Course: Principles of Database Systems CS-GY 6083 Project: Smart Home Energy Management System (SHEMS) Final Report Semester: Fall 2023

Group members:

Name	Email	ID

GitHub Repository:
https://github.com/Loke7132/PdsFinalProjectShems.git

Introduction

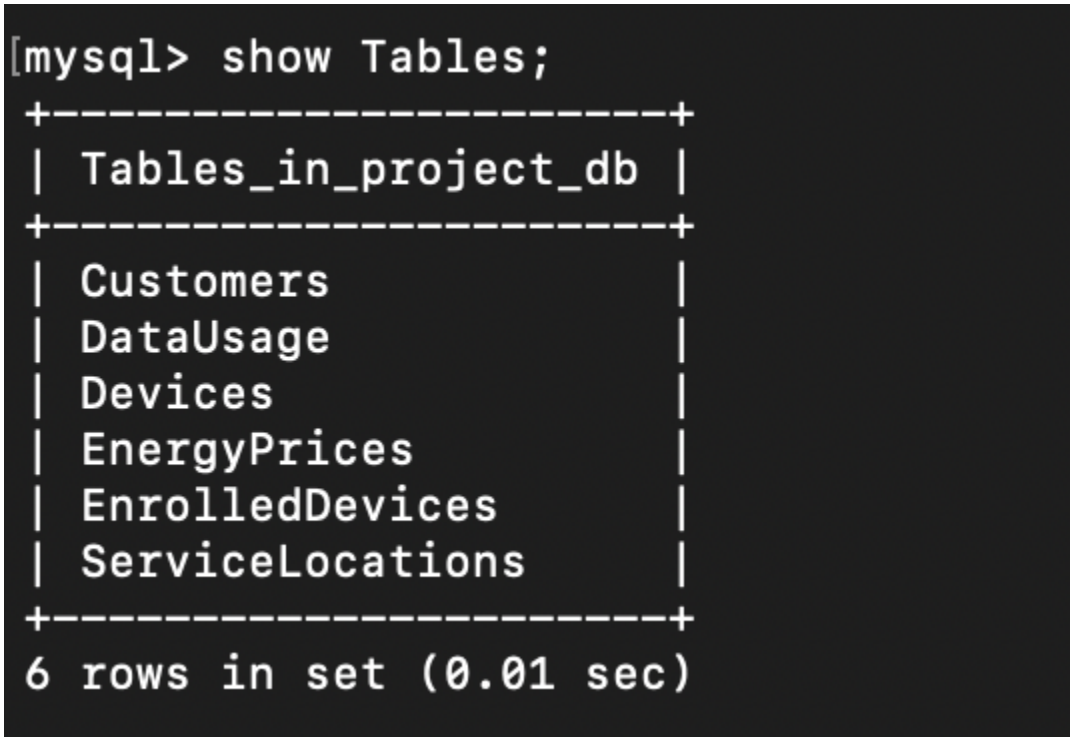
This report concludes the Principles of Database Systems CS-GY 6083 course project, an extension of the initial report. The project focuses on creating the Smart Home Energy Management System (SHEMS) using MySQL, Flask, and React during the fall semester of 2023. The relational schema, developed in the first part, underwent slight modifications to align with final project requirements, including changes to attributes and the addition of a new table. The schema captures relationships between customers, service locations, and devices.

The implementation involved creating a website using MySQL, Flask, and React to visualize mock data. Raw SQL with cursors was employed in the background, adhering to project requirements, while Flask's Object Relational Mapper (ORM) capabilities were not utilized in this context. The report explores the rationale behind design choices, delves into schema SQL design, and discusses backend and frontend designs. The evolution of the schema design is highlighted, and an Entity-Relationship (ER) model illustrates table relationships with an emphasis on foreign keys.

The platform's backend was implemented using Flask, responding to user requests, and MySQL was used to explore the data. Energy data and prices were generated using Python scripts within specified constraints. ReCharts in React handled charting capabilities on the frontend. Cursors were employed for SQL queries, and tables were implemented with constraints, foreign keys, cascading, and trigger capabilities to ensure data integrity.

The report includes screenshots showcasing the platform's capabilities, design code for protection against SQL injection, and details on methodologies, scripts, and testing data. The project aims to demonstrate the practical application of class concepts, meeting technical requirements for a real-world smart home energy management context.

Tables



Customers:

To ensure unique identification for every customer, a dedicated table is employed, featuring a distinct ID for each customer. For ease of search and sorting, the name field is divided into two segments. The AddressNoSt field stores details like apartment number and street name, while the Zipcode, City, and State are stored separately, enhancing the efficiency of search and sorting operations. While CID is a distinct key created to uniquely identify each record within the Customers table and FirstName stores the Customer's initial name. LastName stores the Customer's Last name and Email stores the Customer's Email address.The **Primary Key** is **CID** and there is no **Foreign Key**.

```
mysql> desc Customers;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| CID         | int         | NO   | PRI | NULL    | auto_increment |
| FirstName   | varchar(255) | YES  |     | NULL    |                |
| LastName    | varchar(255) | YES  |     | NULL    |                |
| Email       | varchar(255) | YES  |     | NULL    |                |
| AddressNoSt | varchar(255) | YES  |     | NULL    |                |
| City        | varchar(255) | YES  |     | NULL    |                |
| State       | varchar(255) | YES  |     | NULL    |                |
| ZipCode     | varchar(10)  | YES  |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
8 rows in set (0.01 sec)
```

ServiceLocations:

The necessity for storing user service locations led to the creation of this table. To establish a connection between the customer and location tables, we utilized the CID attribute as a reference. Mirroring the customer table, we divided the address field into four distinct components: AddressNoSt, Zipcode, State, and City. The Zipcode, in particular, plays a crucial role in determining the power price within the specified region. Additionally, SID is a distinct key created to uniquely identify each record within the ServiceLocations table. DateOfTakeover is the date when the customer got control on the location. Bedrooms is the number of bedrooms in the property. Occupants stores number of occupants in the property. **Primary Key** is **SID** and **Foreign Key is CID.**

```
mysql> desc ServiceLocations;
+----------------+-------------+------+-----+---------+----------------+
| Field          | Type        | Null | Key | Default | Extra          |
+----------------+-------------+------+-----+---------+----------------+
| SID            | int         | NO   | PRI | NULL    | auto_increment |
| CID            | int         | YES  | MUL | NULL    |                |
| AddressNoSt    | varchar(255) | YES  |     | NULL    |                |
| City           | varchar(255) | YES  |     | NULL    |                |
| State          | varchar(255) | YES  |     | NULL    |                |
| ZipCode        | varchar(10)  | YES  |     | NULL    |                |
| DateOfTakeover | date        | YES  |     | NULL    |                |
| Bedrooms       | int         | YES  |     | NULL    |                |
| SquareFootage  | int         | YES  |     | NULL    |                |
| Occupants      | int         | YES  |     | NULL    |                |
+----------------+-------------+------+-----+---------+----------------+
10 rows in set (0.01 sec)
```

Devices:

Table stores the information related to the model of each device. While the DeviceID is a distinct key created to uniquely identify each record within the table and DeviceName stores the name of the model device. DeviceType stores the type of the model like AC, Refrigerator etc. **Primary Key** is **DeviceID** and **Foreign Key** is none.

```
mysql> desc Devices;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| DeviceID   | int          | NO   | PRI | NULL    | auto_increment |
| DeviceName | varchar(255) | YES  |     | NULL    |                |
| DeviceType | varchar(255) | YES  |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

EnrolledDevices:

The data in this table has every device owned by a user at a designated location. The connection is established through the use of the DeviceID, which is derived from the devices table, providing details about the device model. Additionally, SID references the service location table to retrieve information about the specific location where the device is currently utilized. EnrollmentID is a distinct key created to uniquely identify each record within the table. **Primary Key** is **EnrollmentID** and **Foreign Key** is **DeviceID, SID**.

```
mysql> desc EnrolledDevices;
+--------------+------+------+-----+---------+----------------+
| Field        | Type | Null | Key | Default | Extra          |
+--------------+------+------+-----+---------+----------------+
| EnrollmentID | int  | NO   | PRI | NULL    | auto_increment |
| DeviceID     | int  | YES  | MUL | NULL    |                |
| SID          | int  | YES  | MUL | NULL    |                |
+--------------+------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

DataUsage:

This table records data transmitted by individual smart home devices through events occurring at 15 minutes intervals. The timestamp indicating when the data is transmitted is stored in the UsageTimestamp field, while the energy usage and event details are stored in the EnergyUsed and EventLabel fields, respectively. DataID is a distinct key created to uniquely identify each record within the table. The **Primary Key** is **DataID** and **Foreign Key** is **EnrollmentID.**

```
mysql> desc DataUsage;
+----------------+--------------+------+-----+---------+----------------+
| Field          | Type         | Null | Key | Default | Extra          |
+----------------+--------------+------+-----+---------+----------------+
| DataUsageID    | int          | NO   | PRI | NULL    | auto_increment |
| EnrollmentID   | int          | YES  | MUL | NULL    |                |
| UsageTimestamp | timestamp    | YES  |     | NULL    |                |
| EventLabel     | varchar(255) | YES  |     | NULL    |                |
| EnergyUsed     | decimal(10,2) | YES  |     | NULL    |                |
+----------------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

EnergyPrices:

The information regarding pricing for each location is stored in this table, organized by zip code and time. PriceID is a distinct key created to uniquely identify each record within the table. The **Primary Key i**s **PriceID** and the **Foreign Key** is None**.**

```
mysql> desc EnergyPrices;
+-----------------+--------------+------+-----+---------+----------------+
| Field           | Type         | Null | Key | Default | Extra          |
+-----------------+--------------+------+-----+---------+----------------+
| PriceID         | int          | NO   | PRI | NULL    | auto_increment |
| ZipCode         | varchar(10)  | YES  |     | NULL    |                |
| PricePerkWh     | decimal(10,2) | YES  |     | NULL    |                |
| PricesTimestamp | timestamp    | YES  |     | NULL    |                |
+-----------------+--------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

Efficiency and Normalisation:

The schema has been meticulously designed and fully adheres to BCNF (Boyce-Codd Normal Form) principles. Each table within the schema exhibits a sole dependency, which is solely on the Candidate Key. No interdependencies between attributes or transitive relations are present. Consequently, it can be confidently asserted that the given schema is in BCNF.

Example:

DeviceID → DeviceName, DeviceType

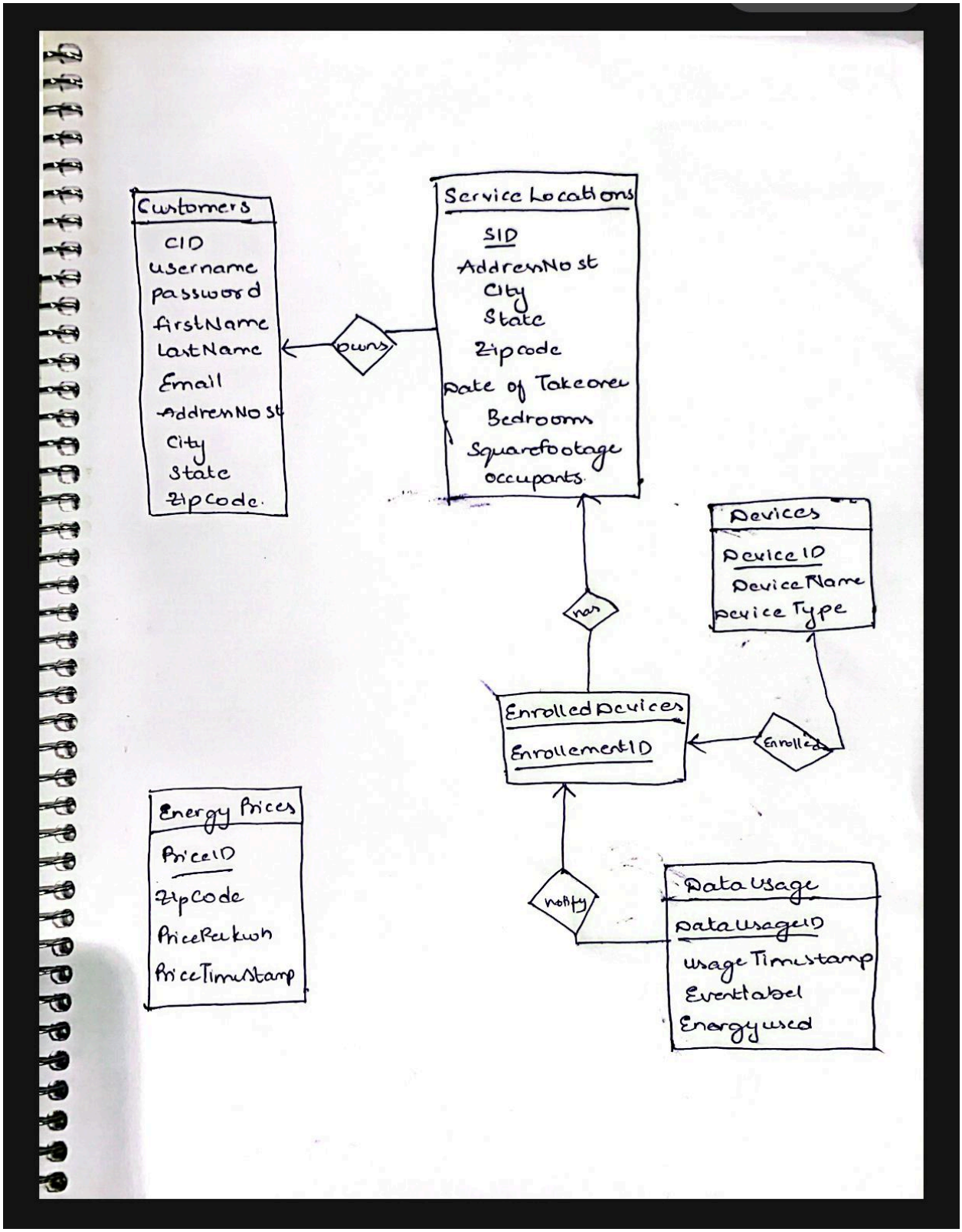
Justification: The primary key (DeviceID) stands as the sole attribute capable of uniquely identifying each record in this table. Both DeviceName and DeviceType are dependent on DeviceID, and no other combinations are relevant for uniquely identifying devices. This identified dependency remains the only plausible one for this table. This analysis can be extended to all tables within the schema, providing conclusive evidence of their adherence to BCNF.

Space Efficiency:

In the case of the "EnrolledDevices" table, where a customer can have multiple devices of the same model, it might appear as if there is a potential for data repetition. However, this situation doesn't constitute redundancy. Since each device operates independently, their respective data is stored separately in the "DataUsage" table, where the "EnrollmentID" serves as a unique identifier. This approach not only avoids redundancy but also maintains the integrity of individual device data. Overall, the design exhibits thoughtful consideration of normalization principles and efficiently manages data relationships.

Constraints

- Primary Key Constraints:
    - Customers Table: The CID column is the primary key, ensuring each customer has a unique identifier.
    - ServiceLocations Table: The SID column is the primary key, ensuring each service location has a unique identifier.
    - Devices Table: The DeviceID column is the primary key, ensuring each device has a unique identifier.
    - EnrolledDevices Table: The EnrollmentID column is the primary key, ensuring each enrolled device has a unique identifier.
    - DataUsage Table: The DataUsageID column is the primary key, ensuring each data record has a unique identifier.
    - EnergyPrices Table: The PriceID column is the primary key, ensuring each price record has a unique identifier.
- Foreign Key Constraints:
    - ServiceLocations Table: The CID column is a foreign key referencing the CID column in the Customers table, ensuring that each service location is associated with a valid customer.
    - EnrolledDevices Table: The SID column is a foreign key referencing the SID column in the ServiceLocations table, and the DeviceID column is a foreign key referencing the DeviceID column in the Devices table. This ensures that each enrolled device is associated with a valid service location and a valid device.
    - DataUsage Table: The EnrollmentID column is a foreign key referencing the EnrollmentID column in the EnrolledDevices table, ensuring that each data record is associated with a valid enrolled device.
- Unique Constraints:
    - Customers Table: The Email column may have a unique constraint to ensure that each email is unique among customers.
    - Devices Table: The combination of DeviceType and DeviceName columns may have a unique constraint to ensure that each device type and Device name pair is unique.

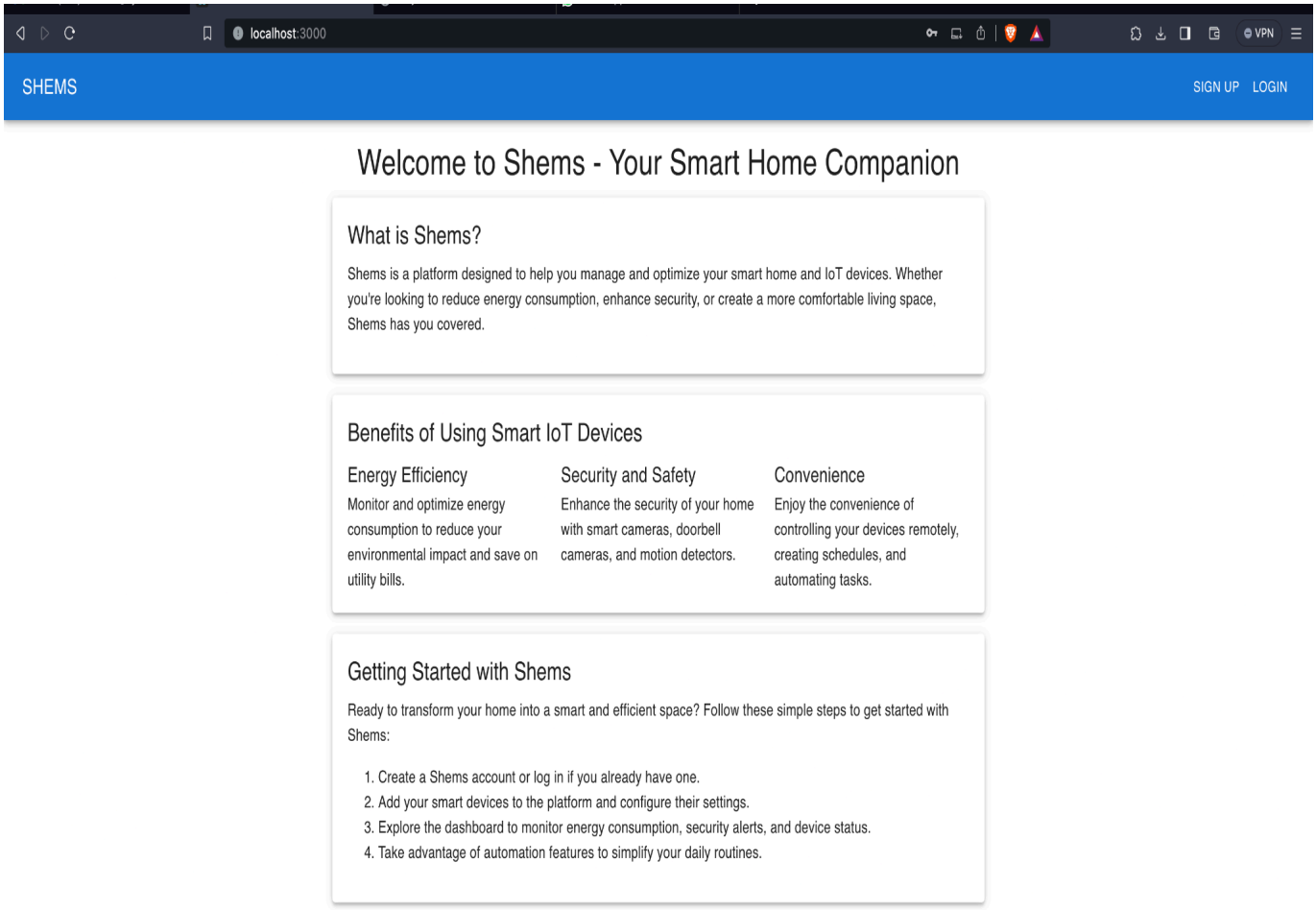


Exploration of the Platform:

This online Smart Home Energy Management System functions as an inclusive dashboard, providing customers with valuable insights to inform decisions regarding energy consumption. Enabling seamless connectivity across multiple service locations, each accommodating a diverse range of devices, the system meticulously records model types, descriptions, and categories of these devices within its database.

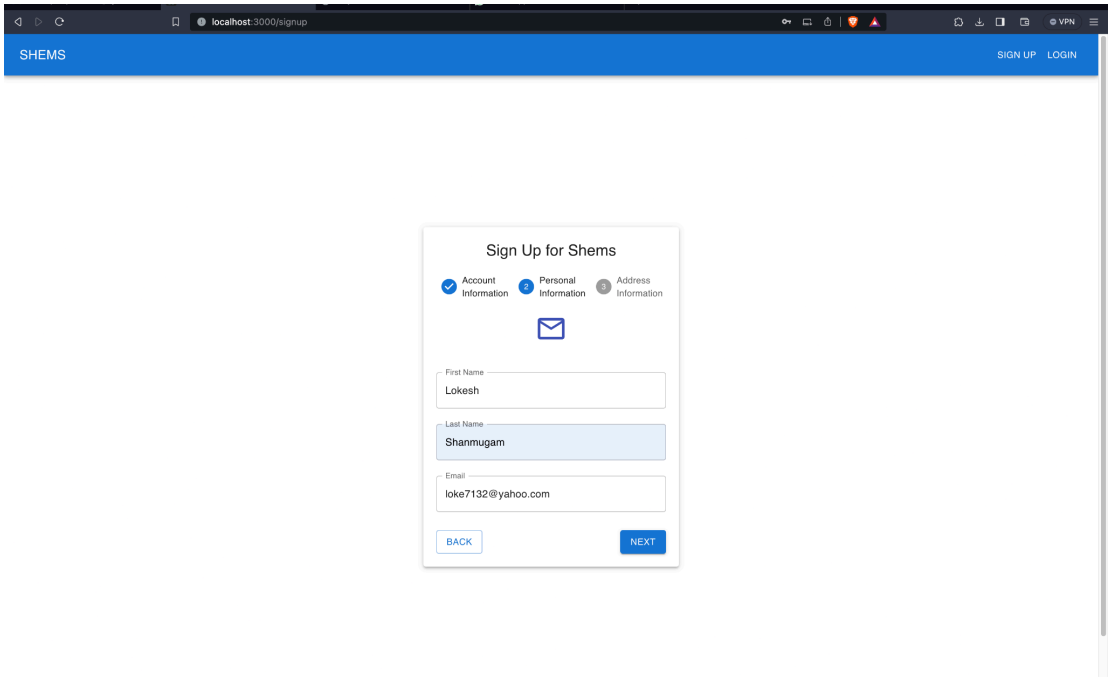
Noteworthy for translating intricate energy consumption data into visually intuitive representations, the platform features dynamic visuals with interactive capabilities. Users can manipulate data in real-time by toggling device visibility on graphs, offering the flexibility to instantly hide or reveal specific devices. Through this platform, customers can easily register, add service locations, integrate various smart devices, and navigate graphical representations of their energy consumption data for informed decision-making.

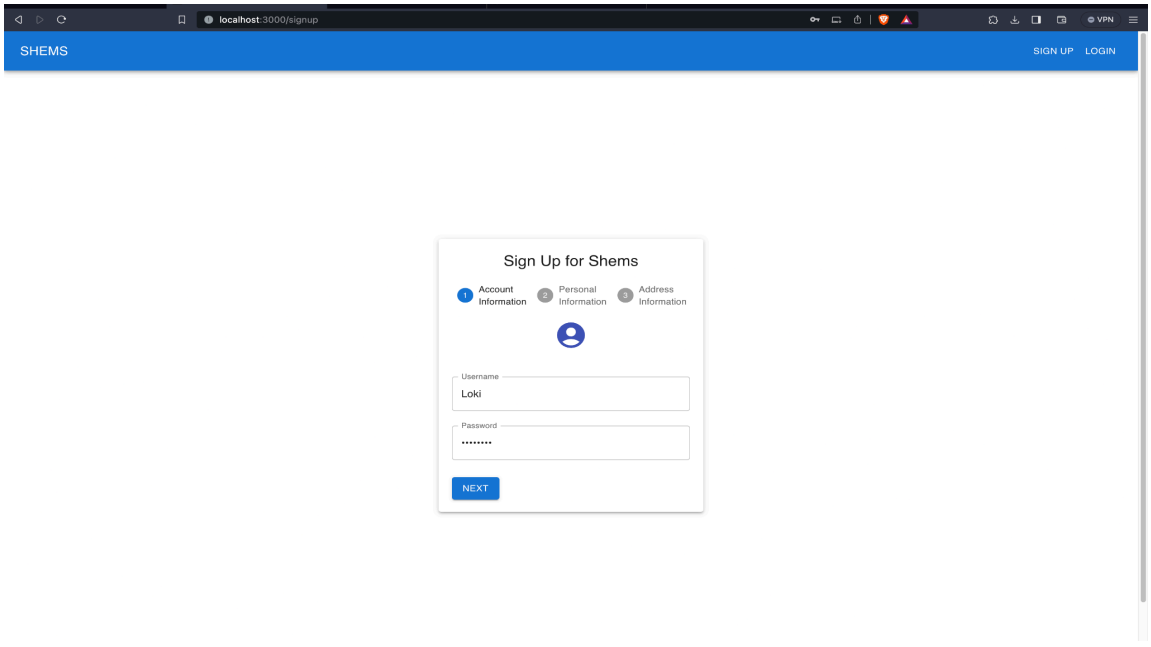
Upon entering the platform, customers are welcomed by a landing page with options to log in or create a new account. The login page includes features such as password recovery, and, notably, credential autofill is incorporated where supported by the browser. In the case of failed credentials, corresponding error messages are displayed; however, for user email protection, specific details about whether the email address is the only incorrect credential are not disclosed.

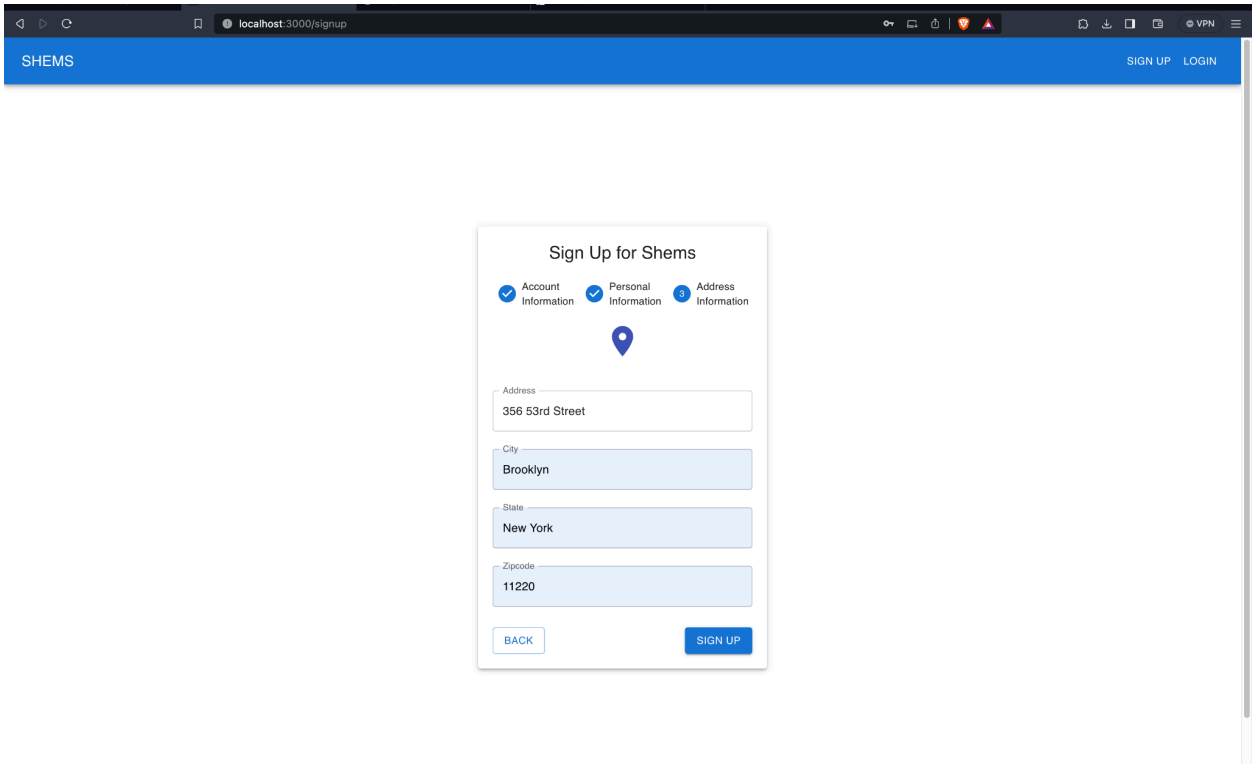


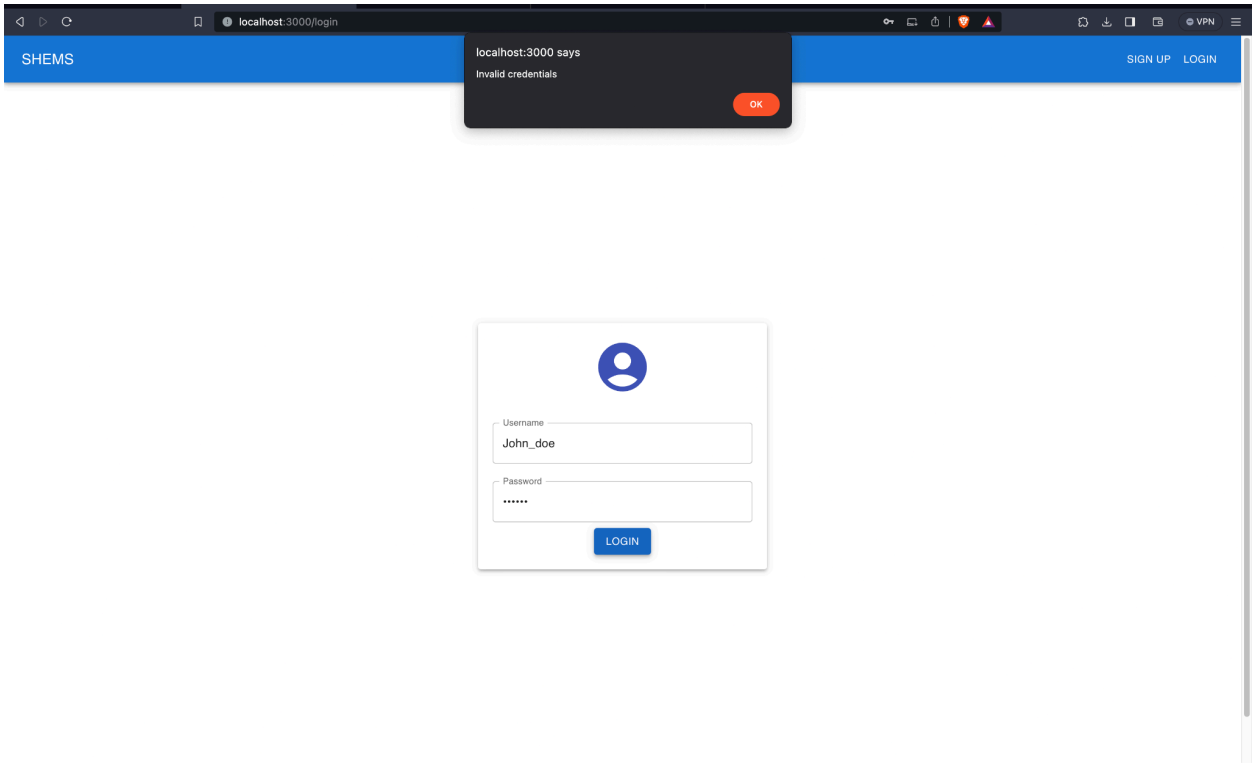
When a new user navigates to the User Registration page through the 'Create a New Account' link, they encounter a form soliciting essential information: Email, Password, Password Confirmation, First Name, Last Name, etc. All data must be input as, by the table definition, no data can be left empty. Upon submitting the form by clicking the 'Submit' button, these fields undergo stringent validation processes to make sure that all data input is the expected data. The data is securely passed as parameters, initiating an insertion process into the customer table through SQL queries.

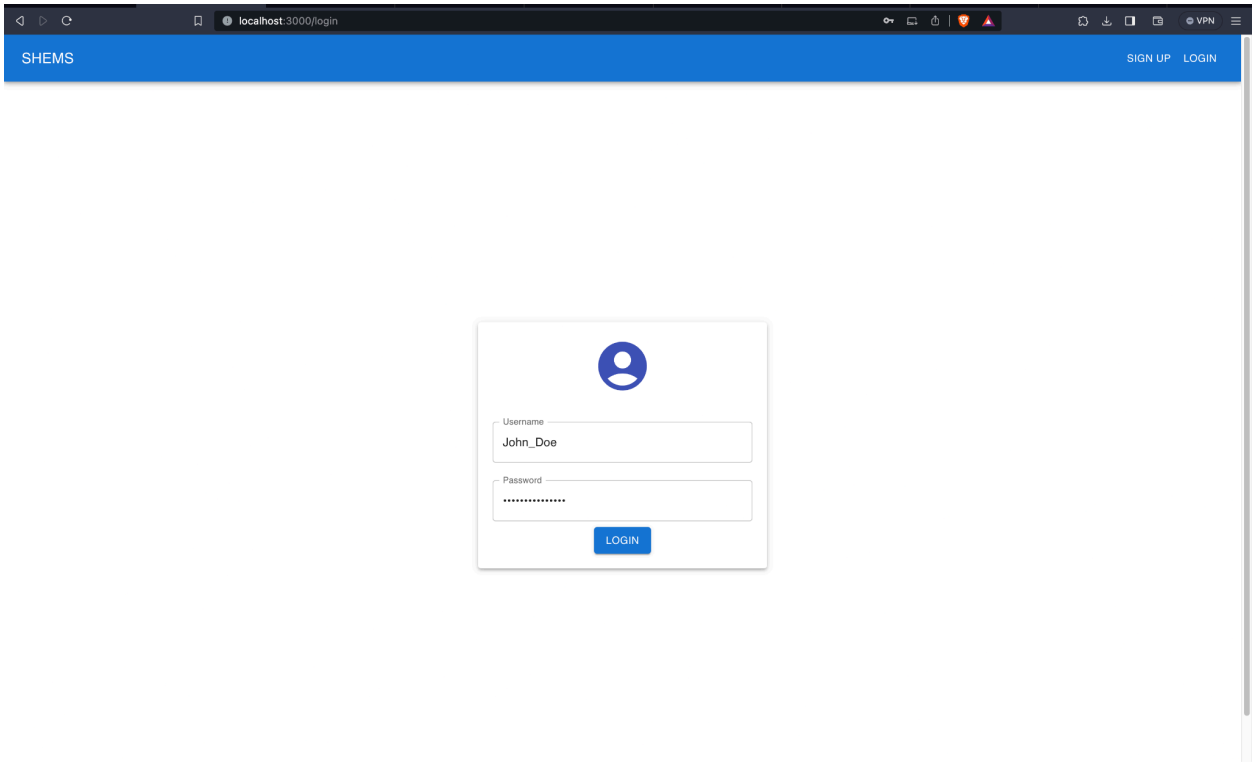
As explained before, we use parameterized queries to safeguard against SQL injections. Any inaccuracies in the entered data type prompt immediate validation errors, signaling users through explicit error messages, ensuring data integrity, and enhancing security measures.



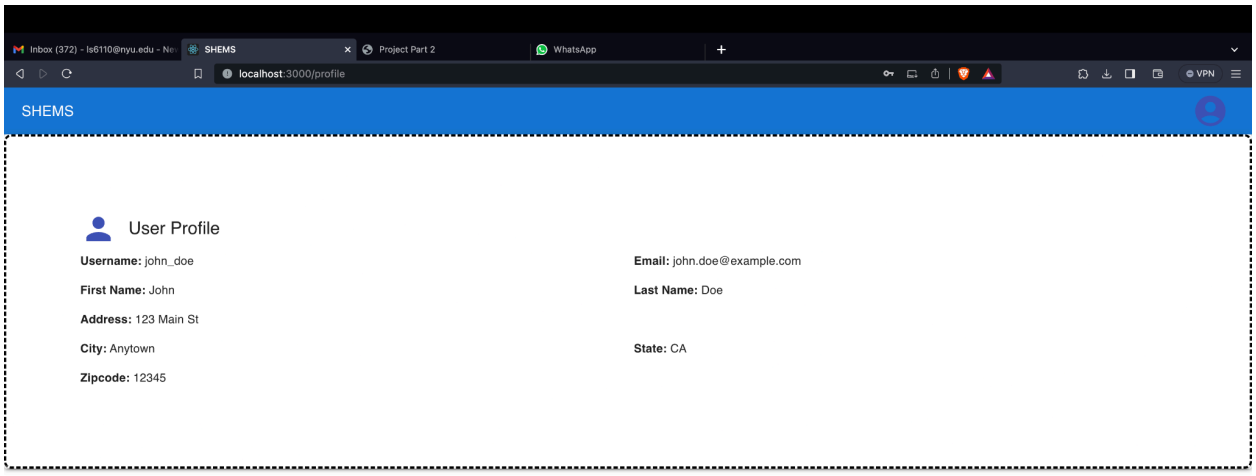


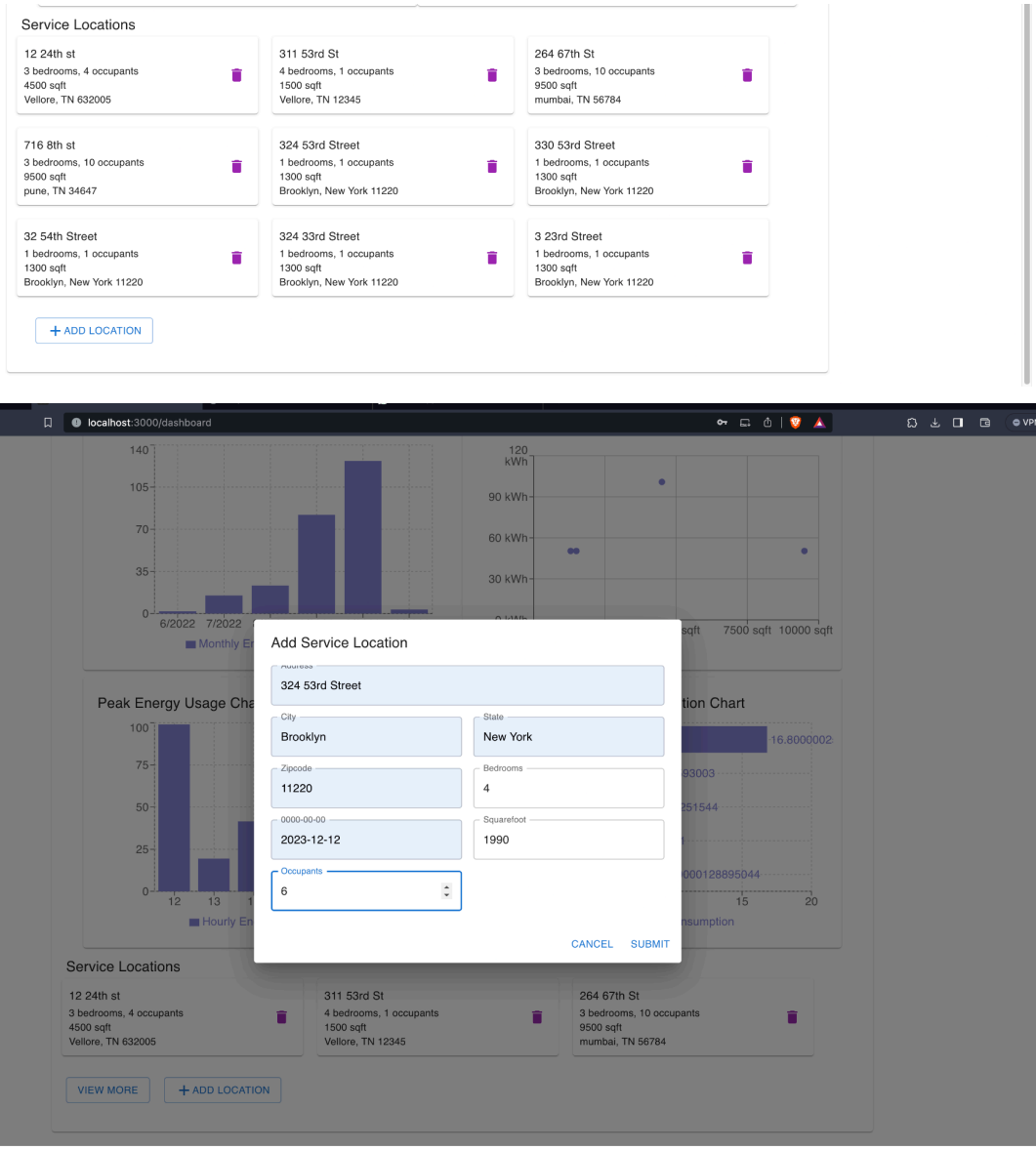

localhost:3000/signup
SHEMS
SIGN UP
LOGIN
Sign Up for Shems
Account Information
Personal Information
3
Address Information
Address
356 53rd Street
City
Brooklyn
State
New York
Zipcode
11220
BACK
SIGN UP


localhost:3000/login
SHEMS
SIGN UP
LOGIN
localhost:3000 says
Invalid credentials
OK
Username
John_doe
Password
LOGIN



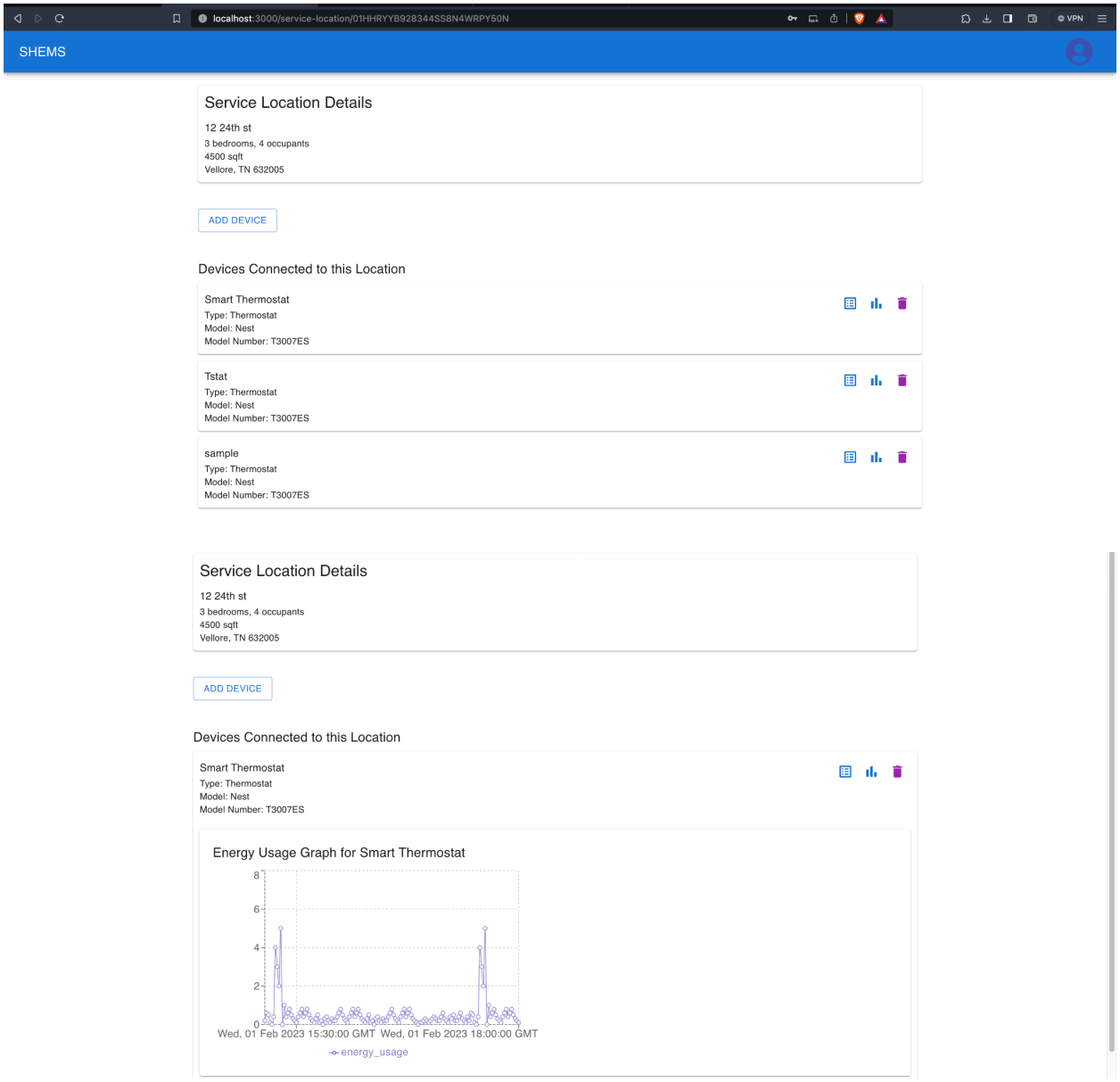
After creating an account, you can incorporate service locations by selecting the 'Add/Delete' option located between the home and logout links. Subsequently, you'll be prompted to input the essential details, all of which are mandatory. These details will then populate an entry in the serviceLocations table within the SQL database. This table is linked to the customer account through a foreign key derived from the session details.

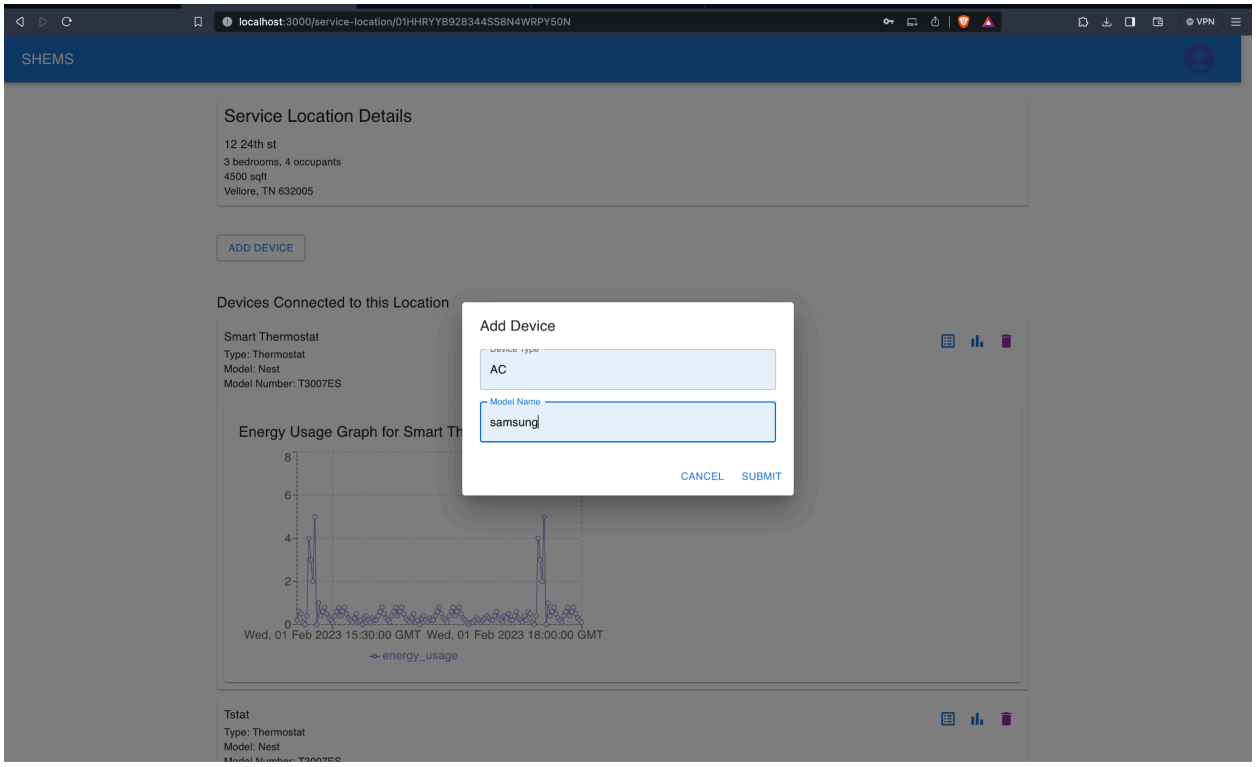




The next step involves adding devices to the list associated with each location, with data sent at 15-minute intervals. The registration page for adding devices resembles other registration pages, although a screenshot is omitted to maintain the report's page limit. The listed devices above will transmit energy data and event data at varying times in 15-minute intervals, which will be recorded in the Energy table and Events table of the database. The provided interface allows for device deactivation; however, it's essential to note that deactivating a device doesn't remove it from the location. Instead, it changes the device's status to

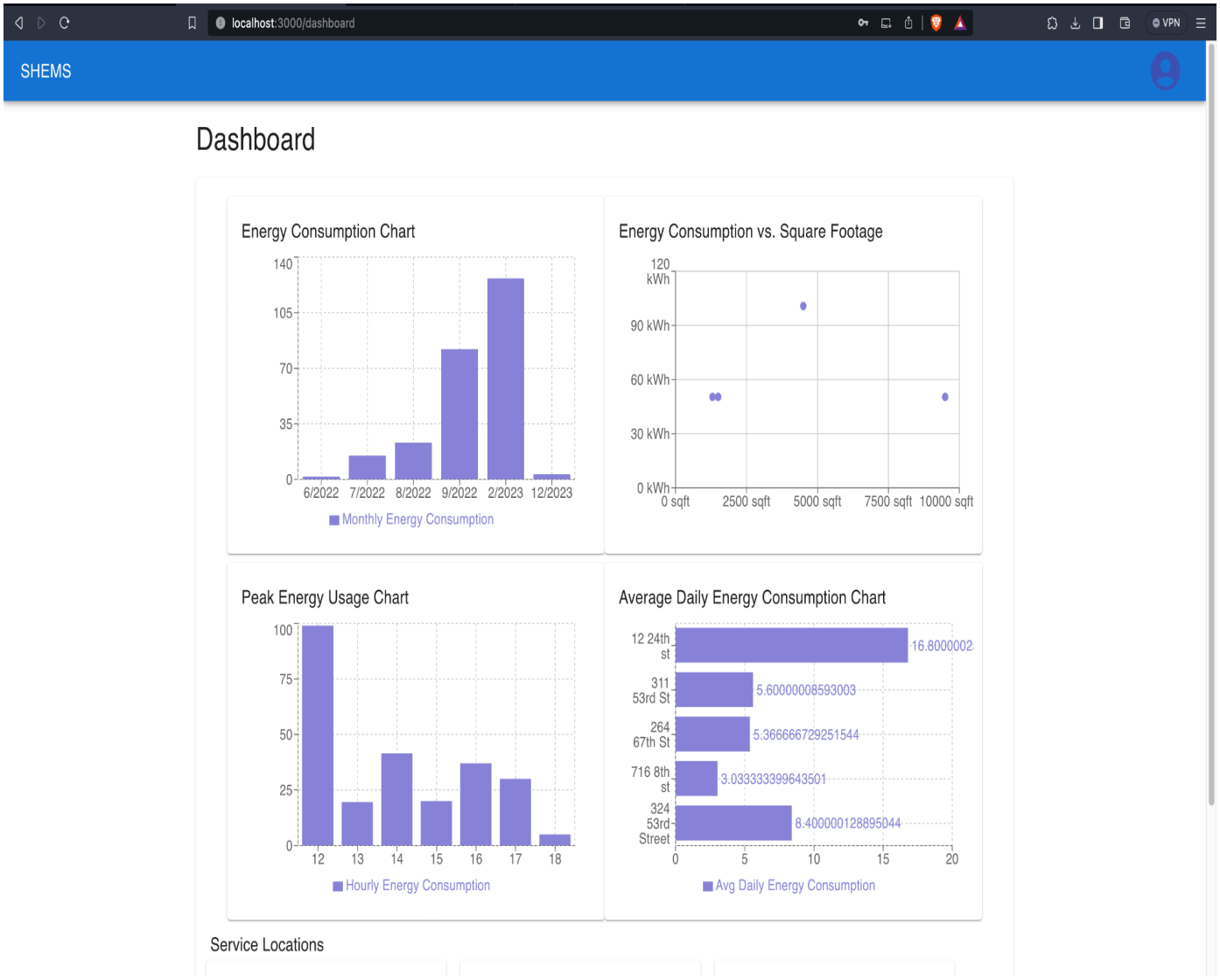
'deactivated.' In real-world scenarios, deactivated devices would cease logging energy data or events, but their data remains included in historical reports.





Displayed above is the homepage you land on when you have service locations and associated devices. We utilize the SQL queries discussed earlier in this report, coupled with ReCharts, to present the Energy Consumption Chart for all locations each month. This chart aids users in comprehending their energy consumption. By default, it illustrates the total energy used across various locations and is interactive.

Beneath the graphs and description, there is a chart depicting Energy Consumption vs. Square Footage, showcasing household energy consumption based on square footage. Additionally, the Peak Energy Usage Chart displays the highest and lowest energy usage, while the Average Daily Energy Consumption Chart illustrates the average energy usage for each location.


localhost:3000/dashboard
VPN
SHEMS
Dashboard
Energy Consumption Chart
140
105
70
35
0
6/2022
7/2022
8/2022
9/2022
2/2023
12/2023
Monthly Energy Consumption
Energy Consumption vs. Square Footage
120 kWh
90 kWh
60 kWh
30 kWh
0 kWh
0 sqft
2500 sqft
5000 sqft
7500 sqft
10000 sqft
Peak Energy Usage Chart
100
75
50
25
0
12
13
14
15
16
17
18
Hourly Energy Consumption
Average Daily Energy Consumption Chart
12 24th st
16.8000002
311 53rd St
5.60000008593003
264 67th St
5.366666729251544
716 8th st
3.033333399643501
324 53rd Street
8.400000128895044
0
5
10
15
20
Avg Daily Energy Consumption
Service Locations

Conclusion:

The system's implementation is a synergy of specialized tools designed for intricate tasks. Flask, along with its Rest Framework, oversees authentication, logic, and template management, while ReChart enhances the platform with graphical representations. Through APIs, users can dynamically update charts, and React templates drive most frontend elements. The refinement of SQL queries optimizes the overall request-response flow, extracting details from sessions or selected locations to ensure precision in data retrieval and interaction handling.

Several specific SQL queries showcase the system's prowess in data retrieval. One query efficiently retrieves device IDs and names while computing individual energy usage over previous months and hours. Another aggregates total energy consumption per service location by merging data from various tables and grouping outcomes based on location identifiers. Concurrency control mechanisms, such as row-level locking during device deletion and atomic transactions for service location additions, fortify data integrity and consistency within the database.

Security is a paramount focus, addressing SQL injection and cross-site scripting attacks. Rigorous data sanitation, type constraints, and parameterized queries counter SQL injection vulnerabilities. Flask output escaping and validation mechanisms combat cross-site scripting threats. Passwords undergo hashing for added security, and CSRF protection is implemented. Despite these measures, the system emphasizes the importance of continuous updates and thorough vulnerability testing to maintain robust security standards.

Functioning as a Smart Home Energy Management System, this platform provides users with a comprehensive dashboard for informed decisions about energy consumption. It streamlines the management of service locations and devices, offering intuitive visualizations through interactive graphs and charts. From initial registration to device addition and energy data visualization, the platform equips users with tools for comprehensive energy usage analysis and management**.**