Logo

Description automatically generated with low confidence**Software Engineering**

**Assignment #4**

**Iteration 3: Execution Phase (OUTLINING SPRINT)**

**Name : Eman Furrukh**

**Roll No : 21i-1726**

**Section : BSDS-U**

**2. Discuss the compete Test policy of your ML based SE project**

The testing policy of this project is carefully designed to ensure that the model is performed accurately and reliably. Here are some general guidelines for this project:

->Data splitting: It is important to split the available data into training, validation, and test sets. The model should be trained on the training set, hyperparameters should be tuned on the validation set, and the final performance of the model should be evaluated on the test set. The split should be done randomly to avoid any biases.

->Metrics selection: The metrics for evaluating the model should be chosen carefully. Metrics such as accuracy, precision, recall, and F1-score can be used for classification tasks, while metrics such as mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE) can be used for regression tasks.

->Cross-validation: To ensure the reliability of the model, cross-validation should be performed.

->Data augmentation: Data augmentation can be used to increase the size of the dataset and improve the model's performance.

->Testing automation: Testing should be automated to ensure that the same tests are executed repeatedly. This helps to detect any changes in the model's performance over time. Continuous integration (CI) and continuous delivery (CD) pipelines can be used to automate testing.

->Human validation: Finally, human validation should be performed to ensure that the model's output makes sense. Domain experts can be used to validate the model's output and provide feedback on its performance.

**3. Design and discuss the higher order test cases of integration testing. (Note: keep software testing principles in your mind)**

Integration testing is a crucial step in the software development life cycle where individual software components are combined and tested as a group to ensure that they work together as expected. Here are some examples of higher order test cases for integration testing for this project:

End-to-End Testing: This type of testing verifies the complete flow of the application from start to finish. It involves testing all the components of the system, including hardware, software, and network components, to ensure that they are working as expected.

Performance Testing: This type of testing is performed to ensure that the system can handle a large volume of users and data without any performance issues. It is important to perform performance testing on all the components of the system to ensure that they can handle the expected load.

Data Integrity Testing: This type of testing verifies that the data is being properly transferred between the different components of the system without any loss or corruption. It is important to test the data transfer between the different modules of the system and ensure that the data is being stored correctly in the database.

Security Testing: This type of testing is performed to ensure that the system is secure from various threats such as hacking, phishing, and other security attacks.

Usability Testing: This type of testing is performed to ensure that the system is easy to use and user-friendly. Usability testing includes testing the user interface, navigation, and overall user experience of the system.

Compatibility Testing: This type of testing is performed to ensure that the system is compatible with different hardware, software, and network configurations. Compatibility testing includes testing the system on different browsers, operating systems, and devices to ensure that the system works as expected in all environments.

**4. Write the detail and comprehensive report of your all above UML models**

Diagram, schematic

Description automatically generated**Use Case Diagram:**

In this Use Case Diagram, we have 3 actors: User, Data Source, and System. The data source is used in order to input the data into the dataset. The user clicks a button in order to input the data source and to generate the prediction. The system consists of a generation of input and output.

Diagram

Description automatically generated**System Sequence Diagram:**

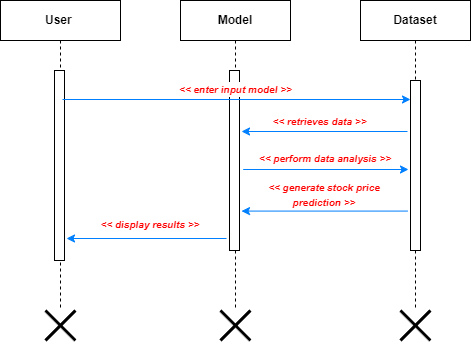
In this System Sequence Diagram, the user inputs data and the system collects the data from the data source and stores it. The system then preprocesses the data, separates it in order to train and test it, applies ML algorithms on it which then produces prediction for the dataset. The system then displays the prediction i.e., the prices to the user who can then give feedback.

**Data Association Diagram:**

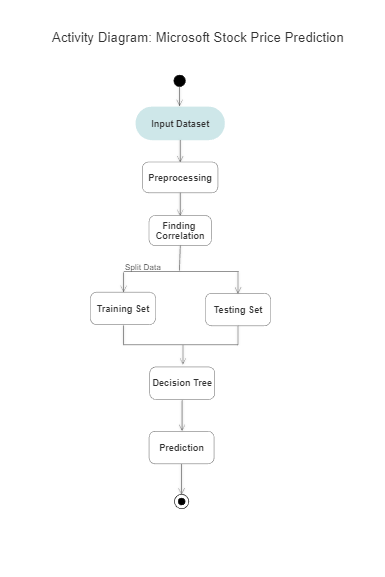
Diagram

Description automatically generatedThe Microsoft Stock Price Prediction system has four main data entities: Stock, Model, Historical Data, and Real-time Data. Each of these entities is represented by a box, and the relationships between them are shown by the lines connecting the boxes.

The arrows indicate the direction of the relationship between entities. For example, a Stock can have historical data associated with it, which can be used to train the predictive Model. The Model can then be used to make predictions on real-time data.

**Sequence Diagram:**

In this system sequence diagram, the user enters their input data into the web application. The web application retrieves the necessary data from the database, performs data analysis, and generates a stock price prediction. The web application then displays the prediction results to the user. The arrows represent the flow of information and actions between the user, the web application, and the database.

**Activity Diagram:**

In this activity diagram, the User Interface collects data from the user when they input the dataset, preprocesses the data, finds the correlation between variables, trains’ and tests the model, and then uses the model to make predictions on new data. Finally, the results are displayed to the user.