**A PROJECT REPORT**

**ON**

**Weather Forecasting using Python**

Project report submitted in partial fulfillment of the requirement for the award of the Degree of

Bachelor of Science

In

Computer Science and Cognitive Systems

By

**B ABHINAV 111720036028**

**C SRIKANTH 111720036031**

Logo

Description automatically generated with medium confidence

**Department of Computer Science and Cognitive Systems**

**Loyola Academy**

(Autonomous and affiliated to Osmania University)

Re-accredited by “NAAC” with “A”

A “College with Potential for Excellence” by UGC.

Secunderabad

2022

**LOYOLA ACADEMY**

Old Alwal, Secunderabad – 500010

(An Autonomous Degree College affiliated to Osmania University)

Accredited by “NAAC” with “A” Grade

A “College with Potential for Excellence” by UGC.

Logo

Description automatically generated with medium confidence

Department of Computer Science and Cognitive Systems

**CERTIFICATE**

This is to certify that the project report entitled for **“Weather Forecasting using Python”** is a record of bonafide work carried out during 2nd Year, 2nd Semester as Partial fulfilment for the award of **DEGREE OF BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND COGNITIVE SYSTEMS** during the academic year 2021-22.

The results embodied in this project report have not been submitted to any other University of any Degree or Diploma.

**NAME: B ABHINAV and C. SRIKANTH**

**CLASS: NCSC**

**UID: 111720036028 and 111720036031**

HEAD OF THE DEPARTMENT PRINCIPAL

EXTERNAL EXAMNIER INTERNAL EXAMINER

**DECLARATION**

We hereby declare that the project entitled **“WEATHER FORECASTING using PYTHON”** done by **B ABHINAV** (111720036028) and **C SRIKANTH** (111720036031) submitted to the HOD, **Mrs. A VARALAKSHMI** of the Department of Computer Science and Cognitive Systems, Loyola Academy UG PG College is a record of original work done by us.

The project has been successfully completed and submitted in partial fulfilment of the requirements for the award of “**DEGREE OF BACHELOR OF SCIECE IN COMPUTER SCIENCE AND COGNITIVE SYSTEMS**” from “Loyola Academy UG PG College”, affiliated to “Osmania University, Hyderabad” in an authentic way and has not been submitted in any other university or institution for the award of degree or diploma.

**B ABHINAV C SRIKANTH**

(111720036028) (111720036031)

**ACKNOWLEDGEMENT**

At the outset, I express my gratitude to the Almighty Lord for showering his grace and blessings upon me to complete this project.

Although our names appear on the cover of this book, many people had contributed in some form or the other to the development of this project. We could not do this project without the assistance or support of everyone, We Thank you all!

I wish to place on my record, our deep sense of gratitude to my Project Guide, **Mrs. A VARALAKSHMI** for their constant motivation and valuable help through the project work.

We also extend our thanks to other faculties for their cooperation during my course.

Finally, we would like to thank our friends for their cooperation to complete this project.

Yours Sincerely

B. Abhinav

C. Srikanth

**Loyola Academy**

(Autonomous and affiliated to Osmania University)

Re-accredited by “NAAC” with ‘A’ grade

A “College with Potential for Excellence” by UGC

**CERTIFICATE**

This is to certify that the following students have successfully completed the project titled **“Weather Forecasting using Python (In House)”** in Loyola Academy, Alwal.

|  |  |
| --- | --- |
| Name of Student | Reg. No |
| **B. Abhinav**  **C. Srikanth** | 111720036028  111720036031 |

They have done this project under the guidance and supervision of Mrs. A. Varalakshmi, HOD of B.Sc. Computer Science and Cognitive Systems, Loyola Academy. The project was completed to our satisfaction, and they showed keen interest and dedication to the project. The project duration is from December 2021 to April 2022. We place our appreciation on records for her best effort.

Mrs. A. Varalakshmi

B.Sc. Computer Science and Cognitive Systems (HOD)

Loyola Academy UG PG College

**INDEX**

1. Introduction
2. Abstract
3. Existing System and Proposed System
4. Hardware and Software Requirements
5. System Analysis
6. System Design
   1. Use Case Diagram
   2. Sequence Diagram
   3. Activity Diagram
7. Implementation
   1. Packages and Modules
   2. Code
8. Output
9. Conclusion
10. Future Scope

**INTRODUCTION**

Weather Forecasting is an application of science and technology to predict the conditions of the atmosphere for a given location and time. People have attempted to predict the weather informally for a millenia and formally since the 19th Century. Weather Forecasts are made by collecting quantitative data about the current state of the atmosphere, land, and ocean and using meteorology to project how the atmosphere will change at a given place.

Once calculated manually based upon changes in barometric pressure, current weather conditions, and sky condition or cloud cover, weather forecasting now relies on computer-based models that take many atmospheric factors into account. Human input is still required to pick the best possible forecast model to base the forecast upon, which involves pattern recognition skills, interconnections, knowledge of model accuracy and performance, and knowledge of model biases. The inaccuracy of forecasting is due to the chaotic nature of the atmosphere, the massive computational power required to solve the equations that describe the atmosphere, the land, and the ocean, the error involved in measuring the initial conditions, and an incomplete understanding of atmospheric and related processes. Hence, forecasts become less accurate as the difference between current time and the time for which the forecast is being made (the range of the forecast) increases. The use of ensembles and model consensus help narrow the error and provide confidence lever in the forecast.

This project focuses on providing the most accurate weather forecast, making use of the most reliable Weather APIs’ that can provide information with the much needed accuracy and precision at any place and any time, about any place (city).

**ABSTRACT**

Weather Forecasting is the application of science and technology to predict the state of the atmosphere for a given location. Ancient weather forecasting methods usually relied on observed patterns of events, also termed pattern recognition. For example, it might be observed that if the sunset was a particularly red, the following day often brought fair weather. However, not all these predictions prove reliable.

Here this system will predict weather based on parameters such as temperature, humidity and wind. User will enter current temperature, humidity, and wind. System will take this parameter and will predict weather (rainfall in inches) from weather API. The OpenWeatherMap Weather API provides a simple way to import weather data and climate information into applications and back-end systems. In addition, the web services allow developers to easily integrate weather data into web sites and other development projects.

The Weather API offers the easiest way to retrieve historical, forecast and statistical forecast data. It offers daily summaries, hourly detail, current conditions and weather alerts in a single, continuous result. The Weather API is perfect for anyone transitioning from the Dark Sky APIs including the forecast and time machine API.

**Team Members:**

B. Abhinav 111720036028

C. Srikanth 111720036031

**EXISTING SYSTEM and PROPOSED SYSTEM**

**Existing System:**

There is existing system like Google weather forecast, weather reports, windows weather report.

**Proposed System:**

It is based on two existing forecasting systems known as **HARMONIE** and **HIRLAM.** The meteorologists then produce a forecast based on the probability of what the weather is likely to be at a certain place and at a certain time. This is called “**Probabilistic Forecasting”.**

**HARDWARE and SOFTWARE REQUIREMENTS**

**Hardware Requirements:**

RAM: 1GB or more

Processor: any modern processor (AMD and intel)

**Software Requirements:**

Operating System: Windows 7 and above

IDLE: PyCharm, VSC, or IDLE (3.8)

**SYSTEM ANALYSIS**

**Problem Definition:**

This project aims at deploying a program that is able to provide the weather forecast about a place, with accuracy and prediction. With all the necessary information like the Maximum Temperature, Minimum Temperature, Wind Speed, Humidity, etc.

**SYSTEM DESIGN**

System Design is the process of defining the architecture, modules, interfaces, and data for a system in order to satisfy specific requirements of a business organization, through the engineering of a coherent system.

It is the core of the process of Software Engineering and is applied irrespecitve of the development procedure used and area of application.

**Architectural Design:**

The Architectural design of a system emphasizes the design of the system architecture that describes the structure, the behaviour and more views of that system and analysis.

**Logical Design:**

The Logical Design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modeling of the actual system. In the context of systems, designs are included. Logical design includes entity-relationship diagrams (ER diagrams).

**Physical Deisgn:**

The Physical Design relates to the actual input and output processes of the system. This is explained in terms of how data is input into a system, how it is verified or authenticated, how it is processed and how it is displayed.

In Physical Design, the following requirements about the system are decided:

* Input Requirements
* Output Requirements
* Storage Requirements
* Processing Requirements
* System Control and Backup or Recovery

Put another way, the physical portion of system design can generally be broken down into three sub-tasks:

* User Interface Deisgn
* Data Design
* Process Design

User Interface design is concerned with how users add information to the system and with how the system presents informantion back to them. Data Design is concerned with how the data is represented and stored within the system. Finally, Process Design is concerned with hoe data moves through the system, with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the system design phase, documentation phase, documentation describing the three sub-tasks is produced and made available for use in the next phase.

Physical design, in this context, does not refer to the tangible physical design of an information system. To use an analogy, a personal computer’s physical design involves input via a keyboard, processing within the CPU, output via a monitor, printer, etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc. It involves a detailed design of a user and a product database structure processor and a control processor. The H/S ersonal specification is developed for the proposed system.

**UML DIAGRAMS:**

* The Unified Modelling Language (UML) is a general-purpose development modelling language that provides a standard way to visualize system design.
* It is used to specify, visualize, modify, construct and document the components of an object-oriented software intensive system under development.
* UML combines the features of data modelling (via entity-relationship diagrams), bussiness modelling (workflows), object modelling and component modelling.
* It can be used for any process, throughout the software development life cycle and across any implementation technology.
* UML is standard modelling language and can be used for modelling concurrent and distributed systems.
  + Actors and activities
  + Bussiness processes
  + Logical and reusable software components
  + Programming statements and database schemas

UML is a modern approach to modelling and documenting software. In fact, it’s one of the most popular business process modelling techniques. It is based on diagrammatic representatives of software components.

**Types of UML diagrams:**

Diagram

Description automatically generated

There are several types of UML diagrams and each one of them serves a different purpose regardless of whether it is being designed before the implementation or after (as apart of documentation).

The two broadest categories that encompass all other types are Behavioral UML diagram and Structural UML diagrams. As the name suggests, some UML diagrams try to analyse and depict the structure of a system or process, whereas other describe the behaviour of the system, its actors, and its building components.

**The different types are broken down as follows:**

* Behavioural UML Diagram
* Activity Diagram
* Use Case Diagram
* Interaction Overview Diagram
* Timing Diagram
* State Machine Diagram
* Communication Diagram
* Sequence Diagram
* Structural UML Diagram
  + Class Diagram
  + Object Diagram
  + Component Diagram
  + Composite Structure Diagram
  + Deployment Diagram
  + Package Diagram
  + Profile Diagram

Not all of the 14 different types of UML diagrams are used on regular basis when documenting systems and/or architecture. The Pareto principle seems to apply in terms of UML diagram usage as well -20% of the diagrams are being used 80% of the time by developers.

The most common UML diagrams are:

* + Use case Diagram
  + Sequence Diagram
  + Activity Diagram

**USE – CASE DIAGRAM:**

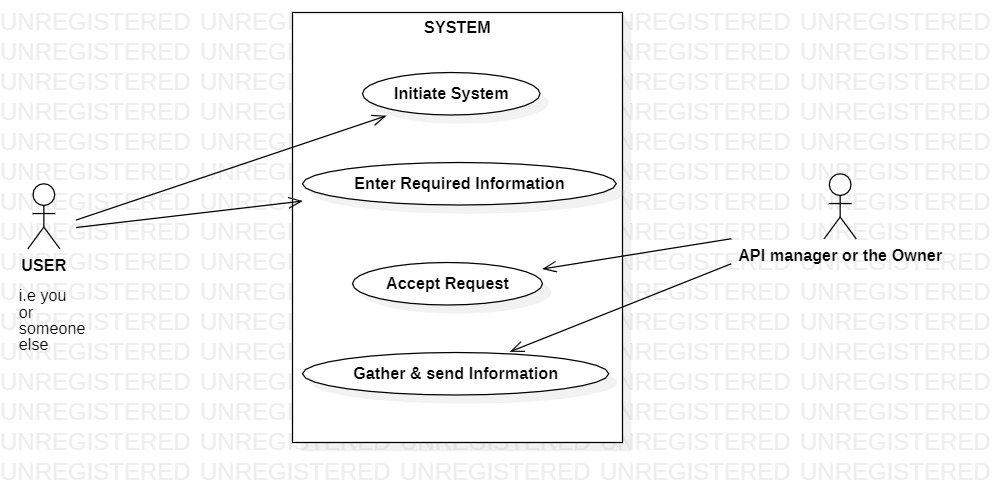
A use case diagrma at its simplest is a representation of a user’s interaction with the system that shows the realationship between the user and the different usecases in which user is involved. A usecase diagram can identify the different types of users of a system and the different usecases and will often be accompanied by the other types of diagrams as well. The use cases are represented by either circles or ellipses.

Use case diagrams can also help to provide a higher levelview of the system.

There are 3 major components in a use case diagram:

* + **Functional requirements** – represented as usecases; a verb describing an action
  + **Actors** – these are entities that interacts with the system. An actor can be a human being, an organisation or an internal or external application
  + **Relationships** – between actors and usecases -represented using straight arrows

Due to their simplistic nature, use case diagrams can be good communication tool for stakeholders. The drawings attempt to mimic the real world and provide a view for the stakeholders to undrerstand how the system is going to be designed.



**SEQUENCE DIAGRAM:**

A sequence ­­­diagram shows object interactions arranged in time sequence. I depict the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with the use case realizations in the logical view of the system under development.

A Sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

Table

Description automatically generated**ACTIVITY DIAGRAM:**

Activity diagrams can be regarded as a form of a structured flowchart combined with a traditional data flow diagram. Typical flowchart

Techniques lack constructs for expressing concurrency. In the Unified Modelling Language, activity diagrams are intended to model to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related acticities. Although activity diagrams primarily show te overall flow of control, they can also include elements showing the flow of data between the activities through one or more data stores.

Activity diagrams can be regarded as a form of a structured flowchart combined with a traditional data flow diagram. Typical flowchart techniques lack constructs for expressing concurrency. However, the join and split symbols in activity diagrams only resolve this for simple cases;

The meaning of the model is not clear when they are arbitarily combined with decisions or loops.

Diagram

Description automatically generated

**IMPLEMENTATION**

**PACKAGES and MODULES:**

**Pip:** Pip is a de facto and rcommended package-management system written in Python and is used to install and manage software packages. It connects to an online repository of public packages, called the Python Package Index.

**Requests:** Requests is a Python module that you can use to send all kinds of HTTP requests. It is an easy-to-use library with lots of features ranging from passing parameters in URLs tp sending custom headers and SSL Verification.

**Pip-PIL and Pillow:** Pillow is a Python Imaging Library (PIL), which adds support form opening, manipulating and saving images. The current version identifies and reads a large number of formats. Write support is intentionally restricted to the most commonly used interchange and presentation formats. Pillow is a fork of PIL that adds some user-friendly features.

**CODE:**

**get\_weather\_icons.py**

import os

import urllib.request

day = ['01d.png', '02d.png', '03d.png', '04d.png', '09d.png', '10d.png', '11d.png', '13n.png', '50d.png']

night = ['01n.png', '02n.png', '03n.png', '04n.png', '09n.png', '10n.png', '11n.png', '13n.png', '50n.png']

base\_url = 'https://openweathermap.org/img/w/'

img\_dir = './img/'

if not os.path.exists(img\_dir):

    os.makedirs(img\_dir)

*# Get the day weather icons*

for name in day:

    file\_name = img\_dir+name

    if not os.path.exists(file\_name):

        urllib.request.urlretrieve(base\_url+name, file\_name)

*# Repeat the same thing for night weather icons*

for name in night:

    file\_name = img\_dir+name

    if not os.path.exists(file\_name):

        urllib.request.urlretrieve(base\_url+name, file\_name)

**main.py**

import tkinter as tk

import pip.\_vendor.requests

from PIL import Image,ImageTk

root=tk.Tk()

root.title("Weather Application")

root.geometry("800x600")

*#Key: 2f5e50542ef3c6e0bce1bc1f239de81f*

*#Api url: api.openweathermap.org/data/2.5/weather?q={city name}&appid={API key}*

def format\_response(weather):

    try:

*#Information that is collected is diplayed in the following format*

        city=weather['name']

        condition=weather['weather'][0]['description']

        temp=weather['main']['temp']

        real\_feel=weather['main']['feels\_like']

        temp1=weather['main']['temp\_min']

        temp2=weather['main']['temp\_max']

        temp3=weather['main']['humidity']

        temp4=weather['main']['pressure']

        wind=weather['wind']['speed']

        coordination1=weather['coord']['lon']

        coordination2=weather['coord']['lat']

        timezone=weather['timezone']

        country=weather['sys']['country']

        final\_str='City : %s\nCondition : %s\nTemprature (°F) : %s\nReal Feel (°F) : %s\nMinimum (°F) : %s\nMaximum (°F) : %s\nHumidity : %s\nPressure : %s\nWind Speed (Kmph) : %s\nLongnitude : %s\nLatitude : %s\nTimezone (UTC) : %s\nCountry : %s'%(city,condition,temp,real\_feel,temp1,temp2,temp3,temp4,wind,coordination1,coordination2,timezone,country)

    except:

*#Reply for not entering the correct or required information*

        final\_str='There was a problem in retrieving that information'

    return final\_str

def get\_weather(city):

    weather\_key='2f5e50542ef3c6e0bce1bc1f239de81f'

    url= 'https://api.openweathermap.org/data/2.5/weather'

    params={'APPID':weather\_key,'q':city,'units':'imperial'}

    response=pip.\_vendor.requests.get(url,params)

*#print(response.json())*

    weather=response.json()

*#print(weather['name'])*

*#print(weather['weather'][0]['description'])*

*#print(weather['main']['temp'])*

    result['text']=format\_response(weather)

    icon\_name=weather['weather'][0]['icon']

    open\_image(icon\_name)

def open\_image(icon):

    size=int(frame\_two.winfo\_height()\*0.25)

    img=ImageTk.PhotoImage(Image.open('./img/'+icon+'.png').resize((size,size)))

    weather\_icon.delete('all')

    weather\_icon.create\_image(0,0,anchor='nw',image=img)

    weather\_icon.image=img

*#the background image is being called here.*

img=Image.open('./bg.jpg')

img=img.resize((800,600),Image.ANTIALIAS)

img\_photo=ImageTk.PhotoImage(img)

*#BG Label*

bg\_lbl=tk.Label(root,image=img\_photo)

bg\_lbl.place(x=0,y=0,width=800,height=600)

*#title*

heading\_title=tk.Label(bg\_lbl,text='Search over 200,000 cities!',fg='black',bg='lightgray',font=('times new roman',18,'bold'))

heading\_title.place(x=155,y=70)

frame\_one=tk.Frame(bg\_lbl,bg="#555555",bd=5)

frame\_one.place(x=150,y=110,width=500,height=60)

*#Text box for the city name to be entered*

txt\_box=tk.Entry(frame\_one, font=('times new roman',30),width=17)

txt\_box.grid(row=0,column=0,sticky='W')

*#button used to obtain the information*

btn=tk.Button(frame\_one,text='Get Weather',fg='black',font=('times new roman',16,'bold'),command=lambda: get\_weather(txt\_box.get()))

btn.grid(row=0,column=1,padx=10)

frame\_two=tk.Frame(bg\_lbl,bg="#555555",bd=5)

frame\_two.place(x=150,y=200,width=500,height=320)

result=tk.Label(frame\_two,font=40,bg='white',justify='left',anchor='nw')

result.place(relwidth=1,relheight=1)

weather\_icon=tk.Canvas(result,bg='white',bd=0,highlightthickness=0)

weather\_icon.place(x=300,y=20,relwidth=1,relheight=0.50)

root.mainloop()

**OUTPUT**

**Step:1** First, we run the get\_weather\_icons.py, just to be sure that the program doesn’t have any errors.

Graphical user interface, text, application

Description automatically generated

This would be the output after running the get\_weather\_icons.py.

**Step:2** Now, we will run the main.py file.

Graphical user interface

Description automatically generated

This would be the window that will open, once we run the main.py file. As you can see, we have the text field for entering the city name, and the text field where the information will be displayed.

**Step:3** Now, we enter the required details, i.e. the city name or the state name.

Text

Description automatically generated

As you can see, we get all the required information like the Maximum Temperature, Minimum Temperature, Humidity, Wind Speed, etc.

We have also checked the accuracy of the program by comparing it with other sources like Google Weather.

Graphical user interface, application

Description automatically generated

We checked the weather of some other cities as well, as shown in the images that follow,

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

We have also tried some invalid inputs like, Antarctica which is an entire continent and Telengana, which is a state.

Graphical user interface

Description automatically generated

Graphical user interface, text

Description automatically generated

**CONCLUSION**

As the results above show that, this program is very much as accurate as any other weather forecasting service on the internet. With the enhancements and upgrades that we have proposed, this program could be an alternative for any other weather forecasting web-app or website on the internet.

**FUTURE SCOPE**

Future enhancements for this project include deploying a website, implementing authentication, designing a database to store the users data (credentials) and search locations for future recommendations.

Other features to be introduced include being able to change the data that you want to see, the metric system used by the website (changing the temperature from Fahrenheit to Celsius).

Being able to check the Hourly Forecast, Monthly Forecast, Air Quality, Auto-Detecting your current location, etc.

This project is one of the most basic weather forecasting programs, and has a lot of scope to be improved in the coming days.