Index/Table of Contents

Aim

To implement the best machine learning algorithm and model to predict rainfall and flood accurately.

RAINFALL AND FLOOD PREDICTION

ABSTRACT :

Flood and heavy Rainfall is the most common and dangerous natural disaster in India and all over the world. India has suffered deeply because of the rainfall or flood. As we know, climate directly affects humans and wildlife. In India, the most important factor for survival is Agriculture. Not only rural but urban areas are also affected badly, livelihood, savings, and all struggles vanish in one go. It all happens because prediction of heavy rainfall or flood is not done, which is a big problem. Predicting the amount of rainfall gives alertness to farmers by knowing early so that they can protect their crops and properties from rain. Not only farmers but we can alert people working in coastal areas like fishermen. We can alert them as well. Children's and people working can also be alerted so that they can know the real-time condition of their area. To help society and resolve the problem we are using the ML algorithms of them are ANN, CART, Genetic Algorithms and SVN.

INTRODUCTION :

Nowadays, rainfall is an important factor for many things. considered to be one of the most. The Primary Sector plays a very vital role in India where Agriculture is one of the most important factors in deciding the economy of the country and agriculture is totally dependent on rainfall. Not only the agriculture sector but rainfall is also more important in coastal areas around the world by getting to know the rainfall is very much necessary to protect their life's from the floods and heavy rainfall. In some of the areas which are having drought, to establish a rainfall harvester, proper prediction of rainfall is necessary. This project completely deals with predicting rainfall using Machine learning and Neural networks. In this, we are executing a comparative study of machine learning approaches and neural network approaches then accordingly selects the perfect approach for prediction of rainfall. First the pre process is

performed, i.e. representing the input dataset in graph form. Then from , ML techniques, LASSO (Least absolute shrinkage and selection operator) regression is used and ANN (Artificial Neural Network) approach is used for neural networks. The prediction of rainfall is made with the approach which has highest accuracy in the outcomes. The prediction of the rainfall is done with the dataset which has the information related to rainfall.

MOTIVATION :

We are Students of SRM residing in Chennai, Tamil Nadu. Every month we face an issue , alert or problems related to heavy rainfall or floods. The most important factor which motivated us was the life of living not only human beings but wildlife as well. From a human perspective, the life of any person is very important on a regular basis. We hear that a certain proportion of people died due to heavy rainfall in a particular area due to lack of knowledge or resources. We just want to convey right and most importantly accurate information to society. Apart from this, our aim is to accurately determine the rainfall for water resources, crops, pre planning of water resources in drought areas and for agricultural purposes. From earlier information it benefits the farmer for better managing their crops from floods. It helps them to increase the economic growth of a country by efficient rainfall. Prediction of precipitation is necessary to save the life of people and properties from flooding. Prediction of rainfall helps people in coastal areas by preventing floods.

BUSINESS TEMPLATE

The Project	-Flood and rainfall prediction using a meteorological data set analyzing the important characteristics using time series and various classifiers. -using data pre-processing procedure by applying RST -using a modular approach.
History	-Lack of implementation of proper algorithm with more accuracy -No effective use of water resources, crop productivity and pre-planning of water structures.
Objective	- to find the most frequent meteorological phenomena which affects our state most

Approach	-Linear regression and Neural Networks implemented to predict rainfall. -Data-driven approach for accurate rainfall prediction. -Data cleaning.
Limitations	-most recent dataset is not available which could result in better accuracy than achieved. -No fully objective way of selecting predictors. -Difficult to perform statistical tests with the results.
Benefits	-Avoidance of loss of life. -Farmers can save their crops according to the prediction that leads to economic growth. Assists with operational activities to improve conveyance/avoid current blockage prior to flooding.

Result: Thus, the project team formed, the project is described, the business case was prepared and the problem statement was arrived.

Aim

To identify the appropriate Process Model for the project and prepare Stakeholder and User Description.

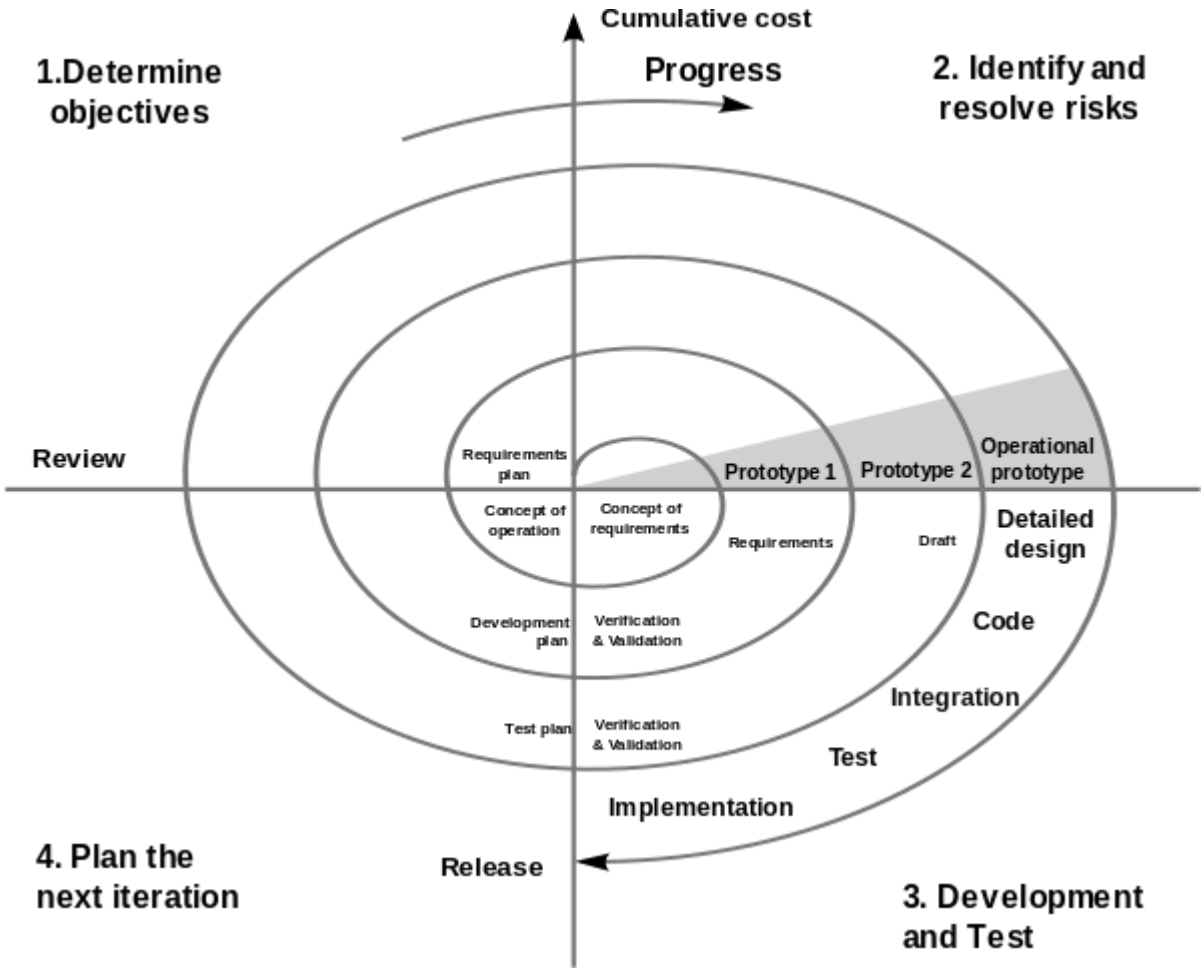
STAKEHOLDERS AND USER DESCRIPTION

Project Name	RAINFALL PREDICTION			
Project Stakeholder Name	Specific Information	Project Interest	Impact on Project	Role
Project Manager	Managing, Designing of aWebsite	Technical and Busines s Handle	Supporter	Decision Making, Participant

Top Manager	Controlling, Dividing work, Overall Handling, Allocating Resources	Leader	Motivational, Influencer	Decision Making, Consultant
Developer(ML Engineer)	Handling Backend - Front end and Designing Programming	Skilled and experienced Programmer	Supporter	Participant
Investors	Financial Support	People who supports Financially	Supporter	Collaborator
Sponsors	Financial Support and Advertisement	People who do Advertisement to grow our market	Supporter	Information Recipient, Participant
Reviewers	Checking the product and giving review	Managing the review	Positive or Negative	Participant
World Meteorological Organization	Organization Head on global level	Organization head	Supporter	Information Recipient, Collaborator

Central Water Commission, Ministry of Water Resource	Organization Head on national level	Organization head	Supporter	Information Recipient, Collaborator

SOFTWARE MODEL USED



The Spiral model is best suitable for our rainfall and flood prediction model. This development model includes a four-phase that is called a Spiral. The spiral model involves identification, design, build, evaluation and analysis of risk of floods and rainfall on the basis of varying dataset.

Stages-

1. Objective-Rainfall prediction based on the dataset
2. Identify and resolve risks
3. Development and test
4. Plan the next iteration on identification of risks.

The main objective of spiral mode is to identify and resolve the risk, the main problem we are facing here is Data Identification, Data collection. So, here basically the initial point let us assume it to be 1950s data, so as time passes the next phase can be in the 1970s and as the spiral keeps on increasing the risk factor also increases at a high rate.

Result: Thus the Project Methodology was identified and the stakeholders were described.

Aim

To identify the system, modules, functional and non-functional requirements for the project.

MODULES

1.Home Page

2.About

3.Latest News about rainfall and flood prediction

4.Rainfall Analysis Home Page

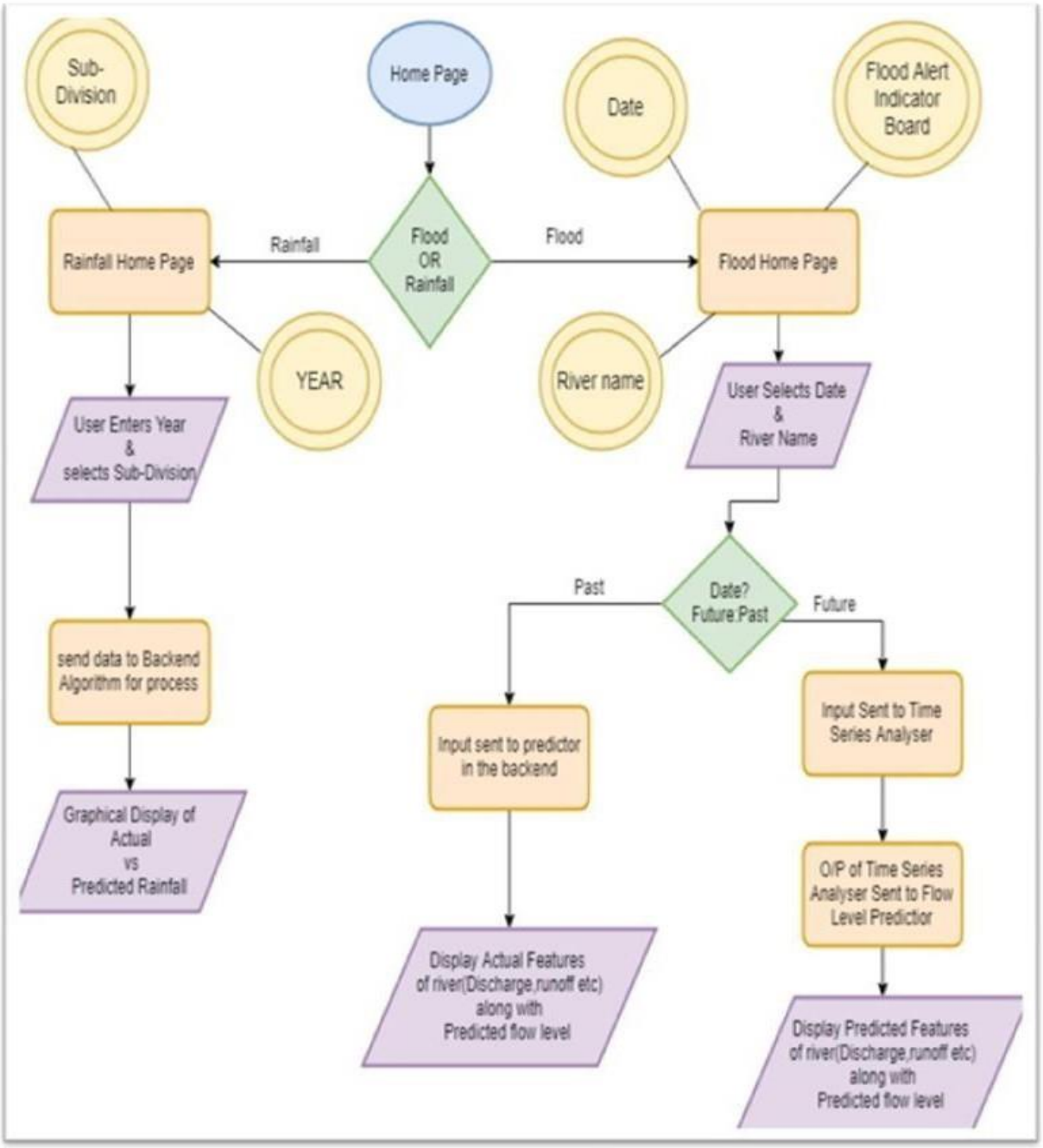
4.1 User enters Year and selects sub-division

4.2 Graphical display of actual vs predicted rainfall

5.Flood Prediction Home Page-The user interface to the common public to check the level of water flow in rivers in future and have provided a mechanism of notification if there is any possibility of flood due to any river in the near future(12 months). Along with that users can also see the historical trends of river flow and can visualize the rainfall patterns also in their Sub-Division(Area).

5.1 User selects Date and River Name

5.2 Display of Actual or predicted features along with predicted flow level


Sub-Division
Home Page
Date
Flood Alert Indicator Board
Rainfall Home Page
Rainfall
Flood OR Rainfall
Flood
Flood Home Page
YEAR
River name
User Enters Year & selects Sub-Division
User Selects Date & River Name
send data to Backend Algorithm for process
Past
Date? Future Past
Future
Graphical Display of Actual vs Predicted Rainfall
Input sent to predictor in the backend
Input Sent to Time Series Analyser
O/P of Time Series Analyser Sent to Flow Level Predictor
Display Actual Features of river(Discharge,runoff etc) along with Predicted flow level
Display Predicted Features of river(Discharge,runoff etc) along with Predicted flow level

COMPARISON BETWEEN SOFTWARE PROCESS MODEL

Properties	Waterfall Model	Incremental Model	Spiral Model	RAD Model
Planning at early stage	Yes	Yes	Yes	No
Returning at early phase	No	Yes	Yes	Yes
Testing	After completion of coding phase	After every Iteration	After end of engineering phase	After completion of coding phase
Detailed Documentation	Necessary	Yes but not much	Yes	Limited
Cost	Low	Low	High	Low
Requirement Specification	Beginning	Beginning	Beginning	Time boxed release
Flexible to change	Difficult	Easy	Easy	Easy
Maintenance	Low	High	Depends	Medium
Duration	Long	Very long	Short	Long
Reusability	Least	To some extent	To some extent	Yes
Customer Control	Low	Yes	Yes	Yes
Team Size	Large Team	Not Large Team	Large Team	Small Team

REQUIREMENTS

SYSTEM REQUIREMENT :

1. Standard hard drive
2. Network Connectivity
3. Disk storage
4. UPS and Surge protector
5. Internet Connectivity
6. Processor - min : 1.9 GigaHertz or x64 bit dual core max : 3.3 GigaHertz or faster 64 bit dual core
7. Memory : minimum 2Gb RAM (4GB recommended)

USER REQUIREMENT :

1. Web browser installed
2. Browser : Google Chrome/MacOS/Linux(updated version)
3. Web browser supporting HTML 3.2 (or later) and cookies
4. Internet connectivity
5. Keyboard and mouse
6. Access to the data

FUNCTIONAL REQUIREMENT :

1. User Profiles-login page
2. Database
3. Updates for upgradation of website
4. News updates on website about red alert areas
5. Usage History
6. User Statistics
7. The website shall provide weather parameters, temperature, pressure, wind, speed, direction, rainfall and humidity.
8. The system shall be able to produce minimum,maximum and the average data of a particular weather parameter when it is requested by an operator/user.

NON FUNCTIONAL REQUIREMENT :

1. Support multiple LogIn/ users.
2. Boot up in less than 5 sec
3. Handle multiple queries
4. Able to retrieve data from backend in less than 10 secs
5. Availability of Image
6. Focusing on user expectation
7. Usability, Reliability, Scalability, performance etc.

Result: Thus the requirements were identified and accordingly described.

Aim

To Prepare Project Plan based on scope, Calculate Project effort based on resources, find Job roles and responsibilities.

ESTIMATION OF PROJECT METRICS

CONSTRUCTIVE COST ESTIMATION MODEL -

COCOMO is one of the most generally used software estimation models in the world. It was proposed by Boehm. Based on the size of the software, it predicts the efforts and schedule of a software product.

MODULES

1. Home Page - 100 KLOC
2. About - 50 KLOC
3. Latest News about rainfall and flood prediction - 200 KLOC
4. Rainfall Analysis Home Page - 702 KLOC
5. Flood Prediction Home Page - 700 KLOC

TOTAL : 1752 KLOC

Additional Info : Data Set - 4117 KLOC

TOTAL : 5869 KLOC

SEMI DETACHED SYSTEM

The projects classified as Semi-Detached are comparatively less familiar and difficult to develop compared to the organic ones and require more experience and better guidance and creativity.

BASIC MODEL:

1. Effort = E = a*(KLOC)^b
2. Time = c*(Effort)^d
3. Person Required = Effort / Time

Software Projects	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semi Detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Efforts = E = 3.0*(5869)^(1.12)
= 21737.19

Time = 2.5*(21737.19)^(0.35)
= 82.40

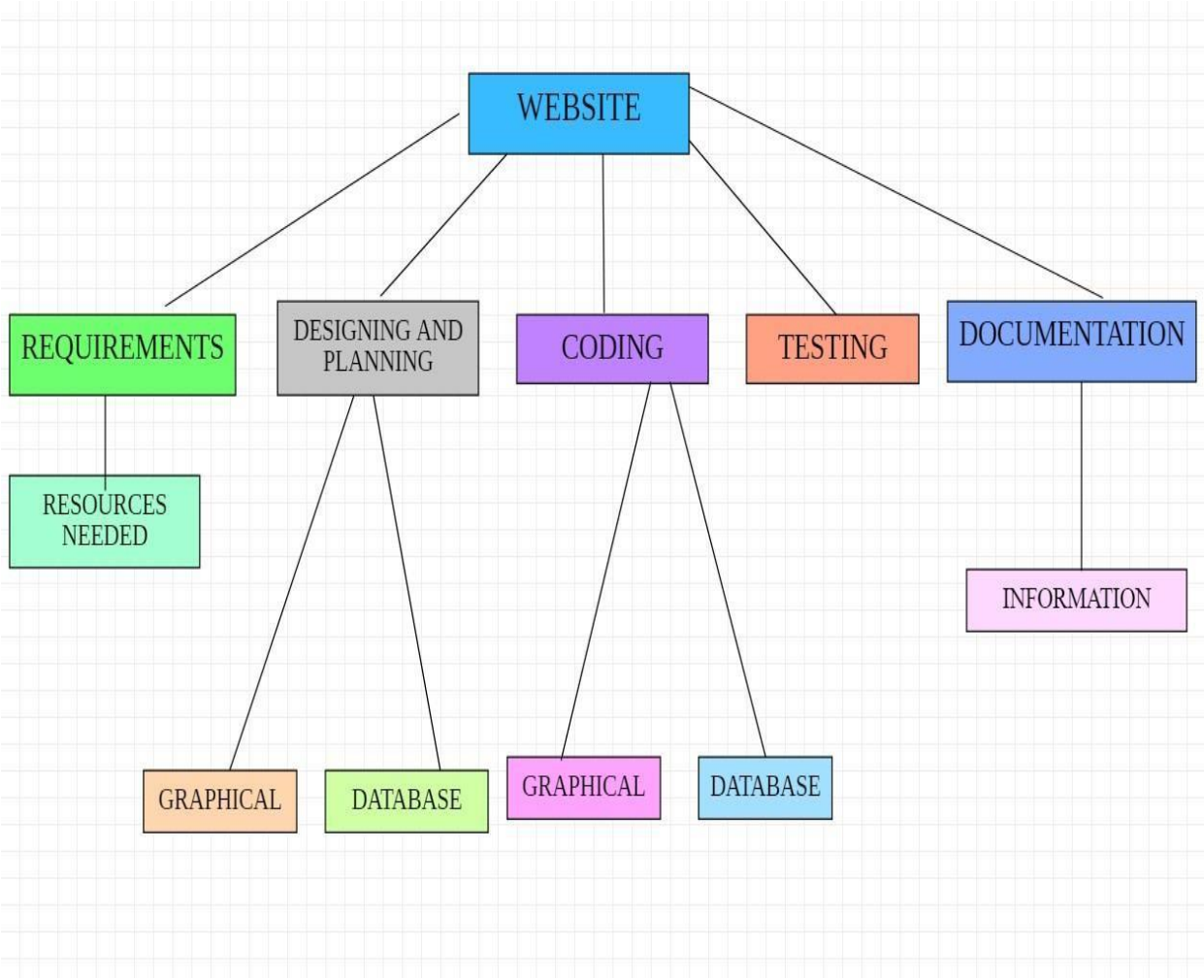
Person Required = Effort / Time
= 21737.19 / 82.40
= 263.80 ~ 263 or 264 persons

Result: Thus, job efforts and productivity were successfully calculated, using the COCOMO model.

Aim

To prepare Work breakdown structure, Timeline chart, Risk identification table.

WORK BREAK DONE STRUCTURE OF WEBSITE



RISK ANALYSIS

- **Risk Compartment Analysis of Rainstorm and Flood Disaster:**

  - Logistic regression model is used for correlation analysis for binary variable of dependent variable, to reduce the impact of indices data on the error of the hybrid PSO-SVR prediction model. In addition, the occurrence of rainstorm and flood disasters is represented by 0 i.e the number of disasters is 0 and 1 i.e the number of disasters is not 0 as the dependent variables of two classifications.

  - In Risk Compartment Analysis of rainstorm and flood disaster includes four part analysis and they are as follows:

    1. Risk analysis of Hazard Factors
    2. Sensitivity analysis of hazard factors i.e mainly analyzing the influencing factors of risk sources.
    3. Vulnerability analysis of hazard-bearing body mainly analyzes the influence of different rainstorm and flood intensity disasters for the distribution of population and the distribution of population and the condition of regional economic and infrastructure.
    4. Compartment Analysis of Disaster Factors

❖ Project risks:

● Based on the reliability and availability of disaster loss data caused by historical rainstorms and floods. ● Poor and inaccurate estimations.

❖ Management:

- Only work on high-priority tasks.
- Include tech spikes.
- Add an allocation factor.
- Think about the cone of uncertainty when giving the estimate.

❖ Technical risks:

- Quality and timeliness of the software to be produced ● Poor or bad quality of code.

❖ Management:

- Do not forget to do the code reviews.

- Test all codes.

- Clear coding standards and guidelines are a must.
- Manage the quality of project

❖ Business Risks:

- They threaten the possibility of the software to be built.
- If they become real ,they put in danger for the project or the product.

Management:
Set a fixed budget, manage and analyze accordingly from the beginning of the project

SWOT ANALYSIS

STRENGTHS

1. Easily Sign in & logout process.
2. Planning upcoming schedule.
3. Support and emergency services time to prepare for flooding.
4. Timely operation of flood control centre like opening gates and dams.
5. Make use of existence facilities

WEAKNESSES

1. The information might be wrong or not up to date.
2. Lack of coordination and communication between agencies of government.
3. Lack of funding resources.

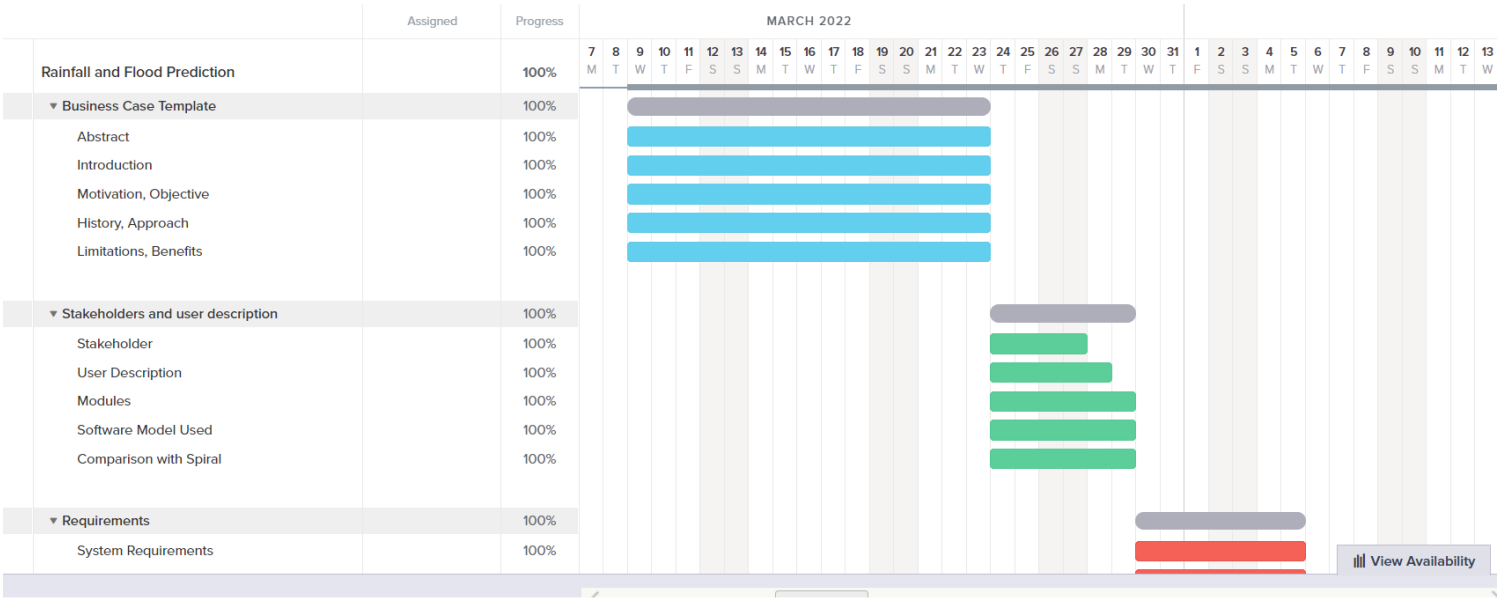
OPPORTUNITIES

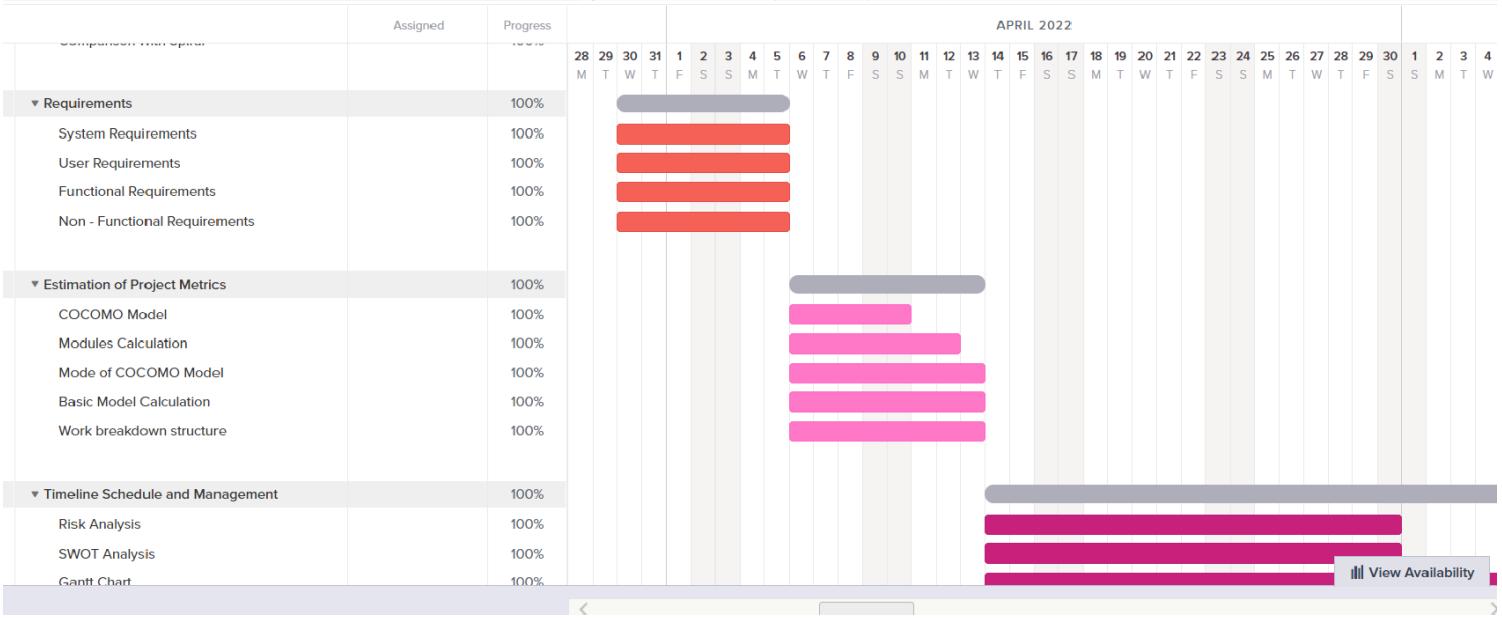
1. Developing in ongoing project.
2. Can work together with NGOs and various public and private sectors.
3. New Technology

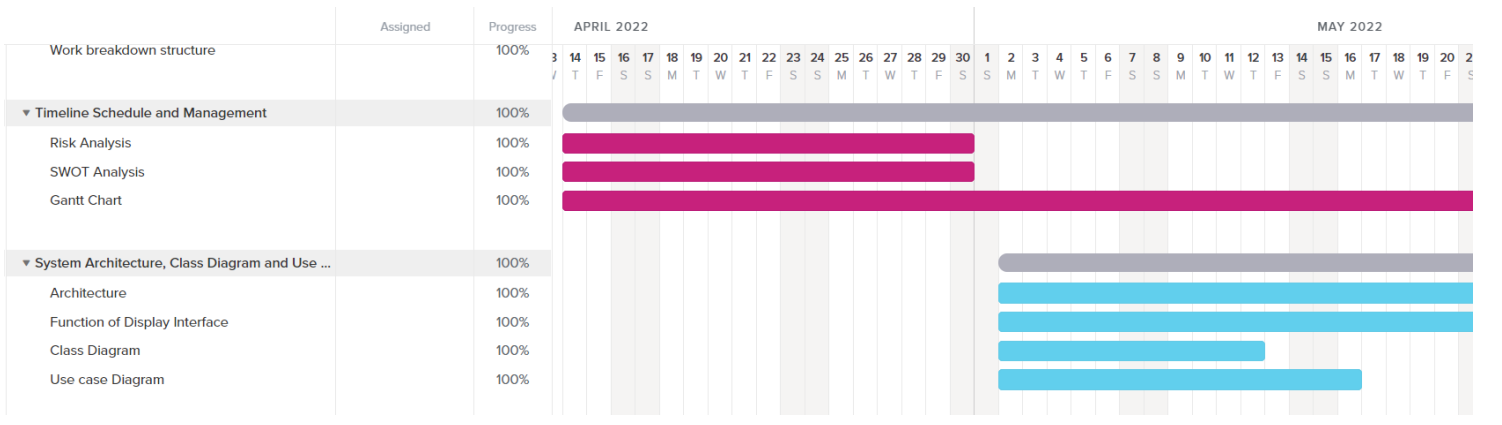
THREATS

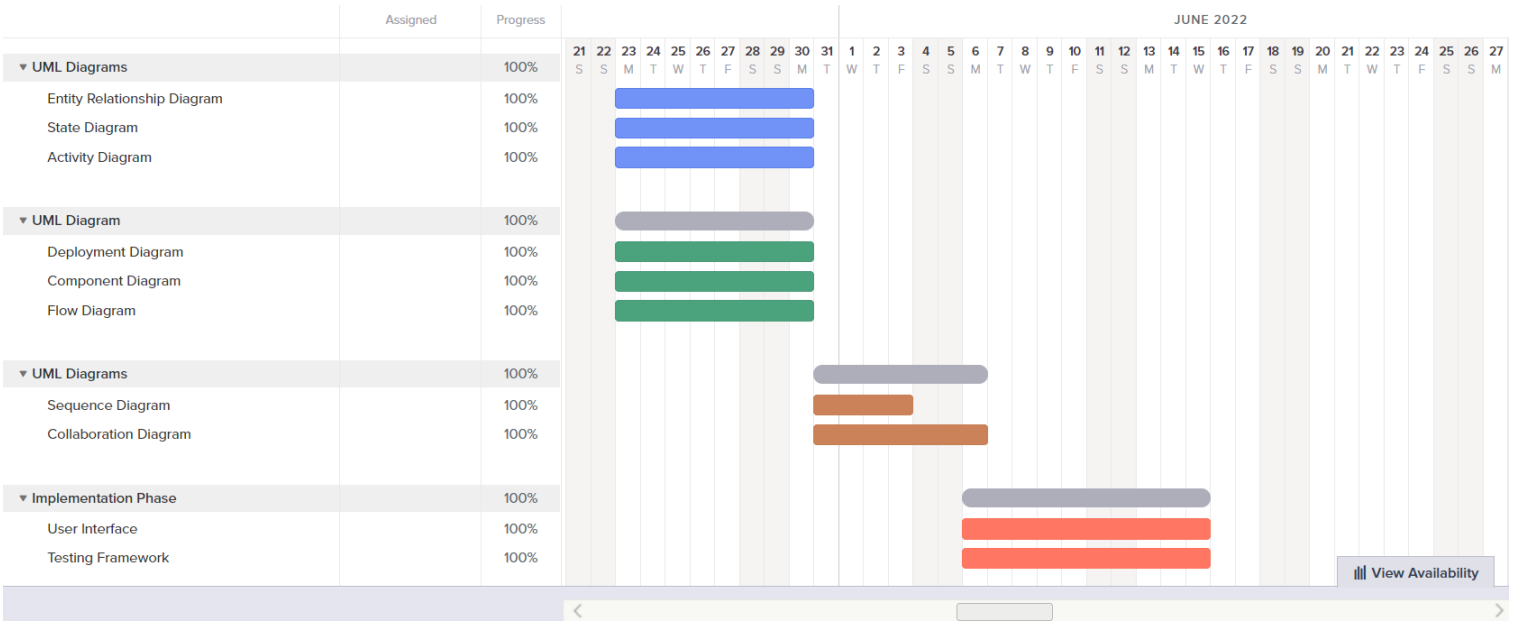
1. Delayed evacuation response.
2. Damage to system provider.
3. Overlapping of responsibilities.
4. Error in sign in and logout.
5. Server down.

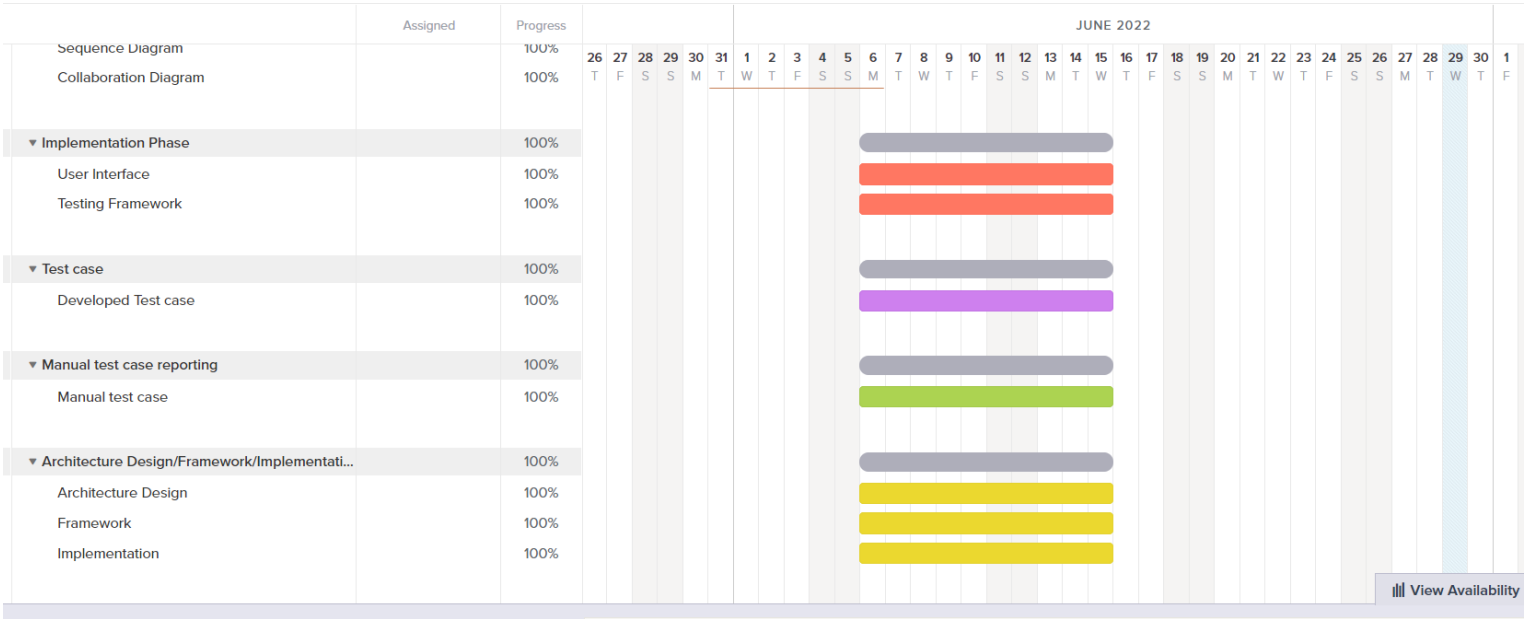
GANTT CHART


Assigned
Progress
MARCH 2022
Rainfall and Flood Prediction 100%
Business Case Template 100%
Abstract 100%
Introduction 100%
Motivation, Objective 100%
History, Approach 100%
Limitations, Benefits 100%
Stakeholders and user description 100%
Stakeholder 100%
User Description 100%
Modules 100%
Software Model Used 100%
Comparison with Spiral 100%
Requirements 100%
System Requirements 100%
View Availability


Assigned
Progress
APRIL 2022
Requirements 100%
System Requirements 100%
User Requirements 100%
Functional Requirements 100%
Non - Functional Requirements 100%
Estimation of Project Metrics 100%
COCOMO Model 100%
Modules Calculation 100%
Mode of COCOMO Model 100%
Basic Model Calculation 100%
Work breakdown structure 100%
Timeline Schedule and Management 100%
Risk Analysis 100%
SWOT Analysis 100%
Gantt Chart 100%
View Availability


Assigned
Progress
APRIL 2022
MAY 2022
Work breakdown structure 100%
Timeline Schedule and Management 100%
Risk Analysis 100%
SWOT Analysis 100%
Gantt Chart 100%
System Architecture, Class Diagram and Use ... 100%
Architecture 100%
Function of Display Interface 100%
Class Diagram 100%
Use case Diagram 100%



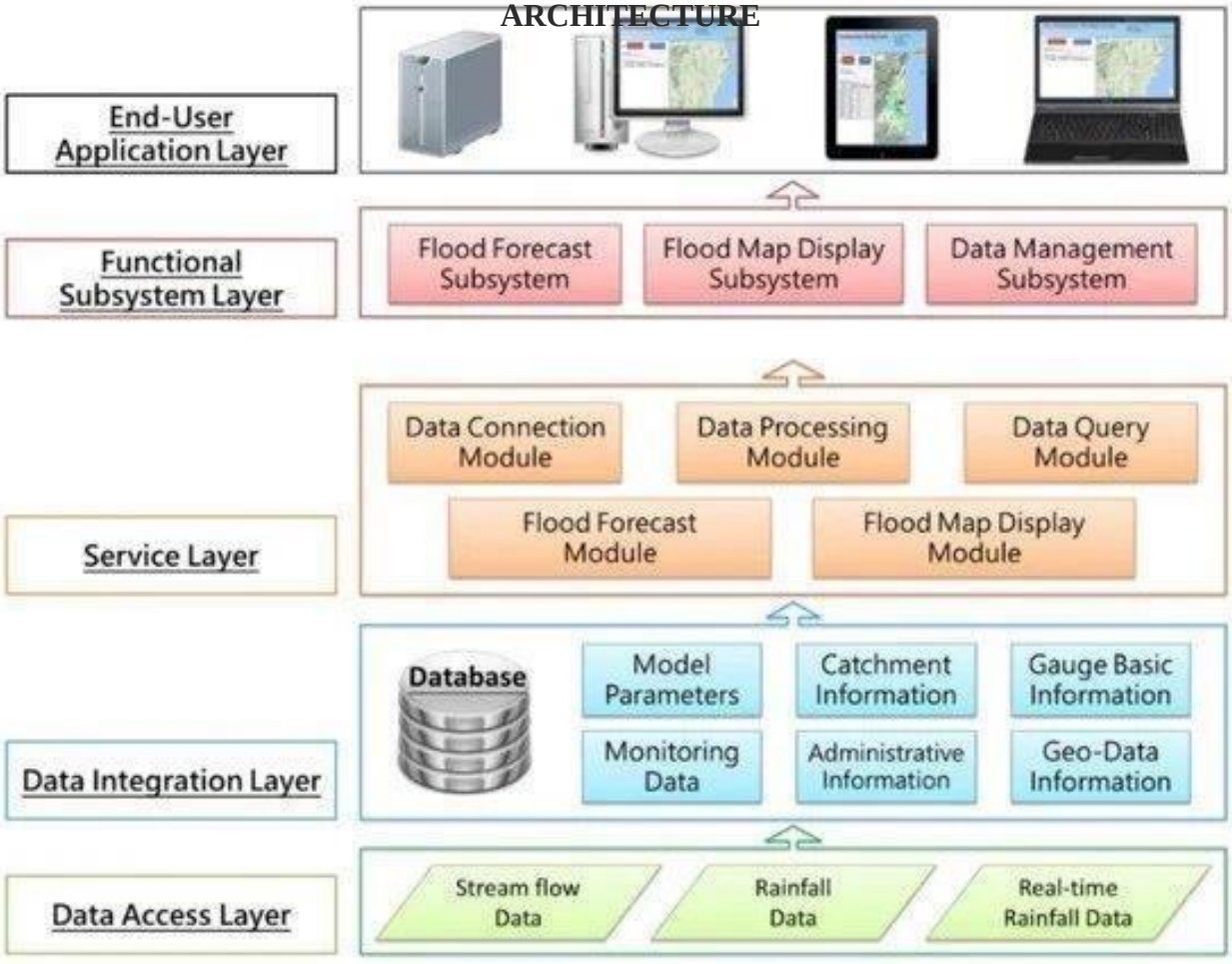


Result: Thus, the Work-Breakdown structure, Timeline diagram and the Risk Table, Swot Analysis and Gantt chart were designed successfully.

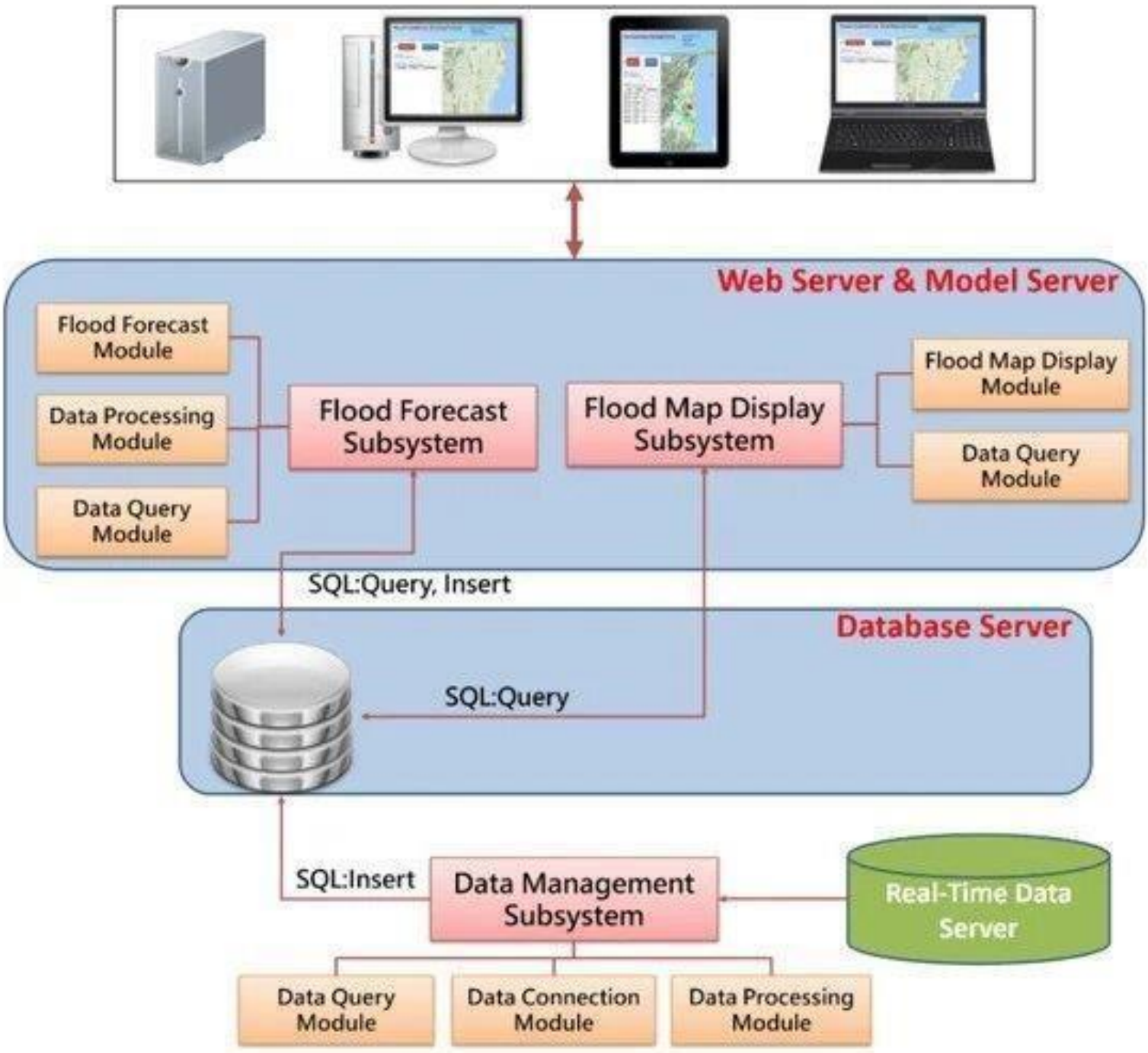
Aim

To Design a System Architecture, Use case and Class Diagram.

SYSTEM ARCHITECTURE



RELATIONSHIP BETWEEN FOUR SERVERS, FIVE MODULES AND THREE SUB-SYSTEMS



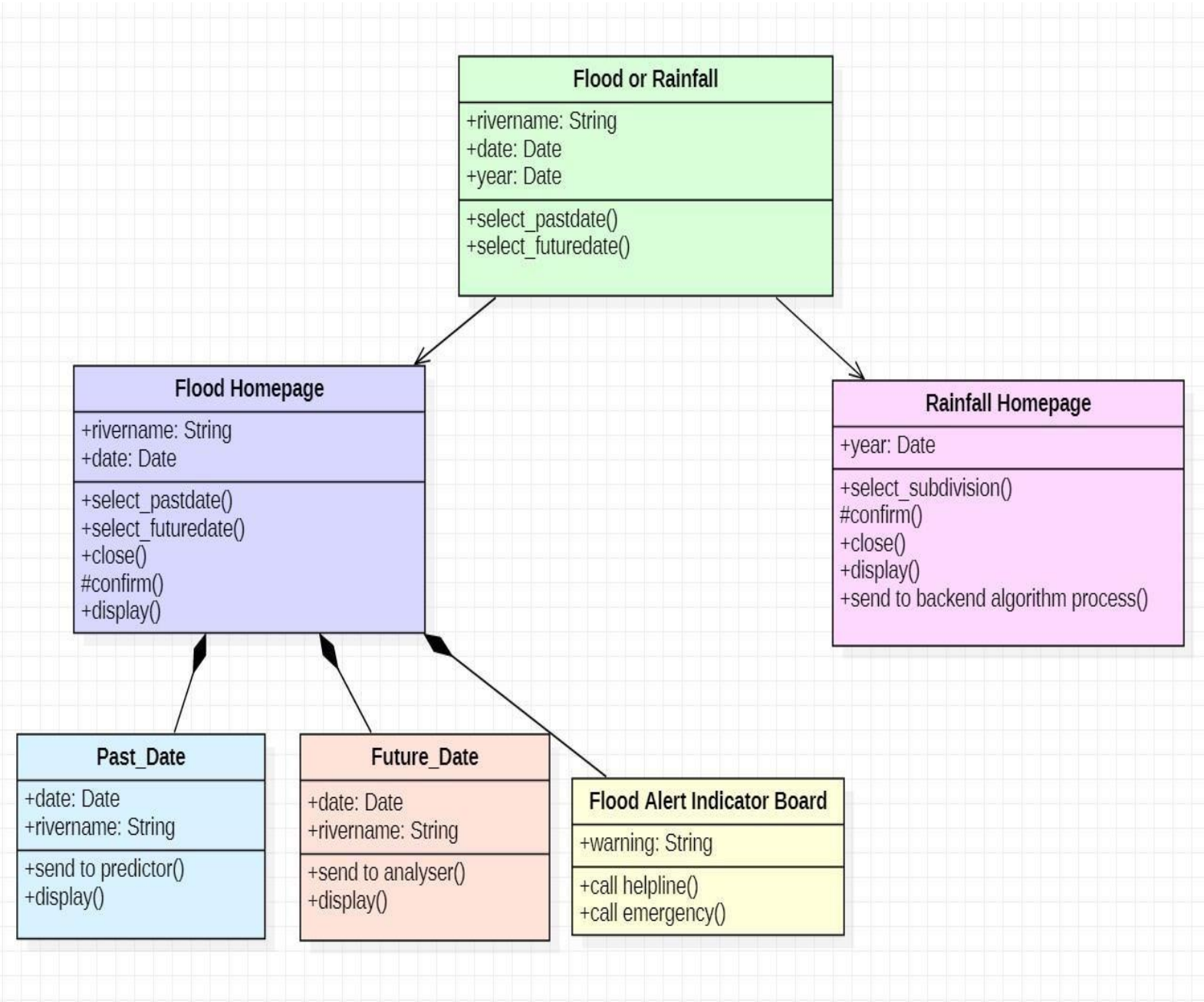
FUNCTION OF DISPLAY INTERFACE

(a)

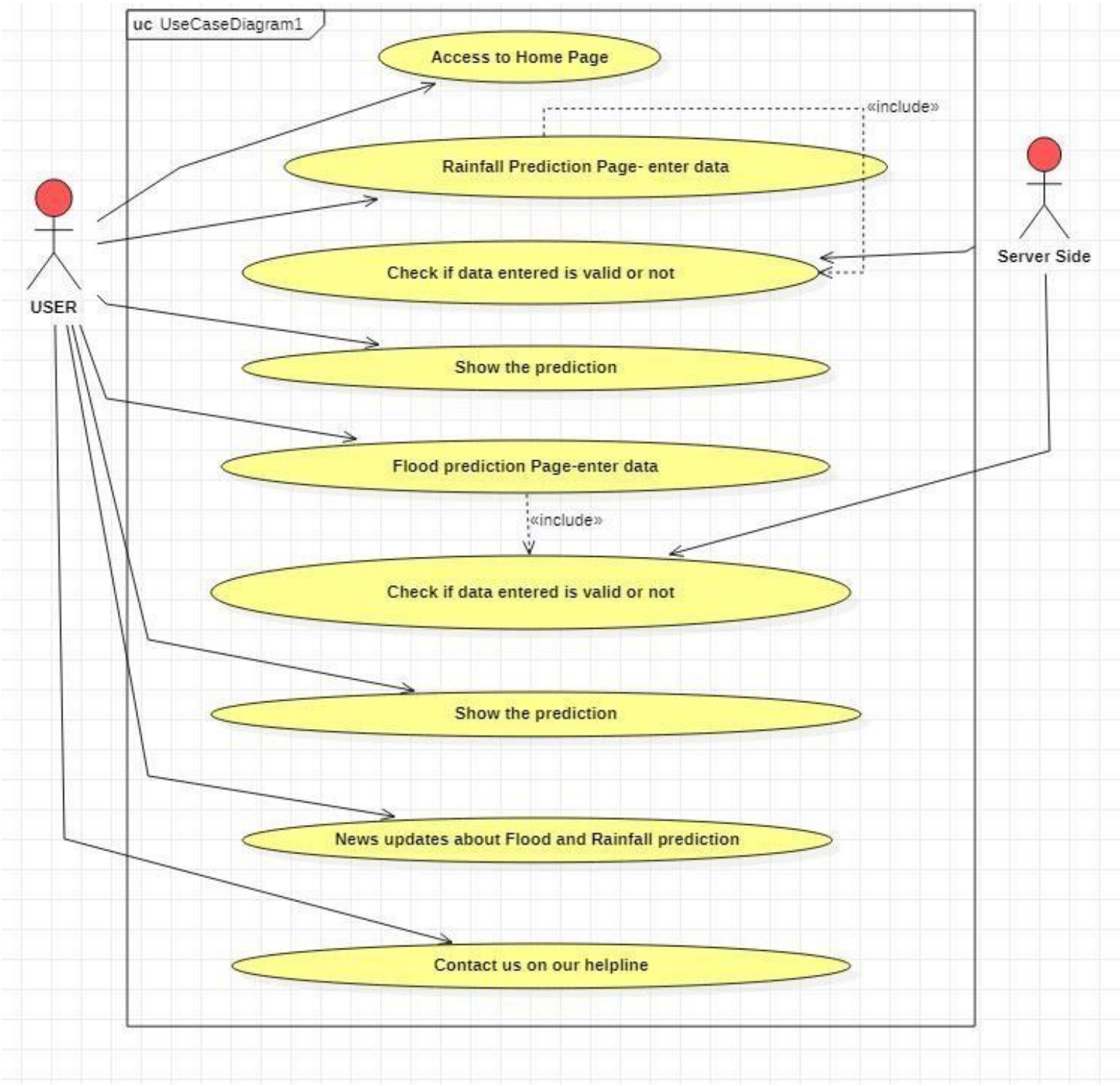
(b)

(c)

CLASS DIAGRAM



USE CASE DIAGRAM



Result: The use-case, class and architecture diagrams for the system were designed successfully.

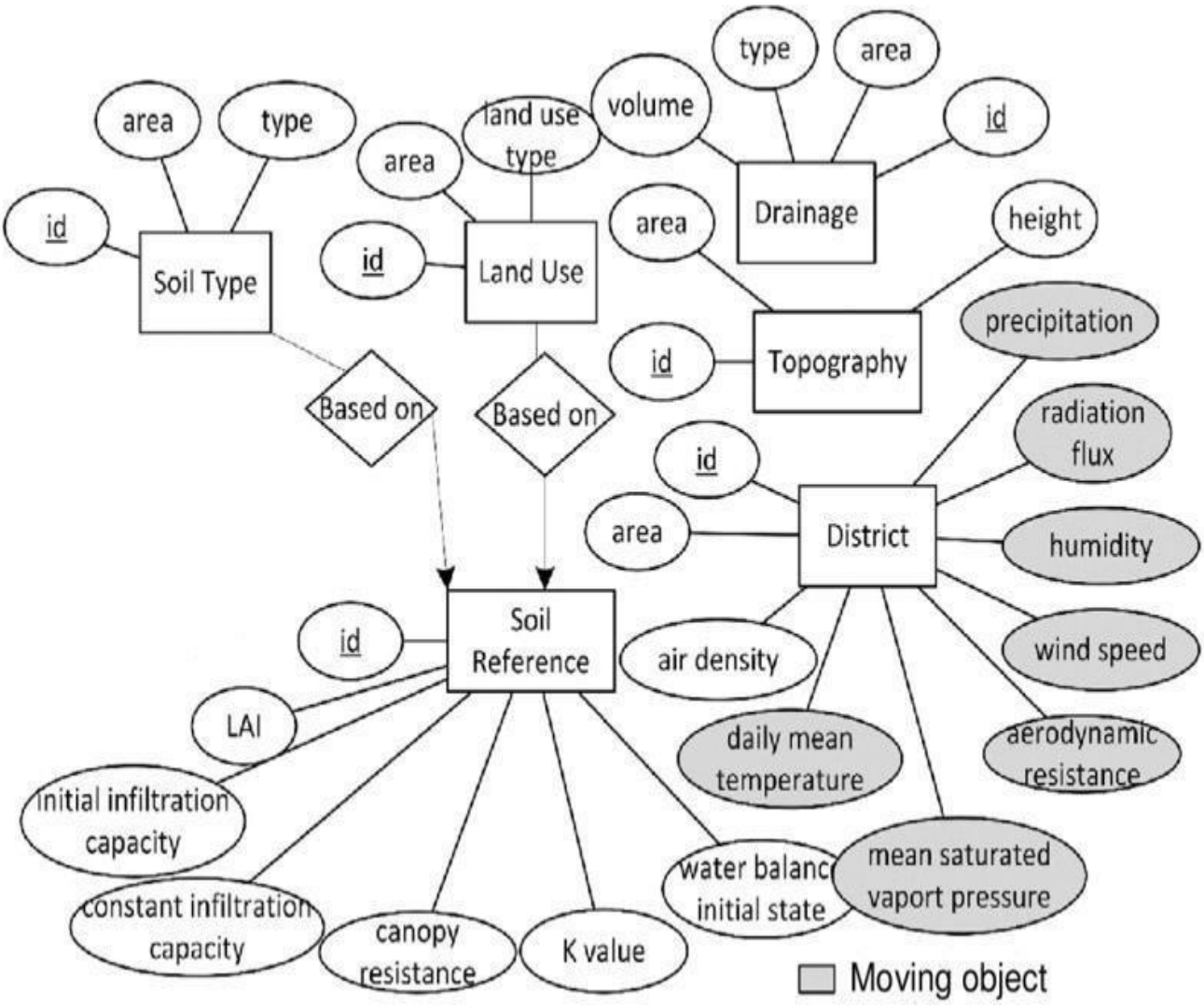
Aim

To Design Entity relationship Diagram.

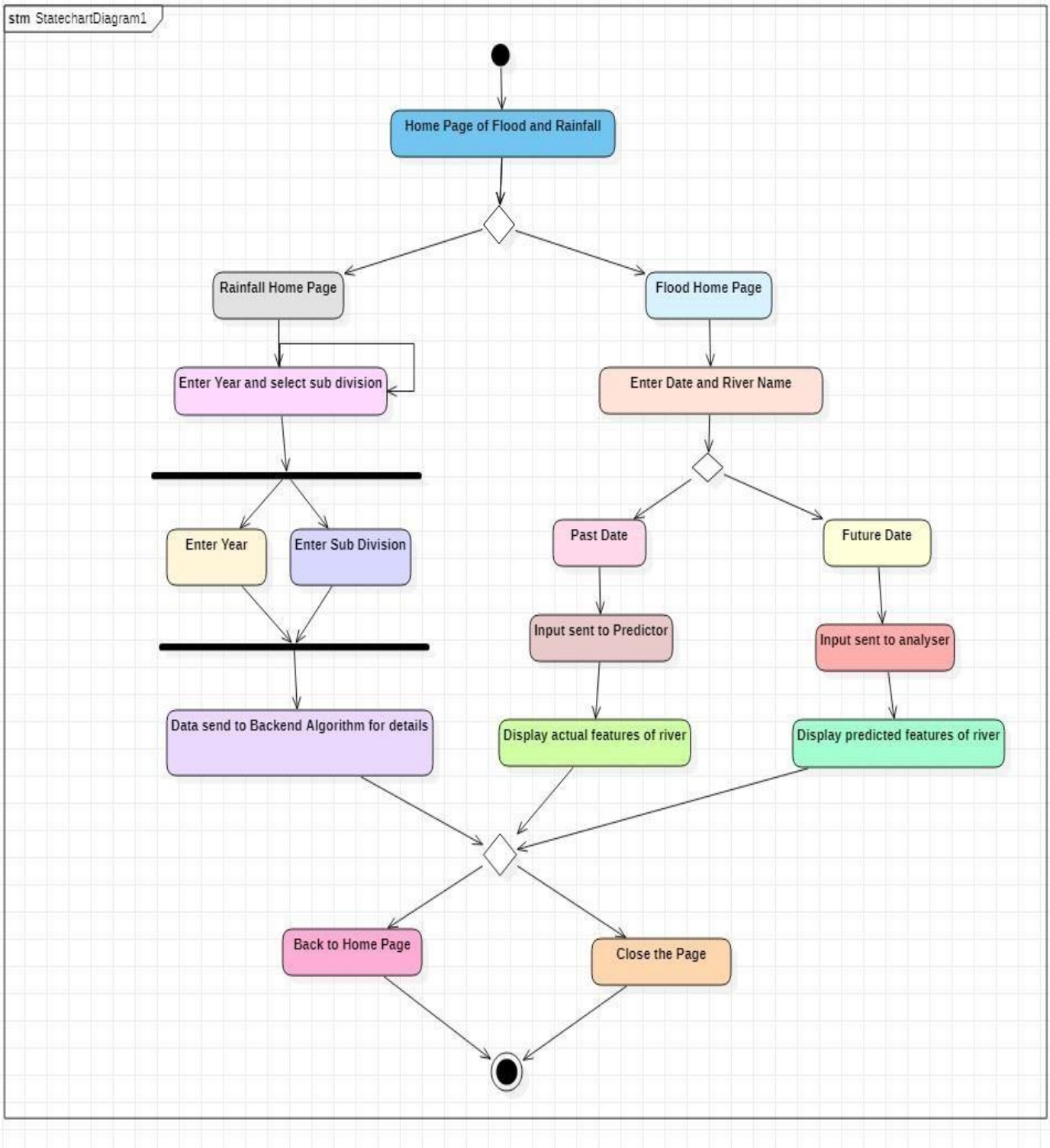
Team Members:

S. No	Register No	Name	Role
1	RA2011033010041	Sanskriti Sinha	Member
2	RA2011033010047	Gayatri Malladi	Member

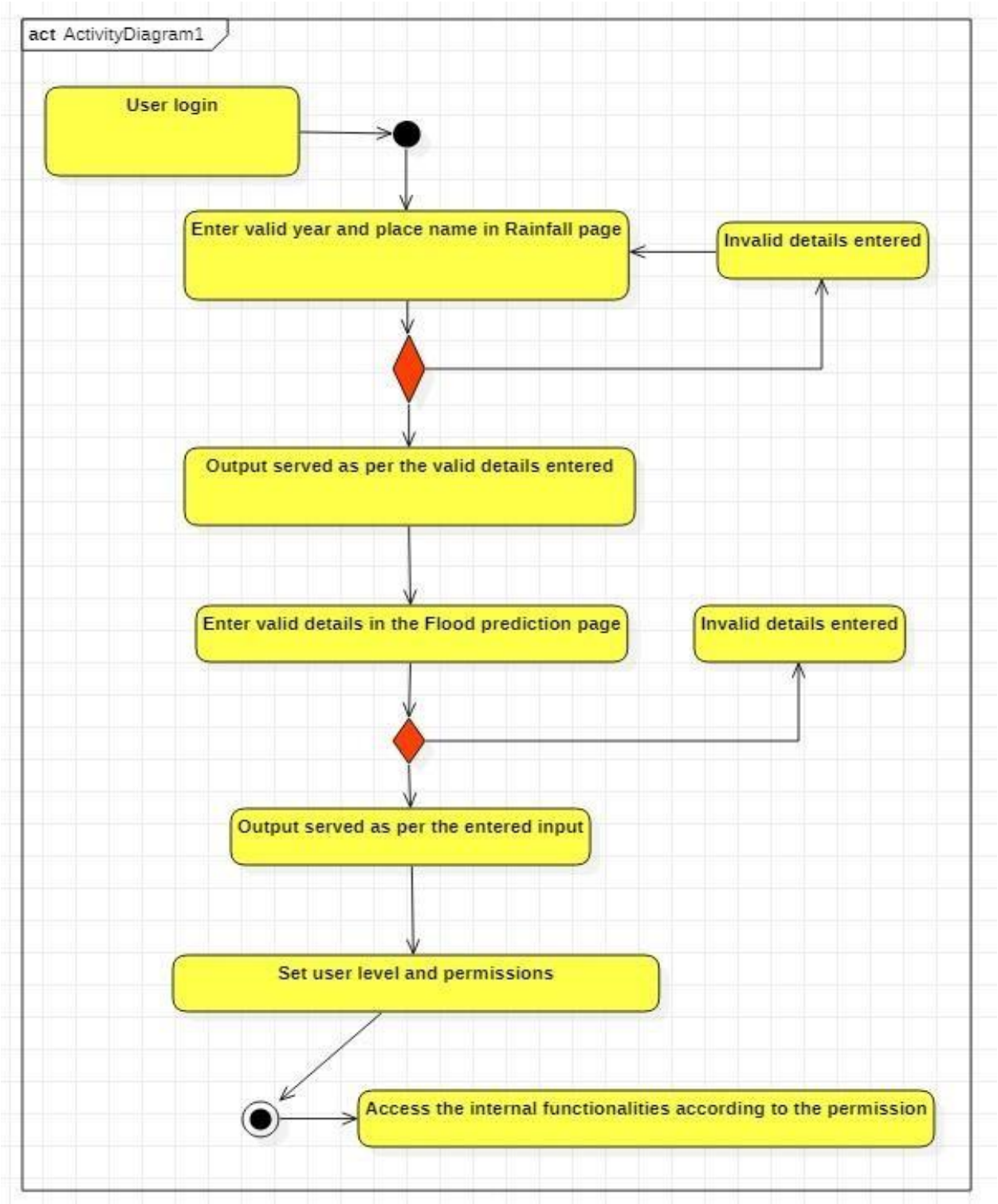
ER DIAGRAM



STATE CHART DIAGRAM



ACTIVITY DIAGRAM

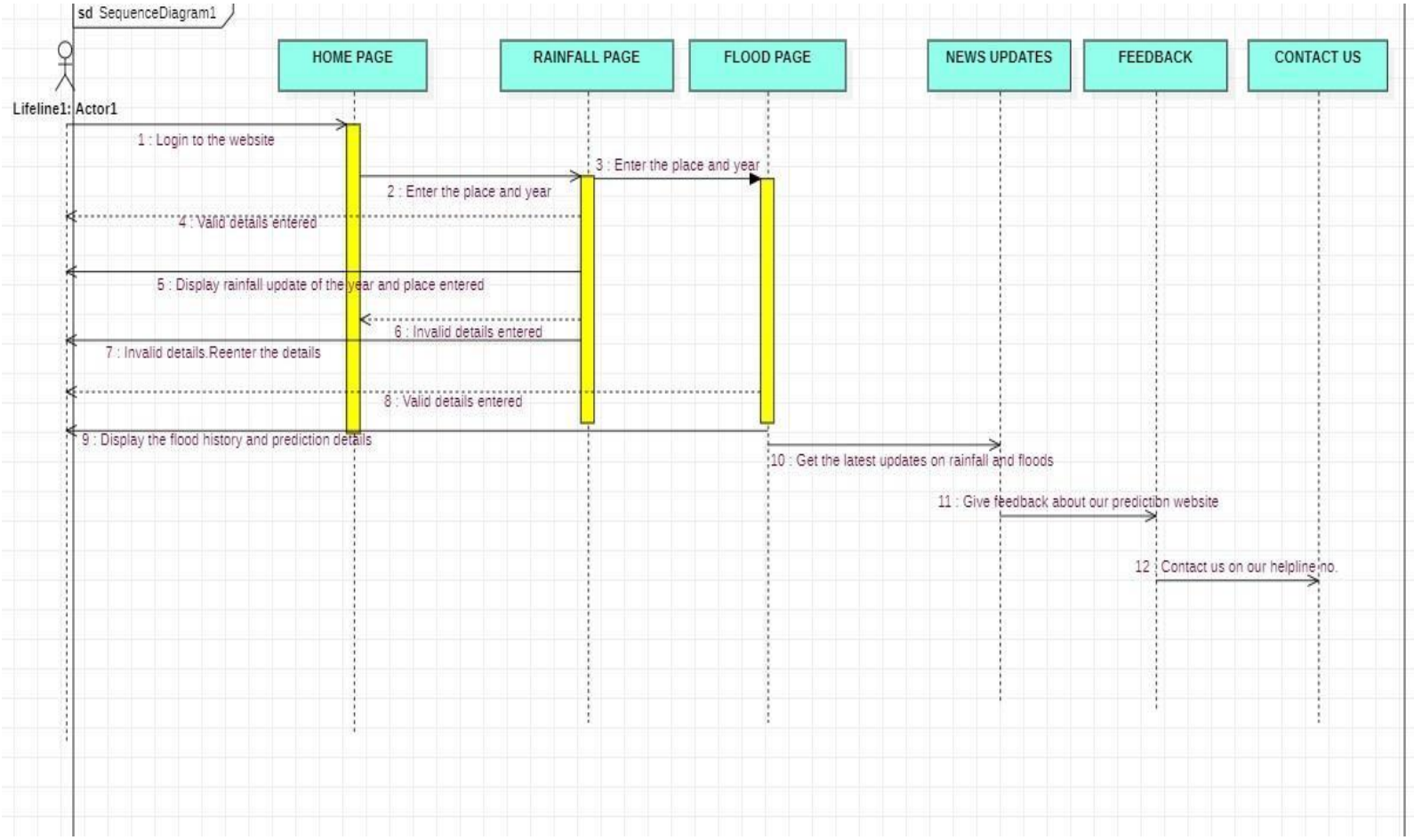


Result: Thus, the data flow diagrams have been created

Aim

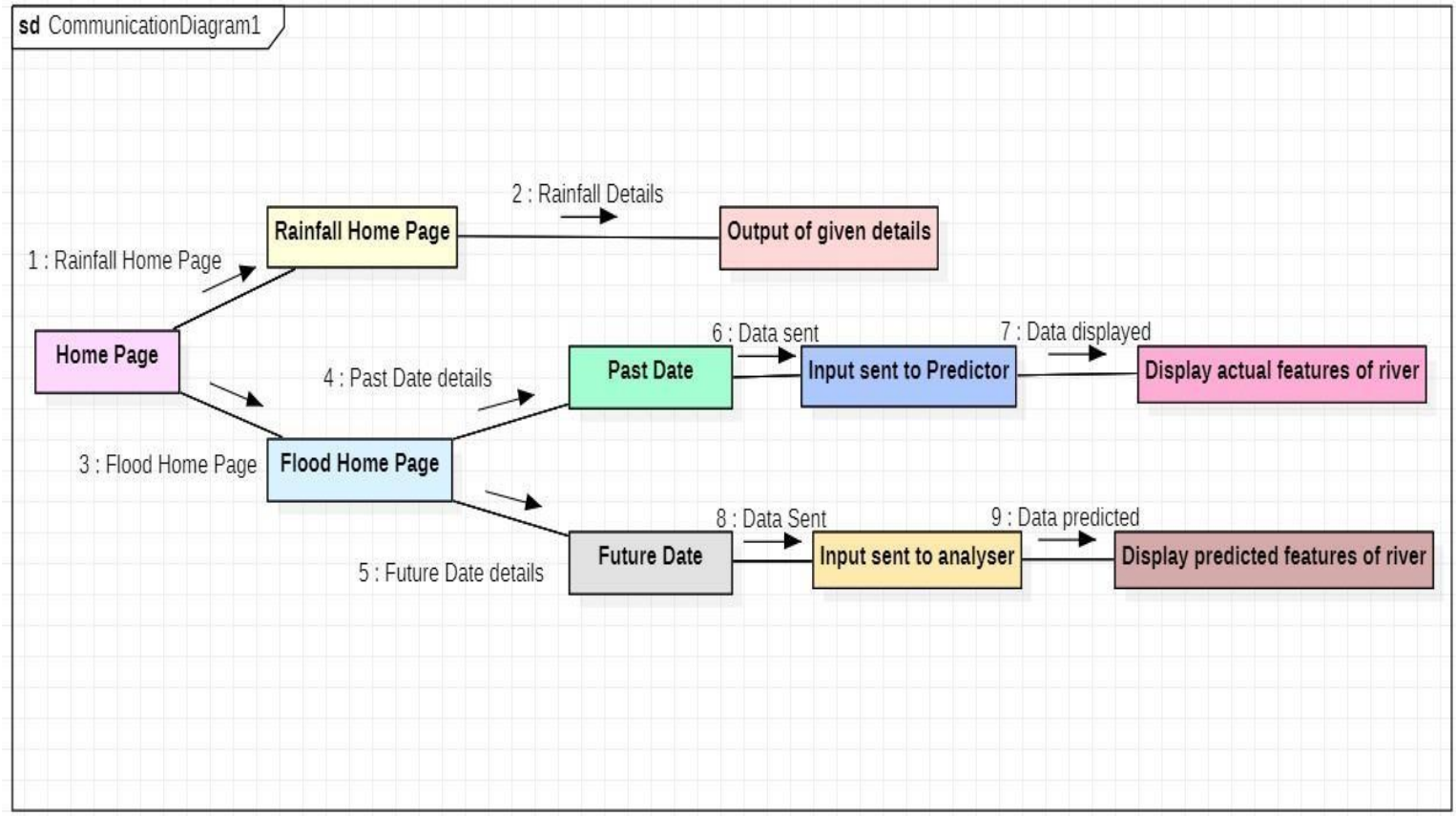
To develop the Sequence and Collaboration Diagram.

SEQUENCE DIAGRAM



The user enters to the login page first. Then the web page proceeds to Rainfall page where the user is asked to enter the place and year .According to the details entered by the user the server rectifies the validity of the data entered. If the data entered is valid the information about rainfall prediction for the entered details is show. If the data entered is invalid, The page requests user to enter valid details again. The user can then proceed to Flood prediction page and again enter details about place and year. The same procedure is repeated again and the prediction is displayed accordingly. The user can further access to News updates and provide feedback accordingly about the web page.

COLLABORATION DIAGRAM



First, user enters the Home Page and there he/ she can go to rainfall home page or flood home page as per the requirement. If he/she choose rainfall home page then they need to enter certain details or requirement and then details about rainfall is displayed on the screen.

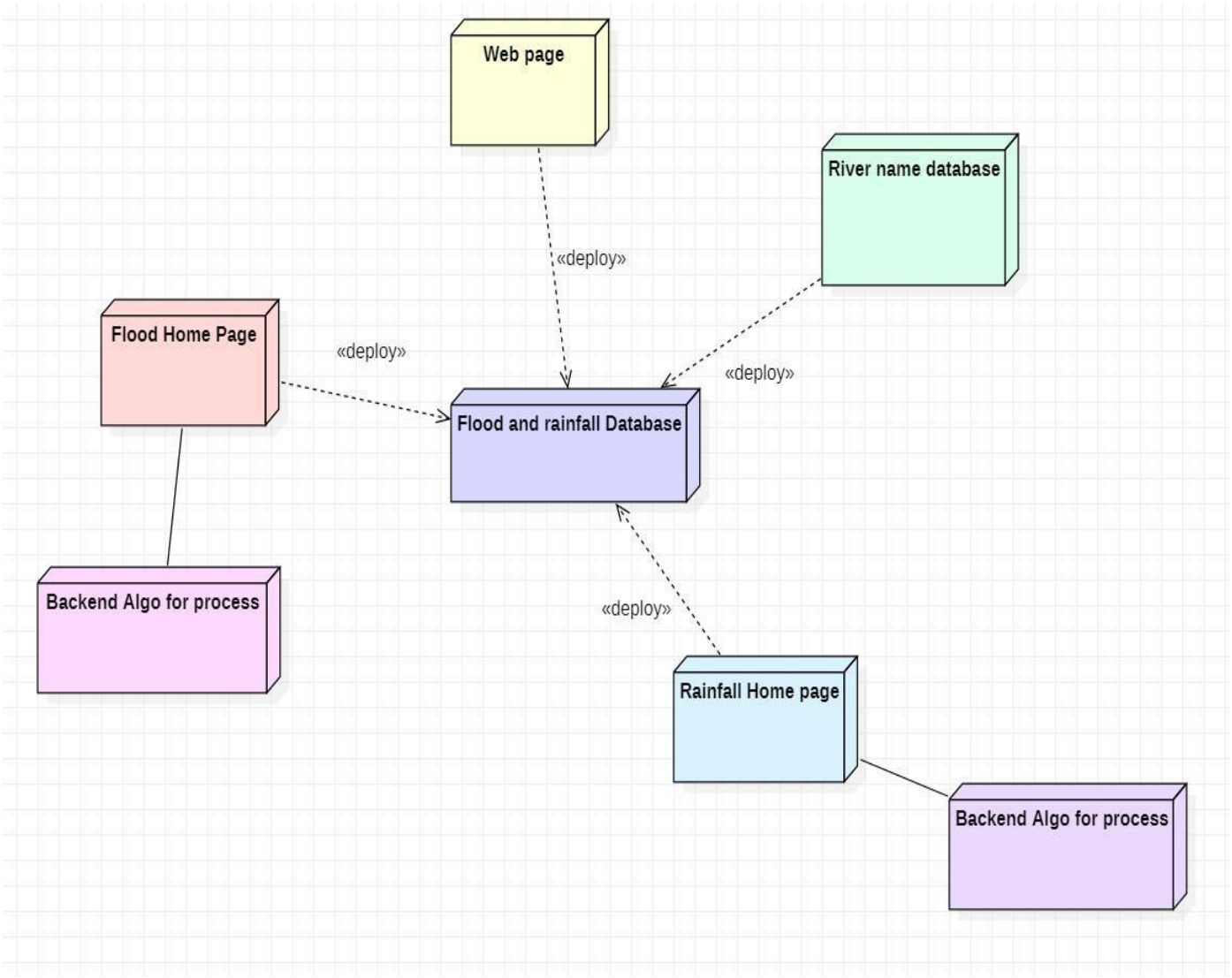
If he/she choose flood home page then they need to make a decision whether to extract the data of past date or future data, if they go with past date then they need to enter date and year and then data is sent to predictor which displays actual features of river and if she/ he choose future date then they need to enter input date and year and data is sent to analyzer who further displays the predicted features of a river based on past data and conditions.

Result: Thus, the sequence and collaboration diagrams were created.

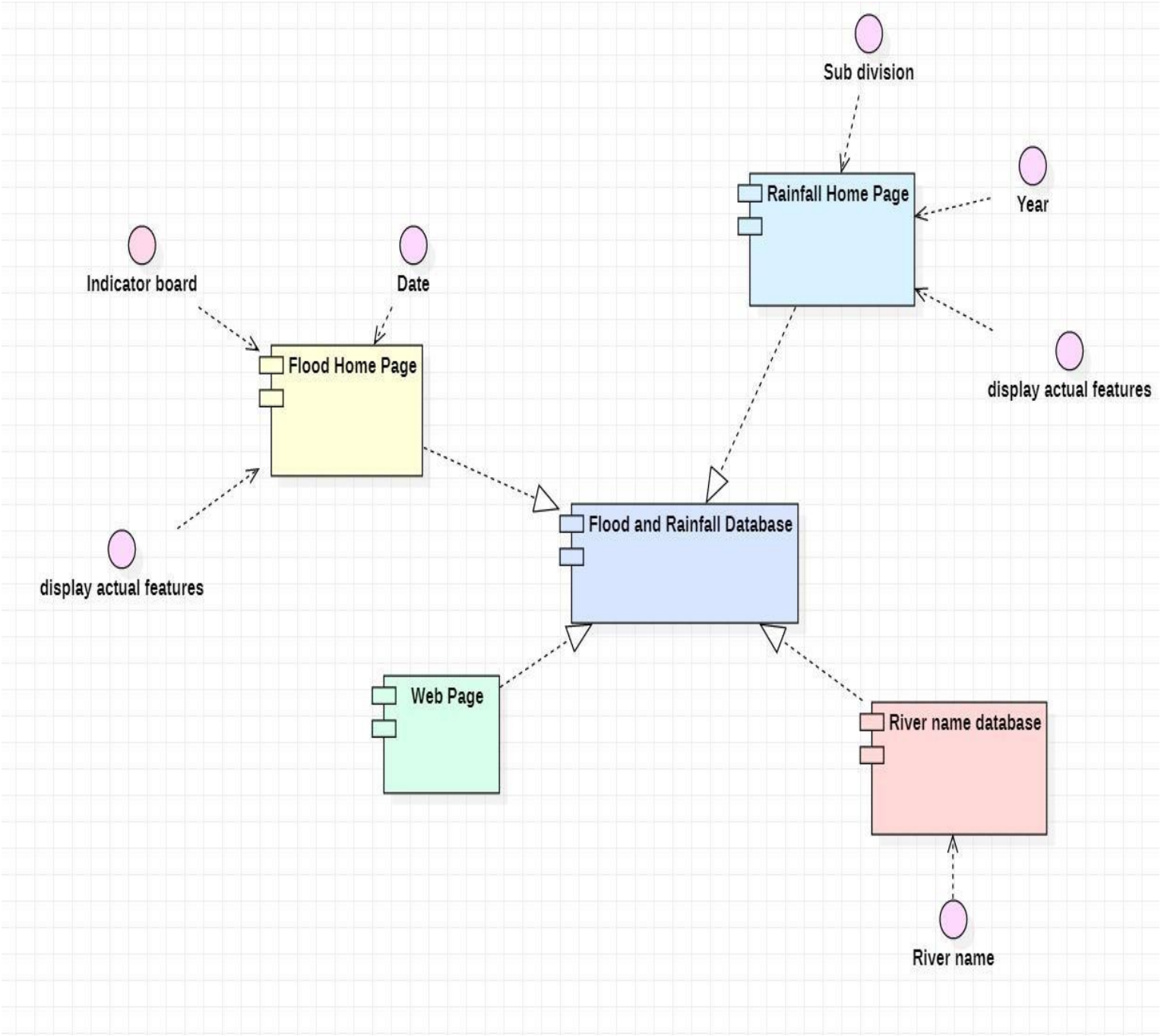
Aim

To develop the Deployment, Component and flow diagram.

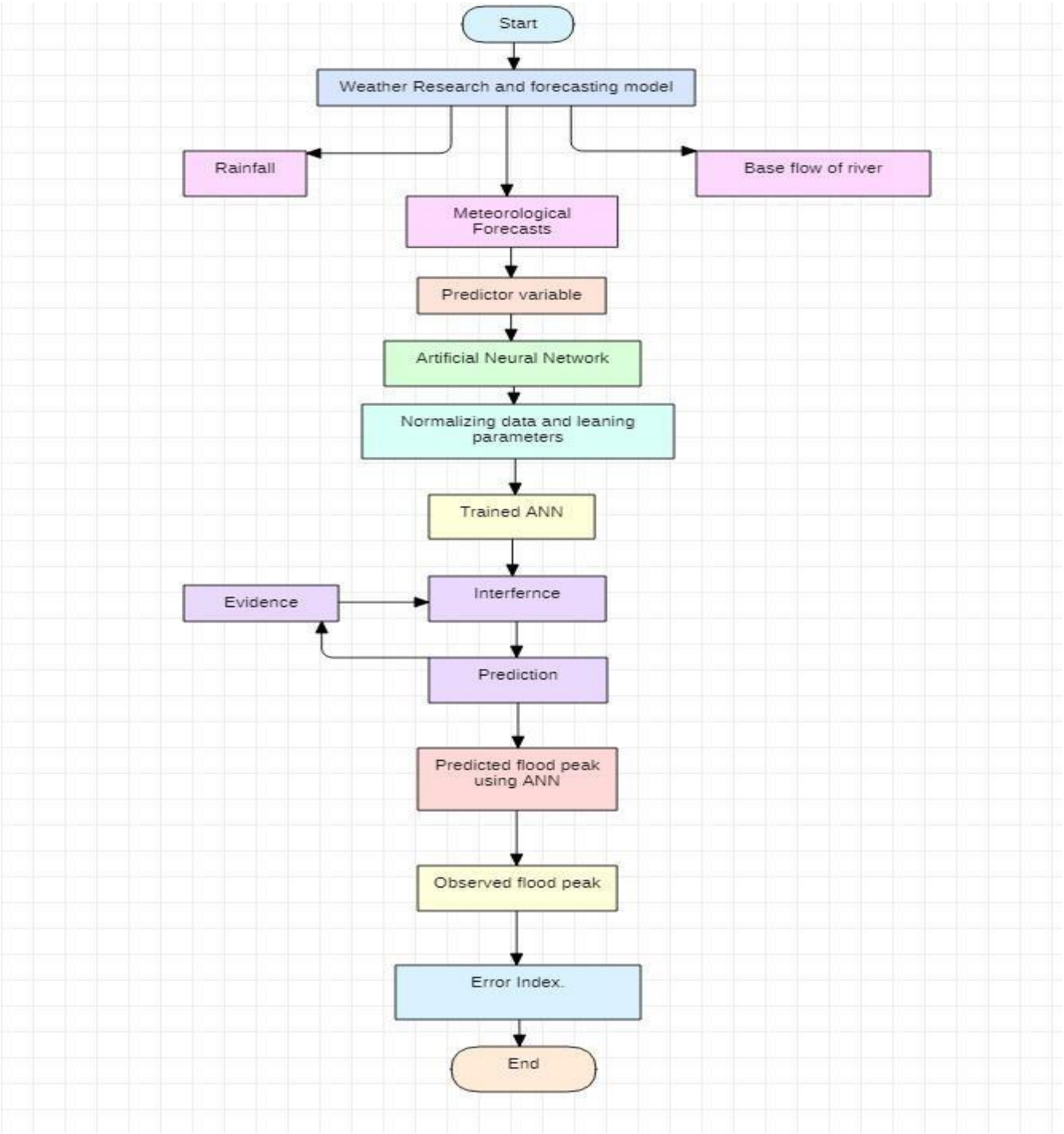
DEPLOYMENT DIAGRAM



COMPONENT DIAGRAM



FLOW DIAGRAM



Result: Thus, the entity relationship diagram was created successfully.

Aim

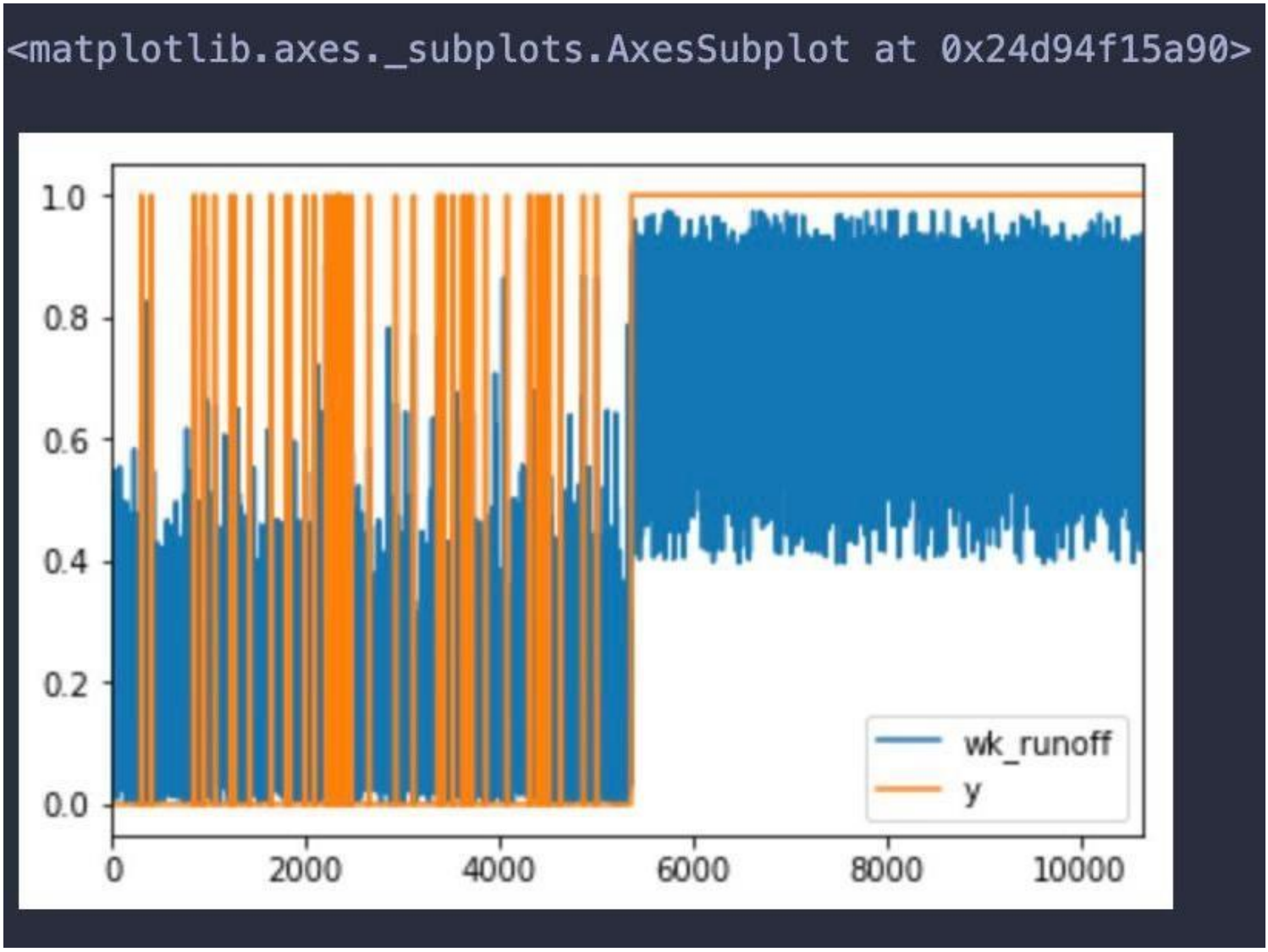
To provide the details of architectural design/framework/implementation

IMPLEMENTATION

MODULE IMPLEMENTATION:

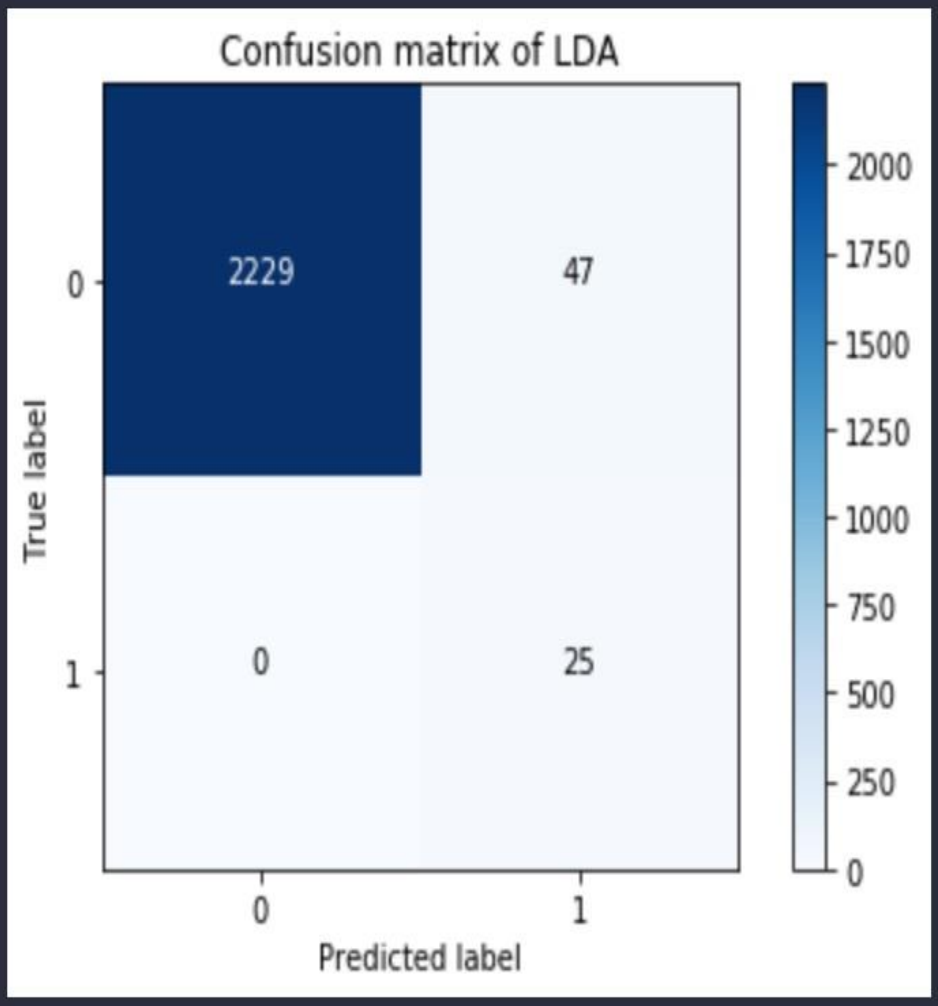
SIGN-UP MODULE OR HOME PAGE:

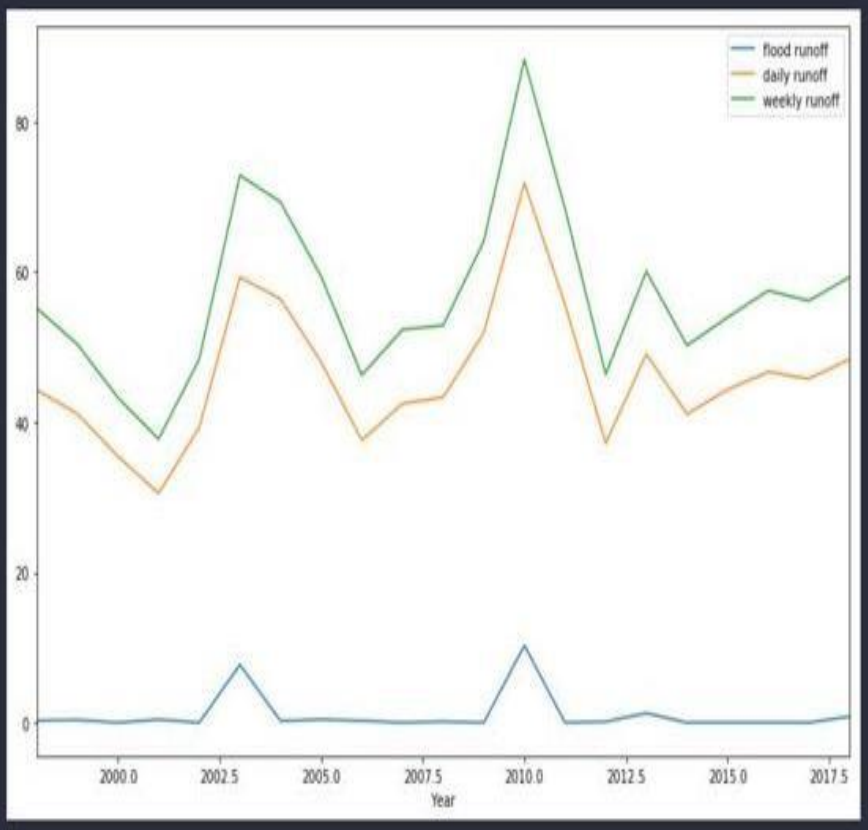
RAINFALL AND FLOOD PREDICTION MODULE



LABELLED DATA VS CONFUSION MATRIC OF LDA:

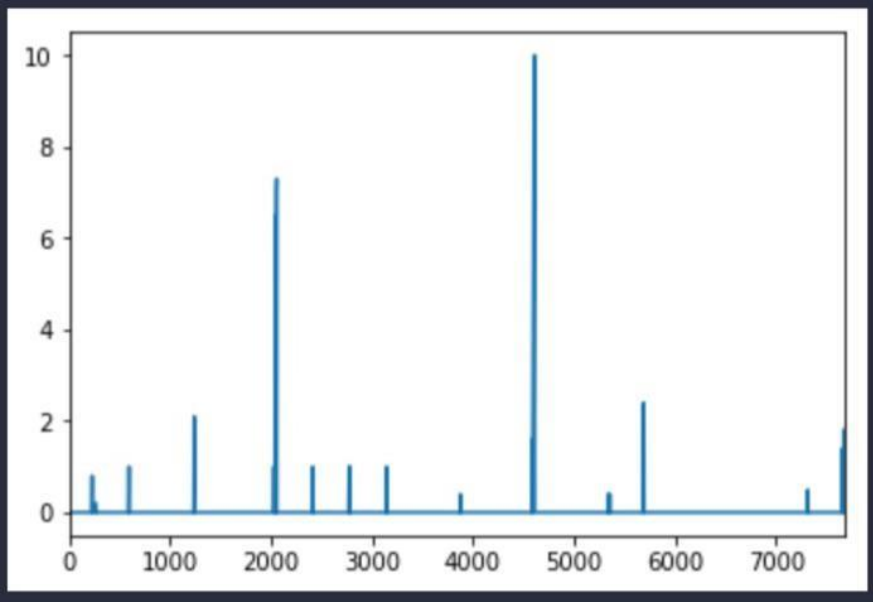
```
...  Recall metric in the train dataset: 100.0%
```

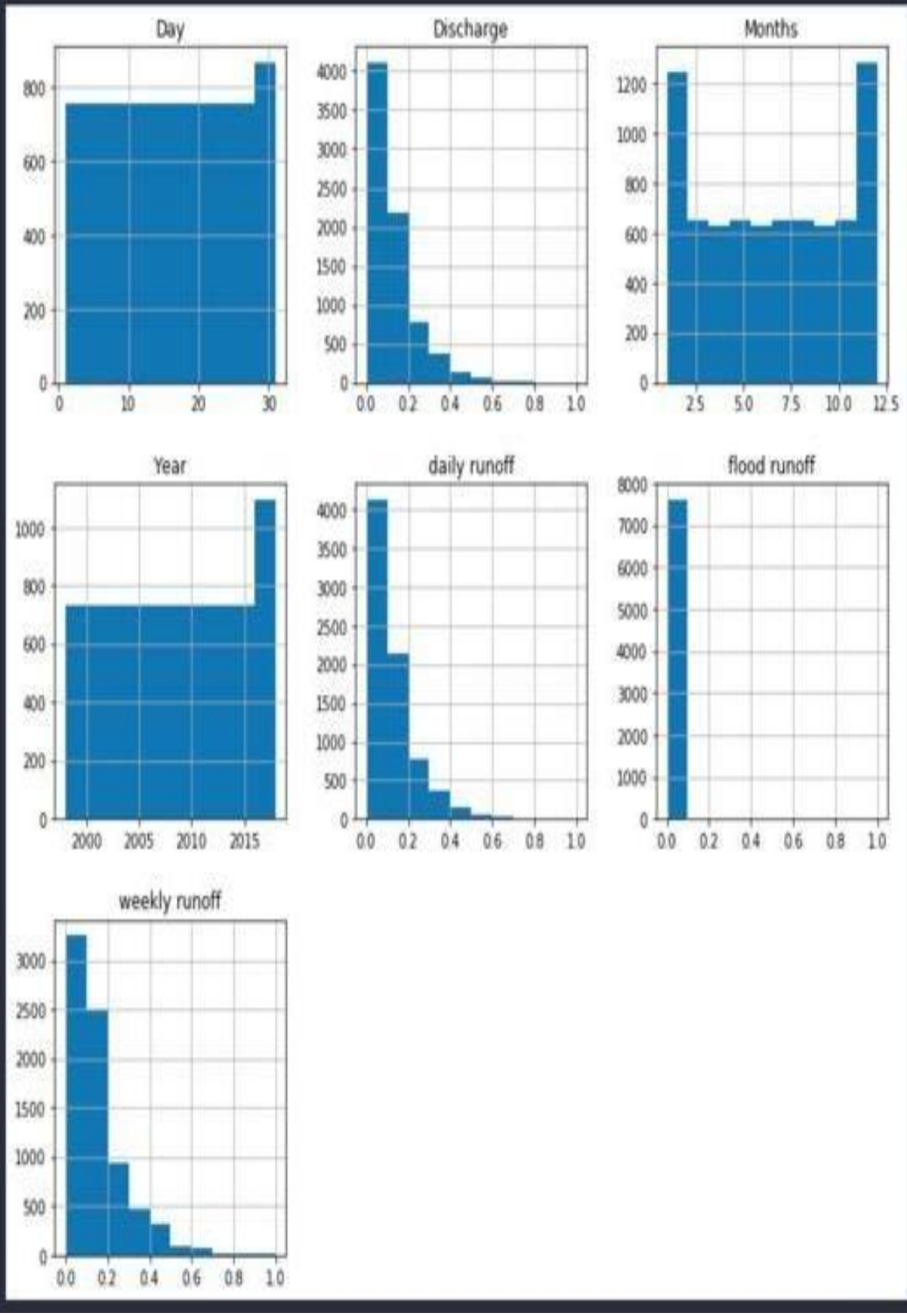





```
...   <matplotlib.axes._subplots.AxesSubplot at 0x24d92e3c860>
```

</>




Day
Discharge
Months
Year
daily runoff
flood runoff
weekly runoff

LOGIN OR HOMPAGE MODULE

HTML,CSS CODE:

```
<!DOCTYPE html>
<html>
<head>
   <title>Login Form</title>
   <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
   <h2>Login Page</h2><br>
   <div class="login">
   <form id="login" method="get" action="login.php">
      <label><b>User Name
      </b>
      </label>
      <input type="text" name="Uname" id="Uname" placeholder="Username">
      <br><br>
      <label><b>Password
      </b>
      </label>
      <input type="Password" name="Pass" id="Pass" placeholder="Password">
      <br><br>
      <input type="button" name="log" id="log" value="Log In Here">
      <br><br>
      <input type="checkbox" id="check">
      <span>Remember me</span>
      <br><br>
      <a href="#">Forgot Password</a>
   </form>
</div>
</body>
</html>
body
{
   margin: 0;
   padding: 0;
   background: url("bg.jpeg");
   background-repeat: no-repeat;
   background-size: cover;
   font-family: 'Arial';
}
.login{
      width: 382px;
      overflow: hidden;
```

```
        margin: auto;
        margin: 20 0 0 450px;
        padding: 80px;
        background: #23463f;
        border-radius: 15px ;

}
h2{
    text-align: center;
    color: #1f423b;
    padding: 20px;
}
label{
    color: #08ffd1;
    font-size: 17px;
}
#Uname{
    width: 300px;
    height: 30px;
    border: none;
    border-radius: 3px;
    padding-left: 8px;
}
#Pass{
    width: 300px;
    height: 30px;
    border: none;
    border-radius: 3px;
    padding-left: 8px;

}
#log{
    width: 300px;
    height: 30px;
    border: none;
    border-radius: 17px;
    padding-left: 7px;
    color: blue;


}
span{
    color: white;
    font-size: 17px;
}
a{
```

```
    float: left;
    color: white;
    background-color: #23463f;
}
```

FOR RAINFALL AND FLOOD PREDICTION:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import keras
#import mpld3

# ## Types of graphs
# - Bar graphs showing distribution of amount of rainfall.
# - Distribution of amount of rainfall yearly, monthly, groups of months.
# - Distribution of rainfall in subdivisions, districts form each month, groups of months.
# - Heat maps showing correlation between amount of rainfall between months.
#

# In[2]:
def rainfall(year,region):
    data = pd.read_csv('data/Sub_Division_IMD_2017.csv')
    data = data.fillna(data.mean())
    data.info()


    # In[3]:

    data.head()


    # In[4]:

    data.describe()


    # In[5]:

    # data.hist(figsize=(12,12));


    # ## Observations
    # - Above histograms show the distribution of rainfall over months.
    # - Observed increase in amount of rainfall over months July, August, September.
```

```
# In[6]:

# data[['YEAR', 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',
#       'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].groupby("YEAR").sum().plot(figsize=(13,8));


# In[7]:

# data[['YEAR','JF', 'MAM',
#       'JJAS', 'OND']].groupby("YEAR").sum().plot(figsize=(13,8));


# ## Observations
# - The above two graphs show the distribution of rainfall over months.
# - The graphs clearly shows that amount of rainfall in high in the months    oda, aug, sep which is monsoon season in India.

# In[8]:

# data[['SUBDIVISION', 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',
#       'AUG', 'SEP', 'OCT', 'NOV',
'DEC']].groupby("SUBDIVISION").mean().plot.barh(stacked=True,figsize=(13,8));


# In[9]:

# data[['SUBDIVISION', 'JF', 'MAM',
#       'JJAS', 'OND']].groupby("SUBDIVISION").sum().plot.barh(stacked=True,figsize=(16,8));


# In[10]:

# plt.figure(figsize=(11,4))
# sns.heatmap(data[['JF', 'MAM',
#       'JJAS', 'OND','ANNUAL']].corr(),annot=True)
# plt.show()


# In[11]:

# plt.figure(figsize=(11,4))
#
sns.heatmap(data[['JAN','FEB','MAR','APR','MAY','JUN','JUL','AUG','SEP','OCT','NOV','DE
C','ANNUAL']].corr(),annot=True)
# plt.show()
```

```
# ## Observations
# - **Heat Map** shows the co-relation(dependency)  odavar the amounts of rainfall over months.
# - From above it is clear that if amount of rainfall is high in the months of  oda, august,
odavari  then the amount of rainfall will be high annually.
# - It is also obwserved that if amount of rainfall in good in the months of  odavar,  odavari,
odavari then the rainfall is going to b good in the overall year.

# In[17]:

#Function to plot the graphs
def plot_graphs(groundtruth,prediction,title):
    N = 9
    ind = np.arange(N) # the x locations for the groups
    width = 0.27       # the width of the bars

    fig = plt.figure(figsize=(18,10))
    fig.suptitle(title, fontsize=12)
    ax = fig.add_subplot(111)
    rects1 = ax.bar(ind, groundtruth, width, color='m')
    rects2 = ax.bar(ind+width, prediction, width, color='c')

    ax.set_ylabel("Amount of rainfall")
    ax.set_xticks(ind+width)
    ax.set_xticklabels( ('APR', 'MAY', 'JUN', 'JUL','AUG', 'SEP', 'OCT', 'NOV', 'DEC') )
    ax.legend( (rects1[0], rects2[0]), ('Ground truth', 'Prediction') )

#     autolabel(rects1)
    for rect in rects1:
        h = rect.get_height()
        ax.text(rect.get_x()+rect.get_width()/2., 1.05*h, '%d'%int(h),
                ha='center', va='bottom')
    for rect in rects2:
        h = rect.get_height()
        ax.text(rect.get_x()+rect.get_width()/2., 1.05*h, '%d'%int(h),
                ha='center', va='bottom')
#     autolabel(rects2)

    #plt.show()
    #mpld3.save_html(fig,'static/img/rainfall.html')
    plt.savefig('static/img/rainfall.png')


def data_generation(year,region):
    temp = data[['SUBDIVISION','JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',
```

```
    'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].loc[data['YEAR'] == year]
  data_year = np.asarray(temp[['JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',
    'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].loc[temp['SUBDIVISION'] == region])
  X_year = None; y_year = None
  for I in range(data_year.shape[1]-3):
    if X_year is None:
      X_year = data_year[:, i:i+3]
      y_year = data_year[:, i+3]
    else:
      X_year = np.concatenate((X_year, data_year[:, i:i+3]), axis=0)
      y_year = np.concatenate((y_year, data_year[:, i+3]), axis=0)

  return X_year,y_year

def data_generation2(region):
  Kerala = np.asarray(data[['JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',
      'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].loc[data['SUBDIVISION'] == region])

  X = None; y = None
  for I in range(Kerala.shape[1]-3):
    if X is None:
      X = Kerala[:, i:i+3]
      y = Kerala[:, i+3]
    else:
      X = np.concatenate((X, Kerala[:, i:i+3]), axis=0)
      y = np.concatenate((y, Kerala[:, i+3]), axis=0)


  return X,y

def prediction2(year,region):
  from keras.models import Model
  from keras.layers import Dense, Input, Conv1D, Flatten

  # NN model
  inputs = Input(shape=(3,1))
  x = Conv1D(64, 2, padding='same', activation='elu')(inputs)
  x = Conv1D(128, 2, padding='same', activation='elu')(x)
  x = Flatten()(x)
  x = Dense(128, activation='elu')(x)
  x = Dense(64, activation='elu')(x)
  x = Dense(32, activation='elu')(x)
  x = Dense(1, activation='linear')(x)
  model = Model(inputs=[inputs], outputs=[x])
  model.compile(loss='mean_squared_error', optimizer='adamax', metrics=['mae'])
  X_testing,Y_testing = data_generation(year,region)
```

```
        from sklearn.metrics import mean_absolute_error
        from sklearn.metrics import explained_variance_score
        # linear model

        X_train,y_train = data_generation2(region)
        model.fit(x=np.expand_dims(X_train, axis=2), y=y_train, batch_size=64, epochs=20, verbose=1,
validation_split=0.1, shuffle=True)

        y_pred = model.predict(np.expand_dims(X_testing, axis=2))
        mae=mean_absolute_error(Y_testing, y_pred)
        score=explained_variance_score(Y_testing, y_pred)
        print(mae)
        print(score)

        Y_year_pred=list(range(9))
        for I in range(9):
            Y_year_pred[i]=y_pred[i][0]
        y_pred=np.array(Y_year_pred)
        plot_graphs(Y_testing,y_pred,"Year: "+str(year)+'  Region: '+str(region))
        return mae,score

    print("############",year,type(int(year)),region,type(region),"77777777777777777777777&&
    &&&&&&&&&&&&&&&&&&&&&&&&&&&&")
     mae,score=prediction2(int(year),region)
     mae=format(round(float(mae),2))
     score=format(round(float(score),2))
     keras.backend.clear_session()
     return mae,score
     import pandas as pd
from datetime import datetime
import time
import matplotlib.pyplot as plt
from sklearn.preprocessing import Normalizer,MinMaxScaler
from imblearn.over_sampling import SMOTE
from sklearn.utils import shuffle
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import classification_report
import numpy as np
import seaborn as sns
import plotly.graph_objs as go
import plotly.plotly as py
import plotly
from sklearn.externals import joblib

import warnings
warnings.filterwarnings("ignore")
```

```
#get_ipython().run_line_magic('matplotlib', 'inline')
#fd-future data set
#validating-0 or 1 (0-tetsing ,1= future prediction)
def flood_classifier(filename,fd,validating=0):

    data1=pd.read_excel('data/'+filename+'.xlsx')

    # In[4]:
    data1.shape
    # In[5]:

    #Fillng null entries with mean of their respective columns
    for I in range(1,len(data1.columns)):
        data1[data1.columns[i]] = data1[data1.columns[i]].fillna(data1[data1.columns[i]].mean())
    # In[6]:
    data1.describe()
    # In[7]:
    y=data1['Flood']
    # In[8]:
    for I in range(len(y)):
        if(y[i] >= 0.1):
            y[i]=1
    # In[9]:

    y=pd.DataFrame(y)

    data1.drop('Flood',axis=1,inplace=True)


    # In[10]:
    data1.head()
    # In[11]:
    data1.hist(figsize=(6,6));

    #Breaking Date column into timestamp

    d1=pd.DataFrame()
    d1["Day"]=data1['Date']
    d1['Months']=data1['Date']
    d1['Year']=data1['Date']
    data1['Date']=pd.to_datetime(data1['Date'])
    d1["Year"]=data1.Date.dt.year
    d1["Months"]=data1.Date.dt.month
    d1["Day"]=data1.Date.dt.day

    #.....Resampling
```

```
#------------not working for    odava
dx=pd.DataFrame()
dx['Date']=data1['Date']
dx['Discharge']=data1['Discharge']
dx=dx.set_index(['Date'])
yearly = dx.resample('Y').sum()

plt.figure(figsize=(9,8))
plt.xlabel('YEARS')
plt.ylabel('Level')
plt.title(filename+" : Year wise Trends")
plt.plot(yearly,'—')

#plt.plot(yearly,style=[':', '—', '-'],title='Year wise Trends')
plt.savefig('static/img/flood.png')
#.....


# In[18]:
data1.drop('Date',inplace=True,axis=1)
# In[19]:


#Scaling the data in range of 0 to 1

# Scaler=MinMaxScaler(feature_range=(0, 1))
# Transform=Scaler.fit_transform(data1)
# # In[20]
# #Transform
# # In[21]:
# Transform=pd.DataFrame(Transform,columns=['Discharge','flood runoff','daily runoff','weekly runoff'])

# # In[22]:
# data1=Transform
# In[23]:
data1=pd.concat([d1,data1],axis=1)
data1.head()

#----------------------for taking data upto 2015 as training and rest for testing----------------------------
------------
locate=0;
for I in range(len(data1["Day"])):
    if(data1["Day"][i]==31 and data1["Months"][i]==12 and data1["Year"][i]==2015):
        locate=I;
        break;
```

```
i=locate+1
print(i)

x_train=data1.iloc[0:I,:]
y_train=y.iloc[0:i]
x_test=data1.iloc[i:,:]
y_test=y.iloc[i:]


# In[25]:


x_train.drop(labels=['Day','Months','Year'],inplace=True,axis=1)
x_test.drop(labels=['Day','Months','Year'],inplace=True,axis=1)


# In[26]:


# nl=Normalizer()
# x_train=nl.fit_transform(x_train)
# x_test=nl.transform(x_test)


# In[27]:


# y_train=nl.transform(y_train)
# y_test=nl.transform(y_test)


# In[28]:


#-----------------Upsampling the data (as very less entries of flood =1 is present)----------------
sm = SMOTE(random_state=2)
X_train_res, Y_train_res = sm.fit_sample(x_train, y_train)
# In[29]:
X_train_res.shape
# In[30]:
x_train, y_train = shuffle( X_train_res, Y_train_res, random_state=0)

x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

```
# In[32]:


# #----------Logistic Regression----------
# from sklearn.linear_model import LogisticRegression
# reg=LogisticRegression()
# reg.fit(x_train,y_train)
# y_predict1=reg.predict(x_test)
# print(set(y_predict1))
# print(reg.score(x_train,y_train))
# print(reg.score(x_test,y_test))
# print(classification_report(y_test, y_predict1))
# print("mean_absolute_error=",mean_absolute_error(y_test, y_predict1))


# # In[34]:

#----------LinearDiscriminantAnalysis----------
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

# clf1=LinearDiscriminantAnalysis()
# clf1.fit(x_train,y_train)

#----------saving & Loading the model----------

path='trained/'+filename+'_LDA'
#joblib.dump(clf1, path+'.pkl')
clf1= joblib.load(path+'.pkl')

#----------

y_predict3=clf1.predict(x_test)
print(set(y_predict3))
print(clf1.score(x_train,y_train))
print(clf1.score(x_test,y_test))
print(classification_report(y_test, y_predict3))
mae=mean_absolute_error(y_test, y_predict3)
print("mean_absolute_error=",mae)

# In[35]:
#----------KneighborsClassifier----------

# from sklearn.neighbors import KneighborsClassifier

# clf2=KneighborsClassifier()
# clf2.fit(x_train,y_train)
```

```
    # y_predict4=clf2.predict(x_test)
    # print(set(y_predict4))
    # print(clf2.score(x_train,y_train))
    # print(clf2.score(x_test,y_test))
    # print(classification_report(y_test, y_predict4))
    # print("mean_absolute_error=",mean_absolute_error(y_test, y_predict4))

    # # In[36]:

    #.....Testing.....
    # In[38]:
    data1.head()
    # In[39]:
    def predicting(future_data):
            # xx=[13214.0,0.0,0.36,2.08]
            #xx=[4990.0,0.0,1.40,15.38]
            xx=future_data
            xx=np.array(xx)
            xx=xx.reshape((-1, 4))
            xx=clf1.predict(xx)
            # xx=reg.predict(xx)
            return xx
    xx=predicting(fd)
    return xx,mae
#xx=predicted value of flood 0 or 1

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler,MinMaxScaler
from datetime import datetime
#data=pd.read_csv("   odavari_daily.csv")
data=pd.read_excel("data/Godavari.xlsx")
y=data['Flood']
# data.drop('Flood',axis=1,inplace=True)
set(y)
y.plot(x = data['Date'], y = y)
for I in range(len(y)):
    if y.iloc[i] >= 0.5 :
        y.iloc[i]=1.0
    else :
        y.iloc[i]=0.0
data.drop('Flood',axis=1,inplace=True)
d1=pd.DataFrame()
d1["Day"]=data['Date']
```

```
d1[‘Months’]=data[‘Date’]
d1[‘Year’]=data[‘Date’]
d1[“Year”]=data.Date.dt.year
d1[“Months”]=data.Date.dt.month
d1[“Day”]=data.Date.dt.day
d1.head()
x=pd.DataFrame(data=data[‘Day’])
x[‘Months’]=data[‘Months’]
x[‘Year’]=data[‘Year’]
x.head()
data1=pd.concat([x,data1],axis=1)
data=data1
data.head()
data[[‘Year’, ‘flood runoff’, ‘daily runoff’, ‘weekly
runoff’]].groupby(“Year”).sum().plot(figsize=(13,8));
```

TRAINING RAINFALL PREDICTION MODEL WITH DIFFERENT MODELS :

We will divide the dataset into training (75%) and test (25%) sets respectively to train the rainfall prediction model. For best results, we will standardize our X_train and X_test data:

CODE-

```
features = MiceImputed[['Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine',
'WindGustDir',
                       'WindGustSpeed', 'WindDir9am', 'WindDir3pm', 'WindSpeed9am', 'WindSpeed3pm',
'Humidity9am',
                       'Humidity3pm', 'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am',
'Temp3pm',
                       'RainToday']]
target = MiceImputed['RainTomorrow']

# Split into test and train
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.25, random_state=12345)

# Normalize Features
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.fit_transform(X_test)
def plot_roc_cur(fper, tper):
    plt.plot(fper, tper, color='orange', label='ROC')
    plt.plot([0, 1], [0, 1], color='darkblue', linestyle='--')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
```

```
    plt.title('Receiver Operating Characteristic (ROC) Curve')
    plt.legend()
    plt.show()
import time
from sklearn.metrics import accuracy_score, roc_auc_score, cohen_kappa_score,
plot_confusion_matrix, roc_curve, classification_report
def run_model(model, X_train, y_train, X_test, y_test, verbose=True):
    t0=time.time()
    if verbose == False:
        model.fit(X_train,y_train, verbose=0)
    else:
        model.fit(X_train,y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    roc_auc = roc_auc_score(y_test, y_pred)
    coh_kap = cohen_kappa_score(y_test, y_pred)
    time_taken = time.time()-t0
    print("Accuracy = {}".format(accuracy))
    print("ROC Area under Curve = {}".format(roc_auc))
    print("Cohen's Kappa = {}".format(coh_kap))
    print("Time taken = {}".format(time_taken))
    print(classification_report(y_test,y_pred,digits=5))

    probs = model.predict_proba(X_test)
    probs = probs[:, 1]
    fper, tper, thresholds = roc_curve(y_test, probs)
    plot_roc_cur(fper, tper)

    plot_confusion_matrix(model, X_test, y_test,cmap=plt.cm.Blues, normalize = 'all')

    return model, accuracy, roc_auc, coh_kap, time_taken
```

RAINFALL PREDICTION MODEL COMPARISON

METRICS FOR ALGORITHMIC COMPARISON :

The implementation of the different machine learning algorithms such as SVM, Bayes, KNN and deep neural network are compared to identify the best algorithm for the prediction of the occurrence of flood. Before the results, a brief explanation about the various metrics used for comparative analysis is given below.

Confusion matrix

This is a binary classifier. A confusion matrix can be of any size depending upon the different number of parameters inputted (labels in our case). The confusion matrix in our case is a 2 × 2 matrix. where TP = true positive; FN = false negative; FP = false positive; TN = true negative. TP and TN denote the number of instances which have been correctly classified as no flood occurrence and flood occurrence respectively. FP and FN signify the number of instances which have been wrongly classified as no flood occurrence and flood occurrence respectively.

Precision and recall

The success of prediction is computed by means of precision-recall where classes are imbalanced. The relevancy of the result is expressed as precision whereas the number of true relevant results returned is expressed as recall. A low false positive rate and low false negative rate result is related to high precision and high recall respectively.

Precision: Termed as the positive predictive value is calculated as given below

$$Precision = \frac{TP}{TP + FP}$$

Recall: Also called as sensitivity is calculated as given below.

$$Recall = \frac{TP}{TP + FN}$$

Accuracy

Based on the confusion matrix created, accuracy computed as an accuracy score function which is either by default the fraction or the count (normalize = false) of correct predictions. Accuracy is calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

F1 score

This is a measure of the test's accuracy where it considers both precision and recall. This is the harmonic average of precision and recall where F1 reaches its best value at 1 and worst at 0. The F1 score conveys the balance between precision and recall:

$$F1\ score = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

MCC

Matthews correlation coefficient considers all four divisions of the confusion matrix when calculated. MCC lies within the range −1 to +1 where a model with a positive score is considered to be perfect whereas the negative score is poor. This makes this metric really useful as it is easy to interpret (Gereon 2018):

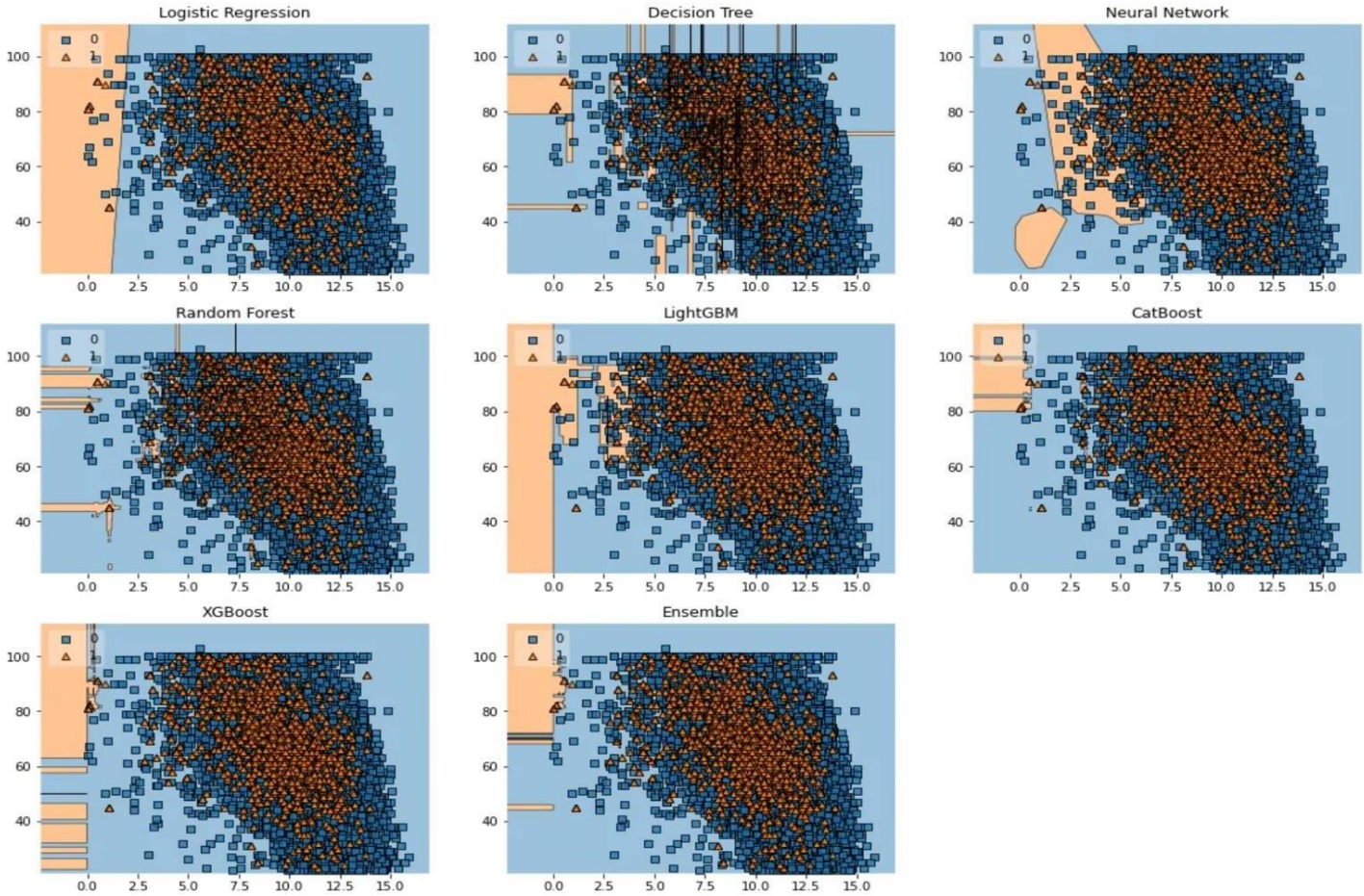
$$MCC = \frac{TP*\mathrm{TN} - \mathrm{FP}*\mathrm{FN}}{\sqrt{(TP + FP) * (\mathrm{TP} + \mathrm{FN}) * (\mathrm{TN} + \mathrm{FP}) * (\mathrm{TN} + \mathrm{FN})}}$$

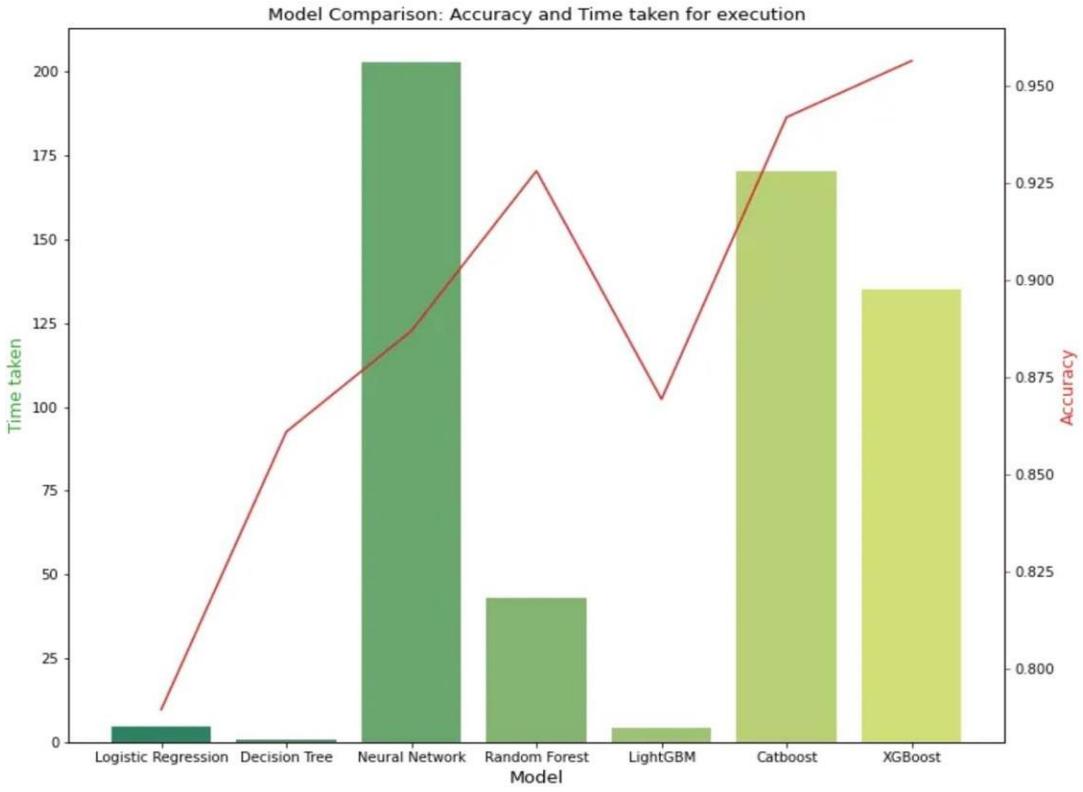
CODE-

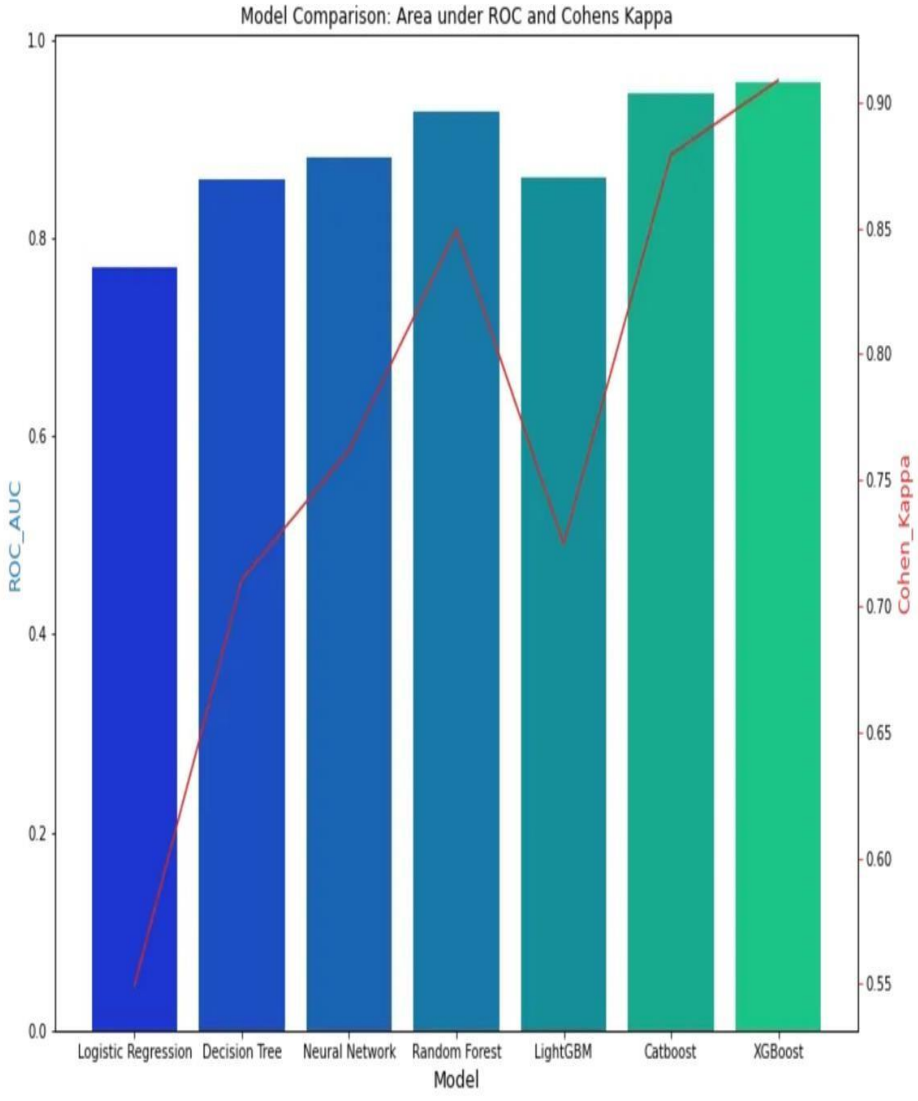
```
Accuracy_scores = [accuracy_lr, accuracy_dt, accuracy_nn, accuracy_rf, accuracy_lgb, accuracy_cb, accuracy_xgb]
roc_auc_scores = [roc_auc_lr, roc_auc_dt, roc_auc_nn, roc_auc_rf, roc_auc_lgb, roc_auc_cb, roc_auc_xgb]
coh_kap_scores = [coh_kap_lr, coh_kap_dt, coh_kap_nn, coh_kap_rf, coh_kap_lgb, coh_kap_cb, coh_kap_xgb]
tt = [tt_lr, tt_dt, tt_nn, tt_rf, tt_lgb, tt_cb, tt_xgb]

model_data = {'Model': ['Logistic Regression','Decision Tree','Neural Network','Random Forest','LightGBM','Catboost','XGBoost'],
        'Accuracy': accuracy_scores,
        'ROC_AUC': roc_auc_scores,
        'Cohen_Kappa': coh_kap_scores,
        'Time taken': tt}
data = pd.DataFrame(model_data)

fig, ax1 = plt.subplots(figsize=(12,10))
ax1.set_title('Model Comparison: Accuracy and Time taken for execution', fontsize=13)
color = 'tab:green'
ax1.set_xlabel('Model', fontsize=13)
ax1.set_ylabel('Time taken', fontsize=13, color=color)
ax2 = sns.barplot(x='Model', y='Time taken', data = data, palette='summer')
ax1.tick_params(axis='y')
ax2 = ax1.twinx()
color = 'tab:red'
ax2.set_ylabel('Accuracy', fontsize=13, color=color)
ax2 = sns.lineplot(x='Model', y='Accuracy', data = data, sort=False, color=color)
ax2.tick_params(axis='y', color=color)
Code language: PHP (php)
   figsize=(12,10))
ax3.set_title('Model Comparison: Area under ROC and Cohens Kappa', fontsize=13)
color = 'tab:blue'
ax3.set_xlabel('Model', fontsize=13)
ax3.set_ylabel('ROC_AUC', fontsize=13, color=color)
ax4 = sns.barplot(x='Model', y='ROC_AUC', data = data, palette='winter')
ax3.tick_params(axis='y')
```

```
ax4 = ax3.twinx()
color = 'tab:red'
ax4.set_ylabel('Cohen_Kappa', fontsize=13, color=color)
ax4 = sns.lineplot(x='Model', y='Cohen_Kappa', data = data, sort=False, color=color)
ax4.tick_params(axis='y', color=color)
plt.show()
```

RAINFALL PREDICTION MODEL COMPARISON:




Model Comparison: Accuracy and Time taken for execution
Time taken
Accuracy
Model
Logistic Regression
Decision Tree
Neural Network
Random Forest
LightGBM
Catboost
XGBoost
0
25
50
75
100
125
150
175
200
0.800
0.825
0.850
0.875
0.900
0.925
0.950


Model Comparison: Area under ROC and Cohens Kappa
ROC_AUC
Cohen_Kappa
Model
Logistic Regression
Decision Tree
Neural Network
Random Forest
LightGBM
Catboost
XGBoost
0.0
0.2
0.4
0.6
0.8
1.0
0.55
0.60
0.65
0.70
0.75
0.80
0.85
0.90

OBSERVATIONS :

We can observe that XGBoost, CatBoost and Random Forest performed better compared to other models. However, if speed is an important thing to consider, we can stick with Random Forest instead of XGBoost or CatBoost.

RESULT :

The main focus of this work is on developing an early flood warning system using a deep neural network. Python's advanced built in data structures, linked with dynamic composing and dynamic binding to associate existing components together, are used. Some of the libraries in Python used for our work are Keras, Pandas and Numpy. A database is created containing the flood rainfall data. The reason for choosing python for implementing machine learning is that Python has many libraries for every need of your AI project. A few names include ‘Numpy’ for scientific computation, ‘Scipy’ for advanced computing and ‘Keras’ for machine learning. Python is a completely open source with a great community. There is a host of resources available which can get any developer up to speed in no time.

CONCLUSION :

To overcome issues with best model of Flood and rainfall Prediction, we have developed this website. The development process for this website is characterized by the efforts made by whole team and it also requires lots hardware and software infrastructures.

The hardware may include PC with core i7 processor, 16 GB RAM and Graphic Card at least MX150, and software like PYTHON,HTML, CSS, JAVA- SCRIPT Etc and Android Studio.

The website is set to meet all the requirements given by the user and looks forward for the essential use of the software. The integrated efforts not only comprise of design and realization of interface but also intense testing of the product.

FUTURE WORK :

India contributes one-fifth of the world's global deaths due to floods and is the world's worst affected country after Bangladesh. The Indian rainfall season is mainly from June to September and accounts for nearly 75% of the total Indian rainfall per year. Much work has been carried out by employing machine learning algorithms such as ANN for flood prediction. Most of the systems employed ANN with a single hidden layer for prediction of flood with parameters such as rainfall, temperature, water flow, water level and humidity. One system was developed using a deep neural network where stream flow was taken into consideration for the prediction of flood. The challenge in all these system is that most used traditional ANN and with the advent of the deep neural network, we can predict the possibility of flood occurrence with higher accuracy based on rainfall and temperature.
This research work employed a deep neural network for forecasting the prediction of occurrence of flood based on temperature and rainfall. Based on the prediction of probability of flood occurrence, an alert can be provided for evacuation which can save human lives and property. The dataset consisted of large amounts of observed rainfall data and collecting factors such as

minimum and maximum temperature and flood occurrence. On comparing the accuracy obtained from four different algorithms, the results indicate that the deep neural network can be efficiently used for flood forecasting. The work carried out here clearly shows that the DNN provides better accuracy when compared with other algorithms such as SVM, KNN and Naïve Bayes with a precision of 89.71%.

The limitation of the research work is that we have a dataset from 1990 to 2002 only for validating the machine learning model towards flood prediction. The most recent dataset is not available which could result in better accuracy than achieved, which is not a major issue.

Many aspects from the future work can be tackled from this research. First, data distribution imbalance was not considered. With the advancement of neural networks for prediction, further work can be implemented by considering a most recent dataset which could achieve higher accuracy. Second, other topographical factors including flood water level can be considered in order to predict floods more precisely for different cities/states in India. Last, by including standard machine learning paradigms, the results obtained open the field for further research and development of new methods.

REFERENCES :

Research paper
Flood prediction based on weather parameters using deep learning | Journal of Water and Climate Change | IWA Publishing (iwaponline.com)

Meteorological Data
https://mausam.imd.gov.in/imd_latest/contents/rainfallinformation.php

Building Deep Learning Model using Keras

Retrieved from: https://towardsdatascience.com/building-a-deep-learning-model-using-keras-1548ca149d37

.