Student Registration System

Software Design Specification
Table of Content

1. Introduction ........................................................................................................................................04
   1.1 Purpose ........................................................................................................................................04
   1.2 Scope ............................................................................................................................................04
   1.3 Glossary .......................................................................................................................................05
   1.4 References ...................................................................................................................................05
   1.5 Overview .....................................................................................................................................06

2. Architectural Design ........................................................................................................................06
   2.1 High level component and their interaction ..............................................................................06
       2.1.1 Components ..........................................................................................................................07
       2.1.2 Interfaces ..............................................................................................................................08
   2.2 Architectural Styles/patterns .......................................................................................................09
       2.2.1 MVC architecture styles .......................................................................................................09
       2.2.2 Three tier client server architecture .....................................................................................09
       2.2.3 Different process and their communications .....................................................................10
   2.3 Physical arrangement of devices and Communication & installed software components on devices ..........................................................................................11
   2.4 Design decisions applied whole application .............................................................................13
       2.4.1 Object oriented software development methods .................................................................13
       2.4.2 Three tier client server architecture .....................................................................................13
       2.4.3 MVC architectural patterns ..................................................................................................13

3. Component and Detail design ...........................................................................................................14
   3.1 Design patterns and other design techniques used ....................................................................14
       3.1.1 Abstract factory Pattern .......................................................................................................14
       3.1.2 Singleton Pattern .................................................................................................................14
       3.1.3 Observer Pattern ..................................................................................................................14
       3.1.4 Adapter Pattern ....................................................................................................................15
       3.1.5 Techniques used ....................................................................................................................15
   3.2 Class Diagram ..............................................................................................................................16
3.3 Sequence Diagram

3.3.1 Sequence Diagram 01: Login into the system. .................................................. 17
3.3.2 Sequence Diagram 02: Change user password............................................... 18
3.3.3 Sequence Diagram 03: Add users................................................................. 19
3.3.4 Sequence Diagram 04: Remove users......................................................... 20
3.3.5 Sequence Diagram 05: Manage users......................................................... 21
3.3.6 Sequence Diagram 06: Edit profile information........................................... 22
3.3.7 Sequence Diagram 07: Remove subjects from student's selection............. 23
3.3.8 Sequence Diagram 08: Create new subjects............................................. 24
3.3.9 Sequence Diagram 09: Edit existing subjects............................................ 25
3.3.10 Sequence Diagram 10: Delete subjects.................................................. 26
3.3.11 Sequence Diagram 11: Create new subject lists...................................... 27
3.3.12 Sequence Diagram 12: Make adjustment of credits per subject........... 28
3.3.13 Sequence Diagram 13: Subject combination tool.................................. 29
3.3.14 Sequence Diagram 14: Publish time table............................................. 30
3.3.15 Sequence Diagram 15: Update time table............................................. 31
3.3.16 Sequence Diagram 16: Current subjects by subject vice..................... 32
3.3.17 Sequence Diagram 17: Past subject selections by subject vice........... 33
3.3.18 Sequence Diagram 18: View the past, present subject selections of a student................................................................. 34
3.3.19 Sequence Diagram 19: Edit student's profile.......................................... 35
3.3.20 Sequence Diagram 20: Select subjects.................................................. 36
3.3.21 Sequence Diagram 21: Remove subjects from the list............................. 37
3.3.22 Sequence Diagram 22: Save................................................................. 38
3.3.23 Sequence Diagram 23: View previous semester information............... 39
3.3.24 Sequence Diagram 24: View Notices...................................................... 39
3.3.25 Sequence Diagram 25: Guest view of the student's profile...................... 40
3.3.26 Sequence Diagram 26: Super user confirmation..................................... 41
3.3.27 Sequence Diagram 27: Access Control Matrix....................................... 42

3.4 Algorithm Design......................................................................................... 43
3.5 Database Design........................................................................................ 51
  3.5.1 Relational Model..................................................................................... 51
  3.5.2 Normalization/DE normalization............................................................ 52
    3.5.2.1 Anomalies...................................................................................... 52
    3.5.2.2 Normalization in our systems...................................................... 53
  3.5.3 Data Types Identification.......................................................................... 54
    3.5.3.1 Indexes......................................................................................... 54
  3.5.6 Triggers.................................................................................................... 58
3.6 User interface design

3.6.1 Rules and guidelines for user interface designing

3.6.1.1 User input validation methods

3.6.1.2 Guidelines for error messages, warnings and tips

3.6.1.3 Guideline for interface designing

3.6.1.4 Guidelines for error messages warnings and tips

3.6.2 User interface for each use case

Interface 1: Login into the system
Interface 2: Change user password
Interface 3: Add users
Interface 4: Remove users
Interface 5: Manage users
Interface 6: Edit profile information
Interface 7: Remove selected subjects from student's selections
Interface 8: Create new subjects
Interface 9: Edit existing subjects
Interface 10: Delete subjects
Interface 11: Create new subject lists
Interface 12: Make adjustment of credits per subject
Interface 13: Subject combination tool
Interface 14: Publish time table
Interface 15: Update time table
Interface 16: Current subjects by subject vice
Interface 17: Past subject selections by subject vice
Interface 18: View past & present subject selections of a student
Interface 19: Edit student's profile
Interface 20: Select subjects
Interface 21: Remove subjects from the list
Interface 22: Save
Interface 23: View previous semester information
Interface 24: View Notices
Interface 25: Guest view of the student's profile
Interface 26: Super user confirmation
Interface 27: Access Control Matrix
1. Introduction

1.1 Purpose

This software design specification is made with the purpose of outlining the software architecture and design of the Student 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.

1.2 Scope

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.

1.3 Intended audience

This document is mainly for the developers and the technical and academic staff of Rajarata university of Sri Lanka and student representatives of the University.
### 1.4 Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm Design</td>
<td>Specific method to create a mathematical process in solving problems</td>
</tr>
<tr>
<td>Architectural Design</td>
<td>Establishing the overall structure of software system</td>
</tr>
<tr>
<td>AS</td>
<td>Applied Science</td>
</tr>
<tr>
<td>Compatible</td>
<td>Capable of orderly efficient integration and operation with other elements in a System with no modification</td>
</tr>
<tr>
<td>Database</td>
<td>A collection of stored related data</td>
</tr>
<tr>
<td>Encapsulate</td>
<td>To express or show the most important facts about something.</td>
</tr>
<tr>
<td>ER diagram</td>
<td>Entity Relationship Diagram; Data model for describing a database in an abstract way</td>
</tr>
<tr>
<td>HPT</td>
<td>Health Promotion Technology</td>
</tr>
<tr>
<td>ICT</td>
<td>Information &amp; Communication Technology</td>
</tr>
<tr>
<td>Instantiate</td>
<td>To represent (an abstract concept) by a concrete or tangible</td>
</tr>
<tr>
<td>SDS</td>
<td>System Design Specification</td>
</tr>
<tr>
<td>Sequence Diagram</td>
<td>An interaction diagram that shows how processes interact with one another and in what order</td>
</tr>
<tr>
<td>SRS</td>
<td>Software Requirements Specification</td>
</tr>
</tbody>
</table>

### 1.5 References

- *Software Requirements Specification (SRS) of the Student Registration System*
- *Golden rules of user interface design by Theo Mandel*
- *Internet*
1.6 Document Overview

The next chapter of the document has described architectural design of the Student 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 chapter 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 with screen shots of the interfaces.
2. Architectural Design

2.1 High level components and interfaces

2.1.1 Components

- **Student component**

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

- **Authentication and user management component**

  This is the major sub system that is responsible for the security of the Student Registration system. It authenticate users and also handles the user management activities such as creating new user accounts, removing accounts from the system etc. Furthermore this component Implements the “control access privilege matrix”.

- **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.

- **Publish component**

  Publish 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.

- **Public component**

  This is a relatively small subsystem compared to the other components of the Student Registration System. This is the component which is responsible for the guest viewer (public) to view student results.
2.1.2 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.

**Authentication and user management component**

authenticateUser: This is the interface that allows the users to login to the system. This will guide the user to the relevant home page.

**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.

**Publish component**

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

Gettimetable: This is the interface where the time table is published. This is connected with the subject component.

**Public component**

viewResults: This is the interface which shows the results of a student to a Guest user, therefore connected with the Student component.
2.2 Architectural Styles / Patterns

The Student Registration System will be developed under two main architectural styles/ patterns. Development of the project will be done in MVC architectural style and also 3 tier Client/Server Architecture. Client can browse the internet and access the Student Registration System provided within the local area network of the University.

2.2.1 MVC Architecture Style (Model – View – Controller)

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.

We will use this MVC Style for the Student Registration System because, there are multiple ways to view and interact with data. Also used when the future requirements for interaction and presentation of data are unknown. In some software systems the code between the process logic and interface are mixed. This will reduce the modularity of application and make the system more difficult to maintain. To avoid this problem we have decided to use MVC architectural style to separate the application logic with the interface. The main advantage of this is style allows the data to change independently of its representation and vice versa. Support presentation of the same data in different ways with changes made in one representation shown all of them.

2.2.2 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. Because servers can be replicated, may also be used when the load on a system is a variable.
• 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 (web/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/application server to make requests and to retrieve data from the database. It then displays to the user the data retrieved from the server.

*Example of the 3-tier architecture in the Student Registration System*

If a student needs to view his current semester subject combination, first he has to login to the system. Then he has to click “User” option, after he has to login “Summary” option. Then system will display the information. In this process, Login screen, users’ main screen and Subject combination summary screen are defined into the Client tier, data for login information and profile information and SQL queries for those information are maintained into the Data tier and controller logic for login process and loading profile information from the database are defined in Middle tier.

2.2.3 Different process and their communication

In the Student Registration System, there are number of different processes, such as database server process, web server process, connections between above servers likewise. When sending mails there should run a mail server. HTTP protocol is using to communicate with web servers, SMTP protocol is using to communicate with mail servers. They should communicate each other well to perform the functions of whole application.
2.3 Physical arrangement of devices and Communication and installed software Components on devices.

The Student Registration System deals mainly with hardware devices and installed software components on devices. The System performs many tasks. It consists of web based system used by Super Users, Administrators and Students of the university. The system helps to record students’ personal details, publish time table, preview student result, select subjects for the semester. Therefore the web based part is expected to run on various operating system platforms. The client server architecture of the system requires to remotely connecting with client and server through the internet connection. The system has two nodes such as the Web server and Clients. The nodes can represent specific instances (workstations) or a class of computers (web server), which is a virtual machine. The applications of the system will run on the web server connected to the database server.

Internet is the worldwide interconnection of all smart communication devices that have a valid IP. There should be installed browser software to access internet. If the user accesses the system, directly through the internet connection the user has to install dongle or modem or relevant device and Wi-Fi or etc... to connect with the system. If the user accesses the system through the intranet connection, the server should install the relevant software. Most of intranet accessing modes refer to the website of the organization which can only be accessed by its employees who have a user name and password. So, considering that type of security purposes, it is better to access this Student Registration through the intranet, but it should be accessed through the internet directly also.

When generating reports such as students’ result and subjects details there should be installed the crystal report software. And also to print the generated reports the user machine should be installed the software relevant to its connected printer.
Physical arrangement of devices in a typical network.

In this diagram, it shows that, the only software a client need is to access this system is a browser.

Arrangement of devices and serves

Our University Student Management System needs some specific set of servers and devices. Such as:
- Server to host web applications and web service applications.
- Database server to store and manage data.
- Personal computer, note book, smart phone etc... to access the website.
- Modem/ router/ switch/ hub/ Wi-Fi network/ cable network etc... and also need an Internet Service Provider to have the internet connectivity.
**Communication among components**

Above devices are communicating with each other. Personal computer communicates with web server and the database server through HTTP protocol. It communicates with mail server through SMTP protocol. Cable network or Wi-Fi network is also a communication method using in connecting different network components.

### 2.4 Design decisions applied whole application

#### 2.4.1 Object oriented software development methods

**Reasons:**
- Improved software maintainability.
- Faster development
- Lower cost development
- Improved software development productivity
- Higher quality software

#### 2.4.2 Three-Tier Client Server 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.

#### 2.4.3 MVC Architectural Pattern

**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.
3. Component and Detailed Design

3.1 Design Patterns and Techniques used

3.1.1 Abstract factory Pattern

Abstract factory pattern is creational design pattern that provides a way to encapsulate a group of individual factories that have a common theme without specifying their concrete classes. Abstract factory pattern offers the interface for creating a family of related objects without explicitly specifying their classes.

Applications:
The design pattern will be applied in creating different user accounts which are the different factories. It will also be used to keep the system.

3.1.2 Singleton Pattern

This is a creational design pattern and is one of the simplest patterns in the field of software engineering. It involves only one class which is responsible to instantiate itself, so that it creates no more than one instance. The singleton pattern is useful when access to limited resource needs to be controlled.

Applications:
In the student registration system this pattern will be used for database manager.

3.1.3 Observer Pattern

The observer pattern is a behavioral pattern which defines a one-to-many dependency between objects where a state change in one object results with all its dependents being notified and updated automatically. This pattern may be used in all situations where more than one display format for state information is required and where it is not necessary for the object that maintains the state information to know about the specific display formats used.

Applications:
Observer pattern will be used in the Student Registration system for the operations of the system users.
3.1.4 Adapter Pattern

The adapter pattern is a structural pattern that translates one interface for a class to a compatible interface. This will convert the interface of a class in to another interface that the user expects. Adapter gives the opportunity for the classes with incompatible interfaces to work together.

Application:
This pattern will be used in the Student Registration system when displaying information from the database.

3.1.5 Techniques Used

Prototyping

In designing the Student registration System prototyping will be used to demonstrate underpinning concepts of the designing and for user interfaces. This technique will provide the opportunity for the system users to experiment the software to a certain extent during the development process.
3.2 Class diagram
3.3 Sequence diagrams

3.3.1 Sequence diagram 1: Login into the system

1 : selectLogin()
2 : display login
3 : enter()
4 : validate()
5 : validate()
6 : validated()
7 : privileged page()
8 : show privileged page()
9 : show privileged page()
10 : invalid details()
11 : display error message()
12 : display login()
13 : enter email()
14 : confirm email
15 : validate email()
16 : validate email()
17 : mail password to the email()
3.3.2 Sequence diagram 2: Change user password

User

Display

Session manager

password

Database

1: edit password
2: set password
3: get password
4:
5: show
6: display
7: display
8: edit password
9: enter new password
10: reenter new password
11: click update button
12: save
13: valid password
14: set new password
15: save password
16: save successful
17: success message

incorrect current password
22: invalid password

invalid reentered password
27: mismatch password

18: save
19: valid password
20: invalid
21: invalid password
23: save
24: valid password
25: invalid
26: mismatch password
3.3.3 Sequence diagram 3: Add users

1: add user
2: display add user
3: add new user
4: add username
5: add password
6: select type
7: select privilege level
8: email
9: confirm
10: confirm admin password
11: save
12: validate
13: valid
14: create new account
15: invalid
16: invalid
17: invalid message
18: display errors
19: invalid
20: invalid
21: invalid message
22: display error
23: invalid
24: invalid
25: invalid message
26: display error
3.3.4 Sequence diagram 4: Remove users

Admin

display

controller

user

1: remove users()

2: display list

3: delete()

4: ask admin password

5: delete()

6: delete()

7: display

8: error message()

9: display

10: error message()

11: display

incorrect current password

invalid reentered password
3.3.5 Sequence diagram 5: Manage users

1: update user
2: display
3: change privileges
4: confirm changes
5: confirm
6: save
7: confirm message
8: logout user
9: change privileges
10: save
11: display user

If there are logged user
3.3.6 Sequence diagram 6: Edit profile information

- **super user/admin**
  - 1: login()
  - 2: display
  - 3: select edit profile()

- **display**
  - 4: make changes()
  - 5: confirm changes()
  - 5: display changes()

- **controller**
  - 6: confirm message
  - 7: save()
  - 9: error message

- **database**
  - 8: confirm message
  - 10: error()
  - 11: terminate connection()
3.3.7 Sequence diagram 7: Remove selected subjects from student's selections
3.3.8 Sequence diagram 8: Create new subjects

- **super user/admin**
- **display**
- **controller**
- **database**

1: login

2: display

3: select subject operations

4: create new subjects

5: make changes

6: display changes

7: save

8: confirm message

9: change imm

10: display changes
3.3.9 Sequence diagram 9: Edit existing subjects

1: login
2: display
3: select subject operations
4: edit subjects
5: make changes
6: display changes
7: save
8: confirm message
9: change imm
10: display changes
3.3.10 Sequence diagram 10: Delete subjects

super user/admin

1: login()

2: display
3: select subject operations()

controller

4: delete subjects()
5: make changes()
6: display changes()
7: save()

display

8: confirm message

database

keep current assignments

9: change imm()

10: display changes
3.3.11 Sequence diagram 11: Create new subject lists

1: select subject selections()

2: select create new subject list()

3: select academic year, sem,stream

4: add new subject list()

5: subject list

6: subject list

7: enter subjects and details()

8: confirm to proceed

9: save()

10: confirm

11: successful notification
3.3.12 Sequence diagram 12: Make adjustment of credits per subject.
3.3.13 Sequence diagram 13: Subject combination tool

```
1: login()
2: display
3: select subject operations()
4: subject selection tool()
5: select details()
6: select subjects()
7: make changes()
8: save()
9: confirm message
```
3.3.14 Sequence diagram 14: Publish time table

1: login
2: display
3: create timetable
4: temporary save
5: display conflicts
5: save
7: confirm message
8: manual change
9: save

system detect some problems
3.3.15 Sequence diagram 15: Update time table

1. login()
2. display
3. edit timetable()
4. select details()
5. display
6. edit subjects()
7. temporary save()
8. display conflicts
9. save()
10. confirm message
11. connect()
12. notify connectivity()
3.3.16 Sequence diagram 16: Current subjects by subject vice

1: login()

2: display

3: select view student choices()

4: enter index()

5: confirm()

6: display

7: display details

8: select subjects()

9: exit()

10: invalid index

11: enter index()

database update fails

database
3.3.17 Sequence diagram 17: Past subject selections by subject vice

```
1: login()
2: display
3: select view student choices()
4: enter index()
5: confirm()
6: display
7: display details
8: select past subjects()
9: exit()
10: invalid index
11: enter index()
```

database update fails
3.3.18 Sequence diagram 18: View the past & present subject selections of a student

```
super user/admin

display

controller

database

1: login() ->
2: display
3: select view student choices() ->
4: enter index() ->
5: confirm() ->
6: display
7: display details
8: select all subjects() ->
9: exit() ->
```
3.3.19 Sequence diagram 19: Edit student's profile

1: login()

2: edit data()

3: confirm changes()

4: save()

5: error message()

6: connect to database()
3.3.20 Sequence diagram 20: Select subjects

1: login()
2: subject list()
3: determine detail()
4: display
5: change subjects()
6: check details()
7: display time
8: save()
9: success
10: error message()

minimum credit mismatch

11: clear()
12: save()
3.3.21 Sequence diagram 21: Remove subjects from the list
3.3.22 Sequence diagram 22: Save

```
user -> Display
| database not available |

Display -> Controller
| detect changes() |

Controller -> Database
| save() |

Database -> Controller
| error message() |

Controller -> Display
| contact message() |
```
3.3.23 Sequence diagram 23: View previous semester information

3.3.24 Sequence diagram 24: View Notices
3.3.25 Sequence diagram 25: Guest view of the student's profile
3.3.26 Sequence diagram 26: Super user confirmation

Super user

Display

Control

Database

1: select login
2: display login
3: send confirmation
4: send key
5: send key to superuser
6: notify
7: send key

Entered key is not match
3.3.27 Sequence diagram 27: Access Control Matrix

Super user

Display

Control

Database

1: select login()  
2: display login  
3: select subject operations()  
4: edit privileges()  
5: save()  
6: send notifications
3.4 Algorithm Design

**Login**

1. Get Username and Password
2. If username is equal to the entered Username & the password is equal to the entered Password
3. Then login successful
4. Else login failed
5. End If.

```
Get Username Get Password
IF FILE EXIST THEN
    READ Password FROM FILE
    IF FILE. Password == Entered Password
        Login successful
    ELSE
        PRINT “incorrect Username or password “
    END IF
ELSE
    PRINT “incorrect Username or password “
END IF
```

**Student - Edit Profile**

1. Student login to account
2. Select student personal details option

```
While Edit Profile
{
    Add details
    Remove details
}
Save changes
EXIT
```
**Student - View Previous Semesters Information**

1. Student login account
2. Select exam details option
3. Then view previous semester result preview
4. EXIT

**Student select & remove subjects**

1. Student login account
2. Select student subject option
3. Select academic year and the semester
4. Then user selects subject from the appeared list as preferred

   IF wrong selection
   Remove Subjects
   ENDIF
   Confirm selections
   EXIT

**Guest - Student Performance Preview**

1. Guest login account
2. Enter relevant Index number and Password
3. IF Index number is equal to the entered Index number and the Password is equal to entered Password
4. Then preview student performance
5. ELSE Print incorrect password or index
6. Back to step 2
7. EXIT
Administrator - Subject Operation

1. Admin login to account
2. Select admin selection option
3. Then go to subject managing tool

IF required privileges given

{ 
   Select relevant course, year, and the semester
   While {
      Viewing relevant subject information
      Edit Subject
      Remove Subject
   }
   Select update information

} ELSE
   Print privileges not given
EXIT
Administrator Account Operations

1. Admin login to account
2. Select login management tool

IF required privileges
{
    IF exiting account
    {
        IF select remove user
        {
            Confirm
        }
        ELSE select Manage User
    }
    IF Suspend account
    {
        Enter relevant details
        Confirm
    }
    ELSE
    {
        Select login Issuing Tool
        Select stream and year

        Enter default and unique option details
        Create account
    }
}
ELSE
Print “Required Privileges not given ”
EXIT
**Administrator - Time Table Publishing**

1. Admin login account
2. Select admin selection option

   IF required privileges given
   {
     Then go to Time Table Preview page
     Select course name, academic year and semester
     View relevant time table
     IF already
     Exit a time table
     Then update as admin prefer
     ELSE
     Create new time table
   }
   ELSE
   Print “Required Privileges not given”
   EXIT
Administrator - View Student Choices

1. Admin login to account
2. Select admin selection option

\[\text{IF Required privileges given} \]
\[\{\]
\[\text{Then go to student choices page} \]
\[\text{Enter relevant index number and registration number} \]
\[\text{IF change subjects} \]
\[\{\]
\[\text{Edit subject choices} \]
\[\text{Save} \]
\[\}\]
\[\text{ELSE} \]
\[\{\text{View past and present subjects}\} \]
\[\text{ELSE} \]
\[\{\text{Print “Privileges not given”}\} \]
\[\text{EXIT} \]
Administrator - View Users’ Profile

1. Admin login to account
2. Select login management option

   \textit{IF required privileges given}
   
   \{ 
   
   Enter user’s registration number
   View users profile
   
   \}
   
   \textit{ELSE}
   
   \{ 
   
   Print “Privileges not given”
   
   \}
   
   EXIT

Super User privilege Access Control Matrix

1. Super user login to system
2. Select privilege access control matrix

   \textit{IF select edit}
   
   \{ 
   
   Edit privileges
   Save changes
   
   \}
   
   \textit{ELSE}
   
   Selected view matrix
   
   EXIT
Super user - Account Operation

1. Super user login to system
2. Select account operations

\[IF \text{select remove user}\]
\{
    \text{Enter user type}
    \text{Enter relevant ID or Registration number}
    \text{Go to relevant user profile}
    \text{Confirm removal}
\}
\[ELSE \text{select add user}\]
\{
    \text{Select user type}
    \text{Enter details}
    \text{Create account}
\}
EXIT
3.5 Database design

3.5.1 Relational model
3.5.2 Normalization / DE normalization

Database normalization is the process of organizing the fields and tables of a relational database to minimize data redundancy and dependency. Normalization usually involves dividing a database into two or more tables and defining relationships between the tables. The objective is to isolate data so that additions, deletions, and modifications of a field can be made in just one table and then propagated through the rest of the database via the defined relationships.

There are three main normal forms, each with increasing levels of normalization:

- **First Normal Form (1NF):** Each field in a table contains different information. For example, in a Student list, each table would contain only one email field.

- **Second Normal Form (2NF):** Each field in a table that is not a determiner of the contents of another field must itself be a function of the other fields in the table.

- **Third Normal Form (3NF):** No duplicate information is permitted. So, for example, if two tables both require an email field, the email information would be separated into a separate table, and the two other tables would then access the email information via a password field in the student table. Any change to an email would automatically be reflected in all tables that link to the student table.

3.5.2.1 Anomalies

When there are too many attributes it is called as anomalies.

For example,

STUDENT (Reg-no, Index no, Fname, Lname, Mobile, Dept-name, Course name, Password, E-mail)

There are basically three types of anomalies.

1. **Deletion anomalies** – Deletion anomalies occur when we delete data from flawed schema.
2. **Insertion anomalies** – Insertion anomalies are occur when we try to insert data into flawed Table.
3. **Update anomalies** – Update anomalies occur when we change data in a flawed schema.
3.5.2.2 Normalization in our System

In this table it is satisfied requirements which should have for the first normalization. In second Normalization tables have changed as following.

Subject table has normalized in two tables as “subject” and “subject-credits”.

- Subject (sub-code, Sub-name)
- Subject-credits (Sub-code, Semester, Year, Credit)

Student table has normalized in to three tables as “Student”, “Student-course name” and “student year”.

- Student (Reg-no, Index-no, Fname, Lname, Mobile)
- Student-year (Reg-no, Course name, Password, Email, User name)
- Student-year (Index-no, Sem-no, Year)

Result preview tabel has normalized in to two tables as “Result preview” and “Result total GPA”.

- Result preview (Reg-no, Index-no, Course code, Name, Sub-Name)
- Result total GPA (Reg-no, Results, Credits, Total GPA)

Likewise we can normalizing our tables into sub tables. One table divide into several tables. For the third normalization also requirements have satisfied. There for now all the tables have satisfied the requirements in First, Second, and third normalization.
### 3.5.3 Data Types Identification

#### 3.5.3.1 Indexes

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size on disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reg-no</td>
<td>Int(11)</td>
<td>4 bytes</td>
</tr>
<tr>
<td>Index no</td>
<td>Int(11)</td>
<td>4 bytes</td>
</tr>
<tr>
<td>Course code</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Name</td>
<td>Varchar(200)</td>
<td>200 bytes</td>
</tr>
<tr>
<td>Sub-name</td>
<td>Varchar(200)</td>
<td>200 bytes</td>
</tr>
<tr>
<td>Result</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Credits</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Total GPA</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size on disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course Name</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Dept-name</td>
<td>Varchar(100)</td>
<td>100 bytes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size on disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reg-no</td>
<td>Int(20)</td>
<td>4 bytes</td>
</tr>
<tr>
<td>Index-no</td>
<td>Int(20)</td>
<td>4 bytes</td>
</tr>
<tr>
<td>Fname</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Lname</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Mobile</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Dept-name</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Course name</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Password</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>E-mail</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Name</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Sem-no</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Year</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
</tbody>
</table>
### Subject

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size on disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code</td>
<td>Varchar (20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Semester</td>
<td>Varchar (20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Year</td>
<td>Varchar (20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Credit</td>
<td>Varchar (20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Subject Name</td>
<td>Varchar (20)</td>
<td>20 bytes</td>
</tr>
</tbody>
</table>

### Important days

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size on Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sem-no</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Year</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Sub name</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>End date</td>
<td>Date</td>
<td></td>
</tr>
<tr>
<td>Start date</td>
<td>Date</td>
<td></td>
</tr>
<tr>
<td>Exam date</td>
<td>Date</td>
<td></td>
</tr>
</tbody>
</table>

### Compulsory

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size On Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject name</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Credit</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Code</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Year</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Semester</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
</tbody>
</table>

### Optional

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size On Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject name</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Credit</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Code</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Year</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Semester</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
</tbody>
</table>
### Notices

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size On Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notice-Id</td>
<td>Int(20)</td>
<td>4 bytes</td>
</tr>
<tr>
<td>Title</td>
<td>Varchar(100)</td>
<td>100 bytes</td>
</tr>
<tr>
<td>Beginning date</td>
<td>Date</td>
<td></td>
</tr>
<tr>
<td>Author</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Ending date</td>
<td>Date</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Varchar(500)</td>
<td>500 bytes</td>
</tr>
<tr>
<td>Admin-Id</td>
<td>Int(20)</td>
<td>4 bytes</td>
</tr>
</tbody>
</table>

### Include

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size On Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course name</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Code</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
</tbody>
</table>

### Current Semester

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size On Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sem-no</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Year</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Sub name</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
</tbody>
</table>

### Admin

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size On Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin-ID</td>
<td>Int(20)</td>
<td>4 bytes</td>
</tr>
<tr>
<td>Admin level</td>
<td>Varchar(50)</td>
<td>20 bytes</td>
</tr>
</tbody>
</table>

### Department

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size On Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department-name</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
</tbody>
</table>
### Time table

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size On Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Subject name</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Course name</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Code</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Time</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Admin Id</td>
<td>Int(20)</td>
<td>4 bytes</td>
</tr>
</tbody>
</table>

### Subject list

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size On Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject name</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Course code</td>
<td>Varchar(20)</td>
<td>20 bytes</td>
</tr>
<tr>
<td>Admin-ID</td>
<td>Int(20)</td>
<td>4 bytes</td>
</tr>
</tbody>
</table>

### Control matrix

<table>
<thead>
<tr>
<th>Field</th>
<th>Data type</th>
<th>Size On Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin-ID</td>
<td>Int(20)</td>
<td>4 bytes</td>
</tr>
<tr>
<td>Name</td>
<td>Varchar(50)</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Add users</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Remove users</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Manage users</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>View users Profile</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Remove selected subject from selection</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Create new subjects</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Edit existing subjects</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Delete subjects</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Create new subject lists</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Make adjustments of no of credits</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Subject combination tool</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Time table management</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Create time table</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>Update time table</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>View student choices</td>
<td>Boolean</td>
<td></td>
</tr>
</tbody>
</table>
3.5.6 Triggers

In a DBMS, a trigger is a SQL procedure that initiates an action when an event (INSERT, DELETE or UPDATE) occurs. The SQL CREATE TRIGGER statement provides a way for the database management system to actively control, monitor, and manage a group of tables whenever an insert, update, or delete operation is performed. The statements specified in the SQL trigger are executed each time an SQL insert, update, or delete operation is performed. An SQL trigger may call stored procedures or user-defined functions to perform additional processing when the trigger is executed.

The CREATE TRIGGER statement defines a trigger at the current server.

Ex:

```
CREATE TRIGGER fname_trigger
BEFORE UPDATE ON student_table
REFERENCING NEW ROW AS n, OLD ROW AS o
FOR EACH ROW
IF n.fname <> o.fname THEN
  END IF;
```
3.6 User interface designing

3.6.1 Rules and guidelines for user interface designing

3.6.1.1 User input validation methods

This system will use java script for basic form validations. But for further validations functions we use php. Using php greatly increases security and usability.

JavaScript will detect empty fields of a form, and php will be used to determine whether inputs are met some per-defined rules to insure only desired inputs are stored in the database. Detecting and avoid accepting erroneous inputs before storing in the database, will make system more efficient.

E.g.

Rules to make sure user enters a correct E-mail address:

1. Input e-mail address must contain a @ sign and at least one dot (.)
2. Also, the @ must not be the first character of the email address, and the last dot must be present after the @ sign and minimum 2 characters before the end. Php will be used to ensure that these rules are met, while entering an E-mail address.

Few instances we hope to use validations

1. On all text inputs, check boxes to ensure user fill all the required fields.
2. E-mail address validations (mentioned above)
3. Input date validation, to make sure entered date is not likely to be an unacceptable one.
3.6.1.2 Guideline for error messages, warnings & supportive information.

System will use simple but informative error messages to inform user what is wrong with a certain operation. The type of the error will be stated top of the error message, enabling user to understand quickly what caused the problem.

![Connection Error!](connection-error.png)

System cannot connect to the database to continue. Please make sure your connection is working and retry later.

![Incomplete Response!](incomplete-response.png)

There are some important data fields required incomplete. You need to fill all the fields with an asterisk mark at the right corner.

Warnings will contain the same characteristics of Error messages but will clearly display the type of it. Occupationally system will display some useful tips and trick for the user. This will reduce user's time to figure out everything on their own.

![Warning!](warning.png)

Password is containing part of your name. Please avoid using easily guessed passwords.
A tips will look like Warning.

3.6.1.3 Guideline for Interface designing.

1. Allow users to use either the keyboard or mouse (flexible)

   Users will be able enter data and navigate using any method, either keyboard or mouse. (Scrolling, Selecting)

2. Allow users to change focus (interruptible)

   Users will open several windows once but focus working on only one interface. This will open different interfaces in different web browser tabs, making it easier to change among tasks. (Navigation will be convenient)

3. Display descriptive messages and text (Helpful)

   Tool tips will be displayed to help users to understand further what important operations and buttons do. User will be able to view them when mouse hover above the relevant texts, buttons.

4. Provide immediate and reversible actions, and feedback (forgiving)

   Most of the time users will be given help to revise their selections in the same interface where they make changes.
   E.g. there will be options to remove subjects on the interface where users select their subject list, making it easier to do changes quickly.
5. Provide meaningful paths and exits (navigable)

Finding a relevant interface and exiting from it should be easy and sensible. Operations should be able to understand without much technical knowledge for all users. Users should be able to guess where to find certain interfaces without complex procedures.

6. Accommodate users with different skill levels (accessible)

System will be used by users with many skill levels. And all of them should be able get their job done without specific technical support. This system will be designed with simple interfaces, easy to understand operations and many tool tips. And occasional warnings will be displayed if the users is likely to do a risky operations to aware the user about its danger.

E.g. Users will be displayed warnings when they choose to delete a subject from database or if a student choose to remove a subject and it makes some issue.es. (Not covered an enough credit level)

7. Principals that make the user interface consistent

a. Sustain the context of users’ tasks (continuity)

Users will be provided points of reference as they navigate through a product interface. Window titles, navigation maps and trees, and other visual aids give users an immediate, dynamic view of where they are and where they’ve been.
b. Maintain consistency within and across products (experience)

One of the most important aspects of the interface is to enable users to learn general concepts about system and products and then apply what they've learned to new situations in different programs or different parts of the system. Consistency in behavior means that the way an object works is the same everywhere.

c. Keep interaction results the same (expectations)

Interaction technique consistency is also important. The same shortcut keys should work in similar programs. Consistency in interface behavior is very important. If users experience different results from the same action, they tend to question their own behavior rather than the product's behavior.

d. Provide aesthetic appeal and integrity (attitude)

The software interfaces will be clean and professional looking but also it will not contain any high resource consuming visual effects and tricks. Because as we hope to implement this on a campus server, we try to use minimal resources as much as possible. Otherwise when there are many students are logged into the system, it will not be able to handle all user requests.

e. Encourage exploration (predictable)

Interface will be clean and user friendly as described above, but we try to make all the interfaces and options more predictable and user finds it is fun to use the software as it is a simple. User should be tempted to use new features and explore them, as there are no use of being quiet.
3.6.1.4 Guideline for error messages, warnings & supportive information.

We use simple but informative error messages to inform user what is wrong with a certain operation. The type of the error will be stated top of the error message, enabling user to understand quickly what caused the problem.

(Reference: Golden rules of user interface design by Theo Mandel)
3.6.2 User interfaces for each use case

Use case 01 : Login into the system.

![User Login Form](image)
Use case 02: Change user password

Change Password

Nimal Gunawardana (Administrator)

Current Password
New Password
Retype Password

Cancel  OK
Use case 03: Add users

Add New Users

User name
Password
Retype Password
E-mail address
Account Type: Student

Privileged Level
- Default Level
- Custom Level

Cancel OK

Custom Privilege level

New User (Administrator)
- Add users
- Define tim
- Remove users
- View Student choices
- View accounts
- Dead lines
- Subject op
- Notices

Cancel OK
Use case 04: Remove users

Remove a User

Account Type: Administrator

Available Users:
- Namal Rathnayake (Administrator/Level 2)
- Sarath Nimathan (Administrator/Level 3)
- Kumara Hewage (Administrator/Level 3)

Confirm Action

Remove Namal Rathnayake (Administrator)

Username
Password

[OK] [Cancel]
Use case 05: Manage users

Manage Users

Manage users and privileges

<table>
<thead>
<tr>
<th>Name</th>
<th>Adjust</th>
<th>Suspend account</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nimal Sagaranayaka</td>
<td>Privileges</td>
<td></td>
</tr>
<tr>
<td>Kamal Nirmal</td>
<td>Privileges</td>
<td></td>
</tr>
<tr>
<td>Sumal Gunasekara</td>
<td>Privileges</td>
<td></td>
</tr>
</tbody>
</table>

Privileges matrix

- Access to others
- Manage timetables
- Change credits
- Change subject
- Delete Subjects
- Access resources

[Cancel] [OK]
Use case 06: Edit profile information

Edit Admin Profile

Image

Name
ID
Previlege Level
E-mail
Contact No
Current Password

Change Image
View privileges

Change Password
View

Important Dates

<table>
<thead>
<tr>
<th>Sun</th>
<th>Mon</th>
<th>Tue</th>
<th>Wed</th>
<th>Thu</th>
<th>Fri</th>
<th>Sat</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
</tr>
<tr>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
</tr>
<tr>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

Cancel
Save
Use case 07: Remove selected subjects from student's selections

Remove Subjects

Stream: Physical
Academic year: 2nd Year
Semester: 1st semester

Relevant Subjects

<table>
<thead>
<tr>
<th>Code</th>
<th>Subject</th>
<th>Credits</th>
<th>Remove</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[Buttons: Cancel, OK]
Use case 08: Create new subjects

Add new Subjects

Stream: Physical
Academic year: 2nd Year
Semester: 1st semester
Subject name
Subject Code
No of credits: 4
Subject Coordinator

[Buttons: Cancel, OK]
## Use case 09: Edit existing subjects

### Edit Subject Details

<table>
<thead>
<tr>
<th>Code</th>
<th>Subject</th>
<th>Credits</th>
<th>Coordinator</th>
<th>Delist</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ICT2402</td>
<td>4</td>
<td>Mr. Dulith</td>
<td>Delist</td>
</tr>
<tr>
<td>2</td>
<td>ICT2354</td>
<td>3</td>
<td>Mrs. Sulakk...</td>
<td>Delist</td>
</tr>
<tr>
<td>3</td>
<td>ICT2204</td>
<td>2</td>
<td>Mr. Namal</td>
<td>Delist</td>
</tr>
<tr>
<td>4</td>
<td>COM2000</td>
<td>0</td>
<td>Miss. Piyumi</td>
<td>Delist</td>
</tr>
</tbody>
</table>
Use case 10: Delete subjects

Delete Subject From the System

Stream: Physical
Academic year: 2nd Year
Semester: 1st semester

<table>
<thead>
<tr>
<th>Code</th>
<th>Subject</th>
<th>Credits</th>
<th>Coordinator</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Computer ...</td>
<td>4</td>
<td>Mr. Dulith</td>
</tr>
<tr>
<td>2</td>
<td>Networks</td>
<td>3</td>
<td>Mrs. Sulakk...</td>
</tr>
<tr>
<td>3</td>
<td>Web</td>
<td>2</td>
<td>Mr. Namal</td>
</tr>
<tr>
<td>4</td>
<td>English</td>
<td>0</td>
<td>Miss. Piyumi</td>
</tr>
</tbody>
</table>

- Delete but keep current assignments
- Delete and remove all assignments

[Buttons: Cancel, Delete]
Use case 11: Create new subject lists

Create New Subject List

Stream: ICT
Academic year: 1st Year
Semester: 1st semester

Use subject code to add subjects to the list:

<table>
<thead>
<tr>
<th>Code</th>
<th>Subject</th>
<th>Credits</th>
<th>Coordinator</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICT2405</td>
<td>Network</td>
<td>4</td>
<td>Mr. Nimmel</td>
</tr>
<tr>
<td>Select</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

New
Cancel OK
Use case 12: Make adjustment of credits per subject.

Adjust number if credits per subject

Stream: ICT
Academic year: 1st Year
Semester: 1st semester

Use drop down list to select no of credits

<table>
<thead>
<tr>
<th>Code</th>
<th>Subject</th>
<th>Credits</th>
<th>Coordinator</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICT2405</td>
<td>Network</td>
<td>4</td>
<td>Mr. Nimmel</td>
</tr>
<tr>
<td>ICT2313</td>
<td>Java</td>
<td>3</td>
<td>Mr. Malan</td>
</tr>
</tbody>
</table>

New
Cancel OK
Use case 13: Subject combination tool

Subject Combination tool
Logged as Super User

View privilege Matrix
Conflicting subjects

Add Subject to Matrix

<table>
<thead>
<tr>
<th>Subject</th>
<th>Subject A</th>
<th>Subject B</th>
<th>Subject D</th>
<th>Optional 1</th>
<th>Optional 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject A</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Subject 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Subject B</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Subject D</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notify Changes to Users

Cancel OK

Subject Combination tool
Logged as Super User

View privilege Matrix
Paired Subjects

Add Subject to Matrix

<table>
<thead>
<tr>
<th>Subject</th>
<th>Subject A</th>
<th>Subject B</th>
<th>Optional 1</th>
<th>Optional 2</th>
<th>Optional 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject A</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Subject 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Subject B</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Subject D</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notify Changes to Users

Cancel OK
Use case 14: Publish time table
Use case 15: Update time table

Publish Timetables

Stream: HPT
Year: 1st Year
Semester: 1st Sem

<table>
<thead>
<tr>
<th></th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Row</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9-10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10-11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11-12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Buttons: Edit x-axis, Edit y-axis, Edit Timetable, Cancel, OK
Use case 16: Current subjects by subject vice

View Student's Choices

Student Index No
2785

View
Cancel  OK

View student's choices

<table>
<thead>
<tr>
<th>Code</th>
<th>Subject</th>
<th>Year</th>
<th>Semester</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AS3014</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>COM3000</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>AS3411</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

- Current subjects by subject vice
- Past subject selections by subject vice
- View the past and present subject selections of a student

Cancel  OK
Use case 17: Past subject selections by subject vice

View student's choices

<table>
<thead>
<tr>
<th>Code</th>
<th>Subject</th>
<th>Year</th>
<th>Semester</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Modling</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Probability</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Graphs</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

- Current subjects by subject vice
- Past subject selections by subject vice
- View the past and present subject selections of a student

[Buttons: Cancel, OK]
Use case 18: View the past and present subject selections of a student

## View Student's Choices

<table>
<thead>
<tr>
<th>Code</th>
<th>Subject</th>
<th>Year</th>
<th>Semester</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AS1014</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>COM1000</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>AS2411</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>AS2215</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>AS3412</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

- [ ] Current subject selections by subject vise
- [ ] Past subject selections by subject vise
- [x] View current and past subject selections by subject vise

[Buttons: Cancel, OK]
Use case 19: Edit student's profile

Student Profile

Image

K. Sugathananda (ICT10/11/093)

First Name
Last Name
Academic Year
Index No
Subject Combination
Year
Semester
E-mail
Mobile

Other

I hereby declare that I have fulfilled all the pre requisites to follow these course units as per the agreements of RUSL.
Use case 20: Select subjects

Select Subjects for the Semester

Index No: 2680
Stream: ICT
Academic year: Second Year
Semester: Second Semester

Use drop down list to select Subjects

Networking and Communication (ICT2452)

Advance Networking
Lecturer: Mr. A.M.S Hemantha
Credits: 4
Practicles: 3 Hours per week
Requirements: Credit pass for Networking basics (ICT1024)
Overview: Students will gain the necessary skills to manage a large network and perform advance tasks for implementation and troubleshooting.

Selected Subjects

<table>
<thead>
<tr>
<th>Code</th>
<th>Subject</th>
<th>Credits</th>
<th>Remove</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ICT2405</td>
<td>Networking</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>ICT2313</td>
<td>Java</td>
<td>3</td>
</tr>
</tbody>
</table>

Requirement for minimum Credits per year

24%

- Hereby I declare that I have fulfilled all the pre requisites.
- I am aware of that alternating this selection is only eligible during next 14 days.
- Dead line for changes: 29 November 2013, 12:00 AM

[Add] [Cancel] [OK]
Use case 21: Remove subjects from the list

Remove Subjects

Ki. Sugathananda (ICT10/11/093)

Index No 2680
Stream ICT
Academic year Second Year
Semester Second Semester

Selected Subjects

<table>
<thead>
<tr>
<th>Code</th>
<th>Subject</th>
<th>Credits</th>
<th>Remove</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICT2405</td>
<td>Networking</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ICT2313</td>
<td>Java</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>COM2240</td>
<td>Community</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Requirement for minimum Credits per year: 33%

- Hereby I declare that I have fulfilled all the pre requisites.
- Deadline for changes: 29 November 2013 12:00 AM
Use case 22 : Save

This use case is for save document details on server.

Use case 23 : View previous semester information

View Previous Semesters

Ki. Sugathananda (ICT10/11/093)

<table>
<thead>
<tr>
<th>Index No</th>
<th>Physical</th>
</tr>
</thead>
<tbody>
<tr>
<td>2680</td>
<td>3rd Year</td>
</tr>
<tr>
<td>Stream</td>
<td>Third Year</td>
</tr>
<tr>
<td>Current year</td>
<td>Second Semester</td>
</tr>
<tr>
<td>Current Semester</td>
<td></td>
</tr>
<tr>
<td>GPA</td>
<td>2.15</td>
</tr>
</tbody>
</table>

Select a Previous semester

3rd Year 2nd Semester

<table>
<thead>
<tr>
<th>Code</th>
<th>Subject</th>
<th>Credits</th>
<th>Results</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AS3014</td>
<td>3</td>
<td>B+</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>COM3000</td>
<td>3</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>AS3411</td>
<td>3</td>
<td>Pending</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>AS3215</td>
<td>2</td>
<td>Pending</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>AS3412</td>
<td>4</td>
<td>E</td>
<td>Repeat</td>
</tr>
</tbody>
</table>

All Sementsers

Cancel OK
View Notices

K. Sugathananda (ICT10/11/093)

* You have 2 new Notices

Notice

From - System Administrator  
To - All ICT Students  
2013-Nov-13 05:12pm (26 hours ago)

[Message]

ICT2015 - Ethnic Harmony(non-credit) subject will be replaced by ICT2116 - Professional English and Communication (2 credits) from this year. However may continue this subject for this semester if you wish.

[/Message]
Use case 25: Guest view of the student’s profile

Guest View

Welcome to Sri Lanka Rajarata university Student Database
Login with Public password

Student Index No

Guest Password

Cancel OK
Use case 26: Super user confirmation

Superuser Login

Username

Password

Forgot password/username

Cancel  OK

Superuser Login Confirmation

Confirm Code

Resend Code

Cancel  OK
Use case 27: Access Control Matrix

View privilege Matrix

Account Operations

<table>
<thead>
<tr>
<th></th>
<th>Add Users</th>
<th>Remove Users</th>
<th>Manage Admin</th>
<th>Manage Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Admin B</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Admin C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Admin D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notify Changes to Users

[Cancel] [OK]
Team name : Humming Bird
Team leader : W.D.R.Y. Jayasundara

Members :

<table>
<thead>
<tr>
<th>Name</th>
<th>Department</th>
<th>ID</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y.M.Y.T.K Yaparatne</td>
<td>ICT/10/11</td>
<td>050</td>
<td>2687</td>
</tr>
<tr>
<td>K.Gunawardana</td>
<td>ICT/10/11</td>
<td>008</td>
<td>2640</td>
</tr>
<tr>
<td>K.A.D.C.P. Kaluarachchi</td>
<td>ICT/10/11</td>
<td>057</td>
<td>2650</td>
</tr>
<tr>
<td>S.D. Thrimawithana</td>
<td>ICT/10/11</td>
<td>032</td>
<td>2675</td>
</tr>
<tr>
<td>W.D.R.Y. Jayasundara</td>
<td>ICT/10/11</td>
<td>013</td>
<td>2648</td>
</tr>
</tbody>
</table>