Weather Analytics Application

Cloud Computing and Big Data Analytics

Master in Data Science - FIB, UPC

Team members:

Miona Dimic, Mateusz Jerzy Galinski, Poly Kinya, Odysseas Kyparissis, Joan Oliveras

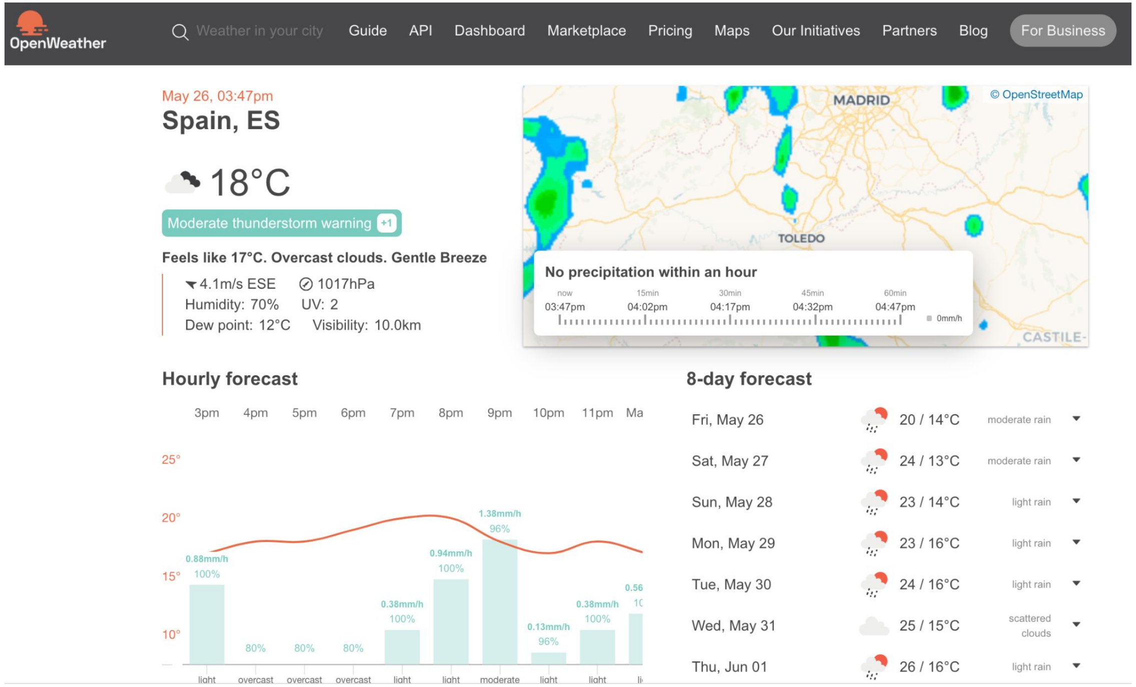
26/05/2023

Outline

- Description and Main Goal of the Project
- Implementation Pipeline
  - Initial Proposed Design
  - Final Design
- AWS Lambda for Historical Data Retrieval
- ETL Process
- Dashboarding
- Django and Elastic Beanstalk (EBS)
- Benefits of Django
- Benefits of Elastic Beanstalk
- Task Allocation and Time Management Tool

Description and Main Goal of the Project

- Weather Data
- Weather Analytics Dashboard
- Future work:
  - Notification system
  - Location specific alert system



May 26, 03:47pm

Spain, ES

18°C

Moderate thunderstorm warning +1

Feels like 17°C. Overcast clouds. Gentle Breeze

4.1m/s ESE 1017hPa
Humidity: 70% UV: 2
Dew point: 12°C Visibility: 10.0km

No precipitation within an hour

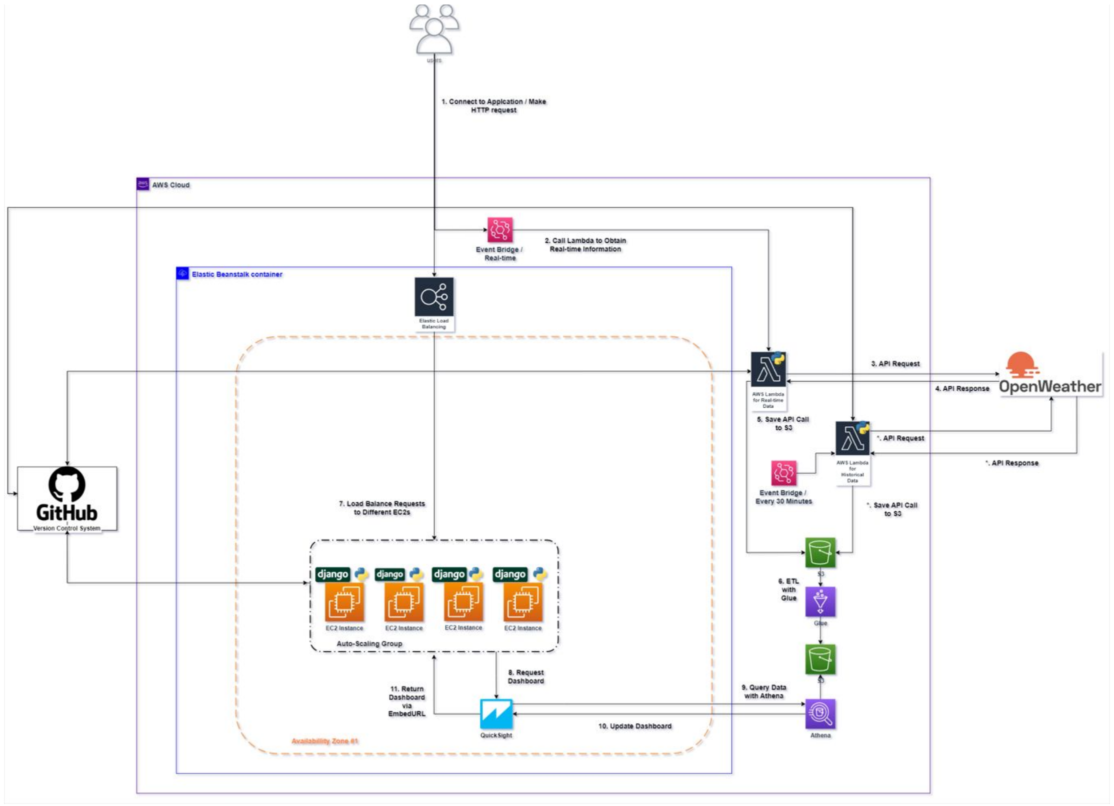
Hourly forecast

8-day forecast

Fri, May 26	20 / 14°C	moderate rain
Sat, May 27	24 / 13°C	moderate rain
Sun, May 28	23 / 14°C	light rain
Mon, May 29	23 / 16°C	light rain
Tue, May 30	24 / 16°C	light rain
Wed, May 31	25 / 15°C	scattered clouds
Thu, Jun 01	26 / 16°C	light rain

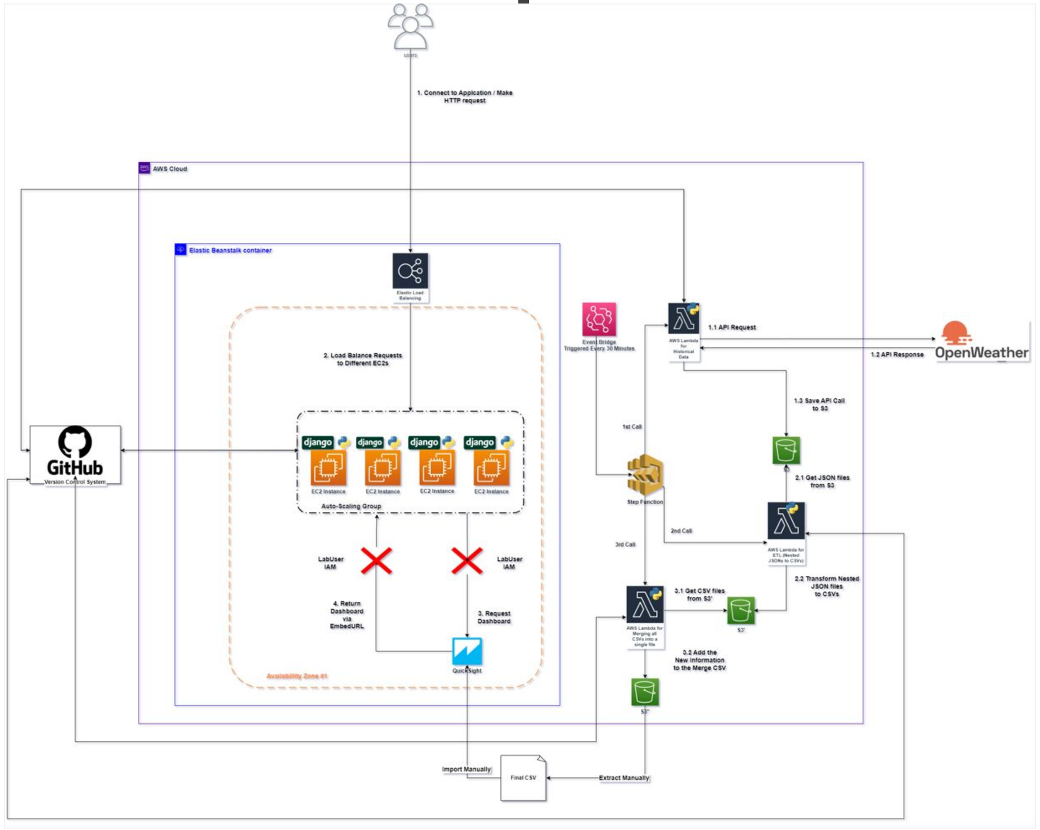
Implementation Pipeline

Initial Proposed Design

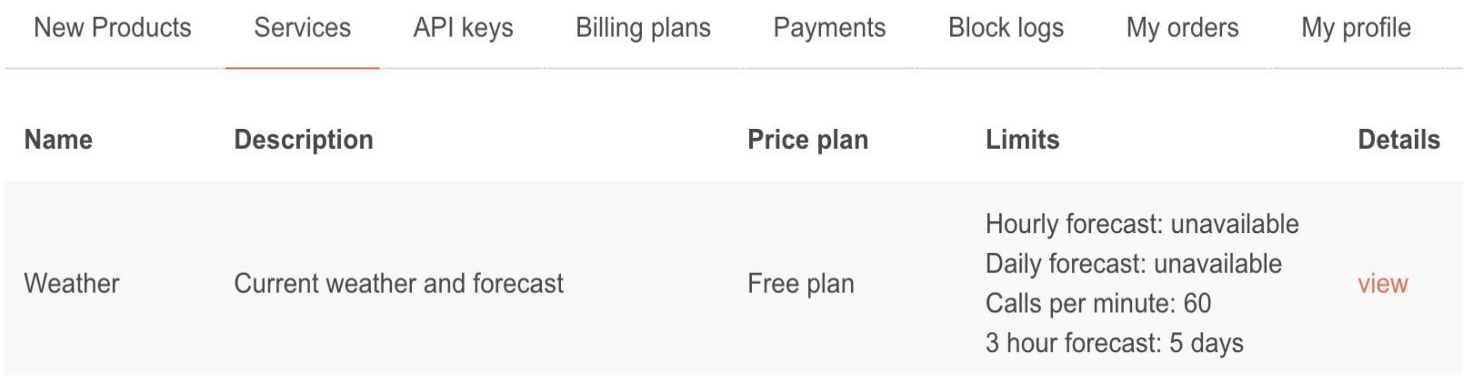


Implementation Pipeline

Final Design


users
1. Connect to Application / Make HTTP request
AWS Cloud
Elastic Beanstalk container
Elastic Load Balancing
2. Load Balance Requests to Different EC2s
django
EC2 Instance
EC2 Instance
EC2 Instance
EC2 Instance
Auto-Scaling Group
GitHub
Version Control System
LabUser IAM
LabUser IAM
4. Return Dashboard via EmbedURL
3. Request Dashboard
QuickSight
Availability Zone #1
Event Bridge Triggered Every 30 Minutes
AWS Lambda for Historical Data
1.1 API Request
1.2 API Response
OpenWeather
1.3 Save API Call to S3
1st Call
S3
2.1 Get JSON files from S3
Step Function
2nd Call
AWS Lambda for ETL (Nested JSONs to CSVs)
3rd Call
2.2 Transform Nested JSON files to CSVs
3.1 Get CSV files from S3'
AWS Lambda for Merging all CSVs into a single file
S3'
3.2 Add the New Information to the Merge CSV
S3''
Import Manually
Final CSV
Extract Manually

OpenWeather API



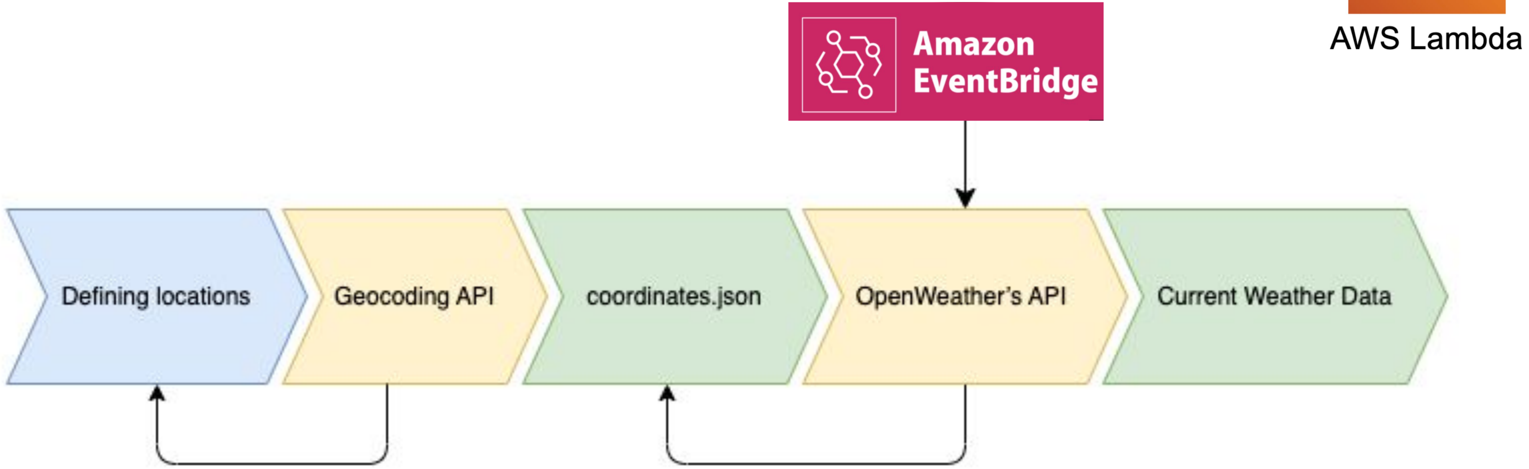
Name	Description	Price plan	Limits	Details
Weather	Current weather and forecast	Free plan	Hourly forecast: unavailable Daily forecast: unavailable Calls per minute: 60 3 hour forecast: 5 days	view

```
http://api.openweathermap.org/geo/1.0/direct?q=Barcelona&limit=5&appid={API key}
```

```
https://api.openweathermap.org/data/2.5/weather?lat={lat}&lon={lon}&appid={API key}
```

AWS Lambda for Historical Data Retrieval

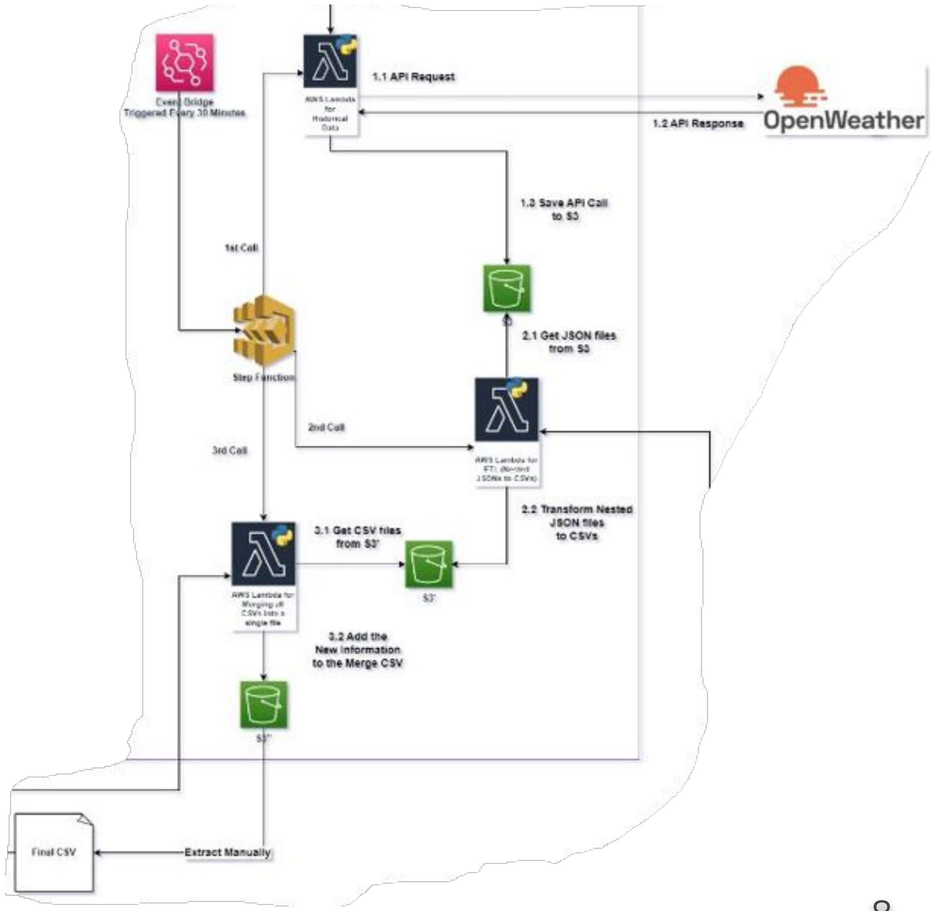
- Real-time simulation
- EventBridge Trigger
- Deployment package with dependencies[1]



[1]https://docs.aws.amazon.com/lambda/latest/dg/python-package.html

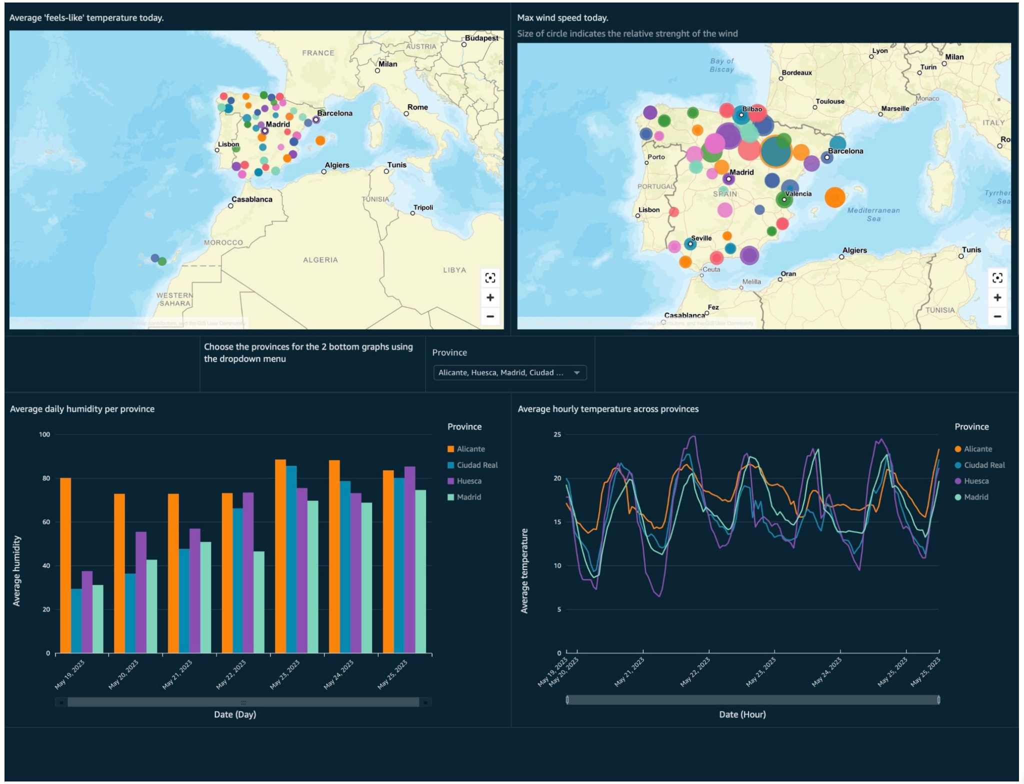
ETL Process

- Total of 3 Lambda functions:
  - Get data from API - json
  - Transform json to csv
  - Merge all the past files into one
- Orchestrated with AWS Step Function
- Triggered with EventBridge every 30 mins


Event Bridge
Triggered Every 30 Minutes
AWS Lambda for Historical Data
1.1 API Request
1.2 API Response
OpenWeather
1.3 Save API Call to S3
1st Call
S3
2.1 Get JSON files from S3
Step Function
2nd Call
3rd Call
AWS Lambda for ETL (Nested JSONs to CSVs)
2.2 Transform Nested JSON files to CSVs
3.1 Get CSV files from S3'
S3'
AWS Lambda for Merging all CSVs into a single file
3.2 Add the New Information to the Merge CSV
S3''
Final CSV
Extract Manually

Dashboarding

- Done using AWS QuickSight
- Interactive
- 2 map graphs
- 2 selectable bar/line graphs
- Updated with new data every 30 mins*

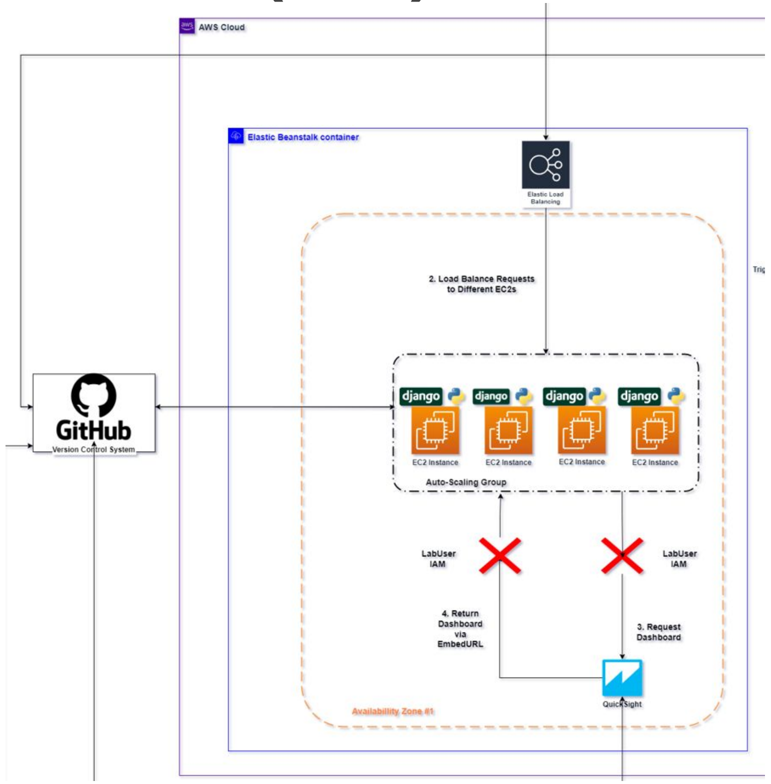

Average 'feels-like' temperature today.
Max wind speed today.
Size of circle indicates the relative strenght of the wind
Average daily humidity per province
Average humidity
Date (Day)
Average hourly temperature across provinces
Average temperature
Date (Hour)
Province
Alicante
Ciudad Real
Huesca
Madrid

Choose the provinces for the 2 bottom graphs using the dropdown menu

Province: Alicante, Huesca, Madrid, Ciudad ...

Django and Elastic Beanstalk (EBS)

- Application Technologies
    - Python
    - Django Framework
    - AWS Elastic Beanstalk
- Version Controlling System
    - Github

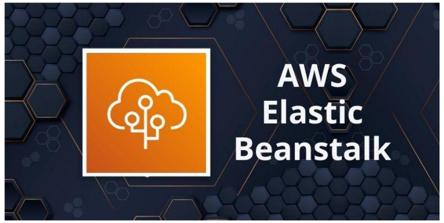


Benefits of Django

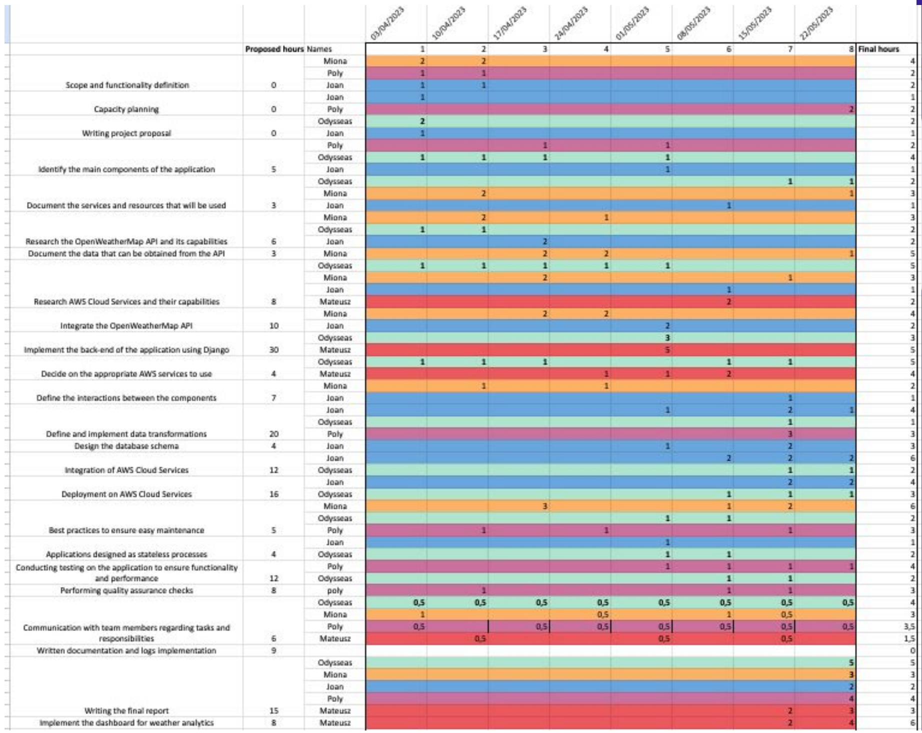
- Collaboration and Version Control Management with a single codebase on GitHub.
- Use of environment variables ensured:
  - Flexibility,
  - Security, and
  - Seamless transitions between development stages.
- Treat backing services as attached resources allowed:
  - Integration with AWS services like:
    - QuickSight and Lambdas.

Benefits of Elastic Beanstalk

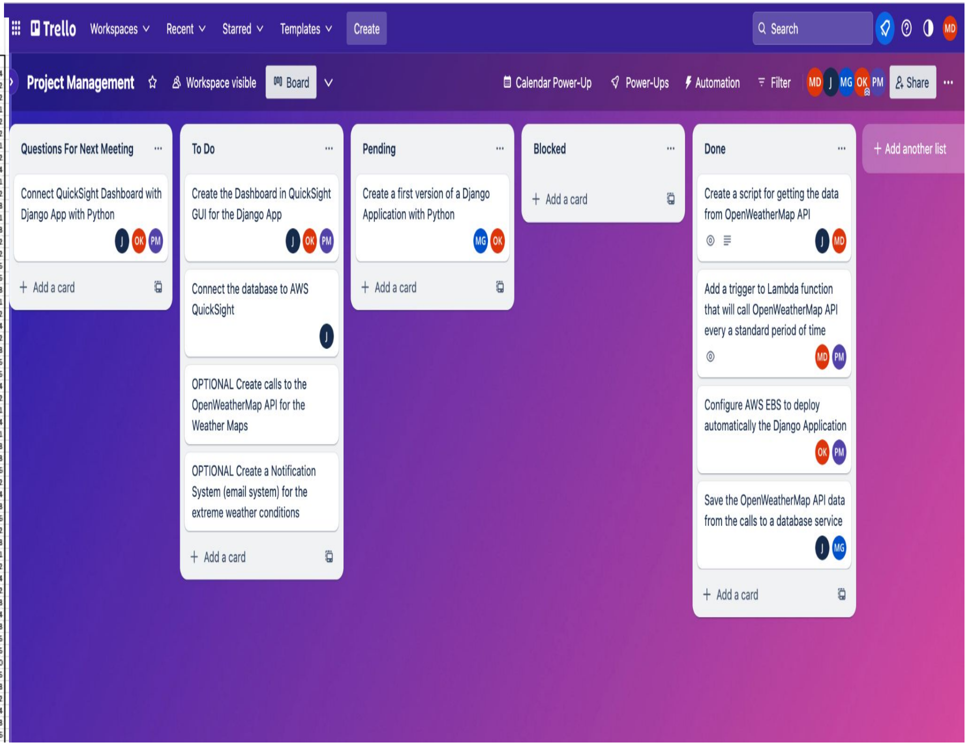
- Configuring the code in a **virtual environment** and **installing dependencies**.
- **Automated creation** of the application's **environment and deployment of code**.
- Allowing **scalability** by:
    - **Executing** the application as a **stateless process** and
    - **Automatically scaling EC2 instances** based on traffic demands.



Task Allocation and Time Management Tool



			03/04/2023	10/04/2023	17/04/2023	24/04/2023	01/05/2023	08/05/2023	15/05/2023	22/05/2023	
	Proposed hours	Names	1	2	3	4	5	6	7	8	Final hours
		Miona	2	2							4
		Poly	1	1							2
Scope and functionality definition	0	Joan	1	1							2
		Joan	1								1
Capacity planning	0	Poly								2	2
		Odysseas	2								2
Writing project proposal	0	Joan	1								1
		Poly			1		1				2
		Odysseas	1	1	1		1				4
Identify the main components of the application	5	Joan					1				1
		Odysseas							1	1	2
		Miona		2						1	3
Document the services and resources that will be used	3	Joan						1			1
		Miona		2		1					3
		Odysseas	1	1							2
Research the OpenWeatherMap API and its capabilities	6	Joan			2						2
Document the data that can be obtained from the API	3	Miona			2	2				1	5
		Odysseas	1	1	1	1	1				5
		Miona			2				1		3
		Joan						1			1
Research AWS Cloud Services and their capabilities	8	Mateusz						2			2
		Miona			2	2					4
Integrate the OpenWeatherMap API	10	Joan					2				2
		Odysseas					3				3
Implement the back-end of the application using Django	30	Mateusz					5				5
		Odysseas	1	1	1			1	1		5
Decide on the appropriate AWS services to use	4	Mateusz				1	1	2			4
		Miona		1		1					2
Define the interactions between the components	7	Joan							1		1
		Joan					1		2	1	4
		Odysseas							1		1
Define and implement data transformations	20	Poly							3		3
Design the database schema	4	Joan					1		2		3
		Joan						2	2	2	6
Integration of AWS Cloud Services	12	Odysseas							1	1	2
		Joan							2	2	4
Deployment on AWS Cloud Services	16	Odysseas						1	1	1	3
		Miona			3			1	2		6
		Odysseas					1	1			2
Best practices to ensure easy maintenance	5	Poly		1		1			1		3
		Joan					1				1
Applications designed as stateless processes	4	Odysseas					1	1			2
Conducting testing on the application to ensure functionality		Poly					1	1	1	1	4
and performance	12	Odysseas						1	1		2
Performing quality assurance checks	8	poly		1				1	1		3
		Odysseas	0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,5	4
		Miona	1			0,5		1	0,5		3
Communication with team members regarding tasks and		Poly	0,5		0,5	0,5	0,5	0,5	0,5	0,5	3,5
responsibilities	6	Mateusz		0,5			0,5		0,5		1,5
Written documentation and logs implementation	9										0
		Odysseas								5	5
		Miona								3	3
		Joan								2	2
		Poly								4	4
Writing the final report	15	Mateusz							2	3	3
Implement the dashboard for weather analytics	8	Mateusz							2	4	6



Trello · Workspaces · Recent · Starred · Templates · Create · Search

Project Management · Workspace visible · Board · Calendar Power-Up · Power-Ups · Automation · Filter · MD J MG OK PM · Share

Questions For Next Meeting
- Connect QuickSight Dashboard with Django App with Python (J, OK, PM)
- + Add a card

To Do
- Create the Dashboard in QuickSight GUI for the Django App (J, OK, PM)
- Connect the database to AWS QuickSight (J)
- OPTIONAL Create calls to the OpenWeatherMap API for the Weather Maps
- OPTIONAL Create a Notification System (email system) for the extreme weather conditions
- + Add a card

Pending
- Create a first version of a Django Application with Python (MG, OK)
- + Add a card

Blocked
- + Add a card

Done
- Create a script for getting the data from OpenWeatherMap API (J, MD)
- Add a trigger to Lambda function that will call OpenWeatherMap API every a standard period of time (MD, PM)
- Configure AWS EBS to deploy automatically the Django Application (OK, PM)
- Save the OpenWeatherMap API data from the calls to a database service (J, MG)
- + Add a card

+ Add another list

Thank you