SOFE4610 - Assignment 2

Raveenth Maheswaran 100704540
Sabesan Sivakumar 100701928
Owais Quadri 100697281

Question 1

Complete steps 1-8 of the design methodology for your project and submit the results for grading. [10]

Step 1: Purpose and Requirements Specification

Purpose:

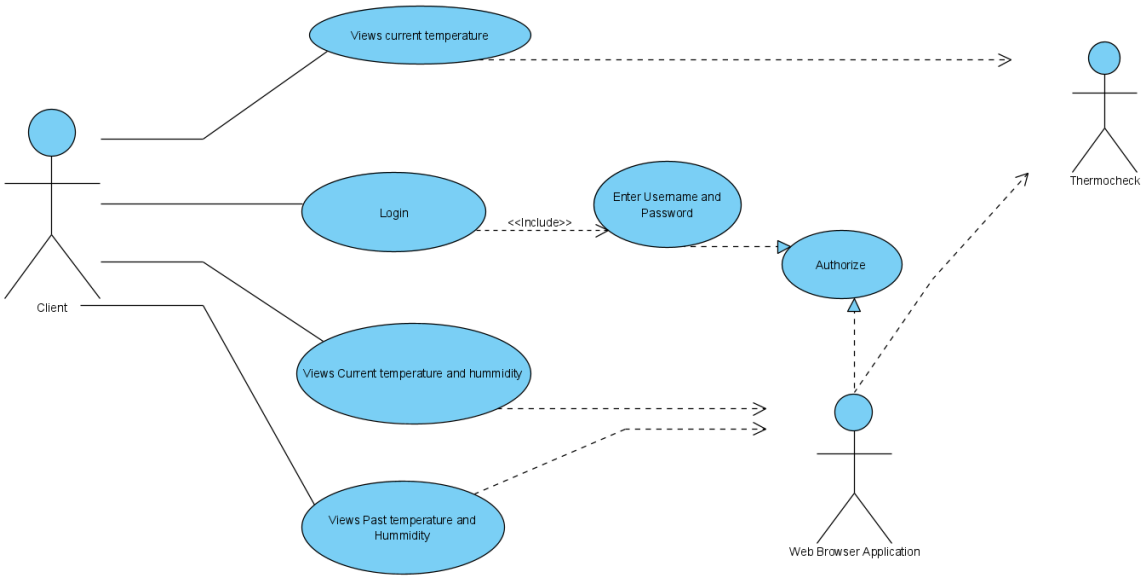
An IoT Thermometer will display the reading of the temperature on an LCD screen and a web browser. This will be accomplished by utilizing the NodeMCU board by connecting it to the temperature and humidity sensor. This sensor will collect the temperature and humidity of the room, and we will program it to display that information on the LCD screen. The NodeMCU board will be powered via USB cable and will be programmed by using Arduino IDE. Appropriate libraries and tools will be used to program the sensor and LCD screen. The values will be shown in real-time and provide accurate results of the temperature and humidity in the room. The user can view previous temperature/humidity values on a web browser application along with the date and time recorded. The software used to collect and store the previous measurement by MySQL and Django python will be used to retrieve the data through Django REST API.

Requirements:

1. Data Collection: Acquire temperature and humidity data from the sensors
2. Data analysis: Analyze the data acquired to ensure that temperature is based on Celsius, and double check data to make sure it is reading correctly
3. System management: Ensure that the connection between the NodeMCU and the temperature/humidity sensor is strong. Also, ensure that the connection between the IoT system and the web browser application is strong as well.
4. Data privacy and security: Users who access through a web browser would need to access the information with the password.
5. User interface requirements: User-friendly application that effectively provides previously saved temperature and humidity sensors

Step 2: Process Specification

Use Case	Description
UC#1	Users see the current temperature by looking at the LCD screen. The temperature/humidity sensor will be connected to the NODEMCU board which will be programed to display the current temperature value onto the LCD screen
UC#2	User looks at the current temperature and humidity levels through a web browser application. The temperature sensor will be connected to the NODEMCU board which will be connected to a server that will be connected to our web browser application. The user would have to log in through the web browser and will be able to see the current humidity and temperature.
UC#3	Users see the previous temperature and humidity levels. The temperature sensor will be connected to the NODEMCU board which will be connected to a server that is connected to a database that will store all the previous temperature and humidity levels of the past 24 hours. The user would have to log in through the web browser application and will be able to see the current humidity and temperature and the previous humidity and temperature.



Step 3: Domain Model Specification
Main concepts:

Entities:
Thermostat:
- Addr : String
- currTemp : float
- currHumid : float

Server:
- Addr : String
- DBpass : String
- currTemp : float
- currHumid : float

WebBrowserApplicationClient:
- clientID : String
- Addr : String
- currTemp : float
- currHumid : float

Database:
- psswrd : String
- TempTable : ArrayList
- currTime : datetime

Relationships:
- Thermostat sends data to Server
- Server stores data in Database
- Server sends data to WebBrowserApplicationClient

Step 4: Information Model Specification
Database is the virtual entity:

Temperature Relation

time	temp	humidity

Step 5: Service Specifications

Service inputs : Temperature/ Humidity Sensor will be retrieving the environment current temperature and humidity levels

Service Output:

LCD Screen: The LCD Screen will be connected to the NODEMCU board and display the current temperature every 5 seconds.

Web Browser Application: It will display both the current and previous temperature and humidity levels.

Service preconditions: Temperature/Humidity sensor connected to NODEMCU with stable and correct readings

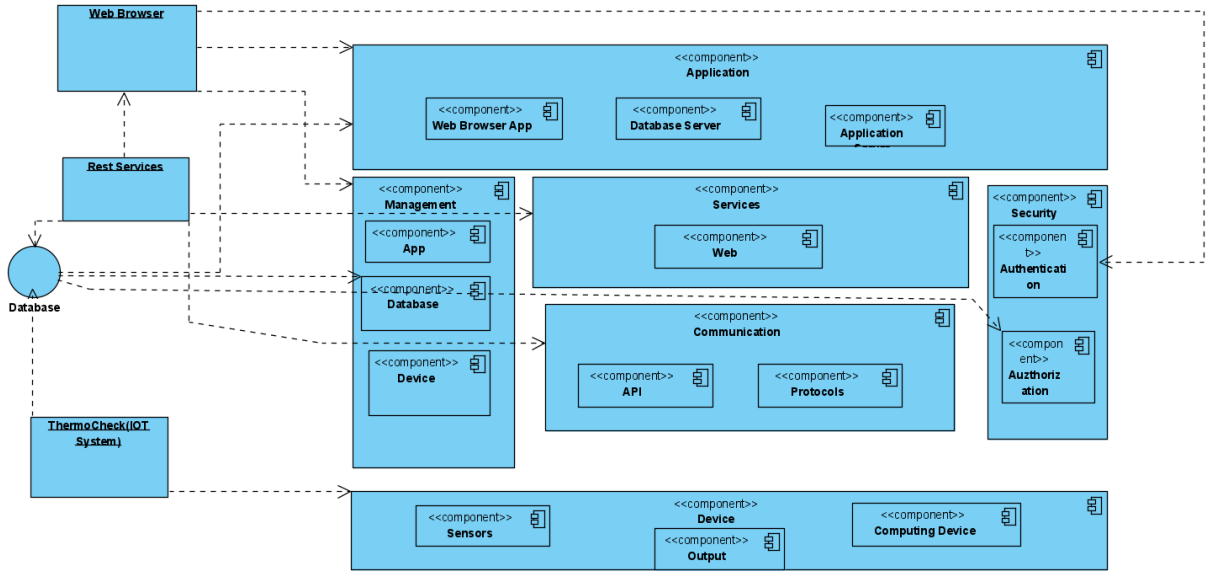
NodeMCU connecting to Arduino with a stable and reliable internet connection

Adrunio connected to a web browser application to showcase current and previous temperature and humidity levels.

Step 6: IoT Level Specification

IoT Level 2 - Our system is a level 2 IoT system. We have both local and cloud layers that are necessary to run our project. The local layer will include the LCD screen as there is no need to have access to cloud layers as the LCD only needs values from the sensor. The cloud layer would consist of our web browser displaying the current temperature and humidity and storing the past 24 hours' temperature and humidity through the MySQL database. It's a single node that performs sensing via the temperature sensor and is stored on the cloud, which can then be accessed by the web browser. In order for users to access their web browser application, they will need to login in order to ensure user security, trust and privacy. This is not computationally intensive, and not much analysis is needed.

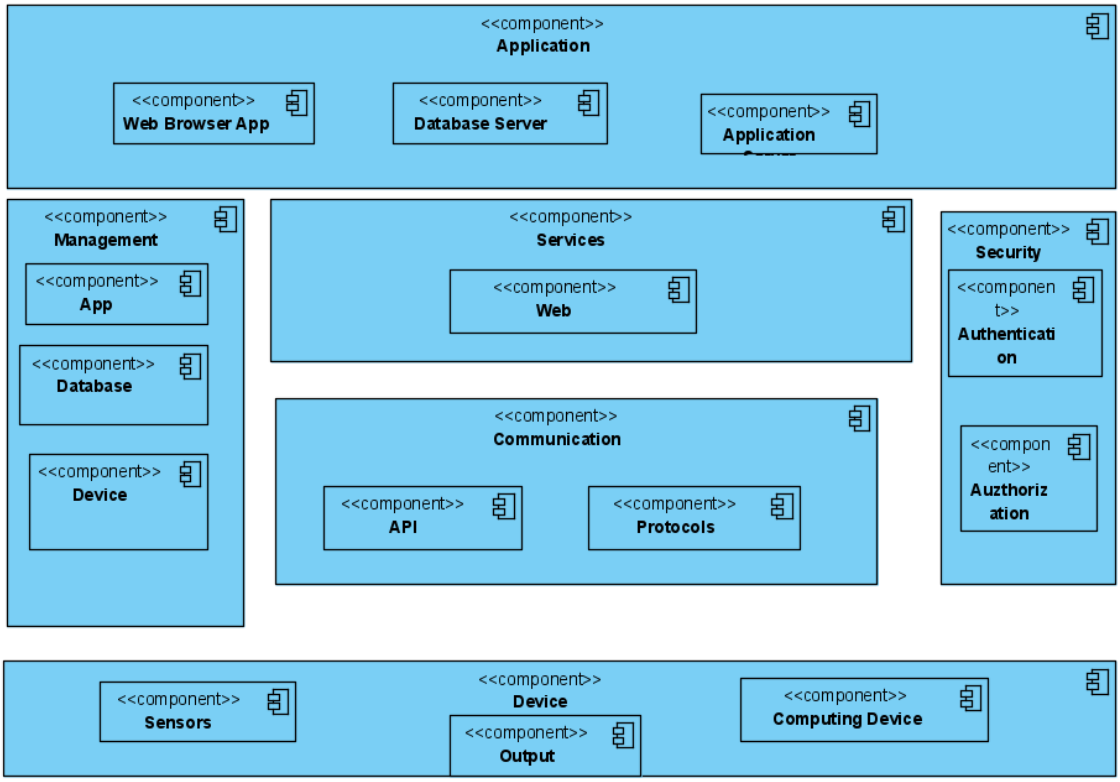
Step 7: Functional View Specification



Below is a table specifying the ThermoCheck devices and its description

IoT System	Description
DHT11 Temperature and Humidity Sensor	Collects the current temperature data needed for our web application
NodeMCU Board	Communicates with the sensor and computer to collect and transfer the data
Workstation PC	Communicates with the NodeMCU that will collect the data and paste it onto our own application

Step 8: Operational View Specification



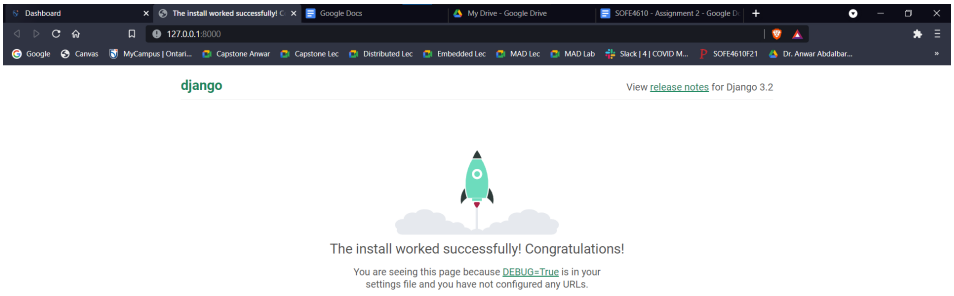
Layer	Description
1 - Application	Web App: Django Web App Application Server: Django App Server Database Server: MySQL
2. Services	Web Service: Mode REST Service, State REST Service
2/3. Security, Management	Authentication Web App, Database Authorization: Web App, Database Application Management: Django App Management Database Management: MySQL Database Device Management: NODEMCU
3. Communication	API: REST API's Protocols: - Network Layer= IPv4/IPv6 - Transport: TCP - Application:HTTP - Link Layer:802.11

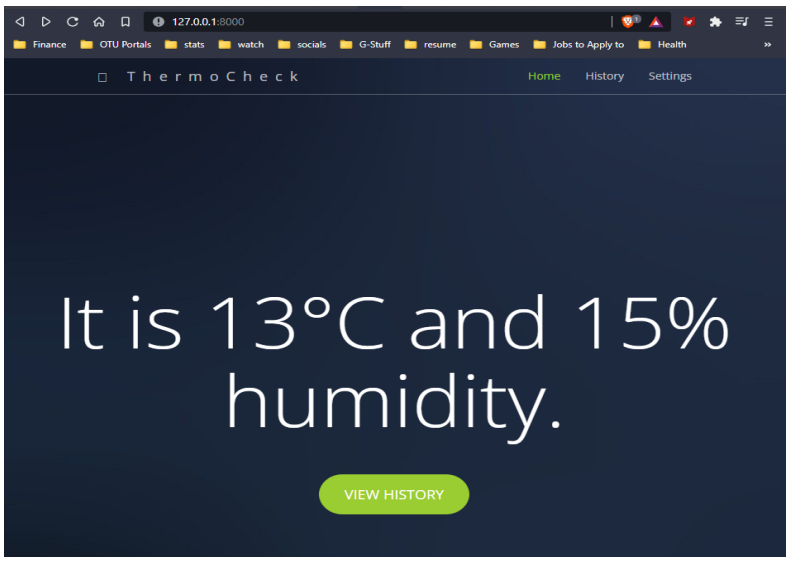
4.Device	Computing Device: WorkStation PC Sensor: DHT11 temperature and Humidity Output: LCD Screen

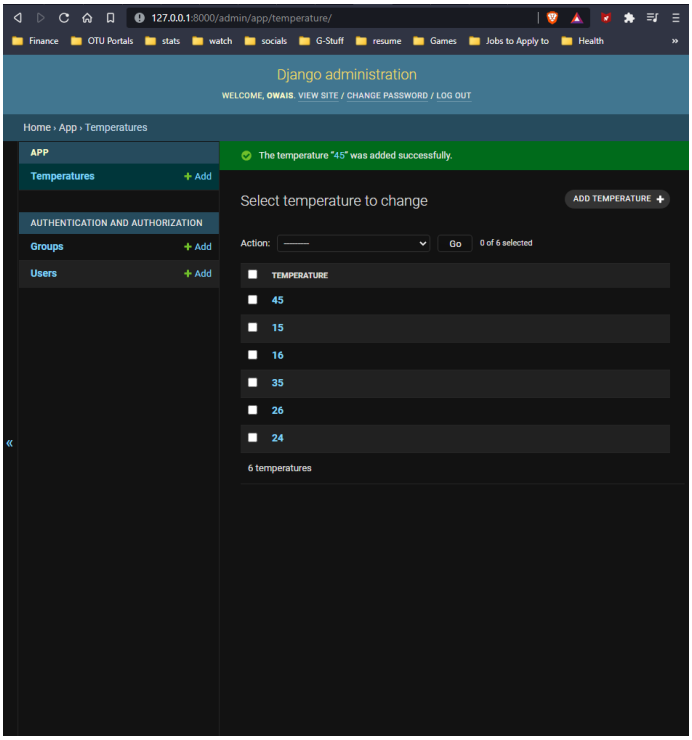
Question 2

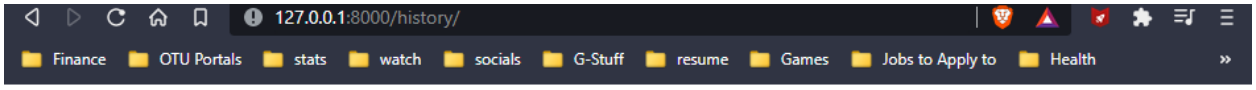
Implement your design using the Django REST framework. Submit a GitHub link of the code and tests for the implementation. In the README file create a table of contents to help navigate your code and show that your implementation performs the functionality if the use case is picked for design.

GitHub link: https://github.com/raveen15/IoTAssignment2/









History

[OrderedDict([('id', 1), ('temperature', '24'), ('humidity', '20'), ('date', '2021-11-01T00:23:10.841483Z')]), OrderedDict([('id', 2), ('temperature', '26'), ('humidity', '0'), ('date', '2021-11-01T00:40:42.605236Z')]), OrderedDict([('id', 3), ('temperature', '35'), ('humidity', '35'), ('date', '2021-11-01T02:09:28.439025Z')]), OrderedDict([('id', 5), ('temperature', '16'), ('humidity', '10'), ('date', '2021-11-01T02:47:51.647594Z')]), OrderedDict([('id', 6), ('temperature', '15'), ('humidity', '15'), ('date', '2021-11-01T03:23:10.487274Z')]), OrderedDict([('id', 7), ('temperature', '45'), ('humidity', '34'), ('date', '2021-11-01T03:35:22.253274Z')]), OrderedDict([('id', 8), ('temperature', '13'), ('humidity', '15'), ('date', '2021-11-01T03:52:31.100898Z')])]