>
<td>int(10)unsigned</td>
</tr>
<tr>
<td>chapterNo(PK)</td>
<td>int(2)unsigned</td>
</tr>
<tr>
<td>description</td>
<td>int(3)unsigned</td>
</tr>
</tbody>
</table>

The subject table stores the concept list, where chapterNo represents unique identifier for each concept. Attribute topic stores the concept name such as loops, pointers.

### Table 6.6: userlogin

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>userid</td>
<td>char(20)</td>
</tr>
<tr>
<td>examid (PK)</td>
<td>int(10)unsigned</td>
</tr>
<tr>
<td>Status</td>
<td>Varchar(10)</td>
</tr>
<tr>
<td>dateoflogin</td>
<td>Date</td>
</tr>
</tbody>
</table>

The userlogin table is designed to store data automatically each time a student takes a tutorial. A unique examid is generated each time the student takes a tutorial.

### Table 6.7: userexamdetail

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>userid (PK)</td>
<td>char(20)</td>
</tr>
<tr>
<td>examid(PK)</td>
<td>int(10)unsigned</td>
</tr>
<tr>
<td>chapterNo</td>
<td>int(2)unsigned</td>
</tr>
<tr>
<td>Questioned</td>
<td>int(10)unsigned</td>
</tr>
<tr>
<td>Mark</td>
<td>int(3)unsigned</td>
</tr>
</tbody>
</table>
Chapter 6: Implementation of StuMOP

The userexamdetail table has a composite key comprised of userid and examid. The attribute “chapterNo” refers from subject table and questionid refers from chapter1questions. Attribute mark stores one for correct answer and zero for wrong answer.

### Table 6.8: result

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>userid (PK)</td>
<td>char(20)</td>
</tr>
<tr>
<td>examid(PK)</td>
<td>int(10)unsigned</td>
</tr>
<tr>
<td>chapterNo</td>
<td>int(2)unsigned</td>
</tr>
<tr>
<td>totalMark</td>
<td>int(3)unsigned</td>
</tr>
<tr>
<td>dateofexam</td>
<td>Date</td>
</tr>
<tr>
<td>Classid</td>
<td>Varchar(15)</td>
</tr>
</tbody>
</table>

The result table is used to store the total mark of students in each concept.

### Table 6.9: user_faildetails

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>userid (PK)</td>
<td>char(20)</td>
</tr>
<tr>
<td>examid(PK)</td>
<td>int(10)unsigned</td>
</tr>
<tr>
<td>chapterNo</td>
<td>int(2)unsigned</td>
</tr>
<tr>
<td>totalMark</td>
<td>int(3)unsigned</td>
</tr>
<tr>
<td>dateofexam</td>
<td>Date</td>
</tr>
<tr>
<td>Classid</td>
<td>Varchar(15)</td>
</tr>
</tbody>
</table>

The user_faildetails table has the structure similar to result table. It is used to store the total mark in each concept by students who have yet to pass in the concept. Once the student passes in the concept the record is deleted through coding. This table is used as an intermediate table to ease the programming logic.
6.3 Software Architecture

The architecture most suitable for the StuMOP tool is the Model-View-Controller (MVC) architecture. This architecture is widely used with Object-Oriented development environments. The aim of MVC architecture is to separate the business logic and data of the application from the presentation of data to the user. The description of each of the components in MVC architecture is as follows:

6.3.1 Model

The model represents the data of an application. Anything that an application will persist becomes a part of model. The model also defines the way of accessing this data (the business logic of application) for manipulation. It knows nothing about the way the data will be displayed by the application. It just provides service to access the data and modify it.

6.3.2 View

The view represents the presentation of the application. The view queries the model for its content and renders it. The way the model will be rendered is defined by the view. The view is not dependent on data or application logic changes and remains same even if the business logic undergoes modification. The usability principles (Nielsen, 2001) are applied in the design of StuMOP tool. It ensures that StuMOP tool provide clear information in the users language and it provides flexibility and efficiency of use.
6.3.3 Controller

All the user requests to the application go through the controller. The controller intercepts the requests from view and passes it to the model for appropriate action. Based on the result of the action on data, the controller directs the user to the subsequent view.

6.4 JSP Model 1 Architecture

The StuMOP tool is built based on the MVC architecture using JSP model 1 architecture as shown in Figure 6.1. Here the JSP page alone is responsible for processing the incoming request and replying back to the client. There is still separation of presentation from content, because all data access is performed using beans. The building blocks are JSP-pages and JavaBeans in this MVC-architecture. Servlets are not used.

Figure 6.1: JSP model 1 architecture (Seshadri, 2007)

Based on JSP model 1 architecture the researcher implemented the MVC-architecture by coding a controller that handles common tasks such as authentication, examines the requests from the user, invokes JavaBeans, controls error handling, controls module
Chapter 6: Implementation of StuMOP

flow. Model coding a set of JavaBeans to handle the database access (or access to other external resources). Coding a View for presentation of the data fetched by the beans

The controller is implemented in JSP-format, which is also commonly used for the View. Model is implemented as Java Bean. The StuMOP tool is therefore simplified by only containing JSP-pages and JavaBeans. In order to clarify the concepts behind the Model 1 architecture in the StuMOP tool, let’s walk through a detailed implementation of it:

6.4.1 View and Controller in the StuMOP Tool:

The main view, or presentation, for our StuMOP tool is facilitated by the JSP page login.jsp as shown in Figure 6.2. The page deals almost exclusively with presenting the main user interface of the application to the student as shown in Figure 6.3 and perform no processing. Also, notice that another JSP page, Elearn.jsp is included within login.jsp via the `<form method="post" action="Elearn.jsp" name=logfrm>`

```html
<!-- login.jsp (view) -->

<form method="post" action="Elearn.jsp" name=logfrm>
  <TABLE class=table_standard cellSpacing=0 cellPadding=5 rules=none width=480 align=center border=2 frame=box>
    <TBODY><TR><TH class=title style="LETTER-SPACING: 4px" colSpan=2>Login For Student</TH></TR><TR><TD>Please enter a username:</TD><TD><INPUT name=userid> </TD></TR><TR><TD>Please enter a password:</TD><TD><INPUT type=password name=userpw> </TD></TR>
    <TR> <TD align=center colSpan=2><INPUT type=submit value="Student Login" name=login>
    </TD></TR><TR><td align=center colSpan=2><a href='../usermanagement/registration.jsp'>New Student</a></td>
    </TR></TABLE></form>

<!-- login.jsp (view) -->
```

Figure 6.2: login.jsp (view)
In turn Elearn.jsp calls a method checkLogin() to authenticate the users. This shows that the JSP program acts both as the view and controller of the MVC architecture. When the authentication is done the Elearn.jsp presents a screen (view) to the student as shown in Figure 6.5. When the student requests the view profile, the Elearn.jsp calls <a href="../exammanagement/ViewProfile.jsp?userid=<%=userid%>">View Profile</a>. The part of coding in ViewProfile.jsp is shown in Figure 6.4, has call to methods viewResultForParticularUser(), vewChartForParticularUser(), viewBestofTutorial() in the program ResultDB.java, where the methods performs the function of the controller. The screen when ViewProfile.jsp is clicked is shown in Figure 6.6. The ResultDB.java has definitions to the above methods.
try
{ maxChapter=admin.SubjectDB.viewChapterNo(); }
catch(Exception ex) {}

try
{ vec=(Vector)exammanagement.ResultDB.viewResultForParticularUser(userid);

try
{ chart=(Vector)exammanagement.ResultDB.viewChartForParticularUser(userid);

catch(Exception ex) {}

try
{ sub=(Vector)admin.SubjectDB.viewSubject(); }

catch(Exception ex) {}

try
{ resultvec=(Vector)exammanagement.ResultDB.viewBestofTutorial(); }

catch(Exception ex) {}

.............

..........
6.4.2 Model in StuMOP Tool

The JavaBean represents the model in the MVC architecture. The model codes the JavaBean to access the database through the SQL statements. The listing in Figure 6.7 shows the definition of viewNameWiseResult() method in the resultDB.java where access to database is made through the SQL "select distinct (userid) from result;". This query retrieves the unique userid from the result table. Where the result entity codes the model that represents the result data of the StuMOP tool Part of the listing is shown in Figure 6.8 represents the attributes of the result table.
public class ResultEntity
{
    private String userid="";
    private int examid;
    private int chapterNo;
    private int totalMark;
    private java.sql.Date dateofexam=null;
    private int countUserid;
    private int totalStrength;

    public ResultEntity() {} 

    public void setUserid(String userid)
    {this.userid=userid;  }

    public String getUserid()
    {return userid;  }

    public void setExamid(int examid)
    {this.examid=examid;  }

    public int getExamid()
    {return examid;  }

    public void setChapterNo(int chapterNo)
    {this.chapterNo=chapterNo;  }

    public int getChapterNo()
    {return chapterNo;  }

    public void setTotalMark(int totalMark)
    {this.totalMark=totalMark;  }

    public int getTotalMark()
    {return totalMark;  }

    public void setDateofexam(java.sql.Date dateofexam)
    {this.dateofexam=dateofexam;  }

    public java.sql.Date getDateofexam()
    {return dateofexam;  }

}

Figure 6.7 Result Entity.java (model)
```java
public static Vector viewNameWiseResult() throws LanException {
    ......
    ResultEntity re=new ResultEntity();
    LanException lex=new LanException();
    try
    {con=Connect.getConnection();  }
    catch(Exception ex)
    {
        flag=false;
        errorstring="problems with connection";
    }
    if(con!=null)
    {
        try
        {
            PreparedStatement pstm=con.prepareStatement("select distinct(userid) from result");
            rs=pstm.executeQuery();
            if(!rs.next())
            {
                flag=false;
                errorstring="data not found";
            }
            else
            {
                do
                {
                    re=new ResultEntity();
                    re.setUserid(rs.getString(1));
                    vec.addElement(re);
                }while(rs.next());
            }
        }
        catch(Exception ex)
        {
            flag=false;
            errorstring=""+ex;
        }
    }
    else
    {
        flag=false;
        errorstring="problems with connection";
    }
    if(flag==false)
    {
        lex.setError(errorstring);
        throw lex;
    }
    return vec;
}
```

Figure 6.8  viewNameWiseResult() in resultDB.java (Model)
6.5 Process Flow

Basically, the StuMOP tool consists of two users, lecturers and students. The Figure 6.9 shows the flow between different processes in the tutorial management that help the lecturer to manage the tutorial and monitor the student. Monitoring the performance of individual and class is facilitated in a various reports using the result, class and student data.

![Diagram of Lecturer Functional Modeling](image)

**Figure 6.9: Lecturer Functional Modeling**

The Figure 6.10 shows the flow between different processes in this tool that helps the student to carry out the learning process and to monitor their learning progress. The JSP files that implement the process are shown in the Figure 6.10.
6.6 Testing

Testing has been performed on the StuMOP tool, after its implementation. Each script was tested during the development phase. To achieve this many test cases were return, based on simple input and output before writing a script. The test cases for each script covered all the functionality that the script is expected to have. A script was only considered to be finished when it had passed all of the test cases. This method of testing during development, in isolation of other layers is known as Unit Testing. Unit testing, however, is not sufficient to ensure a system performs satisfactorily. The application and integration (the communication between scripts) of the units still must be tested. Proper paths through system components such as the login system, the tutorial system

![Diagram of Student Learning Modeling](image-url)
(question and answer) and tutorial management had been rigorously tested to find errors that would otherwise be undetectable when testing individual pieces of code.

6.6.1 System Overview

The StuMOP tool itself can be divided into sections for testing. Testing these individual components in order will effectively test a full path through the system. Each of these components had to be tested exhaustively to ensure that the StuMOP tool will perform correctly and would meet its requirements when deployed in an academic environment.

6.6.2 Test Plan

In testing this StuMOP tool the goal was to find all feasible paths through each system component and establish that each path results in correct output from the system. This involved firstly finding these paths and secondly generalizing so that a feasible number of test cases are remaining. This involves a combination of both boundary testing and testing failure conditions. It is not feasible to test every possible username and password combination for a login, so the tests are generalized into groups, for example, successful login, unsuccessful login etc. The testing cases were restricted to the requirements of the system; it was also ensured that the addition of any extra features does not impact on the required features. The test method used for each component is defined by the method of interaction with that component. The test cases for the login system cover text input whereas the question and answer system were tested by typical browser interaction (i.e. mouse input).

If a data entered by the user is illegal or affects the further processing of those transactions, it results in error condition. Whenever a user enters such invalid data against any field on the screen, the StuMOP tool has to flash appropriate error message on the screen where by the user understands the nature of mistake and correct the entry
suitable. In case of error StuMOP tool should not allow user to proceed with entry of other details without correction of the error.

6.6.3 Test Cases

This section will discuss the test cases for the StuMOP tool. The test cases will be mapped with the requirements derived in section 4.5. This ensures that the StuMOP tool correctly and meets the requirements.

6.6.3.1 Registration in the web-based tool [R1]

The requirement [R1] demands the interactive web-based tool to be controlled by a password system. Hence the user needs to register them.

A valid registration

- **Purpose**

  The purpose of this test is to verify that when a user enters a valid user id the StuMOP tool will register them successfully.

- **Input**

  The input is a variety of personal information such as valid username, password, Email, secret question, secret answer and class code. In this case a name for example “amir” and corresponding details.

- **Expected Output**

  The expected output for this case is that the user is registered successfully.

An invalid registration

- **Purpose**
Chapter 6: Implementation of StuMOP

The purpose of this test is to verify that the StuMOP tool will not register users with redundant username and with an empty class code.

- **Input**

  The input is a variety of personal information such as with redundant username and no data for class code. It should be noted that the username, password, secret question, secret answer, class code and email fields are not passed on as blank fields; in order to ensure this client-side JavaScript validation is done. A student cannot register until his/her lecturer adds a class for the students.

- **Expected Output**

  The expected output for this case is that on all occasions the user is asked to register again, prompting them of the redundant username as shown in figure 6.11 and empty class code as shown in Figure 6.12.

![Image: Student Registration Form](image)

**Figure 6.11: Invalid Student Registration due to redundant username**
6.6.3.1.1 Controlled access to the web-based tool [R1]

The requirement [R1] demands the interactive web-based tool to be controlled by a password system. Lecturers and student need to use their userids and passwords to login.

A valid login

- **Purpose**

  The purpose of this test is to verify that when a user enters a valid user id and password the StuMOP tool will log them in correctly.

- **Input**

  The input is a valid username, in this case a name for example “azhani” and a corresponding password (i.e. an alphanumeric string).

- **Expected Output**

  The expected output for this case is that the user is redirected to the welcome screen of the tutorial.
An Invalid Login

- **Purpose**

The purpose of this test is to verify that the StuMOP tool will not login users who present an invalid username and/or password.

- **Input**

There are four separate inputs for this test case. Firstly a valid username, with an invalid password, followed by an invalid username with a valid password, thirdly an invalid username and an invalid password and finally blank fields for both username and password. It should be noted that the neither of the two fields are not passed on as blank fields; in order to ensure this client-side JavaScript validation is done that alerts with an error message as shown in Figure 6.14.

- **Expected Output**

The expected output for this case is that on all three occasions the user is asked to login again, reminding them that passwords and user names are invalid as shown in Figure 6.13.
6.6.3.2 Lecturer create concept list [R2]

In order to ensure a guided learning process a concept list with a prerequisite hierarchy need to be created.

- **Purpose**

  The purpose of this test is to verify that the StuMOP tool enables the lecturer to create a concept list.

- **Input**

  The input to this test is a list of concepts based on the prerequisite hierarchy.

- **Expected Output**

  The expected output is that the concept is stored in the database and the lecturer is possible to add questions to each concept in the list.
6.6.3.3 Lecturer monitors progress of student [R3]

In order to ensure learning process; the lecturer needs to monitor the learning.

- **Purpose**
  
The purpose of this test is to verify that the StuMOP tool enables the lecturer to monitor learning progress of individual student.

- **Input**
  
The input to this test is the class code and the student code.

- **Expected Output**
  
The expected output is the lecturer able view the performance of student stored in the database. The lecturer is possible to view the number of attempts, the result in each attempt, date the attempt is made.

6.6.3.4 Student monitor their progress [R4]

The student will be able to test herself in the web-based tool and will receive feedback as to progress.

- **Purpose**
  
The purpose of this test is to verify that the StuMOP tool will not show the profile of students who have not attempted at least a single tutorial.

- **Input**
  
The input is a valid username and a corresponding password.

- **Expected Output**
  
The expected output for this case is that an error message will be displayed as shown in Figure 6.15.
6.6.3.5 Lecturer monitor performance in each concept [R5]

In order to enhance teaching process; the lecturer need to monitor the learning.

- **Purpose**

  The purpose of this test is to verify that the StuMOP tool enables the lecturer to trace the concepts that are difficult for the students in their learning progress.

- **Input**

  The input to this test is the class code.

- **Expected Output**

  The expected output is the lecturer able view the overall performance of the class in each concept. The lecturer is possible to view the understanding in percentage.
6.6.3.6 Student centered learning [R6]

The StuMOP tool will select the quiz that the student needs to complete next will be based on the progress the student has made to date. Once the student level has been identified, a quiz on the appropriate level of understanding is given to the student.

- **Purpose**
  
The purpose of this test is to check, whether the StuMOP tool will advance a student to the next concept in the concept list when the student scores minimum eight correct answers on a given level.

- **Input**
  
The input in this case is a student id and a minimum of eight correct answers to twenty questions in a session on a given level.

- **Expected Output**
  
The expected output here is that the student is informed that they are qualified at the current level and presented with another twenty questions at the next level.

**Storing a student’s results**

- **Purpose**
  
The purpose of this test is to verify that the StuMOP tool stores its results correctly and securely when a student finishes their test.

- **Input**
  
The input to this test is a student answering their twenty questions.

- **Expected Output**
The expected output is that the score is stored in the marks database and the student is brought to the welcome page of the tutorial and informed of their overall score in that concept when the student click the “View Profile” link.

6.6.3.7 The student practice concepts [R7]

The student will be able to practice the concepts until sufficient knowledge is attained in the concept in his/her own time until the expiry of the class session.

Class Duration Expiry

• Purpose

The purpose of this test is to verify that this StuMOP tool blocks when a student attempts the tutorial upon the expiry of his/her class.

• Input

The input is a valid username and a corresponding password.

• Expected Output

The expected output for this case is that the user is redirected to an error page as shown in Figure 6.16.

![Figure 6.16: Validation of Class Session Expiry](image)

Figure 6.16: Validation of Class Session Expiry
6.6.3.7.1 The student practice concepts [R7]

The student will be able to practice the concepts until sufficient knowledge is attained in the concept. Hence the student needs to answer the question.

Answering a question

• **Purpose**

The purpose of this test is to verify that a student selects an answer to a question before proceeding to the next question; the StuMOP tool acknowledges it and awards mark appropriately.

• **Input**

The input to this test is an answer, by choosing an answer from multiple choices (a click on the radio button) or filling the blanks.

• **Expected Output**

The expected output for this case is that upon submitting the answer the student is given the next question.

Not answering a question

• **Purpose**

The purpose of this test is to verify that StuMOP tool will not assign the subsequent questions in the tutorial to a student who skips the current question without answering.

• **Input**

The input to this test is submitting the question without choosing the answer from multiple choices or without filling in the blanks.

• **Expected Output**

The expected output is an alert box is prompted with message reminding to select any one answer or to fill in the blanks as shown in Figure 6.17.
Chapter 6: Implementation of StuMOP

Re-entering a tutorial Session

- **Purpose**
  
The purpose of this test is to verify that the StuMOP tool will not assign tutorial on the subsequent concept until the student qualifies in the current concept.

- **Input**
  
The input to this is the tool is valid username, corresponding password and total mark on the latest tutorial and the click on the “Enter Tutorial” button.

- **Expected Output**
  
The expected output is an error message that alerts the student on the poor result of his/her latest tutorial and prompting them to retake the tutorial as shown in Figure 6.18.
Usage of Back Button during the tutorial

- **Purpose**

The purpose of this test is to ensure that students cannot re-answer questions using the back button in the browser. The back button returns a student to their previous page(s), and could therefore enable them to re-answer old questions.

- **Input**

This case tests a session where the student has answered the previous question incorrectly and has returned to the same question to re-answer it. The inputs are therefore a valid username a corresponding password, a question id and an answer id.

- **Expected Output**

The expected output for this case is that the StuMOP tool will disable the function of back button.
6.6.3.8 Performance report in graphical format [R8]

In order to support in the monitoring process; the performance report need to be in graphical format.

- **Purpose**

The purpose of this test is to verify that the StuMOP tool able to generate bar charts to represent the performance of the students.

- **Input**

The input to this test is the class code, student code and the best mark in each concept.

- **Expected Output**

The expected output is the bar chart that represents the performance in each concept. The graph is generated with color coding to represent various performance types Green represents excellent. Orange and red represents good and poor performance respectively.

6.6.3.9 Student view the best performance of the class in each concept [R9]

In order to enhance motivation of the students in their learning process the best performance of the class need to be provided.

- **Purpose**

The purpose of this test is to verify that the StuMOP tool generates the best performance of the class in each concept.

- **Input**

The input to this test is the class code, and the mark of the class in each concept.

- **Expected Output**
The expected output is the bar chart that represents the best performance in each concept of the class.

6.7 Summary

This chapter described the implementation of the StuMOP tool. It also documented the hardware and software requirements followed by database design, software architecture, process flow and testing. The test cases traced that the requirements were implemented.