

PERSONAL EXPENSE TRACKER

A PROJECT REPORT

Submitted By

**DHIWAAGAR P
DHINAKARAN R
HARIPRASANTH P
GOWRISANKAR D**

*in partial fulfilment of the requirements for
the award of the degree
of*

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**



KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI, ERODE-638 060

TABLE OF CONTENT

CHAPTER NO.	TITLE
1	INTRODUCTION
1.1	Project Overview
1.2	Purpose
2	LITERATURE SURVEY
2.1	Existing System
2.2	References
2.3	Problem Statement Definition
3	IDEATION & PROPOSED SYSTEM
3.1	Empathy Map Canvas
3.2	Ideation & Brainstorming
3.3	Proposed Solution
3.4	Problem Solution fit
4	PROJECT DESIGN
4.1	Data Flow Diagrams
4.2	Solution & Technical Architecture
5	PROJECT PLANNING & SCHEDULING
5.1	Sprint Planning & Estimation
5.2	Sprint Delivery Schedule
6	CODING AND SOLUTIONING
7	RESULTS

8	ADVANTAGES & DISADVANTAGES
9	CONCLUSION
10	FUTURE SCOPE

APPENDIX

Source Code

GitHub & Project Demo Link

CHAPTER 1

INTRODUCTION

1.1 Project Overview

The popularity and usability of mobile applications have surpassed that of online applications in terms of user convenience. There are several mobile applications that offer ways to manage individual and group spending, but few of them give a thorough overview of both situations. In this paper, we develop a mobile application for the Android platform that records a user's individual expenses, his or her share of group expenses, top investment options, a view of the stock market at the time, reads authenticated financial news, and allows users to take advantage of the best offers currently available in the market in popular categories. The suggested application will provide the best picture of your costs while getting rid of messy sticky notes, spreadsheet confusion, and inconsistent data handling issues. Our app allows us to control their expenses and decide on their budget more effectively.

1.2 Purpose

It is also known as expense manager and money manager, an expense tracker is a software or application that helps to keep an accurate record of your money inflow and outflow. Many people in India live on a fixed income, and they find that towards the end of the month they don't have sufficient money to meet their needs.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing problem

The issue facing the current generation is that they can't recall where all of the money they earned has gone, forcing them to make do with the meagre amounts of money they have left over to meet their basic necessities. There is currently no perfect solution that enables a person to quickly and effectively manage their daily expenses and alert them to their financial situation. To do this, they must keep extensive ledgers or computer logs for the data, and the user must perform the calculation manually, which could result in errors and losses.

2.2 Reference

1. <https://nevonprojects.com/daily-expense-tracker-system/>
2. <https://data-flair.training/blogs/expense-tracker-python/>
3. <https://phpgurukul.com/daily-expense-tracker-using-php-and-mysql/>
4. <https://ijarsct.co.in/Paper391.pdf>

5. https://kandi.openweaver.com/?landingpage=python_all_projects&utm_source=google&utm_medium=cpc&utm_campaign=promo_kandi_ie&utm_content=kandi_ie_search&utm_term=python_devs&gclid=Cj0KCQiAgribBhDkARIsAASA5bukrZgbI9UZxzpoyf0P-ofB1mZNxzc-okUP-3TchpYMclHTYFYiqP8aAmmwEALw_wcB

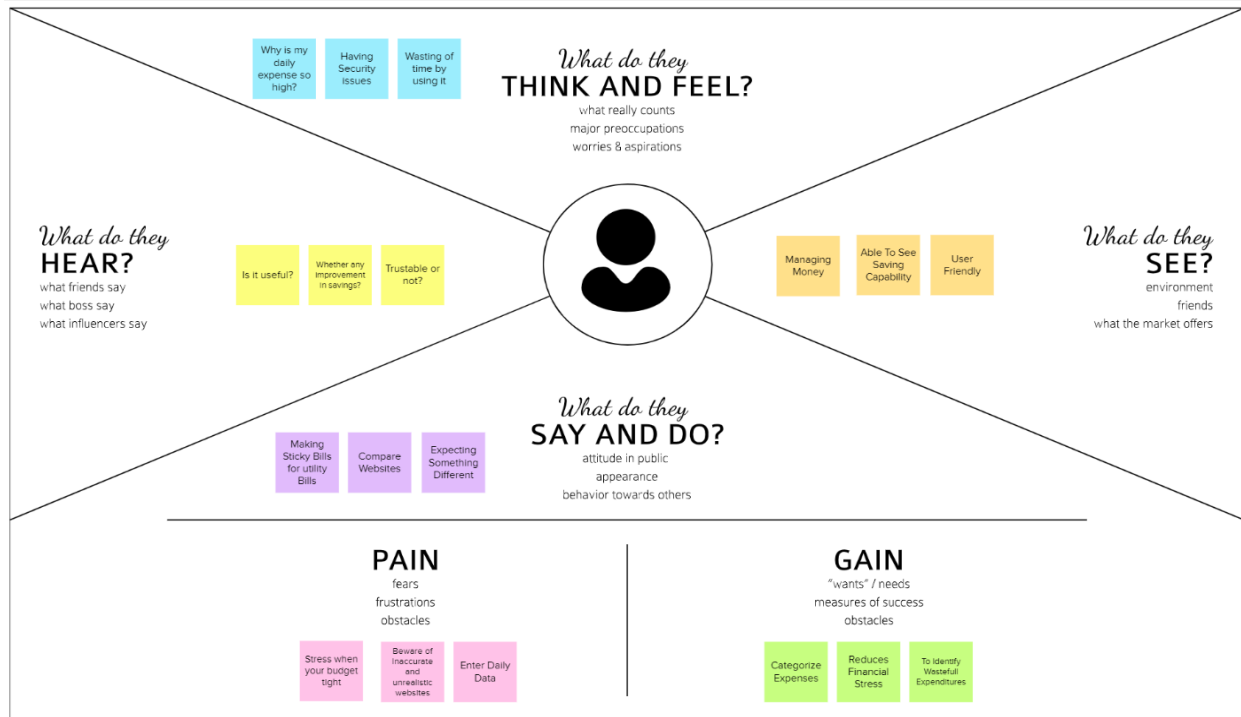
2.3 Problem Statement Definition

A web application called "Expense Tracker" enables users to manage and keep track of both personal and professional expenses. The users of this application can keep digital diaries. It will record a user's earnings and outgoings each day. With the aid of the internet, the user will be able to instantly add his or her expenses and can review them whenever and wherever. Without putting his or her information at danger and effectively protecting his or her privacy, he or she can quickly import transactions from his or her mobile wallets. Files are frequently lost or mistakenly deleted. This spending tracker offers a comprehensive digital fix for this issue. Excel spreadsheets are not very helpful for tracking . Additionally, they lack the sophisticated capability of automatically creating graphical graphics. Not only will it save people time, but it will also guarantee accurate calculations. The user only needs to enter their income and expenses; the system will handle the rest.

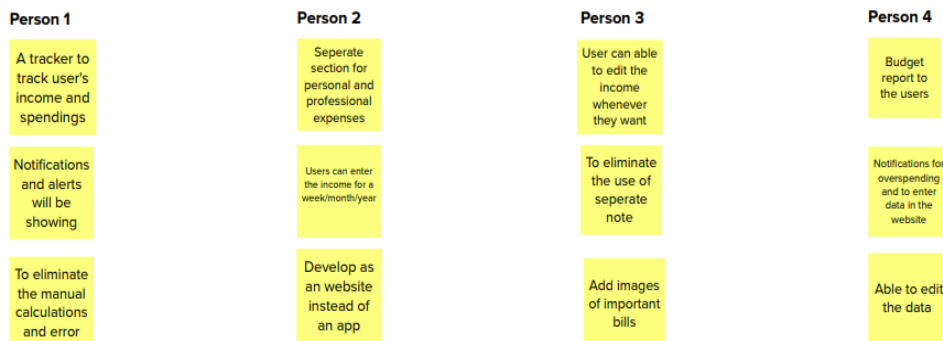
CHAPTER 3

IDEATION & PROPOSED SOLUTION

3.1 Empathy Map canvas



3.2 Ideation & Brainstorming



3.3 Proposed Solution

All people in the earning sector need a way to manage their financial resources and track their expenditure, so that they can improve and monitor their spending habits. This makes them understand the importance of financial management and making better decisions in the future .

They have an option to set a limit for the amount to be used for that particular month if the limit is exceeded the user will be notified with an email alert. The solution to this problem is, the people who get regular payments are able to track their payments and avoid unwanted expenses. If the limit is exceeded the user will be notified with an email alert.

3.4 Proposed Solution Fit

The solution to this problem is, the people who gets regular payments can able to track their payments and avoid unwanted expenses. If the limit is exceeded the user will be notified with an email alert.

- Novelty / Uniqueness Notification can be received through email.
- Social Impact / Customer Satisfaction Using this application one can track their personal expenses and frame a monthly/annual budget. If your expense exceeds the specified limit, the application will show you an alert message .This will make an impact on Mobile Banking for Customers' Satisfaction.
- Business Model (Revenue Model) Business people can use the subscription/premium feature of this application to gain revenue.
- Scalability of the Solution The scalability of the application depends on security, the working of the application even during when the network gets down etc...

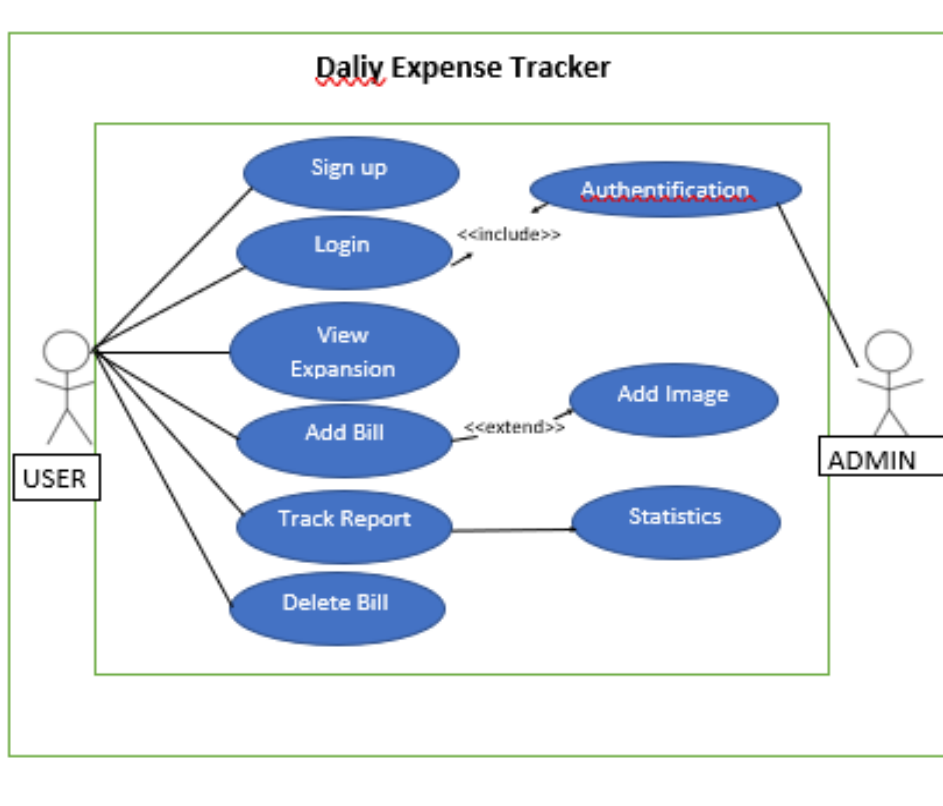
CHAPTER 4

PROJECT DESIGN

4.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

4.2 Solution & Architecture



CHAPTER 5

PROJECT PLANNING & SCHEDULING

5.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Priority
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	High
Sprint-1	Registration	USN-2	As a user, I will receive confirmation email once I have registered for the application	High
Sprint-1	Registration	USN-3	As a user, I can register for the application through Gmail	Medium
Sprint-1	Login	USN-4	As a user, I can log into the application by entering email & password	High
Sprint-1	Dashboard	USN-5	As a user, I can see the stock in hand and how much stock will be received and check other details.	High
Sprint-2	Customer details	USN-6	As a user, I can see the customer details like name, company, location, and so on.	Low
Sprint-2	Invoice management	USN-7	As a user, I can see, manage, and update or modify the invoice of my shop	Low
Sprint-2	Sale and order management	USN-8	As a user, I can see, manage, and update the sale and order	Medium
Sprint-2	Return management	USN-9	As a user, I can manage the returned items and check for damaged or defective items.	Medium
Sprint-2	Purchase order management	USN-10	As a user, I can enter the newly purchased stock and add or remove the stocks. and upload the purchased details as well.	Medium
Sprint-3	Stocks	USN-11	As a user, I can see the stock level, fast-moving, and death stocks.	High
Sprint-3	Report	USN-12	As a user, I can see the report of the stock	Low

Sprint-3	Notification	USN-13	As a user, it is good if I get a notification for low stock.	Medium
Sprint-3	Supplier	USN-14	As a user, I can see the supplier details for a better understanding.	Low
Sprint-2	Profile	USN-15	As a user, I can see my profile and give my details after registering as well.	Low
Sprint-3	bill	USN-16	As a user, I like to print the product the sold now and maintain it.	Medium
Sprint-3	Chatbot	USN-17	As a customer care executive,I can view the complaints on chat box, As a customer, I should be able solve and reply for the customers queries and as a customer, I can close the complaint after assisting	High
Sprint-4	Containerization	USN-18	As a user, I can access the software with high performance	High
Sprint-4	Deployment	USN-19	As a user, I can access the software in the web	High

5.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	31 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

CHAPTER 6

Coding And Solutioning

Codes:

```
# -*- coding: utf-8 -*-  
"""
```

Created on Mon Sep 26 14:27:20 2022

```
@author: user  
"""
```

```
from flask import Flask,render_template, request,redirect,url_for,session  
import ibm_db  
import re
```

```
app=Flask(__name__)  
app.secret_key= 'a'
```

```
conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=125f9f61-9715-  
46f9-9399-  
c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30  
426;Security=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=z  
bf09733;PWD=dLNuxrT4JZsamhxF",",")
```

```
@app.route('/')
```

```
def homer():  
    return render_template('home.html')
```

```
@app.route('/contact')
```

```

def contact():
    return render_template('contact.html')

@app.route('/login',methods =['GET', 'POST'])
def login():
    global userid
    msg = ""

    if request.method == 'POST' :
        username = request.form['username']
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =? AND
password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)
        if account:
            session['loggedin'] = True
            session['id'] = account['USERNAME']
            userid= account['USERNAME']
            session['username'] = account['USERNAME']
            msg = 'Logged in successfully !'

            msg = 'Logged in successfully !'
            return render_template('base.html', msg = msg)
        else:
            msg = 'Incorrect username / password !'
    return render_template('login.html', msg = msg)

```

```

@app.route('/register', methods=['GET', 'POST'])
def register():
    msg = "
    if request.method == 'POST' :
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = 'Account already exists !'
        elif not re.match(r'^[@]+@[^@]+\.[^@]+', email):
            msg = 'Invalid email address !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'name must contain only characters and numbers !'
        else:
            insert_sql = "INSERT INTO users VALUES (?, ?, ?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prepare_stmt, 1, username)
            ibm_db.bind_param(prepare_stmt, 2, email)
            ibm_db.bind_param(prepare_stmt, 3, password)
            ibm_db.execute(prepare_stmt)
            msg = 'You have successfully registered !'
    elif request.method == 'POST':
        msg = 'Please fill out the form !'
    return render_template('register.html', msg = msg)

```

```
@app.route('/Homepage', methods=['GET', 'POST'])
def dash():
```

```
    return render_template('Homepage.html')
```

```
@app.route('/addexpense', methods=['GET', 'POST'])
def addexpense():
```

```
    msg = "
```

```
    if request.method == 'POST' :
```

```
        username = request.form['username']
```

```
        date = request.form['date']
```

```
        expensename = request.form['expensename']
```

```
        amount = request.form['amount']
```

```
        paymode = request.form['paymode']
```

```
        category = request.form['category']
```

```
        sql = "SELECT * FROM users WHERE username =?"
```

```
        stmt = ibm_db.prepare(conn, sql)
```

```
        ibm_db.bind_param(stmt,1,username)
```

```
        ibm_db.execute(stmt)
```

```
        account = ibm_db.fetch_assoc(stmt)
```

```
        print(account)
```

```
        if account:
```

```
            insert_sql = "INSERT INTO add VALUES (?, ?, ?, ?, ?, ?)"
```

```
            prep_stmt = ibm_db.prepare(conn, insert_sql)
```

```
            ibm_db.bind_param(prepare_stmt, 1, username)
```

```
            ibm_db.bind_param(prepare_stmt, 2, date)
```

```
            ibm_db.bind_param(prepare_stmt, 3, expensename)
```

```
            ibm_db.bind_param(prepare_stmt, 4, amount)
```

```
            ibm_db.bind_param(prepare_stmt, 5, paymode)
```

```
    ibm_db.bind_param(prepare_stmt, 6, category)
    ibm_db.execute(prepare_stmt)
    msg = 'You have successfully added'
else:
    msg = 'Enter correct username !'
```

```
elif request.method == 'POST':
    msg = 'Please fill out the form !'
    return render_template('add.html', msg = msg)
```

```
@app.route('/delete/<string:id>', methods = ['POST', 'GET' ])
def delete(id):
    print(id)
    sql = "DELETE FROM expenses WHERE id =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,id)
    ibm_db.execute(stmt)

    return redirect("/display")
```

```
@app.route('/edit/<id>', methods = ['POST', 'GET' ])
def edit(id):
```

```
    sql = "SELECT * FROM expenses WHERE id =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,id)
    ibm_db.execute(stmt)
    row=ibm_db.fetch_tuple(stmt)
```



```
print(row)
return render_template('edit.html', expenses = row)
```

```
@app.route("/limitnum" , methods = ['POST' ])
def limitnum():
    if request.method == "POST":
        number= request.form['number']

        insert_sql = "INSERT INTO limits VALUES (?, ?)"
        prep_stmt = ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prepare_stmt, 1, session['id'])
        ibm_db.bind_param(prepare_stmt, 2, number)
        ibm_db.execute(prepare_stmt)
        msg = 'You have successfully added'

    elif request.method == 'POST':
        msg = 'Please fill out the form !'
    return render_template('limitm.html', msg = msg)
```

```
@app.route("/display")
def display():
    print(session["username"],session['id'])
```

```
sql = "SELECT * FROM add WHERE USERID=?"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,session['id'])
ibm_db.execute(stmt)
list1=[]
row = ibm_db.fetch_tuple(stmt)
while(row):
    list1.append(row)
    row = ibm_db.fetch_tuple(stmt)
print(list1)
```

```
total=0
t_food=0
t_entertainment=0
t_business=0
t_rent=0
t_EMI=0
t_other=0
```

```
for x in list1:
    total += x[4]
    if x[6] == "food":
        t_food += x[4]
    elif x[6] == "entertainment":
        t_entertainment += x[4]
    elif x[6] == "business":
        t_business += x[4]
    elif x[6] == "rent":
        t_rent += x[4]
    elif x[6] == "EMI":
        t_EMI += x[4]
    elif x[6] == "other":
```

```
t_other += x[4]
```

```
return render_template('display.html' ,expense = list1,total = total ,  
    t_food = t_food,t_entertainment = t_entertainment,  
    t_business = t_business, t_rent = t_rent,  
    t_EMI = t_EMI, t_other = t_other)
```

```
@app.route('/logout')
```

```
def logout():  
    session.pop('loggedin', None)  
    session.pop('id', None)  
    session.pop('username', None)  
    return render_template('thanks.html')
```

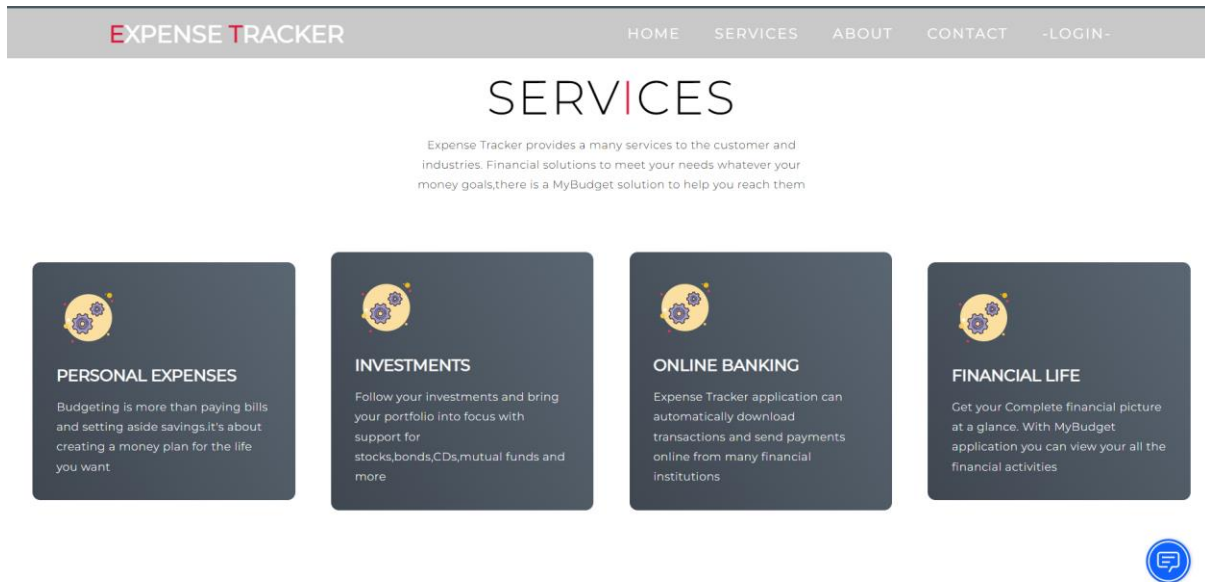
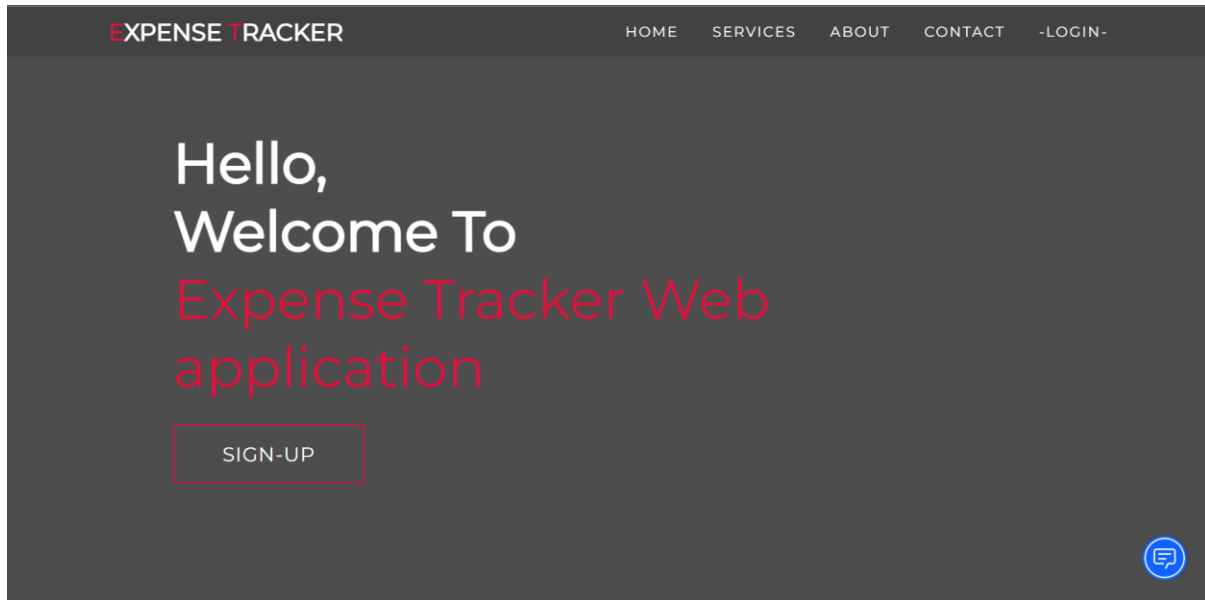
```
if __name__ == '__main__':  
    app.run(host='0.0.0.0')
```

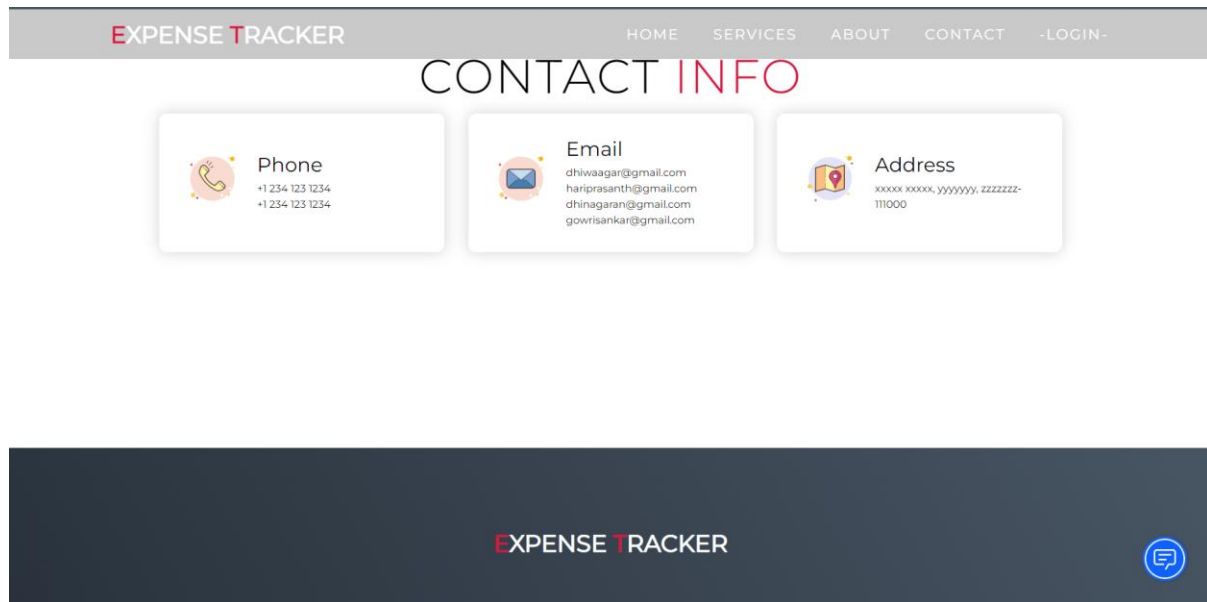
\

CHAPTER 7

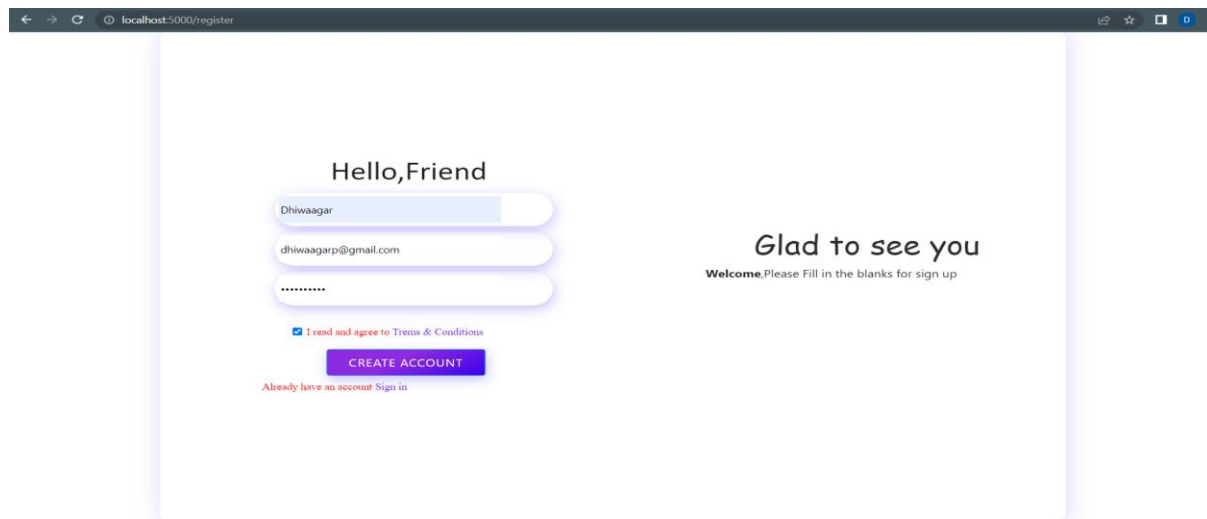
RESULTS

HOME PAGE





SIGNUP PAGE



LOGIN PAGE

Welcome

SIGN IN WITH

USERNAME

PASSWORD

[Forgot Password?](#)

LOGIN

Don't have an account?

[REGISTER here](#)

DASHBOARD

[MyBudget](#) [Home](#) [Add](#) [History](#) [LIMIT](#)

User

[Profile](#)
[Log-Out](#)



ADD EXPENSE

← → ↻ localhost:5000/addexpense

Add Expense

Username
Dhiwaagar

Date
17-11-2022 10:12 PM


Expense name
House Rent

Expense Amount
3000

Paymode
onlinebanking

Category
Rent

Submit



The graphic is a 'MONTHLY EXPENSES template' featuring a man in a suit pointing upwards. It is divided into 'INCOME' and 'EXPENSES' sections. The 'INCOME' section includes 'SALARY' and 'EXTRA INCOME'. The 'EXPENSES' section lists various categories with checkmarks: 'ELECTRIC', 'WATER', 'FOOD', 'CREDIT CARD', 'HOUSE', 'SAVINGS' (with a downward arrow), 'AUTO MAINTENANCE', 'OIL', 'PHONE', 'INTERNET', 'MEDICAL', and 'TRANSPORTATION'. Icons for each category are provided, such as a lightbulb for electric, a faucet for water, a hamburger for food, a credit card, a house, a piggy bank for savings, a car for auto maintenance, a drop of oil, a smartphone for phone, a computer monitor for internet, a medical syringe, and a train for transportation.

LIMIT PAGE

Currently your MONTHLY limit is ₹

ENTER the MONTHLY LIMIT to avoid over EXPENSES

10000 ENTER



CONTACT US PAGE

CONTACT INFO



Phone

+1 234 123 1234
+1 234 123 1234



Email

dhiwaagar@gmail.com
hariprasanth@gmail.com
dthinakaran@gmail.com
gowrisankar@gmail.com



Address

xxxxx xxxxx, yyyyyyy, zzzzzzz-
111000



LOGOUT PAGE

Thank you!

for visiting
our website

♥EXPENSE TRACKER

Manage your expense efficiently

[Return to Website](#)

CHAPTER 8

ADVANTAGES AND DISADVANTAGES

8.1. ADVANTAGES:

One of the major pros of tracking spending is always being aware of the state of one's personal finances. Tracking what you spend can help you stick to your budget, not just in a general way, but in each category such as housing, food, transportation and gifts. While a con is that manually tracking all cash that is spent can be irritating as well as time consuming, a pro is that doing this automatically can be quick and simple.

Another pro is that many automatic spending tracking software programs are available for free. Having the program on a hand-held device can be a main pro since it can be checked before spending occurs in order to be sure of the available budget. Another pro is that for those who just wish to keep tracking spending by hand with a paper and pen or by entering data onto a computer spreadsheet, these options are also available. Some people like to keep a file folder or box to store receipts and record the cash spent each day. A pro of this simple daily tracking system is that it can make one more aware of where the money is going before the end of a pay period or month.

8.2. DISADVANTAGES:

A con with any system used to track spending is that one may start doing it then taper off until it's forgotten about all together. Yet, this is a risk for any new goal such as trying to lose weight or quit smoking. If a person first makes a budget plan, then places money in savings before spending any new pay period or month, the tracking goal can help. In this way, tracking, spending and making sure all receipts are accounted for only needs to be done once or twice a month. Even with constant tracking of one's spending habits, there is no guarantee that financial goals will be met. Although this can be considered to be a con of tracking spending, it could be changed into a pro if one makes

up his or her mind to keep trying to properly manage all finances. Another con that may occur when spending is being tracked is an error, but this may also be able to be changed into a pro if the person does regular tracking. Frequent tracking of cash spending can allow one to catch and correct errors so that the budget plan is still able to be adhered to despite the mistake.

CHAPTER 9

CONCLUSION

A comprehensive money management strategy requires clarity and conviction for decision making. You will need a defined goal and a clear vision for grasping the business and personal finances. That's when an expense tracking app comes into the picture. An expense tracking app is an exclusive suite of services for people who seek to handle their earnings and plan their expenses and savings efficiently. It helps you track all transactions like bills, refunds, payrolls, receipts, taxes, etc., on a daily, weekly, and monthly basis.

CHAPTER 10

FUTURE SCOPE

- Achieve your business goals with a tailored mobile app that perfectly fits your business.
- Scale-up at the pace your business is growing.
- Deliver an outstanding customer experience through additional control over the app.
- Control the security of your business and customer data.
- Open direct marketing channels with no extra costs with methods such as push notifications.
- Boost the productivity of all the processes within the organization.
- Increase efficiency and customer satisfaction with an app aligned to their needs.
- Seamlessly integrate with existing infrastructure.
- Ability to provide valuable insights.
- Optimize sales processes to generate more revenue through enhanced data collection.
- Robo Advisors: Get expert investment advice and solutions with the Robo-advisors
- feature. This feature will analyze, monitor, optimize, and improve diversification in investments by turning data into actionable insights in real-time. Chats: Equip your expense tracking app with a bot that can understand and answer all user queries and address their needs such as account balance, credit score, etc.
- Prediction: With the help of AI, your mobile app can predict your next purchase, according to your spending behavior. Moreover, it can recommend products and provide unique insights on saving money. It brings out the factors causing fluctuations in your expenses.
- Employee Travel Budgeting: Most businesses save money with a travel budgeting app as It helps prepare a budget for an employee's entire business trip. The feature will predict the expenses and allocate resources according to the prediction.

APPENDIX:

The source code , demo video and github are in the link ([github](#))