



Course: PA1484

Date: 2025-11-9

Place: Blekinge Tekniska Högskola (campus)

Participants:

- Hampus Ek
- Kaiser Ahmad Samim
- Max Bacharach
- My Lundblad
- Robert Schrewelius

Project Contributions & Work Log (Week 45)

- “Weather Data Visualization Project”

Weekly Summary

This week we focused on implementing the SMHI API data fetching and setting up the ESP32 environment in VSCode with PlatformIO. The team has been working on the refined requirements as the Project Plan 1 and 2 have already been handed in, and the boot screen already implemented by Hampus.

Team Members, Roles, & Responsibilities

The group assigned the following roles and responsibilities:

Name	Role	Responsibilities
Hampus Ek	Developer	Maintain code quality and consistency. Ensure the GitHub project is up to date. Validate pull/merge actions, with Robert. Responsible for the touchscreen UI and data visualization.
Kaiser Ahmad Samim	ESP Handler & Code Tester	Responsible for ESP32 hardware availability in labs, performs stress testing of the code as well as bug tracking, verification, and report results. Responsible for Historical Weather Data (Upcoming Week).
Max Bacharach	Project overview	Keep the group on track with deadlines and ensure focus on deliverables. Work with the secretary on documentation. Responsible for Historical Weather Data (Month).
My Lundblad	Documentation	Record formal meeting notes and project contributions. Ensure information reaches all members of the group. Assists the chairman with keeping track of deadlines, and agendas.

		Responsible for Weather Forecast Data (Today).
Robert Schrewelius	Developer	Maintain code quality and consistency. Ensure the GitHub project is up to date. Validate pull/merge actions, with Hampus. Responsible for the API SMHI.

- **All members** Implement functionality according to requirements, in collaboration with managers and tester.

Note!

All members are responsible and will participate, but the roles are assigned in order to ensure that at least one person has the final responsibility to make sure that specific sections run smoothly.

Individual Contributions

Name	Main Task This Week	Comments
Hampus	<p>Looked over the graph, adding time and temperature to be displayed in the graph for readability, in order to see what the data points mean.</p> <p>The time data is retrieved from the API.</p> <p>Currently working on changing the hardcoded temperature.</p> <p>Formatting the graph – to be more readable.</p> <p>Placing the tiles in the right order: Boot screen (1), Today's forecast (2), Upcoming Week (3), Month (4), Setting (5).</p>	
Kaiser	<p>Started on the hpp-file "upcomingWeek" where he continues to work on setting up the tile.</p> <p>Created function to call API to get forecast.</p>	The function to call API to get forecast will be moved over to the class "SMHI_API".
Max	Created a tile to draw up a graph and added the function to slide along the x-axis	

	<p>- with the intension to display days on the x-axis and temperature on the y-axis.</p> <p>Worked on SMHI API to apply for tile with month, making it possible to fetch weather data from SMHI API for the tile.</p> <p>Created a tile for the settings function so the user can change the city and variables such as temperature and wind.</p>	
My	<p>Create the “Project Contributions” document.</p> <p>Started on researching on setting up the Weather Forecast Data (Today). Looking over different code examples – reading about them in depth and weighing pros and cons before deciding on approach.</p>	
Robert	<p>Built the API for SMHI och moved code from Max into the class “SMHI_API”.</p>	

Images of proof (voluntarily)

```
const char* param_codes[] = {"1","2","3"}; // 1=Temp, 2=Nederbörd, 3=Vind
You, 21 hours ago | 1 author (You)
struct DataPoint {
    String date;
    float temp;
};

You, 21 hours ago | 1 author (You)
struct Station {
    const char* name;
    const char* id;
};
Station stations[] = {
    {"Karlskrona", "65020"},
    {"Stockholm", "98200"},
    {"Göteborg", "97400"}
};
```

Image 1 – Data structures by Max.

```

static std::vector<DataPoint> weatherData;
You, 21 hours ago | 1 author (You)
class SMHI_API {
public:
    // Constructor
    SMHI_API(const char* apiUrl);
    bool CONNECT();
    // Fetch weather data from SMHI API
    void update_weather_data(int city_idx, int param_idx, String period);
    //Parse the weather data
    void parseWeatherData(const String& jsonData);
private:
    //URL to SMHI API
    const char* apiUrl;
    //Vector for weather data
};

SMHI_API::SMHI_API(const char* apiUrl) : apiUrl(apiUrl) {}

```

Image 2 – Class declaration and constructor by Robert.

```

void SMHI_API::update_weather_data(int city_idx, int param_idx, String period){
    String url=apiUrl;
    url+=param_codes[param_idx];
    url+="/station/";
    url+=stations[city_idx].id;
    url+="/period/"+ period +"/data.json";

    Serial.printf("Fetching data: %s\n",url.c_str());

    WiFiClientSecure client;
    client.setInsecure();

    HTTPClient https;
    if(https.begin(client,url)){
        int httpCode = https.GET();
        if(httpCode==200){
            String payload = https.getString();
            parseWeatherData(payload);
            Serial.println("Weather data fetched OK.");
        } else {
            Serial.printf("HTTP GET failed, code: %d\n",httpCode);
        }
        https.end();
    } else {
        Serial.println("Failed to begin HTTPS request");
    }
}

```

Image 3 - Class function "SMHI_API" update_weather_data by Max.

```

void SMHI_API::parseWeatherData(const String& jsonData){
    DynamicJsonDocument doc(60000);
    DeserializationError err = deserializeJson(doc,jsonData);
    if(err){
        Serial.print("JSON parse error: ");
        Serial.println(err.c_str());
        return;
    }

    weatherData.clear();

    if(doc.containsKey("value")){
        JsonArray arr = doc["value"];
        for(JsonObject v: arr){
            DataPoint dp;
            dp.date = v["date"].as<String>().substring(0,10);
            dp.temp = v["value"].as<float>();
            weatherData.push_back(dp);
        }
    } else if(doc.containsKey("timeSeries")){
        JsonArray ts = doc["timeSeries"];
        for(JsonObject rec : ts){
            String t = rec["validTime"].as<String>();
            JsonArray pars = rec["parameters"];
            for(JsonObject p : pars){
                if(String(p["name"].as<const char*>())=="t"){
                    float val = p["values"][0].as<float>();
                    DataPoint dp;
                    dp.date = t.substring(0,10);
                    dp.temp = val;
                    weatherData.push_back(dp);
                    break;
                }
            }
        }
    }

    } else {
        Serial.println("No known JSON key for data array!");
        return;
    }

    Serial.printf("Parsed %d datapoints.\n",weatherData.size());
}

```

Image 4 & 5 – Function to parse weather data by Max.

Next Two Week's Plan

Name	Task
Hampus	<ol style="list-style-type: none">Work on tile five (settings): Change so more cities than three can be added.Work on tile four (the graph): to show days instead of hours on the x-axis and show the temperature on the y-axis.
Kaiser	<ol style="list-style-type: none">Work on tile three to make it work (upcoming week).
Max	<ol style="list-style-type: none">Work on tile five (settings): Change so more cities than three can be added.Work on tile four (the graph): to show days instead of hours on the x-axis and show the temperature on the y-axis.
My	<ol style="list-style-type: none">Work on tile two to make it work (today's forecast).Documentation (Contributions)
Robert	<ol style="list-style-type: none">Look over the 'SMHI_API' class.Add finished functionality to the 'SMHI_API' class.Support where needed as code manager.