- Description
- Intended User
- Features
- User Interface Mocks
  - Screen 1
  - Screen 2
  - Screen 3
  - Screen 4
  - Screen 5
  - Screen 6
  - Screen 7
  - Screen 8
  - Screen 9
- Key Considerations
  - How will your app handle data persistence?
  - Describe any corner cases in the UX.
  - Describe any libraries you'll be using and share your reasoning for including them.
  - Describe how you will implement Google Play Services.
- Next Steps: Required Tasks
  - Task 1: Project Setup
  - Task 2: Implement UI for Each Activity and Fragment
  - Task 3: Implement Montserrat font
  - Task 4: Modify build.gradle
  - Task 5: Create Activities and Fragments
  - Task 6: Define Menus, Strings, Dimensions, Logs and other Layouts
  - Task 7: Create AsyncTasks
  - Task 8: Create Parcelable data models and Recycler View Adapters
  - Task 9: Verify data and create tests

GitHub Username: devNiharika

Galgotias University mSIM

Description

Galgotias University mSIM is a simple to use app for students which enables them to be able to check important information such as their attendance, books issued from the university library within seconds.
The app also gives general information about the university and the courses it offers.

Intended User

This app is intended for all the students of Galgotias University currently enrolled in a full time or part time course. This app can also be used by guest visitors for knowing about the university and the programs offered by it.

Features

The main features of the app are:

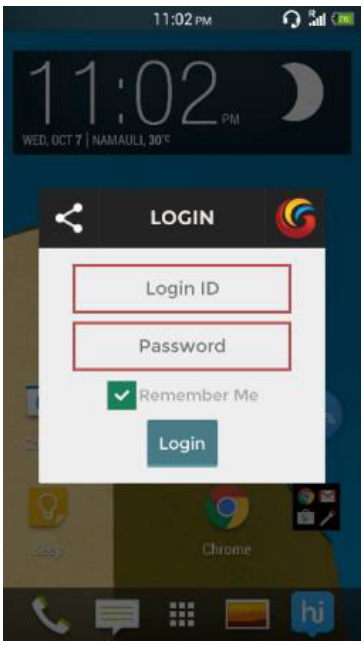
- Students can view their attendance easily. Attendance can be viewed both Date-wise and Subject-wise
- Students can plan their Holidays with the help of Holiday Planner (Attendance Calculator) and easily calculate the number of classes they must attend
- Students will also get alerts if their attendance falls below the minimum attendance criteria defined by the University norms
- Students can also view the books they have issued from the library with the due date of each book along with the library fine if any
- Students can also view their profile and personal details as registered with the University
- Visitors or Guest users can get information about courses the University offers along with route details

User Interface Mocks

Screen 1

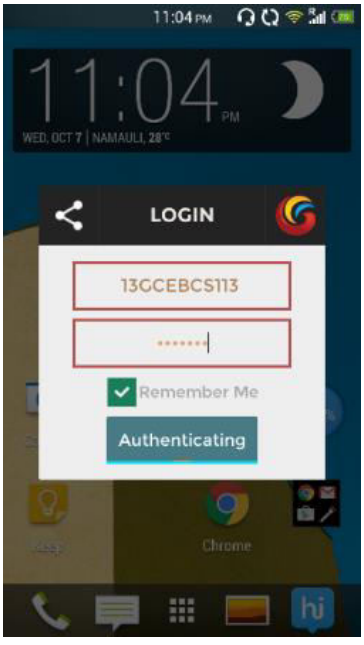
Launch screen with High Quality Logo

Screen 2



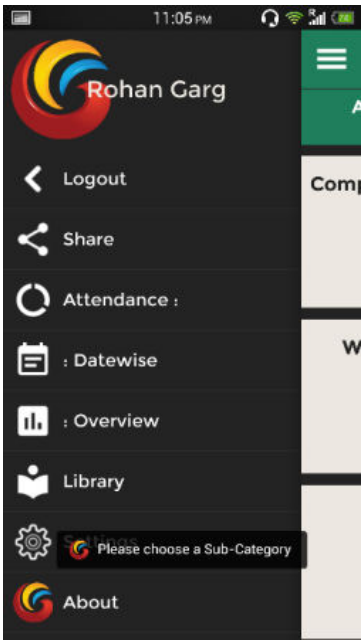
Secure Login screen where students can enter their credentials

Screen 3



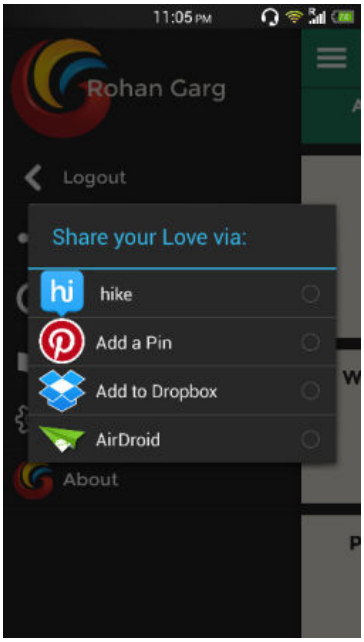
Authentication of student's credentials

Screen 4



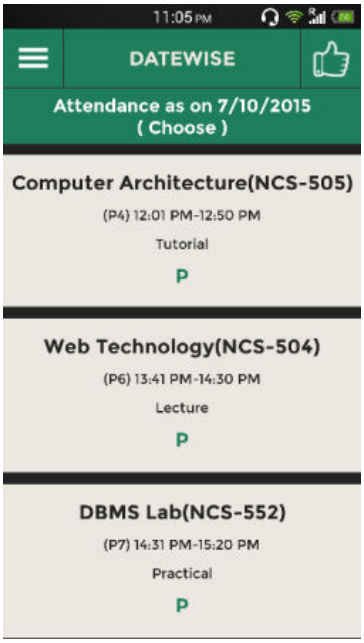
Navigation Drawer for easy in app navigation

Screen 5



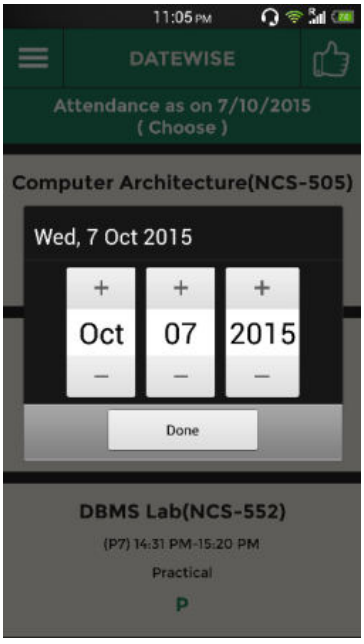
Easy app sharing options

Screen 6



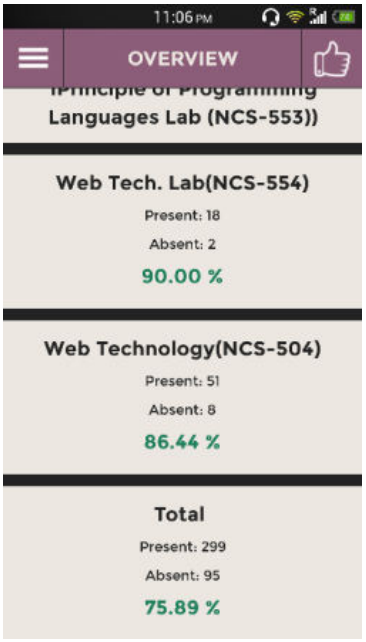
Date-wise attendance with a scrollable list

Screen 7



Date picker to choose the date for which students want to view their attendance

Screen 8



Overview aka Subject wise attendance

Screen 9

Library information with details of books issued if any

Key Considerations

How will your app handle data persistence?

The app would be connected to the existing database of the University with a read only permission which would provide all the required information for the app. Static of less frequently updated items such as Name, Email ID etc. would be stored in Shared Preferences.

Describe any corner cases in the UX.

There can be a corner case when the user rotates his device while data is being fetched because of AsyncTask's natured of being tied with the activity. Hence rotation would be temporarily disabled while the details are being fetched. In case the user rotates his device while no data is being fetched savedInstanceState would be used.

Describe any libraries you'll be using and share your reasoning for including them.

Third party libraries used would be:
OkHttp for fetching content
Jsoup for parsing content
BetterPickers calendar library for providing a calendar picker
ChangeLog library to share the improvements being done with time
UpdateAlertBox for alerts on update of app being available
ProgressButton for indicating loading action
WaveProgress for displaying percentages in graphical form
A scroll aware Floating Action Button for a good reading experience
and ButterKnife for reduction of boiler plate code and performance improvements

Describe how you will implement Google Play Services.

Following components of Google Play Services would be implemented:
AdMob for integration of Ads.
Maps for sharing of location and nearby shops via Google maps. This would make it easy for new students and visitors to navigate the University campus.

Next Steps: Required Tasks

Task 1: Project Setup

- Configure libraries
- Import required modules: ChangeLog, ProgressButton, VersionManager and WaveProgress

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
- Build UI for Library fragment
- Build UI for Profile fragment
- Build UI for Attendance fragment
- Build UI for rows of RecyclerView
- Build UI for About activity

Task 3: Implement Montserrat font

Define a class MyApp.java which extends “android.app.Application” for implementing custom fonts on app start.

- Import Montserrat Bold and Regular to assets folder

Task 4: Modify build.gradle

- Use pro-guard
- Create build types
- Set minifyEnabled and shrinkResources true for release build type and false for debug build type
- Define API keys for Google Play Services

Task 5: Create Activities and Fragments

Define activities and tie them to their corresponding UI.

- Define Activities in Manifest.xml
- Handle corner cases
- Handle data persistence
- Handle errors

Task 6: Define Menus, Strings, Dimensions, Logs and other Layouts

- Define Navigation Drawer menu
- Define Toolbar menu
- Define Change Log
- Define strings in String.xml
- Define dimensions in dimens.xml with support for tablets
- Add animation xmls in anims folder
- Define colors.xml and styles.xml for a uniform colour scheme

Task 7: Create AsyncTasks

Fetch data using AsyncTasks.

- Define AsyncTask for loading Captcha
- Define AsyncTask for getting session id from server
- Define AsyncTask for loading Day by Day attendance
- Define AsyncTask for loading Subject-wise attendance
- Define AsyncTask for loading Monthly attendance
- Define AsyncTask for loading Library details
- Define AsyncTask for loading Profile details

Task 8: Create Parcelable data models and Recycler View Adapters

Display data using recycler view and define data models for easy parsing of data.

- Build Adapter for Day by Day attendance
- Build Adapter for Subject-wise attendance
- Build Adapter for Monthly attendance
- Build Adapter for Library details

Task 9: Verify data and create tests

Verify the data being fetched from the server and create tests to ensure the app does not crash at any given point of time.