Universidad Distrital Francisco José de Caldas

Facultad de Ingeniería

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Weather.com

Andrey Camilo Gonzalez Caceres
Cristian Andres Gamez Nuñez

June 10, 2024

Abstraction

Wheather.com, more than an application, is a solution to the many weather applications that do not fulfill their main objective, we are based on designing a solution for the common shortcomings of weather applications, with some very nice new functions. and useful for the user. Finally offering a totally different weather application that provides a color palette along with its interface in a very visually friendly way, and processes carried out providing accurate and current weather data.

Introduction

Weather.com is a weather application, but not just any weather application, a weather application with an interface and color palette that are psychologically more susceptible and pleasant to the user. Carrying out processes to deliver the most precise and exact data according to location and place choosen for the data sample.

The function of the application is to provide beyond climate data, an experience never before perceived by users, so that the set of interfaces shown feels very friendly, also showing new data display functions, location selection and interfaces psychologically pleasing to the human eye. Our application is different from the rest due to its interfaces, its attractive selection of locations, its incredible and fast response to data sample requests, the fulfillment of its main objective.

We are based on the use of graphic design tools and programming languages such as HTTML, CSS, JavaScript and Django, to carry out process tools such as, mainly python, FastAPI and OpenWeatherMap, and finally for the hosting of usernames, passwords, locations, among others, we implement SQLalchemy, postgreSQL and DBeaber.

Weather.com will be taken as a solution to the great problem of nonfulfillment of the main objective of common climate applications, we propose to be a solution to the shortcomings and failures of other climate applications, carrying out faster and more effective processes, with much friendlier interfaces. before the user, and saving data with greater effectiveness, solving each of the main errors.

We plan to provide the best weather application possible, focusing our attention on user experience, pleasant interfaces and effectiveness in data display, hosting it on an functional and fast server, and hosting the respective data in an effective but very secure database of data. We expect excellent reception, and great experience and opinions from users, having great scalability, through recommendations between users and new users.

This project is very ambitious since it allows us to know a lot of very important data in daily life that will help us make decisions according to the respective climate at the given moment, giving us to understand that both in this project and in others, computing techniques will always be A great tool that improves every day and is more useful.

Methods and Materials

For the development of our weather application, we have followed an object-oriented approach, using the Python programming language. Next, we will describe the methods and techniques used during the development process:

Technology Selection:

Python was selected as the primary programming language due to its versatility, readability, and extensive community support. The following Python libraries were used: Faker: It will be used to generate fictitious data for the development and testing of the application. FastAPI: It will be used for the development of high-performance web APIs, which will allow easy integration with the OpenWeatherMap API and the creation of endpoints for the application's functionalities. SQLAlchemy: It will be used to interact with the database, facilitating the management and manipulation of user data.

Figure 1: OpenWeatherMap, FastAPI, Python logos

Application Architecture:

A monolithic architecture was chosen for the weather application, meaning that all application components are integrated into a single system. This simplifies the initial development and deployment of the application, resulting in a faster and less complex process.

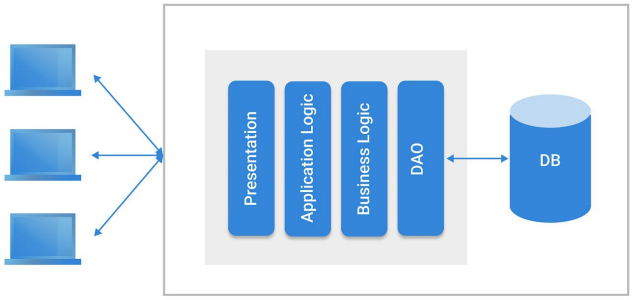

Figure 2: Monolithic Arquitecture reference

OpenWeatherMap API Integration:

The OpenWeatherMap API will be used to obtain accurate and up-to-date weather data for the application. This will be achieved by integrating HTTP requests to the OpenWeatherMap API within our application.

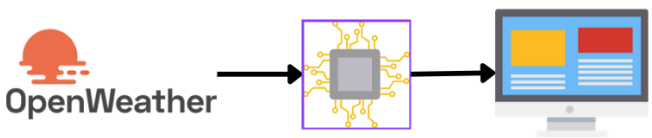

Figure 3: OpenWeatherMap reference

Functional Development:

The following main functionalities of the application will be developed: Display of current weather: Methods will

be created to obtain and display current weather data, including temperature, weather conditions, and wind chill. Weather forecast: Methods will be implemented to obtain and display the future weather forecast for a specific location. User Registration: Methods will be developed to allow users to register in the application, providing basic information such as name, password and location. Viewing users (for administrators): Methods will be created so that administrators can view and manage the list of users registered in the application.

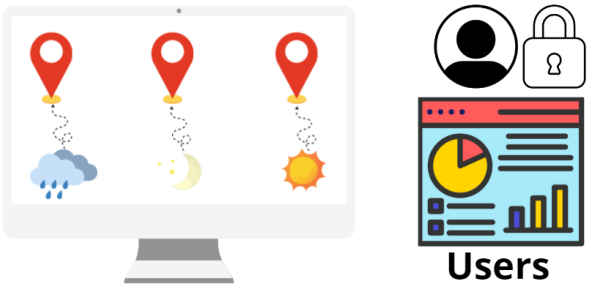

Figure 4: Ubication weather and users reference

Data Management and Persistence:

SQLAlchemy will be used to handle the interaction with the database. Methods will be implemented to store, retrieve, update and delete user data in the database, ensuring adequate persistence of user information.

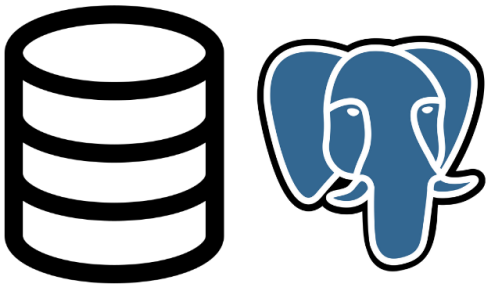

Figure 5: PostgreSQL reference

The selection of these languages and tools were the best due to the flexibility, speed, connectivity, security and comfort of use that these technological tools offer us. Python being the most used programming language for good reason, offering us ease, organization and multiple options in writing the code, using the monolith architecture, which allows us to have great clarity of the correlation between its fixed structures, the use of OpenWeatherMap for weather data and postgreSQL, DBebear, SQLalchemy for data storage.

Results

In developing the climate application, extensive unit and end-to-end testing was carried out to ensure its quality and optimal performance. Among these tests, 10 tests dedicated to user registration stand out, which showed a notable pass rate of 90%.

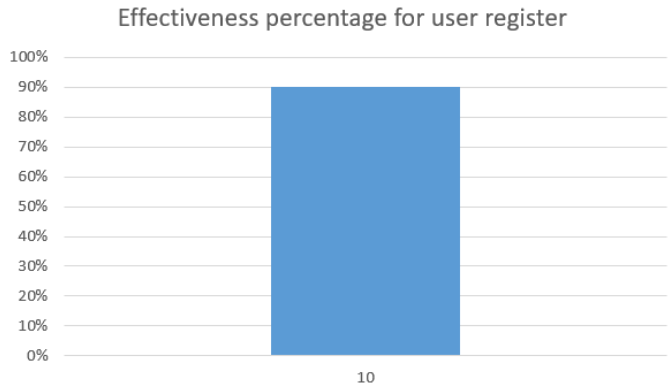

Figure 6: Efectiveness user register reference

Additionally, 10 additional tests were carried out to evaluate the login process, also obtaining a 90% approval rate.

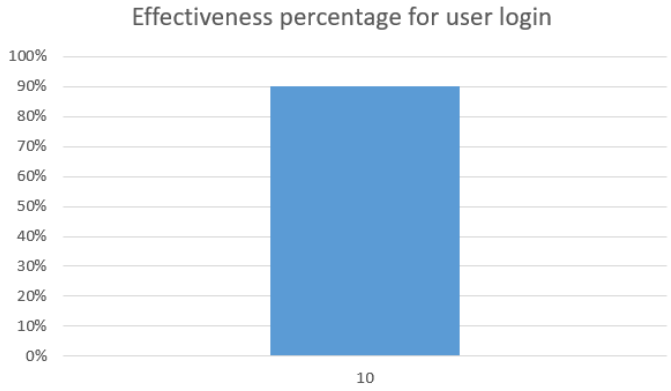

Figure 7: Efectiveness user login reference

In addition, 20 specific tests were carried out to evaluate the visualization of climate data, with a passing rate of 75%.

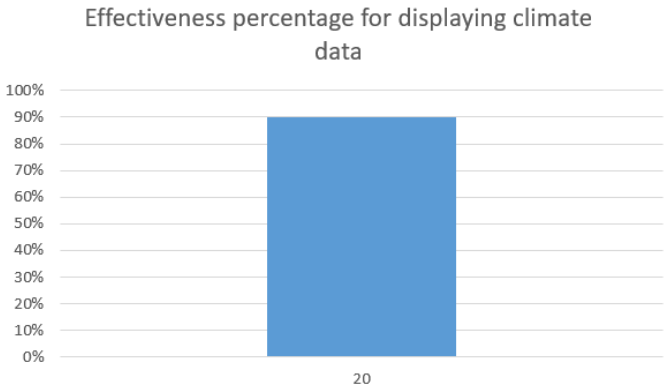

Figure 8: Efectiveness display of climate data reference

These results reflect the development team's commitment to excellence in the implementation of the application, ensuring a reliable and satisfactory experience for users.

Conclusion

When we created Weather.com, our goal was to design a weather app that not only offers weather information but also enhances the user experience with its well-thought-out interface and seamless functionality. Using Python and essential libraries such as FastAPI and SQLAlchemy, we seamlessly integrate real-time weather data from the OpenWeatherMap API, ensuring users have access to real-time updates. Our streamlined development approach made it easy to build and deploy the app, while maintaining clarity and efficiency throughout the project.

Throughout this process, we addressed issues seen in weather apps and provided a reliable and visually appealing tool that truly meets user needs. Weather.com stands out for its fast response times, user-friendly design and efficient data handling, proving that we successfully achieve our goals.

Bibliography

[1] *¿Qué son las reglas de negocio? — IBM*. IBM in Deutschland, Österreich und der Schweiz. URL: https://www.ibm.com/es-es/topics/business-rules (visited on 04/10/2024).

[2] *Poster X-Meeting 2013*. Poster X-Meeting 2013.pdf. EngAndres. (Visited on 06/10/2024).

[3] *Poster X-Meeting 2014*. Poster X-Meeting 2014.pdf. EngAndres. (Visited on 06/10/2024).

[4] C. Sierra. *ud-public/courses/advanced-programming/project/Paper Guidelines.pdf at main · EngAndres/ud-public*. GitHub. URL: https://github.com/EngAndres/ud-public/blob/main/courses/advanced-programming/project/Paper_Guidelines.pdf (visited on 06/10/2024).

[5] *Todos los diagramas UML. Teoría y ejemplos*. DiagramasUML.com. URL: https://diagramasuml.com/ (visited on 04/10/2024).

[5] [1] [4] [2] [3]