Software Design Specification

for

Student Management System (SIS)

Version 1.0 approved

Prepared by

Muiez Arif(BSSE02153166)

Taimoor Mushtaq(BSSE02153139)

Haris Akmal(BSSE02153163)

Ali riaz(BSSE02153075)

Muhammad Damil(BSSE02153171)

Date: April 8,2018

**Course enrollment:**

**Software Design Specification:**

Table of contents:

1. Introduction
2. Design considerations
3. Architectural Strategies
4. System Architecture
5. Policies and Tactics
6. Detailed System Design
7. Glossary
8. **Introduction**
   1. **Document Outline:**

The purpose of this document is to produce a detailed outlining of student management system. This document will help the developers to fully understand the system’s functionalities of student management system, also this document contains UML diagrams such as class diagram, use case diagram and sequence diagrams which provide the abstract as well as dynamic view of the whole system.

The document also provides the architecture design of system in detail and the strategies that need to be implemented to achieve the goal. Design policies and rules are clearly mentioned in it. In the end design patterns are applied on the whole system or to any component of system.

* 1. **Document Description:**

The SDD document provides an abstract and dynamic aspect of system which need to developed. The behavior of system is defined through the sequence diagram, the class diagram provides the system’s abstractions and the user interaction is described through usecase diagram. The functional and non-functional requirements which are described in software requirement and specification document are implemented in SDD.

**1.2.1) System Overview:**

The student management system is designed to let students enroll/drop their courses, view their attendance, print fee voucher, view exam details. There are three types of users.

1. Admin
2. Students
3. Teachers

Students can enroll/drop their courses, view their attendance, print fee voucher and view their exam details. Teachers can upload attendance, upload exam marks and can update or change it whereas Admin’s task is to manage the teachers and students in different ways i.e offer courses, assign courses to teachers, allot room numbers and handle exceptions.

**2) Design Consideration**

This section addresses some issues that need to be handled while developing the system.

**2.1) Assumptions and Dependencies:**

The student management system depends on the availability of operating environment.

As this is a Java-based application so it is assumed that both the server and client machines have eclipse installed in it and the version should be 1.5.0 or later.

Operating system will be windows or mac.

**2.2) General constraints:**

a) **Requirements**

All the functional and non-functional requirements that are mentioned in SRS document should be implemented.

b) **Protocol requirements:**

Student management system uses http protocol as it is distributive system.

c) **Hardware Constraints:**

Hardware system should be at least corei7 for best performance.

d) **Software Constraints:**

Student management system is developed in java so user should have eclipse 1.5.0 or latest version installed in its system.

e) **Memory and capacity limitations:**

Operating system must have at least 8Gb of ram and 500Gb HDD in it.

f) **Testing:**

The system will be tested as unit testing and on complete system test.

**2.3) Goals and Guidelines:**

* The system will be available all the time.
* It must perform all the functional and non-functional requirements as stated in SRS document.

**2.4) Development method:**

The incremental software development process will be used for the student management system as requirements are highly volatile or unstable, so changes are easy to made.

**4) System Architecture:**

**4.1) Components:**

* **Student Component:**

In this component the main functionalities that need to be performed are:

* Student can view his attendance and exam details.
* Student can print their fee vouchers.
* Student can enroll subjects.
* Students can drop subjects.
* **Teacher component:**

In this component the main functionalities that need to be performed are:

* Teacher can upload attendance.
* Teacher can upload exam marks.
* Teacher can modify records.
* **Admin Component**

In this component the main functionalities that need to be performed are:

* Admin can modify student details after 2 weeks of uploaded.
* Admin can assign teachers, courses.
* Admin can assign rooms for classes.
* Can make student not allowed for exams.
* **Database Component**

All of system’s data is stored in database and mange it. The framework 4.5 will be used in the student management system that is entity framework.

**4.2) Interfaces:**

* **Student Interface:**

This will be the homepage for students when logged in. the following tags will be shown on the left of page side.

* View profile
* Update profile
* View attendance
* View exam details
* Print fee voucher
* Add/drop courses
* Change password
* **Admin Interface:**

The admin has rights to grant access to teachers and students in a controlled way. The admin can restrict any student or teacher and maintain their records. It can also delete their records.

* **Teacher Interface:**

The homepage for teachers has the following tags and placed on the left of the page as shown in student’s interface.

* Upload attendance
* View attendance
* Upload exam marks
* Notify students

**4.3) Architecture Design:**

The Student Management System will be designed under two (hybrid) architectural patterns. These are:

* MVC architecture.
* Client/Server Architecture.
* **MVC:**

MVC architecture is used because there are multiple views against the same business logic so MVC is preferred for such system. The system is divided into three main components which are:

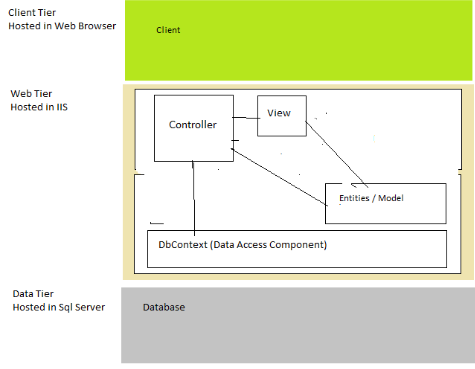
* **Model**:

All of the system’s business logic is written in model it manages the data and interact with databases.

* **View**:

It manages, how data is represented to users. There are multiple views, which are logically set against some interfaces.

* **Controller**:

It contains functions which are associated with model and views. Controller passes request to model to fetch data from database and then calls view to display it.

* **Client/Server Architecture:**

In a client server architecture services are served by server and the client uses it separately which is the beauty of this architecture.  
It is used when data is used from different locations and by different users.

It is divided into three tires which are:

* Data tire
* Middle tire
* Client tier

• Data Tire

Its only work is to maintain data.

• Middle Tire

The business logics and functional work that controller performs in MVC.

Because this is a functional layer that is why client cannot access it directly or indirectly.

• Client Tire

user interacts with directly this tier and its work is to retrieve data from server and the display it to user.

**Diagram:**

**Pc2**

**Pc1**

**SERVER**

**Pc3**

**Design Decisions:**

* **MVC Architecture:**

Reasons:

• Flexible pattern, can interact with other devices easily.  
• Easy to change code.  
• best for multiple viewed systems.

.

**POLICIES AND TACTICS:**

There are some of the policies and tactics which are finalized with our respectable client.

Some of the main point and policies are discussed below.

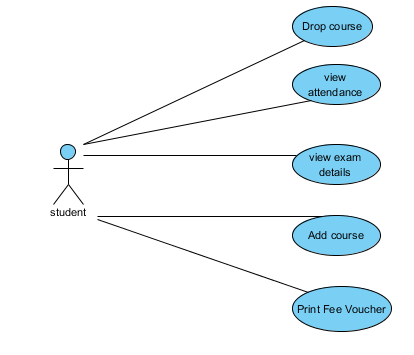
Before delivering the SYSTEM, whole payment should be made by the customer

* We will provide them a Database which will contain all data of students enrolled in the system
* We will provide them the source code
* We will provide maintenance which will cost 10 percent of the whole system after every year.
* There will be proper testing of the system and then it will be delivered to the client.
* System installation will be charged separately but training of one week will be given for free.
* Prototype will be showed to customer first.
* 3 months of time period is required for complete development.

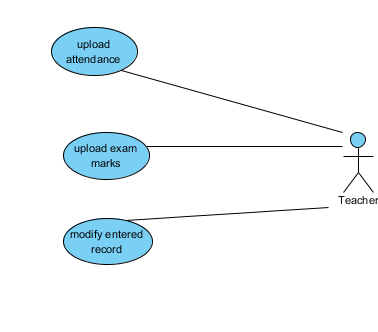
# Detailed System Design:

**UseCase Diagram:**

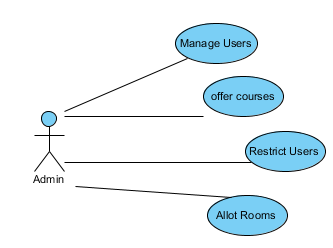
* **Student:**



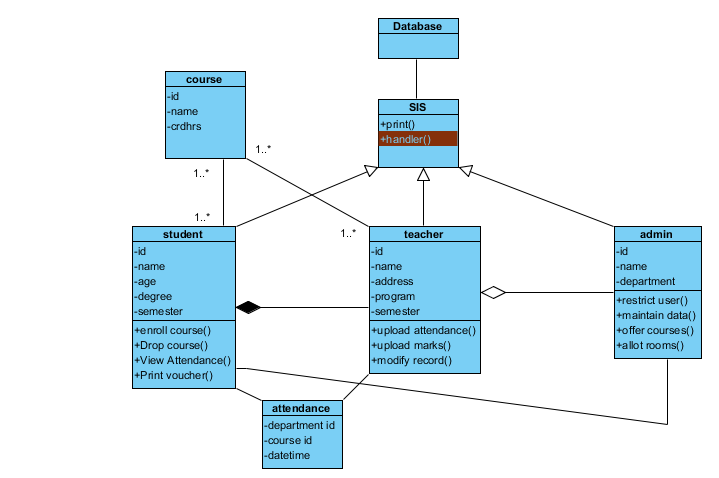
* **Teacher:**



* **Admin:**

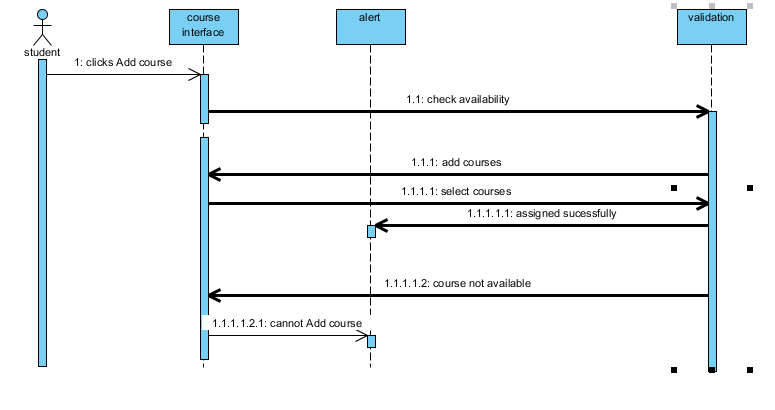


* **CLASS DIAGRAM:**

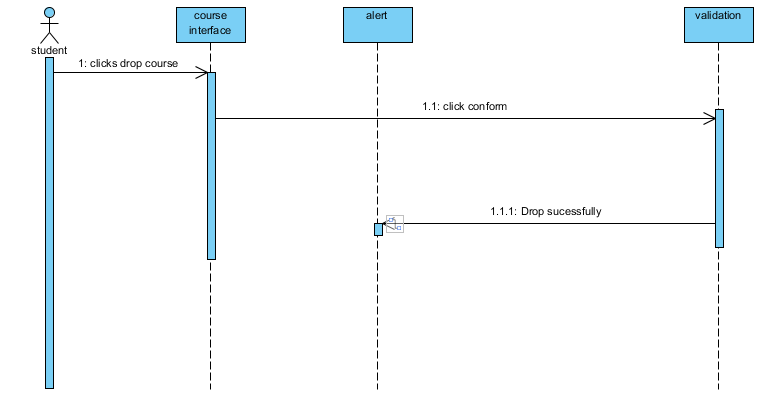


* **Sequence Diagram:**

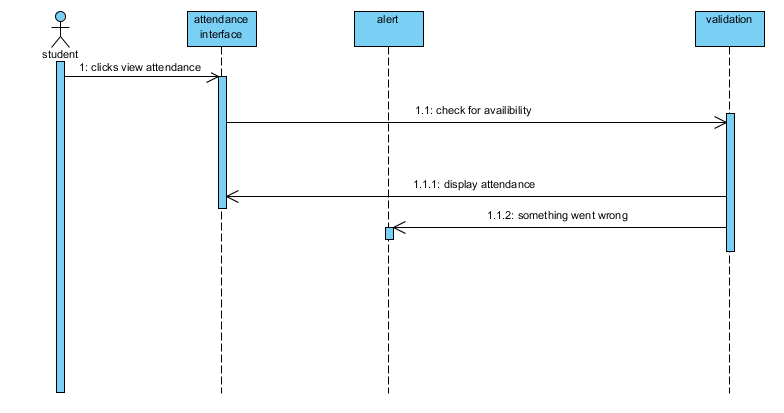
1. course enrollment:



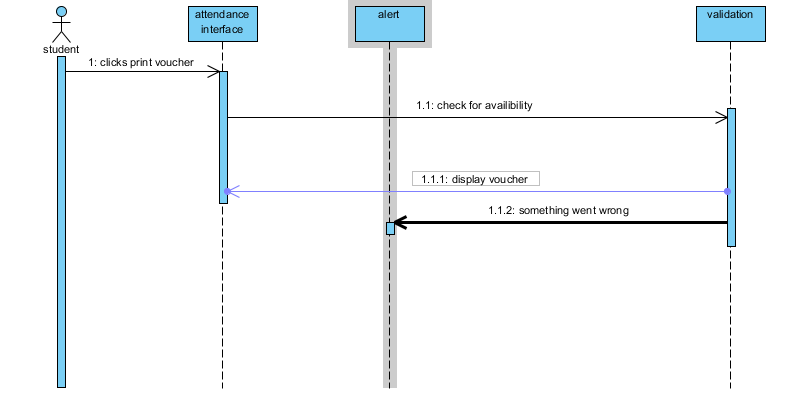
1. Drop course:



1. View Attendance:

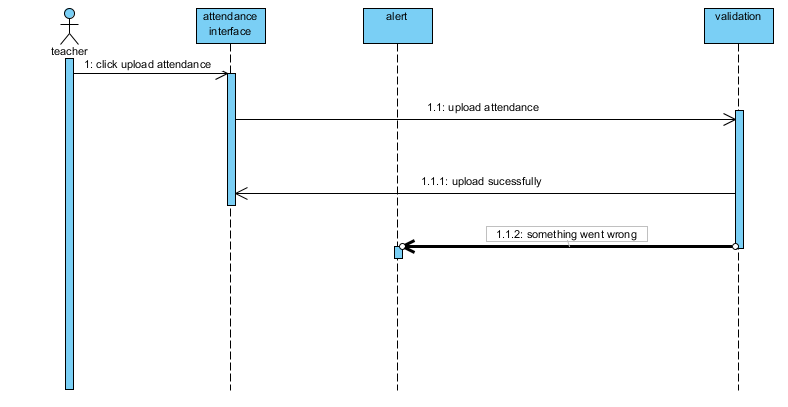


1. Print Voucher:

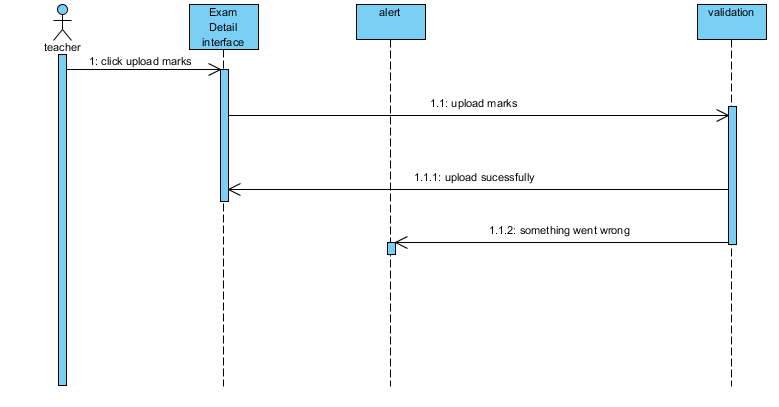


* Teacher

1. Upload Attendance:

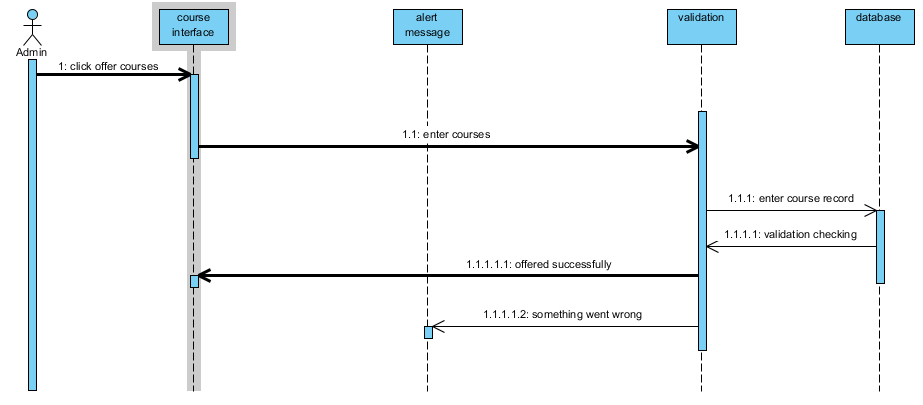


1. Upload Marks:



* **Admin**

1. Offer Courses:



**Glossary:**

| **Word** | **Description** |
| --- | --- |
| SDD | Software design document |
| SRS | Software requirement and specification |
| HDD | Hard disk drive |
| Eclipse | Platform for java. |
| EFR | Entity framework. |