A

PROJECT REPORT

ON

**ATTENDENCE SYSTEM**

Submitted in partial fulfillment of the

Requirement for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

**By**

SHAIKH MUSHARRAF & NITESH RAMCHANDRA PRAJAPATI

Roll No. - 137 & 57

**Under the esteemed guidance of**

Mrs. Smita Dalvi, Mrs. Vaishali Kadam & Mrs. Supriya Sudhir



**SHREE SHANKAR NARAYAN COLLEGE EDUCATION TRUSTS**

**SHANKAR NARAYAN COLLEGE**

(DEPARTMENT OF INFORMATION TECHNOLOGY)

**BHAYANDAR, 401 105**

MAHARASHTRA

**2019-2020**

**PREFACE**

Now a day, it is highly possible to adapt mobile computing in various applications.

The portability, open source nature of smart phones and android development platform has made the development of application software for various environments as handy.

Smartphone applications are resulted in paper less work, easy to use and time saving in nature.

The wireless communication technology of smart phone enables the information transfer from the current client to remote database server, where ever the network range is available.

The Attendance System is a mobile computing software application, which focuses on an activity or function, which is based on management information system of academic institutions.

In this work, the systems functionalities are categorized as a group and the similarities in grouped functionalities are designed and developed with reuse of software components.

# Index

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr.**  **No** |  |  | **Contents** | **Pg. No** |
|  |  |  |  |  |
| **1.** |  |  | **ACKNOWLEDGEMENT** | 6 |
|  |  |  |  |  |
| **2.** |  |  | **INTRODUCTION** | 7-9 |
|  | 2.1 |  | Synopsis | 8 |
|  | 2.2 |  | Software And Hardware Requirements | 9 |
|  |  |  |  |  |
| **3.** |  |  | **SCOPE AND OBJECTIVE** | 10 |
|  |  |  |  |  |
| **4.** |  |  | **THEORITICAL BACKGROUND** | 11-15 |
|  |  |  |  |  |
| **5.** |  |  | **PROBLEM DEFINITION** | 16-17 |
|  | 5.1 |  | Drawbacks of Existing System | 17 |
|  | 5.2 |  | Advantages of Proposed System | 17 |
|  |  |  |  |  |
| **6.** |  |  | **PLANNING** | 18-22 |
|  | 6.1 |  | GANTT Chart | 19-20 |
|  | 6.2 |  | PERT Chart | 21-22 |
|  |  |  |  |  |
| **7.** |  |  | **COST AND BENEFIT ANALYSIS** | 23-26 |
|  | 7.1 |  | COCOMO Model | 24-26 |
|  |  |  |  |  |
| **8**. |  |  | **METHODOLOGY ADOPTED** | 27-28 |
|  |  |  |  |  |
| **9.** |  |  | **SYSTEM ANALYSIS AND DESIGN** | 29-58 |
|  | 9.1 |  | Feasibility Study | 30 |
|  | 9.2 |  | E-R Diagram | 31-32 |
|  | 9.3 |  | Class Diagram | 33-34 |
|  | 9.4 |  | Use Case Diagram | 35-36 |
|  | 9.5 |  | Sequence Diagram | 37-38 |
|  | 9.6 |  | Collaboration Diagram | 39-40 |
|  | 9.7 |  | Object Diagram | 41-42 |
|  | 9.8 |  | Activity Diagram | 43-44 |
|  | 9.9 |  | State Chart | 45-46 |
|  | 9.10 |  | Component Diagram | 47-48 |
|  | 9.11 |  | Deployment Diagram | 49-50 |
|  | 9.12 |  | Data Dictionary | 51 |
|  |  | 9.12.1 | Table Structure | 52 |
|  | 9.13 |  | Screen Design | 53- |
|  |  |  |  |  |
| **10** |  |  | **PROCESS INVOLVED(Description of modules in detail)** | 47-67 |
|  |  |  |  |  |
| **11** |  |  | **TESTING METHODOLOGY** | 47 |
|  | 11.1 |  | Unit Testing | 60 |
|  | 11.2 |  | System Testing |  |
|  | 11.3 |  | Strategies of Testing |  |
|  |  | 11.3.1 | Black box Testing | 60 |
|  |  | 11.3.2 | Equivalence Class Partitioning | 62 |
|  |  | 11.3.3 | White box Testing | 64 |
|  | 11.4 |  | Test Reports | 66 |
|  |  |  |  |  |
| **12** |  |  | **FUTURE ENHANCEMENTS** | 69 |
| **13** |  |  | **REFERENCE AND BIBLIOGRAPHY** |  |

**ACKNOWLEDGEMENT**

Achievement is finding out what you would be doing rather than what you must do. It is not until you undertake such a project that you realize how much effort and hard work it is, what your capabilities are and how well you can represent yourself or other things. I would like to add a few heartfelt words for the people who were a part of it in a numerous way. People who gave me unending support right from the stage the project ideas were conceived. I take this opportunity to express my profound gratitude to management of **Shankar Narayan College** for giving me this opportunity to accomplish this project work.

I express my special thanks to our Principal **Dr. V. N. Yadav** of Shankar Narayan College for extending their support and allowing me to work on this project.

We are very much thankful to **Prof. Vaishali Kadam**,the Coordinator for her kind co-operation in the completion of my project. A special thanks to **Prof. Supriya Sudhir** who is our project guide for her most sincere, useful and encouraging contribution throughout the project span.

Last but not the least; I would like to thank all our Teachers, friends & our family for their support, motivation and encouragement.

(SHAIKH MUSHARRAF)

(NITESH RAMCHANDRA PRAJAPATI)

INTRODUCTION

**2.1 SYSNOPSIS**

1. **Title of the project:**

Attendance System

1. **Objective of the project:**

Attendance System is software developed for daily student attendance in schools, colleges and institutes. If facilitates to access the attendance information of a particular student in a particular class. The information is sorted by the operators, which will be provided by the teacher for a particular class. This system will also help in evaluating attendance eligibility criteria of a student. The purpose of developing attendance tracking system is to computerized the tradition way of taking attendance. Another purpose for developing this software is to generate the report automatically at the end of the session or in the between of the session.

**The main objective of this project is to computerize the manual system & reduce the time consumption.**

In other words we can say that our project has the following objectives.

* Make the attendance system computerized.
* Reduce time consumption
* Reduce error scope
* Paper work is reduced
* Need not to maintain registers for future references.

1. **Project category:**

The Project category is Mobile Application (Android) developed in Java using Android Studio.

1. **Modules:**

* Registration form
* Login
* Admin
* Teacher
* Attendance Report

It has modules to store

1. Student information
2. modify student class
3. Division, roll no.
4. Select course
5. Add edit subjects
6. Enter attendance class and subject wise
7. Generate daily and consolidate attendance in attractive format.

**2.2**  **Software and hardware requirement**

* **Language and software tool used:**
* Front End: Android, Java
* Back End: MySQL
* Development Environment: Android Studio
* Operating System: Window 7

**Description of Tool:**

**Android Studio:**

Android Studio is the official [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) for [Google](https://en.wikipedia.org/wiki/Google)'s [Android](https://en.wikipedia.org/wiki/Android_%28operating_system%29) [operating system](https://en.wikipedia.org/wiki/Operating_system), built on [Jet Brain’s](https://en.wikipedia.org/wiki/JetBrains) [IntelliJ IDEA](https://en.wikipedia.org/wiki/IntelliJ_IDEA) software and designed specifically for [Android development](https://en.wikipedia.org/wiki/Android_software_development). It is available for download on [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS) and [Linux](https://en.wikipedia.org/wiki/Linux) based operating systems. It is a replacement for the [Eclipse Android Development Tools](https://en.wikipedia.org/wiki/Eclipse_%28software%29#Android_Development_Tools) (ADT) as primary IDE for native Android application development.

**Java:**

Java Database Connectivity (JDBC) is an [application programming interface](https://en.wikipedia.org/wiki/Application_programming_interface) (API) for the programming language [Java](https://en.wikipedia.org/wiki/Java_%28programming_language%29), which defines how a client may access a [database](https://en.wikipedia.org/wiki/Database). It is Java based data access technology and used for Java database connectivity. It is part of the [Java Standard Edition](https://en.wikipedia.org/wiki/Java_Standard_Edition) platform, from [Oracle Corporation](https://en.wikipedia.org/wiki/Oracle_Corporation). It provides methods to query and update data in a database, and is oriented towards [relational databases](https://en.wikipedia.org/wiki/Relational_database). A JDBC-to-[ODBC](https://en.wikipedia.org/wiki/ODBC) bridge enables connections to any ODBC-accessible data source in the [Java virtual machine](https://en.wikipedia.org/wiki/Java_virtual_machine) (JVM) host environment.

1. **Hardware requirement:**

* Operating system: Window 7
* Hard disks: 40GB
* RAM: 256 MB
* Android device

**3** **Scope And Objective**

* **Scope Of System :**
* Login to the First page of the application.
* Change the password after logging in to the system.
* View/Change his/her details.
* An admin login should be present who can read as well as remove any uploads.
* In this project manage the student details like student Attendance, monthly record of each student.
* **Objective of System:**
* To design and develop the android based mobile based attendance system.
* “Attendance System” is software developed for maintaining the attendance of the student on the daily basis in the collage.
* Here the staffs, who are handling the subjects, will be responsible to mark the attendance of the students.
* Each staff will be given with a separate username and password based on the subject they handle. An accurate report based on the student attendance is generated here.
* This system will also help in evaluating attendance eligibility criteria of a student. Report of the student’s attendance on weekly and monthly basis is generated.
* The scope of the project is the system on which, the software is installed, i.e. the project is developed as an Android application and it will work for a particular institute. But later on the project can be modified to operate it online.
* This system will help to make an entry of attendance easy.
* This system also tracks the monthly record of attendance.

**4** **Theoretical Background (Technology)**

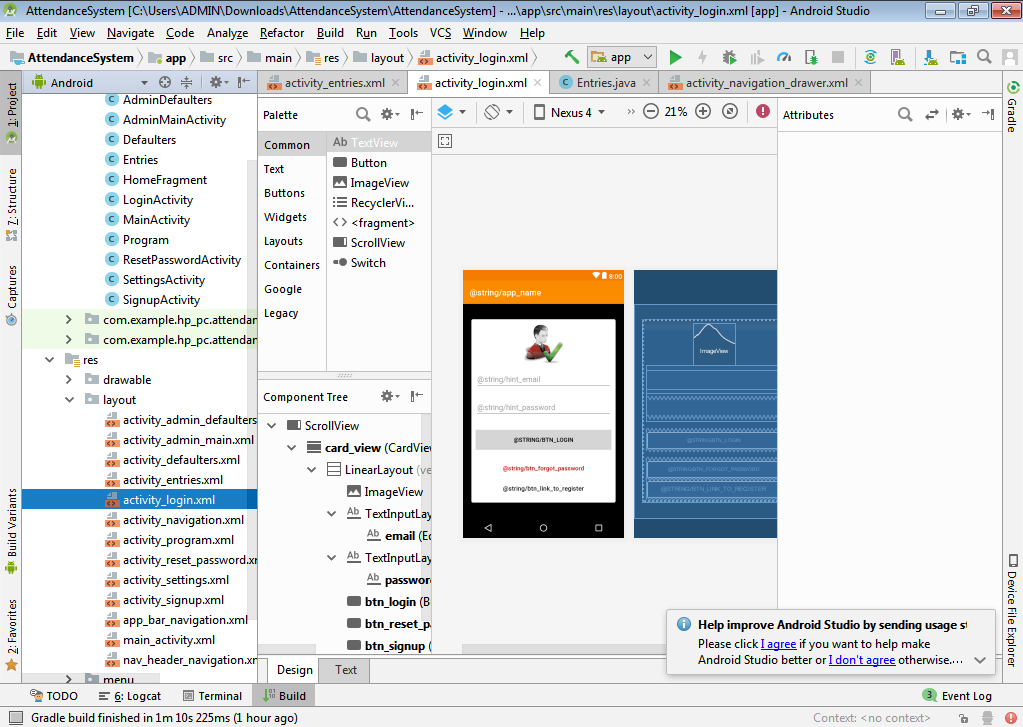
**FRONT END:-**

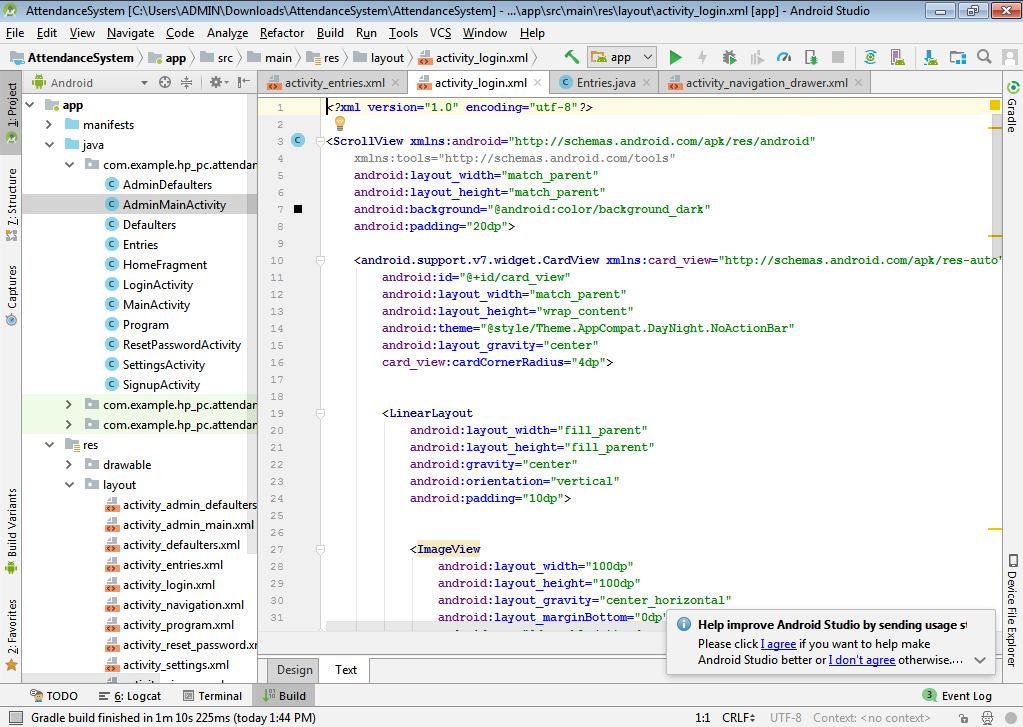
**Android Studio 3.0.1**

Android Studio is the official Integrated Development Environment for Android application development, based on [IntelliJ IDEA.](https://www.jetbrains.com/idea/) On top of the capabilities expected from IntelliJ, Android Studio also offers:

1. Flexible Gradle-based build system
2. Build variants and multiple APK file generation
3. Code templates to help you build common app features
4. Rich layout editor with support for drag and drop them editing
5. Lint tools to catch performance, version compatibility, usability and other problems
6. ProGuard and app-signing capabilities

Built-in support for [Google Cloud Platform,](http://developers.google.com/cloud/devtools/android_studio_templates/) making it easy to integrate Google Cloud Messaging and App Engine.

****

****

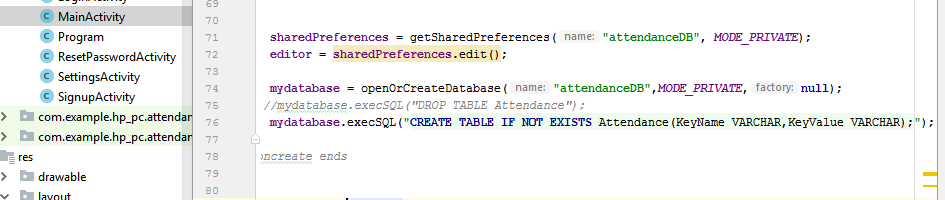
**BACK END**:-

1. **SQLite**

* SQLite is a software library that implements a self-contained, server less, zero-configuration, transactional SQL database engine. SQLite is the most widely deployed SQL database engine in the world. The source code for SQLite is in the public domain. This tutorial will give you a quick start with SQLite and make you comfortable with SQLite programming.

## **Why SQLite?**

* SQLite does not require a separate server process or system to operate (server less).
* SQLite comes with zero-configuration, which means no setup or administration needed.
* A complete SQLite database is stored in a single cross-platform disk file.
* SQLite is very small and light weight, less than 400KiB fully configured or less than 250KiB with optional features omitted.
* SQLite is self-contained, which means no external dependencies.
* SQLite transactions are fully ACID-compliant, allowing safe access from multiple processes or threads.
* SQLite supports most of the query language features found in SQL92 (SQL2) standard.
* SQLite is written in ANSI-C and provides simple and easy-to-use API.
* SQLite is available on UNIX (Linux, Mac OS-X, Android, iOS) and Windows (Win32, WinCE, WinRT).

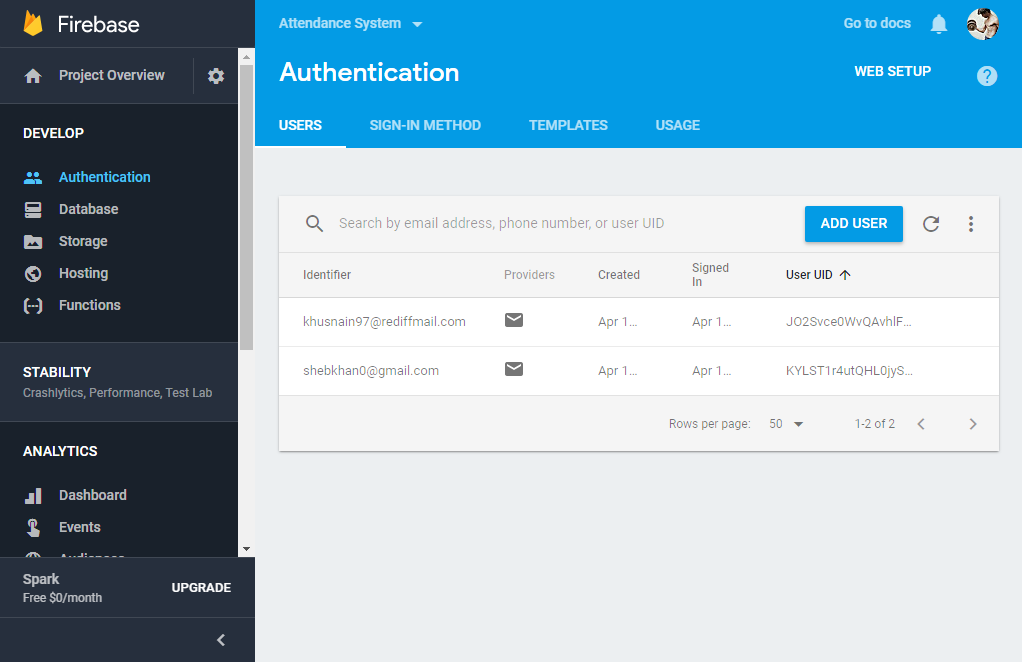


1. **Firebase**

* Firebase is a mobile and web app development platform that provides developers with a plethora of tools and services to help them develop high-quality apps, grow their user base, and earn more profit.

Firebase Services can be divided into two groups:





**PROBLEM DEFINITION**

**5.1 Problems with Existing System**

* The Existing system is a manual entry for the students. Here the attendance will be carried out in the hand written registers. It will be a tedious job to maintain the record for the user. The human effort is more here. The retrieval of the information is not as easy as the records are maintained in the hand written registers.
* This application requires correct feed on input into the respective field. Suppose the wrong inputs are entered, the application resist to work. So the user find it difficult to use.
* Manual maintenance of data.
* No attendance percentage calculation

**5.2** **Advantages of Proposed System**

To overcome the drawbacks of the existing system, the proposed system has been evolved. This project aims to reduce the paper work and saving time to generate accurate results from the student’s attendance. The system provides with the best user interface. The efficient reports can be generated by using this proposed system.

* **Advantages of Proposed System**

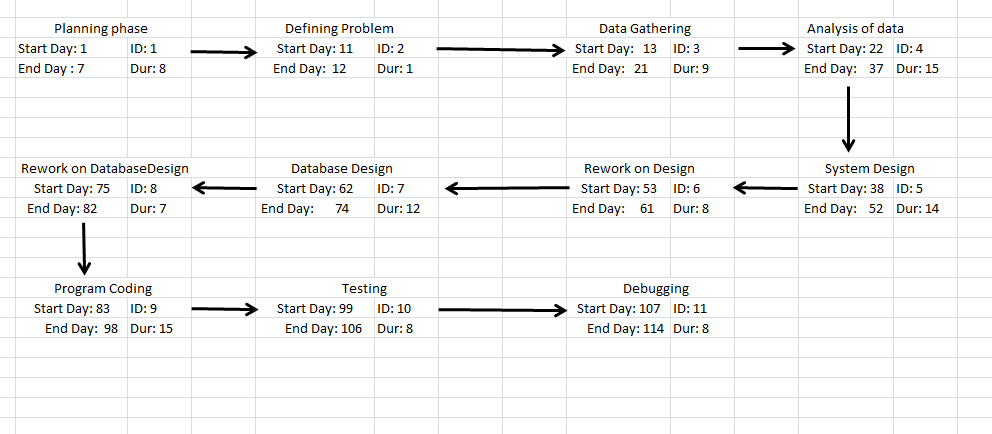
1. It is trouble-free to use.
2. It is a relatively fast approach to enter attendance
3. Is highly reliable, approximate result from user
4. Best user Interface
5. Efficient reports
6. Easy to handle and feasible.
7. Cost reduction.
8. Fast and Convenient
9. User friendly
10. Computer operator control
11. Very less paper work

**PLANNING**

* 1. **Gantt Chart**
* A Gantt chart is a type of bar chart illustrates a project schedule.
* Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project.
* Terminal elements and summary elements comprise the work breakdown structure of the project. Some Gantt chart also shows the dependency (precedence network) relationships between activities.
* Although now regarded as a common charting technique, Gantt charts were considered revolut7ionary when they were introduced.
* This chart is used also in Information Technology to represent data that have been collected.
* Gantt chart can be used in any project that involves effort, resources and milestone and deliverables.
* Gantt chart can also be used for breaking the utilization of the resource in the project.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **WEEKS**  **TASKS** | **11-**  **DEC** | **14-**  **DEC** | **21-**  **DEC** | **28-**  **DEC** | **5-**  **JAN** | **12-**  **JAN** | **19-**  **JAN** | **26-**  **JAN** | **2-**  **FEB** | **9-**  **FEB** | **16-**  **FEB** | **23-**  **FEB** | **28-**  **FEB** | **6-**  **MAR** | **13-**  **MAR** | **20-**  **MAR** | **27-**  **MAR** | **3-**  **APR** |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Planning**  **Phase** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Defining**  **Problem** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Data**  **Gathering** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Analysis of data** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **System Design** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Rework on**  **System Design** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Database**  **Design** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Rework on**  **Database design** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Program**  **Coding** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Testing** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Debugging** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

* 1. **PERT CHART**
* The Program (or Project) Evaluation and Review Technique, commonly abbreviated PERT, is a model for project management designed to analyze and represent the tasks involved in completing a given project.
* PERT is a method to analyze the involved tasks in completing a given project especially the time needed to complete each task, and identifying the minimum time needed to complete the total project.
* A PERT chart is a project management tool used to schedule organize and coordinate tasks within a project PERT stands for Program Evaluation Review Technique a methodology developed by the U S Navy in the 1950s to manage the Polaris submarine missile program. A similar methodology the Critical Path Method (CPM) was developed for project management in the private sector at about the same time.
* A PERT chart presents a graphic illustration of a project as a network diagram consisting of numbered nodes (either circles or rectangles) representing events or marks in the project. The direction of the arrows on the lines indicates the sequence of tasks**.**



**COST AND BENEFITS ANALYSIS**

* 1. **Cost Benefit Analysis.**

Cost-Benefit Analysis (CBA) estimates and totals up the equivalent money value of the benefits and costs to the community of projects to establish whether they are worthwhile. It is important to identify cost and benefit factors. Therefore a cost based study was done on Book Hive to identify the factors. Cost and benefits were categorized as follows: -

Development costs and Operating costs, Cost benefit analysis is done in three steps:

* The first is estimate the anticipated development & operational cost. Development costs are those that are incurred during the development of system. Operational costs are those that are anticipated financial benefits.
* Financial benefits are the expected annual saving or increase in revenue derived from installation of the system.
* Third cost benefits analysis calculated based on the detailed estimates of cost & benefits.

The cost of project OFFICE MANAGEMENT is estimated by using cocomo model for estimation which is based on size estimation. The most fundamental calculation in the COCOMO model is the use of the Effort Equation to estimate the number of Person-  
Months required developing a project.

**COCOMO MODEL1**

COCOMO'81 models depend on the two main equations.

Development effort: MM = a \* KDSIb based on MM - man-month / person month / staff-month is one month of effort by one person. In COCOMO'81, there are 152 hours per Person month. According to organization these values may differ from the standard by 10% to 20%.

Effort and development time (TDEV): TDEV = 2.5 \* MMc.

The coefficients a, b and c depend on the mode of the development. There are three modes of development:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Development Mode | Project Characteristics | |  | |
|  | Size | Innovation | Deadline/Constraint | Dev.  Environment |
| Organic | Small | Little | Not light | Stable |
| Semi-  Detached | Medium | Medium | Medium | Medium |
| Embedded | Large | Greater | Tight | Complex hardware |

BASIC COCOMO

The basic COCOMO applies the parameterized equation without much detailed consideration of project characteristics.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Basic COCOMO | A | B | C | D |
| Organic | 2.4 | 1.05 | 2.5 | 0.38 |
| Semi-detached | 3.0 | 1.12 | 2.5 | 0.35 |
| Embedded | 3.6 | 1.20 | 2.5 | 0.32 |

SM = a \* KDSIb.

TDEV = c \* SMd

DSI = 4000

KDSI = DSI/1000

= 4000/1000

= 4

SM = 2.5 \* 41.05.

= 10.28

TDEV = 2.5 \* 10.280.38

= 6.06

Average Staff = SM / TDEV

= 10.28 / 6.06

= 1.69

= 2 FSP

1. **Methodology Adopted (SDLC Model)**

Steps for software development:-

System Development Life Cycle (SDLC) is the overall process of developing information systems through a multi-step process from investigation of initial requirements through analysis, design, implementation and maintenance. There are many different models:

1. Waterfall Model

2. Spiral Model

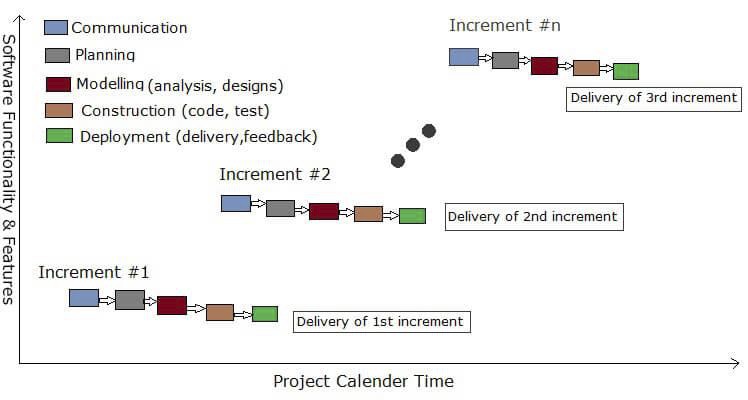
3. Rapid prototype Model

4. Incremental Model

In software engineering the SDLC concept underpins many kinds of software development methodologies. These methodologies form the framework for planning and controlling the creation of an information system. Waterfall model: Model used for proposed system.

**INCREMENTAL MODEL:-**

Incremental model in software engineering is a one which combines the elements of waterfall model in an iterative manner. It delivers a series of releases called increments which provide progressively more functionality for the client as each increment is delivered. In the incremental model of software engineering, [waterfall model](http://www.technotrice.com/what-is-waterfall-model-software-engineering/) is repeatedly applied in each increment. The incremental model applies linear sequences in a required pattern as calendar time passes. Each linear sequence produces an increment   
in the work.



As from the diagram, you can see that there are 5 phases(tasks) which are carried out in each increment. If you want to see what activity is carried out in each phase, then check out this post: [Phases of waterfall model](http://www.technotrice.com/what-is-waterfall-model-software-engineering/) as the phases are same.

The first increment is often a [core product](http://en.wikipedia.org/wiki/Core_product) where the necessary requirements are addressed, and the extra features are added in the next increments. The core product is used and evaluated by the client. Once the customer assesses the core product, there is plan development for the next increment. Thus in every increment, the needs of the client are kept in mind, and more features and functions are added, and the core product is updated. This process continues till the complete product is produced.

* Initial product delivery is faster.
* Lower initial delivery cost.
* Core product is developed first i.e. main functionality is added in the first increment.
* After each iteration, [regression testing](http://en.wikipedia.org/wiki/Regression_testing) should be conducted. During this testing, faulty elements of the software can be quickly identified because few changes are made within any single iteration.
* It is generally easier to test and debug than other methods of software development because relatively smaller changes are made during each iteration. This allows for more targeted and rigorous testing of each element within the overall product.
* With each release a new feature is added to the product.
* Customer can respond to feature and review the product. Risk of changing requirement is reduced
* Work load is less.

**SYSTEM ANALYSIS AND DESIGN**

**9.1 Feasibility Study**

Feasibility analysis begins once the goals are defined. It starts by generating broad possible solutions, which are possible to give an indication of what the new system should look like. This is where creativity and imagination are used. Analysts must think up new ways of doing things- generate new ideas. There is no need to go into the detailed system operation yet. The solution should provide enough information to make reasonable estimates about project cost and give users an indication of how the new system will fit into the organization. It is important not to exert considerable effort at this stage only to find out that the project is not worthwhile or that there is a need significantly change the original goal.

Feasibility of a new system means ensuring that the new system, which we are going to implement, is efficient and affordable. There are various types of feasibility to be determined. They are,

**9.1.1 Economically feasibility**

* Development of this application is highly economically feasible. The only thing to be done is making an environment with an effective supervision.
* It is cost effective in the sense that has eliminated the paper work completely. The system is also time effective because the calculations are automated which are made at the end of the month or as per the user requirement.

**9.1.2** **Technical feasibility**

* The technical requirement for the system is economic and it does not use any other additional Hardware and software.
* Technical evaluation must also assess whether the existing systems can be upgraded to use the new technology and whether the organization has the expertise to use it.

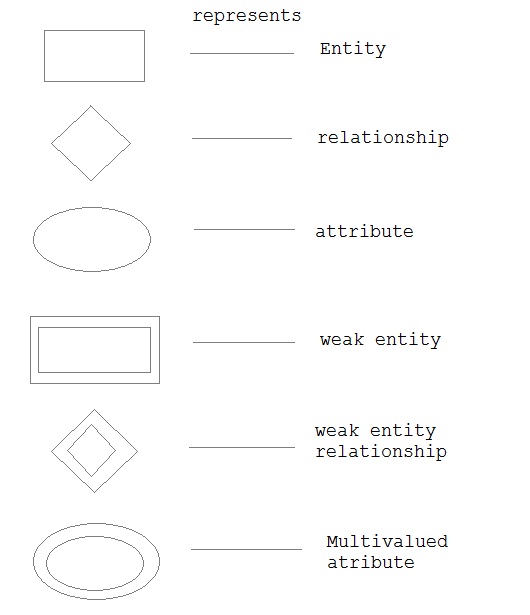
**9.1.3** **Operational feasibility**

* The system working is quite easy to use and learn due to its simple but attractive interface.
* User requires no special training for operating the system.
* Technical performance include issues such as determining whether the system can provide the right information for the Department personnel student details, and whether the system can be organized so that it always delivers this information at the right place and on time using intranet services. Acceptance revolves around the current system and its personnel.

**9.2** **ER Diagram**

The Entity Relationship Diagram (Model) is based on perception of a real world that consists of a collection of basic objects called as Entity and relationships among these objects. Entities in database are described by a set of attributes.

* A Relationship is an association among several Entities.
* The set of Entities of the same type are called as Entity Set.
* The set of Relationships of same type are called as Relationship Set.
* Notations:



Makes present and absent

Teacher

Student

Contains

Select

Class

On

Subject

ER diagram

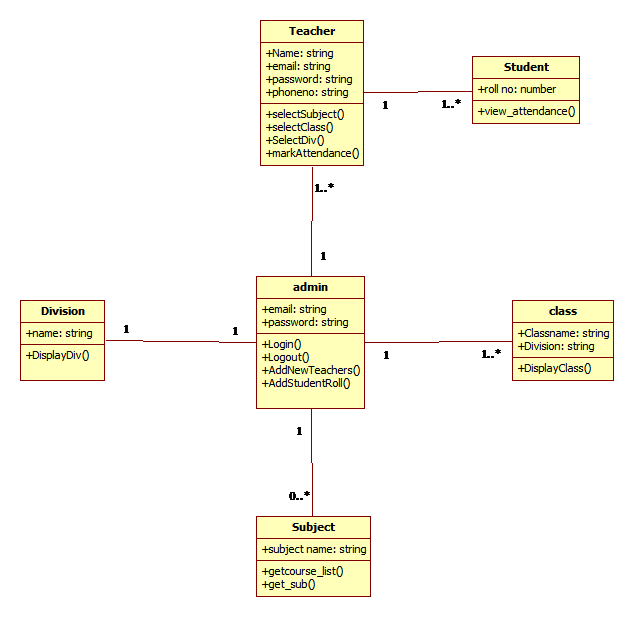
**9.3** **Class Diagram**

* A class diagram is an illustration that shows the relationship between systems and describes different types of objects.
* The classes in a class diagram represent both the main objects and or interactions in the application and the objects to be programmed.
* The format of the class is as follows:-

|  |
| --- |
| Operation |
| Name |
| Attribute |

* The first section is name where we define the name of the class.
* In attribute section we define the attribute with data type.

In operation section we define the operations performed by the system.



**Class Diagram**

**9.4** **Use Case Diagram**

* A Use Case represents the sequence of actions that the systems perform to produce something of value to the actor interacting with the system.
* An actor may participate in more than one use case and, conversely, more than one actor may participate in the same use case.
* The Use Case Diagram is a graphical depiction with an accompanying textual description of use cases and the actor that participate in them.
* Use Case Diagrams model the functionality of system by using Actors and Use Cases.
* ACTOR :-

User

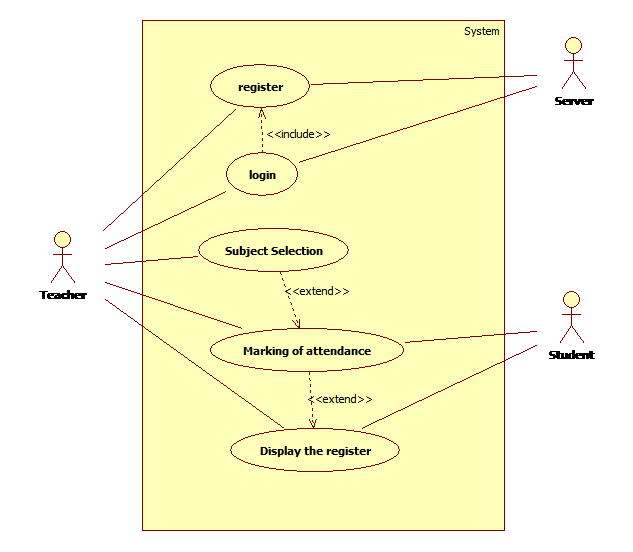
An actor is always outside the automation system boundary of the system. An actor can play several roles.

* LINES :-

This indicates that which actor participates with which case.

* USE CASES:

Use Cases are services or functions provided by the system to its users. Each use case describes one logical interaction between the Actor and the system.



**Use Case Digram**

**9.5 Sequence Diagram**

Shows sequence of interactions between objects and flow of events in a single use case.

Focuses on message details.

Used more frequently in industry.

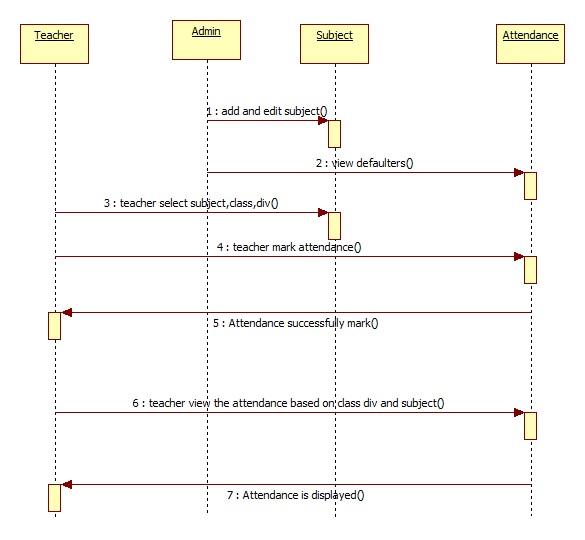
**SSD Notation**

Actor represented by stick figure person (or role) that “Interacts” with system by entering input data and receiving output data. Object notation is rectangle with name of object underline shows individual object and not class of all similar objects. Lifeline is vertical line under object or actor to show passage of time for object. Messages use arrows to show messages sent or recovered by actor or system.

ACTOR SYMBOL

OBJECT SYMBOL

LIFELINE SYMBOL

****

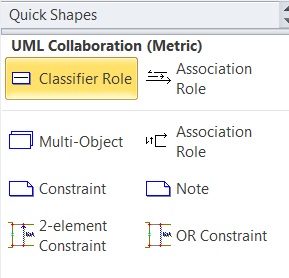
**Sequence Diagram**

**9.6 Collaboration Diagram**

A collaboration diagram is a type of visual presentation that shows how various software objects interact with each other within an overall IT architecture and how users can benefit from this collaboration. A collaboration diagram often comes in the form of a visual chart that represents a flow chart.

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

* Notations:

****

**1: Teacher Collaboration**

: Teacher table

Teacher register 3 2: 2: Login(us,pw)

: Class, :div, :subject

1: enter all mandatory fields 3: Selection of class, div, subject

4: mark attendance 5: view defaulter

Attendance

Defaulter

**2: admin collaboration**

: Login

Admin login 3 2: 2: Login(us,pw)

: Class, :div, :subject

1: enter all mandatory fields 3: Selection of class, div,

Subject

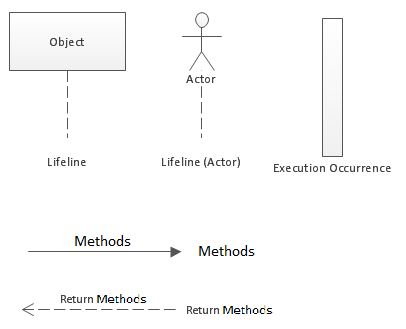
4: view defaulter

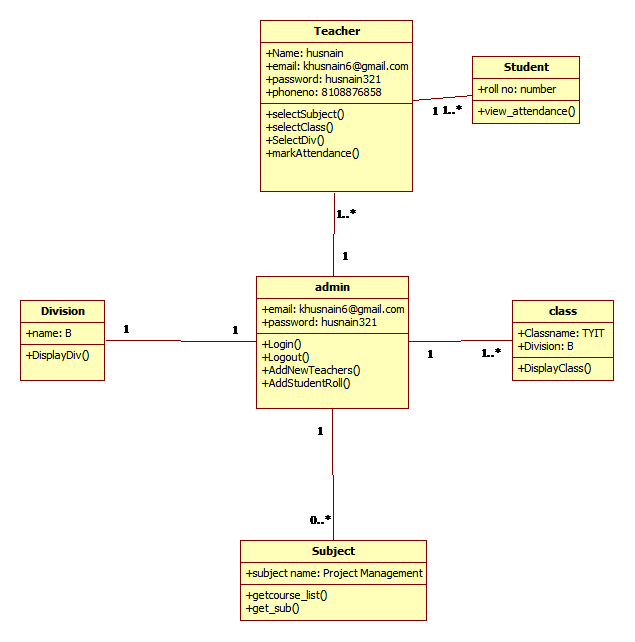
Attendance

**9.7 Object Diagram**

An **object diagram** in the Unified Modelling Language (UML), is a diagram that shows a complete or partial view of the structure of a modelled system at a specific time.

In the Unified Modelling Language (UML), an object diagram focuses on some particular set of objects and attributes, and the links between these instances. A correlated set of object diagrams provides insight into how an arbitrary view of a system is expected to evolve over time. Object diagrams and class diagrams are closely relatedand use almost identical notation.



****

**Object Diagram**

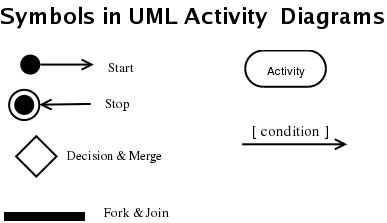
**9.8** **Activity Diagram**

Activity diagram is basically a flow chart to represent the flow form 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. Before drawing an activity diagram, we should identify the following elements:

* Activities
* Association
* Conditions
* Constraints

The most important shape types:

* Rounded rectangles represent activities;
* Diamonds represent decisions;
* Bars represent the start (split) or end (join) of concurrent activities;
* A black circle represents the start (initial state) of the workflow;
* An encircled black circle represents the end (final state).



Login Admin/Teacher

Invalid

Authentication

Whether teacher or student

Logout

View attendance

Mark attendance

Select subject

Select class,div

Register

View defaulters

Edit subject

Add subject

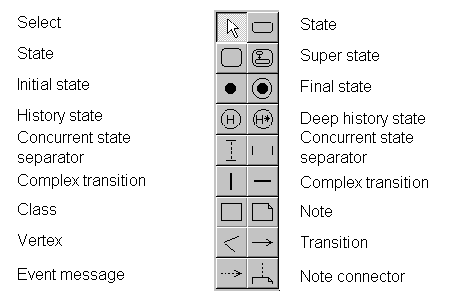
Valid

**Activity Diagram**

**9.9** **State chart Diagram**

* This diagram shows the life of an object in states and transitions.
* State diagram describes the dynamic behaviour of a system in response to external stimuli.
* It helps to model dynamic behaviour of objects based on states.
* It also helps to model reactive objects whose states are triggered by specific events.
* It helps to describe passive objects which go through several distinct phases during their lifetime.

**Notations:**

****

Mark attendance

View defaulter

Select class, div, subject

Teacher register

Login as

View defaulter

Select class, div, subject

Admin

**State chart Diagram**

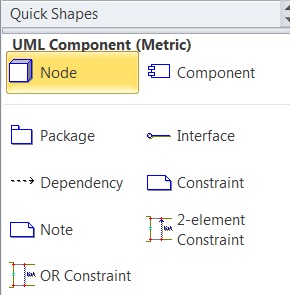
**9.10** **Component Diagram**

Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems.

The purpose is different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities.

The purpose of the component diagram can be summarized as:

* Visualize the components of a system.
* Construct executable by using forward and reverse engineering.
* Describe the organization and relationships of the components.





**user admin/teacher**

**class,division**



**Subject**





**Attendance**

**Component Diagram**

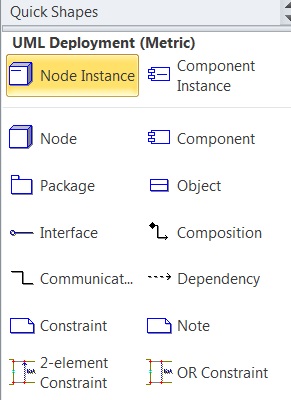
**9.11** **Deployment Diagram**

A **deployment diagram** the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected

There are two types of Nodes:

* Device Node
* Execution Environment Node

Device nodes are physical computing resources with processing memory and services to execute software, An execution environment node (EEN) is a software computing resource that runs within an outer node.



Firebase

Admin database

Attendance DB

Device

User Interface

User android phone

SQLite

Teacher Database

Firebase

**Deployment Diagram**

**9.12 Data Dictionary**

A **data dictionary**, or metadata repository, as defined in the *IBM Dictionary of Computing*, is a "centralized repository of information about data such as meaning, relationships to other data, origin, usage, and format. *Oracle* defines it as a collection of tables with metadata. The term can have one of several closely related meanings pertaining to data bases and database management system (DBMS):

* A document describing a database or collection of databases
* An integral component of a DBMS that is required to determine its structure
* A piece of middleware that extends or supplants the native data dictionary of a DBMS

A data dictionary is a file or a set of files that contains a database's metadata. The data dictionary contains records about other objects in the database, such as data ownership, data relationships to other objects, and other data.

**9.12.1 Table Structure**

User/Registration

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| Name | varchar(50) |
| Email | varchar(50) |
| Password | varchar(50) |
| Phone | number(10) |

Login

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| Email | varchar(50) |
| Password | varchar(50) |

Admin login

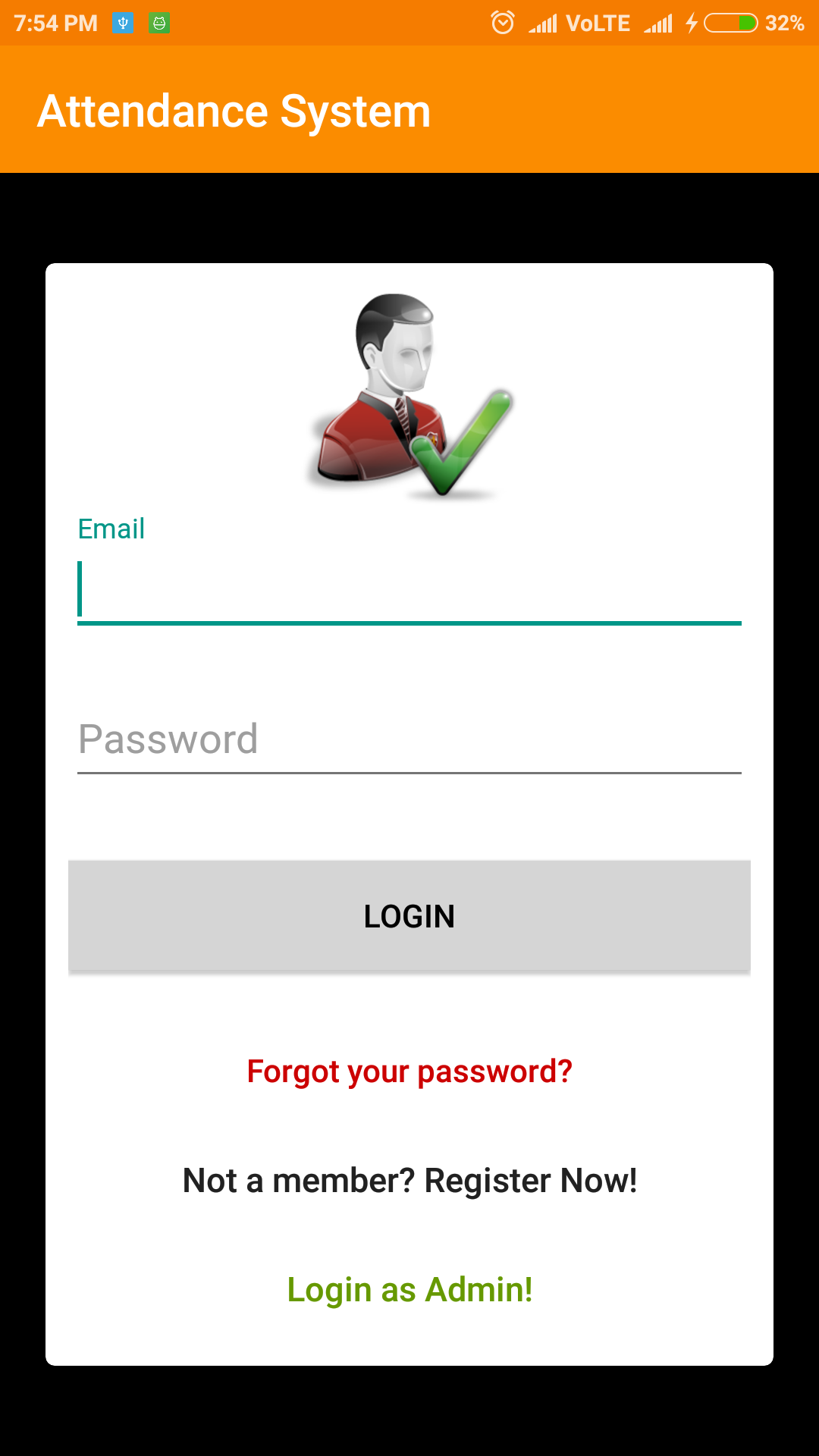
|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| username | varchar(50) |
| Password | varchar(50) |

Attendance

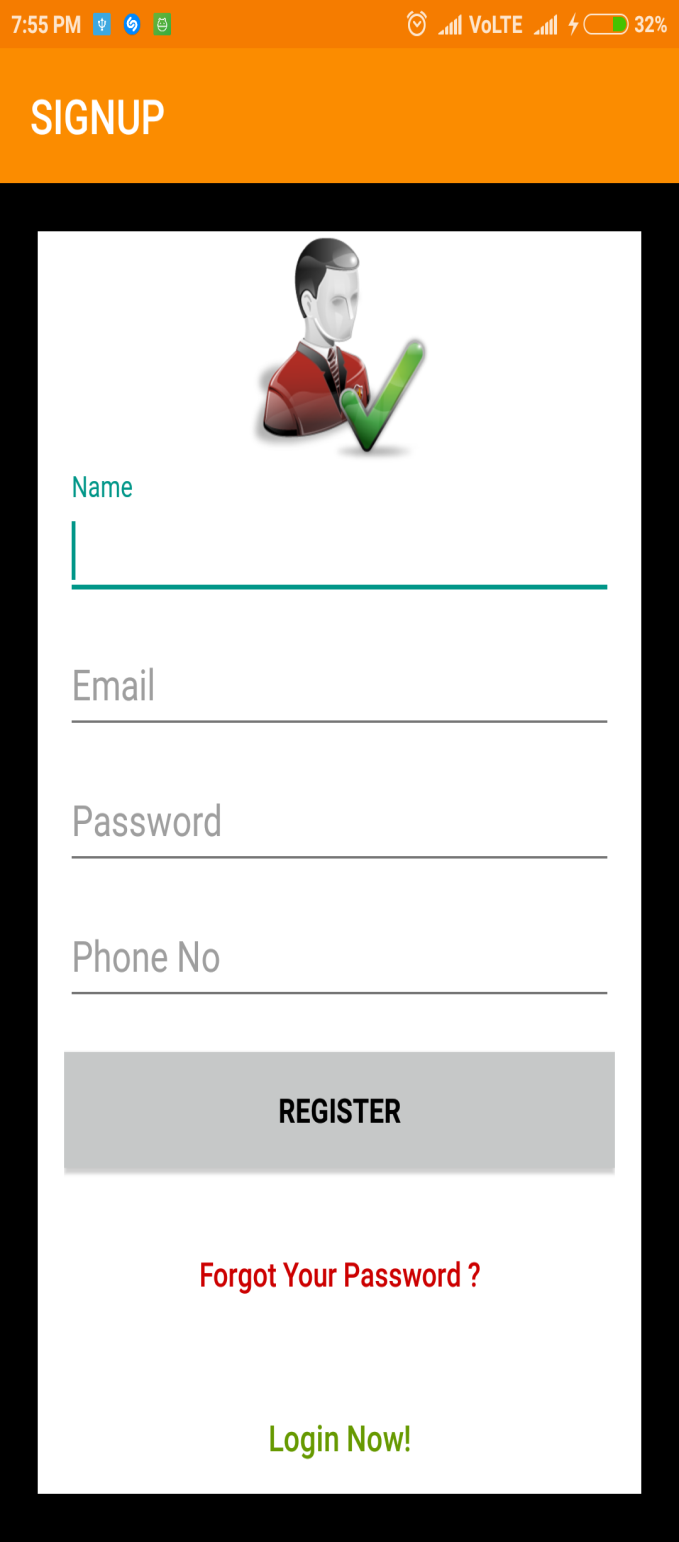
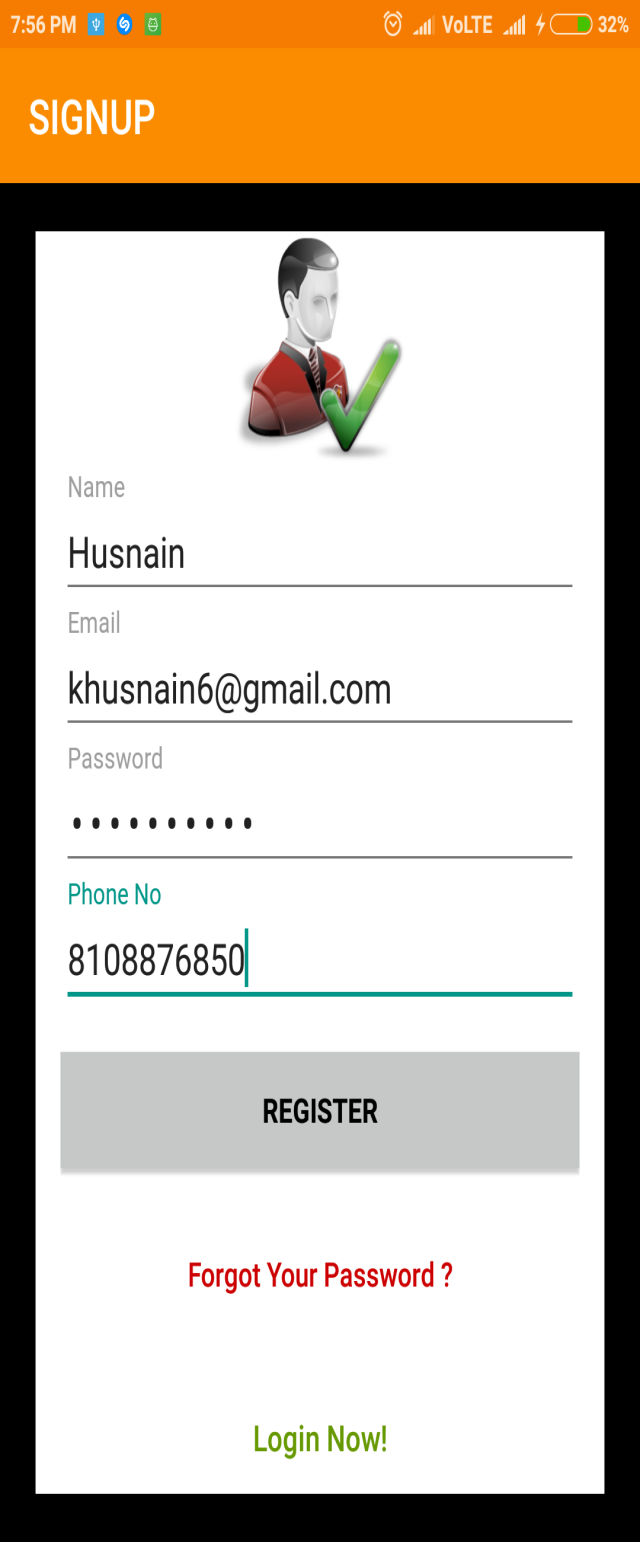
|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| keyname | number(10) |
| keyvalues | number(10) |

**9.13** **Screen Design**

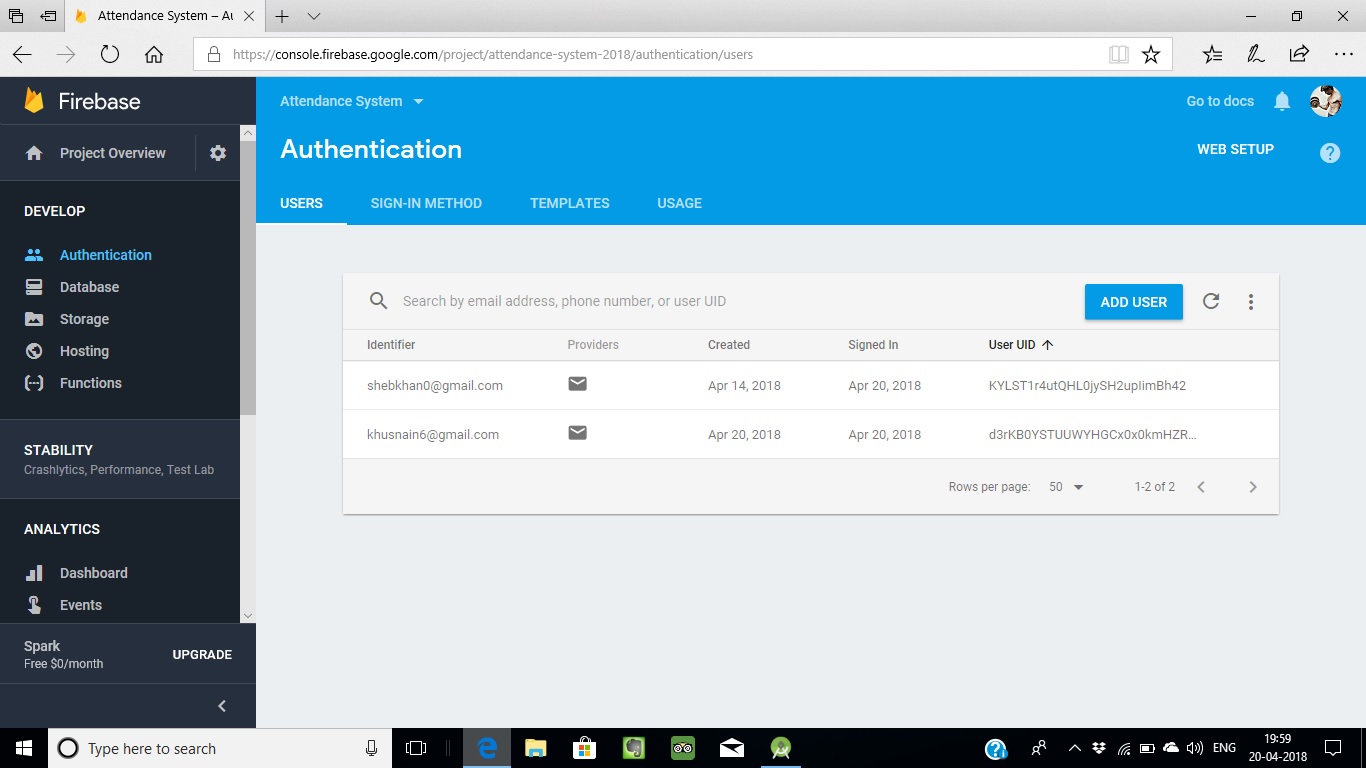
* **Main screen**

****

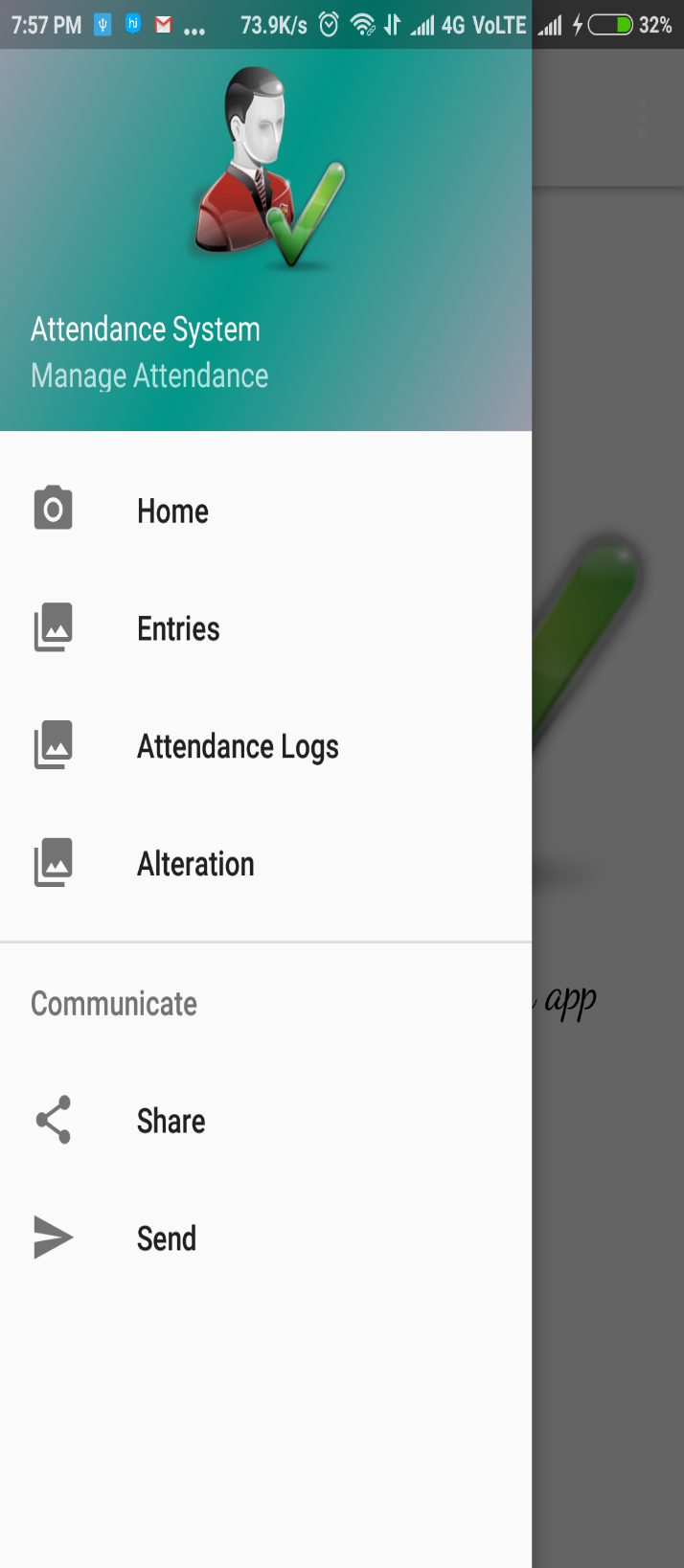
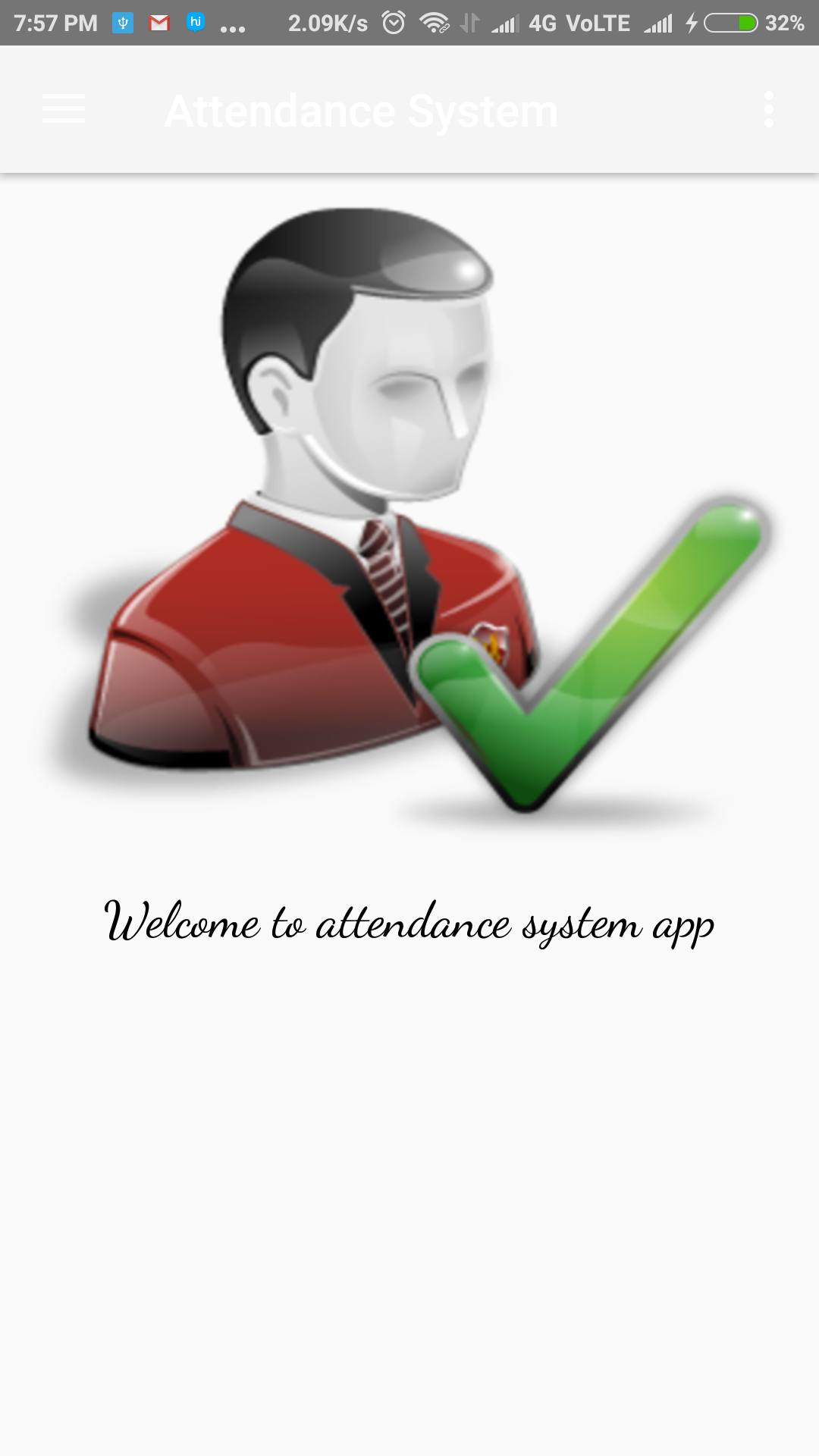
* **Admin/user registration**



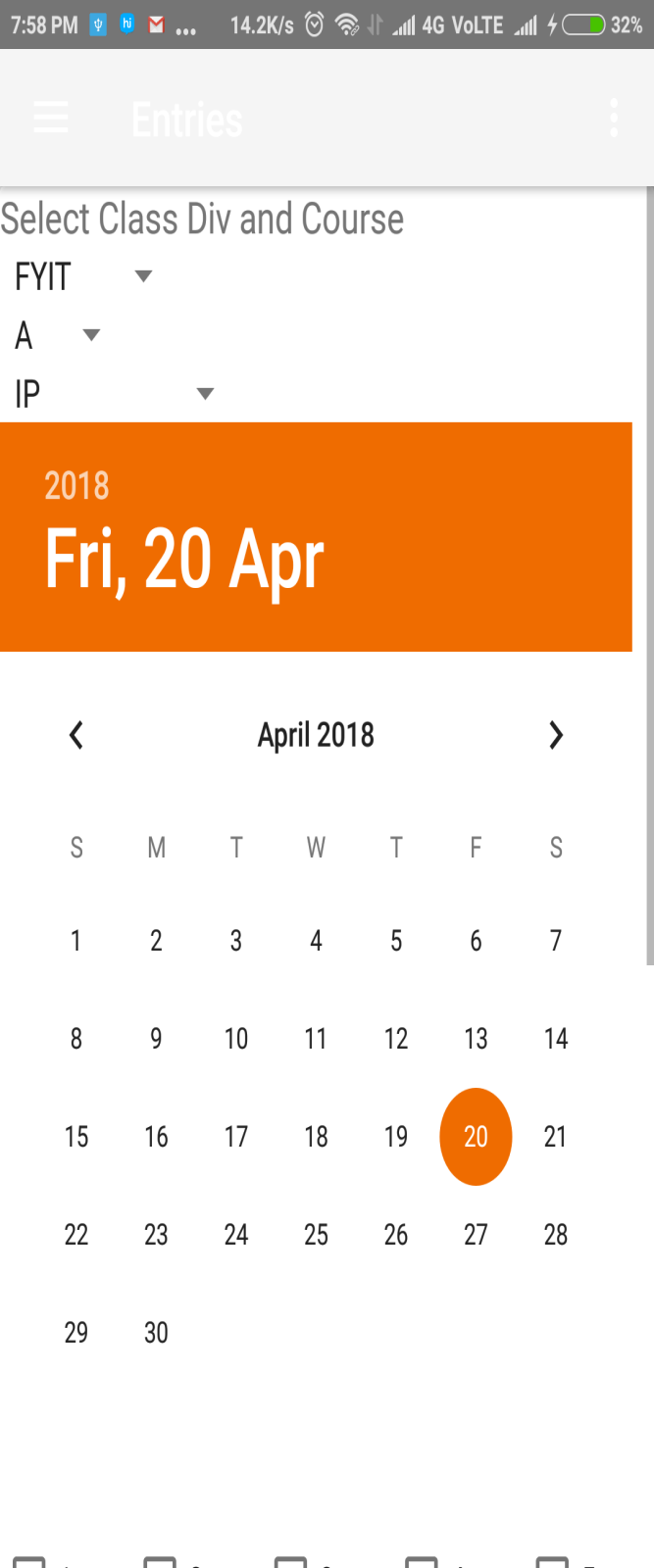
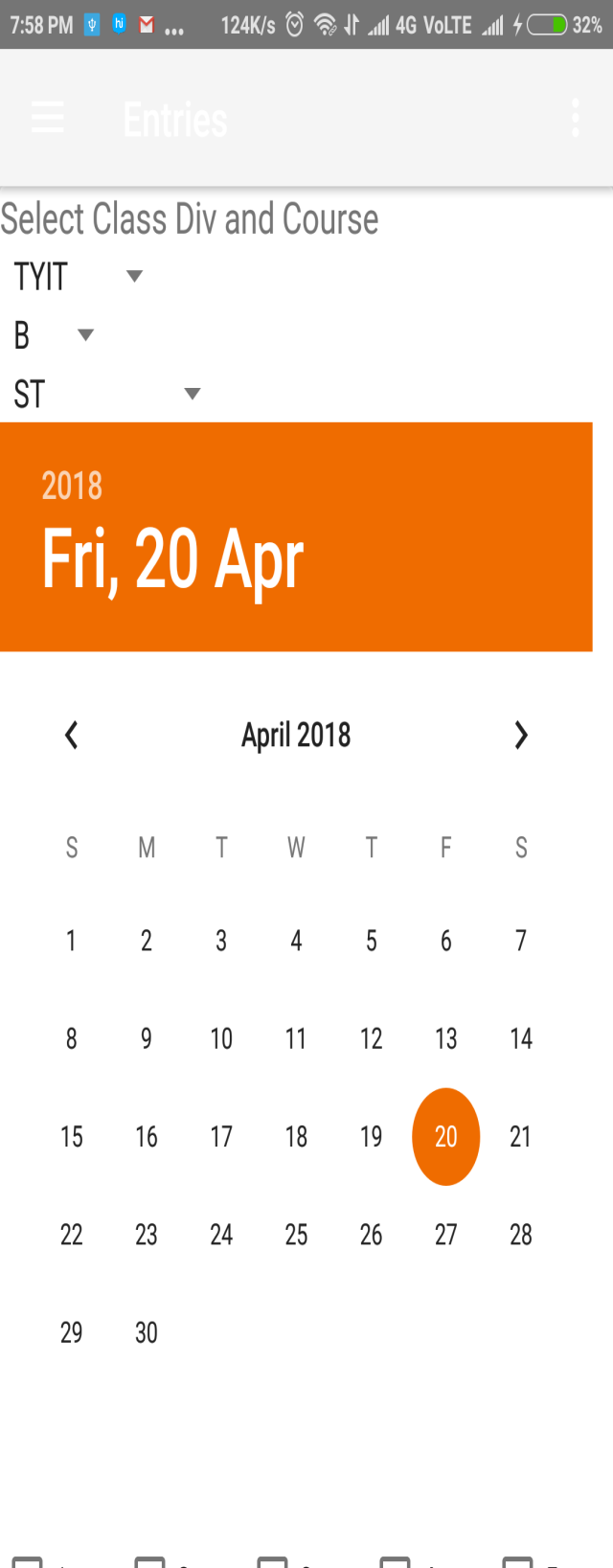
* **Firebase**

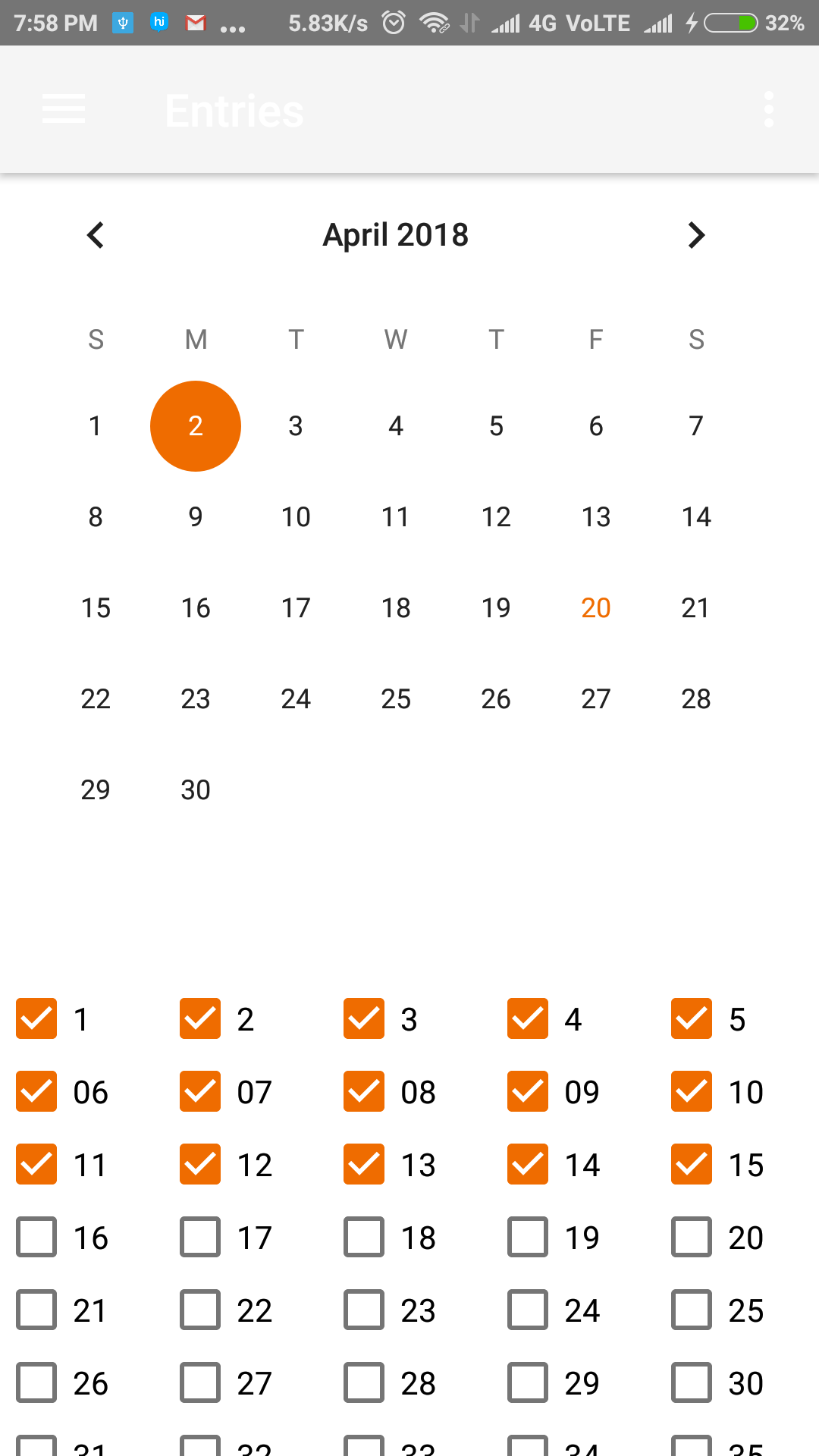
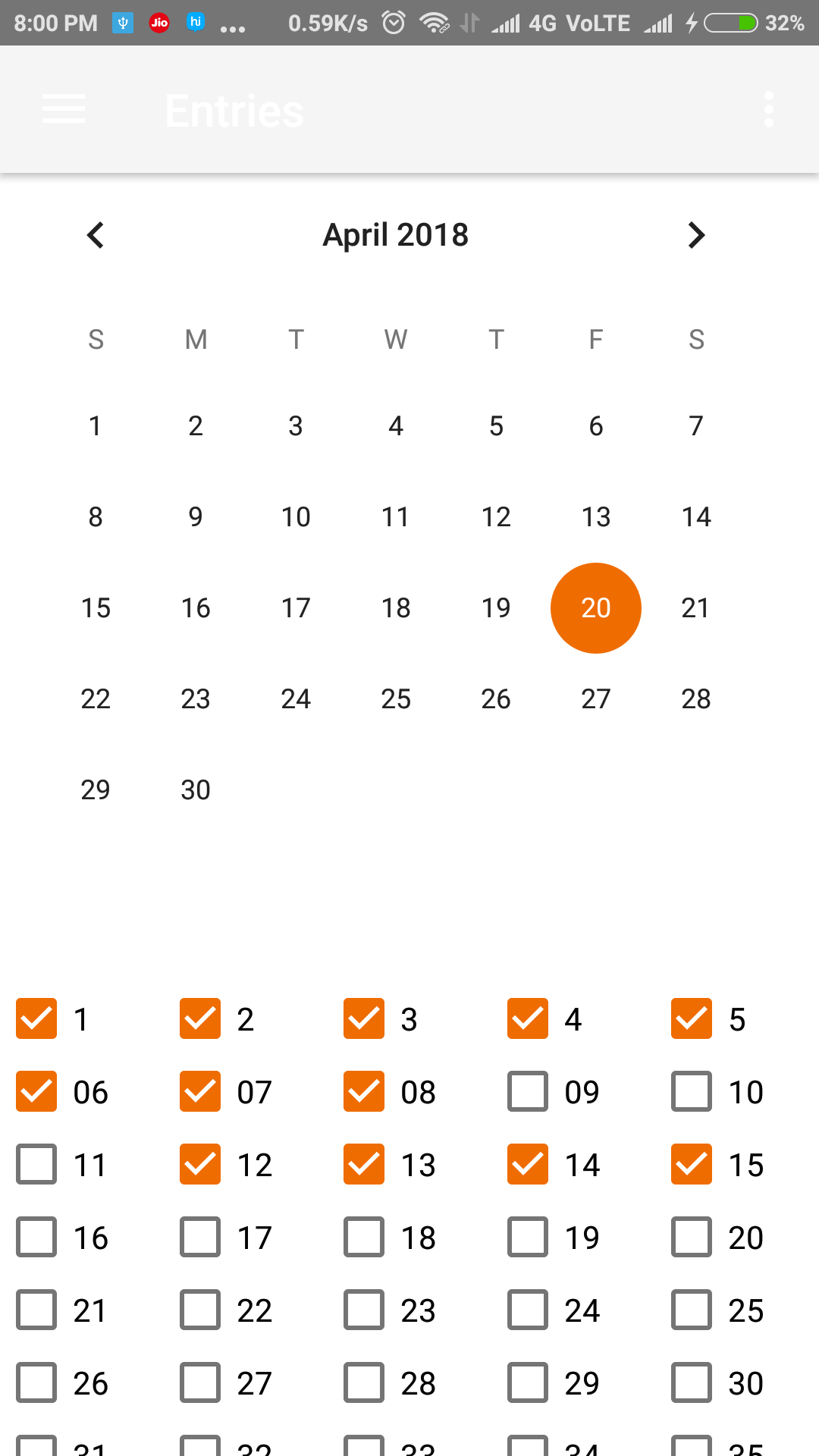
****

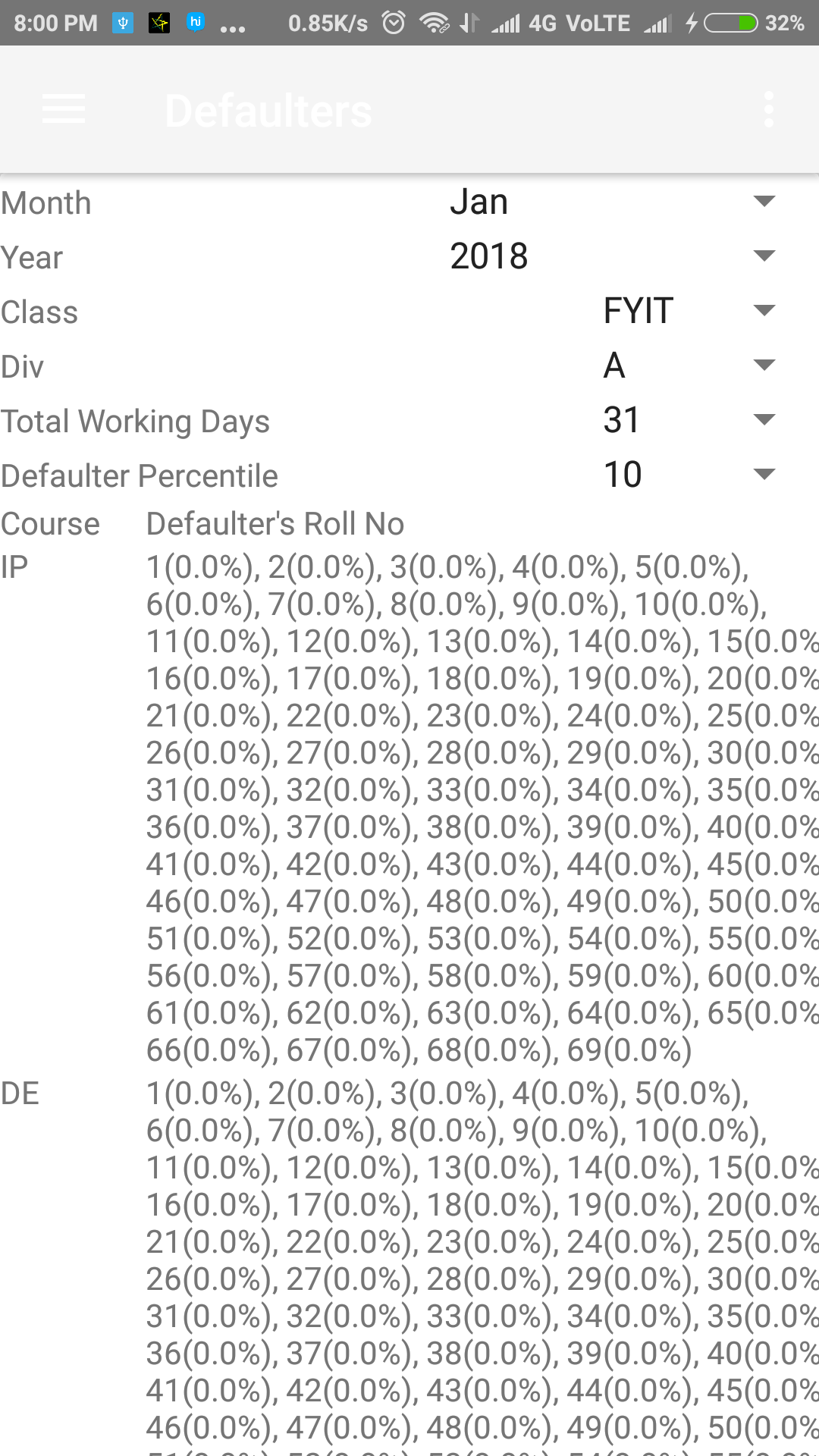
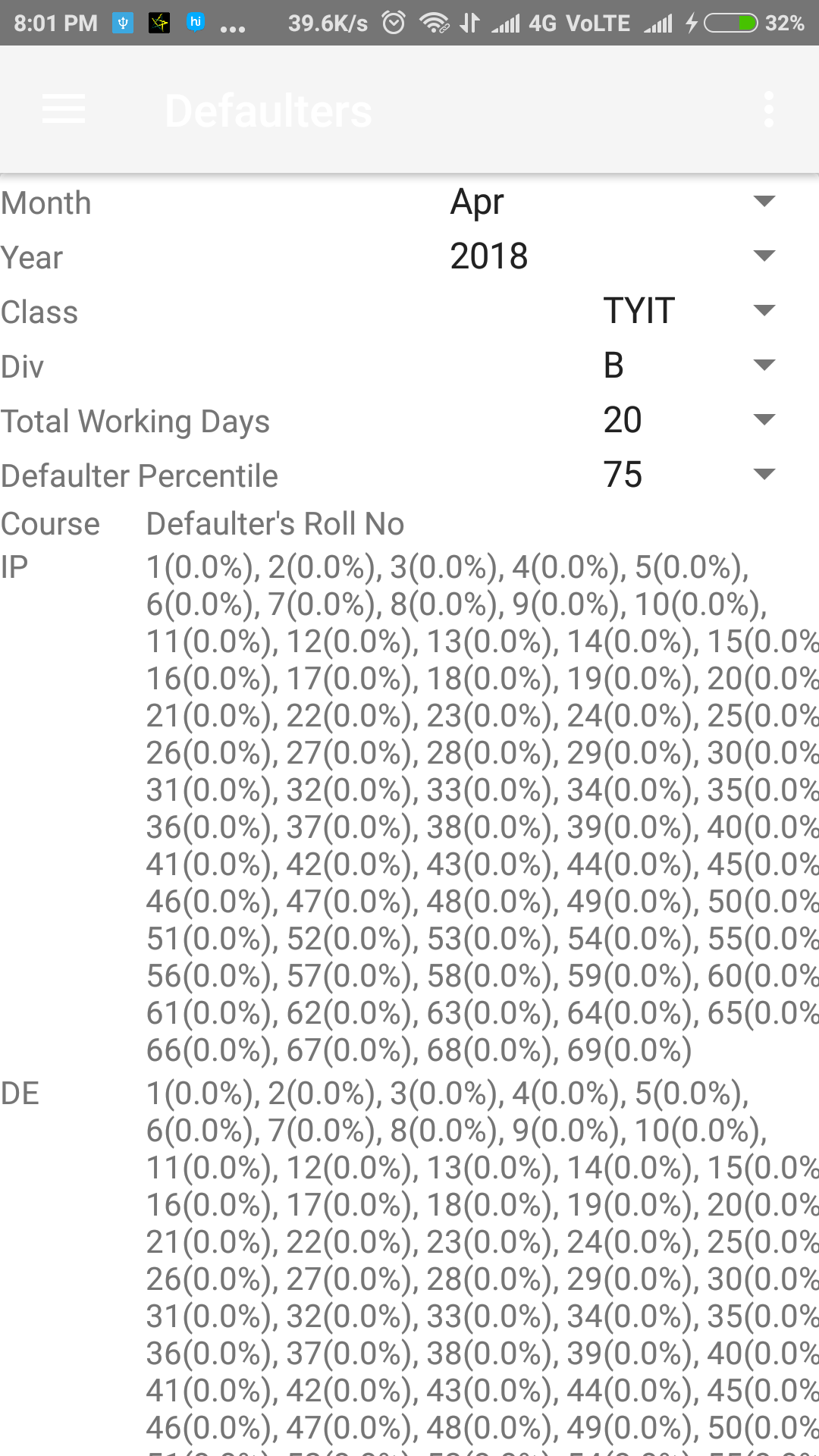
* **Homepage and navigation page**

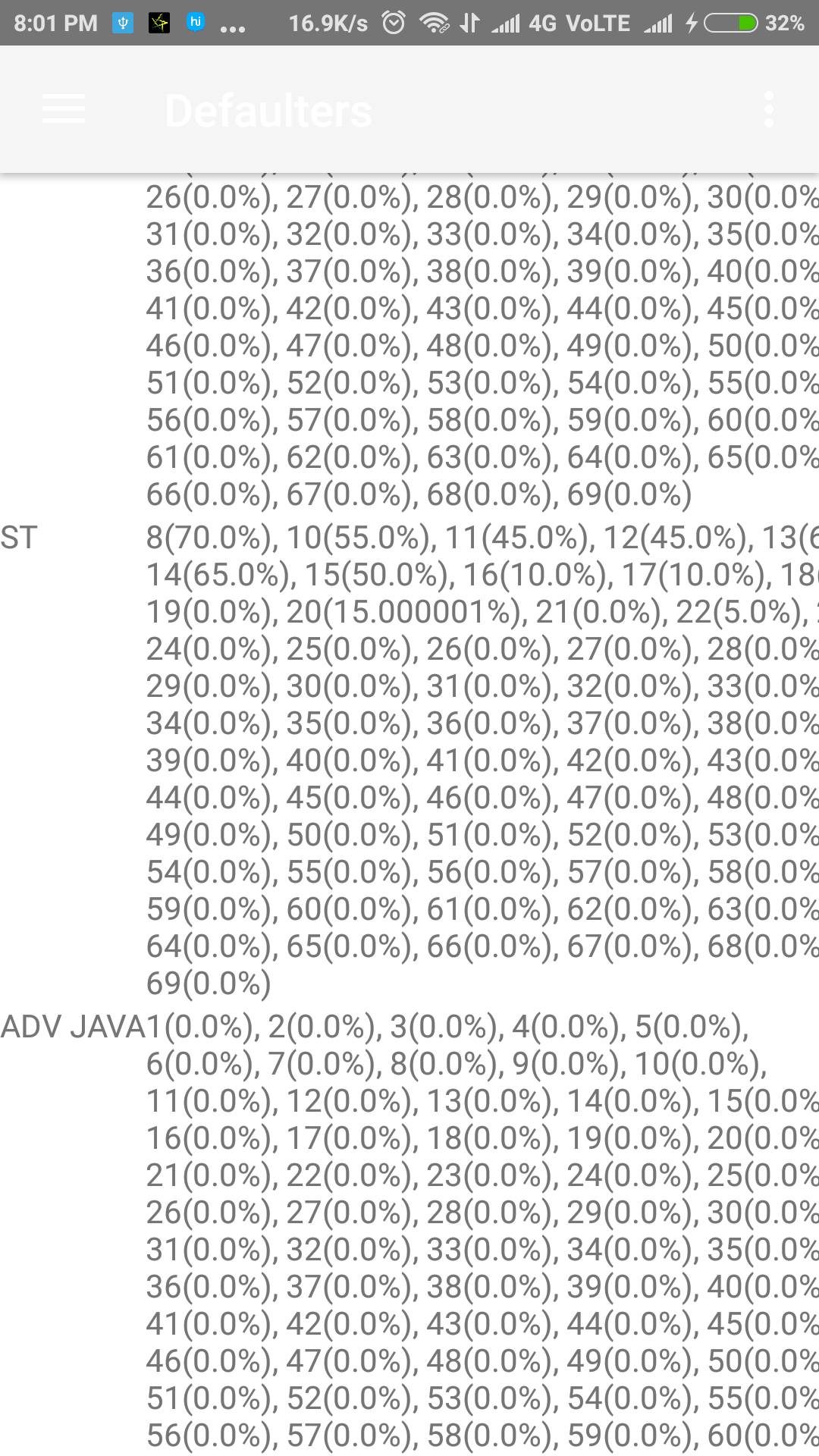
****

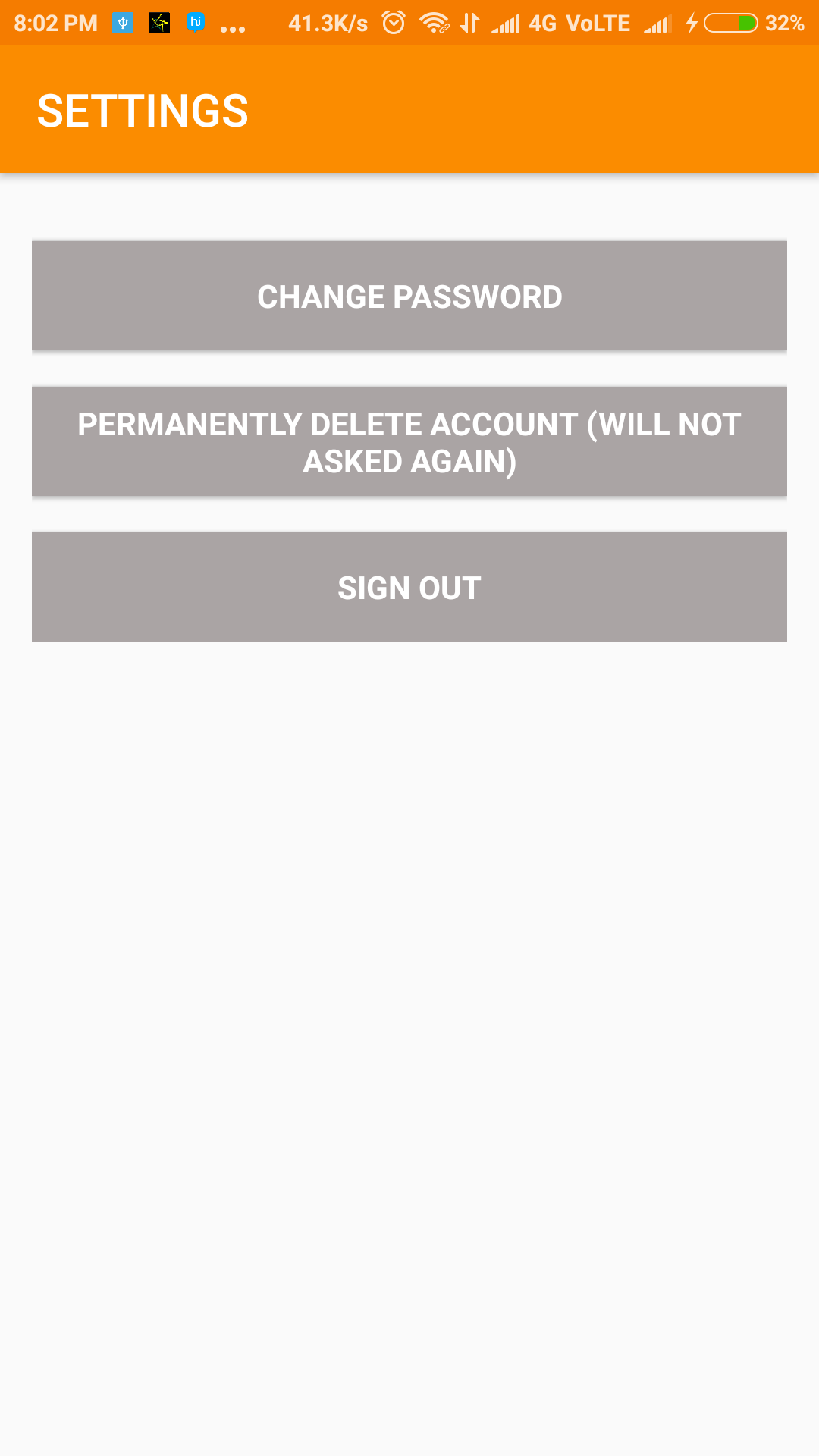
* **Entries page, attendance from 2 April to 20th April and check defaulter also.**









* **More option for user**

**PROCESS INVOLVED**

1. **Process Involved (Description of Modules in details)**
2. Registration form:-It contains a registration details in which teacher or user have to provide required information Such as, Name, Email, Password, Phone no.

So this information is stored in Database.

1. Login:-After the registration is successful then the user can do a login using his/her registered email and password.

After login the user may be,

* Admin
* Teacher

1. Authentication:-After the login details is entered then it will check the data from database whether the user is valid or not
2. Student information:-Student information is provided with a roll no.
3. Student class:-In this system classes is there such as FYBSCIT, SYBSCIT and TYBSCIT.
4. Select Division:-After the selection of class it will ask a division such as A, B and C.
5. Select Course:-In this section the subject name will be show such as Data warehouse, Internet Technology, Project Management, Geographical information System for TYBSCIT.
6. Enter attendance class and subject wise: - Now the attendance will marked using checkbox if checkbox is checked then student is present.
7. Attendance Report: - In this section monthly attendance record will store.
8. Defaulter:- the record is shown in percentage.

**TESTING METHODOLOGY**

* 1. **Unit Testing**

In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer modules together with associated control data, usage procedures, and operating procedures are tested to determine whether they are fit for use.

Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could unit be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. It forms the basis for component testing.

Unit testing is commonly automated, but may still be performed manually. The IEEE does not favor one over the other. The objective in unit testing is to isolate a unit and validate its correctness. A manual approach to unit testing may employ a step-by-step instructional document. However, automation is efficient for achieving this, and enables the many benefits listed in this article. Conversely, if not planned carefully, a careless manual unit test case may execute as an integration test case that involves many software components, and thus preclude the achievement of most if not all of the goals for unit testing.

As a consequence, unit testing is traditionally a motivator for programmers to create decoupled and cohesive code bodies. This practice promotes healthy habits in software development. Deign patterns, unit testing, and refactoring often work together so that the best solution may emerge.

Advantages:

* 1. Find problems early.
  2. Facilitates change.
  3. Simplifies integration.
  4. Documentation.
  5. Design.

Limitations and Disadvantages:

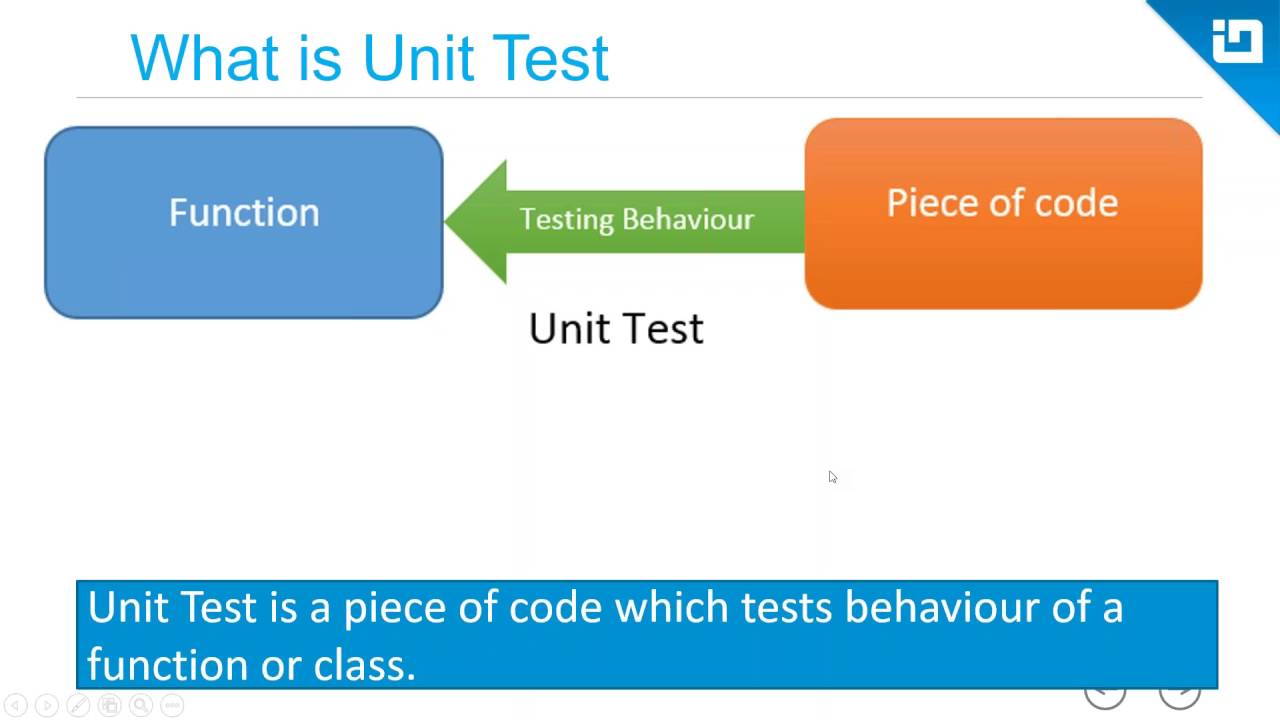
1. Decision problem.
2. Not integration testing.
3. Combinatorial problem.
4. Realism.
5. Record keeping.
6. Sustainability challenges.
7. Platform differences.
8. External work.

Applications:

1) Extreme programming.

2) Unit test frameworks.

3) Language- level unit testing support.

****

* 1. **System Testing**

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system’s compliance with its specified requirements. System testing fails within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic.

As a rule, system testing takes, as its input, all of the “integrated” software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the “inter-assemblages” and also within the system as a whole.

System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS). System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification(s).

Usually, Black Box Testing method is used. System testing is the third level of software testing performed after Integration Testing and before Acceptance Testing. Normally, independent Testers perform System Testing.

Tasks:

* System Test Plans
* Prepare
* Review
* Rework
* Baseline
* System Test Cases
* Prepare
* Review
* Rework
* Baseline
  1. **Strategies of Testing**

A test strategy is an outline that describes the testing approach of the software development cycle. It is created to inform project managers, testers, and developers about some of the testing process. This includes the testing objective, methods of testing new functions, total time and resources required for the project, and the testing environment.

Test strategies describe how the product risks of the stakeholders are mitigated at the test-level, which types of testing are to be performed, and which entry and exit criteria apply. They are created based on development design documents. System design documents are primarily used and occasionally, conceptual design documents may be referred to. Design documents describe the functionality of the software to be enabled in the upcoming release. For every stage of development design, a corresponding design, a corresponding test strategy should be created to test the new feature sets.

* Test Levels :

The test strategy describes the test levels to be performed.

* Roles and Responsibilities :

The roles and responsibilities of test leader, individual testers, project manager are to be clearly defined at a project level in this section.

* Environment requirements:

Environment requirements are an important part of the test strategy. It describes what operating systems are used for testing.

* Testing Tools:

There are two methods used in executing test cases; manual and automated and the combination of both methods is considered to be the best testing method.

* Risks and mitigation:

Any risks that will affect the testing process must be listed along with the mitigation. By documenting a risk, its occurrence can be anticipated well ahead of time.

* Test Schedule:

A test plan should make an estimation of how long it will take to complete the testing phase.

* Regression Test Approach:

Regression tests will make sure that one fix does not create some other problems in that program or in any other interface.

* Test Groups:

From the list of requirements, we can identify related areas, whose functionality is similar. These areas are called the test groups.

* Test Priorities:

Among test cases, we need to establish priorities.

* Test Status Collections and Reporting:

When test cases are executed, the test leader and the project manager must know where exactly the project stands in terms of testing activities.

* Test Records and Maintenance:

When the test cases are executed, we need to keep track of the execution details like when it is executed, who did it, how long it took, what is the result etc.

* Requirements Traceability Matrix :

In a requirements traceability matrix, the rows will have the requirements. The columns represent each document.

* Test Summary :
* The senior management may like to have a test summary on a weekly or monthly basis. If the project is very critical, they may need it even on daily basis.
  + 1. **Black Box Testing**

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: unit, integration, system and acceptance. It is sometimes referred to as specification-based testing.

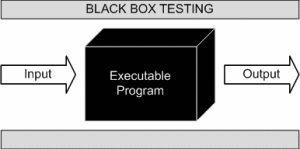
Test design techniques:

Typical black-box test techniques include:

Decision table testing, Equivalence Testing, Boundary Value Analysis, Cause-effect Graph, Error Guessing, State Transition Testing, Use Case Testing, Domain Analysis, Syntax Testing etc.

Black-box testing also known as Behavioral Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.

* **Advantages**:
* Tests are done from the user’s point of view and will help in exposing discrepancies in the specifications.
* Tester need not know programming languages or how the software has been implemented.
* Tests can be conducted by a body independent from the developers, allowing for an objective perspective and the avoidance of developer-bias.
* Test cases can be designed as soon as the specifications are complete.
* **Disadvantages:**
* Only a small number of possible inputs can be tested and many program paths will be left untested.
* Without clear specifications, which is the situation in many projects, test cases will be difficult to design.
* Tests can be redundant if the software designer/ developer has already run a test case.



* + 1. **Equivalence Class partitioning**

Equivalence partitioning or equivalence class partitioning (ECP) is a software testing technique that divides the input data of a software unit into partitions of equivalent data from which test cases can be derived. In principle, test cases are designed to cover each partition at least once.

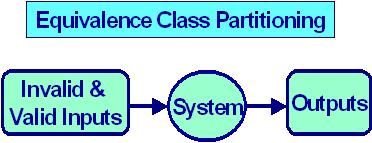
This technique tries to define test cases that uncover classes of errors, thereby reducing the total number of test cases that must be developed. An advantage of this approach is reduction in the time required for testing a software due to lesser number of test cases.

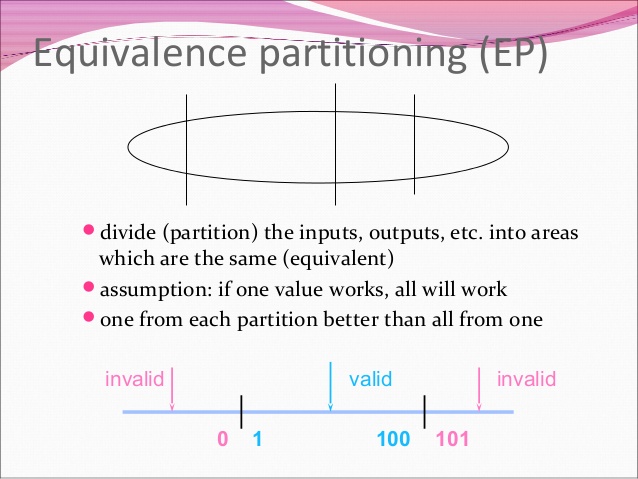
Equivalence partitioning is typically applied to the inputs of a tested component, but may be applied to the outputs in rare cases. The equivalence partitions are usually derived from the requirements specification for input attributes that influence the processing of the test object.

Equivalence Partitioning is a testing technique where input values set into classes for testing.

* Valid Input Class = Keeps all valid inputs.
* Invalid Input Class = Keeps all invalid inputs.

In equivalence-partitioning technique we need to test only one condition from each partition. This is because we are assuming that all conditions in one partition will be treated in the same way by the software. If one condition in a partition works, we assume all of the conditions in a partition does not work, then we assume that none of the conditions in that partition will work so again there is little point in testing any of these others. Similarly, if one of the conditions in a partition does not work, then we assume that none of the partition will work so again there is little point in more partition.



****

* + 1. **White Box Testing**

White-box testing also known as clear box testing or glass box testing or transparent box testing and structural testing is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the expected outputs. This is analogous to testing nodes in a circuit. White-box testing can be applied at the unit, integration and system levels of the software testing process. Although

Traditional testers tended to think of white-box testing as being done at the unit level, it is used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system-level test. Though this method of test design can uncover many errors or problems, It has the potential to miss unimplemented parts of the specification or missing requirements.

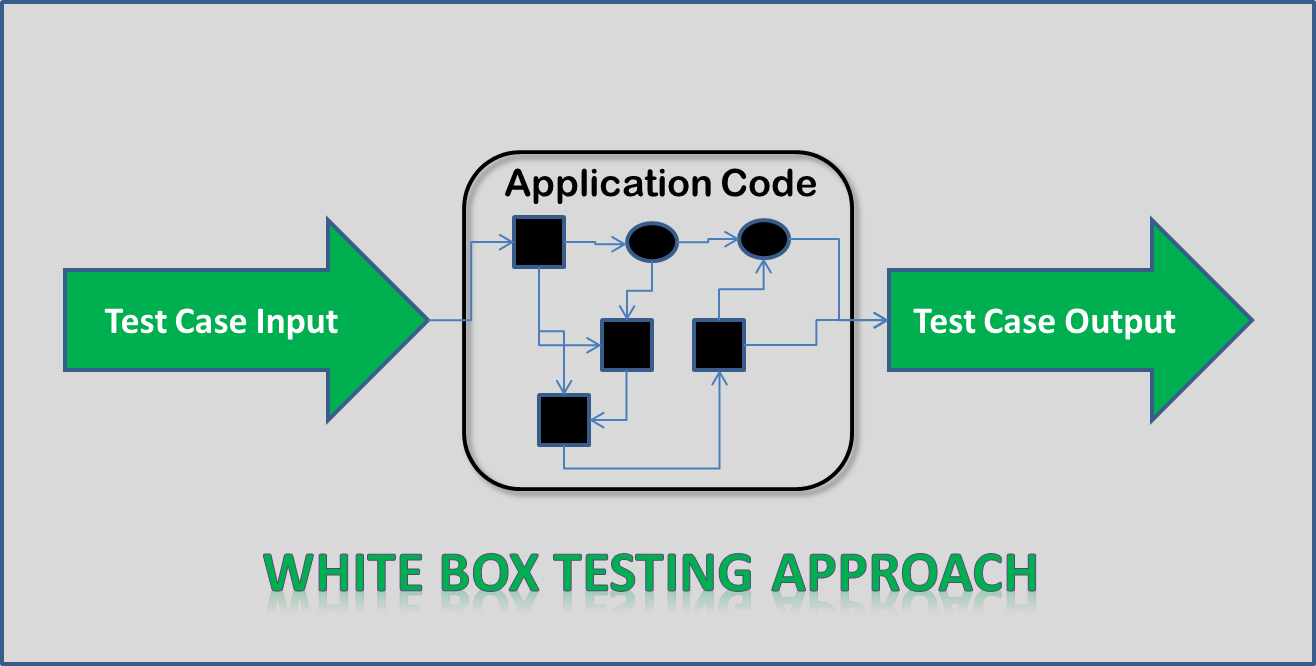
* **Advantages**

1. Side effects of having the knowledge of the

source code is beneficial to thorough testing.

1. Optimization of code becomes easy as inconspicuous bottlenecks are exposed.
2. Gives the programmer introspection because developers carefully describe any new implementation.
3. Provides traceability of tests from the source, thereby allowing future changes to the source to be easily captured in the newly added or modifies tests.
4. Easy to automate.
5. Provides clear, engineering-based rules for when to stop testing.

* **Disadvantages**

1. White-box testing brings complexity to testing because the tester must have knowledge of the program, including being a programmer.
2. On some occasions, it is not realistic to be able to test every single existing condition of the application and some conditions will be untested.
3. The tests focus on the software as it exists, and missing functionality may not be discovered.
4. The resulting test can be fragile because they are tightly coupled to the specific implementation of the thing being tested. The code under test could be written to implement the same functionality in a different way that invalidates the assumptions baked into the test. This could result in tests that fail unnecessarily or, in the worst case, tests that now give false positives and mask errors in the code. ****
   1. **TEST REPORTS:-**

**Form: Teacher Sign up**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sr. No | Test Case | Action Performed | Input | Excepted Output | Actual output | Remark |
| 1 | Enter Name | Name entered  in textbox | Husnain | It should accept | It accepted | Pass |
| 2 | Enter Name | Invalid Name entered  in textbox | Hus1324%77 | It should not accept | It did not accepted | Pass |
| 3 | Enter Email Id | Email ID entered  in textbox | khusnain6@gmail.com | It should accept | It accepted | Pass |
| 4 | Enter Email Id | Invalid Email ID entered  in textbox | Husnain321 | It should not accept | It did not accepted | Pass |
| 5 | Enter Password | Invalid Password entered  in textbox below 6 | Hus2 | It should not accept | It did not accepted | Pass |
| 6 | Enter Password | password  in textbox | Husnain321 | It should accept | It accepted | Pass |
| 7 | Enter Phone number | Valid phone number entered in Text Area | 8108876858 | It should accept | It accepted | Pass |
| 8 | Enter Phone number | Invalid phone number | 8865797568757 | It should not accept | It did not accepted | Pass |

**12 Future Enhancement**

The project has a very vast scope in future. The project can be implemented on intranet in future. Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion. With the proposed software of database Space Manager ready and fully functional the client is now able to manage and hence run the entire work in a much better, accurate and error free manner. The following are the future scope for the project.

* Discontinue of particular student eliminate potential attendance.
* Bar code Reader based attendance system.
* Individual Attendance system With photo using Student login.

### References And Bibliography:-

**Videos:**

* Android Programming Tutorial 2015 | The Complete Tutorial to Learn Android
* Developing with SQLite Android
* How to use firebase

**Websites:**

* Developer.android.com
* www.tutorialspoint.com
* Stackoverflow.com
* www.couchbase.com
* www.firebase.com

**Books:**

* Programming Android (By: Zigurd Mednieks, Laird Dornin, G. Blake Meike & Masumi Nakamura)

### Bibliography.

* [http://www.google.co.in](http://www.google.co.in/)
* <http://stackoverflow.com/questions/tagged/android>
* <http://developer.android.com/guide>
* <http://www.tutorialspoint.com/android>
* https://www.youtube.com