

Project Development phase - Sprint 2

Team ID	PNT2022TMID04422
Project Name	Personal Expense Tracker

Dashboard

Add expenses

Personal Expense Tracker | Home | Add | History | Charts | Report | Logout

Add Expense

Date: 11/18/2022

Time: 10:35 AM

Expense name: personal

Expense Amount: 200

Payment type: cash

Category: food

Add

History

127.0.0.1:5555/expense-tracker

Navigation

Document

Personal Expense Tracker

Home

Add

History

Charts

Report

Logout

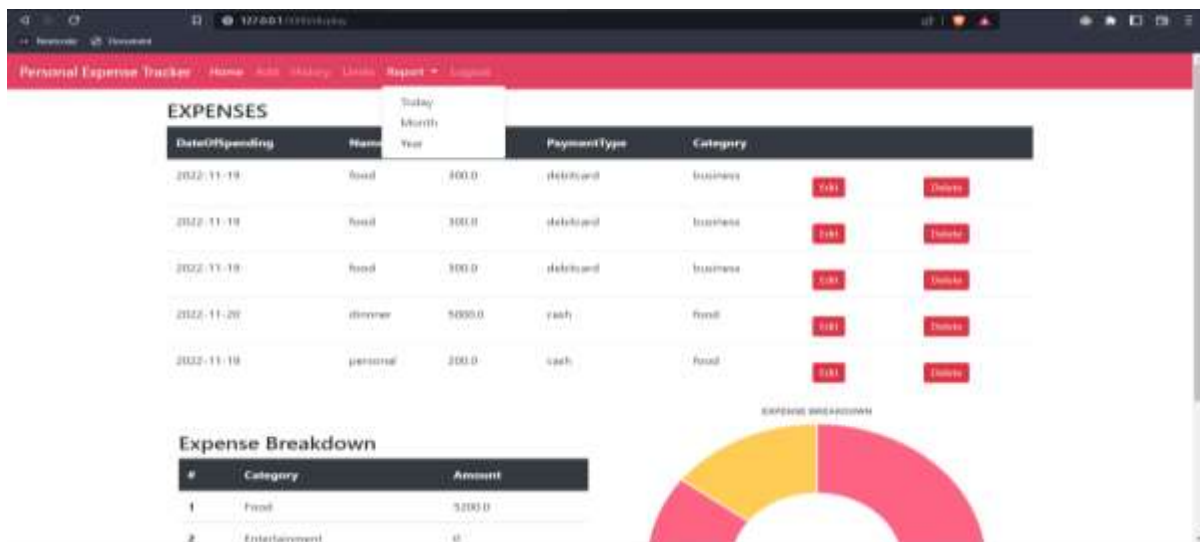
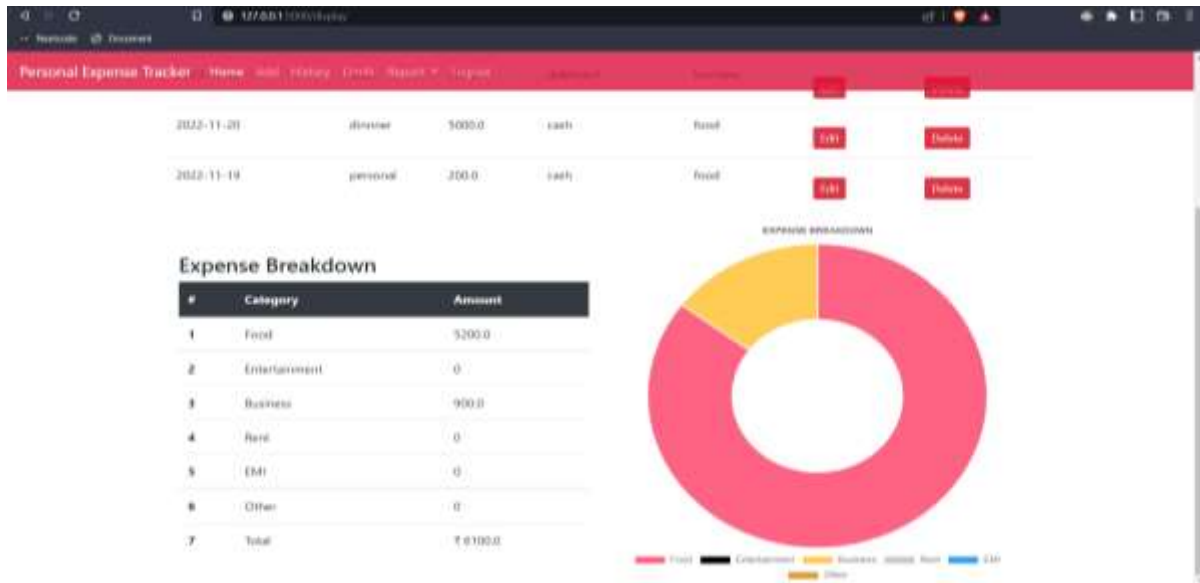
EXPENSES

DateOfSpending	Name	Amount	PaymentType	Category		
2022-11-19	food	400.0	debitcard	business	500	Delete
2022-11-19	food	300.0	debitcard	business	500	Delete
2022-11-19	food	300.0	debitcard	business	500	Delete
2022-11-20	dinner	1000.0	cash	food	500	Delete
2022-11-18	personal	200.0	cash	food	500	Delete

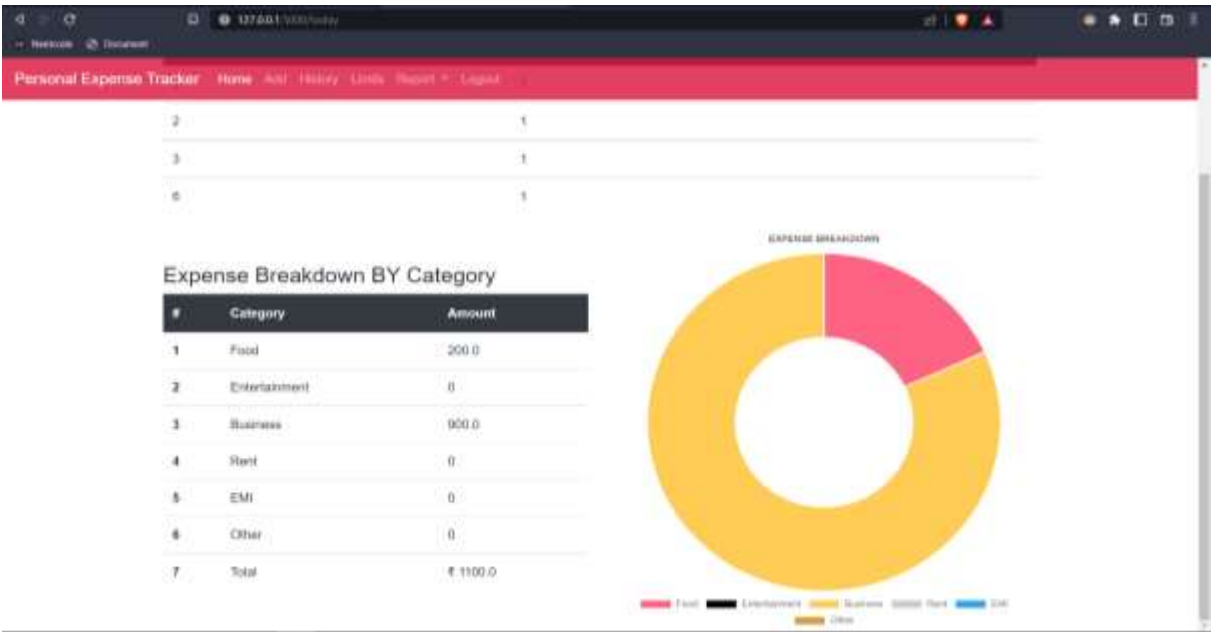
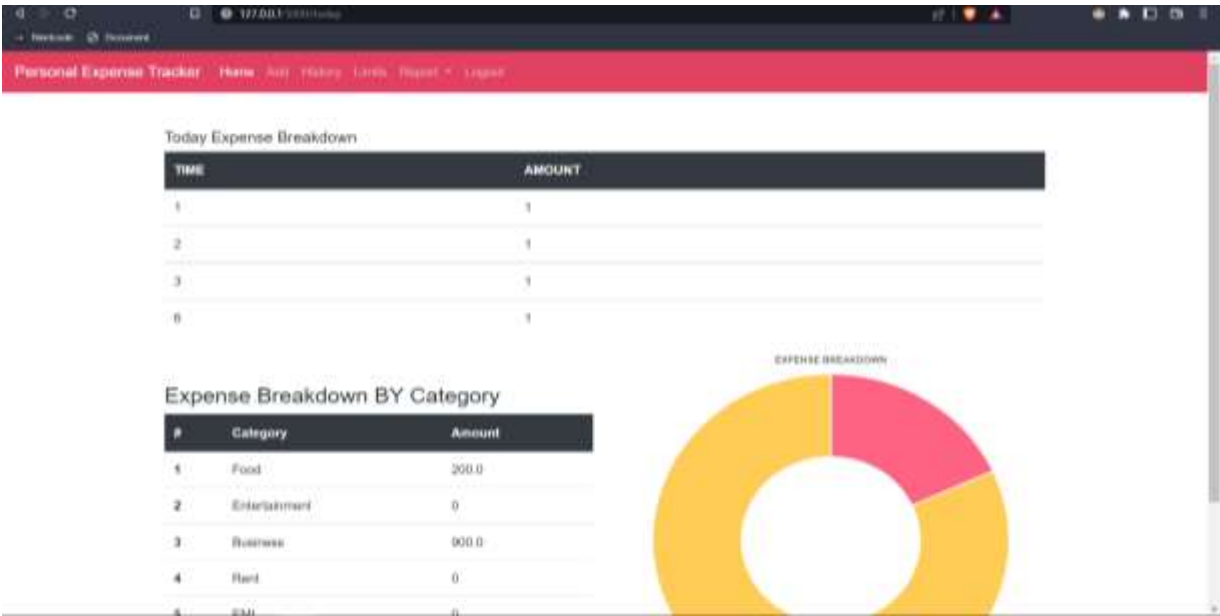
Expense Breakdown

#	Category	Amount
1	Food	5200.0
2	Entertainment	0

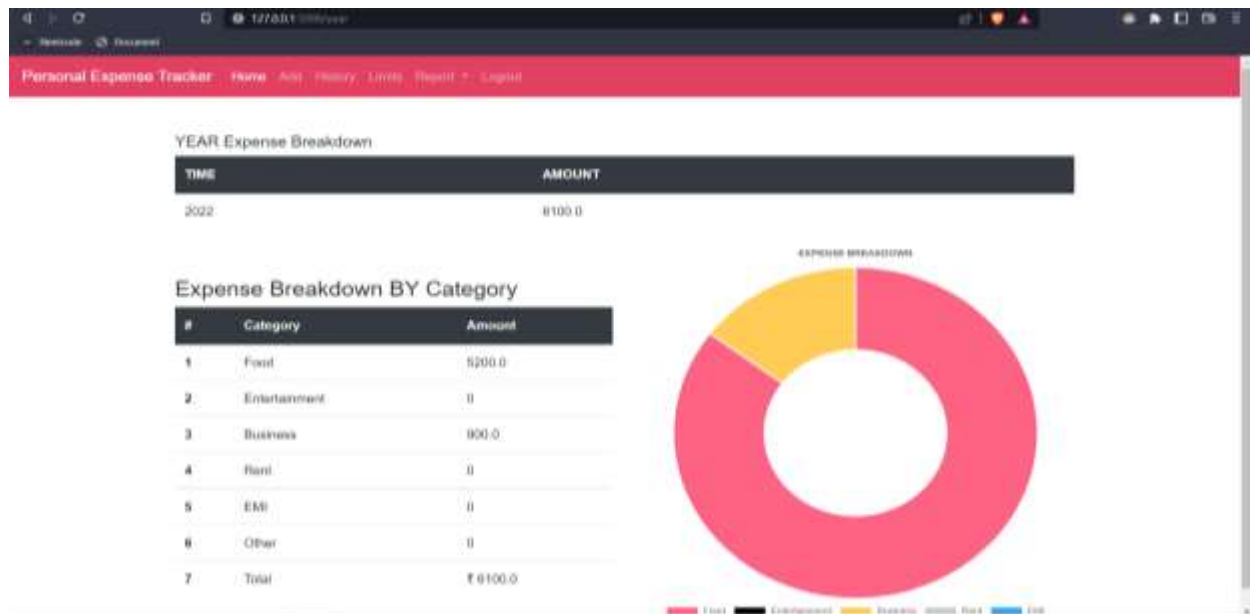
EXPENSE BREAKDOWN



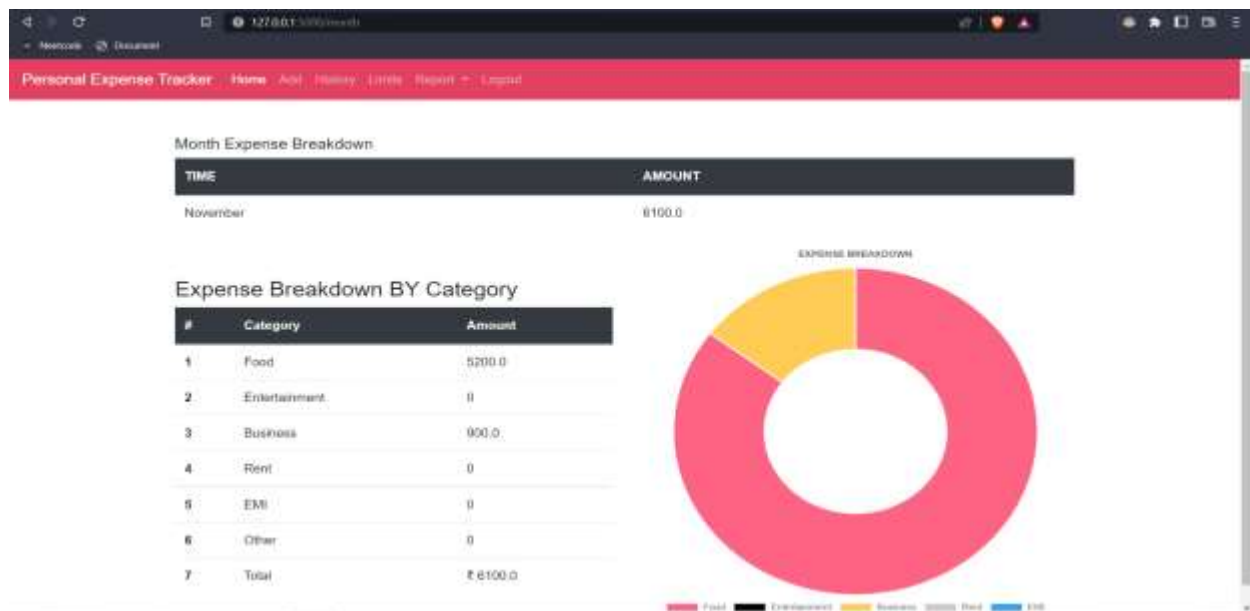
Report (Today)



Report (year)



Report (month)



Set monthly limit



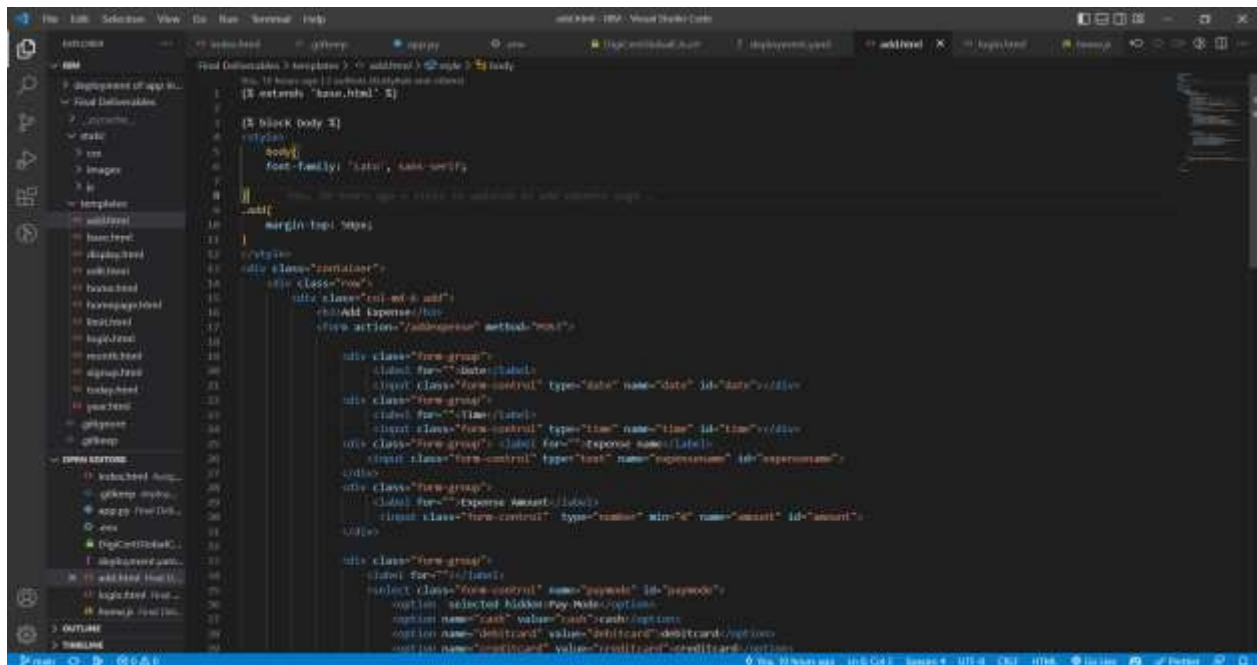
Personal Expense Tracker / Home / Add / History / Limits / Report / Logout

Currently your MONTHLY limit is ₹ (10000.0)

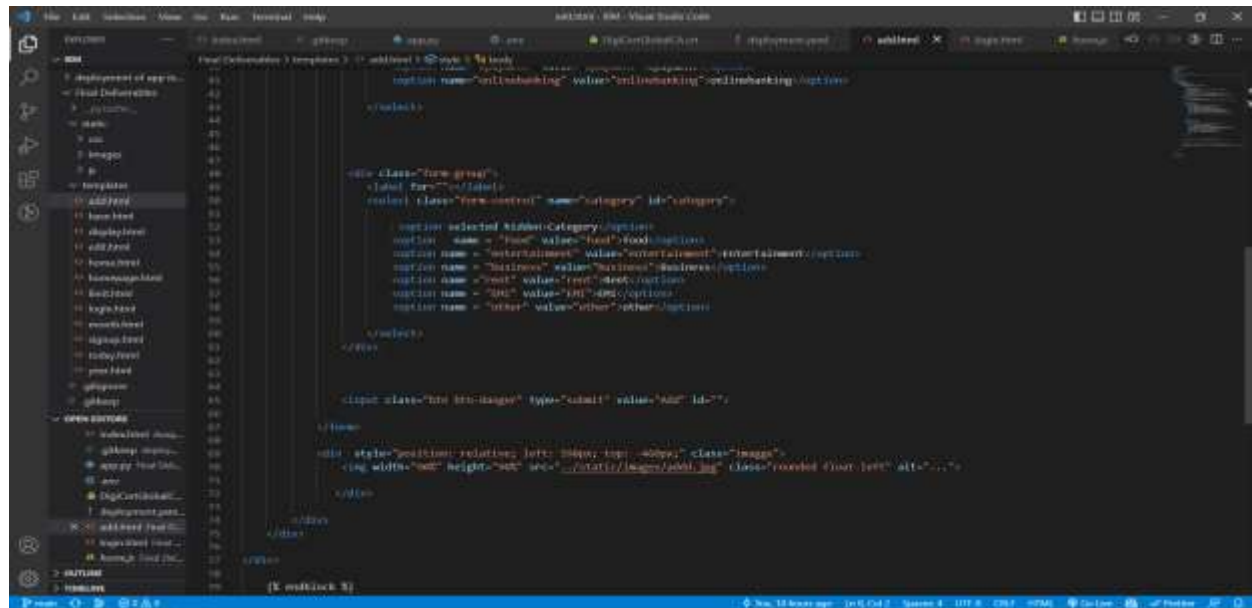
ENTER the MONTHLY LIMIT to avoid over-EXPENSES

Front End Code

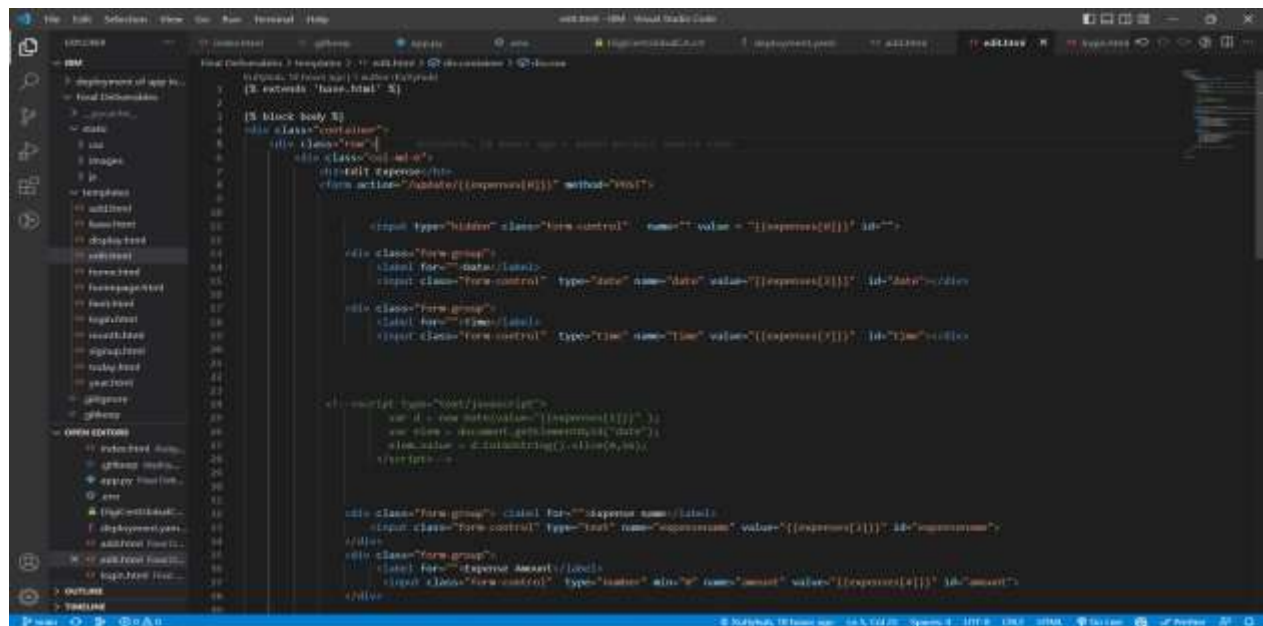
Add expense



```
1 <!-- Add Expense Form -->
2 <!-- This is how you can create the add.html file -->
3 <!-- [X] extends "base.html" -->
4 <!-- [X] block body -->
5 <!-- [X] class="form" -->
6 <!-- [X] class="form" -->
7 <!-- [X] class="form" -->
8 <!-- [X] class="form" -->
9 <!-- [X] class="form" -->
10 <!-- [X] class="form" -->
11 <!-- [X] class="form" -->
12 <!-- [X] class="form" -->
13 <!-- [X] class="form" -->
14 <!-- [X] class="form" -->
15 <!-- [X] class="form" -->
16 <!-- [X] class="form" -->
17 <!-- [X] class="form" -->
18 <!-- [X] class="form" -->
19 <!-- [X] class="form" -->
20 <!-- [X] class="form" -->
21 <!-- [X] class="form" -->
22 <!-- [X] class="form" -->
23 <!-- [X] class="form" -->
24 <!-- [X] class="form" -->
25 <!-- [X] class="form" -->
26 <!-- [X] class="form" -->
27 <!-- [X] class="form" -->
28 <!-- [X] class="form" -->
29 <!-- [X] class="form" -->
30 <!-- [X] class="form" -->
31 <!-- [X] class="form" -->
32 <!-- [X] class="form" -->
33 <!-- [X] class="form" -->
34 <!-- [X] class="form" -->
35 <!-- [X] class="form" -->
36 <!-- [X] class="form" -->
37 <!-- [X] class="form" -->
38 <!-- [X] class="form" -->
39 <!-- [X] class="form" -->
40 <!-- [X] class="form" -->
41 <!-- [X] class="form" -->
42 <!-- [X] class="form" -->
43 <!-- [X] class="form" -->
44 <!-- [X] class="form" -->
45 <!-- [X] class="form" -->
46 <!-- [X] class="form" -->
47 <!-- [X] class="form" -->
48 <!-- [X] class="form" -->
49 <!-- [X] class="form" -->
50 <!-- [X] class="form" -->
51 <!-- [X] class="form" -->
52 <!-- [X] class="form" -->
53 <!-- [X] class="form" -->
54 <!-- [X] class="form" -->
55 <!-- [X] class="form" -->
56 <!-- [X] class="form" -->
57 <!-- [X] class="form" -->
58 <!-- [X] class="form" -->
59 <!-- [X] class="form" -->
60 <!-- [X] class="form" -->
61 <!-- [X] class="form" -->
62 <!-- [X] class="form" -->
63 <!-- [X] class="form" -->
64 <!-- [X] class="form" -->
65 <!-- [X] class="form" -->
66 <!-- [X] class="form" -->
67 <!-- [X] class="form" -->
68 <!-- [X] class="form" -->
69 <!-- [X] class="form" -->
70 <!-- [X] class="form" -->
71 <!-- [X] class="form" -->
72 <!-- [X] class="form" -->
73 <!-- [X] class="form" -->
74 <!-- [X] class="form" -->
75 <!-- [X] class="form" -->
76 <!-- [X] class="form" -->
77 <!-- [X] class="form" -->
78 <!-- [X] class="form" -->
79 <!-- [X] class="form" -->
80 <!-- [X] class="form" -->
81 <!-- [X] class="form" -->
82 <!-- [X] class="form" -->
83 <!-- [X] class="form" -->
84 <!-- [X] class="form" -->
85 <!-- [X] class="form" -->
86 <!-- [X] class="form" -->
87 <!-- [X] class="form" -->
88 <!-- [X] class="form" -->
89 <!-- [X] class="form" -->
90 <!-- [X] class="form" -->
91 <!-- [X] class="form" -->
92 <!-- [X] class="form" -->
93 <!-- [X] class="form" -->
94 <!-- [X] class="form" -->
95 <!-- [X] class="form" -->
96 <!-- [X] class="form" -->
97 <!-- [X] class="form" -->
98 <!-- [X] class="form" -->
99 <!-- [X] class="form" -->
100 <!-- [X] class="form" -->
```



Edit

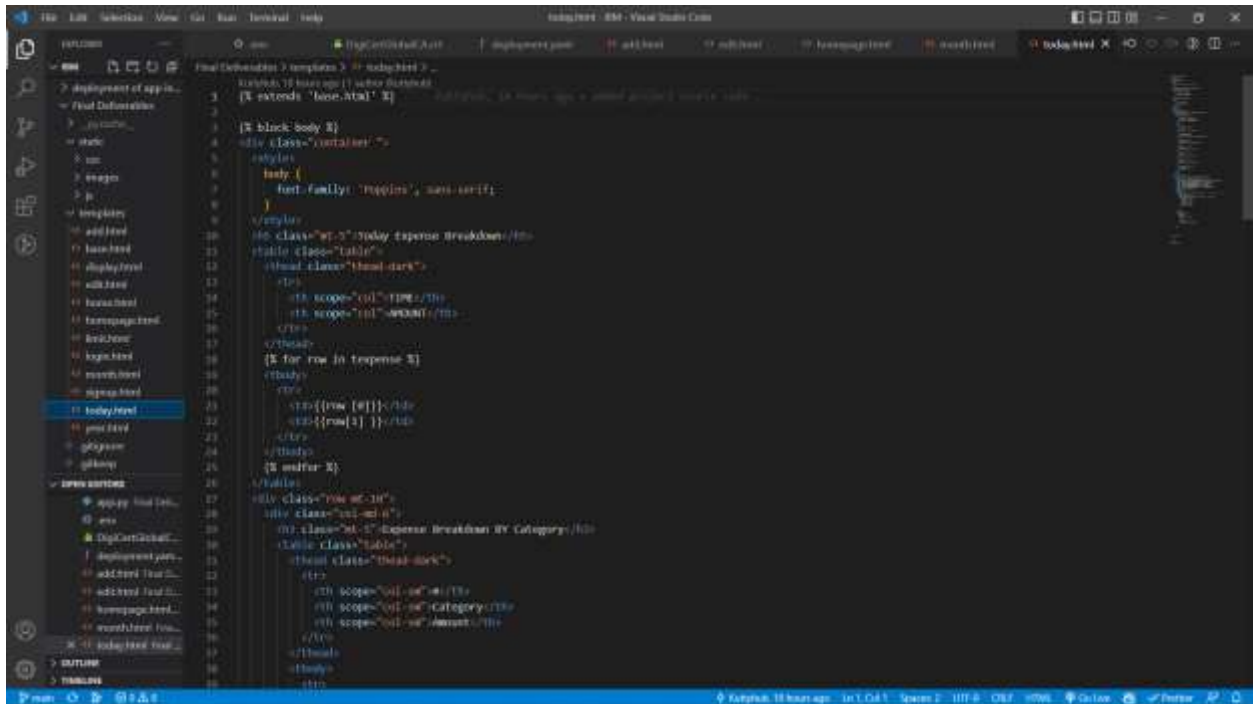




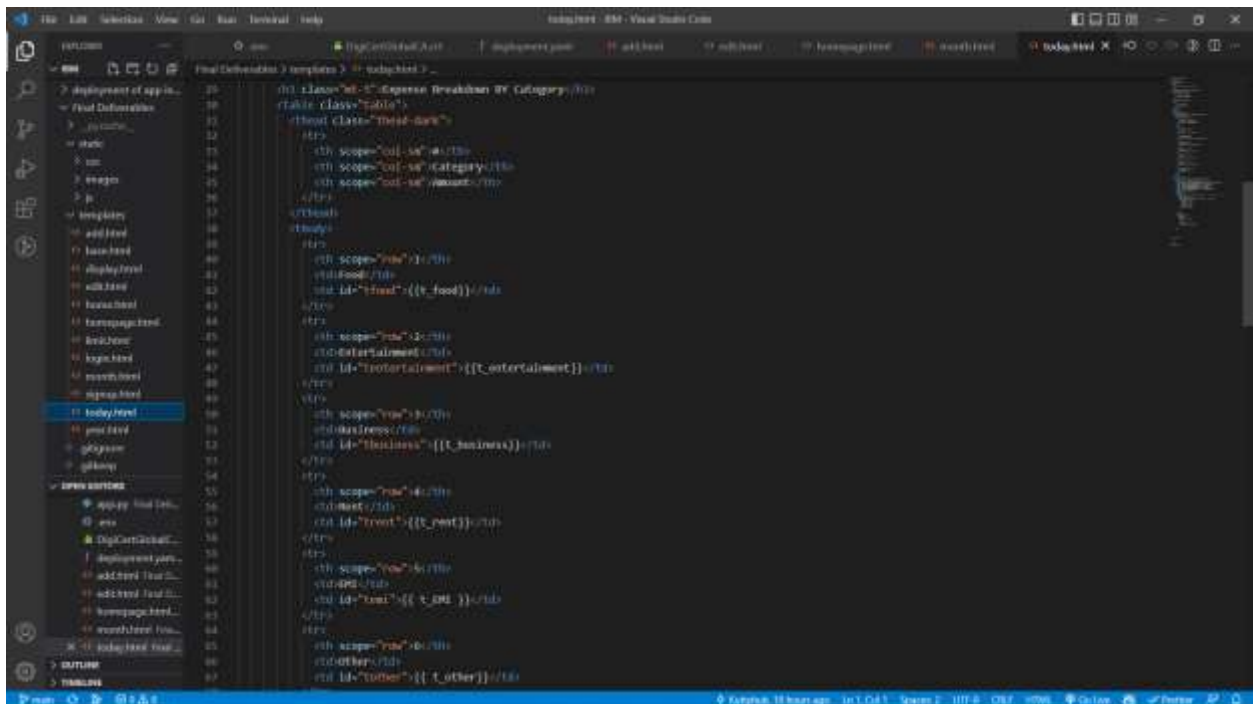
Report (month)[illegible]

The screenshot shows a Visual Studio Code editor with a Jekyll site template. The left sidebar displays a file explorer with a tree structure of files and folders. The main editor area shows the content of 'layouts/much.html', which includes a header, a main content area with a table, and a footer. The table is titled 'Expense Breakdown by Category' and has columns for 'Category' and 'Amount'. The right sidebar shows a preview of the rendered HTML.

Report (today)



```
Final Definition: > templates > > today.html > ...
Liquid: 15 hours ago (1 week outdated)
[X extends 'base.html' X]
{{X block: body X}}
<div class="container">
  <div>
    <body>
      <h1 class="text-center">Today Expense Breakdown</h1>
      <table class="table">
        <thead class="thead-dark">
          <tr>
            <th scope="col">TIME</th>
            <th scope="col">AMOUNT</th>
          </tr>
        </thead>
        <tbody>
          <{% for row in response %}>
            <tr>
              <td>{{row[0]]</td>
              <td>{{row[1]}}</td>
            </tr>
          </tbody>
          <{% endfor %}>
        </table>
      <div class="row mt-3">
        <div class="col-md-6">
          <h2 class="text-center">Expense Breakdown BY Category</h2>
          <table class="table">
            <thead class="thead-dark">
              <tr>
                <th scope="col">cat</th>
                <th scope="col">category</th>
                <th scope="col">Amount</th>
              </tr>
            </thead>
            <tbody>
              <tr>
                <th scope="row">1</th>
                <td>Food</td>
                <td id="food">{{t_food}}</td>
              </tr>
              <tr>
                <th scope="row">2</th>
                <td>Entertainment</td>
                <td id="entertainment">{{t_entertainment}}</td>
              </tr>
              <tr>
                <th scope="row">3</th>
                <td>Business</td>
                <td id="business">{{t_business}}</td>
              </tr>
              <tr>
                <th scope="row">4</th>
                <td>Rent</td>
                <td id="rent">{{t_rent}}</td>
              </tr>
              <tr>
                <th scope="row">5</th>
                <td>Other</td>
                <td id="other">{{t_other}}</td>
              </tr>
            </tbody>
          </table>
        </div>
      </div>
    </div>
  </div>
</body>
</div>
```



```
Final Definition: > templates > > today.html > ...
Liquid: 15 hours ago (1 week outdated)
[X extends 'base.html' X]
{{X block: body X}}
<div class="container">
  <div>
    <body>
      <h1 class="text-center">Expense Breakdown BY Category</h1>
      <table class="table">
        <thead class="thead-dark">
          <tr>
            <th scope="col">cat</th>
            <th scope="col">category</th>
            <th scope="col">Amount</th>
          </tr>
        </thead>
        <tbody>
          <tr>
            <th scope="row">1</th>
            <td>Food</td>
            <td id="food">{{t_food}}</td>
          </tr>
          <tr>
            <th scope="row">2</th>
            <td>Entertainment</td>
            <td id="entertainment">{{t_entertainment}}</td>
          </tr>
          <tr>
            <th scope="row">3</th>
            <td>Business</td>
            <td id="business">{{t_business}}</td>
          </tr>
          <tr>
            <th scope="row">4</th>
            <td>Rent</td>
            <td id="rent">{{t_rent}}</td>
          </tr>
          <tr>
            <th scope="row">5</th>
            <td>Other</td>
            <td id="other">{{t_other}}</td>
          </tr>
        </tbody>
      </table>
    </div>
  </div>
</body>
</div>
```



Flask Code

Add expenses

```

116 @app.route("/add")
117 def adding():
118     return render_template("add.html")
119
120 @app.route("/addexpense", methods=["GET", "POST"])
121 def addexpense():
122
123     date = request.form["date"]
124     expensename = request.form["expensename"]
125     amount = request.form["amount"]
126     paymode = request.form["paymode"]
127     category = request.form["category"]
128     time = request.form["time"]
129
130     sql = "INSERT INTO EXPENSES(DATEID,DATE,EXPENSENAME,AMOUNT,PAYMENTMODE,CATEGORY,TIME) VALUES(?, ?, ?, ?, ?, ?, ?)"
131     stat = db.cursor(cursor, sql)
132     db.cursor(cursor, stat, 1, session["id"])
133     db.cursor(cursor, stat, 2, date)
134     db.cursor(cursor, stat, 3, expensename)
135     db.cursor(cursor, stat, 4, amount)
136     db.cursor(cursor, stat, 5, paymode)
137     db.cursor(cursor, stat, 6, category)
138     db.cursor(cursor, stat, 7, time)
139     db.cursor(cursor, stat)
140
141     print(date + " " + expensename + " " +
142           amount + " " + paymode + " " + category)
143
144     sql = "SELECT * FROM EXPENSES WHERE USERID=? AND MONTH(DATE)=MONTH(DATE(NOW()))"
145     stat1 = db.cursor(cursor, sql1)
146     db.cursor(cursor, stat1, 1, session["id"])
147     db.cursor(cursor, stat1)
148     list2 = []
149     expense1 = db.cursor(cursor, stat1)
150     while expense1:
151         list2.append(expense1)
152         expense1 = db.cursor(cursor, stat1)
153     total = 0
154     for a in list2:
155         total += a[4]
156
157

```

```

144 category = request.form["category"]
145 time = request.form["time"]
146
147 sql = "INSERT INTO EXPENSES(DATEID,DATE,EXPENSENAME,AMOUNT,PAYMENTMODE,CATEGORY,TIME) VALUES(?, ?, ?, ?, ?, ?, ?)"
148 stat = db.cursor(cursor, sql)
149 db.cursor(cursor, stat, 1, session["id"])
150 db.cursor(cursor, stat, 2, date)
151 db.cursor(cursor, stat, 3, expensename)
152 db.cursor(cursor, stat, 4, amount)
153 db.cursor(cursor, stat, 5, paymode)
154 db.cursor(cursor, stat, 6, category)
155 db.cursor(cursor, stat, 7, time)
156 db.cursor(cursor, stat)
157
158 print(date + " " + expensename + " " +
159       amount + " " + paymode + " " + category)
160
161 sql = "SELECT * FROM EXPENSES WHERE USERID=? AND MONTH(DATE)=MONTH(DATE(NOW()))"
162 stat1 = db.cursor(cursor, sql1)
163 db.cursor(cursor, stat1, 1, session["id"])
164 db.cursor(cursor, stat1)
165 list2 = []
166 expense1 = db.cursor(cursor, stat1)
167 while expense1:
168     list2.append(expense1)
169     expense1 = db.cursor(cursor, stat1)
170 total = 0
171 for a in list2:
172     total += a[4]
173
174 sql2 = "SELECT EXPENSEID FROM LIMITS ORDER BY LIMITS.ID DESC LIMIT 1"
175 stat2 = db.cursor(cursor, sql2)
176 db.cursor(cursor, stat2)
177 limit = db.cursor(cursor, stat2)
178
179 if len(list2) + 1 > 0 and total < limit[0]:
180     sendEmail(session["email"])
181     return redirect("/display")
182

```


Display details

```

@app.route("/display")
def display():
    print(session["username"], session["id"])
    sql = "SELECT * FROM EXPENSE WHERE USED=1"
    stat = db.cursor(cursor, sql)
    db.execute(stat, 1, session["id"])
    list1 = []
    row = db.fetch_tuple(stat)
    while row:
        list1.append(row)
        row = db.fetch_tuple(stat)
    print("list1, exp=", list1)
    total = 0
    t_food = 0
    t_entertainment = 0
    t_business = 0
    t_rent = 0
    t_rent1 = 0
    t_rent2 = 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] == "rent1":
            t_rent1 += x[4]
        elif x[6] == "rent2":
            t_rent2 += x[4]
        elif x[6] == "other":
            t_other += x[4]

    return render_template(
        "display.html",
        expense=list1
    )

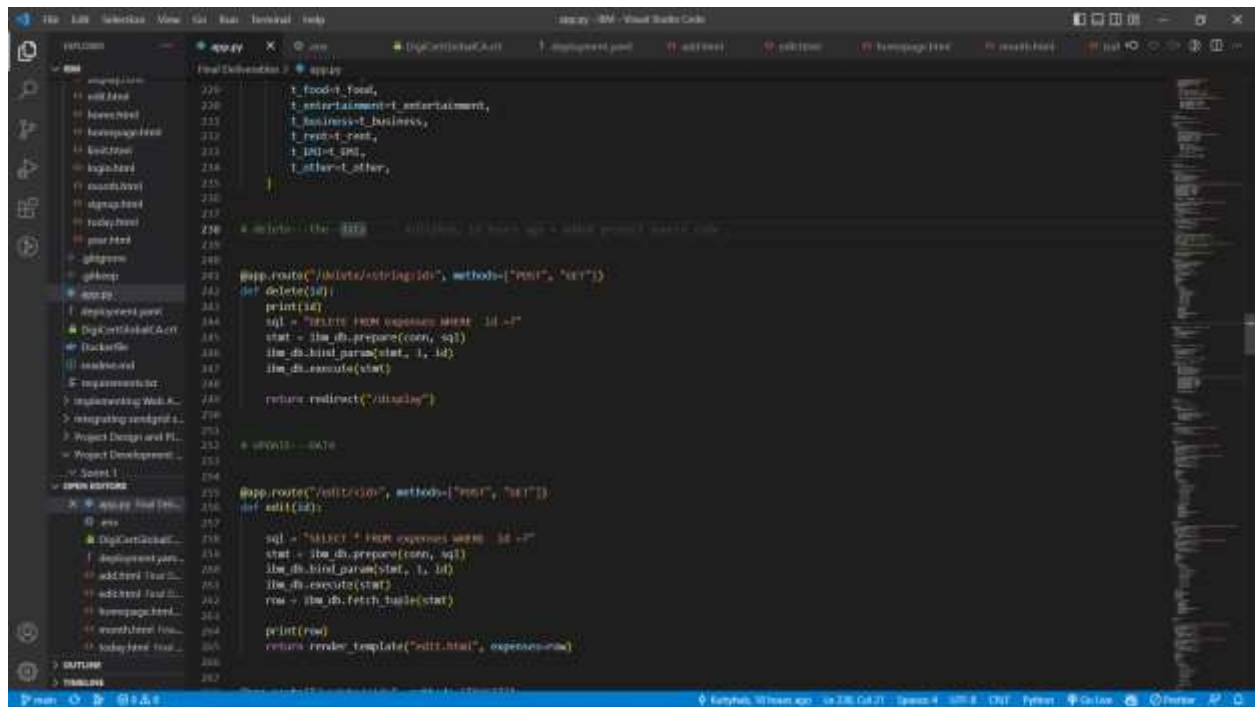
```

```

100 while True:
101     list1.append(row)
102     row = db.fetch_tuple(stmt)
103     print("list1, sep='\n'")
104     total = 0
105     t_food = 0
106     t_entertainment = 0
107     t_business = 0
108     t_rent = 0
109     t_img = 0
110     t_other = 0
111
112 for x in list1:
113     total += x[4]
114     if x[6] == "food":
115         t_food += x[4]
116     elif x[6] == "entertainment":
117         t_entertainment += x[4]
118     elif x[6] == "business":
119         t_business += x[4]
120     elif x[6] == "rent":
121         t_rent += x[4]
122     elif x[6] == "img":
123         t_img += x[4]
124     elif x[6] == "other":
125         t_other += x[4]
126
127 return render_template(
128     "display.html",
129     expense=list1,
130     total=total,
131     t_food=t_food,
132     t_entertainment=t_entertainment,
133     t_business=t_business,
134     t_rent=t_rent,
135     t_img=t_img,
136     t_other=t_other,
137 )

```

Delete the data



The screenshot shows the VS Code editor with the file explorer on the left displaying a project structure. The main editor window shows the `app.js` file with the following code:

```
const express = require('express');
const app = express();
const mongoose = require('mongoose');
const bcrypt = require('bcrypt');
const jwt = require('jsonwebtoken');
const User = mongoose.model('User');
const Expense = mongoose.model('Expense');

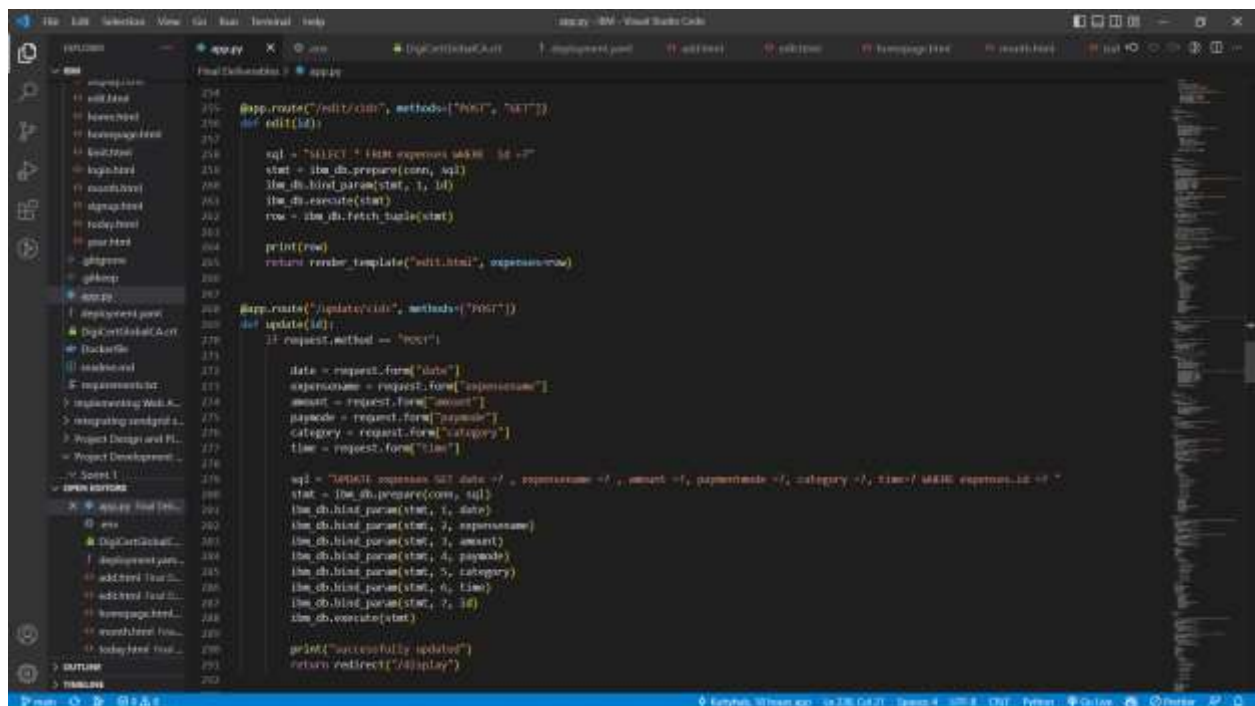
// Middleware
app.use(express.json());
app.use(express.urlencoded({ extended: true }));

// Routes
// ... (previous routes) ...

// DELETE route
app.delete('/delete/:id', async (req, res) => {
  const id = req.params.id;
  const query = { _id: id };
  const result = await Expense.deleteOne(query);
  if (result.deletedCount > 0) {
    res.status(200).json({ message: 'Expense deleted successfully' });
  } else {
    res.status(404).json({ message: 'Expense not found' });
  }
});

// UPDATE route
app.put('/update/:id', async (req, res) => {
  const id = req.params.id;
  const { date, expenseName, amount, paymode, category, time } = req.body;
  const query = { _id: id };
  const result = await Expense.updateOne(query, {
    date,
    expenseName,
    amount,
    paymode,
    category,
    time
  }, { upsert: true });
  if (result.modifiedCount > 0) {
    res.status(200).json({ message: 'Expense updated successfully' });
  } else {
    res.status(404).json({ message: 'Expense not found' });
  }
});
```

Update the data



The screenshot shows the VS Code editor with the file explorer on the left displaying a project structure. The main editor window shows the `app.js` file with the following code:

```
const express = require('express');
const app = express();
const mongoose = require('mongoose');
const bcrypt = require('bcrypt');
const jwt = require('jsonwebtoken');
const User = mongoose.model('User');
const Expense = mongoose.model('Expense');

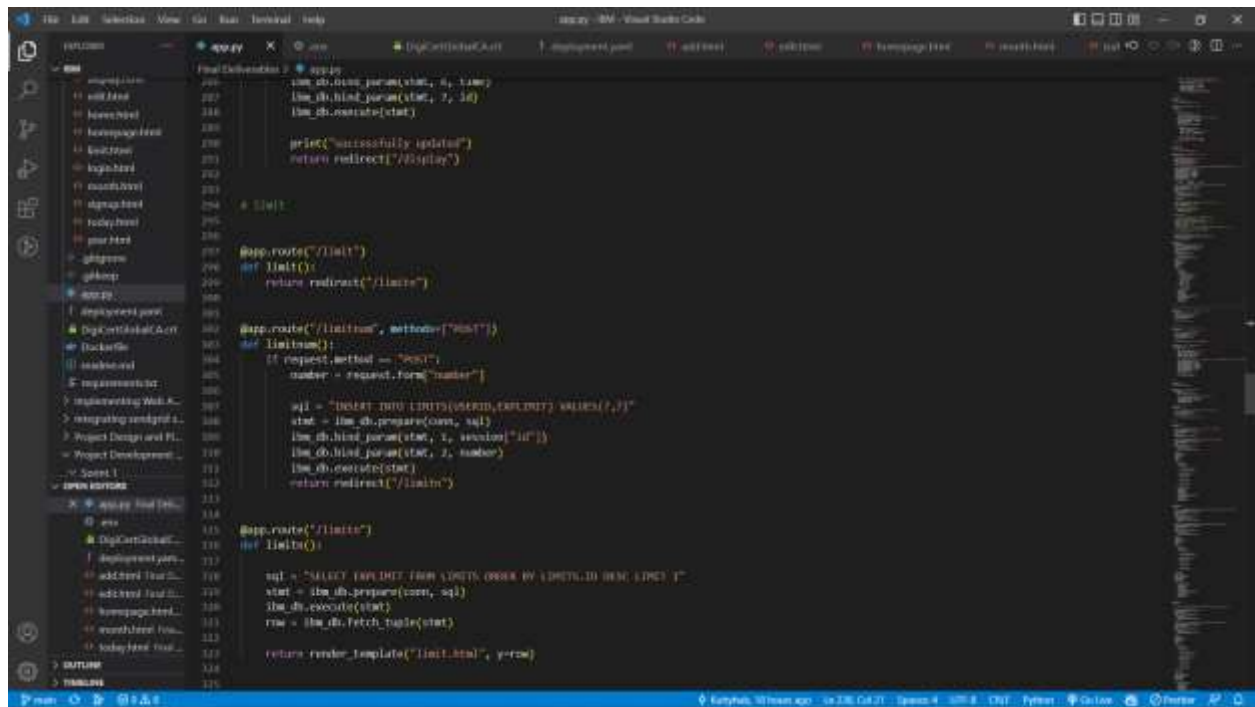
// Middleware
app.use(express.json());
app.use(express.urlencoded({ extended: true }));

// Routes
// ... (previous routes) ...

// DELETE route
app.delete('/delete/:id', async (req, res) => {
  const id = req.params.id;
  const query = { _id: id };
  const result = await Expense.deleteOne(query);
  if (result.deletedCount > 0) {
    res.status(200).json({ message: 'Expense deleted successfully' });
  } else {
    res.status(404).json({ message: 'Expense not found' });
  }
});

// UPDATE route
app.put('/update/:id', async (req, res) => {
  const id = req.params.id;
  const { date, expenseName, amount, paymode, category, time } = req.body;
  const query = { _id: id };
  const result = await Expense.updateOne(query, {
    date,
    expenseName,
    amount,
    paymode,
    category,
    time
  }, { upsert: true });
  if (result.modifiedCount > 0) {
    res.status(200).json({ message: 'Expense updated successfully' });
  } else {
    res.status(404).json({ message: 'Expense not found' });
  }
});
```


Limit



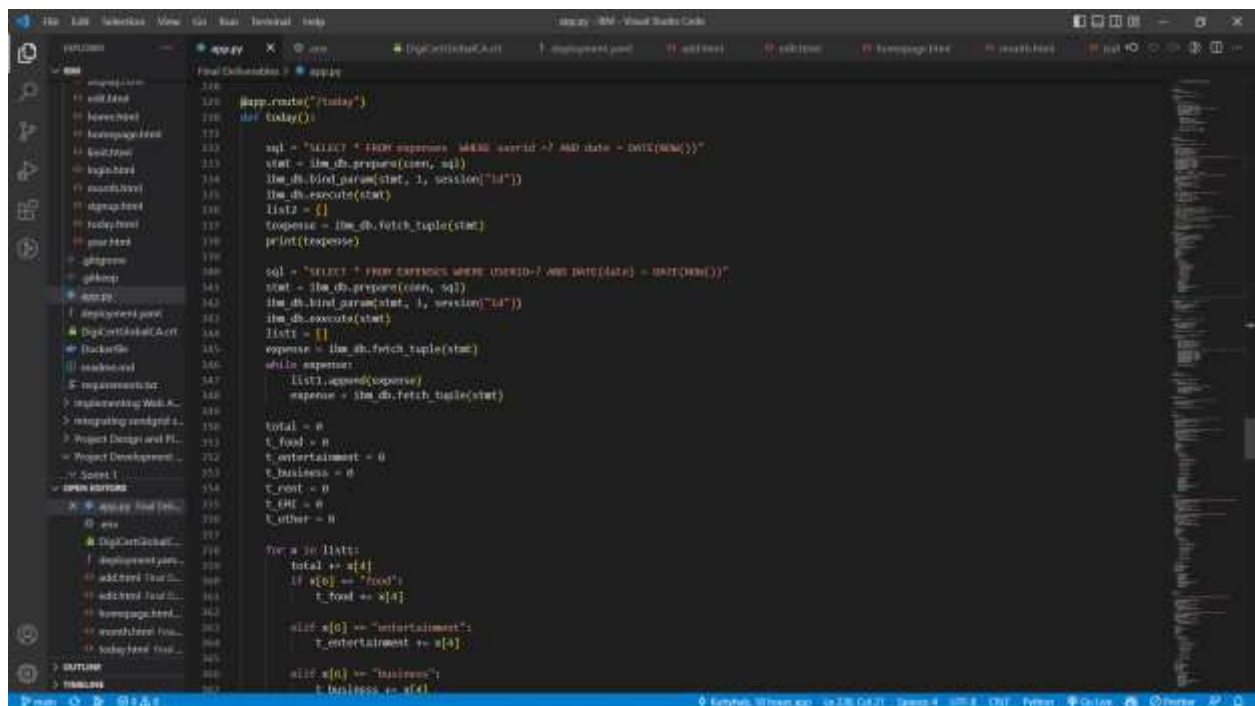
```
def update_limit(self, user_id, limit):
    cursor = self.db.cursor()
    cursor.execute("UPDATE users SET limit = %s WHERE user_id = %s", (limit, user_id))
    self.db.commit()
    return jsonify({"message": "Limit updated successfully"}), 200

@app.route("/limit", methods=["POST"])
def limit():
    if request.method == "POST":
        user_id = request.json.get("user_id")
        limit = request.json.get("limit")

        sql = "UPDATE users SET limit = %s WHERE user_id = %s"
        cursor = self.db.cursor()
        cursor.execute(sql, (limit, user_id))
        self.db.commit()
        return jsonify({"message": "Limit updated successfully"}), 200

    return jsonify({"message": "Invalid request"}), 400
```

Report (today)



```
def report(self, date):
    cursor = self.db.cursor()
    cursor.execute("SELECT * FROM expenses WHERE DATE(date) = DATE(%s)", (date,))
    results = cursor.fetchall()
    return jsonify(results), 200

@app.route("/report", methods=["POST"])
def report():
    if request.method == "POST":
        date = request.json.get("date")

        sql = "SELECT * FROM expenses WHERE DATE(date) = DATE(%s)"
        cursor = self.db.cursor()
        cursor.execute(sql, (date,))
        results = cursor.fetchall()
        return jsonify(results), 200

    return jsonify({"message": "Invalid request"}), 400
```