System Design Document (SDD)

Project: Student Information Management System

Purpose of the document

This document mainly discusses the system design aspect of our student information management system. Design goals will be provided in the introduction of the document to identify the qualities that our system will focus on. An overview of the current system architecture will be included for comparison with the proposed system architecture. Besides that, the proposed system architecture, its subsystem decomposition, hardware and software mapping, persistent data management, access control and security, global software control, and subsystem services will also be included in this document.

Ultimately, the goal of this system design document is to provide design specification of SIMS to facilitate our project implementation process.

About the System Design Document

Version Control

Version No.	Name (Title)	Author	Date	Authorization
1.0	SDD Report	Zhi Kang Wong Fan Wu Gabriel Alvarez	1/23/2019	Michael Barkdoll
2.0	SDD Report	Zhi Kang Wong Fan Wu Gabriel Alvarez	2/7/2019	Michael Barkdoll
3.0	SDD Report	Zhi Kang Wong Fan Wu Gabriel Alvarez	2/24/2019	

Document Properties

Items	**Details**
Document Title	ZFG System Design Document
Author	Zhi Kang Wong Fan Wu Gabriel Alvarez
Creation Date	1/23/2019
Last Updated	2/24/2019

Table of Contents

1. Introduction

1.1 Purpose of the system

The purpose of this system is to provide an all-in-one system that will handle student related information, schedule courses, and handle exit surveys whilst maintaining an user friendly interface. Users should be able to update the information given the correct authorization. Additional features include adding, deleting, sorting, searching, updating. Each user will have a role which consists of graduate program director, undergraduate program director, faculty, thesis advisor, project advisor, and student. Depending upon the role/authorization, each user will have access to certain features.

1.2 Design Goals

Performance:

In terms of response time, our Student Information Management System (SIMS) should be able to handle all requests from users such as logging in, searching and updating information, immediately. The throughput of the system will be high with the help of load balancing. The ideal case will be the system can accomplishes up to 100 tasks in a fixed period of time. Since it is a web application, not much memory will be required for the system to run.

Dependability:

The robustness of SIMS will be strong as it could survive invalid user input such as displaying notification or warning messages to users when their credentials are wrong during the login process. SIMS will also be reliable as it will deliver results as requested by users instead of producing unrelated output. The availability of this system will be 24/7 to allow users use it anytime. There will be implementation of encryption on SIMS database to protect user's private information from malicious attacks. The system will be safe and will not endanger human lives.

Cost:

The development cost and other cost for this system such as deployment, upgrade, maintenance, and administration cost will be covered by our stakeholders SIU CS department staff.

Maintenance:

SIMS will be extensible to allow the addition of new functionalities to the system besides the current features. The system will also be easy to modify as long as the developer(s) has the knowledge of AngularJS, ASP.NET Core, etc. In terms of portability, our system should be able to run on any platform such as computer and mobile devices with the presence of internet. Readability for this system will be maintained with the addition of comment in system code. Ultimately, SIMS should implement all functional requirements listed in the requirement analysis document to facilitate traceability of requirements.

End User Criteria:

The proposed SIMS contains functions that are requested by the stakeholders. To increase the user's utility and usability to our system, SIMS will be having user interface that are user friendly, reactive, and scalable.

1.3 Definitions, acronyms, abbreviations

- SIMS: Student Information Management System
- SIU: Southern Illinois University
- CS: Computer Science
- MVC: Model- View- Controller
- SDD: System Design Document
- RAD: Requirement Analysis Document
- DBMS: Database Management System
- API: Application Programming Interface

1.4 References

Before reading this system design document (SDD), it is better to read the prerequisite document, Requirement Analysis Document (RAD), which gives all the functional requirements and nonfunctional requirements of this system. For more details on our development tool, please refer to the online course titled "Build an app with ASPNET Core and Angular from scratch" by Neil Cummings from Udemy.com. The following table summarizes the references.

References	References' Location	Date
Requirement Analysis Document (RAD)	https://www2.cs.siu.edu/~sims/	1/20/2019
Build an app with ASPNET Core and Angular from scratch	https://www.udemy.com/build-an-app-with-aspnet-core-and-angular-from-scratch/learn/v4/content	11/16/2018

1.5 Overview

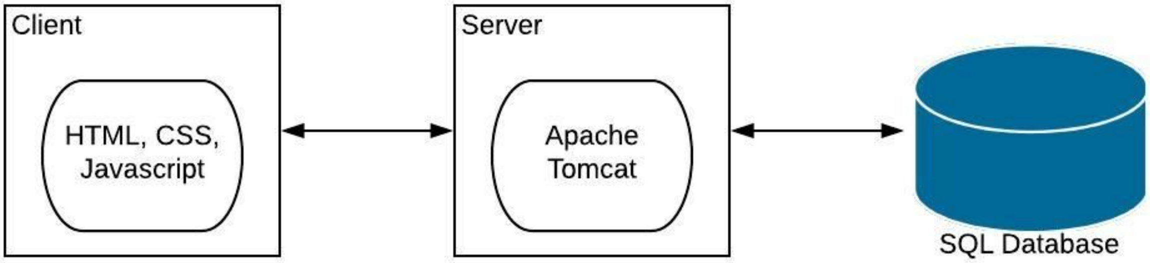
To help with design, this document was constructed to facilitate comprehension of the Student Information Management System (SIMS) and it's inner workings. Hopefully, this documents will aid development of the system and provide a well structured blueprint.

In this document we will break down the many aspects of the Student Information System. First we will delve into the current software architecture, to have a better understanding of the next topic which will focus on the proposed software architecture. We will decompose the system into subsystems, then we will look at hardware/software mapping, data management, access control and security, and global software control. Finally we will discuss subsystem services provided by SIMS.

2. Current Software Architecture

2.1 Overview

In our requirement analysis document, the exit survey system and course scheduling system are no longer being used due to the lack of maintenance with the absence of previous developer. For the existing student exit survey system, no framework is being used and simple user interface is created through HTML5, CSS, and Javascript. Apache Tomcat is used as a server and connects the frontend with the database.



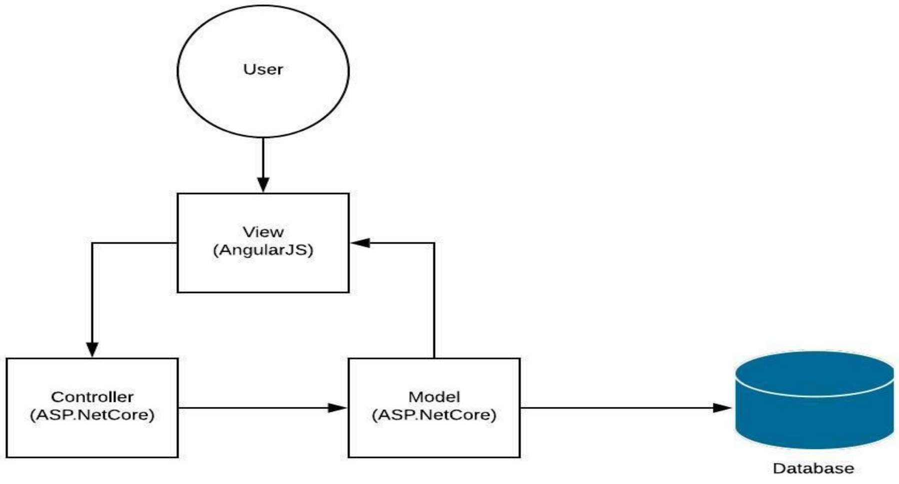
The architecture for the existing course scheduling system is unknown due to its design and source code unavailability.

3. Proposed Software Architecture

3.1 Overview

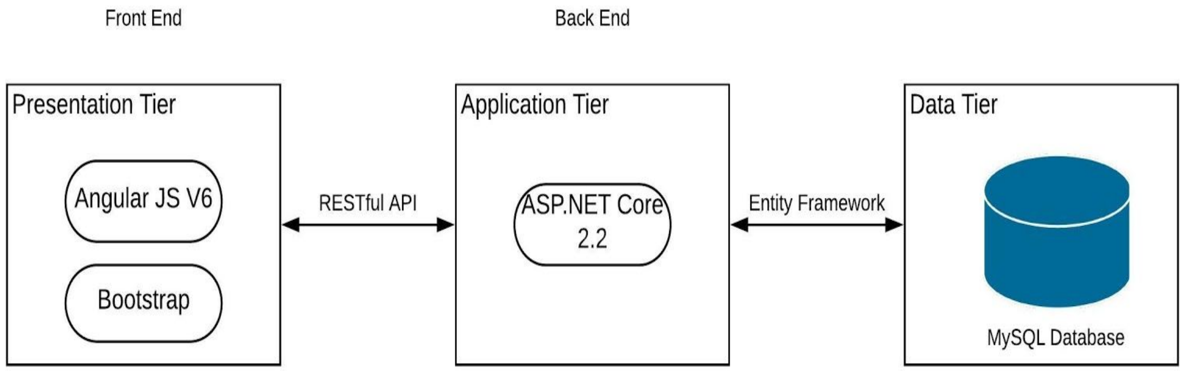
Our proposed system make use of frameworks and APIs such as AngularJS, ASP.Net Core, Entity Framework, RESTful API, and ASP. NET Web API. Since our system is a web application, we will be using MVC (Model View Controller) architecture. However, the view engine from ASP. NET Core (razor) will not be implemented, instead we will be using AngularJS for the client side templates (or views). The model layer is responsible for handling data and allows user retrieve, insert, and update information in database through controller. The view layer is considered as an interface that is being presented to the user. Data from model layers are displayed through this layer. The controller layer acts as an intermediary between model layer, view layer and other related resources to process the HTTP request and generate the web application.

Figure of MVC architecture:



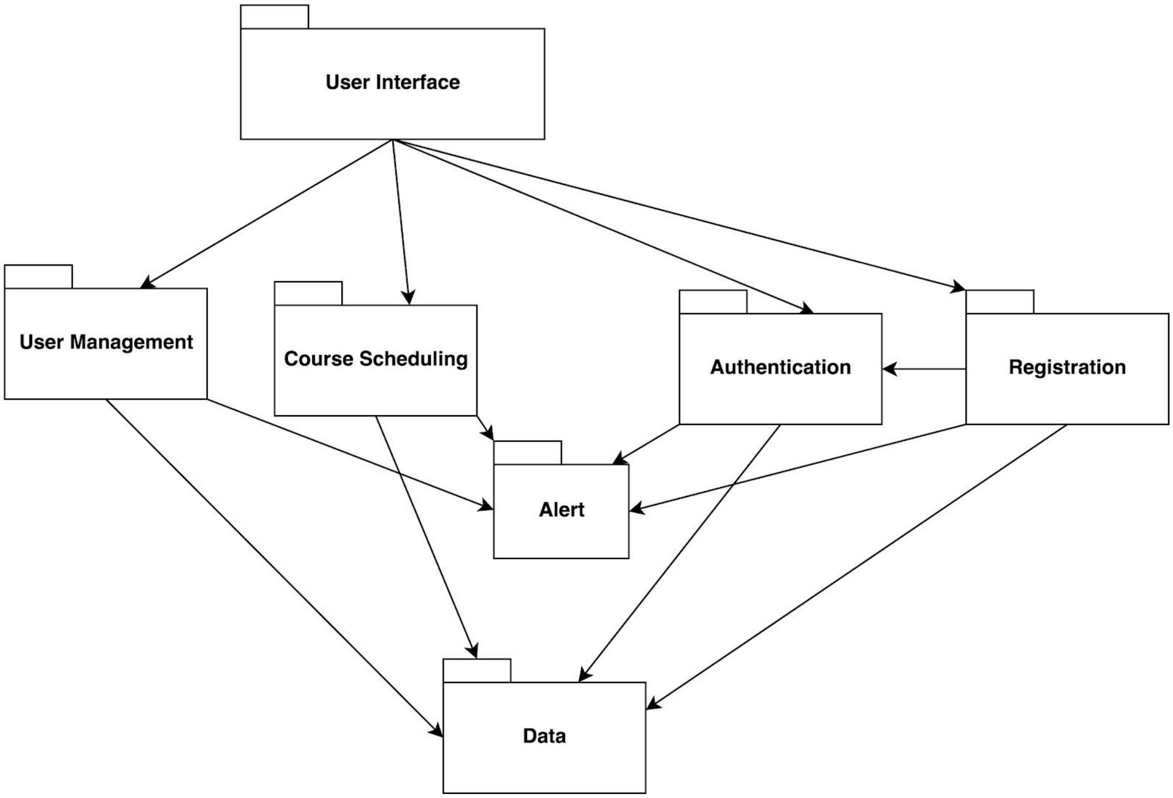
ZFG SIMS is a client-server system following the 3-tier architecture. The first tier is the presentation tier, which represents the user interface and the frontend of the system. This tier will be built on web technologies like Angular 6 JS, Bootstrap, etc and communicates with subsequent tier through API calls The purpose of this tier is to provides user interface for users and allow them to interact with the system. The application tier can be considered as the controller of the system as it contains the subsystems and its services. This tier will receive user actions from the presentation tier and processes them accordingly. The data access tier contains DBMS for data storage.

Figure of SIMS software architecture:



SIMS will be consisting of several subsystems that carry out different functionalities. The subsystems of SIMS include authentication, user interface, register, scheduling, user management, database, and alert.

3.2 Subsystem decomposition



Authentication

The main function of the Authentication subsystem is to authenticate user login before allowing access to the system. Authentication is needed in order to use the system. Requires services by the Data subsystem.

User Interface

The User Interface subsystem will provide each user with a predefined interface depending on their respective roles within the system. From here the users will have access to the system's functions. Authentication services are required in order to login and display the interface. Provides communication between the users the rest of the subsystems and required the services of the Registration, Authentication, Course Scheduling, User Management subsystem.

Register

The Register subsystem is responsible for checking credential and registering users. Uses the Data, Authentication, and Alert subsystems.

Course Scheduling

The Course scheduling subsystem will take care of the course scheduling functional requirement. It can be used to make the timetable of the courses by dragging and dropping, and create new courses tuple by entering relative information including course number, course name, course location and so on. In addition, after creating the schedule, it can be saved to the database or exported to pdf.

User Management

One of the biggest subsystems. It will be responsible for managing user related info. The subsystem will provide services such as editing profile information, adding/removing files, adding/removing pictures. Utilized the services of the Data subsystem and Alert subsystem.

Data

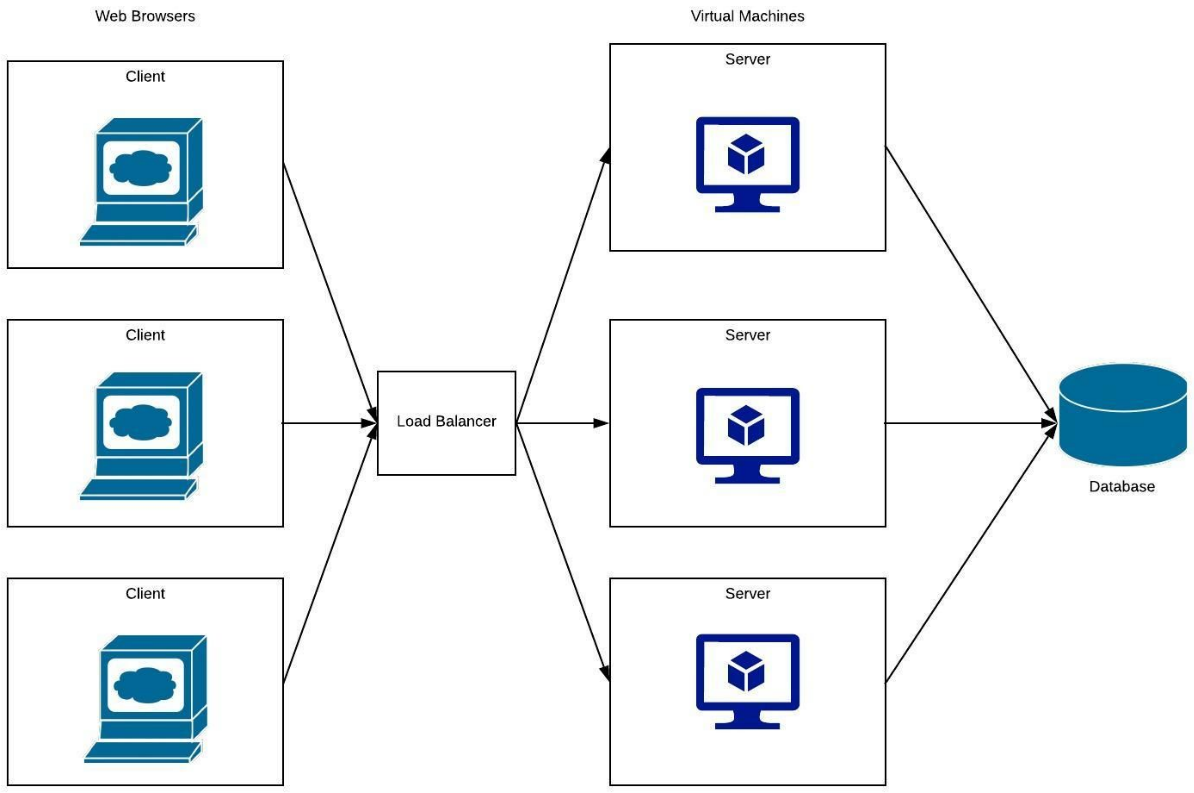
The Data subsystem is responsible for storage and retrieval of all persistent data in the system. Other subsystems are reliant on the Database subsystem service for completing queries.

Alert

The Alert subsystem provides other subsystem the means of displaying alerts to the user when something has gone wrong, missing items, invalid request, or they have done something illegal. Other subsystems utilize its services in order to produce feedback to the user.

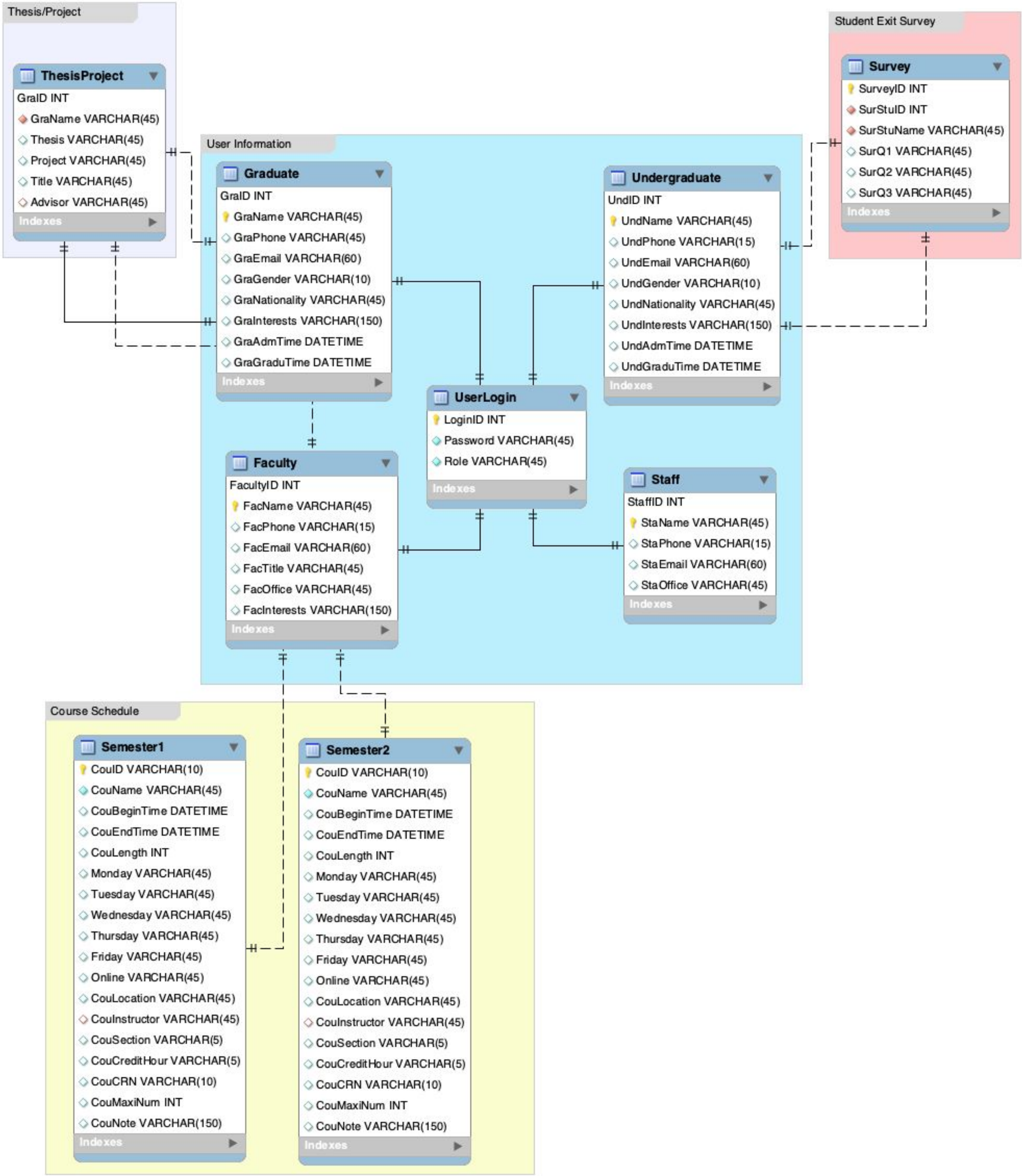
3.3 Hardware/ Software Mapping

The hardware configurations of SIMS are client machine and server machine with virtual machines. Each virtual machine in the server will have development tools and subsystems installed in it to handle client request. Load Balancer is used to distribute our web application traffic from client to respective servers, which increase the capacity and reliability of applications. Example of off-the-shelf components includes web browser Google Chrome, MySQL Database, Oracle VM VirtualBox, and Linux OS. Web browsers are to be deployed at the client side node, while Database server MySQL on the server side node.



3.4 Persistent Data Management

For persistent data management, a relational database created by MySQL will be used as the database management system (DBMS) of the Student Information Management System (SIMS). This database is used to store all data related to the SIMS, including user information data, course scheduling data, student exit survey data, and thesis/project data. SIMS shall implement some authentication mechanism for the requests of access to the database. The following figure shows the relational database schema of the SIMS.


Thesis/Project
ThesisProject
GraID INT
GraName VARCHAR(45)
Thesis VARCHAR(45)
Project VARCHAR(45)
Title VARCHAR(45)
Advisor VARCHAR(45)
Indexes
Student Exit Survey
Survey
SurveyID INT
SurStuID INT
SurStuName VARCHAR(45)
SurQ1 VARCHAR(45)
SurQ2 VARCHAR(45)
SurQ3 VARCHAR(45)
Indexes
User Information
Graduate
GraID INT
GraName VARCHAR(45)
GraPhone VARCHAR(45)
GraEmail VARCHAR(60)
GraGender VARCHAR(10)
GraNationality VARCHAR(45)
GraInterests VARCHAR(150)
GraAdmTime DATETIME
GraGraduTime DATETIME
Indexes
Undergraduate
UndID INT
UndName VARCHAR(45)
UndPhone VARCHAR(15)
UndEmail VARCHAR(60)
UndGender VARCHAR(10)
UndNationality VARCHAR(45)
UndInterests VARCHAR(150)
UndAdmTime DATETIME
UndGraduTime DATETIME
Indexes
UserLogin
LoginID INT
Password VARCHAR(45)
Role VARCHAR(45)
Indexes
Faculty
FacultyID INT
FacName VARCHAR(45)
FacPhone VARCHAR(15)
FacEmail VARCHAR(60)
FacTitle VARCHAR(45)
FacOffice VARCHAR(45)
FacInterests VARCHAR(150)
Indexes
Staff
StaffID INT
StaName VARCHAR(45)
StaPhone VARCHAR(15)
StaEmail VARCHAR(60)
StaOffice VARCHAR(45)
Indexes
Course Schedule
Semester1
CouID VARCHAR(10)
CouName VARCHAR(45)
CouBeginTime DATETIME
CouEndTime DATETIME
CouLength INT
Monday VARCHAR(45)
Tuesday VARCHAR(45)
Wednesday VARCHAR(45)
Thursday VARCHAR(45)
Friday VARCHAR(45)
Online VARCHAR(45)
CouLocation VARCHAR(45)
CouInstructor VARCHAR(45)
CouSection VARCHAR(5)
CouCreditHour VARCHAR(5)
CouCRN VARCHAR(10)
CouMaxiNum INT
CouNote VARCHAR(150)
Indexes
Semester2
CouID VARCHAR(10)
CouName VARCHAR(45)
CouBeginTime DATETIME
CouEndTime DATETIME
CouLength INT
Monday VARCHAR(45)
Tuesday VARCHAR(45)
Wednesday VARCHAR(45)
Thursday VARCHAR(45)
Friday VARCHAR(45)
Online VARCHAR(45)
CouLocation VARCHAR(45)
CouInstructor VARCHAR(45)
CouSection VARCHAR(5)
CouCreditHour VARCHAR(5)
CouCRN VARCHAR(10)
CouMaxiNum INT
CouNote VARCHAR(150)
Indexes

3.5 Access Control and Security

Access control and security will be enforced with required logins, JSON tokens, and data encryption.

Actor	Privileges
Admin	● Register users ● Delete users ● Assign roles ● Manage student information ● Manage exit surveys ● Update students' thesis/projects ● Schedule courses ● Search students ● Edit profile
Staff	● Delete users ● Assign roles to user ● Manage student information ● Manage exit surveys ● Update students' thesis/projects ● Schedule courses ● Search students ● Edit profile
Thesis/Project Advisor	● Update students' thesis/project ● Search and sort students ● Edit profile
Faculty	● Manage exit surveys ● Search and sort students ● Edit profile
Student	● Manage exit surveys ● Edit profile

Required Login

In order for an user to gain access and utilize the system they must first login to the system using valid credentials

JSON Tokens

After successful login, the system will generate a token for the user that will be used to indicate the identity of the current user, his/her roles, and authorizations.

Data Encryption

Delicate data will be encrypted using SHA512 hashing.

3.6 Global Software Control

The control flow of SIMS is event-driven. According to the Requirement analysis document (RAD), SIMS should make sure many users can interact at a time and should not affect data consistency. This can be implemented by the event handlers. The event handler can monitor the movement of the user like onclick( ), onkeypress( ), and so on to achieve the interaction with the user. In other words, different users can trigger different events at the same time. The application responds to events in the order they arrive. Each time the user interacts with a component, an event is sent to the specific subsystem. Different events are sent to different subsystems. One subsystem can work correctly based on the order of the events, and it ignores events that are not relevant to its purpose.

4. Subsystem Services

User Interface

The User Interface will be the bridge between users and the services of the subsystems.

Authentication

In order for a user to utilize the system, first they must prove that they have access to the system through the Authentication subsystem. The Authentication subsystem takes in two inputs, username and password. It must first determine if the username entered does in fact exist within the database by using the Data subsystem. If the user does not exist then the subsystem will call forth the Alert subsystem to let the user know that the credential were incorrect, otherwise the password entered will be go through a verification process. If the password entered is incorrect, the subsystem will utilize the Alert subsystem to let the user know that the input credentials were incorrect, otherwise, a JSON token will be generated and stored in local storage for further authorization checks.

Subsystem services:

- Login()
- VerifyPasswordHash()
- UserExists()
- LoggedIn()

Register

The Register subsystem creates an entry into the database of a new user. Input includes typical user info such username, password, siu id, name, major, etc. The first step to registration is determining if the entered username is already in use; this is done by calling the UserExist() services from Authorization subsystem. If the username is already in use then the subsystem will use the Alert subsystem to notify the user of such error, otherwise a password hash will be generated using the input password and a password salt. Finally, all the input information is saved in the database using the Data subsystem.

Subsystem Services:

- Register()
- CreatePasswordHash()

Course Scheduling

The course scheduling subsystem can be used to make a course schedule in an intuitive way. The services provided by this subsystem including creating new courses tuples, update courses information, delete courses, and set time of the courses. The courses can be shown in a timetable if the time of the courses has been set. In addition, after creating the timetable, it can be exported to PDF. This subsystem will be encapsulated as a complete API which can connect to the application.

Subsystem Services:

- CreateCourses()
- CourseSchedule()
- ExportToPDF()

User Management

The function of the User Management system is to manage the information of the users within the system. This will allow Staff, Faculty, Admins to make changes to their profiles and to others profiles. Students will be able to make changes to their own profiles only. Aside, from managing information, the system will allow the retrieval of profiles, search of other profiles by certain fields and the ability to sort the results as well. This subsystem will use the services of the Data subsystem in order to query for the users.

Subsystem Services:

- SubmitChanges()
- GetUsers()
- GetUser()
- SearchBy()
- SortBy()
- Add()
- Delete()

Data

The User Management, Register, Course Scheduling, and Authentication subsystem all rely on the Data subsystem to complete database queries and changes.

Subsystem services:

- Add()
- Delete()
- Get()
- Save()

Alert

The User Management, Register, Course Scheduling, and Authentication subsystem all use the Alert subsystem as a means of notifying the user of errors, warnings, messages, successes, and to prompt for confirmations.

Subsystem services:

- Confirm()
- Success()
- Error()
- Warning()
- Message()