**Software Design Document for**

GradeUP

Version 1.0

Prepared by

|  |  |  |
| --- | --- | --- |
| Akash Ravi | Nikhil Tiware | Parikshit Tiwari |
| Shashank Sharma | Shruti Nafday | Shweta Amla |

A-Team

September 18, 2015

Contents

[1.0 Introduction 6](#_Toc430380893)

[1.1 Goals and objectives 6](#_Toc430380894)

[1.2 Statement of scope 6](#_Toc430380895)

[2. Create a group 7](#_Toc430380896)

[1.3 Software context 8](#_Toc430380897)

[1.4 Major constraints 8](#_Toc430380898)

[2.0 Data design 8](#_Toc430380899)

[2.1 Internal software data structure 8](#_Toc430380900)

[2.2 Global data structure 8](#_Toc430380901)

[2.3 Temporary data structure 9](#_Toc430380902)

[2.4 Database description 9](#_Toc430380903)

[2.4.1 Database System 9](#_Toc430380904)

[2.4.2 Entity-Relationship Diagram 9](#_Toc430380905)

[3.0 Architectural and component-level design 10](#_Toc430380906)

[3.1 System Structure 10](#_Toc430380907)

[3.1.1 Architecture diagram 10](#_Toc430380908)

[3.2.1 Description for Component “Login/Register” 11](#_Toc430380909)

[3.2.1.1 Processing narrative (PSPEC) for component “*Login/Register*” 11](#_Toc430380910)

[3.2.1.2 Component “*Login/Register*” interface description. 11](#_Toc430380911)

[3.2.1.3 Component “*Login/Register*” processing detail 11](#_Toc430380912)

[3.2.2 Description for Component “User Dashboard” 12](#_Toc430380913)

[3.2.2.1 Processing narrative (PSPEC) for component “User Dashboard” 12](#_Toc430380914)

[3.2.2.2 Component “User Dashboard” interface description. 12](#_Toc430380915)

[3.2.2.3 Component “User Dashboard” processing detail 12](#_Toc430380916)

[3.2.3 Description for Component “Group Viewer” 13](#_Toc430380917)

[3.2.3.1 Processing narrative (PSPEC) for component “*Group Viewer*” 13](#_Toc430380918)

[3.2.3.2 Component “*Group Viewer*” interface description. 13](#_Toc430380919)

[3.2.3.3 Component “*Group Viewer*” processing detail 13](#_Toc430380920)

[3.2.4 Description for Component “Schedule Meeting” 14](#_Toc430380921)

[3.2.4.1 Processing narrative (PSPEC) for component “*Schedule Meeting*” 14](#_Toc430380922)

[3.2.4.2 Component “*Schedule Meeting”* interface description. 14](#_Toc430380923)

[3.2.4.3 Component “*Schedule Meeting*” processing detail 14](#_Toc430380924)

[3.2.5 Description for Component “Notifications 15](#_Toc430380925)

[3.2.5.1 Processing narrative (PSPEC) for component “Notifications” 15](#_Toc430380926)

[3.2.5.2 Component “Notifications” interface description. 15](#_Toc430380927)

[3.2.5.3 Component “Notifications” processing detail 15](#_Toc430380928)

[3.2.6 Description for Component “Discussion Portal” 16](#_Toc430380929)

[3.2.6.1 Processing narrative (PSPEC) for component “Discussion Portal” 16](#_Toc430380930)

[3.2.6.2 Component “Discussion Portal” interface description. 16](#_Toc430380931)

[3.2.6.3 Component “Discussion Portal” processing detail 16](#_Toc430380932)

[3.2.7 Description for Component “User Schedule Handler” 16](#_Toc430380933)

[3.2.7.1 Processing narrative (PSPEC) for component “User Schedule Handler” 16](#_Toc430380934)

[3.2.7.2 Component “User Schedule Handler” interface description. 16](#_Toc430380935)

[3.2.7.3 Component “User Schedule Handler” processing detail 17](#_Toc430380936)

[3.2.8 Description for Component “User Credential Verifier” 17](#_Toc430380937)

[3.2.8.1 Processing narrative (PSPEC) for component “User Credential Verifier” 17](#_Toc430380938)

[3.2.8.2 Component “User Credential Verifier” interface description. 17](#_Toc430380939)

[3.2.8.3 Component “User Credential Verifier” processing detail 17](#_Toc430380940)

[3.2.9 Description for Component “Scheduler” 18](#_Toc430380941)

[3.2.9.1 Processing narrative (PSPEC) for component “Scheduler” 18](#_Toc430380942)

[3.2.9.2 Component “Scheduler” interface description. 18](#_Toc430380943)

[3.2.9.3 Component “Scheduler” processing detail 18](#_Toc430380944)

[3.2.10 Description for Component “Notifications Handler” 19](#_Toc430380945)

[3.2.10.1 Processing narrative (PSPEC) for component “Notifications Handler” 19](#_Toc430380946)

[3.2.10.2 Component “Notifications Handler” interface description. 19](#_Toc430380947)

[3.2.10.3 Component “Notifications Handler” processing detail 19](#_Toc430380948)

[3.2.11 Description for Component “Group Request Handler” 20](#_Toc430380949)

[3.2.11.1 Processing narrative (PSPEC) for component “*Group Request Handler*” 20](#_Toc430380950)

[3.2.11.2 Component “*Group Request Handler*” interface description. 20](#_Toc430380951)

[3.2.11.3 Component “*Group Request Handler*” processing detail 20](#_Toc430380952)

[3.2.12 Description for Component “Discussion Handler” 20](#_Toc430380953)

[3.2.12.1 Processing narrative (PSPEC) for component “Discussion Handler” 20](#_Toc430380954)

[3.2.12.2 Component “Discussion Handler” interface description. 21](#_Toc430380955)

[3.2.12.3 Component “Discussion Handler” processing detail 21](#_Toc430380956)

[3.2.13 Description for Component “Discussion Threads” 21](#_Toc430380957)

[3.2.13.1 Processing narrative (PSPEC) for component “Discussion Threads” 21](#_Toc430380958)

[3.2.13.2 Component “Discussion Threads” interface description. 21](#_Toc430380959)

[3.2.13.3 Component “Discussion Threads” processing detail 21](#_Toc430380960)

[3.2.14 Description for Component “User Portfolio” 22](#_Toc430380961)

[3.2.14.1 Processing narrative (PSPEC) for component “User Portfolio” 22](#_Toc430380962)

[3.2.14.2 Component “User Portfolio” interface description. 22](#_Toc430380963)

[3.2.14.3 Component “User Portfolio” processing detail 22](#_Toc430380964)

[3.2.15 Description for Component “Group Portfolio” 23](#_Toc430380965)

[3.2.15.1 Processing narrative (PSPEC) for component “Group Portfolio” 23](#_Toc430380966)

[3.2.15.2 Component “Group Portfolio” interface description. 23](#_Toc430380967)

[3.2.15.3 Component “Group Portfolio” processing detail 23](#_Toc430380968)

[3.2.16 Description for Component “Meeting Calendar” 23](#_Toc430380969)

[3.2.16.1 Processing narrative (PSPEC) for component “Meeting Calendar” 23](#_Toc430380970)

[3.2.16.2 Component “Meeting Calendar” interface description. 24](#_Toc430380971)

[3.2.16.3 Component “Meeting Calendar” processing detail 24](#_Toc430380972)

[3.3 Dynamic Behavior for Component n 24](#_Toc430380973)

[3.3.1 Interaction Diagrams 24](#_Toc430380974)

[4.0 User interface design 24](#_Toc430380975)

[4.1.1 Screen images 24](#_Toc430380977)

[4.2 Interface design rules 34](#_Toc430380978)

[4.3 Components available 36](#_Toc430380979)

[5.0 Restrictions, limitations, and constraints 37](#_Toc430380980)

[6.0 Testing Issues 38](#_Toc430380981)

[6.1 Classes of tests 38](#_Toc430380982)

[6.3 Performance bounds 40](#_Toc430380983)

[6.4 Identification of critical components 41](#_Toc430380984)

[7.0 Appendices 42](#_Toc430380985)

[7.1 Requirements traceability matrix 42](#_Toc430380986)

[A matrix that traces stated components and data structures to software requirements will be updated at a later stage. 42](#_Toc430380987)

[7.2 Packaging and installation issues 42](#_Toc430380988)

[7.3 Design metrics to be used 42](#_Toc430380989)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# 1.0 Introduction

1. The purpose of this document is to provide an insight into the design aspects of the GradeUP application. The topics covered in this document are:

* Architectural Design
* Database implementation
* User interface design
* Class hierarchies
* Design Constraints
* Test Cases

1. This document is intended for both the stakeholders and the developers of the system.

## 1.1 Goals and objectives

The GradeUP android application is being designed to ease the user’s efforts in studies. The intended users for this application are students who need guidance of any sort on study topics but are not well versed with networking and sharing knowledge on an open platform. The application explores few of the networking possibilities by connecting one student to other students with some level of expertise by creating study groups. Also, it provides an opportunity to the users to participate in open discussion forums, group discussions and organize small group meetings seamlessly.

## 1.2 Statement of scope

It is targeted for anyone who wants to collaborate & share knowledge efficiently by the means of study groups but are facing scheduling problems due to conflicting timetables. The application helps all users in a group find a common time slot during which they are all free to meet for study. It helps the user in finding friends who are willing to help with the specific subjects and form study groups to simplify the process of studying. It also aims at providing a forum for general subject related discussions. The application can be used to effectively manage the group meetings. This will reduce the effort required in getting help for completing a course or subject and would ideally help improve the grade of a student who uses it and makes it easy for anyone trying to learn any subject. It will help the user finish their coursework in an organized manner. Thus, the application has universal scope and can be used by any student. The long term vision is to make every student’s life easier and help them get organized.

**1.2.1 ESSENTIAL REQUIREMENTS:**

The following includes the list of features we wish to implement in the course of three sprints:

1. **Register/Login/Logout**

The Register/Login/Logout feature allows the user to register/login/logout of the Application. The user will have to enter their first name, last name, email ID, field of study and contact number.

## 2. Create a group

This feature allows the user to create a new group. When a user creates a group he/she automatically becomes the admin for that group

1. **Join/Leave a group**

This feature allows the user to join an existing group. A user can also leave the group that they are a part of.

1. **View my Dashboard**

The dashboard screen is a central location where the user can see his/her schedule for the day, the list of groups the user is a part of and the general discussion portal.

1. **Free time in schedule**

The user on entering a group should be able to see the combined schedule of the group and

free time available to all.

1. **Schedule/Cancel meetings**

The user should be able to schedule a meeting with a particular study group based in the

common free time available to all.

1. **Setup Agenda for meetings**

The user should be able to setup an agenda for the meeting

1. **Start/Comment discussion within the group**

The user should be able to start a discussion thread, where he/she can post doubts and

queries within a group. Other members in the group should be able to respond to the

queries.

1. **Start/comment on public discussion thread**

The user should be able to start a public discussion thread where he/she would be able to

post any doubt or queries. Other users should be able to see these threads and be able to

comment on them.

1. **Update user profile**

The Update Profile feature allows users to update their personal information including their schedule and study record(s).

## 

**1.2.2 DESIRABLE REQUIREMENTS**

This includes the list of features that we would like to include if at all time permits:

1. **SMS and email Reminder**

The user should receive an SMS and email reminder on their registered phone number and email ID 15 minutes before the start of the meeting.

1. **Block social media notification**

For the duration of the study group meeting social media notifications should be blocked to the group members from getting distracted from studying.

## 1.3 Software context

The software is developed as a project, which is part of the Computer Science Graduate level coursework. This app will be made public for no extra cost through the android play store. No commercial release of this software for business purposes is intended as of now.

## 1.4 Major constraints

Due to lack of experience of team members with android development environment, the learning curve will be steep. Thus the team members will need to learn the new technology as well as progress with the app development in a limited timeframe. As a consequence, the delivery of certain application features may be delayed due to a hard deadline. This might result in a reduced number of features being delivered, with the core features untouched.

# 2.0 Data design

## 2.1 Internal software data structure

The gradeUP application has a server and a client side as mentioned in the architecture.

The client side has temporary variables that will be used to send and temporarily store the data from the server side. The user data and session information will be stored in the cache in the client side.

The server side has temporary variables just like the client side that is used to store temporarily and send data to the client side.

## 2.2 Global data structure

All the global data in the application will be stored in the form of tables in a Relation Database and SQLite will be used to manage this database system. The tables will be related to each other through keys and will ensure that all the data can be efficiently retrieved through queries. All the data fetched from the database will be stored in temporary variables and the variables used to insert data into it are temporary and these will expire once the user is logged out. The data will be fetched and inserted into the database using SQL queries through JDBC (Java Database Connectivity).

## 2.3 Temporary data structure

Temporary data involves all the variables that are created to insert data into the database from the UI or to fetch data to be populated onto the view. All the variables will have only temporary data. The data will be converted to JSON for ease of transmission between the client-server and vice versa.

* All variables fetched from the database will be converted into JSON format, sent to the client and then finally be converted into the format required before projecting onto the view
* The data to be inserted into the database will be converted into JSON in the front end, will be transmitted through JDBC and will be split into the required type and format before insertion

All the above mentioned data in the variables will be lost once the particular function terminates.

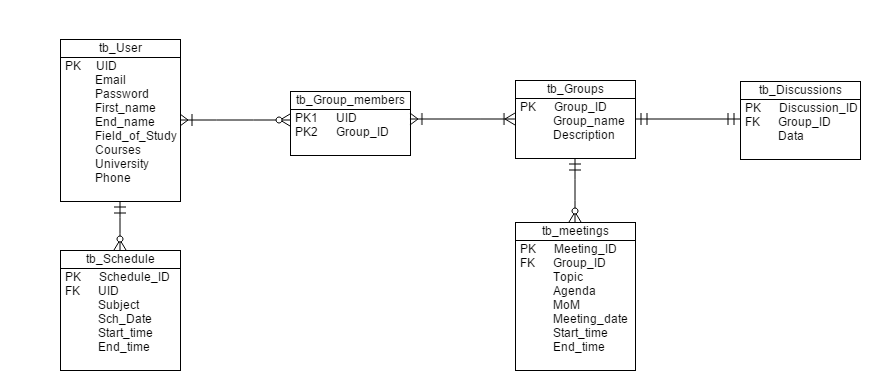
## 2.4 Database description

This section gives an insight into the database structure being used in the application project and elaborates on the different parts of the ERD, known as Entity-Relationship Diagram, which is a visual representation of the relationship between various database entities.

### 2.4.1 Database System

The GradeUP application will be using SQLite database for storing all the application and user data. SQLite is an in-process library that implements a self-contained, transactional SQL database engine with the biggest advantage of having an open source code. Apart from this, SQLite is a Relational Database Management System and thus makes a good platform for learning SQL which will stand beneficial for the development team.

### 2.4.2 Entity-Relationship Diagram



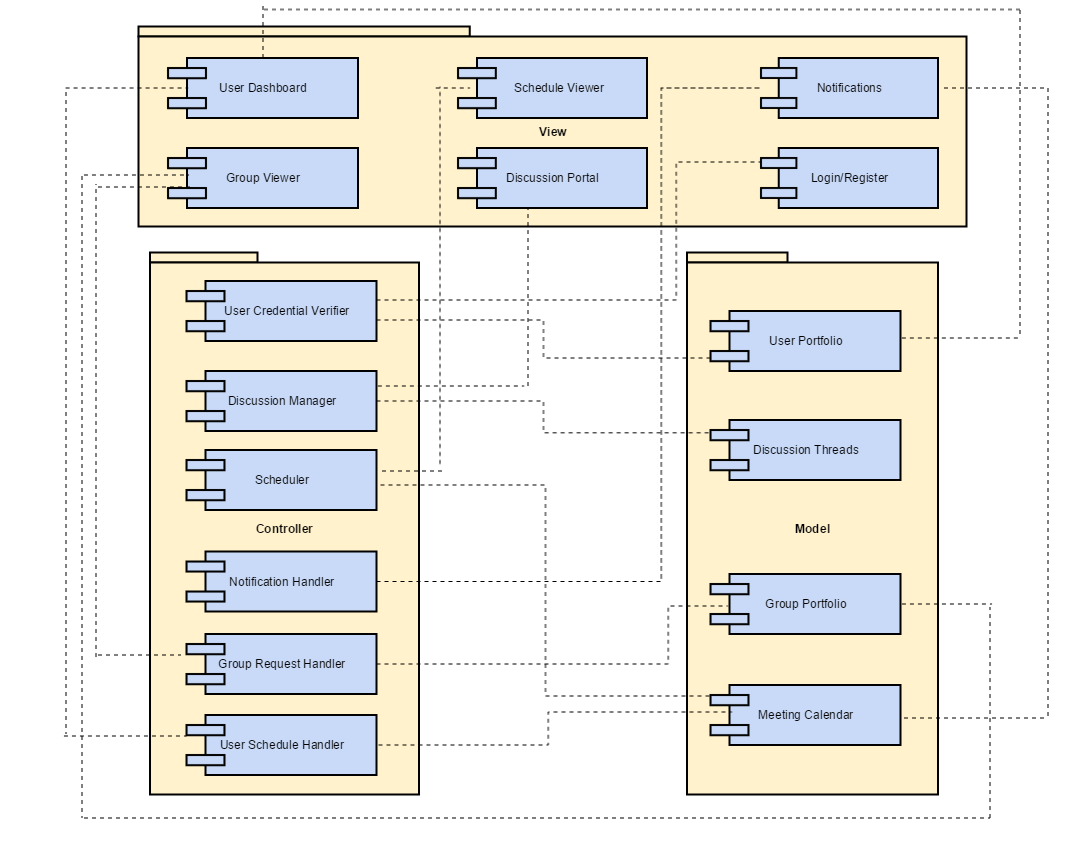
# 3.0 Architectural and component-level design

A detailed description of the program architecture is presented here.

## 3.1 System Structure

The GradeUP application system structure is based on MVC design structure. MVC stands for Model-View-Controller. Essentially, it separates the application data structure, code logic and interfaces design from one another. The main advantage being that any minor changes in one of the three does not affect the other two.

### 3.1.1 Architecture diagram



MVC Component Diagram

## 3.2.1 Description for Component “Login/Register”

This component deals rendering initial user interfaces meant for creation of user accounts for the application. If user account is already created it will ask for the login credentials and in case of wrong credential gives error or provides options to retrieve password.

### 3.2.1.1 Processing narrative (PSPEC) for component “*Login/Register*”

The object of the class takes username and password to match it with Database entry and if matched it will grant access and transfer to login the application. Also password is always displayed as encrypted. And at registration it takes following details from user and if same user name is used again displays the error.

1. First name
2. Last name
3. Field of study
4. Password
5. Confirm password

### 3.2.1.2 Component “*Login/Register*” processing detail

Algorithm to ensure encryption of password while displayed on screen.

#### 3.2.1.3.1 Restrictions/limitations for component Login/Register

User can not register with different usernames from same device as could cause issue while interacting with main server.

#### 3.2.1.3.2 Performance issues for component Login/Register

Very basic data is rendered on the screen, so no performance issue.

#### 3.2.1.3.3 Processing detail for each operation of component Login/Register

Login: Object of this type takes user name and password at login screens, matches it in DB and returns true if matched successfully.

Register: Object of this type takes user basic data, put it in database and returns true if user successfully created.

Forgot Password: Creates object that will ask for your user name and send password to your registered email-id.

## 3.2.2 Description for Component “User Dashboard”

User Dashboard is used for rendering following user data on screen:

1. User Picture
2. User Name
3. University Name
4. User Daily Schedule
5. List of groups that user is part of.
6. Discussion Portal
7. Update Profile
8. Update Schedule

### 3.2.2.1 Processing narrative (PSPEC) for component “User Dashboard”

User dashboard shows renders all the users information in one place. User can update its profile, check its schedule, and browse the groups he/she is member of. Also user can go through the news feed and post over it.

### 3.2.2.2 Component “User Dashboard” interface description.

NA

### 3.2.2.3 Component “User Dashboard” processing detail

This component simply displays user data, so no algorithm.

#### 3.2.2.3.1 Design Class hierarchy for component User Dashboard

No class hierarchy is their but it will association with groups and discussion portal.

#### 3.2.2.3.2 Restrictions/limitations for component User Dashboard

It will display only 5 groups at a time on dashboard rest can be browsed using scroll bar. Similarly last couple of news feed will be displayed based on size.

#### 3.2.2.3.3 Performance issues for component User Dashboard

Simple rendering of data, no complex operations to undergo, so performance issue.

#### 3.2.2.3.4 Processing detail for each operation of component User Dashboard

1. User Basic Info: Displays user name, picture and university name.

2. Schedule: Displays user schedule by taking data from user schedule handler.

3. Group list: Get group list from user portfolio to display in scroll menu on dashboard

4. Update Schedule: Users can update their schedules by selecting start time and end time, or week days in case of periodic schedule.

5. Discussion Portal: Displays data retrieved from discussion manager

6. Update profile: Provides option for updating user data like phone number and email Id

## 3.2.3 Description for Component “Group Viewer”

This component renders following data related to groups:

1. Group Name
2. Group members
3. Group meetings
4. Group search
5. Join group
6. Discussions

### 3.2.3.1 Processing narrative (PSPEC) for component “*Group Viewer*”

It renders data like group name, group members, group meetings using group portfolio. Also it displays group discussions and provide option to post to group discussion. This also renders different groups available to join.

### 3.2.3.2 Component “*Group Viewer*” interface description.

NA

### 3.2.3.3 Component “*Group Viewer*” processing detail

Algorithms to ensure user get to see only the group it is a member.

#### 3.2.3.3.1 Design Class hierarchy for component Group Viewer

Group viewer has association with schedule viewer to set meeting with members of the group.

#### 3.2.3.3.2 Restrictions/limitations for component Group Viewer

User cannot see data of group of whom he is not a member.

#### 3.2.3.3.3 Performance issues for component Group Viewer

Very basic data is rendered on the screen, so no performance issue.

#### 3.2.3.3.4 Processing detail for each operation of component Group Viewer

Display Group: Displays groups basic information like name, members, meetings

Join Group: Provides list of groups available in the application to user for joining.

Request Viewer: Group admin to get option to allow users become member of the group.

Discussions: All the discussions related to the group will be displayed here in chronological order.

## 3.2.4 Description for Component “Schedule Meeting”

This component renders data to select a meeting time with the group members based on best possible time interval when no one has any prior commitments and set then set that meeting for the group.

### 3.2.4.1 Processing narrative (PSPEC) for component “*Schedule Meeting*”

Using this screen, user will be given options of best time possible between all the group members and set it.

#### 3.2.4.3.1 Design Class hierarchy for component Schedule Meeting

Group viewer has association with scheduler to compute the best time slot available for meeting.

#### 3.2.4.3.2 Restrictions/limitations for component Schedule Meeting

Meeting can be scheduled at intervals of 30 minutes.

#### 3.2.4.3.3 Performance issues for component Schedule Meeting

Very basic data is rendered on the screen, so no performance issue.

## 3.2.5 Description for Component “Notifications

Notification renders information like meeting scheduled for the group, request for joining group has been accepted, meeting is cancelled and reply received on discussion portal

### 3.2.5.1 Processing narrative (PSPEC) for component “Notifications”

Pop up the shown to the user when any of following activity happens over the application:

1. Meeting scheduled for the group
2. Request for joining group has been accepted
3. Meeting is cancelled
4. Reply received on discussion portal

#### 3.2.5.2.1 Restrictions/limitations for component Notifications

It will display only 5 latest notification and rest can be seen by scrolling down.

#### 3.2.5.2.2 Performance issues for component Notifications

Simple rendering of data, no complex operations to undergo, so performance issue.

## 3.2.6 Description for Component “Discussion Portal”

This will render discussion going on in the group and as well on the main screen by fetching data from Discussion Handler.

### 3.2.6.1 Processing narrative (PSPEC) for component “Discussion Portal”

At user dashboard and group page discussion will be displayed with few latest at a time and a text box to revert on these discussion. These discussions would be synced across all users every second.

#### 3.2.6.2.1 Design Class hierarchy for component Discussion Portal

It is composition of User Dashboard and Group Viewer.

#### 3.2.6.2.2 Restrictions/limitations for component Discussion Portal

It will display only few latest news feed and rest can be seen by scrolling down. Maximum data posted in one post will also be defined.

#### 3.2.6.2.3 Performance issues for component Discussion Portal

Simple rendering of data, no complex operations to undergo, so performance issue.

## 3.2.7 Description for Component “User Schedule Handler”

User Schedule Handler provides the algorithms to convert data entered by while updating schedule to be further added to the database and vice versa.

### 3.2.7.1 Processing narrative (PSPEC) for component “User Schedule Handler”

User Schedule Handler takes input from User Dashboard and converts it into dates over the calendar and send it further to the Meeting Calendar to be furthered stored into the database. Also do vice versa for sending it display over the screen.

### 3.2.7.2 Component “User Schedule Handler” interface description.

NA

### 3.2.7.3 Component “User Schedule Handler” processing detail

Algorithm for converting periodic schedule into the dates over the calendar.

#### 3.2.7.3.1 Design Class hierarchy for component User Schedule Handler

No class hierarchy is there.

#### 3.2.7.3.2 Performance issues for component User Schedule Handler

As data is efficiently stored using efficient data structure and database, so no performance issue.

#### 3.2.7.3.3 Processing detail for each operation of component User Schedule Handler

Convert Schedule: Convert the periodic schedule entered by the user into dates over the calendar based on start date and end date.

Update Schedule: Update user schedule by adding it to the database using Meting Calendar.

## 3.2.8 Description for Component “User Credential Verifier”

This component verifies the password entered by user matches with one stored in database and also provides option to change the password and send it to user’s email id in case he forgets

### 3.2.8.1 Processing narrative (PSPEC) for component “User Credential Verifier”

It takes input from login screen decrypts it and check if the password and email matches using User Portfolio and send the return back. If new password is set it encrypts it, and further adds to database using user portfolio. In case user forgets the password, it decrypts it and further send it to the email-Id.

### 3.2.8.2 Component “User Credential Verifier” interface description.

NA

### 3.2.8.3 Component “User Credential Verifier” processing detail

Algorithm for encryption and decryption of password it will be implemented.

#### 3.2.8.3.1 Design Class hierarchy for component User Credential Verifier

No class hierarchy is there.

#### 3.2.8.3.2 Performance issues for component User Credential Verifier

As data is efficiently stored using efficient data structure and database, so no performance issue.

#### 3.2.8.3.3 Processing detail for each operation of component User Credential Verifier

Match password: Match the password and user name with the database.

Decrypt password: Decrypt password to send it to email-Ids

Encrypt password: At time of new password is set, its encryption occurs to be further stored into database.

## 3.2.9 Description for Component “Scheduler”

It calculates the common time slots available for all the members of the groups when meeting has to be scheduled.

### 3.2.9.1 Processing narrative (PSPEC) for component “Scheduler”

Scheduler receives data from the Meeting Viewer to providing the choice of common time slots, it takes schedule of all the users from Meeting Calendar. And provides common time slot available according to the time requested.

#### 3.2.9.2.1 Restrictions/limitations for component Scheduler

If few users are not available in the slot request, it will show that option with conflict of that user.

#### 3.2.9.2.2 Performance issues for component Scheduler

As data is efficiently stored using efficient data structure and database, so no performance issue.

## 3.2.10 Description for Component “Notifications Handler”

Notification Handler sends information to Notifications if any such activity happens like meeting scheduled for the group, request for joining group has been accepted, meeting is cancelled and reply received on discussion portal

### 3.2.10.1 Processing narrative (PSPEC) for component “Notifications Handler”

If keeps on polling the calendar and send data to Notification if any of following activity happens over the application:

1. Meeting scheduled for the group
2. Request for joining group has been accepted
3. Meeting is cancelled
4. Reply received on discussion portal

Triggers SMS and Emails before meetings

### 3.2.10.2 Component “Notifications Handler” processing detail

It keeps on polling the Meeting Calendar, and if any meeting is scheduled it sends the notification. Also triggers SMS and Emails before the scheduled meeting.

#### 3.2.10.2.3 Performance issues for component Notifications Handler

As data is efficiently stored using efficient data structure and database, so no performance issue.

## 3.2.11 Description for Component “Group Request Handler”

This component ensures that user can request for the group and group admin can approve or disapprove of the request.

### 3.2.11.1 Processing narrative (PSPEC) for component “*Group Request Handler*”

When user search groups, it will provide all the group available to user to join. Once request is made, it will provide notification to group admin to approve the request. And if user disapproves the request, user cannot see group data.

#### 3.2.11.2.1 Design Class hierarchy for component Group Request Handler

Group Request Handler has association with Group Viewer to search and join groups.

#### 3.2.11.2.2 Restrictions/limitations for component Group Request Handler

User cannot see data of group of whom he is not a member.

#### 3.2.11.2.3 Performance issues for component Group Request Handler

Very basic data which is efficiently stored, so no performance issue.

## 3.2.12 Description for Component “Discussion Handler”

It stores the discussion going on in the group and as well on the main screen by fetching data from Discussion Handler. Also keep all the users in sync.

### 3.2.12.1 Processing narrative (PSPEC) for component “Discussion Handler”

Every message post by user over the Discussion Portal is stored in the Database via Discussion Threads. And the data posted by other users is also synchronized latest data would be kept in Discussion Threads

#### 3.2.12.2.2 Restrictions/limitations for component Discussion Handler

Maximum data posted in one post will be defined.

#### 3.2.12.2.3 Performance issues for component Discussion Handler

As data is efficiently stored using efficient data structure and database, so no performance issue.

## 3.2.13 Description for Component “Discussion Threads”

It database class that stores and retrieves data corresponding to discussions in the database.

### 3.2.13.1 Processing narrative (PSPEC) for component “Discussion Threads”

Every message post by user over the Discussion Portal is stored in the Database via Discussion Threads.

#### 3.2.13.2.1 Performance issues for component Discussion Threads

As data is efficiently stored using efficient data structure and database, so no performance issue.

#### 3.2.13.2.2 Design constraints for component Discussion Threads

NA

## 3.2.14 Description for Component “User Portfolio”

It database class that stores and retrieves data corresponding to users in the database.

### 3.2.14.1 Processing narrative (PSPEC) for component “User Portfolio”

It will store all the user data:

1. User Picture
2. User Name
3. University Name
4. User Daily Schedule
5. List of groups that user is part of.
6. Discussion Portal

#### 3.2.14.2.1 Performance issues for component User Portfolio

As data is efficiently stored using efficient data structure and database, so no performance issue.

## 3.2.15 Description for Component “Group Portfolio”

It database class that stores and retrieves data corresponding to groups in the database.

### 3.2.15.1 Processing narrative (PSPEC) for component “Group Portfolio”

It will store all the group data:

1. Group Picture
2. Group Name
3. Members Name
4. Group Schedule
5. Discussion Portal

#### 3.2.15.2 Performance issues for component Group Portfolio

As data is efficiently stored using efficient data structure and database, so no performance issue.

## 3.2.16 Description for Component “Meeting Calendar”

It database class that stores and retrieves data corresponding to meetings of groups and schedule of user in the database.

### 3.2.16.1 Processing narrative (PSPEC) for component “Meeting Calendar”

When user updates schedule it will be added to the Meeting calendar as date further entered into database. Also when group meeting is schedules that’s also added to it and further to database.

#### 3.2.16.1.2 Performance issues for component Meeting Calendar

As data is efficiently stored using efficient data structure and database, so no performance issue.

## 3.3 Dynamic Behavior for Component n

A description of the interaction of the classes is presented.

### 3.3.1 Interaction Diagrams

A sequence diagram, for each use case the component realizes, is presented.

# 4.0 User interface design

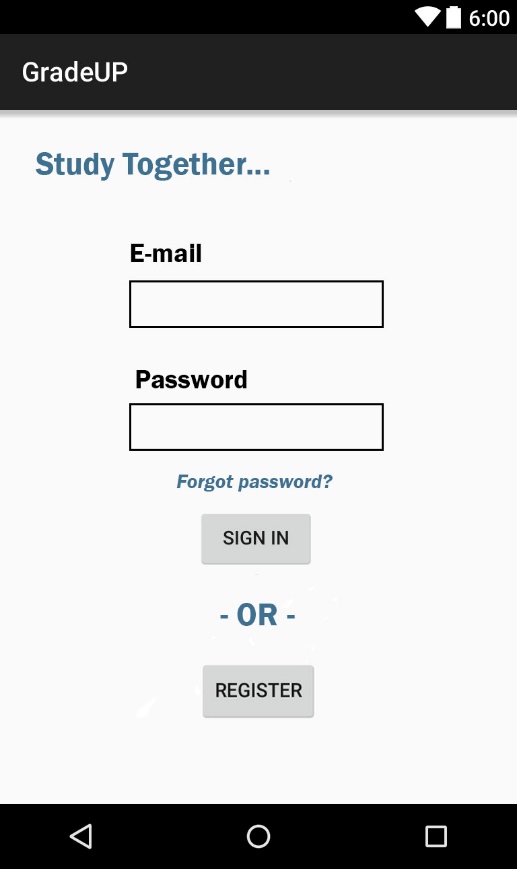
## The user interface of the GradeUP application consists of a series of menus and screens which will make the use of the application simple and intuitive. The screens are as follows. The “Login” screen, “Registration” screen, “User Dashboard” screen, “Sidebar” menu, “Add Schedule” screen, “Create Group” screen, “Group Homepage” screen, “Join Group” screen, “Schedule Meeting” screen and “Discussion Forum” 4.1 Description of the user interface

A detailed description of user interface including screen images or prototype is presented.

### 4.1.1 Screen images

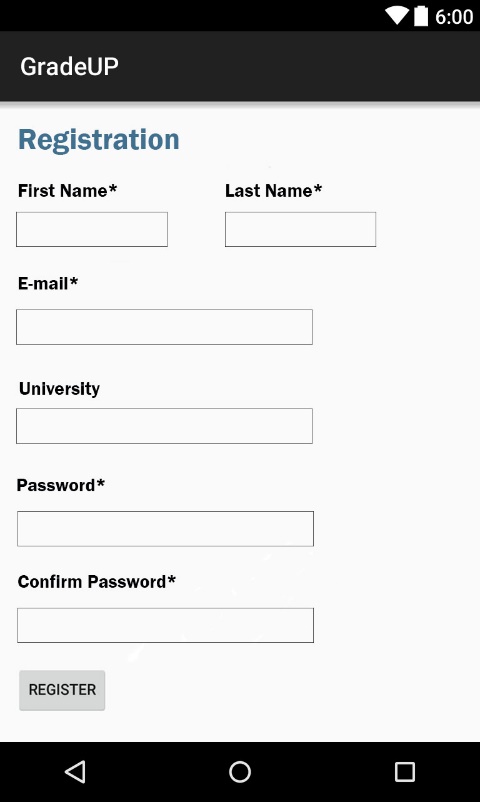
The following section describes the various screen the user will see and the action he/she will be able to perform

LOGIN PAGE



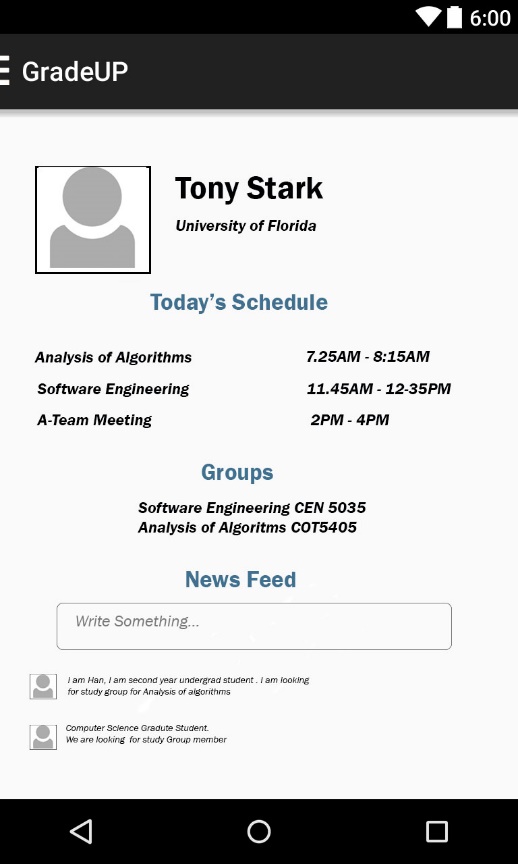
* The first screen the user will see when entering the application will be the Login Page. In here first time user will have to register themselves in to the application by clicking on the “Register” button. This will redirect the users to the “Registration” screen.
* Returning users will have to enter their Email Id and password in the textboxes provided and then click on ‘Sign In” button. If the email id and password combination is correct the user will be redirected to User DashBoard screen
* If the user does not enter the valid Email Id and password combination a error message saying “Please enter correct Email Id or Password”

REGISTRATION



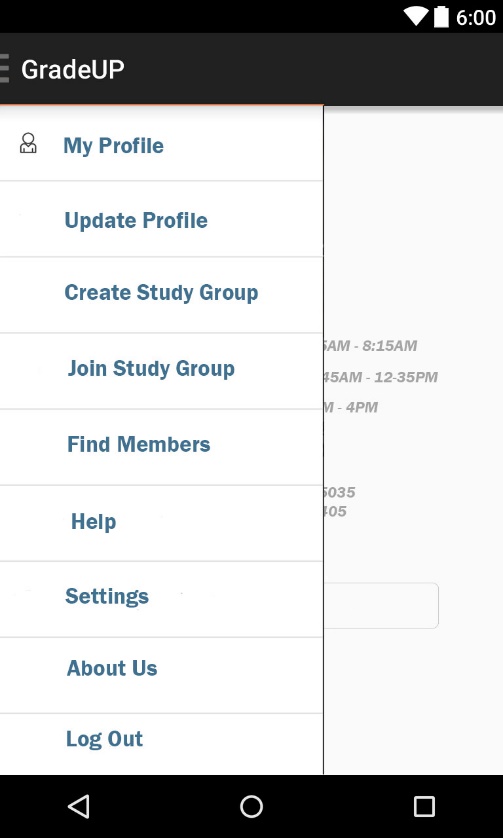
* A first time user has to be registered and will be redirected to a registration page where the user will be asked to enter his/her first name, last name, field of study, a password and confirm password.
* If the user enters an incorrect email id then an error message saying “Enter a valid Email Id”
* The user will not be allowed to register if the Email ID is invalid and if the Password and Confirm Password does not match
* If the information is validated the user will click REGISTER and be redirected to the “User Dashboard” screen

USER DASHBOARD



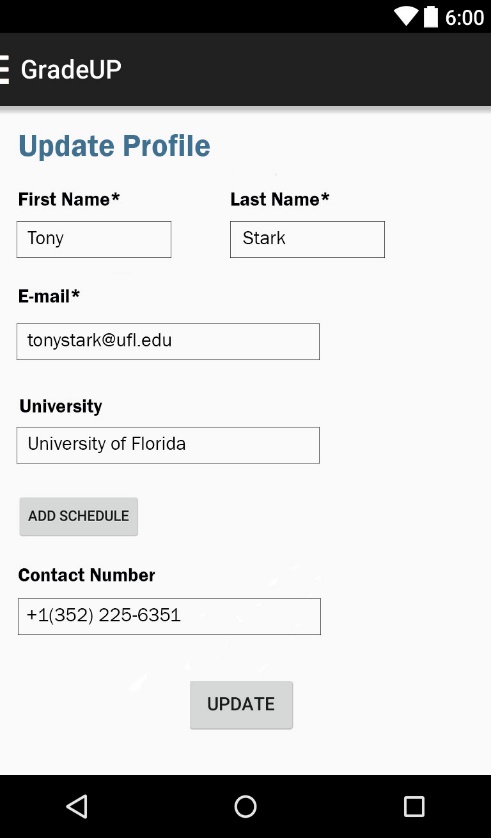
* On the dashboard the user will be able to see his/her schedule the groups the user is a part of and the news feed.
* The user will be able to navigate to his/her groups list by clicking on the Groups button

SIDEBAR



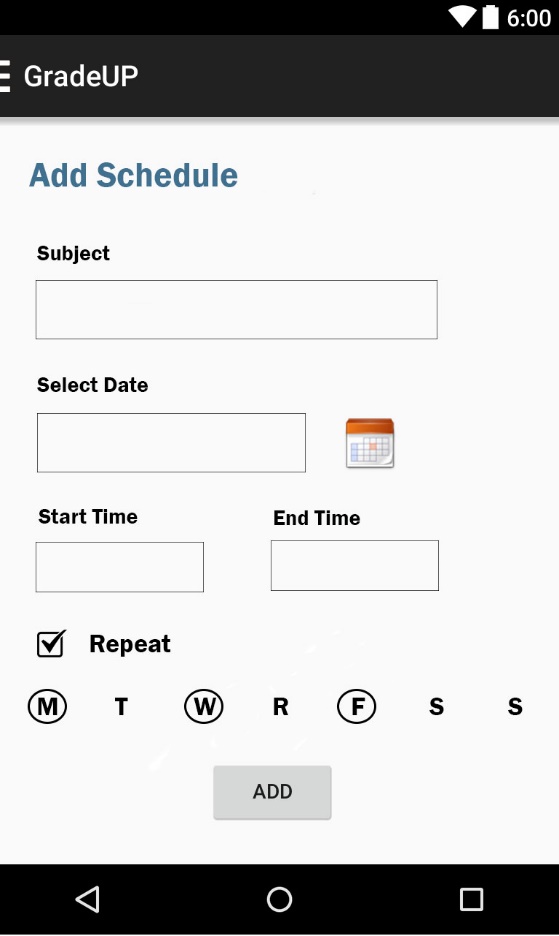
* The Sidebar is a menu which the user can see from any page by swiping to the right, from this menu the user can navigate to the “My Profile”, “Update Profile”, “Create Study Group”, “Join Study”, “My Groups”, “Help”, “Settings”, “About Us” and “Logout”
* The user will be able navigate to any of these pages by clicking on them.

UPDATE PROFILE



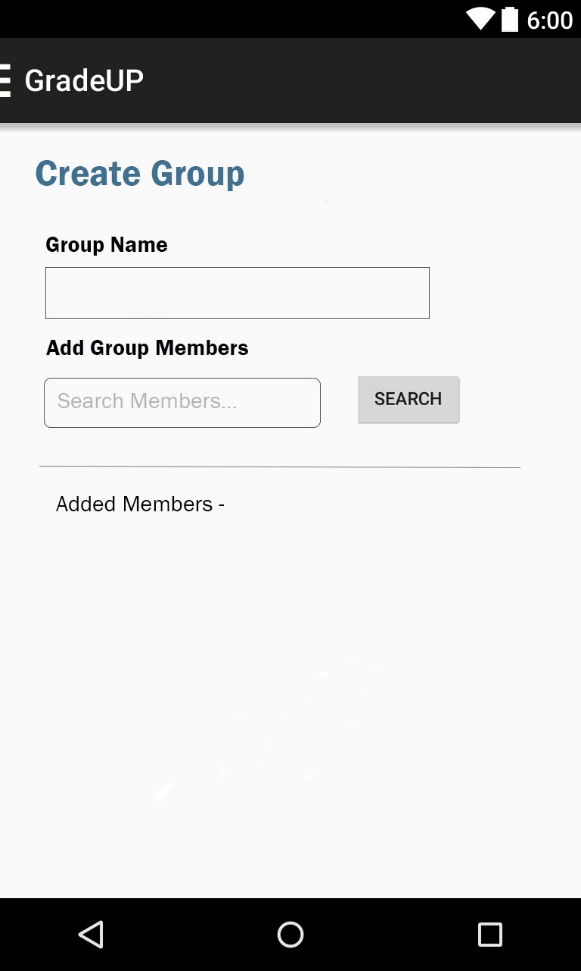
* The user can update his/her profile by navigating to the Update Profile screen from the sidebar. Here the user can update his/her personal information, such as first name, last name, email id, university, course, schedule and contact number.
* The email id will have to be a valid email id. If the user enters an invalid email address, an error message will be displayed saying, “Please enter a valid Email ID”
* When the user clicks on the UPDATE button his/her profile will be updated

ADD SCHEDULE



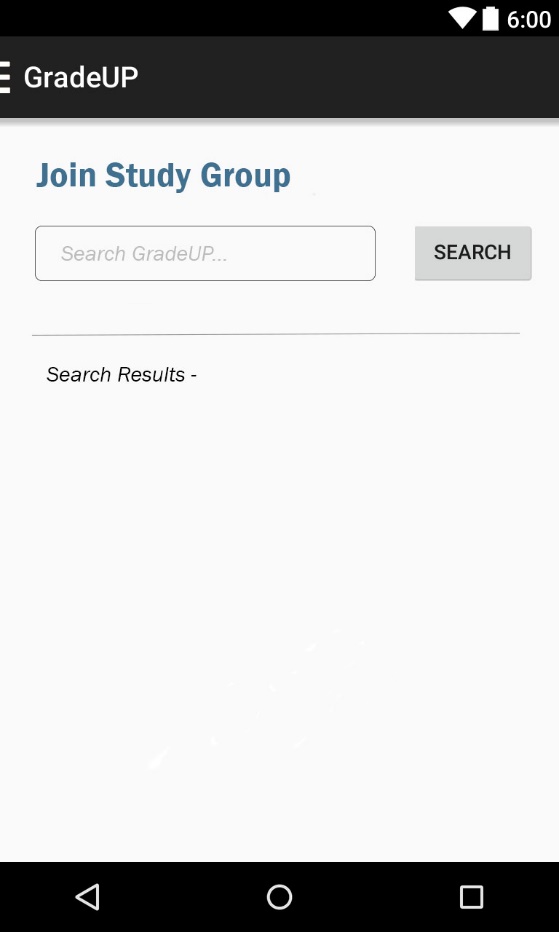
* On the “Add Schedule” screen the user will be able to enter his/her schedule. The page allows the time slot to be given a title. A date field will allow the user select a date.
* The user will click on the Start Time and End Time dropdown to select his time slot. The dropdown will contain time on the 24 hour format in half hour intervals.

CREATE GROUP



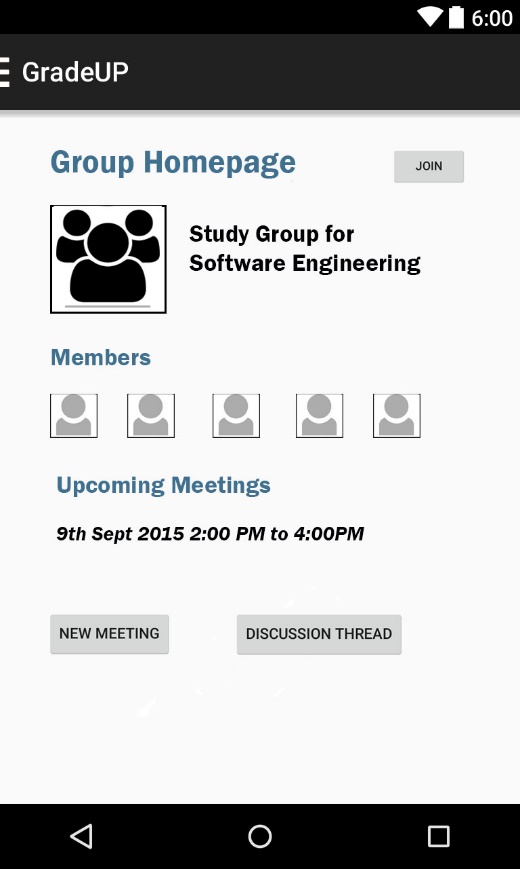
* On the Create Group page the user will be able to create a page. In the group name field the user will be asked to enter a unique group name. A message saying please enter a unique group name will be shown
* The user can also add other users to the group using the email id to search.
* The group will be created on click of the CREATE GROUP, when at least one other user other than the creator is added to the group and if the group has a unique name.

JOIN GROUP



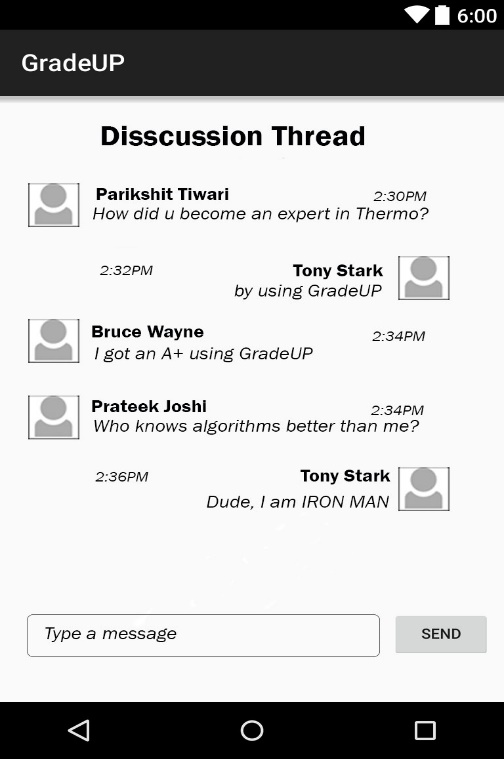
* On the Join Group page the user will enter group names and click search. The search results will be returned in a list. The user will then click on the group name and be redirected to the Group Homepage.

GROUP HOMEPAGE



* On the group homepage the user will be able to see the members present in the group, the group schedules and upcoming meetings.
* The user can join the group by clicking the JOIN button. On click of the JOIN button a request is sent to the Admin of the group, who approves or denies the request.
* Once a member joins the group, the user will be able to see the NEW MEETING and DISCUSSION THREADS buttons. The new meeting button will allow the user to create a new meeting based on the free time shown in the GROUP SCHEDULE page
* On click of the DISCUSSION THREAD the user will be redirected to the DISCUSSION PAGE.

DISCUSSION THREAD



* The DISCUSSION THREAD screen is a place where people can post their queries or comments by typing a message in the text area and hitting send.

## 4.2 Interface design rules

The interface design rules for GradeUP are derived from the following Rules of Interface Design, some of them being derived from Ben Shneiderman’s “Eight Golden rules of Interface design”. Here’s a description of each rule, as well as how the rule applies to GradeUP.

* **Strive for consistency**

Consistent sequences of actions should be required in similar situations. Identical terminology should be used in prompts, menus, and help screens, and consistent commands should be employed throughout. GradeUP inherits this in the form of standard sidebar menu, common help screen, standard user prompts in the form of screen popups etc.

For each screen in GradeUP: all of the major and frequently used options are displayed in the dynamic sidebar menu such as My Profile, Update Profile, Create/Join Study Group(s), Logout etc

* **Offer Informative feedback**

While interacting with the application, for every valid gesture of the user while accessing any of the GUI components, such as buttons, there should be a system feedback. In GradeUP, the user gets a system feedback only for major operations such as Login, Logout, Create Group, Leave group etc. Actions such as view group or view profile will simply redirect the user to the desired pages, in this case Group list and profile dashboard respectively.

* **Offer simple error handling**

As much as possible, the system should be designed to prevent chances of any fatal error caused by the user. However, if an error is made, the user system should immediately be prompted with a message stating an easy to implement solution to the error. GradeUP will give display an error message if the user tries to perform an illegal operation.

* **Permit easy reversal of actions**

This feature provides the user a sense of confidence in a way that in case an unwanted action is performed, it can be reverted. As an example, in GradeUP, in case a user accidently sends a request for getting added to a group, the request can be cancelled using the cancel request option.

* **Cater to universal usability**

Recognize the needs of diverse users and design for plasticity, facilitating transformation of content. Novice to expert differences, age ranges, disabilities, and technological diversity each enrich the spectrum of requirements that guides design. Adding features for novices, such as explanations, and features for experts, such as shortcuts and faster pacing, can enrich the interface design and improve perceived system quality. GradeUP will be easy and intuitive to use irrespective of expertise, age or any other such factor.

* **Design dialogs to yield closure**

Sequences of actions should be organized into groups with a beginning, middle, and end. Informative feedback at the completion of a group of actions gives operators the satisfaction of accomplishment, a sense of relief, a signal to drop contingency plans from their minds, and an indicator to prepare for the next group of actions.

The GradeUP application will give users feedback when actions such as Create Group, Exit group, join group, create profile, update profile, schedule meeting and join meeting are completed.

* **Prevent errors**

As much as possible, design the system such that users cannot make serious errors; for example, gray out menu items that are not appropriate and do not allow alphabetic characters in numeric entry fields. If a user makes an error, the interface should detect the error and offer simple, constructive, and specific instructions for recovery.

GradeUP will provide users with feedback whenever they make an error, usually in the form of an error message being displayed on the screen which instructs them on how to correct the error.

* **Support internal locus of control**

Experienced users strongly desire the sense that they are in charge of the interface and that the interface responds to their actions. They don’t want surprises or changes in familiar behavior, and they are annoyed by tedious data-entry sequences, difficulty in obtaining necessary information, and inability to produce their desired result.

Data entry work will be limited to competition of registration and at the time of entering schedules.

* **Reduce short-term memory load.**

Humans’ limited capacity for information processing in short-term memory (the rule of thumb is that we can remember "seven plus or minus two chunks" of information) requires that designers avoid interfaces in which users must remember information from one screen and then use that information on another screen. It means that cell phones should not require re-entry of phone numbers, web-site locations should remain visible, multiple-page displays should be consolidated, and sufficient training time should be allotted for complex sequences of actions.

Users will not be required to remember any data other than their email id and password at the time of login.

## 4.3 Components available

The following GUI components provided by Android will be used by GradeUP

* **View**

This class represents the basic building block for user interface components. A View occupies a rectangular area on the screen and is responsible for drawing and event handling.

* **Button**

Buttons will allow users to trigger on click event

* **TextView**

The TextView allows the application to display static text

* **ListView**

The ListView allows the application to display a list of data, such as Group Names, Members etc.

**4.4 UIDS Description**

Android Studio provides an in-built User Interface Design System. Android Studio allows you to create User Interface from a graphical drag and drop designer. The UIDS generates XML code from the drag drop design. The XML code can be written directly to form the UI but that might prove time consuming. The XML is code is linked to a Java class where we can insert logic for events.

# 5.0 Restrictions, limitations, and constraints

The major limitation of GradeUp is the time constraint. Also, due to lack of experience of team members with android development environment, the learning curve will be steep. Thus the team members will need to learn the new technology as well as progress with the app development in a limited timeframe. As a consequence, the delivery of certain application features may be delayed due to a hard deadline.

A software limitation of GradeUp is that it is restricted to Android platform. Users with Windows and iOS phones would not be able to access this app. However once the application is developed on the Android platform it could easily be implemented on other platforms and also on web interface, currently this is out of scope of our project.

Another constraint is that this application requires the user to have internet access. In case of an internet connection outage, the application features will be partially available but no new data updates will happen during this time.

Also the users need to have a valid email address to use the application.

# 6.0 Testing Issues

Test strategy and preliminary test case specification are presented in this section.

## 6.1 Classes of tests

**6.1.1 White Box Testing**

The developers are expected to perform white box testing throughout the development. Developers will be expected to test each and every class they have written to ensure that individual modules of the code are working.

**6.1.2 Black Box Testing**

Black box testing is done by a tester who has no knowledge of the internal specification and working of the code. The code will be tested once it has been entirely assembled

**6.1.3 Feature Testing**

Individual Feature will be tested in the following way

* + User Registration
    - All fields marked with \* are required. If any such field is empty then user should not be allowed to register.
    - Numeric characters are not allowed in the First Name and Last Name field
    - Email id should contain only one @ character and should be a valid email address
    - User should not be allowed to proceed unless Password and Confirm Password fields match
    - Password length should be between 6-12 characters
  + Login Page
    - Both email id and password field are required fields.
    - Email id should contain only one @ character and should be a valid email address.
    - User should only be allowed to Login if a correct combination of email id and password are entered.
    - Password length should be between 6-12 characters.
    - Application should navigate to the Dashboard page when correct email id and password are entered and login is clicked.
    - Forgot password should navigate to appropriate page.
  + User Dashboard
    - The user’s First Name, Last Name, University, Course, Schedule, Groups and News feed should be accurately displayed for that user
    - On click of the Group name the user should get redirected to that group’s homepage
  + Group Homepage
    - The groups details like Groupname, Members list, Upcoming meetings should be accurately displayed
    - User’s request to join the group should be sent to the Admin alone when user clicks the Join button.
    - On click of New meeting the user should get redirected to the Schedule Meeting page
    - On click of Discussion Thread the user should get redirected to the Discussion Thread page
    - If the user is a part of the group then the Join button toggles to the Leave button
    - On click of Leave button the user should be exited from the group and his/her schedule should be removed from the group
  + Schedule Meeting
    - Subject is a required field
    - The meeting date cannot be an invalid or past date
    - On click of the calendar icon a calendar widget should open, from which the user should be able to select a date.
    - On date selection available time slots for that date only should be visible.
    - The start time and end time entered should not allow to enter time period outside the suggested slots.
    - When the user clicks the schedule button the meeting should be scheduled and the user should be redirected to the Group homepage.
  + Discussion Thread
    - The message entered by the user in the textbox should be displayed on the forum, when the user clicks send.
    - Send button should be disabled when the textbox is empty
  + Update Profile
    - All fields marked with \* are required. If any such field is empty then user should not be allowed to update his/her profile.
    - Numeric characters are not allowed in the First Name and Last Name field
    - Email id should contain only one @ character and should be a valid email address
    - Clicking on the Add Schedule button should redirect to Add Schedule page.
    - Clicking on the update button should update the user profile
  + Add Schedule
    - Title is a required field
    - The meeting date cannot be an invalid or past date
    - On click of the calendar icon a calendar widget should open, from which the user should be able to select a date.
    - When the user clicks the schedule button the meeting should be scheduled and the user should be redirected to the Group homepage.
    - If the repeat option is selected the particular time slot should be repeated for that time slot on the days selected by the user.
  + Create Group
    - The Group name is a required field. An error message will be displayed if the user hits Create group button with empty group name
    - The user should be able to search other users based on a valid email id only
    - If the search members textbox is empty the Search button should be disabled
    - At least one member must be added before the group can be created
    - When a member is searched and selected that member should be visible in the Added Members list
    - On click of Create Group button the group should be created and the user should be redirected to the Group Homepage
  + Join Study Group
    - The user should be able to search groups based on group name
    - If the search textbox is empty the Search button should be disabled
    - When a group is searched and selected that group should be visible in the Search Results list
    - When the Search Result is clicked the user is redirected to the Group Homepage
  + Sidebar
    - When the user swipes right on any screen the Sidebar should be displayed. It should show the “My Profile”, “Update Profile”, “Create Study Group”, “Join Study Group”,” My Groups,” “Help”, “Settings”, “About Us” and “Logout” options
    - On click of the option the user should be redirected to that particular page.

.

## 6.3 Performance bounds

The execution time of the application should be negligible. Interaction time between the mobile application and the database should be small. With the current scope of the project we do not expect to be supporting more that 50-60 users.

## 6.4 Identification of critical components

The most critical component of the application is the interval tree algorithm which will calculate the free time period between the various users.

# 7.0 Appendices

Presents information that supplements the design specification.

## 7.1 Requirements traceability matrix

## A matrix that traces stated components and data structures to software requirements will be updated at a later stage.

## 7.2 Packaging and installation issues

There are no special considerations for software packaging and installation for this application. The user can download and install the application package from Google play store once it is made available.

## 7.3 Design metrics to be used

Complexity and goodness of the design will be measured through stability and flexibility of the application which is ensured on developer end as the application is based on MVC-Model View Controller design architecture. As mentioned earlier, MVC architecture helps developer to only change parts of the design model without any requirement to alter the complete design.