**Weather\_Bot Project Report**

**I. Introduction**

The Weather\_Bot project, titled "Weather\_Bot," aims to provide users with instantaneous weather forecasts from cities across the globe through a streamlined interface on the Telegram platform. This initiative simplifies access to real-time weather information, catering to users' needs promptly and efficiently.

**II. Project Idea and Value Added**

The project addresses the need for swift and accessible weather updates for cities worldwide. By leveraging the Telegram bot interface, Weather\_Bot facilitates quick access to real-time weather forecasts, significantly enhancing user convenience and eliminating the need for multiple searches across different platforms.

**III. Similar or Related Works of Projects**

There are many apps or websites that shows the weather forecasts of the world, and some even have it built into the system. But to me, for older people who is not technologically capable of accessing that information, this is the solution. As most elders tend to only use their phones to chat/message and nothing else, having a telegram bot that can show them the weather forecast from all over the world by just writing the name of the city is an added convenience.

**IV. SDLC Phases, Algorithms, Pseudocode, Flowchart, etc.**

The project progressed through multiple phases of the Software Development Life Cycle (SDLC), encompassing planning, design, implementation, and finalization. Algorithms, pseudocode, and flowcharts were utilized to outline the logic flow and facilitate the integration of the Telegram bot interface with the OpenWeatherMap API. Most of it was learn from YouTube and some help from my seniors. This code using telegram framework, so it really confused me a lot of time and it was very challenging.

**V. System Features and Functions**

Weather\_Bot's primary function is to enable users to input the name of their desired city, subsequently receiving real-time weather forecasts for that location. The streamlined interface simplifies this process, providing users with immediate access to current weather conditions.

**VI. Code Implementation**

The bot fetches real-time weather data from the OpenWeatherMap API based on user requests, ensuring accurate and prompt information retrieval. And then I just prompted it to respond to different input(cities). Using the telegram frame work is a bit hard to explain but mostly it was just organizing/assigning responses to the user’s inputs.

  api\_key = "4c0b058647d489799cb157d186004397"

    url = f"https://api.openweathermap.org/data/2.5/weather?q={city}&appid={api\_key}"

this where the magic happens, this is how our bot retrieve its info via api key from a website that offers free api.

def start(update, context):

    greet = """WELCOME TO WEATHER-FORECAST💧⛄️☁️☔️"""

    context.bot.send\_chat\_action(

        chat\_id=update.effective\_message.chat\_id, action="typing"

    )

    context.bot.send\_message(chat\_id=update.effective\_chat.id, text=greet)

This is the method that is evoked once the user click “start”. It prints “"""WELCOME TO WEATHER-FORECAST💧⛄️☁️☔️"""

def help(update, context):

    text = """Available Commands :-

    Enter a city name to get the weather forecast - Example: //London"""

    context.bot.send\_chat\_action(

        chat\_id=update.effective\_message.chat\_id, action="typing"

    )

    context.bot.send\_message(chat\_id=update.effective\_chat.id, text=text)

And this one is for when the user clicks “help”.

  def get\_weather():

        if data["cod"] == 200:

            temp = data["main"]["temp"]

            weather = data["weather"][0]["main"]

            wind = data["wind"]["speed"]

            temp -= 273.15

            return round(temp, 2), weather, wind

        else:

            return "City not found"

This checks if the response code (data["cod"]) is equal to 200, indicating a successful response. If so, it extracts temperature (temp), weather description (weather), and wind speed (wind) from the provided data dictionary. It returns "City not found" if the response code is not 200.

**VII. Code Testing/Debugging**

It took a whole while to get this code to run. I’ve experience so many problems with my version of python and the version of the telegram bot extension that I was using. Some computers can’t run the code for some reason. I am not sure. As of right now, one of my peer’s computer can run it and the bot will only work when the code is being ran.

**VIII. System Results and Evaluation**

Upon completion, the Weather\_Bot demonstrated efficient and accurate delivery of real-time weather forecasts. User experience evaluations highlighted the simplicity and convenience of accessing weather information through the Telegram platform. This journey had also shown me how simple it is nowadays to create bots that fetch many varieties of information from free sources.

**IX. Conclusion and Future Development**

The successful implementation of Weather\_Bot signifies a significant step in providing accessible weather forecasts to users (especially technologically handicapped ones). Future developments could include enhancements to user interactions, such as implementing additional weather-related commands or expanding the bot's capabilities.