Table Discription:

1.login_Information:

- ip_address – VARCHAR[45]
- hit_time – TIMESTAMP
- random_number – SMALLINT
- user_key- SMALLINT
- status – BOOLEAN

MAX per row size – 55 byte

Note : user_key is only for internal purpose not for the client use.

The rows in login_information which present in the table for more than 12 hrs should be cleared automatically.

2.user_inforamtion:

- user_key – SMALLINT PRIMAY KEY
- user_id – VARCHAR[50] UNIQUE NOT NULL
- login_password – INT
- company_name – VARCHAR[50]

MAX per row size – 106 byte

3. device_mapping:

- user_key – SMALLINT
- device_id – SMALLINT UNIQUE
- device_name – VARCHAR[20]

MAX per row size – 24 byte

4.modbus_data_(device_id):

- device_id – SMALLINT

- batch_name – VARCHAR[15]
- set_weight - INT
- actual_weight - INT
- total_weight - INT

Work flow:

1.Receiving data from Rasberry pi:

- The data sent from the Raspberry Pi will be in Base64 format; decode the data and store it in the corresponding table based on its device_id in the database.

2.Login Process:

- When a user lands on the login page, a request is sent to the backend along with the user's device IP address. The backend generates a random number, stores it in the login_information table along with the IP address, and returns the random number to the frontend.
- After the user enters their login credentials, the user_id and password are concatenated using "///" as a separator. This combined string is then converted into a sequence of ASCII values. Each ASCII value is modified: the first value is incremented by the random number, the second is decremented by the same number, and this pattern continues alternately. The resulting values are concatenated into a single string.
- This encoded string, along with the user's IP address, is then sent to the backend. The backend uses the IP address to retrieve the corresponding random number from the login_information table, decodes the string to extract the original user_id and password, and verifies the credentials against the records in the user_information table.
- If the login is successful, the backend responds with a JSON object containing the company_name and an array of associated device_names with device_id. The frontend then redirects the user to the dashboard page.

3.Dashboard page:

- The company name received during login should be displayed prominently at the top of the page.
- A dropdown menu should be provided to list all available device_names. When a user selects a device_name, the corresponding device_id is sent to the backend to request data.
- By default, data for the first device in the list should be fetched and displayed initially.
- The backend processes the request by querying the modbus_data_<device_id> table, then returns the result as a JSON response to the frontend.
- The frontend should display the received data in a format that aligns with the client's design and layout specifications.

4.Admin Portal :

4.1 Admin Input & Device ID Validation:

- The admin must enter a device_id in an input field.
- When submitted, the backend attempts to insert the device_id into the device_mapping table, where device_id is unique.
- If a duplicate device_id is detected, an exception is thrown and handled gracefully, returning the message: "The Device ID already exists. Try a different Device ID."

4.2 Table Creation on Success:

- If the device_id is inserted successfully, a new table named modbus_data_<device_id> is created dynamically to store Modbus data specific to that device.

4.3 Device Mapping and User Handling:

- The next step is to map the device_id to a device_name and associate it with a user.
- If the user is new, the user is prompted to enter:
    - Device Name
    - Company Name
- The system automatically generates:
    - user_id using the first 6 letters of the company name (excluding spaces and special characters) + the 5-digit device_id
    - password as a 6-digit device_key
- The following records are updated:
    - user_information table with user_key, user_id, password, and company_name
    - device_mapping table with user_key and device_name for the respective device_id

4.4 Handling Existing Users:

- If the user already exists, the admin can search for them by company name using a search bar.
- Once identified, the new device_id is simply mapped to the existing user by updating the device_mapping table.