###### A

Major Project On

**A MACHINE LEARNING APPROACH FOR RAINFALL ESTIMATIONINTEGRATINGHETEROGENEOUS DATA SOURCES**

(Submitted in partial fulfillment of the requirements for the award of Degree)

###### BACHELOR OF TECHNOLOGY

In

###### COMPUTER SCIENCE AND ENGINEERING

by

###### PASUPUNUTI VIVEK (217R5A0512)

Under the Guidance of

**CH.REKHA**

(Assistant Professor)

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**CMR TECHNICAL CAMPUS UGC AUTONOMOUS**

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2020-2024

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



#### CERTIFICATE

This is to certify that the project entitled “**A Machine Learning Approach For Rainfall Estimation Integrating Heterogeneous Data Sources ”** being submitted by **PASUPUNUTI VIVEK (217R5A0512)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-24.

The results embodied in this project have not been submitted to any other University or Institute for the award of any degree or diploma.

**CH.Rekha Dr. A. Raji Reddy**

(Assistant Professor) INTERNAL GUIDE

DIRECTOR

Dr. K. Srujan Raju

HOD

**EXTERNAL EXAMINER**

**Submitted for viva voice Examination held on**

**ACKNOWLEDGEMENT**

Apart from the efforts of Mine, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project.

I take this opportunity to express my profound gratitude and deep regard to my guide **Mrs.CH.Rekha,** Assistant Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her shall carry me a long way in the journey of life on which I am about to embrak.

I also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **G.Vinesh Shanker, Dr.J.Narasimharao, Ms.Shilpa, & Dr.K. Maheswari** for their cordial support, valuable information and guidance, which helped me in completing this task through various stages.

I also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

I am obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughoutthe course of this project. I also express my sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project. The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help

Finally, I would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. I sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

**PASUPUNUTI VIVEK (217R5A0512)**

#### ABSTRACT

Providing an accurate rainfall estimate at individual points is a challenging problem in order to mitigate risks derived from severe rainfall events, such as floods and landslides. Dense networks of sensors, named rain gauges (RGs), are typically used to obtain direct measurements of precipitation intensity in these points. These measurements are usually interpolated by using spatial interpolation methods for estimating the precipitation field over the entire area of interest. This project focuses on developing an innovative machine learning-based framework for accurate rainfall estimation by integrating diverse and heterogeneous data sources. Traditional methods often struggle to provide precise predictions due to limited data variety. Our approach leverages a combination of meteorological data, satellite imagery, and ground-based sensors to enhance the accuracy of rainfall predictions. We employ advanced machine learning algorithms to analyze and assimilate these diverse datasets, creating a robust model that adapts to changing environmental conditions. The integration of multiple data sources not only improves prediction accuracy but also enhances the reliability of rainfall estimates in areas with sparse ground-based monitoring.However, these methods are computationally expensive, and to improve the estimation of the variable of interest in unknown points, it is necessary to integrate further information. To overcome these issues, this work proposes a machine learning-based methodology that exploits a classifier based on ensemble methods for rainfall estimation and is able to integrate information from different remote sensing measurements. The proposed approach supplies an accurate estimate of the rainfall where RGs are not available, permits the integration of heterogeneous data sources exploiting both the high quantitative precision of RGs and the spatial pattern recognition ensured by radars and satellites, and is computationally less expensive than the interpolation methods. Experimental results, conducted on real data concerning an Italian region, Calabria, show a significant improvement in comparison with Kriging with external drift (KED), a well-recognized method in the field of rainfall estimation, both in terms of the probability of detection and mean-square error.

**LIST OF FIGURES**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIGURE NO** | | **FIGURE NAME** | **PAGE NO** |
| Figure 3.1 | | Project Architecture of A Machine Learning Approach for Rainfall Estimation Integrating Heterogeneous Data Sources. | 6 |
| Figure 3.2 | | Use Case Diagram for A Machine Learning Approach for Rainfall Estimation Integrating Heterogeneous Data Sources. | 7 |
| Figure 3.3 | | Class Diagram for Project Architecture of A Machine Learning Approach for Rainfall Estimation Integrating Heterogeneous Data Sources. | 8 |
| Figure 3.4 | | Sequence diagram for A Project Architecture of A Machine Learning Approach for Rainfall Estimation Integrating Heterogeneous Data Sources. | 9 |
|  |
|  |

**SCREENSHOTS**

|  |  |  |
| --- | --- | --- |
| **SCREENSHOT NO.** | **SCREENSHOT NAME** | **PAGE NO**. |
| Screenshot 5.1 | Registration Page | 27 |
| Screenshot 5.2 | Service Provider Page | 28 |
| Screenshot 5.3 | User Login Page | 29 |
| Screenshot 5.4 | Output Page | 30 |

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **ABSTRACT** | **i** |
| **LIST OF FIGURES**  **SCREENSHOTS** | **ii**  **iii** |
| **1. INTRODUCTION** | **1** |
| 1.1 PROJECT SCOPE | 1 |
| 1.2 PROJECT PURPOSE | 1 |
| 1.3 PROJECT FEATURES | 2 |
| **2. SYSTEM ANALYSIS** | **3** |
| 2.1 PROBLEM DEFINITION | 3 |
| 2.2 EXISTING SYSTEM | 3 |
| 2.2.1 LIMITATIONS OF THE EXISTING SYSTEM | 4 |
| 2.3 PROPOSED SYSTEM | 4 |
| 2.3.1 ADVANTAGES OF PROPOSED SYSTEM | 4 |
| 2.4 HARDWARE & SOFTWARE REQUIREMENTS | 5 |
| 2.4.1 HARDWARE REQUIREMENTS | 5 |
| 2.4.2 SOFTWARE REQUIREMENTS | 5 |
| **3. ARCHITECTURE** | **6** |
| 3.1 PROJECT ARCHITECTURE | 6 |
| 3.2 USE CASE DIAGRAM | 7 |
| 3.3 CLASS DIAGRAM | 8 |
| 3.4 SEQUENCE DIAGRAM | 9  **10** |
| **4. IMPLEMENTATION**  **5. SCREENSHOTS**  5.1 REGISTARTION  5.2 SERVICE PROVIDER PAGE  5.3 USER LOGIN PAGE  5.4 OUTPUT PAGE  **6.TESTING**  6.1 INTODUCTION TO TESTING  6.2 TYPES OF TESTING    6.2.1 UNIT TESTING  6.2.2 INTEGRATION TESTING  6.2.3 FUNCTIONAL TESTING  6.2.4 SYSTEM TESTING  6.2.5 WHITE BOX TESTING  6.2.6 BLACK BOX TESTING  6.2.7 ACCEPTANCE TESTING  6.3 TEST CASES  **7.CONCLUSION**  **8.BIBILOGRAPHY**  8.1 REFERENCES  8.2 GITHUB LINK | **27**  27  28  29  30  **31**  31  31  31  32  32  32  32  32  33  33  34  **36**  37  37 |

# INTRODUCTION

#### INTRODUCTION

##### 1.1 PROJECT SCOPE

This project aims to develop a comprehensive machine learning-based approach for rainfall estimation by integrating heterogeneous data sources. The scope encompasses a detailed investigation into various types of data, including but not limited to meteorological satellite imagery, ground-based weather station data, and remote sensing information. The project will involve a thorough review and selection of appropriate machine learning algorithms suitable for handling diverse data modalities.

The research will focus on creating a robust framework that can effectively fuse information from these disparate sources to enhance the accuracy and reliability of rainfall predictions. The algorithmic models will be tailored to account for the inherent challenges posed by the heterogeneity in data, such as varying spatial and temporal resolutions. Special attention will be given to addressing potential biases and uncertainties associated with each data type.

##### 1.2 PROJECT PURPOSE

The purpose of the "Machine Learning Approach for Rainfall Estimation Integrating Heterogeneous Data Sources" project is likely to improve accuracy in rainfall prediction by leveraging diverse data sets. This could involve integrating information from various sources such as satellite imagery, weather stations, and other relevant data to enhance the effectiveness of rainfall estimation models using machine learning techniques. The goal is to develop a more robust and precise system for forecasting rainfall, which has implications for sectors like agriculture, water resource management, and disaster preparedness.

##### 1.3 PROJECT FEATURES

Features of "Machine Learning Approach for Rainfall Estimation Integrating Heterogeneous Data Sources" project is a comprehensive initiative aimed at advancing rainfall prediction through the integration of diverse data sets. Leveraging cutting-edge machine learning techniques, the project focuses on amalgamating heterogeneous data sources to enhance the accuracy and reliability of rainfall estimation.

The key features of this project include the utilization of various data types, such as satellite imagery, weather station data, and atmospheric pressure readings. By integrating these disparate sources, the model aims to capture a more holistic understanding of the factors influencing rainfall patterns, thereby improving the overall precision of predictions. Additionally, the project emphasizes real-time data processing to ensure up-to-the-minute accuracy in rainfall estimates.

One noteworthy aspect is the incorporation of advanced statistical models to analyze historical rainfall patterns and identify trends. This historical perspective enhances the project's ability to predict future rainfall events by learning from past occurrences and adapting to changing environmental conditions. Moreover, the machine learning algorithms employed in the project undergo continuous refinement through iterative training processes, ensuring adaptability to evolving climate dynamics.

* 1. **SYSTEM ANALYSIS**

#### SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The proposed project aims to develop a machine learning-based system for rainfall estimation by integrating heterogeneous data sources. The system analysis involves a comprehensive examination of diverse data inputs, including meteorological data, satellite imagery, and ground-based measurements. Through this integrated approach, the model seeks to enhance the accuracy of rainfall predictions. The system will undergo a thorough evaluation to ensure robustness and reliability, considering factors such as data quality, model performance, and real-time applicability. The project's system analysis will play a crucial role in identifying optimal algorithms, refining feature selection, and establishing a framework for seamless integration of disparate data types, ultimately contributing to more precise and timely rainfall estimates.

##### PROBLEM DEFINITION

The project aims to develop a machine learning model that combines diverse data sources to accurately estimate the rainfall, to addressing for the challenge of integrating heterogeneous data for more precise rainfall predictions.

##### EXISTING SYSTEM

The existing is based on the ensemble paradigm include the work in which, similar to our work, employs a probabilistic ensemble and merges two sources of data (i.e., rain gauges and radar) even if the aim of this work is to develop a run-off analysis. Afterward, a blending technique is applied to the results of the runoff hydrologic models to determine a single runoff hydrograph. Experimental results show that the hydrologic models are accurate and can help to make more effective decisions in the flood warning define a technique for deriving a probabilistic spatial analysis of daily precipitation from rain gauges. The final model represents an ensemble of possible fields, conditional on the observations, which can be explained as a Bayesian predictive distribution measuring the uncertainty due to the data sampling from the station network. An evaluation of a real case study, located in the European Alps, proves the capability of the approach in providing accurate predictions for a hydrological partitioning of the region.

##### LIMITATIONS OF EXISTING SYSTEM

Following are the disadvantages of existing system:

* + - * Hierarchical probabilistic ensemble classifier (HPEC) rainfall prediction.
      * Artificial neural networks (ANNs) prediction is not accurate.

##### 2.3 PROPOSED SYSTEM

The proposed system leverages a machine learning approach to enhance rainfall estimation by integrating heterogeneous data sources. By combining meteorological satellite data, ground-based weather station measurements, and historical precipitation records, our model aims to provide a more comprehensive and accurate rainfall prediction. The machine learning algorithms employed, such as neural networks or ensemble methods, will be trained on this diverse dataset to learn complex patterns and relationships, enabling the system to adapt to varying geographical and climatic conditions. This integrative approach not only improves the precision of rainfall estimation but also enhances the robustness and reliability of the system, making it a valuable tool for effective water resource management and disaster preparedness.

##### ADVANTAGES OF THE PROPOSED SYSTEM

* + - * The proposed system is more effective due to presence of many ml classifiers .
      * The proposed system implemented with an accurate prediction for the correspond dataset

#### 2.4 HARDWARE & SOFTWARE REQUIREMENTS

##### 2.4.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

System Processor : Pentium –IV

Key Board : Standard Windows Keyboard

Monitor : SVGA

Mouse : Two or Three Button Mouse

Hard Disk : Minimum of 8GB and above

Ram : Minimum of 8GB and above

##### SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system.

Operating Systems : Window 8 or above

Coding Languages : Python

Front-End : Css,Html,Javascript.

Back-End : Django-ORM

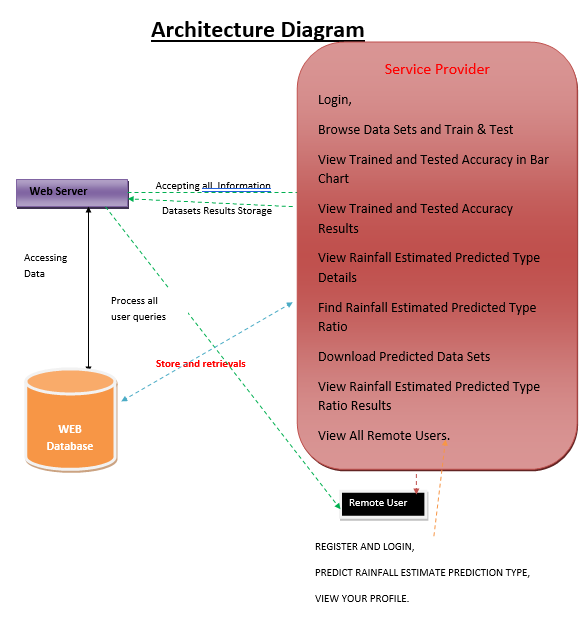
Data Base : MySQL (WAMP Server).

## ARCHITECTURE

### ARCHITECTURE

##### 3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.



**Figure 3.1:** Project Architecture of A Machine Learning Approach for Rainfall Estimation Integrating Heterogeneous Data Sources.

##### 3.2 USE CASE DIAGRAM

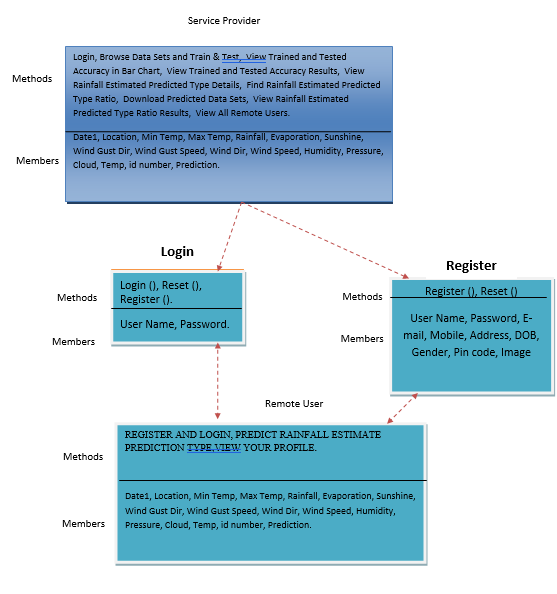
In the use case diagram, we have basically one actor who is the user in trained model. A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures. 

**Figure 3.2:** Use Case Diagram for A Robust Approach For Effective Spam Detection Using Supervised Learning Techniques

CMRTC 7

##### 3.3 CLASS DIAGRAM

Class diagram is a type of static structure diagram that describes the structureof a system by showing the system’s classes, their attributes, operations(or methods), and the relationships among objects.

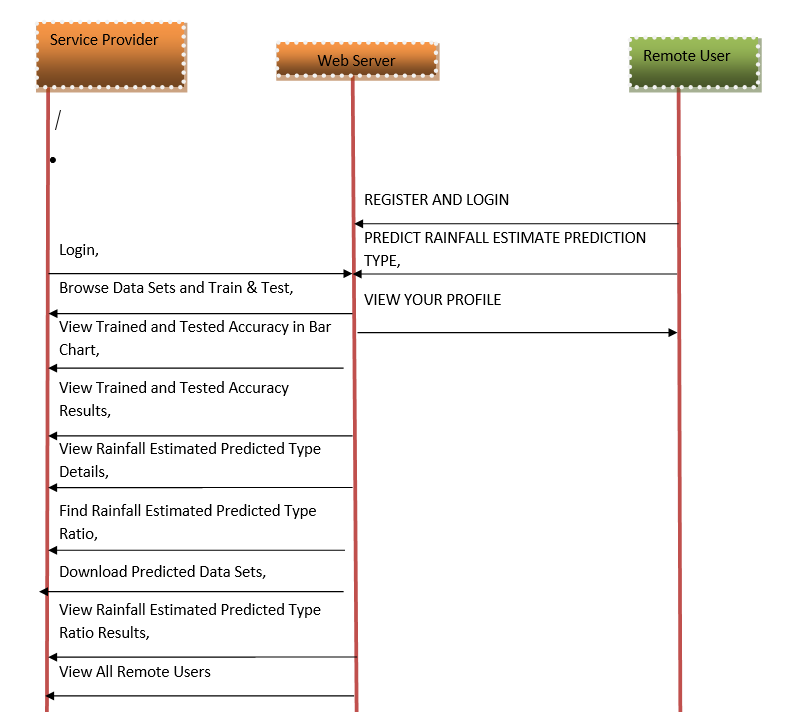


**Figure 3.3**: Class Diagram for A Machine Learning Approach for Rainfall Estimation Integrating Heterogeneous Data Sources

CMRTC 8

##### 3.4 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects 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 use case realizations in the logical view of the system under development.



**Figure 3.4:** Sequence Diagram for A Machine Learning Approach for Rainfall Estimation Integrating Heterogeneous Data Sources

CMRTC

1

9

## IMPLEMENTATION

##### 4.1 SAMPLE CODE

import functools

import os

import sys

import os.path

from io import StringIO

from pip.\_vendor.pygments.formatter import Formatter

from pip.\_vendor.pygments.token import Token, Text, STANDARD\_TYPES

from pip.\_vendor.pygments.util import get\_bool\_opt, get\_int\_opt, get\_list\_opt

try:

import ctags

except ImportError:

ctags = None

\_\_all\_\_ = ['HtmlFormatter']

\_escape\_html\_table = {

ord('&'): '&amp;',

ord('<'): '&lt;',

ord('>'): '&gt;',

ord('"'): '&quot;',

ord("'"): '&#39;',

}

def escape\_html(text, table=\_escape\_html\_table):

"""Escape &, <, > as well as single and double quotes for HTML."""

return text.translate(table)

def webify(color):

if color.startswith('calc') or color.startswith('var'):

return color

else:

return '#' + color

10

def \_get\_ttype\_class(ttype):

fname = STANDARD\_TYPES.get(ttype)

if fname:

return fname

aname = ''

while fname is None:

aname = '-' + ttype[-1] + aname

ttype = ttype.parent

fname = STANDARD\_TYPES.get(ttype)

return fname + aname

CSSFILE\_TEMPLATE = '''\

/\*

generated by Pygments <https://pygments.org/>

Copyright 2006-2022 by the Pygments team.

Licensed under the BSD license, see LICENSE for details.

\*/

%(styledefs)s

'''

DOC\_HEADER = '''\

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"

"http://www.w3.org/TR/html4/strict.dtd">

<!--

generated by Pygments <https://pygments.org/>

Copyright 2006-2022 by the Pygments team.

Licensed under the BSD license, see LICENSE for details.

-->

<html>

<head>

<title>%(title)s</title>

<meta http-equiv="content-type" content="text/html; charset=%(encoding)s">

<style type="text/css">

''' + CSSFILE\_TEMPLATE + '''

</style>

</head>

<body>

<h2>%(title)s</h2>

'''

11

<html>

<head>

<title>%(title)s</title>

<meta http-equiv="content-type" content="text/html; charset=%(encoding)s">

<link rel="stylesheet" href="%(cssfile)s" type="text/css">

</head>

<body>

<h2>%(title)s</h2>

'''

DOC\_FOOTER = '''\

</body>

</html>

'''

class HtmlFormatter(Formatter):

name = 'HTML'

aliases = ['html']

filenames = ['\*.html', '\*.htm']

def \_\_init\_\_(self, \*\*options):

Formatter.\_\_init\_\_(self, \*\*options)

self.title = self.\_decodeifneeded(self.title)

self.nowrap = get\_bool\_opt(options, 'nowrap', False)

self.noclasses = get\_bool\_opt(options, 'noclasses', False)

self.classprefix = options.get('classprefix', '')

self.cssclass = self.\_decodeifneeded(options.get('cssclass', 'highlight'))

self.cssstyles = self.\_decodeifneeded(options.get('cssstyles', ''))

self.prestyles = self.\_decodeifneeded(options.get('prestyles', ''))

self.cssfile = self.\_decodeifneeded(options.get('cssfile', ''))

self.noclobber\_cssfile = get\_bool\_opt(options, 'noclobber\_cssfile', False)

self.tagsfile = self.\_decodeifneeded(options.get('tagsfile', ''))

self.tagurlformat = self.\_decodeifneeded(options.get('tagurlformat', ''))

self.filename = self.\_decodeifneeded(options.get('filename', ''))

self.wrapcode = get\_bool\_opt(options, 'wrapcode', False)

self.span\_element\_openers = {}

self.debug\_token\_types = get\_bool\_opt(options, 'debug\_token\_types', False)

if self.tagsfile:

12

if not ctags:

raise RuntimeError('The "ctags" package must to be installed '

'to be able to use the "tagsfile" feature.')

self.\_ctags = ctags.CTags(self.tagsfile)

linenos = options.get('linenos', False)

if linenos == 'inline':

self.linenos = 2

elif linenos:

# compatibility with <= 0.7

self.linenos = 1

else:

self.linenos = 0

self.linenostart = abs(get\_int\_opt(options, 'linenostart', 1))

self.linenostep = abs(get\_int\_opt(options, 'linenostep', 1))

self.linenospecial = abs(get\_int\_opt(options, 'linenospecial', 0))

self.nobackground = get\_bool\_opt(options, 'nobackground', False)

self.lineseparator = options.get('lineseparator', '\n')

self.lineanchors = options.get('lineanchors', '')

self.linespans = options.get('linespans', '')

self.anchorlinenos = get\_bool\_opt(options, 'anchorlinenos', False)

self.hl\_lines = set()

for lineno in get\_list\_opt(options, 'hl\_lines', []):

try:

self.hl\_lines.add(int(lineno))

except ValueError:

pass

self.\_create\_stylesheet()

def \_get\_css\_class(self, ttype):

"""Return the css class of this token type prefixed with

the classprefix option."""

ttypeclass = \_get\_ttype\_class(ttype)

if ttypeclass:

return self.classprefix + ttypeclass

return ''

def \_get\_css\_classes(self, ttype):

"""Return the CSS classes of this token type prefixed with the classprefix option."""

cls = self.\_get\_css\_class(ttype)

13

while ttype not in STANDARD\_TYPES:

ttype = ttype.parent

cls = self.\_get\_css\_class(ttype) + ' ' + cls

return cls or ''

def \_get\_css\_inline\_styles(self, ttype):

"""Return the inline CSS styles for this token type."""

cclass = self.ttype2class.get(ttype)

while cclass is None:

ttype = ttype.parent

cclass = self.ttype2class.get(ttype)

return cclass or ''

def \_create\_stylesheet(self):

t2c = self.ttype2class = {Token: ''}

c2s = self.class2style = {}

for ttype, ndef in self.style:

name = self.\_get\_css\_class(ttype)

style = ''

if ndef['color']:

style += 'color: %s; ' % webify(ndef['color'])

if ndef['bold']:

style += 'font-weight: bold; '

if ndef['italic']:

style += 'font-style: italic; '

if ndef['underline']:

style += 'text-decoration: underline; '

if ndef['bgcolor']:

style += 'background-color: %s; ' % webify(ndef['bgcolor'])

if ndef['border']:

style += 'border: 1px solid %s; ' % webify(ndef['border'])

if style:

t2c[ttype] = name

# save len(ttype) to enable ordering the styles by

# hierarchy (necessary for CSS cascading rules!)

c2s[name] = (style[:-2], ttype, len(ttype))

def get\_style\_defs(self, arg=None):

"""

Return CSS style definitions for the classes produced by the current

highlighting style. ``arg`` can be a string or list of selectors to

14

insert before the token type classes.

"""

style\_lines = []

style\_lines.extend(self.get\_linenos\_style\_defs())

style\_lines.extend(self.get\_background\_style\_defs(arg))

style\_lines.extend(self.get\_token\_style\_defs(arg))

return '\n'.join(style\_lines)

def get\_token\_style\_defs(self, arg=None):

prefix = self.get\_css\_prefix(arg)

styles = [

(level, ttype, cls, style)

for cls, (style, ttype, level) in self.class2style.items()

if cls and style

]

styles.sort()

lines = [

'%s { %s } /\* %s \*/' % (prefix(cls), style, repr(ttype)[6:])

for (level, ttype, cls, style) in styles

]

return lines

def get\_background\_style\_defs(self, arg=None):

prefix = self.get\_css\_prefix(arg)

bg\_color = self.style.background\_color

hl\_color = self.style.highlight\_color

lines = []

if arg and not self.nobackground and bg\_color is not None:

text\_style = ''

if Text in self.ttype2class:

text\_style = ' ' + self.class2style[self.ttype2class[Text]][0]

lines.insert(

0, '%s{ background: %s;%s }' % (

prefix(''), bg\_color, text\_style

15

)

)

if hl\_color is not None:

lines.insert(

0, '%s { background-color: %s }' % (prefix('hll'), hl\_color)

)

return lines

def get\_linenos\_style\_defs(self):

lines = [

'pre { %s }' % self.\_pre\_style,

'td.linenos .normal { %s }' % self.\_linenos\_style,

'span.linenos { %s }' % self.\_linenos\_style,

'td.linenos .special { %s }' % self.\_linenos\_special\_style,

'span.linenos.special { %s }' % self.\_linenos\_special\_style,

]

return lines

def get\_css\_prefix(self, arg):

if arg is None:

arg = ('cssclass' in self.options and '.'+self.cssclass or '')

if isinstance(arg, str):

args = [arg]

else:

args = list(arg)

def prefix(cls):

if cls:

cls = '.' + cls

tmp = []

for arg in args:

tmp.append((arg and arg + ' ' or '') + cls)

return ', '.join(tmp)

return prefix

@property

def \_pre\_style(self):

return 'line-height: 125%;'

16

@property

def \_linenos\_style(self):

return 'color: %s; background-color: %s; padding-left: 5px; padding-right: 5px;' % (

self.style.line\_number\_color,

self.style.line\_number\_background\_color

)

@property

def \_linenos\_special\_style(self):

return 'color: %s; background-color: %s; padding-left: 5px; padding-right: 5px;' % (

self.style.line\_number\_special\_color,

self.style.line\_number\_special\_background\_color

)

def \_decodeifneeded(self, value):

if isinstance(value, bytes):

if self.encoding:

return value.decode(self.encoding)

return value.decode()

return value

def \_wrap\_full(self, inner, outfile):

if self.cssfile:

if os.path.isabs(self.cssfile):

# it's an absolute filename

cssfilename = self.cssfile

else:

try:

filename = outfile.name

if not filename or filename[0] == '<':

# pseudo files, e.g. name == '<fdopen>'

raise AttributeError

cssfilename = os.path.join(os.path.dirname(filename),

self.cssfile)

except AttributeError:

print('Note: Cannot determine output file name, '

'using current directory as base for the CSS file name',

file=sys.stderr)

cssfilename = self.cssfile

# write CSS file only if noclobber\_cssfile isn't given as an option.

try:

17

if not os.path.exists(cssfilename) or not self.noclobber\_cssfile:

with open(cssfilename, "w") as cf:

cf.write(CSSFILE\_TEMPLATE %

{'styledefs': self.get\_style\_defs('body')})

except OSError as err:

err.strerror = 'Error writing CSS file: ' + err.strerror

raise

yield 0, (DOC\_HEADER\_EXTERNALCSS %

dict(title=self.title,

cssfile=self.cssfile,

encoding=self.encoding))

else:

yield 0, (DOC\_HEADER %

dict(title=self.title,

styledefs=self.get\_style\_defs('body'),

encoding=self.encoding))

yield from inner

yield 0, DOC\_FOOTER

def \_wrap\_tablelinenos(self, inner):

dummyoutfile = StringIO()

lncount = 0

for t, line in inner:

if t:

lncount += 1

dummyoutfile.write(line)

fl = self.linenostart

mw = len(str(lncount + fl - 1))

sp = self.linenospecial

st = self.linenostep

anchor\_name = self.lineanchors or self.linespans

aln = self.anchorlinenos

nocls = self.noclasses

lines = []

for i in range(fl, fl+lncount):

print\_line = i % st == 0

18

special\_line = sp and i % sp == 0

if print\_line:

line = '%\*d' % (mw, i)

if aln:

line = '<a href="#%s-%d">%s</a>' % (anchor\_name, i, line)

else:

line = ' ' \* mw

if nocls:

if special\_line:

style = ' style="%s"' % self.\_linenos\_special\_style

else:

style = ' style="%s"' % self.\_linenos\_style

else:

if special\_line:

style = ' class="special"'

else:

style = ' class="normal"'

if style:

line = '<span%s>%s</span>' % (style, line)

lines.append(line)

ls = '\n'.join(lines)

# If a filename was specified, we can't put it into the code table as it

# would misalign the line numbers. Hence we emit a separate row for it.

filename\_tr = ""

if self.filename:

filename\_tr = (

'<tr><th colspan="2" class="filename">'

'<span class="filename">' + self.filename + '</span>'

'</th></tr>')

# in case you wonder about the seemingly redundant <div> here: since the

# content in the other cell also is wrapped in a div, some browsers in

# some configurations seem to mess up the formatting...

yield 0, (f'<table class="{self.cssclass}table">' + filename\_tr +

'<tr><td class="linenos"><div class="linenodiv"><pre>' +

19

ls + '</pre></div></td><td class="code">')

yield 0, '<div>'

yield 0, dummyoutfile.getvalue()

yield 0, '</div>'

yield 0, '</td></tr></table>'

def \_wrap\_inlinelinenos(self, inner):

# need a list of lines since we need the width of a single number :(

inner\_lines = list(inner)

sp = self.linenospecial

st = self.linenostep

num = self.linenostart

mw = len(str(len(inner\_lines) + num - 1))

anchor\_name = self.lineanchors or self.linespans

aln = self.anchorlinenos

nocls = self.noclasses

for \_, inner\_line in inner\_lines:

print\_line = num % st == 0

special\_line = sp and num % sp == 0

if print\_line:

line = '%\*d' % (mw, num)

else:

line = ' ' \* mw

if nocls:

if special\_line:

style = ' style="%s"' % self.\_linenos\_special\_style

else:

style = ' style="%s"' % self.\_linenos\_style

else:

if special\_line:

style = ' class="linenos special"'

else:

style = ' class="linenos"'

if style:

linenos = '<span%s>%s</span>' % (style, line)

else:

20

linenos = line

if aln:

yield 1, ('<a href="#%s-%d">%s</a>' % (anchor\_name, num, linenos) +

inner\_line)

else:

yield 1, linenos + inner\_line

num += 1

def \_wrap\_lineanchors(self, inner):

s = self.lineanchors

# subtract 1 since we have to increment i \*before\* yielding

i = self.linenostart - 1

for t, line in inner:

if t:

i += 1

href = "" if self.linenos else ' href="#%s-%d"' % (s, i)

yield 1, '<a id="%s-%d" name="%s-%d"%s></a>' % (s, i, s, i, href) + line

else:

yield 0, line

def \_wrap\_linespans(self, inner):

s = self.linespans

i = self.linenostart - 1

for t, line in inner:

if t:

i += 1

yield 1, '<span id="%s-%d">%s</span>' % (s, i, line)

else:

yield 0, line

def \_wrap\_div(self, inner):

style = []

if (self.noclasses and not self.nobackground and

self.style.background\_color is not None):

style.append('background: %s' % (self.style.background\_color,))

if self.cssstyles:

style.append(self.cssstyles)

style = '; '.join(style)

yield 0, ('<div' + (self.cssclass and ' class="%s"' % self.cssclass) +

21

(style and (' style="%s"' % style)) + '>')

yield from inner

yield 0, '</div>\n'

def \_wrap\_pre(self, inner):

style = []

if self.prestyles:

style.append(self.prestyles)

if self.noclasses:

style.append(self.\_pre\_style)

style = '; '.join(style)

if self.filename and self.linenos != 1:

yield 0, ('<span class="filename">' + self.filename + '</span>')

# the empty span here is to keep leading empty lines from being

# ignored by HTML parsers

yield 0, ('<pre' + (style and ' style="%s"' % style) + '><span></span>')

yield from inner

yield 0, '</pre>'

def \_wrap\_code(self, inner):

yield 0, '<code>'

yield from inner

yield 0, '</code>'

@functools.lru\_cache(maxsize=100)

def \_translate\_parts(self, value):

"""HTML-escape a value and split it by newlines."""

return value.translate(\_escape\_html\_table).split('\n')

def \_format\_lines(self, tokensource):

"""

Just format the tokens, without any wrapping tags.

Yield individual lines.

"""

nocls = self.noclasses

lsep = self.lineseparator

tagsfile = self.tagsfile

lspan = ''

22

line = []

for ttype, value in tokensource:

try:

cspan = self.span\_element\_openers[ttype]

except KeyError:

title = ' title="%s"' % '.'.join(ttype) if self.debug\_token\_types else ''

if nocls:

css\_style = self.\_get\_css\_inline\_styles(ttype)

if css\_style:

css\_style = self.class2style[css\_style][0]

cspan = '<span style="%s"%s>' % (css\_style, title)

else:

cspan = ''

else:

css\_class = self.\_get\_css\_classes(ttype)

if css\_class:

cspan = '<span class="%s"%s>' % (css\_class, title)

else:

cspan = ''

self.span\_element\_openers[ttype] = cspan

parts = self.\_translate\_parts(value)

if tagsfile and ttype in Token.Name:

filename, linenumber = self.\_lookup\_ctag(value)

if linenumber:

base, filename = os.path.split(filename)

if base:

base += '/'

filename, extension = os.path.splitext(filename)

url = self.tagurlformat % {'path': base, 'fname': filename,

'fext': extension}

parts[0] = "<a href=\"%s#%s-%d\">%s" % \

(url, self.lineanchors, linenumber, parts[0])

parts[-1] = parts[-1] + "</a>"

# for all but the last line

for part in parts[:-1]:

if line:

if lspan != cspan:

line.extend(((lspan and '</span>'), cspan, part,

(cspan and '</span>'), lsep))

23

else: # both are the same

line.extend((part, (lspan and '</span>'), lsep))

yield 1, ''.join(line)

line = []

elif part:

yield 1, ''.join((cspan, part, (cspan and '</span>'), lsep))

else:

yield 1, lsep

# for the last line

if line and parts[-1]:

if lspan != cspan:

line.extend(((lspan and '</span>'), cspan, parts[-1]))

lspan = cspan

else:

line.append(parts[-1])

elif parts[-1]:

line = [cspan, parts[-1]]

lspan = cspan

# else we neither have to open a new span nor set lspan

if line:

line.extend(((lspan and '</span>'), lsep))

yield 1, ''.join(line)

def \_lookup\_ctag(self, token):

entry = ctags.TagEntry()

if self.\_ctags.find(entry, token.encode(), 0):

return entry['file'], entry['lineNumber']

else:

return None, None

def \_highlight\_lines(self, tokensource):

"""

Highlighted the lines specified in the `hl\_lines` option by

post-processing the token stream coming from `\_format\_lines`.

"""

hls = self.hl\_lines

for i, (t, value) in enumerate(tokensource):

if t != 1:

yield t, value

24

if i + 1 in hls: # i + 1 because Python indexes start at 0

if self.noclasses:

style = ''

if self.style.highlight\_color is not None:

style = (' style="background-color: %s"' %

(self.style.highlight\_color,))

yield 1, '<span%s>%s</span>' % (style, value)

else:

yield 1, '<span class="hll">%s</span>' % value

else:

yield 1, value

def wrap(self, source):

"""

Wrap the ``source``, which is a generator yielding

individual lines, in custom generators. See docstring

for `format`. Can be overridden.

"""

output = source

if self.wrapcode:

output = self.\_wrap\_code(output)

output = self.\_wrap\_pre(output)

return output

def format\_unencoded(self, tokensource, outfile):

source = self.\_format\_lines(tokensource)

if not self.nowrap and self.linenos == 2:

source = self.\_wrap\_inlinelinenos(source)

if self.hl\_lines:

source = self.\_highlight\_lines(source)

if not self.nowrap:

if self.lineanchors:

source = self.\_wrap\_lineanchors(source)

if self.linespans:

25

source = self.\_wrap\_linespans(source)

source = self.wrap(source)

if self.linenos == 1:

source = self.\_wrap\_tablelinenos(source)

source = self.\_wrap\_div(source)

if self.full:

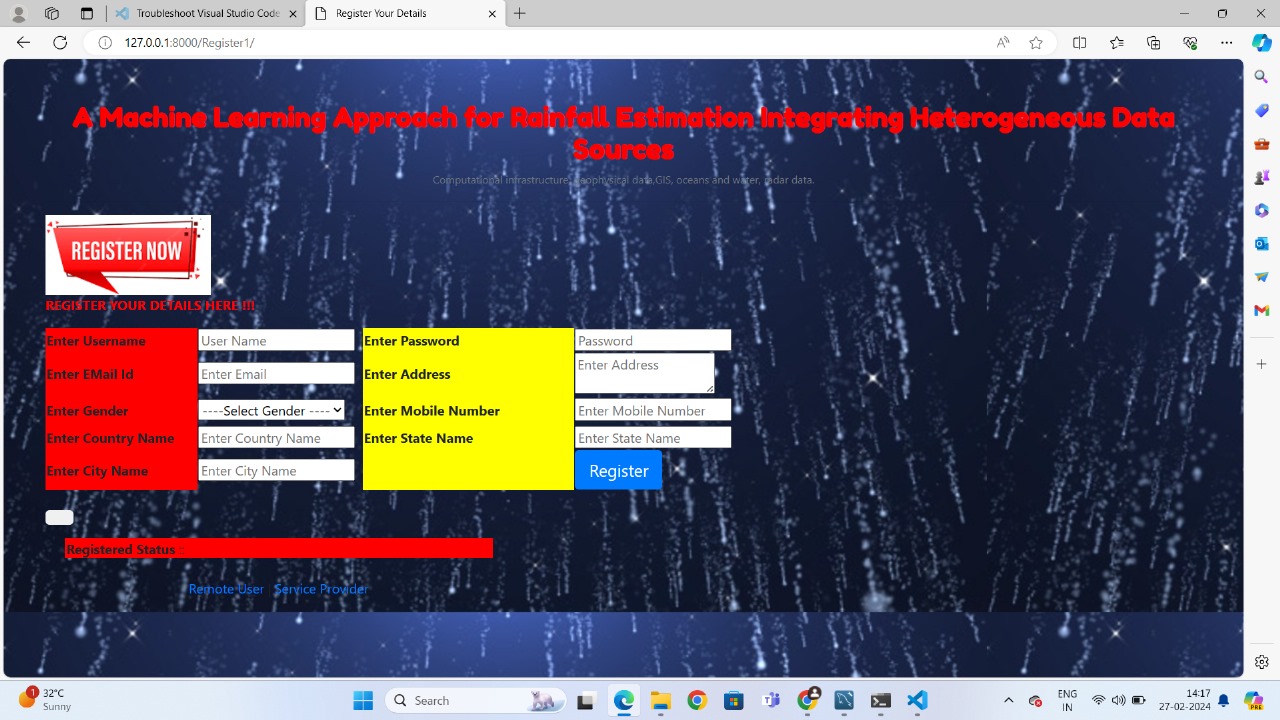
source = self.\_wrap\_full(source, outfile)

for t, piece in source:

outfile.write(piece)

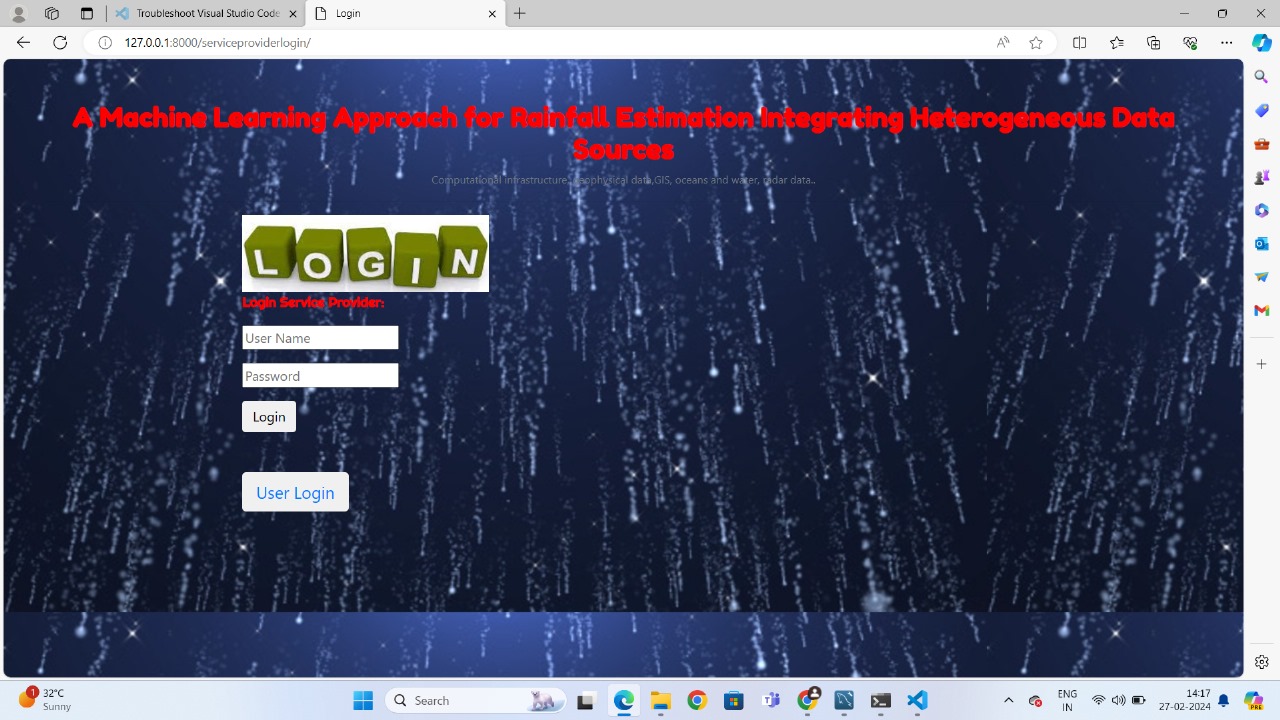
26

## SCREENSHOTS



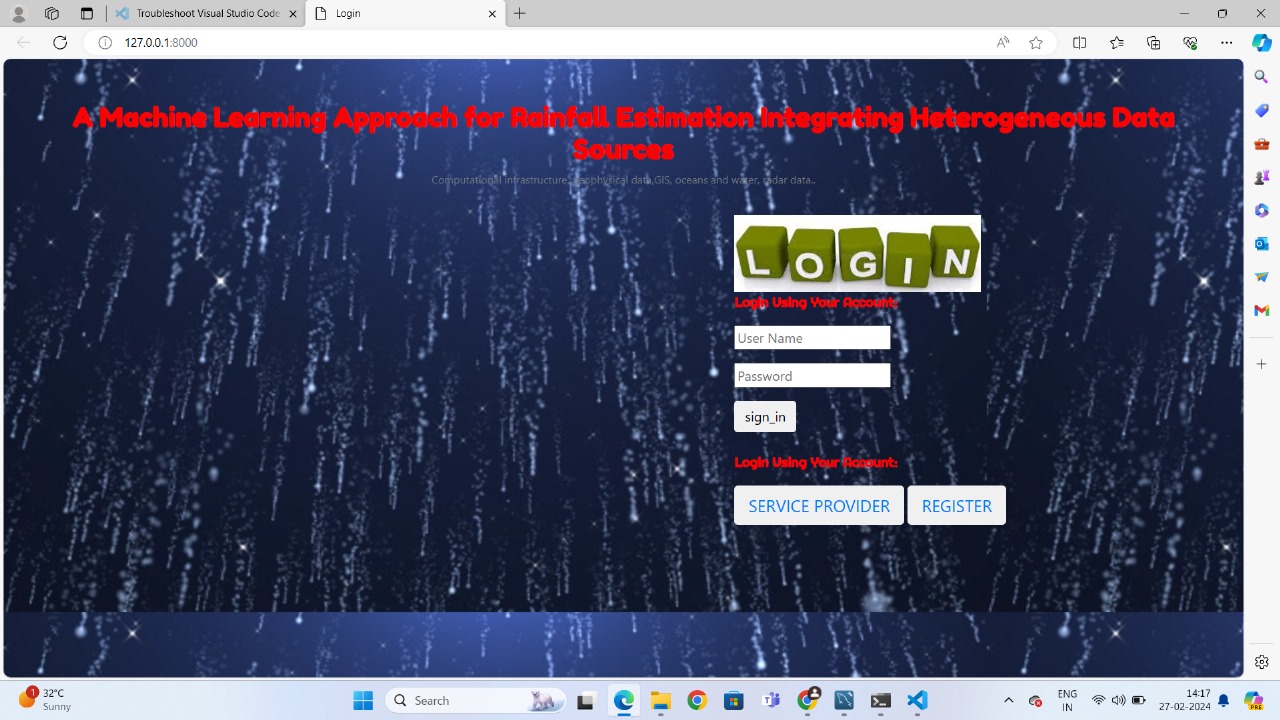
**SCREENSHOT 5.1:** Registration Page

27



**SCREENSHOT 5.2:** Service Provider Login Page

28



**SCREENSHOT 5.3:** User Login Page

29



**SCREENSHOT 5.3:** Output Page

30

## TESTING

### 6. TESTING

##### 6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

##### 6.2 TYPES OF TESTING

**6.2.1 UNIT TESTING :**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .It is done after the completion of an individual unit before integration. This is a structural testing that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

31

##### INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

##### 6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

**Valid Input :** identified classes of valid input must be accepted. **Invalid :** identified classes of invalid input must Input be rejected. **Functions :** identified functions must be exercised.

**Output :** identified classes of application outputs must be exercised.

##### 6.2.4 SYSTEM TESTING

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points

##### 6.2.5 WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose.

32

##### 6.2.6 BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works*.*

##### 6.2.7 ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

##### TEST STRATEGY AND APPROACH

Field testing will be performed manually and functional tests will be written in

detail.

Test Objectives

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

Features To Be Tested

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

33

**6.3 TEST CASES**

|  |  |
| --- | --- |
| Use case ID | YOUTUBE VIDEO PROMOTION BY CROSS- NETWORK |
| Use case Name | Home button | |
| Description | Display home page of registration | |
| Primary actor | User | |
| Precondition | User must open registration | |
| Post condition | Display the Home Page of an registration | |
| Frequency of Use case | Many times | |
| Use case Diagrams | N/A | |
| Attachments | N/A | |

|  |  |
| --- | --- |
| Use case ID | YOUTUBE VIDEO PROMOTION BY CROSS- NETWORK |
| Use case Name | Registration |
| Description | It display the registration form |
| Primary actor | User |
| Precondition | User Must have Email ID |

34

|  |  |
| --- | --- |
| Use case ID | YOUTUBE VIDEO PROMOTION BY  CROSS-NETWORK |
| Use case Name | Login Form |
| Description | Display Login form to the User |
| Primary actor | User |
| Precondition | User must have username &password |
| Post condition | Display the Home Page |
| Alternative use case | Forgot password |

# 7.CONCLUSION

#### 7. CONCLUSION

In conclusion, our proposed ML-based approach for spatial rainfall field estimation, leveraging heterogeneous data sources like RGs, radars, and satellites, proves to be highly effective. The methodology, incorporating spatial pattern recognition from radars and satellites, employs a random uniform under-sampling strategy and an ensemble model (HPEC) for severity estimation. Experimental results, validated with real data from the Department of Civil Protection, demonstrate significant improvements over the widely used Kriging method. HPEC exhibits enhanced capability in detecting rainfall events, as evidenced by higher POD (0.58) and lower MSE (0.11) compared to Kriging (0.48 and 0.15, respectively). The computational efficiency of our approach, attributed to the quadratic complexity of ML algorithms like RF and the scalability of ensemble methods, further underscores its advantages. Additionally, our analysis underscores the crucial contributions of all data sources, with the removal of any source adversely impacting algorithm performance. Future work involves validating the method over larger time intervals, considering seasonal variability, and exploring the incremental building of the ensemble model with new data. Furthermore, we aim to assess the algorithm's effectiveness in identifying highly localized heavy precipitation events through time series analysis of individual feature contributions from radar and Meteosat data.

**8.BIBLIOGRAPHY**

#### 8.BIBLIOGRAPHY

##### REFERENCES

[1] <https://ieeexplore.ieee.org/document/9272667>

[2] <https://staff.icar.cnr.it/folino/papers/tgrs20.pdf>

[3]<https://www.researchgate.net/publication/346656388_A_Machine_Learning_Approach_for_Rainfall_Estimation_Integrating_Heterogeneous_Data_Sources>

##### GITHUB LINK

[https://github.com/A-Machine-Learning-Approach-for-Rainfall-Estimation-Integrating-Heterogeneous-Data-Sources](https://github.com/vishnuvardhan81/A-Machine-Learning-Approach-for-Rainfall-Estimation-Integrating-Heterogeneous-Data-Sources)

**CMRTC** 37