PREDICTIVE INVENTORY CLOUD

PROJECT REPORT

Submitted by

KEERTHI P

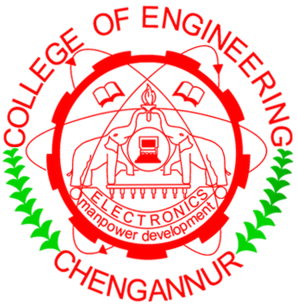
CHN22MCA2028

to

APJ Abdul Kalam Technological University

in partial fulfillment of the requirements for the award of Degree in

Master of Computer Application



DEPARTMENT OF COMPUTER ENGINEERING

COLLEGE OF ENGINEERING CHENGANNUR, ALAPPUZHA

APRIL 2024

DEPARTMENT OF COMPUTER ENGINEERING

COLLEGE OF ENGINEERING CHENGANNUR

ALAPPUZHA

CERTIFICATE

This is to certify that the project report titled ***Predictive Inventory Cloud*** *is a bonafide record of the* ***20MCA246 MAIN PROJECT*** *presented by* ***KEERTHI P*** *(CHN22MCA2028), Fourth Semester Master of Computer Application student, under my guidance and supervision. This project is submitted in partial fulfillment of the requirements for the award of the degree* ***Master of Computer Application*** *of* ***APJ Abdul Kalam Technological University****.*

Smt.Neethu Treasa Jacob	**Sri. Ajoy Thomas**	**Dr. Manju S Nair**
Guide	Project Coordinator	Head of the Dept.
Assistant Professor	Assistant Professor	Associate Professor
Dept.of Computer Engineering	Dept.of Computer Engineering	Dept.of Computer Engineering

DECLARATION

I undersigned hereby declare that the project report "**Predictive Inventory Cloud**" , submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Application of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under the supervision of **Smt.Neethu Treasa Jacob** ,Assistant Professor,Department of Computer Engineering. This submission represents my ideas in my own words, and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to the ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma, or similar title of any other University.

Place: Chengannur **KEERTHI P**

Date : 18/04/2024 **CHN22MCA2028**

ACKNOWLEDGEMENT

This work would not have been possible without the support of many people. First and foremost, I give thanks to Almighty God who gave us the inner strength, resources, and ability to complete our project successfully.

I would like to thank **Dr. Smitha Dharan**, The Principal College of Engineering Chengannur, has provided with the best facilities and atmosphere for the project completion and presentation. I would also like to thank HOD **Dr. Manju S Nair** ,Associate Professor, Department of Computer Engineering, our project coordinator **Sri. Ajoy Thomas** ,Assistant Professor, Department of Computer Engineering, our project coordinator **Smt. Leya G** ,Assistant Professor, Department of Computer Engineering, our project coordinator **Smt. Dr. Sushitha Susan Joseph** ,Assistant Professor, Department of Computer Engineering, our project guide **Smt. Neethu Treasa Jacob**,Assistant Professor, Department of Computer Engineering for the extended help and the encouragement and support given to me while doing the project.

I would like to thank our dear friends and faculties for extending their cooperation and encouragement throughout the project work, without which I would never have completed the project this well. Thank you all for your love and also for being very understanding.

KEERTHI P

CHN22MCA2028

ABSTRACT

The project proposes a predictive inventory management system designed to optimize stock levels using machine learning forecasts. By analysing historical sales data, seasonality, and market trends, the system will predict future inventory requirements. Amazon Forecast, integrated with machine learning techniques, will provide accurate demand forecasts. Amazon SageMaker will be used for custom model development, tailored to specific business needs. The inventory data will be stored and managed in Amazon RDS, ensuring high availability and reliability. AWS Lambda functions will trigger restocking actions based on predictive outputs. This approach aims to minimize stock-outs and overstock situations, leading to improved operational efficiency and cost savings.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Project Area

The project aims to develop a predictive inventory management system powered by machine learning, leveraging historical sales data, seasonal trends, and market insights for accurate forecasting. Utilizing Amazon Forecast and SageMaker, the system will generate precise demand predictions tailored to the business's specific needs. By automating restocking through AWS Lambda functions and ensuring data reliability with Amazon RDS, the system aims to minimize stockouts and overstock situations, thus enhancing operational efficiency and cost-effectiveness. Ultimately, the goal is to revolutionize inventory management, providing businesses with the tools to optimize stock levels, streamline operations, and improve customer satisfaction through consistent product availability. Predict the stock of a particular item.Depending on the report of sale to fulfill the next stocking.It can maintain the stock for a particular time without excess losing .

1.2 Objectives

- **Optimize Stock Levels**: The primary objective is to optimize inventory levels by accurately predicting future demand. This will help in ensuring that the right amount of stock is available at the right time, minimizing stockouts and overstock situations.
- **Utilize Machine Learning Forecasts**: Leverage machine learning techniques, particularly Amazon Forecast, to analyze historical sales data, seasonality, and market trends. This will enable accurate demand forecasting, enhancing inventory management decisions.
- **Custom Model Development**:Utilize Amazon SageMaker for developing custom machine learning models tailored to specific business needs. This allows for the incorporation of additional factors or nuances that might not be captured by standard forecasting methods.
- **Ensure High Availability and Reliability**: Store and manage inventory data in Amazon RDS to ensure high availability and reliability. This ensures that inventory information is always accessible and up-to-date for making informed decisions.
- **Automate Restocking Actions**: Use AWS Lambda functions to trigger restocking actions

based on predictive outputs. This automation streamlines the inventory replenishment process, reducing manual intervention and improving operational efficiency.

- **Minimize Stockouts and Overstock Situations**:The overarching goal is to minimize stockouts (insufficient inventory) and overstock situations (excess inventory). By accurately predicting demand and automating restocking actions, the system aims to strike a balance that optimizes operational efficiency and cost savings.
- **Improve Operational Efficiency**:By optimizing stock levels, automating restocking actions, and leveraging machine learning forecasts, the system aims to improve overall operational efficiency. This includes reducing the time and resources spent on inventory management tasks while ensuring that inventory levels align closely with demand.
- **Achieve Cost Savings**:More efficient inventory management practices, including minimizing stockouts and overstock situations, the system aims to achieve cost savings for the business. This could result from reduced carrying costs associated with excess inventory, as well as avoiding lost sales due to stockouts.

CHAPTER 2

Problem Definition and Motivations

2.1 Existing System

The current inventory management system relies heavily on manual forecasting methods, which are often inaccurate and time-consuming. This leads to frequent stockouts or overstock situations, resulting in lost sales and increased holding costs. Without a comprehensive analysis of historical sales data and market trends, the system struggles to adapt to changing demand patterns and seasonal fluctuations. Additionally, the lack of integration with advanced machine learning techniques limits the system's ability to generate accurate demand forecasts. As a result, the company faces challenges in optimizing stock levels and maintaining operational efficiency.

2.1.1 Limitation Of Existing System

- **Manual Inventory Management**: The current system likely relies on manual methods for inventory management, such as Excel sheets or basic software. This manual approach is time-consuming, prone to errors, and lacks the ability to adapt to changing demand patterns efficiently.
- **Limited Forecasting Capability**: Without the use of advanced machine learning techniques, the system may struggle to accurately forecast future inventory requirements. Traditional methods may not account for seasonality, market trends, or other variables that can impact demand fluctuations.
- **Reactive Restocking**: The system likely operates on a reactive restocking model, where restocking actions are triggered only when inventory levels reach a critical point or when orders are placed. This can lead to stockouts during peak demand periods or overstock situations when demand is low.
- **Poor Data Management**: Inventory data may be scattered across multiple platforms or stored in disparate systems, making it difficult to consolidate and analyze effectively. This can hinder the accuracy of demand forecasts and lead to suboptimal inventory management decisions.

- **Limited Scalability**: The existing system may lack scalability to handle large volumes of data or adapt to the growing needs of the business. As sales volume increases or new product lines are introduced, the system may struggle to keep up with demand forecasting and inventory optimization requirements.
- **High Costs and Inefficiencies**: Manual inventory management and reactive restocking practices can result in higher operational costs and inefficiencies. Stockouts can lead to lost sales opportunities and dissatisfied customers, while overstock situations tie up capital and warehouse space unnecessarily.

2.2 Problem Statement

To revolutionize manual stocking processes, we'll employ predictive analytics to forecast current item sales. These predictions will serve as invaluable references for future stocking decisions, streamlining inventory management and optimizing supply chain efficiency.Developing a predictive inventory management system utilizing Amazon Forecast and SageMaker to optimize stock levels, minimize stockouts, and reduce overstock situations for enhanced operational efficiency and cost savings.

2.3 Proposed System

The project proposes a predictive inventory management system leveraging machine learning forecasts to optimize stock levels. By analyzing historical sales data, seasonality, and market trends,the system utilizes Amazon Forecast for accurate demand predictions. Custom models developed through Amazon SageMaker cater to specific business requirements, while inventory data is stored in Amazon RDS for reliability. AWS Lambda functions automate restocking based on predictive outputs, aiming to minimize stockouts and overstock situations.It presents an automatic prediction.Overall, the system enhances operational efficiency and yields cost savings by ensuring optimal inventory management..

CHAPTER 3

LITERATURE REVIEW

3.1 A Data-Based Model Predictive Decision Support System for Inventory Management in Hospitals by Maria Isabel Fernández; Paula Chanfreut; Isabel Jurado in 2020[1]

This paper presents experimental results from the application of a data-based model predictive decision support system to drug inventory management in the pharmacy of a mid-size hospital in Spain. The underlying objective is to improve the efficiency of their inventory policy by exploiting pharmacy historical data. To this end, the pharmacy staff was aided by a decision support system that provided them with quantities needed for the satisfaction of clinical needs and the risk of stockout in case no order is placed for different time horizons. With this information in mind, the pharmacy service takes the final order decisions. The results obtained during a test period of four months are provided and compared with those of a previous model predictive control approach, which was implemented in the same hospital in the past, and with the usual policy of the pharmacy department.

3.2 A Profit Function-Maximizing Inventory Backorder Prediction System Using Big Data Analytics by Petr Hajek; Mohammad Abedin in 2021[2]

The abstract inventory backorder prediction is widely recognized as an important component of inventory models. However, backorder prediction is traditionally based on stochastic approximation, thus neglecting the substantial amount of useful information hidden in historical inventory data. To provide those inventory models with a big data-driven backorder prediction, we propose a machine learning model equipped with an undersampling procedure to maximize the expected profit of backorder decisions. This is achieved by integrating the proposed profit-based measure into the prediction model and optimizing the decision threshold to identify the optimal backorder strategy. We show that the proposed inventory backorder prediction model shows better prediction

and profit function performance than the state-of-the-art machine learning methods used for large imbalanced data. Notably, the proposed model is computationally effective and robust to variation in both warehousing/inventory cost and sales margin. In addition, the model predicts both major (non-backorder items) and minor (backorder items) classes in a benchmark dataset.

3.3 Real-time RFID-based item tracking using IoT and efficient inventory management using Machine Learning by Ayaskanta Mishra; Manaswini Mohapatro in 2022[3]

Internet of Things (IoT) and Machine Learning (ML) based connected intelligence framework has become a technology enabler in various segments of supply chain like inventory management. In this paper, we have proposed an IoT-cloud architecture for passive RFID tag based real-time Stock Keeping Units (SKUs) tracking and ML algorithms for predictive stock analysis. The SKUs are fitted with RFID tags, those are scanned at entry and exit of the warehouse and this real-time data is sent to cloud server over internet. The acquired data is then processed using ML based software engine using techniques like classification, training and testing. We have considered, ABC Inventory Classification for the SKUs in warehouse and three ML algorithms as: Support Vector Machine (SVM), K-nearest neighbors (KNN) and Bayes for predictive analysis of SKUs in inventory. The result indicates SVM is outperforming with 84.8% accuracy. The accuracy of KNN is 83.6% and Bayes at 74%.

3.4 Factor Analysis of Inventory Management in Thai Construction Industry by Kanyanat Jakkraphobyothin; Suparat Srifa; Thanwadee Chinda in 2018[11]

This paper presents The possession of well-organized inventory management is a key factor for a construction company to maintain its competence among competitors. Efficient inventory management leads to a higher-quality financial performance of construction projects as inventory-related cost can account for 60-65% of total construction cost. The purpose of this research study is to examine key factors affecting inventory management in Thai construction industry. A questionnaire survey was developed to gather data for an exploratory factor analysis (EFA). The results extract four key factors affecting inventory management, including ‘performance’, ‘cost’, ‘strategy’

and ‘inventory policy’, with a total of 15 associated items. The four key factors provide better understanding of inventory management in Thai construction industry.

3.5 Smart Inventory Optimization using Machine Learning Algorithms by Shreyas Bailkar; Kunal Shenoy; Amogh Bedekar; Sandip Bankar; Pranav More in 2024[12]

This paper presents the industries increasingly rely on efficient inventory management, logistics, and supply chain operations, and ML-based intelligence frameworks have emerged as indispensable tools. In this study, ML algorithms are tailored for intelligent inventory management. This research centers around the application of ML techniques within the framework of the ABC Inventory Classification methodology. To assess the effectiveness of these algorithms, a comprehensive training and testing was conducted, followed by model classification. The five ML algorithms examined in this study include Decision Tree Algorithm, Support Vector Machines (SVM), Random Forest Classifier, Naïve Bayes Classifier and K-nearest neighbors (KNN). The goal is to shed light on how these ML algorithms perform in comparison to one another when used for inventory management. Through analysis using inventory dataset, the inference made was that Random Forest Classifiers provides the highest accuracy at 93%. Improving the effectiveness, precision, and financial performance of inventory management practices across various industries and supply chain segments by leveraging the power of ML is an important step of this research. This research study makes significant progress towards integrating intelligent ML-driven solutions into the intricate world of supply chain operations.

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 Hardware Requirements

The minimum hardware configuration required for the proper functioning of the system can be outlined below:

- **CPU**: Intel Core i5 or above
- **Operation Speed**: 2.0 GHz or above
- **Ram**: 4GB or above

4.2 Software Requirements

The minimum software configuration required for the proper functioning of the system can be outlined below:

- **Operating System (OS)**: Windows 8 or above, or the latest version of any Linux ubuntu .
- **Code Editor(CE)**: Visual Studio Code.
- **Development Languages Frontend**: Bootstrap,HTML,CSS,JavaScript.
- **Development Languages Backend**:Php, Python 3.8 with python supporting framework like Flask and libraries like OpenCV ,Pandas, Scikit Learn.

4.3 Funtional Requirements

- **User Interface:**: Develop a user-friendly interface for accessing inventory data, demand forecasts, and restocking recommendations.Provide visualization tools to display historical sales data, forecasted demand, and inventory levels.Allow users to customize views and generate reports for further analysis.
- **API Testing**: Test the functionality and reliability of APIs used for data exchange between system components.Verify API endpoints for data retrieval, forecast generation, and restocking triggers.Conduct stress testing to evaluate API performance under heavy loads..
- **End-to-End Testing**: Perform end-to-end testing to validate the entire workflow from data

ingestion to restocking actions.Simulate real-world scenarios to assess system behavior under various conditions, such as sudden demand spikes or data anomalies. .

- **Data Integration and Preprocessing**: The system should integrate with various data sources, including historical sales data, inventory records, seasonality data, and market trends.Data preprocessing should involve cleaning, normalization, and feature engineering to prepare it for analysis.
- **Inventory Management**: Maintain a centralized repository for inventory data using Amazon RDS for high availability and reliability.Track current inventory levels, including quantities and locations.Implement algorithms to calculate optimal stock levels based on demand forecasts and lead times.
- **Integration Testing**: Verify seamless integration between different components of the system, including Amazon Forecast, SageMaker, RDS, Lambda functions, and user interface. Test data flow and communication protocols to ensure reliability and consistency.

4.4 Non Functional Requirements

- **Performance**: The system should be able to handle large volumes of data efficiently and provide timely forecasts even during peak usage periods.
- **Scalability**: It should be capable of scaling both vertically and horizontally to accommodate increasing data volumes and user demand over time. Reliability: The system should have a high level of uptime, with minimal downtime for maintenance or upgrades. It should also be resilient to failures and able to recover gracefully from errors.
- **Security**: Data privacy and security should be maintained at all times, with appropriate measures in place to protect sensitive information from unauthorized access or breaches.
- **Usability**: The system should be user-friendly and intuitive, with clear interfaces for data input, model configuration, and result visualization. Training and documentation should be provided to support users in utilizing the system effectively.
- **Compatibility**: The system should be compatible with various data formats, integration protocols, and existing infrastructure within the organization.
- **Maintainability**: It should be designed with modular components and well-documented code to facilitate future updates, modifications, and troubleshooting.

- **Cost-effectiveness**: The solution should provide value for money, with considerations for the cost of AWS services, infrastructure maintenance, and ongoing support.

4.4.1 Performance Requirements

- **Forecast Accuracy**: By leveraging Amazon Forecast and custom machine learning models developed using Amazon SageMaker, the predictive inventory management system aims to achieve high accuracy in forecasting future inventory requirements. The system will continuously analyze historical sales data, seasonality, and market trends to refine and improve its predictions over time. Regular evaluation against actual sales data will ensure that the forecasts remain accurate and reliable.
- **Inventory Turnover**: The implementation of the predictive inventory management system is expected to optimize inventory turnover by aligning stock levels with demand fluctuations. By accurately predicting future inventory requirements, the system can help businesses maintain optimal stock levels, minimizing excess inventory while ensuring that products are available when needed. This optimization can lead to a higher inventory turnover ratio, indicating efficient utilization of resources and reduced carrying costs.
- **Reduction in Stockouts**: One of the primary goals of the system is to minimize stockouts, where products are unavailable when customers demand them. By proactively analyzing demand patterns and triggering restocking actions based on predictive outputs, the system aims to ensure that sufficient inventory is always available to meet customer needs. This reduction in stockouts can enhance customer satisfaction, prevent revenue loss due to missed sales opportunities, and preserve the reputation of the business.

4.4.2 Quality Requirements

- **User-Friendliness**: The interface of the system should be intuitive and easy to use for the end-users, such as inventory managers or purchasing teams.
- **Scalability**: The system should be able to scale seamlessly to accommodate increasing volumes of data and growing business needs.
- **Adaptability to Changing Market Conditions**: The system should be capable of adapting to changing market dynamics, such as shifts in consumer demand patterns or seasonal variations.

CHAPTER 5

DESIGN AND IMPLEMENTATION

5.1 Overall Design

5.1.1 System Design

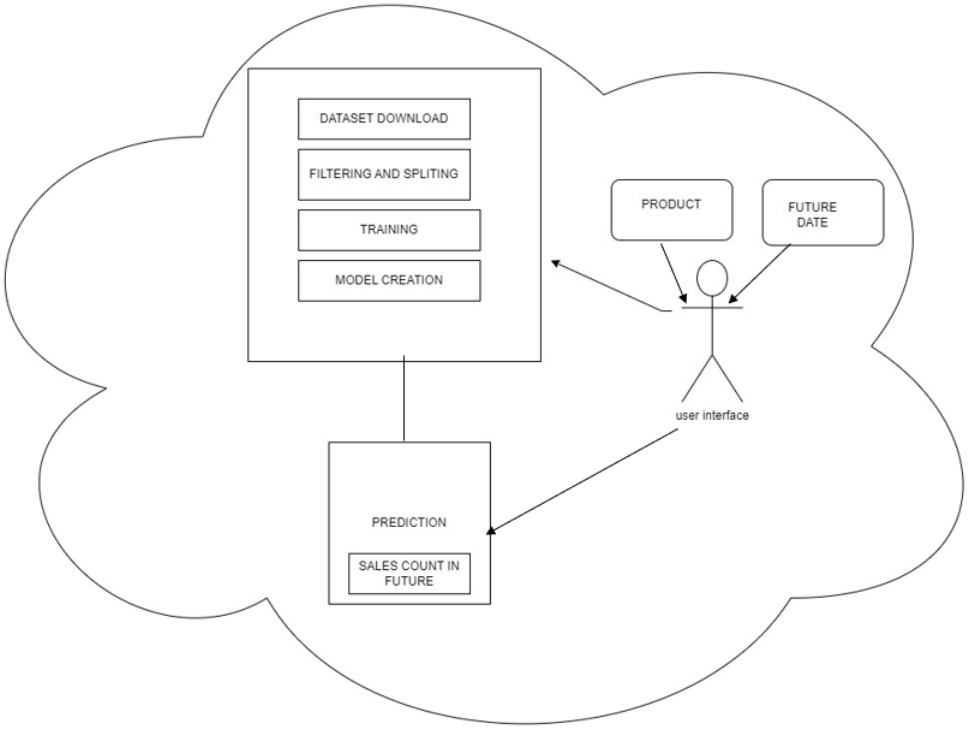


Figure 5.1: System Design

5.1.2 System Architecture

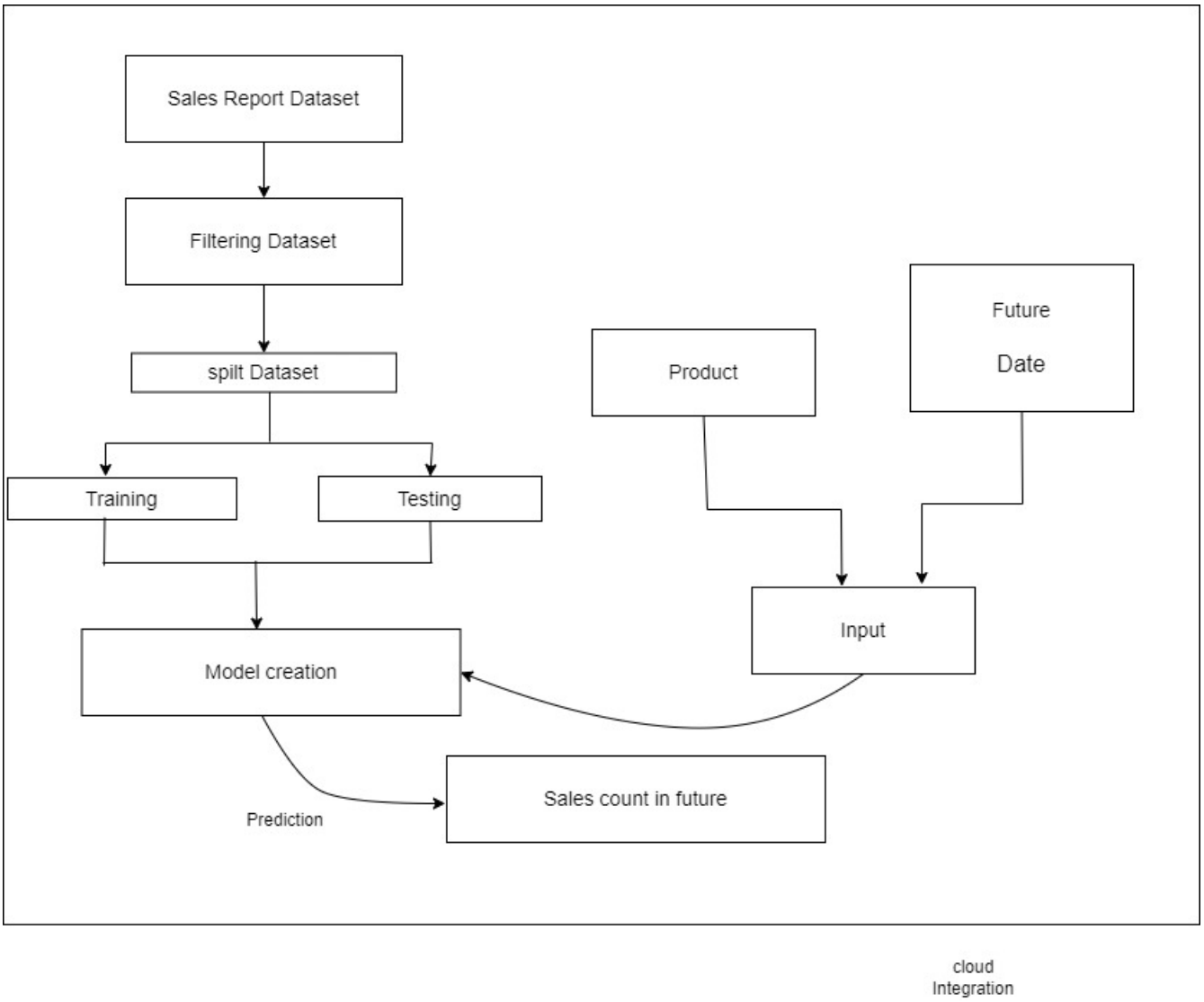


Figure 5.2: Architecture of the system

5.2 Methodology

5.2.1 Data Collection and Preparation

- Gather historical sales data, including transactional records, stock levels, and any relevant metadata.
- Clean and preprocess the data to handle missing values, outliers, and inconsistencies.
- Organize the data into a suitable format for analysis, ensuring compatibility with the chosen machine learning algorithms.

5.2.2 Feature Engineering

- Extract relevant features from the dataset, such as product attributes, seasonality indicators, historical trends, and market dynamics.
- Transform and engineer these features to enhance their predictive power and relevance to inventory forecasting.

5.2.3 Model Selection and Development

- Utilize Amazon Forecast for demand forecasting, leveraging its advanced machine learning algorithms and built-in functionalities.
- Employ Amazon SageMaker for custom model development, tailoring algorithms to specific business requirements and datasets.
- Train and fine-tune the selected models using historical data, optimizing performance metrics such as accuracy and robustness.

5.2.4 Integration and Deployment

- Store and manage inventory data in Amazon RDS (Relational Database Service), ensuring scalability, reliability, and high availability.
- Integrate Amazon Forecast and SageMaker models into the inventory management system, allowing seamless access to demand forecasts.
- Implement AWS Lambda functions to trigger restocking actions based on predictive outputs, automating inventory replenishment processes.

5.2.5 Validation and Performance Monitoring:

- Evaluate the performance of the predictive inventory system using metrics such as forecast accuracy, inventory turnover, and cost savings.
- Conduct regular validation checks to assess the system's effectiveness in minimizing stock-outs and overstock situations.
- Monitor key performance indicators (KPIs) over time and make adjustments to the system as needed to improve efficiency and reliability.

5.2.6 Continuous Improvement and Optimization

- Iterate on the model development and deployment process, incorporating feedback and insights gained from real-world usage.
- Explore additional features, algorithms, or data sources to further enhance the accuracy and effectiveness of inventory forecasts.
- Continuously optimize the system's parameters and configurations to adapt to changing market conditions and business needs.

5.2.7 Python

Purpose: Python serves as the core programming language for Predictive Inventory Cloud, providing a robust foundation for the implementation of prediction of inventory products recognition algorithms and the overall system logic.

Details: The strategic adoption of Python is underpinned by its exceptional readability, versatility, and the wealth of specialized libraries. Python's human-readable syntax facilitates collaborative development, ensuring that the intricacies of Predictive Inventory Cloud's algorithms can be comprehended and extended by developers. The language's versatility proves crucial for Predictive Inventory Cloud's diverse requirements, seamlessly accommodating tasks ranging from image processing to machine learning model implementation and graphical user interface (GUI) development.

Python's extensive array of libraries is instrumental in transforming Predictive Inventory Cloud from a concept to a functional reality. In the realm of image processing, Python leverages libraries like OpenCV and Pandas, providing a robust toolkit for filtering . For machine learning, Python serves as the conduit for leveraging frameworks like flask and algorithm randam forest , enabling the training of intricate models for inventory model as trainingmodel.pkl file.

Moreover, Python's excellence in GUI development is exemplified by the utilization of Flask

in Predictive Inventory Cloud . This ensures the creation of an intuitive, user-friendly interface that seamlessly integrates with the underlying recognition algorithms. Overall, Python empowers Predictive Inventory Cloud to deliver robust predictive capabilities, driving enhanced inventory optimization and business performance.Python's scalability ensures that the system can handle large volumes of data efficiently, crucial for inventory management applications.

5.3 Data Set

5.3.1 Data Exploration

The project harnesses the image dataset available on Kaggle, specifically the "inventory management items " dataset . It consists of a wide range of text across different categories, providing ample diversity for training and testing the model.The historical sales data is loaded from a CSV file named 'sales_ data.csv' into a pandas DataFrame.

- **Import Necessary Libraries**: Import pandas for data manipulation and exploration.
- **Load the Dataset**: Load the dataset into a pandas DataFrame.
- **Data Exploration**: Explore the dataset to understand its structure, columns, and general statistics.
- **Data Preprocessing**: If necessary, preprocess the data, including handling missing values, encoding categorical variables, and performing feature engineering.
- **Process the 10,000 Days**: Slice the DataFrame to include only the 10,000 days of data.Convert the 'date' column to datetime format using pd.to_datetime() for further analysis. Extract features like month, day, and day of the week from the 'date' column using pandas datetime functionality. These features are extracted to enrich the dataset with temporal information that might be relevant for predicting sales.

5.4 Algorithm

A potent machine learning tree learning method is the Random Forest algorithm. It functions by building many Decision Trees in the training stage. A random subset of the data set is used to measure a random subset of characteristics in each partition throughout the construction of each tree. Because of the variety that this randomness brings to the individual trees, the likelihood of overfitting is decreased and the overall prediction performance is enhanced. In prediction, the method combines all tree results, averaging (for problems involving regression) or voting (for tasks

involving classification). With the help of several trees and their insights, this cooperative decision-making process yields findings that are accurate and consistent. Given its reputation for handling complex data, random forests are frequently utilized for regression and classification tasks.

- The Random Forest algorithm is implemented in the provided code as follows:

1. **Import the Necessary Library**: from sklearn.ensemble import RandomForestRegressor
2. **Instantiate the Random Forest Regressor**:model = RandomForestRegressor(n_estimators=100, random_state=42)
   (a) n_estimators specifies the number of decision trees in the forest. Here, it's set to 100.
   (b) random_state ensures reproducibility by fixing the random seed.
3. **Train the Model** :model.fit(X_train, y_train)

   The fit() method is called with the training features (X_train) and target (y_train) to train the Random Forest model.
4. **Make Predictions**:

   Predictions are made on the test set (X_test) using the predict() method of the trained model.

CHAPTER 6

Report of Project Implementation Done

6.1 Implementation

- To implement the predictive inventory management system using the proposed approach, follow these steps:

    1. **Data Collection and Preparation**: Gather historical sales data, including product sales, stock levels, and any relevant factors affecting demand (e.g., seasonality, promotions). Clean and preprocess the data to handle missing values, outliers, and inconsistencies.

    2. **Amazon RDS Setup**: Set up an Amazon Relational Database Service (RDS) instance to store and manage the inventory data. Design the database schema to efficiently store and retrieve inventory-related information.

    3. **Amazon Forecast Integration**: Utilize Amazon Forecast to generate demand forecasts based on historical sales data. Train and fine-tune Forecast models to improve forecast accuracy, considering factors like seasonality and market trends.

    4. **Custom Model Development with Amazon SageMaker**; Use Amazon SageMaker to build custom machine learning models tailored to your specific business needs. Explore advanced forecasting techniques and algorithms to enhance prediction accuracy.

    5. **Integration and Automation with AWS Lambda**: Develop AWS Lambda functions to automate inventory management tasks. Set up triggers to execute Lambda functions based on predictive outputs from Forecast or custom models. Implement logic to initiate restocking actions when inventory levels reach certain thresholds or predicted demand exceeds current stock levels.

    6. **Testing and Validation**: Test the system thoroughly to ensure accurate demand forecasting and timely restocking actions. Validate the system performance against historical data and real-world scenarios.

    7. **Deployment and Monitoring**: Deploy the predictive inventory management system in

a production environment. Monitor system performance, including forecast accuracy, inventory turnover, and cost savings. Implement mechanisms for continuous improvement, such as retraining models with updated data and refining restocking strategies based on feedback.

8. **Training and User Adoption**: Provide training to relevant stakeholders on using the new inventory management system. Encourage user adoption by highlighting the benefits of improved operational efficiency and cost savings.

6.2 Test Case

The goal of testing is to find mistakes. The process of testing includes attempting to find every potential flaw or vulnerability in a piece of work. It offers a means of verifying the operation of parts, assemblies, sub assemblies, and/or the final product It is the act of working out.software with the goal of making sure the system satisfies its specifications and user expectations.Expectations and doesn't perform poorly enough to be unsatisfactory. Different test kinds exist. Every examination type responds to a particular testing need.

- **Test Case for Registration Page**:
  1. **Test Case** :Case1
     - Test Case Name: Successful Registration
     - Description:Verify that a user can successfully register by providing valid registration details.
     - Preconditions: The user is on the registration page.
     - Test Steps:
       (a) Enter valid registration details (e.g., username, email, password).
       (b) Click on the "Register" button.
     - Expected Result: The user is registered successfully, and they are redirected to the login page.
  2. **Test Case** :Case2
     - Test Case Name:Registration with Existing Email

- Description: Verify that a user cannot register with an email address that is already associated with an existing account.
- Preconditions: The user is on the registration page.
- Test Steps:
  (a) Enter registration details with an email address that is already registered.
  (b) Click on the ”Register” button.
- Expected Result: The system displays an error message indicating that the email address is already in use, and the user is not registered.

3. **Test Case** : Case3
   - Test Case Name:Empty Fields Validation
   - Description:Verify that the system validates and prompts the user to fill in all required fields before registration.
   - Preconditions:The user is on the registration page.
   - Test Steps:
     (a) Submit the registration form without filling in any fields.
   - Expected Result:The system displays error messages next to each empty field, prompting the user to fill in the required information.

- **Test Case for Login Page**:

1. **Test Case** :Case1
   - Test Case Name: Successful Login
   - Description:Verify that a registered user can successfully log in using valid credentials.
   - Preconditions: The user is on the login page.
   - Test Steps:
     (a) Enter valid login credentials (e.g., email, password).
     (b) Click on the ”Login” button.

- Expected Result:The user is logged in successfully and redirected to the homepage/dashboard.

2. **Test Case** :Case2
   -
   - Test Case Name:Incorrect Password
   - Description: Verify that the system does not allow login with incorrect password.
   - Preconditions: The user is on the login page.
   - Test Steps:
     (a) Enter a registered email address with an incorrect password.
     (b) Click on the ”Login” button.
   - Expected Result: The system displays an error message indicating that the password is incorrect, and the user is not logged in.

3. **Test Case** :Case3
   - Test Case Name: Login with Non-existing Email
   - Description:Verify that the system does not allow login with an email address that is not registered.
   - Preconditions: The user is on the login page.
   - Test Steps:
     (a) Enter a non-existing email address.
     (b) Enter a valid password.
     (c) Click on the ”Login” button.
   - Expected Result: The system displays an error message indicating that the email address is not registered, and the user is not logged in.

CHAPTER 7

RESULT AND CONCLUSION

7.1 Result

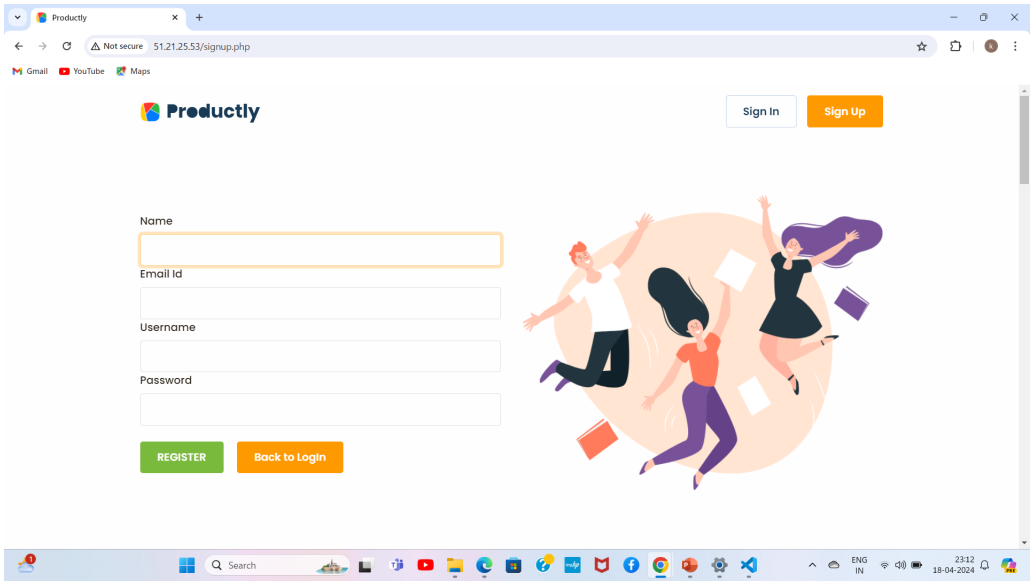


Figure 7.1: Registeration Page

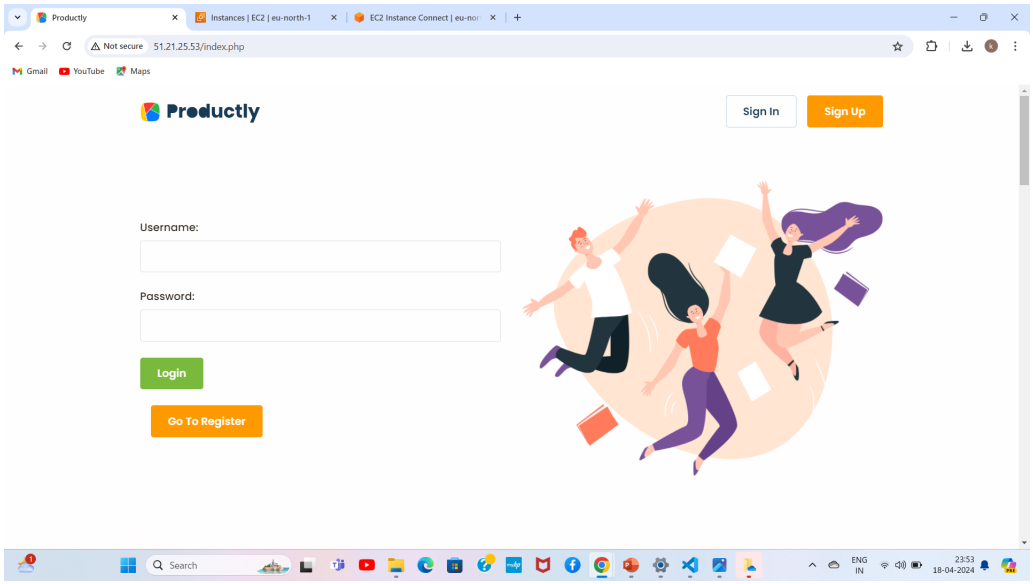


Figure 7.2: Login Page

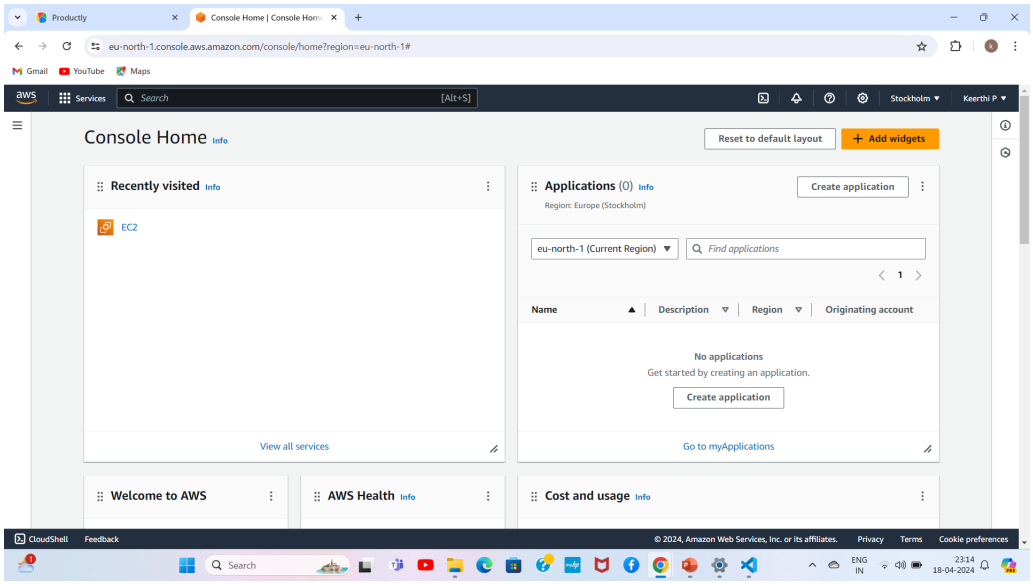


Figure 7.3: AWS Console Page

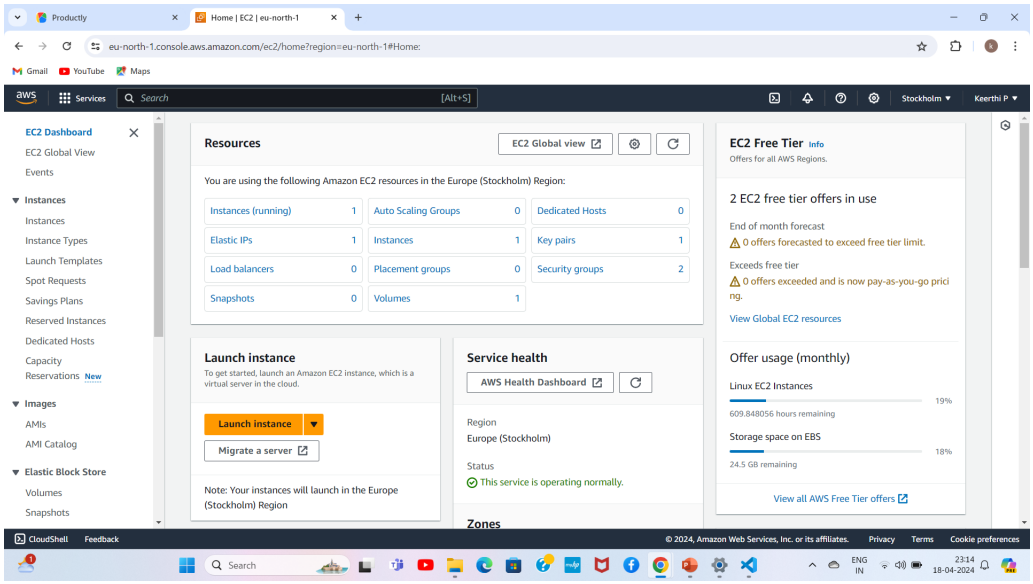
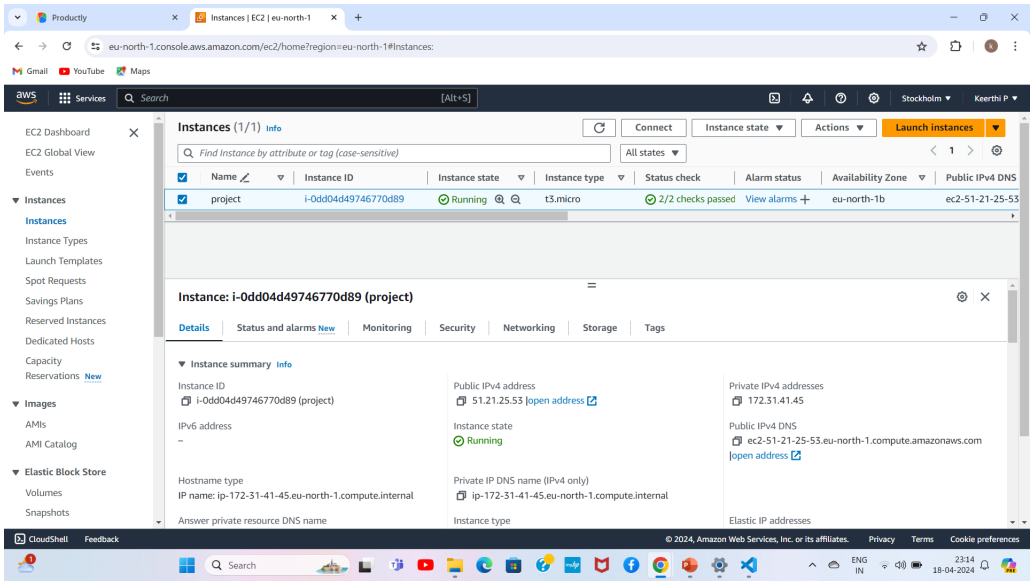


Figure 7.4: Instance Page



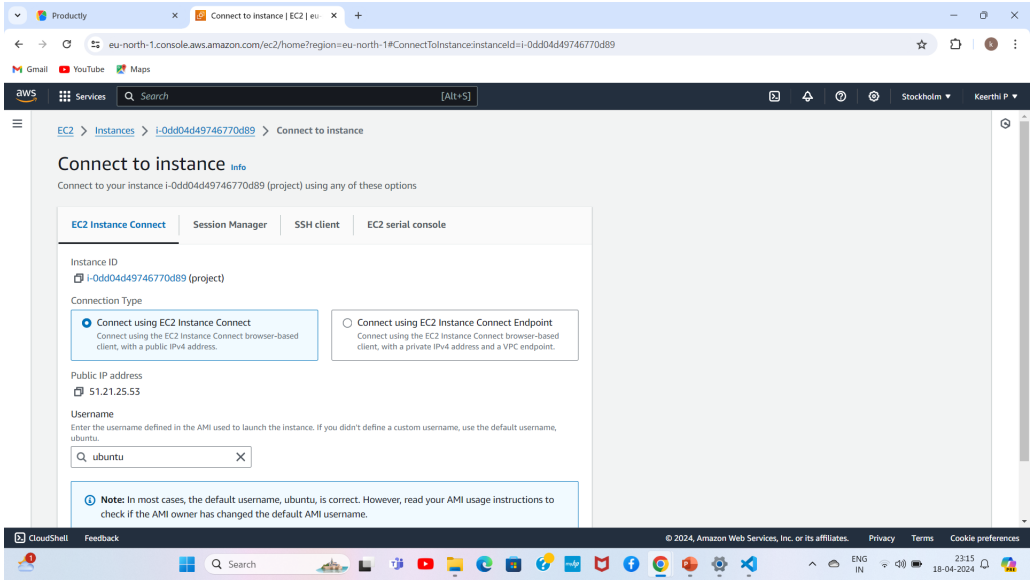


Figure 7.5: Connetion page

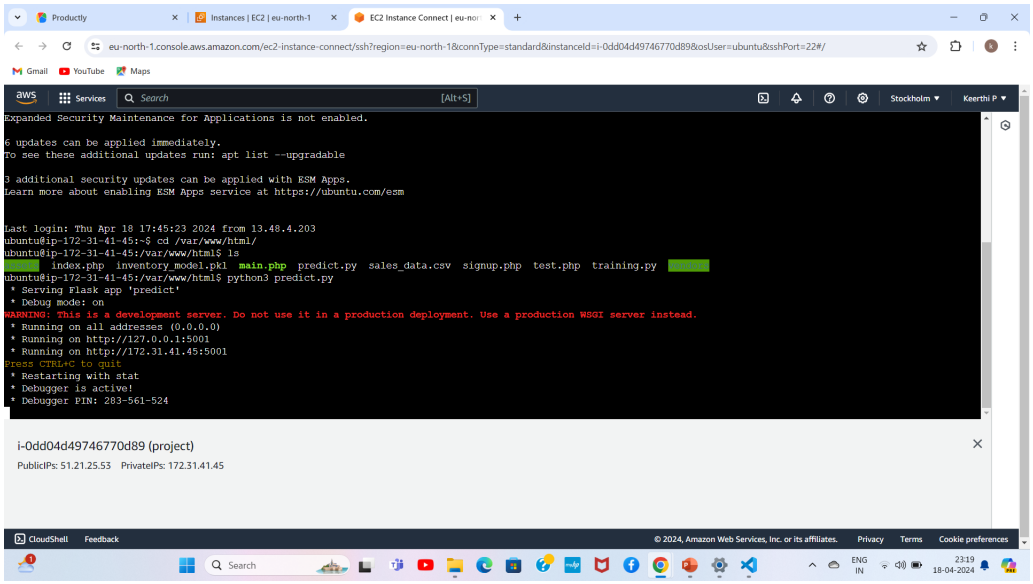


Figure 7.6: Prediction Running In Linux

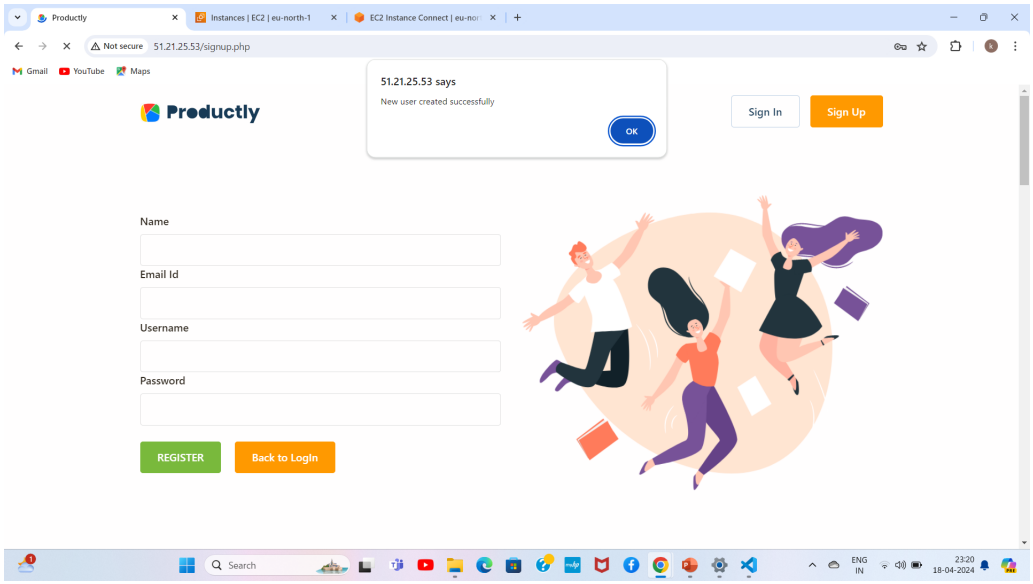


Figure 7.7: Registered successfully

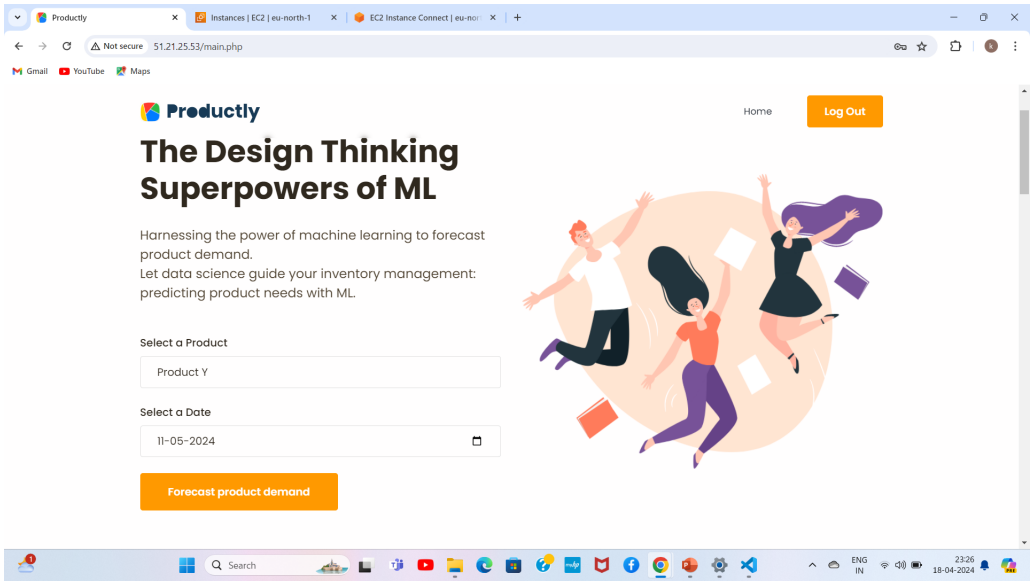


Figure 7.8: Home Page

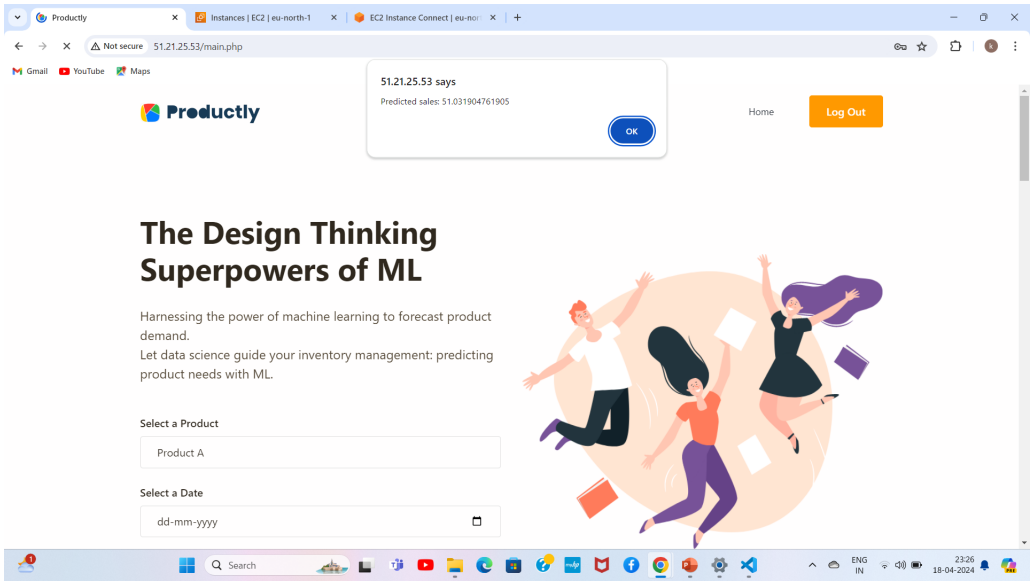


Figure 7.9: Forecasting

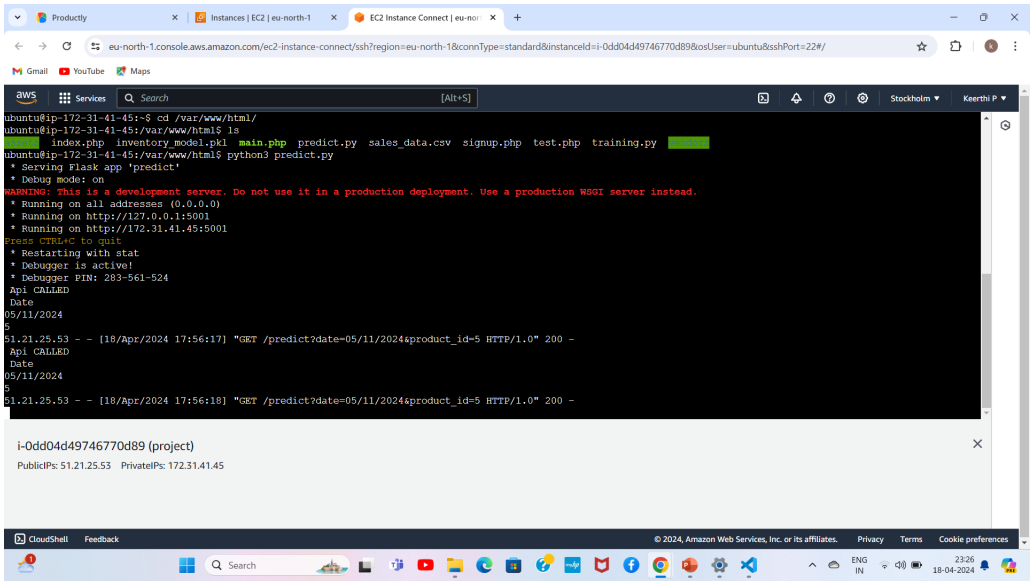


Figure 7.10: Successfully Predicted

7.2 GitHub

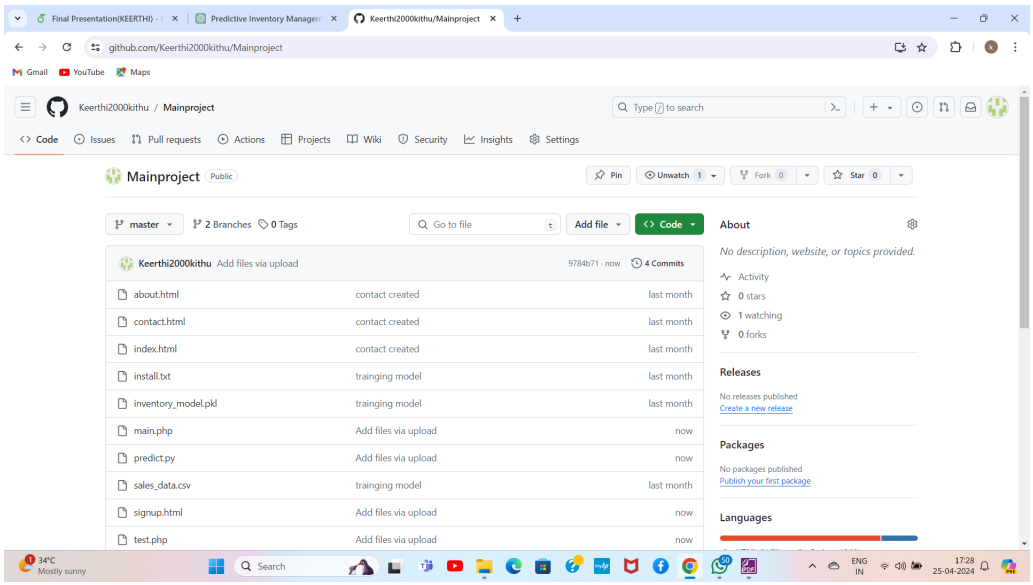


Figure 7.11: GitHub History

7.3 Conclusion

The project has successfully demonstrated the feasibility of using machine learning techniques to Aims to minimize stockouts and overstock situations.The predictive inventory management system harnesses the power of machine learning and AWS services to revolutionize inventory management practices. By leveraging historical sales data, seasonality patterns, and market trends, the system generates accurate demand forecasts using Amazon Forecast. Additionally, custom models developed with Amazon SageMaker enhance the precision of predictions, tailored to the unique requirements of the business.

Storing and managing inventory data in Amazon RDS ensures high availability and reliability, crucial for seamless operations. The integration of AWS Lambda functions automates restocking actions, triggered by predictive outputs, thereby minimizing stockouts and overstock situations.

Ultimately, this approach promises to optimize stock levels, improve operational efficiency, and drive cost savings. By proactively managing inventory based on data-driven insights, businesses can enhance customer satisfaction, streamline operations, and stay ahead in today's dynamic market landscape.

7.4 Future Scope

- **Resource Management Integration**: Integrate the system with AWS Resource Groups to optimize resource allocation based on predicted inventory demands. This ensures that computational resources are dynamically allocated and scaled according to the workload, maximizing efficiency and cost-effectiveness.
- **Mobile Application Development**: Develop a mobile application for inventory managers and staff to access real-time inventory data, receive alerts on low stock levels or predicted surges in demand, and initiate restocking actions remotely. This enhances flexibility and responsiveness in inventory management, enabling proactive decision-making on the go.
- **CRM Tool Integration**: Integrate with a Customer Relationship Management (CRM) tool, such as Salesforce or HubSpot, to correlate inventory data with customer interactions and purchasing patterns. By analyzing customer behavior and preferences, the system can further refine demand forecasts and tailor inventory management strategies to enhance customer satisfaction and retention.
- **Predictive Maintenance**: Extend the predictive capabilities of the system beyond inven-

tory forecasting to incorporate predictive maintenance for equipment and machinery involved in the supply chain and warehouse operations. By analyzing historical maintenance data and equipment telemetry, the system can anticipate potential failures and schedule preventive maintenance tasks, minimizing downtime and optimizing asset utilization.

REFERENCES

[1] M. I. Fernández, P. Chanfreut, I. Jurado and J. M. Maestre, "A Data-Based Model Predictive Decision Support System for Inventory Management in Hospitals," in IEEE Journal of Biomedical and Health Informatics, vol. 25, no. 6, pp. 2227-2236, June 2021, doi: 10.1109/JBHI.2020.3039692.

[2] P. Hajek and M. Z. Abedin, "A Profit Function-Maximizing Inventory Backorder Prediction System Using Big Data Analytics," in IEEE Access, vol. 8, pp. 58982-58994, 2020, doi: 10.1109/ACCESS.2020.2983118.

[3] A. Mishra and M. Mohapatro, "Real-time RFID-based item tracking using IoT And efficient inventory management using Machine Learning," 2020 IEEE 4th Conference on Information And Communication Technology (CICT), Chennai, India, 2020, pp. 1-6, doi: 10.1109/CICT51604.2020.9312074.

[4] https://www.kaggle.com/datasets/fayez1/inventory-management

[5] https://docs.python.org/3/library/index.html

[6] Y. Fan, "Development of inventory management system," 2010 2nd IEEE International Conference on Information Management and Engineering, Chengdu, China, 2010, pp. 207-210, doi: 10.1109/ICIME.2010.5478077.

[7] F. Badilini, T. Erdem, W. Zareba and A. J. Moss, "ECGScan: A method for conversion of paper electrocardiographic printouts to digital electrocardiographic files", *J. Electrocardiol., vol. 38, no. 4, pp. 310-318, 2005.*

[8] M. Aazam and E. N. Huh, "Inter-cloud Media Storage and Media Cloud Architecture for Inter-cloud Communication," 2014 IEEE 7th International Conference on Cloud Computing, Anchorage, AK, USA, 2014, pp. 982-985, doi: 10.1109/CLOUD.2014.151.

[9] A. Jayaram, "An IIoT quality global enterprise inventory management model for automation and demand forecasting based on cloud," 2017 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, India, 2017, pp. 1258-1263, doi: 10.1109/CCAA.2017.8230011.

[10] W. Li, Z. Lin, X. Zhang and J. Zhou, "Research on Distributed Logistics Inventory Model

Based on Cloud Computing," 2016 12th International Conference on Computational Intelligence and Security (CIS), Wuxi, China, 2016, pp. 73-77, doi: 10.1109/CIS.2016.0025.

[11] K. Jakkraphobyothin, S. Srifa and T. Chinda, "Factor Analysis of Inventory Management in Thai Construction Industry," 2018 3rd Technology Innovation Management and Engineering Science International Conference (TIMES-iCON), Bangkok, Thailand, 2018, pp. 1-5, doi: 10.1109/TIMES-iCON.2018.8621817.

[12] S. Bailkar, K. Shenoy, A. Bedekar, S. Bankar and P. More, "Smart Inventory Optimization using Machine Learning Algorithms," 2024 2nd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT), Bengaluru, India, 2024, pp. 1395-1400, doi: 10.1109/IDCIoT59759.2024.10467512.