**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

JNANA SANGAMA, BELAGAVI – 590 018



**An Internship Project Report**

**on**

***Finance App***

Submitted in partial fulfillment of the requirements for the VIII Semester of degree of **Bachelor of Engineering in Information Science and Engineering** of Visvesvaraya Technological University, Belagavi

**by**

**Venkatesh Ramakrishnan**

**1RN16IS119**

**Under the Guidance of**

**Mr. T.S Bhagavath Singh**

**Associate Professor**

**Department of ISE**



**Department of Information Science and Engineering**

**RNS Institute of Technology**

**Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post, Channasandra, Bengaluru-560098**

**2021-2022**

**RNS INSTITUTE OF TECHNOLOGY**

Dept. of ISE, RNSIT 2021-2022 Page 2

**Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,**

**Channasandra, Bengaluru - 560098**

**DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING**



**CERTIFICATE**

Certified that the Internship work entitled ***FINANCE APP*** has been successfully completed by **Venkatesh Ramakrishnan (1RN16IS119)** a bonafide student of **RNS Institute of Technology, Bengaluru** in partial fulfillment of the requirements of 8th semester for the award of degree in **Bachelor of Engineering in Information Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during academic year **2021-2022**. The internship report has been approved as it satisfies the academic requirements in respect of internship work for the said degree.

|  |  |  |
| --- | --- | --- |
| **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
| **Mr. T.S.Bhagavath Singh** | **Dr. Suresh L** | **Dr. M K Venkatesha** |
| Internship Guide | Professor and HoD | Principal |
| Associate Professor | Department of ISE | RNSIT |
| Department of ISE | RNSIT |  |
|  |  |  |
|  | **External Viva** |  |
| **Name of the Examiners** |  | **Signature with Date** |
| **1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | | **1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
| **2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | | **2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |

# ABSTRACT

Money management is an important and unavoidable activity which most people dread. Money management not only involves handling investments but also includes managing multiple accounts and tracking expenses. Each of these activities involves accessing information from different locations and so collecting and consolidating monetary information is not easy.

Currently, there are some stand alone personal finance applications which address different issues of financial management. There are individual software packages available for portfolio management, budgeting and investment tracking. But each of these applications is limited to only a specific aspect of personal finance. Also, these applications being stand-alone in nature, their usage is limited to the specific system on which they have been installed.

In our project, we are addressing the above issues by creating a customizable and secure Web 2.0 application for Personal Finance Management. This application can be used as a one stop shop for maintaining a user’s personal finances including income, investments and expenses. Users can tailor the application based on their financial requirements to effectively handle and manage multiple financial accounts. This application eliminates the need for multiple software applications and reduces the complexity involved in managing money.

**ACKNOWLEDGMENT**

I, **VENKATESH RAMAKRISHNAN [USN: 1RN16IS119]** student of **VIII** Semester BE, in Information Science and Engineering, RNS Institute of Technology hereby declare that the Internship work entitled **FINANCE APP** has been carried out by us and submitted in partial fulfillment of the requirements for the VIII Semester degree of Bachelor of Engineering in Information Science and Engineering of Visvesvaraya Technological University, Belagavi during academic year 2021-2022.

Place : Bengaluru

Date :

**VENKATESH RAMAKRISHNAN**

**(1RN16IS119)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | **TABLE OF CONTENTS** |  |
| **CERTIFICATE** | | | | |
| **[ABSTRACT](#_bookmark0)** | | |  | **[i](#_bookmark0)** |
| **[ACKNOWLEDGEMENT](#_bookmark1)** | | | | **[ii](#_bookmark1)** |
| **[TABLE OF CONTENTS](#_bookmark2)** | | | | **[iii](#_bookmark2)** |
| **[LIST OF FIGURES](#_bookmark3)** | | | | **vi** |
| **[LIST OF TABLES](#_bookmark4)** | | | | **vii** |
| **ABBREVIATIONS** | | | | **viii** |
| **[1](#_bookmark5)** | **[INTRODUCTION](#_bookmark5)** | | | **[1](#_bookmark5)** |
|  | [1.1](#_bookmark5) | [Background](#_bookmark5) | | [1](#_bookmark5) |
|  | 1.2 | About the mini project | | 1 |
| **[2](#_bookmark6)** | **LITERATURE REVIEW** | | | **3** |
| **[3](#_bookmark7)** | **[ANALYSIS AND DESIGN OF THE SYSTEM](#_bookmark7)** | | | **6** |
|  | [3.1](#_bookmark7) | [Analysis of the system](#_bookmark7) | | 6 |
|  | 3.2 | Design of the system | | 6 |
|  |  | 3.2.1 Architecture of the system | | 6 |
|  |  | 3.2.2 Use-Case Diagram | | 7 |

|  |  |  |  |
| --- | --- | --- | --- |
| **[4 IMPLEMENTATION](#_bookmark8)** | | | **9** |
| [4.1](#_bookmark8) | [Front-end and back-end used.](#_bookmark8) | | 9 |
|  | [4.1.1](#_bookmark8) | [Features of front-end - PHP.](#_bookmark8) | 9 |
|  | [4.1.2](#_bookmark8) | [Features of backend – MySQL (WAMP server).](#_bookmark8) | 10 |
| [4.2](#_bookmark9) | [Discussion](#_bookmark9) of code segments | | 11 |
|  | [4.2.1](#_bookmark9) | Database Connection | 11 |
|  | 4.2.2 | Admin Login | 11 |
|  | 4.2.3 | Health Status. | 12 |
| 4.3 | Applications of project work. | | 13 |
| **5 TESTING AND RESULTS** | | | **14** |
| 5.1 | Testing |  | 14 |
|  | 5.1.1 | Unit testing. | 14 |
|  |  | 5.1.1.1 Unit test case 1 | 14 |
|  | 5.1.2 | Integration testing | 15 |
|  |  | 5.1.2.1 Integration test case 1 | 15 |
|  |  | 5.1.2.2 Integration test case 2 | 15 |
|  | 5.1.3 | System testing | 16 |
|  |  | 5.1.3.1 System test case 1 | 16 |
| 5.2 | Discussion of results | | 17 |
|  | 5.2.1 Login page | | 17 |
|  | 5.2.2 Dashboard | | 17 |
|  | 5.2.3 Registration page | | 18 |
|  | 5.2.4 Payments page | | 18 |
|  | 5.2.5 Health status page | | 19 |
|  | 5.2.6 Workout routines page | | 19 |

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Description** | **Page No.** |
| Figure 3.1 | System Data Flow Diagram | 7 |
| Figure 3.2 | System Use Case Diagram | 8 |
| Figure 5.1 |  | 17 |
| Figure 5.2 | Dashboard | 17 |
| Figure 5.3 | Registration page | 18 |
| Figure 5.4 | Payments page | 18 |
| Figure 5.5 | Health status page | 19 |
| Figure 5.6 | Workout routines page | 19 |
|  |  |  |

**Chapter 1**

# INTRODUCTION

Financial management is an extremely pivotal life skill that could impact one’s life. Knowing how to manage your personal finance would have a positive impact on your life

## Background

The “finance app” is a service provided by a [bank](https://en.wikipedia.org/wiki/Bank" \t "Bank) or other [financial institution](https://en.wikipedia.org/wiki/Financial_institution" \t "Financial institution) that allows its customers to conduct [financial transactions](https://en.wikipedia.org/wiki/Financial_transaction" \t "Financial transaction) remotely using a [mobile device](https://en.wikipedia.org/wiki/Mobile_device" \t "Mobile device) such as a [smartphone](https://en.wikipedia.org/wiki/Smartphone" \t "Smartphone) or [tablet](https://en.wikipedia.org/wiki/Tablet_computer" \t "Tablet computer). Unlike the related [internet banking](https://en.wikipedia.org/wiki/Internet_banking" \t "Internet banking) it uses software, usually called an [app](https://en.wikipedia.org/wiki/Mobile_app" \t "Mobile app), provided by the financial institution for the purpose. Mobile banking is usually available on a 24-hour basis. Some financial institutions have restrictions on which accounts may be accessed through mobile banking, as well as a limit on the amount that can be transacted. Mobile banking is dependent on the availability of an internet or data connection to the mobile device.

## About the mini project

The Finance App is an interface which maintains the record of all the payments,. Administrator can log in to the system using the credentials which takes him to the main page. He can add a new member, update

Admin can view, edit/update the users or the inventory along with payment option which specifies joining date, expiry date, workout routine and health status. Trainers need not use any paperwork for maintaining the information about his members, which will be a tedious job and also there are chances of it getting misplaced. This mini project in turn helps him/her to maintain the records in a database which can be updated, managed and changed at any point of time.

New members can be added in the registration column which includes all their basic details along with joining details. Random function is being used to avoid redundancy of the primary key. Workout routines can be viewed as well as edited to the likes of the administrator

and new plans can also be added. Members per month, year and plan can be viewed by selecting the year and month from the drop-down box. All the data entered in the front end will be stored in the backend in their respective tables. The mini project organizes, manages and updates the entire database which eases the work of the administrator.

* This Finance app is developed to provide the following services:
* The Administrator can add new members to the gym by adding their credentials like name, address, phone number, joining date and exercise routine.
* Payment status of every member is maintained along with the exercise routine, joining date and expiry date of their plan.
* Admin profile can also be changed by providing the previous password and updating the credentials of the new one

**Chapter 2**

# LITERATURE REVIEW

Personal finance is the process of planning and managing personal financial activities such as [income](https://corporatefinanceinstitute.com/resources/knowledge/accounting/annual-income/) generation, spending, saving, [investing](https://corporatefinanceinstitute.com/resources/knowledge/trading-investing/investing-beginners-guide/), and protection. The process of managing one’s personal finances can be summarized in a [budget](https://corporatefinanceinstitute.com/resources/knowledge/accounting/types-of-budgets-budgeting-methods/) or financial plan. This guide will analyze the most common and important aspects of individual financial management.

#### #1 Income

Income refers to a source of cash inflow that an individual receives and then uses to support themselves and their family. It is the starting point for our financial planning process.

Common sources of income are:

* Salaries
* Bonuses
* Hourly wages
* Pensions
* [Dividends](https://corporatefinanceinstitute.com/resources/knowledge/finance/dividend/)

These sources of income all generate cash that an individual can use to either spend, save, or invest. In this sense, income can be thought of as the first step in our personal finance roadmap.

#### #2 Spending

Spending includes all types of expenses an individual incurs related to buying goods and services or anything that is consumable (i.e., not an investment). All spending falls into two categories: cash (paid for with cash on hand) and credit (paid for by borrowing money). The majority of most people’s income is allocated to spending.

Common sources of spending are:

* Rent
* [Mortgage payments](https://corporatefinanceinstitute.com/resources/knowledge/finance/mortgage/)
* Taxes
* Food
* Entertainment
* Travel
* [Credit card payments](https://corporatefinanceinstitute.com/resources/knowledge/other)

**Chapter 3**

# ANALYSIS AND DESIGN OF THE SYSTEM

### Analysis of the system

The Finance app includes:

* + - Wealth Management: This includes maintaining the information about all the members, their spending and earnings.
    - Flutter Technology used for the development of the application.

### Design of the system

**Systems design** is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. It is the application of systems theory to product development. There is some overlap with system analysis, system architecture and system engineering.

#### Architecture of the system

A **Data Flow Diagram** (**DFD**) is a graphical representation of the "flow" of data through an [information system](https://en.wikipedia.org/wiki/Information_system), modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFD’s can also be used for the [visualization](https://en.wikipedia.org/wiki/Data_visualization) of [data processing](https://en.wikipedia.org/wiki/Data_processing) (structured design).

A DFD shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored. It does not show information about process timing or whether processes will operate in sequence or in parallel, unlike a traditional structured [flowchart](https://en.wikipedia.org/wiki/Flowchart) which focuses on control flow, or a UML activity workflow diagram, which presents both control and data, flows as a unified model.

Figure 3.1 shows the data flow diagram for the Gym Management System. It illustrates the flow of data from and to the Gym Management in the system. The administrator can add new

member to the database, as well as make payment transactions, maintain health status of each and every member, workout routine management and finally updating or adding a new admin profile to the system. The system uses a MySQL database to store all the required information in the form of relational tables. In response to user input, the system uses the database to store the member information, their health status, payment transactions, workout routines and admin information.

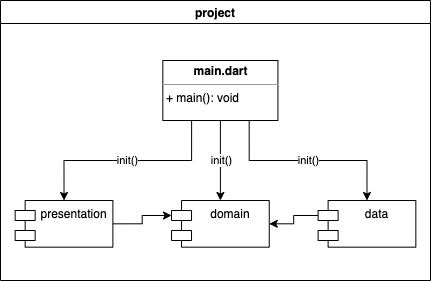


Figure 3.1 System Data Flow Diagram

#### Use-Case Diagram

A **Use-Case Diagram** (**UCD**) at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different [use-cases](https://en.wikipedia.org/wiki/Use_case) in which the user is involved. A use-case diagram can identify the different types of users of a system and the different use-cases and will often be accompanied by other types of diagrams as well.

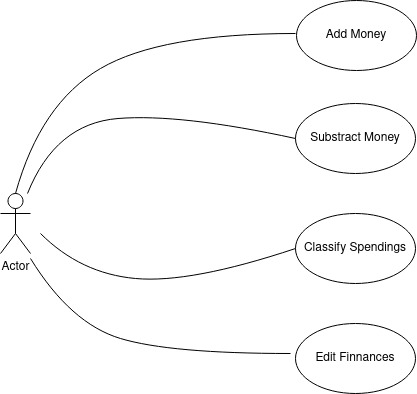


Figure 3.2 System Use-Case Diagram

Internship Project work Finance App

**Chapter 4**

# IMPLEMENTATION

Implementation is the process of defining how the system should be built, ensuring that it is operational and meets quality standards. It is a systematic and structured approach for effectively integrating a software-based service or component into the requirements of end users.

## Front-end and back-end used

The front-end is everything involved with what the user sees. The back-end, or the "server-side", is basically how the site works, updates and changes. This refers to everything the user can't see in the browser, like [databases](http://pluralsight.com/training/courses/TableOfContents?courseName=intro-sql-server&highlight=dan-sullivan_what-is-relational&what-is-relational) and [servers.](http://pluralsight.com/training/courses/TableOfContents?courseName=sql-server-fundamentals&highlight)

## Flutter

Flutter is a cross-platform tool intended for creating Android and iOS apps from a single code base by using a modern, reactive framework. Flutter apps are built using Dart, a simple object-oriented programming language. The central idea of Flutter revolves around widgets. The entire UI is made of combining different widgets, each of which defines a structural element (like a button or menu), a stylistic element (like a font or color scheme), an aspect of layout (like padding), and so on. Flutter does not use OEM widgets, but provides its own ready-made widgets which look native either to Android (Material Design) or iOS apps (Cupertino). It’s also possible to create custom widgets.

Flutter uses DART programming language which was introduced by Google in 2011 and is rarely used by developers.

Dart syntax is easy to understand for JavaScript or Java developers as it supports most of the object-oriented concepts. It’s easy to get started with Dart as there is great and easy-to-follow documentation available on the official Dart site

Flutter framework is bundled with UI rendering components, device API access, navigation, testing, stateful management and loads of libraries. This rich set of components removes the need to use third-party libraries. Flutter also has Widgets for Material Design and Cupertino that allow developers to easily render the UI on both iOS and Android platform.

Dept. of ISE, RNSIT 2021-2022 Page 9

Internship Project work Finance App

### 4.1.2 Xamarin

Xamarin has loads of modules and a great development API; however, it heavily depends on the

Visual Studio IDE. Developers from the non-Microsoft stack will struggle to learn all the concepts

of VS Code or a similar IDE. Also, learning C# will be require a long learning curve. And when

compared to editors, IDEs are heavy-weight and building and compiling things takes time.

Xamarin uses AOT compilation for iOS for the build and JIT/AOT for Android, so getting the UI

changes in the devices might take some time. Xamarin renders UI in terms of the native UI Controllers. There are a lot of resources online for Xamarin developers to solve the common issues.

**MySQL** is an open-source relational database management system (RDBMS). MySQL is a central component of the XAMPP open-source web application software stack (and other "AMP" stacks). MySQL is written in C and C++. MySQL works on many system platforms, including Linux, macOS, Microsoft Windows, etc.

MySQL performs extremely well in the average case, the developer interfaces are there, and the documentation is very, very good. It has also been tested to be a fast, stable and true multi-user, multi-threaded SQL database server.

## Discussion of code segments

This section includes the segments of code used to provide various user functionalities.

## Main Page

The below code shows how to establish connection with database

import 'package:finacash/screen/InicialPage.dart';

import 'package:flutter/material.dart';

import 'package:flutter/services.dart';

import 'package:intl/date\_symbol\_data\_local.dart';

main() {

  initializeDateFormatting().then((\_){

    runApp(*MaterialApp*(

    home: *InicialPage*(),

    debugShowCheckedModeBanner: false,

  ));

  });

}

## Home Page

The below code shows how admin logs in to the system by providing the credentials

import 'dart:ui';

import 'package:finacash/Helper/Movimentacoes\_helper.dart';

import 'package:finacash/Widgets/AnimatedBottomNavBar.dart';

import 'package:finacash/Widgets/CardMovimentacoesItem.dart';

import 'package:finacash/Widgets/CustomDialog.dart';

import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

import 'package:flutter/rendering.dart';

import 'package:intl/intl.dart';

import 'package:table\_calendar/table\_calendar.dart';

class *HomePage* extends *StatefulWidget* {

*@override*

*\_HomePageState* createState() => *\_HomePageState*();

}

class *\_HomePageState* extends *State*<*HomePage*> {

*String* saldoAtual = "";

*var* total;

*var* width;

*var* height;

*bool* recDesp = false;

  final *GlobalKey*<*ScaffoldState*> \_scafoldKey = *GlobalKey*<*ScaffoldState*>();

*MovimentacoesHelper* movHelper = *MovimentacoesHelper*();

*TextEditingController* \_valorController = *TextEditingController*();

*CalendarController* calendarController;

*MovimentacoesHelper* movimentacoesHelper = *MovimentacoesHelper*();

*List*<*Movimentacoes*> listmovimentacoes = *List*();

*List*<*Movimentacoes*> ultimaTarefaRemovida = *List*();

*var* dataAtual = new *DateTime*.now();

*var* formatter = new *DateFormat*('dd-MM-yyyy');

*var* formatterCalendar = new *DateFormat*('MM-yyyy');

*String* dataFormatada;

*String* format(*double* n) {

    return n.toStringAsFixed(n.truncateToDouble() == n ? 0 : 2);

  }

  \_addValor() {

*String* valor = \_valorController.text;

    setState(() {

      saldoAtual = valor;

    });

  }

  \_saldoTamanho(*String* conteudo) {

    if (conteudo.length > 8) {

      return width \* 0.08;

    } else {

      return width \* 0.1;

    }

  }

  \_salvar() {

    dataFormatada = formatter.format(dataAtual);

*Movimentacoes* mov = *Movimentacoes*();

    mov.valor = 20.50;

    mov.tipo = "r";

    mov.data = "10-03-2020"; //dataFormatada;

    mov.descricao = "CashBack";

*MovimentacoesHelper* movimentacoesHelper = *MovimentacoesHelper*();

    movimentacoesHelper.saveMovimentacao(mov);

    mov.toString();

  }

  \_allMov() {

    movimentacoesHelper.getAllMovimentacoes().then((list) {

      setState(() {

        listmovimentacoes = list;

      });

      print("All Mov: $*listmovimentacoes*");

    });

  }

  \_allMovMes(*String* data) {

    movimentacoesHelper.getAllMovimentacoesPorMes(data).then((list) {

      if (list.isNotEmpty) {

        setState(() {

          listmovimentacoes = list;

          //total =listmovimentacoes.map((item) => item.valor).reduce((a, b) => a + b);

        });

        total =

            listmovimentacoes.map((item) => item.valor).reduce((a, b) => a + b);

        saldoAtual = format(total).toString();

      } else {

        setState(() {

          listmovimentacoes.clear();

          total = 0;

          saldoAtual = total.toString();

        });

      }

      //print("TOTAL: $total");

      //print("All MovMES: $listmovimentacoes");

    });

  }

*@override*

*void* initState() {

    // TODO: implement initState

    super.initState();

    calendarController = *CalendarController*();

    if (*DateTime*.now().month != false) {

      //saldoAtual = "1259";

    }

    //\_salvar();

    dataFormatada = formatterCalendar.format(dataAtual);

    print(dataFormatada);

    \_allMovMes(dataFormatada);

    //\_allMov();

  }

  \_dialogAddRecDesp() {

    showDialog(

        context: context,

        builder: (context) {

          return *CustomDialog*();

        });

  }

*@override*

*Widget* build(*BuildContext* context) {

    width = *MediaQuery*.of(context).size.width;

    height = *MediaQuery*.of(context).size.height;

    \_allMovMes(dataFormatada);

    return *Scaffold*(

      key: \_scafoldKey,

      body: *SingleChildScrollView*(

        primary: false,

        physics: *NeverScrollableScrollPhysics*(),

        //physics: ClampingScrollPhysics(),

        //height: height,

        //width: width,

        child: *Column*(

          children: <*Widget*>[

*Stack*(

              children: <*Widget*>[

*Container*(

                  width: *double*.infinity,

                  height: height \* 0.334, //300,

                  color: *Colors*.black,

                ),

*Positioned*(

                  top: 0,

                  left: 0,

                  right: 0,

                  child: *Container*(

                      width: *double*.infinity,

                      height: height \* 0.28, //250,

                      decoration: *BoxDecoration*(

                        color: *Colors*.lightGreen, //Colors.indigo[400],

                      )),

                ),

*Positioned*(

                  top: width \* 0.18, //70

                  left: width \* 0.07, //30,

                  child: *Text*(

                    "FinanceApp",

                    style: *TextStyle*(

                        color: *Colors*.white, fontSize: width \* 0.074 //30

                        ),

                  ),

                ),

*Positioned*(

                  bottom: 0,

                  left: width \* 0.07, // 30,

                  right: width \* 0.07, // 30,

                  child: *Container*(

                    height: height \* 0.16, //150,

                    width: width \* 0.1, // 70,

                    decoration: *BoxDecoration*(

                        color: *Colors*.white,

                        borderRadius: *BorderRadius*.circular(20),

                        boxShadow: [

*BoxShadow*(

                              color: *Colors*.grey[400],

                              blurRadius: 5,

                              offset: *Offset*(0, 2))

                        ]),

                    child: *Column*(

                      mainAxisAlignment: *MainAxisAlignment*.center,

                      crossAxisAlignment: *CrossAxisAlignment*.start,

                      mainAxisSize: *MainAxisSize*.min,

                      children: <*Widget*>[

*Padding*(

                          padding: *EdgeInsets*.only(

                            left: width \* 0.05,

                            top: width \* 0.04,

                            bottom: width \* 0.02,

                          ),

                          child: *Text*(

                            "Total",

                            style: *TextStyle*(

                                color: *Colors*.grey[600], fontSize: width \* 0.1),

                          ),

                        ),

*Row*(

                          crossAxisAlignment: *CrossAxisAlignment*.center,

                          mainAxisAlignment: *MainAxisAlignment*.spaceBetween,

                          children: <*Widget*>[

*Padding*(

                              padding: *EdgeInsets*.only(left: width \* 0.05),

                              child: *Container*(

                                width: width \* 0.6,

                                child: *Text*(

                                  saldoAtual,

                                  overflow: *TextOverflow*.ellipsis,

                                  style: *TextStyle*(

                                    color: *Colors*

                                        .lightBlue[700], //Colors.indigo[400],

                                    fontWeight: *FontWeight*.bold,

                                    fontSize: \_saldoTamanho(saldoAtual),

                                    //width \* 0.1 //\_saldoTamanho(saldoAtual)

                                  ),

                                ),

                              ),

                            ),

*Padding*(

                              padding: *EdgeInsets*.only(right: width \* 0.04),

                              child: *GestureDetector*(

                                onTap: () {

                                  \_dialogAddRecDesp();

                                  /\* Navigator.push(

                                      context,

                                      MaterialPageRoute(

                                          builder: (context) => AddReceita()));

                                 \*/

                                },

                                child: *Container*(

                                  width: width \* 0.12,

                                  height: width \* 0.12, //65,

                                  decoration: *BoxDecoration*(

                                      color: *Colors*.red, //Colors.indigo[400],

                                      borderRadius: *BorderRadius*.circular(50),

                                      boxShadow: [

*BoxShadow*(

                                          color: *Colors*.grey,

                                          blurRadius: 7,

                                          offset: *Offset*(2, 2),

                                        )

                                      ]),

                                  child: *Icon*(

*Icons*.add,

                                    size: width \* 0.07,

                                    color: *Colors*.white,

                                  ),

                                ),

                              ),

                            )

                          ],

                        ),

*SizedBox*(

                          height: height \* 0.008,

                        )

                      ],

                    ),

                  ),

                )

              ],

            ),

*TableCalendar*(

              calendarController: calendarController,

              locale: "en\_US",

              headerStyle: *HeaderStyle*(

                formatButtonShowsNext: false,

                formatButtonVisible: false,

                centerHeaderTitle: true,

              ),

              calendarStyle: *CalendarStyle*(outsideDaysVisible: false),

              daysOfWeekStyle: *DaysOfWeekStyle*(

                weekdayStyle: *TextStyle*(color: *Colors*.transparent),

                weekendStyle: *TextStyle*(color: *Colors*.transparent),

              ),

              rowHeight: 0,

              initialCalendarFormat: *CalendarFormat*.month,

              onVisibleDaysChanged: (dateFirst, dateLast, *CalendarFormat* cf) {

                print(dateFirst);

                dataFormatada = formatterCalendar.format(dateFirst);

                \_allMovMes(dataFormatada);

                print("DATA FORMAT CALENDAR $*dataFormatada*");

                //print("Data Inicial: $dateFirst ....... Data Final: $dateLast");

              },

            ),

*Padding*(

                padding:

*EdgeInsets*.only(left: width \* 0.04, right: width \* 0.04),

                child: *Row*(

                  mainAxisAlignment: *MainAxisAlignment*.spaceBetween,

                  children: <*Widget*>[

*Text*(

                      "Transactions",

                      style: *TextStyle*(

                          color: *Colors*.grey[600], fontSize: width \* 0.04),

                    ),

*Padding*(

                      padding: *EdgeInsets*.only(right: width \* 0.02),

                      child: *Icon*(

*Icons*.sort,

                        size: width \* 0.07,

                        color: *Colors*.grey[400],

                      ),

                    )

                  ],

                )),

*Padding*(

              padding: *EdgeInsets*.only(

                  left: width \* 0.04, right: width \* 0.04, top: 0),

              child: *SizedBox*(

                width: width,

                height: height \* 0.47,

                child: *ListView*.builder(

                  itemCount: listmovimentacoes.length,

                  itemBuilder: (context, index) {

*Movimentacoes* mov = listmovimentacoes[index];

*Movimentacoes* ultMov = listmovimentacoes[index];

                    return *Dismissible*(

                      direction: *DismissDirection*.endToStart,

                      onDismissed: (direction) {

                        //\_dialogConfimacao(context, width, mov,index);

                        setState(() {

                          listmovimentacoes.removeAt(index);

                        });

                        movHelper.deleteMovimentacao(mov);

                        final snackBar = *SnackBar*(

                          content: *Container*(

                            padding: *EdgeInsets*.only(bottom: width \* 0.025),

                            alignment: *Alignment*.bottomLeft,

                            height: height \* 0.05,

                            child: *Text*(

                              "Undo Action",

                              style: *TextStyle*(

                                  color: *Colors*.black,

                                  fontWeight: *FontWeight*.bold,

                                  fontSize: width \* 0.05),

                            ),

                          ),

                          duration: *Duration*(seconds: 2),

                          backgroundColor: *Colors*.orange[800],

                          shape: *RoundedRectangleBorder*(

                              borderRadius: *BorderRadius*.only(

                                  topLeft: *Radius*.circular(15),

                                  topRight: *Radius*.circular(15))),

                          action: *SnackBarAction*(

                            label: "Withdrawal",

                            textColor: *Colors*.white,

                            onPressed: () {

                              setState(() {

                                listmovimentacoes.insert(index, ultMov);

                              });

                              movHelper.saveMovimentacao(ultMov);

                            },

                          ),

                        );

                        \_scafoldKey.currentState.showSnackBar(snackBar);

                      },

                      key: *ValueKey*(mov.id),

                      background: *Container*(

                        padding: *EdgeInsets*.only(right: 10, top: width \* 0.04),

                        alignment: *Alignment*.topRight,

                        color: *Colors*.yellow,

                        child: *Icon*(

*Icons*.delete\_outline,

                          color: *Colors*.white,

                          size: width \* 0.07,

                        ),

                      ),

                      child: *CardMovimentacoesItem*(

                        mov: mov,

                        lastItem:

                            listmovimentacoes[index] == listmovimentacoes.last

                                ? true

                                : false,

                      ),

                    );

                  },

                ),

              ),

            ),

*Padding*(

              padding: *EdgeInsets*.only(top: 20),

              child: *Text*("EEEEEEEEE"),

            )

          ],

        ),

      ),

    );

  }

}

## Navigation Bar

import 'package:flutter/material.dart';

class *AnimatedBottomBar* extends *StatefulWidget* {

  final *List*<*BarItem*> barItems;

  final *Duration* animationDuration;

  final *Function* onBarTap;

  final *BarStyle* barStyle;

*AnimatedBottomBar*(

      {this.barItems,

      this.animationDuration = const *Duration*(milliseconds: 500),

      this.onBarTap, this.barStyle});

*@override*

*\_AnimatedBottomBarState* createState() => *\_AnimatedBottomBarState*();

}

class *\_AnimatedBottomBarState* extends *State*<*AnimatedBottomBar*>

    with *TickerProviderStateMixin* {

*int* selectedBarIndex = 1;

*@override*

*Widget* build(*BuildContext* context) {

*double* width = *MediaQuery*.of(context).size.width;

*double* height = *MediaQuery*.of(context).size.height;

    return *Material*(

      elevation: 10,

      child: *Padding*(

        padding:  *EdgeInsets*.only(

          bottom: width \* 0.07, //32.0,

          top: width \* 0.04,//16.0,

          left:width \* 0.04,// 16.0,

          right:width \* 0.04,// 16.0,

        ),

        child: *Row*(

          mainAxisSize: *MainAxisSize*.max,

          mainAxisAlignment: *MainAxisAlignment*.center,

          children: \_buildBarItems(context, width),

        ),

      ),

    );

  }

*List*<*Widget*> \_buildBarItems(*BuildContext* contex,*double* largura) {

*List*<*Widget*> \_barItems = *List*();

    for (*int* i = 0; i < widget.barItems.length; i++) {

*BarItem* item = widget.barItems[i];

*bool* isSelected = selectedBarIndex == i;

      \_barItems.add(*InkWell*(

        splashColor: *Colors*.transparent,

        onTap: () {

          setState(() {

            selectedBarIndex = i;

            widget.onBarTap(selectedBarIndex);

          });

        },

        child: *AnimatedContainer*(

          padding:  *EdgeInsets*.symmetric(horizontal: largura \* 0.03 , vertical: largura \* 0.008),

          duration: widget.animationDuration,

          decoration: *BoxDecoration*(

              color: isSelected

                  ? item.color.withOpacity(0.15)

                  : *Colors*.transparent,

              borderRadius: *BorderRadius*.all(*Radius*.circular(30))),

          child: *Row*(

            mainAxisAlignment: *MainAxisAlignment*.spaceEvenly,

            children: <*Widget*>[

*Icon*(

                item.iconData,

                color: isSelected ? item.color : *Colors*.black,

                size: widget.barStyle.iconSize,

              ),

*SizedBox*(

                width: largura \* 0.01,

              ),

*AnimatedSize*(

                duration: widget.animationDuration,

                curve: *Curves*.easeInOut,

                vsync: this,

                child: *Text*(

                  isSelected ? item.text : "",

                  style: *TextStyle*(

                      color: item.color,

                      fontWeight: widget.barStyle.fontWeight,

                      fontSize: widget.barStyle.fontSize),

                ),

              )

            ],

          ),

        ),

      ));

    }

    return \_barItems;

  }

}

class *BarStyle* {

  final *double* fontSize, iconSize;

  final *FontWeight* fontWeight;

*BarStyle*({this.fontSize = 16.0, this.iconSize = 32, this.fontWeight = *FontWeight*.w600});

}

class *BarItem* {

*String* text;

*IconData* iconData;

*Color* color;

*BarItem*({this.text, this.iconData, this.color});

}

**4.2.4 Withdrawal**

import 'package:finacash/Helper/Movimentacoes\_helper.dart';

import 'package:finacash/Widgets/TimeLineItem.dart';

import 'package:flutter/material.dart';

class *DespesasResumo* extends *StatefulWidget* {

*@override*

*\_DespesasResumoState* createState() => *\_DespesasResumoState*();

}

class *\_DespesasResumoState* extends *State*<*DespesasResumo*> {

*MovimentacoesHelper* movimentacoesHelper = *MovimentacoesHelper*();

*List*<*Movimentacoes*> listmovimentacoes = *List*();

  \_allMovPorTipo() {

    movimentacoesHelper.getAllMovimentacoesPorTipo("d").then((list) {

      setState(() {

        listmovimentacoes = list;

      });

      print("All Mov: $*listmovimentacoes*");

    });

  }

*@override*

*void* initState() {

    // TODO: implement initState

    super.initState();

    \_allMovPorTipo();

  }

*@override*

*Widget* build(*BuildContext* context) {

*double* width = *MediaQuery*.of(context).size.width;

*double* height = *MediaQuery*.of(context).size.height;

    return *Scaffold*(

      backgroundColor: *Colors*.redAccent.withOpacity(0.8),

      body: *SingleChildScrollView*(

        physics: *ClampingScrollPhysics*(),

        child: *Column*(

          crossAxisAlignment: *CrossAxisAlignment*.start,

          children: <*Widget*>[

*Padding*(

              padding: *EdgeInsets*.only(left: width \* 0.05, top: width \* 0.2),

              child: *Text*(

                "Withdrawal Amount",

                style: *TextStyle*(

                    color: *Colors*.white, //Colors.grey[400],

                    fontWeight: *FontWeight*.bold,

                    fontSize: width \* 0.08),

              ),

            ),

*Padding*(

              padding: *EdgeInsets*.only(left: width \* 0.03, top: width \* 0.08),

              child: *SizedBox*(

                width: width,

                height: height \* 0.74,

                child: *ListView*.builder(

                  itemCount: listmovimentacoes.length,

                  itemBuilder: (context, index) {

*List* movReverse = listmovimentacoes.reversed.toList();

*Movimentacoes* mov = movReverse[index];

                    if (movReverse[index] == movReverse.last) {

                      return *TimeLineItem*(

                        mov: mov,

                        colorItem: *Colors*.red[900],

                        isLast: true,

                      );

                    } else {

                      return *TimeLineItem*(

                        mov: mov,

                        colorItem: *Colors*.red[900],

                        isLast: false,

                      );

                    }

                  },

                ),

              ),

            ),

          ],

        ),

      ),

    );

  }

}

## Applications of project work

* Connected platform
* Building a community
* End to end conversation
* Temporary group creation
* Secured conversations
* Sharing of confidential information

**Chapter 5**

# RESULTS

## Home page

Figure 5.1 shows the Homepage of the Finance app displayed before user’s login.

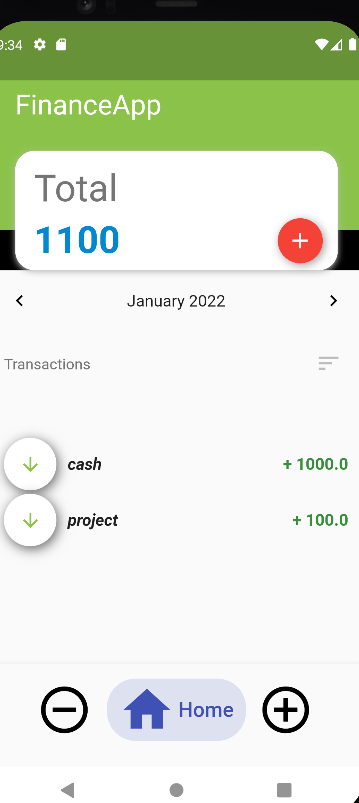


Figure 5.1 Login page

## Deposit

Figure 5.2 shows the Deposit page which is displayed after the user logs in to the system

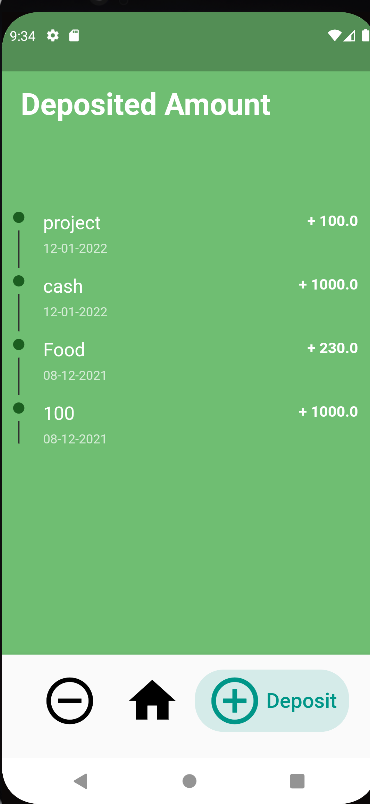
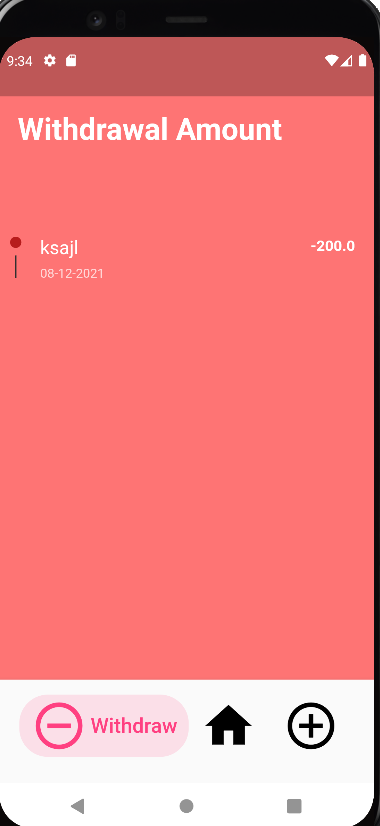


Figure 5.2 Deposit

## Withdrawal page

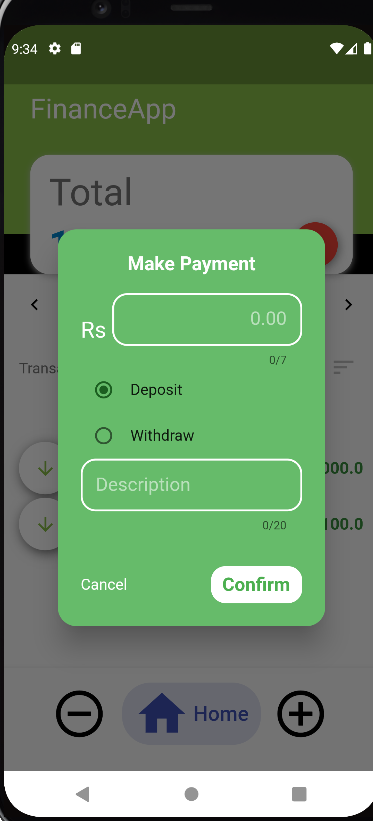
Figure 5.3 shows the withdrawal page where the admin enters all the details if the user.



## Payments page

Figure 5.3 Registration page

Figure 5.4 shows the Payments page which consists of all the transactions on the members



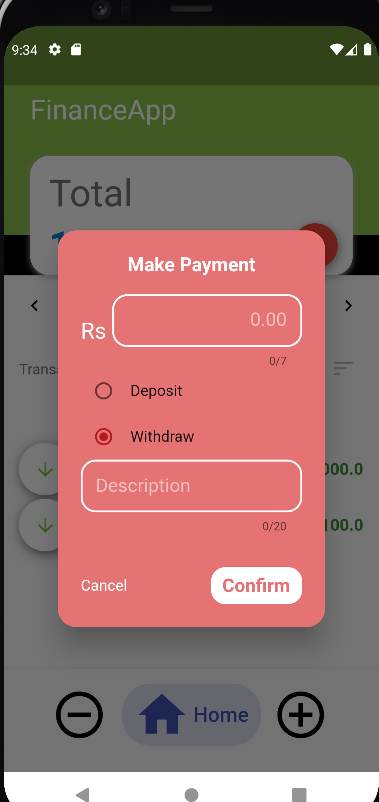


Figure 5.4 Payments page

## Edit page

Figure 5.5 shows the Edit of the members which is maintained by the administrator

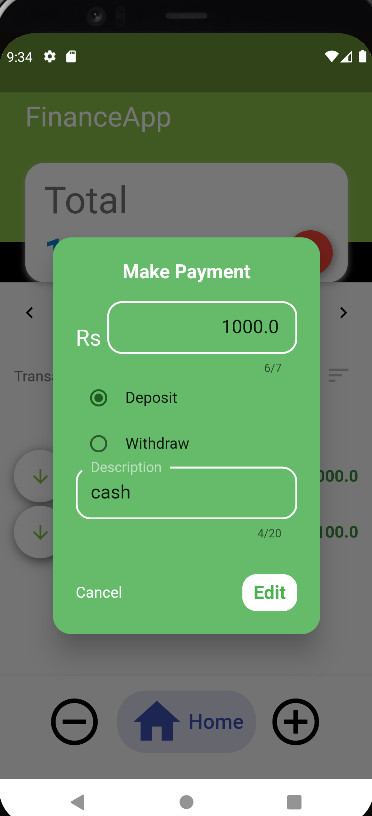


Figure 5.5 Edit page

## Chapter 6

**CONCLUSION AND FUTURE ENHANCEMENTS**

The architecture of mobile banking apps is usually prone to some serious mobile banking vulnerabilities that may lead to financial security breaches. Basically, a mobile-based online banking app is a type of software that is directly connected to the bank’s back-end system via Application Programming Interfaces (APIs).

The future enhancements that can be made to this mini project are:

* End to end Encryption.
* Add a multi-authentication feature
* Offer real time alerts
* Utilize behavioral analysis

# REFERENCES

* https://flutter.dev/
* https://www.youtube.com/channel/UCW5YeuERMmlnqo4oq8vwUpg