Weather Forecasting

A Mini Project Report Submitted in partial fulfilment of the Requirement For

III Semester of

MASTER OF COMPUTER APPLICATIONS

Submitted By

KEDARASETTI RAMYA KEERTHI

20551F0027

Under the Esteemed Guidance of

Dr. R. Tamilkodi

HOD, MCA Dept.

DEPARTMENT OF COMPUTER APPLICATIONS

GODAVARI INSTITUTE OF ENGINEERING & TECHNOLOGY (A)

[Affiliated to J.N.T.U.K, Kakinada |Recognized by UGC under section 2(f) & 12(B) | NAAC 'A+' GRADE]

NH-16 CHAITANYA KNOWLEDGE CITY, RAJAHMUNDRY (A.P)

2022

GODAVARI INSTITUTE OF ENGINEERING & TECHNOLOGY (A)

[Affiliated to J.N.T.U.K, Kakinada |Recognized by UGC under section 2(f) & 12(B) | NAAC 'A+' GRADE]
NH-16 CHAITANYA KNOWLEDGE CITY, RAJAHMUNDRY (A.P)
2022

DEPARTMENT OF COMPUTER APPLICATIONS

CERTIFICATE

This is to certify that this Mini project **"Weather Forecasting"** is being submitted by **KEDARASETTI RAMYA KEERTHI (20551F0027)** in partial fulfilment of the requirements for III Semester of **MASTER OF COMPUTER APPLICATIONS** in the academic year 2021-2022 to the **GIET AUTONOMOUS** is a bonafide work carried out her under my guidance and supervision.

The result embodied in this project has not been submitted to any other University or Institute for the award of degree

CO-ORDINATOR **SUPERVISOR**

ACKNOWLEDGEMENT

It is privilege for me to have under taken the project **"WEATHER FORECASTING"** in **GIET (A), Rajahmundry**.

I avail this opportunity to express my deep sense of gratitude and heart full thanks to **Mr. K. SASI KIRAN VARMA**, Honorable managing director of **GODAVARI INSTITUTE OF ENGINEERING & TECHNOLOGY, RAJAHMUNDRY.**

I am thankful to beloved Principal **Dr. P.M.M.S.SARMA,** for permitting encouraging me in doing this project.

I am deeply indebted to **Dr. R. TAMILKODI**, Head of the Department of Computer Applications, GIET, whose motivation in the field of software development made me overcome all the hardships during the course of study.

I am heartily thankful to my internal guide **Dr. R. Tamilkodi** HOD, MCA dept, GIET, for valuable guidance, keen interest, open minded discussion and his constant encouragement throughout the project work.

Finally I would like to extend my heartfelt thanks to my **TEACHING AND NON TEACHING STAFF** who had shared their technical knowledge and their encouragement were always there as a source of strength and inspiration.

Although a leaflet title "Acknowledgement" cannot represent my true feelings for all these persons, I feel very much thankful to all of them and also to my **PARENTS** and **FRIENDS** for encouraging and giving me all the moral support required for making this endeavour a reality.

KEDARASETTI RAMYA KEERTHI

(20551F0027)

DECLARATION

I **KEDARASETTI RAMYA KEERTHI (PIN 20551F0027)** declare that the mini project title **"WEATHER FORECASTING"** is a bona fide work carried out by me and has not been submitted to any other university or college for the award of degree.

KEDARASETTI RAMYA KEERTHI

(20551F0027)

TABLE OF CONTENTS

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 of events, also termed pattern recognisation. Here the present system weather forecasting will predict weather based on parameters such as temperature, humidity and wind. There is a vast variety of end uses to weather forecasts. Weather warnings are important because they are used to protect life and property.

The traditional forecast process employed by most NHMs(Natural History Museum) involves forecasters producing text based , sensible ,weather- element forecast products(e.g.,maximum and minimum temperature ,cloud cover) using numerical weather prediction(NWP) output as guidance. The process is typically schedule driven, product oriented and labour intensive.

CHAPTER-1

INTRODUCTION

1.1 Purpose

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 mainly 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.[1] Human input is still required to pick the best possible forecast model to base the forecast upon, which involves pattern recognition skills, teleconnections, knowledge of model 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 level in the forecast.

There is a vast variety of end uses to weather forecasts. Weather warnings are important forecasts because they are used to protect life and property. Forecasts based on temperature and precipitation are important to agriculture, and therefore to traders within commodity markets. Temperature forecasts are used by utility companies to estimate demand over coming days. On an everyday basis, many use weather forecasts to determine what to wear on a given day. Since outdoor activities are severely curtailed by heavy rain, snow and wind forecasts can be used to plan activities around these events, and to plan ahead and survive them.

The aim of weather prediction is to provide information people and organizations can use to reduce weather- related losses and enhance social benefits, including protection of life and property , public health and safety, and support of economic prosperity and quality of life.

1.2 Scope

Weather and Forecasting (WAF)(ISSN:0882-8156; eISSN:1520-0434) publishes research that is relevant to operational forecasting. The scope of WAF(Weather and Forecasting) includes research relevant to forecast lead times ranging from short – term “nowcasts” through seasonal time scales out to approximately two years.

.

CHAPTER-2

SYSTEM ANALYSIS

2.1 Existing System

. The traditional forecast process employed by most NHMs(Natural History Museum) involves forecasters producing text based , sensible ,weather- element forecast products(e.g.,maximum and minimum temperature ,cloud cover) using numerical weather prediction(NWP) output as guidance. The process is typically schedule driven, product oriented and labour intensive.

2.1.1 Disadvantages

- It consists less features compared to the proposed system.
- It consists complex algorithms.
- Illiterates does not understood the textile information.

2.2 Proposed System

This system provides instant information availability and this system consists more features compared to existing system. In this system user enter the city name then it will display the weather with images. It shows the image in console oriented wttr.in uses wego for visualization and various data sources for weather forecast information. It supports various information representation methods like terminal-oriented ANSI-sequences for console HTTP clients(curl, httpie, or wget),HTML for web browsers.

Advantages

1.Illiterates are also understand the weather information by seeing weather images.

2.Accuarte results.

Modules Description

REQUESTS MODULE

Requests library is one of the integral part of Python for making HTTP requests to a specified URL. Whether it be REST APIs or Web Scrapping, requests is must to be learned for proceeding further with these technologies. When one makes a request to a URI, it returns a response. Python requests provides inbuilt functionalities for managing both the request and response.

Requests library is one of the integral part of python for making HTTP requests to a specified URL. Whether it be Reset API's or web scrapping, requests is most to be learned for proceeding further with these technologies.

When one makes a request to URL ,it receives a response. Python requests provides inbuilt functionalities for managing both the request and response

National Weather Services

National weather services uses super computers around the clock to accurately produce forecasts, watches, warnings and a whole host of data for the public. These computers make use of virtually all observational data that the NWS collects. This data comes from satellites, weather balloons, buoys ,radar and more. In addition to helping the NWS forecast weather, the computer data is also made available worldwide, which can help other countries predict their weather as well.

FEASIBILITY STUDY

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time.

There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operational Feasibility
- Economical Feasibility

ECONOMIC FEASIBILITY

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

OPERATIONAL FEASIBILITY

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

TECHNICAL FEASIBILITY

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipments have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to the users.

CHAPTER-3

SYSTEM REQUIREMENTS

3.1 FUNCTIONAL REQUIREMENTS

User

User search the weather information and make a weather report in both image and text format.

Weather Satellites

Weather Satellites helps the process of looking over a large area, as does the network of weather observations.

Super computers

These computers make use of virtually all observational data that the NWS news collects. This data comes from satellites, weather ballons, buoys, rader and more. From the data the super computers able to help predict every kind of weather imaginable, including hurricanes, tornadoes, extreme heat and even space weather.

3.2 NON FUNCTIONAL REQUIREMENTS

The major non-functional Requirements of the system are as follows

Usability

The system is designed with completely automated process hence there is no or less user intervention.

Reliability

The system is more reliable because of the qualities that are inherited from the chosen platform java. The code built by using java is more reliable.

Performance

This system is developing in the high level languages and using the advanced front-end and back-end technologies it will give response to the end user on client system with in very less time.

Supportability

The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is having JVM, built into the system.

Implementation

The system is implemented in web environment using struts framework. The http is used as the web server and windows xp professional is used as the platform. Interface the user interface is based on Struts provides HTML Tag .

3.3 Software Requirements

Language	:	Python
Frontend	:	Python, html
IDE	:	python3
Operating System	:	windows 10
Server	:	HTTP

3.4 Hardware Requirements

Processor	:	i3
Hard Disk	:	80GB
RAM	:	4GB

CHAPTER-4

SYSTEM DESIGN

The purpose of the design phase is to plan a solution of the problem specified by the requirement document. This phase is the first step in moving from the problem domain to the solution domain. In other words, starting with what is needed, design takes us toward how to satisfy the needs. The design of a system is perhaps the most critical factor affection the quality of the software; it has a major impact on the later phase, particularly testing, maintenance. The output of this phase is the design document. This document is similar to a blueprint for the solution and is used later during implementation, testing and maintenance. The design activity is often divided into two separate phases System Design and Detailed Design. System Design also called top-level design aims to identify the modules that should be in the system, the specifications of these modules, and how they interact with each other to produce the desired results.

System Model

Introduction to UML

The unified Modeling Language (UML) is a standard language for writing software blueprints. The UML may be used to visualize, specify , construct and document the artifacts of software-intensive system.

The goal of UML is to provide a standard notation that can be used by all object - oriented methods and to select and integrate the best elements .UML is itself does not prescribe or advice on how to use that notation in a software development process or as part of an object - design methodology. The UML is more than just bunch of graphical symbols. Rather , behind each symbol in the UML notation is well-defined semantics.

The system development focuses on three different models of the system.

- Functional model
- Object model
- Dynamic model

Functional model : In UML is represented with use case diagrams , describing the functionality of the system from user point of view.

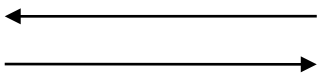
Object model: In UML is represented with class diagrams , describing the structure of the system in terms of objects , attributes , associations and operations

Data Flow Diagrams

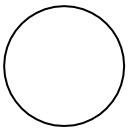
A graphical tool used to describe and analyze the moment of data through a system manual or automated including the process, stores of data, and delays in the system. Data Flow Diagrams are the central tool and the basis from which other components are developed. The transformation of data from input to output, through processes, may be described logically and independently of the physical components associated with the system. The DFD is also know as a data flow graph or a bubble chart.

DFDs are the model of the proposed system. They clearly should show the requirements on which the new system should be built. Later during design activity this is taken as the basis for drawing the system's structure charts. The Basic Notation used to create a DFD's are as follows:

1. Dataflow: Data move in a specific direction from an origin to a destination.

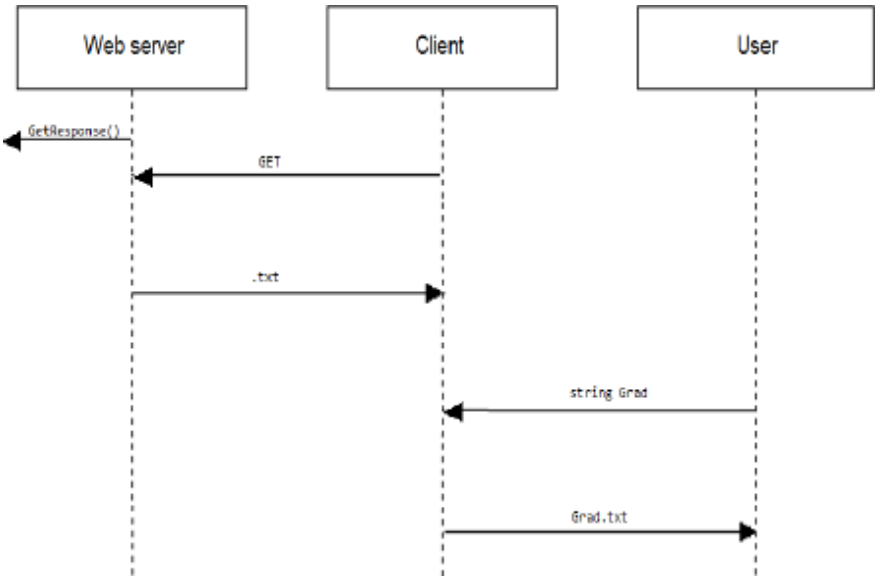


2. Process: People, procedures, or devices that use or produce (Transform) Data. The physical component is not identified.



3.Rectangle: The vertical rectangles denote execution specification.

Sequence diagram



CHAPTER-5

SYSTEM IMPLEMENTATION

5.1 Technology Description

Python:

Python is an interpreted high-level general–purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, Including structured(particularly procedural),object-oriented and functional programming. It is often described as a “batteries included” language due to its comprehensive standard library.

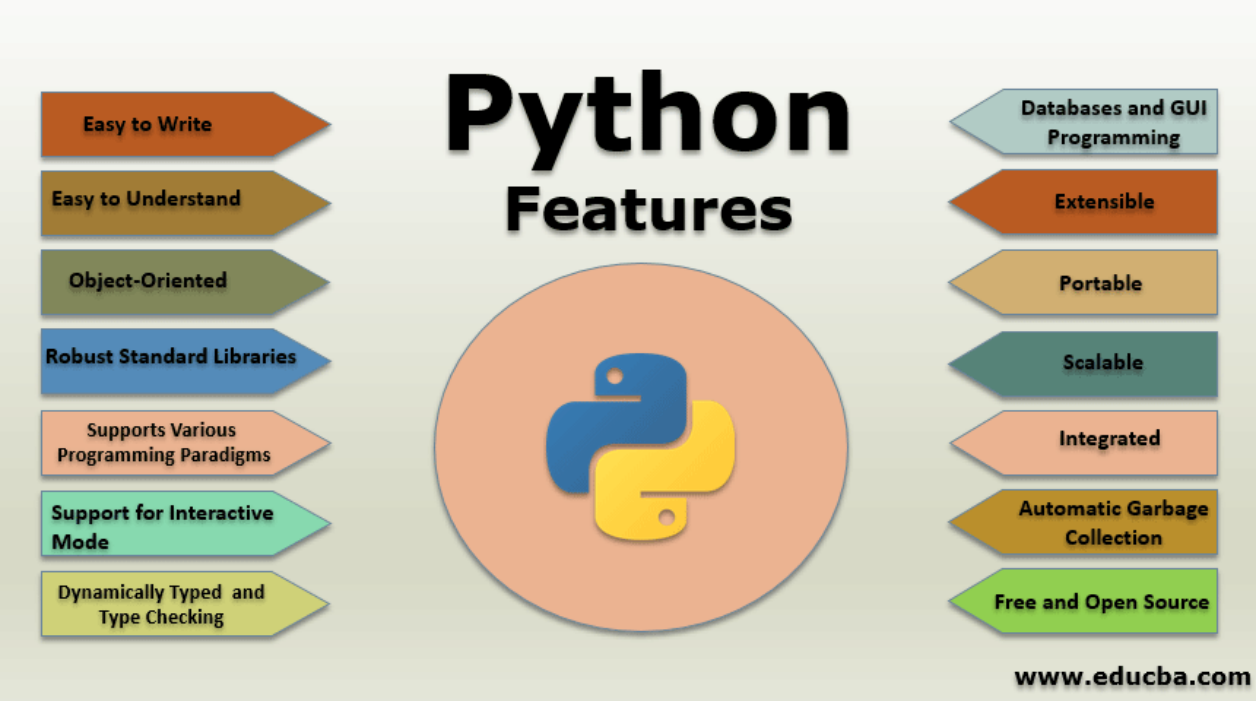
Guido van Rossum began working on Python in the late 1960’s as a successor to the ABC programming language, and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features, such as list comprehensions and a cycle-detecting garbage collection system (in addition to reference counting).Python 3.0 was released in 2008 and was a major revision of the language that is not completely backward-compatible. Python 2 was discontinue with version 2.7.18 in 2020 .

Features of Python

Python is a dynamic, high level, free open source and interpreted programming language. Itsupports object-oriented programming as well as procedural oriented programming as well as procedural oriented programming.

In Python, we don't need to declare the type of variable because it is a dynamically typed language.
For example,x=10
Here, x can be anything such as String, int etc.,

There are many features in Python, some of which are discussed below-



1. Easy to Write

- It is easy to write as compared to other programming languages. It's syntax is straightforward and much the same as the English language.
- There is no use of the semicolon or curly-bracket, the indentation defines the code block.

2. Object-Oriented Language

- Python supports object-oriented language and concepts of classes and objects come into existence.
- It supports inheritance, polymorphism, and encapsulation, etc.,

3.Robust standard Library

- It provides a vast range of libraries for the various fields such as machine learning, web developer, and also for scripting.
- There are various machine learning libraries, such as Tensor flow, Pandas, Numpy, Keras, and Pytorch.

4.Databases and GUI Programming

- Graphical User Interface is used for the developing Desktop applications, PyQT5, Tkinter, Kivy are libraries which are used for developing the web application.

5.Extensible

- It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our Python code.

Applications Of Python

1. Web Applications
2. Desktop GUI Applications
3. Console Based Applications
4. Software Development
5. Scientific and Numeric

HTML

HTML stands for hyper text markup language.HTML is the standard markup language for creating web pages. HTML describes the structure of a web page. HTML consists of a series of elements.HTML elements tell the browser how display the content. HTML elements label pieces of contents such as “this is heading” etc.,

Advantages of HTML

HTML is easy to learn

HTML is supported all browsers

HTML is the most friendly Search Engine

5.2 sample code

```
#!/usr/bin/env python
# vim: set encoding=utf-8

from gevent.pywsgi import WSGIServer
from gevent.monkey import patch_all
patch_all()

# pylint: disable=wrong-import-position,wrong-import-order
import sys
import os
import jinja2

from flask import Flask, request, send_from_directory, send_file
APP = Flask(__name__)

MYDIR = os.path.abspath(
    os.path.dirname(os.path.dirname(__file__)))
sys.path.append("%s/lib/" % MYDIR)

import wttr_srv
from globals import TEMPLATES, STATIC, LISTEN_HOST, LISTEN_PORT
# pylint: enable=wrong-import-position,wrong-import-order

from view.v3 import v3_file

MY_LOADER = jinja2.ChoiceLoader([
    APP.jinja_loader,
    jinja2.FileSystemLoader(TEMPLATES),])
```

```
PP.jinja_loader = MY_LOADER

@APP.route('/v3/<string:location>')
def send_v3(location):
    filepath = v3_file(location)
    if filepath.startswith("ERROR"):
        return filepath.rstrip("\n") + "\n"
    return send_file(filepath)

@APP.route('/files/<path:path>')
def send_static(path):
    "Send any static file located in /files/"
    return send_from_directory(STATIC, path)

@APP.route('/favicon.ico')
def send_favicon():
    "Send static file favicon.ico"
    return send_from_directory(STATIC, 'favicon.ico')

@APP.route('/malformed-response.html')
def send_malformed():
    "Send static file malformed-response.html"
    return send_from_directory(STATIC, 'malformed-response.html')

@APP.route("/")
@APP.route("/<string:location>")
def wttr(location=None):
    "Main function wrapper"
return wttr_srv.wttr(location, request)
SERVER = WSGIServer(
```

```
    (LISTEN_HOST, int(os.environ.get('WTTRIN_SRV_PORT', LISTEN_PORT))),
    APP)
SERVER.serve_forever()
        #!/usr/bin/env python
        # vim: set encoding=utf-8

        from gevent.pywsgi import WSGIServer
        from gevent.monkey import patch_all
        patch_all()

        # pylint: disable=wrong-import-position,wrong-import-order
        import sys
        import os
        import jinja2

        from flask import Flask, request, send_from_directory, send_file
        APP = Flask(__name__)

        MYDIR = os.path.abspath(
            os.path.dirname(os.path.dirname('__file__')))
        sys.path.append("%s/lib/" % MYDIR)

        import wttr_srv
        from globals import TEMPLATES, STATIC, LISTEN_HOST,
        LISTEN_PORT
        # pylint: enable=wrong-import-position,wrong-import-order

        from view.v3 import v3_file

        MY_LOADER = jinja2.ChoiceLoader([
            APP.jinja_loader,
```

```
    jinja2.FileSystemLoader(TEMPLATES),
])

APP.jinja_loader = MY_LOADER

@APP.route('/v3/<string:location>')
def send_v3(location):
    filepath = v3_file(location)
    if filepath.startswith("ERROR"):
        return filepath.rstrip("\n") + "\n"
    return send_file(filepath)

@APP.route('/files/<path:path>')
def send_static(path):
    "Send any static file located in /files/"
    return send_from_directory(STATIC, path)

@APP.route('/favicon.ico')
def send_favicon():
    "Send static file favicon.ico"
    return send_from_directory(STATIC, 'favicon.ico')

@APP.route('/malformed-response.html')
def send_malformed():
    "Send static file malformed-response.html"
    return send_from_directory(STATIC, 'malformed-response.html')

@APP.route("/")
@APP.route("/<string:location>")
def wttr(location=None):
    "Main function wrapper"
```

```
    return wttr_srv.wttr(location, request)


SERVER = WSGIServer(
    (LISTEN_HOST,                int(os.environ.get('WTTRIN_SRV_PORT',
LISTEN_PORT))),
    APP)
SERVER.serve_forever()
        #vim: fileencoding=utf-8


        """
        The proxy server acts as a backend for the wttr.in service.
        It caches the answers and handles various data sources
        transforming their
        answers into format supported by the wttr.in service.
        If WTTRIN_TEST is specified, it works in a special test mode:
        it does not fetch and does not store the data in the cache,
        but is using the fake data from "test/proxy-data".
        """
        from__future__import print_function


        from gevent.pywsgi import WSGIServer
        from gevent.monkey import patch_all
        patch_all()


        # pylint: disable=wrong-import-position,wrong-import-order
        import sys
        import os
        import time
        import json
        import hashlib
```

```
import requests
import cyrtranslit

from flask import Flask, request
APP = Flask(__name__)



MYDIR = os.path.abspath(
    os.path.dirname(os.path.dirname(__file__)))
sys.path.append("%s/lib/" % MYDIR)


from globals import PROXY_CACHEDIR, PROXY_HOST,
PROXY_PORT, USE_METNO, USER_AGENT,
MISSING_TRANSLATION_LOG
from metno import create_standard_json_from_metno,
metno_request
from translations import PROXY_LANGS
# pylint: enable=wrong-import-position


def is_testmode():
    """Server is running in the wttr.in test mode"""


    return "WTTRIN_TEST" in os.environ


def load_translations():
    """
    load all translations
    """
    translations = {}


    for f_name in PROXY_LANGS:
        f_name = 'share/translations/%s.txt' % f_name
```

```
        translation = {}
        lang = f_name.split('/')[-1].split('.', 1)[0]
        with open(f_name, "r") as f_file:
            for line in f_file:
                if ':' not in line:
                    continue
                if line.count(':') == 3:
                    _, trans, orig, _ = line.strip().split(':', 4)
                else:
                    _, trans, orig = line.strip().split(':', 3)
                trans = trans.strip()
                orig = orig.strip()

                translation[orig.lower()] = trans
        translations[lang] = translation
    return translations
TRANSLATIONS = load_translations()

def _is_metno():
    return USE_METNO

def _find_srv_for_query(path, query):    # pylint: disable=unused-
argument
    if _is_metno():
        return 'https://api.met.no'
    return 'http://api.worldweatheronline.com'

def _cache_file(path, query):
    """Return cache file name for specified `path` and `query`
    and for the current time.
    Do smooth load on the server, expiration time
```

```
    is slightly varied basing on the path+query sha1 hash digest.
    """

    digest = hashlib.sha1(("%s %s" % (path, query)).encode("utf-
8")).hexdigest()
    digest_number = ord(digest[0].upper())
    expiry_interval = 60*(digest_number+90)

    timestamp = "%010d" %
(int(time.time())//expiry_interval*expiry_interval)
    filename = os.path.join(PROXY_CACHEDIR, timestamp,
path, query)

    return filename


def _load_content_and_headers(path, query):
    if is_testmode():
        cache_file = "test/proxy-data/data1"
    else:
        cache_file = _cache_file(path, query)
    try:
        return (open(cache_file, 'r').read(),
                json.loads(open(cache_file+".headers", 'r').read()))
    except IOError:
        return None, None

def _touch_empty_file(path, query):
    cache_file = _cache_file(path, query)
    cache_dir = os.path.dirname(cache_file)
    if not os.path.exists(cache_dir):
        os.makedirs(cache_dir)
```

```
    open(cache_file, 'w').write("")

def _save_content_and_headers(path, query, content, headers):
    cache_file = _cache_file(path, query)
    cache_dir = os.path.dirname(cache_file)
    if not os.path.exists(cache_dir):
        os.makedirs(cache_dir)
    open(cache_file + ".headers", 'w').write(json.dumps(headers))
    open(cache_file, 'wb').write(content)

def translate(text, lang):
    """
    Translate `text` into `lang`.
    If `text` is comma-separated, translate each term independently.
    If no translation found, leave it untouched.
    """

    def _log_unknown_translation(lang, text):
        with open(MISSING_TRANSLATION_LOG % lang, "a") as
f_missing_translation:
            f_missing_translation.write(text+"\n")

    if "," in text:
        terms = text.split(",")
        translated_terms = [translate(term.strip(), lang) for term in
terms]
        return ", ".join(translated_terms)

    if lang not in TRANSLATIONS:
        _log_unknown_translation(lang,
"UNKNOWN_LANGUAGE")
        return text
```

```
    if text.lower() not in TRANSLATIONS.get(lang, {}):
        _log_unknown_translation(lang, text)
        return text

    translated = TRANSLATIONS.get(lang, {}).get(text.lower(),
text)
    return translated


def cyr(to_translate):
    """
    Transliterate `to_translate` from latin into cyrillic
    """
    return cyrtranslit.to_cyrillic(to_translate)


def _patch_greek(original):
    return original.replace(u"Ηλιόλουστη/ο", u"Ηλιόλουστη")


def add_translations(content, lang):
    """
    Add `lang` translation to `content` (JSON)
    returned by the data source
    """

    if content == "{}":
        return {}

    languages_to_translate = TRANSLATIONS.keys()
    try:
        d = json.loads(content)         # pylint: disable=invalid-name
    except (ValueError, TypeError) as exception:
```

```
        print("---")
        print(exception)
        print("---")
        return {}

    try:
        weather_condition = d['data']['current_condition'
            ][0]['weatherDesc'][0]['value'].capitalize()
        d['data']['current_condition'][0]['weatherDesc'][0]['value'] = \
            weather_condition
        if lang in languages_to_translate:
            d['data']['current_condition'][0]['lang_%s' % lang] = \
                [{'value': translate(weather_condition, lang)}]
        elif lang == 'sr':
            d['data']['current_condition'][0]['lang_%s' % lang] = \
                [{'value': cyr(
                    d['data']['current_condition'][0]['lang_%s' % lang][0]['value']\
                    )}]
        elif lang == 'el':
            d['data']['current_condition'][0]['lang_%s' % lang] = \
                [{'value': _patch_greek(
                    d['data']['current_condition'][0]['lang_%s' % lang][0]['value']\
                    )}]
        elif lang == 'sr-lat':
            d['data']['current_condition'][0]['lang_%s' % lang] = \

[{'value':d['data']['current_condition'][0]['lang_sr'][0]['value']\
                        }]

        fixed_weather = []
```

```
        for w in d['data']['weather']: # pylint: disable=invalid-name
            fixed_hourly = []
            for h in w['hourly']:       # pylint: disable=invalid-name
                weather_condition = h['weatherDesc'][0]['value']
                if lang in languages_to_translate:
                    h['lang_%s' % lang] = \
                        [{'value': translate(weather_condition, lang)}]
                elif lang == 'sr':
                    h['lang_%s' % lang] = \
                        [{'value': cyr(h['lang_%s' % lang][0]['value'])}]
                elif lang == 'el':
                    h['lang_%s' % lang] = \
                        [{'value': _patch_greek(h['lang_%s' % lang][0]['value'])}]
                elif lang == 'sr-lat':
                    h['lang_%s' % lang] = \
                        [{'value': h['lang_sr'][0]['value']}]
                fixed_hourly.append(h)
            w['hourly'] = fixed_hourly
            fixed_weather.append(w)
        d['data']['weather'] = fixed_weather

        content = json.dumps(d)
    except (IndexError, ValueError) as exception:
        print(exception)
    return content

def _fetch_content_and_headers(path, query_string, **kwargs):
    content, headers = _load_content_and_headers(path, query_string)
```

```
    if content is None:
        srv = _find_srv_for_query(path, query_string)
        url = '%s/%s?%s' % (srv, path, query_string)

        attempts = 10
        response = None
        while attempts:
            try:
                response = requests.get(url, timeout=2, **kwargs)
            except requests.ReadTimeout:
                attempts -= 1
                continue
            try:
                json.loads(response.content)
                break
            except ValueError:
                attempts -= 1

        _touch_empty_file(path, query_string)
        if response:
            headers = {}
            headers['Content-Type'] = response.headers['content-
type']
            _save_content_and_headers(path, query_string,
response.content, headers)
            content = response.content
        else:
            content = "{}"
    else:
        print("cache found")
    return content, headers
```

```
@APP.route("/<path:path>")
def proxy(path):
    """
    Main proxy function. Handles incoming HTTP queries.
    """

    lang = request.args.get('lang', 'en')
    query_string = request.query_string.decode("utf-8")
    query_string = query_string.replace('sr-lat', 'sr')
    query_string = query_string.replace('lang=None', 'lang=en')
    content = ""
    headers = ""
    if _is_metno():
        path, query, days = metno_request(path, query_string)
        if USER_AGENT == '':
            raise ValueError('User agent must be set to adhere to metno
ToS: https://api.met.no/doc/TermsOfService')
        content, headers = _fetch_content_and_headers(path, query,
headers={
                                'User-Agent': USER_AGENT
                            })
        content = create_standard_json_from_metno(content, days)
    else:
        # WWO tweaks
        query_string += "&extra=localObsTime"
        query_string += "&includelocation=yes"
        content, headers = _fetch_content_and_headers(path,
query_string)

    content = add_translations(content, lang)
```

```
    return content, 200, headers


if__name__== "_main_":
    #app.run(host='0.0.0.0', port=5001, debug=False)
    #app.debug = True
    if len(sys.argv) == 1:
        bind_addr = "0.0.0.0"
        SERVER = WSGIServer((bind_addr, PROXY_PORT),
APP)
        SERVER.serve_forever()
    else:
        print('running single request from command line arg')
        APP.testing = True
        with APP.test_client() as c:
            resp = c.get(sys.argv[1])
            print('Status: ' + resp.status)
            # print('Headers: ' + dumps(resp.headers))
            print(resp.data.decode('utf-8'))
  <!DOCT
  YPE
  html>
                    <html lang="en">
                      <head>
                        <meta charset="utf-8">
                        <meta http-equiv="x-ua-compatible"
                    content="ie=edge">
                        <meta name="viewport"
                    content="width=device-width, initial-scale=1">
                        <title>wttr-switcher</title>
                        <link rel="stylesheet"
                    href="assets/css/bootstrap.min.css">
                        <style>
```

```
        #weather-input {
            width:76px;
        }
    </style>
</head>
<body>
    <div class="container mt-3">
        <div class="d-inline-block">
            <img id="weather-img" class="img-fluid rounded" src="https://wttr.in/London_0pq_transparency=200_lang=en.png" alt="Current weather in London">
            <form id="weather-form">
                <div class="input-group mt-1">
                    <input id="weather-input" class="form-control" type="text" placeholder="Weather in..." aria-label="Weather in...">
                    <div class="input-group-append">
                        <button id="weather-btn" class="btn btn-primary" type="button">OK</button>
                    </div>
                </div>
            </form>
        </div>
    </div>
    <script defer src="assets/js/main.js"></script>
</body>
</html>
```

CHAPTER 6
SYSTEM TESTING

TESTING METHODOLOGIES

The following are the Testing Methodologies:

- **Unit Testing.**
- **Integration Testing.**
- **User Acceptance Testing.**
- **Output Testing.**
- **Validation Testing.**

Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.

During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

The following are the types of Integration Testing:

1) Top Down Integration

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main

program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

2. Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

Output Testing

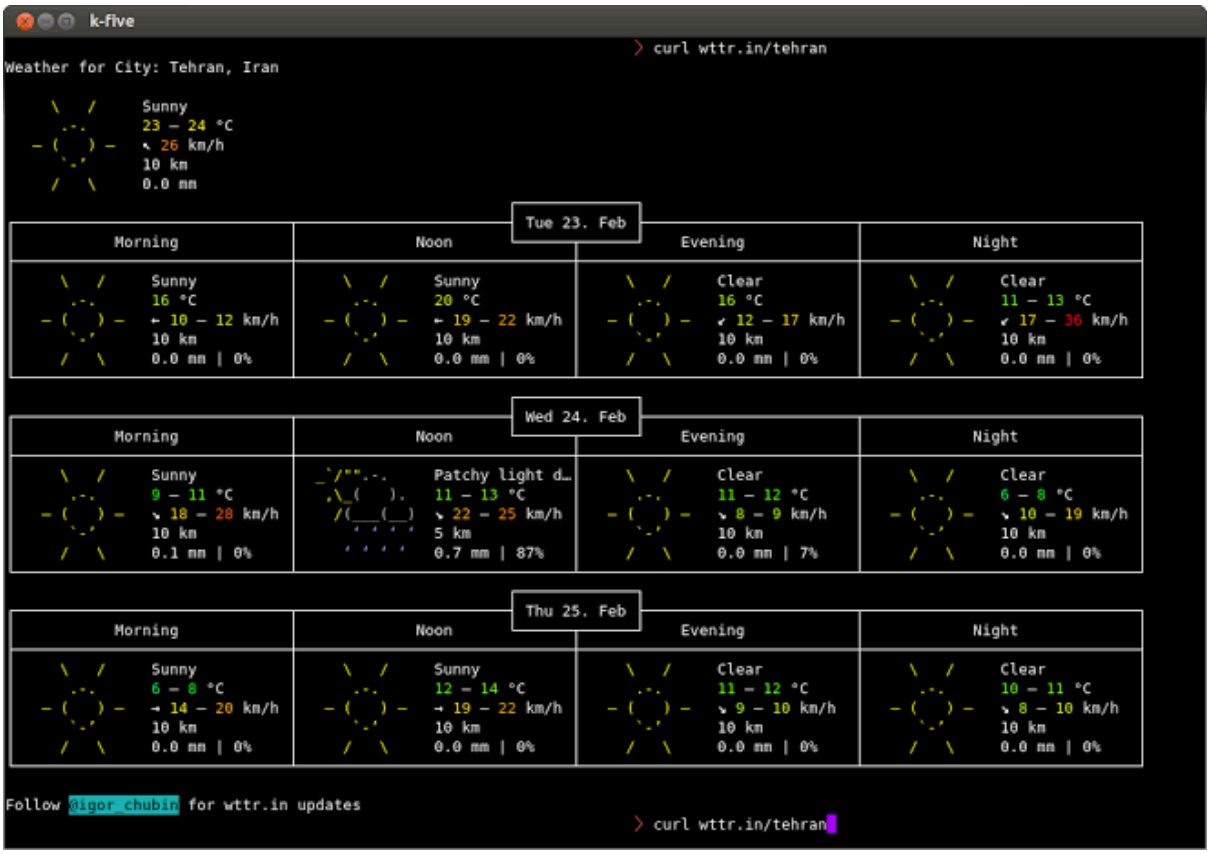
After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format

CHAPTER-7

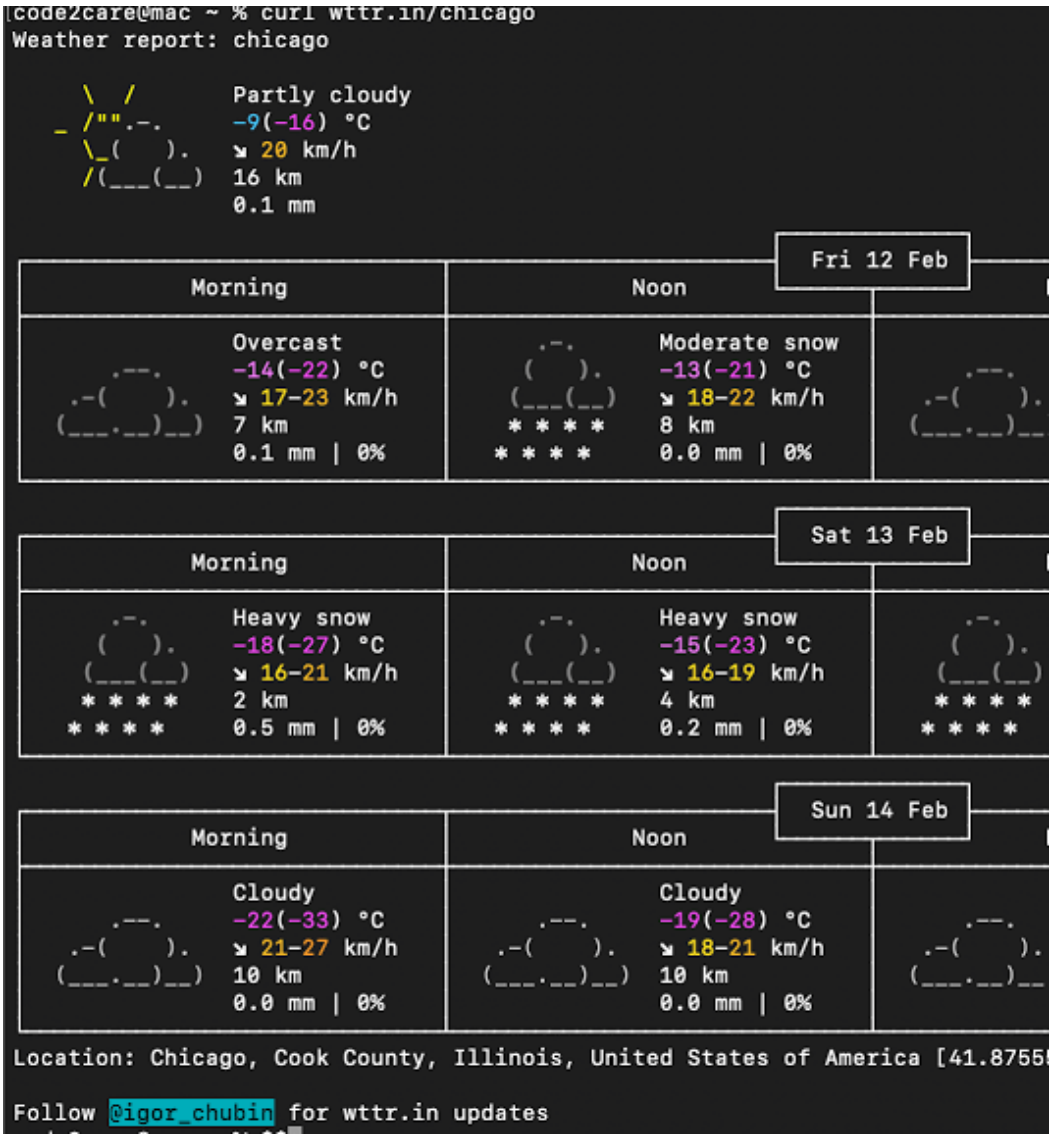
RESULTS

7.1 weather report of Iran

..



7.2 Weather report of Chicago



CHAPTER-8

CONCLUSION & FUTURE SCOPE

Weather forecasts still have their limitations despite the use of modern technology and improved techniques to predict the weather. Weather forecasting is complex and not always accurate ,especially for days in further future, because the weather can chaotic and unpredictable. If weather patterns are relatively stable, the persistence method of forecasting provides a relatively useful technique to predict the weather for the next day. Weather observation techniques improved and there have been technological advancements in predicting the weather in recent times. The accuracy of individual weather forecasts varies significantly.

REFERENCES

1. Beebe, R. G., "Forecasting Winter Precipitation for Atlanta, Ga." *Mon. We. Rev. Wash.*, 78:59–68 (1950).CrossRefGoogle Scholar

2. Besson, L., "Attempts at Methodical Forecasting of the Weather." Translation. *Mon. Wea. Rev. Wash.*, 32:311–313 (1904).CrossRefGoogle Scholar

3. Boyce, R. E., and others, *Tropical Cyclone Forecasting*. Chicago, University of Chicago, Dept. Meteor., 1950.Google Scholar

4. Brier, G. W., *Predicting the Occurrence of Winter Time Precipitation for Washington, D. C.* U. S. Weather Bureau, Washington, D. C., 1945.Google Scholar

5. Brier, G. W., "A Study of Quantitative Precipitation Forecasting in the Tva Basin." *U. S. W.a. Bur. Res. Pap*. No. 26 (1946).Google Scholar

6. Charney, J., and Eliassen, A., "A Numerical Method for Predicting the Perturbations of the Middle Latitude Westerlies." *Tellus*, Vol. 1, No. 2, pp. 38–54 (1949).CrossRefGoogle Scholar

7. Dines, W. H., "The Element of Chance Applied to Vari ous Meteorological Problems." *Quart. J. R. meteor. Soc*., 28:53–68 (1902).CrossRefGoogle Scholar

8. Gringorten, I. I., "A Study in Objective Forecasting." *Bull. Amer. meteor. Soc*., 30:10–15 (1949).Google Scholar

9. Hallenbeck, C., "Forecasting Precipitation in Percent ages of Probability." *Mon. Wea. Rev. Wash.*, 48:645–647 (1920).CrossRefGoogle Scholar

10. Hess, S. L., "Reply to Mr. Namias' Comments on the 'Prognostic Usefulness of the Correlation between Pressure and Temperature Changes at 3 km.'" *Bull. Amer. meteor. Soc*., 30:59–60 (1949).Google Scholar