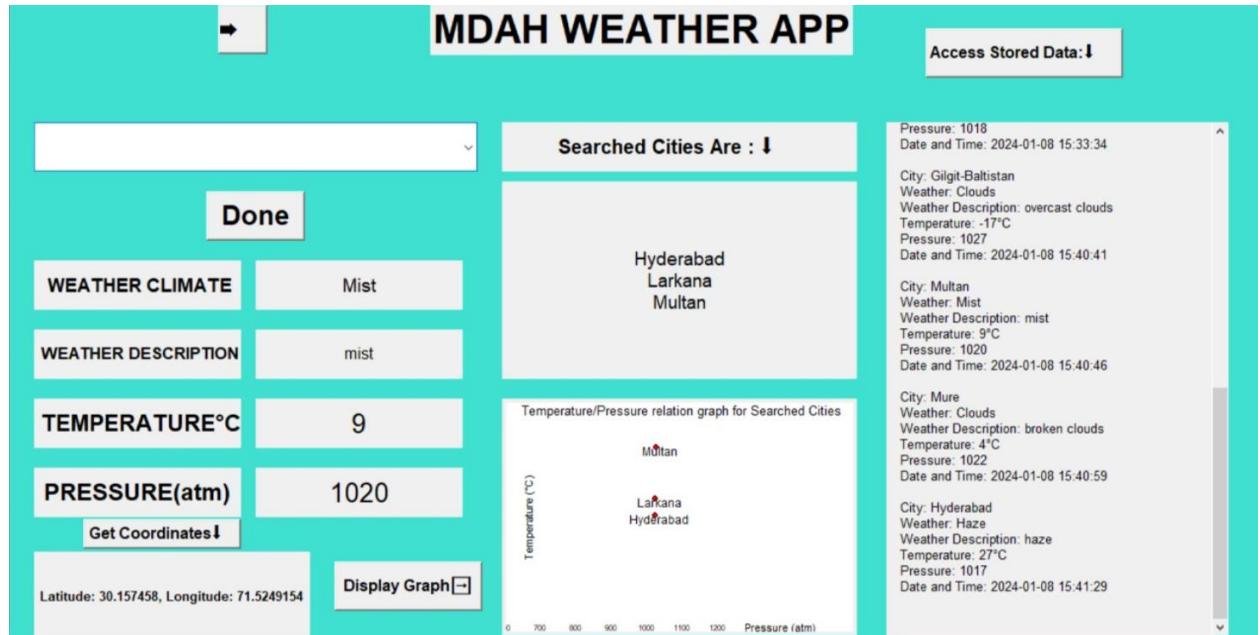


MDAH WEATHER APP

1. OUTPUT:



1. Tools and Techniques Used:

Tools:

- ◆ Python: The primary programming language used for development.
- ◆ Tkinter: A Python GUI toolkit used for creating the application's graphical interface.
- ◆ Requests: A library for making HTTP requests, utilized for fetching weather data from the OpenWeatherMap API.
- ◆ JSON: Used for handling and parsing data obtained from the API.
- ◆ CSV: Utilized for reading city coordinates from a CSV file.
- ◆ Datetime: Employed for obtaining the current date and time.

Techniques:

- ◆ Linked List: Implemented a linked list data structure to store and manage a list of searched cities.
- ◆ Stacks: To switch pages, push and pop operations have been used.
- ◆ Insertion Sort: Used for sorting the list of searched cities alphabetically.
- ◆ Linear Search: For searching the city name.
- ◆ File Handling: Created functions to save weather data to a text file and read/display content from it.
- ◆ Canvas and Scrollbar: Incorporated a canvas widget with a scrollbar to display stored data with scrolling capability.
- ◆ Graphical Visualization: Developed a graphical representation of temperature-pressure relations for searched cities.

- ◆ Combobox: Implemented a dropdown list for city selection using the Tkinter Combobox widget.
- ◆ API Integration: Fetched weather data from the OpenWeatherMap API using HTTP requests.
- ◆ Event Handling: Employed event handling to update the Combobox dynamically as the user types.
- ◆ City Coordinates CSV File: A dataset containing latitude and longitude coordinates for various cities. This data is utilized for displaying the coordinates of a selected city in the application.

2. Main Algorithms Implemented in the Project:

- **Insertion Sort Algorithm:**

Description: Used to sort the list of searched cities alphabetically.

Implementation: The insertion sort algorithm iterates through the list, comparing and rearranging elements to maintain the sorted order.

- **Linear Search Algorithm:**

Description: Used to search city names

Implementation: The algorithm updates the combobox suggestions by filtering the list of city names based on a case-insensitive substring match with the user's input. It then dynamically displays the matching cities in the combobox and searching co-ordinates from the csv files and then saving the co-ordinates in the text file

- **Fetch Weather Data Algorithm:**

Description: Retrieves weather data from the OpenWeatherMap API for a given city.

Implementation: Utilizes the Requests library to make an HTTP request to the API, providing the city name and API key. The obtained data is then processed and returned.

- **Get Coordinates Algorithm:**

Description: Retrieves latitude and longitude coordinates for a given city from a CSV file.

Implementation: Reads the city coordinates from a CSV file using the CSV library, searching for a match based on the city name.

- **Display Graph Algorithm:**

Description: Generates a graphical representation of the temperature-pressure relations for searched cities.

Implementation: Uses the Canvas widget to draw circles at specific coordinates based on temperature and pressure values. City names are displayed next to the corresponding points, creating a scatter plot.
