**CHAPTER1: PRELIMINARY INVESTIGATION**

1.1 Synopsis

1.2 Organizational Overview

1.3 Working of the current system

1.4 Limitations of the current system

1.5 The Proposed System

1.6 Advantages of the current system

1.7 Tools and Technologies to be used

1.8 Feasibility study

|  |  |
| --- | --- |
| **1.1** | **SYNOPSIS** |

# BUDGET BUCKET

# Scope

* Budget Bucket is an app which manage expenses/costs accordingly. This app allows user to list down their expenses and savings on daily basis.
* The project consists of how we arrange our expenses our savings and how we will manage our budget on daily as well as monthly basis. This project is having categories travelling, medical, shopping, food expenses which we are going to manage in this app. The user can add/browse their expense/savings as per provided categories to add it in the expenses/savings list.
* So as we know the costs of each goods and services increases frequently hence savings are necessary nowadays so we can decide our proper budget and we can save it using this app.

# Features

* It can provide convenient way of maintaining records of customers and expenses.
* This project gives us a futuristic view of how user can save their money and how they will manages the budget using this android application.
* This application can be modified and enhanced to support more operations.

# Modules

* Registration page
* User login page
* User interface pages(which will provide various options to user)
  1. Home Page
  2. Expense Categories
  3. Day Activity
  4. Month Activity
  5. Choose Date Activity
  6. About us

|  |
| --- |
| **1.2 ORGANIZATIONAL OVERVIEW** |

BUDGET BUCKET is an online android application to store expenses and savings. In addition to it will gives the savings from the expenses which provided by this app and gives you the final monthly budget.

This android application is a perfect application for the people who have habit of purchasing, shopping, travelling and then note down their overall spends anywhere so this android application provides Add note feature also with saving expenses for customers to add the note in each expenditure.

|  |
| --- |
| **1.3 WORKING OF THE CURRENT SYSTEM** |

* In the existing system, the customer need to maintain the diary to keep all the records written or use notes to add such expenses and it will take time as well as concentration

to add such data correctly.

* A record has to be kept for all the expenditures which given in this android application.
* This used to increase lot of paper work and to manually handle all this operations is a tedious task.
* So to overcome all these issues there was a need to come up with an Android application that can provide all these operations.

|  |
| --- |
| **1.4 LIMITATIONS OF THE CURRENT SYSTEM** |

* All the information is maintained in books and broaches, so there are possibilities of errors.
* The customer has to note some other expenses which we don’t have mentioned in this android application.
* It takes more effort and physical space to keep track of paper documents to find information and keep details secure.
* The expenses we have provided are limited.

|  |
| --- |
| **1.5 THE PROPOSED SYSTEM** |

* The proposed system keeps the records online. This means that the information of the expenditure and registered users will be stored in databases which can be accessed easily.
* A registered user would be able to view the expense categories and details online at their convenience.
* Here the customer can store there note’s too in this application

|  |
| --- |
| **1.6 ADVANTAGES OF PURPOSED SYSTEM** |

* With the help of BUDGET BUCKET app, you can easily add your expenses and notes in it and gives the savings as well as monthly budget.
* Users can register themselves.
* Each user has his/her own id through which they can check there savings.
* Security of data - data is well protected.
* Ensures data accuracy during accessing any data of user.
* Minimized manual data entry.
* Greater efficiency since data processing is very fast.
* User friendly and interactive interface for customers with provision of a visual confirmation that the data was stored successfully.

|  |
| --- |
| **1.7 TOOLS AND TECHNOLOGIES TO BE USED** |

* ANDROID STUDIO
* PHP
* MYSQL
* SQLite
* XML

# Android studio

**Android Studio** is the official [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) for [Google's](https://en.wikipedia.org/wiki/Google) [Android](https://en.wikipedia.org/wiki/Android_(operating_system)) [operating system,](https://en.wikipedia.org/wiki/Operating_system) built on [jetbrains'](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_(software)#Android_Development_Tools) (ADT) as primary IDE for native Android application development.

Android Studio was announced on May 16, 2013 at the [Google I/O](https://en.wikipedia.org/wiki/Google_I/O) conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0. The current stable version is 3.1 released in March 2018.

**PHP**

**PHP** Hypertext Pre processor (or simply PHP) is a server-side scripting language designed for Web development, but also used as a general-purpose programming language. It was originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Group PHP originally stood for Personal Home Page, but it now stands for the recursive initialism PHP: Hypertext Pre processor. PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management systems, and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications. The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge. The PHP language evolved without a written formal specification or standard until 2014, with the original implementation acting as the de facto standard which other implementations aimed to follow. Since 2014 work has gone on to create a formal PHP specification

# MySQL

**MySQL** is an [open-source](https://en.wikipedia.org/wiki/Open-source) [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS). Its name is a combination of "My" the name of co-founder [Michael Widenius's](https://en.wikipedia.org/wiki/Michael_Widenius) daughter, and "[SQL"](https://en.wikipedia.org/wiki/SQL), the abbreviation for [Structured Query Language.](https://en.wikipedia.org/wiki/Structured_Query_Language) The MySQL development project has made its [source code](https://en.wikipedia.org/wiki/Source_code) available under the terms of the [GNU General Public License,](https://en.wikipedia.org/wiki/GNU_General_Public_License) as well as under a variety of [proprietary](https://en.wikipedia.org/wiki/Proprietary_software) agreements. MySQL was owned and sponsored by a single [for-profit](https://en.wikipedia.org/wiki/Business) firm, the [Swedish](https://en.wikipedia.org/wiki/Sweden) company [MySQL AB,](https://en.wikipedia.org/wiki/MySQL_AB) now owned by [Oracle Corporation.](https://en.wikipedia.org/wiki/Oracle_Corporation) For proprietary use, several paid editions are available, and offer additional functionality.

MySQL is a central component of the [LAMP](https://en.wikipedia.org/wiki/LAMP_(software_bundle)) open-source web application software stack (and other "[AMP"](https://en.wikipedia.org/wiki/List_of_AMP_packages) stacks). LAMP is an acronym for "[Linux,](https://en.wikipedia.org/wiki/Linux) [Apache,](https://en.wikipedia.org/wiki/Apache_HTTP_Server) MySQL, [Perl](https://en.wikipedia.org/wiki/Perl)[/PHP](https://en.wikipedia.org/wiki/PHP)[/Python"](https://en.wikipedia.org/wiki/Python_(programming_language)). Applications that use the MySQL database include: [TYPO3,](https://en.wikipedia.org/wiki/TYPO3) [MODx,](https://en.wikipedia.org/wiki/MODx) [Joomla,](https://en.wikipedia.org/wiki/Joomla) [WordPress,](https://en.wikipedia.org/wiki/WordPress) [Simple Machines Forum,](https://en.wikipedia.org/wiki/Simple_Machines_Forum) [phpBB,](https://en.wikipedia.org/wiki/PhpBB) [MyBB,](https://en.wikipedia.org/wiki/MyBB) and [Drupal.](https://en.wikipedia.org/wiki/Drupal) MySQL is also used in many high-profile, largescale [websites,](https://en.wikipedia.org/wiki/Website) including [Google,](https://en.wikipedia.org/wiki/Google) [Facebook,](https://en.wikipedia.org/wiki/Facebook) [Twitter,](https://en.wikipedia.org/wiki/Twitter) [Flickr,](https://en.wikipedia.org/wiki/Flickr) and [YouTube.](https://en.wikipedia.org/wiki/YouTube)

**SQLite**

**SQLite** is a relational database management system contained in a C programming library. In contrast to many other database management systems, SQLite is not a client–server database engine. Rather, it is embedded into the end program.

SQLite is ACID-compliant and implements most of the SQL standard, using a dynamically and weakly typed SQL syntax that does not guarantee the domain integrity.

SQLite is a popular choice as embedded database software for local/client storage in application software such as web browsers. It is arguably the most widely deployed database engine, as it is used today by several widespread browsers, operating systems, and embedded systems (such as mobile phones), among others. SQLite has bindings to many programming languages.

**XML**

**XML** In computing, Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The W3C's XML 1.0 Specification and several other related specifications all of them free open standards define XML.

The design goals of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures such as those used in web services.

Several schema systems exist to aid in the definition of XML-based languages, while programmers have developed many application programming interfaces (APIs) to aid the processing of XML data.

|  |
| --- |
| **1.8 FEASIBILITY STUDY** |

# Introduction:

* A feasibility study is an analysis of how successfully a project can be completed.
* It is the initial design stage of any project, which brings together the elements of knowledge.
* All activities of feasibility study are directed towards helping answer the question.
* “Should we proceed with the proposed project idea?”
* “Does the proposed system contributes to the overall objectives for which the system was proposed for?”

**Types of Feasibility Study:**

1. **Technical Feasibility.**
2. **Operational Feasibility.**
3. **Economic Feasibility.**

# 1.Technical feasibility

# Since Budget Bucket is an android application which developed under Android Studio. There was an option between SQLite and MySQL databases for database connectivity. For local data storage we used SQLite database which is provided by Android Studio and to store remote access data we used MySQL database. Hence the decision was taken that MySQL database will store the remote data access. For creating connection between android and MySQL database there was a choice between JSP and PHP, for development of Budget Bucket app PHP decided to be used for database connectivity.

# 2.Operational feasibility

The Application has been developed for Android OS and will work on all the Android Device which has Android version 5.1 and above. The database which is used for this application is been hosted online using free hosting service provided by 000webhost.

# 3.Economic feasibility

The Application is been developed for the Third Year College Project. Hence, there is no mode of payment or expenses involved therefore there is no economic feasibility.

# CHAPTER 2: SYSTEM ANALYSIS

2.1 Event Table

2.2 Entity Relationship Diagram

2.3 Class Diagram

2.4 Object Diagram

2.5 Use Case Diagram

2.6 Activity Diagram

2.7 State Chart Diagram

2.8 Sequence Diagram

|  |  |  |
| --- | --- | --- |
| **2.1** |  | **EVENT TABLE** |

Events are objects or messages used when a software components wants to notify a state change to other components.

An Event model is a software architecture (a set of classes and interfaces)that determines how components occur.

On the event source side:-

* create and describe events
* trigger (or fire)events
* distribute events to interested components On the event listener side:-
* subscribe to event sources
* react to events when received
* remove the subscription to event sources when desired

Terminology often used refers to:-

* Event Source or Provider :- the sender of events
* Event :- the object sent
* Event Listener or Event Sink or Consumer :- the receiver of events

# For registered user

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Event | Trigger | Source | Response | Destination |
| Member Sign Up | Sign Up page | Unregistered user | Sign Up Successfully | Registered user |
| Member logs in | Login page | Registered user | Successfully login | Registered user |
| Member View the categories | Homepage | Registered user | Display Categories | Registered user |
| Member Adds the  expenses | Categories page | Registered user | Option to add expenses and note | Registered user |
| Checks the categories | Category Activity | Registered user | Expenses Added | Registered user |
| Add the daily savings | Expenses Page | Registered user | Savings Added | Registered user |
| View the overall Monthly Budget | Expense Page | Registered user | Monthly Savings added | Registered user/System |
| Display the monthly budget | Expense Page | Registered user | Budget Displayed | Registered user/System |
| Logout | Log out | Registered user | Homepage | Registered user |

|  |  |  |
| --- | --- | --- |
| **2.2** |  | **ENTITY RELATIONSHIP DIAGRAM** |

In software engineering Entity Relationship Model (ER Model) is a data model for describing the data or the information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database. The main components of ER model are entities (things) and the relationships that can exist among them.

An entity relationship model is the systematic way of describing and defining a business process. The process is modeled as component (entities) that are linked with each other by a relationship that express the dependencies and requirements between them such as: one building maybe divided into zero or more apartments, but one apartment can only be located in one building. Entities may have various properties (attributes) that characterize them. Diagrams are created to represent these entities, attributes and relationship graphically are called entity relationship diagrams.

An ER model is typically implemented as a database. In the case of a relational database, which stores data in tables, every row of each table represents one instance of an entity. Some data fields in these tables points to indexes in other tables; such pointers represents the relationships.

**Entity**

**Relationship**

**Attributes**

**Limitations:**

* ER model assume information content that can readily be represented in a relational database. They describe only a relational structure for this information.
* They are inadequate for systems in which the information cannot readily be represented in relational form, such as with semi structured data.



|  |  |  |
| --- | --- | --- |
| **2.3** |  | **CLASS DIAGRAM** |

The class diagram is the static diagram. It represents the static view of an application. Class Diagram is not only used for visualizing, describing and documenting different aspects of a system but also the constructing executable code of the software application.

The class diagrams describe the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams which can be mapped directly with object oriented languages.

The class diagrams shows collection of classes, interfaces, associations, collaborations and constraints. It is also known as a structural diagram.

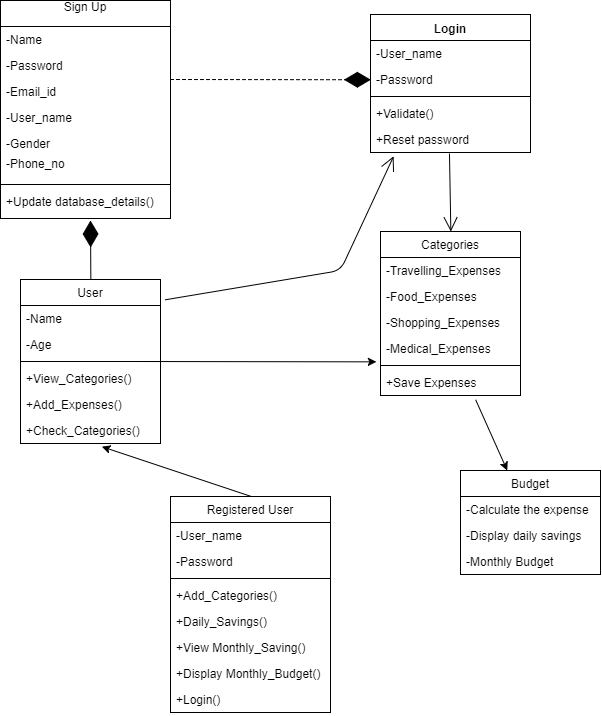
**Purpose:**

The purpose of class diagram is to model the static view of an application. The class diagrams are the only diagrams which can be mapped directly with object oriented languages and thus widely used at the time of construction.

The UML diagrams like activity diagram, sequence diagram can only give the sequence flow of the application but a class diagram is a bit different. So it is the most popular UML diagram in the coder community.

So the purpose of the class diagram can be summarized as:

* Analysis and design of static view of an application.
* Describes responsibilities of the system.
* Base for component and deployment diagrams.



|  |  |  |
| --- | --- | --- |
| **2.4** |  | **OBJECT DIAGRAM** |

Object diagrams are derived from class diagrams so object diagrams are dependent upon class diagrams.

Object diagram represent an instance of a class diagram. The basic concepts are similar for class diagrams and object diagrams. Object diagrams also represent the static view of a system but this static view is a snapshot of the system at a particular moment.

Object diagrams are used to render a set of objects and their relationships as an instance.

**Purpose:**

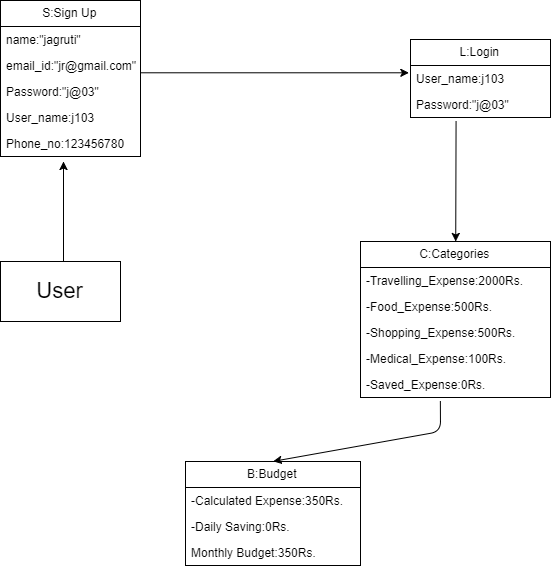
The purpose of a diagram should be understood clearly to implement it practically. The purposes of object diagrams are similar to class diagrams.

The difference is that a class diagram represents an abstract model consisting of classes and their relationships. But an object diagram represents an instance at a particular moment which is concrete in nature.

It means the object diagram is more close to system behaviour. The purpose is to capture static view of a system at a particular moment.

So the purposes of the object diagram can be summarized as:

* Forward and reverse engineering.
* Object relationship of a system.
* Understand object behaviour and their relationship from practical perspective.



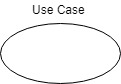
|  |  |  |
| --- | --- | --- |
| **2.5** |  | **USE CASE DIAGRAM** |

To model a system the most important aspect is to capture the dynamic behaviour. To clarify a bit in details, *dynamic behaviour* means the behaviour of the system when it is running /operating.

So only static behaviour is not sufficient to model a system rather dynamic behaviour is more important than static behaviour. In UML there are five diagrams available to model dynamic nature and use case diagram is one of them. Now as we have to discuss that the use case diagram is dynamic in nature there should be some internal or external factors for making the interaction.

These internal and external agents are known as actors. So use case diagrams are consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system.

So to model the entire system numbers of use case diagrams are used.

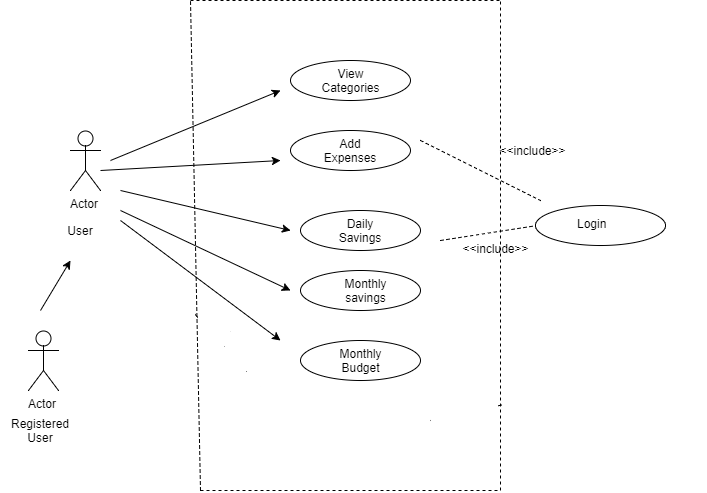
 

**Purpose:**

The purpose of use case diagram is to capture the dynamic aspect of a system. But this definition is to generic to describe the purpose.

Because other four diagrams (activity, sequence, collaboration, statechart) are also having the same purpose. So we will look into some specific purpose which will distinguish it from other four diagrams.

Use case diagram are used to gather the requirement of a system including internal and external influences. These requirements are mostly design requirements. So when system is analyzed to gather its functionalities use cases are prepared and actors are identified.



|  |  |  |
| --- | --- | --- |
| **2.6** |  | **ACTIVITY DIAGRAM** |

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. 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. This flow can be sequential, branched or concurrent. Activity diagrams deals with all type of flow control by using different elements like fork, join etc.

**Purpose:**

It captures the dynamic behaviour of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is the particular operation of the system. activity diagrams are not only used for visualizing dynamic nature of the system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is message part.

It does not show any message flow from one activity to another. Activity diagram is some time considered as the flow chart. Although the diagrams look like a flow chart but it is not. It shows different flow like parallel, branched, concurrent and single.

So the purposes can be described as:

* Draw the activity flow of a system.
* Describe the sequence from one activity to another.
* Described the parallel, branched and concurrent flow of the system.
* Draw the activity flow of a system.



|  |  |  |  |
| --- | --- | --- | --- |
| **2.7** |  |  | **STATE DIAGRAM** |

The name of the diagram itself clarifies the purpose of the diagram and other details. It describes different states of a component in a system. The states are specific to a component/object of a system.

A state chart diagram describes a state machine. Now to clarify it state machine can be defined as a machine which defines different states of an object and those states are controlled by external or internal events.

Activity diagram is a special kind of state chart diagram. As state chart diagram defines states it used to model lifetime of an object.

**Purpose:**

State chart diagram is one of the five UML diagrams used to model dynamic nature of a system. They define different states of an object during its lifetime. And these states are changed by events. So State Chart Diagrams are useful to model reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

State Chart Diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. So the most important purpose of State Chart Diagram is to model life time of an object from creation to termination.

State Chart Diagrams are also used for forward and reverse engineering of a system.But the main purpose is to model reactive system.

Following are the main purpose of using State Chart Diagrams:-

* To model dynamic aspect of a system.
* To model life time of a reactive system.
* To describe different states of an object during its life time.
* Define a state machine to model of an object.



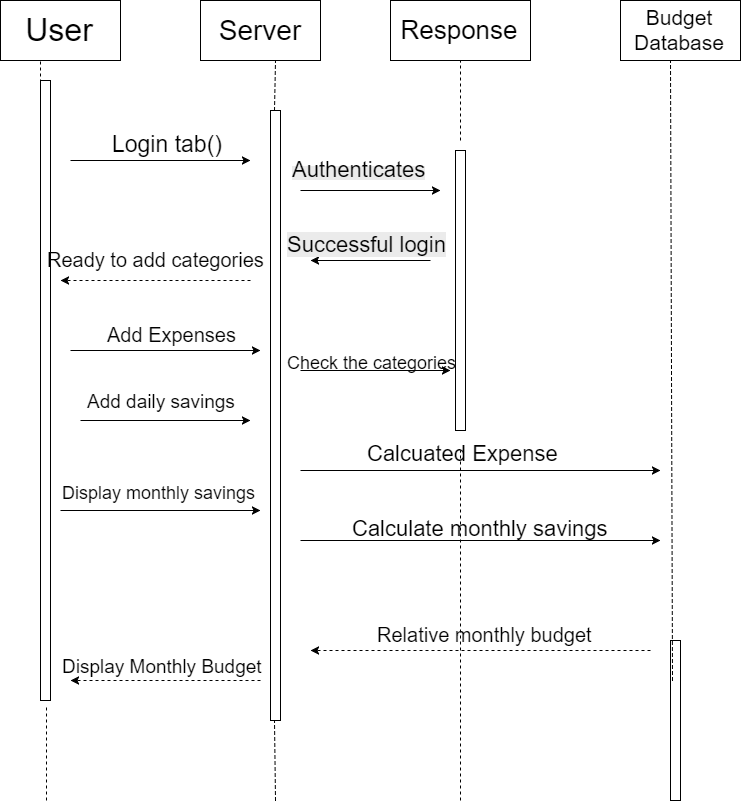
|  |  |  |
| --- | --- | --- |
| **2.8** |  | **SEQUENCE DIAGRAM** |

A Sequence Diagram is an interaction diagram that shows how processes operate with one another and what is their order. It is a construct of a Message Sequence Chart. A Sequence Diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence Diagrams are sometimes called event diagrams or event scenarios.

A Sequence Diagram shows parallel vertical lines (lifelines),different processes or objects that live simultaneously and as horizontal arrows, the messages exchanged between them, in order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

If the lifeline is that of an object, it demonstrates a role. Leaving the instance name blank can represent anonymous and unnamed instances. Messages, written with horizontal arrows with the message name written above them, display interaction. Solid arrow heads represent synchronous calls, open arrow heads represent asynchronous messages, and dashed lines represent reply messages. If a caller sends a synchronous message, it must wait until the message is done, such as invoking a subroutine. If a caller sends an asynchronous message, it can continue processing and doesn’t have to wait for a response. Asynchronous calls are present in multithreaded applications and in message-oriented middleware. Activation boxes, or method-call boxes, are opaque rectangles drawn on top of lifelines to represent that processes are being performed response to the message (Execution Specification in UML).

**For User:**



**CHAPTER 3: SYSTEM DESIGN**

3.1 Component Diagram

3.2 Package Diagram

3.3 Deployment Diagram

3.4 System Flow Chart

3.5 Structured Chart

Component Diagrams are different in terms of nature and behaviour. Component diagrams are used to model physical aspects of a system.

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

**Purpose:**

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the component used to make those functionalities

So from that point component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc.

Component diagrams can also be described as a static implementation view of system. Static implementation represents the organization of the components at a particular moment.

A single Component Diagram cannot represent the entire system but a collection of diagrams are used to represent the whole.

|  |  |  |
| --- | --- | --- |
| **3.1** |  | **COMPONENT DIAGRAM** |

So 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.



When modelling a large scale system, you would probably be working with a high volume of model elements. They describe a model from different views and different phases, hence are in different types.

UML package helps to organize and arrange model elements and diagrams into logical groups, through which you can manage a chunk of project data together.

You can also use packages to represent different views of the systems architecture .in addition, developers can use package to model the physical package or namespace structure of the application to build.

Package diagram visualizes packages and depicts the dependency, Import, access, generalization, realization and merge relationships between them. Package diagram enables you to gain a high level understanding of the collaboration among model elements through analyzing the relationships among their parent package. This also helps explain the systems architecture from a broad view.



|  |  |  |
| --- | --- | --- |
| **3.2** |  | **PACKAGE DIAGRAM** |

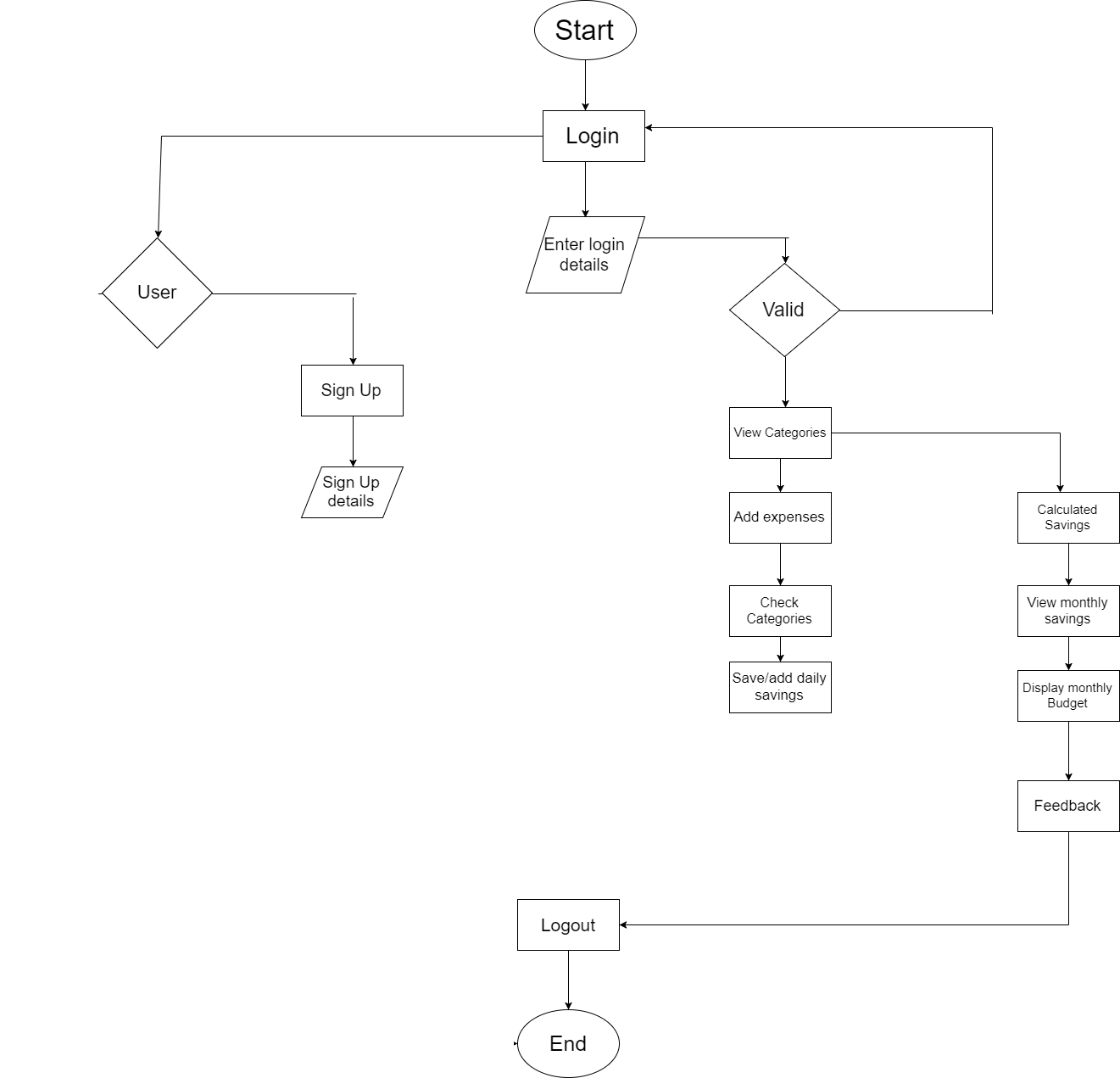
|  |
| --- |
| **3.3 DEPLOYMENT DIAGRAM** |

* Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed.
* So deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.
* Deployment diagrams are used for describing the hardware components where software components are deployed.
* Deployment diagrams shows how components are deployed in hardware.



**3.4 SYSTEM FLOW CHART**

* A flowchart is a type of diagram that represents an algorithm or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows.
* This diagrammatic representation illustrates a solution to a given problem.
* Process operations are represented in these boxes, and arrows; rather, they are implied by the sequencing of operations.
* Flowcharts are used in analysing, designing, documenting or managing a process or program in various fields.
* Flowcharts are used in designing and documenting complex processes or programs.
* Like other types of diagrams, they help visualize what is going on and thereby help the viewer to understand a process, and perhaps also find flaws, bottlenecks, and other less-obvious features within it.
* There are many different types of flowcharts, and each type has its own repertoire of boxes and notational conventions.
* The two most common types of boxes in a flowchart are:
  + A processing step, usually called activity, and denoted as a rectangular box
  + A decision, usually denoted as a diamond



|  |  |  |
| --- | --- | --- |
| **3.5** |  | **STRUCTURED CHART** |

A structure chart (SC) in software engineering and organizational theory, is a chart which shows the breakdown of a system to its lowest manageable levels.

They are used in structured programming to arrange program modules into a tree. Each module is represented by a box, which contain the module’s name. A structure chart (SC) in software engineering and organizational theory, is a chart which shows the breakdown of a system to its lowest manageable levels.

They are used in structured programming to arrange program modules into a tree. Each module is represented by a box, which contains the module’s name.

Structure diagram is a chart derived from data flow diagram. The system structure chart represents hierarchical structure of modules.

A structure chart depicts

* The size and complexity of the system, and number of a readily identifiable functions and modules within each function and whether each identifiable function is a manageable entity or should be broken down into smaller components.
* A structure chart is also used to diagram associated elements that comprise a run stream or thread. It is often developed as a hierarchal diagram, but other representations are allowable.



|  |
| --- |
| **CHAPTER 4: SYSTEM CODING** |

4.1 Site Map

4.2 Data Dictionary

4.3 Source Code

|  |  |  |
| --- | --- | --- |
| **4.1** | **SITEMAP** |  |

* LOGIN
* SIGN UP
* HOMEPAGE
* EXPENSES
* CATEGORIES
* EXPENSE ADDING
* MENU
* DAY
* MONTH
* CHOOSE DATE
* ABOUT US
* LOGOUT

|  |  |  |
| --- | --- | --- |
| **4.2** |  | **DATA DICTIONARY** |

**TABLES:**

**user\_details**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Key** |
| user\_name | varchar(30) |  |
| Name | varchar(20) |  |
| Password | varchar(30) | (user\_name)\*primary key |
| email\_id | varchar(50) |  |
| Gender | varchar(10) |  |
| phone\_number | bigInt(10) |  |
| age | bigInt(30) |  |

**categories**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Key** |
| user\_name | varchar(30) |  |
| expenses | varchar(20) |  |
| expense\_type | varchar(30) | (user\_name)#foreign key |
| date\_details | Date |  |
| Total | int |  |
| note | varchar(255) |  |
| Budget | Int |  |

**budget**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Key** |
| user\_name  calculated\_expense  daily\_savings  monthly\_savings | varchar(30)  int  int  int | (user\_name)#foreign key |

|  |  |  |
| --- | --- | --- |
| **4.3** |  | **Source Code** |

# activity\_main1.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity1"

android:background="@drawable/background03">

<ImageView

android:layout\_width="245dp"

android:layout\_height="177dp"

android:layout\_alignParentBottom="true"

android:layout\_centerHorizontal="true"

android:layout\_marginBottom="181dp"

android:src="@mipmap/app\_logo4" />

<TextView

android:id="@+id/t1"

android:layout\_width="305dp"

android:layout\_height="66dp"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

android:layout\_marginTop="64dp"

android:fontFamily="cursive"

android:text="Budget Bucket"

android:textAlignment="center"

android:textSize="40dp"

android:textStyle="italic|bold" />

</RelativeLayout>

# mainActivity1.java

package com.example.jagrutirane.budgetbucket;

import android.content.Intent;

import android.os.Bundle;

import android.os.Handler;

import android.support.v7.app.AppCompatActivity;

import android.view.View;

public class MainActivity1 extends AppCompatActivity {

private static int SPLASH\_TIME\_OUT=2000;

Intent intent;

View view;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main1);

view = this.getWindow().getDecorView();

new Handler().postDelayed(new Runnable() {

@Override

public void run() {

Intent homeintent = new Intent(MainActivity1.this,login\_page.class);

startActivity(homeintent);

finish();

}

}, SPLASH\_TIME\_OUT);

}

}

**activity\_login\_page.xml**

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android" xmlns:app="http://schemas.android.com/apk/res-auto"xmlns:tools=<http://schemas.android.com/tools>

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".login\_page"

android:background="@drawable/background03">

<ProgressBar

# android:id="@+id/progress1"

# android:layout\_width="match\_parent"

# android:layout\_height="wrap\_content"

# android:layout\_below="@+id/imageView"

# android:layout\_alignParentStart="true"

# android:layout\_marginStart="0dp"

# android:layout\_marginTop="26dp" />

# android:layout\_centerHorizontal="true"

# android:layout\_marginTop="14dp"

# app:srcCompat="@mipmap/budget\_img2" />

# <TextView

# android:id="@+id/t1"

# android:layout\_width="140dp"

# android:layout\_height="39dp"

# android:layout\_alignParentStart="true"

# android:layout\_alignParentTop="true"

# android:layout\_marginStart="13dp"

# android:layout\_marginTop="214dp"

# android:background="@drawable/input\_shape"

# android:drawableLeft="@drawable/username"

# android:icon="@drawable/username"

# android:padding="5dp"

# android:text="Username:"

# android:textColor="@color/colorAccent"

# android:textSize="20dp" />

# <android.support.design.widget.TextInputLayout

# android:layout\_width="171dp"

# android:layout\_height="50dp"

# android:layout\_alignParentEnd="true"

# android:layout\_alignTop="@+id/t1"

# android:layout\_marginEnd="18dp">

# <android.support.design.widget.TextInputEditText

# android:id="@+id/et1"

# android:layout\_width="175dp"

# android:layout\_height="42dp"

# android:layout\_alignTop="@+id/t1"

# android:layout\_alignParentEnd="true"

# android:layout\_marginEnd="17dp"

# android:hint="Username"

# android:inputType="textPersonName" />

# </android.support.design.widget.TextInputLayout>

# <android.support.design.widget.TextInputLayout

# android:layout\_width="176dp"

# android:layout\_height="49dp"

# android:layout\_alignParentEnd="true"

# android:layout\_alignTop="@+id/t2"

# android:layout\_marginEnd="15dp"

# android:layout\_marginTop="0dp"

# android:backgroundTint="@color/color"

# app:passwordToggleDrawable="@drawable/visiblity"

# app:passwordToggleEnabled="true">

# <android.support.design.widget.TextInputEditText

# android:id="@+id/et2"

# android:layout\_width="match\_parent"

# android:layout\_height="46dp"

# android:drawableRight="@drawable/visiblity\_activity"

# android:hint="Password"

# android:inputType="textPassword" />

# </android.support.design.widget.TextInputLayout>

# <TextView

# android:id="@+id/t2"

# android:layout\_width="140dp"

# android:layout\_height="45dp"

# android:layout\_alignStart="@+id/t1"

# android:layout\_alignParentBottom="true"

# android:layout\_marginStart="0dp"

# android:layout\_marginBottom="193dp"

# android:background="@drawable/input\_shape"

# android:drawableLeft="@drawable/password"

# android:icon="@drawable/password"

# android:padding="5dp"

# android:text="Password:"

# android:textColor="@color/colorAccent"

# android:textSize="20dp" />

# <Button

# android:id="@+id/b01"

# android:layout\_width="121dp"

# android:layout\_height="wrap\_content"

# android:layout\_alignParentBottom="true"

# android:layout\_centerHorizontal="true"

# android:layout\_marginBottom="105dp"

# android:background="@drawable/input\_shape"

# android:focusedByDefault="true"

# android:padding="10dp"

# android:text="Login" />

# <Button

# android:id="@+id/b02"

# android:layout\_width="125dp"

# android:layout\_height="wrap\_content"

# android:layout\_alignParentBottom="true"

# android:layout\_centerHorizontal="true"

# android:layout\_marginBottom="25dp"

# android:background="@drawable/input\_shape"

# android:focusedByDefault="true"

# android:padding="10dp"

# android:text="Sign Up" />

</RelativeLayout>

**login\_page.java**

package com.example.jagrutirane.budgetbucket;

import android.content.DialogInterface;

import android.content.Intent;

import android.net.ConnectivityManager;

import android.net.NetworkInfo;

import android.os.Bundle;

import android.support.v7.app.AlertDialog;

import android.support.v7.app.AppCompatActivity;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.ProgressBar;

import android.widget.TextView;

import android.widget.Toast;

import com.android.volley.AuthFailureError;

import com.android.volley.Request;

import com.android.volley.RequestQueue;

import com.android.volley.Response;

import com.android.volley.VolleyError;

import com.android.volley.toolbox.StringRequest;

import com.android.volley.toolbox.Volley;

import org.json.JSONException;

import org.json.JSONObject;

import java.util.HashMap;

import java.util.Map;

public class login\_page extends AppCompatActivity {

private static final String TAG = "login\_page";

Intent i;

View view;

TextView username,password;

Button login,signup;

String s,p;

ProgressBar progress;

String db\_username,db\_password;

private static String loginUrl="https://jrane.000webhostapp.com/loginUser.php";

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_login\_page);

if(SharedPreferenceManager.getInstance(this).isLoggedIn())

{

finish();

i=new Intent(login\_page.this,home\_navigation.class);

startActivity(i);

return;}

username =(TextView) findViewById(R.id.et1);

login= (Button) findViewById(R.id.b01);

password = (TextView) findViewById(R.id.et2);

s= username.getText().toString();

p=password.getText().toString();

progress=(ProgressBar)findViewById(R.id.progress1);

progress.setVisibility(View.GONE);

login.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

db\_username = username.getText().toString();

db\_password =password.getText().toString();

if (username.getText().length() == 0 && !(s.matches("^[a-z-0-9]{3,15}$")) &&

password.getText().length()==0 && !(p.matches("^[a-z-0-9]")))

{username.setError("Enter your username");

password.setError("Enter valid password");

progress.setVisibility(View.GONE);

//Toast.makeText(getApplicationContext(), "Successfully logged in", Toast.LENGTH\_SHORT).show();}

else{if (online())

{ progress.setVisibility(View.VISIBLE);

userLogin();

else {Toast.makeText(getApplicationContext(), "Login..", Toast.LENGTH\_SHORT).show(); }}}

});

signup = (Button) findViewById(R.id.b02);

signup.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

Intent signupintent = new Intent(login\_page.this, signUp\_page.class);

startActivity(signupintent);

Toast.makeText(getApplicationContext(), "Sign Up..", Toast.LENGTH\_SHORT).show();}

});}

protected boolean online()

{

ConnectivityManager connectivityManager=(ConnectivityManager)getApplicationContext().

getSystemService(CONNECTIVITY\_SERVICE);

NetworkInfo info=connectivityManager.getActiveNetworkInfo();

if(info != null && info.isConnectedOrConnecting())

{

return true;}

else

{return false;

}}

public void userLogin()

{

StringRequest stringRequest = new StringRequest(Request.Method.POST, Constant.URL\_LOGIN,

new Response.Listener<String>() {

@Override

public void onResponse(String response) {

try {

JSONObject jsonObject = new JSONObject(response);

if (jsonObject.getInt("success") == 1) {

//SharedPreferenceManager.getInstance(getApplicationContext());

Toast.makeText(getApplicationContext(), jsonObject.getString("message"), Toast.LENGTH\_SHORT).show();

Toast.makeText(getApplicationContext(), "Successfully logged in", Toast.LENGTH\_SHORT).show();

progress.setVisibility(View.VISIBLE);

i = new Intent(login\_page.this, home\_navigation.class);

startActivity(i);

//finish();

} else if (jsonObject.getInt("success") == 0) {

progress.setVisibility(View.GONE);

Toast.makeText(getApplicationContext(), jsonObject.getString("message"), Toast.LENGTH\_SHORT).show();

Log.d(TAG,"onResponse"+response);

Log.d(TAG,"onResponse value error");

} else {

Toast.makeText(getApplicationContext(), "Check your INTERNET CONNECTION", Toast.LENGTH\_SHORT).show();

} catch (JSONException e) {

e.printStackTrace();

} }},

new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG,"onErrorResponse:Login Failed");

Toast.makeText(getApplicationContext(),error.getMessage(),Toast.LENGTH\_SHORT).show();

}

}){

@Override

protected Map<String,String> getParams() throws AuthFailureError

{

Map<String,String> params = new HashMap<>();

params.put("username",db\_username);

params.put("user\_password",db\_password);

return params;}};

RequestQueue requestQueue = Volley.newRequestQueue(this);

requestQueue.add(stringRequest);}

@Override

public void onBackPressed()

{

final AlertDialog.Builder builder = new AlertDialog.Builder(login\_page.this);

builder.setMessage("Are you sure you want to Exit?");

builder.setCancelable(true);

builder.setNegativeButton("No", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

dialogInterface.cancel();

} });

builder.setPositiveButton("Yes", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

finish();

Toast.makeText(getApplicationContext(),"Click again to exit",Toast.LENGTH\_SHORT).show();

}

});

AlertDialog alertDialog = builder.create();

alertDialog.show();}}

**activity\_sign\_up\_page.xml**

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".signUp\_page"

android:background="@drawable/background03">

<ProgressBar

android:id="@+id/progressBar"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="185dp"/>

<TextView

android:id="@+id/t1"

android:layout\_width="69dp"

android:layout\_height="30dp"

android:layout\_alignParentTop="true"

android:layout\_marginTop="22dp"

android:layout\_marginEnd="-5dp"

android:layout\_toStartOf="@+id/e5"

android:icon="@drawable/username"

android:text="Name:"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="20dp" />

<EditText

android:id="@+id/e1"

android:layout\_width="wrap\_content"

android:layout\_height="56dp"

android:layout\_alignStart="@+id/e2"

android:layout\_alignBottom="@+id/t1"

android:layout\_marginStart="0dp"

android:layout\_marginBottom="-14dp"

android:ems="10"

android:hint="jagruti j. rane"

android:inputType="textPersonName"

android:visibility="visible" />

<TextView

android:id="@+id/t2"

android:layout\_width="wrap\_content"

android:layout\_height="30dp"

android:layout\_alignParentStart="true"

android:layout\_alignParentTop="true"

android:layout\_marginStart="0dp"

android:layout\_marginTop="82dp"

android:icon="@drawable/username"

android:text="Username:"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="20dp" />

<EditText

android:id="@+id/e2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignBaseline="@+id/t2"

android:layout\_alignParentEnd="true"

android:layout\_marginEnd="29dp"

android:ems="10"

android:hint="jr@03"

android:inputType="textPersonName"

android:visibility="visible" />

<TextView

android:id="@+id/t3"

android:layout\_width="98dp"

android:layout\_height="30dp"

android:layout\_alignStart="@+id/t1"

android:layout\_alignParentTop="true"

android:layout\_marginStart="0dp"

android:layout\_marginTop="139dp"

android:icon="@drawable/email"

android:text="Email id:"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="20dp" />

<EditText

android:id="@+id/e3"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignBaseline="@+id/t3"

android:layout\_alignStart="@+id/e1"

android:layout\_marginStart="0dp"

android:ems="10"

android:hint="jagrutir03@gmail.com"

android:inputType="textEmailAddress" />

<TextView

android:id="@+id/t4"

android:layout\_width="106dp"

android:layout\_height="35dp"

android:layout\_alignStart="@+id/t1"

android:layout\_alignParentTop="true"

android:layout\_marginStart="0dp"

android:layout\_marginTop="204dp"

android:icon="@drawable/password"

android:text="Password:"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="20dp" />

<android.support.design.widget.TextInputLayout

android:layout\_width="208dp"

android:layout\_height="wrap\_content"

android:layout\_alignTop="@+id/t2"

android:layout\_alignParentEnd="true"

android:layout\_marginTop="114dp"

android:layout\_marginEnd="23dp"

android:backgroundTint="@color/color"

app:passwordToggleDrawable="@drawable/visiblity"

app:passwordToggleEnabled="true">

<android.support.design.widget.TextInputEditText

android:id="@+id/e4"

android:layout\_width="match\_parent"

android:layout\_height="44dp"

android:drawableRight="@drawable/visible\_on"

android:hint="Password"

android:inputType="textPassword" />

</android.support.design.widget.TextInputLayout>

<TextView

android:id="@+id/t5"

android:layout\_width="54dp"

android:layout\_height="32dp"

android:layout\_alignStart="@+id/t1"

android:layout\_alignParentBottom="true"

android:layout\_marginStart="0dp"

android:layout\_marginBottom="218dp"

android:text="Age:"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="20dp" />

<EditText

android:id="@+id/e5"

android:layout\_width="44dp"

android:layout\_height="wrap\_content"

android:layout\_alignTop="@+id/t5"

android:layout\_toStartOf="@+id/e2"

android:ems="10"

android:inputType="number"

android:visibility="visible"/>

<TextView

android:id="@+id/t6"

android:layout\_width="wrap\_content"

android:layout\_height="38dp"

android:layout\_alignTop="@+id/t5"

android:layout\_centerHorizontal="true"

android:text="Gender:"

android:textColor="@color/colorAccent"

android:textSize="20dp"

android:textAlignment="center"/>

<RadioGroup

android:id="@+id/radioGroup1"

android:layout\_width="wrap\_content"

android:layout\_height="69dp"

android:layout\_alignParentEnd="true"

android:layout\_alignTop="@+id/t5"

android:layout\_marginEnd="26dp"

android:scrollbarAlwaysDrawHorizontalTrack="true"

android:visibility="visible">

<RadioButton

android:id="@+id/radioButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Female" />

<RadioButton

android:id="@+id/radioButton2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Male" />

<RadioButton

android:id="@+id/radioButton3"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Others"/>

</RadioGroup>

<TextView

android:id="@+id/t7"

android:layout\_width="112dp"

android:layout\_height="33dp"

android:layout\_alignStart="@+id/t1"

android:layout\_alignParentBottom="true"

android:layout\_marginStart="0dp"

android:layout\_marginBottom="139dp"

android:text="Phone No:"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="20dp" />

<EditText

android:id="@+id/editText6"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignTop="@+id/t7"

android:layout\_alignParentEnd="true"

android:layout\_marginTop="0dp"

android:layout\_marginEnd="6dp"

android:ems="10"

android:hint="1234567890"

android:inputType="phone"

android:visibility="visible" />

<Button

android:id="@+id/button1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignStart="@+id/t6"

android:layout\_alignParentBottom="true"

android:layout\_marginStart="0dp"

android:layout\_marginBottom="47dp"

android:background="@drawable/input\_shape"

android:focusedByDefault="true"

android:padding="10dp"

android:text="Submit"

android:visibility="visible" />

</RelativeLayout>

**signUp\_page.java**

package com.example.jagrutirane.budgetbucket;

import android.content.Intent;

import android.net.ConnectivityManager;

import android.net.NetworkInfo;

import android.os.Bundle;

import android.support.v7.app.AppCompatActivity;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ProgressBar;

import android.widget.RadioButton;

import android.widget.RadioGroup;

import android.widget.Toast;

import com.android.volley.AuthFailureError;

import com.android.volley.Request;

# import com.android.volley.RequestQueue;

# import com.android.volley.Response;

# import com.android.volley.VolleyError;

# import com.android.volley.toolbox.StringRequest;

# import com.android.volley.toolbox.Volley;

# import org.json.JSONException;

# import org.json.JSONObject;

# import java.util.HashMap;

# import java.util.Map;

# public class signUp\_page extends AppCompatActivity {

# private static final String TAG = "signUp\_page";

# private static String url\_create\_user="https://jrane.000webhostapp.com/RegisterUser1.php";

# private EditText editTextEmail;

# Intent i;

# View v;

# Button submit;

# RadioGroup gender;

# EditText name;

# EditText username ;

# EditText email;

# EditText password;

# EditText age;

# EditText phone\_number;

# RadioButton female,male,others;

# ProgressBar pb;

# String n,u,e,p,a;

# String selectedText,selectedText1,selectedText2;

# int db\_gender;

# String selectedGender;

# String db\_name,db\_user\_name,db\_email\_id,db\_password;

# String db\_phone\_number;

# String db\_age;

# @Override

# protected void onCreate(Bundle savedInstanceState) {

# super.onCreate(savedInstanceState);

# setContentView(R.layout.activity\_sign\_up\_page);

# if(SharedPreferenceManager.getInstance(this).isLoggedIn()){

# finish();

# i=new Intent(signUp\_page.this,login\_page.class);

# startActivity(i);

# return;}

# pb=(ProgressBar)findViewById(R.id.progressBar);

# pb.setVisibility(View.GONE);

# name = (EditText) findViewById(R.id.e1);

# username = (EditText) findViewById(R.id.e2);

# email = (EditText) findViewById(R.id.e3);

# password =(EditText) findViewById(R.id.e4);

# age=(EditText) findViewById(R.id.e5);

# phone\_number=(EditText) findViewById(R.id.editText6);

# gender=(RadioGroup)findViewById (R.id.radioGroup1);

# female=(RadioButton)findViewById(R.id.radioButton);

# String selectedText=(String)female.getText();

# male=(RadioButton)findViewById(R.id.radioButton2);

# String selectedText1=(String)male.getText();

# others=(RadioButton)findViewById(R.id.radioButton3);

# String selectedText2=(String)female.getText();

# n = name.getText().toString();

# u = username.getText().toString();

# e = email.getText().toString();

# a = age.getText().toString();

# p=password.getText().toString();

# submit = (Button) findViewById(R.id.button1);

# submit.setOnClickListener(new View.OnClickListener() {

# @Override

# public void onClick(View view) {

# int radioButtonId=gender.getCheckedRadioButtonId();

# View checkedRadio = gender.findViewById(radioButtonId);

# db\_name=name.getText().toString();

# db\_user\_name=username.getText().toString();

# db\_email\_id=email.getText().toString();

# db\_password=password.getText().toString();

# db\_gender=gender.indexOfChild(checkedRadio);

# RadioButton rdb = (RadioButton) gender.getChildAt(db\_gender);

# selectedGender = rdb.getText().toString();

# db\_phone\_number=phone\_number.getText().toString();

# db\_age=age.getText().toString();

# if (name.getText().length() == 0 && !(n.matches("^[a-z-A-Z]")) && (username.getText().length() == 0 && !(u.matches("^[a-z-A-Z,0-9]"))) && (email.getText().length() == 0

# && !(e.matches("^[a-z-0-9]") && (password.getText().length()==0 && !(p.matches("^[a-z-0-9-A-Z]]"))) )&& !(age.getText().length()==0 && (a.matches("^10-100"))))) {

# name.setError("Enter your valid Name");

# username.setError("Enter your valid Username");

# email.setError("Enter your valid EmailId");

# password.setError("Enter your correct Password");

# age.setError("Enter your valid age");

# pb.setVisibility(View.GONE);

# } else {if (online())

# { pb.setVisibility(View.VISIBLE);

# registerUser();}

# else{

# Toast.makeText(getApplicationContext(),"Not connected to internet",Toast.LENGTH\_SHORT).show();

# } }}

# });}

# protected boolean online()

# { ConnectivityManager cm = (ConnectivityManager)getApplicationContext()

# .getSystemService(CONNECTIVITY\_SERVICE);

# NetworkInfo info = cm.getActiveNetworkInfo();

# if(info!=null && info.isConnectedOrConnecting())

# {return true;}

# else

# {return false;}}

# public void registerUser() {

# StringRequest stringRequest = new StringRequest(Request.Method.POST,Constant.URL\_REGISTER,

# new Response.Listener<String>()

# { @Override

# public void onResponse(String response){

# try {

# JSONObject jsonObject = new JSONObject(response);

# Log.d(TAG, "onResponse: "+response);

# Toast.makeText(signUp\_page.this, "Sign Up Successful", Toast.LENGTH\_SHORT).show();

# Intent submitintent = new Intent(signUp\_page.this, login\_page.class);

# startActivity(submitintent);

# pb.setVisibility(View.GONE);

# } catch (JSONException e) {

# e.printStackTrace();}}

# }, new Response.ErrorListener() {

# @Override

# public void onErrorResponse(VolleyError error) {

# Log.d(TAG, "onErrorResponse: Sign up failed");

# Toast.makeText(getApplicationContext(), error.getMessage(), Toast.LENGTH\_SHORT).show();

# Log.d(TAG, "onErrorResponse: "+error);}

# }) {

# @Override

# protected Map<String, String> getParams() throws AuthFailureError{

# Map<String, String> params = new HashMap<>();

# params.put("name", db\_name);

# params.put("username", db\_user\_name);

# params.put("email\_id", db\_email\_id);

# params.put("password", db\_password);

# params.put("gender", selectedGender );

# params.put("phone\_number", db\_phone\_number);

# params.put("age", db\_age);

# return params;} };

# RequestQueue requestQueue = Volley.newRequestQueue(this);

# requestQueue.add(stringRequest);}}

# Content\_home\_navigation.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

app:layout\_behavior="@string/appbar\_scrolling\_view\_behavior"

tools:context=".home\_navigation"

android:orientation="vertical"

android:weightSum="10"

tools:showIn="@layout/app\_bar\_home\_navigation"

android:background="@drawable/background03">

<LinearLayout

android:layout\_width="wrap\_content"

android:layout\_height="35dp"

android:orientation="horizontal">

<TextView

android:id="@+id/tv1"

android:layout\_width="422dp"

android:layout\_height="wrap\_content"

android:layout\_alignParentBottom="true"

android:layout\_alignParentStart="true"

android:layout\_marginStart="0dp"

android:fontFamily="serif"

android:text="CATEGORIES:"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="25dp"> </TextView>

</LinearLayout>

<LinearLayout

android:layout\_width="358dp"

android:layout\_height="145dp"

android:layout\_marginTop="20dp"

android:orientation="horizontal">

<android.support.v7.widget.CardView

android:layout\_width="160dp"

android:layout\_height="135dp"

android:layout\_marginLeft="3dp"

android:clickable="true"

android:onClick="onClick1">

<ImageView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:src="@drawable/food\_expense" />

<TextView

android:id="@+id/tv2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:fontFamily="serif"

android:text="FOOD"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="15sp" />

</android.support.v7.widget.CardView>

<android.support.v7.widget.CardView

android:layout\_width="160dp"

android:layout\_height="135dp"

android:layout\_gravity="right"

android:layout\_marginLeft="33dp"

android:clickable="true"

android:onClick="onClick2"

android:orientation="vertical">

<ImageView

android:layout\_width="165dp"

android:layout\_height="137dp"

android:src="@drawable/medical\_expense" />

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:fontFamily="serif"

android:text="MEDICAL"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="15sp" />

</android.support.v7.widget.CardView></LinearLayout>

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="131dp"

android:layout\_marginTop="10dp"

android:orientation="horizontal">

<ImageView

android:layout\_width="110dp"

android:layout\_height="125dp"

android:layout\_centerInParent="true"

android:src="@drawable/circular\_view" />

<TextView

android:layout\_width="112dp"

android:layout\_height="30dp"

android:layout\_alignStart="@+id/editBudget"

android:layout\_alignParentTop="true"

android:layout\_marginStart="-15dp"

android:layout\_marginTop="9dp"

android:text="Expense"

android:textAllCaps="true"

android:textColor="@color/colorAccent"

android:textSize="25sp" />

<EditText

android:id="@+id/editBudget"

android:layout\_width="72dp"

android:layout\_height="wrap\_content"

android:layout\_alignParentBottom="true"

android:layout\_centerHorizontal="true"

android:layout\_marginBottom="34dp"

android:focusable="false" />

</RelativeLayout>

<LinearLayout

android:layout\_width="409dp"

android:layout\_height="149dp"

android:layout\_marginTop="25dp">

<android.support.v7.widget.CardView

android:layout\_width="160dp"

android:layout\_height="135dp"

android:layout\_marginStart="4dp"

android:clickable="true"

android:onClick="onClick3"

android:orientation="vertical">

<ImageView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:src="@drawable/shopping\_expense" />

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:fontFamily="serif"

android:text="SHOPPING"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="15sp" />

</android.support.v7.widget.CardView>

<android.support.v7.widget.CardView

android:layout\_width="160dp"

android:layout\_height="135dp"

android:layout\_marginLeft="35dp"

android:clickable="true"

android:onClick="onClick4"

android:orientation="horizontal">

<ImageView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:src="@drawable/travelling\_expense" />

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:fontFamily="serif"

android:text="TRAVELLING"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="15sp" />

</android.support.v7.widget.CardView>

</LinearLayout>

</LinearLayout>

# activity\_main.xml

*<?*xml version="1.0" encoding="utf-8"*?>*

<RelativeLayout

xmlns:android="http://schemas.android.com/apk/res/android" xmlns:tools="http://schemas.android.com/tools" xmlns:app="http://schemas.android.com/apk/res-auto" android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:background="@drawable/bg2"

tools:context="com.example.sneha.myapplication.MainActivity">

<EditText android:layout\_width="250dp" android:layout\_marginLeft="60dp" android:layout\_height="wrap\_content" android:hint="User Name"

android:drawableLeft="@drawable/ic\_action\_user"

android:textSize="25dp"

android:textColor="@android:color/black" android:layout\_marginTop="100dp" android:id="@+id/et1"/>

<EditText android:id="@+id/et2" android:layout\_width="250dp" android:layout\_height="wrap\_content" android:hint="Password"

android:drawableLeft="@drawable/ic\_action\_pass"

android:textSize="25dp"

android:textColor="@android:color/black" android:layout\_marginTop="17dp" android:ems="10" android:inputType="textPassword" android:layout\_below="@+id/et1" android:layout\_alignLeft="@+id/et1"

android:layout\_alignStart="@+id/et1" />

<Button android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="SIGN IN" android:textSize="30dp" android:layout\_below="@id/et2" android:layout\_marginLeft="130dp" android:layout\_marginTop="30dp" android:id="@+id/signin"

android:onClick="signin"/>

<TextView android:id="@+id/reg" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_below="@id/signin" android:layout\_marginLeft="135dp" android:layout\_marginTop="30dp" android:onClick="reg" android:text="Register Now" android:textColor="@android:color/holo\_red\_dark"

android:textSize="20dp"

android:textStyle="bold" />

<TextView android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_below="@id/reg" android:layout\_marginLeft="125dp" android:layout\_marginTop="20dp" android:text="Forgot Password" android:textColor="@android:color/holo\_red\_dark"

android:textSize="20dp"

android:onClick="forpass"/>

</RelativeLayout>

# home\_navigation.java

package com.example.jagrutirane.budgetbucket;

import android.content.Intent;

import android.os.Bundle;

import android.support.design.widget.NavigationView;

import android.support.v4.view.GravityCompat;

import android.support.v4.widget.DrawerLayout;

import android.support.v7.app.ActionBarDrawerToggle;

import android.support.v7.app.AppCompatActivity;

import android.support.v7.widget.Toolbar;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.EditText;

import android.widget.Toast;

import java.util.ArrayList;

public class home\_navigation extends AppCompatActivity

implements NavigationView.OnNavigationItemSelectedListener { @Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_home\_navigation);

Toolbar toolbar = findViewById(R.id.toolbar);

setSupportActionBar(toolbar);

budgetDatabase bgDatabase = new budgetDatabase(getApplicationContext());

ArrayList<ExpensePOJO> data = bgDatabase.getCategories();

DrawerLayout drawer = findViewById(R.id.drawer\_layout);

ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(

this, drawer, toolbar, R.string.navigation\_drawer\_open, R.string.navigation\_drawer\_close);

drawer.addDrawerListener(toggle);

toggle.syncState();

NavigationView navigationView = findViewById(R.id.nav\_view);

navigationView.setNavigationItemSelectedListener(this);

int total = getTotal(data);

int budget=getBudget(data);

EditText editTotal =(EditText) findViewById(R.id.editBudget);

editTotal.setText(""+total);

Toast.makeText(getApplicationContext(),"Expense Added"+ editTotal.getText().toString(),Toast.LENGTH\_SHORT).show();

editTotal.setText(""+budget);

Toast.makeText(getApplicationContext(),"Monthly Budget",Toast.LENGTH\_SHORT).show();}

public int getTotal(ArrayList<ExpensePOJO> recieved){

int total=0;

for(ExpensePOJO each : recieved){

total = total+each.getExpenses();

EditText editTotal=(EditText)findViewById(R.id.editBudget);

editTotal.setText(""+total);}

return total;}

public int getBudget(ArrayList<ExpensePOJO> gives)

{int budget=0;

int total=0;

for(ExpensePOJO each:gives)

{budget = ((total)/30);

EditText editBudget=(EditText) findViewById(R.id.editBudget);}

return budget;}

@Override

public void onBackPressed() {

DrawerLayout drawer = findViewById(R.id.drawer\_layout);

if (drawer.isDrawerOpen(GravityCompat.START)) {

drawer.closeDrawer(GravityCompat.START);

} else {

super.onBackPressed();}}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.home\_navigation, menu);

return true;}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

int id = item.getItemId();

if (id == R.id.action\_aboutUs) {

Intent i=new Intent(home\_navigation.this,aboutUs.class);

startActivity(i);

return true;}

return super.onOptionsItemSelected(item);}

@SuppressWarnings("StatementWithEmptyBody")

@Override

public boolean onNavigationItemSelected(MenuItem item) {

budgetDatabase bgDatabase = new budgetDatabase(getApplicationContext());

ArrayList<ExpensePOJO> data = bgDatabase.getCategories();

int id = item.getItemId();

if (id == R.id.nav\_choose\_date)

{Intent intent1 = new Intent(home\_navigation.this,choose\_date.class);

startActivity(intent1);

Toast.makeText(getApplicationContext(),"Choose date selected",Toast.LENGTH\_SHORT).show();}

else if (id == R.id.nav\_Day)

{Intent intent2 =new Intent(home\_navigation.this,day.class);

startActivity(intent2);

Toast.makeText(getApplicationContext(),"Day selected",Toast.LENGTH\_SHORT).show();

int total = getTotal(data);

EditText editTotal =(EditText) findViewById(R.id.editBudget);

editTotal.setText(""+total);}

else if (id == R.id.nav\_month)

{Intent intent3 = new Intent(home\_navigation.this,month.class);

startActivity(intent3);

Toast.makeText(getApplicationContext(),"Month Selected",Toast.LENGTH\_SHORT).show();

int total = getTotal(data);

int budget=getBudget(data);

EditText editTotal =(EditText) findViewById(R.id.editBudget);

Toast.makeText(getApplicationContext(),"Expense Added"+ editTotal.getText().toString(),Toast.LENGTH\_SHORT).show();

editTotal.setText(""+budget);

Toast.makeText(getApplicationContext(),"Monthly Budget",Toast.LENGTH\_SHORT).show();}

else if (id == R.id.logOut)

{Intent intent = new Intent(home\_navigation.this,login\_page.class);

startActivity(intent);

Toast.makeText(getApplicationContext(),"Logout",Toast.LENGTH\_SHORT).show();}

DrawerLayout drawer = findViewById(R.id.drawer\_layout);

drawer.closeDrawer(GravityCompat.START);

return true;}

public void onClick1(View view)

{budgetDatabase bgDatabase = new budgetDatabase(getApplicationContext());

ArrayList<ExpensePOJO> data = bgDatabase.getCategories();

Intent i1 = new Intent(home\_navigation.this,food\_expense.class);

startActivity(i1);

int total = getTotal(data);

EditText editTotal =(EditText) findViewById(R.id.editBudget);

editTotal.setText(""+total);

Toast.makeText(getApplicationContext(),"Food expense Added"+ editTotal.getText().toString(),Toast.LENGTH\_SHORT).show();}

public void onClick2(View view)

{budgetDatabase bgDatabase = new budgetDatabase(getApplicationContext());

ArrayList<ExpensePOJO> data = bgDatabase.getCategories();

Intent i2 = new Intent(home\_navigation.this,medical\_expense.class);

startActivity(i2);

int total = getTotal(data);

EditText editTotal =(EditText) findViewById(R.id.editBudget);

editTotal.setText(""+total);

Toast.makeText(getApplicationContext(),"Medical Expense Added"+ editTotal.getText().toString(),Toast.LENGTH\_SHORT).show();}

public void onClick3(View view)

{budgetDatabase bgDatabase = new budgetDatabase(getApplicationContext());

ArrayList<ExpensePOJO> data = bgDatabase.getCategories();

Intent i3 = new Intent(home\_navigation.this,shopping\_expense.class);

startActivity(i3);

int total = getTotal(data);

EditText editTotal =(EditText) findViewById(R.id.editBudget);

editTotal.setText(""+total);}

public void onClick4(View view)

{budgetDatabase bgDatabase = new budgetDatabase(getApplicationContext());

ArrayList<ExpensePOJO> data = bgDatabase.getCategories();

Intent i4 = new Intent(home\_navigation.this,travelling\_expense.class);

startActivity(i4);

int total = getTotal(data);

EditText editTotal =(EditText) findViewById(R.id.editBudget);

editTotal.setText(""+total);}}

**CHAPTER5: SYSTEM IMPLEMENTATION**

5.1 H/W and S/W Requirements

5.2 Screen/Report Layouts

|  |  |  |
| --- | --- | --- |
| **5.1** |  | **H/W AND S/W REQUIREMENTS** |

**Hardware Requirements:**

The minimum Hardware and System Software requirements for development and using this are:

Processor: 1.2GHz and above

RAM: Minimum requirement is 2 GB

**Software Requirements:**

**Client side requirements**

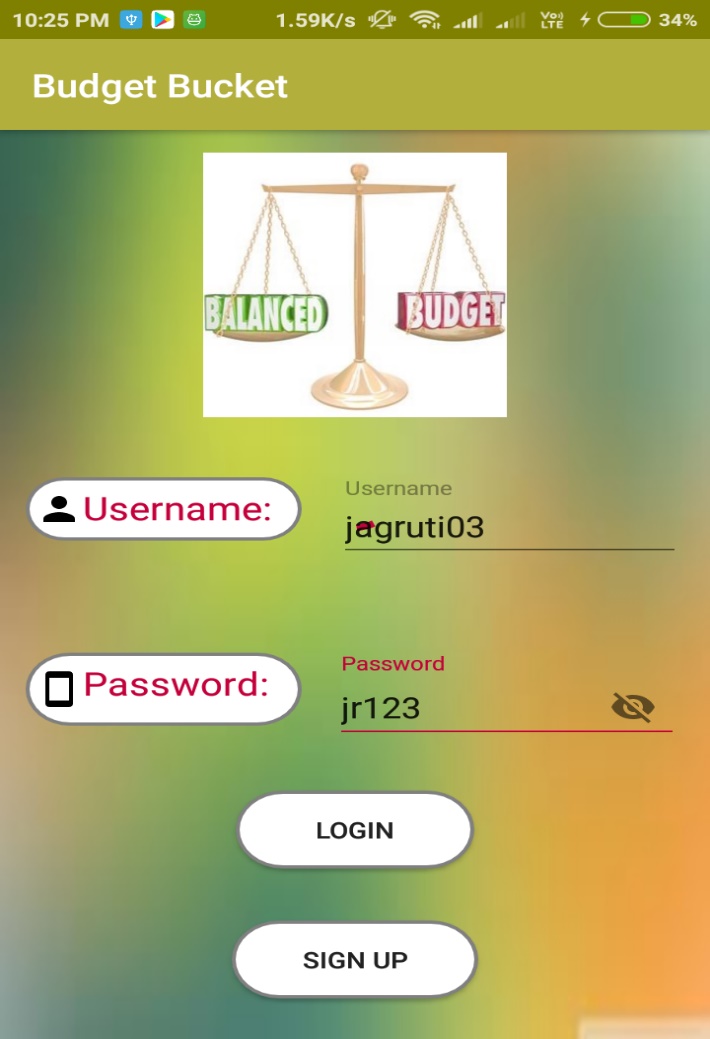
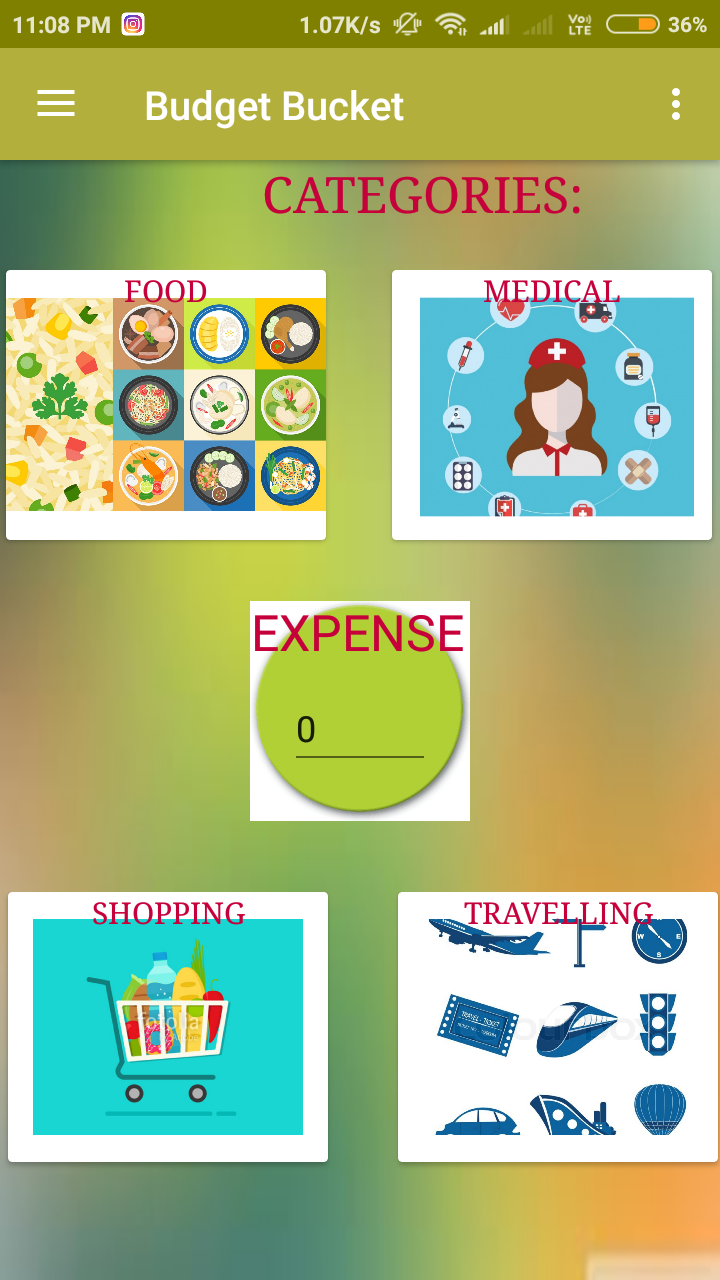
Android version: 5.1 and above

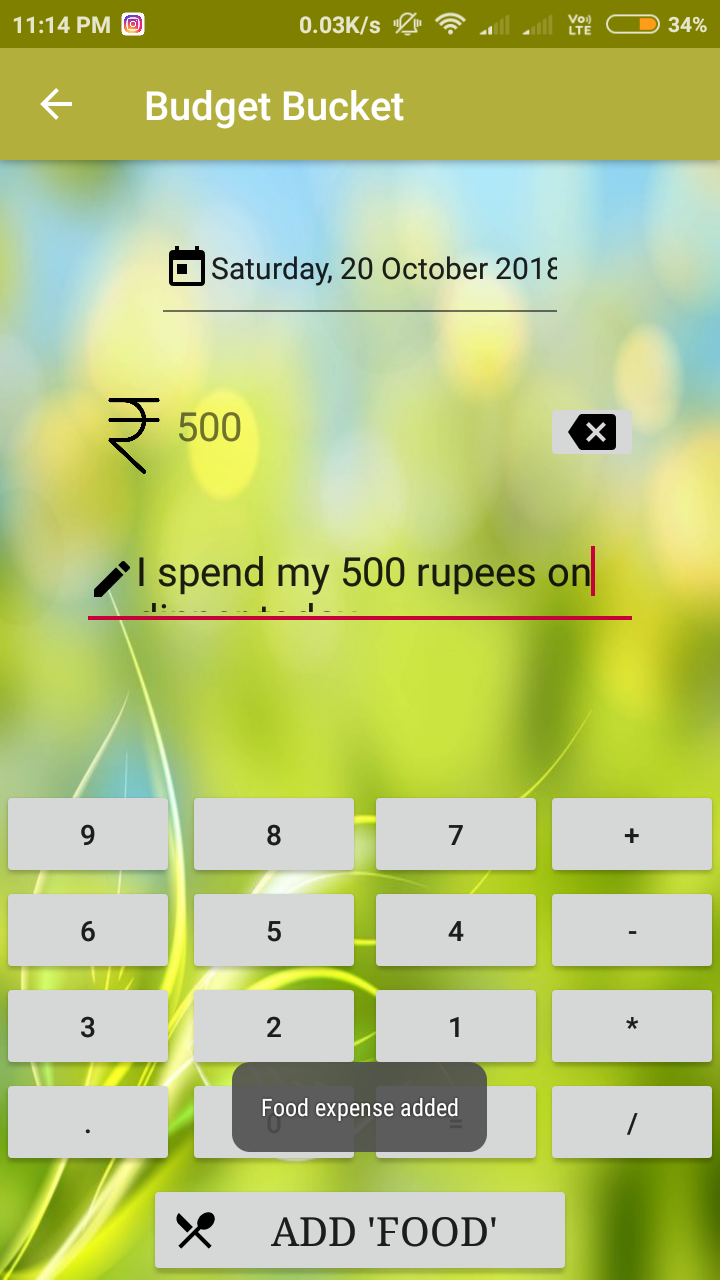
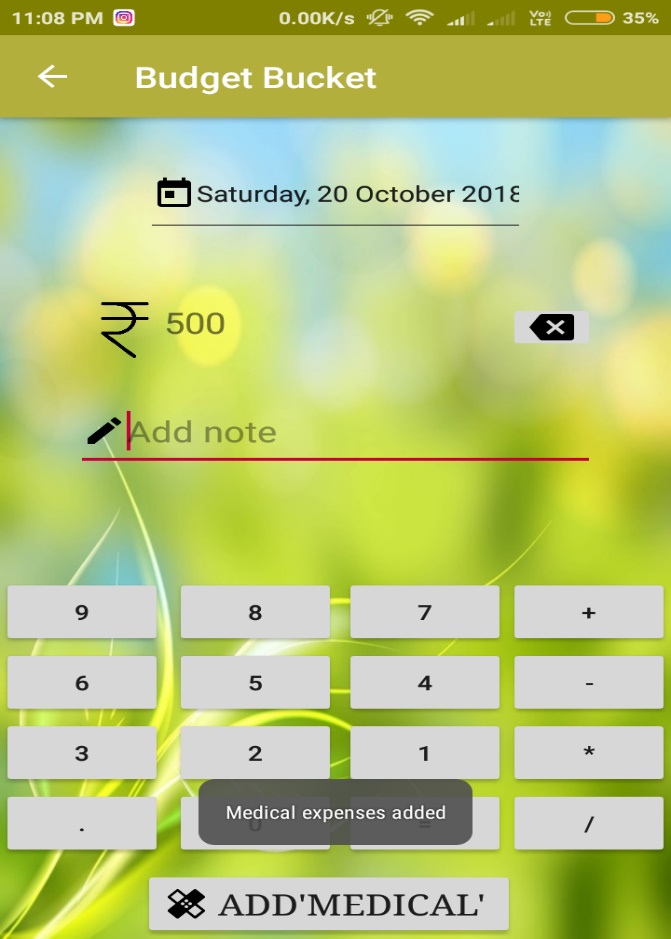
**Sever side requirements**

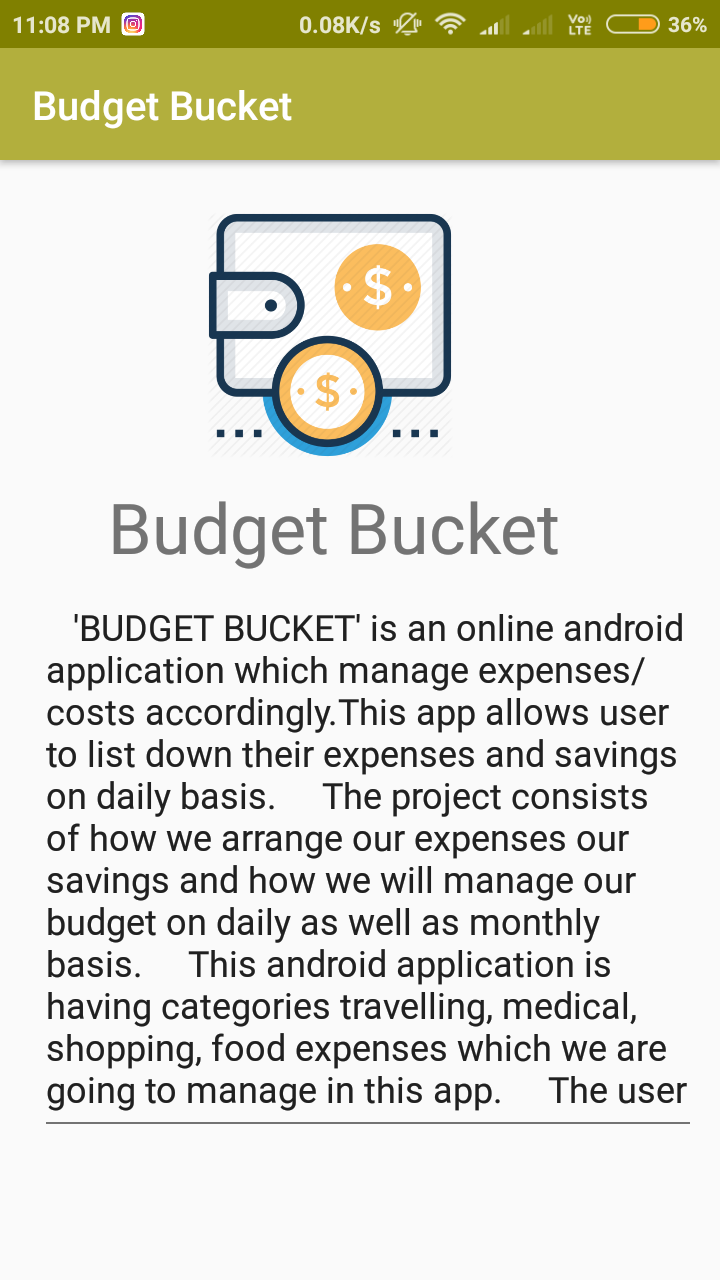
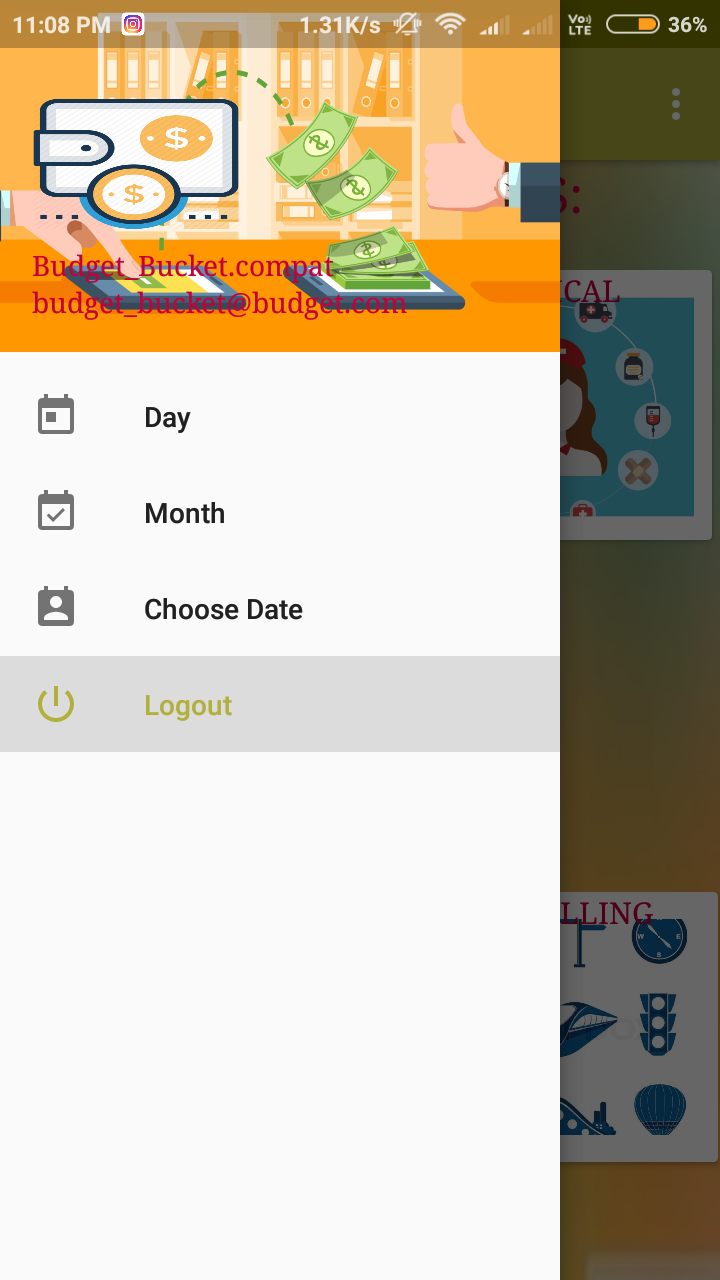
Windows 10

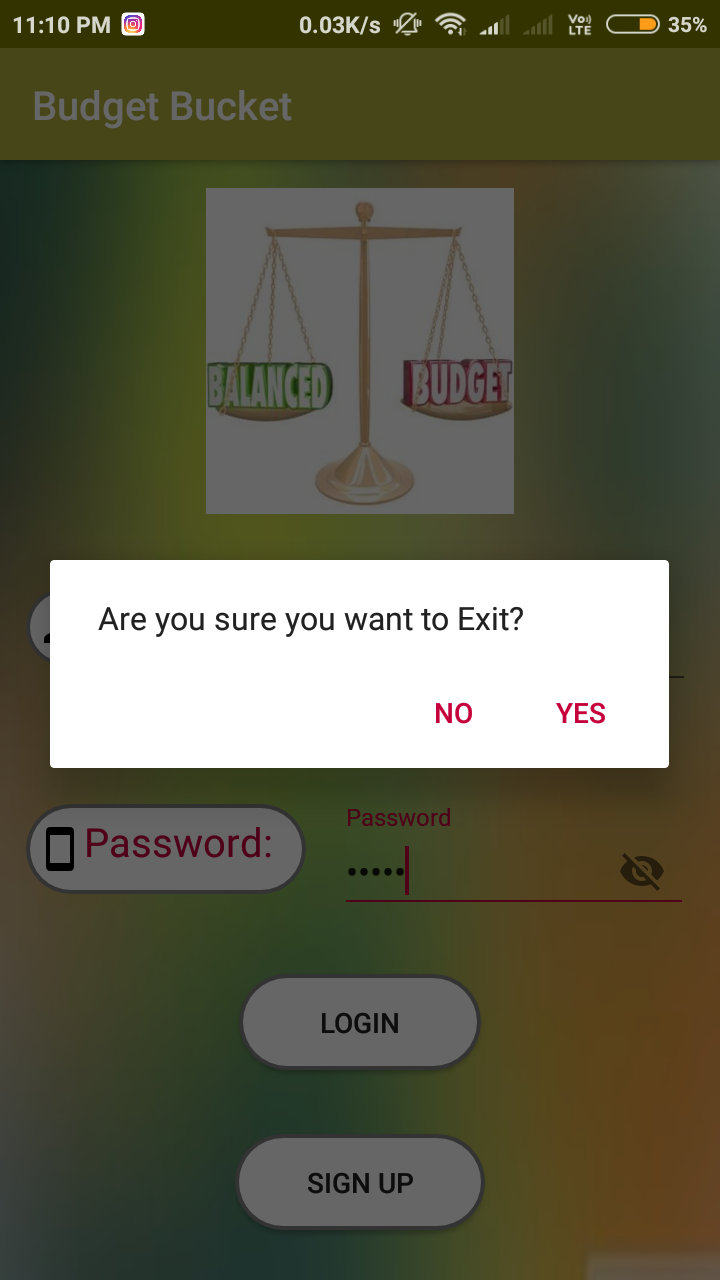
MySQL Database

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **5.2** |  | **SCREEN LAYOUTS** |   **App Layouts:** |  |



**CHAPTER6: FUTURE ENHANCEMENT**

If there is any rectification or enhancement proposed by the user, the application will be modified and the desired module will be implemented into current application. Future enhancements include developing the following as stated below:

* To include more expenses as per requirements
* User can maintain their own budgets with more expenses.
* Guest user can also able to add their expenses.
* To update the application as per the customer’s demands.

**CHAPTER7: REFERENCE AND BIBILOGRAPHY**

**Websites:**

* [https://www.youtube.com](https://www.youtube.com/)
* [https://www.google.com](https://www.google.com/)
* [https://stackoverflow.com](https://stackoverflow.com/)
* [https://www.w3schools.com](https://www.w3schools.com/)
* [https://www.tutorialspoint.com](https://www.tutorialspoint.com/)

**Books:**

* Android Programming for beginners.