CLCP-1200-02 - Capstone

Final Capstone Project Report

Submitted by Team 05:

Shijin Joseph	100913453
Sampathraja Thulasiraja	100900613
Kiran Sajan	100853913

Instructor:

Kabir Sultanul

Date: 13th April 2024

1. Introduction

In the intricate retail environment of today, where the digital and physical spheres collide, price is a critical factor in a company's ability to succeed or fail. Introducing the Smart Price Retail Platform, an initiative that blends modern technology with the traditional goal of achieving business excellence. Our mission is to use cloud computing and machine learning to transform retail pricing optimization and provide retailers the knowledge they need to prosper in a time of fast change.

The Smart Price Retail Platform is more than simply a piece of technology. We're witnessing a shift in how retailers approach pricing strategies, embracing a thorough, data-driven approach. Our aim is to empower merchants of all scales with the necessary tools to navigate today's marketplace with confidence and strategy. Our research began with recognizing the challenges merchants encounter in a fiercely competitive and digitally transformed environment. To stay competitive, merchants must reassess their pricing strategies amidst the dominance of e-commerce giants and the influence of social media on consumer habits. Our objective is to address these challenges by leveraging the latest advancements in data analytics and technology.

We have created a thorough framework for pricing optimisation by using a wide range of research and observations. Through investigating developments in data analytics, machine learning, and cloud computing, we hope to develop a solution that not only addresses retailers' present needs but also foresees their future opportunities and challenges.

We are aiming to transform pricing tactics in the retail industry by utilising cutting edge methods like the **Random Forest Regression** algorithm for accurate model prediction. (builtin.com/random-forest-algorithm, n.d.) The foundation of our predictive analytics system is this algorithm, which is well-known for its accuracy and adaptability, and which allows us to estimate ideal retail product prices with unmatched accuracy. Furthermore, our project demonstrates a dedication to innovative infrastructure deployment techniques, as demonstrated using **Terraform scripts**. These scripts help build the foundation for a reliable and scalable cloud environment by making it easier to create crucial infrastructure elements like load balancers, EC2 instances, Virtual Private Clouds (VPCs), and S3 buckets. (developer.hashicorp.com, n.d.)

Our project tackles themes of technology innovation and societal advancement, going beyond the retail industry. Our aim is to establish a retail environment that is more competitive and focused on the needs of consumers by enabling businesses to make educated decisions and democratising access to price knowledge. Our goal as we set out on this adventure is to use technology to transform retail in a positive way. With the help of our initiative, which combines entrepreneurship and knowledge, we may influence the direction of commerce in the future by combining quality and smart pricing, creating countless opportunities for expansion and innovation.

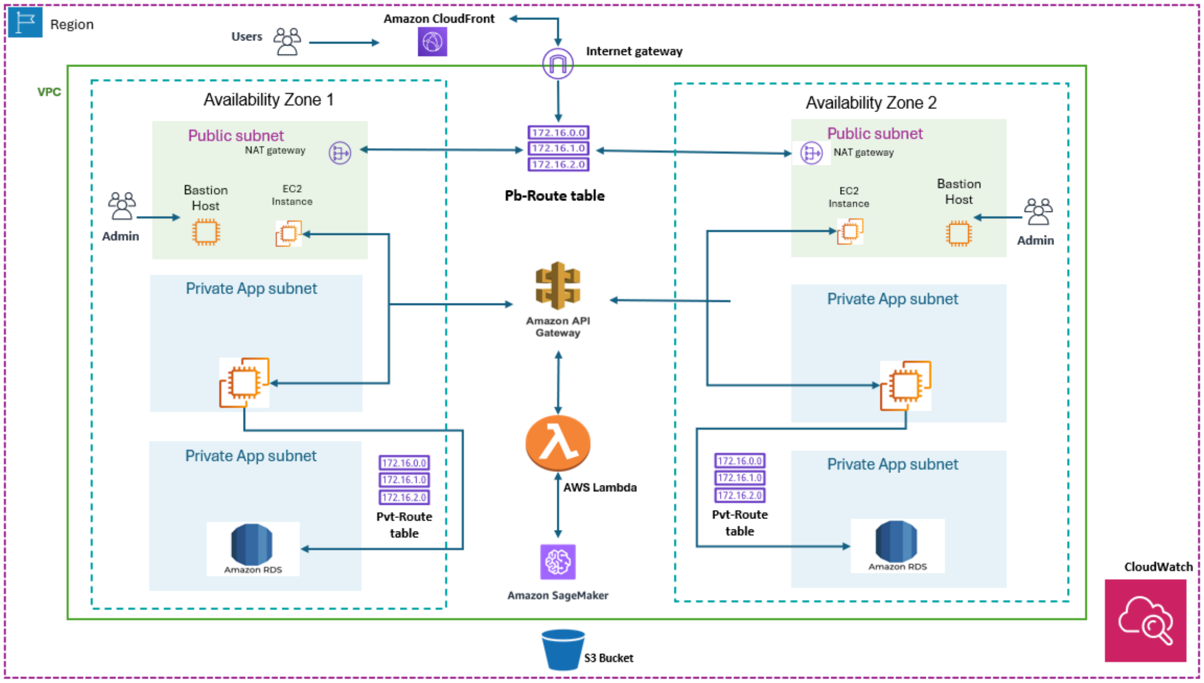
2. Background

Our project, the Smart Price Retail Platform, was inspired by several things that highlight how crucial pricing optimisation is in today's retail environment. Our deep-seated Understanding of the complex issues retailers faces in a time of digital disruption, fierce Rivalry, and changing consumer preferences drove us to take on this specific project. Before starting the project, we conducted preliminary research and background checks, Revealing a stark reality: traditional pricing strategies were no longer sufficient to keep up With the demands of today's rapidly evolving economy. Retailers are grappling with a more Intricate pricing landscape due to the rise of e-commerce giants, the proliferation of price Comparison tools, and the expanding influence of social media on consumer choices.

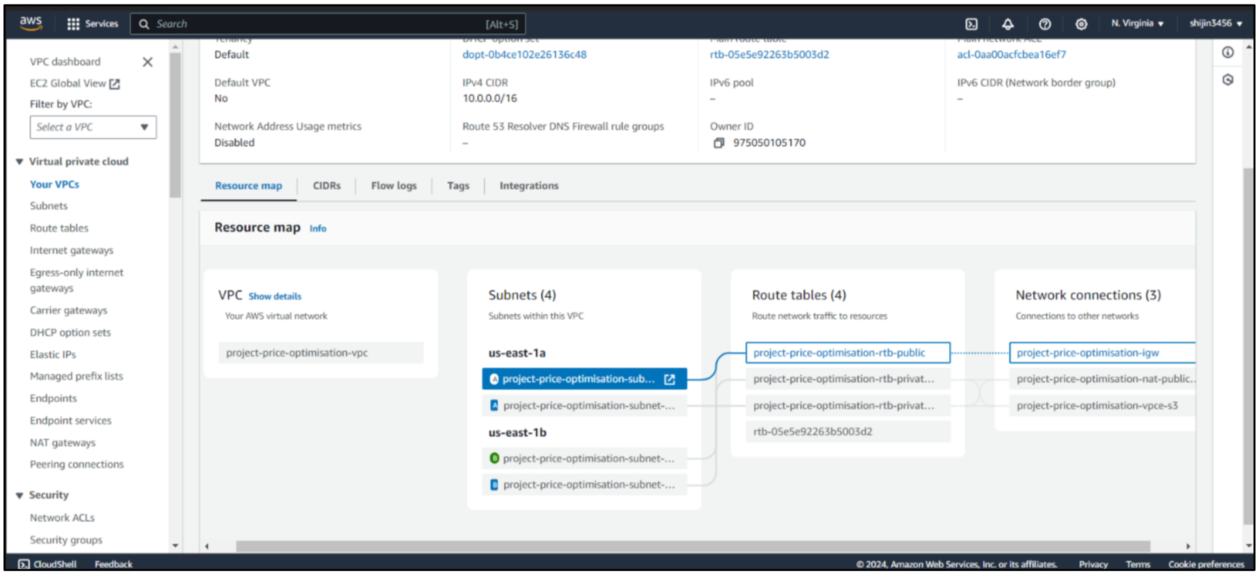
Experiences and personal insights further demonstrated how urgent it is to address this urgent Problem. We were deeply aware of the revolutionary potential of using cutting-edge Technologies like cloud computing and machine learning to address persistent issues in the Retail industry because we are folks with a strong interest in both technology and business. Our project has two main objectives: firstly, to effectively harness emerging technologies to Unlock new sources of value and gain a competitive edge; and secondly, to encourage Retailers to adopt more flexible and data-driven pricing strategies to keep pace with a rapidly Changing marketplace. Through the creation of the Smart Price Retail Platform, our aim is to Equip retailers with the knowledge and tools necessary to navigate today's complex retail Landscape with confidence and strategic insight.

Our decision to undertake this project was driven by a strong belief in the transformative Potential of technology, as well as a blend of personal insights and empirical evidence. Beyond simply enhancing retailers' competitiveness and profitability, we also seek to Contribute to broader discussions on innovation and entrepreneurship in the digital age by Addressing the pressing need for pricing optimization within the retail sector.

3. Architecture Design



Well Architected Framework:



Default

dopt-0b4ce102e26136c48

rtb-05e5e92263b5003d2

acl-0aa00acfcbea16ef7

Default VPC
No

IPv4 CIDR
10.0.0.0/16

IPv6 pool
–

IPv6 CIDR (Network border group)
–

Network Address Usage metrics
Disabled

Route 53 Resolver DNS Firewall rule groups
–

Owner ID
975050105170

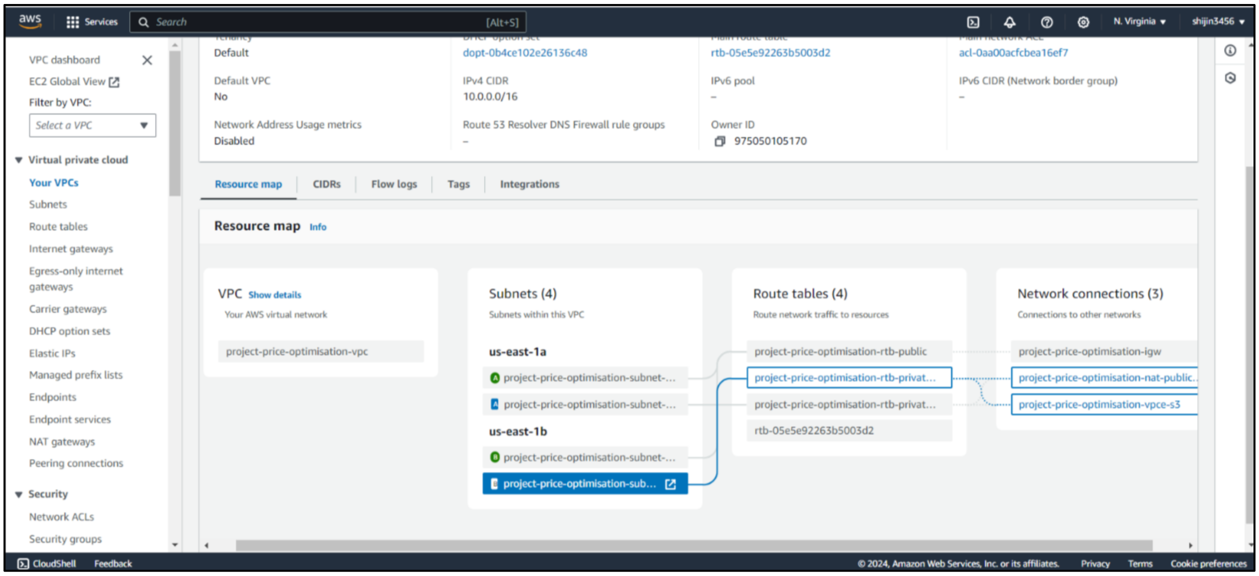
Resource map Info

VPC Show details
Your AWS virtual network
project-price-optimisation-vpc

Subnets (4)
Subnets within this VPC
us-east-1a
project-price-optimisation-sub...
project-price-optimisation-subnet-...
us-east-1b
project-price-optimisation-subnet-...
project-price-optimisation-subnet-...

Route tables (4)
Route network traffic to resources
project-price-optimisation-rtb-public
project-price-optimisation-rtb-privat...
project-price-optimisation-rtb-privat...
rtb-05e5e92263b5003d2

Network connections (3)
Connections to other networks
project-price-optimisation-igw
project-price-optimisation-nat-public...
project-price-optimisation-vpce-s3



Default

dopt-0b4ce102e26136c48

rtb-05e5e92263b5003d2

acl-0aa00acfcbea16ef7

Default VPC
No

IPv4 CIDR
10.0.0.0/16

IPv6 pool
–

IPv6 CIDR (Network border group)
–

Network Address Usage metrics
Disabled

Route 53 Resolver DNS Firewall rule groups
–

Owner ID
975050105170

Resource map Info

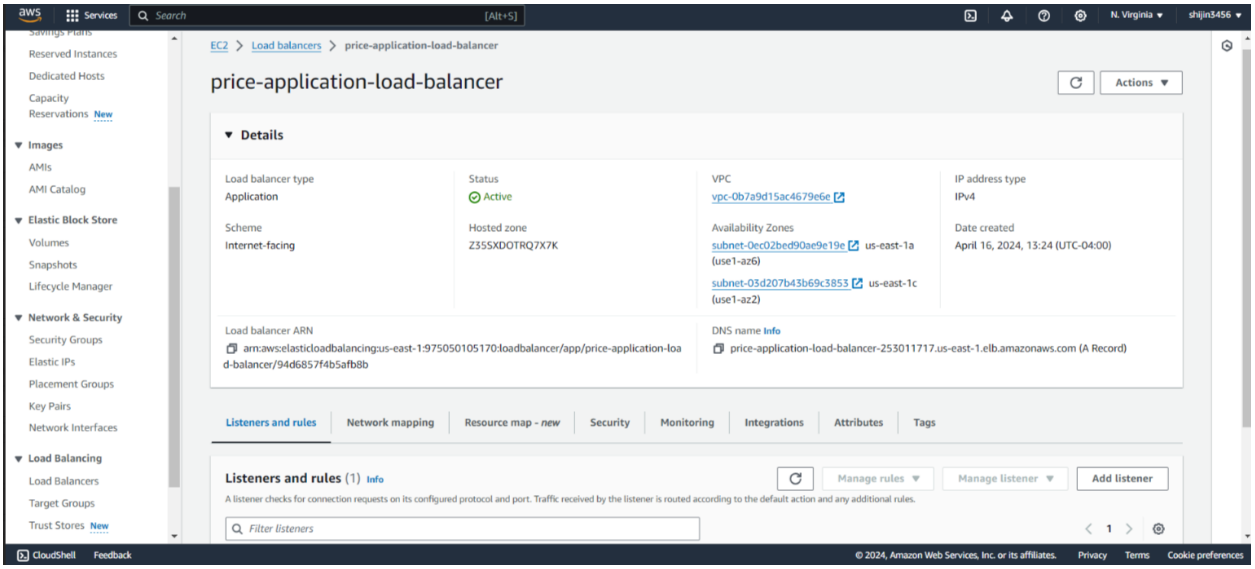
VPC
Your AWS virtual network
project-price-optimisation-vpc

Subnets (4)
Subnets within this VPC
us-east-1a
project-price-optimisation-subnet-...
project-price-optimisation-subnet-...
us-east-1b
project-price-optimisation-subnet-...
project-price-optimisation-sub...

Route tables (4)
Route network traffic to resources
project-price-optimisation-rtb-public
project-price-optimisation-rtb-privat...
project-price-optimisation-rtb-privat...
rtb-05e5e92263b5003d2

Network connections (3)
Connections to other networks
project-price-optimisation-igw
project-price-optimisation-nat-public...
project-price-optimisation-vpce-s3

Load Balancer:



EC2 > Load balancers > price-application-load-balancer

price-application-load-balancer

Details

Load balancer type
Application

Status
Active

VPC
vpc-0b7a9d15ac4679e6e

IP address type
IPv4

Scheme
Internet-facing

Hosted zone
Z35SXDOTRQ7X7K

Availability Zones
subnet-0ec02bed90ae9e19e us-east-1a (use1-az6)
subnet-03d207b43b69c3853 us-east-1c (use1-az2)

Date created
April 16, 2024, 13:24 (UTC-04:00)

Load balancer ARN
arn:aws:elasticloadbalancing:us-east-1:975050105170:loadbalancer/app/price-application-load-balancer/94d6857f4b5afb8b

DNS name Info
price-application-load-balancer-253011717.us-east-1.elb.amazonaws.com (A Record)

Listeners and rules (1) Info

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

4. Gap Analysis

Currently, the system lacks the capability to predict prices for multiple products simultaneously and integrate them seamlessly with the webpage, hindering user experience and limiting functionality. Additionally, there is a deficiency in showcasing ML model accuracy and performance metrics due to the absence of a robust backend deployment strategy. Real-time price monitoring is also missing from the system architecture. Furthermore, the website's design and functionality require enhancement to improve user engagement and satisfaction.

Proposed Solutions: To address these gaps, it is imperative to develop and implement backend solutions capable of supporting multiple product price prediction and seamless integration with the webpage. Establishing a robust deployment pipeline will facilitate showcasing ML model accuracy and performance metrics, while integrating real-time data monitoring mechanisms will enable timely updates of pricing information. Additionally, a website redesign and enhancement process should be undertaken to improve user experience and aesthetics, ensuring optimal engagement and usability.

There was insufficient capacity to manage rising loads, an uneven distribution of traffic during high usage hours, and the possibility of single points of failure due to inadequate availability zones, scalability mechanisms, and load balancing capabilities in the AWS architecture.

We have added: The AWS architecture now includes more availability zones to improve resilience and fault tolerance. To ensure optimal performance and availability, auto-scaling groups and elastic load balancing have been implemented to dynamically modify resources based on demand. By distributing traffic among several instances and zones, load balancers reduce the chance of outage and performance degradation while enhancing system scalability and dependability. The availability, performance, and worldwide reach of the system are further improved by utilising AWS services like Amazon CloudFront for content delivery and Amazon Route 53 for DNS administration.

Strategic Alliances:

Current State: Strategic alliances with regional distributors, merchants, and payment processors are not currently established.
Desired State: Establishing strategic alliances with local partners can facilitate market expansion into new territories.
Gap: Lack of strategic alliances with regional partners is a gap that needs to be addressed to support international growth and market penetration.

Closing these gaps will enable the platform to effectively target international markets, capitalize on opportunities, and achieve its expansion goals.

5. Cloud Implementation

The implementation process of the Smart Price Retail Platform involved a meticulous integration of various cloud services and resources to create a robust and scalable infrastructure capable of supporting our pricing optimization system. This section delves into the technical aspects and development stages of our project, focusing on the cloud implementation.

Cloud Resources

Our project leveraged several key cloud services provided by Amazon Web Services (AWS) to build and deploy the Smart Price Retail Platform. These services included:

1. **Amazon EC2 (Elastic Compute Cloud):** To host the application components and supply processing power for our machine learning algorithms and backend services, we employed EC2 instances. (aws.amazon.com/ec2, n.d.)
2. **Amazon S3 (Simple Storage Service):** S3 was essential in helping us store backup data, machine learning models, datasets, and other pertinent items that our application needed. It provided low-latency data access, scalability, and durability.
3. **AWS Lambda**: Lambda functions were used to run code in response to events that were set off by modifications to the system, like data processing jobs or real-time price updates. By offering a serverless computing environment, they maximised resource usage and did away with the necessity for server management.
4. **Amazon SageMaker**: Machine learning models that forecast the best retail pricing were developed, trained, and implemented using SageMaker. It provided a managed platform with machine learning model building tools and algorithms integrated in, cutting down on complexity and development time. (aws.amazon.com, n.d.)
5. **API Gateway** : To provide secure and scalable access to our application's functionalities, we chose API Gateway as our primary entry point for client requests. API Gateway offers capabilities like request/response transformation, caching, and logging in addition to making it simple to develop, publish, and manage APIs. We can make sure that users may safely and smoothly interact with our application by utilising API Gateway. (docs.aws.amazon.com, n.d.)

6. Configuration and Deployment:

The process of configuring our cloud environment and deploying the Smart Price Retail Platform involved several steps:

- **Setting up EC2 Instances**: Based on the needs of our application, we chose the instance types and configurations and started by allocating the necessary computing resources to EC2 instances. In order to enable our backend services and machine learning algorithms, we installed the required software dependencies and libraries.
- **Uploading Data to S3**: After applying the proper access permissions and classifying the datasets, machine learning models, and other files into logical folders, we uploaded them to S3 buckets. To effectively manage data retention and storage expenses, we employed S3 lifecycle policies.

i. **Training and Deploying Models with SageMaker**: We trained machine learning models with SageMaker by utilizing past price information and product attributes. We chose the best algorithms, adjusted the hyperparameters, and used the evaluation metrics that were already included in the model to assess its performance. After the models were trained, we made them available as endpoints through RESTful APIs. (www.run.ai, n.d.)

ii. **Developing Lambda Functions**: To manage tasks like data pretreatment, model inference, and real-time price updates, we created Lambda functions. We utilized Python, along with other supported programming languages, to write the Lambda function code. Event triggers were configured to automatically invoke these functions based on predefined conditions.

iii. **Creating API gateway**: API Gateway served as the gateway between our frontend and backend components, facilitating seamless communication and data exchange. We set up RESTful APIs within API Gateway to create endpoints that enable access to our backend services. This setup allows our frontend components to communicate with our application. By connecting API Gateway with our backend services, we've built a strong and scalable architecture that can efficiently manage client requests while maintaining security and reliability. Since our frontend components are hosted on a different domain than our backend services, we needed to enable CORS to permit cross-origin requests. Without CORS, browsers restrict cross-origin requests due to security policies, potentially leading to issues such as blocked requests or security errors.

    a. To enable CORS, we configured CORS settings within API Gateway, which acts as the entry point for client requests to our backend services. This configuration involved specifying allowed origins (domains), methods, headers, and other parameters to control access to our APIs from different origins. (docs.aws.amazon.com/AmazonS3/latest, n.d.)

Development stages

1. Data Setup and Model Training:

Action: We began by setting up an S3 bucket to securely store our project data and uploaded the necessary CSV file containing price optimization data. Following this, leveraging AWS SageMaker, we imported a sample Jupyter notebook to train a predictive model capable of optimizing prices. This involved creating a SageMaker notebook instance, uploading the notebook file, and executing it to train the model. (www.kaggle.com, n.d.)Lamb

Achievement: Through meticulous data organization and model training, we laid the foundation for accurate price optimization.

2. Training SageMaker Model:

Action: We trained a robust machine learning model using SageMaker's advanced capabilities, enabling it to predict optimal prices based on historical pricing data and product attributes. Through experimentation, we identified the most effective methods, fine-tuned the model's parameters, and evaluated its performance using metrics like R^2 Score, Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Squared Error (MSE).

Our achievement speaks volumes: the model demonstrated exceptional performance, evidenced by a test MSE of 0.0577, MAE of 0.0747, RMSE of 0.2402, and an impressive

R^2 Score of 0.9997. These results underscore the model's remarkable accuracy in predicting prices.

3. **Website Deployment and User Interaction:**

Action: Deployed the website on an EC2 instance, ensuring high availability and security using load balancing and auto-scaling features across two availability zones. Users interacted with the website by inputting product attributes, which were then sent to a REST API via API Gateway for price prediction.

Achievement: Provided users with a secure and responsive platform to interact with our price optimization system, facilitating seamless user experience.

4. **Lambda Function Development:**

Action: Developed Lambda functions to manage tasks like data preprocessing, model inference, and real-time price updates. The Lambda functions, written in Python, processed user input data and invoked the SageMaker endpoint to generate price predictions. (docs.aws.amazon.com, n.d.)

Achievement: Enabled efficient execution of backend logic and real-time prediction capabilities, enhancing system responsiveness and scalability.

5. **API Gateway Configuration:**

Action: Setup the API Gateway to handle communication between the frontend and backend services and serve as the point of entry for frontend queries. Cross-Origin Resource Sharing, or CORS, policies were put into place to enable safe communication between the frontend and backend housed on separate domains.

Achievement: Achieved secure and effective data sharing by facilitating a smooth interface between the frontend and backend components.

6. **CORS Implementation:**

Action: To enable secure cross-origin communication between the frontend and backend services, CORS policies were implemented. This required setting up API Gateway so that it could safely allow browsers to make cross-origin queries by include the necessary CORS headers in answers to frontend requests.

Achievement: Minimised potential security risks related to cross-origin requests by ensuring secure and regulated access to backend resources from the frontend.

7. Challenges and Resolutions

During the implementation process, we encountered several challenges, including:

1. **Security and Compliance**: Security and compliance were paramount concerns during the implementation phase. Safeguarding sensitive customer and financial data required stringent measures to ensure data security and regulatory adherence. At both the infrastructure and application levels, we implemented robust auditing procedures,

enforced access restrictions, and employed encryption protocols to address these challenges effectively.

a. **Scalability and Performance**: Handling fluctuating workloads and traffic surges while achieving scalability and optimal performance presented another difficulty. We used CloudWatch to track system performance, optimised resource allocation, and auto-scaling strategies to dynamically change capacity in response to demand.
b. **Integration and Compatibility**: Compatibility between components and the integration of different cloud services presented integration issues. We overcame these difficulties by utilising AWS SDKs and APIs for smooth integration, adhering to best practices, and carrying out extensive testing.

We successfully designed and launched the Smart Price Retail Platform on the AWS cloud by working together, being precise with our planning, and solving problems. This laid the groundwork for a scalable, dependable, and effective pricing optimisation system. A preview of our cloud resource dashboard, including the several services and configurations we used in our implementation, may be seen in the screenshots below.

7. Learning Outcome

We have gained a wide range of knowledge and abilities in the technical, managerial, and interpersonal domains over the course of our capstone project. These learning outcomes have been crucial to our personal and professional development.

1. **Development of Technical Skills**: Working together, we refined our technical abilities in a variety of software development, machine learning, and cloud computing domains. We became proficient in setting up and utilising cloud services via platforms such as Amazon Web Services (AWS), where we learned about topics like EC2 instances, Lambda functions, and SageMaker models. We also explored the nuances of model training approaches, data pretreatment strategies, and machine learning algorithms, arming us with the knowledge and skills required to create predictive pricing models. Furthermore, by using frontend and backend development frameworks, we were able to create dependable backend systems and user interfaces that merged well with our cloud infrastructure.
2. **Managerial Skills Enhancement**: Through efficient project planning, coordination, and organisation, each team member improved their managerial abilities on an individual basis. In order to meet project milestones and deliverables, we learnt how to clearly define objectives, create schedules, and manage resources effectively. We also developed the capacity to prioritise work, reduce risks, and adjust to changing needs, exhibiting resilience and adaptability in the face of difficulties. Our joint efforts promoted a climate of accountability, cooperation, and communication that made project execution easier and guaranteed stakeholder expectations were met.
3. **Cultivation of Interpersonal Skills**: Our project gave participants lots of chances to hone crucial soft skills including leadership, teamwork, and communication. We promoted open communication, helpful criticism, and mutual support through frequent team meetings, and brainstorming sessions. This helped to build a climate of trust and respect within the team.
4. **Insights into Problem-Solving**: One of the most important lessons learned from our project is the value of approaching problem-solving methodically and iteratively. We

gained knowledge on how to deconstruct difficult problems into smaller, more doable jobs, examine several options before settling on the best one. Furthermore, the practical limitations and real-world circumstances we encountered emphasised the importance of creativity, critical thinking, and resourcefulness in solving complex challenges.

By embracing a growth mindset and recognizing failure as an opportunity to learn, we effectively tackled obstacles, honed our solutions, and ultimately achieved our project objectives.

Our capstone project has been truly transformative, equipping us with a diverse skill set, invaluable experiences, and profound insights into creativity, problem-solving, and teamwork dynamics. As we embark on new professional endeavors, we carry the lessons learned and connections forged during this enriching experience, confident in our ability to thrive in demanding, high-pressure environments.

8. FUTURE SCOPE

With a view to the future, our project, the Smart Price Retail Platform, has a great deal of potential for improvements and growth, opening the door for further creativity and flexibility in response to changing market conditions. The following are a few directions for further advancement and development:

1. **Improved Predictive Models**: One significant area for future development is enhancing and expanding the predictive models used by the platform. This can be achieved by employing advanced machine learning algorithms and incorporating additional data sources such as sentiment analysis from social media, pricing data from competitors, and macroeconomic indicators. Moreover, integrating techniques like ensemble learning and deep learning frameworks could further improve the platform's forecasting capabilities. These enhancements would enable merchants to make more accurate predictions regarding market trends and consumer preferences.

2. **Dynamic Pricing techniques:** Using increasingly complex dynamic pricing techniques is a viable path forward for development. Retailers have the ability to adjust prices in real-time based on various factors such as changes in demand, inventory levels, seasonal patterns, and competitor strategies. This is made possible through the use of real-time data feeds and advanced optimization algorithms. Additionally, personalized pricing algorithms tailored to specific customer segments can be incorporated to maximize revenue opportunities and enhance customer engagement and loyalty.

3. **Expansion of Platform Functionality**: The platform has the potential to expand by incorporating new modules and features to cater to the diverse requirements of merchants across different markets and industries. By integrating modules such as demand forecasting, inventory management, and CRM systems, retailers can gain comprehensive visibility and control over their operations. Furthermore, the inclusion of analytics dashboards and reporting tools allows merchants to derive valuable insights from their data, facilitating informed decision-making and strategic planning.

4. **Scalability and Adaptability**: It's essential for the platform to remain adaptable and scalable to accommodate the evolving dynamics of the retail industry. This involves investing in robust architecture and infrastructure capable of supporting growing user bases, increasing data volumes, and emerging technologies. Additionally, by adopting flexible and modular design principles, the platform can future-proof itself against technological advancements and ensure compatibility with third-party systems and services.

5. **Geographic Expansion and Localization**: The platform has the potential to adapt to different languages, currencies, and legal regulations to capitalize on opportunities in international markets. This involves conducting market research and implementing localization efforts to tailor the platform to the specific needs and preferences of diverse geographic regions and cultural demographics. Additionally, forming strategic partnerships with local distributors, retailers, and payment processors can facilitate market expansion into new territories previously untouched.

9. CONCLUSION

After a journey filled with creativity, teamwork, and an unwavering pursuit of quality, the Smart Price Retail Platform has emerged as a ground-breaking solution in the field of retail pricing optimisation. Our project has gone through several stages, each of which has contributed to its development and improvement, from conception to realisation as a fully functional system.

The project's primary driving force was a deep understanding of the difficulties merchants confront in navigating the intricacies of the contemporary economy. We determined the urgent need for a comprehensive pricing optimisation solution that took advantage of the game-changing capabilities of cloud computing and machine intelligence through preliminary research and background studies. Our team decided to take on this challenging project because of our own experiences and perspectives, which further demonstrated the importance of tackling this crucial problem statement.

Our project design's development over time is evidence of our dedication to innovation and constant improvement. We were able to find the restrictions and flaws in the earlier version by contrasting the original and current designs, which led to the required adjustments and enhancements. We succeeded in achieving notable improvements in functionality, scalability, and performance through iterative development and feedback-driven refinement, filling in important gaps and bringing the platform closer to our stakeholders' requirements.

With a primary focus on cloud deployment, the Smart Price Retail Platform's implementation approach involved a number of technical issues and development stages. Through the utilisation of a wide range of cloud resources and services, such as database solutions, hosting services, and computing resources, we established a stable and expandable cloud environment to facilitate the implementation of our application. Even though there were difficulties along the route, like setting up intricate cloud infrastructures and making the most use of resources, proactive problem-solving and teamwork allowed us to get over them and accomplish our goals.

Screenshots of our cloud resource dashboard and pertinent configurations clearly off the complexity and scalability of our approach by giving a visual depiction of the complex cloud setup. These images demonstrate our technological expertise and meticulous attention to detail in a concrete way. They also show how our application is supported by a complicated underlying infrastructure and a smooth integration of multiple cloud services.

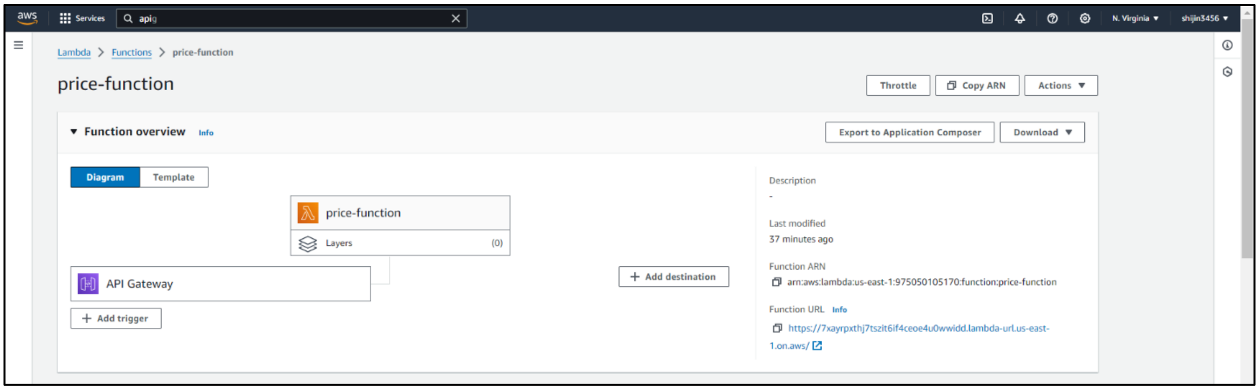
Along with refining our technical skills, the implementation process helped us build our managerial and interpersonal competencies. A key component of our project management strategy was managing project schedules and priorities, working well as a team, and explaining difficult technical ideas to stakeholders who weren't technical. Our learning experience was further enhanced by newfound understandings of the problem-solving process, such as the iterative nature of software development and the significance of adaptability and tenacity in the face of difficulties.

The Smart Price Retail Platform has a great deal of room to grow and improve in the future. The platform's value proposition and competitive advantage may be further strengthened by new features like real-time market monitoring, sophisticated predictive analytics capabilities, and integration with cutting-edge technologies like blockchain and IoT. Enhancements to current features, such speed optimisations, security improvements, and user interface upgrades, are also crucial factors to take into account in order to make sure the platform stays useful and relevant in a constantly shifting environment.

In addition to accomplishing notable accomplishments, the Smart Price Retail Platform has significantly advanced the field of retail pricing optimisation. With the help of our multidisciplinary team, technical know-how, and unwavering commitment, we have created a ground-breaking solution that enables shops to prosper in a fast-paced, cutthroat industry. We are steadfastly dedicated to pushing the envelope of what is conceivable and having a significant impact on the future of retailing as we carry on with our innovative and exploratory journey.

10.SCREENSHOTS

1. LAMDA



price-function

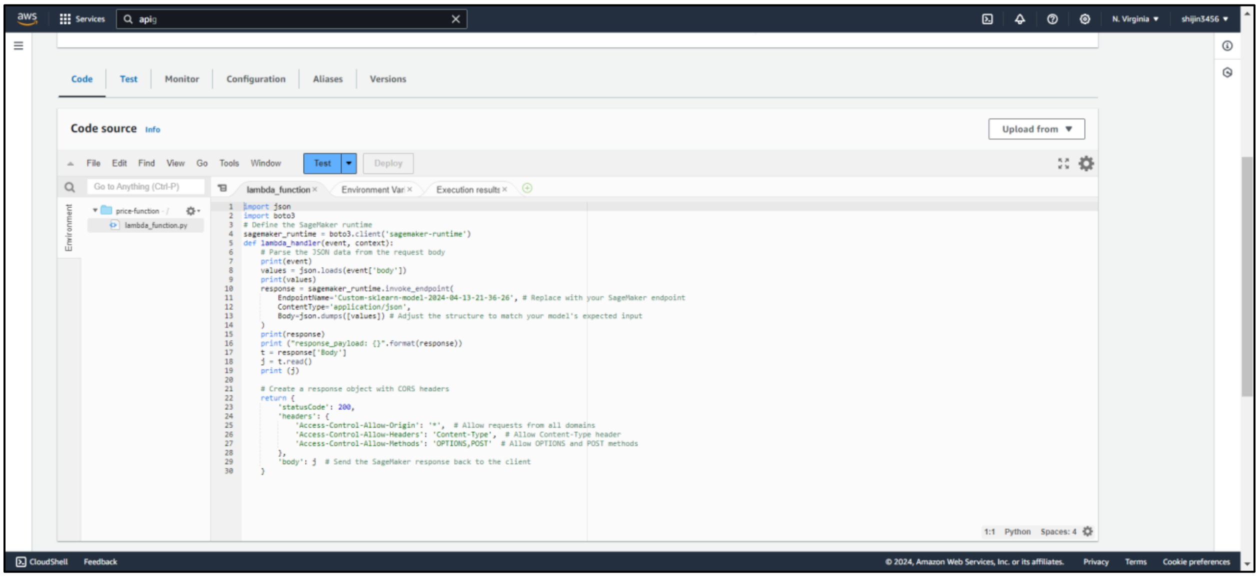
Description
-

Last modified
37 minutes ago

Function ARN
arn:aws:lambda:us-east-1:975050105170:function:price-function

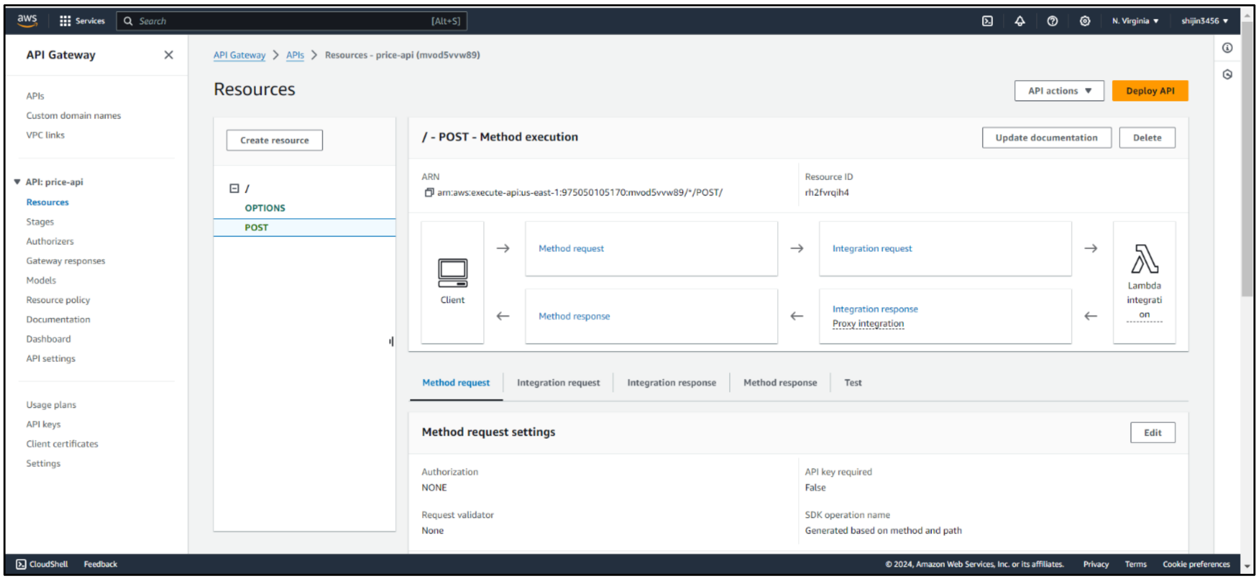
Function URL
https://7xayrpxthj7tszit6if4ceoe4u0wwidd.lambda-url.us-east-1.on.aws/

Lambda Function Code



```python
import json
import boto3
# Define the SageMaker runtime
sagemaker_runtime = boto3.client('sagemaker-runtime')
def lambda_handler(event, context):
    # Parse the JSON data from the request body
    print(event)
    values = json.loads(event['body'])
    print(values)
    response = sagemaker_runtime.invoke_endpoint(
        EndpointName='Custom-sklearn-model-2024-04-13-21-36-26', # Replace with your SageMaker endpoint
        ContentType='application/json',
        Body=json.dumps([values]) # Adjust the structure to match your model's expected input
    )
    print(response)
    print ("response_payload: {}".format(response))
    t = response['Body']
    j = t.read()
    print (j)

    # Create a response object with CORS headers
    return {
        'statusCode': 200,
        'headers': {
            'Access-Control-Allow-Origin': '*',  # Allow requests from all domains
            'Access-Control-Allow-Headers': 'Content-Type',  # Allow Content-Type header
            'Access-Control-Allow-Methods': 'OPTIONS,POST'  # Allow OPTIONS and POST methods
        },
        'body': j  # Send the SageMaker response back to the client
    }
```

2. API GATEWAY



Resources

/ - POST - Method execution

ARN
arn:aws:execute-api:us-east-1:975050105170:mvod5vvw89/*/POST/

Resource ID
rh2fvrqih4

Method request settings

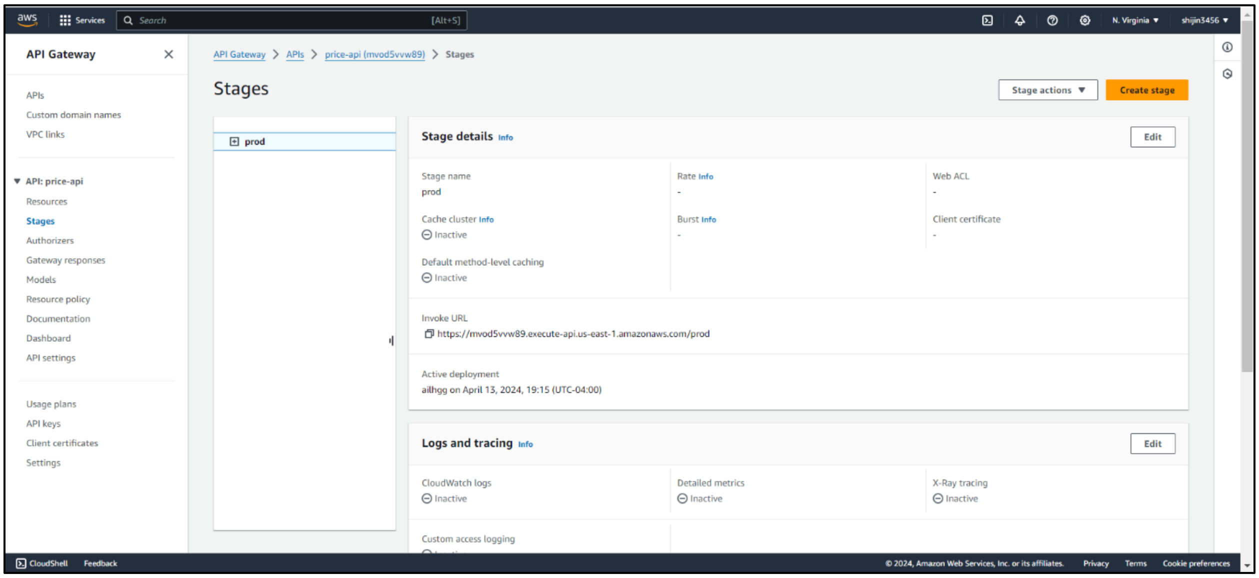
Authorization
NONE

API key required
False

Request validator
None

SDK operation name
Generated based on method and path

API Gateway Stage with CORS Deployed:



Stages

Stage details

Stage name
prod

Rate
-

Web ACL
-

Cache cluster
Inactive

Burst
-

Client certificate
-

Default method-level caching
Inactive

Invoke URL
https://mvod5vvw89.execute-api.us-east-1.amazonaws.com/prod

Active deployment
ailhgg on April 13, 2024, 19:15 (UTC-04:00)

Logs and tracing

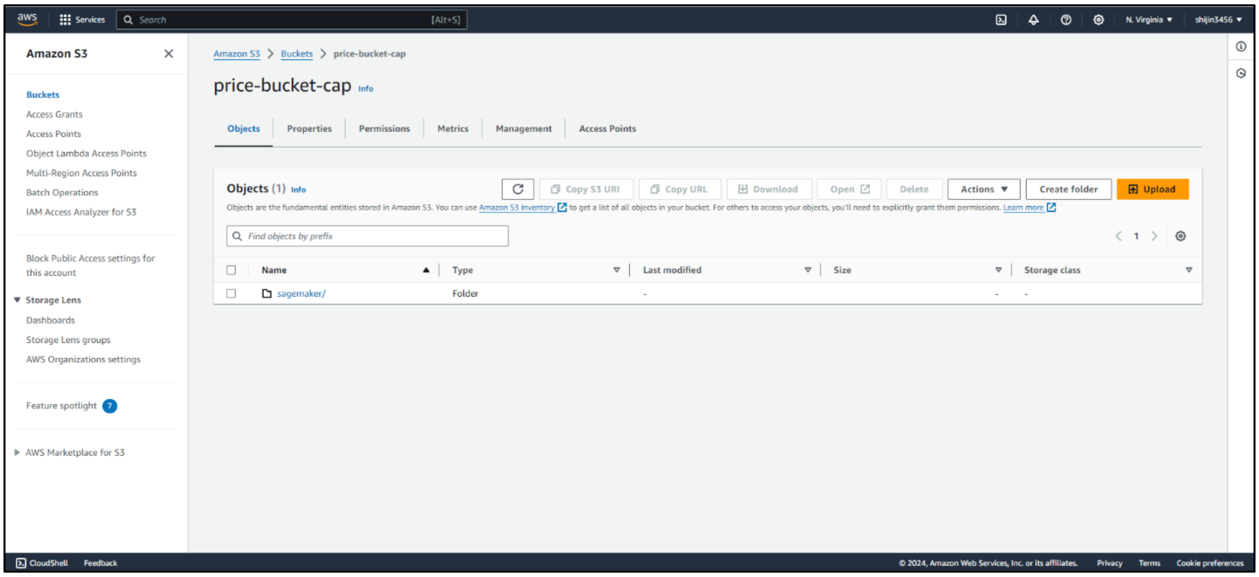
CloudWatch logs
Inactive

Detailed metrics
Inactive

X-Ray tracing
Inactive

Custom access logging

3. S3 BUCKET



Amazon S3 > Buckets > price-bucket-cap

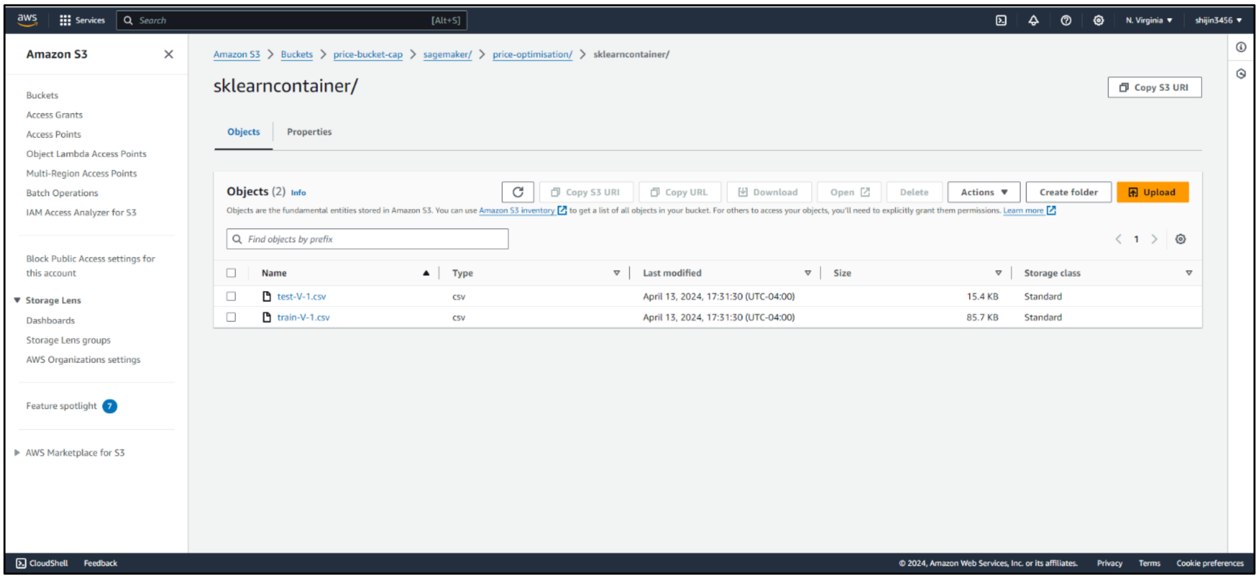
price-bucket-cap Info

Objects (1) Info

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

□	Name	Type	Last modified	Size	Storage class
□	sagemaker/	Folder	-	-	-

4. TRAINING AND TESTING DATA INSIDE BUCKET



Amazon S3 > Buckets > price-bucket-cap > sagemaker/ > price-optimisation/ > sklearncontainer/

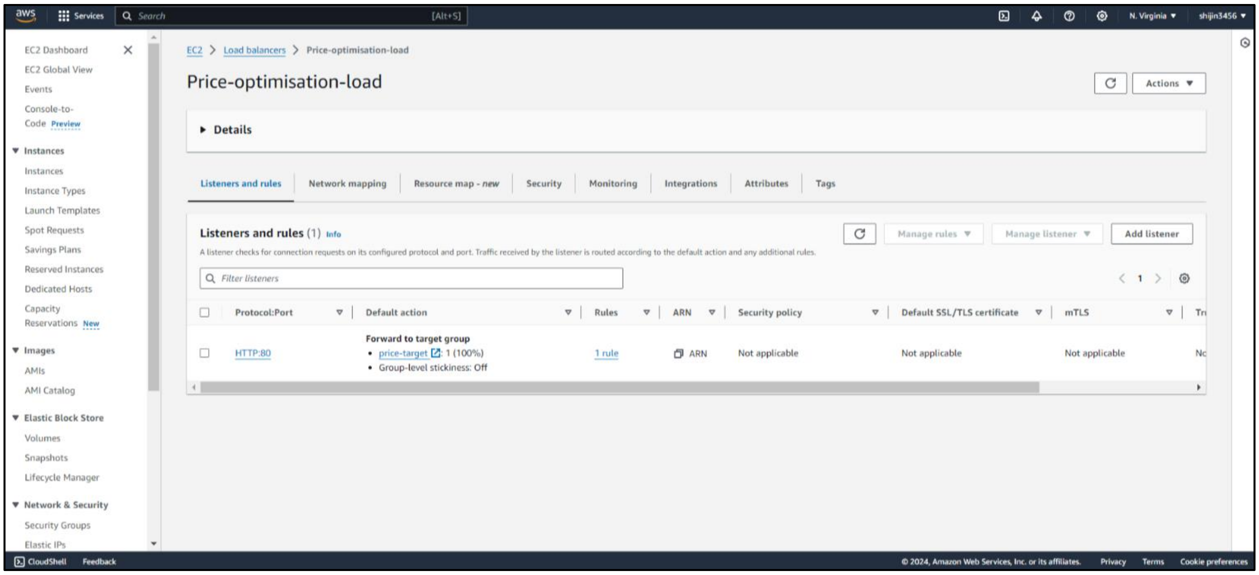
sklearncontainer/

Objects (2) Info

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

□	Name	Type	Last modified	Size	Storage class
□	test-V-1.csv	csv	April 13, 2024, 17:31:30 (UTC-04:00)	15.4 KB	Standard
□	train-V-1.csv	csv	April 13, 2024, 17:31:30 (UTC-04:00)	85.7 KB	Standard

5. LOAD BALANCER:



EC2 > Load balancers > Price-optimisation-load

Price-optimisation-load

Actions

▸ Details

Listeners and rules | Network mapping | Resource map - *new* | Security | Monitoring | Integrations | Attributes | Tags

Listeners and rules (1) Info

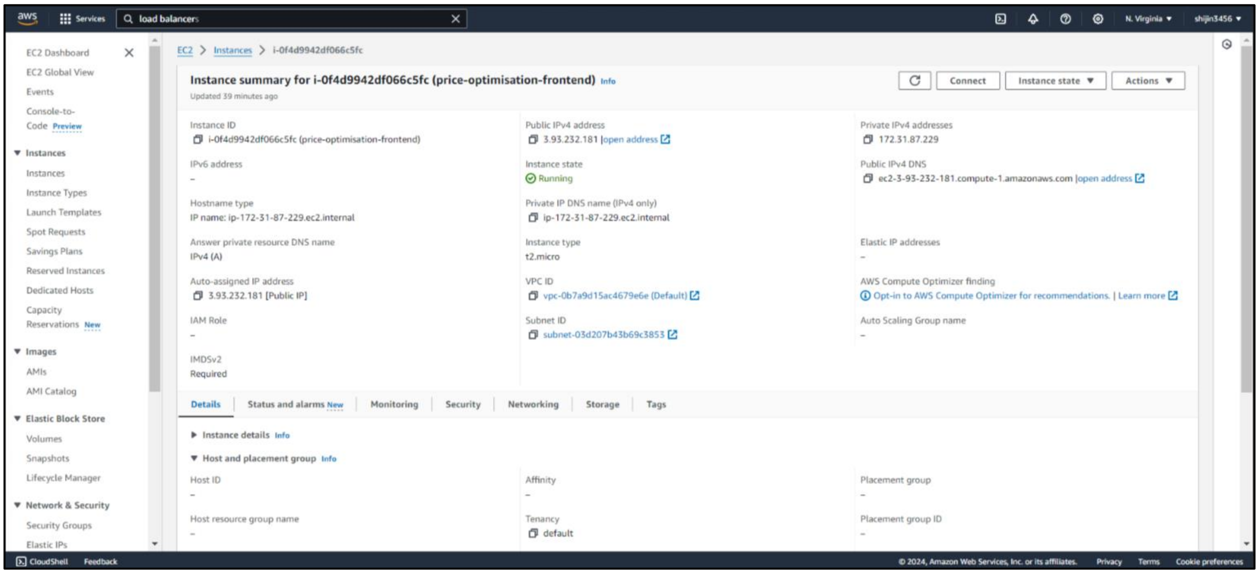
Manage rules | Manage listener | Add listener

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Filter listeners

□	Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS	Tr
□	HTTP:80	Forward to target group • price-target: 1 (100%) • Group-level stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not applicable	Nc

6. EC2 INSTANCE:



EC2 > Instances > i-0f4d9942df066c5fc

Instance summary for i-0f4d9942df066c5fc (price-optimisation-frontend) Info

Updated 39 minutes ago

Connect | Instance state | Actions

Instance ID
i-0f4d9942df066c5fc (price-optimisation-frontend)

Public IPv4 address
3.93.232.181 |open address

Private IPv4 addresses
172.31.87.229

IPv6 address
–

Instance state
Running

Public IPv4 DNS
ec2-3-93-232-181.compute-1.amazonaws.com |open address

Hostname type
IP name: ip-172-31-87-229.ec2.internal

Private IP DNS name (IPv4 only)
ip-172-31-87-229.ec2.internal

Answer private resource DNS name
IPv4 (A)

Instance type
t2.micro

Elastic IP addresses
–

Auto-assigned IP address
3.93.232.181 [Public IP]

VPC ID
vpc-0b7a9d15ac4679e6e (Default)

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

IAM Role
–

Subnet ID
subnet-03d207b43b69c3853

Auto Scaling Group name
–

IMDSv2
Required

Details | Status and alarms New | Monitoring | Security | Networking | Storage | Tags

▸ Instance details Info

▾ Host and placement group Info

Host ID
–

Affinity
–

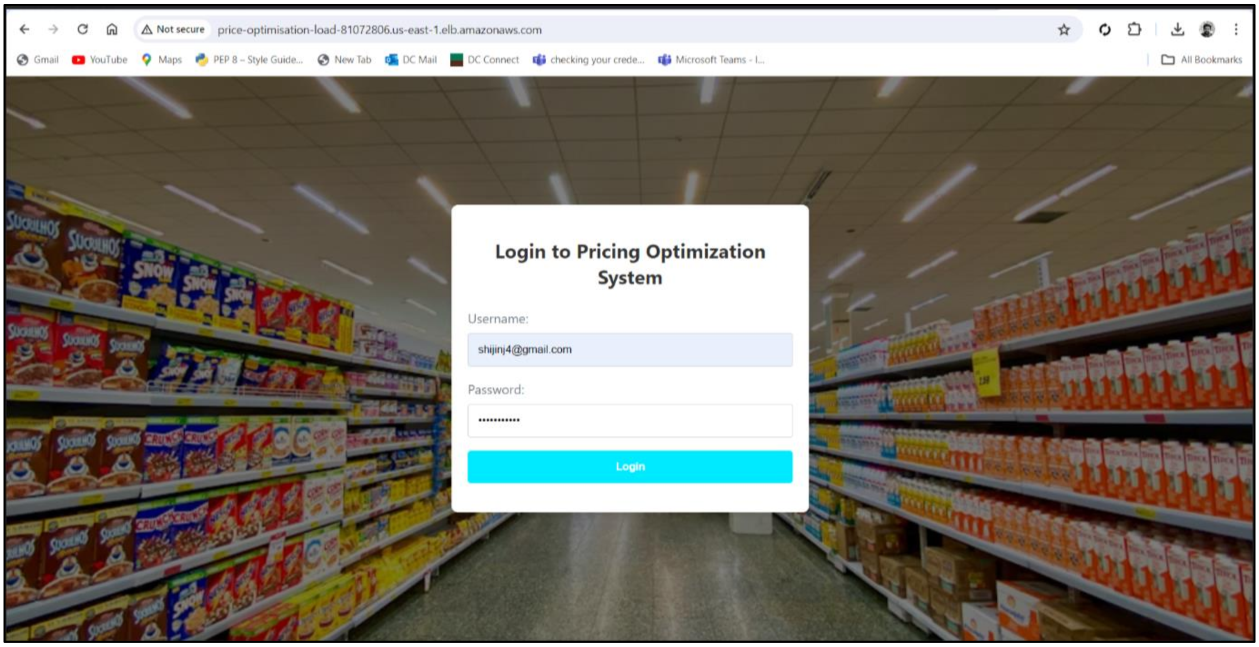
Placement group
–

Host resource group name
–

Tenancy
default

Placement group ID
–

7. WEBSITE HOSTED IN LOAD BALANCER:

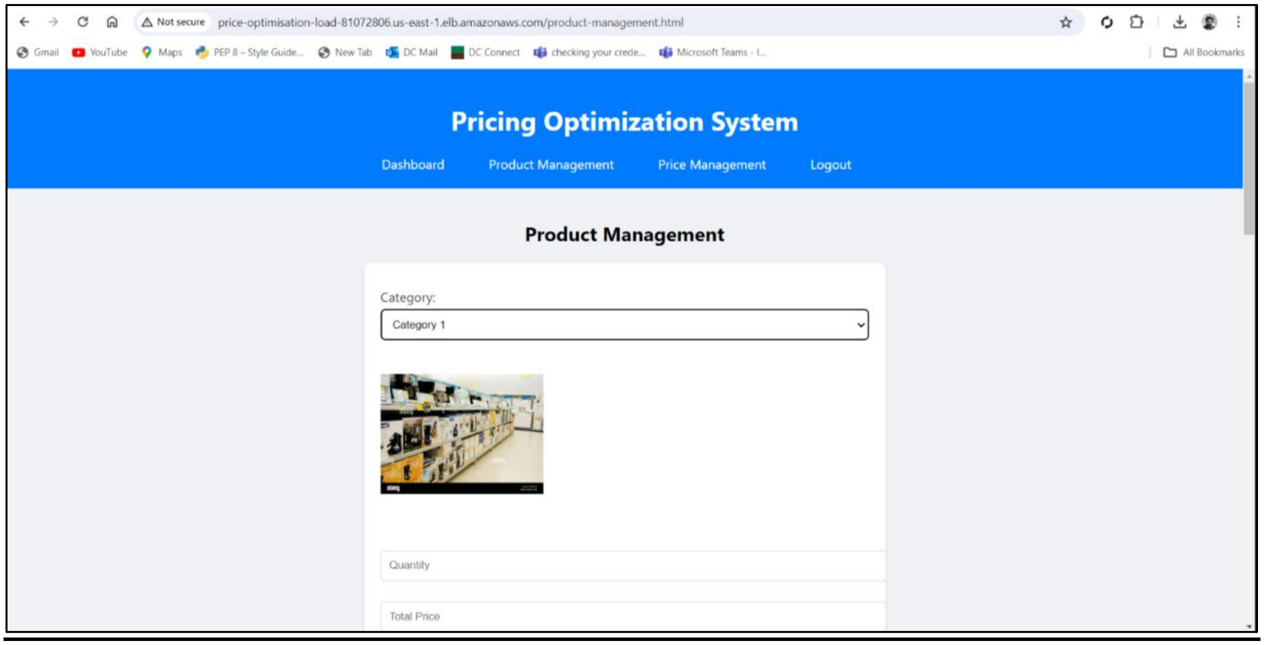


Login to Pricing Optimization System

Username: shijinj4@gmail.com

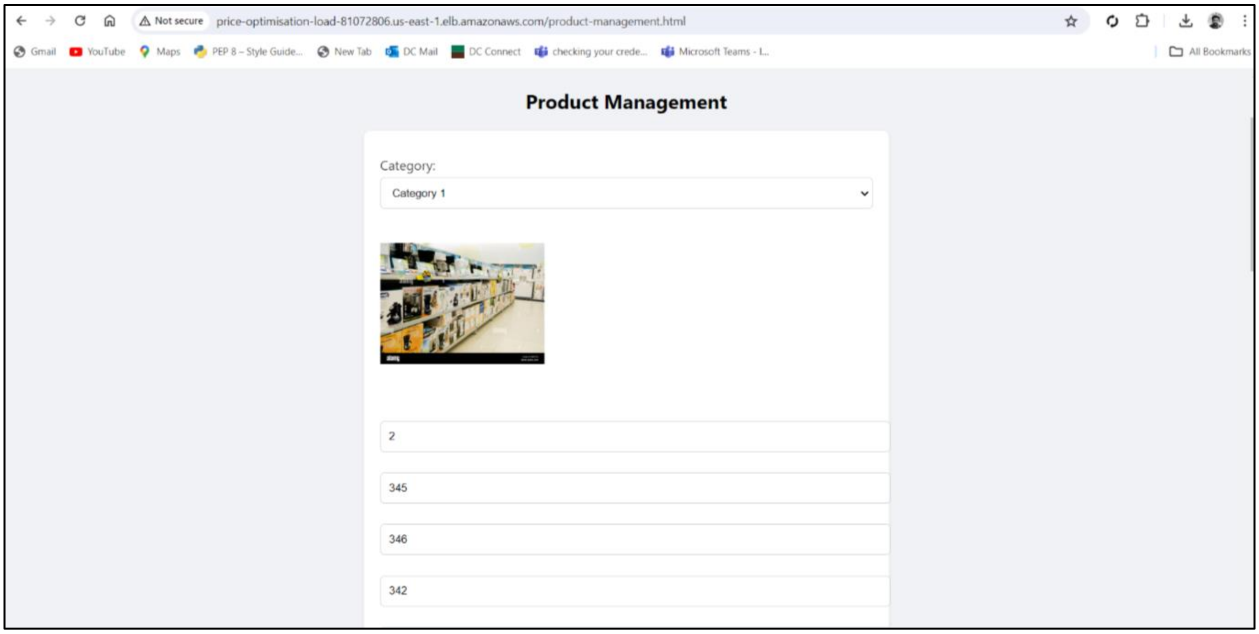
Password: •••••••••••

8. PRODUCT MANAGEMENT PAGE:



Product Management

Category: Category 1



Product Management

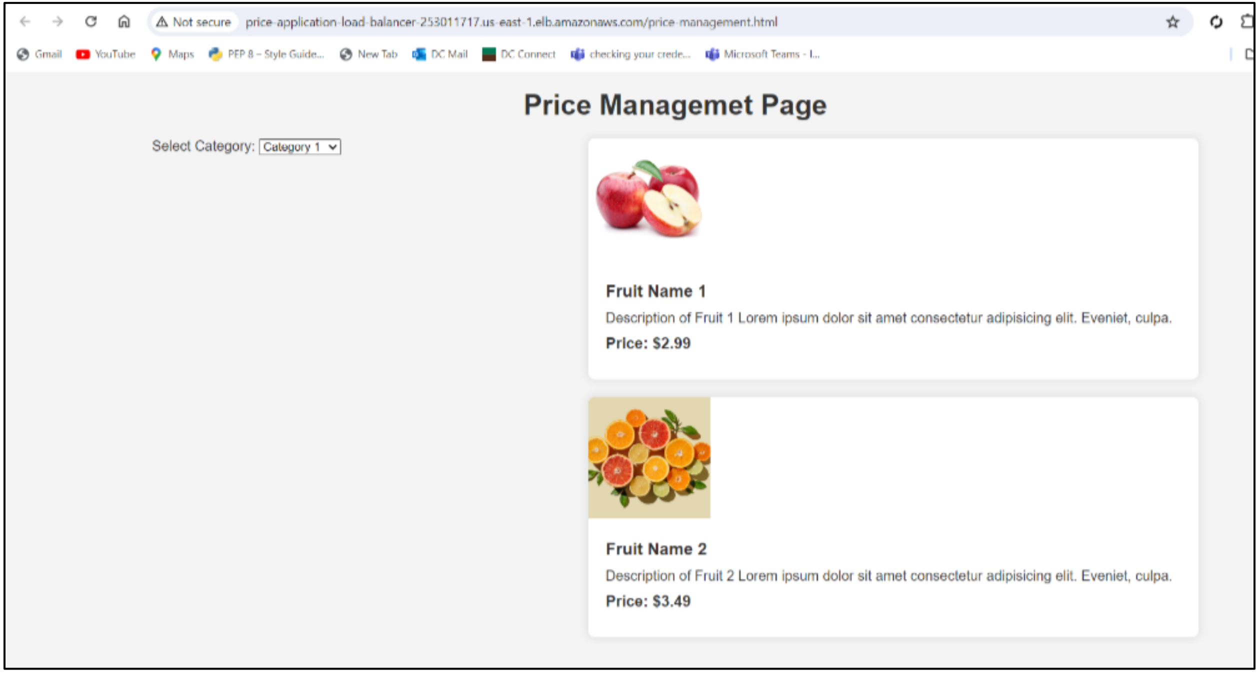
Category: Category 1

2

345

346

342



Price Managemet Page

Select Category: Category 1

Fruit Name 1

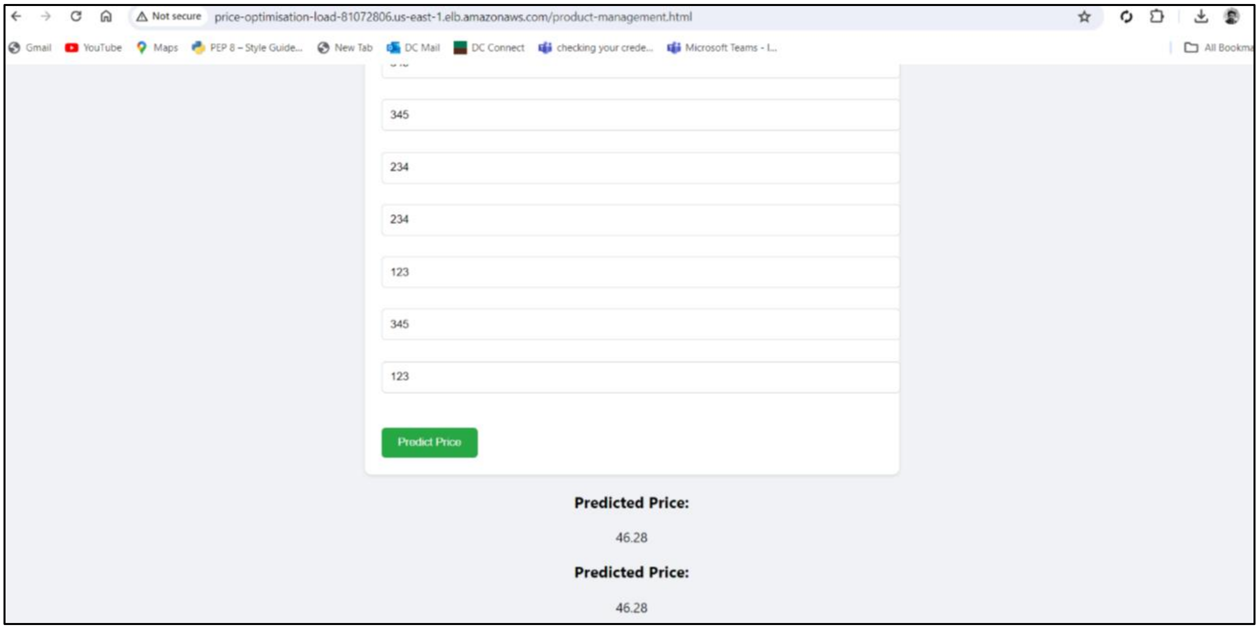
Description of Fruit 1 Lorem ipsum dolor sit amet consectetur adipisicing elit. Eveniet, culpa.

Price: $2.99

Fruit Name 2

Description of Fruit 2 Lorem ipsum dolor sit amet consectetur adipisicing elit. Eveniet, culpa.

Price: $3.49



345

234

234

123

345

123

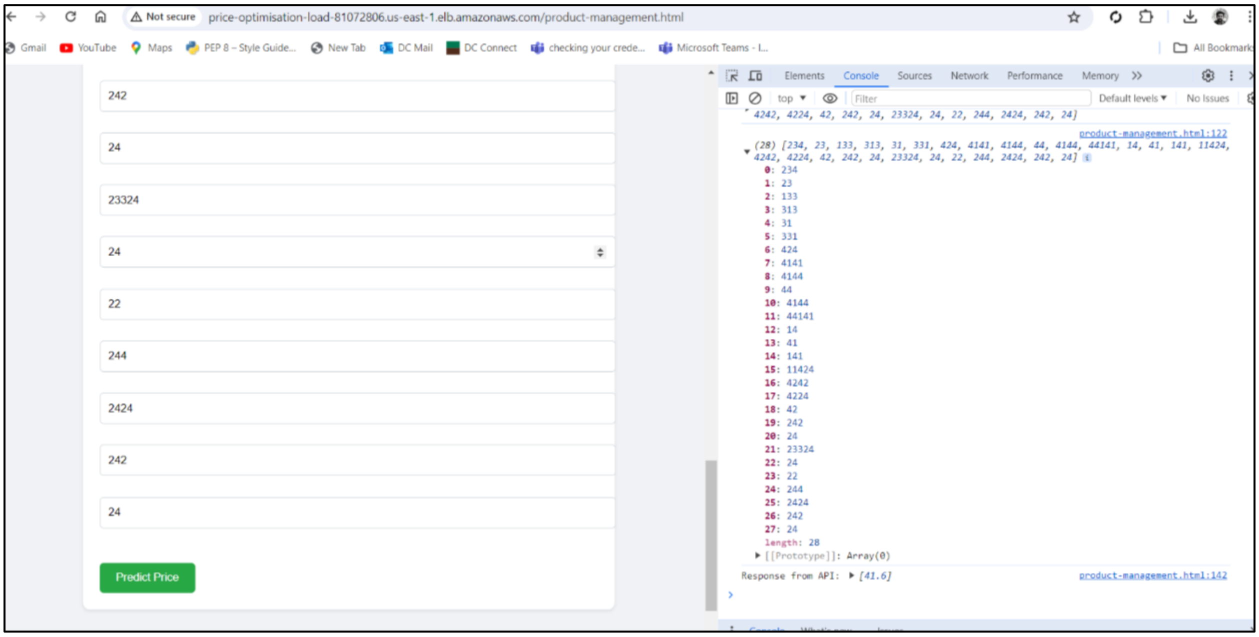
Predicted Price:

46.28

Predicted Price:

46.28

9. GETTING PROPER RESPONSE FROM API



242

24

23324

24

22

244

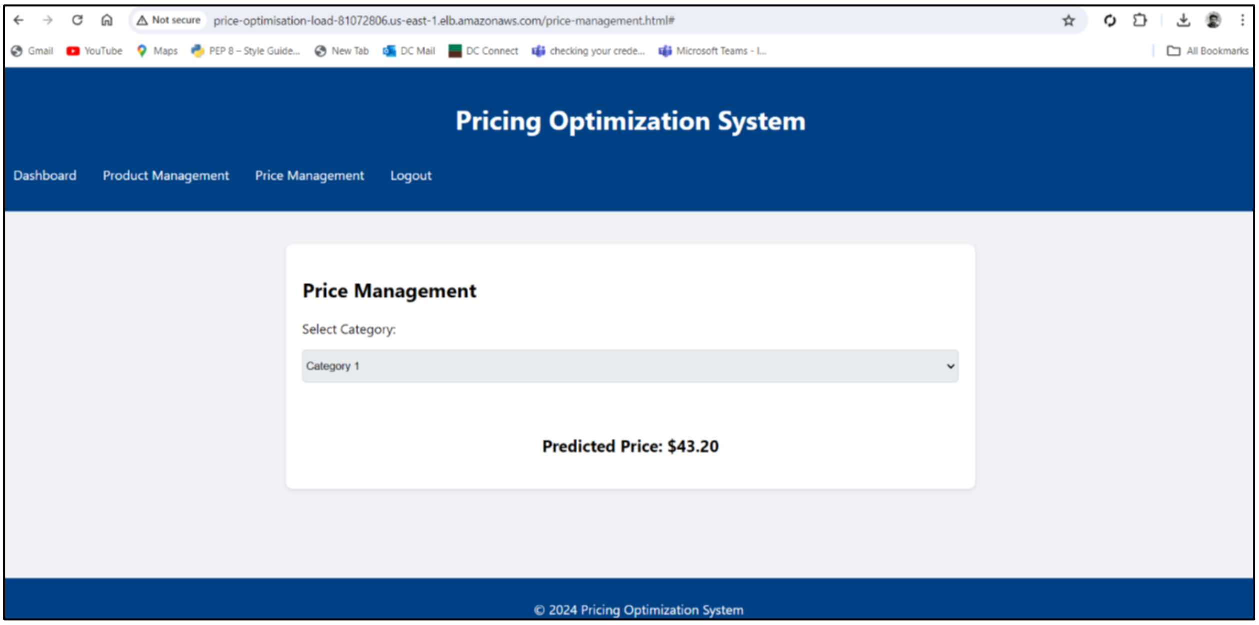
2424

242

24

```
  4242, 4224, 42, 242, 24, 23324, 24, 22, 244, 2424, 242, 24]
                                                    product-management.html:122
▾ (28) [234, 23, 133, 313, 31, 331, 424, 4141, 4144, 44, 4144, 44141, 14, 41, 141, 11424,
  4242, 4224, 42, 242, 24, 23324, 24, 22, 244, 2424, 242, 24]
    0: 234
    1: 23
    2: 133
    3: 313
    4: 31
    5: 331
    6: 424
    7: 4141
    8: 4144
    9: 44
    10: 4144
    11: 44141
    12: 14
    13: 41
    14: 141
    15: 11424
    16: 4242
    17: 4224
    18: 42
    19: 242
    20: 24
    21: 23324
    22: 24
    23: 22
    24: 244
    25: 2424
    26: 242
    27: 24
    length: 28
  ▸ [[Prototype]]: Array(0)
Response from API:  ▸ [41.6]                        product-management.html:142
>
```

PREDICTED PRICE WILL BE DISPLAYED IN THIS PAGE FOR SELECTED CATEGORY:



Pricing Optimization System

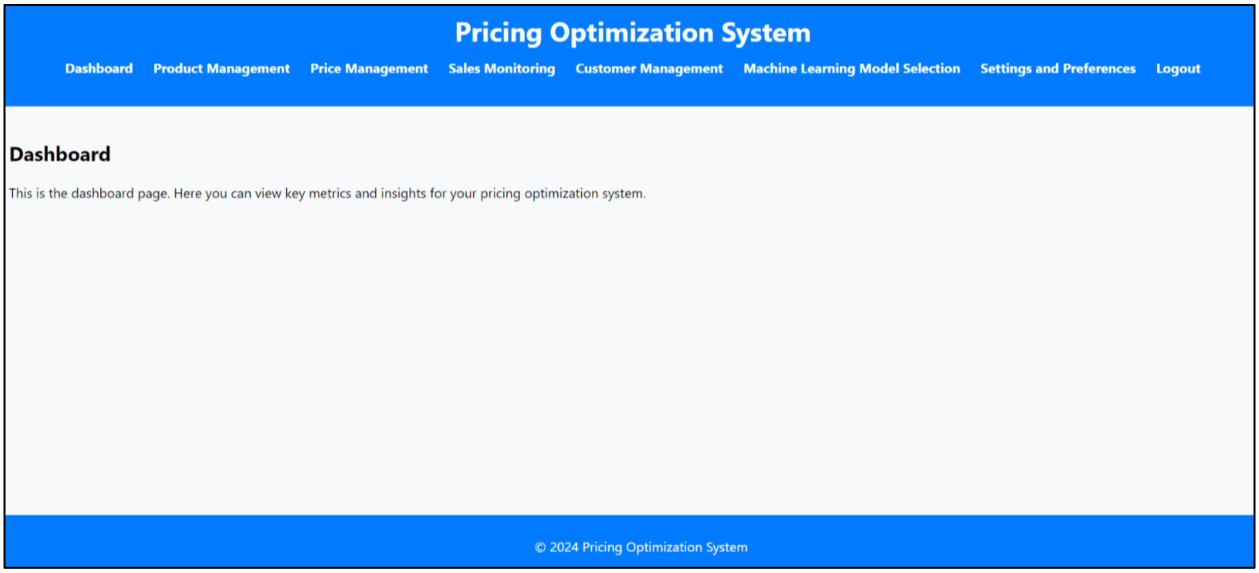
Price Management

Select Category: Category 1

Predicted Price: $43.20

© 2024 Pricing Optimization System

DASHBOARD PAGE :



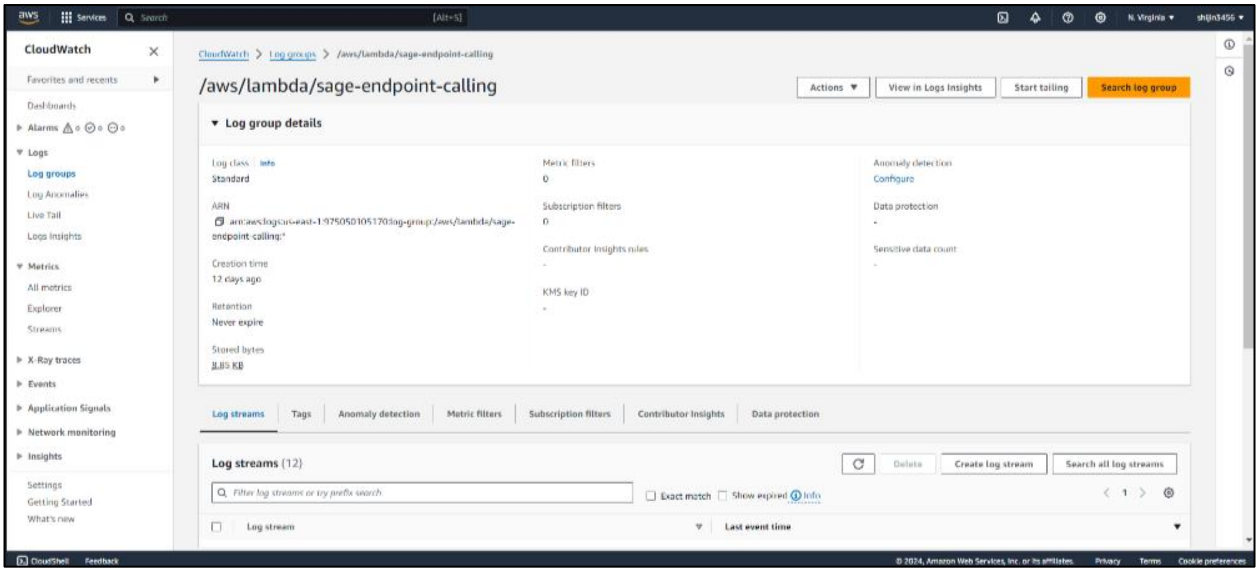
Pricing Optimization System

Dashboard

This is the dashboard page. Here you can view key metrics and insights for your pricing optimization system.

© 2024 Pricing Optimization System

10. AWS CLOUDWATCH LOGS FOR ENDPOINT MONITORING:



/aws/lambda/sage-endpoint-calling

▼ Log group details

Log class Info
Standard

ARN
arn:aws:logs:us-east-1:975050105170:log-group:/aws/lambda/sage-endpoint-calling:*

Creation time
12 days ago

Retention
Never expire

Stored bytes
8.85 KB

Metric filters
0

Subscription filters
0

Contributor Insights rules
-

KMS key ID
-

Anomaly detection
Configure

Data protection
-

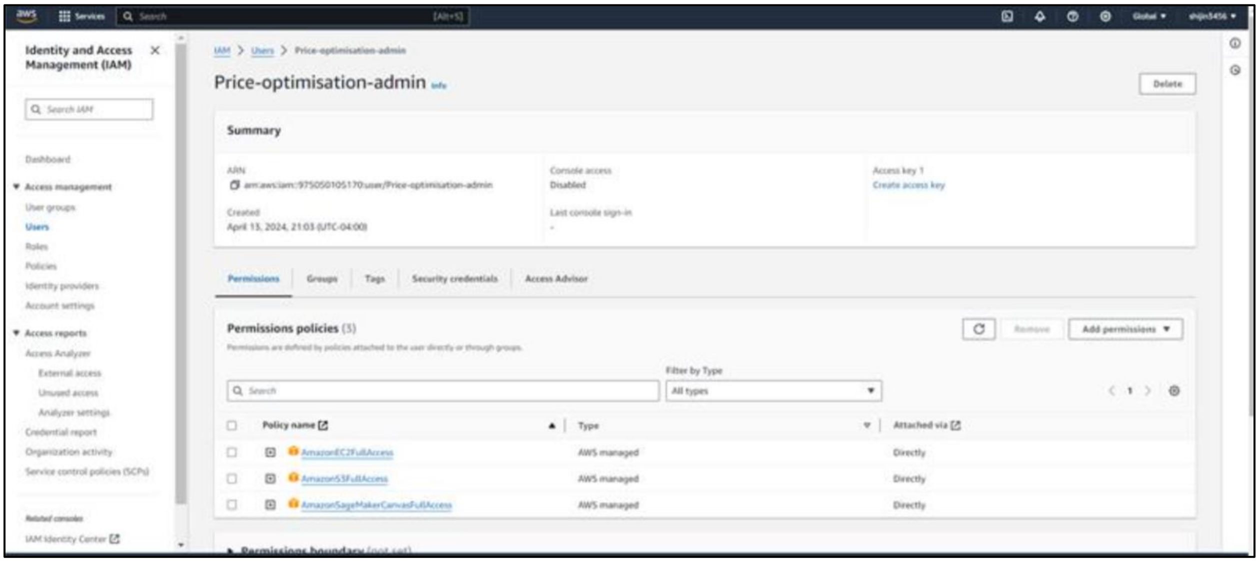
Sensitive data count
-

Log streams (12)

□ Exact match □ Show expired

□	Log stream	Last event time

11. IAM ROLE CREATED:



Price-optimisation-admin

Summary

ARN
arn:aws:iam::975050105170:user/Price-optimisation-admin

Created
April 13, 2024, 21:03 (UTC-04:00)

Console access
Disabled

Last console sign-in
-

Access key 1
Create access key

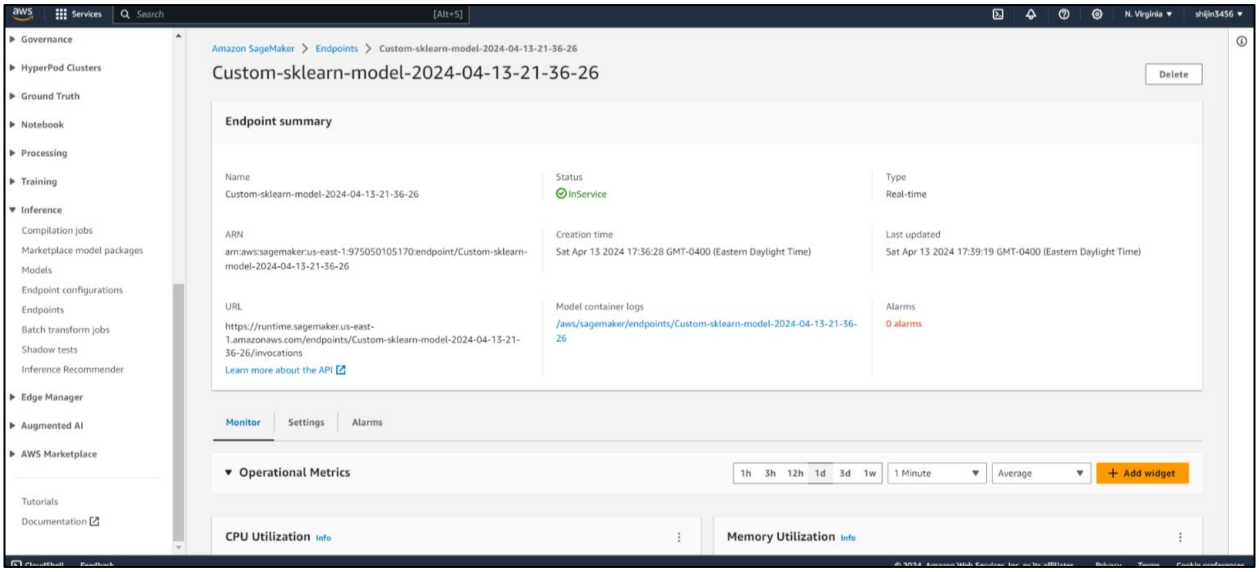
Permissions policies (3)

Permissions are defined by policies attached to the user directly or through groups.

Filter by Type
All types

□	Policy name	Type	Attached via
□	AmazonEC2FullAccess	AWS managed	Directly
□	AmazonS3FullAccess	AWS managed	Directly
□	AmazonSageMakerCanvasFullAccess	AWS managed	Directly

12. SAGEMAKER ENDPOINT



Custom-sklearn-model-2024-04-13-21-36-26

Endpoint summary

Name
Custom-sklearn-model-2024-04-13-21-36-26

Status
InService

Type
Real-time

ARN
arn:aws:sagemaker:us-east-1:975050105170:endpoint/Custom-sklearn-model-2024-04-13-21-36-26

Creation time
Sat Apr 13 2024 17:36:28 GMT-0400 (Eastern Daylight Time)

Last updated
Sat Apr 13 2024 17:39:19 GMT-0400 (Eastern Daylight Time)

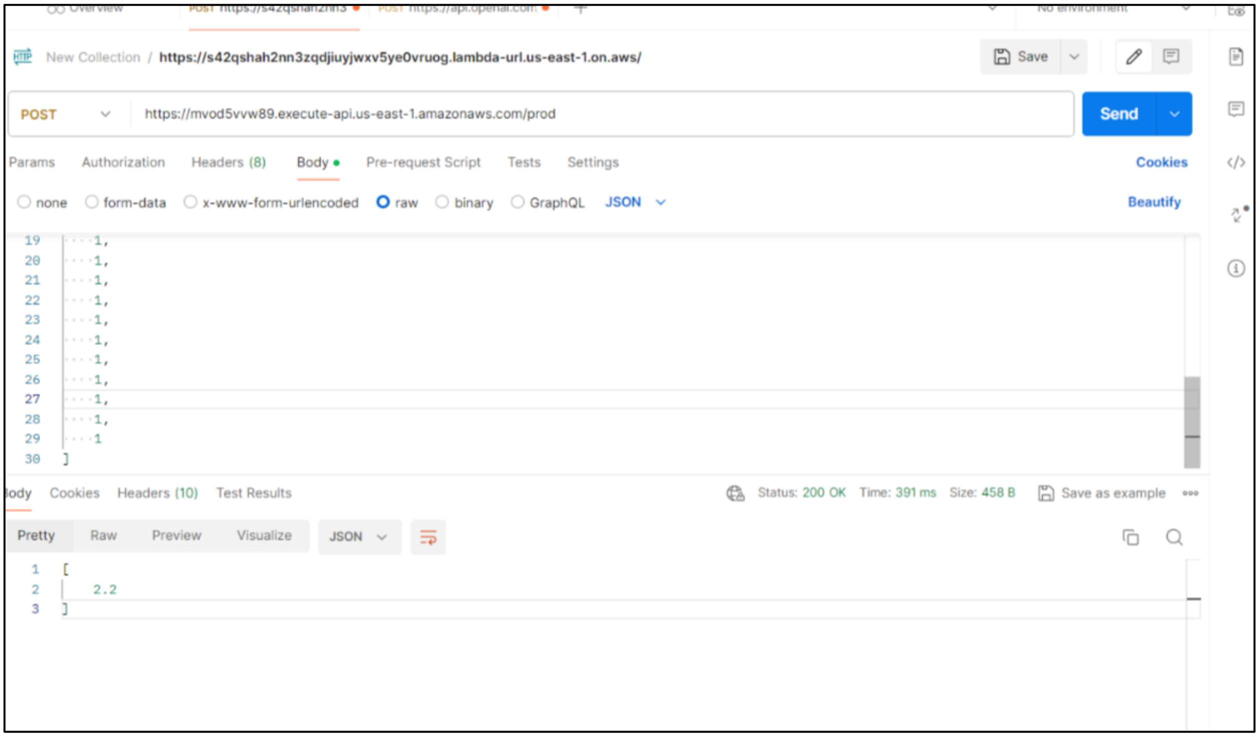
URL
https://runtime.sagemaker.us-east-1.amazonaws.com/endpoints/Custom-sklearn-model-2024-04-13-21-36-26/invocations
Learn more about the API

Model container logs
/aws/sagemaker/endpoints/Custom-sklearn-model-2024-04-13-21-36-26

Alarms
0 alarms

Operational Metrics

POSTMAN TESTING FOR THE ENDPOINT:



New Collection / https://s42qshah2nn3zqdjiuyjwxv5ye0vruog.lambda-url.us-east-1.on.aws/

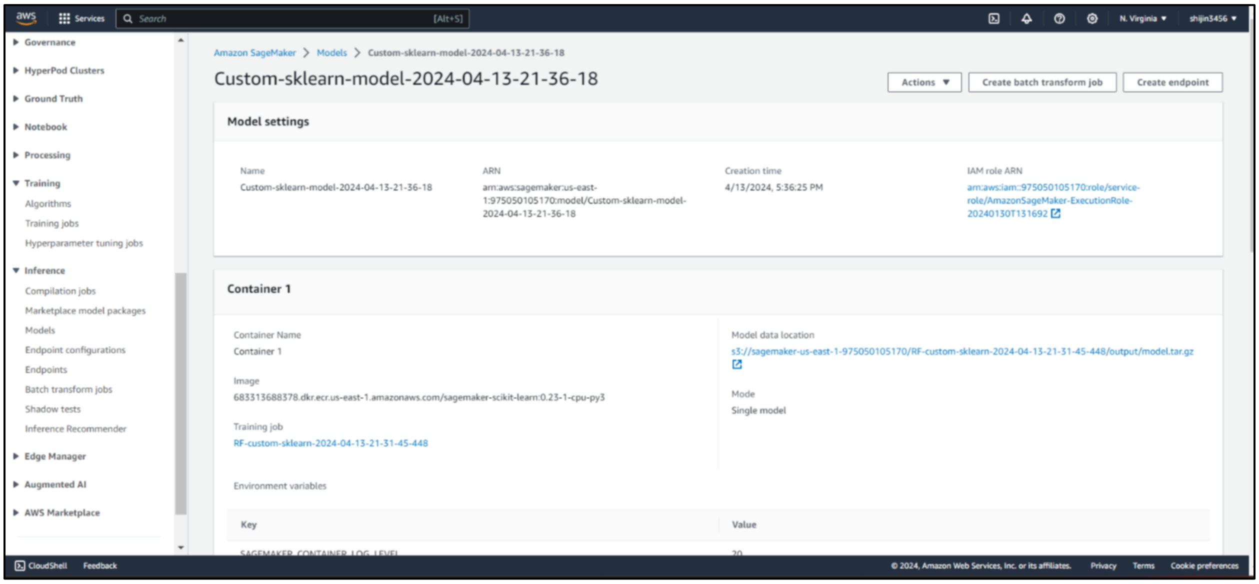
POST https://mvod5vvw89.execute-api.us-east-1.amazonaws.com/prod

```
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1
]
```

Status: 200 OK Time: 391 ms Size: 458 B

```
[
    2.2
]
```

13. SAGEMAKER MODEL FOR RETAIL PRICE PREDICTION



Amazon SageMaker > Models > Custom-sklearn-model-2024-04-13-21-36-18

Custom-sklearn-model-2024-04-13-21-36-18

Model settings

Name
Custom-sklearn-model-2024-04-13-21-36-18

ARN
arn:aws:sagemaker:us-east-1:975050105170:model/Custom-sklearn-model-2024-04-13-21-36-18

Creation time
4/13/2024, 5:36:25 PM

IAM role ARN
arn:aws:iam::975050105170:role/service-role/AmazonSageMaker-ExecutionRole-20240130T131692

Container 1

Container Name
Container 1

Image
683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-scikit-learn:0.23-1-cpu-py3

Training job
RF-custom-sklearn-2024-04-13-21-31-45-448

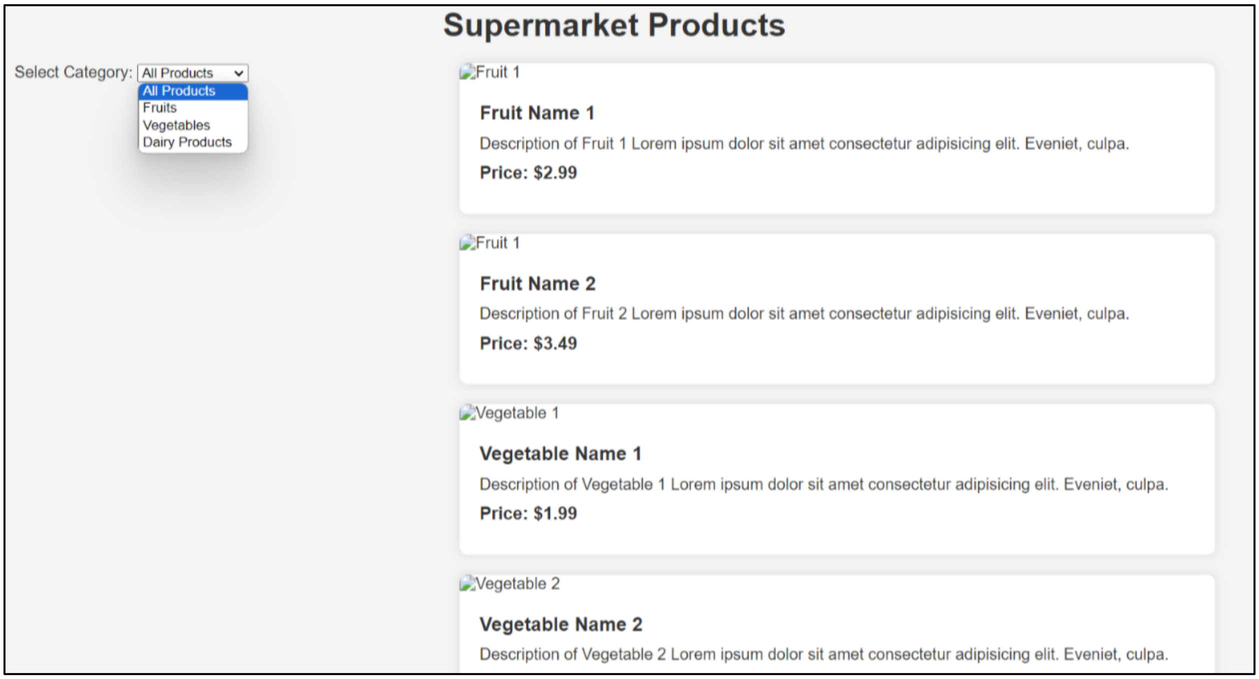
Model data location
s3://sagemaker-us-east-1-975050105170/RF-custom-sklearn-2024-04-13-21-31-45-448/output/model.tar.gz

Mode
Single model

Environment variables

Key	Value
SAGEMAKER_CONTAINER_LOG_LEVEL	20

14. LIST OF PRODUCTS



Supermarket Products

Select Category: All Products

Fruit 1

Fruit Name 1

Description of Fruit 1 Lorem ipsum dolor sit amet consectetur adipisicing elit. Eveniet, culpa.

Price: $2.99

Fruit 1

Fruit Name 2

Description of Fruit 2 Lorem ipsum dolor sit amet consectetur adipisicing elit. Eveniet, culpa.

Price: $3.49

Vegetable 1

Vegetable Name 1

Description of Vegetable 1 Lorem ipsum dolor sit amet consectetur adipisicing elit. Eveniet, culpa.

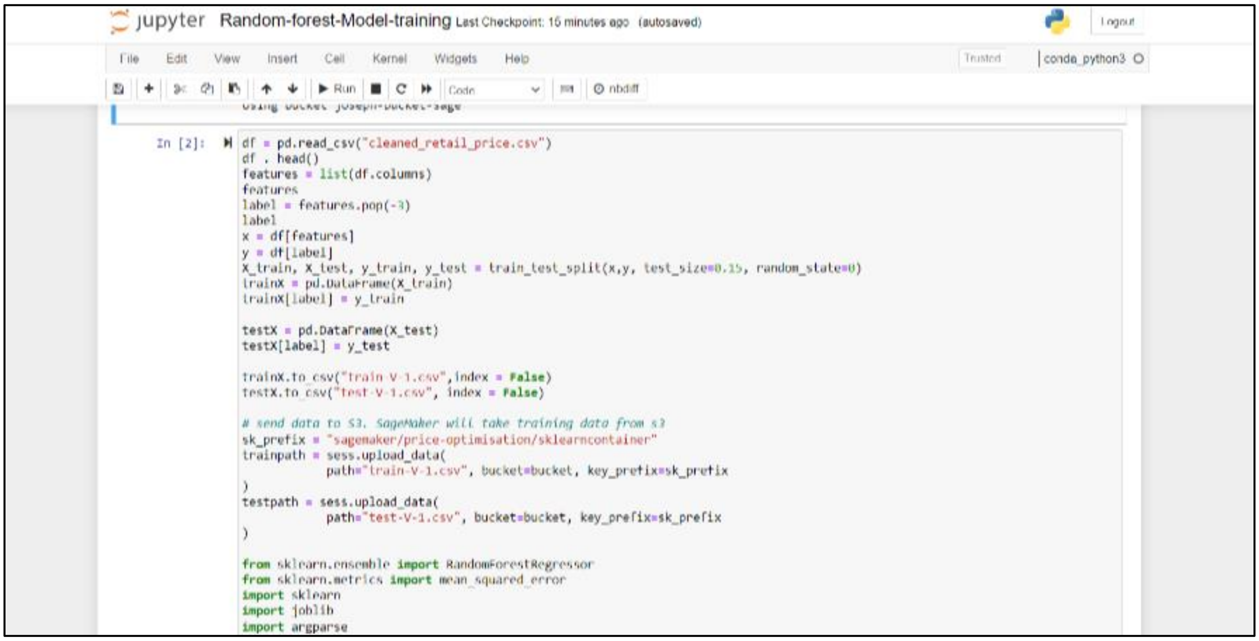
Price: $1.99

Vegetable 2

Vegetable Name 2

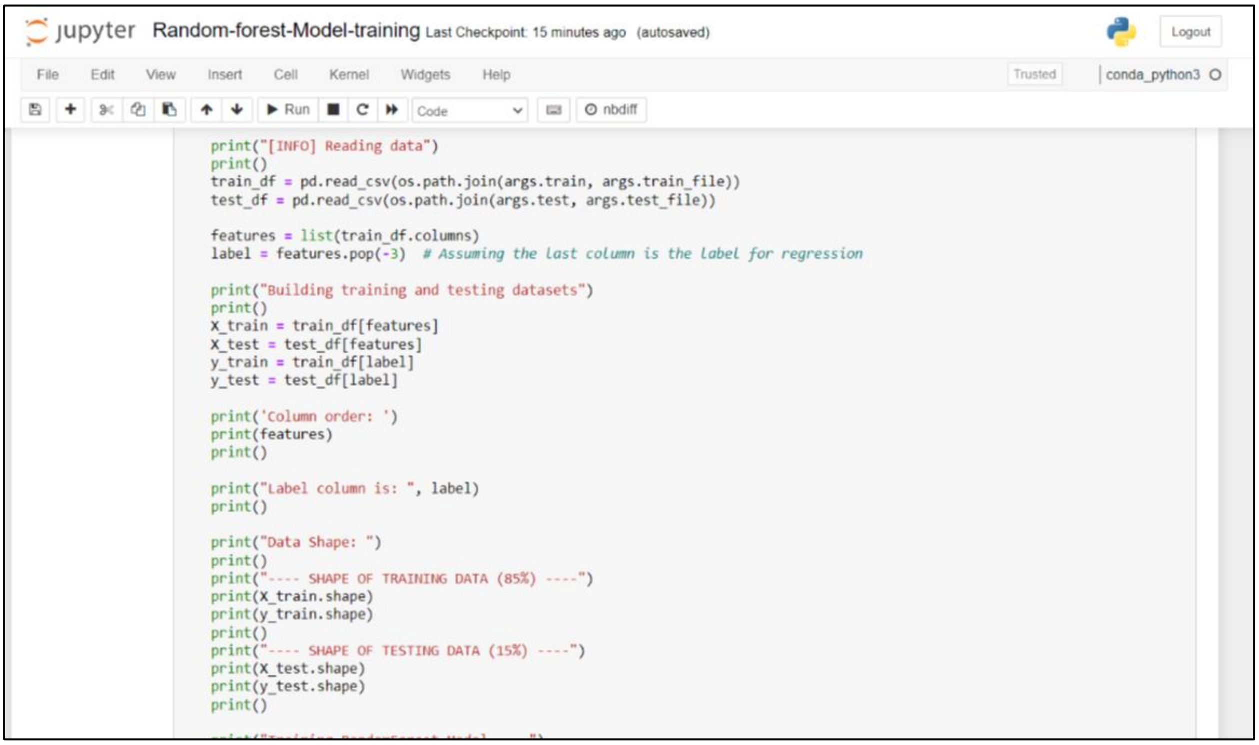
Description of Vegetable 2 Lorem ipsum dolor sit amet consectetur adipisicing elit. Eveniet, culpa.

15. MODEL TRAINING SCRIPT DEPLOYED IN SAGEMAKER NOTEBOOK INSTANCES:

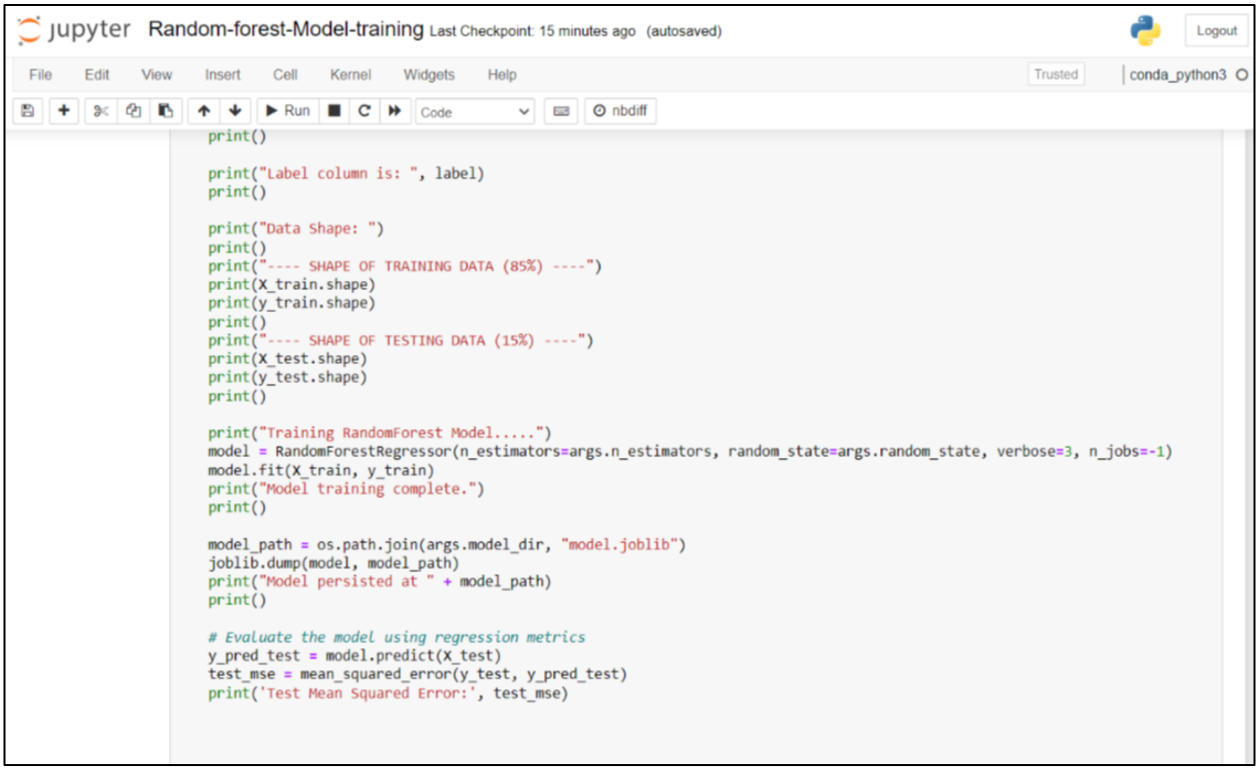

jupyter Random-forest-Model-training Last Checkpoint: 15 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
conda_python3
Run
Code
nbdiff

```
using bucket joseph-bucket-sage
```

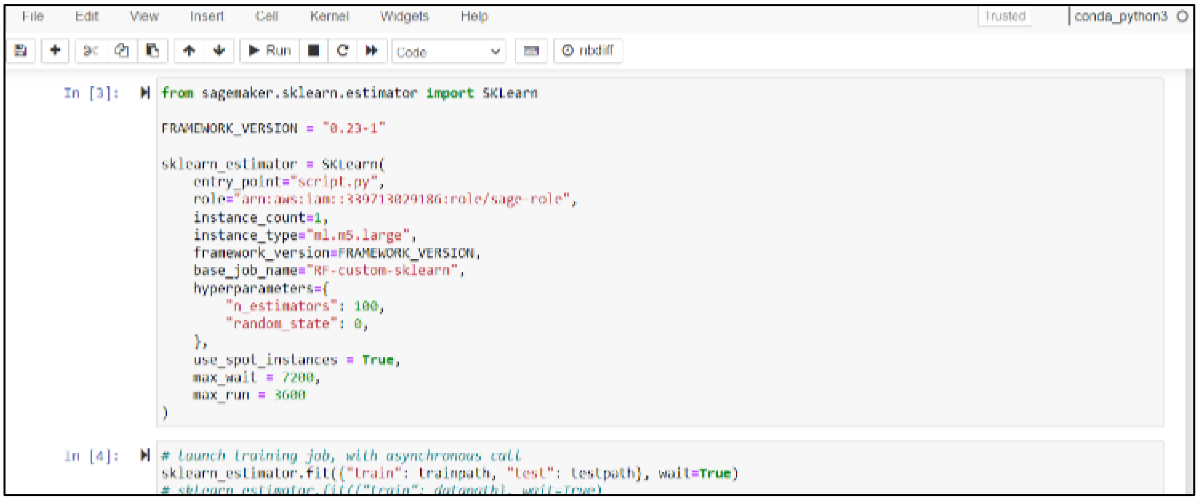
```
In [2]:  df = pd.read_csv("cleaned_retail_price.csv")
         df . head()
         features = list(df.columns)
         features
         label = features.pop(-3)
         label
         x = df[features]
         y = df[label]
         X_train, X_test, y_train, y_test = train_test_split(x,y, test_size=0.15, random_state=0)
         trainX = pd.DataFrame(X_train)
         trainX[label] = y_train

         testX = pd.DataFrame(X_test)
         testX[label] = y_test

         trainX.to_csv("train-V-1.csv",index = False)
         testX.to_csv("test-V-1.csv", index = False)

         # send data to S3. SageMaker will take training data from s3
         sk_prefix = "sagemaker/price-optimisation/sklearncontainer"
         trainpath = sess.upload_data(
                     path="train-V-1.csv", bucket=bucket, key_prefix=sk_prefix
         )
         testpath = sess.upload_data(
                     path="test-V-1.csv", bucket=bucket, key_prefix=sk_prefix
         )

         from sklearn.ensemble import RandomForestRegressor
         from sklearn.metrics import mean_squared_error
         import sklearn
         import joblib
         import argparse
```


jupyter Random-forest-Model-training Last Checkpoint: 15 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
conda_python3
Run
Code
nbdiff

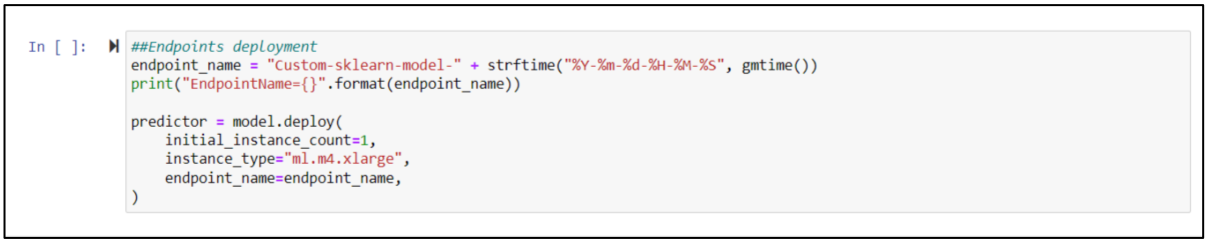
```
print("[INFO] Reading data")
print()
train_df = pd.read_csv(os.path.join(args.train, args.train_file))
test_df = pd.read_csv(os.path.join(args.test, args.test_file))

features = list(train_df.columns)
label = features.pop(-3)  # Assuming the last column is the label for regression

print("Building training and testing datasets")
print()
X_train = train_df[features]
X_test = test_df[features]
y_train = train_df[label]
y_test = test_df[label]

print('Column order: ')
print(features)
print()

print("Label column is: ", label)
print()

print("Data Shape: ")
print()
print("---- SHAPE OF TRAINING DATA (85%) ----")
print(X_train.shape)
print(y_train.shape)
print()
print("---- SHAPE OF TESTING DATA (15%) ----")
print(X_test.shape)
print(y_test.shape)
print()
```


jupyter Random-forest-Model-training Last Checkpoint: 15 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
conda_python3
Run
Code
nbdiff

```
print()

print("Label column is: ", label)
print()

print("Data Shape: ")
print()
print("---- SHAPE OF TRAINING DATA (85%) ----")
print(X_train.shape)
print(y_train.shape)
print()
print("---- SHAPE OF TESTING DATA (15%) ----")
print(X_test.shape)
print(y_test.shape)
print()

print("Training RandomForest Model.....")
model = RandomForestRegressor(n_estimators=args.n_estimators, random_state=args.random_state, verbose=3, n_jobs=-1)
model.fit(X_train, y_train)
print("Model training complete.")
print()

model_path = os.path.join(args.model_dir, "model.joblib")
joblib.dump(model, model_path)
print("Model persisted at " + model_path)
print()

# Evaluate the model using regression metrics
y_pred_test = model.predict(X_test)
test_mse = mean_squared_error(y_test, y_pred_test)
print('Test Mean Squared Error:', test_mse)
```

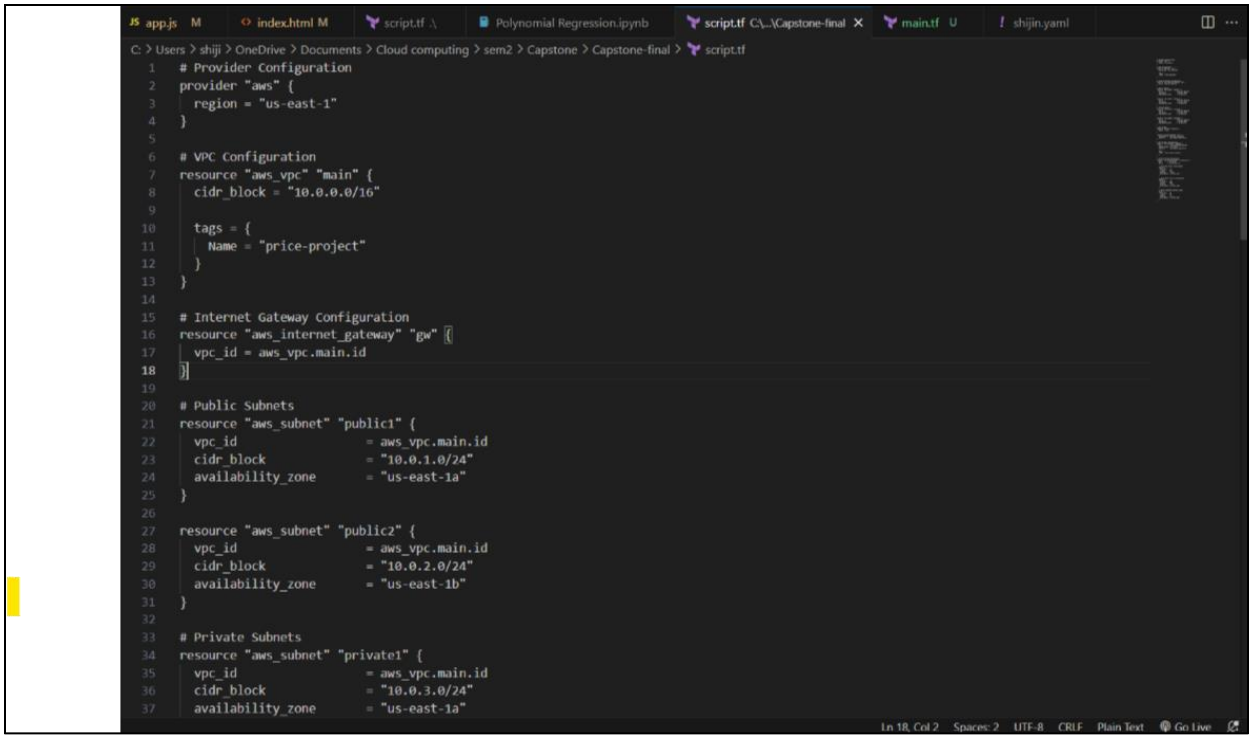


```
In [3]: from sagemaker.sklearn.estimator import SKLearn

        FRAMEWORK_VERSION = "0.23-1"

        sklearn_estimator = SKLearn(
            entry_point="script.py",
            role="arn:aws:iam::339713029186:role/sage-role",
            instance_count=1,
            instance_type="ml.m5.large",
            framework_version=FRAMEWORK_VERSION,
            base_job_name="RF-custom-sklearn",
            hyperparameters={
                "n_estimators": 100,
                "random_state": 0,
            },
            use_spot_instances = True,
            max_wait = 7200,
            max_run = 3600
        )

In [4]: # launch training job, with asynchronous call
        sklearn_estimator.fit({"train": trainpath, "test": testpath}, wait=True)
        # sklearn_estimator.fit({"train": datapath}, wait=True)
```



```
In [ ]: ##Endpoints deployment
        endpoint_name = "Custom-sklearn-model-" + strftime("%Y-%m-%d-%H-%M-%S", gmtime())
        print("EndpointName={}".format(endpoint_name))

        predictor = model.deploy(
            initial_instance_count=1,
            instance_type="ml.m4.xlarge",
            endpoint_name=endpoint_name,
        )
```

16. TERRAFORM SCRIPT FOR INFRASTRUCTURE DEPLOYMENT:


app.js M
index.html M
script.tf .\
Polynomial Regression.ipynb
script.tf C:\...\Capstone-final
main.tf U
shijin.yaml
C: > Users > shiji > OneDrive > Documents > Cloud computing > sem2 > Capstone > Capstone-final > script.tf
Ln 18, Col 2 Spaces: 2 UTF-8 CRLF Plain Text Go Live

```
# Provider Configuration
provider "aws" {
  region = "us-east-1"
}

# VPC Configuration
resource "aws_vpc" "main" {
  cidr_block = "10.0.0.0/16"

  tags = {
    Name = "price-project"
  }
}

# Internet Gateway Configuration
resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main.id
}

# Public Subnets
resource "aws_subnet" "public1" {
  vpc_id                  = aws_vpc.main.id
  cidr_block              = "10.0.1.0/24"
  availability_zone       = "us-east-1a"
}

resource "aws_subnet" "public2" {
  vpc_id                  = aws_vpc.main.id
  cidr_block              = "10.0.2.0/24"
  availability_zone       = "us-east-1b"
}

# Private Subnets
resource "aws_subnet" "private1" {
  vpc_id                  = aws_vpc.main.id
  cidr_block              = "10.0.3.0/24"
  availability_zone       = "us-east-1a"
```

11.REFRENCES:

RESEARCH PAPER: On the Integration of Google Cloud and SAP HANA for Adaptive Supply Chain in Retailing - ScienceDirect

RESEARCH PAPER: (PDF) AI AND MACHINE LEARNING INTEGRATION WITH AWS SAGEMAKER: CURRENT TRENDS AND FUTURE PROSPECTS (researchgate.net)

Github Link For Our Codes

References

aws.amazon.com. (n.d.). Retrieved from https://aws.amazon.com/tutorials/machine-learning-tutorial-label-training-data/

aws.amazon.com/ec2. (n.d.). Retrieved from https://aws.amazon.com/ec2/instance-types/?p=itt#compute-optimized

builtin.com/random-forest-algorithm. (n.d.). Retrieved from https://builtin.com/data-science/random-forest-algorithm

developer.hashicorp.com. (n.d.). Retrieved from https://developer.hashicorp.com/terraform/tutorials/aws-get-started

docs.aws.amazon.com. (n.d.). Retrieved from https://docs.aws.amazon.com/lambda/latest/dg/getting-started.html

docs.aws.amazon.com. (n.d.). Retrieved from https://docs.aws.amazon.com/apigateway/latest/developerguide/getting-started.html

docs.aws.amazon.com/AmazonS3/latest. (n.d.). Retrieved from https://docs.aws.amazon.com/AmazonS3/latest/userguide/cors.html

www.kaggle.com. (n.d.). Retrieved from https://www.kaggle.com/datasets/suddharshan/retail-price-optimization

www.run.ai. (n.d.). Retrieved from https://www.run.ai/guides/machine-learning-in-the-cloud/sagemaker-tutorial