# Criterion E: Product development

Techniques used to address the client's requirements:

- Utilizing Python to do Backend Processing
- Utilizing Python functions for controlling data within the SQL Database
- Cascading Style Sheets
- Use of HTML to display information and data validation
- JavaScript to show data more effectively
- Unique Properties inside SQL database
- Photoshop to make Logo

## Structure of website:

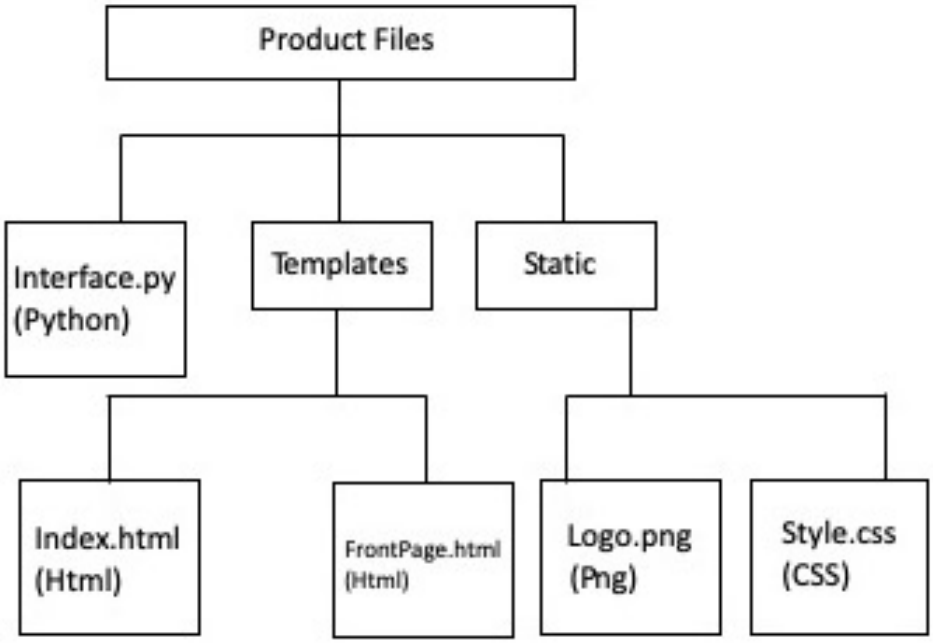



**Figure 1: Layout of the file organization:**

The content for the website has been organized into folders dependent on their purpose, templates stored both html files. Static contains the logo and the style sheet, the name static because it is unlikely that these files will change, but also because it makes it easy to locate. The Interface file has been placed outside of folders because it is the file that holds the attributes for the server, therefore it is the first that will be executed.

For the image I have decided to utilize PNG over JPEG since it allows for images to have a transparent background.

## The structure and organization of the webpages
## Technique: Python Backend Processing

```
def check_auth(username, password):
    '''THIS IS FOR THE USERNAME AND PASSWORD VARIABLE'''
    return username == os.environ['USERNAME']; password == os.environ['PASSWORD'];

def authenticate():
    return Response(
    'Could not verify your access level for that URL.\n'
    'You have to login with proper credentials', 401,
    {'WWW-Authenticate': 'Basic realm="Login Required"'})

def requires_auth(f):
    @wraps(f)
    def decorated(*args, **kwargs):
        auth = request.authorization
        if not auth or not check_auth(auth.username, auth.password):
            return authenticate()
        return f(*args, **kwargs)
    return decorated

@app.route('/')
def mainidea():
    return render_template('frontpage.html')
```

**Figure 2: Python file that runs on the Server**

The python file holds all the attributed for how the website should act. It states what pages should load up based on the URL. For example, '/' would open the front page, when '/home' would load up the main webpage.

The python file also fetches sensitive data that is stored on the server, the reason this was used is so users can't use Inspect Elements on modern browsers to get data such as the database URL, or the username and password. Also, tools such as user agents won't work since this is a python file.

Functions created on python can be used by the html page by doing "action='/functionName'". Python file also shows errors on the webpage if something goes wrong, with troubleshooting tips listed within the log.

| | |
|---|---|
| @requires_auth | Calls the function that shows a banner on the screen for users to input credentials. |
| @authenticate | Show correct error message for users stating that they need to use to correct credential |
| @check_auth | Checks entered credentials compared to what is stored on the server. |
| @app.route | Routes to what webpage should load based on the URL. |

## Technique: Utilizing Python functions for controlling data within the SQL Database

```
@app.route('/home')
@requires_auth
def hello_world():
        conn = psycopg2.connect(connection_string)
        cursor = conn.cursor(cursor_factory=psycopg2.extras.DictCursor)
        cursor.execute("SELECT * FROM expenses ORDER BY reading_date DESC LIMIT 5")
        records = cursor.fetchall()

        cursor.execute("SELECT * FROM expenses ORDER BY temperature DESC LIMIT 1 ")
        highest = cursor.fetchall()
        max = highest[0]

        cursor.execute("SELECT * FROM expenses ORDER BY temperature ASC LIMIT 1")
        lowest = cursor.fetchall()
        low = lowest[0]
```

**Figure 3: Showcases input data into database through QUERIES**

Python was utilized here to fetch information from the database and sort it based on characteristics such as ascending. This is done utilizing an SQL query, which is a unique statement to fetch or insert data from a SQL database.

```
@app.route('/handle',methods=['POST'])
@requires_auth
def handle():
    conn = psycopg2.connect(connection_string)
    cursor = conn.cursor(cursor_factory=psycopg2.extras.DictCursor)
    amount = float(request.form['amount'])
    location = str(request.form['location'])
    current_date = datetime.datetime.now()
    weekdays = datetime.datetime.now().strftime('%A')
    query = "INSERT INTO expenses(famount, reading_date, locations, weekday) VALUES (%s, %s, %s, %s)"
    val = (amount, current_date, location, weekdays)
    cursor.execute(query, val)
    conn.commit()
    conn.close()
    return redirect('/home')

@app.route('/deletedatabase',methods=['POST'])
@requires_auth
def deletedatabase():
    conn = psycopg2.connect(connection_string)
    cursor = conn.cursor(cursor_factory=psycopg2.extras.DictCursor)
    amount = float(request.form['amount'])
    query = 'DELETE FROM expenses WHERE famount=(%s)'
    val = (amount)
    cursor.execute(query, val)
    conn.commit()
    conn.close()
    return redirect('/home')
```

**Figure 4: Showcases deletion of records in the database through QUERIES**

As shown in the screenshot below SQL queries can also be utilized to input data into the database. Python grabs the variables from the HTML file through request. Form and stores them. Then it inputs all of them into the database. SQL queries within python can also be used to delete data inside of the database, as seen in the function deletedatabase().
This is another reason python was used because it provides native support for SQL queries. A simple HTML file wouldn't be able to do this without needing some sort of third-party library.

## Technique: Cascading Style Sheets

```css
body {
    opacity: 0.8;
}
    span {
        border-color:  white;
        border-style: solid;
        border-width: 2px;
        background-color: white;
    }
.LOGOSF{
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: hidden;
    background-color: black;
    float: right;
    display: block;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;
    font-family: "Courier"
}

table {
    border-collapse: collapse;
    width: 80%;
    margin-left: auto;
    margin-right: auto;
    color: white;

}

td {
    text-align: left;
    padding: 8px;
    font-family: "Trebuchet MS"
}
th {
    text-align: left;
    padding: 8px;
    background-color: rgba(255,255,255,0.8);
    color: #f24343;
    font-family: "Courier"
}
```

**Figure 7: Showcasing the Cascading Style Sheet**

This website was created through hard coding every part of it. I utilized cascading style sheets to decrease the amount of code on the webpage, making it easier for troubleshooting and maintenance. This file controls all aspects of styling of the webpage, you can set attributes to certain things by adding a "class" or "id".
If you want to utilize the same styling for other title or text, instead of repeating the styling you just add the same ID or Class that you used before.

```
<h1 class="LOGOSF"
```

**Figure 5: Showcasing how classes are used**

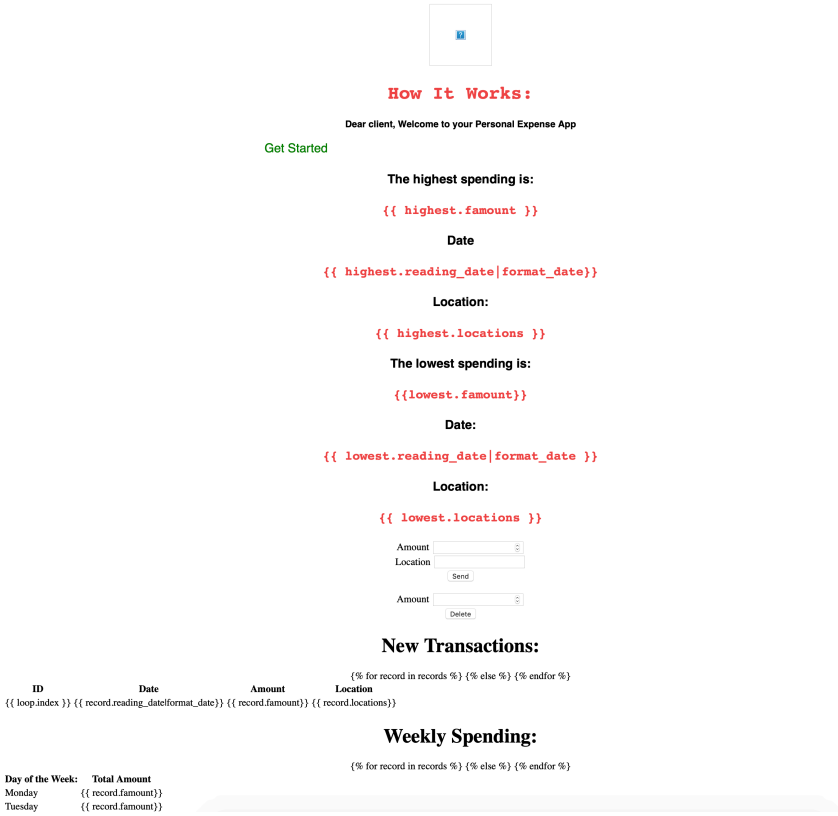
To link the file to the webpage the following statement is used.

```
<link
href="../static/style.css">
```

**Figure 6: Showcases how CSS file is linked to HTML file**

# Effects of CSS and Python on Webpage:

## Before:



How It Works:

Dear client, Welcome to your Personal Expense App

Get Started

The highest spending is:

{{ highest.famount }}

Date

{{ highest.reading_date|format_date}}

Location:

{{ highest.locations }}

The lowest spending is:

{{lowest.famount}}

Date:

{{ lowest.reading_date|format_date }}

Location:

{{ lowest.locations }}

Amount

Location

Send

Amount

Delete

New Transactions:

{% for record in records %} {% else %} {% endfor %}

| ID | Date | Amount | Location |
|---|---|---|---|
| {{ loop.index }} | {{ record.reading_datelformat_date}} | {{ record.famount}} | {{ record.locations}} |

Weekly Spending:

{% for record in records %} {% else %} {% endfor %}

| Day of the Week: | Total Amount |
|---|---|
| Monday | {{ record.famount}} |
| Tuesday | {{ record.famount}} |

Figure 8: Website with no CSS

## After:

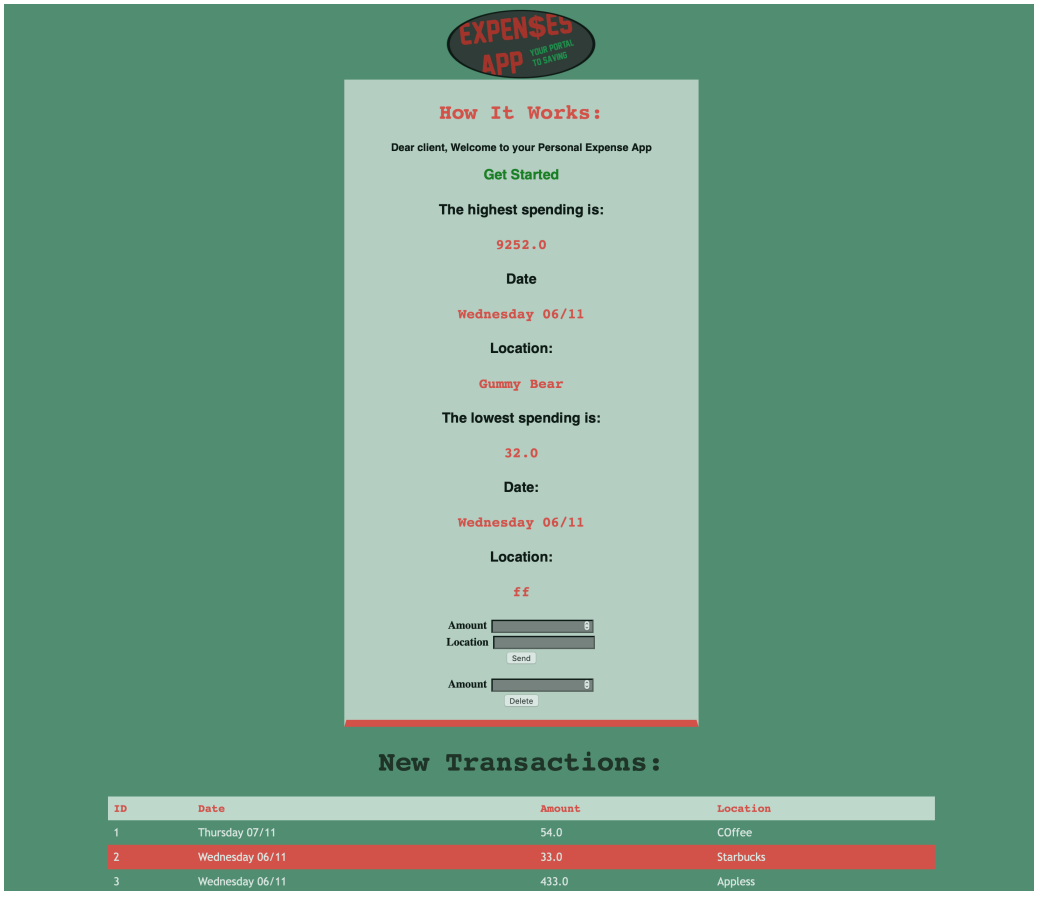


**Figure 9: Website with CSS**

## Technique: Use of HTML to display information and data validation

```
<h4 style="text-align: center;
color:black;
font-size:20px;
font-family: sans-serif">Location:</h4>
<h4 style="text-align: center;
    color:#f24343;
    font-size:20px;
    font-family: Courier">{{ highest.locations }}</h4>

    <h4 style="text-align: center;
    color:black;
    font-size:20px;
    font-family: sans-serif">The lowest spending is:</h4>
    <h4 style="text-align: center;
        color:#f24343;
        font-size:20px;
        font-family:Courier">{{lowest.famount}}</h4>
    <h4 style="text-align: center;
    color:black;
    font-size:20px;
    font-family: sans-serif">Date:</h4>
        <h4 style="text-align: center;
        color:#f24343;
        font-size:20px;
        font-family: Courier">{{ lowest.reading_date|format_date }}</h4>
    <h4 style="text-align: center;
    color:black;
    font-size:20px;
    font-family: sans-serif">Location:</h4>
```

**Figure 9: How the webpage showcases information from the database**

HTML as seen below can gather information straight from the database and embedded it based on desired formatting, this ensures that information will always appear quickly on screen.

```
<form method="post" action="/handle">
    Amount <input id="textbox"type="number" step="0.01" name="amount"/><br>
    <form method="post" action="/handle">
        Location <input id="textbox"type="text" name="location"/><br>
    <input type="submit" value="Send"/>
</form>
<form method="post" action="/deletedatabase">
    Amount <input id="textbox"type="number" name="amount"/><br>
<form method="post" action="/deletedatabase"><input type="submit" value="Delete"/></form>
```

**Figure 10: How user input is taken**

There is also data validation in place because of the “type” property that can be added to a textbox. “number” ensures that the user is not able to input anything but numbers. This is useful to ensure that data is correctly added into the database, otherwise it could lead to breaking the database.

## Technique: JavaScript to show data more easily

```
<script>
        function myFunction() {
          var popup = document.getElementById("myPopup");
          popup.classList.toggle("show");
        }
        </script>
```

**Figure 11: JavaScript text**

In order to make the webpage less crowded I utilized JavaScript to show information that isn't required to be present at all times. The function below creates an animation that when pressed shows the user how to use the app. This is only required when the user first starts using the webpage.

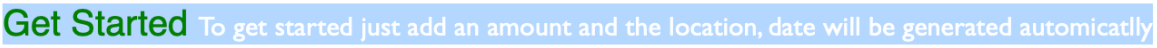



**Figure 11: No JavaScript being utilized**


**Figure 12: JavaScript being used**

## Technique: Unique Properties inside SQL database

| # | column_name | data_type | is_nullable | check | column_default | foreign_key | comment |
|---|---|---|---|---|---|---|---|
| 2 | famount | float4 | YES | NULL | NULL | EMPTY | NULL |
| 4 | reading_date | timestamp | YES | NULL | NULL | EMPTY | NULL |
| 5 | locations | text | YES | NULL | NULL | EMPTY | NULL |
| 6 | weekday | text | YES | NULL | NULL | EMPTY | NULL |

**Figure 13: Unique properties of the database**

The SQL database is a relational management system that allows for data to be added using 'queries. This database is active on the same server that broadcasts the website. Each of the column has a primary key, which is shown by the #. Each of the columns also include a different data type, this ensures that only data within that type will be added. This database also includes a Nullable feature which ensures that it accepts blank values. This is useful when the user wants to input a value but doesn't remember the location for example.

## Technique: Using Photoshop to create the logo

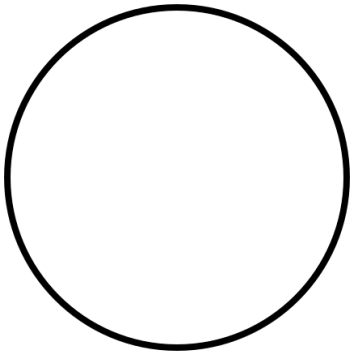



**Figure 13: Circle (logo before being edited) Figure 14: Final logo after being edited**

This logo was made utilizing Photoshop. I started with a simple outline of a circle and did the following steps to turn it into the logo on the right.

1. Stretch the circle using the transform tool.
2. I added a stroke with a position of inside to create the grey background around the circle.
3. Used the text tool to type “EXPEN$ES APP” and “YOUR PORTAL TO SAVING”.
4. I then used color overlay within blending options to change the color.
5. I then used the transform tool to resize it.

## Python libraries used:

| | |
|---|---|
| Flask | Flask is what allows to manage a website with python. It allows to create redirected based on URL’s but also request data from HTML files. |
| Functools | Functools allows to create authentication for webpages within python |
| Psycopg2 | A tool that allows to connect to POSTGRESQL database in order to send and receive data. |
| Datetime | Allows to get the date and time from the system running the python program |
| Os | Allows to access elements that are stored in the os of the server. For example, the username and password. |
| UrLLib | Allows to request and send data using SQL queries. |
| SSL | Allows to make a connection with the database using a secure socket layer for added encryption |

Word Count: 945