Project Topic

Mana Staff Management System

Abstract

D.S. Senanyake College is one of many institutes in Sri Lanka that keeps track of an incredible amount of data. The college currently operates on a paper based system where information about staff and students is stored manually. This system has proven to cause a lot of problems such as inefficiency in performance, loss of data integrity due to human error etc. The proposed system is a web-based system that will be used by the staff to manage information. This includes and is not limited to student details, student marks and grades, teachers’ timetable details, attendance marking and leave management. An event management module will also be created for staff members to handle school events.

Developing the system must be done with caution as most of the intended users are not computer savvy and the UI has to be simple and intuitive enough for inexperienced users. Security is also a prime concern as the system will be deployed on the internet and therefore needs adequate protection from any known threats.

From this project, we hope to overcome the difficulties faced in the manual system and serve as a suitable replacement for the current system.

Declaration

We declare that the project work entitled “**Mana Staff Management System**” for “**D.S. Senanyake College**” was carried out by us. This project was undertaken as part of the academic curriculum (2nd year 2nd semester ITP module) and was done according to the Sri Lanka Institute of Information Technology rules and norms. This is our original work.

Project Details

|  |  |
| --- | --- |
| Project Title | Mana Staff Management System |
| Project ID | ITP-14-B1-06 |

Group Members

|  |  |  |
| --- | --- | --- |
| **Reg. No** | **Name** | **Signature** |
| IT13006426 | Mendis B.T.M. |  |
| IT13001304 | Alimudeen M.Y. |  |
| IT13008338 | Peiries M.S.E |  |
| IT13013424 | Liyanage M.C. |  |
| IT13031312 | Joseph V.S. |  |
| IT13014650 | Rathnayake R.M.D.A. |  |
| IT13028206 | Jayakody J.A.I |  |
| IT13024000 | De Silva G.L.N.A.M. |  |

Acknowledgement

We would like to acknowledge the support of the Old Boys Association of D.S. Senanyake College who provided us with the resources we need when it came to data on the school.

We are also grateful to our Project Supervisor Mr. Prasanna S. Haddela for his constant support in our project and for giving us his valuable insight and opinions. Mr. Haddela was instrumental in the development phase and this project would have been entirely different without him.

We would also like to thank Ms. Laksri Wijerathna and Ms. Geethanjali Wimalaratne for guiding us during lectures and also for providing the required material to do our project.

Table of Contents

Abstract i

Acknowledgement ii

Declaration ii

Table of Contents iv

List of Figures v

List of Tables v

List of Acronyms and Abbreviations Error! Bookmark not defined.

1. Introduction 1

1.1 Problem Statement 1

1.2 Product Scope 1

1.3 Project Report Structure 2

2. Methodology 3

2.1 Requirements and Analysis 3

2.2 Design **Error! Bookmark not defined.**

2.3 Implementation 45

2.4 Testing 47

3. Evaluation 49

3.1 Assessment of the Project results 49

3.2 Lessons Learned 49

3.3 Future Work 50

4. Conclusion 50

5. References 51

Appendix A: Design Diagrams 52

Appendix B: Test Results 53

Appendix C: Selected Code Listings 54

List of Figures

|  |  |  |
| --- | --- | --- |
| Figure Number | Name | Page Number |
| 1.0 | Function Interaction Diagram |  |
| Use Case Diagrams | | |
| 2.0 | Staff Registration |  |
| 2.1 | Leave Management |  |
| 2.2 | Timetable |  |
| 2.3 | Event Management |  |
| 2.4 | Student Information Management |  |
| 2.5 | Attendance |  |
| 2.6 | Marks and Grading |  |
| 2.7 | Security |  |

List of Tables

|  |  |  |
| --- | --- | --- |
| Table Number | Name | Page Number |
| Staff Registration Use Case Scenarios | | |
| 1.0 | Staff Registration |  |
| 1.1 | Fill Teachers Achievement Form |  |
| 1.2 | View Staff Details |  |
| 1.3 | Search Staff Details |  |
| 1.4 | Add Staff Member to Black List |  |
| 1.5 | View Black List member details |  |
| 1.6 | Generate Report |  |
| 1.7 | Update Staff Details |  |
| Leave Management Use Case Scenarios | | |
| 2.0 | Apply for Leave |  |
| 2.1 | Approve Leave |  |
| 2.2 | Generate Leave Report |  |
| Time Table Use Case Scenarios | | |
| 3.0 | View Timetable |  |
| 3.1 | Create timetable by teacher |  |
| 3.2 | Substitute Teacher by teacher |  |
| 3.3 | Substitute Teacher by class |  |
| Event Management Use Case Scenarios | | |
| 4.0 | View Event List |  |
| 4.1 | Add Event |  |
| 4.2 | Edit Event |  |
| 4.3 | Add Transaction |  |
| 4.4 | Print Transaction Report |  |
| Student Information Management System Use Case Scenarios | | |
| 5.0 | Register Student |  |
| 5.1 | Update Student Details |  |
| 5.2 | Search Student |  |
| 5.3 | Search class |  |
| 5.4 | Generate Report |  |
| Attendance System Use Case Scenarios | | |
| 6.0 | Mark Attendance |  |
| 6.1 | View Attendance |  |
| 6.2 | View year plan |  |
| Marks and Grading System Use Case Scenarios | | |
| 7.0 | Insert Term Test Marks |  |
| 7.1 | Insert Ordinary Level Marks |  |
| 7.2 | Insert Advanced Level Marks |  |
| 7.3 | Search Results |  |
| 7.4 | Generate Report |  |
| Security System Use Case Scenarios | | |
| 8.0 | Login |  |
| 8.1 | Change Password |  |
| 8.2 | Request Password Reset |  |
| 8.3 | Register User |  |
| 8.4 | Delete User |  |
| 8.5 | Reset User Password |  |
| 8.6 | Backup Database |  |

# Introduction

## Problem Statement

The system to be developed will be based on the current manual system they have in place. The reports generated will mimic reports and other data files already in use, so as to make the transition to the new system easier.

The Ministry of Education (under which the school operates) requires several reports from the school as it is the governing body. When producing these reports, they have to be made as similar to the existing ones as possible while still enhancing their current production model, e.g. Generating reports faster than they are being produced currently.

## Product Scope

Mana is a staff management system to be developed for D.S Senanayake College, Colombo. During the requirement gathering phase it was understood that at present the school is finding it hard to keep track of the data manually. All changes to the system have to be worked out manually. The intended system will focus on the staff information management, but will still deal with student information management, marks and grading, event planning, student attendance.

With a workforce of more than 300, it is highly inefficient and not to mention time consuming to store and manage data manually, not to mention the several thousand students.

Developing the system must be done with caution as most of the intended users are not computer savvy. Any user interface has to be simple and intuitive enough for inexperienced users. Security is also a prime concern as the system will be deployed on the internet and therefore needs adequate protection from any known threats.

**Key Benefits of the proposed system**

* All the staff, student, attendance and grading information will be available to anyone authorized to view this information.
* The automated system can provide access levels to different users to restrict access to material that they are not authorized to.
* The various detailed reports generated by this system will provide useful information to the teachers using this system.
* As it is web-based, the system can be accessed anywhere, at any given time without restrictions on platform either.
* Single centralized storage of information.
* Easy record keeping.

**Objectives**

* Provide a system the client can use to store and manage their staff members’ information.
* Fast and efficient retrieval of data of staff members.
* Ensure no data is lost to system errors by making multiple backups.
* Reduce the data entry time to a minimum so that users don’t have to waste time.
* Maintain a single copy of the information for many users to work on thereby eliminating data duplication and maintaining the integrity of data.
* Eliminating human error by having form validations therefore data accuracy is maintained.
* Provide descriptive statistical reports to the users for reviewing purposes.

**Goals**

* Provide an easy user experience with simple functions to introduce users to computers.
* Eliminate the limitations that are present in the manual system.
* Provide a system that rapidly processes data and information is generated quickly.

## Project Report Structure

In this report, the contents under “*Methodology*” gives the requirements specification and analysis of the Mana Staff Management System. It also contains Use Case diagrams and scenarios specifying how the system operates. In addition it contains various non-functional requirements, required for the system.

The “*Design*” section contains the Entity-Relationship Diagram and Activity Diagrams that help visualize the software solution.

Moreover in the “*Implementation*” section describes how the Mana Staff Management System is implemented and structured.

# Methodology

## Requirements and Analysis

We obtained several reports generated by the school and it helped us greatly to get a clear idea of how the current manual system operated. Furthermore meetings were held once every two weeks with our client. In the first few meetings we carried out question and answer sessions to obtain the requirements set. After we had an idea of the system, we developed dummy interfaces and obtained our clients feedback and understood how to improve our system.

### Product Functions

* Staff Management
* Leave Management
* Timetables
* Student Information Management
* Marks and Grading
* Attendance
* Event Management
* User and Access Management

The following is a diagram of how the mentioned function will interact with each other and constitute the system as a whole.

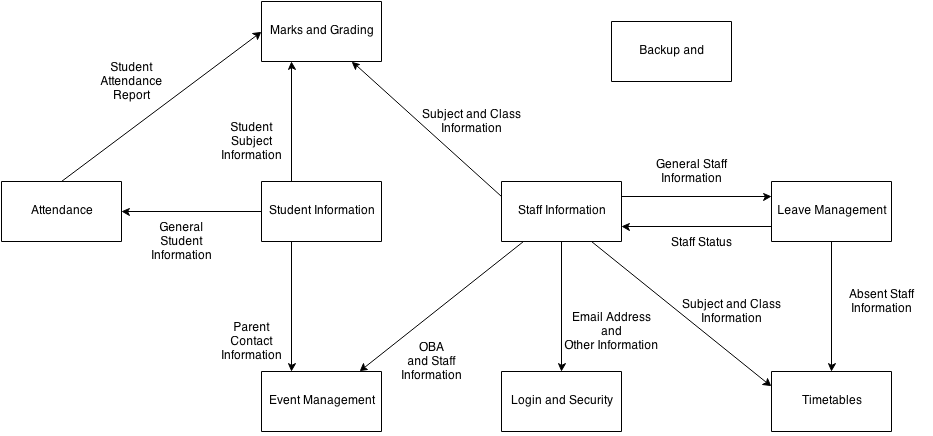
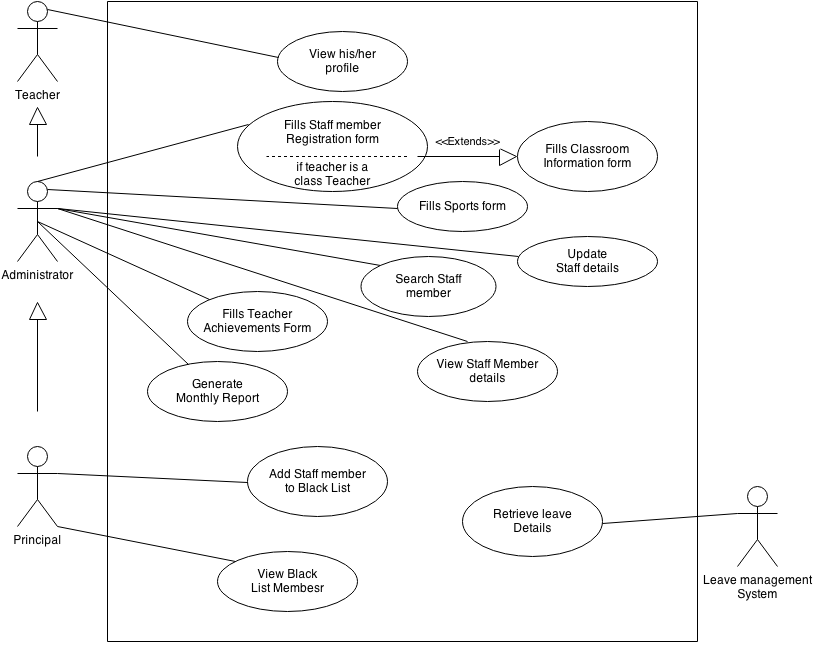


Figure 1.0

### Use Case Diagrams

##### **Staff Registration System**



Staff Registration & Management

##### 

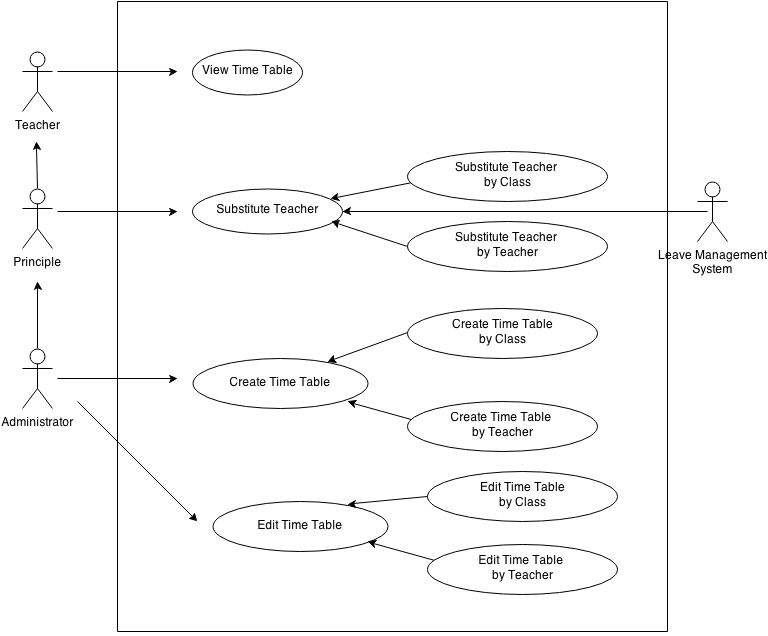
Figure 2.0

##### **Leave Management System**



Figure 2.1

**Time Table System**



Time Table

Figure 2.2

**Event Management System**

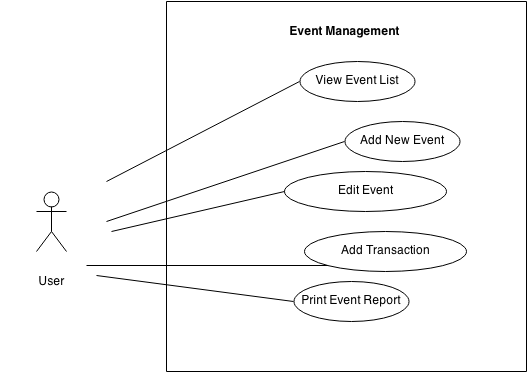


Figure 2.3

**Student Information Management System**

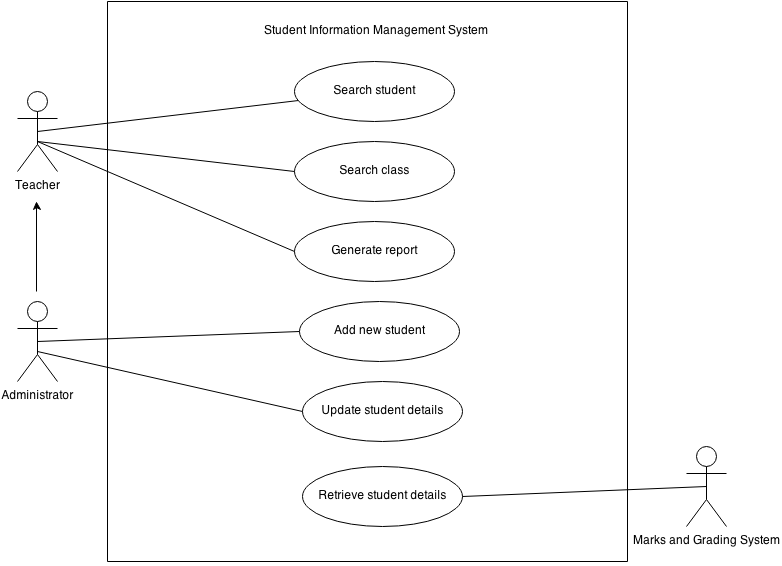


Figure 2.4

**Attendance System**

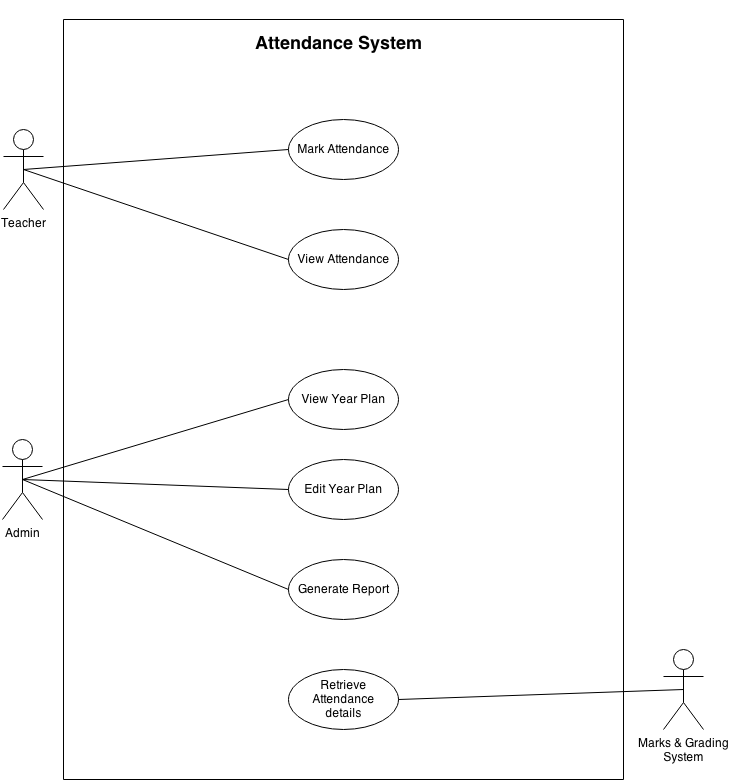


Figure 2.5

**Marks and Grading System**

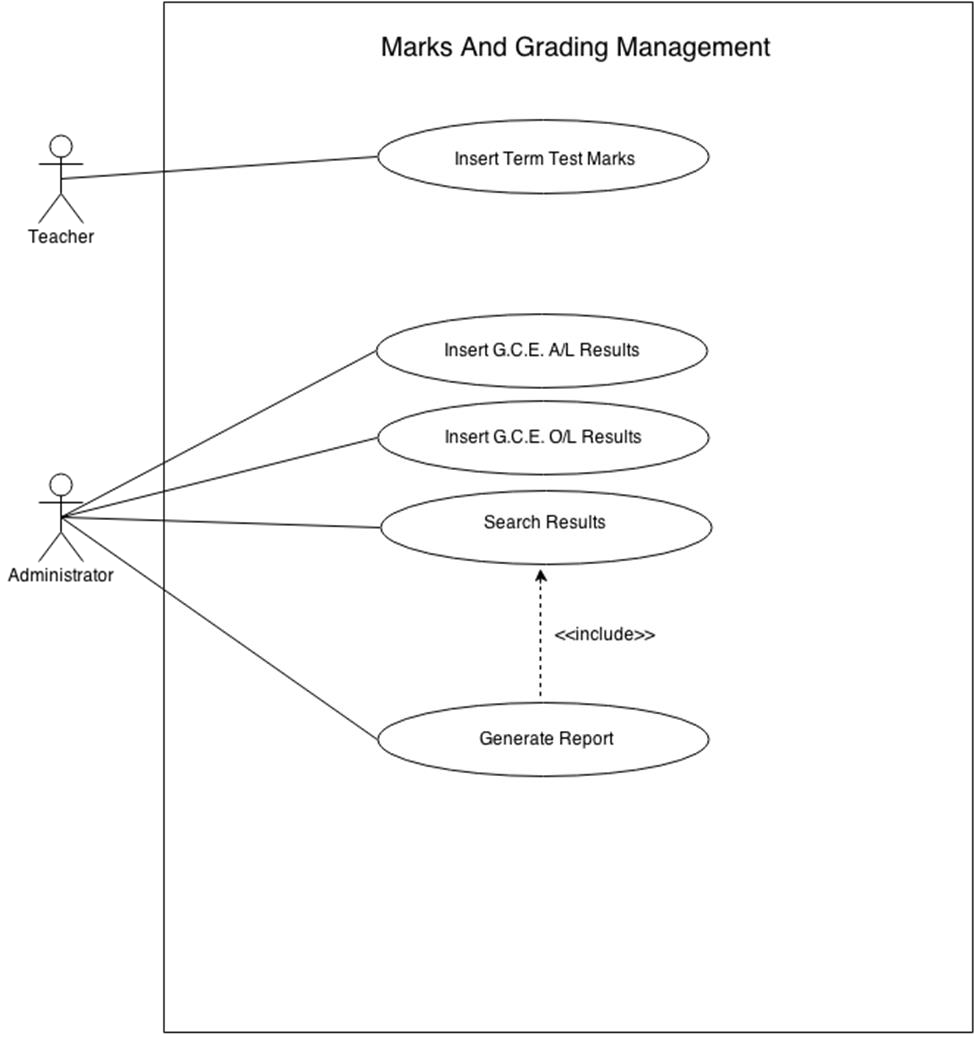


Figure 2.6

**Security System**

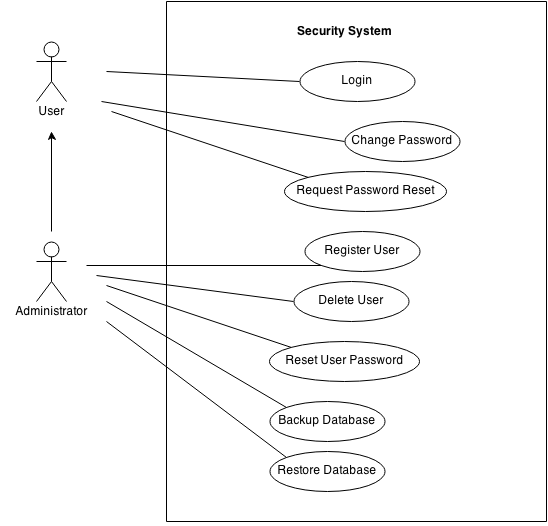


Figure 2.7

­

### Use Case Scenarios

**Staff Registration and Management System**

|  |  |
| --- | --- |
| Use Case  Name | Fill Staff Member Registration Form |
| Primary Actor | Administrator |
| Main Flow | 1). Login to the System  2). Administrator selects the Register Staff Member tab in the website.  3). Website displays the register interface with Staff registration form.  4). Administrator enters Staff member details and submit.  5). System will display a successful message and use case ends. |
| Extensions | 4) a) If Administrator provides wrong information system will displays an error  Message.  b) If staff member is a Class teacher needs to fill classroom Information form. |

Table 1.0

|  |  |
| --- | --- |
| Use Case  Name | Fills Teacher Achievement Form |
| Primary Actor | Administrator |
| Pre-condition | Should have filled the staff member registration form and submit |
| Main Flow | 1) Login to the system.  2). Administrator selects the Teacher Achievements tab.  3). Website displays the Teacher Achievements interface.  4). Administrator enters details and clicks submit. |
| Extensions | 4) a) If Administrator didn’t fill required fields system will displays an error Message. |

Table 1.1

|  |  |
| --- | --- |
| Use Case  Name | View Staff Details |
| Primary Actors | Administrator, principal |
| Main Flow | 1. Login to the system. 2. Click the “Staff management”. 3. Click the “View staff details” tab. 4. System prompts the Assign interface. 5) View the staff member Details |

Table 1.2

|  |  |
| --- | --- |
| Use Case  Name | Search Staff Details. |
| Primary Actors | Administrator. Principal |
| Main Flow | 1. Login to the system. 2. Click the “Search Staff member”. 3. System prompts the Assign interface. 4. User can search Staff member by select the required field. 5. System views search results.   6)If he wants he can generate a report |
| Extensions | 5) a .if user didn’t select any required Field it will display all the details of staff members |

Table 1.3

|  |  |
| --- | --- |
| Use Case  Name | Add Staff Member to Black List. |
| Primary Actor | Principal |
| Main Flow | 1. Login to the system. 2. Click the “Add Staff member to Black List”. 3) System prompts the Assign interface. 3. Principal enter staff member details and clicks submit. 4. System display successful message. |
| Extensions | 5) a) If principal didn’t fill required field system will displays an error Message |

Table 1.4

|  |  |
| --- | --- |
| Use Case  Name | View Black List member details. |
| Primary Actor | Principal |
| Main Flow | 1. Login to the system. 2. Click the “View Black List”. 3. System prompts the Assign interface   4)If principal needs he /she can generate a report |

Table 1.5

|  |  |
| --- | --- |
| Use Case  Name | Generate Monthly Staff Report |
| Primary Actor | Administrator |
| Secondary Actor | Leave management System |
| Pre-condition | Should have filled the staff member registration form and submit |
| Main Flow | 1. Login to the system. 2. User selects the view staff Details form tab in the website. 3) Select view staff details with leave details 4) Clicks print button. |

Table 1.6

|  |  |
| --- | --- |
| Use Case  Name | Update Staff member Details |
| Primary Actor | Administrator |
| Pre-condition | Should have filled the staff member registration form and submit |
| Main Flow | 1) Login to the system.  2). Administrator selects the Assigned tab  3). Website displays the related interface.  4). Administrator update and submit |
| Extensions | 4) a) If Administrator didn’t fill required fields system will displays an error message |

Table 1.7

**Leave Management System**

|  |  |
| --- | --- |
| Use Case  Name | Apply for Leave |
| Primary Actor | Clerk |
| Pre-Conditions | Clerk has authorized login |
| Main Flow | 1. Clerk enters the Staff ID 2. Staff ID is validated and checked if exists 3. Clerk enters the start date 4. Clerk enters the end date 5. Clerk enters the number of leave days 6. Clerk picks the leave type 7. System generates the number of leave days left 8. Clerk clicks apply for leave after finished 9. System logs the leave request and sends for approval |
| Extensions | 2 a. If Invalid Staff ID is entered, display error message.  2 b. Go back to step 1.     1. a1. If Staff member doesn’t exist, display error message. 2 b1. Go back to step 1.      1. a. Invalid dates entered, display error message.   3 b1. Repeat step.    7a. Number of leave days over, display message. |

Table 2.0

|  |  |
| --- | --- |
| Use Case  Name | Approve Leave |
| Primary Actor | Principal |
| Pre-Conditions | Principal has an authorized login |
| Main Flow | 1. Principal selects the approve leave from main menu 2. Principal selects leave request from the list 3. Principal expands details 4. Principal selects approves/rejects leave request 5. System updates staff leave days left |
| Extensions | 2 a. No leave requests, display message saying no leave requests. |

Table 2.1

|  |  |
| --- | --- |
| Use Case  Name | Generate Leave Report |
| Primary Actor | Principal |
| Pre-Conditions | Principal has an authorized login |
| Main Flow | 1. Principal selects leave report from the main menu 2. Principal see generate leave report form 3. Principal selects what sort of leave report it should be 4. Principal clicks generate report 5. System generates report 6. Principal can see the generated report |

Table 2.2

**Timetable**

|  |  |
| --- | --- |
| Use Case Name | View Timetable |
| Primary Actor | Principle, Administrator |
| Main Flow | 1. Login 2. User will redirect to the Main Interface 3. User Click View Time Table Tab 4. User will Redirect to Time Table Interface 5. Then User Select View time table tab 6. User Enters Teacher’s Name or Class 7. Relevant Time Table will Display |

Table 3.0

|  |  |
| --- | --- |
| Use Case Name | Create timetable by Teacher |
| Primary Actors | Administrator |
| Main flow | 1. User Log in to the System 2. User will Redirect to the Main Interface 3. User Clicks Time Table Tab 4. User will Redirect to Time Table Interface 5. Then User Select Create time table tab 6. User Tick ‘ by teacher ’ Combo Box 7. Text box will appear to enter Teacher’s name 8. User enters the teacher’s name 9. User Feed the time table 10. User press save 11. System prompts “Time Table Successfully Save” |

Table 3.1

|  |  |
| --- | --- |
| Use Case Name | Substitute Teacher by Teacher |
| Primary Actors | Administrator , Principle |
| Main flow | 1. User Log in to the System 2. User will Redirect to the Main Interface 3. User Clicks Substitute Teacher Tab 4. User will Redirect to Substitute Teacher Interface 5. User Ticks ‘ by teacher ’ Combo Box 6. Text box will prompt to enter Teacher’s name 7. User enters the teacher’s name 8. System Prompts relevant teacher’s time Table 9. User Selects relevant(concerned) period 10. Available Teachers list will appear with their contact details 11. User selects one of the available Teacher 12. System sends email and SMS for selected teacher |

Table 3.2

|  |  |
| --- | --- |
| Use Case Name | Substitute Teacher by class |
| Primary Actors | Administrator , Principle |
| Main flow | 1. User Log in to the System 2. User will Redirect to the Main Interface 3. User Clicks Substitute Teacher Tab 4. User will Redirect to Substitute Teacher Interface 5. User Ticks ‘ by class ’ Combo Box 6. Text box will prompt to enter class 7. User enters the class 8. System Prompts relevant class Table 9. User Selects relevant(concerned) period 10. Available Teachers list will appear with their contact details 11. User selects one of the available Teacher 12. System sends email and SMS for selected teacher |

Table 3.3

**Event Management System**

|  |  |
| --- | --- |
| Use Case Name | View Event List |
| Primary Actor | Principal |
| Pre-Condition | Should have valid login |
| Main Flow | 1.Login  2.Click Event Management in the Main Interface |

Table 4.0

|  |  |
| --- | --- |
| Use Case Name | Add New Event |
| Primary Actor | Principal |
| Pre-Condition | Should have valid login |
| Main Flow | 1.Click Event Management in the Main menu  2.Click Add New Event button in the Event List interface  3.Fill the Event details  4.Click Save Event |

Table 4.1

|  |  |
| --- | --- |
| Use Case Name | Edit Event |
| Primary Actor | Principal, Event Manager |
| Pre-Condition | Should have valid login |
| Main Flow | 1.Login  2.Click Event Management in the Main Interface  3.Click Manage button next to the related event  4.Edit required fields  5.Click Save Event |

Table 4.2

|  |  |
| --- | --- |
| Use Case Name | Add Transaction |
| Primary Actor | Event Manager |
| Pre-Condition | Event is already created |
| Main Flow | 1.Login  2.Click manage button in the related event  3.Fill the transaction details |

Table 4.3

|  |  |
| --- | --- |
| Use Case Name | Print Transaction Report |
| Primary Actor | Event Manager |
| Pre-Condition | Transactions were fed to the system |
| Main Flow | 1.Login  2.Click manage button in the related event  3.Click Print Transaction Button |

Table 4.4

**Student Information Management System**

|  |  |
| --- | --- |
| Use Case  Name | Add new student |
| Primary Actor | Administrator |
| Pre-Conditions | Admin has authorized login |
| Main Flow | 1. Selects the add new student from menu 2. Enters the student details   3.Clicks submit button  4.Receives a successful message from the system |

Table 5.0

|  |  |
| --- | --- |
| Use Case  Name | Update student Details |
| Primary Actor | Administrator |
| Pre-Conditions | Admin has authorized login |
| Main Flow | 1. Selects the update student details 2. Update the necessary student details 3. Clicks submit button 4. Receives a successful message from the system |

Table 5.1

|  |  |
| --- | --- |
| Use Case  Name | Search Student |
| Primary Actor | Administrator and Teacher |
| Main Flow | 1. Selects the search student 2. Fill in the student admission number 3. Clicks the search button 4. Can view the searched results of the student |
| Extensions | 2.a : If the admission number is wrong, user will receive an error message from the system |

Table 5.2

|  |  |
| --- | --- |
| Use Case  Name | Search Class |
| Primary Actor | Administrator and Teacher |
| Main Flow | 1.Selects the search class  2.Fill in the grade/year and the class  3.Clicks the search button  4.Can view the searched results of the student |
| Extensions | 2.a : If the grade and the class is wrong, user will receive an error message from the system |

Table 5.3

|  |  |
| --- | --- |
| Use Case  Name | Generate Report |
| Primary Actor | Administrator |
| Main Flow | 1. User selects report from the main menu 2. User see generate report form 3. User selects what sort of report it should be 4. User clicks generate report 5. System generates report 6. User can see the generated report |

Table 5.4

##### **Attendance System**

|  |  |
| --- | --- |
| Use Case  Name | Mark attendance. |
| Primary Actor | Teacher |
| Precondition | Teacher should have log in to the system. |
| Main Flow | 1. Log in to the system. 2. Select keep attendance. 3. Mark attendance. 4. Save. |

##### Table 6.0

|  |  |
| --- | --- |
| Use Case  Name | View Attendance. |
| Primary Actor | Teacher |
| Precondition | Select class, grade and date. |
| Main Flow | 1. Teacher select view attendance from attendance. 2. Teacher enter student details and get attendance. |

Table 6.1

|  |  |
| --- | --- |
| Use Case  Name | View year plan. |
| Primary Actor | Administrator |
| Precondition | Select year plan then view. |
| Main Flow | 1. Log in to the system. 2. Select the year plan. 3. View year plan. |
| Use Case  Name | Edit year plan. |
| Primary Actor | Administrator |
| Precondition | Go attendance interface and select year plan. |
| Main Flow | 1. Log in to the system. 2. Select year plan. 3. Edit year plan. 4. Save. |

Table 6.2

**Marks and Grading System**

|  |  |
| --- | --- |
| Use Case  Name | Insert Term Test Marks |
| Primary Actor | Teacher |
| Main Flow | 1. Select Insert term test marks from Menu. 2. Teacher enters class details and term details and submits. 3. Teacher get student list of preferred class. 4. Teacher enters subject marks and remarks and submits. 5. Teacher gets success massage from the system. |
| Extensions | 2 a. If Teacher leave combo boxes empty an error massage will display. |

Table 7.0

|  |  |
| --- | --- |
| Use Case  Name | Insert G.C.E. O/L results |
| Primary Actor | Administrator |
| Main Flow | 1. Select Insert G.C.E O/L results from Menu 2. Administrator enters index number and submit 3. Administrator gets Subject list along with the name, index number and year 4. Administrator enters G.C.E. O/L results 5. Administrator gets the success massage from the system |
| Extensions | 2 a .Administrator gets an error massage if he enters an invalid index number |

Table 7.1

|  |  |
| --- | --- |
| Use Case  Name | Insert G.C.E. A/L results |
| Primary Actor | Administrator |
| Main Flow | 1. Select Insert G.C.E A/L results from Menu 2. Administrator enters index number and submit 3. Administrator gets Subject list along with the name, index number and year 4. Administrator enters G.C.E. A/L results 5. Administrator gets the success massage from the system |
| Extensions | 2 a .Administrator gets an error massage if he enters an invalid index number |

Table 7.2

|  |  |
| --- | --- |
| Use Case  Name | Search Results |
| Primary Actor | Administrator |
| Main Flow | 1. Administrator select Search results from main menu. 2. Administrator selects what sort of result it should be. 3. Administrator clicks search. 4. Administrator can see the results 5. If he wants he can generate a report |
| Extensions | 2 a. If details are incorrect, an error message will be displayed. |

Table 7.3

|  |  |
| --- | --- |
| Use Case  Name | Generate Report |
| Primary Actor | Administrator |
| Main Flow | 1. Select report item from the menu 2. Administrator sees the generate report form 3. Administrator selects what sort of report it should be. 4. Click generate report 5. Administrator can see generated report 6. Prints the report |

Table 7.4

##### **Security System**

|  |  |
| --- | --- |
| Use Case  Name | Login |
| Primary Actor | All Users |
| Preconditions | User has login credentials provided to them |
| Main Flow | 1. Enter username and password 2. Submit |
| Extensions | 2 a. If username/password combination is incorrect, log the attempt and allow a reattempt. |

Table 8.0

|  |  |
| --- | --- |
| Use Case  Name | Change Password |
| Primary Actor | All Users |
| Preconditions | User is logged in |
| Main Flow | 1. Enter new password and confirm 2. Submit |
| Extensions | 2 a. If new passwords do not match, display an error message and prompt reattempt. |

Table 8.1

|  |  |
| --- | --- |
| Use Case  Name | Request Password Reset |
| Primary Actor | All Users |
| Main Flow | 1. Enter email address 2. Confirm password reset |

Table 8.2

|  |  |
| --- | --- |
| Use Case  Name | Register User |
| Primary Actor | Administrator |
| Main Flow | 1. Enter user information 2. Select user’s access rights 3. Provide user a password 4. Submit |

Table 8.3

|  |  |
| --- | --- |
| Use Case  Name | Delete User |
| Primary Actor | Administrator |
| Main Flow | 1. Enter email 2. Confirm deletion 3. Send user email notifying account deletion |

Table 8.4

|  |  |
| --- | --- |
| Use Case  Name | Reset User Password |
| Primary Actor | Administrator |
| Main Flow | 1. Select User from list of Users that have requested password resets 2. Confirm Selection 3. Provide user a password 4. Submit |

Table 8.5

|  |  |
| --- | --- |
| Use Case  Name | Backup database |
| Primary Actor | Administrator |
| Main Flow | 1. Select Backup database |
|  | 2. Confirm back up |

Table 8.6

### Non Functional Requirements

**Security Requirements**

Due to the system being hosted on the internet, it will be vulnerable to security threats present. Protection must be given to known threats such as SQL Injection. For user authentication, their email address will be used for them to sign in.

Many different users will have access to the system so it is important that no user views data they aren’t meant to, so appropriate access rights must be allocated for the system.

**Precision and Accuracy Requirements**

Because the system is going to be storing highly sensitive data about staff, students and even transaction data it has to be precise and accurate this means that any sort of human error should be avoided and the system has to be validated.

**Software Quality Assurance**

Reliability and robustness are features that will be focused on during development. It is important that the product handles all tasks presented to it and does not crash at all as this may affect the end-user’s impression of the system and go on to hinder the transfer to the system.

The product will have to be made as easy to use as possible as the intended end users have little experience with computers.

**Business Rules**

Marks and other grading information can only be entered after the respective exams have been completed. A teacher should only enter marks for students they’ve taught.

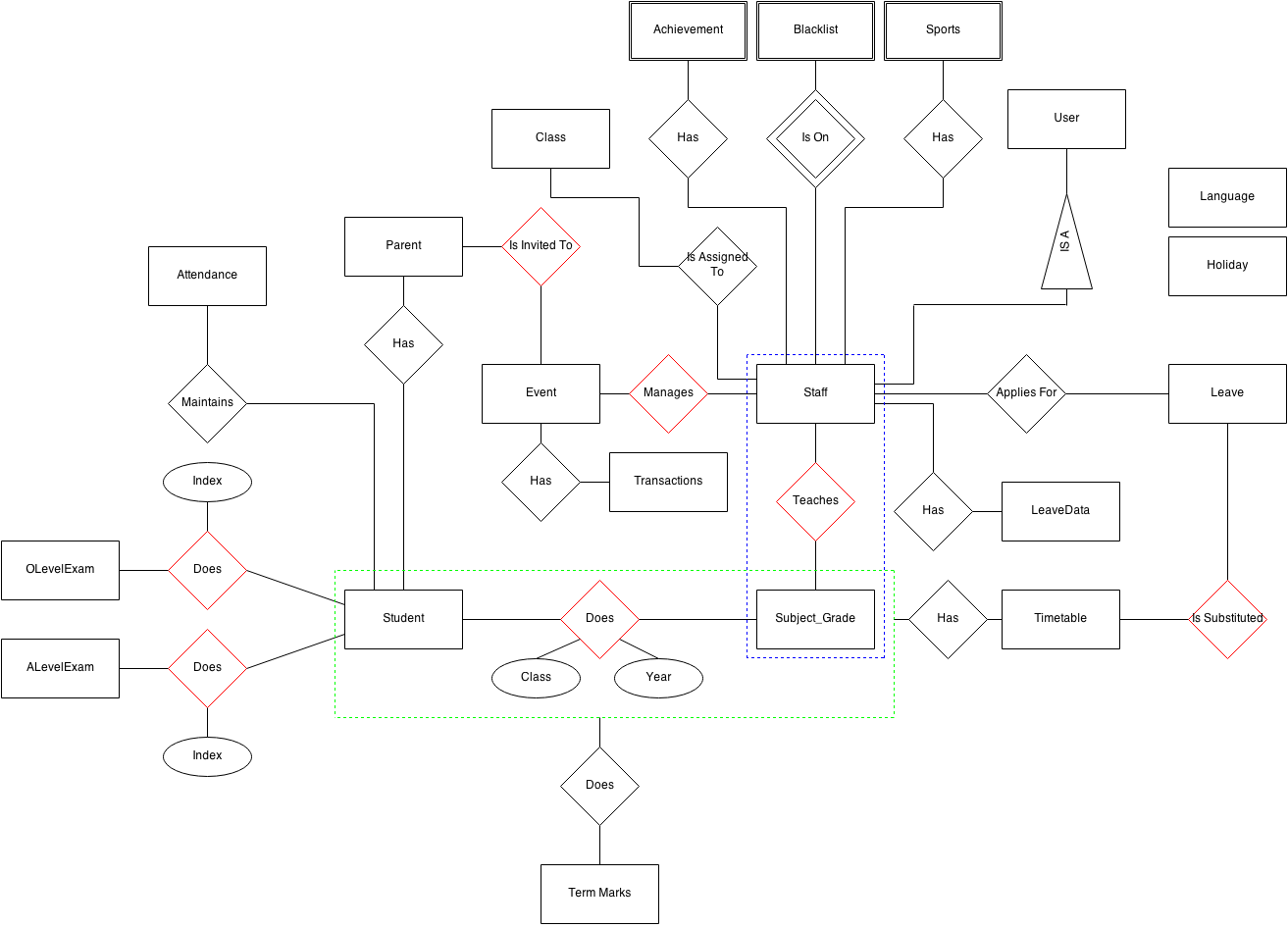
The year plan has to be created at the beginning of the academic year, changes will be allowed though.

Only the principal can approve leave.

Only the class teacher of a class can mark the attendance of that class.

## Design

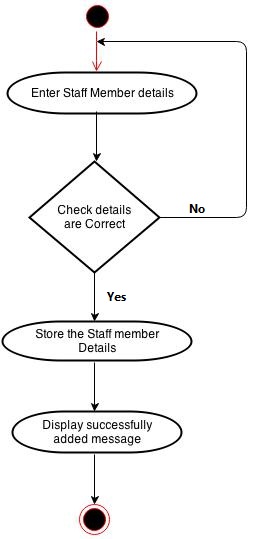
### Entity Relationship Diagram



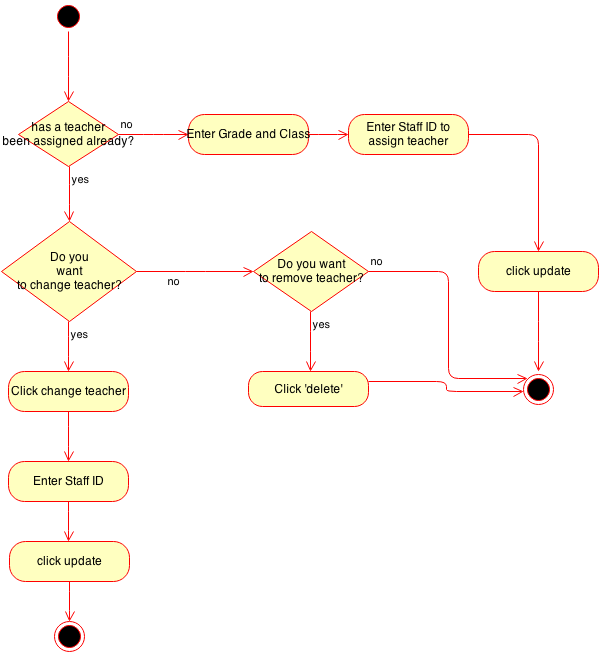
### Activity Diagrams

**Staff Management Module**

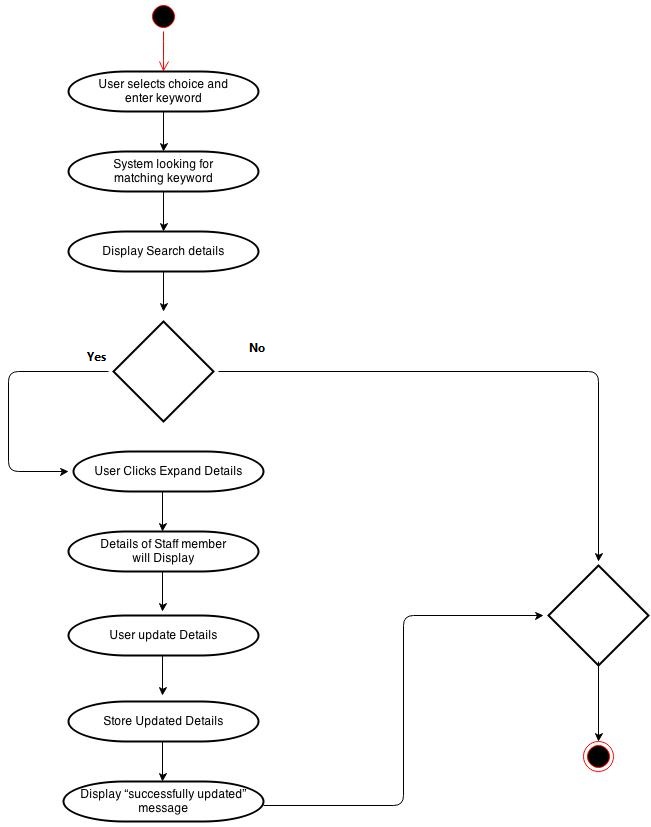
Register Staff Member



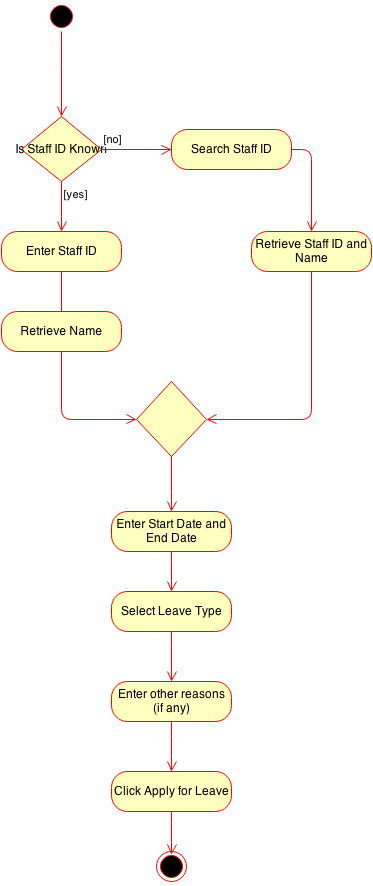
Assign Teacher to Class



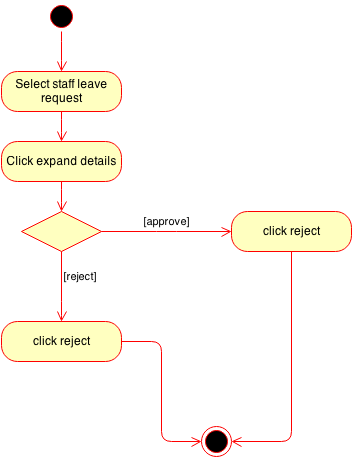
Search and Update Staff



**Leave Management Module**

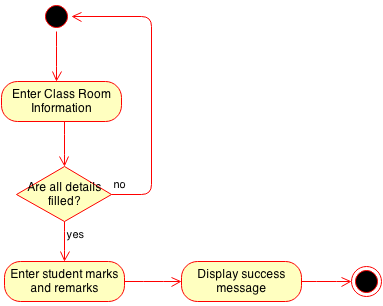
Apply for Leave

Approve Leave

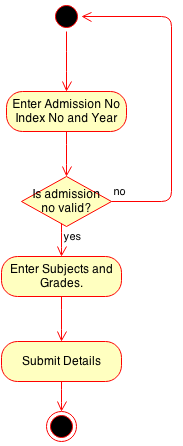


**Marks and Grading Module**

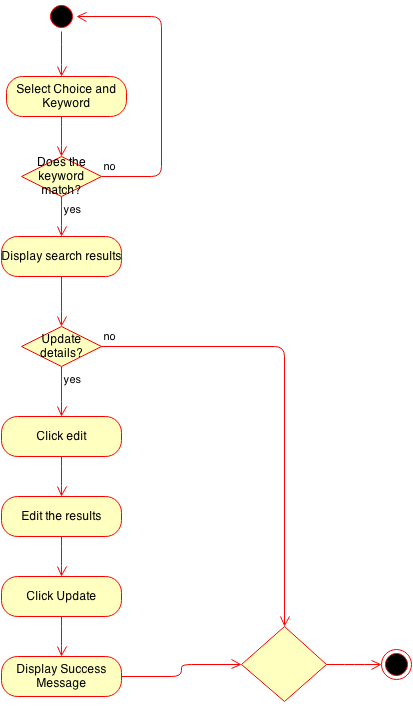
Enter Term Test Marks



Advanced Level and Ordinary Level Input

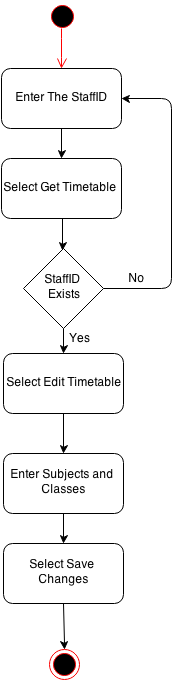


Search and Update Marks

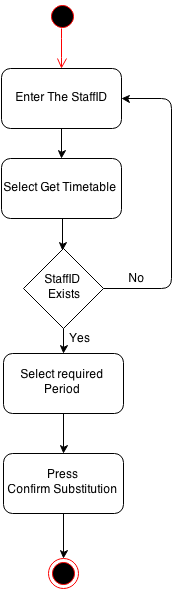


**Timetable Module**

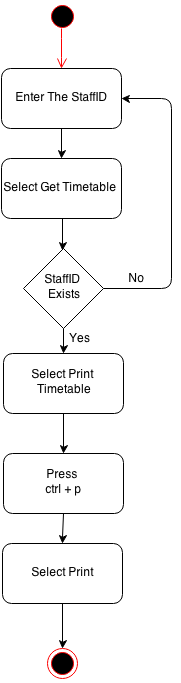
Create Timetable



Substitute Teacher

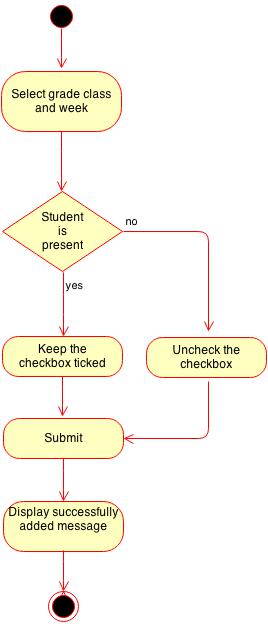


Print Timetable



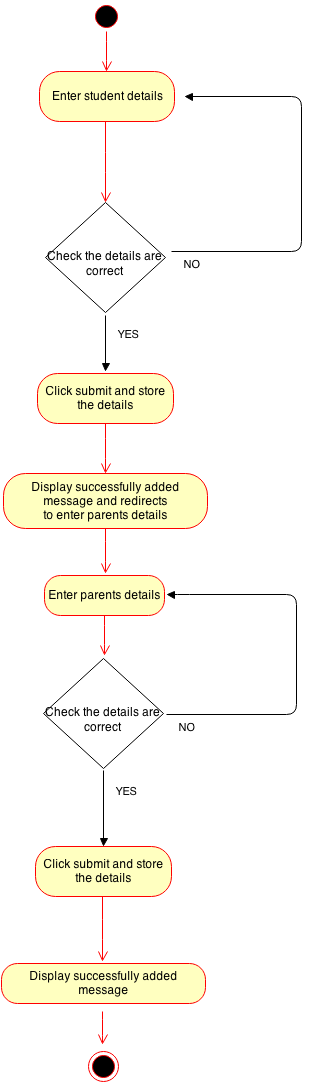
**Attendance Module**

Mark Attendance

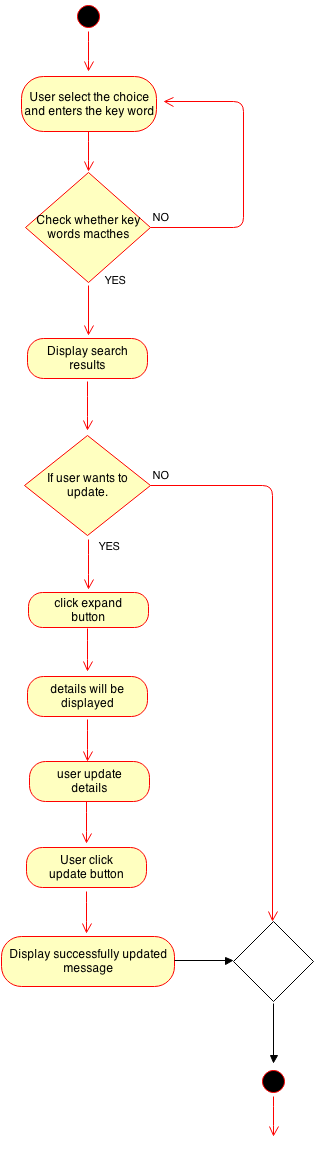


**Student Information Management Module**

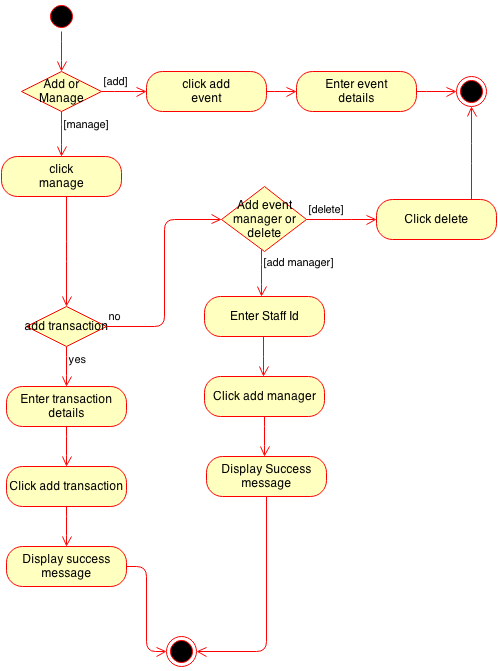
Register Student



Search Student



**Event Management Module**



## Implementation

During the design phase it was identified that the system be split into 8 major modules with several smaller modules complementing them. Here, the minor modules and design objects will be discussed before discussing the major modules.

### Common Systems and Objects

#### Master Page

The master page provides a unified interface to the system. It includes the menu, links to the home page, logging in or out and links to language changing links.

#### Database Connection and Querying

A separate class was implemented for creating and handling the database connection. The singleton design pattern was used to use the same object to minimise conflicts.

A single file was used to store all the database querying and retrieving functions.

#### Server-side Functions

Common files was used to store functions that all modules frequently used including form validation code as well as other functions.

#### Client-Side Functions

Several client-side functions were implemented to increase the usability of the system, using JavaScript. These functions were also included in common files.

### Staff Management

The staff management module is the most critical module from the client’s perspective. It comprises of staff member registration, searching and updating and producing the staff reports.

### Leave Management

A module was created for staff members to apply for leave and for any higher levelled users to approve leave requests. Access levels have been implemented in the system for only higher levelled users to approve leave.

### Timetables

Teacher’s timetables are stored with the system. Timetables have to be entered teacher-wise. The system then allows timetables to be viewed class-wise. Teachers can also be substituted and the teachers who are free to be substituted are suggested.

### Student Information

The student information module handles the registering of students and searching for and updating student details.

### Event Management

This module was developed for staff members to manage school events and record transaction information for the events.

### Attendance

The attendance of students can be recorded using the attendance module. This module also includes a year plan that has to be set at the beginning of the academic year with the school holidays.

### Marks and Grading

Student marks including the examination results from O/Level and A/Level examinations can be stored and searched for with this module.

### User Management

This system allows the creation and deletion of users to the system. Access levels can also be granted for users to have different rights.

## Testing

<Describe your test plan and Evidence that all aspects of the system have been tested>

Testing was done primarily on the server side using regular expressions and other custom functions.

The below functions were called after form submission to validate fields.

function isFilled($value){

if (($value == NULL) || ($value == "")){

return false;

}

else{

return true;

}

}

function isNIC($nicNumber)

{

if (preg\_match("/^[0-9]{9}[v|x]$/i", $nicNumber)){

return true;

}

else{

return false;

}

}

function isNumeric($courseOfStudy)

{

if(preg\_match("/^[0-9]$/", $courseOfStudy)){

return true;

}

else{

return false;

}

}

function isAlphabetic($nameWithInitials)

{

if (preg\_match("/^([A-Z]|\.|\s)\*$/i", $nameWithInitials)){

return true;

}

else{

return false;

}

}

function isContactNumber($contactnumber)

{

if(preg\_match("/^[0-9]{10}$/", $contactnumber)){

return true;

}

else{

return false;

}

}

function hasSpaces($value)

{

for ($i = 0; $i < strlen($value); $i++){

$curr = substr($value, $i, 1);

if(strcmp($curr, " ") == 0){

return true;

}

}

return false;

}

# Evaluation

## Assessment of the Project results

The project result/outcomes was analyzed by carrying out meetings with the client. According to the client the design had to be improved to be more user friendly but as far as functionality goes it met most of the requirements specified in the SRS document.

In comparison to the SRS document, the final software solution meets the functions stated. However there is a difference between processes carried out in the each of the 8 business functions compared to that of the final software solution. These differences were caused because of changing client requirements as he thought that it was a better approach to improve the software functionality.

## Lessons Learned

The most important aspect we’ve learnt is that it is important to work as a team. If we didn’t have proper team work this project would have been impossible to accomplish.

Furthermore in this project we’ve come to realize that time management plays a huge role in project management. Keeping to a proper schedule would result in increase of software quality and having a less risk of project failure. That being said the team is thankful to have managed to meet every milestone and come to meet the requirements specified.

Invaluable experience was gained doing this project and a lot of exposure was acquired on how industrial project work is carried out.

## Future Work

In the future a short messaging system can be implemented where a staff member can be notified if their leave request (in the leave management system) has been approved or rejected, and can also be used to notify a teacher if they’ve been substituted to a class in case a teacher is absent.

A mobile application can be developed to access the system so that the principal or any other staff member can use it on the go.

Much more functionality can be added. This system mainly focuses on the staff, a library system component can be implemented so that borrowing and lending of books or retrieval of book information can be done online.

Furthermore once the system is deployed, as part of the software development life cycle maintenance will be carried out. This includes corrective, adaptive and perfective maintenance methods.

# Conclusion

< This section sums up the whole project. Discuss the realization of the original objectives/goals and how work can be taken further. Highlight the weaknesses/limitations of your proposed technique but you must always suggest a solution to all these (especially in future work). Also highlight the benefits of developing this project to the client organization>

While developing the system, some requirements that were initially agreed upon were dropped, but the essential system was developed to the customer’s requirements.

## Identified limitations and their solutions

Language data is currently all stored in a single table. This could take up time when retrieving as the same table has to retrieve several language records for each form.

A solution would be to appropriately fragment this table into labels and drop-down options and all draw options for a combo box in a single query. Alternatively, language data can be stored in a separate data store like a static file with frequently used data cached and used immediately when required. This is a more efficient solution that could be used when scaling up the system

# References

<Include a list of references done in the IEEE referencing style>

1. jQuery API Documentation. Available: api.jquery.com

Appendix A: Design Diagrams

<Include the main UML diagrams in the main text and supplementary design diagrams can be included here>

Appendix B: Test Results

< Additional test result tables and figures can be included here >

Appendix C: Selected Code Listings

**Generating the Initial Year Plan**

<?php

function InsertBlankTd($numberOfTdsToAdd) {

$tdString = "";

for($i = 1; $i <= $numberOfTdsToAdd; $i++) {

$tdString .= "<td></td>";

}

return $tdString;

}

function FriendlyDayOfWeek($dayNum) //Converts Sunday to 7

{

if ($dayNum == 0)

{

return 7;

}

else

{

return $dayNum;

}

}

for ($mC=1;$mC<=12;$mC++)

{ //mc is Month, dDay is digit of day

$currentDT = mktime(0, 0, 0, $mC, $dDay, $dYear);

echo "<tr><td class='monthName'><div>" . date("F", $currentDT) . "</div></td>";

$daysInMonth = date("t", $currentDT);

echo InsertBlankTd(FriendlyDayOfWeek(date("w", $currentDT)) - 1);

for ($i = 1; $i <= $daysInMonth; $i++) {

$exactDT = mktime(0, 0, 0, $mC, $i, $dYear);

$formattedDate = date("d/m/Y", $exactDT); /\*$i . "/$mC" . "/$dYear";\*/

$class = "";

echo "<td class='" . $class. " days day" . FriendlyDayOfWeek(date("w",$exactDT)) . "' name=\"" . $formattedDate . "\" >$i</td>";

}

echo InsertBlankTd($dDaysOnPage - $daysInMonth - FriendlyDayOfWeek(date("w",$currentDT)) + 1);

echo "</tr>";

}

?>

**jQuery for the Navigation Menu**

$(document).ready(function () {

$('#nav > li > a').click(function(){

if ($(this).attr('class') != 'active'){

$('#nav li ul').slideUp(300);

$(this).next().slideToggle(300);

$('#nav li a').removeClass('active');

$(this).addClass('active');

}

else{

$('#nav li ul').slideUp(300);

$('#nav li a').removeClass('active');

}

});

$("#logInLink").on("click", function(e){

logIn();

});

});

var w = Math.max(document.documentElement.clientWidth, window.innerWidth || 0);

if (w <= 1280)

{

moveNav();

}

function moveNav()

{

var menuBut = document.getElementById('menuButton');

$(menuBut).removeClass('hidden');

var element = document.getElementById('nav');

element.style.left = -230 + "px";

}

**Generating a Timetable**

$timeArray = array("07.50-08.30", "08.30-09.10", "09.10-09.50", "09.50-10.30", "10.50-11.30", "11.30-12.10", "12.10-12.50", "12.50-01.30" );

$colourArray = array("#f69988", "#f48fb1", "#ce93d8", "#b39ddb", "#9fa8da", "#afbfff", "#81d4fa", "#80deea", "#80cbc4", "#72d572", "#c5e1a5",

"#e6ee9c", "#ffcc80", "#fff59d", "#ffe082"); //15

$classColour = array();

for($i = 0; $i < 8; $i++){

if(($i == 4)){ //INTERVAL

$intervalRow = "\t<tr class=interval>\t<td>10.30-10.50</td> \t<td colspan=\"5\" >" . "INTERVAL" . "</td>\t</tr>\n";

echo $intervalRow;

}

for($x = 0; $x < 6; $x++){

$thisCell = "";

if($x == 0){

$thisCell = "\t<tr>\t";

$thisCell .= "<td class='time'>" . $timeArray[ $i ] . "</td>";

}

else{ //Normal rows are here.

$number=($i + (8 \* ($x - 1) ));

$thisCell = "";

$subject = $myTime->slot[$number]->Subject;

$class = $myTime->slot[$number]->Grade . " " . $myTime->slot[$number]->Class;

$currColour = $colourArray[ (rand(0,14)) ];

while ( in\_array($currColour, $classColour) )

{

$currColour = $colourArray[ (rand(0,14)) ];

}

if( trim($class) == ""){

$classColour[$class] = "#dedede";

}

if ($classColour[$class] == ""){

$classColour[$class] = $currColour;

}

else{

$currColour = $classColour[$class];

}

$classDiv = "<div class='classroom'><input id='classroom\_$number' name='classroom\_$number' readonly value='" . $class . "' /></div>";

$thisCell .= "\t<td class='subject' style='background-color:$currColour;' id=\"" . $number . "\">" . $classDiv;

$thisCell .= "<textarea id='subject\_$number' name='subject\_$number' readonly style='background-color:$currColour;'>" . $subject . "</textarea></td>";

if ($x % 5 == 0){

$thisCell .= "\t</tr>\n";

}

}

echo $thisCell;

}

}

?>