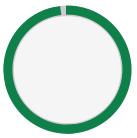
Dupli Checker

PLAGIARISM SCAN REPORT

2%
Plagiarised



98%
Unique

Date	2023-11-23
Words	997
Characters	7426

Content Checked For Plagiarism

2nd
The last research paper introduces an advanced weather monitoring system leveraging IoT technology to monitor a wide range of weather parameters in real-time. It aims to provide accessible data for meteorological departments, weather stations, aviation, marine industries, and agriculture. While the system offers benefits like portability, low maintenance, and cost-effectiveness, it has limitations, including sensor measurement ranges, placement accuracy, solar panel power constraints, and communication range limitations. Nonetheless, the project represents a significant step forward in environmental monitoring technology, offering timely weather alerts and paving the way for future advancements in the field.
The research papers of weather forecasting represent weather to help in agricultural industry, transportation, pollution, metrological, marine and aviation industry, weather stations and so on signifying the future scope of representation of weather for other domains, therefore, we are creating a weather application for city to city travelling commuters that works around the globe to help plan the trip, time, clothing, accessories and things accordingly.
Project Scope:
The project scope includes the following:
1. Development of a Python application using Tkinter for the graphical user interface.
2. Integration with the OpenWeatherMap API for real-time weather data.
3. Display of current weather information, including temperature, humidity, coordinates, city, and country.
4. Dynamic background image change to reflect day or night.
5. Display of current date and time.
6. A "Search" button to trigger weather data retrieval.
7. Error handling for API requests.
Limitations:
The Weather App project has some limitations:
1. Dependency on the availability and reliability of the OpenWeatherMap API, which may be subject to usage limitations or downtime.
2. Limited features; the app provides basic weather information but may not offer advanced features such as forecasts or historical data.
3. The application does not account for user preferences or settings; it provides information in a fixed format.
4. The GUI may vary in appearance and functionality based on the capabilities and limitations of the Tkinter library and the user's system.
5. The application's security model for storing the API key should be further strengthened to ensure data privacy.

3. System analysis:
Existing System:
The existing system for obtaining weather information typically involves web-based weather websites, mobile apps, and television weather broadcasts. Users access weather information by visiting websites or using dedicated weather applications on their devices. However, the existing system has several limitations:

• Limited interactivity with weather data.
• Variation in user interfaces and features among different weather sources.
• Dependence on data source quality and accuracy.
• Limited customization options for users.
• Lack of additional context, such as time of day, in weather presentation.
Scope and Limitations of Existing System:
• The existing system lacks interactivity, customization, and additional contextual information.
• Users have to rely on different sources for weather data, which may vary in accuracy.
• The user interface and user experience vary widely based on the source of weather information.
Project Perspective:
The Weather App project aims to address the limitations of the existing system by providing a user-friendly and visually appealing application with the following features:
• A graphical user interface (GUI) created using the Tkinter library.
• Integration with the OpenWeatherMap API to fetch real-time weather data.
• Display of real-time weather information, including temperature, humidity, coordinates, city name, and country.
• Dynamic background changes to represent the time of day (day or night).
• Display of the current date and time to provide additional context to the user.
• Interactivity that allows users to input a city name and retrieve specific weather information.
• Error handling to ensure the reliability and robustness of the application.

Features:
1. User Interface: The project offers an intuitive and user-friendly graphical interface, making it easy for users to interact with the application.
2. Real-Time Weather Information: The project integrates with the OpenWeatherMap API to provide up-to-date and accurate weather information, including temperature, humidity, and more.
3. Dynamic Background: The application dynamically changes its background to reflect the time of day, enhancing the user's experience and providing context.
4. Date and Time Display: Users can view the current date and time within the application, allowing them to make weather-related decisions with temporal context.
5. Interactivity: The app allows users to input a city name and fetch specific weather information for that location.
6. Error Handling: The project includes error handling to ensure that the application can gracefully handle issues, such as failed API requests.
Stakeholders:
1. Developers: Those responsible for designing, coding, and maintaining the application.
2. Users: Individuals seeking real-time weather information in a user-friendly format.
3. OpenWeatherMap: The project relies on the OpenWeatherMap API as a data source and therefore, OpenWeatherMap is an indirect stakeholder.
4. System Administrators: Individuals responsible for deploying and maintaining the application.

Requirement analysis:

Functional Requirement: The system must allow users to register, log in, and view weather data by city. It should fetch and display current temperature, humidity, and other details using an external API.
Performance Requirement: The system should retrieve weather data swiftly, ensuring real-time updates and responsive display within 3 seconds of user input.
Security Requirement: Implement secure password storage using encryption techniques, ensure data encryption during transmission (HTTPS), and validate user input to prevent injection attacks. Employ secure API key handling and implement access controls to protect sensitive information.

4. System design:

Design constraints:

The system must adhere to API usage policies and rate limits, preventing excessive requests. It should flexibly accommodate variations in API response structures. Compatibility across multiple platforms and browsers must be ensured. Implementation should incorporate robust error handling, gracefully degrading the system during API disruptions or failures.

System Model:
• User Interface (UI): Developed using Tkinter, offering an interactive interface for users to input city names and view weather data.
• OpenWeatherMap API: An external source for retrieving real-time weather data based on user input.
• Data Processing: Involves data retrieval, parsing JSON responses, and updating the UI with fetched weather information.
• Error Handling: Ensures the system is robust by managing API request

Matched Source

Similarity 4%

Title:**github.com › rjkumarwankhade › software-exe-fileGitHub**

#Overview This Python project consists of a graphical user interface (GUI) created using the Tkinter library. The GUI provides buttons for various fun and interactive Python programs. Each button corresponds to a different Python function or game.

https://github.com/rjkumarwankhade/software-exe-file/blob/main/README.md/