PROJECT REPORT

Project Title :
Personal Finance Manager

MADE BY:

EDWIN CHAZHOOR (22BAI1002)

COURSE CODE: BCSE302L
TITLE: DATABASE SYSTEMS

PROBLEM STATEMENT

People struggle to manage finances effectively, which leads to a necessity of a Personal Finance Manager. This solution aims to empower users by organizing and optimizing income, expenses, savings, and investments. Ensuring user-friendly interfaces, security, it encourages planned financial decisions, reducing stress and promoting proactive financial planning.

PROBLEM DESCRIPTION

Lots of people have a hard time handling their money. It's tough for them to keep track of how much they make, spend, save, and invest, and this can make them feel stressed and cause them to make not-so-great financial choices. The tools available right now aren't easy to use, safe, or personalized, which makes it even more difficult for people to make smart decisions about their money. What's really needed is a simple and safe Personal Finance Manager that can help solve these problems and give people the confidence to improve their financial situation.

ASSUMPTIONS

- Users will actively engage with the platform and consistently input accurate financial data.

- The implemented security measures (authentication using unique user_id and password) are strong enough to protect user data.

- Budgeting and Savings: Users will be able to set up and track monthly budgets and keep a track of their savings.

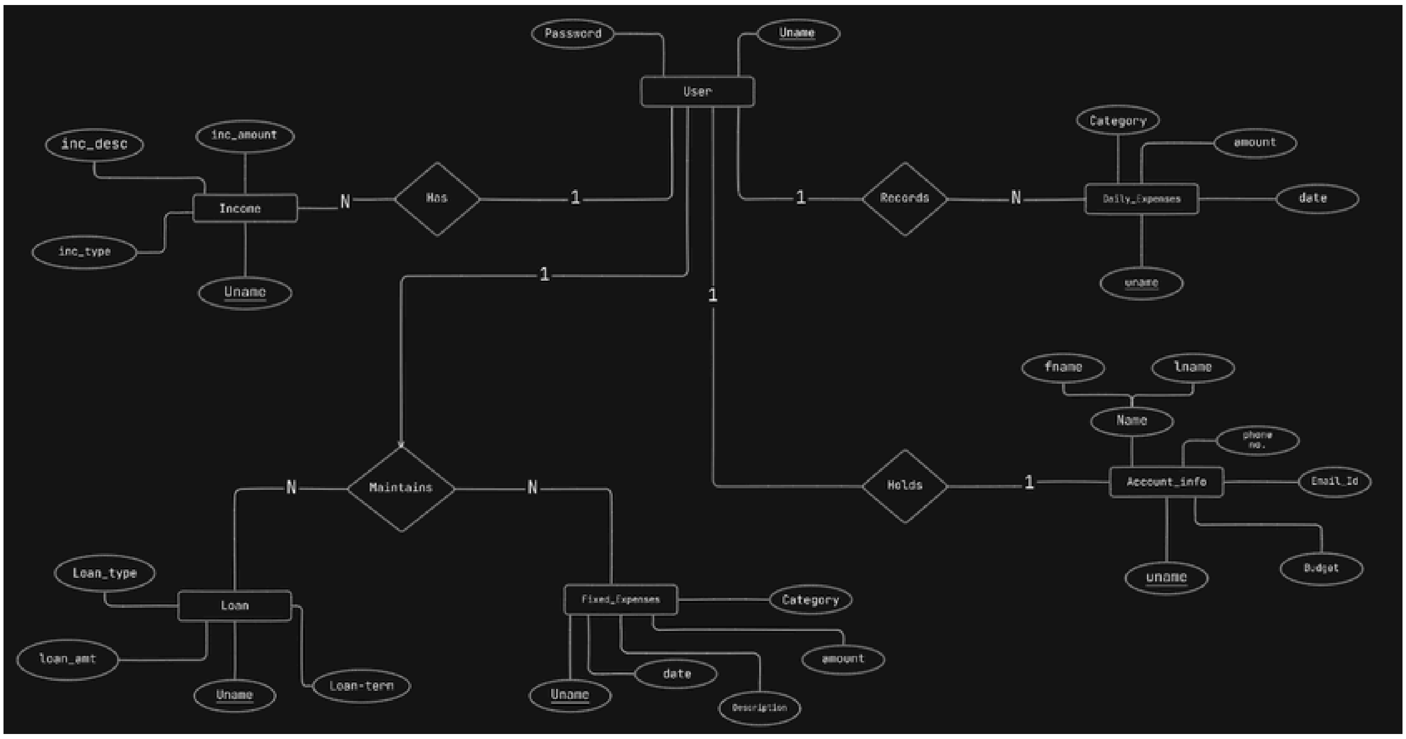
- Expense Tracking: The expenditures of users will be taken as input and categorized into the type of expenses and it will be tracked against the set budget to help users stay within their financial limits.

- Upcoming bill payments will be displayed helping users avoid late fees and will be automatically cut from the budget on the due date.

- The database will be designed to handle structured financial data, including tables for fixed and variable expenses, , budgets, loans and user information.

- The database can scale to accommodate a growing number of users and increasing volumes of financial data.

ER-MODEL



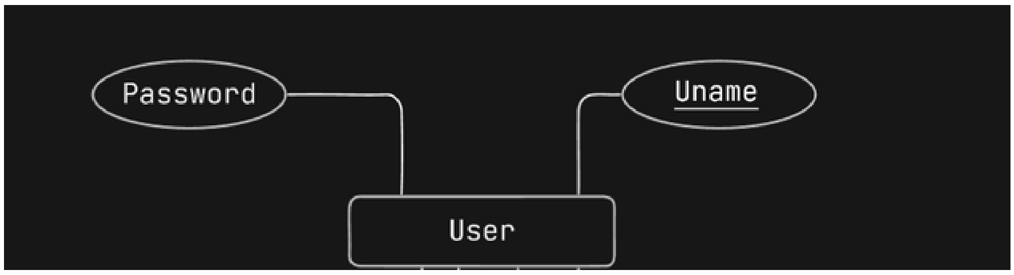
Entities present:

- User(uname, password)
- Account_info(uname,fname,lname, email, budget ,savings, phone_no.)
- Daily_Expenses(uname, category, amount, description date)
- Fixed_Expenses(uname, category, desc, amount, start_date, type)
- Loan(uname, Loan_type, Loan_amount, loan_term, interest_rate)
- Income(uname, inc_type, inc_desc, inc_amount)

Relationships:

- Many variable expenses(daily expenses) of a single user is recorded thus the cardinality ratio between user and varied_expenses is 1:N.
- A single user maintains multiple fixed expenses(phone bill, rent) so the cardinality ratio between user and fixed_expenses is 1:N.
- A user can have n number of loans thus the the cardinality ratio between user and loan is 1:N.
- A user can have multiple income sources therefore the cardinality ratio between user and income is 1:N.

RELATIONAL MODEL CONVERSION

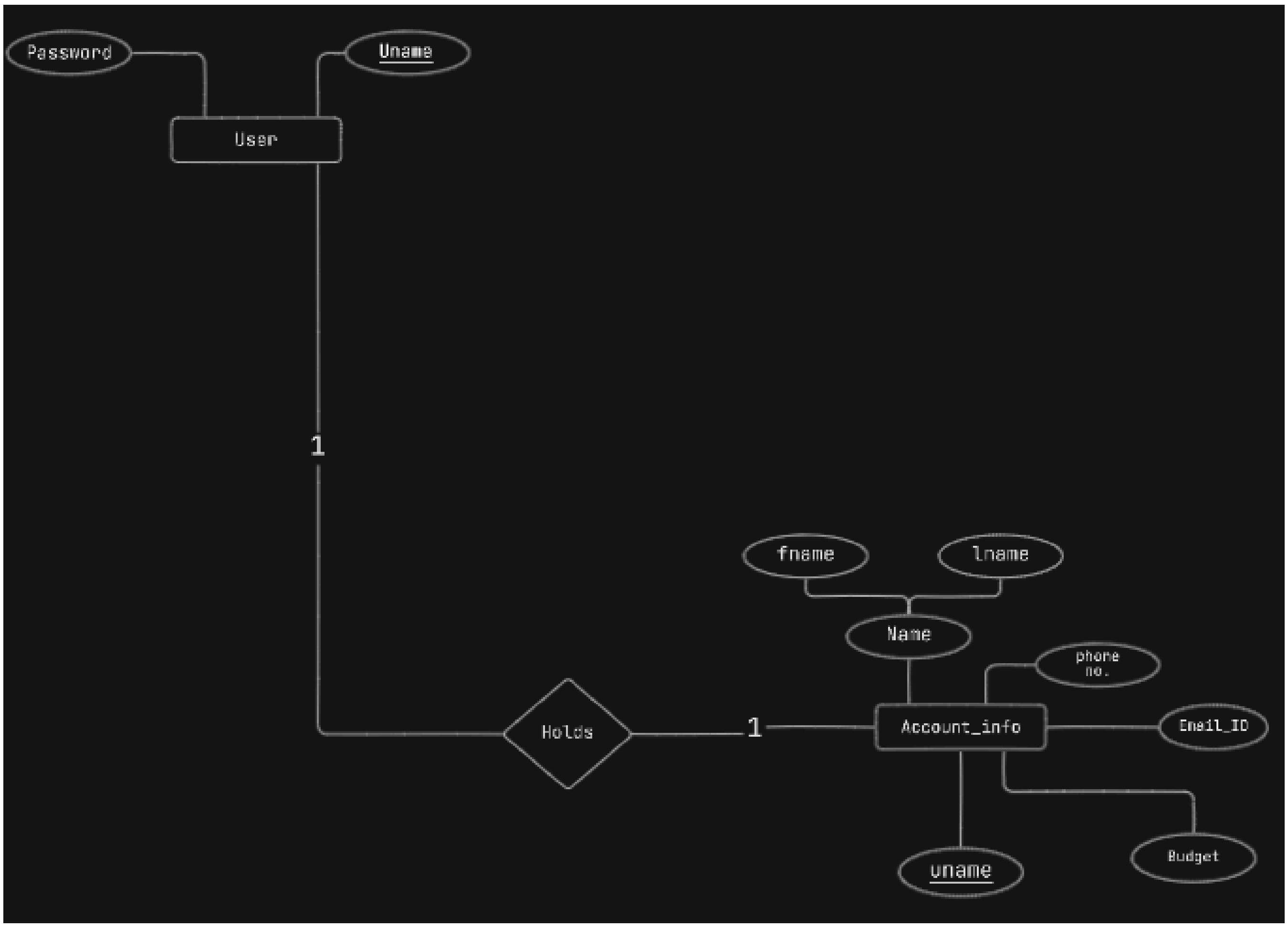


User

<u>**Uname**</u>	**Password**

Primary key: Uname

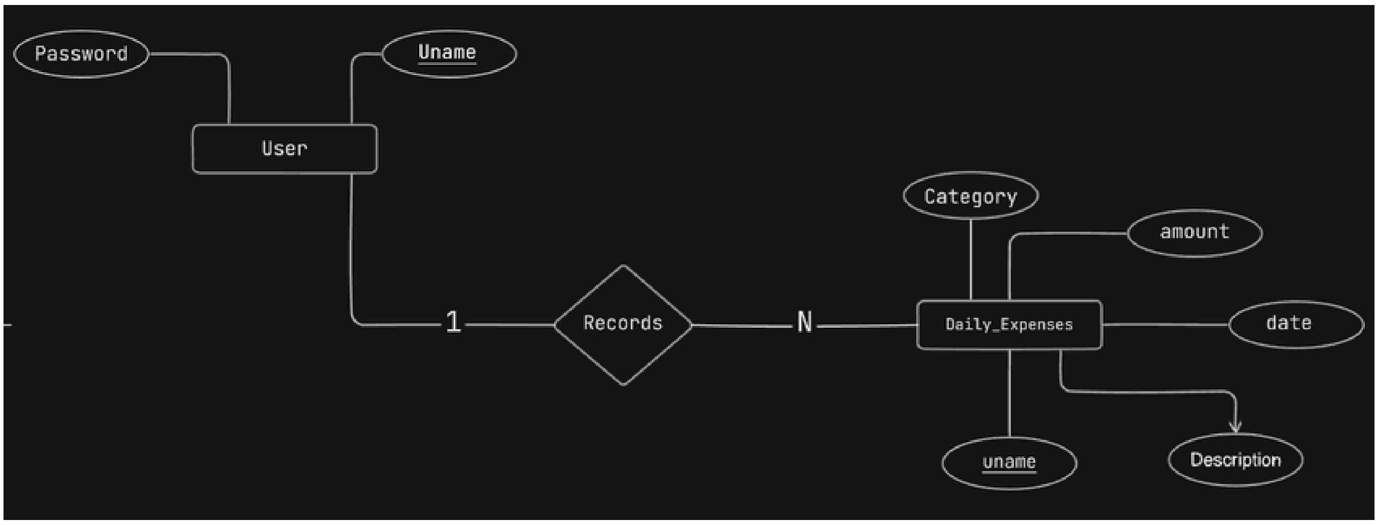
RELATIONAL MODEL CONVERSION



Account_info

Uname	Name	Email	Phone_no	Budget	Savings

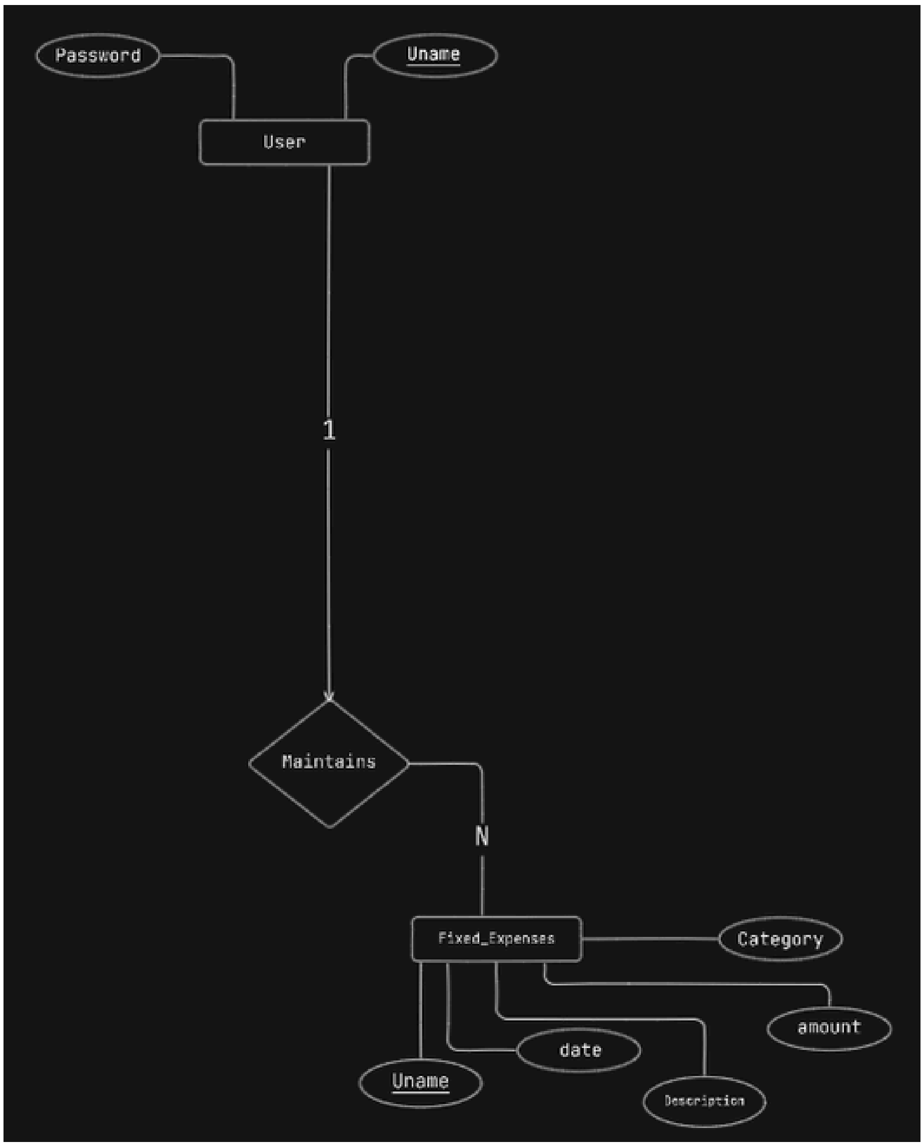
RELATIONAL MODEL CONVERSION



Record_Daily_ expenses

Uname	desc	amount	date	category

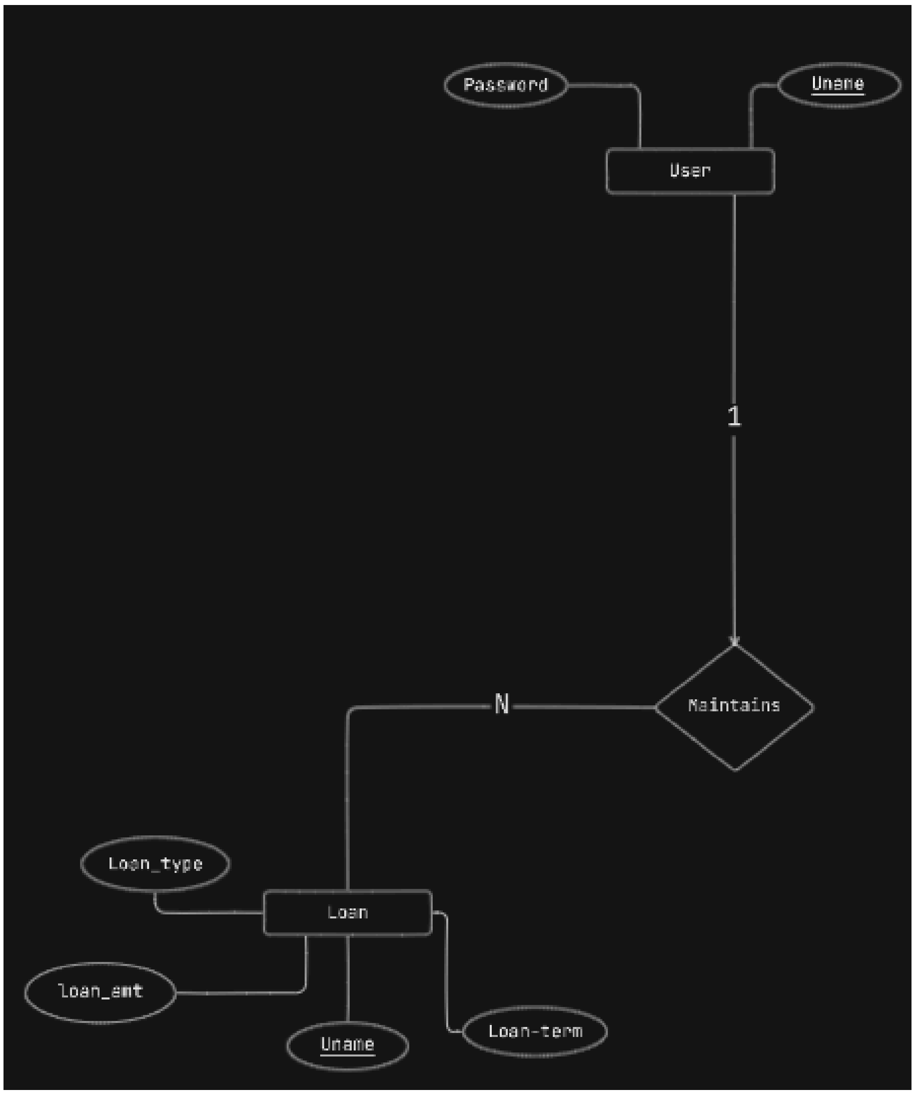
RELATIONAL MODEL CONVERSION



Maintain_Fixed_expenses

Uname	desc	amount	date	category

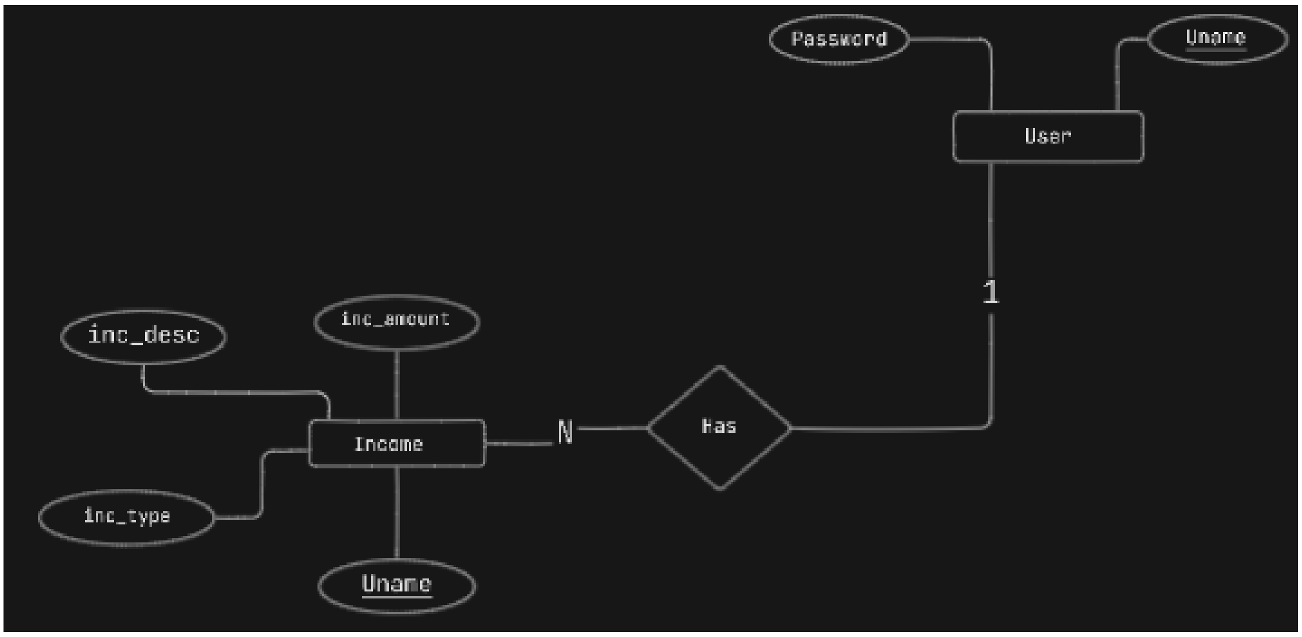
RELATIONAL MODEL CONVERSION



Maintain_Loan

Uname	lona_amt.	loan_type	loan_term

RELATIONAL MODEL CONVERSION



Has_Income

Uname	<u>inc_desc</u>	inc_type	inc_amount

TECH STACK

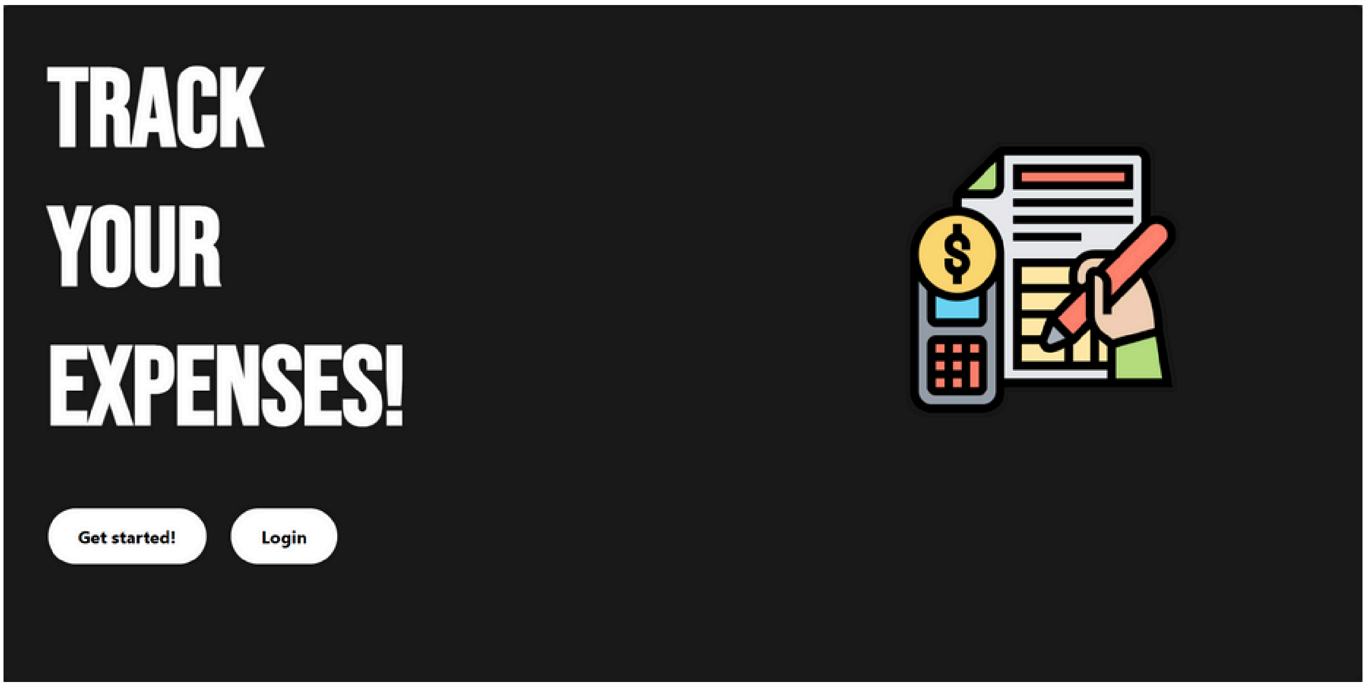
- **Python**: Python is the primary programming language used for server-side logic in this Flask application. It handles routing, form submissions, database interactions, and other backend operations.

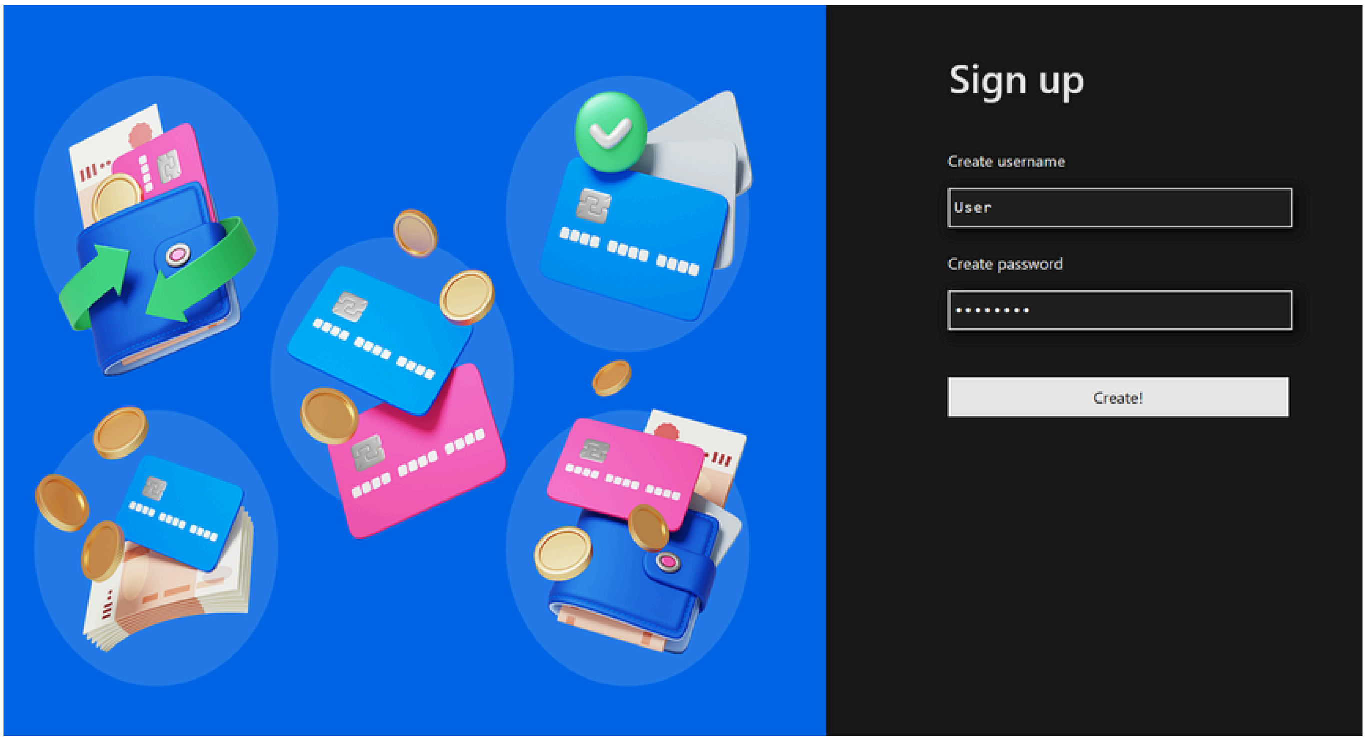
- **Flask**: Flask is a micro web framework for Python used to develop web applications. It provides tools, libraries, and technologies for building web applications.

- **MySQL Database**: MySQL is an open-source relational database management system. It is being used here to store user data such as account information, expenses, income, and loans

- **HTML Templates**: The application utilizes HTML templates for rendering the front-end views. Flask's render_template function is used to render these templates, allowing dynamic generation of HTML content.

OUTPUT


TRACK YOUR EXPENSES!


Sign up

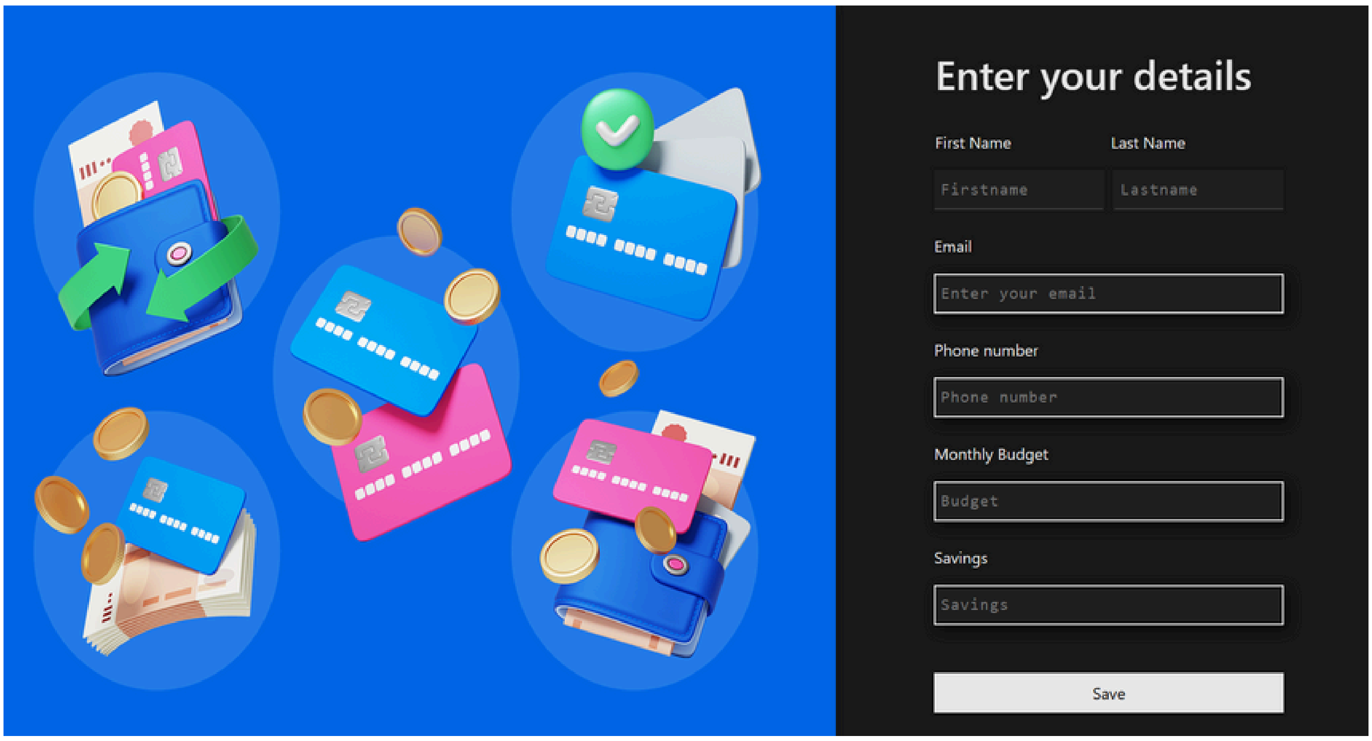
Create username

User

Create password

....

OUTPUT



Enter your details

First Name

Last Name

Firstname

Lastname

Email

Enter your email

Phone number

Phone number

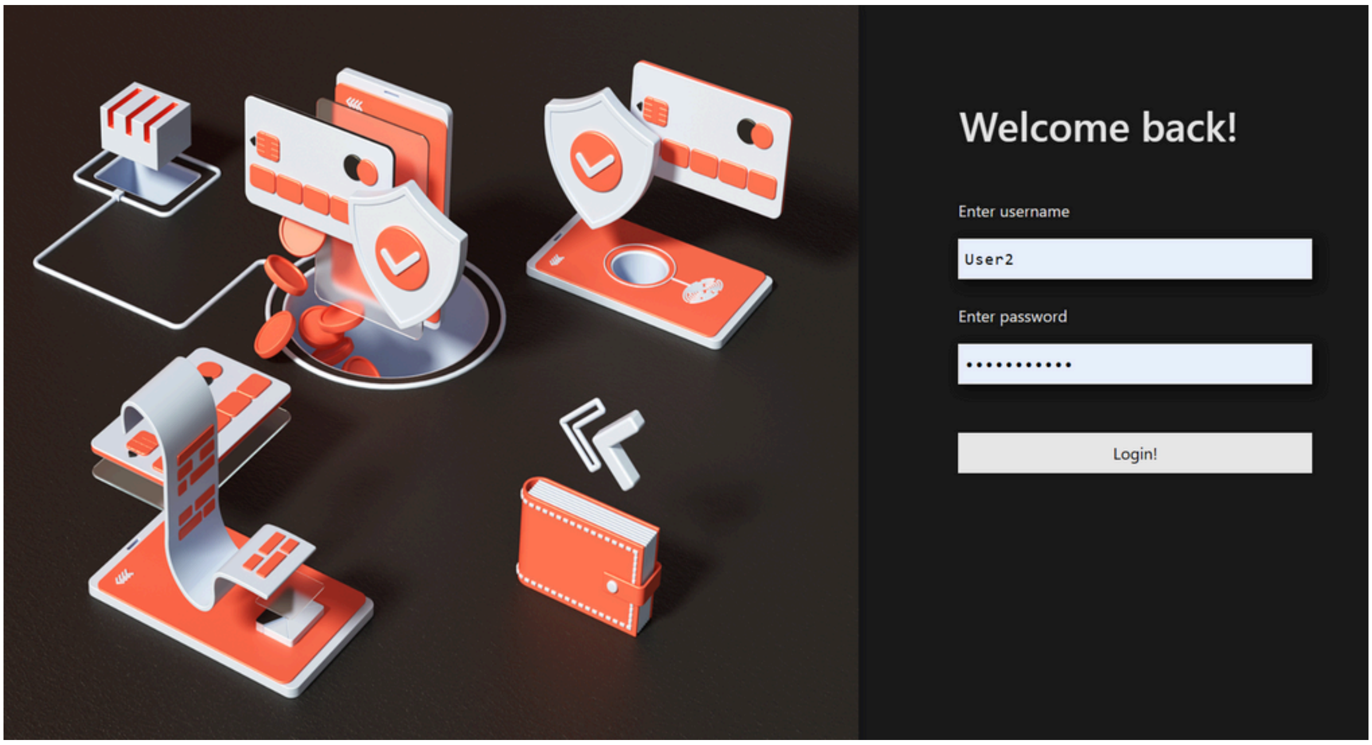
Monthly Budget

Budget

Savings

Savings

Save



Welcome back!

Enter username

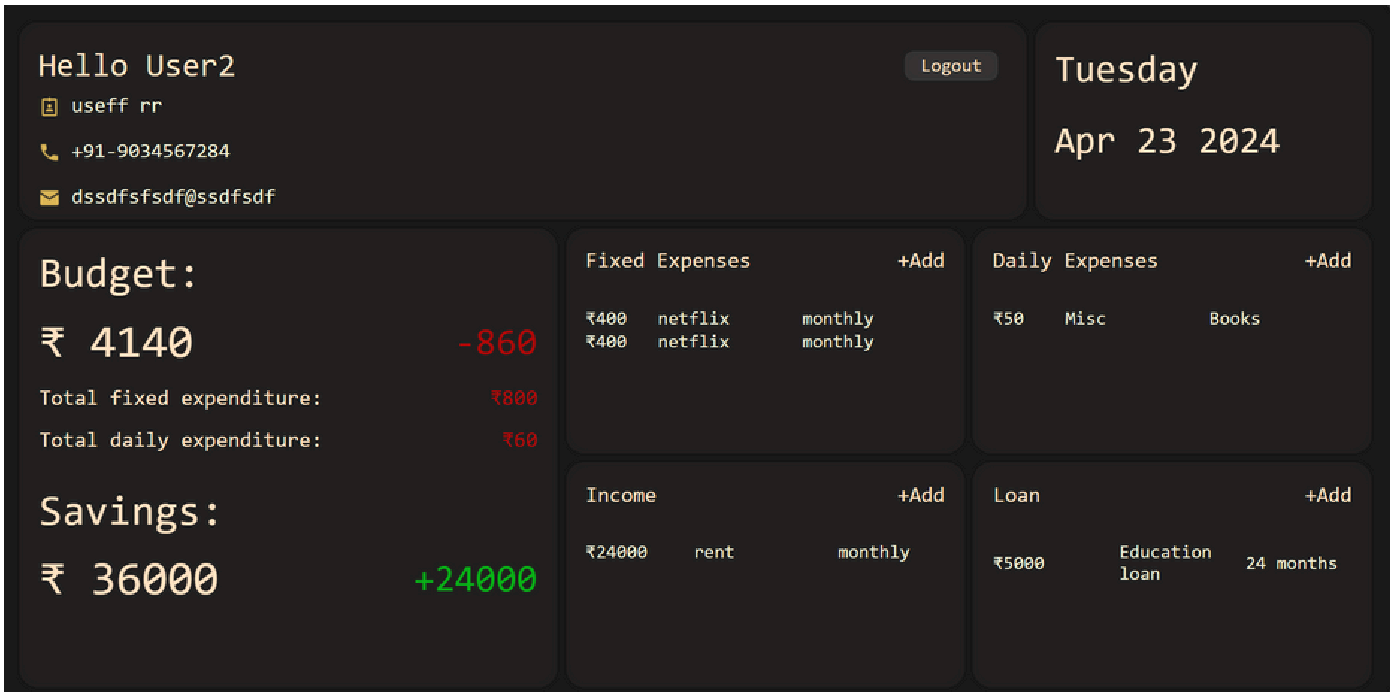
User2

Enter password

•••••••••••

Login!

OUTPUT



Hello User2 Logout
useff rr
+91-9034567284
dssdfsfsdf@ssdfsdf

Tuesday
Apr 23 2024

Budget:
₹ 4140 -860
Total fixed expenditure: ₹800
Total daily expenditure: ₹60

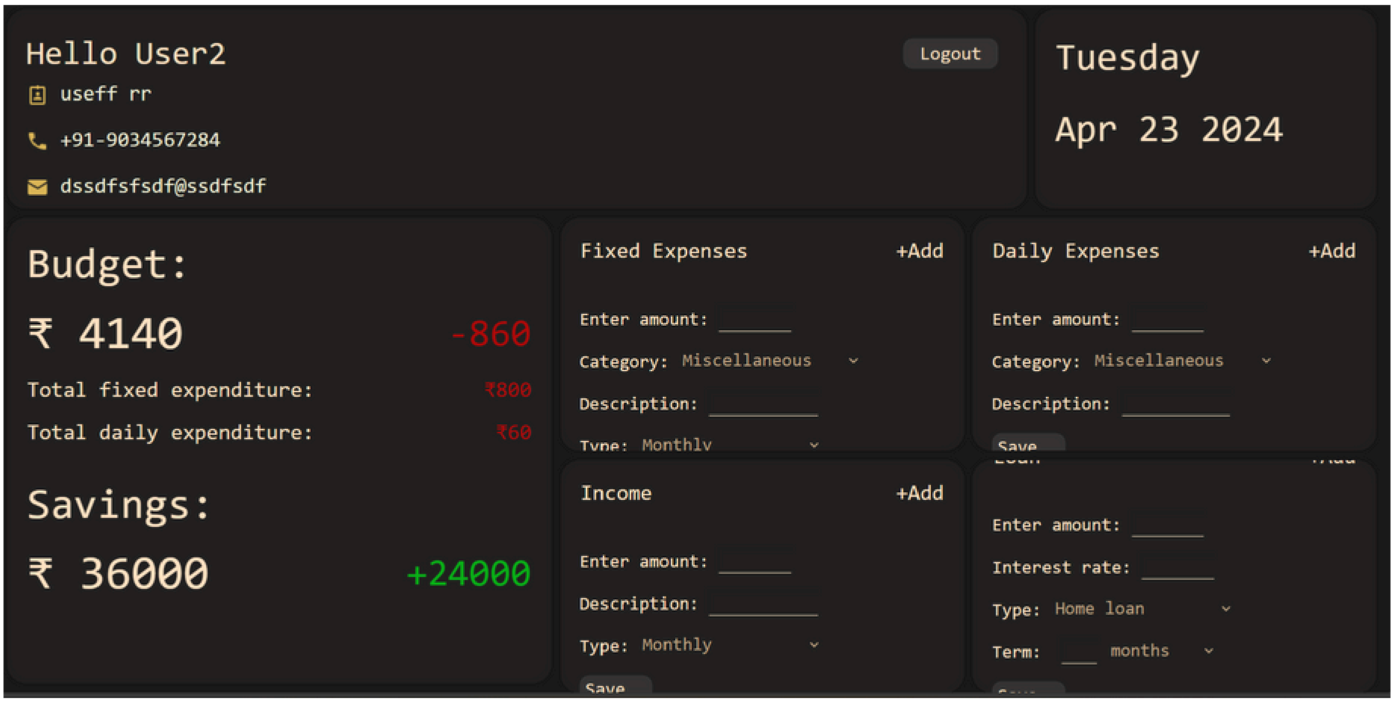
Savings:
₹ 36000 +24000

Fixed Expenses +Add
```
₹400   netflix    monthly
₹400   netflix    monthly
```

Daily Expenses +Add
```
₹50    Misc       Books
```

Income +Add
```
₹24000   rent     monthly
```

Loan +Add
```
₹5000    Education loan   24 months
```



Hello User2 Logout
useff rr
+91-9034567284
dssdfsfsdf@ssdfsdf

Tuesday
Apr 23 2024

Budget:
₹ 4140 -860
Total fixed expenditure: ₹800
Total daily expenditure: ₹60

Savings:
₹ 36000 +24000

Fixed Expenses +Add
Enter amount: ______
Category: Miscellaneous
Description: ________
Type: Monthly

Daily Expenses +Add
Enter amount: ______
Category: Miscellaneous
Description: ________
Save

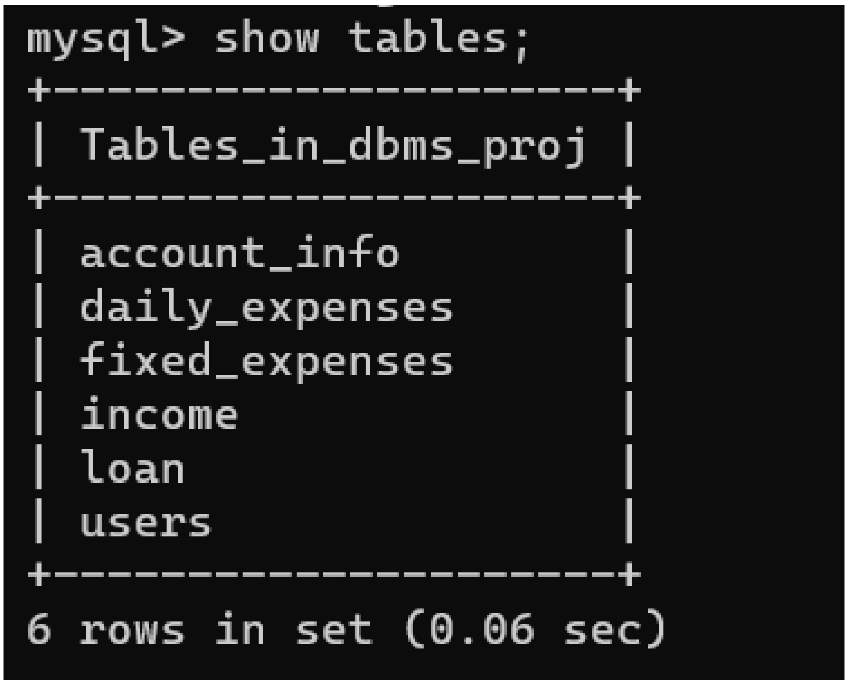
Income +Add
Enter amount: ______
Description: ________
Type: Monthly
Save

Enter amount: ______
Interest rate: ______
Type: Home loan
Term: ___ months

MYSQL TABLES

Tables Created:

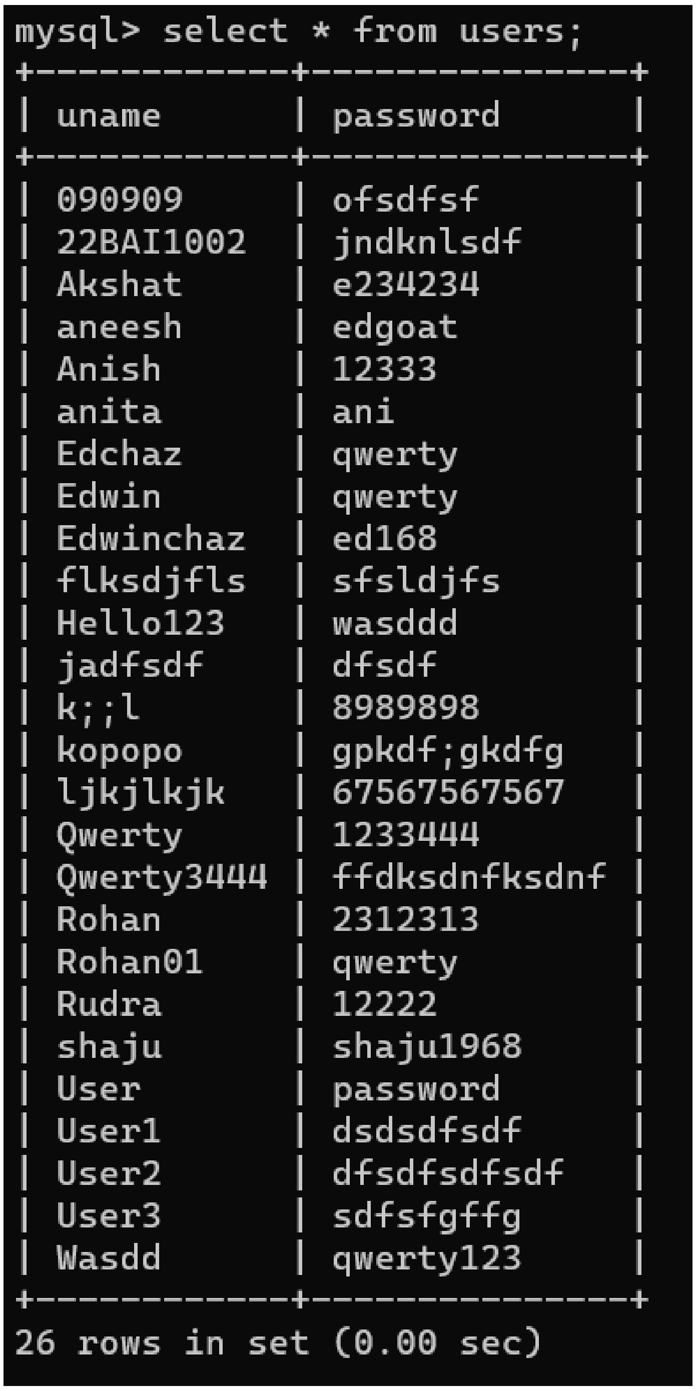
- User(uname, password)
- Account_info(uname,fname,lname, email, budget ,savings, phone_no.)
- Daily_Expenses(uname, category, amount, description date)
- Fixed_Expenses(uname, category, desc, amount, start_date, type)
- Loan(uname, Loan_type, Loan_amount, loan_term, interest_rate)
- Income(uname, inc_type, inc_desc, inc_amount)



```
mysql> show tables;
+---------------------+
| Tables_in_dbms_proj |
+---------------------+
| account_info        |
| daily_expenses      |
| fixed_expenses      |
| income              |
| loan                |
| users               |
+---------------------+
6 rows in set (0.06 sec)
```

MYSQL TABLES

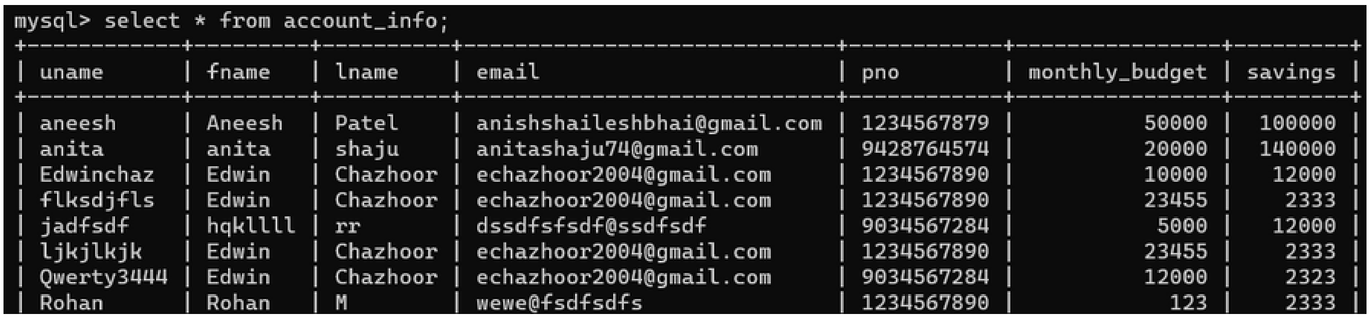
User(uname, password)



```
mysql> select * from users;
+------------+---------------+
| uname      | password      |
+------------+---------------+
| 090909     | ofsdfsf       |
| 22BAI1002  | jndknlsdf     |
| Akshat     | e234234       |
| aneesh     | edgoat        |
| Anish      | 12333         |
| anita      | ani           |
| Edchaz     | qwerty        |
| Edwin      | qwerty        |
| Edwinchaz  | ed168         |
| flksdjfls  | sfsldjfs      |
| Hello123   | wasddd        |
| jadfsdf    | dfsdf         |
| k;;l       | 8989898       |
| kopopo     | gpkdf;gkdfg   |
| ljkjlkjk   | 67567567567   |
| Qwerty     | 1233444       |
| Qwerty3444 | ffdksdnfksdnf |
| Rohan      | 2312313       |
| Rohan01    | qwerty        |
| Rudra      | 12222         |
| shaju      | shaju1968     |
| User       | password      |
| User1      | dsdsdfsdf     |
| User2      | dfsdfsdfsdf   |
| User3      | sdfsfgffg     |
| Wasdd      | qwerty123     |
+------------+---------------+
26 rows in set (0.00 sec)
```

MYSQL TABLES

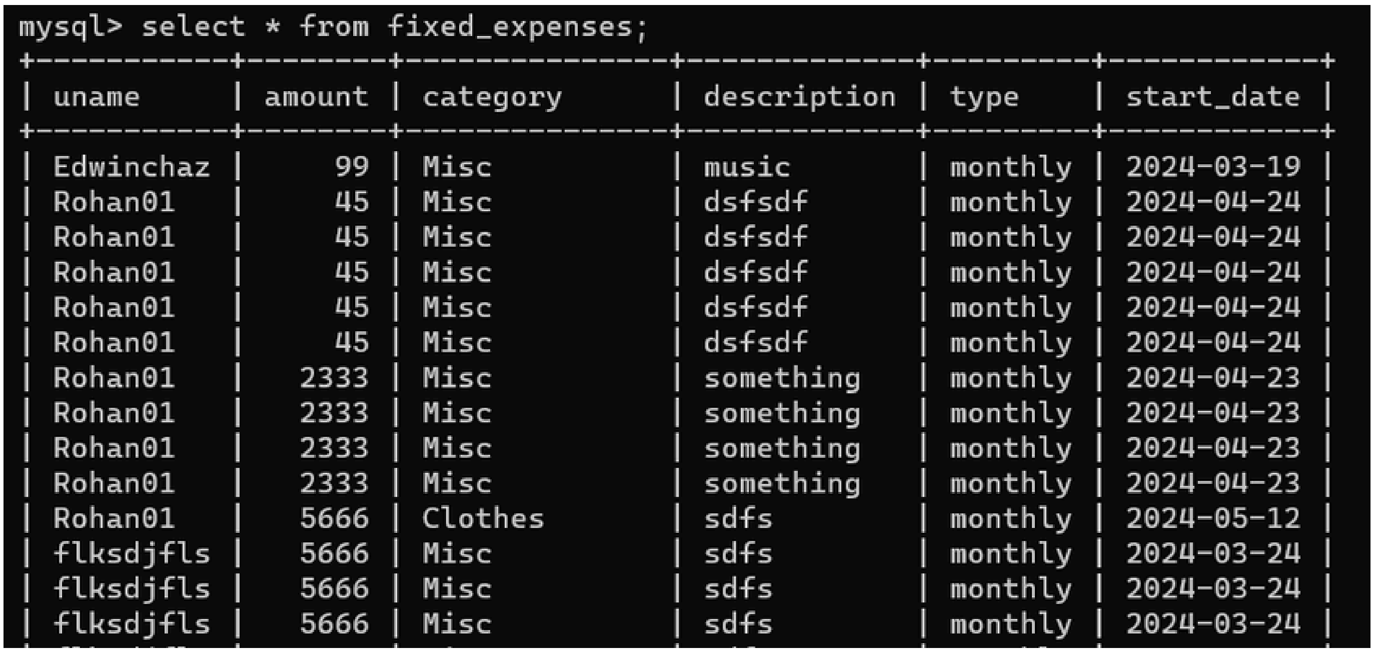
Account_info(uname,fname,lname, email, budget , savings, phone_no.)



```
mysql> select * from account_info;
+------------+---------+----------+-----------------------------+------------+----------------+---------+
| uname      | fname   | lname    | email                       | pno        | monthly_budget | savings |
+------------+---------+----------+-----------------------------+------------+----------------+---------+
| aneesh     | Aneesh  | Patel    | anishshaileshbhai@gmail.com | 1234567879 |          50000 |  100000 |
| anita      | anita   | shaju    | anitashaju74@gmail.com      | 9428764574 |          20000 |  140000 |
| Edwinchaz  | Edwin   | Chazhoor | echazhoor2004@gmail.com     | 1234567890 |          10000 |   12000 |
| flksdjfls  | Edwin   | Chazhoor | echazhoor2004@gmail.com     | 1234567890 |          23455 |    2333 |
| jadfsdf    | hqkllll | rr       | dssdfsfsdf@ssdfsdf          | 9034567284 |           5000 |   12000 |
| ljkjlkjk   | Edwin   | Chazhoor | echazhoor2004@gmail.com     | 1234567890 |          23455 |    2333 |
| Qwerty3444 | Edwin   | Chazhoor | echazhoor2004@gmail.com     | 9034567284 |          12000 |    2323 |
| Rohan      | Rohan   | M        | wewe@fsdfsdfs               | 1234567890 |            123 |    2333 |
```

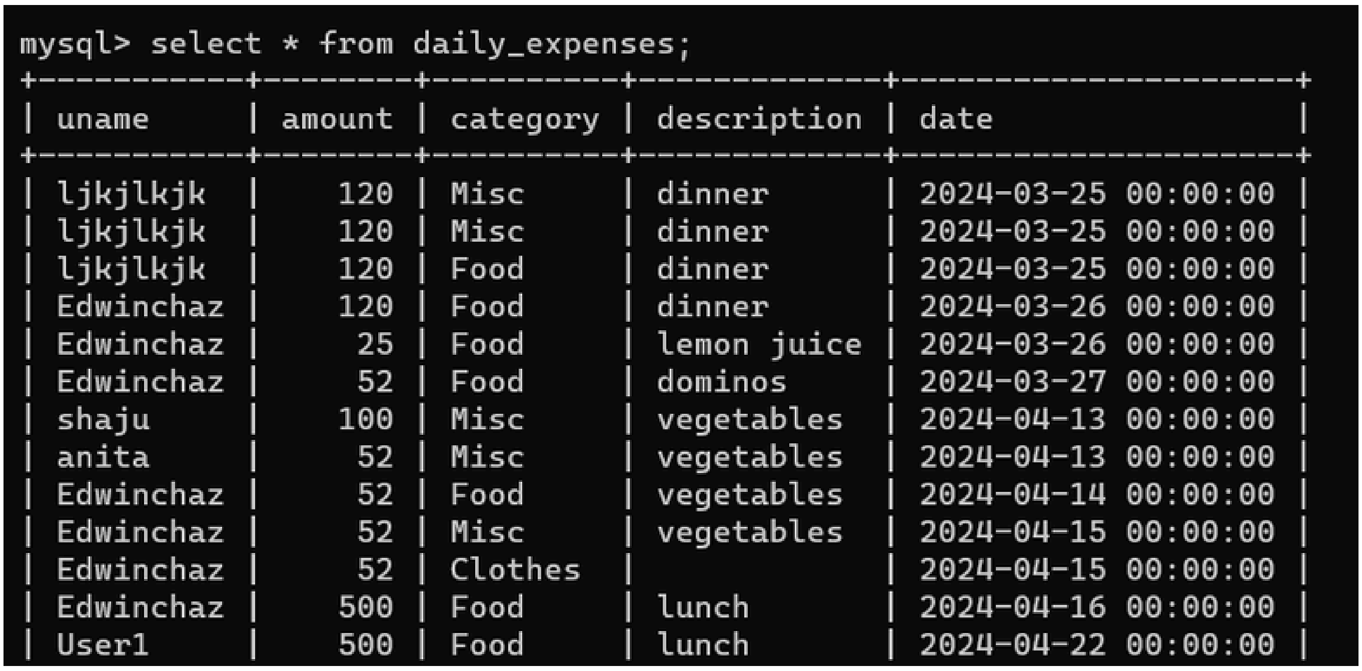
MYSQL TABLES

Fixed_Expenses(uname, category, desc, amount, start_date, type)


```
mysql> select * from fixed_expenses;
+-----------+--------+---------------+-------------+---------+------------+
| uname     | amount | category      | description | type    | start_date |
+-----------+--------+---------------+-------------+---------+------------+
| Edwinchaz |     99 | Misc          | music       | monthly | 2024-03-19 |
| Rohan01   |     45 | Misc          | dsfsdf      | monthly | 2024-04-24 |
| Rohan01   |     45 | Misc          | dsfsdf      | monthly | 2024-04-24 |
| Rohan01   |     45 | Misc          | dsfsdf      | monthly | 2024-04-24 |
| Rohan01   |     45 | Misc          | dsfsdf      | monthly | 2024-04-24 |
| Rohan01   |     45 | Misc          | dsfsdf      | monthly | 2024-04-24 |
| Rohan01   |   2333 | Misc          | something   | monthly | 2024-04-23 |
| Rohan01   |   2333 | Misc          | something   | monthly | 2024-04-23 |
| Rohan01   |   2333 | Misc          | something   | monthly | 2024-04-23 |
| Rohan01   |   2333 | Misc          | something   | monthly | 2024-04-23 |
| Rohan01   |   5666 | Clothes       | sdfs        | monthly | 2024-05-12 |
| flksdjfls |   5666 | Misc          | sdfs        | monthly | 2024-03-24 |
| flksdjfls |   5666 | Misc          | sdfs        | monthly | 2024-03-24 |
| flksdjfls |   5666 | Misc          | sdfs        | monthly | 2024-03-24 |
```

MYSQL TABLES

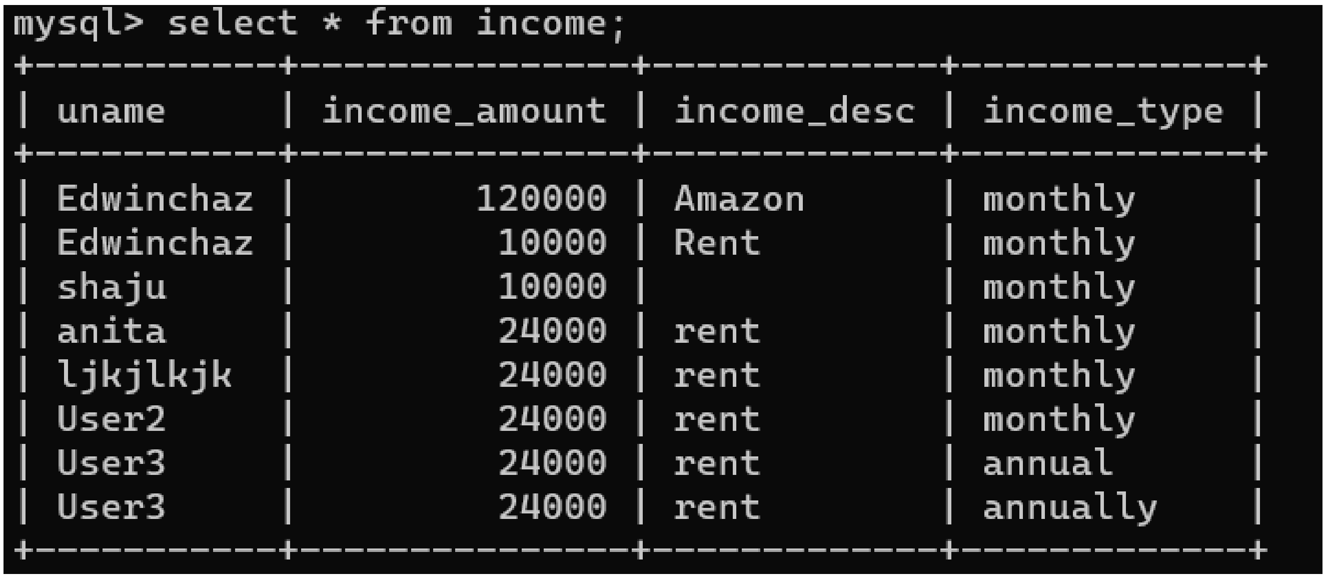
Daily__Expenses(uname, category, amount, description date)



```
mysql> select * from daily_expenses;
+-----------+--------+----------+-------------+---------------------+
| uname     | amount | category | description | date                |
+-----------+--------+----------+-------------+---------------------+
| ljkjlkjk  |    120 | Misc     | dinner      | 2024-03-25 00:00:00 |
| ljkjlkjk  |    120 | Misc     | dinner      | 2024-03-25 00:00:00 |
| ljkjlkjk  |    120 | Food     | dinner      | 2024-03-25 00:00:00 |
| Edwinchaz |    120 | Food     | dinner      | 2024-03-26 00:00:00 |
| Edwinchaz |     25 | Food     | lemon juice | 2024-03-26 00:00:00 |
| Edwinchaz |     52 | Food     | dominos     | 2024-03-27 00:00:00 |
| shaju     |    100 | Misc     | vegetables  | 2024-04-13 00:00:00 |
| anita     |     52 | Misc     | vegetables  | 2024-04-13 00:00:00 |
| Edwinchaz |     52 | Food     | vegetables  | 2024-04-14 00:00:00 |
| Edwinchaz |     52 | Misc     | vegetables  | 2024-04-15 00:00:00 |
| Edwinchaz |     52 | Clothes  |             | 2024-04-15 00:00:00 |
| Edwinchaz |    500 | Food     | lunch       | 2024-04-16 00:00:00 |
| User1     |    500 | Food     | lunch       | 2024-04-22 00:00:00 |
```

MYSQL TABLES

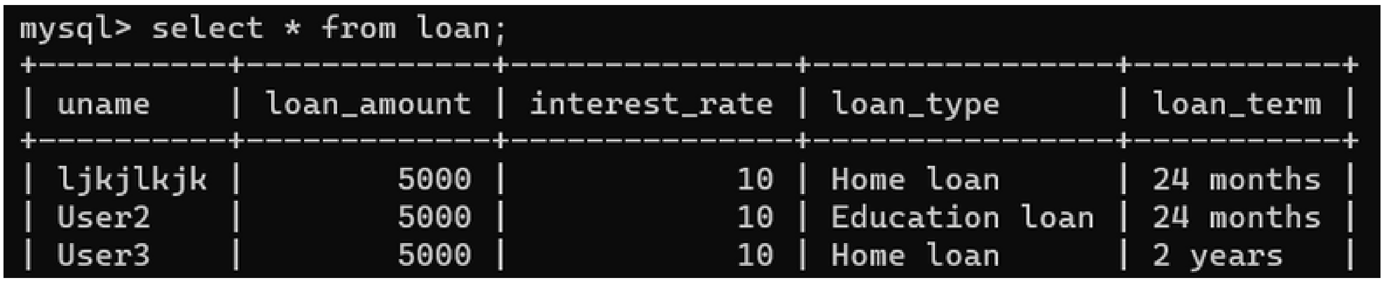
Income(uname, inc_type, inc_desc, inc_amount)



```
mysql> select * from income;
+-----------+---------------+-------------+-------------+
| uname     | income_amount | income_desc | income_type |
+-----------+---------------+-------------+-------------+
| Edwinchaz |        120000 | Amazon      | monthly     |
| Edwinchaz |         10000 | Rent        | monthly     |
| shaju     |         10000 |             | monthly     |
| anita     |         24000 | rent        | monthly     |
| ljkjlkjk  |         24000 | rent        | monthly     |
| User2     |         24000 | rent        | monthly     |
| User3     |         24000 | rent        | annual      |
| User3     |         24000 | rent        | annually    |
+-----------+---------------+-------------+-------------+
```

MYSQL TABLES

Loan(uname, Loan_type, Loan_amount, loan_term, interest_rate)



```
mysql> select * from loan;
+----------+-------------+---------------+----------------+-----------+
| uname    | loan_amount | interest_rate | loan_type      | loan_term |
+----------+-------------+---------------+----------------+-----------+
| ljkjlkjk |        5000 |            10 | Home loan      | 24 months |
| User2    |        5000 |            10 | Education loan | 24 months |
| User3    |        5000 |            10 | Home loan      | 2 years   |
```

CODE - MYSQL CONNECTION USING FLASK

```
from flask import Flask, render_template, request, redirect, session,url_for
import mysql.connector as ms

conn = ms.connect(host="localhost", port=3306, user="root",
passwd="Edchaz168", database="dbms_proj")

if conn.is_connected():
  print("Hi")

mc=conn.cursor()

app = Flask(__name__)

@app.route('/')
def main_page():
 return render_template("start_page.html")

@app.route('/signup')
def signup_page():
 return render_template("Signup.html")

@app.route('/fill_details', methods=['POST'])
def enter_details():
 if request.method == 'POST':
  global uname
  uname=request.form['username']
  passwd=request.form['password']
  mc.execute("select uname from users where uname=%s",(uname,))
  result=mc.fetchall()
  conn.commit()
  if result!=[]:
   err='Username already exists'
   return render_template("Signup.html",err=err)
  else:
   mc.execute("insert into users values(%s,%s)",(uname,passwd))
   conn.commit()
   return render_template("fill_details.html",result=result)
```

```
@app.route('/start/login')
def login_page():
 return render_template("login.html")
@app.route('/login', methods=['POST'])
def success_page():
 if request.method == 'POST':
  fname=request.form['firstname']
  lname=request.form['lastname']
  email=request.form['email']
  phone=request.form['pnumber']
  budget=request.form['budget']
  savings=request.form['savings']
  print(uname,fname,lname,email,phone,budget,savings)
  budget=int(budget)
  savings=int(savings)
  mc.execute("insert into account_info values(%s,%s,%s,%s,%s,%s,%s)",
(uname,fname,lname,email,phone,budget,savings))
  conn.commit()
  return render_template("login.html")
 else:
  return render_template("login.html")


@app.route('/dashboard', methods=['POST'])
def dashboard_page():
 if request.method == 'POST':
  global uname
  uname=request.form['username']
  passwd=request.form['password']
  mc.execute("select * from users where uname=%s and password=%s",
(uname,passwd))
  result=mc.fetchall()
  conn.commit
```

```
if result!=[]:

mc.execute("select * from account_info where uname=%s",(uname,))
result=mc.fetchall()
conn.commit()
for result in result:
fname=result[1]
lname=result[2]
email=result[3]
pno=result[4]
budget=result[5]
savings=result[6]


mc.execute("select * from fixed_expenses where uname=%s order by
start_date desc",(uname,))
result_fixed=mc.fetchall()
conn.commit()
mc.execute("select * from daily_expenses where uname=%s and
date(date)=curdate()",(uname,))
result_today=mc.fetchall()
conn.commit()
mc.execute("select * from income where uname=%s",(uname,))
result_income=mc.fetchall()
conn.commit()
mc.execute("select * from loan where uname=%s",(uname,))
result_loan=mc.fetchall()
conn.commit()

mc.execute("select * from daily_expenses where uname=%s and
extract(month from date(date))=extract(month from curdate())",(uname,))
result_daily=mc.fetchall()
conn.commit()
```

```
  if result_income!=[]:
   # Calculate total income
   mc.execute("SELECT SUM(income_amount) FROM income WHERE
uname=%s", (uname,))
   total_inc = mc.fetchall()
   conn.commit()

   total_income=total_inc[0][0]
   savings += total_income
  else:
   total_income=0

  if result_fixed!=[] and result_daily==[]:
   mc.execute("select sum(amount) from fixed_expenses where
uname=%s",(uname,))
   l=mc.fetchall()
   print(l)
   fixed_expense=l[0][0]
   total_expense=l[0][0]
   conn.commit()
   budget=budget-total_expense
   return render_template("dashboard.html",total_expense=-
total_expense,fixed_expense=fixed_expense,daily_expense=0,total_inco
me=total_income,result_loan=result_loan,result_income=result_income
,result_fixed=result_fixed,uname=uname,fname=fname,lname=lname,em
ail=email,pno=pno,budget=budget,savings=savings)
  elif result_fixed==[] and result_daily!=[]:
   mc.execute("select sum(amount) from daily_expenses where
uname=%s",(uname,))
   l=mc.fetchall()
   print(l)
   daily_expense=l[0][0]
   total_expense=l[0][0]
   conn.commit()
   budget=budget-total_expense
   return render_template("dashboard.html",total_expense=-
total_expense,fixed_expense=0,daily_expense=daily_expense,total_inco
me=total_income,result_loan=result_loan,result_income=result_income
,result_today=result_today,uname=uname,fname=fname,lname=lname,e
mail=email,pno=pno,budget=budget,savings=savings)
```

```
 elif result_fixed!=[] and result_daily!=[]:
 mc.execute("select sum(amount) from fixed_expenses where
uname=%s",(uname,))
 l=mc.fetchall()
 print(l)
 fixed_expense=l[0][0]
 conn.commit()
 mc.execute("select sum(amount) from daily_expenses where uname=%s",
(uname,))
 l1=mc.fetchall()
 print(l)
 daily_expense=l1[0][0]
 conn.commit()
 total_expense=fixed_expense + daily_expense
 budget=budget-total_expense
 return
render_template("dashboard.html",fixed_expense=fixed_expense,daily_
expense=daily_expense,total_expense=-
total_expense,result_today=result_today,total_income=total_income,res
ult_loan=result_loan,result_income=result_income,result_fixed=result_
fixed,result_daily=result_daily,uname=uname,fname=fname,lname=lnam
e,email=email,pno=pno,budget=budget,savings=savings)
 elif result_fixed==[] and result_daily==[] and result_income==[] and
result_loan==[]:
 return
render_template("dashboard.html",daily_expense=0,fixed_expense=0,un
ame=uname,total_income=0,fname=fname,lname=lname,email=email,pn
o=pno,budget=budget,savings=savings)
 else:
 err="Invalid username or password!"
 return render_template("login.html",err=err)
```

```
@app.route('/user_dashboard')
def user_dashboard_page():
  mc.execute("select * from account_info where uname=%s",(uname,))
  result=mc.fetchall()
  conn.commit()
  for result in result:
    fname=result[1]
    lname=result[2]
    email=result[3]
    pno=result[4]
    budget=result[5]
    savings=result[6]


  mc.execute("select * from fixed_expenses where uname=%s order by
start_date desc",(uname,))
  result_fixed=mc.fetchall()
  conn.commit()
  mc.execute("select * from daily_expenses where uname=%s and
date(date)=curdate()",(uname,))
  result_today=mc.fetchall()
  conn.commit()
  mc.execute("select * from income where uname=%s",(uname,))
  result_income=mc.fetchall()
  conn.commit()
  mc.execute("select * from loan where uname=%s",(uname,))
  result_loan=mc.fetchall()
  conn.commit()

  mc.execute("select * from daily_expenses where uname=%s",(uname,))
  result_daily=mc.fetchall()
  conn.commit()
  if result_income!=[]:
   # Calculate total income
   mc.execute("SELECT SUM(income_amount) FROM income WHERE 
uname=%s", (uname,))
   total_inc = mc.fetchall()
   conn.commit()

   total_income=total_inc[0][0]
   savings += total_income
  else:
   total_income=0


  if result_fixed!=[] and result_daily==[]:
   mc.execute("select sum(amount) from fixed_expenses where 
uname=%s",(uname,))
   l=mc.fetchall()
   fixed_expense=l[0][0]
   total_expense=l[0][0]
   conn.commit()
   budget=budget-total_expense
   return render_template("dashboard.html",total_expense=-
total_expense,daily_expense=0,fixed_expense=fixed_expense,total_inco
me=total_income,result_loan=result_loan,result_income=result_income
,result_fixed=result_fixed,uname=uname,fname=fname,lname=lname,em
ail=email,pno=pno,budget=budget,savings=savings)
  elif result_fixed==[] and result_daily!=[]:
   mc.execute("select sum(amount) from daily_expenses where 
uname=%s",(uname,))
   l=mc.fetchall()
   print(l)
   daily_expense=l[0][0]
   total_expense=l[0][0]
   conn.commit()
   budget=budget-total_expense
   return render_template("dashboard.html",total_expense=-
total_expense,fixed_expense=0,daily_expense=daily_expense,total_inco
me=total_income,result_loan=result_loan,result_income=result_income
,result_today=result_today,uname=uname,fname=fname,lname=lname,e
mail=email,pno=pno,budget=budget,savings=savings)
```

```
  elif result_fixed!=[] and result_daily!=[]:
   mc.execute("select sum(amount) from fixed_expenses where
uname=%s",(uname,))
   l=mc.fetchall()
   print(l)
   fixed_expense=l[0][0]
   conn.commit()
   mc.execute("select sum(amount) from daily_expenses where
uname=%s",(uname,))
   l1=mc.fetchall()
   print(l)
   daily_expense=l1[0][0]
   conn.commit()
   total_expense=fixed_expense + daily_expense
   budget=budget-total_expense
   return
render_template("dashboard.html",fixed_expense=fixed_expense,dail
y_expense=daily_expense,total_expense=-
total_expense,result_today=result_today,total_income=total_income,
result_loan=result_loan,result_income=result_income,result_fixed=r
esult_fixed,result_daily=result_daily,uname=uname,fname=fname,lna
me=lname,email=email,pno=pno,budget=budget,savings=savings)
  elif result_fixed==[] and result_daily==[] and result_income==[] and
result_loan==[]:
   return
render_template("dashboard.html",daily_expense=0,fixed_expense=0,
total_income=0,uname=uname,fname=fname,lname=lname,email=ema
il,pno=pno,budget=budget,savings=savings)
```

```
@app.route('/fixed_expenses',methods=['POST'])
def fixed_expenses():
 if request.method == 'POST':
  amt=request.form['fixed_amount']
  amt=int(amt)
  cat=request.form['category']
  des=request.form['desc']
  ma=request.form['m/a']
  sdate=request.form['start_date']
  mc.execute("insert into fixed_expenses
values(%s,%s,%s,%s,%s,%s)",(uname,amt,cat,des,ma,sdate))
  conn.commit()
  return redirect(url_for('user_dashboard_page'))

@app.route('/daily_expenses',methods=['POST'])
def daily_expenses():
 if request.method == 'POST':
  amt=request.form['daily_amount']
  amt=int(amt)
  cat=request.form['daily_category']
  des=request.form['daily_desc']
  mc.execute("insert into daily_expenses
values(%s,%s,%s,%s,curdate())",(uname,amt,cat,des))
  conn.commit()
  return redirect(url_for('user_dashboard_page'))
```

```
@app.route('/income',methods=['POST'])
def income():
 if request.method=='POST':
  amt=request.form['inc_amount']
  amt=int(amt)
  des=request.form['inc_desc']
  type=request.form['inc_type']
  mc.execute("insert into income values(%s,%s,%s,%s)",
(uname,amt,des,type))
  conn.commit()
  return redirect(url_for('user_dashboard_page'))

@app.route('/loan',methods=['POST'])
def loan():
 if request.method=='POST':
  amt=int(request.form['loan_amount'])
  rate=int(request.form['loan_rate'])
  type=request.form['loan_type']
  term1=request.form['loan_term']
  term2=request.form['loan_m/y']
  term=term1+' '+term2
  mc.execute("insert into loan values(%s,%s,%s,%s,%s)",
(uname,amt,rate,type,term))
  conn.commit()
  return redirect(url_for('user_dashboard_page'))



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

FUTURE SCOPE

We can incorporate features like:

- Financial Goal Setting: Allow users to set specific financial goals (e.g., saving for a down payment, vacation) and track progress towards them.
- Goal-Based Saving: Create features that allow users to set up automatic transfers towards specific saving goals
- Multi-Factor Authentication: Implement multi-factor authentication for added security and user protection.
- NoSQL Databases: Depending on your project's growth and data structure complexity, consider NoSQL databases that offer greater flexibility and scalability compared to traditional relational databases.