Software Design Specification

for

course enrollment System

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:**

This software design specification is made with the purpose of outlining the software architecture and design of the course Registration System in detail. The document will provide developers an insight in meeting client’s needs efficiently and effectively. Moreover the document facilitates communication  
and understanding of the system by providing several views of the system design.

The next part of the document has described architectural design of the course Registration System. The high level components and their interactions, suitable architectural patterns, physical arrangement of components and design decisions applied to the whole system.  
The 3rd and final part of the System Design Specification is on Component and detailed design.  
Includes design patterns, sequence diagrams, class diagrams, database design in detail and user interface design.

* 1. **Document Description:**

The Software design document would demonstrate how the design will accomplish the functional and non- functional requirements captured in the Software Requirement specification (SRS). The document will provide a framework to the programmers through describing the high level components and architecture, sub systems, interfaces, database design and algorithm design. This is achieved through the use of architectural patterns, design patterns, sequence diagrams, class diagrams, relational models and user interfaces.

**1.2.1) System Overview:**

The course registration and enrollment system is being design for university students to get enroll their courses that are offered by university administration. There are two types of accessors.

1. Administration
2. Users
3. Students
4. Teachers

Admin’s work is to manage the users and to overlook their works, whereas users are of two types, student can get enroll in their courses also their course grades are also be shown, uploaded by their respective teachers. The system also manage the attendance record of each user.

**2) Design Consideration**

The design considerations section describes many of the issues that need to be addressed or resolved before attempting to devise a complete design solution.

**2.1) Assumptions and Dependencies:**

Course registration system depends on the availability of the external system web service. Since this provider web service is being developed on a different schedule on a different program, any modifications or issues with the interface will be difficult to resolve.

As a Java-based application, it assumes that the server and client machines running the software program will have a eclipse installed version 1.5.0 or later.

Operating system will be windows or mac.

**2.2) General constraints:**

a) **Requirements:** Refer to the course enrollment Software Requirements Specification (SRS) for the complete set of functional and non-functional requirements.

b) **Network Communications:** Course enrollment system is a distributed project that uses SOAP over HTTP to transfer data between the Program and the External System. Networking design is out of scope for this and will be handled by the network group of the Program.

**c) Hardware Constraints:**

None

**d) Software Constraints:**

Course registration system depends on the availability of the external system web service. Since this provider web service is being developed on a different schedule on a different program, any modifications or issues with the interface will be difficult to resolve.

**2.3) Goals and Guidelines:**

* Other than network errors, no internal software failures are acceptable.
* Support a Service Oriented Architecture (SOA) using industry accepted standards and specifications.
* Where possible, use known design patterns and employ software best practices.

**2.4) Development method:**

The incremental software development process will be used for the course enrollment architecture development as requirements are volatile or not stable.so change is easy to made.  
An incremental software development process is necessary to incrementally deliver functionality by a deadline

**4) System Architecture:**

**4.1) Components:**

* **Student Component:**

This is one of the key components of the Course Registration System. This is where the student  
subject selection option is implemented. This also includes the result preview, semester preview and  
profile functions.

* **management component**

This is the major sub system that is responsible for the security of the Course Registration system. It  
authenticate users and also handles the user management activities such as creating new user accounts, removing accounts, offering courses from the system.

* **Subject Component**

This is the key component that implements the functions related to the subject operations of administrator such as adding a new subject, editing credits of an existing subject and removing subjects etc.  
Subject component is also responsible for displaying the available subject list for every semester.

* **Notifying Component**

Notifying component is the component responsible for publishing notices created by the administrator  
and also the time tables. This component has the ability to publish multiple notices and time tables  
at the same time.

**4.2) Interfaces:**

* **Student Component**

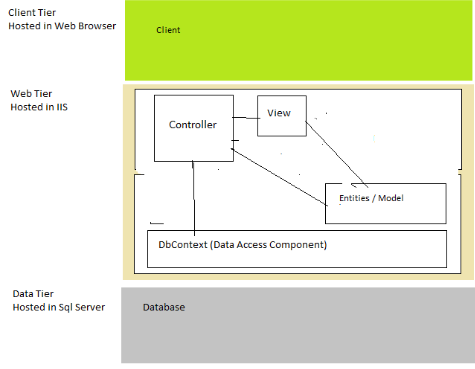
StudentSelectSubjects: This interface will provide the available subject list for the semester. This allows the student to choose the subjects, hence a busy interface. This interface is the bridge between the student component and the subject component.  
StudentProfile: This interface is used for the student to view and edit personnel information.

* **Management Component**

AuthenticateUser: This is the interface that allows the users to login to the system.

* **Subject Component**

NewSubject: This interface is where administrator adds new subjects to the courses offered. The Subject and student components are connected.  
EditSubject: In this interface the administrator edit existing subjects. The Subject and student components are connected.

* **Notifying Component**

Getnotice: This is the interface where the notifications are published. It is connected with the subject component.

**4.3) Architecture Design:**

The Student Registration System will be developed under two main architectural patterns. Development of the project will be done in MVC architectural style and also n-tier Client/Server Architecture.

* **MVC:**

MVC style separates presentation and interaction from the system data. The system is structured into  
three logical components that interact with each other.  
• The Model component :

Manages the system data and associated operations on that data.  
• The View component:

Defines and manages how the data is presented to the user.  
• The Controller component:

Manages user interaction and passes these interactions to the View and the Model.

The main advantage of this is style allows the data to change independently of its representation and vice versa.

* **Three-Tier Client Server Architecture:**

In a client server architecture, the functionality of the system is organized into services, with each service delivered from separate server. Clients are users of these services and access servers to make use of them.  
We will use this 3- Tier Client Server Architecture because, when data in a shared database has to be accessed from a range of locations.

• Data Tire  
The data tire maintains the applications data such as Users’ data , Departments’ data , subjects’ data , courses’ data , time tables’ data and the SQL queries . It stores these data in a relational database management system (RDBMS). All the connections with the RDBMS are managed in this tier.

• Middle Tire  
The middle tier (application server ) implements the business logic , controller logic and presentation logic to control the interaction between the applications’ clients and data. Business rules enforced by the business logic dictate how clients and cannot access application data and how applications process data.  
• Client Tire  
The client tire is the applications user interface connecting data entry forms and client side applications. It displays data to the user. User interact directly with the application through user interface. The client tier interacts with the web server to make requests and to retrieve data from the database. It then displays to the user the data retrieved from the server.

**Physical arrangement of devices in a typical network:**

Pc3

Pc2

Pc1

Wifi Router

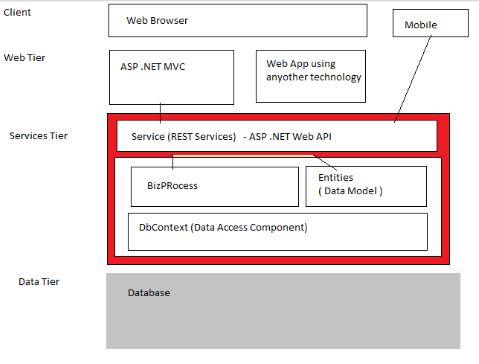
SERVER

**Design Decisions:**

* **MVC Architecture:**

Reasons:

• It should interact with other machines or users effectively.  
• For more efficient interaction system should have flexible interfaces.  
• MVC can be taken as for a popular and easy to handle web application development style that has the feature of separating the presentation, Business & intermediate logics.  
• Ease to coding and provide well defined interfaces within each logic.



* **Three Tier Architecture:**

Reasons:  
• As more users access the system a three-tier solution is more scalable than the other solutions because you can add as many middle tiers as needed to ensure good  
performance.  
• Security is also the best in the three-tier architecture because the middle layer protects the database tier.

POLICIES AND TACTICS:

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

Some of the main pointa 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
* There would be a library for the admin side
* We will not provide them the source code
* We will provide maintenace which will cost 10 percent of the whole system after every one year.
* There will be proper testing of the sytem and then it will be delivered to the client
* System installation will be free of cost and training of one week will be given
* Interfaces will be provided of three types. One for student second for teacher and third for the teacher end use.
* Prototyping would be done. And would be shown to customer first.