SOFTWARE DESIGN DOCUMENT

**(SDD)**

**For**

**Tuition Management System (TMS)**

**Prepared for:**

**SPACE EDGE INC.**

**Generated By:**

|  |  |
| --- | --- |
| **NAME** | **MATRIC ID** |
| PUANG SING CHUAN | CB14049 |
| MOHD AMIRULHANIF BIN ABDUL RAZAK | CB14015 |
| NORMALIANA BINTI CHE ZAHARI | CB14059 |
| NABILA BINTI ZAINAL | CB14036 |
| FATIN SYAHIRA BINTI RAHMAT | CB14038 |

**MARCH 2016**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Approval List Table** | | | | | | |
| Index | 01 | 02 | 03 | 04 | 05 | 06 |
| Written by:  Name:  Fatin Syahira binti Rahmat | Date: | Date: | Date: | Date: | Date: | Date: |
| Verified by:  Quality Manager  Name:  Nabila binti Zainal | Date: | Date: | Date: | Date: | Date: | Date: |
| Check by:  Configuration Manager  Name:  Mohd Amirulhanif bin Abdul Razak | Date | Date: | Date: | Date: | Date: | Date: |
| Approved by:  Work Package Manager  Name:  Normaliana binti Che Zahari | Date | Date: | Date: | Date: | Date: | Date: |
| Authenticated by:  Project Manager  Name:  Puang Sing Chuan | Date | Date: | Date: | Date: | Date: | Date: |

|  |  |
| --- | --- |
| **Revision History** | |
| **Revision** | **Description** |
| 01 |  |
| 02 |  |
| 03 |  |
| 04 |  |
| 05 |  |
| 06 |  |

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **1. SCOPE**...............................................................................................................  1.1. Identification ..................................................................................................  1.2. Overview of the System ….............................................................................  1.3. Overview of the Document …........................................................................ | 1  1  1  3 |
| **2. REFERENCED DOCUMENTS**...................................................................... 2.1. Overview of the Documents ........................................................................  2.2 Reference Documents .................................................................................. | 4  4  4 |
| **3. PRELIMINARY DESIGN**...............................................................................  3.1. System Overview ............................................................................................  3.1.1 System Architecture ...................................................................................  3.1.2 System States and Modes...........................................................................  3.2. CSCI Design Description ................................................................................  3.2.1. Registration Subsystem.............................................................................  3.2.2. Payment Subsystem...................................................................................  3.2.3. Attendance Subsystem..............................................................................  3.2.4. Examine Subsystem..................................................................................  3.2.5. Teaching Subsystem……………………………………………………. | 5  5  6  12  15  15  16  17  18  19 |
| **4. DETAILED DESIGN**.......................................................................................  4.1 Registration Subsystem............................................................................  4.1.1 Registration Package......................................................................  4.2 Payment Subsystem.................................................................................  4.2.1 Payment Package..............................................................................  4.3 Attendance Subsystem................................................................................  4.3.1 Attendance Package.........................................................................  4.4 Examine Subsystem……..........................................................................  4.4.1 Examine Package…......................................................................  4.5 Teaching Subsystem……………………………………………………  4.5.1 Teaching Package….................................................................... | 20  20  20  39  39  63  63  82  82  96  96 |
| **5. DATA DICTIONARY……………………………………………………..** | 119 |
| **6. REQUIREMENTS TRACEABILITY………………………………………**  **7. NOTES**............................................................................................................... | 122  123 |

**List of Figures**

Figure 3.1: TMS Design Overview.......................................................................5

Figure 3.2: Static Organization of TMS................................................................6

Figure 3.3: Dynamic Organization of TMS........... ..............................................10

Figure 3.4: Package/ Subsystem Interfaces...........................................................11

Figure 3.5: State Diagram for Registration ..........................................................12

Figure 3.6: State Diagram for Fees............................... .......................................12

Figure 3.7: State Diagram for Wage............................ ........................................13

Figure 3.8: State Diagram for Examine....................... ........................................13

Figure 3.9: State Diagram for Subject Registration .... ........................................14

Figure 3.10: State Diagram for Upload Document.… .........................................14

Figure 3.11: Visibility of Registration................... ........................ .....................15

Figure 3.12: Visibility of Payment.......................................................................16

Figure 3.13: Visibility of Attendance...................................................................17

Figure 3.14: Visibility of Examine.......................................................................18

Figure 3.15: Visibility of Teaching………...........................................................19

Figure 4.1: Registration Detail Design.................................................................20

Figure 4.2: Payment Detail Design......................................................................39

Figure 4.3: Attendance Detail Design..................................................................63

Figure 4.4: Evaluation Detail Design...................................................................82

Figure 4.5: Teaching Detail Design ....................................................................96

**List of Tables**

Table 4.1.1.1.1: Local Data Definition for data elements ………......…………21

Table 4.1.1.1.2: Local Data Definition for data elements………...........………21

Table 4.1.1.1.3: Local Data Definition for data elements……...................……21

Table 4.1.1.1.4: Local Data Definition for data elements……………...............22

Table 4.1.1.1.5: Local Data Definition for data elements…………….......……22

Table 4.1.1.1.6: Local Data Definition for data elements…………...................22

Table 4.1.1.1.7: Local Data Definition for data elements…...............................22

Table 4.1.1.2.1: Local Data Definition for data elements………….....………..25

Table 4.1.1.2.2: Local Data Definition for data elements…………..............….25

Table 4.1.1.2.3: Local Data Definition for data elements…...............................25

Table 4.1.1.2.4: Local Data Definition for data elements…………...................25

Table 4.1.1.2.5: Local Data Definition for data elements ……..........................25

Table 4.1.1.2.6: Local Data Definition for data elements………...........…........25

Table 4.1.1.2.7: Local Data Definition for data elements ……….....………….25

Table 4.1.1.2.8: Local Data Definition for data elements……….....……..........26

Table 4.1.1.2.9: Local Data Definition for data elements…….....................…..26

Table 4.1.1.2.10: Local Data Definition for data elements….....………………26

Table 4.1.1.3.1: Local Data Definition for data elements…….................……..29

Table 4.1.1.3.2: Local Data Definition for data elements ……….....………….29

Table 4.1.1.3.3: Local Data Definition for data elements…………….…..........29

Table 4.1.1.4.1: Local Data Definition for data elements……………........……31

Table 4.1.1.4.2: Local Data Definition for data elements………………………31

Table 4.1.1.5.1: Local Data Definition for data elements………………..……..32

Table 4.2.1.1.1: Local Data Definition for data elements………………......…..40

Table 4.2.1.1.2: Local Data Definition for data elements………………...……40

Table 4.2.1.2.1: Local Data Definition for data elements ………………..…….43

Table 4.2.1.2.2: Local Data Definition for data elements………………....……43

Table 4.2.1.2.3: Local Data Definition for data elements………………....……43

Table 4.2.1.2.4: Local Data Definition for data elements………………....……43

Table 4.2.1.2.5: Local Data Definition for data elements……………....………44

Table 4.2.1.2.6: Local Data Definition for data elements………………………44

Table 4.2.1.3.1: Local Data Definition for data elements………………………45

Table 4.2.1.3.2: Local Data Definition for data elements………………………46

Table 4.2.1.3.3: Local Data Definition for data elements………………………46

Table 4.2.1.3.4: Local Data Definition for data elements………………………46

Table 4.2.1.3.5: Local Data Definition for data elements………………………46

Table 4.2.1.3.6: Local Data Definition for data elements………………………46

Table 4.2.2.1.1: Local Data Definition for data elements………………………48

Table 4.2.2.1.2: Local Data Definition for data elements………………………48

Table 4.2.2.2.1: Local Data Definition for data elements………………………50

Table 4.2.2.2.2: Local Data Definition for data elements………………………50

Table 4.2.2.2.3: Local Data Definition for data elements………………………50

Table 4.2.2.2.4: Local Data Definition for data elements………………………51

Table 4.2.2.2.5: Local Data Definition for data elements………………………51

Table 4.2.2.3.1: Local Data Definition for data elements………………………52

Table 4.2.2.3.2: Local Data Definition for data elements………………………52

Table 4.2.2.3.3: Local Data Definition for data elements………………………53

Table 4.2.2.3.4: Local Data Definition for data elements………………………53

Table 4.2.2.3.5: Local Data Definition for data elements………………………53

Table 4.2.3.1.1: Local Data Definition for data elements………………………54

Table 4.2.3.1.2: Local Data Definition for data elements………………………55

Table 4.2.3.2.1: Local Data Definition for data elements………………………57

Table 4.2.3.2.2: Local Data Definition for data elements………………………57

Table 4.2.3.2.3: Local Data Definition for data elements………………………57

Table 4.2.3.2.4: Local Data Definition for data elements………………………57

Table 4.2.3.2.5: Local Data Definition for data elements………………………57

Table 4.2.3.2.6: Local Data Definition for data elements………………………58

Table 4.2.3.2.7: Local Data Definition for data elements………………………58

Table 4.2.3.3.1: Local Data Definition for data elements………………………60

Table 4.2.3.3.2: Local Data Definition for data elements………………………60

Table 4.2.3.3.3: Local Data Definition for data elements………………………60

Table 4.2.3.3.4: Local Data Definition for data elements………………………60

Table 4.2.3.3.5: Local Data Definition for data elements………………………60

Table 4.2.3.3.6: Local Data Definition for data elements………………………61

Table 4.2.3.3.7: Local Data Definition for data elements………………………61

Table 4.3.2.1: Local Data Definition for data elements……………………...…66

Table 4.3.2.2: Local Data Definition for data elements……………………...…66

Table 4.3.2.3: Local Data Definition for data elements……………………...…66

Table 4.3.2.4: Local Data Definition for data elements……………………...…66

Table 4.3.3.1: Local Data Definition for data elements……………………...…69

Table 4.3.3.2: Local Data Definition for data elements……………………...…69

Table 4.3.3.3: Local Data Definition for data elements……………………...…69

Table 4.3.3.4: Local Data Definition for data elements……………………...…69

Table 4.3.3.5: Local Data Definition for data elements……………………...…70

Table 4.3.4.1: Local Data Definition for data elements…………………...……73

Table 4.3.6.1: Local Data Definition for data elements……………………...…76

Table 4.3.8.9: Local Data Definition for data elements……………………...…79

Table 4.4.1.1: Local Data Definition for data elements……………………...…83

Table 4.4.1.2: Local Data Definition for data elements……………………...…83

Table 4.4.1.3: Local Data Definition for data elements……………………...…83

Table 4.4.1.4: Local Data Definition for data elements…………………...……83

Table 4.4.1.5: Local Data Definition for data elements…………………...……84

Table 4.4.1.6: Local Data Definition for data elements……………………...…84

Table 4.4.2.1.1: Local Data Definition for data elements………………………86

Table 4.4.2.1.2: Local Data Definition for data elements………………………86

Table 4.4.2.1.3: Local Data Definition for data elements………………………86

Table 4.4.2.1.4: Local Data Definition for data elements………………………87

Table 4.4.2.1.5: Local Data Definition for data elements………………………87

Table 4.4.2.1.6: Local Data Definition for data elements………………………87

Table 4.4.3.1: Local Data Definition for data elements………………………...88

Table 4.4.3.2: Local Data Definition for data elements………………………...88

Table 4.4.3.3: Local Data Definition for data elements…………………...……88

Table 4.4.4.1: Local Data Definition for data elements………………………90

Table 4.4.5.1: Local Data Definition for data elements………………………91

Table 4.4.6.1: Local Data Definition for data elements………………………92

Table 4.4.7.1: Local Data Definition for data elements………………………93

Table 4.4.7.2: Local Data Definition for data elements………………………94

Table 4.4.8.1: Local Data Definition for data elements………………………95

Table 4.5.1.1.1: Local Data Definition for data elements………………….…97

Table 4.5.1.1.2: Local Data Definition for data elements…………………….97

Table 4.5.1.2.1: Local Data Definition for data elements………………….…99

Table 4.5.1.2.2: Local Data Definition for data elements………………….…99

Table 4.5.1.2.3: Local Data Definition for data elements…………………….99

Table 4.5.1.2.4: Local Data Definition for data elements………………….…100

Table 4.5.1.2.5: Local Data Definition for data elements…………………….100

Table 4.5.1.3.1: Local Data Definition for data elements………………….…103

Table 4.5.1.3.2: Local Data Definition for data elements…………………….103

Table 4.5.1.3.3: Local Data Definition for data elements………………….…103

Table 4.5.1.4.1: Local Data Definition for data elements………………….…105

Table 4.5.1.4.2: Local Data Definition for data elements………………….…105

Table 4.5.1.5.1: Local Data Definition for data elements…………………….106

Table 4.5.1.5.2: Local Data Definition for data elements………………….…107

Table 4.5.1.5.3: Local Data Definition for data elements………………….…107

Table 5.1: Data Dictionary…………………............................................……119

Table 7.1: Requirement Traceability Matrix…….................…………………122

# SCOPE

## Identification

System Name: Tuition Management System

Abbreviation: TMS

System ID No.: TMS-SE-2016

## Overview of the System

The main purpose of the system is to provide the following functions:

Registration

* This module is used by the staff to assists the student and teacher to complete the registration procedure. The student and teacher will register manually. Then, the staff will key in their information. Students can choose which tuition package they want to take, while teachers will be assign to teach subject based on their credential. Their detail will be recorded into the system and the ID will be generated for each and every new students and teachers that have registered. The student and teacher are allowed to use the system by login using their id number given.

Payment

* This module used by the staff to keep track the payment transaction when the student or parents paying the fees. The payment is made every month according to the total the total subjects they have taken. Receipt will be generated in every payment.

Attendance

* This module is used by the teacher to mark and keep track the attendance of the students for every class. Any student who absent for that class can be checked. The warning letter will be issued by the admin to the student who absent more that the limitation period.

Examine

* This module is used by the teacher to evaluate student’s performance. Once teacher login, they can either evaluate student by key in result, comments or view the student’s result. For student, they can view result when they login to the system.

Teaching

* This module used by the student to register subjects and for the teacher to upload syllabus documents of a particular subject like exercises, references / notes or extra study material for student who taking this subject. Therefore, student can log into their account to view and download the latest uploaded syllabus for only those taken subject. Student also able to view their timetable based on subject taken.

**Purpose of the System**

The main purpose of this system is to design better management of daily operations of or activities of the tuition centre for the student, teacher and staff. By using this system it can use to prove the student attend to the tuition class so that we can keep track of student who play truant, improve retrieving accurate information system and allow the student to view marks and quizzes for evaluations. Student also are able to view and download notes and exercises from this system.

## Overview of the Document

This paragraph summarizes the purpose and contents of this Software Design Document (SDD). It specifies the requirement aspects related to the Tuition Management System.

In general, this SDD is divided into 8 sections as the following:

|  |  |
| --- | --- |
| **Chapter 1** | Describes the scope identification, system overview and the document overview. |
| **Chapter 2** | Referenced documents, government documents and non-government documents. |
| **Chapter 3** | Describes the preliminary design for the CSCI. The preliminary design will identify CSC, the description of each CSC design, the characteristics of each CSC and the traceability of requirements set forth in SRS and IRS. |
| **Chapter 4** | Describes the detail design. |
| **Chapter 5** | Describes the global data elements within the CSCI. |
| **Chapter 6** | Describes each of the shared data files of the CSCI. |
| **Chapter 7** | Requirements traceability to SRS and IRS documents. |
| **Chapter 8** | Notes and abbreviations. |

# 

# REFERENCED DOCUMENTS

**2.1. Overview of the Documents**

Below is the standard that been used in this document.

1. IEEE Std 1016-1998, “IEEE Recommended Practice for Software Design Descriptions”, 1998 Edition, IEEE, 1998

<http://standards.ieee.org/findstds/standard/1016-1998.html>

**2.2. Reference Documents**

This section consists of the contractual documents and non-contractual documents.

2.2.1 Contractual documents

Not Applicable

2.2.2 Non-contractual documents

1. **Software Requirements Specification (SRS) for Tuition Management System (TMS)**

Version : 1.0

Date : Disember 2015

Author : Shaun The Sheep Company

Publisher : -

1. **Software Development Plan (SDP) for Tuition Management System (TMS)**

Version : 1.0

Date : November 2015

Author : Shaun The Sheep Company

Publisher : -

# PRELIMINARY DESIGN

## 3.1. System Overview

This paragraph identifies the TMS external interface components, which are describe in the following figure.

|  |
| --- |
|  |

Figure 3.1: TMS Design Overview

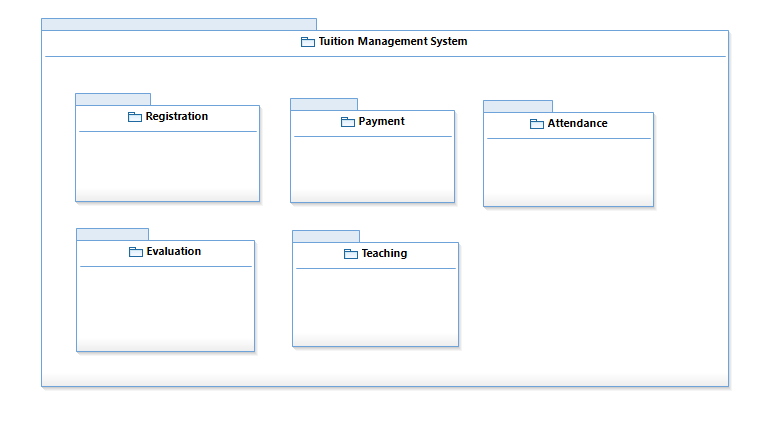
Figure 3.1 shows external interface diagram between the TMS and the actor. The staff use this system to manage their daily operations and activities while the student and teacher use the system to update their information by editing but not delete own profile. Students are also allowed to obtain teaching documents from their teacher.

The system’s functions shall manage registered student information, manage teacher information, manage payment, manage student’s attendance, manage teacher’s examine and teacher’s teaching materials. The server shall hold the database containing all the data that will be used by the system. The system will be written in Java and shall use Java Spring framework.

## 3.1.1 TMS Architecture

## 3.1.1.1 Static Organization

Figure 2 shows the static organization of TMS. The TMS consists of five module. The detail of the TMS will describe in next section.



**Examine**

**Figure 3.2: Static Organization for TMS**

This section describes the detail for each subsystem/package.

* 1. **Registration**

This subsystem is responsible to assists the student and teacher to complete the registration procedure. The student and teacher will register manually. Then, the staff will key in their information. Students can choose which tuition package they want to take, while teachers will be assign to teach subject based on their credential or qualification. Their detail will be recorded into the system and the ID will be generated for each and every new students and teachers that have registered. The student and teacher are allowed to use the system by login using their id number given.

* + 1. LoginPage
    2. LogController
    3. Authentication
    4. StudentPage
    5. StudentController
    6. Student
    7. Package
    8. Sub\_package
    9. TeacherPage
    10. TeacherController
    11. Teacher
  1. **Payment**

This subsystem is responsible to manage payment process in TMS. Payment process was divide into 3 sub process which its Student Fees, Teacher Wage and Claim Purpose. At student fees process tuition staff can insert detail about student fees and can be view by student. At teacher wage process tuition staff can insert detail about teacher wage and teacher view the detail. Teacher also can do a claim by giving the receipt to tuition staff and status of claim can be view by teacher after the detail about claim was insert by tuition staff.

* + 1. FeesController
    2. Fees
    3. FeesView
    4. WageController
    5. Wage
    6. WageView
    7. ClaimController
    8. Calim
    9. CalimView
  1. **Attendance**

This subsystem is responsible to manage attendance of students in TMS. Students scan their ID card and the scanner will capture the attendance. Attendance will be recorded to the database and students are able to view attendance. Attendance can be viewed by choosing 3 options which are viewing report by SubjectReport, WeeklyReport or MonthlyReport.

* + 1. Student
    2. Attendance Recorder Controller
    3. Scanner Controller
    4. Report
    5. Subject Report
    6. Monthly Report
    7. Weekly Report
    8. Subject Report Interface
    9. Monthly Report Interface
    10. Weekly Report Interface
  1. **Examine**

This subsystem is responsible to manage the result of the student. This package consists of the following classes or unit

* + 1. Evaluation Manager
    2. Comment
    3. Quiz
    4. Test
    5. ViewController
    6. ViewInterface
    7. StdEvaluationController
    8. stdEvaluation
  1. **Teaching**

This subsystem is responsible to allow the teacher to manage documents in TMS. The student are allowed to register subjects and get the document provided by their teacher. This package consists of the following classes or unit

* + 1. Register Subject
    2. Registration Controller
    3. Registration Form
    4. Subject Schedule
    5. Classroom
    6. Subject
    7. Subject Offer Controller
    8. Manage Subject
    9. Document
    10. Document Controller
    11. Attach Document
    12. Std Document Controller
    13. Document Doc

**3.1.1.2. Dynamic Organization**

Figure 3 diagram shows components and their relationships between each other in System.

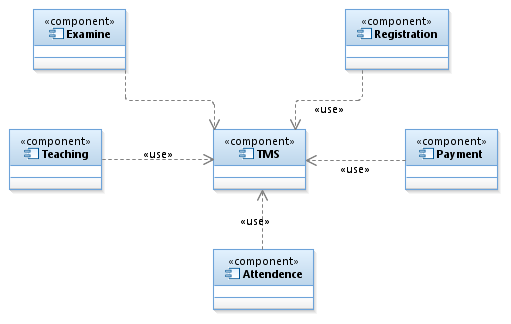


Figure 3.3: Dynamic Organization of Tuition Management System

Figure 3 shows the dynamic organization, which describes the dependency between components in the TMS. It shows the relationship of main program with its related CSCs and CSUs.

**3.1.1.3. CSCs Interfaces**

Figure 4 shows all the interaction between package / subsystem and non-development external subsystem/unit.

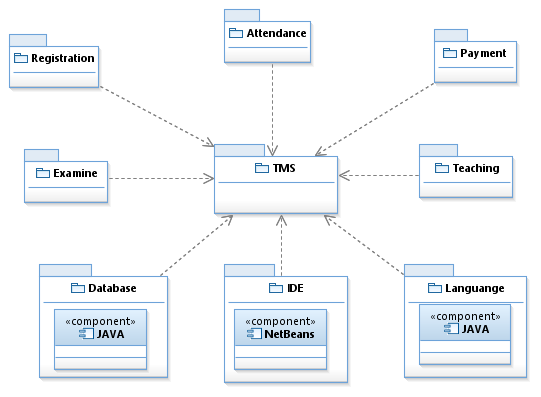
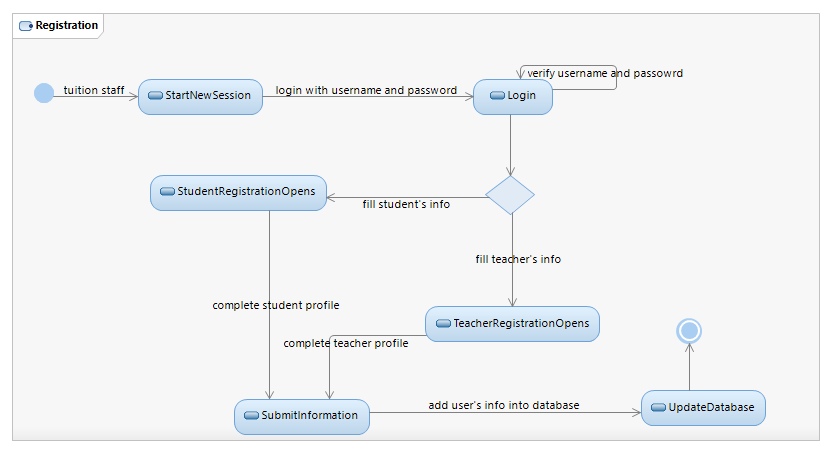
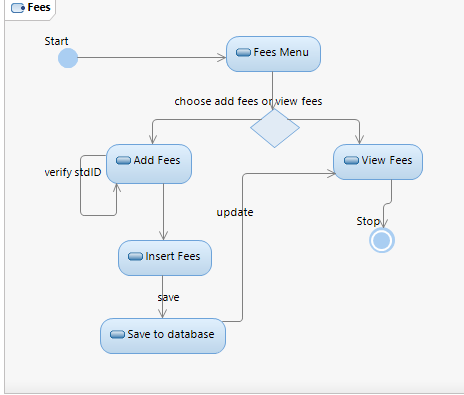


Figure 3.4:Package / Subsystem Interfaces

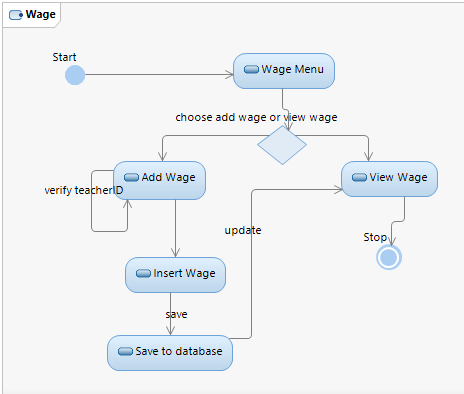
### 3.1.2. System States and Modes

This section describes states diagrams for Tuition Management System.

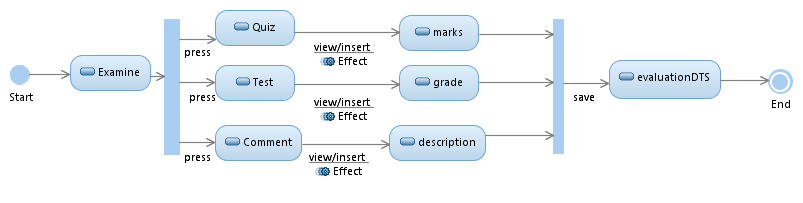
Figure 3.5 : State Diagram for Registration



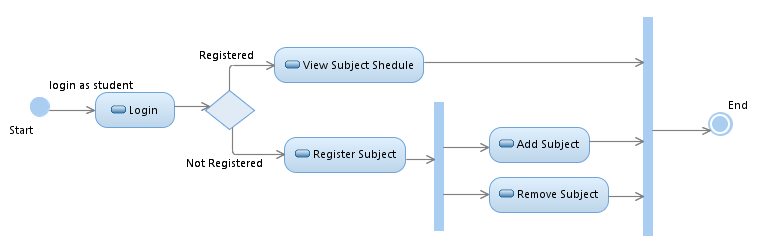
**Figure 3.6 : State Diagram for Fees**



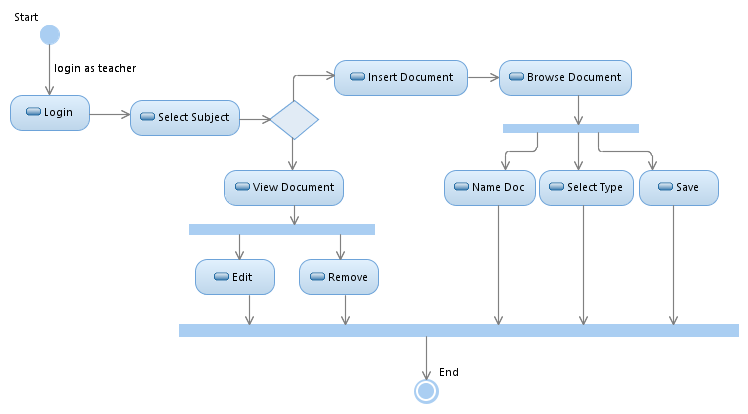
**Figure 3.7 : State Diagram for Wage**



**Figure 3.8 : State Diagram for Examine System**



**Figure 3.9:State Diagram for Subject Registration**



**Figure 3. 10:State Diagram for Upload Document**

## 

## 3.2. CSCI Design Description

Tuition Management System consists of following subsystems which provide design description of each CSC.

### 3.2.1 Registration Subsystem

This section shows the interaction in the registration subsystem.



Figure 3.11: Visibility of Registration

Figure 8 show the visibility of Registration Package. In registration packaged has including three classes to manage registration process in TMS. This figure also show connection between registration and other package in TMS.

### 

### 3.2.2 Payment Subsystem

This section shows the interaction in the payment subsystem

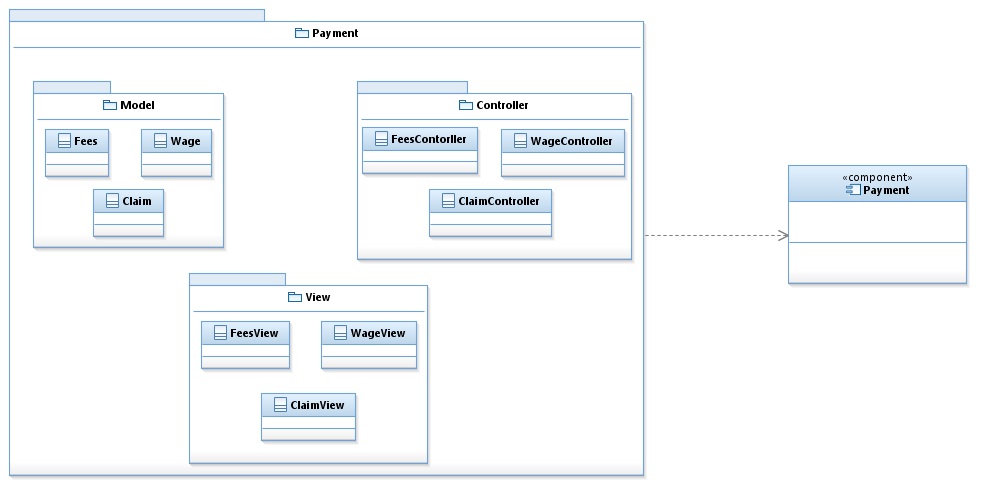


Figure 3.12: Visibility of Payment

Figure 7 show the visibility of Payment. This visibility manage payment process int TMS. This figure also show connection between Payment and other packages in TMS. This process is divided into three parts. Each parts has controller to communicate a process between model and view.

**3.2.3 Attendance Subsystem**

This section shows the interaction in the attendance subsystem.

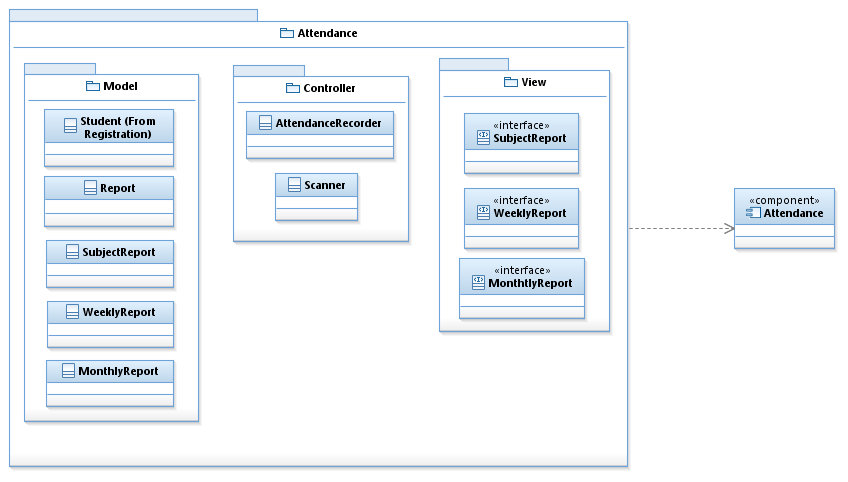


Figure 3.13: Visibility of Attendance

Figure 10 show the visibility of Attendance. In attendance packaged has including three classes to manage attendance process in TMS. This figure also show connection between attendance and other package in TMS.

### 3.2.4 Examine Subsystem

This section shows the interaction in the examine subsystem.

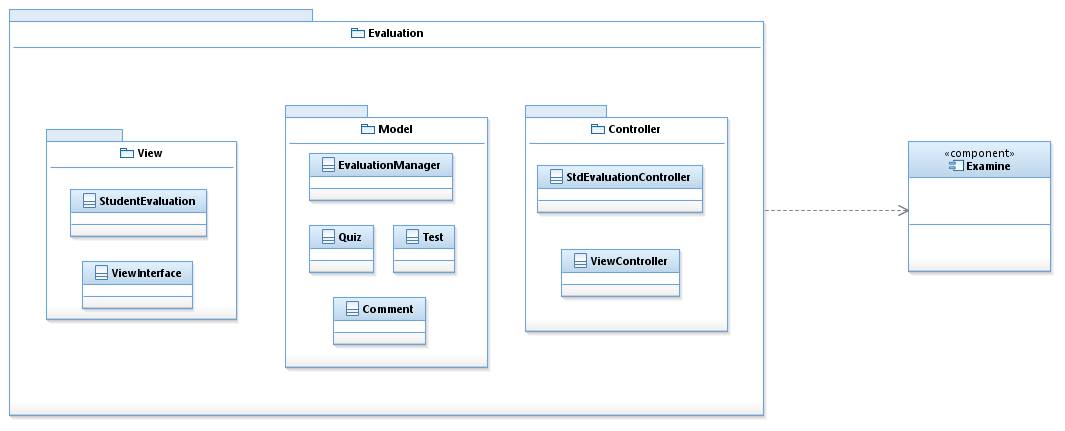


Figure 3.14: Visibility of Examine

Figure 11 show the visibility of Examine. In attendance examine packaged has including three classes to manage examine process in TMS. This figure also show connection between examine and other package in TMS.

### 3.2.5 Teaching Subsystem

This section shows the interaction in the teaching subsystem.

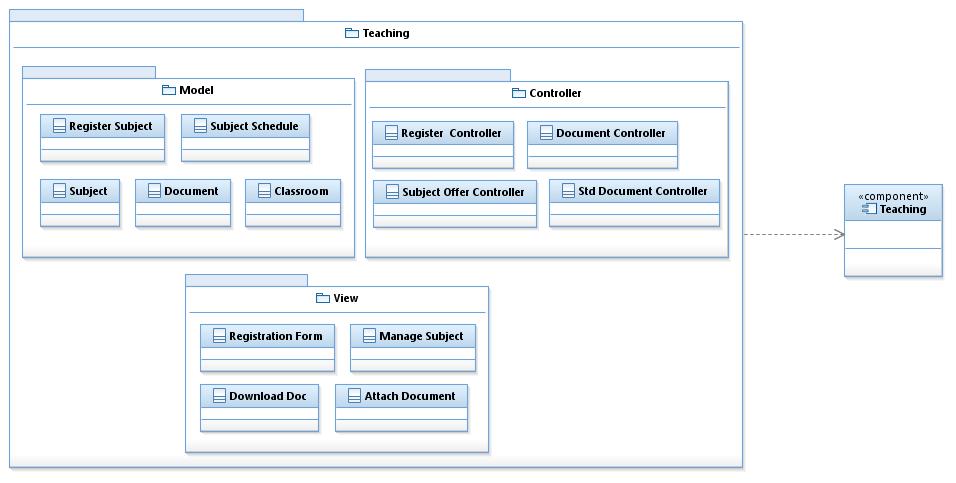


Figure 3.15: Visibility of Teaching

Figure 12 show the visibility of Teaching. In teaching packaged has including three classes to manage teaching process in TMS. This figure also show connection between attendance and other package in TMS.

1. **DETAILED DESIGN**

This section divided into the following paragraphs and subparagraphs to describe the detailed design.

**4.1 Registration Subsystem (SDD-REQ-S01)**

Figure 15 shows registration module. This subsystem need to describe the relationship among the other subsystem classes.



**Figure 4.1: Registration Detail Design**

**4.1.1 Model class design(SDD-REQ-4.1.1)**

This subparagraph specifies the design of model class.

1. Student
2. Teacher
3. Package
4. Sub\_package
5. Authentication

4.1.1.1 Student class (SDD-REQ-4.1.1.1)

1. **Input/output data elements**

List of input and output data elements :

Input : stdID, stdName, stdIC, stdPackage, stdAddress, stdGender, stdPhoneNo

Output : get student information

1. **Local data elements**

Table 4.1.1.1.1: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | stdID |
| Description | Contain student id |
| Data type | String |
| Precision/ resolution | Alphanumeric |

Table 4.1.1.1.2: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | stdName |
| Description | Contain student name |
| Data type | String |
| Precision/ resolution | - |

Table 4.1.1.1.3: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | stdIC |
| Description | Contain student ic number |
| Data type | Integer |
| Precision/ resolution | - |

Table 4.1.1.1.4: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | stdPackage |
| Description | Contain student package |
| Data type | Package class |
| Precision/ resolution | - |

Table 4.1.1.1.5: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | stdAddress |
| Description | Contain student address |
| Data type | String |
| Precision/ resolution | - |

Table 4.1.1.1.6: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | stdGender |
| Description | Contain student gender |
| Data type | char |
| Precision/ resolution | F and M |

Table 4.1.1.1.7: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | stdPhoneNo |
| Description | Contain student contact number |
| Data type | Integer |
| Precision/ resolution | - |

1. **Algorithms**

The purpose of this class is to allow tuition staff to input information of student into system.

Class type : Model class

Responsibility : To input information of student into system

Attributes : stdID : String

stdName : String

stdIC : Integer 12

stdPackage : Package

stdAddress : String

stdGender : char

stdPhoneNo : Integer

Method : getStdID, setStdID, getStdName, setStdName, getStdIC, setStdIC, getStdPackage, getStdAddress, setStdAddress, getStdGender, setStdGender, setStdPhoneNo, getStdPhoneNo

1. getStdID() and setStdID()

Responsibility : get and set student id

Input parameter :stdID

Output parameter :getStdID

Algorithm :

BEGIN

getStdID();

setStdID(stdID);

END

1. getStdName()and setStdName()

Responsibility : get and set student name

Input parameter : stdName

Output parameter : getStdName

Algorithm :

BEGIN

getStdName();

setStdName(stdName);

END

1. getStdIC()and setStdIC()

Responsibility : get and set student ic number

Input parameter : stdIC

Output parameter : getStdIC

Algorithm :

BEGIN

getStdIC();

setStdIC(stdIC);

END

1. getStdAddress()and setStdAddress()

Responsibility : get and set student address

Input parameter : stdAddress

Output parameter : getStdAddress

Algorithm :

BEGIN

getStdAddress();

setStdAddress(stdAddress);

END

1. getStdGender()and setStdGender()

Responsibility : get and set student gender

Input parameter : stdGender

Output parameter : getStdGender

Algorithm :

BEGIN

getStdGender();

setStdGender(stdGender);

END

1. getStdPhoneNo()and setStdPhoneNo()

Responsibility : get and set student phone number

Input parameter : stdPhoneNo

Output parameter : getStdPhoneNo

Algorithm :

BEGIN

getStdPhoneNo();

setStdPhoneNo(stdPhoneNo);

END

1. getStdPackage()

Responsibility : get student package

Input parameter : packID

Output parameter : getStdPackage

Algorithm :

BEGIN

getStdPackage();

END

**4.1.1.2 Teacher class (SDD-REQ-4.1.1.2)**

1. **Input/output data elements**

List of input and output data elements :

Input :tID, tName, tIC, tAddress, tGender, tPhoneNo, tEmail, tQualification, tContract, tAccNo

Output : get teacher information

1. **Local data elements**

Table 4.1.1.2.1: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | tID |
| Description | Contain teacher id |
| Data type | String |
| Precision/ resolution | Alphanumeric |

Table 4.1.1.2.2: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | tName |
| Description | Contain teacher name |
| Data type | String |
| Precision/ resolution | - |

Table 4.1.1.2.3: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | tIC |
| Description | Contain teacher ic number |
| Data type | Integer |
| Precision/ resolution | - |

Table 4.1.1.2.4: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | tAddress |
| Description | Contain teacher address |
| Data type | String |
| Precision/ resolution | - |

Table 4.1.1.2.5: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | tGender |
| Description | Contain teacher gender |
| Data type | Char |
| Precision/ resolution | F and M |

Table 4.1.1.2.6: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | tPhoneNo |
| Description | Contain teacher contact no |
| Data type | Integer |
| Precision/ resolution | - |

Table 4.1.1.2.7: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | tEmail |
| Description | Contain teacher email |
| Data type | String |
| Precision/ resolution | - |

Table 4.1.1.2.8: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | tQualification |
| Description | Contain teacher qualification |
| Data type | String |
| Precision/ resolution | - |

Table 4.1.1.2.9: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | tContract |
| Description | Contain teacher contract |
| Data type | Integer |
| Precision/ resolution | - |

Table 4.1.1.2.10 :Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | tAccNo |
| Description | Contain teacher account no |
| Data type | Integer |
| Precision/ resolution | - |

1. **Algorithms**

The purpose of this class is to allow tuition staff to input information of teacher into system.

Class type : Model class

Responsibility : To input information of teacher into system

Attributes : tID : String

tName : String

tIC : Integer 12

tAddress : String

tGender : char

tPhoneNo : Integer

tEmail : String

tQualification : String

tContract : integer

tAccNo : Integer

Methods : getTID (), setTID(), getTName(),setTName(),getTIC (),setTIC(), getTEmai()l,setTEmail(),getTAddress (), setTAddress, getTPhone (), setTPhone (), getTAccNo (), setTAccNo (), getTGender(), setTGender(), getTQualification(), setTQualification(), getTContract(), setTContract().

1. getTID() and setTID()

Responsibility : get and set teacher id

Input parameter :tID

Output parameter :getTID

Algorithm :

BEGIN

getTID();

setTID(tID);

END

1. getTName() and setTName()

Responsibility : get and set teacher name

Input parameter :tName

Output parameter :getTName

Algorithm :

BEGIN

getTName();

setTName(tName);

END

1. getTIC() and setTIC()

Responsibility : get and set teacher’s ic number

Input parameter :tIC

Output parameter :getTIC

Algorithm :

BEGIN

getTIC();

setTIC(tIC)

END

1. getTAddress() and setTAddress()

Responsibility : get and set teacher’s address

Input parameter :tAddress

Output parameter :getTAddress

Algorithm :

getTAddress();

setTAddress(tAddress );

BEGIN

END

1. getTGender() and setTGender()

Responsibility : get and set teacher’s gender

Input parameter :tGender

Output parameter :getTGender

Algorithm :

BEGIN

getTGender();

setTGender(tGender);

END

1. getTPhoneNo() and setTPhoneNo()

Responsibility : get and set teacher’s phone number

Input parameter :tPhoneNo

Output parameter :getTPhoneNo

Algorithm :

BEGIN

getTPhoneNo();

setTPhoneNo(tPhoneNo);

END

1. getTEmail() and setTEmail()

Responsibility : get and set teacher’s email

Input parameter :tEmail

Output parameter :getTEmail

Algorithm :

BEGIN

getTEmail();

setTEmail(tEmail);

END

1. getTQualification() and setTQualification()

Responsibility : get and set teacher qualification

Input parameter : tQualification

Output parameter :getTQualification

Algorithm :

BEGIN

getTQualification();

setTQualification(tQualification);

END

1. getTContract() and setTContract()

Responsibility : get and set teacher contract

Input parameter :tContract

Output parameter :getTContract

Algorithm :

BEGIN

getTContract();

setTContract(tContract);

END

1. getTAccNo() and setTAccNo()

Responsibility : get and set teacher account number

Input parameter :tAccNo

Output parameter :getTAccNo

Algorithm :

BEGIN

getTAccNo();

setTAccNo(tAccNo );

END

**4.1.1.3 Package class (SDD-REQ-4.1.1.3)**

1. **Input/output data elements**

List of input and output data elements :-

Input : packID, packName ,totalAmount

Output : get package

1. **Local data elements**

Table 4.1.1.3.1: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | packID |
| Description | Contain package id |
| Data type | String |
| Precision/ resolution | - |

Table 4.1.1.3.2: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | packName |
| Description | Contain package name |
| Data type | String |
| Precision/ resolution | - |

Table 4.1.1.3.3: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | totalAmount |
| Description | Contain total package |
| Data type | Double |
| Precision/ resolution | - |

1. **Algorithms**

The purpose of this class is to allow student to choose their tuition package

Class type : Model class

Responsibility : To input information of teacher into system

Attributes : packID : String

packName : String

totalAmount : Double

Methods :setPackID(), setPackName, setTotalAmount(), getPackID(), getPackName(), getTotalAmount()

1. getPackID() , setPackID()

Responsibility : get and set package id

Input parameter : String packID

Output parameter :packID

Algorithm :

BEGIN

getPackID();

setPackID(packID );

END

1. getPackName(), setPackName()

Responsibility : get and set package name

Input parameter : String packName

Output parameter :packName

Algorithm :

BEGIN

getPackName();

setPackName(packName);

END

1. getTotalAmount(), setTotalAmount()

Responsibility : get and set total amount of package

Input parameter : Double Total Amount

Output parameter : total maount

Algorithm :

BEGIN

getTotalAmount();

setTotalAmount(totalAmount );

END

4.1.1.4 Authentication class (SDD-REQ-4.1.1.4)

1. **Input/output data elements**

List of input and output data elements :-

Input : username, password

Output : verify username and password

1. **Local data elements**

Table 4.1.1.4.1: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | username |
| Description | Contain username for staff, teacher and student |
| Data type | String |
| Precision/ resolution | - |

Table 4.1.1.4.2: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | password |
| Description | Contain password for staff, teacher and student |
| Data type | String |
| Precision/ resolution | - |

1. **Algorithms**

The purpose of this class is to store the username and password.

Class type : Model class

Responsibility : To verify username and password

Attributes : username : String

Password : String

Methods : setUsername(), getUsername(), setPassword(), getPassword()

1. getUsername() , setUsername()

Responsibility : get and set username

Input parameter : String username

Output parameter : username

Algorithm :

BEGIN

getUsername();

setUsername( username );

END

1. getPassword(), setPassword()

Responsibility : set and password

Input parameter : String password

Output parameter : password

Algorithm :

BEGIN

getPassword();

setPassword( password );

END

4.1.1.5 Authentication class (SDD-REQ-4.1.1.5)

1. **Input/output data elements**

List of input and output data elements :-

Input : subjectCode

Output : get subject code

1. **Local data elements**

Table 4.1.1.5.1: Local Data Definition for Data Element

|  |  |
| --- | --- |
| Name | subjectCode |
| Description | Contain subject code |
| Data type | Subject |
| Precision/ resolution | - |

1. **Algorithms**

The purpose of this class is to list the subject based on package.

Class type : Model class

Responsibility : To list subject based on package

Attributes : subjectCode : String

Methods :getSubject()

1. getSubject()

Responsibility : get subject

Input parameter : String subjectCode

Output parameter : Subject

Algorithm :

BEGIN

getSubject(Subject);

END

4.1.2 Controller class design (SDD-REQ-4.1.2)

1. Reg Controller
2. Log Contoller

4.1.2.1 Student Controller class

1. **Input/output data elements**

List of input and output data elements :-

Input : none

Output : none

1. **Local data elements**

Not applicable

1. **Algorithms**

The purpose of this class is to control student registration into system.

Class type : Control class

Responsibility : To control the student registration

Attributes :

Methods : updateStudent() :Student

selectPackage() : Package

remStudent() : Student

1. updateStudent()

Responsibility : allow students to update and edit their information

Input parameter : stdID

Output parameter : none

Algorithm :

BEGIN

updateStudent(stdID);

END

1. remStudent()

Responsibility : to remove student from system

Input parameter : stdID

Output parameter : none

Algorithm :

BEGIN

remStudent(stdID);

END

1. selectPackage()

Responsibility : allow student to select tuition package

Input parameter : packID

Output parameter : none

Algorithm :

BEGIN

selectPackage(packID);

END

4.1.2.2 Teacher Controller class (SDD-REQ-4.1.2.2)

1. **Input/output data elements**

List of input and output data elements :-

Input : none

Output : none

1. **Local data elements**

Not applicable

1. **Algorithms**

The purpose of this class is to control teacher registration into system.

Class type : Control class

Responsibility : To control the teacher registration

Attributes : none

Methods : updateTeacher() : Teacher

remTeacher() : Teacher

Algorithm :

1. updateTeacher

Responsibility : allow teachers to update or edit their information

Input parameter : tID

Output parameter : none

Algorithm :

BEGIN

updateTeacher(tID);

END

1. remTeacher()

Responsibility : allow tuition staff to remove teacher from system.

Input parameter : tID

Output parameter : none

Algorithm :

BEGIN

remTeacher(tID);

END

4.1.2.3 Log Controller class (SDD-REQ-4.1.2.3)

1. **Input/output data elements**

List of input and output data elements :-

Input : none

Output : none

1. **Local data elements**

Not applicable

1. **Algorithms**

The purpose of this class is to verify the username and password.

Class type : Control class

Responsibility : To control the registration

Attributes : none

Methods : verify( username: String, password : String)

1. Verify()

Responsibility : to verify username and password

Input parameter : none

Output parameter : none

Algorithm :

BEGIN

Verify(username, password);

END

4.1.3 View class design

1. Login page
2. Registration page

4.1.3.1 Login page class (SDD-REQ-4.1.3.1)

1. **Input/output data elements**

List of input and output data elements :-

Input : none

Output : none

1. **Local data elements**

Not applicable

1. **Algorithms**

The purpose of this class is to login into system.

Class type : view class

Responsibility : to create interface for login

Attributes : not applicable

Methods : doLogin()

1. doLogin()

Responsibility : to create interface for login

Input parameter : none

Output parameter : none

Algorithm :

BEGIN

doLogin( username, password);

END

4.1.3.2Student view class

1. **Input/output data elements**

List of input and output data elements :-

Input : none

Output : none

1. **Local data elements**

Not applicable

1. **Algorithms**

The purpose of this class is to view student’s details.

Class type : view class

Responsibility : print student’s detail

Attributes : not applicable

Methods : printStudentDetails()

1. printStudentDetails()

Responsibility : to print student’s detail

Input parameter : stdID

Output parameter : getStudent

Algorithm :

BEGIN

printStudentDetails( stdID );

END

4.1.3.3Teacher view class (SDD-REQ-4.1.3.3)

1. **Input/output data elements**

List of input and output data elements :-

Input : none

Output : none

1. **Local data elements**

Not applicable

1. **Algorithms**

The purpose of this class is to view teacher’s detail.

Class type : view class

Responsibility : print teacher’s detail

Attributes : not applicable

Methods : printTeacherDetails()

1. printTeacherDetails()

Responsibility : print teacher’s details

Input parameter : tID

Output parameter : getTeacher

Algorithm :

BEGIN

printTeacherDetails( tID );

END

### 4.2 Payment (SDD-REQ-S02)

### 

### Figure 4.2 : Payment Detail Design

### Figure 4.2 shows Payment Package. This package divided to 3 subclass which it Student Fees, Teacher Wage and Claim Purpose.

**4.2.1 TMS Payment FeesController (SDD-REQ-4.2.1)**

This class for tuition staff manage student fees every month.

Class Type : Control Class

Responsibility : To control fees process in TMS.

Attributes : fees : Fees

view : FeesView

**Methods : TMS Payment FeesController contains the following methods :**

#### 4.2.1.1 FeesController (SDD-REQ-4.2.1.1)

This subparagraph specifies the design of FeesController

1. **Input/Output data elements**

List of input and output data elements:

Input : none

Output : none

1. **Local data elements**

Table 4.2.1.1.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | fees |
| Description | Class Fees |
| Data Type | Fees |
| Precision/resolution | - |

Table 4.2.1.1.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | view |
| Description | Class to view fees |
| Data Type | FeesView |
| Precision/resolution | - |

1. **Algorithms**

Class Type : Controller class

Responsibility :Control data flow between model fees and view fees

Attributes : fees : Fees

view : FeesView

**Methods :**

**1. *FeesController***

**Responsibility :** get and set model of fees and fees view

**Input Parameter :**None

**Output Parameter :** None

**Algorithm :**

BEGIN

Fees fees;

FeesView view;

FeesController(Fees fees, FeesView view)

END

**2.*SetFees and getFees***

**Responsibility :** To set and get attribute from Fees.

**Input Parameter :**setStdID, setStdName, setStdPackage, setFeesID, setFeesAmount, setFeesDate.

**Output Parameter :** getStdID, getStdName, getStdPackage, getFeesID, getFeesAmount, getFeesDate.

**Algorithm :**

BEGIN

fees.setStdID(stdID)

fees.getStdID()

fees.setStdName(stdName)

fees.getStdName()

fees.setStdPackage(stdPackage)

fees.getStdPackage()

fees.setFeesID(feesID)

fees.getFeesID()

fees.setFeesAmount(feesAmount)

fees.getFeesAmount()

fees.setFeesDate(feesDate)

fees.getFeesDate()

END

**3.*view()***

**Responsibility :** To view data from fees.

**Input Parameter :**none

**Output Parameter :**getStdID, getStdName, getStdPackage, getFeesID, getFeesAmount, getFeesDate

**Algorithm :**

BEGIN

view.viewFeesDetail(

fees.getStdID(),

fees.getStdName(),

fees.getStdPackage(),

fees.getFeesID(),

fees.getFeesAmount(),

fees.getFeesDate());

END

#### 4.2.1.2Fees (SDD-REQ-4.2.1.2)

This subparagraph specifies the design of Fees

1. **Input/Output data elements**

List of input and output data elements:

Input : stdID, stdName, stdPackage, FeesID, FeesAmount, FeesDate

Output : none

1. **Local data elements**

Table 4.2.1.2.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | stdID |
| Description | Get from Model Student |
| Data Type | Class |
| Precision/resolution | - |

Table 4.2.1.2.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | stdName |
| Description | Get from Model Student |
| Data Type | Class |
| Precision/resolution | - |

Table 4.2.1.2.3: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | stdPackage |
| Description | Get from Model Package |
| Data Type | Class |
| Precision/resolution | - |

Table 4.2.1.2.4: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | FeesID |
| Description | Number of fees ID |
| Data Type | Integer |
| Precision/resolution | Number |

Table 4.2.1.2.5: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | FeesAmount |
| Description | Amount of student fees |
| Data Type | Float |
| Precision/resolution | Number in format 00.00 |

Table 4.2.1.2.6: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | FeesDate |
| Description | Date of student pay a fees |
| Data Type | String |
| Precision/resolution | day/month/year (dd/mm/yy) |

1. **Algorithms**

Class Type : Class Model

Responsibility :To insert fees data

Attributes : student : Student

package : Package

FeesID : Integer

FeesAmount : Float

FeesDate : String

**Methods :**

**1. set*Fees()***

**Responsibility :** To set student fees

**Input Parameter :** setStdID, setStdName, setStdPackage, setFeesID, setFeesAmount, setFeesDate.

**Output Parameter :** None

**Algorithm :**

BEGIN

Student student = new Student();

Package pack = new Package();

fees.setFeesID(FeesID);

fees.setFeesAmount(FeesAmount);

fees.setFeesDate(FeesDate);

END

**2.get*Fees()***

**Responsibility :** To get student fees

**Input Parameter :** none

**Output Parameter :** getStdID, getStdName, getStdPackage, getFeesID, getFeesAmount, getFeesDate.

**Algorithm :**

BEGIN

Student student = new Student();

Package pack = new Package();

fees.getFeesID();

fees.getFeesAmount();

fees.getFeesDate();

END

#### 4.2.1.3FeesView (SDD-REQ-4.2.1.3)

This subparagraph specifies the design of FeesView

1. **Input/Output data elements**

List of input and output data elements:

Input : none

Output : stdID, stdName, stdPackage, FeesID, FeesAmount, FeesDate.

1. **Local data elements**

Table 4.2.1.3.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | stdID |
| Description | Get from Model Student |
| Data Type | Class |
| Precision/resolution | - |

Table 4.2.1.3.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | stdName |
| Description | Get from Model Student |
| Data Type | Class |
| Precision/resolution | - |

Table 4.2.1.3.3: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | studentPackage |
| Description | Get from Model Package |
| Data Type | Class |
| Precision/resolution | - |

Table 4.2.1.3.4: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | FeesID |
| Description | Number of fees ID |
| Data Type | Integer |
| Precision/resolution | Number |

Table 4.2.1.3.5: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | FeesAmount |
| Description | Amount of student fees. |
| Data Type | Float |
| Precision/resolution | Number in format 00.00 |

Table 4.2.1.3.6: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | FeesDate |
| Description | Date of student pay a fees |
| Data Type | String |
| Precision/resolution | day/month/year (dd/mm/yy) |

1. **Algorithms**

Class Type : Class View

Responsibility :To view student fees data

Attributes : student : Student

package : Package

FeesID : Integer

FeesAmount : Float

FeesDate : String

**Methods :**

**1. view*Fees()***

**Responsibility :** To view student fees

**Input Parameter :** none

**Output Parameter :** stdID, stdName, stdPackage, FeesID, FeesAmount, FeesDate.

**Algorithm :**

BEGIN

public void viewFeesDetail(

String stdID,

String stdName,

String stdPackage,

int feesID,

float feesAmount,

String feesDate)

END

**4.2.2 TMS Payment WageController (SDD-REQ-4.2.2)**

This class for tuition staff manage teacher wage every month.

Class Type : Control Class

Responsibility : To control wage process in TMS.

Attributes : wage : Wage

view : WageView

**Methods : TMS Payment WageController contains the following methods :**

#### 4.2.2.1WageController (SDD-REQ-4.2.2.1)

This subparagraph specifies the design of WageController

1. **Input/Output data elements**

List of input and output data elements:

Input : none

Output : none

1. **Local data elements**

Table 4.2.2.1.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | wage |
| Description | Class Wage |
| Data Type | Wage |
| Precision/resolution | - |

Table 4.2.2.1.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | view |
| Description | Class View Wage |
| Data Type | WageView |
| Precision/resolution | - |

1. **Algorithms**

Class Type : Controller class

Responsibility : To control data flow between model wage and view wage

Attributes : wage : Wage

view : WageView

**Methods :**

**1. *WageController***

**Responsibility :** get and set model of wage and wage view

**Input Parameter :**None

**Output Parameter :** None

**Algorithm :**

BEGIN

Wage wage;

WageView view;

WageController(Wage wage, WageView view)

END

**2.*SetWage and getWage***

**Responsibility :** To set and get attribute from Wage.

**Input Parameter :**setTID, setTName, setWageID, setWageAmount, setWageDate.

**Output Parameter :** getTID, getTName, getWageID, getWageAmount, getWageDate.

**Algorithm :**

BEGIN

wage.setTID(teacherID)

wage.getTeacherID()

wage.setTeacherName(teacherName)

wage.getTeacherName()

wage.setWageID(wageID)

wage.getWageD()

wage.setWageAmount(wageAmount)

wage.getWageAmount()

wage.setWageDate(wageDate)

wage.getWageDate()

END

**3.*view()***

**Responsibility :** To view data from wage.

**Input Parameter :**none

**Output Parameter :** getTID, getTName, getWageID, getWageAmount, getWageDate.

**Algorithm :**

BEGIN

view.viewWageDetail(

wage.getTID(),

wage.getTName(),

wage.getWageID(),

wage.getWageAmount(),

wage.getWageDate());

END

#### 4.2.2.2Wage (SDD-REQ-4.2.2.2)

This subparagraph specifies the design of Wage

1. **Input/Output data elements**

List of input and output data elements:

Input : tID, tName, WageID, WageAmount, WageDate

Output : none

1. **Local data elements**

Table 4.2.2.2.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | tID |
| Description | Get from Model Teacher |
| Data Type | Class |
| Precision/resolution | Alphabet and Number |

Table 4.2.2.2.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | tName |
| Description | Get from Model Teacher |
| Data Type | String |
| Precision/resolution | Alphabet |

Table 4.2.2.2.3: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | WageID |
| Description | Number of wage ID |
| Data Type | Integer |
| Precision/resolution | Number |

Table 4.2.2.2.4: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | WageAmount |
| Description | Amount of teacher fees |
| Data Type | Float |
| Precision/resolution | Number in format 00.00 |

Table 4.2.2.2.5: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | WageDate |
| Description | Date of teacher get a wage |
| Data Type | String |
| Precision/resolution | day/month/year (dd/mm/yy) |

1. **Algorithms**

Class Type : Class Model

Responsibility :To insert wage data

Attributes : teacher : Teacher

WageID : Integer

WageAmount : Float

WageDate : String

**Methods :**

1. ***setWage()***

**Responsibility :** To setteacher wage

**Input Parameter :** setTID, setTName, setWageID, setWageAmount, setWageDate.

**Output Parameter :** None

**Algorithm :**

BEGIN

Teacher teacher = new Teacher();

wage.setWageID(WageID);

wage.setWageAmount(WageAmount);

wage.setWageDate(WageDate);

END

1. ***getWage()***

**Responsibility :** To get teacher fees

**Input Parameter :**None

**Output Parameter :** getTID, getTName, getWageID, getWageAmount, getWageDate.

**Algorithm :**

BEGIN

Teacher teacher = new Teacher()

wage.getWageID();

wage.getWageAmount();

wage.getWageDate();

END

#### 4.2.2.3WageView (SDD-REQ-4.2.2.3)

This subparagraph specifies the design of WageView

1. **Input/Output data elements**

List of input and output data elements:

Input : none

Output : tID, tName, WageID, WageAmount, WageDate

1. **Local data elements**

Table 4.2.2.3.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | tID |
| Description | Get from Model Teacher |
| Data Type | Class |
| Precision/resolution | Alphabet and Number |

Table 4.2.2.3.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | tName |
| Description | Get from Model Teacher |
| Data Type | String |
| Precision/resolution | Alphabet |

Table 4.2.2.3.3: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | WageID |
| Description | Number of wage ID |
| Data Type | Integer |
| Precision/resolution | Number |

Table 4.2.2.3.4: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | WageAmount |
| Description | Amount of teacher wage |
| Data Type | Float |
| Precision/resolution | Number in format 00.00 |

Table 4.2.2.3.5: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | WageDate |
| Description | Date of teacher get wage |
| Data Type | String |
| Precision/resolution | day/month/year (dd/mm/yy) |

1. **Algorithms**

Class Type : Class View

Responsibility :To view teacher wage data

Attributes : teacher : Teacher

WageID : Integer

WageAmount : Float

WageDate : String

**Methods :**

1. ***viewWage()***

**Responsibility :** To view teacher wage

**Input Parameter :** None

**Output Parameter :** getTID, getTName, getWageID, getWageAmount, getWageDate.

**Algorithm :**

BEGIN

Public void viewWageDetail (

String tID,

String tName,

int wageID,

float wageAmount,

String wageDate)

END

**4.2.3TMS Payment ClaimController (SDD-REQ-4.2.3)**

This class for tuition staff manage teacher claim every month.

Class Type : Control Class

Responsibility : To control claim process in TMS.

Attributes : claim : Claim

view : ClaimView

**Methods : TMS Payment ClaimController contains the following methods :**

#### 4.2.3.1ClaimController (SDD-REQ-4.2.3.1)

This subparagraph specifies the design of ClaimController

1. **Input/Output data elements**

List of input and output data elements:

Input : none

Output : none

1. **Local data elements**

Table 4.2.3.1.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | claim |
| Description | Class Claim |
| Data Type | Claim |
| Precision/resolution | - |

Table 4.2.3.1.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | view |
| Description | Class View Claim |
| Data Type | ClaimView |
| Precision/resolution | - |

1. **Algorithms**

Class Type : Controller class

Responsibility : To control data flow between model claim and view claim

Attributes : claim : Claim

view : ClaimView

**Methods :**

**1. *ClaimController***

**Responsibility :** get and set model of claim and claim view

**Input Parameter :**None

**Output Parameter :** None

**Algorithm :**

BEGIN

Claim claim;

ClaimView view;

ClaimController(Claim claim, ClaimView view)

END

**2.*SetClaim and getClaim***

**Responsibility :** To set and get attribute from Claim.

**Input Parameter :**setTID, setTName, setClaimID, setClaimAmount, setClaimDate, setClaimInfo, setClaimStatus.

**Output Parameter :** getTID, getTName, getClaimID, getClaimAmount, getClaimDate, getClaimInfo, getClaimStatus.

**Algorithm :**

BEGIN

claim.setTID(tID)

claim.getTID()

claim.setTName(tName)

claim.getTName()

claim.setClaimID(claimID)

claim.getClaimID()

claim.setClaimAmount(claimAmount)

claim.getClaimAmount()

claim.setClaimStatus(claimStatus)

claim.getClaimStatus()

claim.setClaimInfo(claimInfo)

claim.getClaimInfo()

END

**3.*view()***

**Responsibility :** To view data from claim.

**Input Parameter :**none

**Output Parameter :** getTID, getTName, getClaimID, getClaimAmount, getClaimDate, getClaimInfo, getClaimStatus.

**Algorithm :**

BEGIN

view.viewClaimDetail(

claim.getTID(),

claim.getTName(),

claim.getClaimID(),

claim.getClaimAmount(),

claim.getClaimDate(),

claim.getClaimInfo(),

claim.getClaimStatus());

END

#### 4.2.3.2Claim(SDD-REQ-4.2.3.2)

This subparagraph specifies the design of Claim

1. **Input/Output data elements**

List of input and output data elements:

Input : tID, tName, ClaimID, ClaimAmount, ClaimDate, ClaimInfo, ClaimStatus

Output :

1. **Local data elements**

Table 4.2.3.2.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | tID |
| Description | Get from Model Teacher |
| Data Type | Class |
| Precision/resolution | Alphabet and Number |

Table 4.2.3.2.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | tName |
| Description | Get from Model Teachre |
| Data Type | String |
| Precision/resolution | Alphabet |

Table 4.2.3.2.3: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | ClaimID |
| Description | Number of claim ID |
| Data Type | Integer |
| Precision/resolution | Number |

Table 4.2.3.2.4: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | ClaimAmount |
| Description | Amount of teacher claim |
| Data Type | Float |
| Precision/resolution | Number in format 00.00 |

Table 4.2.3.2.5: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | ClaimDate |
| Description | Date of teacher get claim |
| Data Type | String |
| Precision/resolution | day/month/year (dd/mm/yy) |

Table 4.2.3.2.6: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | ClaimInfo |
| Description | Information about claim |
| Data Type | String |
| Precision/resolution | Alphabet |

Table 4.2.3.2.7: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | ClaimStatus |
| Description | Status of claim |
| Data Type | String |
| Precision/resolution | Alphabet |

1. **Algorithms**

Class Type : Class Model

Responsibility :To insert claim data

Attributes : teacher : Teacher

ClaimID : Integer

ClaimAmount : Float

ClaimDate : String

ClaimInfo : String

ClaimStatus : String

**Methods :**

1. ***setClaim()***

**Responsibility :** To set claim data

**Input Parameter :** setTID, setTName, setClaimID, setClaimAmount, setClaimDate, setClaimInfo, setClaimStatus.

**Output Parameter :** None

**Algorithm :**

BEGIN

Teacher teacher = new Teacher();

claim.setClaimID(ClaimID);

claim.setClaimAmount(ClaimAmount);

claim.setClaimDate(ClaimDate);

claim.setClaimInfo(ClaimInfo);

claim.setClaimStatus(ClaimStatus);

END

1. ***getClaim()***

**Responsibility :** To get claim data

**Input Parameter :** None

**Output Parameter :** getTID, getTName, getClaimID, getClaimAmount, getClaimDate, getClaimInfo, getClaimStatus.

**Algorithm :**

BEGIN

Teacher teacher = new Teacher();

claim.getClaimID();

claim.getClaimAmount();

claim.getClaimDate();

claim.getClaimInfo();

claim.getClaimStatus();

END

#### 4.2.3.3ClaimView(SDD-REQ-4.2.3.3)

This subparagraph specifies the design of ClaimView

1. **Input/Output data elements**

List of input and output data elements:

Input : None

Output : tID, tName, ClaimID, ClaimAmount, ClaimDate, ClaimInfo, ClaimStatus

1. **Local data elements**

Table 4.2.3.3.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | tID |
| Description | Get from Model Teacher |
| Data Type | Class |
| Precision/resolution | Alphabet and Number |

Table 4.2.3.3.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | tName |
| Description | Get from Model Teachre |
| Data Type | String |
| Precision/resolution | Alphabet |

Table 4.2.3.3.3: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | ClaimID |
| Description | Number of claim ID |
| Data Type | Integer |
| Precision/resolution | Number |

Table 4.2.3.3.4: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | ClaimAmount |
| Description | Amount of teacher claim |
| Data Type | Float |
| Precision/resolution | Number in format 00.00 |

Table 4.2.3.3.5: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | ClaimDate |
| Description | Date of teacher get claim |
| Data Type | String |
| Precision/resolution | day/month/year (dd/mm/yy) |

Table 4.2.3.3.6: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | ClaimInfo |
| Description | Information about claim |
| Data Type | String |
| Precision/resolution | Alphabet |

Table 4.2.3.3.7: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | ClaimStatus |
| Description | Status of claim |
| Data Type | String |
| Precision/resolution | Alphabet |

1. **Algorithms**

Class Type : Class View

Responsibility :To view claim data

Attributes : teacher : Teacher

ClaimID : Integer

ClaimAmount : Float

ClaimDate : String

ClaimInfo : String

ClaimStatus : String

**Methods :**

1. ***viewClaim()***

**Responsibility :** To view teacher claim

**Input Parameter :** None

**Output Parameter :** getTID, getTName, getClaimID, getClaimAmount, getClaimDate, getClaimInfo, getClaimStatus.

**Algorithm :**

BEGIN

Public void viewClaimDetail (

String tID,

String tName,

int claimID,

float claimAmount,

String claimDate

String claimInfo

String claimStatus )

END

**4.3 Attendance Module (SDD-REQ-S03)**

## Figure 15 shows Attendance Module subsystem.

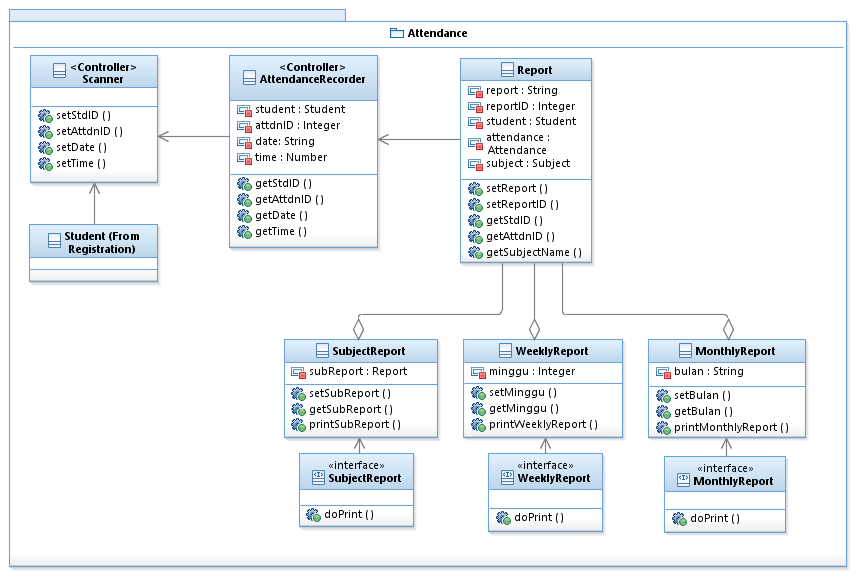


Figure 4.3 :Attendance Module subsystem Detail Design

**4.3.1 Class Scanner Design (SDD-REQ-4.3.1)**

This subparagraph specifies the design of Scanner class

1. **Input/Output data elements**

List of input and output data elements:

Input : stdID

Output : none

1. **Local data elements:** Not applicable
2. **Algorithms**

Class Type : Control class

Responsibility : To set and save StdID, AttdnID, date and time

Attributes : none

**Methods :**

***1.* setStdID()**

**Responsibility :** to set student ID

**Input Parameter :** String

**Output Parameter :** none

**Algorithm :**

BEGIN

Public void setStdID(stdID);

END

***2.* setAttdnID ()**

**Responsibility :** to set attendance ID

**Input Parameter :** none

**Output Parameter :** attdnID

**Algorithm :**

BEGIN

Public void setAttdnID(attdnID);

END

***3.* setDate()**

**Responsibility :** to set date

**Input Parameter :** none

**Output Parameter :** date

**Algorithm :**

BEGIN

Public void setDate(date);

END

***4.* setTime()**

**Responsibility :** to set time

**Input Parameter :** none

**Output Parameter :** time

**Algorithm :**

BEGIN

Public void setTime(time);

END

#### 4.3.2 Class AttendanceRecorder Design (SDD-REQ-4.3.2)

This subparagraph specifies the design of AttendanceRecorder class

1. **Input/Output data elements**

List of input and output data elements:

Input : stdID

Output : Verify attendance

1. **Local data elements**

Table 4.3.2.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | StdID |
| Description | Contain student ID |
| Data Type | Student |
| Precision/resolution | - |

Table 4.3.2.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | AttdnID |
| Description | Contain attendance ID |
| Data Type | String |
| Precision/resolution | Alphabet and Number |

Table 4.3.2.3: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | Date |
| Description | Capture date |
| Data Type | String |
| Precision/resolution | dd/mm/yyyy |

Table 4.3.2.4: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | Time |
| Description | Capture time |
| Data Type | Number |
| Precision/resolution | hh:mm:ss |

1. **Algorithms**

Class Type : Control class

Responsibility : Set attendance ID and send to database

Attributes : student: Student

attdnID: Integer

date: String

time: Number

**Methods :**

***1.* getStdID()**

**Responsibility :** to get student ID from Student

**Input Parameter :** none

**Output Parameter :** stdID

**Algorithm :**

BEGIN

Public String getStdID();

END

***2.* getAttdnID()**

**Responsibility :** to attendance ID from Scanner

**Input Parameter :** none

**Output Parameter :** attdnID

**Algorithm :**

BEGIN

Public String getAttdnID();

END

***3.* getDate()**

**Responsibility :** to get date from Scanner

**Input Parameter :** none

**Output Parameter :** date

**Algorithm :**

BEGIN

Public String getDate();

END

***4.*  getTime ()**

**Responsibility :** to get time from Scanner

**Input Parameter :** none

**Output Parameter :** time

**Algorithm :**

BEGIN

Public getTime();

END

**4.3.3 Class Report Design (SDD-REQ-4.3.3)**

This subparagraph specifies the design of Report class

1. **Input/Output data elements**

List of input and output data elements:

Input : stdID

Output : Show attendance report by subject

1. **Local data elements**

Table 4.3.3.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | report |
| Description | Contain report of attendance |
| Data Type | String |
| Precision/resolution | - |

Table 4.3.3.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | reportID |
| Description | Contain ID of report |
| Data Type | Integer |
| Precision/resolution | Number |

Table 4.3.3.3: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | stdID |
| Description | Contain ID of student |
| Data Type | Student |
| Precision/resolution | - |

**Table 4.3.3.4: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | attdnID |
| Description | Contain attendance ID |
| Data Type | Attendance |
| Precision/resolution | - |

Table 4.3.3.5: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | subjectName |
| Description | Contain name of subject |
| Data Type | String |
| Precision/resolution | - |

**c) Algorithms**

Class Type : Model class

Responsibility : Displays students’ report of attendance

Attributes : report: String

reportID: Integer

student: Student

attendance: Attendance

subject: Subject

**Methods :**

***1.* setReport()**

**Responsibility :** to write report

**Input Parameter :** none

**Output Parameter :** report

**Algorithm :**

BEGIN

Public void setReport(Report);

END

***2.* setReportID()**

**Responsibility :** to set report ID

**Input Parameter :** none

**Output Parameter :** none

**Algorithm :**

BEGIN

Public void setReportID(reportID);

END

***3.* getStdID()**

**Responsibility :** to get student ID from Student

**Input Parameter :** none

**Output Parameter :** stdID

**Algorithm :**

BEGIN

Public String getStdID();

END

***4.* getAttdnID()**

**Responsibility :** to get attendance ID from Attendance

**Input Parameter :** none

**Output Parameter :** attdnID

**Algorithm :**

BEGIN

Public String getAttdnID();

END

***5.* getSubjectName ()**

**Responsibility :** to get subject name from Subject

**Input Parameter :** none

**Output Parameter :** subReport

**Algorithm :**

BEGIN

Public String getSubjectName();

END

**4.3.4 Class SubjectReport Design (SDD-REQ-4.3.4)**

This subparagraph specifies the design of SubjectReport class

1. **Input/Output data elements**

List of input and output data elements:

Input : stdID

Output : Show attendance report by subject

1. **Local data elements**

Table 4.3.4.1 : Local Data Definition for data element

|  |  |
| --- | --- |
| Name | subReport |
| Description | Contain report of attendance |
| Data Type | Report |
| Precision/resolution | - |

1. **Algorithms**

Class Type : Model class

Responsibility : Displays students’ report according to the subjects registered

Attributes : subReport: Report

**Methods :**

***1.*** setSubReport() and getSubReport()

**Responsibility :** to print Subject Report attendance

**Input Parameter :** stdID

**Output Parameter :** subReport

**Algorithm :**

BEGIN

Public void setSubReport(subReport);

Public String getSubReport()

END

***2.*** printSubReport()

**Responsibility :** to print Subject Report attendance

**Input Parameter :** none

**Output Parameter :** subReport

**Algorithm :**

BEGIN

Public void printSubReport();

END

**4.3.1.5 Class SubjectReport Interface Design (SDD-REQ-4.3.1.5)**

This subparagraph specifies the design of MonthlyReport class

1. **Input/Output data elements**

List of input and output data elements:

Input : none

Output : Show students’ attendance by month

1. **Local data elements** :Not applicable
2. **Algorithms**

Class Type : Interface class

Responsibility : Displays students’ report according to the month

Attributes : none

**Methods :**

***1.*  doPrint()**

**Responsibility :** to create interface to print SubReport

**Input Parameter :** none

**Output Parameter :** none

**Algorithm :**

BEGIN

doPrint (subReport, stdID);

END

#### 4.3.6 Class Weekly Design (SDD-REQ-4.3.6) EEKLY

This subparagraph specifies the design of WeeklyReport class

1. **Input/Output data elements**

List of input and output data elements:

Input : stdID

Output : Show students’ attendance by week

1. **Local data elements**

Table 4.3.6.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | minggu |
| Description | Contain number of week |
| Data Type | Integer |
| Precision/resolution | Number |

1. **Algorithms**

Class Type : Model class

Responsibility : Displays students’ report according to the week

Attributes : minggu: Integer

**Methods :**

***1.* setMinggu()** and **getMinggu()**

**Responsibility :** to set number of week of lesson

**Input Parameter :** stdID

**Output Parameter :** WeeklyReport

**Algorithm :**

BEGIN

Public void setMinggu(minggu);

Public String getMinggu();

END

***2.* printWeeklyReport()**

**Responsibility :** to print Weekly Report attendance

**Input Parameter :** none

**Output Parameter :** WeeklyReport

**Algorithm :**

BEGIN

Public void printWeeklyReport();

END

**4.3.7 Class WeeklyReport Interface Design (SDD-REQ-4.3.7)**

This subparagraph specifies the design of MonthlyReport class

1. **Input/Output data elements**

List of input and output data elements:

Input : none

Output : Show students’ attendance by month

1. **Local data elements** : Not applicable
2. **Algorithms**

Class Type : Interface class

Responsibility : Displays students’ report according to the month

Attributes : none

**Methods :**

***1.*  doPrint()**

**Responsibility :** to create interface to print WeeklyReport

**Input Parameter :** none

**Output Parameter :** none

**Algorithm :**

BEGIN

doPrint (minggu, stdID);

END

**4.3.8 Class MonthlyReport Design (SDD-REQ-4.3.8)**

This subparagraph specifies the design of MonthlyReport class

1. **Input/Output data elements**

List of input and output data elements:

Input : stdID

Output : Show students’ attendance by month

1. **Local data elements**

Table 4.3.8.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | bulan |
| Description | Contain name of month |
| Data Type | String |
| Precision/resolution | - |

1. **Algorithms**

Class Type : Model class

Responsibility : Displays students’ report according to the month

Attributes : bulan: String

**Methods :**

***1.*  setBulan()** and **getBulan()**

**Responsibility :** to set month of lesson

**Input Parameter :** stdID

**Output Parameter :** MonthlyReport

**Algorithm :**

BEGIN

Public void setBulan(bulan);

Public String getBulan();

END

***2.*  printMonthlyReport()**

**Responsibility :** to print Monthly report attendance

**Input Parameter :** none

**Output Parameter :** MonthlyReport

**Algorithm :**

BEGIN

Public void printMonthlyReport();

END

**4.3.9 Class MonthlyReport Interface Design (SDD-REQ-4.3.9)**

This subparagraph specifies the design of MonthlyReport class

1. **Input/Output data elements**

List of input and output data elements:

Input : none

Output : Show students’ attendance by month

1. **Local data elements** : Not applicable
2. **Algorithms**

Class Type : Interface class

Responsibility : Displays students’ report according to the month

Attributes : none

**Methods :**

***1.*  doPrint()**

**Responsibility :** to create interface to print MonthlyReport

**Input Parameter :** none

**Output Parameter :** none

**Algorithm :**

BEGIN

doPrint (bulan, stdID);

END

**4.4 TMS Examine System (SDD-REQ-S04)**

Figure 15 shows evaluation. This subsystem need to describe the relationship among the other subsystem classes

|  |
| --- |
|  |

**Figure 4.4: Evaluation Detail Design**

**4.4.1 EvaluationManager (SDD-REQ-4.4.1)**

**Class Type :** Controller Class

**Responsibility:** Evaluation Manager is responsibility to control and manage the student Evaluation in this system.

**Attribute :** student :Student

subject :Subject

Marks :Quiz

grade :Test

description :Comment

**Method :** Evaluation Manager class contain the following method:

**Local data elements**

Table 4.4.1.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | stdID |
| Description | Get from model Student |
| Data Type | Class |
| Precision/resolution | Number and Alphabet |

**Table 4.4.1.2: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | stdName |
| Description | Get from model Student |
| Data Type | Class |
| Precision/resolution | Alphabet |

**Table 4.4.1.3: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | subjectCode |
| Description | Get from model Subject |
| Data Type | Class |
| Precision/resolution | Alphabet |

**Table 4.4.1.4: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | Marks |
| Description | Get from model Quiz |
| Data Type | Class |
| Precision/resolution | Number |

**Table 4.4.1.5 : Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | grade |
| Description | Get from model Test |
| Data Type | Class |
| Precision/resolution | Alphabet |

**Table 4.4.1.6 : Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | description |
| Description | Get from model Comment |
| Data Type | Class |
| Precision/resolution | Alphabet |

**4.4.1.1 getEvaluation**

**Responsibility:** To display the evaluation each student

**Input parameter:** none

**Output parameter:** stdID, stdName, subjectCode, Marks, grade, comment

**Pre-conditions:** Each student ID and name must identify.

**Algorithm:**

BEGIN

class EvaluationManager{

private String stdName;

private String stdID;

private String subjectCode;

private int Marks,Grade;

public EvaluationManager()

{

stdName = "Unsigned";

}

public String getStdName()

{

return stdName;

}

public String getStdID()

{

Return stdID;

}

public String getSubejctCode()

{

Return subjectCode;

}

public int getMarks()

{

return Marks;

}

public int getGrade()

{

return grade;

}

public String getDescription()

{

return description;

}

}

END

**Post condition:** Evaluation information are get.

**4.4.2 ViewController (SDD-REQ-4.4.2)**

**Class Type :** View class

**Responsibility:** To handling output of the evaluation each student

**Attribute :** student :Student

subject :Subject

Marks :Quiz

grade :Test

comment :Comment

**Method :** viewController class contain the following method:

**4.4.2.1 ViewEvaluation**

**Responsibility:** To display the evaluation each student

**Input parameter:** none

**Output parameter:** none

**Pre-conditions:** Each student ID and name must identify.

**Local data elements**

Table 4.4.2.1.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | stdID |
| Description | Get from model Student |
| Data Type | Class |
| Precision/resolution | Number and Alphabet |

**Table 4.4.2.1.2: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | stdName |
| Description | Get from model Student |
| Data Type | Class |
| Precision/resolution | Alphabet |

**Table 4.4.2.1.3: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | subjectCode |
| Description | Get from model Subject |
| Data Type | Class |
| Precision/resolution | Alphabet |

**Table 4.4.2.1.4: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | Marks |
| Description | Get from model Quiz |
| Data Type | Class |
| Precision/resolution | Number |

**Table 4.4.2.1.5: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | grade |
| Description | Get from model Test |
| Data Type | Class |
| Precision/resolution | Alphabet |

**Table 4.4.2.1.6: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | description |
| Description | Get from model Comment |
| Data Type | Class |
| Precision/resolution | Alphabet |

**Algorithm:**

BEGIN

Public class ViewEvaluation{

public void viewEvaluationDetail(String StdName, String StdID, String SubjectCode, String Test, Integer Quiz, String Comment)

System.out.println ("student name is" + stdName);

System.out.println ("student ID is" + stdID);

System.out.println ("subject code is" + subjectCode);

System.out.println (stdName +" test reulst is" + Test);

System.out.println (stdName + "quiz result is" + Quiz);

System.out.println (stdName + "comment is" + description);

}

END

**Post condition:** Evaluation displayed

**4.4.3 StdEvaluationController (SDD-REQ-4.4.3)**

**Class Type :** model class

**Responsibility:** To handling input of the marks and grade each student

**Attribute :** Marks :Quiz

grade :Test

**Method :** StdEvaluationController class contain the following method:

**Local data element**

Table 4.4.3.1: Local Data Definition for data element

|  |  |
| --- | --- |
| Name | Marks |
| Description | Get from model Quiz |
| Data Type | Class |
| Precision/resolution | Number |

**Table 4.4.3.2: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | grade |
| Description | Get from model Test |
| Data Type | Class |
| Precision/resolution | Alphabet |

**Table 4.4.3.3: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | description |
| Description | Get from model Comment |
| Data Type | Class |
| Precision/resolution | Alphabet |

**4.4.3.1 StdEvaluationView**

**Responsibility:** To view the evaluation and set the marks and grade for the test and quiz each student

**Input parameter:** none

**Output parameter:** none

**Pre-conditions:** Each student ID and name must identify.

**Algorithm:**

BEGIN

private int Marks,Grade;

Public class ViewEvaluation{

public void viewEvaluationDetail(String StdName, String StdID, String SubjectCode, String Test, Integer Quiz, String Comment)

System.out.println ("student name is" + stdName);

System.out.println ("student ID is" + stdID);

System.out.println ("subject code is" + subjectCode);

System.out.println (stdName +" test reulst is" + Test);

System.out.println (stdName + "quiz result is" + Quiz);

System.out.println (stdName + "comment is" + description);

public void setMarks(int marks)

{

This.Marks = marks

}

public void setGrades(int grade)

{

This.Grade = grade;

}

}

}

END

**Post Condition:** Evaluation displayed and marks and grade filled.

**4.4.4 Quiz (SDD-REQ-4.4.4)**

**Class Type :** Controller class

**Responsibility:** To handling input of the quiz each student

**Attribute :** Quiz : Integer

**Method :** Quiz class contain the following method:

**Local data element**

Table 4.4.4.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | Marks |
| Description | Get from model Quiz |
| Data Type | Class |
| Precision/resolution | Number |

**4.4.4.1 setQuiz**

**Responsibility:** To set the quiz each student

**Input parameter:** Quiz

**Output parameter:** none

**Pre-conditions:** none.

**Algorithm:**

BEGIN

public void setStdQuiz(int quiz)

{

Quiz = quiz;

}

END

**Post condition:** quiz is set

**4.4.4.2 getQuiz**

**Responsibility:** To get the quiz each student

**Input parameter:** none

**Output parameter:** Quiz

**Pre-conditions:** none

**Algorithm:**

BEGIN

public int getStdQuiz()

{

return Quiz;

}

END

**Post condition:** quiz is get

**4.4.5 Test (SDD-REQ-4.4.5)**

**Class Type :** Controller class

**Responsibility:** To handling input of the Test each student

**Attribute :** Test : String

**Method :** Test class contain the following method:

Table 4.4.5.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | grade |
| Description | Get from model Test |
| Data Type | Class |
| Precision/resolution | Alphabet |

**4.4.5.1 setTest**

**Responsibility:** To set the test each student

**Input parameter:** Test

**Output parameter:** none

**Pre-conditions:** none

**Algorithm:**

BEGIN

public void setStdTest(int test)

{

This.Test = test;

}

END

**Post condition:** test is set.

**4.4.5.2 getTest**

**Responsibility:** To get the test each student

**Input parameter:** none

**Output parameter:** Test

**Pre-conditions:**none

**Algorithm:**

BEGIN

public int getStdTest()

{

return Test;

}

END

**Post condition:** test is get

**4.4.6 Comment (SDD-REQ-4.4.6)**

**Class Type :** Controller class

**Responsibility:** To handling input of the comment each student

**Attribute :** Comment : String

**Method :** Comment class contain the following method:

**Local data element**

Table 4.4.6.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | comment |
| Description | Get from model Comment |
| Data Type | Class |
| Precision/resolution | Alphabet |

**4.4.6.1 setComment**

**Responsibility:** To set the comment each student

**Input parameter:** Comment

**Output parameter:** none

**Pre-conditions:** none

**Algorithm:**

BEGIN

public void set Comment (int comment)

{

This. Comment = comment;

}

END

**Post condition:** Comment is set.

**4.4.6.2 getComment**

**Responsibility:** To get the Comment each student

**Input parameter:** none

**Output parameter:** Comment

**Pre-conditions:**none

**Algorithm:**

BEGIN

public int get Comment ()

{

return Comment;

}

END

**Post condition:** Comment is get.

**4.4.7 Student (SDD-REQ-4.4.7)**

**Class Type :** Model class

**Responsibility:** To handling input of the student information each student

**Attribute :** stdID :Student

stdName :Student

**Method :** Student class contain the following method:

**Local data elements**

Table 4.4.7.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | stdID |
| Description | Get from model Student |
| Data Type | Class |
| Precision/resolution | Number and Alphabet |

Table 4.4.7.2: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | stdName |
| Description | Get from model Student |
| Data Type | Class |
| Precision/resolution | Alphabet |

**4.4.7.1 getStdName**

**Responsibility:** To get the student name each student.

**Input parameter:** none

**Output parameter:** stdName.

**Pre-conditions:** Each student name must identify.

**Algorithm:**

BEGIN

public int getStdName ()

{

return stdName;

}

END

**Post condition:** get the student name.

**4.4.7.2 getStdID**

**Responsibility:** To get the student ID each student

**Input parameter:** none

**Output parameter:** stdID

**Pre-conditions:** Each student ID must identify.

**Algorithm:**

BEGIN

public int getStdID ()

{

return stdID;

}

END

**Post condition:** get the student ID.

**4.4.8 Subject (SDD-REQ-4.4.8)**

**Class Type :** Model class

**Responsibility:** To handling input of the student information each student

**Attribute :** subjectCode :subject

**Method :** Subject class contain the following method:

Table 4.4.8.1: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | subjectCode |
| Description | Get from model Subject |
| Data Type | Class |
| Precision/resolution | Alphabet |

**4.4.8.1 getSubjectCode**

**Subject**

**Responsibility:** To get the subject code each subject for each student from the teaching module

**Input parameter:** none

**Output parameter:** subjectCode

**Pre-conditions:** Each student ID and name must identify.

**Algorithm:**

BEGIN

public int getSubjectCode()

{

return subjectCode;

}

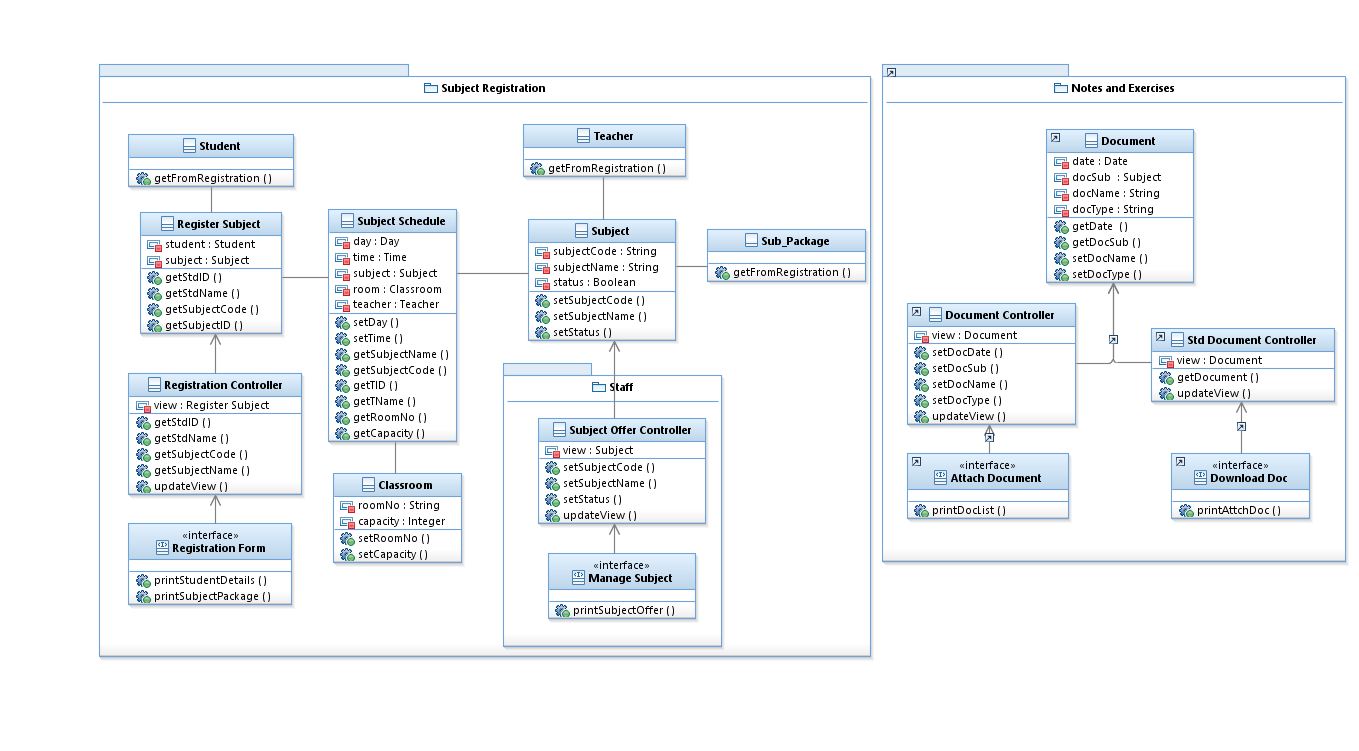
END

**Post condition:** Student subject code must be exist in the teaching module.

**4. DETAILED DESIGN**

**4.5 Teaching Subsystem (SDD-REQ-S05)**

Figure 1 shows Teaching Subsystem.



**Figure 4.5 : Teaching Detail Design**

**4.5.1 Model Package**

The purpose of this class is to describe the relationship among the other subsystem classes.

This package consists of :-

1. Register Subject
2. Subject Schedule
3. Subject
4. Classroom
5. Document

**4.5.1.1 Class Register Subject Design (SDD-REQ-4.5.1.1)**

This subparagraph specifies the design of Register Subject

1. **Input / Output data elements**

List of input and output data elements:

Input : student, subject

Output : Display student registered subject

1. **Local data elements**

**Table 4.5.1.1.1: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | student |
| Description | Contain the information of the registered student |
| Data Type | Student |
| Precision/resolution | Alphabet and Number |

**Table 4.5.1.1.2: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | subject |
| Description | Contain the detail of the subject |
| Data Type | Subject |
| Precision/resolution | Alphabet and Number |

1. **Algorithms**

Class Type : Model class

Responsibility : To record the registration subject of the student

Attributes : student : Student

subject : Subject

Methods : getStdID, getStdName, getSubjectCode, getSubjectName

1. ***getStdID ()***

**Responsibility :** To get the registered student ID

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getStdID (){

return student.stdID();}

END

1. ***getStdName ()***

**Responsibility :** To get the registered student name

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getStdName (){

return student.stdName();}

END

1. ***getSubjectCode ()***

**Responsibility :** To get subject code from class subject

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getSubjectCode (){

return subject.subjectCode();}

END

1. ***getSubjectName ()***

**Responsibility :** To get name of the subject code from class subject

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getSubjectName (){

return subject.subectName ();}

END

**4.5.1.2 Class Subject Schedule Design (SDD-REQ-4.5.1.2)**

This subparagraph specifies the design of Subject Schedule

1. **Input / Output data elements**

List of input and output data elements:

Input : Day, Time, subject, room, teacher

Output : Display the subject schedule

1. **Local data elements**

**Table 4.5.1.2.1: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | day |
| Description | Contain the day for the subject |
| Data Type | Day |
| Precision/resolution | Alphabet |

**Table 4.5.1.2.2: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | time |
| Description | Contain the time of subject |
| Data Type | Time |
| Precision/resolution | hh : mm |

**Table 4.5.1.2.3: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | room |
| Description | Contain room for the class |
| Data Type | Integer |
| Precision/resolution | Number |

**Table 4.5.1.2.4 : Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | subject |
| Description | Contain details of the subject |
| Data Type | String |
| Precision/resolution | Alphabet |

Table 4.5.1.2.5 : Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | teacher |
| Description | Contain the teacher that assign for the subject |
| Data Type | String |
| Precision/resolution | Alphabet |

1. **Algorithms**

Class Type : Model class

Responsibility : To lists all subjects that offered by the tuition center.

Attributes : day : Day

time : Time

subject : Subject

room : Classroom

teacher : Teacher

Methods **:** setDay, setTime, getSubject getClassroom, getTeacher.

1. ***setDay()***

**Responsibility :** Set the day of subject offered

**Input Parameter :** Day

**Output Parameter :** Display day

**Algorithm :**

BEGIN

public void setDay (String day) {

this.setDay(String day);}

END

1. ***setTime ()***

**Responsibility :** Set the time of the subject

**Input Parameter :** Time

**Output Parameter :** Display time in **“**hh : mm” format

**Algorithm :**

BEGIN

public void setTime ( Time ) {

this.setTime (hh, mm); }

END

1. ***getSubjectCode ()***

**Responsibility :** Get the list of code subject registered.

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getSubjectCode(){

return subject.subjectCode ();}

END

1. ***getSubjectName()***

**Responsibility :** Get the list of subject name offered by the tuition centre

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getSubjectName(){

return subject.subjectName ();}

END

1. ***getTID()***

**Responsibility :** Get teacher id that assign for the subject

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getTID ( ){

return teacher.tID();}

END

1. ***getTName()***

**Responsibility :** Get the teacher name base on their id

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getTName (){

return teacher.tName();}

END

1. ***getRoomNo()***

**Responsibility :** Get the room no for the subject

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getRoomNo (){

return classroom.roomNo();}

END

1. ***getCapacity()***

**Responsibility :** Get the capacity for each room

**Input Parameter :** tID, tName

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getCapacity(){

return subject.capacity ();}

END

**4.5.1.3 Class Subject Design (SDD-REQ-4.5.1.3)**

This subparagraph specifies the design of Subject

1. **Input/Output data elements**

List of input and output data elements:

Input : SubjectName, SubjectCode, Status

Output : Display the list of subjects

1. **Local data elements**

**Table 4.5.1.3.1: Local Data Definition for data element**

|  |  |
| --- | --- |
| Name | SubjectName |
| Description | Contain the name of the subject |
| Data Type | String |
| Precision/resolution | Alphabet |

**Table 4.5.1.3.2: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | SubjectCode |
| Description | Contain the code of the subject |
| Data Type | String |
| Precision/resolution | Alphabet and Number |

Table 4.5.1.3.3.: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | Status |
| Description | Indicate the status of the subject either its available or not |
| Data Type | Boolean |
| Precision/resolution | Available / Not Available |

1. **Algorithms**

Class Type : Model class

Responsibility : To list the subject offered by the tuition center

Attributes : SubjectName : String

SubjectCode : String

Status : Boolean

Methods **:** setSubjectName, setSubjectCode, setStatus

1. ***setSubjectName()***

**Responsibility :** Set the name of subject offered

**Input Parameter :** SubjectName, SubjectCode

**Output Parameter :** Display subject name

**Algorithm :**

BEGIN

public String setSubjectName(String subjectName,subjectCode){

SET (subjectCode) = subjectName;

return subject.setSubjectName(SubjectName);}

END

1. ***setSubjectCode ()***

**Responsibility :** Set the code for subject offered

**Input Parameter :** SubjectCode

**Output Parameter :** Display subject code

**Algorithm :**

BEGIN

public void setSubjectCode(String subjectCode){

SET subjectCode = “BCN1223”

return subject.setSubjectCode(subjectCode);}

END

1. ***setStatus()***

**Responsibility :** Set status of the subject offered

**Input Parameter :** Status

**Output Parameter :** Display the status of the subject

**Algorithm :**

BEGIN

IF capacity >= max\_capacity

Print status = “Not Available”;

ELSEIF

Print status = “Available”;

return status;

END

**4.5.1.4 Class Classroom Design (SDD-REQ-4.5.1.4)**

This subparagraph specifies the design of Classroom

1. **Input/Output data elements**

List of input and output data elements:

Input : RoomNo, Capacity

Output : Display subject classroom

1. **Local data elements**

**Table 4.5.1.4.1: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | RoomNo |
| Description | Contain the number of the room |
| Data Type | Integer |
| Precision/resolution | - |

**Table 4.5.1.4.2: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | Capacity |
| Description | Contain the capacity for the class |
| Data Type | Integer |
| Precision/resolution | - |

1. **Algorithms**

Class Type : Model class

Responsibility : To display the classroom of the subject with its capacity

Attributes : RoomNo : String

Capacity : Capaity

Methods :setRoomNo, setCapacity

1. ***setRoomNo ()***

**Responsibility :** Set Room Number from classroom class

**Input Parameter :** String roomNo, subjectCode

**Output Parameter :** RoomNo

**Algorithm :**

BEGIN

public void setRoomNo (String RoomNo, subjectCode){

SET (subjectcode) = roomNo;

return classroom.setRoomNo (RoomNo) ;}

END

1. ***setCapacity ()***

**Responsibility :** Set capacity for classroom

**Input Parameter :** RoomNo

**Output Parameter :** capacity

**Algorithm :**

BEGIN

public void setCapacity(String RoomNo) {

SET roomNo = 30;

return classroom.setCapacity(capacity);}

END

**4.5.1.5 Class Document Design (SDD-REQ-4.5.1.5)**

This subparagraph specifies the design of document

1. **Input/Output data elements**

List of input and output data elements:

Input : Date, DocSubject, DocName, DocType

Output : Display the details of the file

1. **Local data elements**

**Table 4.5.1.5.1: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | Date |
| Description | Contain day, month and year document being uploaded |
| Data Type | Date |
| Precision/resolution | dd / mm / yyyy |

**Table 4.5.1.5.2: Local Data Definition for data elements**

|  |  |
| --- | --- |
| Name | DocSubject |
| Description | Contain document subject |
| Data Type | String |
| Precision/resolution | Alphabet |

Table 4.5.1.5.3: Local Data Definition for data elements

|  |  |
| --- | --- |
| Name | DocType |
| Description | Contain types of the subject either notes or exercises |
| Data Type | String |
| Precision/resolution | Alphabet |

1. **Algorithms**

Class Type : Model class

Responsibility : To save documents that have been attach by the teacher

Attributes : Date : date

DocSubject : Subject

DocName : String

DocType : String

**Methods :** getDate, getDocSub, setDocName, setDocType

1. ***getDate()***

**Responsibility :** Get date of the document being upload

**Input Parameter :** Date

**Output Parameter :** Day, Month, Year

**Algorithm :**

BEGIN

public Date getDate(){

return date;

END

1. ***getDocSub()***

**Responsibility :** Get the document subject

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getDocSub()

return Document.getDocSub();

END

1. ***setDocName()***

**Responsibility :** Set the name or description of the document

**Input Parameter :** Sting

**Output Parameter :** DocName

**Algorithm :**

BEGIN

public void setDocName(String DocName)

SET docName = “aaa”

return document.setDocName(DocName);

END

1. ***setDocType()***

**Responsibility :** set the type of the document either note or exercise

**Input Parameter :** DocName

**Output Parameter :** DocType

**Algorithm :**

BEGIN

public void setDocType (String DocName){

SET DocType = “Note” or

SET DocType = “Exercise”

return document.setDocType (DocType);}

END

**4.5.2 Controller Package (SDD-REQ-4.5.2)**

The purpose of this class is to describe the relationship among the other subsystem classes.

1. Register Controller
2. Subject Offer Controller
3. Document Controller
4. Std Document Controller

**4.5.2.1 Class Registration Controller Design (SDD-REQ-4.5.2.1)**

This subgraph specifies the design of Registration Controller

1. **Input/Output data elements**

List of input and output data elements:

Input : Not Applicable

Output : Not Applicable

1. **Local data elements**

Not Applicable

1. **Algorithms**

Class Type : Controller class

Responsibility : To manage the register subject of the student

Attributes : ViewRegistration

Methods : getStdID, getStdName, getSubjectCode, getSubjectName, updateView

1. ***getStdID ()***

**Responsibility :** Get the registered student ID

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getStdID (){

return student.stdID();}

END

1. ***getStdName ()***

**Responsibility :** Get the student name from their ID

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getStdName (){

return student.stdName();}

END

1. ***getSubjectCode ()***

**Responsibility :** Get the code of the subject

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getSubjectCode (){

return subject.subjectCode();}

END

1. ***getSubjectName ()***

**Responsibility :** Get the subject name base on its code

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getSubjectdName (){

return student.subjectName();}

END

1. ***UpdateView ()***

**Responsibility :** Update the view for registration subject

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

Display RegistrationSubject;

IF changes = Update RegistrationSubject;

END

**4.5.2.2 Class Subject Offer Controller Design (SDD-REQ-4.5.2.2)**

This subparagraph specifies the design of Subject Offer Controller

1. **Input/Output data elements**

List of input and output data elements:

Input : Not Applicable

Output : Not Applicable

1. **Local data elements**

Not Applicable

1. **Algorithms**

Class Type : Controller class

Responsibility : To manage the subject offered by the tuition center

Attributes : viewSubject

Methods : setSubjectCode, setSubjectName, UpdateView

1. ***setSubjectCode()***

**Responsibility :** Set the subject that need to be offered

**Input Parameter :** SubjectCode

**Output Parameter :** SubjetCode

**Algorithm :**

BEGIN

public void setSubjectCode(String subjectCode){

SET subjectCode = “BCN1223”

return subject.setSubjectCode(subjectCode);}

END

1. ***setSubjectName()***

**Responsibility :** Set the subject that need to be offered

**Input Parameter :** SubjectCode

**Output Parameter :** SubjectName

**Algorithm :**

BEGIN

public void setSubjectCode(String subjectCode){

SET SubjectName(subjectCode) = “Physics”

return subject.setSubjectName(subjectName);}

END

1. ***UpdateView()***

**Responsibility :** Update the view for Subject class

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

Display Subject;

IF changes = Update Subject;

END

**4.5.2.3 Class Document Controller Design (SDD-REQ-4.5.2.3)**

This subparagraph specifies the design of Document Controller

1. **Input/Output data elements**

List of input and output data elements:

Input : Not Applicable

Output : Not Applicable

1. **Local data elements**

Not Applicable

1. **Algorithms**

Class Type : Controller Class

Responsibility : To manage the document that need to attach.

Attributes : ViewDocument

Methods :setDocSubject, setDocName, setDocType, UpdateView

1. ***setDocSubject()***

**Responsibility :** Set the subject of the document

**Input Parameter :** SubjectName, SubjectCode

**Output Parameter :** DocSubject

**Algorithm :**

BEGIN

public void setDocSubject (String SubjectName, SubjectCode)

return document.setDocSubject (DocSubject);

END

1. ***setDocName()***

**Responsibility :** Set the name or description of the document

**Input Parameter :** String

**Output Parameter :** DocName

**Algorithm :**

BEGIN

public void setDocName (String DocName)

SET docName = “Chapter 1”

return document.setDocName (DocName);

END

1. ***setDocType()***

**Responsibility :** Set the type of the document

**Input Parameter :** DocName

**Output Parameter :** DocType

**Algorithm :**

BEGIN

public void setDocType ( String DocName)

Set DocType (DocName) = “Exercise”;

return document.SetDocType (DocType);

END

1. ***UpdateView()***

**Responsibility :** Update view for Document Class

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

Display Document;

IF changes = Update Document;

END

**4.5.2.4 Class Std Document Controller Design (SDD-REQ-4.5.2.4)**

This subparagraph specifies the design of Std Document Controller

1. **Input/Output data elements**

List of input and output data elements:

Input : Not Applicable

Output : Not Applicable

1. **Local data elements**

Not Applicable

1. **Algorithms**

Class Type : Controller Class

Responsibility : To manage the document to be downloaded by the student

Attributes : ViewDocument

Methods :getDocument

1. ***getDoc()***

**Responsibility :** get document that have been uploaded

**Input Parameter :** None

**Output Parameter :** None

**Algorithm :**

BEGIN

public String getDoc()

return document.Doc();

END

**4.5.3 View Package**

The purpose of this class is to describe the relationship among the other subsystem classes.

1. Registration Form
2. Manage Subject
3. Download Doc
4. Attach Document

**4.5.3.1 Class Registration Form Design (SDD-REQ-4.5.3.1)**

This subparagraph specifies the design of Registration Form

1. **Input/Output data elements**

List of input and output data elements:

Input : Not Applicable

Output : Not Applicable

1. **Local data elements**

Not Applicable

1. **Algorithms**

Class Type : View Class

Responsibility : To control the operation for subject registration

Attributes : None

Methods **:** printStudentDetails, printSubjectPackgae

1. ***printStudentDetails* *()***

**Responsibility :** To view registered student name and code

**Input Parameter :** None

**Output Parameter :** stdID, stdName

**Algorithm :**

BEGIN

public void viewRegForm(){

view.printStudentDetails (student.getStdID (), student.getStdName ());}

END

1. ***printSubjectPackage ()***

**Responsibility :** To view subject offered base on student package

**Input Parameter :** None

**Output Parameter :** subjectCode, subjectName

**Algorithm :**

BEGIN

public void viewRegForm(){

view.printSubjectPackage (subject.getSubjectCode(), subject.getSubjectName());}

END

**4.5.3.2 Class Manage Subject Design (SDD-REQ-4.5.3.2)**

This subparagraph specifies the design of Manage Subject

1. **Input/Output data elements**

List of input and output data elements:

Input : Not Applicable

Output : Not Applicable

1. **Local data elements**

Not Applicable

1. **Algorithms**

Class Type : View Class

Responsibility : To control the operation of the subject offered

Attributes : None

Methods **:** printSubjectOffer

1. ***printSubjectOffer()***

**Responsibility :** To view subject offered by the tuition

**Input Parameter :** None

**Output Parameter :** subjectCode, subjectName

**Algorithm :**

BEGIN

public void viewManageSubject(){

view.printSubjectOffer (subject.getSubjectCode(), subject.getSubjectName());}

END

**4.5.3.3 Class Download Document Design (SDD-REQ-4.5.3.3)**

This subparagraph specifies the design of Download Document

1. **Input/Output data elements**

List of input and output data elements:

Input : Not Applicable

Output : Not Applicable

1. **Local data elements**

Not Applicable

1. **Algorithms**

Class Type : Interface class

Responsibility : To control the operation to download the document

Attributes : None

Methods **:** printAttachDoc

1. ***printAttachDoc()***

**Responsibility :** To view the attached document

**Input Parameter :** None

**Output Parameter :** docSub, docName, docType

**Algorithm :**

BEGIN

public void viewDldDoc(){

view.printAttachDoc (document.getDocSub(), document.getDocName()document.getDocType());}

END

**4.5.3.4 Class Attach Document Design (SDD-REQ-4.5.3.4)**

This subparagraph specifies the design of Attach Document

1. **Input/Output data elements**

List of input and output data elements:

Input : Not Applicable

Output : Not Applicable

1. **Local data elements**

Not Applicable

1. **Algorithms**

Class Type : Interface class

Responsibility : To control the operation for document attachment

Attributes : None

Methods **:** printDocList

1. ***printDocList()***

**Responsibility :** To view the document that need to be attach

**Input Parameter :** None

**Output Parameter :** DocSub, DocName, DocType

**Algorithm :**

BEGIN

public void viewAttchDoc(){

view.printDocList(document.getDocSub(), document.getDocName()document.getDocType());}

END

# 5. DATA DICTIONARY

**Table 5.1 : Data Dictionary**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Key | Reference | Description | |
| StdID | varchar(8) | PK |  | A unique ID to identify each student | |
| FullName | varchar(50) |  |  | A field that indicate student and teacher full name | |
| ICNo | varchar(12) |  |  | A field that indicate student and teacher ICNo | |
| Gender | char(1) |  |  | A field that indicate student and teacher gender | |
| PhoneNo | varchar(20) |  |  | A field that indicate student phone number | |
| Address | varchar(50) |  |  | A field that indicate student address | |
| Package | varchar(15) |  |  | A field that indicate student package | |
| Guardian | varchar(50) |  |  | A field that indicate student guardian | |
| tID | varchar(8) | PK |  | A unique ID to identify each teacher | |
| Contract | varchar(30) |  |  | A field that indicate teacher contract | |
| Qualification | varchar(20) |  |  | A field that indicate teacher qualification | |
| ContactNo | varchar(20) |  |  | A field that indicate teacher contact number | |
| Email | varchar(20) |  |  | A field that indicate teacher email | |
| AccountNo | varchar(20) |  |  | A field that indicate account number | |
| PackID | varchar(20) | PK |  | A unique ID to identify each package | |
| PackName | varchar(10) |  |  | A field that indicate package name | |
| TotalAmount | varchar(10) |  |  | A field that indicate total amount of packaged | |
| FeesID | integer | PK |  | A unique ID to identify each fees ID | |
| FeesAmount | float |  |  | | A field that indicate fees amount |
| FeesDate | varchar(15) |  |  | | A field that indicate fees date. |
| WageID | integer | PK |  | | A unique ID to identify each wage ID |
| WageAmount | float |  |  | | A field that indicate wage amount |
| WageDate | varchar(15) |  |  | | A field that indicate wage date. |
| ClaimID | integer | PK |  | | A unique ID to identify each wage ID |
| ClaimAmount | float |  |  | | A field that indicate wage amount |
| ClaimDate | varchar(15) |  |  | | A field that indicate wage date. |
| ClaimInfo | varchar(50) |  |  | | A field that indicate information about claim. |
| ClaimStatus | varchar(10) |  |  | | A field that indicate status of claim. |
| AttdnID | varchar(10) | FK |  | | A field that indicate attendance ID |
| Date | varchar(20) |  |  | | A field that indicate date |
| Time | varchar(10) |  |  | | A field that indicate time |
| ReportID | varchar(10) | PK |  | | A field that indicate report ID |
| Minggu | varchar(2) |  |  | | A field that indicate number of week |
| Bulan | varcahr(10) |  |  | | A field that indicate month |
| RoomNo | varchar(10) |  |  | | A number that indicate classroom of a subject |
| Capacity | varchar(5) |  |  | | A field that indicate the capacity of the classroom |
| SubjectCode | varchar(20) | FK |  | | A unique code to identify each subject |
| SubjectName | varchar(10) |  |  | | A field that indicate the name of the subject |
| Status | varchar(8) |  |  | | A field that indicate the status of the subject |
| DocSubject | varchar(20) |  |  | | A field that indicate the subject of the document |
| DocName | varchar(50) |  |  | | A field that indicate the name or description of the subject |
| DocType | varchar(10) |  |  | | A field that indicate the type of the document (notes / exercises) |

**6. REQUIREMENTS TRACEBILITY**

This chapter provides the requirements traceability for TMS.

**Table 6.1: Requirement Traceability Matrix**

|  |  |  |
| --- | --- | --- |
| Requirement | | Description |
| Source | Allocated |
| SRS-FAD-01 | SDD-REQ-4.1.1 | Admin is allow to enter student and teacher credentials into the system. |
| SRS-FAD-02 | SDD-REQ-4.1.2 | Teacher and student able to view, update, and delete the profile |
| SRS-FAD-03 | SDD-REQ-4.2.2 | Admin can update student’s receipt number and teacher’s wage in the system. |
| SRS-FTE-01 | SDD-REQ-4.2.3 | Teacher is able to key in amount of claim purpose. |
| SRS-FTE-02 | SDD-REQ-4.4.3 | Teacher is allow to key in student’s marks, description and grade into the system. |
| SRS-FTE-03 | SDD-REQ- 4.4.2 | Teacher is allow to view students evaluation |
| SRS-FTE-04 | SDD-REQ- 4.4.3 | Teacher is allow to delete, update, save the student evaluation |
| SRS-FTE-05 | SDD-REQ-4.4.7 | Student is allow to view evaluations |
| SRS-FTE-05 | SDD-REQ-4.3.1 | Teacher must scan their ID card in order to record their attendance. |
| SRS-NF-04 | SDD-REQ- 4.5.1. | Student able to register subject based on package |
| SRS-NF-05 | SDD-REQ- 4.5.3. | Teacher able to upload document for notes and exercise |
| SRS-NF-06 | SSD-REQ- 4.5.3.3 | Student able to download document |
| Non- Functional Requirement | | |
| SRS-NF-01 | | System only allow the user to login for 3 times only. |
| SRS-NF-02 | | Time changes from one function to another must not exceed 7 seconds. |
| SRS-NF-03 | | The system will automatically log out if the user did not make any operation with the system for 30 minutes. |

# 8. NOTES AND ABBREVIATION

Abbreviation used:

* UMP Universiti Malaysia Pahang
* CSCI Computer Software Configuration Items
* DOD Department of Defence
* SDD Software Development Document
* SDP Software Development Plan
* SRS Software Requirement Specification
* UML Unified Modelling Language