Deployment Of AI

Group-C

Final Report

Price Optimization for Retail Industry

Our Team:

Student Name	Student ID
Abhishek Pandey	500227770
Ram Krishna Dhakal	500228601
Bibek Khadka	500216129

Govind Ram Gupta Belde	**500228074**
Kamalpreet Kaur	**500227884**

Detailed Report on AI Project for Price Optimization in the Retail Industry (Maven Toy Store)

Project Overview

The objective of this project is to optimize pricing strategies in the retail toy industry using machine learning techniques. The project leverages the Maven Toy Store dataset to build a decision tree-based machine learning model for predicting Maximum increase in Profit a retailer can make and deployed the solution using modern DevOps tools such as Jenkins, Docker, and Git. Additionally, a user-friendly interface has been developed to visualize the model's predictions and performance.

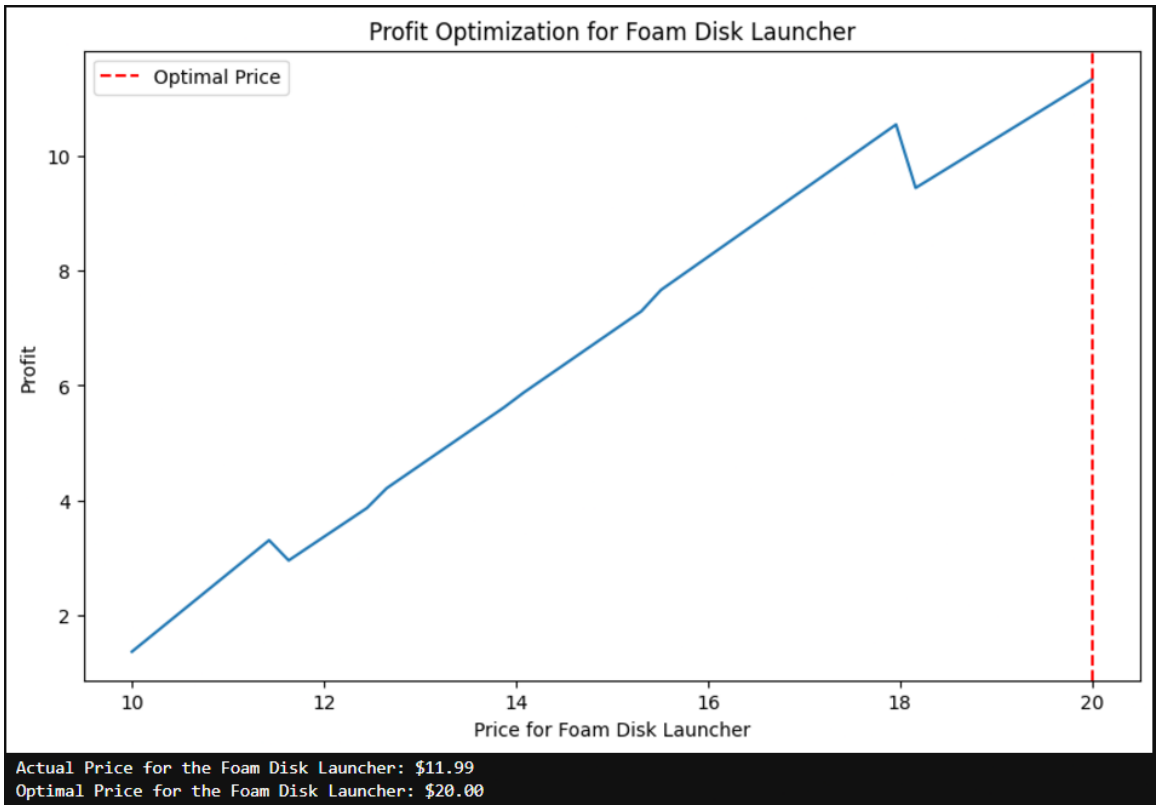
Dataset

- **Dataset Source**: Maven Toy Store Dataset
- **Description**:
    - Contains historical sales data of toys, including features such as toy categories, pricing, seasonal trends, customer demographics, and sales volumes.
    - Includes over 800,000 records with fields such as:
        - Toy ID
        - Product Category
        - Product name
        - Product price
        - Product cost

- Store name
- Store city
- No.of units

Data Preprocessing

1. **Cleaning**:
    a. Removed duplicate records.
    b. Handled missing values using mean/median imputation for numerical columns and mode imputation for categorical columns.
2. **Feature Engineering**:
    a. Created new features such as "Product price" and "Profit"
    b. One-hot encoded categorical variables such as "Category" and "Season."
    c. Merge sales with products data on 'Product_ID'
    d. Merge the result with stores data on 'Store_ID'
    e. Remove dollar signs and convert 'Product_Cost' and 'Product_Price' to numeric
    f. Convert 'Date' to datetime and extract year, month
3. **Splitting**:
    a. Dataset split into training (80%) and testing (20%) sets.
4. **Evaluation Metrics**:
    a. Random Forest Mean Squared Error: 0.6435769340355606
    b. Random Forest R-squared: 0.05284726033696985

c. 

From the above picture we can see that the model predicted an increase in 8 dollars for the above product is possible considering the market scenario.

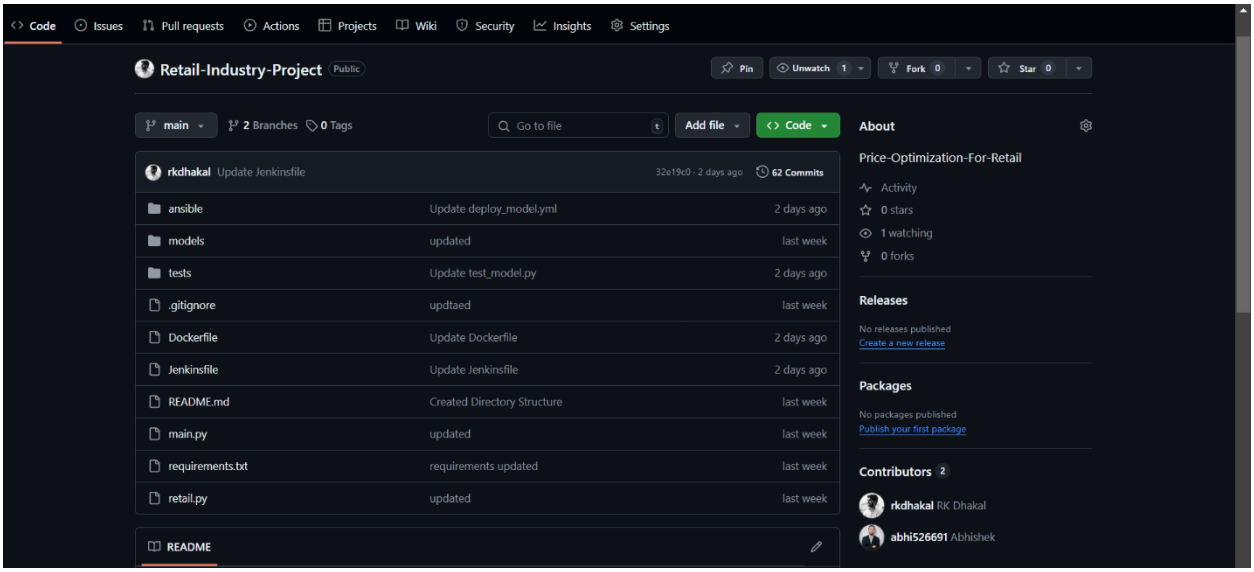
Machine Learning Model

- **Model Type**: Decision Tree
- **Reason for Selection**:
  - Handles categorical and numerical data effectively.
  - Captures non-linear relationships between pricing and sales volume.
  - Provides interpretable results for business stakeholders.
- **Model Training**:
  - Hyperparameter tuning performed using grid search to optimize parameters
  - Created a function to calculate Profit for a given price
  - Created a Dataframe to maintain the feature names
  - Plotted the profits for the different product categories using random forest regressor.

Deployment Pipeline

1. Version Control with Git:

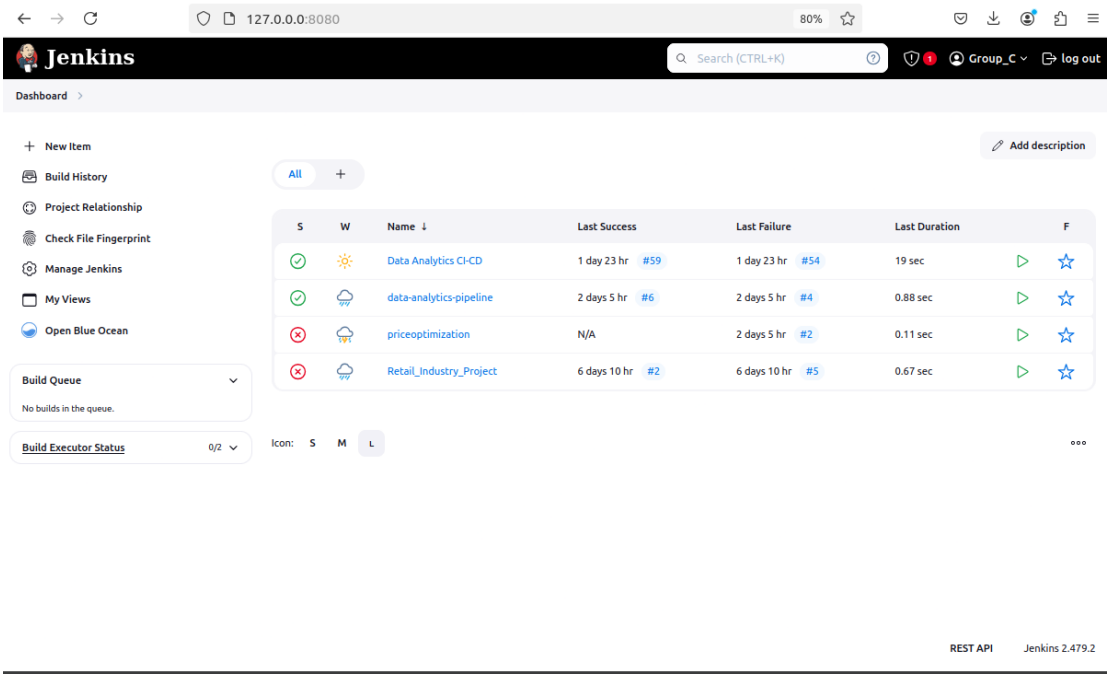
- Repository hosted on GitHub.
- Contains structured directories for:
    - **Data**: Raw and processed data.
    - **Models**: Saved decision tree models.
    - **Scripts**: Python scripts for preprocessing, training, and prediction.
    - **UI**: Source code for the user interface.
    - Jenkins: for CI/CD pipelines
    - Docker: To automate the setup

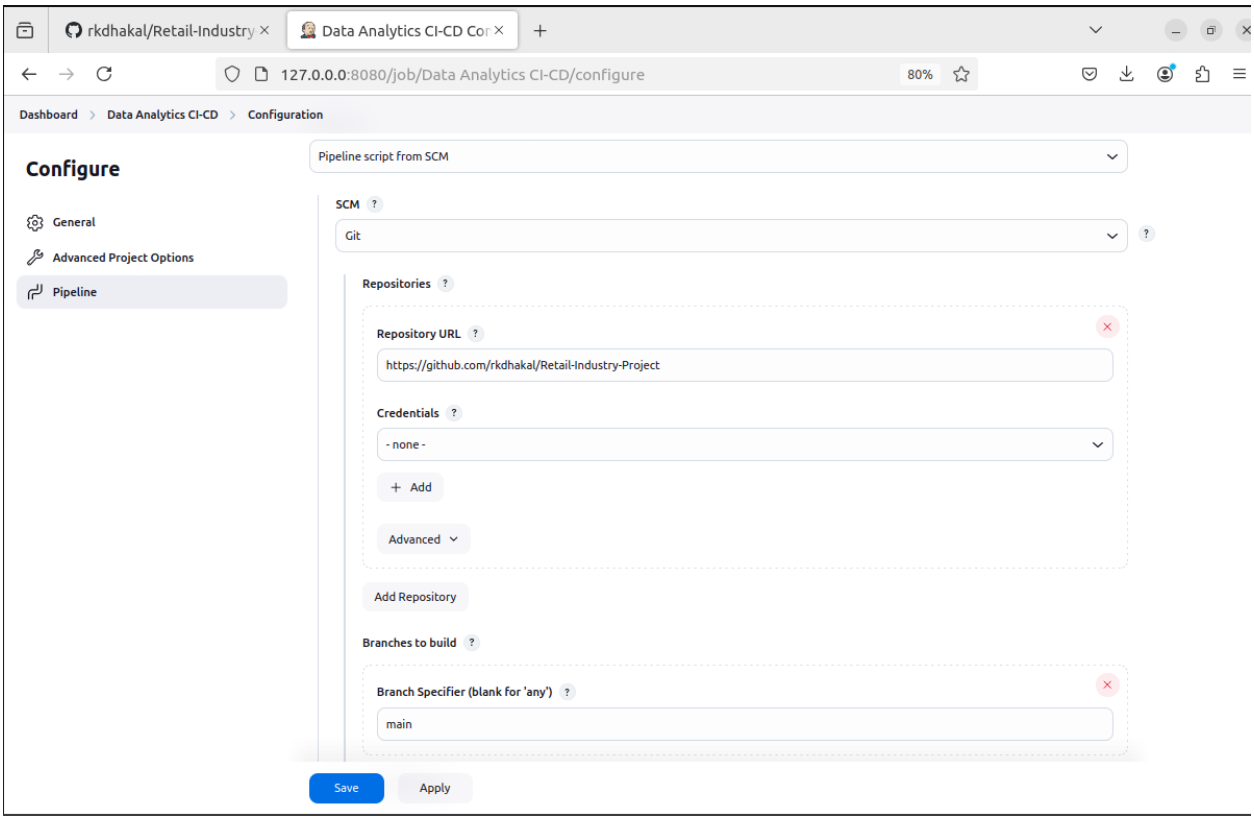


Git link: https://github.com/rkdhakal/Retail-Industry-Project

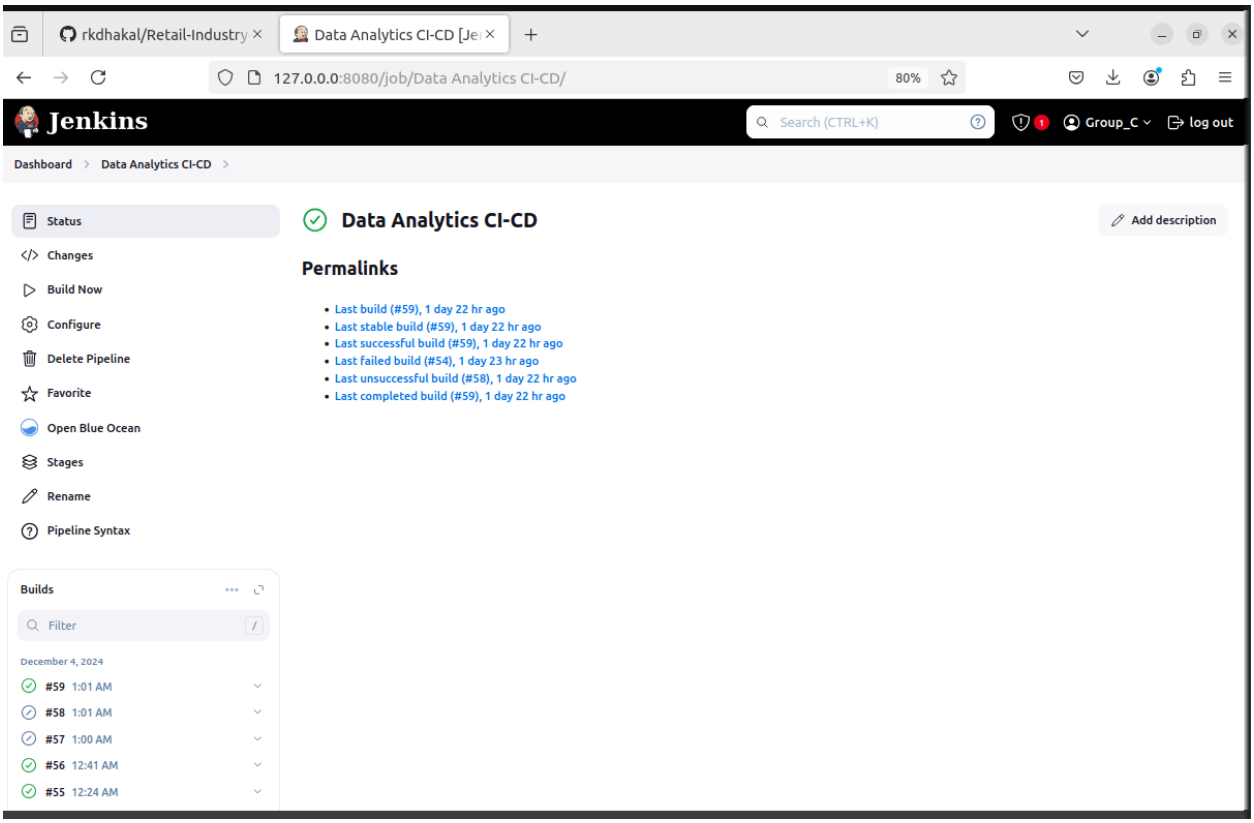
2. Continuous Integration/Deployment with Jenkins:

- Automated pipeline setup using Jenkins to:
    - Fetch the latest code from Git.
    - Run unit tests to ensure code quality.
    - Build the Docker container containing the ML model and dependencies.
    - Deploy the container to a staging environment for validation.





From the above pictures we can see the Jenkins is set up and running.

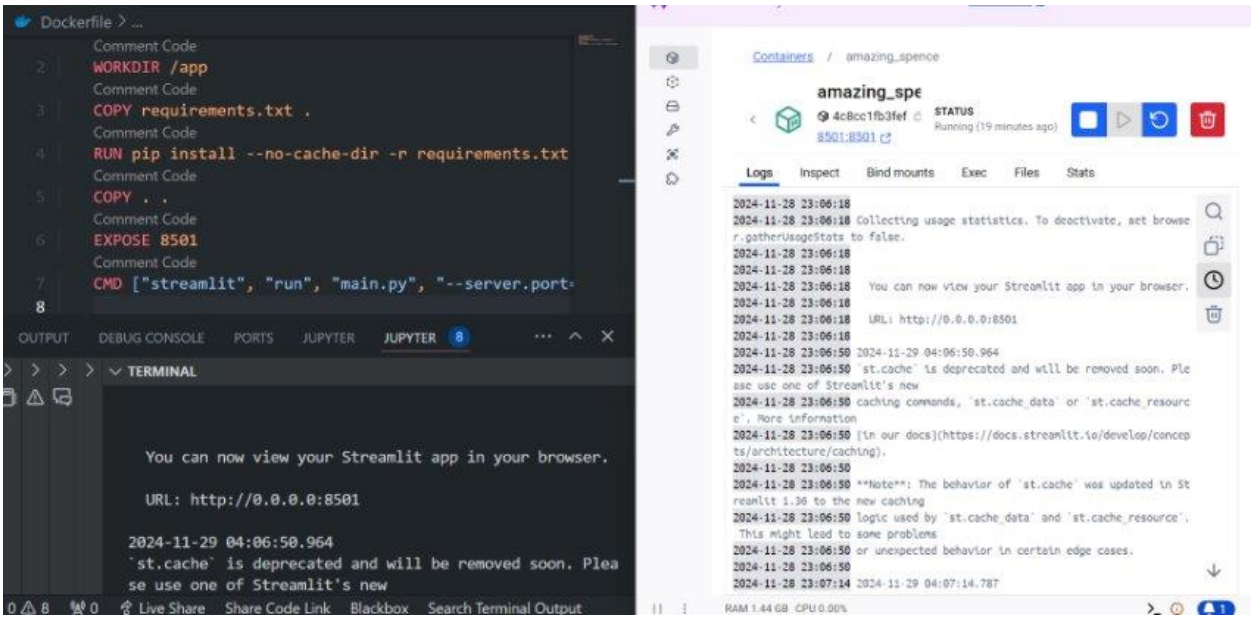


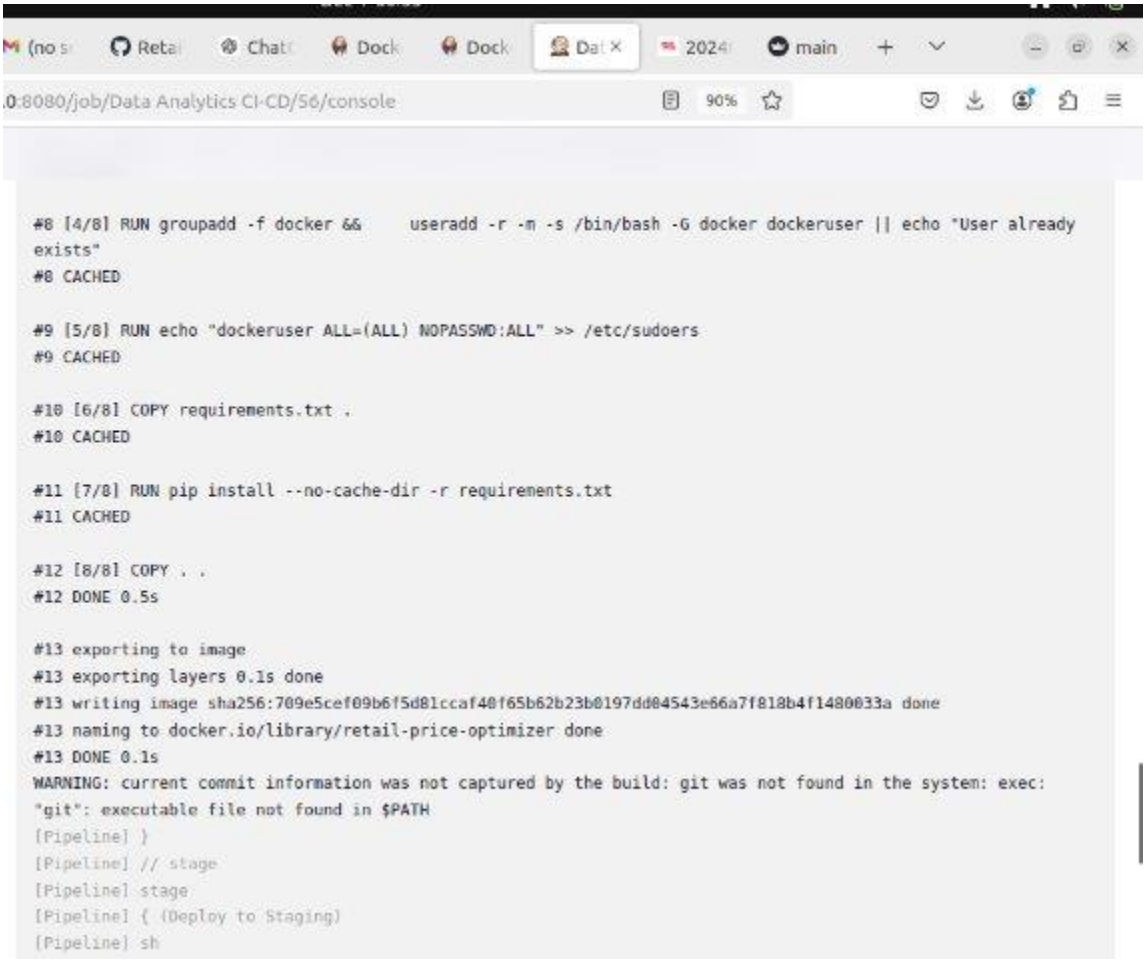
After running this is how it looks like.

3. Containerization with Docker:

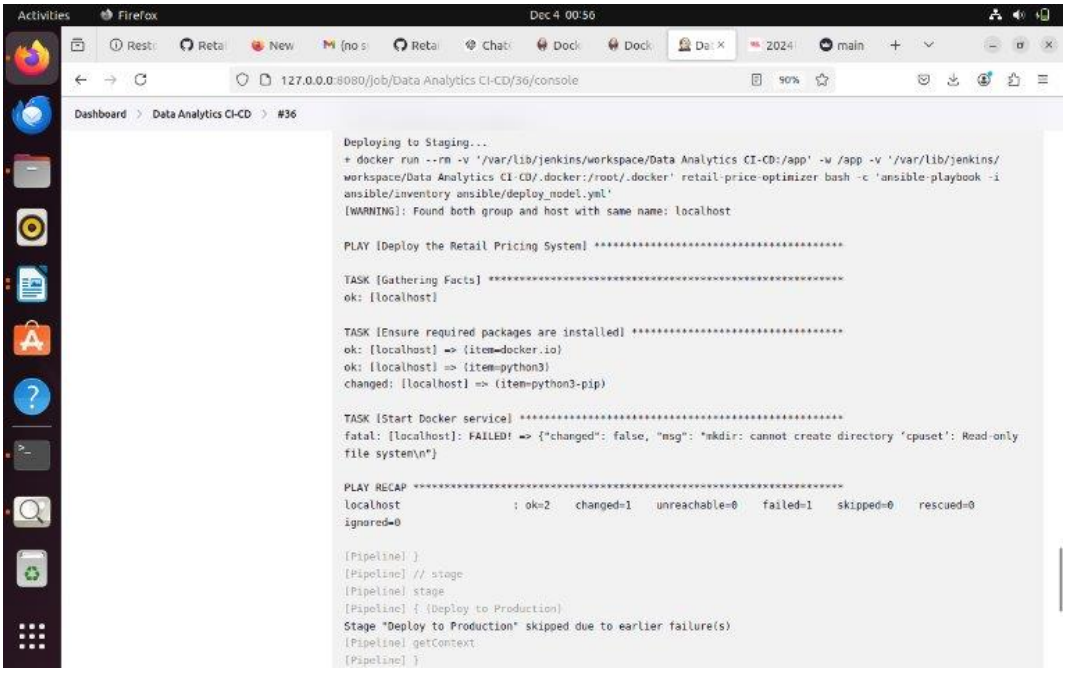
- Created a Dockerfile to package the application.
- Image contains:
    - Preprocessed data pipeline.
    - Decision tree model.
    - Flask application for the user interface.
- Pushed the Docker image to DockerHub for easy access.

We have setup the docker.

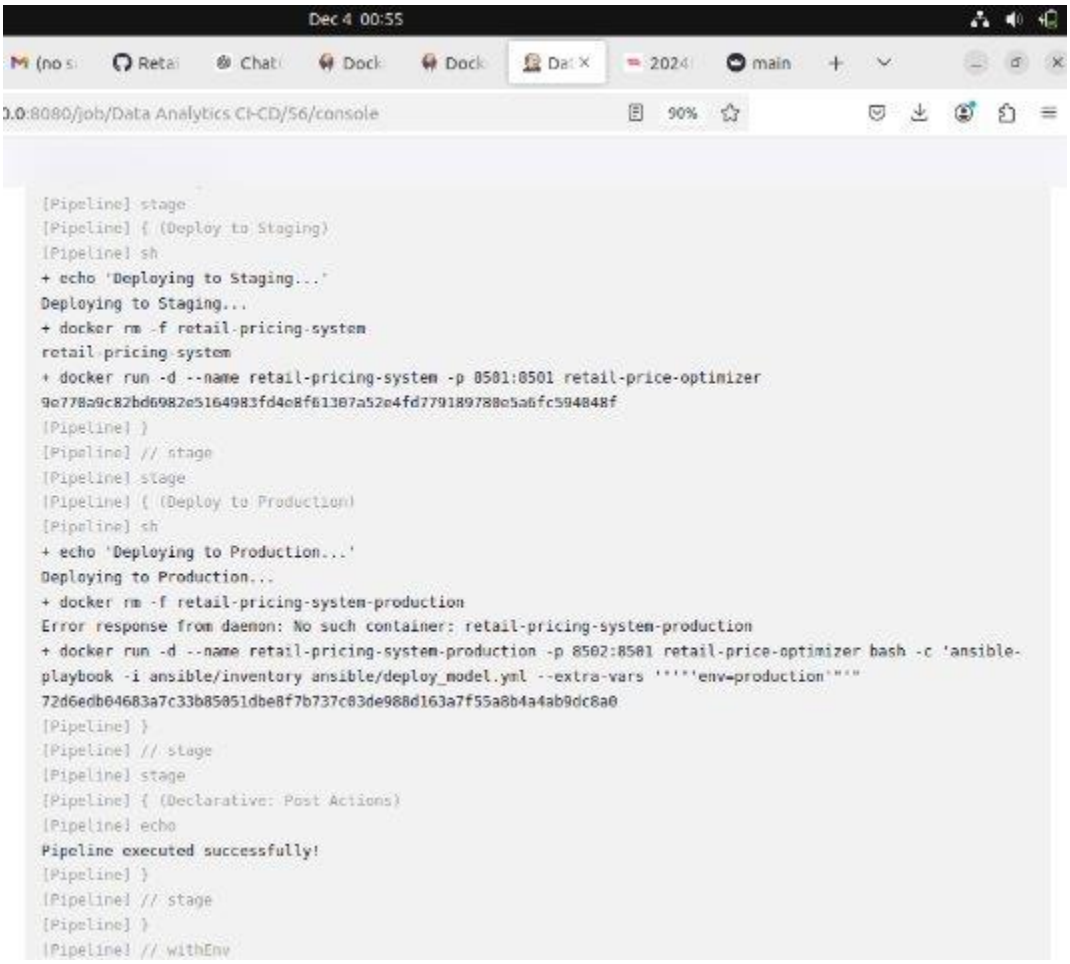


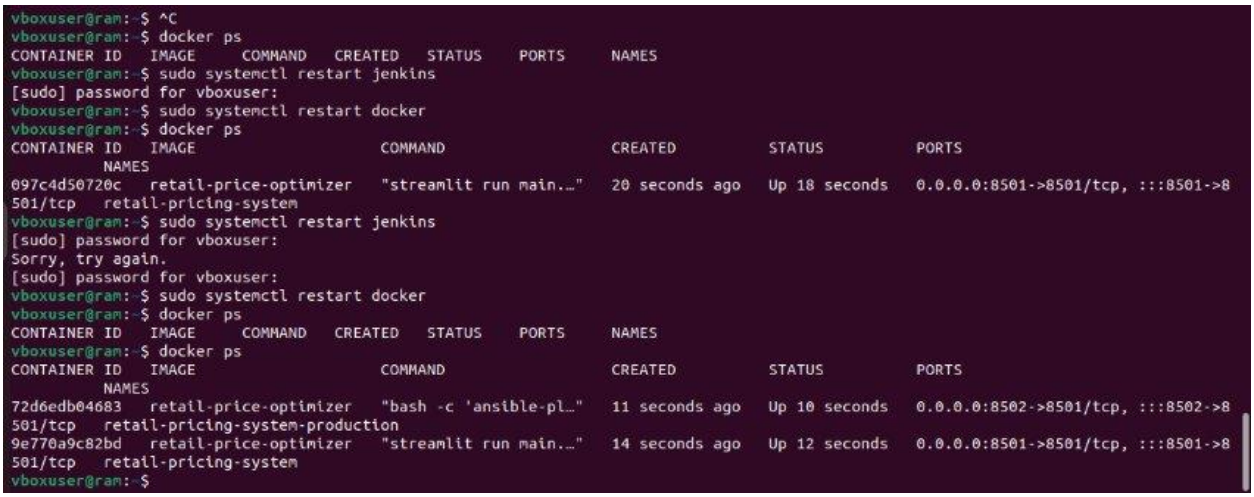


Deployed to staging



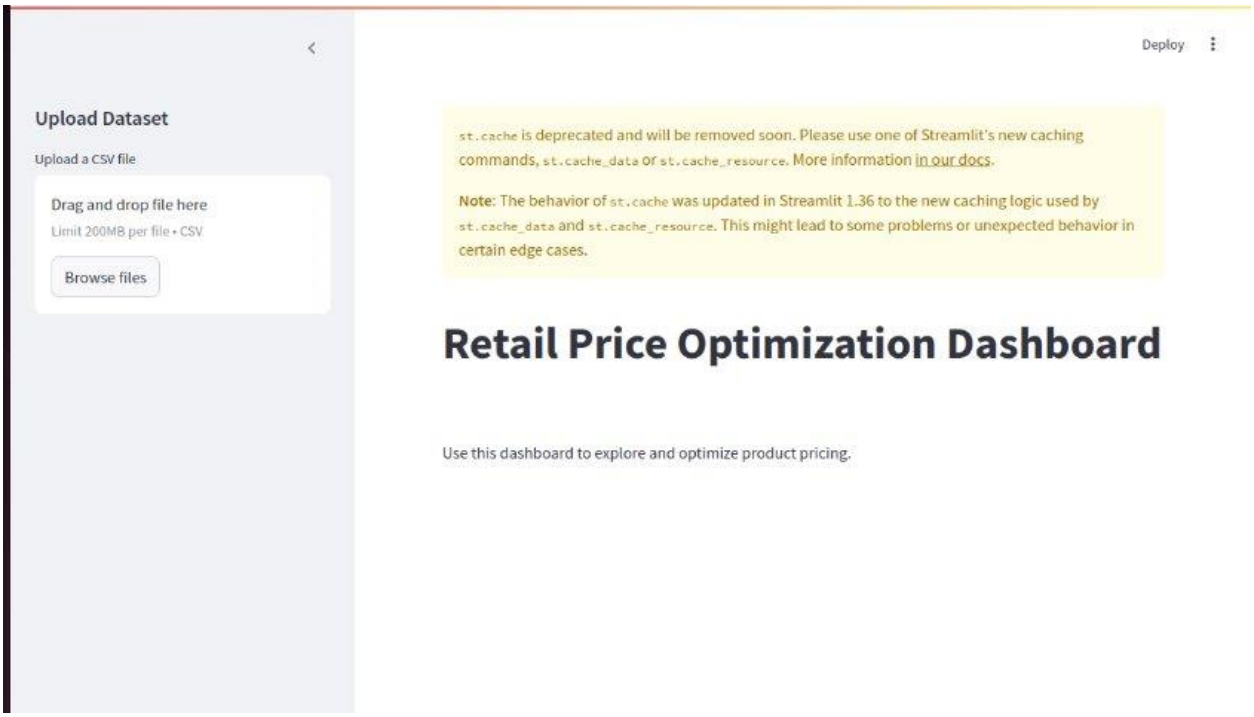
Deployed to production is successful





User Interface

- **Framework Used**: Flask (Python-based web framework).
- **Key Features**:
  - **Input Panel**: Allows users to input toy attributes such as category, base price, and seasonal index to predict optimal price.
  - **Output Panel**:
    - Displays the predicted price.
    - Showcases the impact of discounts and seasonal factors.
  - **Visualization**:
    - Line charts showing actual vs. predicted sales.
    - Heatmaps depicting the relationship between price and sales volume.

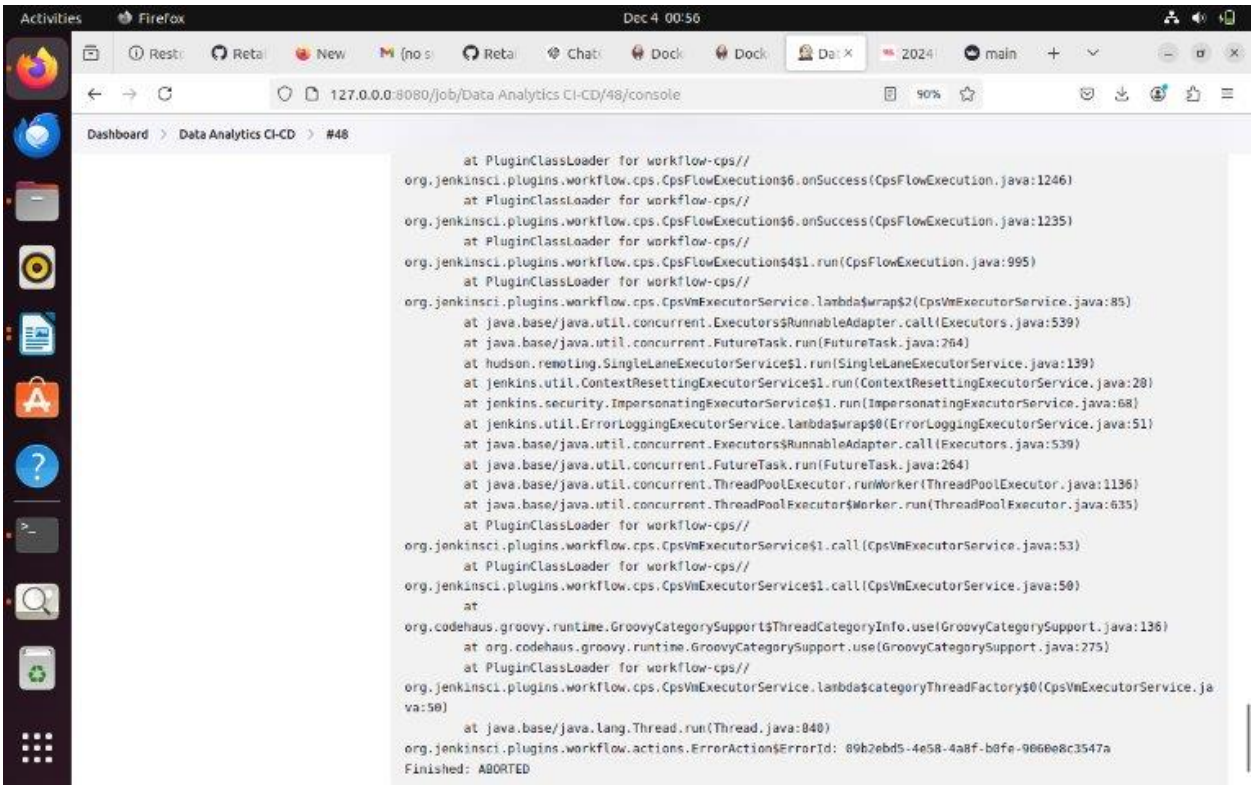


Key Challenges and Solutions

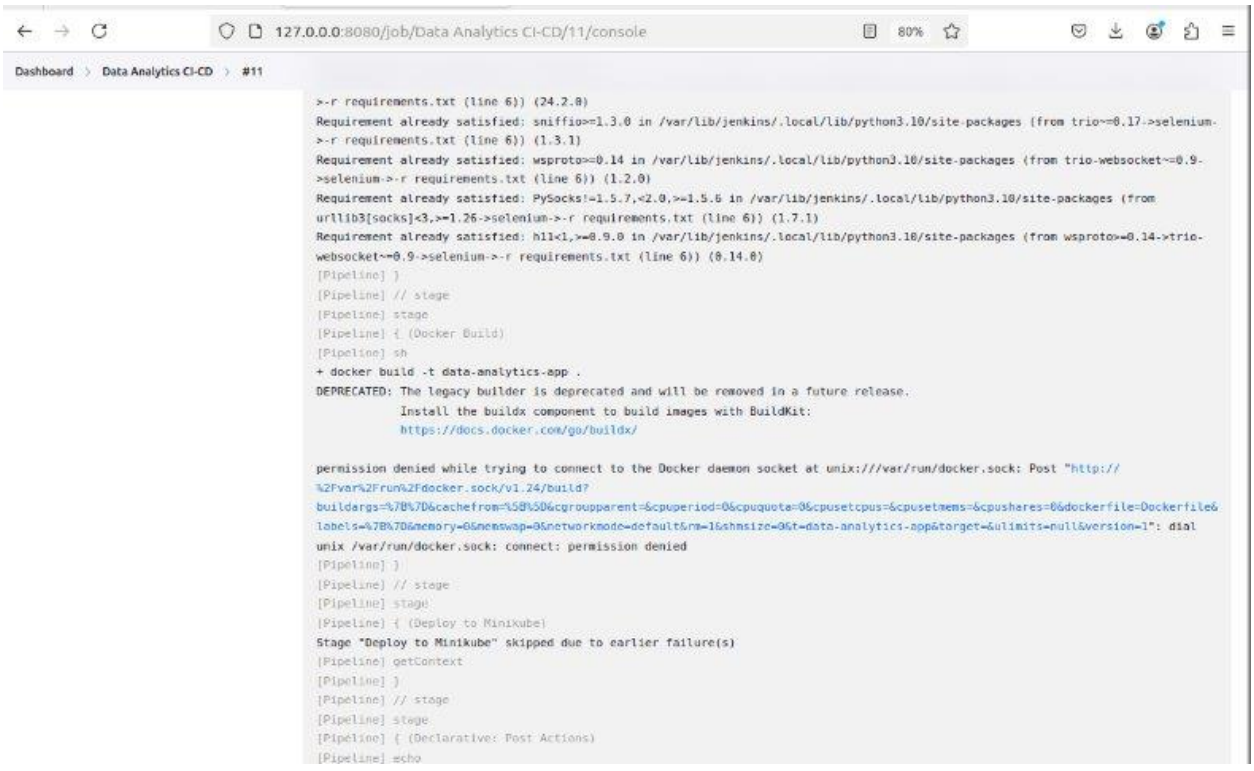
1. **Data Quality**:
    a. Challenge: Missing values and outliers in sales data.
    b. Solution: Implemented robust cleaning and outlier detection mechanisms.
2. **Model Interpretability**:
    a. Challenge: Business stakeholders required easy-to-understand results.
    b. Solution: Selected decision tree models for their inherent interpretability and provided feature importance visualizations.
3. **Deployment Complexity**:
    a. Challenge: Coordinating CI/CD and containerization workflows.
    b. Solution: Streamlined the pipeline using Jenkins' declarative pipeline scripts and Docker.

Errors we faced while working in this project:

Java Error



Docker Permission issue:



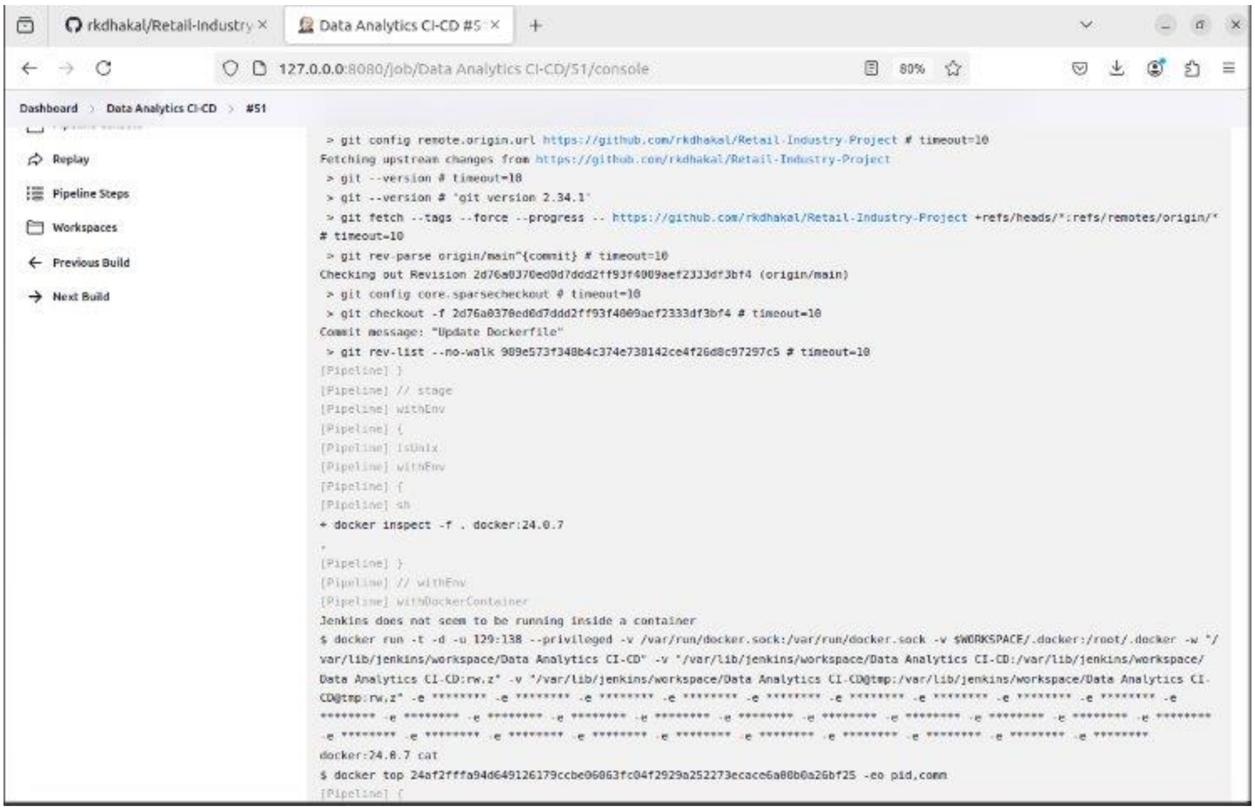
Solution for docker permission issue:

```
# Create a non-root user with Docker permissions
RUN groupadd -f docker && \
    useradd -r -m -s /bin/bash -G docker dockeruser || echo "User already exists"

# Allow the user to use sudo without a password
RUN echo "dockeruser ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers

# Set the environment variable for Docker CLI
ENV DOCKER_HOST=unix:///var/run/docker.sock
```

Jenkins Issue:



Solution to jenkins issue:

```
pipeline {
    agent {
        docker {
            image 'docker:24.0.7'
            args '--privileged -v /var/run/docker.sock:/var/run/docker.sock -v $WORKSPACE/.docker:/root/.docker'
        }
    }

    environment {
        DOCKER_IMAGE = "retail-price-optimizer"
        REPO_URL = "https://github.com/rkdhakal/Retail-Industry-Project"
        DOCKER_CONFIG = "$WORKSPACE/.docker"
    }
```

Conclusion and Future Work

The project successfully demonstrates the application of AI to optimize pricing strategies in a retail toy store. The integration of machine learning with modern DevOps tools ensures a robust and scalable solution.

Future Enhancements:

1. Integrate additional features such as competitor pricing and inventory levels.
2. Enhance the user interface with predictive analytics dashboards.
3. Explore advanced ML models (e.g., ensemble methods) for improved accuracy.
4. Implement real-time data pipelines for dynamic price optimization.