Student: Balaciu Darius Alexandru

PROJECT for SOFTWARE ENGINEERING LABORATORY

CS2 Skin Market Analyzer

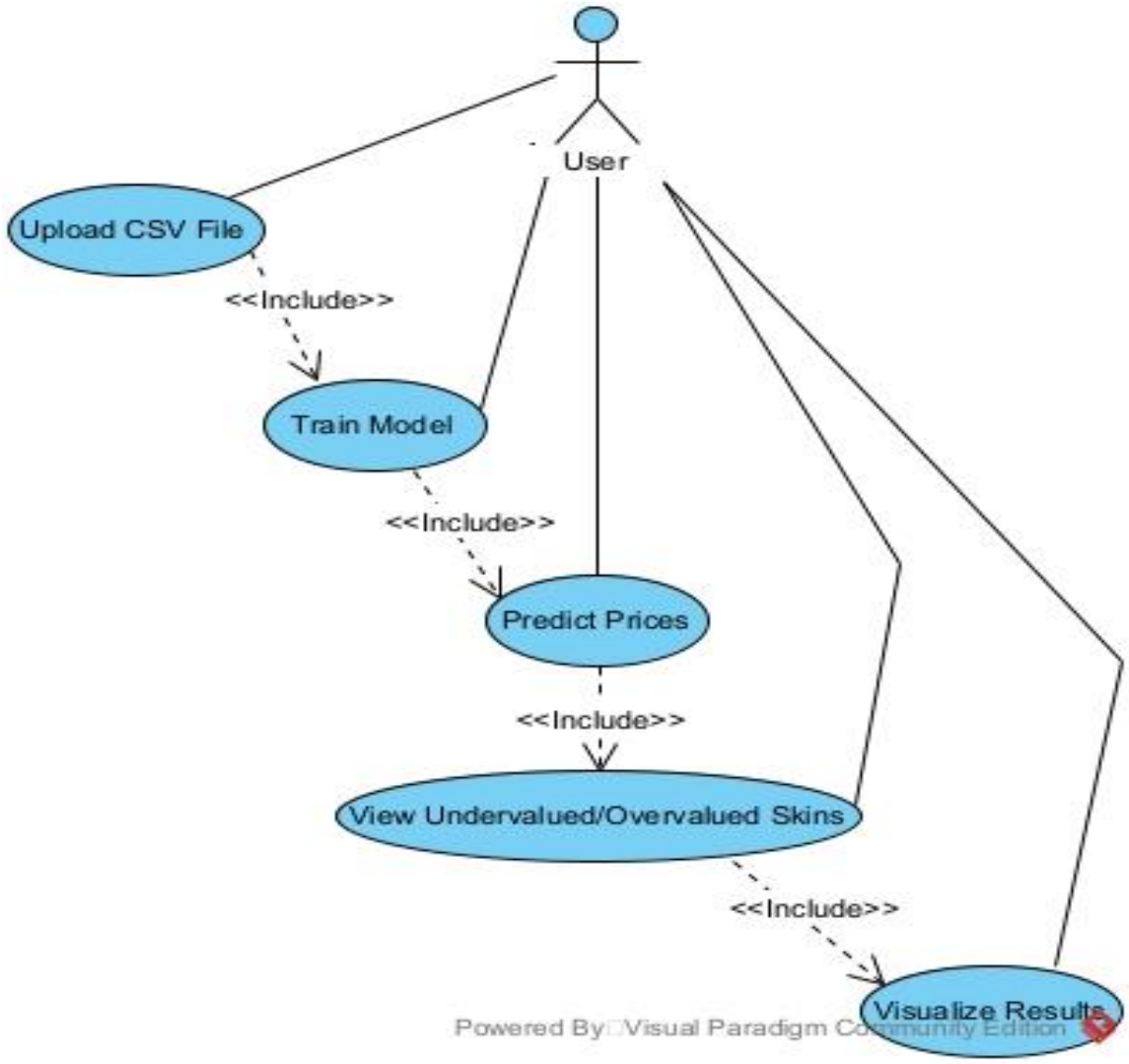
1. Application Description

This application analyzes and predicts the market value of Counter-Strike 2 (CS2) cosmetic items using historical price data. It identifies undervalued and overvalued skins using a machine learning model (Random Forest Regressor), helping users make informed decisions when buying or selling skins.

The system:

- Loads and cleans CS2 skin data.
- Trains a regression model to predict skin prices.
- Compares predicted vs actual prices to detect discrepancies.
- Visualizes findings using static and interactive plots.
- Outputs top undervalued/overvalued items.
- Recommends decisions related to buying/selling skins/items.

2. Use Case Diagram



3. Use Case Details

A) Upload CSV File

Actor: User

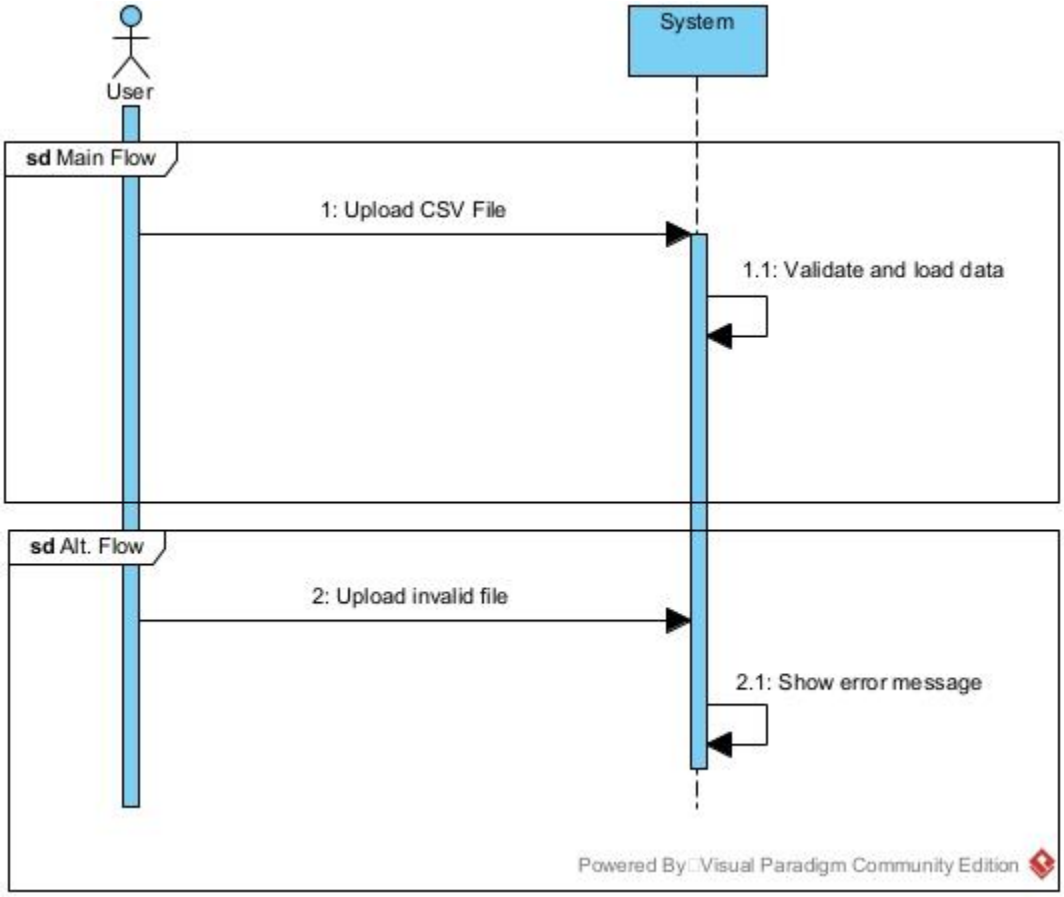
Description: The user uploads a CSV file containing CS2 skin data, which the system validates and loads into memory.

Preconditions:

- The user has access to a valid CSV file with the required columns (**`name`**, **`rarity`**, **`quality`**, **`float_value`**, etc.).
- The system is ready to receive and process the uploaded file.

Postconditions:

- The CSV file is validated and loaded into the system.
- If the file is invalid, an error message is displayed to the user.



B) Train Model

Actor: User

Description: The user requests the system to train a Random Forest model using the uploaded and preprocessed data.

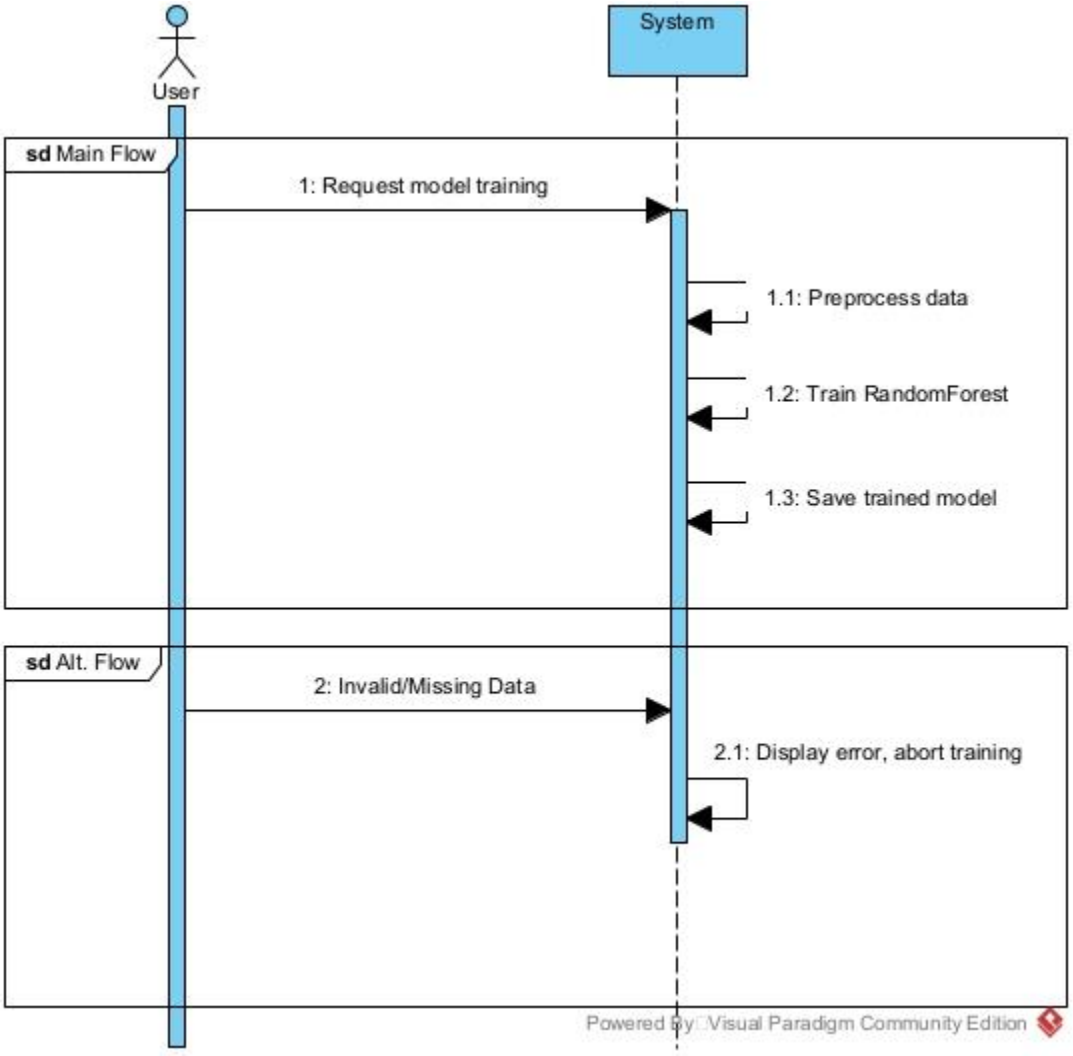
Preconditions:

- The CSV file has been successfully uploaded and validated.

- The data contains all necessary features and target variable (**`price`**).

Postconditions:

- The Random Forest model is trained and saved.
- If the data is invalid or missing, an error message is displayed, and training is aborted.



C) Predict Prices

Actor: User

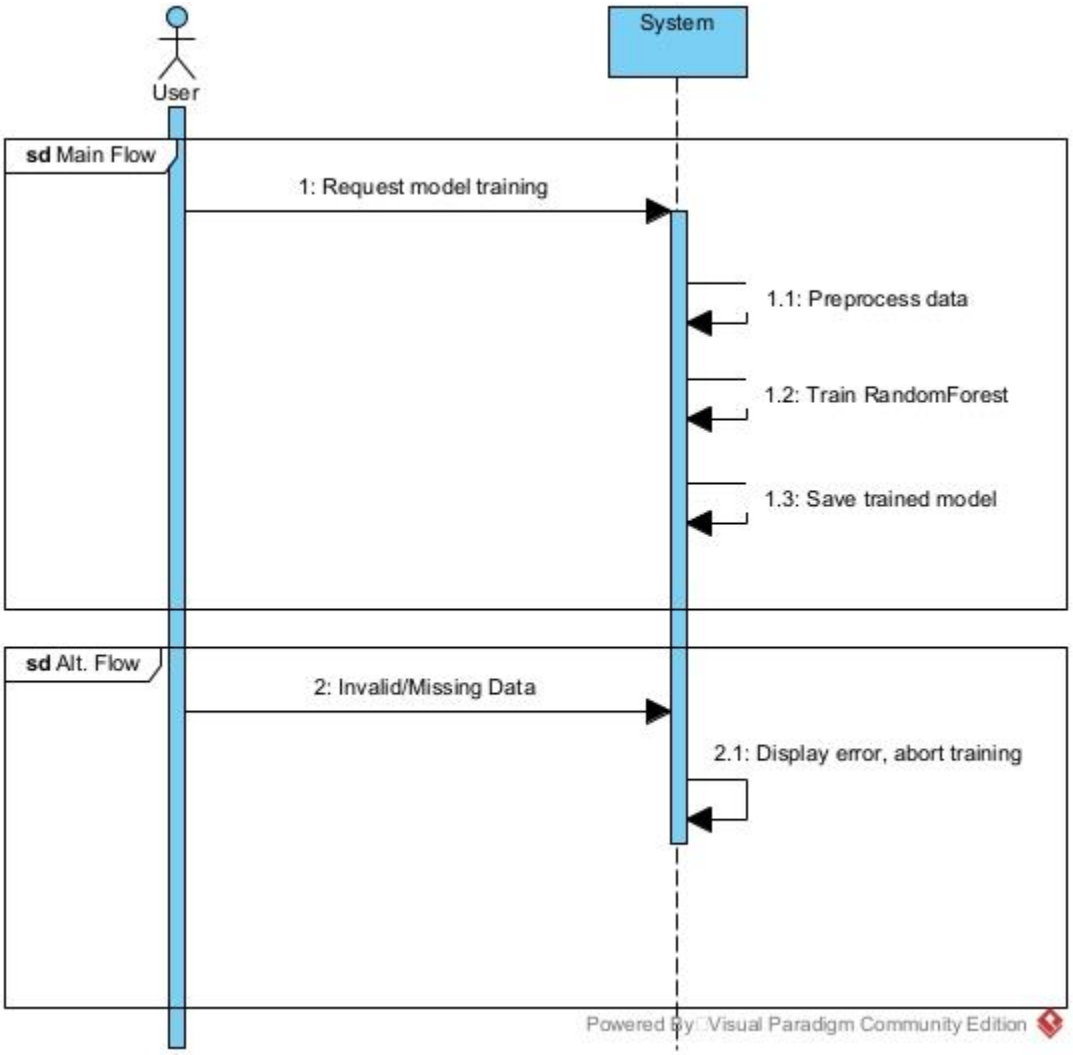
Description: The user requests the system to predict prices for new or existing items using the trained Random Forest model.

Preconditions:

- A trained Random Forest model exists.
- The input data for prediction is available.

Postconditions:

- Predicted prices are returned to the user.
- If no model is trained, the system prompts the user to train the model first.



D) View Undervalued/Overvalued Items

Actor: User

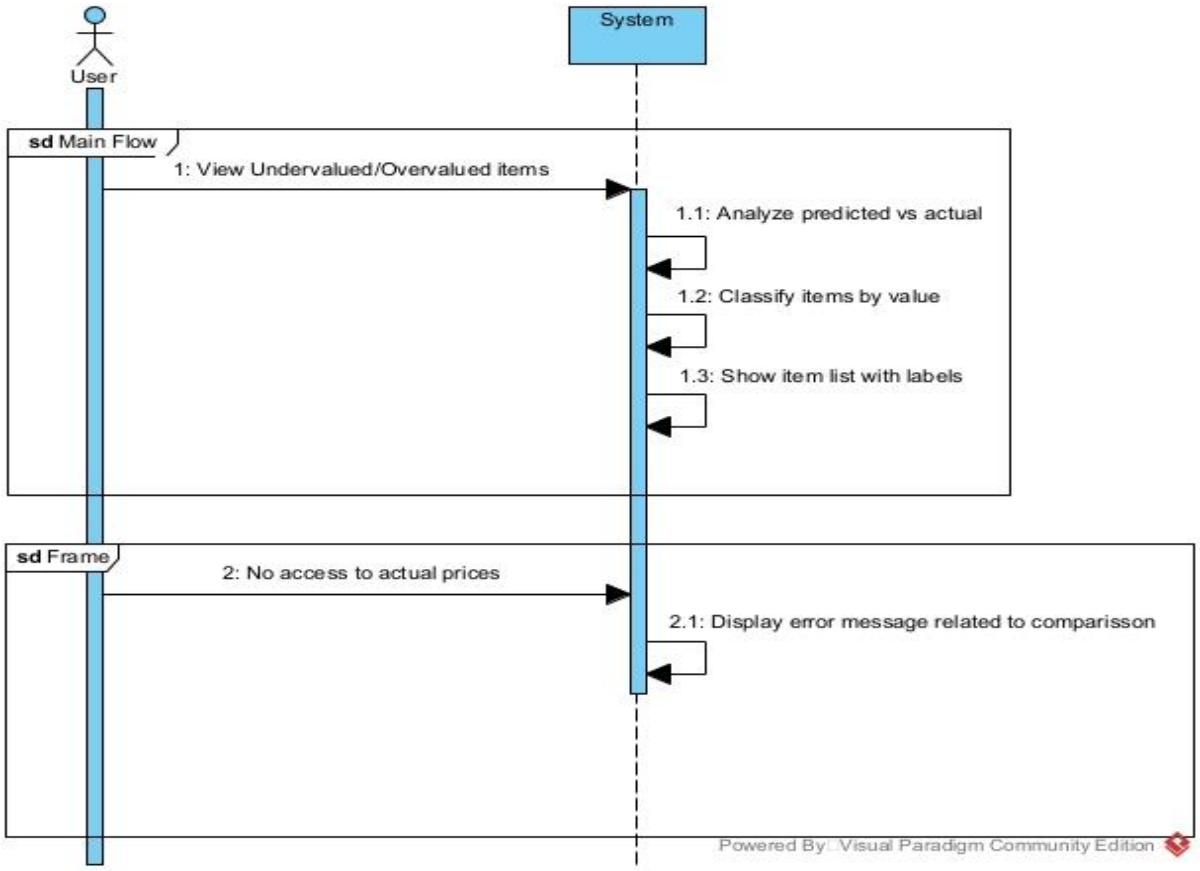
Description: The user requests the system to analyze predicted vs. actual prices and classify items as undervalued, overvalued, or fairly valued.

Preconditions:

- Predicted prices are available.
- Actual prices are available for comparison.

Postconditions:

- Items are classified based on price differences.
- A list of items with labels (e.g., "Undervalued," "Overvalued") is displayed to the user.



E) Visualise Results

Actor: User

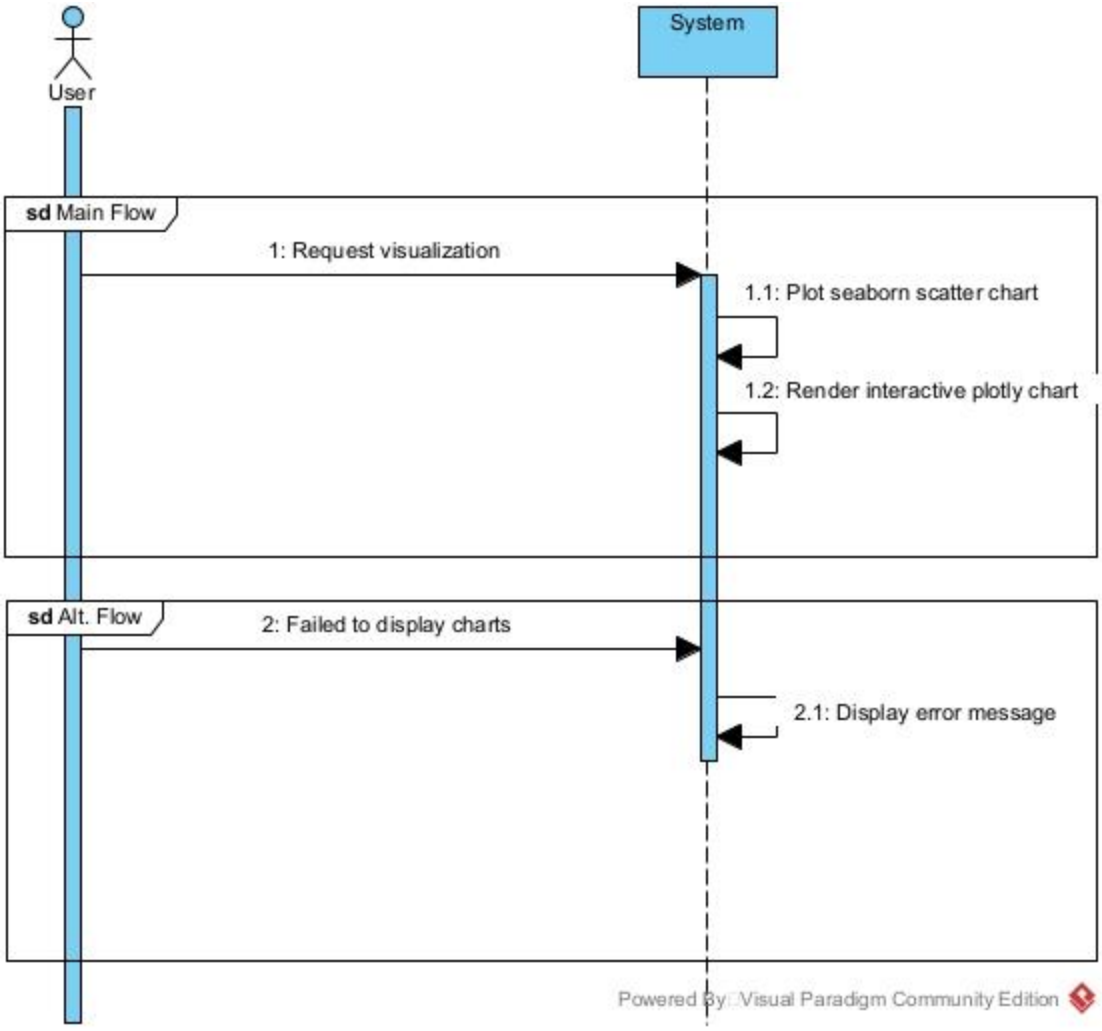
Description: The user requests the system to generate visualizations comparing actual vs. predicted prices using Seaborn scatter plots and Plotly interactive charts.

Preconditions:

- Comparison is available.
- The visualization libraries (Seaborn and Plotly) are properly configured.

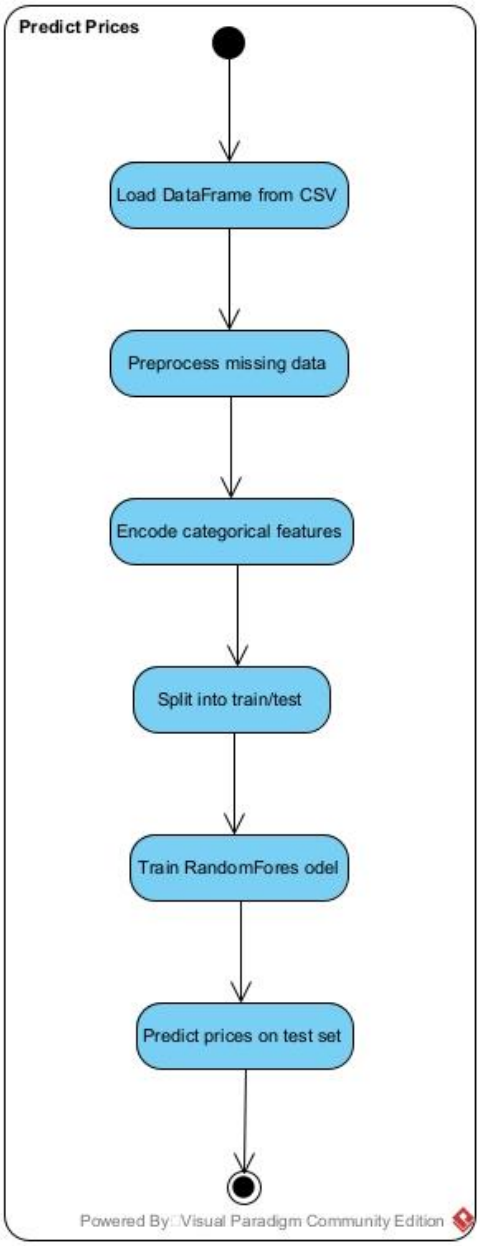
Postconditions:

- A static Seaborn scatter plot is generated.
- An interactive Plotly chart is rendered and displayed to the user.

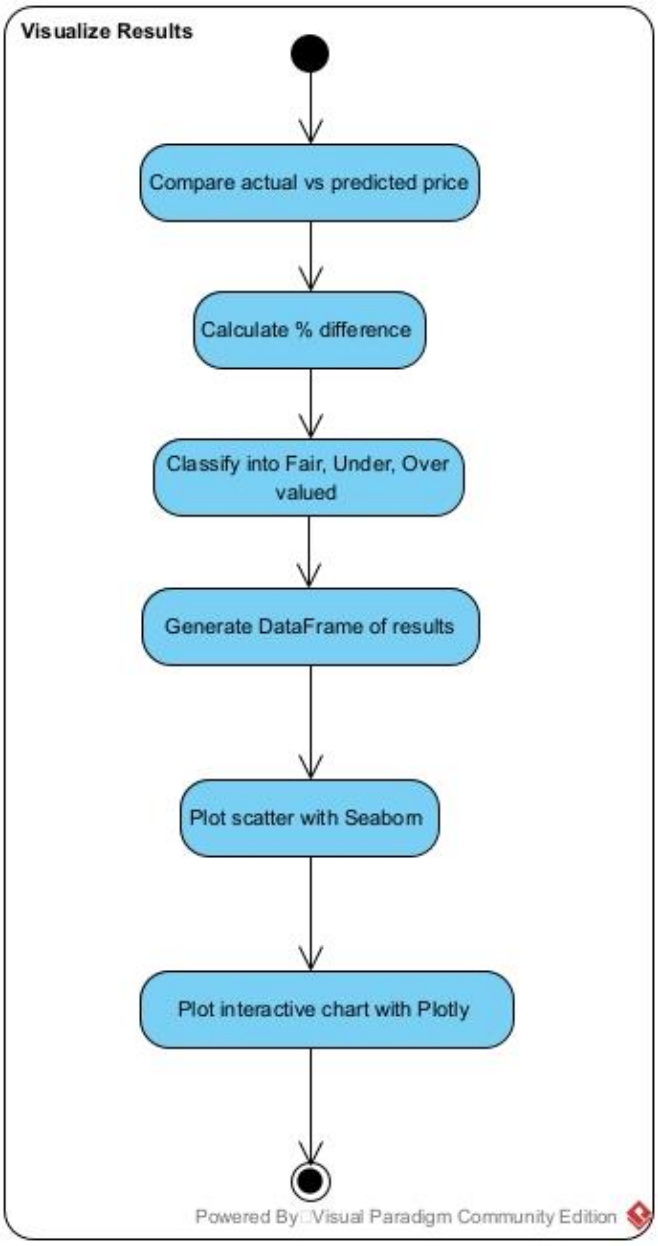


4. Activity Diagrams

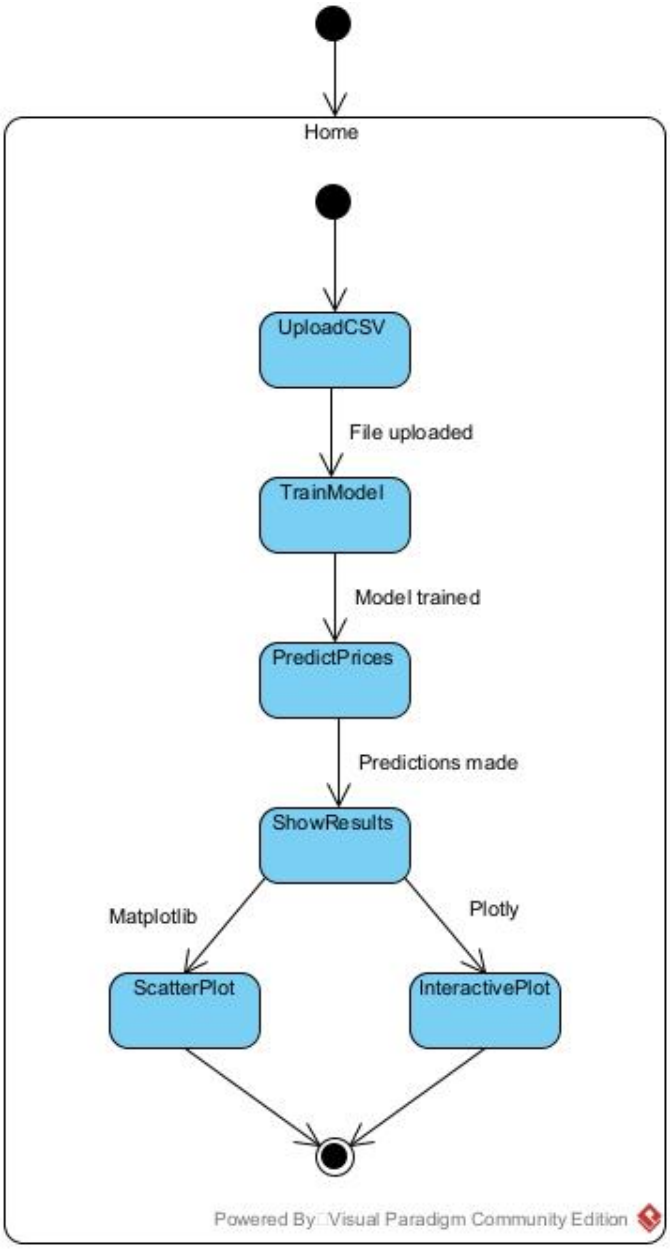
a) Predict Prices



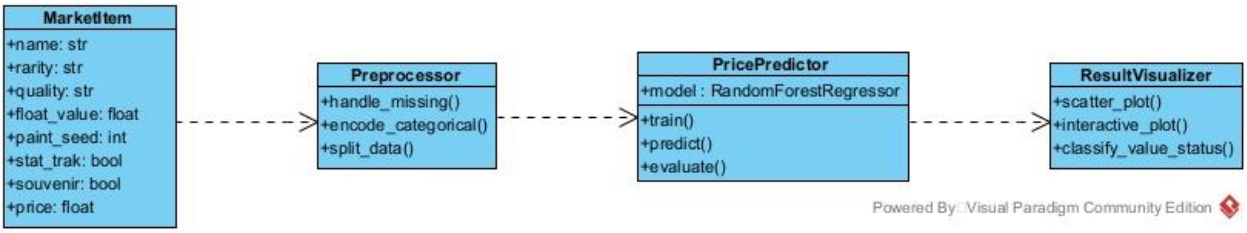
b) Visualize Results



5. GUI Prototype

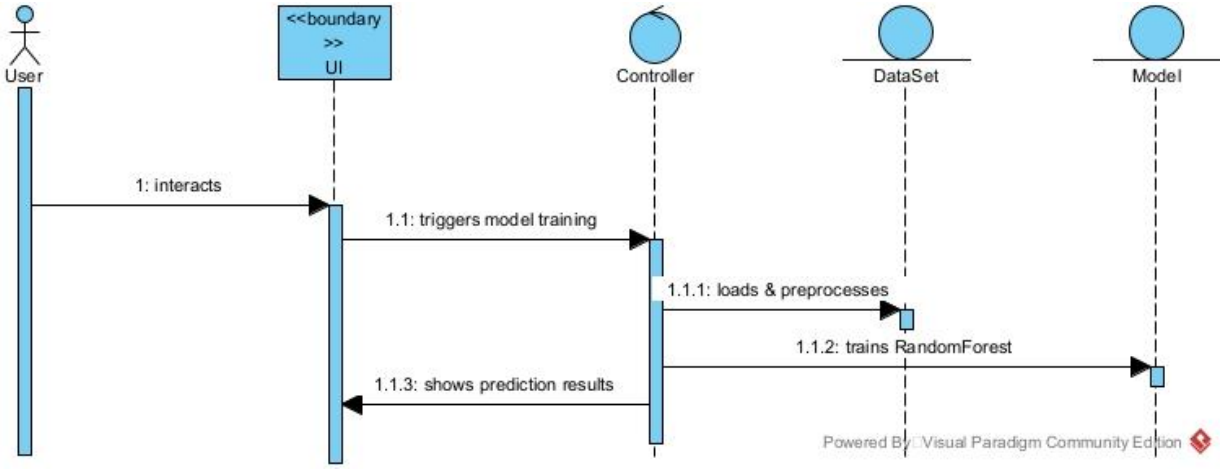


6. Domain Model

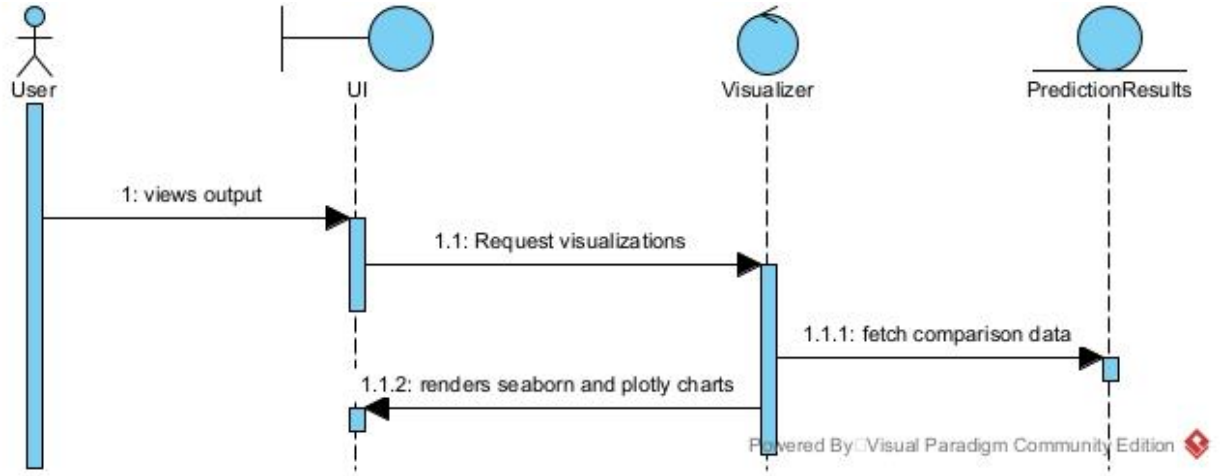


7. Robustness Diagrams

a) Predict Prices

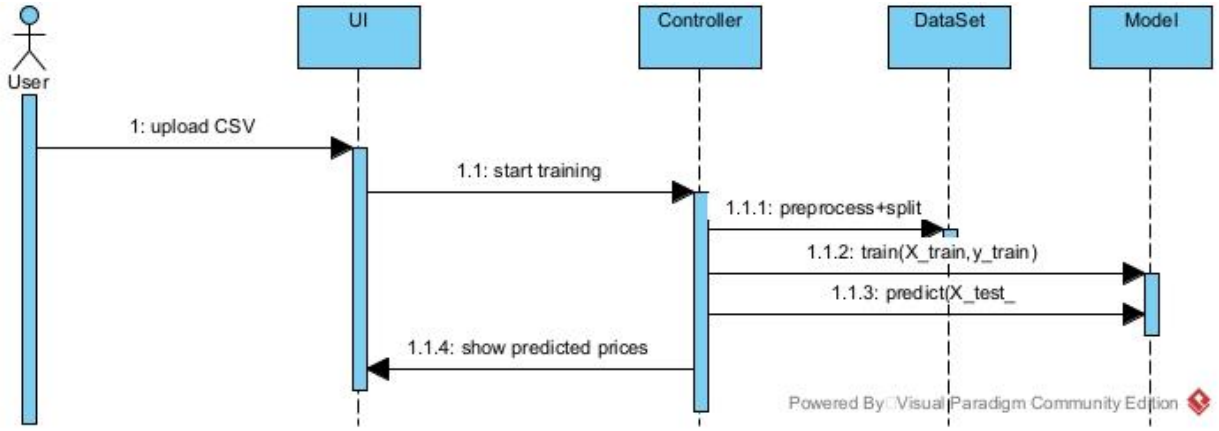


b) Visualize Results

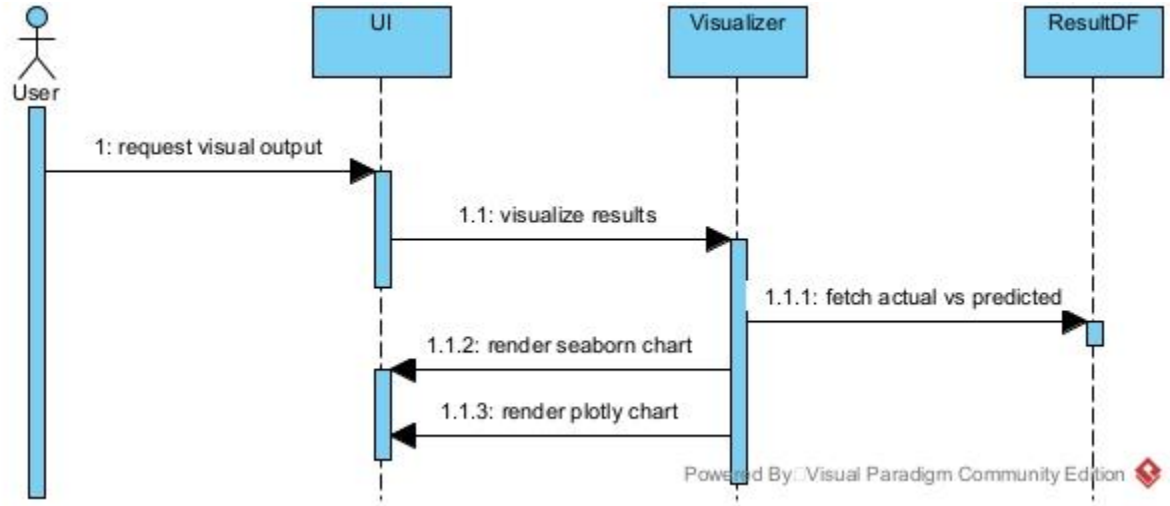


8. Sequence Diagrams

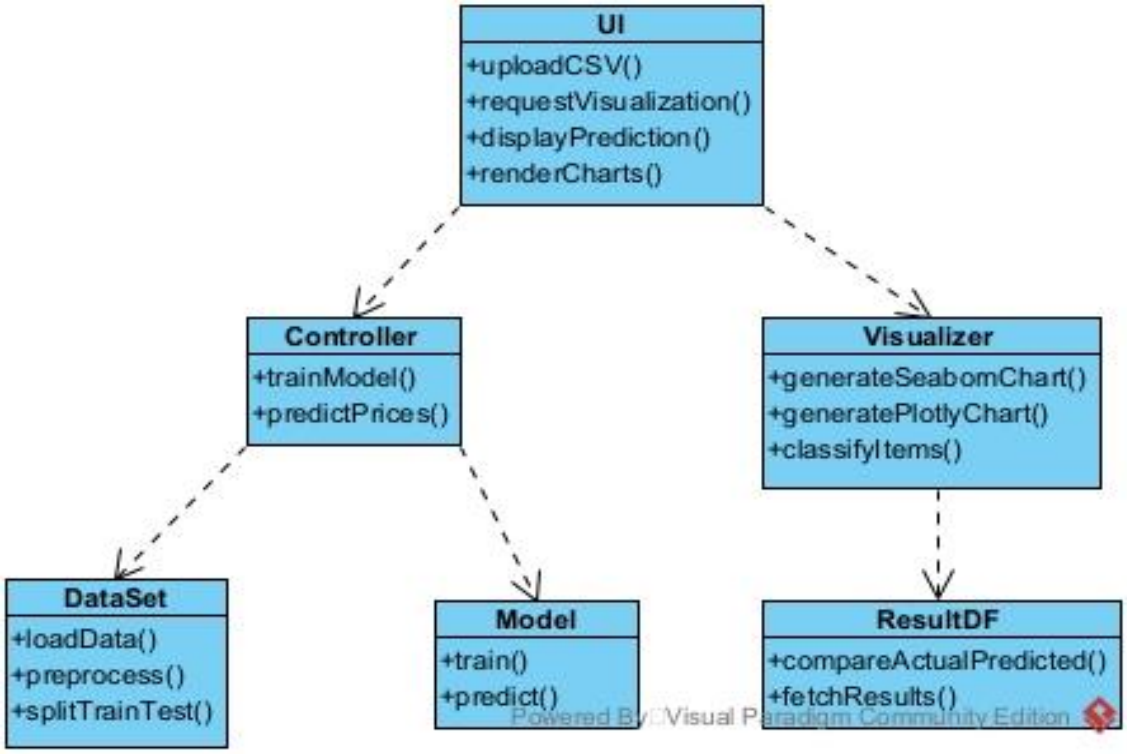
a) Predict Prices



b) Visualize Results



9. Extended Class Diagram


UI
+uploadCSV()
+requestVisualization()
+displayPrediction()
+renderCharts()
Controller
+trainModel()
+predictPrices()
Visualizer
+generateSeabornChart()
+generatePlotlyChart()
+classifyItems()
DataSet
+loadData()
+preprocess()
+splitTrainTest()
Model
+train()
+predict()
ResultDF
+compareActualPredicted()
+fetchResults()
Powered By Visual Paradigm Community Edition