WEATHER APP

Development Report

Student ID: 202843

Introduction

With the development of the weather application, a response has been given to a basic need: real, reachable information regarding the weather at the location of the user, accessible and provided on time.

It is organized in this document, with the origin of the idea, the solution meant to be provided, the approach to software engineering taken, and the current status of the product. The Appendix provides an itemized list of the required capabilities of the solution.

Underlying Need or Idea

The project was conceptualized upon the needs that were arising from an individual who wanted to have a real-time update on the weather. Before, broad data was given to a broad population, which might not necessarily apply to their individual case. This clearly indicated the need for a weather-reporting tool that is customizable and works in real time, most especially the one that could disseminate accurate weather forecast reports and current weather condition situations directly into the users' place or at the exact time and place required by the user.

Envisioned Solution

The solution was visualized as a desktop application that provides real-time weather forecasts to its user and current weather conditions based on either the immediate geographical location or one the user defines.

The application should provide an extremely friendly interface with views, such as the current temperature, weather conditions, and a week's forecast. It was to be designed to change the background with respect to the current weather condition (e.g., rainy, sunny, and snowy) in order to make it have a better look, as well as seperately display the local time of the location.

Software Engineering Method Used

The weather application is developed with the Agile software engineering methodology. Due to the iterative development cycles and continued iteration of incorporation of feedback in it, the Agile methodology—with the best reputation for flexibility and client orientation—best fitted this project.

Accordingly, the whole project has been divided into sprints, and attention was paid to the functionality of the developed application. Reviewed and adapted regularly, the outcomes of testing and feedback are to ensure that the end product meets the user requirement and expectations in full.

Current Status of the Product

With that, it would be the end of the "weather app" project. The application can now be considered a full-fledged desktop application that is running smoothly with the specifications of the applications and requirements of the users.

In this application, it shows current and even weekly real-time weather information: current temperature, weather conditions for location of user at a given time, or specified city.

The most drastically pleasing and highly dynamic feature is the background, which changes according to the weather conditions in the city; and has found particular satisfaction in that it allows the visual appearance and the usability of the interface to be more intuitive.

Conclusion

The weather application project demonstrates that, with a clear understanding of what the users need and good software development methodology, one may come up with a very useful tool. With the Agile method, the development of this project was a process that readily allowed for changes and responses to easily fit user requirements in at different stages.

The end product met not only the identified need in this niche of offering personalized information on weather but even afforded a lively experience with its dynamic interface.

Appendix: Software Solution Capabilities

- Real-time Weather Data: This retrieves the current weather, which reflects the temperature and general status of the weather of a given geographical location.
- Geographical Flexibility: This allows the user to manually input any city name to check the weather in the specified location.
- Dynamic Background: The application changes the background dynamically according to weather conditions; for example, if it is sunny, cloudy, rainy, or snowy.
- Weekly Weather Forecast: This is a prediction of the general climatic conditions for a week, including the daily forecast of temperatures and conditions.
- Local Time Display: It shows the local time for the current city and other specified cities, therefore providing context for the given weather information.
- User-friendly Interface: Offers a simple, intuitive interface that ensures ease of use for all demographics.
- Customization: Set allows the users to interface with the application by specifying the places, with real-time updates showing what the user inputs.
- Error Handling: Strong error handling mechanism to let the user know about the failures, for example, in determining the location or retrieving the weather data.

All of these powerful features combined in one application really make these suite of applications very powerful for pretty much anybody that's looking for very in-depth and personal weather information directly on their desktop.

Weather Application Handbook

For Users

This is the desktop application software that aids in forecasting weather and current conditions in real-time regarding a location or city of choice. "How to get started and get the best out of your weather application.

Getting Started

Application launch: Double click on the application icon from the desktop to launch the Weather Application. It picks up weather information for the current geographical location at launch.

Viewing Weather Information: The main window is one with a dynamic background changing according to the current temperature and weather conditions. For instance, when the weather is sunny, the background is usually a blue sky. In addition, at the bottom of this dynamic weather window, there is a weekly weather forecast showing daily temperatures together with weather conditions each day.

Changing Location - To view weather information for a different city:
- Locate the text entry box at the top of the application window.
- Type the name of the city you're interested in.
- Click on the "Search" button or just hit the enter key. The app will be refreshed to show the weather information on the city that you searched.

Local Time Display: The application also provides the local time just above the displayed temperature for the current or specified city. With this, therefore, the user will be able to fit the weather information in reference to the time of the day.

Tips for a Better Experience

Correct City Names: Ensure you spell the city names correctly to avoid errors in fetching weather data.

Internet Connection: Internet connection has to be enabled for fetching the latest weather and updating the dynamic background of the application.

For Programmers

This Weather Application is a Python-based one, developed using the following libraries: tkinter as the GUI; python_weather, geocoder, and timezonefinder with pytz for handling the time zone. It was the very first chunk of code that can actually show a user how to navigate with the codebase and make changes to the app in order to potentially expand its functionality.

Code Structure

Main Application Loop: Located at the bottom of the script, orchestrating the GUI setup and event handling.

Weather Data Fetching: The weather data fetching is done through a Python asynchronous function, fetch_weather_and_display, which fetches the weather data from end API.

GeoSource value from geoposition and handle timezone: Determine the current or specified location by geoposition and get the local time using timezone detection supported by timezonefinder and geonamescache to handle the ambiguity in time at the feature boundary of a time zone.

UI Update Functions: Functions dynamically called to update the GUI in respect of the data on weather and the received conditions include the update_weather_display and update_background.

Customizing and Extending

Addition of More Functionalities: The python_weather API has features that can integrate air quality indices or weather alert, as well as a myriad of other features in the other libraries used within the script, including updating GUI to integrate new information.

UI Enhancements: Change the tkinter widgets and styling to look different or enhance user interaction. - Optimization: Check the possibility of place or time weather data caching using the provided API in places or times where there are repeated queries to minimize the number of calls made.

Development and Testing

Environment Setup: Ensure that all dependencies are installed with pip according to the requirements.txt file provided.

Running the Application: Run the script with a Python interpreter. Python 3.6 or greater is recommended so that there is support for the asynchronous function.

Debugging and testing: You may need this to debug by putting in some logger, and if applicable, you may write unit tests for some important functions, especially when interacting with other APIs.

Compare with other sources: When running the app, compare your results with other data aggregates to ensure reliability, as done in the beta testing screenshots below: