CHAPTER 5

SYSTEM DESIGN

The purpose of the design phase is to plan a solution of the problem specified by the requirement document. The design of a system is perhaps the most critical factor affecting the quality of the software, and has a major impact on the later phases, particularly testing and maintenance. The output of this phase is the design document. The design activity is often divided into two separate phases. They are system design and detailed design.

5.1 High Level Design

High-level design which is sometimes also called system design, aims to identify the modules that should be in the system, the specifications of these modules, and how they interact with each other to produce the desired results. At the end of system design all the major data structures, file formats, output formats, as well as the major modules in the system and their specifications are decided.

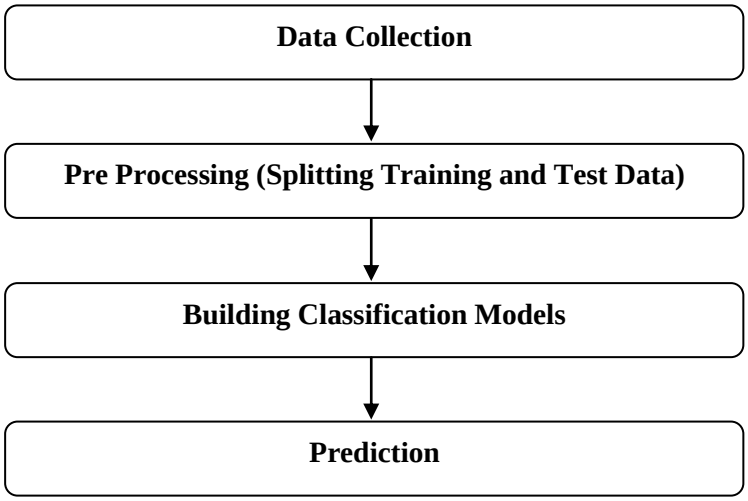

Figure 5.1: *Process Flow Diagram*

Figure 5.1 shows the process flow of Flood Prediction using AI. To get the required data, the system uses a request module which requests for the data from a repository using an API. The API will provide the data in a JSON format which is then converted to CSV format to be processed by the pandas library. The data is then cleaned up, such that any information that is not relevant, missing values, duplicate values etc. are ignored and only the relevant data is used for the next step which is splitting the data into two parts, training data and testing data. Here a ratio of 80:20 is used, where 80% of the

data is used for training the model and 20% of the data is used for testing purpose. Then we use different algorithms to prepare a model for the given data. The models are then used to predict the probability of the flooding in a given region for a given month.

5.2 Detailed Design

During detailed design the internal logic of each of the modules specified in system design is decided. During this phase further details of the data structures and algorithmic design of each of the modules is specified. The logic of module is usually specified in a high-level design description language, which is independent of target language in which the software will eventually be implemented.

5.2.1 Use Case Diagram of Fruit Recognition using Image Processing

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. A use case diagram is a dynamic or behaviour diagram in UML. The use cases are represented by either circles or ellipses. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. Use case diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities. They also help identify any internal or external factors that may influence the system and should be taken into consideration.

There is only one actor that is user in the proposed system of Flood Prediction using AI. The user enters the desired region and time for the prediction of flood. After reading this data, the system will load these as parameters for pre-processing where only the data regarding the user’s choice is selected from the dataset and is used for processing. This is to increase the efficiency of the system such that it will use only the data which is relevant to the particular use case and nothing else. The data which is now pre-processed to remove irrelevant fields is then passed to each of the four algorithms, K – Nearest Neighbour, Logistic Regression, Support Vector Machine and Decision Tree. Each algorithm will output its prediction values along with the accuracy of the prediction.

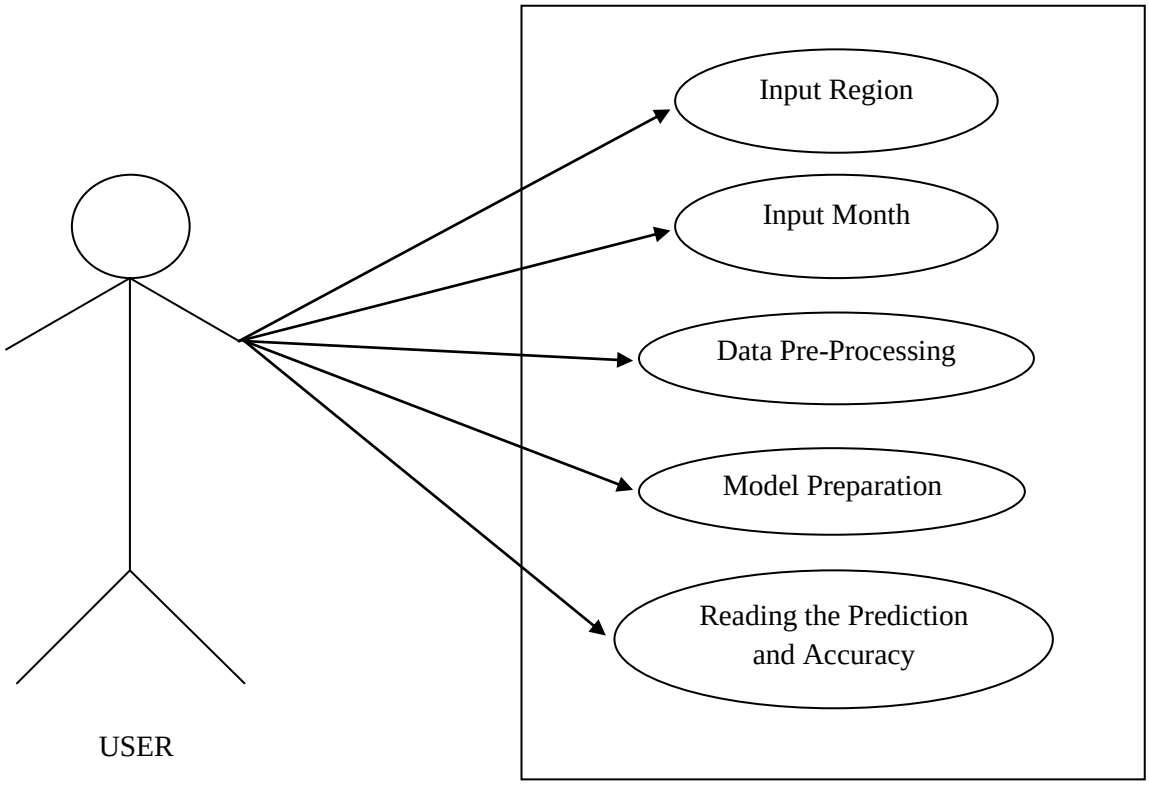

Figure 5.2: *Use Case Diagram for Floor Prediction using AI*

5.2.2 Data Flow Diagram of Fruit Recognition using Image Processing

A Data Flow Diagram (DFD) is a graph showing flow of data values from their sources in objects through processes that transform them to destination in other objects. A DFD also known as “bubble chart” has the purpose of clarifying the system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design phase that functionally decomposes the requirements specifications down to the lowest level of detail. The bubbles represent data transformations and the lines represent data flows in the system. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated.

Figure 5.3 shows the data flow diagram for Flood Prediction using AI. User can give input in the form of region and time of year. These values are captured and are passed to the system where it will pick the data related to the chosen region and time of year. Data is extracted from the dataset and is passed to the models for prediction. The algorithms take the input data and predict the output values which are then compared with the test data for validation and to measure accuracy of the algorithm. The predication and the accuracy are then shown to the user as output.

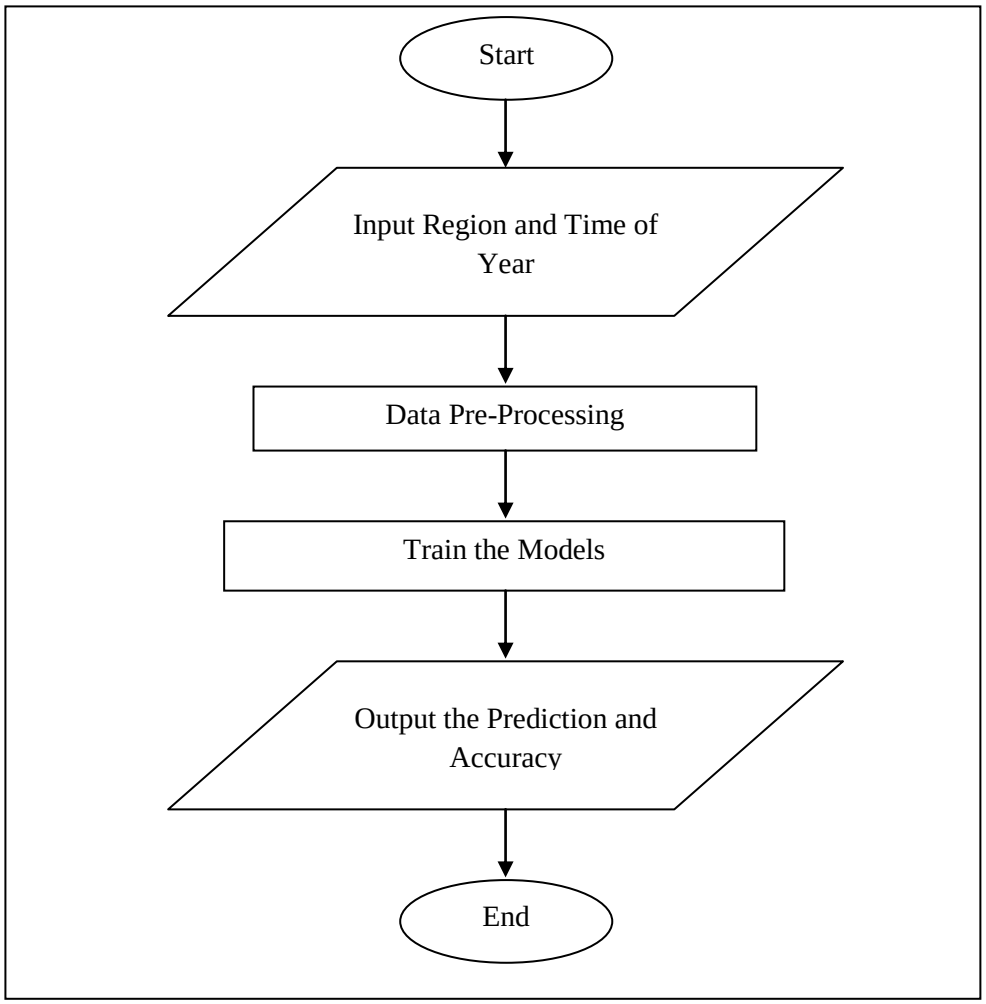

Figure 5.3: *Data Flow Diagram for Flood Predication using AI*