

“BACHAT”

An Online Funds Record and Management Website

A PROJECT REPORT

Submitted By

Nainsi Kansal - 2100290140093

Prachi Verma - 2100290140103

Kiran Chauhan - 2100290140074

Anshima Saini - 2100290140029

**Submitted in partial fulfillment of the
Requirements for the Degree of**

MASTER OF COMPUTER APPLICATIONS

Under the Supervision of

**Dr. Vipin Kumar
Associate Professor**



Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS

KIET Group of Institutions, Ghaziabad

Uttar Pradesh-201206

(December 2022)

CERTIFICATE

Certified that Nainsi Kansal - 2100290140093, Prachi Verma - 2100290140103, Kiran Chauhan- 2100290140074, and Anshima Saini - 2100290140029 have carried out the project work having **“BACHAT”- An Online Funds Record and Management Website** for Master of Computer Applications from Dr. A.P.J. Abdul Kalam Technical University (AKTU), Technical University, Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date:

Nainsi Kansal - 2100290140093
Prachi Verma - 2100290140103
Kiran Chauhan - 2100290140074
Anshima Saini - 2100290140029

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

Dr. Vipin Kumar
Assistant Professor
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Name of Internal Examiner

Signature of External Examiner

Dr. Arun Kumar Tripathi
Head, Department of Computer Applications
KIET Group of Institutions, Ghaziabad

ABSTRACT

Bachat is a project which aims in developing an online application cum website to provide an Online platform to the user to, maintain weekly, monthly, and yearly expenditures of a person or family based on income. It helps in maintaining records and keeping the track of Payments and other expenses. This project has a variety of features, and logical and rational factors affecting the balance of one's budget. Users can log in to their own portal separately with a unique ID and password set. The server side is the admin side that can maintain and manage the database and security. This System can be used to maintain a track of income, money spent on a daily basis and analysis all expenses, and notify when the person overspends the money. Overall, this project of ours is being developed to help the people(users) to maintain and analyze the economy and notify them to maintain it.

It is built using React, Spring Boot, Figma, and My SQL

ACKNOWLEDGEMENT

Success in life is never attained single-handedly. My deepest gratitude goes to my thesis supervisor, **Dr. Vipin Kumar** for his guidance, help, and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Arun Kumar Tripathi, Professor and Head, of the Department of Computer Applications**, for his insightful comments and administrative help at various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Nainsi Kansal- 2100290140093

Prachi Verma - 2100290140103

Kiran Chauhan - 2100290140074

Anshima Saini - 2100290140029

Table of Contents

Certificate	2
Abstract	3
Acknowledgment	4
 Chapter 1: Introduction	 7
1.1 Project Description	7
Chapter 2: Literature Review	8
Chapter 3: Project Requirements	9
2.1 Technologies Used	9
Chapter 4: Software Requirement Specification	10
4.1 General Specification	10
4.2 Problem Statement	10
4.3 System Objective	11
4.4 Requirement Specification	11
4.5 Functional Requirements	12
4.6 Non-functional Requirements	12
4.7 Software And Hardware Requirements	13
4.8 Existing VS Proposed System	14
4.9 Software System Attributes	14
4.10 Features Of User Performance System	15
4.11 Preliminary Investigation	16
4.12 Model Used	17
4.13 Preliminary Description	18
Chapter 5: Analysis	19
5. Feasibility Study	19
Chapter 6: Planning and Scheduling	21
6.1 Gantt Chart	21
6.2 Data Flow Diagram	23
6.3 Class Diagram	26
6.4 Entity Relationship Diagram	27
6.5 Use Case Diagram	29
6.6 Activity Diagram	31
6.7 Sequence Diagram	33
6.8 Input Output Diagram	35
Chapter 7: CODING AND IMPLEMENTATION	37
7.1 Login	37

7.2 Sign Up.....	43
7.3 Dashboard.....	50
7.4 Contact Us.....	54
7.5 Transaction.....	59
7.6 Income.....	63
7.7 Sidebar.....	66
7.8 Spendings.....	69
7.9 Calendar.....	71
7.10 About Us.....	74
7.11 Set Goal.....	79
7.12 Home.....	82
Conclusion.....	88
References.....	89

CHAPTER 1

INTRODUCTION

1.1 PROJECT DESCRIPTION

This project has been developed to override the problems prevailing in the practicing manual system. This application is supported to eliminate and in some cases reduce the hardships faced by the existing system. Moreover, this application is designed for moving the office toward secure digitalization. It will enhance the transparency in the work.

No formal knowledge is needed for the user to use this system. Thus, by this it proves it is user-friendly, This **“BACHAT”- An Online Funds Record and Management Website** can lead to error-free, secure, reliable and fast system.

This application provides the way to manage the team, office and project for productive outcome. Through this applications we are approaching towards the less use of papers by virtue of which we are indirectly sustaining the environment.

CHAPTER 2

LITERATURE REVIEW

Emerging markets have been the subject of significant attention in the finance and trade literature in recent years, as their economic growth potential and increasing integration into the global economy have made them an important focus for researchers and policymakers alike. In this literature review, we will explore some of the key themes and debates in emerging markets finance and trade, including the challenges and opportunities associated with investing in these markets, the role of financial institutions in facilitating trade, and the impact of globalization on emerging markets.

One of the central challenges in emerging markets finance is the issue of risk. Emerging markets are often characterized by high levels of political and economic uncertainty, as well as relatively underdeveloped financial systems. This can make it difficult for investors to assess the risks associated with investing in these markets, and can lead to higher costs of capital and lower levels of investment. A number of studies have explored different approaches to mitigating these risks, including portfolio diversification, the use of hedging strategies, and the development of local financial markets.

Another important area of research in emerging markets finance is the role of financial institutions in facilitating trade. Financial intermediaries such as banks and other lenders play a critical role in enabling trade, by providing the necessary financing and risk management services that allow firms to expand into new markets. Studies have explored the factors that influence the availability and cost of credit in emerging markets, as well as the impact of financial sector reforms on trade and investment.

The impact of globalization on emerging markets is another area of significant interest in the literature. Globalization has brought many benefits to emerging markets, including increased trade and investment flows, improved access to technology and knowledge, and greater opportunities for economic growth. However, globalization has also exposed emerging markets to a range of new risks and challenges, including heightened competition from established firms in developed countries, and increased volatility in global financial markets.

CHAPTER 3

PROJECT REQUIREMENTS

3.1 TECHNOLOGIES USED

Operating Systems: Microsoft Windows

Front End: React, HTML/CSS, JavaScript

UI/UX: Figma

Software Requirements:

- Windows 10/11 or equivalent
- React
- Spring Boot
- RDBMS (Back end): MySQL

Hardware Requirements:

- Processor – Intel i3 5th generation or higher
- RAM – Minimum 4 GB, recommended 8 GB
- Disk space - Minimum 10 GB of free disk space
- Network Connectivity

CHAPTER 4

SOFTWARE REQUIREMENT SPECIFICATION

4.1 GENERAL DESCRIPTION

This combined aggregation of information and workplace activity constructs a general, specific program or aim which is to be executed or produced within the workplace while working with others as a squad. The history of coaction began many centuries ago, long before the B.C. or A.D. epochs, where at least two persons had to pass on in the attempt of finishing a undertaking, undertaking, or written papers. Therefore, coaction is non a new term, but an enhanced and improved one in the professional workplace.

4.2 PROBLEM STATEMENT

The problem occurred before having computerized system includes:

After discussing our application functions and comparing them to other existing applications, some features were found lacking. This is a new application that will attract the public user through its features. There are always some challenges. We have to face some challenges as well, since the main purpose of our application is to track the user's expenses.

- This is an Android-based mobile application, so if a user does not have an Android phone then this application will not help him.
- After getting notifications if a user doesn't check his phone for full information then the main motto of this app will fail.

4.3 SYSTEM OBJECTIVES

Improvement in control and performance

The system is developed to cope up with the current issues and problems of forgetting the traditional mechanism. The system identify who is accessing the profile and the data/information will be updated on the portal. To declare the Project and performance of the user and details.

Save cost

The existing system is based on the pen paper mode and several in the digital mode but is not secured and efficient to work.

Save Time

People at any location will able to perform or know there seniors subordinate team and there uniqueness etc. by registering or Logging in the Portal.

4.4 Requirement Specification

The application requirement specification is produced at the analysis task. The function and performance allocated to application as part of system engineering are refined by establishing a complete information description, a detailed functional and behavioral description, an indication of performance requirements and design constraints.

4.5 Functional Requirements

User Generated Email and Password

The application will work with the Email and password generated by the admin after joining the application.

Delete and Update the User

Admin will add, update and delete a user.

Register and Login

To work on the web application one should be registered and should have to login with the personal email and password which is also linked by their bank account.

4.6 Non-functional Requirements

Performance Requirements

- **User friendly:** The system should be user friendly so that it can easily be understood by the user without any difficulty.
- **Ease of maintenance :-** System should be easy to maintain and use.
- **Less time consuming:** The system should be less time consuming which could be achieved by good programming.
- **Error free:** The system should easily handle the user error in any case.
- **Static:** Application runs on stand alone machine . Support only single user.

4.7 SOFTWARE AND HARDWARE REQUIREMENTS

This section describes the software and hardware requirements of the system.

SOFTWARE REQUIREMENTS

Operating system- Windows/Linux Operating System This is the web Application which can run on any of the Operating System.

Database- MySQL is used in storing the data in structured manner.

SpringBoot- is a Software used for server which is use to serve the client what he/she wants from the server.

Browser- Any of the browser can be used to run and test the web application's Appearance and working e.g. Internet Explorer, Google Chrome, Mozilla Firefox etc.

Development tools and Programming language- React, UI/UX: Figma, HTML/CSS, JavaScript is used to write the whole web designing and operational code. SpringBoot is used for backend maintenance.

HARDWARE REQUIREMENTS

- Desktop/Laptop any configuration.

4.8 EXISTING VS PROPOSED SYSTEM

Existing system does not have a secure facility of **“BACHAT”- An Online Funds Record and Management Website** with transparency in Workplace whereas proposed system is secure and transparency in the work of the people.

Existing system does not have any facility of generating Email Online whereas proposed system is working on the facility of generating email and password online by the admin with security .

Existing System does not have the facility of registering and generating organizational password Whereas proposed system are more focused on it.

4.9 SOFTWARE SYSTEM ATTRIBUTES

- **Portability:-** The system should be machine independent.
- **Security:-** The system is designed in such a way that it will store the recorded data in the system of the owner. The system will be secure from unauthorized access of the application.
- **Maintainability:** The system will be designed in a maintainable order. The system can be easily modified and renewed according to the need of the organization

4.10 FEATURES OF USER PERFORMANCE SYSTEM

- Internet connection required against the computer.
- Multiple users can login and register on the same portal remotely.
- People can register and login in the system.
- Graphics with a classic look and the feel of a royal Web Application.
- Classic Profile Details to display profile of each user.
- Security of data to be stored.
- Ensures data accuracy (number of alerts generated).
- Minimize manpower.
- Minimize time consumption.
- Greater efficiency.
- Fast.
- Better services.
- User-friendliness and Interactive.
- Minimum time required.
- Easy to add, update and delete.
- User friendly.
- Free for the user.

4.11 Preliminary Investigation

After obtaining the background knowledge, we began to collect data on the existing system.

The tools that are used in information gathering are as follows:

- Online Apps observation.
- Review of the people

The model we have used is Prototype Model. In this model, first of all the existing system is observed, then customer requirements are taken in consideration then planning, modelling, construction and finally deployment and again adding the new system if asked by the customer to do so.

4.12 Model Used: Prototype Model

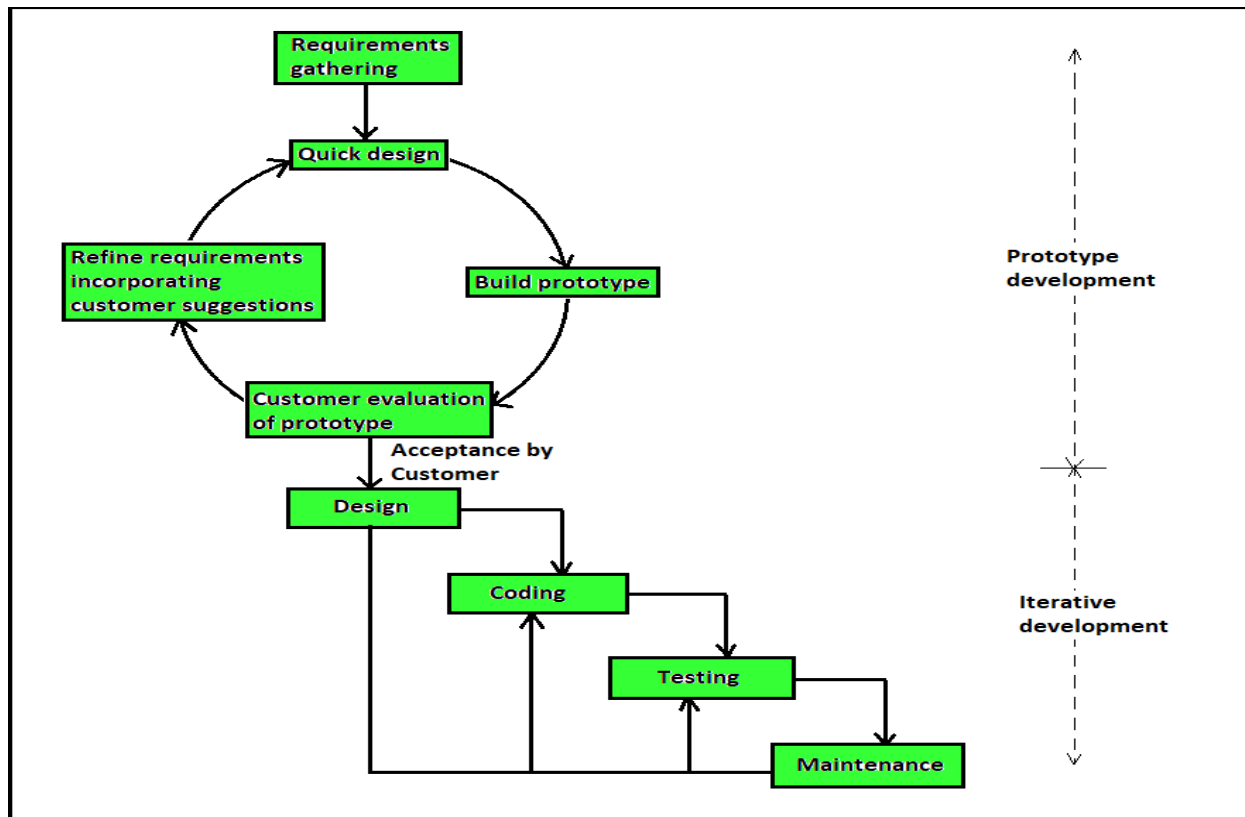


Fig. 4.12.1 : Prototype Model

The prototype model requires that before carrying out the development of actual software, a working prototype of the system should be built. A prototype is a toy implementation of the system. A prototype usually turns out to be a very crude version of the actual system, possibly exhibiting limited functional capabilities, low reliability, and inefficient performance as compared to actual software. In many instances, the client only has a general view of what is expected from the software product.

4.13 Preliminary Description

The first step in the system development life cycle is the preliminary investigation to determine the feasibility of the system. The purpose of preliminary investigation is to evaluate project requests. It is not a design study nor does it include the collection of details to describe the system in all respect. Rather, it is the collecting of information that helps committee members to evaluate the merits of project request and make an informed judgement about the feasibility of the proposed project.

An analyst working on the preliminary investigation should accomplish the following objectives:

- Clarify and understand the project request.
 - Determine the size of the project.
 - Access costs and benefits of alternative approaches.
 - Determine the technical and operational feasibility of alternative approaches.
 - Report the findings to management with recommendations outlining the acceptance and rejection of the proposal
-
-

CHAPTER 5

ANALYSIS

5.1 FEASIBILITY STUDY

After studying and analyzing all the existing and requires functionalities of the system, the next task is to do the feasibility study for the project. Feasibility study includes consideration of all the possible ways to provide a solution to a given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be easily done based on the future upcoming requirements.

- **ECONOMICAL FEASIBILITY**

For the economic feasibility, Economic analysis or cost/benefits analysis is most frequently used technique the effectiveness of a proposed system. it is a procedure to determine the benefits and saving those are expected from the proposes system and compare them with cost .if the benefits outweigh the costs, a decision is taken to design and implement the system. otherwise, further justification or alternative in proposed system will have to be made if it is to have a chance of being approved this is ongoing effort that improves in accuracy at each phase of a system life cycle

- **TECHNICAL FEASIBILITY**

This included the study of function, performance and constraints that may affect the ability to achieve an acceptable system. For this feasibility study, we studied complete functionalities to be provided in the system, as described in the System Requirement Specification (SRS), and checked if everything was possible using different type of front end and back end platform.

○ **OPERATIONAL FEASIBILITY**

No doubt the technically growing world needs more enhancement in technology, this apps is very user friendly and all inputs to be taken all self-explanatory even to a layman. As far our study is concerned, the clients will be comfortable and happy as the system has cut down their loads and bring the young generation to the same virtual world they are growing drastically.

Operational feasibility cover two aspects.one technical performance aspects and the other is acceptance within the organization.

Operation feasibility determine how the proposed the system will fit in with the current operation and what needs to implement the system.

CHAPTER 6

PLANNING AND SCHEDULING

6.1 GANTT CHART

A Gantt chart can be developed for the entire project or a separate chart can be developed for each function. A tabular form is maintained where rows indicate the task with milestones and columns indicate duration (Weeks).

Week	1	2	3	4	5	6	7	8	9	10	11	12
Activities												
Research												
Define Specification												
Project Planning												
Design												
Development												
Test Plan												
Testing and Q A												
Delivery												

Fig. 6.1.1: Gantt Chart

SOFTWARE REQUIREMENTS WITH SPECIFICATION

Name of Components	Specifications
Operating system	Windows
Language	React,HTML/CSS,Javascript
Software Development kit	Springboot, Google Chrome
Markup Language Enable	HTML

HARDWARE REQUIREMENTS WITH SPECIFICATIONS

Name of Components	Specifications
Desktop/Laptop	Any Configuration
Memory Used	6.31 MB

6.2 DATA FLOW DIAGRAM

Data flow diagram Are used to graphically represent the flow of data in a **“BACHAT”- An Online Funds Record and Management Website**. DFD describes the processes that are involved in a system to transfer data from the admin to the user, user to the user, user to admin etc.

“BACHAT”- An Online Funds Record and Management Website this system shows the flow of data in admin Modules on many Action. It shows the flow of data among the sub module in it Admin data flow on the sub screen.

User Performance System this system shows the flow of data in User Modules on many Action. It shows the flow of data among the sub module in it User data flow on the sub screen. It is with who is someone's expense manager.

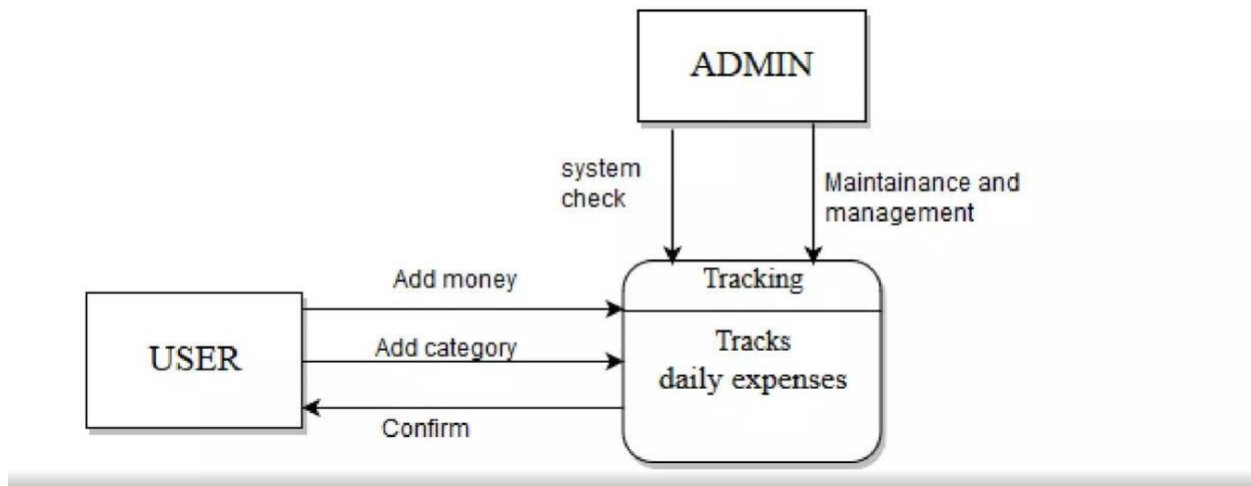


Fig. 6.2.1 DFD Level

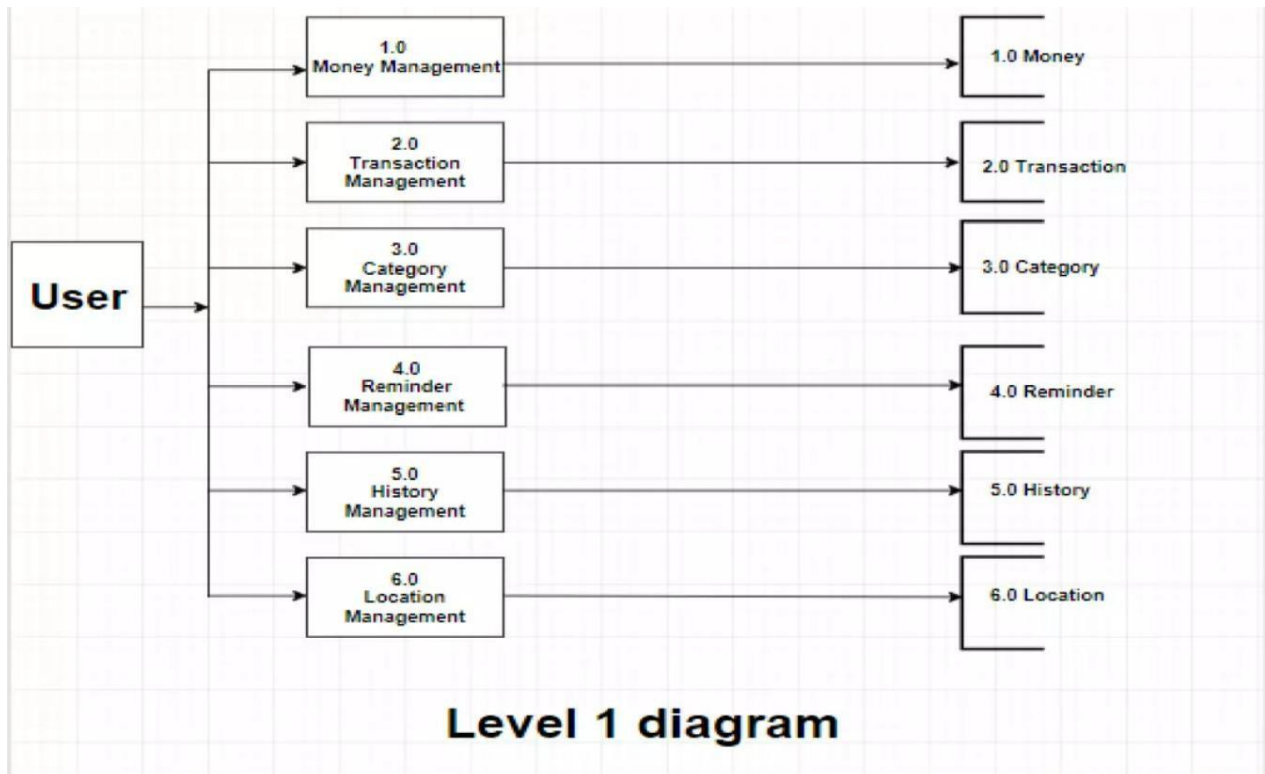


Fig. 6.2.2: User DFD Level 1

User Performance System this system shows the flow of data in User Modules on many Action. It shows the flow of data among the sub module in it User data flow on the sub screen. It is with who is not someone's expense manager.

6.3 CLASS DIAGRAM

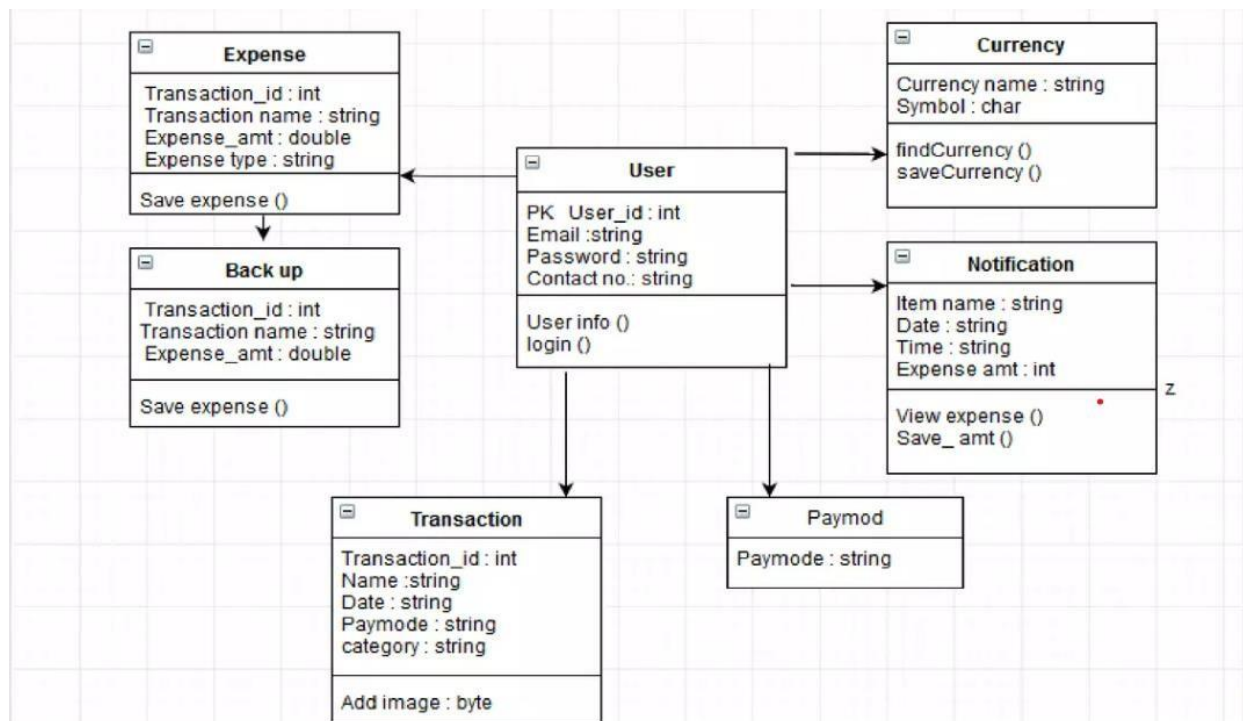


Fig. 6.3.1: Class Diagram

6.4 ENTITY RELATIONSHIP DIAGRAM

This ER Diagram represents the model of WorkPlace CoAction System Entity. The Entity Relationship Diagram show all visual instrument of Database table and relation between HomePage, Admin Page, User Page. All of it have Structured data and every entity may have some attributes.

User Performance System Entity and their Attributes:

1. Admin: Attribute of Admin: Email id, Password, Forget Password.
2. Insert New User Details: Attributes are: Name, Email, Password, Gender, Phone.
3. Delete New User: Attributes are: Name, Email, Password, Gender, Phone.
4. Update Details of Self: Attributes are: Name, Email, Password, Gender, Phone.
5. User: Attribute of User: Email id, Password, Forget Password.
6. User Update Details: Attributes are: Name, Email, Gender, Phone.
7. Search User: Attributes are: Name, Email, Gender, Phone.
8. View Profile of User: Attributes are: Name, Email, Gender, Phone.

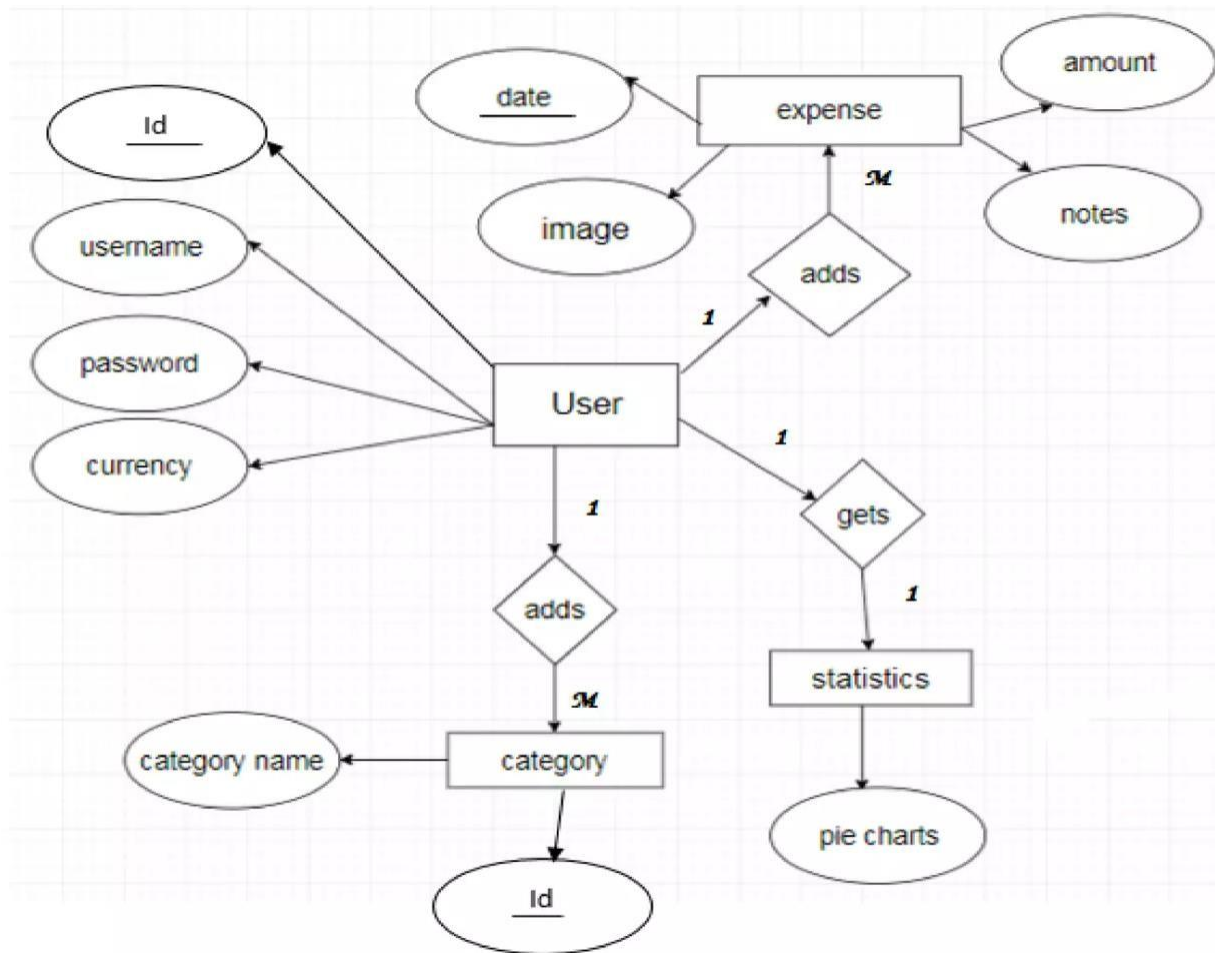


Fig. 6.4.1: Entity Relationship Diagram

6.5 USE CASE DIAGRAM

Use-case diagrams model the behavior of a system and help to capture the requirements of the system. Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors.

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system.

Purposes of a use case diagram given below:

1. It gathers the system's needs.
2. It depicts the external view of the system.
3. It recognizes the internal as well as external factors that influence the system.
4. It represents the interaction between the actors.

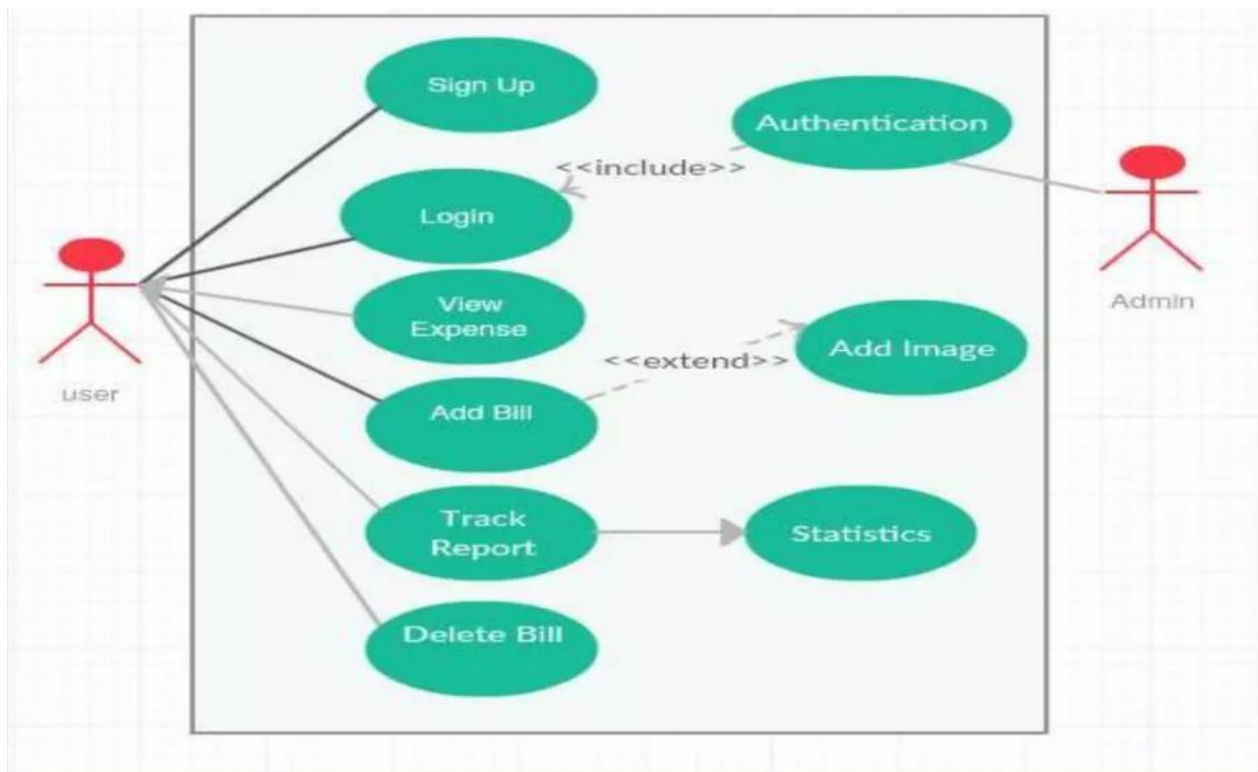


Fig. 6.5.1: Use Case Diagram

6.6 ACTIVITY DIAGRAM

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.

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

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc

The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior 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 a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

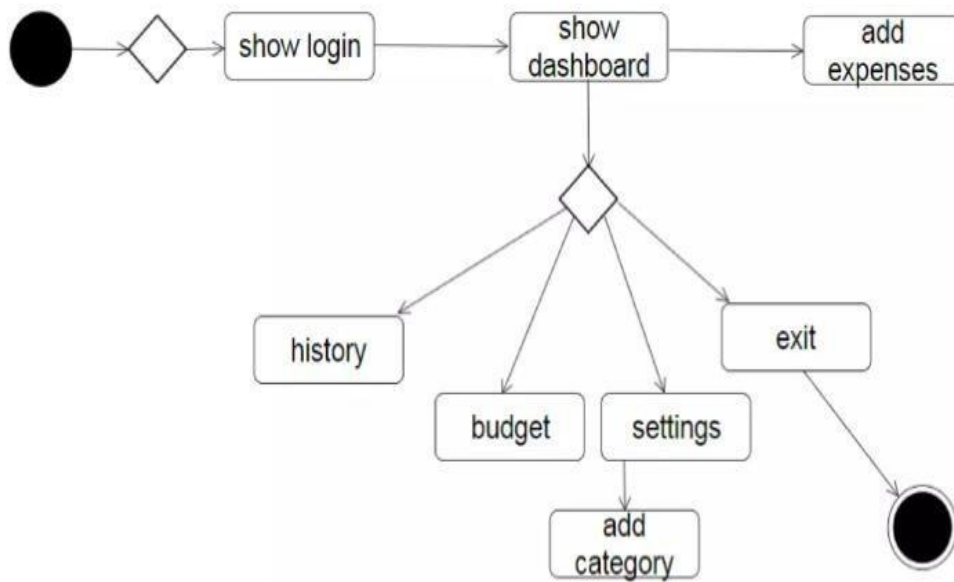
Activity Diagram

Fig. 6.6.1: Activity Diagram

6.7 SEQUENCE DIAGRAM

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.

Sequence diagram

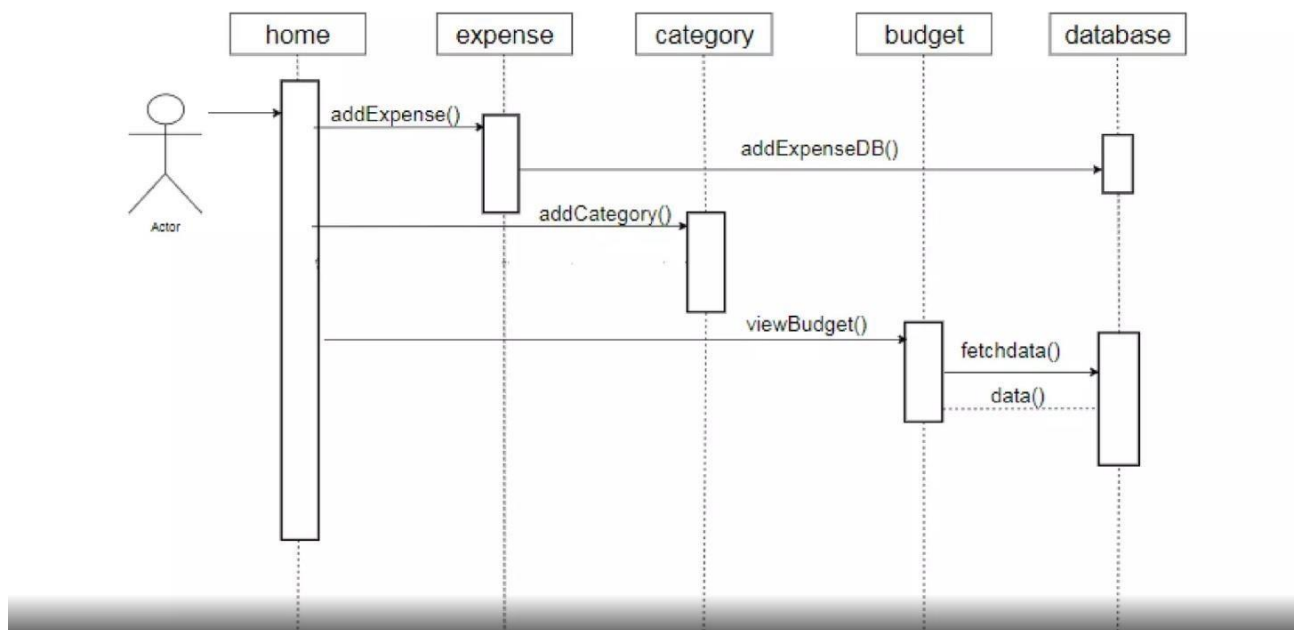


Fig. 6.7.1: Sequence Diagram

6.8 INPUT OUTPUT DIAGRAM

The **input–process–output (IPO) model** is a widely used approach in systems analysis and software engineering for describing the structure of an information processing program or another process. Many introductory programming and systems analysis texts introduce this as the most basic structure for describing a process.

Input Output Diagram

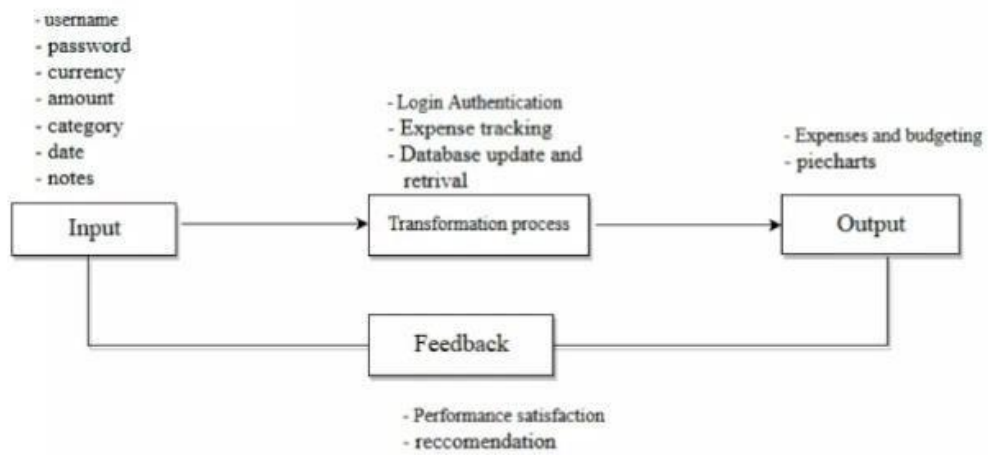


Fig. 6.8.1: Input Output Diagram

CHAPTER 7

CODING AND IMPLEMENTATION

7.1 LOGIN

```
import React, { useRef, useState } from "react";
import { useNavigate } from "react-router-dom";

const Login = () => {
  const emailRef = useRef("");
  const passwordRef = useRef("");
  const [error, setError] = useState(false);
  const navigate = useNavigate();

  const handleSignUp = () => {
    navigate("/signUp");
  };

  async function submitHandler(event) {
    event.preventDefault();

    const user = {
      emailId: emailRef.current.value,
      password: passwordRef.current.value,
```

```

};

const response = await fetch("http://localhost:8080/user/login", {
  method: "POST",
body: JSON.stringify(user),
  headers: {
    "Content-Type": "application/json",
  },
});

if (response.ok) {
  navigate("/dashboard");
} else {
  setError(true);
}
}

return (
  <section className="vh-50 bg-image">
    <div className="mask d-flex align-items-center h-100 gradient-custom-3">
      <div className="container h-100">
        <div className="row d-flex justify-content-center align-items-center h-100">
          <div className="col-12 col-md-9 col-lg-7 col-xl-6">
            <div className="card">

```

Login to Application

```
</h2>
```

```
<div>
```

```
  className="error"
```

```
  style={ {
```

```
    display: error ? "" : "none",
```

```
    textAlign: "center",
```

```
    color: "red",
```

```
    fontSize: "18px",
```

```
    fontStyle: "italic",
```

```
    fontWeight: "bold",
```

```
  } }
```

```
>
```

```
<p>
```

```
  Invalid Email Or Password...<br></br>
```

```
</p>
```

```
</div>
```

```
<form onSubmit={submitHandler}>
```

```
  <label htmlFor="email">
```

```
    <h6>E-Mail / UserName</h6>
```

</label>

</br>

<input

type="text"

id="email"

className="form-control"

placeholder="E-mail"

required={true}

ref={emailRef}

/>

</br>

<label htmlFor="password">

<h6>Password:</h6>

</label>

</br>

<input

type="text"

id="password"

className="form-control"

SignUp

</button>

</form>

</div>

</div>

</div>

</div>

</div>

</div>

</section>

);

export default Login;

7.2 SIGNUP

```
import React, { useRef } from "react";  
import { useNavigate } from "react-router-dom";
```

```
const SignUp = () => {  
  const emailRef = useRef("");  
  const firstNameRef = useRef("");  
  const lastNameRef = useRef("");  
  const mobileNumRef = useRef("");  
  const passwordRef = useRef("");  
  const confirmPasswordRef = useRef("");
```

```
  const navigate = useNavigate();
```

```
  async function submitSignUp(event) {  
    event.preventDefault();
```

```
const user = {  
  emailId: emailRef.current.value,  
  firstName: firstNameRef.current.value,  
  lastName: lastNameRef.current.value,  
  mobileNum: mobileNumRef.current.value,  
  password: passwordRef.current.value,  
  confirmPassword: confirmPasswordRef.current.value,  
};  
  
const response = await fetch("http://localhost:8080/user/signup", {  
  method: "POST",  
  body: JSON.stringify(user),  
  headers: {  
    "Content-Type": "application/json",  
  },  
});  
  
if (response.ok) {  
  navigate("/login");  
}  
}
```

```

return (
  <section className="vh-50 bg-image">
    <div className="mask d-flex align-items-center h-100 gradient-custom-3">
      <div className="container h-100">
        <div className="row d-flex justify-content-center align-items-center h-100">
          <div className="col-12 col-md-9 col-lg-7 col-xl-6">

<div className="card">
  <div className="card-body p-5">
    <h2 className="text-uppercase text-center mb-5">
      New User Registration
    </h2>

    <form onSubmit={ submitSignUp }>
      <br></br>
      <label htmlFor="email">
        <h6>Email_Id / UserName:</h6>
      </label>
      <input
        type="text"
        id="email"

```

```

className="form-control"
    placeholder="Email/Username"
    required={true}
    ref={emailRef}
  />
  <br></br>
  <label htmlFor="firstname">
    <h6>First_Name:</h6>
  </label>
  <input
    type="text"
    id="firstname"
    className="form-control"
    placeholder="FirstName"
    required={true}

ref={firstNameRef}
  />
  <br></br>
  <label htmlFor="lastName">
    <h6>Last_Name:</h6>
  </label>
  <input>

```

```
type="text"
    id="lastName"
    className="form-control"
    placeholder="lastName"
    required={true}
    ref={lastNameRef}
  />
<br></br>
<label htmlFor="phoneNum">
  <h6>Contact_Number:</h6>
</label>
<input
  type="text"
  id="phonenum"
  className="form-control"
  placeholder="PhoneNum"
  required={true}
  ref={mobileNumRef}
/>
<br></br>
<label htmlFor="password">
  <h6>Password:</h6>
```

```
</label>

<br></br>

<input
  type="text"
  id="password"
  placeholder="Password"
  className="form-control"
  required={true}
  ref={passwordRef}
/>

<br></br>

<label htmlFor="cpassword">
  <h6>Confirm:</h6>
</label>

<br></br>

<input
  type="text"
  id="cpassword"
  className="form-control"
  placeholder="Confirm Password"
  required={true}
```


7.3 DASHBOARD

```
import React from "react";
import { Link } from "react-router-dom";
import "./SideBar.css";

const SideBar = () => {
  return (
    <div>
      <input type="checkbox" id="check" />
      <label htmlFor="check">
        <i className="fas fa-times" id="cancel"></i>
        <i className="fas fa-bars" id="btn"></i>
      </label>

      <div className="sidebar">
        <header>
          <i className="fa fa-rupee-sign"> &nbsp;&nbsp;  Bachat 💰 </i>
        </header>
        <Link to="/transactions">
          <i className="fa fa-credit-card"></i>
          <span>Transactions</span>
        </Link>
      </div>
    </div>
  );
};
```

```
<Link to="/income">
```

```
  <i className="fa fa-calculator"></i>
```

```
  <span>Income</span>
```

```
</Link>
```

```
<Link to="/spendings">
```

```
  <i className="fa fa-archive"></i>
```

```
  <span>Spendings</span>
```

```
</Link>
```

```
<Link to="/savings">
```

```
  <i className="fa fa-briefcase" />
```

```
  <span>Savings</span>
```

```
</Link>
```

```
<Link to="/setgoals">
```

```
  <i className="fas fa-bullseye"></i>
```

```
  <span>Set Goals</span>
```

```
</Link>
```

```
<Link to="/charts">
```

```
  <i className="fa fa-database"></i>
```

```
  <span>Charts</span>
```

```

</Link>
  <Link to="/calendarApp">
    <i className="fa fa-calendar"></i>
    <span>Calendar</span>
  </Link>

  <Link to="/contactUs">
    <i className="far fa-envelope"></i>
    <span>Contact</span>
  </Link>
  <Link to="/login">
    <i className="fa fa-arrow-circle-left"></i>
    <span>Log Out</span>
  </Link>
</div>
</div>
);
};
export default SideBar;
import React from "react";
import SideBar from "../SideBar";

```

```
const DashBoard = () => {  
  return (  
    <div>  
      <div id="DashBoard">  
        <SideBar pageWrapId={"page-wrap"} outerContainerId={"DashBoard"} />  
      </div>  
  
    </div>  
  );  
};  
  
export default DashBoard;
```

7.4 CONTACT US

```
import React from "react";
import { Button } from "react-bootstrap";
import SideBar from "../SideBar";
const ContactUs = () => {
  const [formStatus, setFormStatus] = React.useState("Send Message");
  const onSubmit = (e) => {
    e.preventDefault();
    setFormStatus("Submitting...");
    const { name, email, message } = e.target.elements;
    let conFom = {
      name: name.value,
      email: email.value,
      message: message.value,
    };
    console.log(conFom);
  };

  return (
    <div>
      <div>
```

```
<SideBar />
```

```
</div> ;
```

```
<div className="container mt-5">
```

```
<div className="row d-flex justify-content-center align-items-center h-100">
```

```
<div className="col-12 col-md-9 col-lg-7 col-xl-6">
```

```
<div className="card">
```

```
<div className="card-body p-5">
```

```
<h1 className="mb-3" style={{ textAlign: "center" }}>
```

```
  Contact Us
```

```
</h1>
```

```
<p
```

```
  style={{
```

```
    textAlign: "center",
```

```
    fontWeight: "bold",
```

```
    color: "green",
```

```
    fontSize: "1.2rem",
```

```
  }}

```

```
>
```

```
  Got a Question ? We'd to love hear from you.Send us a message
```

```
  and we'll respond as soon as possible
```

```
</p>
```

```

<form onSubmit={onSubmit}>
  <div className="mb-3">
    <label className="form-label" htmlFor="name">
      Name
    </label>
    <input
      className="form-control"
      type="text"
      id="name"
      required
    />
  </div>
  <div className="mb-3">
    <label className="form-label" htmlFor="email">
      Email
    </label>
    <input
      className="form-control"
      type="email"
      id="email"

```


required

```
    />
  </div>
  <div className="mb-3">
    <label className="form-label" htmlFor="message">
      Message
    </label>
    <textarea
      className="form-control"
      rows="10"
      id="message"
      required
    />
  </div>
  <Button type="submit">{formStatus}</Button>
```


7.5 TRANSACTIONS

```
import React, { useState } from "react";
import SideBar from "../SideBar";
import DateRangePicker from "react-bootstrap-daterangepicker";
import "bootstrap/dist/css/bootstrap.css";
import "bootstrap-daterangepicker/daterangepicker.css";
import moment from "moment";

const Transactions = () => {
  const [quote, setQuote] = useState("");
  const [index, setIndex] = useState(0);
  const [user, setUser] = useState();
  const [fromDate, setFromDate] = useState(new Date());
  const [toDate, setToDate] = useState(new Date());
```

```

const range = {
  Today: [moment(), moment()],
};

const handleEvent = (event, picker) => {
  console.log("start: ", picker.startDate._d);
  console.log("end: ", picker.endDate._d);
  setFromDate(picker.startDate._d.toISOString());
  setToDate(picker.endDate._d.toISOString());
};

const QuoteAPI = async () => {
  let arrayOfQuotes = [];
  const data = await axios.post("");
  arrayOfQuotes = data.data;
  console.log(arrayOfQuotes);
  const quote = arrayOfQuotes.map((arrayOfQuote) => (
    <div key={arrayOfQuote.id}>
      <h3>{arrayOfQuote.text}</h3>
      <p>{arrayOfQuote.author}</p>
    </div>
  ));

```



```

<DatePicker
  ranges={range}
  alwaysShowCalendars={false}
  onEvent={handleEvent}
>
  <button>
    {moment(fromDate).format("LL")} to {moment(toDate).format("LL")}
  </button>
</DatePicker>

</div>
</div>
);
};
export default Transactions;

```

7.6 INCOME

```
import './Savings.css';
import SideBar from './SideBar';
import { useRef } from 'react';
import React, { useState } from 'react';

const Income = () => {
  const incomeRef = useRef("");
  const sourceRef = useRef("");
  const dateRef = useRef("");

  async function submitHandler(event) {
    event.preventDefault();
    const incomeData = {
      income: incomeRef.current.value,
      sourceIncome: sourceRef.current.value,
      incomeDate: dateRef.current.value,
    };
  }
}
```

```

const response = await fetch("http://localhost:8080/income/", {
  method: "POST",
  body: JSON.stringify(incomeData),
  headers: {
    "Content-Type": "application/json",
  },
});
if (response.ok) {
  console.log("ok");
  return response.json();
}
}
return (
  <form onSubmit={handleSubmit}>
    <div>
      <SideBar />
    </div>
    <div className="new-expense">
      <div className="new-expense__control">
        <label>Source Of Income</label>
        <input type="text" required={true} ref={sourceRef} />
      </div>

```



```

<div className="new-expense__control">
  <label>Income</label>
  <input type="number" required={true} ref={incomeRef} />
</div>
<div className="new-expense__control">
  <label>Date</label>
  <input type="date" required={true} ref={dateRef} />
</div>
<div className="new-expense__actions">
  <button
    type="submit"
    onChange={(event) => {
      getSaveIncome(event.target.value);
    }}
  >
    Add Income
  </button>
</div>
</div>
</form>
);
};
export default Income;

```

7.7 SIDEBAR

```
import React from "react";
import { NavLink } from "react-router-dom";
import "./SideBar.css";

const SideBar = () => {
  return (
    <div>
      <input type="checkbox" id="check" />
      <label htmlFor="check">
        <i className="fas fa-times" id="cancel"></i>
        <i className="fas fa-bars" id="btn"></i>
      </label>
      <div className="sidebar">
        <header>
          <i className="fa fa-rupee-sign"> &nbsp;&nbsp;  Bachat 💰 </i>
        </header>
        <NavLink to="/transactions">
          <i className="fa fa-credit-card" style={{ color: "green" }}></i>
        </NavLink>
      </div>
    </div>
  );
}
```

```
<span>Transactions</span>
```

```
</NavLink>
```

```
<NavLink to="/income">
```

```
  <i className="fa fa-calculator" style={{ color: "green" }}></i>
```

```
  <span>Income</span>
```

```
</NavLink>
```

```
<NavLink to="/spendings">
```

```
  <i className="fa fa-archive" style={{ color: "green" }}></i>
```

```
  <span>Expense</span>
```

```
</NavLink>
```

```
<NavLink to="/savings">
```

```
  <i className="fa fa-briefcase" style={{ color: "green" }} />
```

```
  <span>Savings</span>
```

```
</NavLink>
```

```
<NavLink to="/setgoals">
```

```
  <i className="fas fa-bullseye" style={{ color: "green" }}></i>
```

```
  <span>Set Goals</span>
```

```
</NavLink>
```

```

<NavLink to="/history">
  <i className="fa fa-credit-card" style={{ color: "green" }}></i>
  <span>History</span>
</NavLink>

<NavLink to="/charts">
  <i className="fa fa-database" style={{ color: "green" }}></i>
  <span>Charts</span>
</NavLink>

<NavLink to="/calendarApp">
  <i className="fa fa-calendar" style={{ color: "green" }}></i>
  <span>Calendar</span>
</NavLink>

<NavLink to="/">
  <i className="fa fa-arrow-circle-left" style={{ color: "green" }}></i>
  <span>Log Out</span>
</NavLink>
</div>
</div>

);

};

export default SideBar;

```

7.8 SPENDINGS

```
import React, { useState } from "react";

import SideBar from "../SideBar/SideBar.js";

import Expenses from "../Expenses/Expenses.js";

import NewExpense from "../NewExpense/NewExpense.js";

const DUMMY_EXPENSES = [];

const Spendings = () => {
  const [expenses, setExpenses] = useState(DUMMY_EXPENSES);
  const addExpenseHandler = (expense) => {
    setExpenses((prevExpenses) => {
      return [expense, ...prevExpenses];
    });
  };

  return (
    <div className="vh-100" style={{ background: "#DCF2C4" }}>
      <SideBar />
```

```
<NewExpense onAddExpense={addExpenseHandler} />  
  <Expenses items={expenses} />  
</div>  
  
);  
  
};  
  
export default Spendings;
```

7.9 CALENDER

```
import "react-calendar/dist/Calendar.css";
import Calendar from "react-calendar";
import SideBar from "../SideBar/SideBar";
import { useState } from "react";

const CalendarApp = () => {
  const [date, setDate] = useState(new Date());
  return (
    <div>
      <SideBar />
      <div
        style={{
          marginLeft: "650px",
          padding: "50px",
        }}
      >
```

```

<h1
  style={{
    color: "black",
    fontSize: "18px",
    fontStyle: "italic",
    fontWeight: "bold",
    marginLeft: "30px",
    padding: "20px",
  }}
>
  You can select your range here
</h1>
<div>
  <Calendar onChange={setDate} value={date} selectRange={true} />
</div>
{date.length > 0 ? (
  <p
    style={{
      color: "black",

```



```

        fontSize: "18px",
        fontStyle: "italic",
        fontWeight: "bold",
        padding: "10px",
    }}
    >
    <span>Start:</span> { date[0].toDateString()}
    &nbsp; to &nbsp;
    <span>End:</span> { date[1].toDateString()}
    </p>
    ) : (
    <p>
    <span>Default selected date:</span> { date.toDateString()}
    </p>
    )}
    </div>
    </div>
    );
};
export default CalendarApp;

```

7.10 ABOUT US

```
import Carousel from "react-bootstrap/Carousel";

import image0 from "../Images/image0.jpg";
import image1 from "../Images/image1.png";
import image2 from "../Images/image2.png";
import image3 from "../Images/image3.png";
import image4 from "../Images/image4.png";
import { NavLink } from "react-router-dom";

import "../AboutUs.css";

function AboutUs() {
  return (
    <div>
      <nav className="topnav1">
        <NavLink to="/">Home</NavLink>
        <NavLink to="/login">Login</NavLink>
        <NavLink to="/contactUs">ContactUs</NavLink>
      </nav>
```

```
<div className="text-center" style={{ backgroundColor: "#F8F1BA" }}>
```

```
  <Carousel fade>
```

```
    <Carousel.Item>
```

```
      <img
```

```
        className="mx-auto d-block"
```

```
        width="800"
```

```
        height="800"
```

```
        src={image0}
```

```
        style={{
```

```
          marginTop: "70px",
```

```
        }}
      </img>
```

```
      alt="First slide"
    </Carousel.Item>
```

```
  </Carousel>
```

```
</div>
```

```
<Carousel.Item>
```

```
  <img
```

```
    className="image-fluid"
```

```
    width="800"
```

```
    height="800"
```

```
    src={image1}
  </img>
```

```

    style={{
      marginTop: "70px",
    }}
    alt="Second slide"
  />
</Carousel.Item>
<Carousel.Item>
  <img
    className="mx-auto d-block"
    width="800"
    height="800"
    src={image2}
    style={{
      marginTop: "70px",
    }}
    alt="Third slide"
  />
</Carousel.Item>
<Carousel.Item>
  <img

```

```
        className="mx-auto d-block"
        width="800"
        height="800"
        src={image3}
        style={{
            marginTop: "70px",
        }}
        alt="Fourth slide"
    />
</Carousel.Item>
<Carousel.Item>
    <img
        className="mx-auto d-block"
        width="800"
        height="800"
        style={{
            marginTop: "70px",
        }}
        src={image4}
        alt="Fifth slide"
    />
```

```
</Carousel.Item>
```

```
  </Carousel>
```

```
</div>
```

```
</div>
```

```
);
```

```
}
```

```
export default AboutUs;
```

7.11 SET GOAL

```
import React, { useRef } from "react";
import SideBar from "../SideBar/SideBar.js";
import Popup from "reactjs-popup";
import "reactjs-popup/dist/index.css";

const SetGoal = () => {
  const monthRef = useRef("");
  return (
    <div>
      <SideBar />
      <div className="new-expense">
        <div className="new-expense__control">
          <label>Select Month:</label>
          <select required={true} ref={monthRef}>
            <option value="JANUARY">January</option>
            <option value="FEBRUARY">February</option>
```

```

<option value="MARCH">March</option>
<option value="APRIL">April</option>
<option value="MAY">May</option>
<option value="JUNE">June</option>
<option value="JULY">July</option>
<option value="AUGUST">August</option>
<option value="SEPTEMBER">September</option>
<option value="OCTOBER">October</option>
<option value="NOVEMBOR">Novembor</option>
<option value="DECEMBER">December</option>
</select>

<label>Enter Spending Target:</label>

<input type="number" required={true} />

<div className="new-expense__actions">

  <Popup trigger={<button type="submit"> Add Target</button>}>

    <div style={{ color: "green", fontWeight: "bold" }}>

      Added Successfully 💰
    
```



```
</div>
  </Popup>
</div>
</div>
</div>
</div>
);
};

export default SetGoal;
```

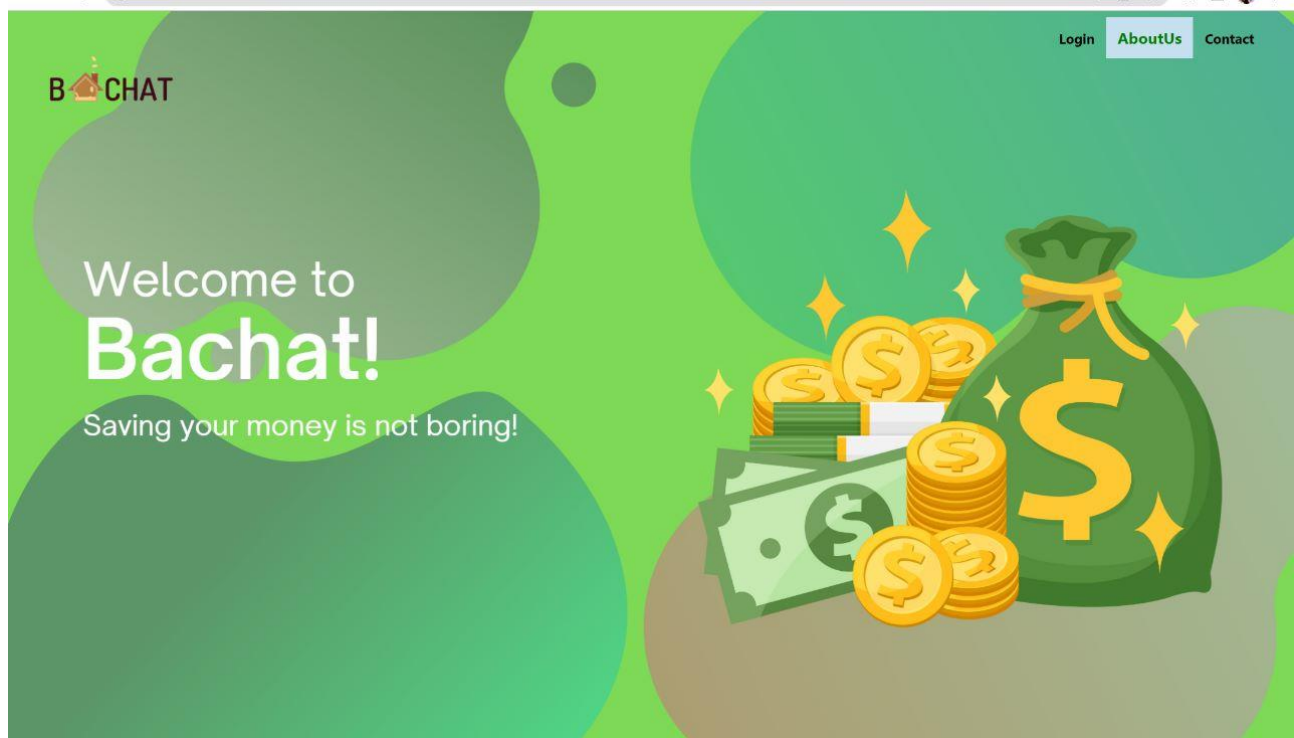
7.12 HOME

```
import { NavLink } from "react-router-dom";

import "./Home.css";

const Home = () => {
  return (
    <div className="bg-img">
      <div className="page">
        <div className="topnav">
          <NavLink to="/login">Login</NavLink>
          <NavLink to="/aboutUs">AboutUs</NavLink>
          <NavLink to="/contactUs">Contact</NavLink>
        </div>
      </div>
    </div>
  );
};

export default Home;
```



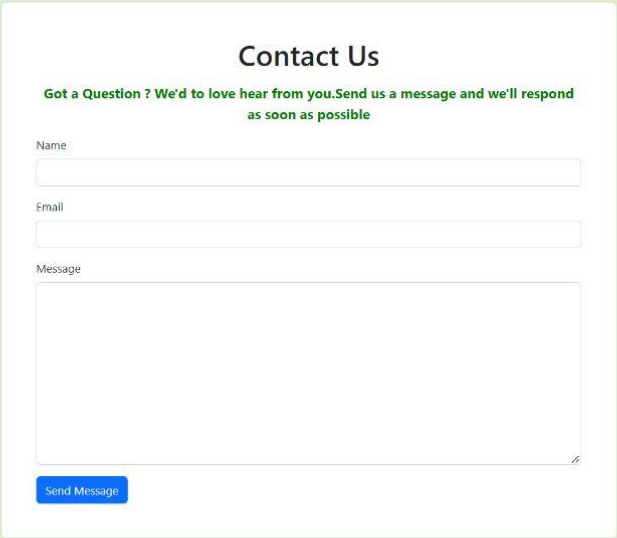
HOMEPAGE

This is the first page of the project website. This page contains the logo of the website and there are three navigation options available at the right side of the page.

The user can login to the website after clicking on the Login option.

The user can get an overview of the website after clicking on the AboutUs page.

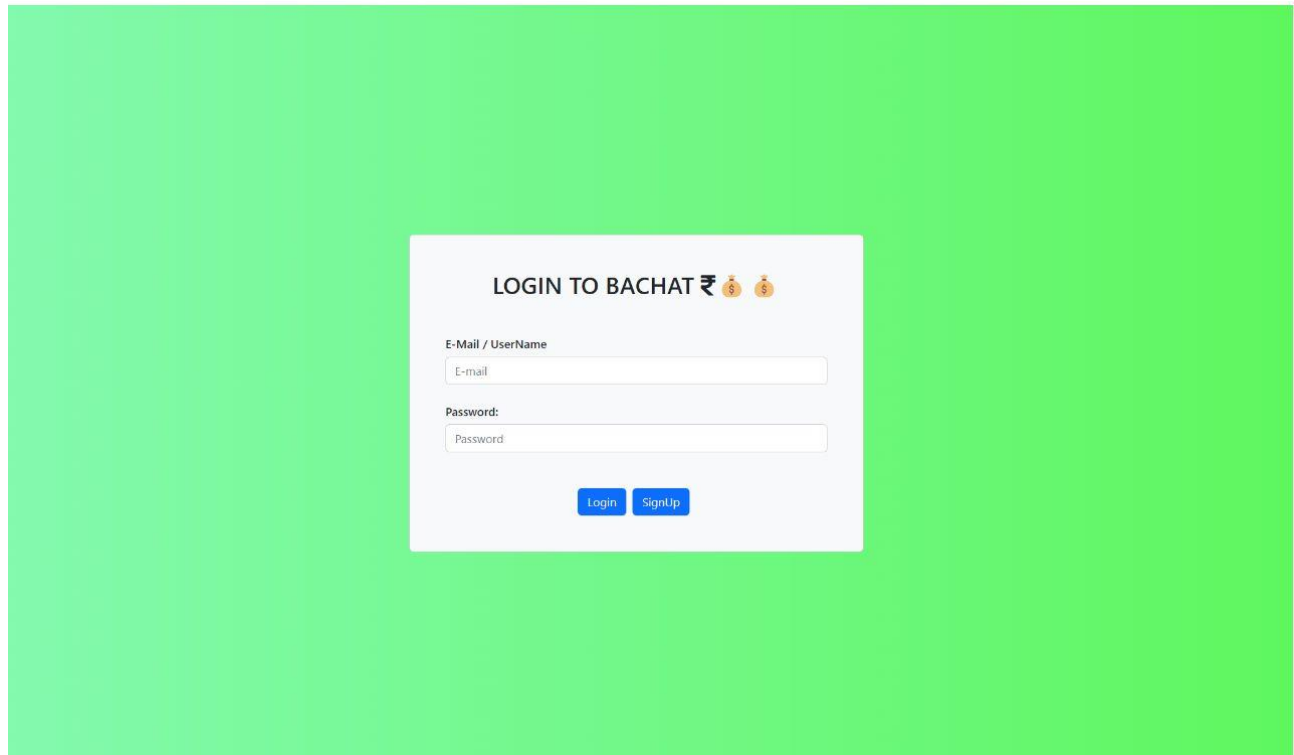
The user can contact the project developers after clicking on the Contact option.



The image shows a 'Contact Us' form centered on a light green background. The form is a white rectangle with a thin grey border. At the top, the title 'Contact Us' is centered in a bold, black font. Below the title, a green message reads: 'Got a Question ? We'd to love hear from you.Send us a message and we'll respond as soon as possible'. The form contains three input fields: a 'Name' field, an 'Email' field, and a larger 'Message' field. Each field is preceded by its respective label. At the bottom left of the form is a blue button with the text 'Send Message' in white. A small cursor icon is visible at the bottom right of the message text area.

CONTACT PAGE

This page is designed for the user to interact with the developers. User can share their feedback and also provide some suggestions to the developers in order to overcome the project errors.



The image shows a login page for a website called "BACHAT". The page has a solid green background. In the center, there is a white rectangular box containing the login form. The form has a title "LOGIN TO BACHAT ₹ \$ \$" at the top. Below the title, there are two input fields: "E-Mail / UserName" and "Password:". The "E-Mail / UserName" field has a placeholder text "E-mail". The "Password:" field has a placeholder text "Password". At the bottom of the form, there are two blue buttons: "Login" and "SignUp".

LOGIN PAGE

In order to use this project website the user must require to login to the website by entering a valid email Id and password.

If an user does not have an account he must require to create an account by clicking on the SignUp button where he need to enter his name, email Id, password then he need to click on the Login button to with the previous information entered while creating the account .

₹ Bachat ✕

TRANSACTIONS

INCOME

EXPENSE

SAVINGS

SET GOALS

HISTORY

CHARTS

CALENDAR

LOG OUT

Category

Amount

Clothes

Date

dd-mm-yyyy

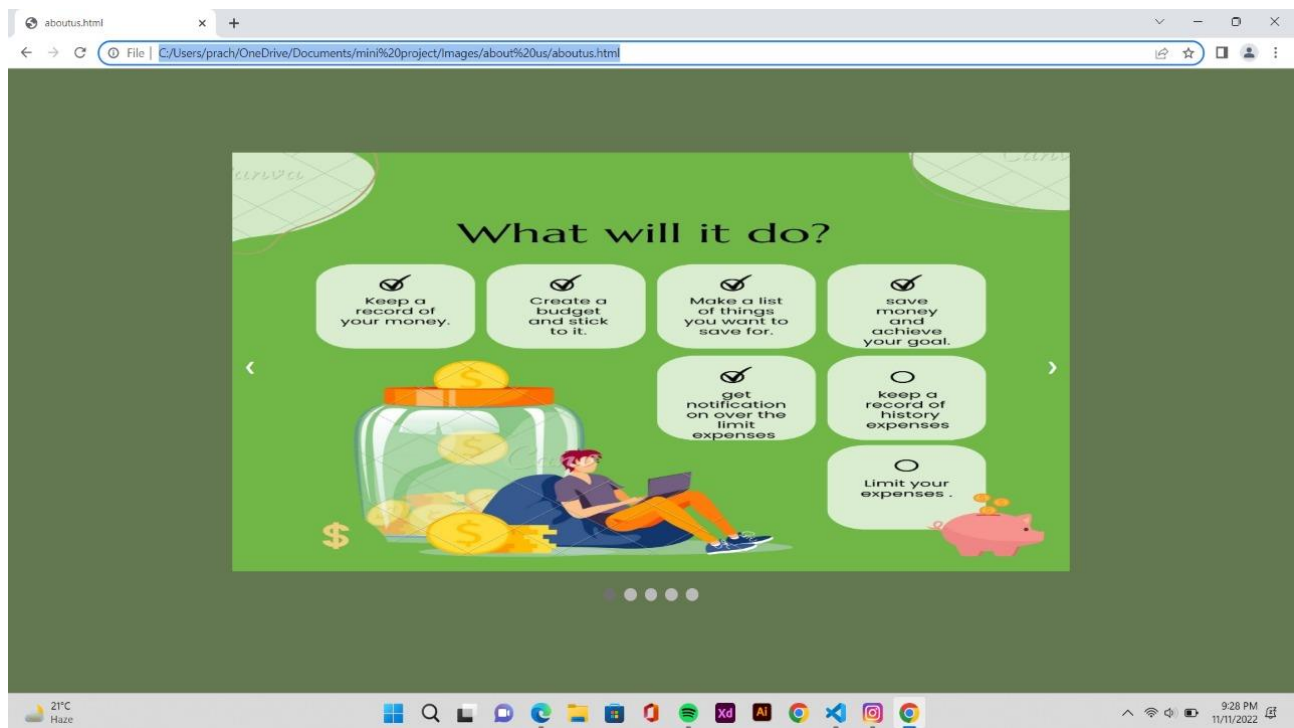
Add Expense

MAIN PAGE

This is the page which will appear after logging on the project website. It contains dashboard that provides platform to perform operations like adding transactions, add savings etc.

Here user can get full overview of his previous transactions. He can also set a target to limit his future expenses.

After using the website user can log out. The details will be stored and can be accessed in the time of need.



ABOUT

This about page will give a brief introduction about what this project is all about and what are the functionalities that are being provided to the user.

The benefits of using this website are also mentioned in this page. The slideshow of this page helps the user to quickly get an overview of this project.

CONCLUSION

At the end, the project helps in savings and maintaining the funds of a family or person. It helps users like women who manage the family expenses and thrive to save as Indians tend to save. The goal is achieved as all the data is stored easily and analysis is done simultaneously. When the economy of people improves the economy of the country improves and it helps in the development of all. The future goal is to make a Mobile application with updated features to make it more user-friendly and easier and more comfortable.

REFERENCES

<https://stackoverflow.com/>

<https://www.geeksforgeeks.org/>

<https://www.behance.net/>

Clean Code: A Handbook of Agile Software Craftsmanship is a book written by Robert. C. Martin.