Chapter 1: INTRODUCTION

Android application that can remind us of any event/Task we mark on it as a public event or private event. It can be like reminding us about a meeting/event/any function etc. The app can take from the user the event name, date, time, location, venue longitude and latitude of the event. Store the details to the database and the tab of the notification will active before the few hour of the event, on that period creator of that event or we can say that admin can send notification to the their subscriber of event. When event got created user can check it's location in maps. Events venue will be used to serach in google maps for user help to find the correct location for event. Application will be used as a "reminder" for a specific type of event with the ability to save multiple events. Events are based on alternating calendar dates & frequency (user input). Notification/reminders will be sent via Text Message.

Whether you are a home user, student or professional, it is always important to keep and organize important information, dates and events. TimeKeeper is mobile reminder application that helps you manage different kinds of information with ease and efficiency. Time Keeper comes with lots of useful features, allowing you to save your time, increase productivity and focus on your job.

All the information related to event reminder stores for you are classified depending on how it relates to a time. For example, if you have a meeting on Monday at 11:00 AM then you should set up a reminder one day before, which will give remainder before your meeting in the form of message. If you have a deadline, you set up a reminder to allow you to do the work in a timely fashion.

There are two types of event public event and private event. The above scenario is related to private event where as public event is a event which can be subscribed by many users. For example it can be a workshop, to which students can subscribe and app will give remainder to each and every student who had subscribed to the same event in the form of message.

App also have feature of timer in which we can reset timer as well as set the particular time from which it can start in the form of (minutes and second). And it also

have feature of Stopwatch with the help of which user can calculate the time of any activity. And it also have an welcome dashboard, i.e profile window where the name of user is their along with the number of the events and his subscribed event. We have also designed module for a Time management tips, in which we add some tips from internet. So it can be helpful for user of the application to understand the importance of time.

1.1 Project Objectives:

Objective of this app to provide simple interface for reminders of various technical as well as non technical events. You can customize the duration, multiple selections, update the events and etc.

Some Objectives is as Follow:

1) Manage all your reminders.
2) Create event
3) Generate and fulfill your idea about events
4) Plan your time.
5) Search the location of event.
6) Save Time.

Chapter 2: PROBLEM DEFINITION

2.1 Problem Statement:

In today's fast paced life, keeping track of all the events which has travel schedules is a tedious task. Though there are number of event reminder and calendar applications available for android, none of them supports dynamic reminders based on user's current location. All the current available applications give an event reminder before a fixed amount of time from the event start time. Notification will also include details of event for users to start. The application has been implemented using Android and its services for the front end. No other app provide the feature to notify users when event creator wants to. So in timekeeper app we decided to provide the functionality for public event creator to send notification to subscribed users. Notifications can be send in one click when action creator want to notify to all subscribed event.

To make this app more efficient for user we decide to add all time related functionality in single app. This app has facility for user to use timer. In case, user want to use timer facility for some event he can use it from same app instead of using inbuilt timer. It will reduce app's switch timing. User might need Stopwatch facility in same app. this will provide all this functionality in one app. This app also maintains user profile information and can identify or search the location of event from the Google map.

Chapter 3: REQUIREMENT SPECIFICATIONS

3.1 Software Specifications:

1. Front End: Android
2. Back end: Firebase
3. IDE: Android Studio
4. Operating system: windows 7/8/10, linux etc
5. Documentation: Ms-Office

3.2 Hardware Specifications:

3.2.1Windows requirements

- Microsoft Windows 7/8/10 (32-bit or 64-bit)
- 3 GB RAM minimum, 8 GB RAM recommended (plus 1 GB for the Android Emulator)
- 2 GB of available disk space minimum, 4 GB recommended (500 MB for IDE plus 1.5 GB for Android SDK and emulator system image)
- 1280 x 800 minimum screen resolution

3.2.2Linux OS requirements

- GNOME or KDE desktop. Tested on Ubuntu 14.04 LTS, Trusty Tahr (64-bit distribution capable of running 32-bit applications)
- 64-bit distribution capable of running 32-bit applications
- GNU C Library (glibc) 2.19 or later
- 3 GB RAM minimum, 8 GB RAM recommended (plus 1 GB for the Android Emulator)
- 2 GB of available disk space minimum, 4 GB recommended (500 MB for IDE plus
- 1.5 GB for Android SDK and emulator system image)

3.3 Android Platform:

Android software development is the process by which new applications are created for devices running the Android operating system. Google states that "Android apps can be written using kotlin, Java, and C++ languages" using the Android software development kit (SDK), while using other languages is also possible. All non-JVM languages, such as Go, JavaScript, C, C++ or assembly, need the help of JVM language code, that may be supplied by tools, likely with restricted API support. Some programming languages and tools allow cross-platform app support (i.e. for both Android and iOS). Third party tools, development environments, and language support have also continued to evolve and expand since the initial SDK was released in 2008. In addition, with major business entities like Walmart, Amazon, and Bank of America eyeing to engage and sell through mobiles, mobile application development is witnessing a transformation.The official Android app distribution mechanism to end users is Google Play; it also allows staged gradual app release, as well as distribution of pre-release app versions to testers.

3.4 Firebase:

Firebas**e** is a Backend-as-a-Service BaaS that started as a YC11 startup and grew up into a next-generation app-development platform on Google Cloud Platform. Firebase frees developers to focus crafting fantastic user experiences. You don't need to manage servers.

The firebase realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data. Firebase Storage provides secure file uploads and downloads for Firebase apps, regardless of network quality. The developer can use it to store images, audio, video, or other user-generated content.

3.5 Non-Functional Requirements

Usability:

The app's UI should be user friendly. Concerning from User's point of view, it should be understandable. All menus must have a consistent format.

Security:

All passwords, sensitive credentials should in a safe storage

Reliability:

The app should be available 100% of the time. The app should never crash

Localizability:

The app's language must be US English so that it can be universally Acceptable.

Compatibility:

The app must be compatible with at least Android and iOS Smartphones

Performance:

At any point, the app must be able to perform to its 100%.

Chapter 4: FEASIBILITY STUDY

A feasibility study was an evaluation of a proposal designed to determine the difficulty in carrying out a designated task. Generally, a feasibility study precedes technical development and project implementation.

4.1. Economic Feasibility:

To develop the proposed system, it needs no extra facilities and devices. All dependencies are satisfied from the open source projects. All tools used are free, open source and the programming language is Java and JavaScript and hence its development is economically.

4.2. Technical Feasibility:

Proposed system is technically feasible because the proposed system requires only those H/W and S/W tools that are available in the system. It requires the installation of Andriod SDK and Firebase which can be done for free. More over expandability will be maintained in the new system. New modules can be added later on the application, if required in the future. Additionally, the application will have User-friendly Forms, extra API and Screens.

4.3. Behavioral Feasibility:

Behavioral feasibility determines how much effort will go in the proposed information system, and in educating and training the users on the new system. Since the user interface is very simple and easily understandable, no training is required for using this software.

Chapter 5: SYSTEM DESIGN

5.1 Modules:

The proposed system is divided into eight different modules as follows:

- Login & User Authentication
- Registration form
- Calendar
- Daily View
- Timer
- Stopwatch
- Profile
- Time management Tips

5.1.1 Login & User Authentication:

This screen is shown first when user opens the app, if user is not logged in. If user is logged in then the user is redirected to the calendar view. From here user can do three actions:

1. Login
2. Go to Registration page
3. Reset Password

1. Login:

User can Login using Email and password. The Authentication API used here is the Google Firebase Email-Password Authentication API. It is safer to use the Firebase API in this project so that we can leverage Google's own security mechanisms for data security. To login user should enter Email and Password and press the login button, if login is successful user is redirected to Calendar page if not then appropriate errors are shown.

2. Go to Registration page:

To go to Registration page user should click the text view at bottom with text "Register here" in it. This would be useful for unregistered users.

3. Reset Password:

Clicking on the reset password link will send the password reset link to the user's email automatically, and let them change their password.

5.1.2 Registration Form:

This screen is shown to users if they navigate here using the go to Registration link of login page. From here user can do Two Actions :

1. Register
2. Go back to login page

1. Register:

User can Register on this page using name email password and phone number. Email and Password is used to create the users account on Firebase and the user id is then used to store the name, phone-number and other information along with user's uid in the firebase database. User is automatically signed in after successfully registering otherwise proper errors are shown.

2. Go back to login page:

The users who are already registered can go to login page by clicking on the login here link.

5.1.3 Calendar:

This screen is the first screen shown to logged in users. This screen shows a calendar of current month along with today's date. You can click on the date to go to a daily screen for that date and you can change the month and year by clicking on them respectively. The daily list provides a list of days in the month on which we can click to go in the daily screen for current month it does not change even if the month changes.

From here user can do three actions :

1. Go to daily view.
2. Create an event.
3. Change Month/Year.

1. Go to daily view :

A user can go to the daily view by clicking on the calendar date in calendar or by clicking on the date in the daily view.

2. Create an event :

User can create an event by clicking on the plus icon on top of the daily view. The event is added by filling out a pop-up form that opens after clicking the button. Event name, event description, event location, latitude and longitude has to be provided.

3. Change Month/Year :

Click on Month or the Year Text view to change it.

5.1.4 Daily View:

The daily view shows the events of the day both personal and public and you can also add events from this view. This view also shows how many days of difference is there from current day to that day. From here user can do Two Actions:

1. Go back to calendar View.
2. Add an event.
3. Watch all public and private events.
4. Notify to subscribed users.
5. Search event location after event created
6. Delete an event

1. Go back to calendar View:

A user can go to the calendar view from this view by clicking on the back button on the top left.

2. Add an event:

User can add an event by clicking on the plus icon on the bottom of the personal/public events from that day.

3. Watch all public and private events:

All public and private events from the public database are shown in a popup dialog.

4. Notify to subscribed users:

Event creator of public event have option to notify subscribed users. We enable this feature before few hour of event time

5. Search event location after event created:

When user will create event with proper venue details user or subscriber will able to find it's correct location on map. We have provided option for search the event location

6. Delete an event:

When user will create event then after clicking that particular event we have an option to delete the event.

5.1.5 Timer:

The timer screen contains a timer app where we can set a timer and it will countdown to that timer. It can be useful for time management. In this section we can do Four Action namely:

1. Set a timer
2. Start the timer
3. Stop the timer
4. Reset the timer

5.1.6 Stopwatch:

The stopwatch screen contains a stopwatch app where we can start a stopwatch and it continues to count time until stopped by user. In this section we can do Three Actions namely:

1. Start Stopwatch

2. Stop Stopwatch
3. Reset Stopwatch

5.1.7 Profile:

The profile screen displays the number of personal/public events of the user along with subscribed events as well as, along with a log out button.

5.1.8 Time Management Tips:

This screen shows time management tips to the user and you can also search more tips my clicking the tips so it will directly search more management tips from Google.

5.2 System Analysis Model's

5.2.1 Data Flow Diagram:

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated.

5.2.1.1 Data Flow Diagram

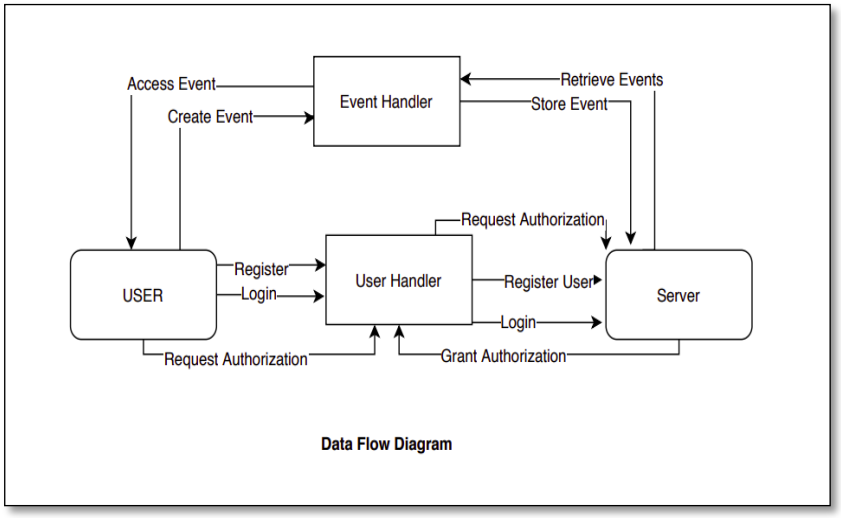


Fig 5.2.1.1 Data Flow Diagram (Level 0)

5.2.2 Use Case Diagram:

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

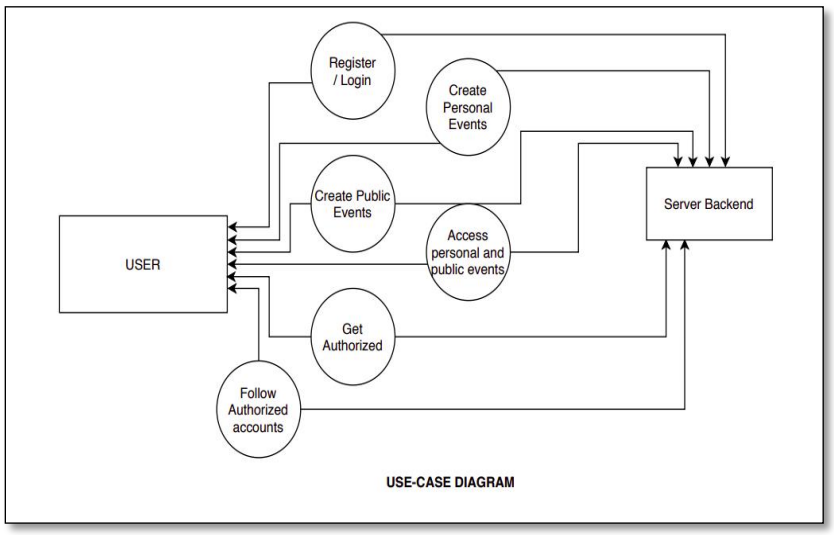


Fig 5.2.2.1 Use Case Diagram

5.2.3 Activity Diagram:

Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another.

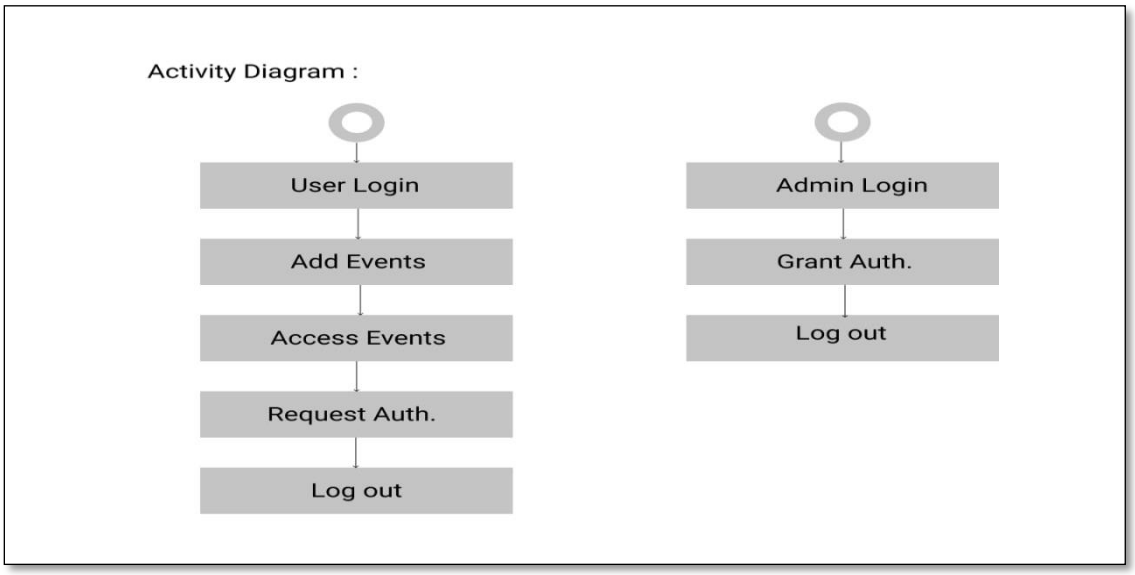


Fig 5.2.3.1 Activity Diagram

5.2.4 ER Diagram:

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. At first glance an entity relationship diagram looks very much like a flowchart

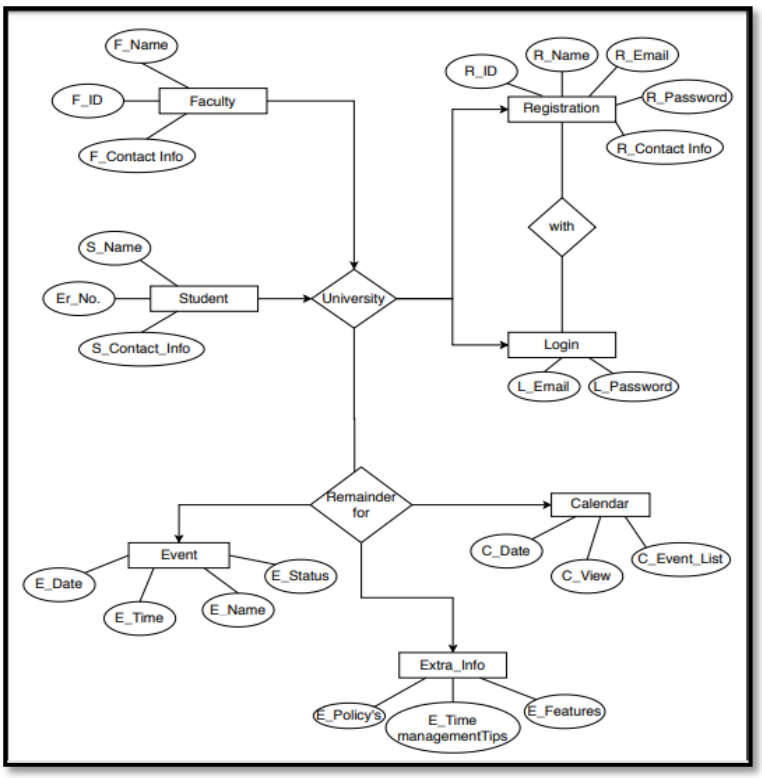


Fig 5.2.4.1 ER Diagram

5.2.5 Class Diagram:

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects

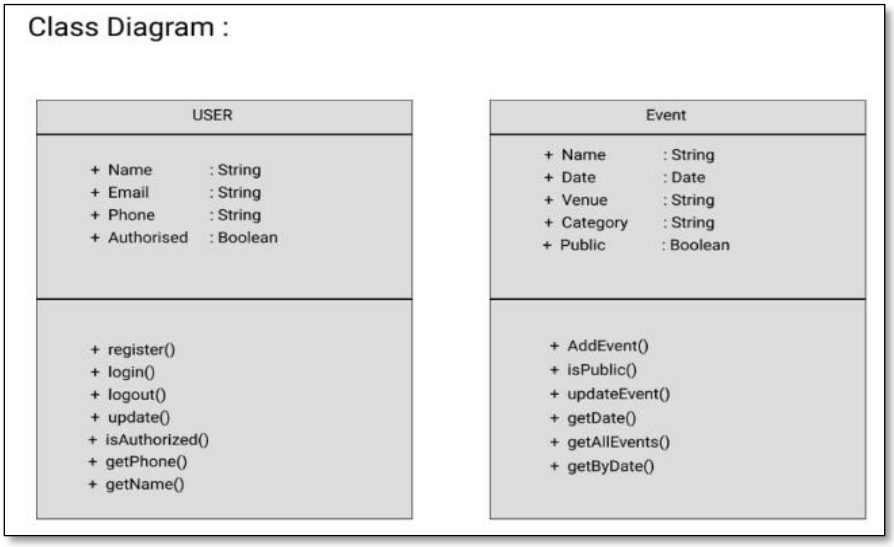


Fig 5.2.5.1 Class Diagram

5.2.6 Component Diagram:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation and details.

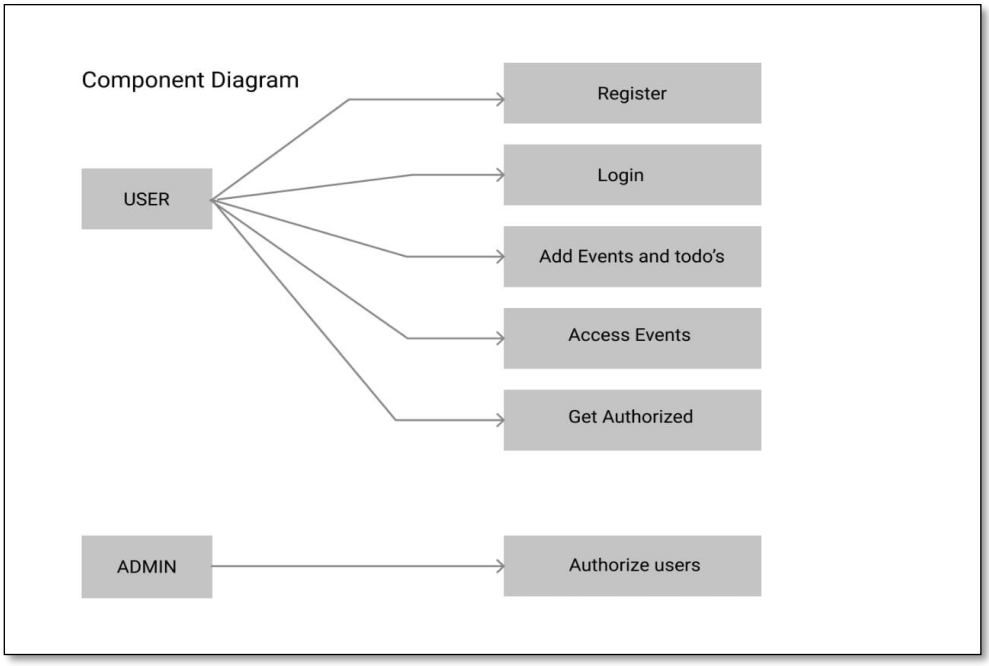


Fig 5.2.6.1 Component Diagram

5.2.7. Package Diagram:

A package diagram in the Unified Modeling Language depicts the dependencies between the packages that make up a model. It means a relationship between an importing namespace and a package.

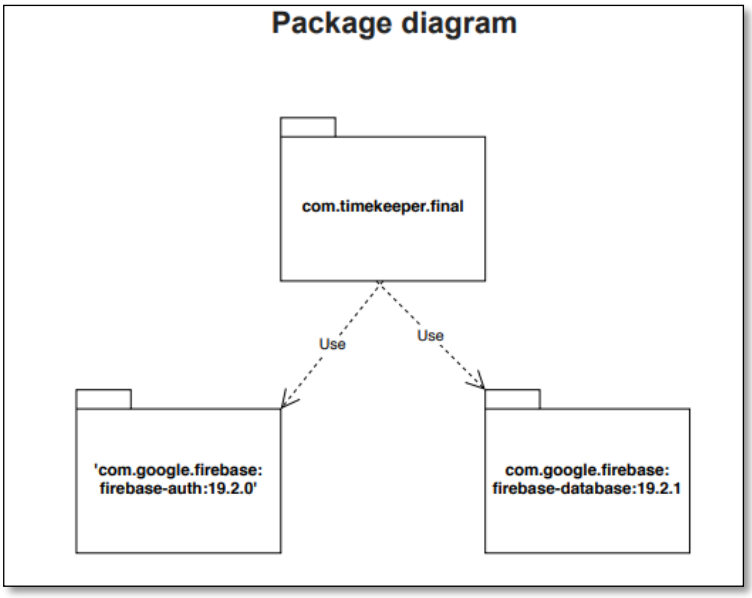


Fig 5.2.7.1 Package Diagram

5.3 Database Design

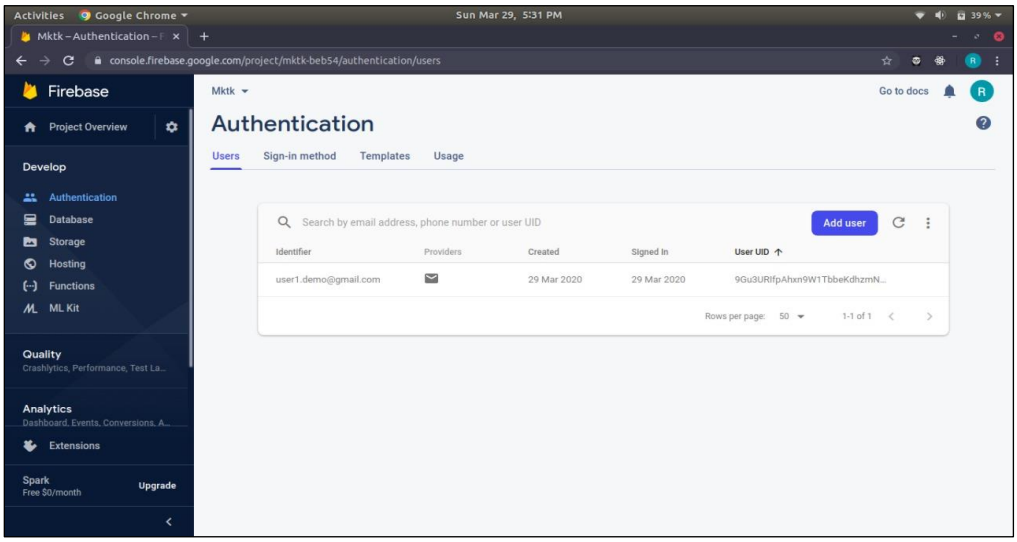


Fig 5.3.1 Authentication

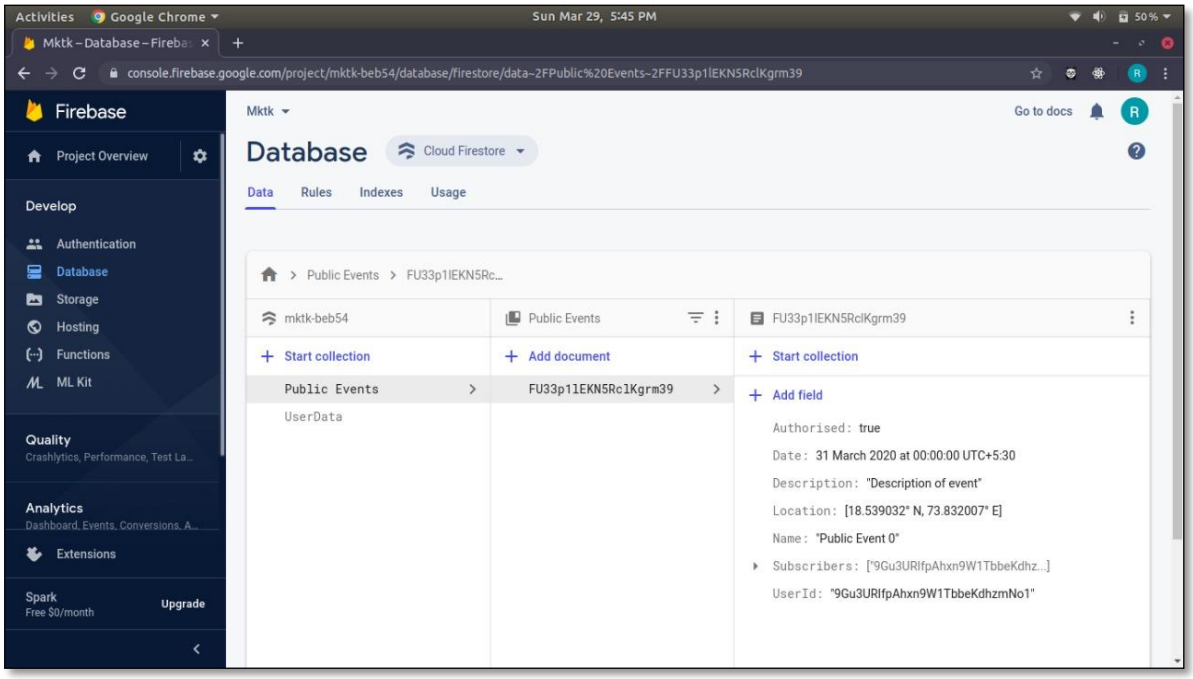


Fig 5.3.2 Database

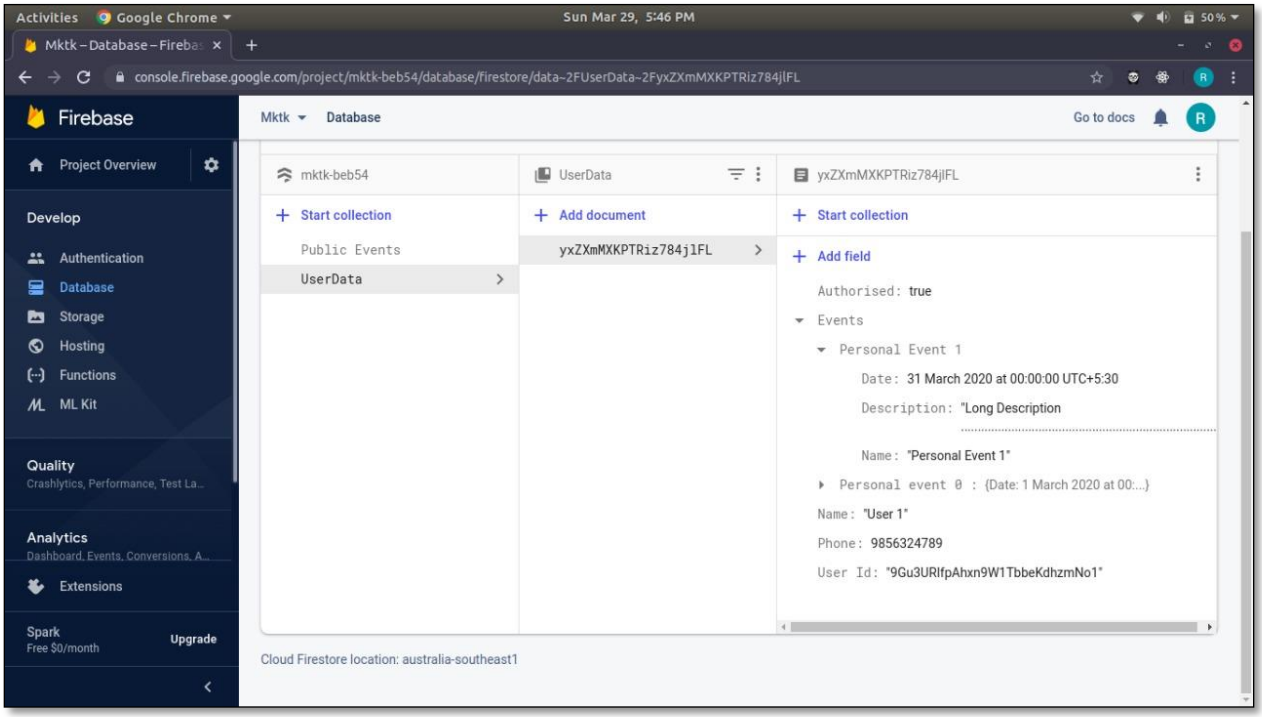


Fig 5.3.3 Firebase

Chapter 6: SCREENSHOTS

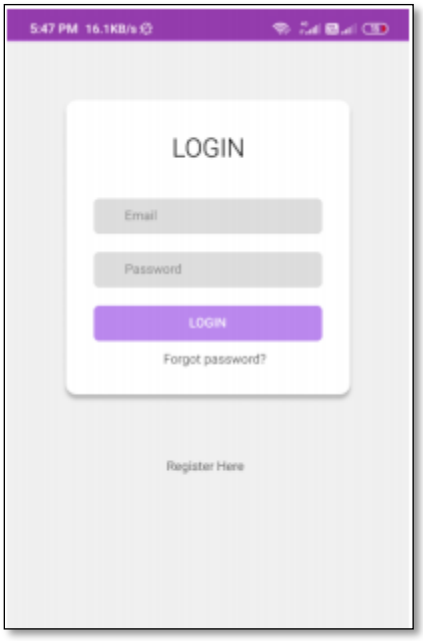


Fig 6.1. Login Screen

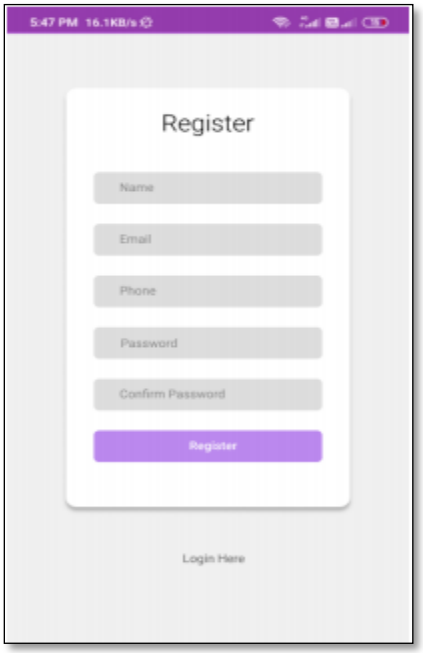


Fig 6.2. Registration Screen

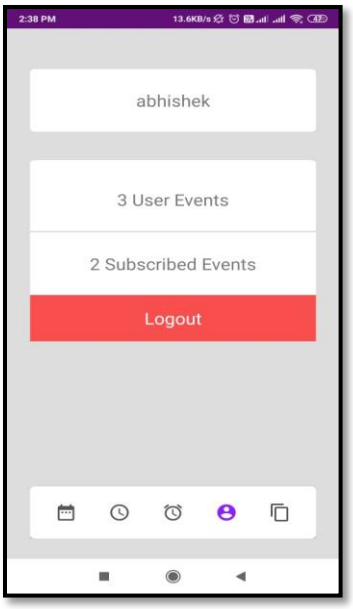


Fig 6.3 Profile view

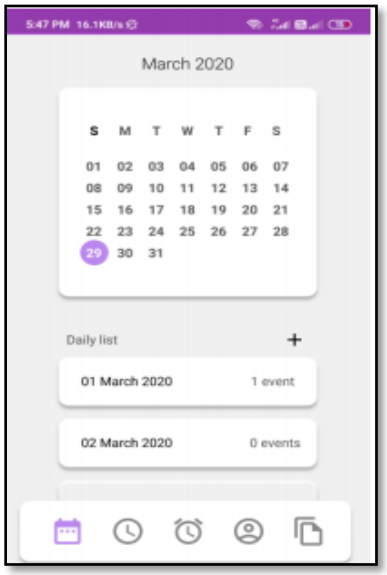
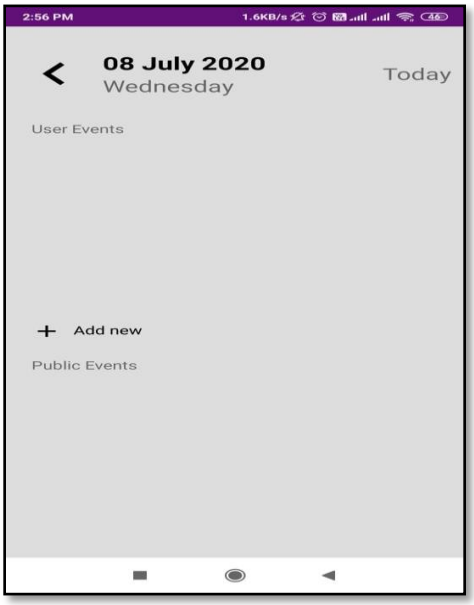


Fig 6.3. Calendar



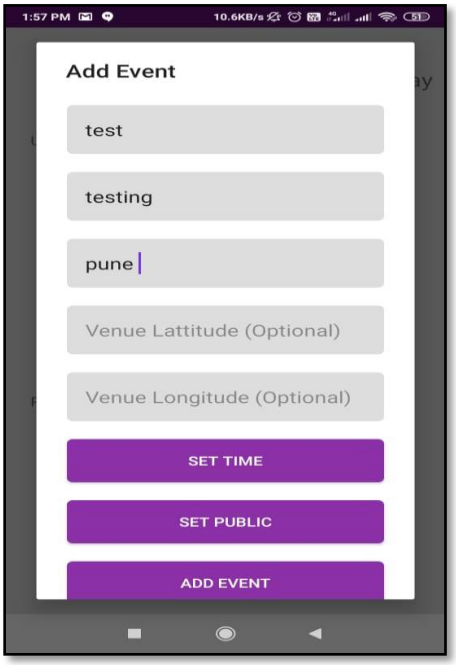


Fig 6.4 Add event

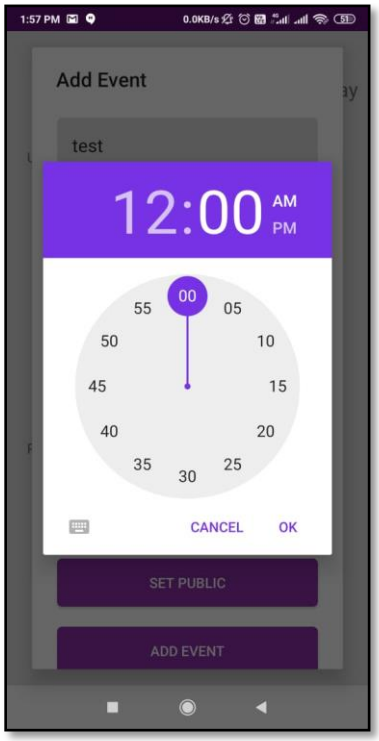


Fig 6.5 Set Time

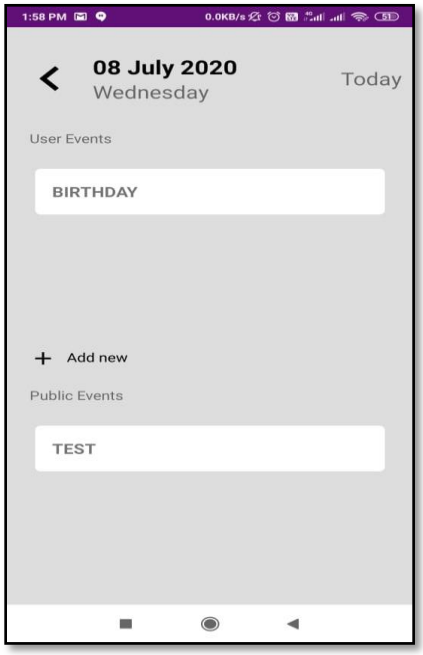


Fig 6.6. Daily View

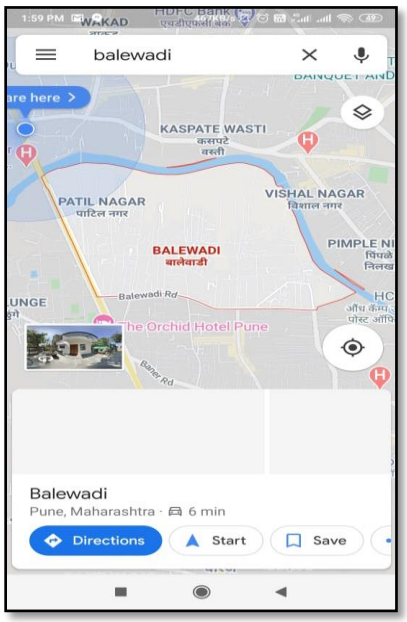


Fig. 6.6 location

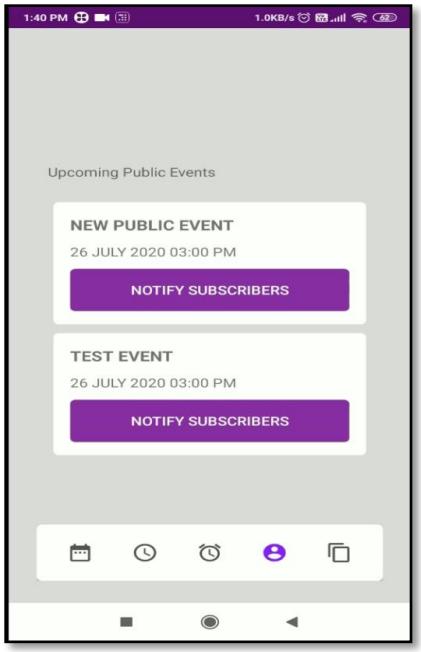


Fig.6.7 Admin Send Notification to User

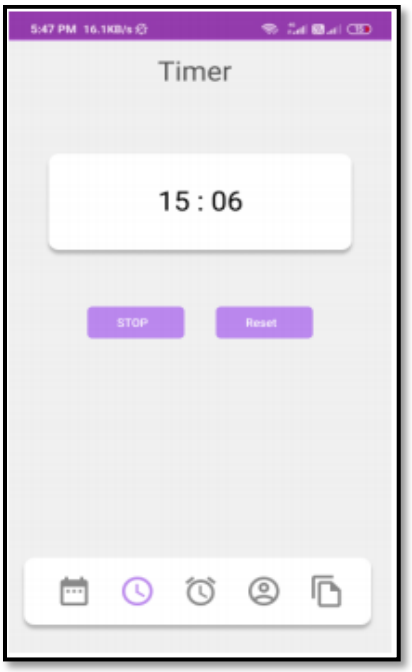


Fig 6.5. Timer

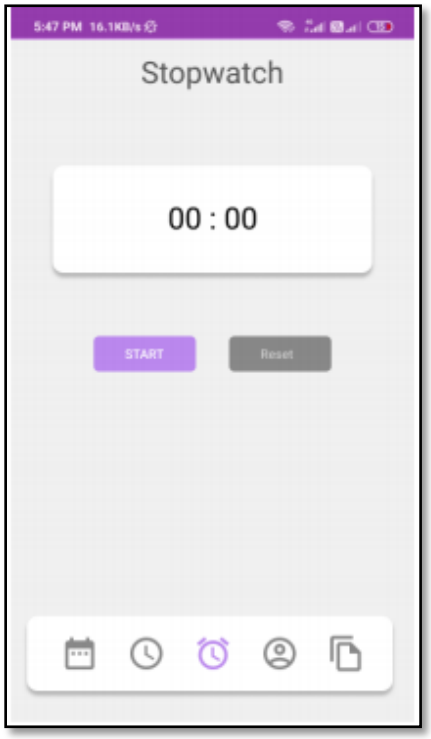


Fig.6.6 Stopwatch

Fig.6.7 Time Management Tips

Chapter 7: CONCLUSION

Chapter 8: REFERENCE

[1] https://blog.hubstaff.com/best-time-tracking-app-for-android/

[2] https://www.projecttopics.info/android/android-event-scheduler-reminder.php

[3] http://www.xsyssoftwaretechnologies.com/products/mobile-applications/android-event-scheduler

[4] http://mrbool.com/android-task-reminder-how-to-design-the-task-reminder.html

[5] https://www.intelegain.com/app-ideas-for-startups-to-launch-in-2020/

[6] https://www.cio.com/article/2370594/why-google-s-new--keep--service-might-not-be-a.html

[7] https://1000projects.org/events-manager-planner.html