Configuration Manual

MSc Research Project
MSc in Cloud Computing

Sravanthi Challa
Student ID: 21156239

School of Computing
National College of Ireland

Supervisor: Shaguna Gupta

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Sravanthi Challa		
Student ID:	21156239		
Programme:	MSc in Cloud Computing	**Year:**	2023
Module:	MSc Research Project		
Lecturer:	Shaguna Gupta		
Submission Due Date:	14 December 2023		
Project Title:	Machine Learning & PBFT Blockchain Methodology on AWS for Proteomics Analytics		
Word Count:	…………… **Page Count:** ……………		

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……………………………………………………………………………………………………

Date: ……………………………………………………………………………………………………

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Sravanthi Challa
Student ID: 21156239

1 Introduction

Proteomics has transformed the study of proteins and provided an immense amount of new information about biological systems and disease states. However, proteomics data analysis is a difficult and computationally demanding task that frequently calls for advanced machine learning techniques. This configuration manual presents a thorough method of proteomics data analytics on Amazon Web Services (AWS) that makes use of PBFT blockchain technology and machine learning to overcome these obstacles.
The objective of this project is to improve protein identification and quantification accuracy by utilising machine learning, and to guarantee data confidentiality and integrity by integrating blockchain technology. For the purpose of deploying and managing the blockchain application and machine learning model, AWS offers a stable and scalable platform. The integration of these technologies has the potential to revolutionise proteomics research by facilitating more dependable and effective data analysis.

1.1 Target Audience

The purpose of this configuration manual is to assist researchers and developers who want to implement an AWS proteomics data analytics solution based on machine learning. From data collection and pre-processing to model training, deployment, and blockchain integration, it offers detailed guidance for every step of the procedure.

1.2 Prerequisites

Before proceeding with this configuration manual, ensure that you have the following prerequisites:

- Basic understanding of machine learning, Blockchain and proteomics
- Familiarity with AWS services, such as ECS, Docker, and Kubernetes
- Experience with Python

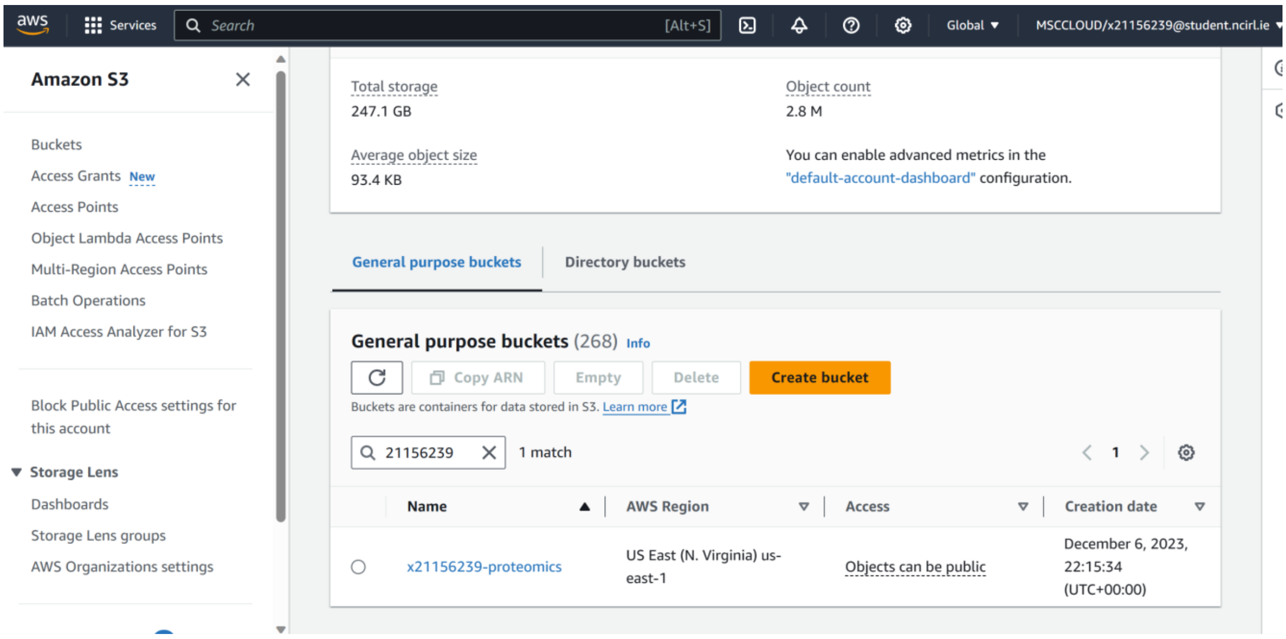
2 Data Collection and Pre-Processing

2.1 Data Collection

Proteomics data collection requires collecting information from a variety of experimental methods, including chromatography, NMR (Nuclear Magnetic Resonance), and mass spectrometry. These methods produce a variety of data formats, such as raw output files, chromatograms, and spectra. The data collection process is meticulous and crucial because it often involves large volumes, a variety of biological samples, and multi-dimensional features.

Data will be collected and stored in AWS S3 bucket.

1. First create AWS S3 bucket in AWS.



aws
Services
Search [Alt+S]
Global
MSCCLOUD/x21156239@student.ncirl.ie
Amazon S3
Buckets
Access Grants New
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings

Total storage: 247.1 GB
Object count: 2.8 M
Average object size: 93.4 KB
You can enable advanced metrics in the "default-account-dashboard" configuration.

General purpose buckets | Directory buckets

General purpose buckets (268) Info

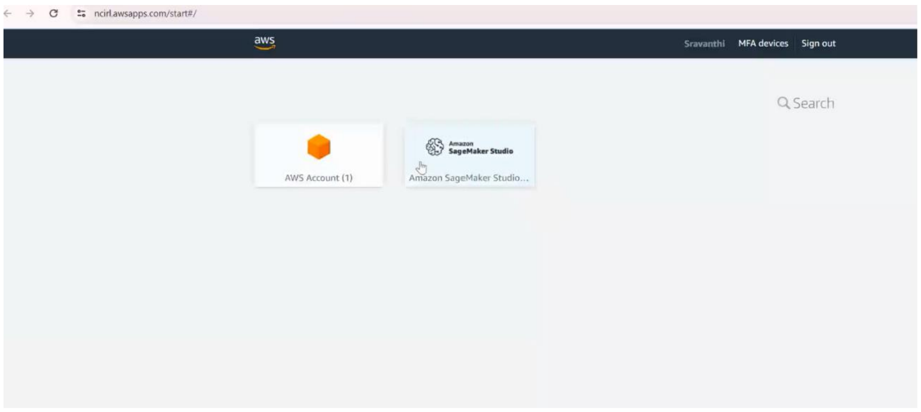
Copy ARN | Empty | Delete | Create bucket

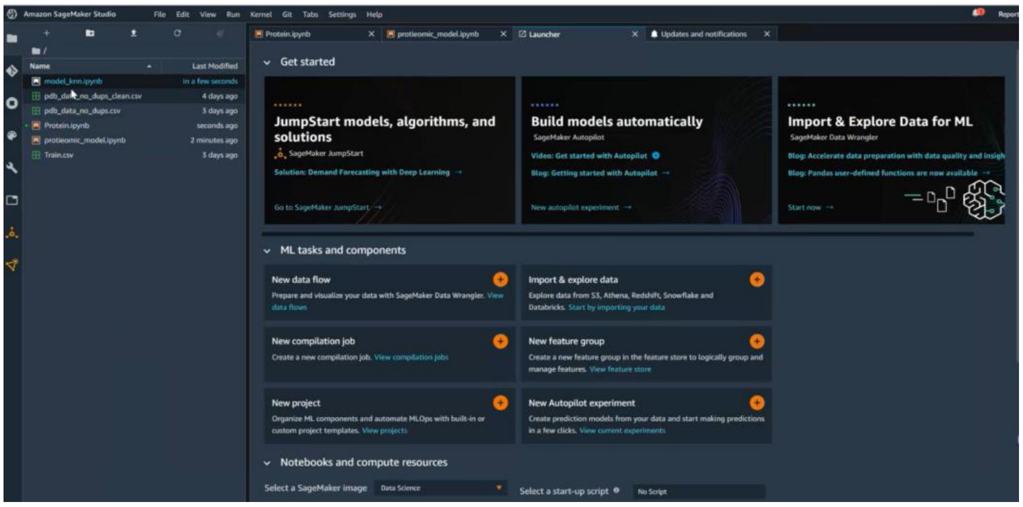
Buckets are containers for data stored in S3. Learn more

21156239 — 1 match

Name	AWS Region	Access	Creation date
x21156239-proteomics	US East (N. Virginia) us-east-1	Objects can be public	December 6, 2023, 22:15:34 (UTC+00:00)

2. Then check wheather you have access for SageMaker in AWS





Name	Last Modified
model_knn.ipynb	in a few seconds
pdb_data_no_dups_clean.csv	4 days ago
pdb_data_no_dups.csv	3 days ago
Protein.ipynb	seconds ago
protieomic_model.ipynb	2 minutes ago
Train.csv	3 days ago

Get started

JumpStart models, algorithms, and solutions
SageMaker JumpStart
Solution: Demand Forecasting with Deep Learning →
Go to SageMaker JumpStart →

Build models automatically
SageMaker Autopilot
Video: Get started with Autopilot
Blog: Getting started with Autopilot →
New autopilot experiment →

Import & Explore Data for ML
SageMaker Data Wrangler
Blog: Accelerate data preparation with data quality and insigh
Blog: Pandas user-defined functions are now available →
Start now →

ML tasks and components

New data flow
Prepare and visualize your data with SageMaker Data Wrangler. View data flows

Import & explore data
Explore data from S3, Athena, Redshift, Snowflake and Databricks. Start by importing your data

New compilation job
Create a new compilation job. View compilation jobs

New feature group
Create a new feature group in the feature store to logically group and manage features. View feature store

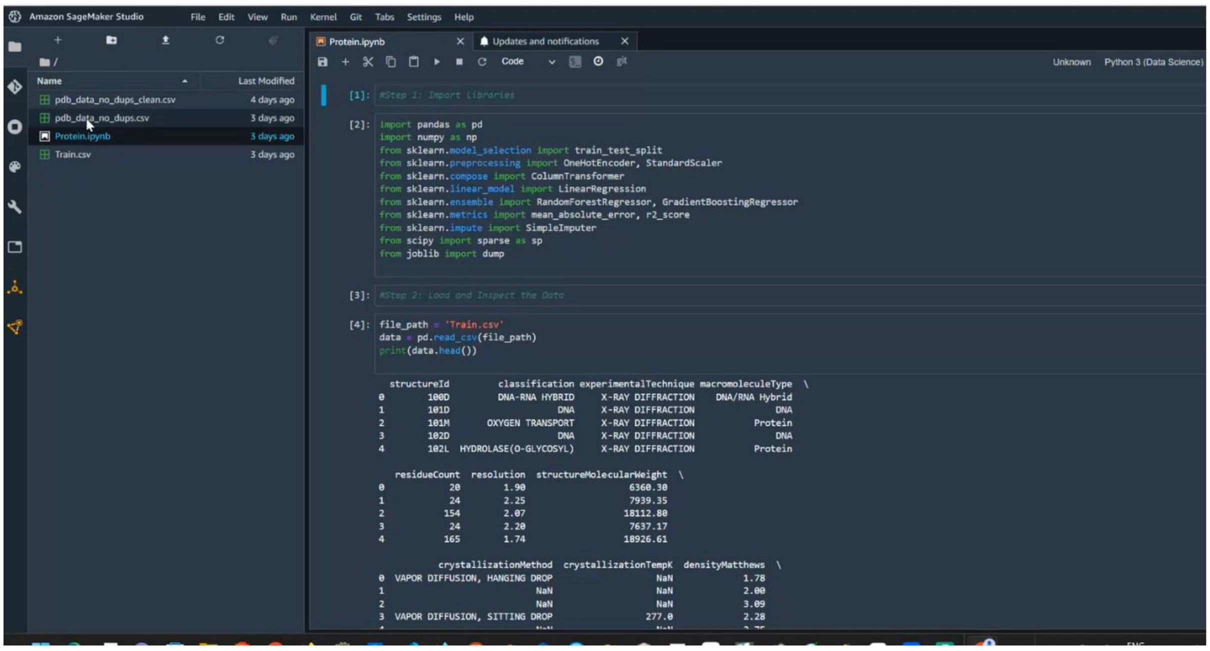
New project
Organize ML components and automate MLOps with built-in or custom project templates. View projects

New Autopilot experiment
Create prediction models from your data and start making predictions in a few clicks. View current experiments

Notebooks and compute resources

Select a SageMaker image: Data Science
Select a start-up script: No Script

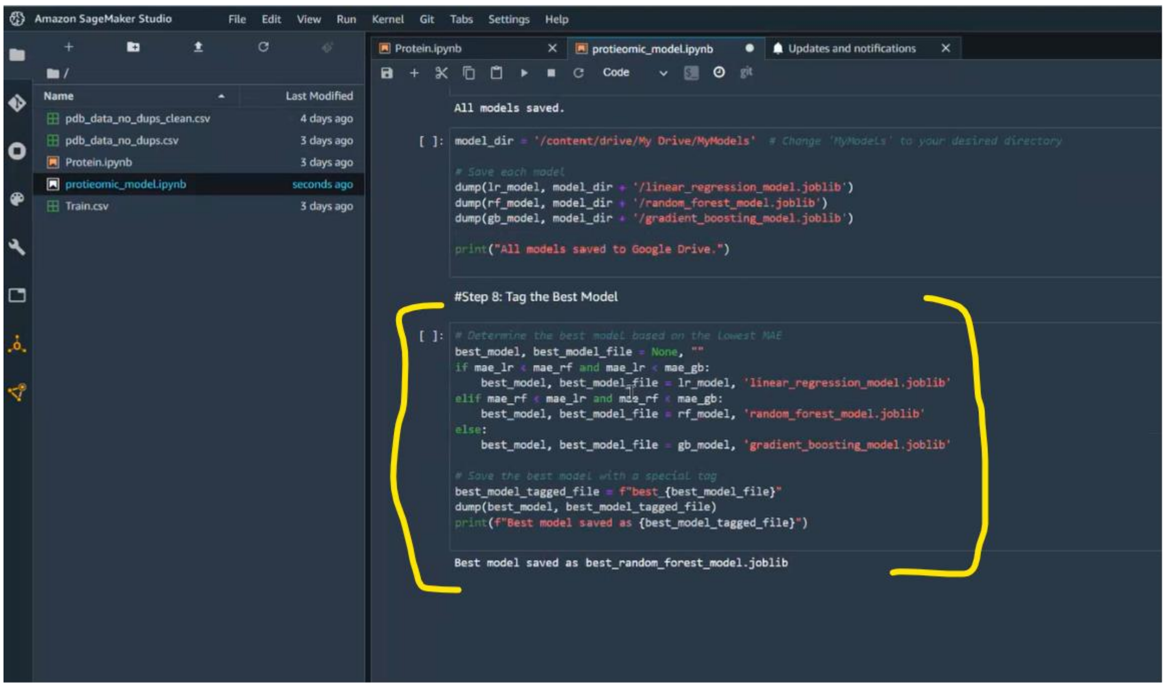
3. Go to SageMaker and create the machine learning algorithm file and upload PDB file. Here, the SageMaker will train the data for different models.



Amazon SageMaker Studio
File Edit View Run Kernel Git Tabs Settings Help
Name
Last Modified
pdb_data_no_dups_clean.csv 4 days ago
pdb_data_no_dups.csv 3 days ago
Protein.ipynb 3 days ago
Train.csv 3 days ago
Protein.ipynb
Updates and notifications
Code
Unknown
Python 3 (Data Science)

```
[1]: #Step 1: Import Libraries

[2]: import pandas as pd
     import numpy as np
     from sklearn.model_selection import train_test_split
     from sklearn.preprocessing import OneHotEncoder, StandardScaler
     from sklearn.compose import ColumnTransformer
     from sklearn.linear_model import LinearRegression
     from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
     from sklearn.metrics import mean_absolute_error, r2_score
     from sklearn.impute import SimpleImputer
     from scipy import sparse as sp
     from joblib import dump

[3]: #Step 2: Load and Inspect the Data

[4]: file_path = 'Train.csv'
     data = pd.read_csv(file_path)
     print(data.head())

  structureId         classification experimentalTechnique macromoleculeType  \
0        100D          DNA-RNA HYBRID     X-RAY DIFFRACTION    DNA/RNA Hybrid
1        101D                     DNA     X-RAY DIFFRACTION               DNA
2        101M       OXYGEN TRANSPORT     X-RAY DIFFRACTION           Protein
3        102D                     DNA     X-RAY DIFFRACTION               DNA
4        102L  HYDROLASE(O-GLYCOSYL)     X-RAY DIFFRACTION           Protein

   residueCount  resolution  structureMolecularWeight  \
0            20        1.90                   6360.30
1            24        2.25                   7939.35
2           154        2.07                  18112.80
3            24        2.20                   7637.17
4           165        1.74                  18926.61

          crystallizationMethod  crystallizationTempK  densityMatthews  \
0  VAPOR DIFFUSION, HANGING DROP                   NaN             1.78
1                           NaN                   NaN             2.00
2                           NaN                   NaN             3.09
3  VAPOR DIFFUSION, SITTING DROP                 277.0             2.28
```

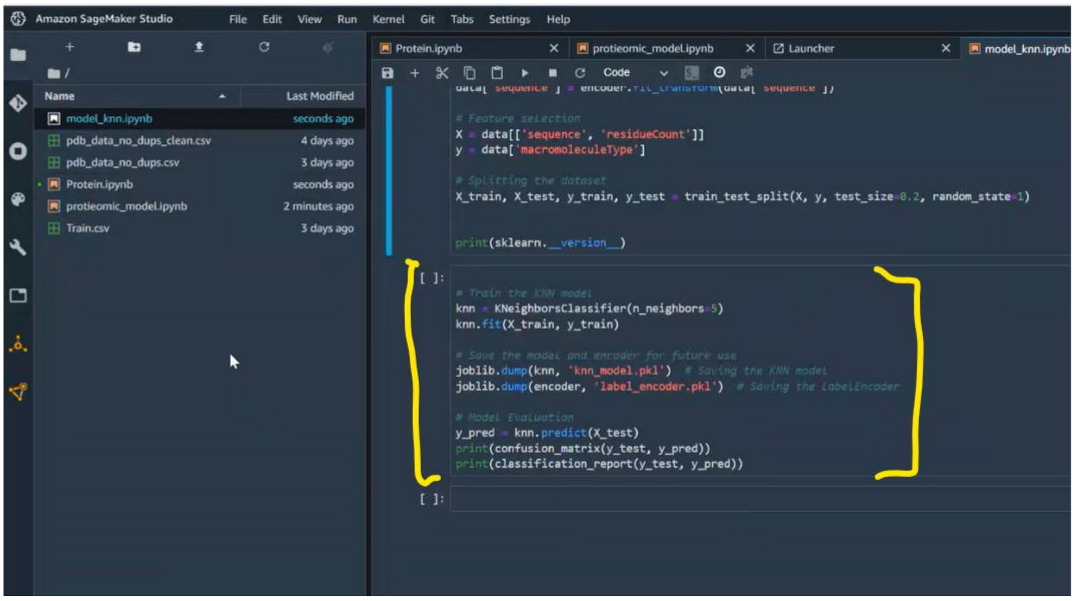
4. We will be declaring the best model prediction in SageMaker in our algorithm.



Amazon SageMaker Studio
File Edit View Run Kernel Git Tabs Settings Help
Name
Last Modified
pdb_data_no_dups_clean.csv 4 days ago
pdb_data_no_dups.csv 3 days ago
Protein.ipynb 3 days ago
protieomic_model.ipynb seconds ago
Train.csv 3 days ago
Protein.ipynb
protieomic_model.ipynb
Updates and notifications
Code

```
All models saved.

[ ]: model_dir = '/content/drive/My Drive/MyModels'  # Change 'MyModels' to your desired directory

     # Save each model
     dump(lr_model, model_dir + '/linear_regression_model.joblib')
     dump(rf_model, model_dir + '/random_forest_model.joblib')
     dump(gb_model, model_dir + '/gradient_boosting_model.joblib')

     print("All models saved to Google Drive.")

#Step 8: Tag the Best Model

[ ]: # Determine the best model based on the lowest MAE
     best_model, best_model_file = None, ""
     if mae_lr < mae_rf and mae_lr < mae_gb:
         best_model, best_model_file = lr_model, 'linear_regression_model.joblib'
     elif mae_rf < mae_lr and mae_rf < mae_gb:
         best_model, best_model_file = rf_model, 'random_forest_model.joblib'
     else:
         best_model, best_model_file = gb_model, 'gradient_boosting_model.joblib'

     # Save the best model with a special tag
     best_model_tagged_file = f"best_{best_model_file}"
     dump(best_model, best_model_tagged_file)
     print(f"Best model saved as {best_model_tagged_file}")

Best model saved as best_random_forest_model.joblib
```

5. Write down the model training code.

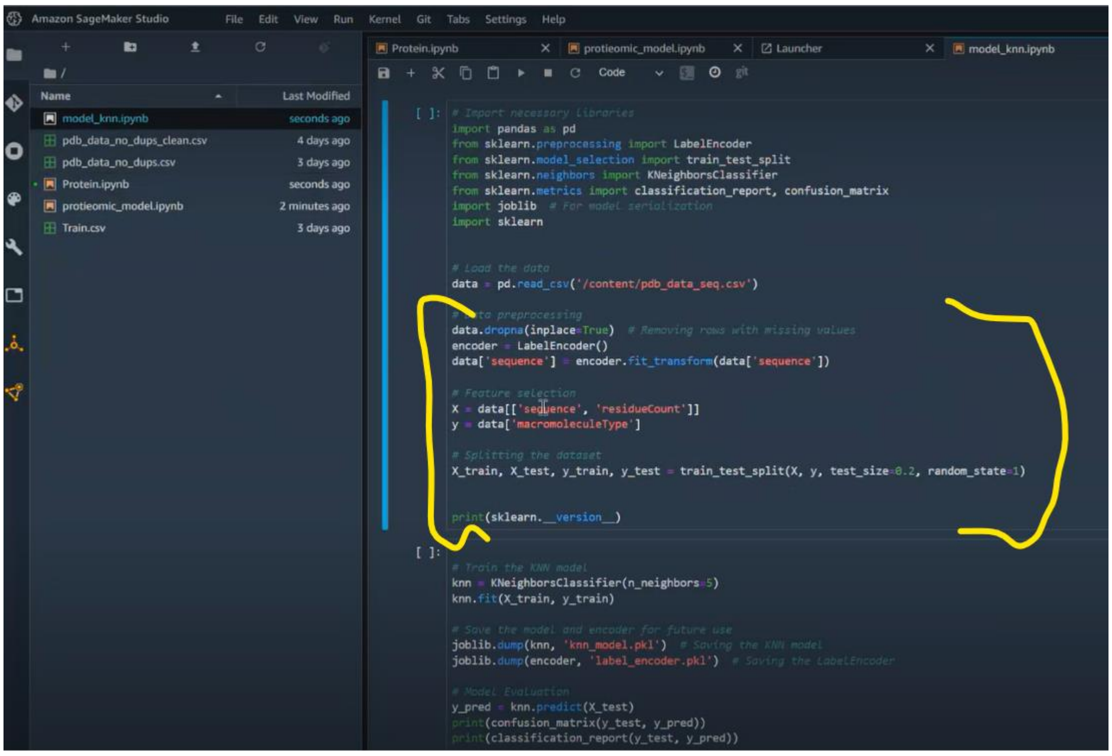


Name	Last Modified
model_knn.ipynb	seconds ago
pdb_data_no_dups_clean.csv	4 days ago
pdb_data_no_dups.csv	3 days ago
Protein.ipynb	seconds ago
protieomic_model.ipynb	2 minutes ago
Train.csv	3 days ago

```
data['sequence'] = encoder.fit_transform(data['sequence'])

# Feature selection
X = data[['sequence', 'residueCount']]
y = data['macromoleculeType']

# Splitting the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)


print(sklearn.__version__)
```

```
[ ]:
# Train the KNN model
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)

# Save the model and encoder for future use
joblib.dump(knn, 'knn_model.pkl')  # Saving the KNN model
joblib.dump(encoder, 'label_encoder.pkl')  # Saving the LabelEncoder

# Model Evaluation
y_pred = knn.predict(X_test)
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))

[ ]:
```

6. Data pre-processing declaration is done, Prediction of protiens will be done by X and Y-axis data

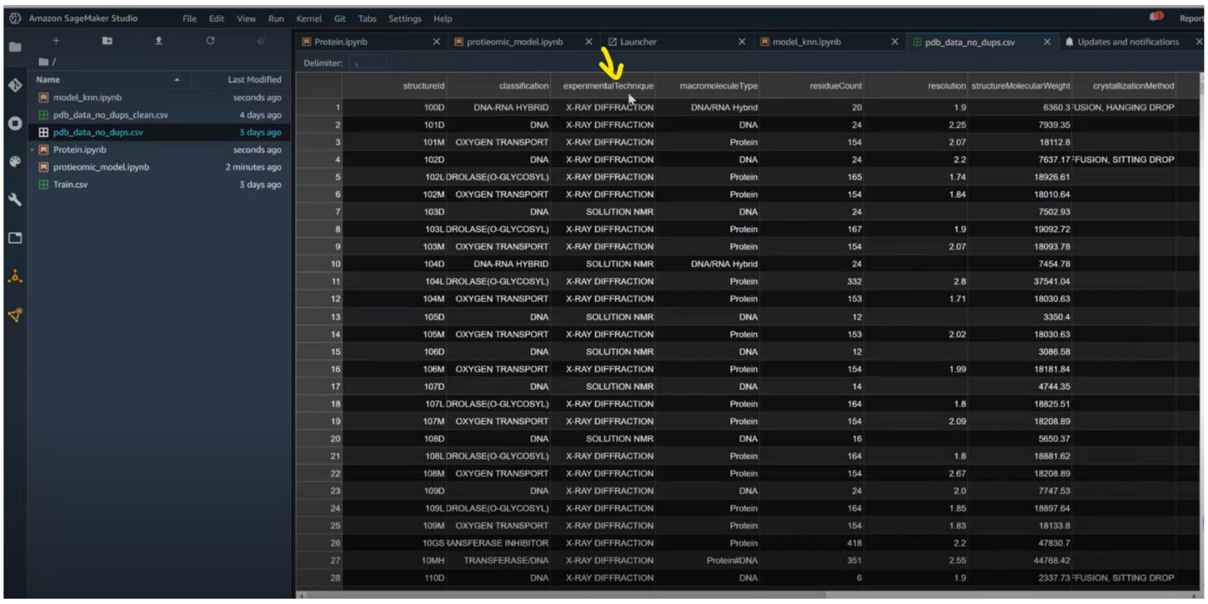


Name	Last Modified
model_knn.ipynb	seconds ago
pdb_data_no_dups_clean.csv	4 days ago
pdb_data_no_dups.csv	3 days ago
Protein.ipynb	seconds ago
protieomic_model.ipynb	2 minutes ago
Train.csv	3 days ago

```
[ ]: # Import necessary libraries
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, confusion_matrix
import joblib  # For model serialization
import sklearn


# Load the data
data = pd.read_csv('/content/pdb_data_seq.csv')

# Data preprocessing
data.dropna(inplace=True)  # Removing rows with missing values
encoder = LabelEncoder()
data['sequence'] = encoder.fit_transform(data['sequence'])

# Feature selection
X = data[['sequence', 'residueCount']]
y = data['macromoleculeType']

# Splitting the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)


print(sklearn.__version__)
```

```
[ ]:
# Train the KNN model
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)

# Save the model and encoder for future use
joblib.dump(knn, 'knn_model.pkl')  # Saving the KNN model
joblib.dump(encoder, 'label_encoder.pkl')  # Saving the LabelEncoder

# Model Evaluation
y_pred = knn.predict(X_test)
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

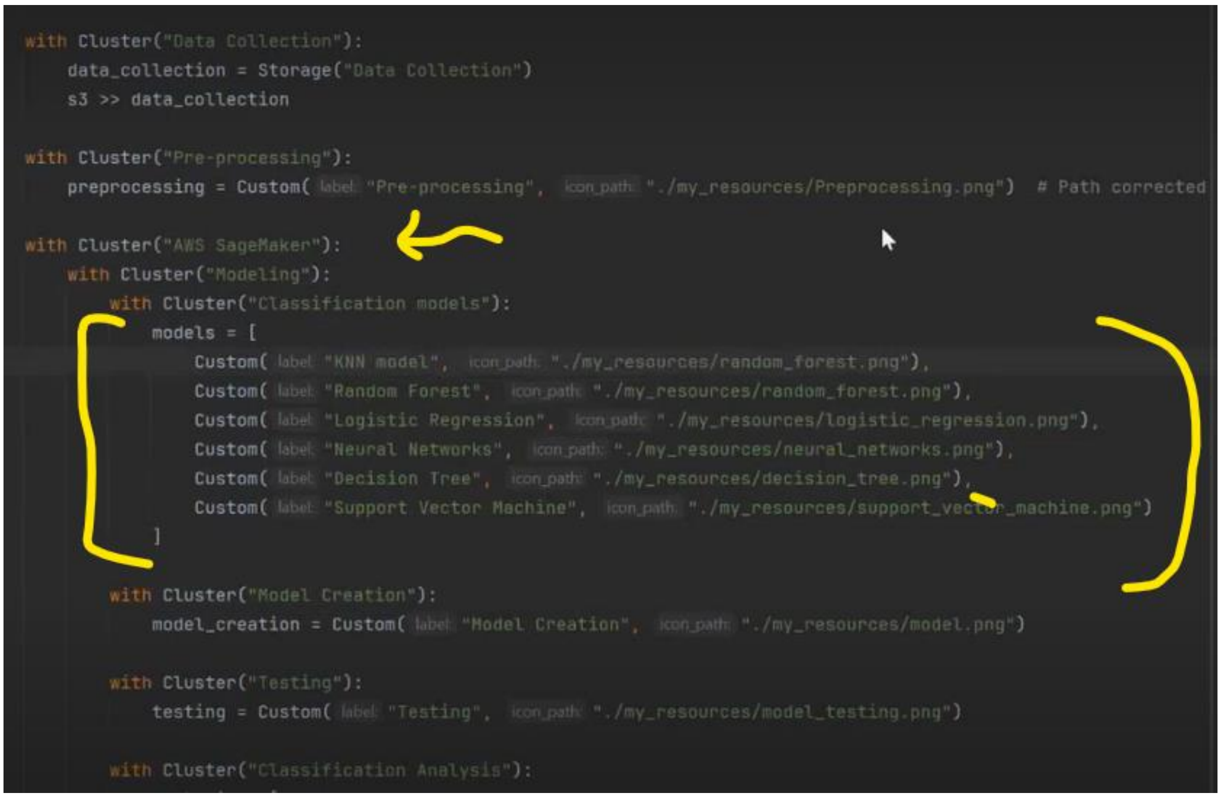
7. Based on the macromolecule, Sequence and residuecount the prediction is done. The column can be seen in the data file.



Name	Last Modified
model_knn.ipynb	seconds ago
pdb_data_no_dups_clean.csv	4 days ago
pdb_data_no_dups.csv	3 days ago
Protein.ipynb	seconds ago
protieomic_model.ipynb	2 minutes ago
Train.csv	3 days ago

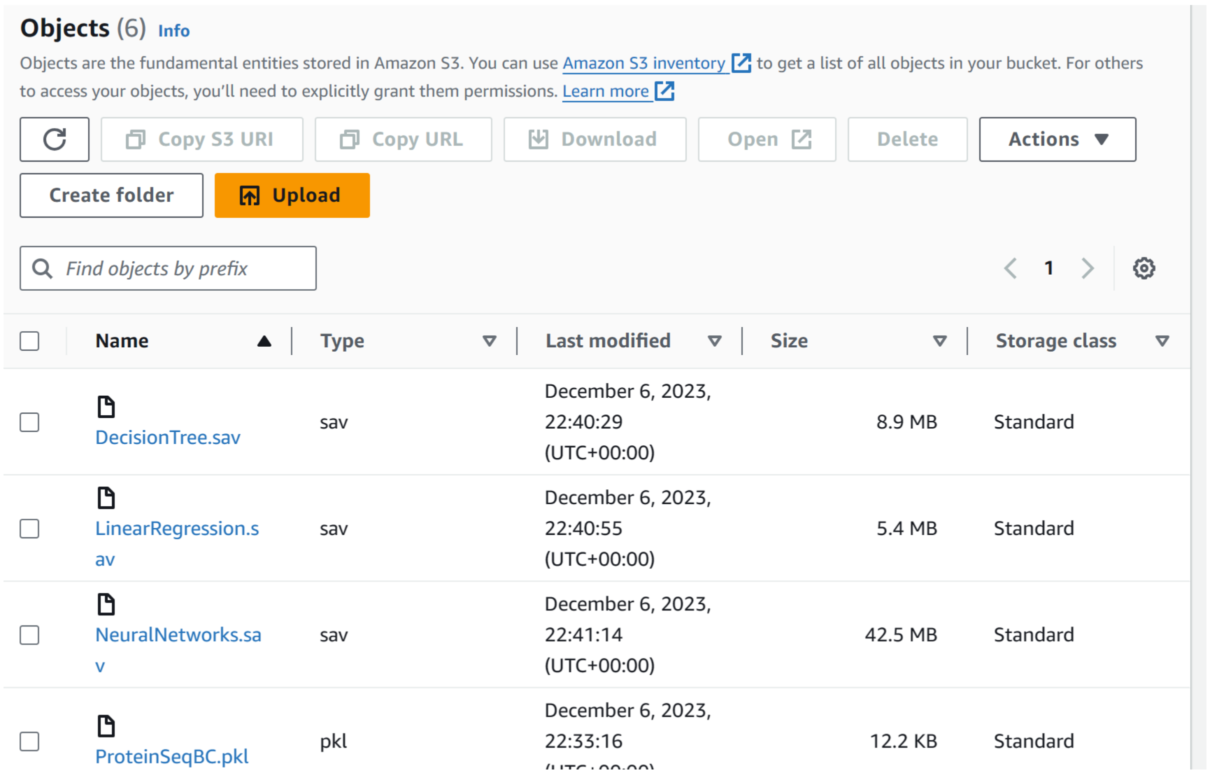
	structureId	classification	experimentalTechnique	macromoleculeType	residueCount	resolution	structureMolecularWeight	crystallizationMethod
1	100D	DNA-RNA HYBRID	X-RAY DIFFRACTION	DNA/RNA Hybrid	20	1.9	6360.3	USION, HANGING DROP
2	101D	DNA	X-RAY DIFFRACTION	DNA	24	2.25	7939.35	
3	101M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	154	2.07	18112.8	
4	102D	DNA	X-RAY DIFFRACTION	DNA	24	2.2	7637.17	FUSION, SITTING DROP
5	102L	DROLASE(O-GLYCOSYL)	X-RAY DIFFRACTION	Protein	165	1.74	18926.61	
6	102M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	154	1.84	18010.64	
7	103D	DNA	SOLUTION NMR	DNA	24		7502.93	
8	103L	DROLASE(O-GLYCOSYL)	X-RAY DIFFRACTION	Protein	167	1.9	19092.72	
9	103M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	154	2.07	18093.78	
10	104D	DNA-RNA HYBRID	SOLUTION NMR	DNA/RNA Hybrid	24		7454.78	
11	104L	DROLASE(O-GLYCOSYL)	X-RAY DIFFRACTION	Protein	332	2.8	37541.04	
12	104M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	153	1.71	18030.63	
13	105D	DNA	SOLUTION NMR	DNA	12		3350.4	
14	105M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	153	2.02	18030.63	
15	106D	DNA	SOLUTION NMR	DNA	12		3086.58	
16	106M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	154	1.99	18181.84	
17	107D	DNA	SOLUTION NMR	DNA	14		4744.35	
18	107L	DROLASE(O-GLYCOSYL)	X-RAY DIFFRACTION	Protein	164	1.8	18825.51	
19	107M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	154	2.09	18208.89	
20	108D	DNA	SOLUTION NMR	DNA	16		5650.37	
21	108L	DROLASE(O-GLYCOSYL)	X-RAY DIFFRACTION	Protein	164	1.8	18881.62	
22	108M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	154	2.67	18208.89	
23	109D	DNA	X-RAY DIFFRACTION	DNA	24	2.0	7747.53	
24	109L	DROLASE(O-GLYCOSYL)	X-RAY DIFFRACTION	Protein	164	1.85	18897.64	
25	109M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	154	1.83	18133.8	
26	10GS	RANSFERASE INHIBITOR	X-RAY DIFFRACTION	Protein	418	2.2	47830.7	
27	10MH	TRANSFERASE/DNA	X-RAY DIFFRACTION	Protein#DNA	351	2.55	44768.42	
28	110D	DNA	X-RAY DIFFRACTION	DNA	6	1.9	2337.73	FUSION, SITTING DROP

8. Sagemaker is connected to our streamlit application, and cluster is been created.



```
with Cluster("Data Collection"):
    data_collection = Storage("Data Collection")
    s3 >> data_collection

with Cluster("Pre-processing"):
    preprocessing = Custom( label: "Pre-processing",  icon_path: "./my_resources/Preprocessing.png")  # Path corrected

with Cluster("AWS SageMaker"):
    with Cluster("Modeling"):
        with Cluster("Classification models"):
            models = [
                Custom( label: "KNN model",  icon_path: "./my_resources/random_forest.png"),
                Custom( label: "Random Forest",  icon_path: "./my_resources/random_forest.png"),
                Custom( label: "Logistic Regression",  icon_path: "./my_resources/logistic_regression.png"),
                Custom( label: "Neural Networks",  icon_path: "./my_resources/neural_networks.png"),
                Custom( label: "Decision Tree",  icon_path: "./my_resources/decision_tree.png"),
                Custom( label: "Support Vector Machine",  icon_path: "./my_resources/support_vector_machine.png")
            ]

        with Cluster("Model Creation"):
            model_creation = Custom( label: "Model Creation",  icon_path: "./my_resources/model.png")

        with Cluster("Testing"):
            testing = Custom( label: "Testing",  icon_path: "./my_resources/model_testing.png")

        with Cluster("Classification Analysis"):
```

9. Once we run the model training it will create a pickle file (PKL) of the that particular model in S3 bucket

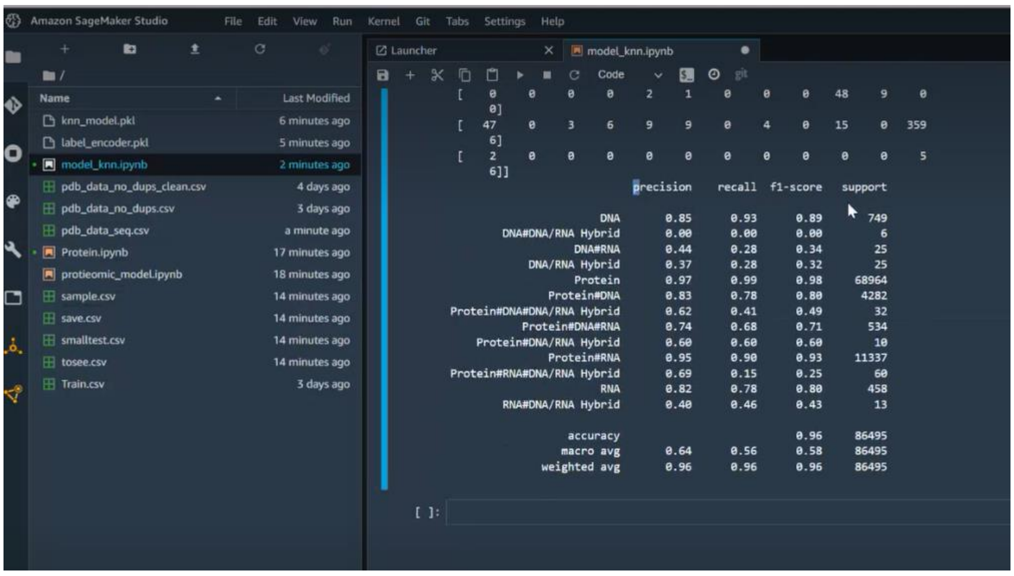


Objects (6) Info

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
DecisionTree.sav	sav	December 6, 2023, 22:40:29 (UTC+00:00)	8.9 MB	Standard
LinearRegression.sav	sav	December 6, 2023, 22:40:55 (UTC+00:00)	5.4 MB	Standard
NeuralNetworks.sav	sav	December 6, 2023, 22:41:14 (UTC+00:00)	42.5 MB	Standard
ProteinSeqBC.pkl	pkl	December 6, 2023, 22:33:16	12.2 KB	Standard

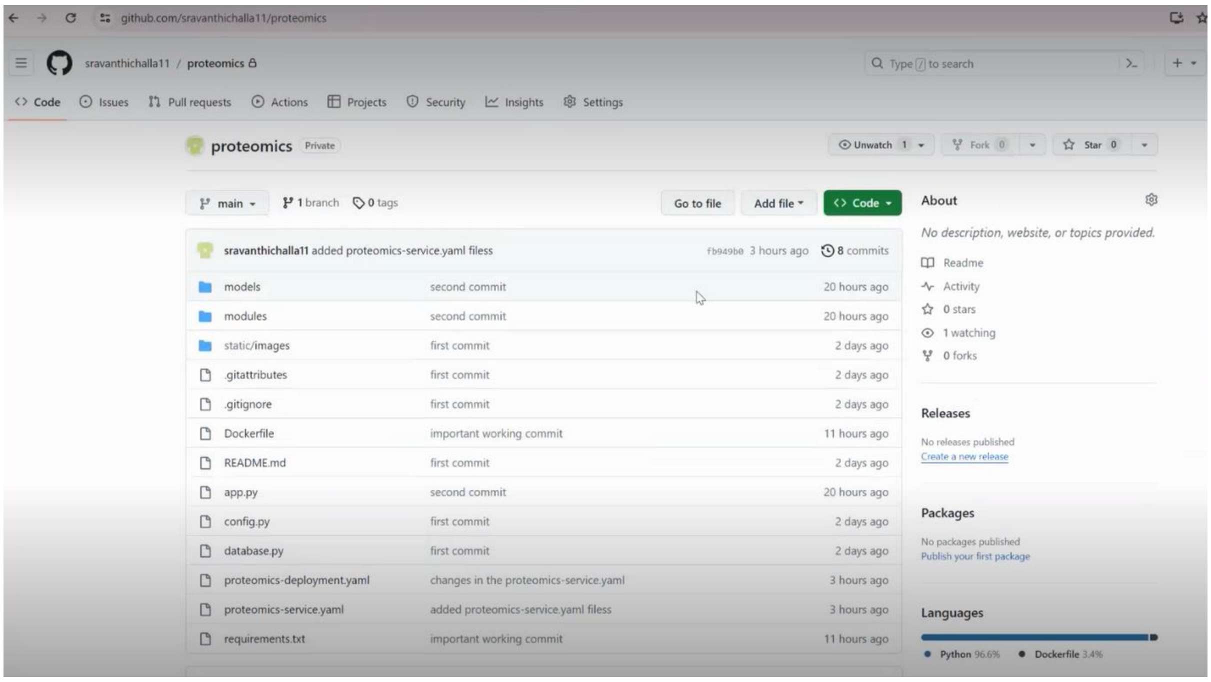
10. The trained file, gives us precision, recall, f1score and support



Name	Last Modified
knn_model.pkl	6 minutes ago
label_encoder.pkl	5 minutes ago
model_knn.ipynb	2 minutes ago
pdb_data_no_dups_clean.csv	4 days ago
pdb_data_no_dups.csv	3 days ago
pdb_data_seq.csv	a minute ago
Protein.ipynb	17 minutes ago
protieomic_model.ipynb	18 minutes ago
sample.csv	14 minutes ago
save.csv	14 minutes ago
smalltest.csv	14 minutes ago
tosee.csv	14 minutes ago
Train.csv	3 days ago

```
 [   0    0    0    0    2    1    0    0    0   48    9    0
     0]
 [  47    0    3    6    9    9    0    4    0   15    0  359
     6]
 [   2    0    0    0    0    0    0    0    0    0    0    5
     6]]
                            precision    recall  f1-score   support

                       DNA       0.85      0.93      0.89       749
        DNA#DNA/RNA Hybrid       0.00      0.00      0.00         6
                   DNA#RNA       0.44      0.28      0.34        25
            DNA/RNA Hybrid       0.37      0.28      0.32        25
                   Protein       0.97      0.99      0.98     68964
               Protein#DNA       0.83      0.78      0.80      4282
Protein#DNA#DNA/RNA Hybrid       0.62      0.41      0.49        32
           Protein#DNA#RNA       0.74      0.68      0.71       534
    Protein#DNA/RNA Hybrid       0.60      0.60      0.60        10
               Protein#RNA       0.95      0.90      0.93     11337
Protein#RNA#DNA/RNA Hybrid       0.69      0.15      0.25        60
                       RNA       0.82      0.78      0.80       458
        RNA#DNA/RNA Hybrid       0.40      0.46      0.43        13

                  accuracy                           0.96     86495
                 macro avg       0.64      0.56      0.58     86495
              weighted avg       0.96      0.96      0.96     86495

[ ]:
```

11. Create Github repository and upload all your UI code in it.



github.com/sravanthichalla11/proteomics

sravanthichalla11 / proteomics

Code · Issues · Pull requests · Actions · Projects · Security · Insights · Settings

proteomics Private

main · 1 branch · 0 tags · Go to file · Add file · Code

sravanthichalla11 added proteomics-service.yaml filess · fb949b0 3 hours ago · 8 commits

models	second commit	20 hours ago
modules	second commit	20 hours ago
static/images	first commit	2 days ago
.gitattributes	first commit	2 days ago
.gitignore	first commit	2 days ago
Dockerfile	important working commit	11 hours ago
README.md	first commit	2 days ago
app.py	second commit	20 hours ago
config.py	first commit	2 days ago
database.py	first commit	2 days ago
proteomics-deployment.yaml	changes in the proteomics-service.yaml	3 hours ago
proteomics-service.yaml	added proteomics-service.yaml filess	3 hours ago
requirements.txt	important working commit	11 hours ago

About

No description, website, or topics provided.

Readme · Activity · 0 stars · 1 watching · 0 forks

Releases

No releases published
Create a new release

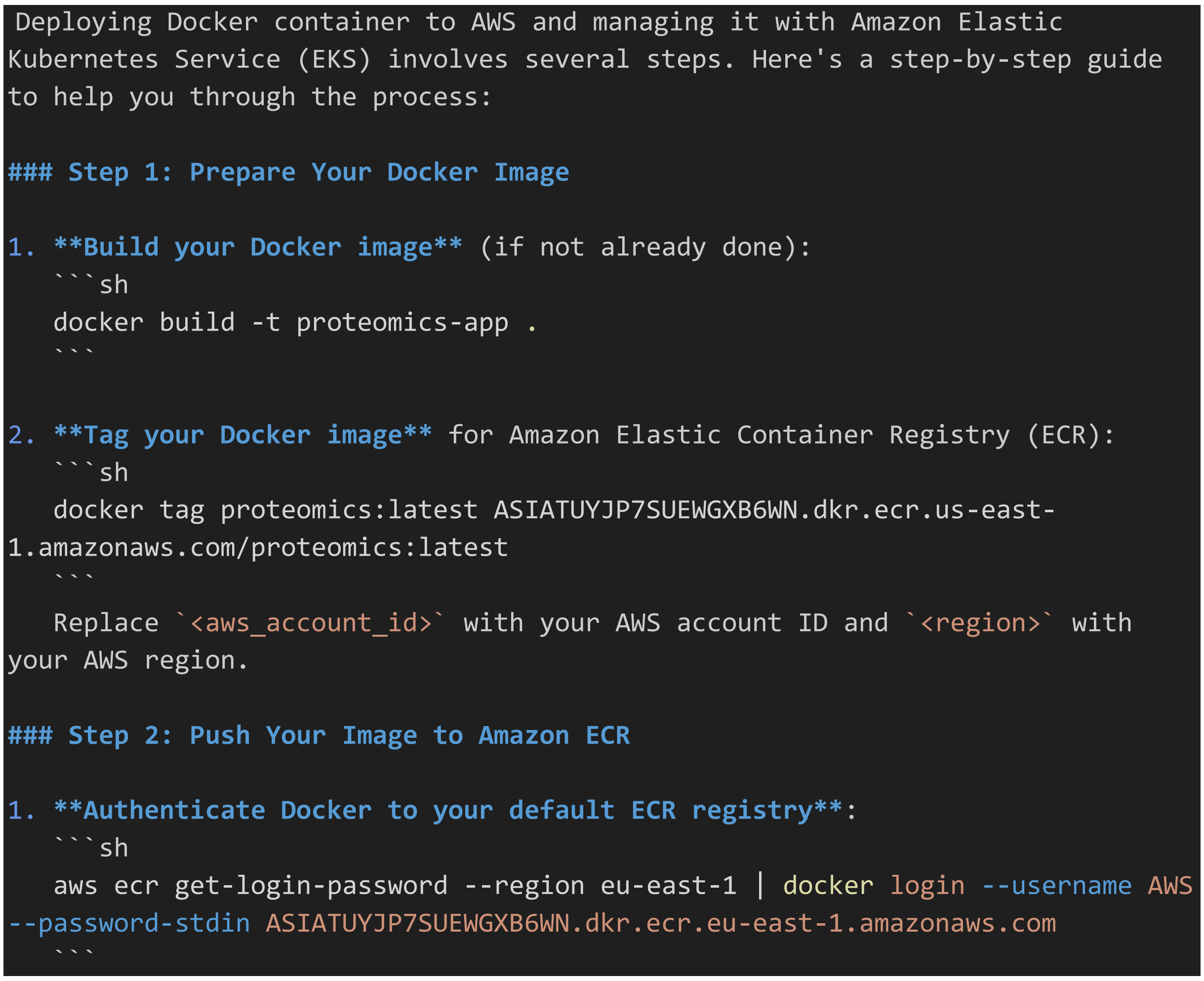
Packages

No packages published
Publish your first package

Languages

Python 96.6% · Dockerfile 3.4%

12. For creating cluster, Docker and Kubernetes setup has to be done.



~~~
Deploying Docker container to AWS and managing it with Amazon Elastic
Kubernetes Service (EKS) involves several steps. Here's a step-by-step guide
to help you through the process:

### Step 1: Prepare Your Docker Image

1. **Build your Docker image** (if not already done):
   ```sh
   docker build -t proteomics-app .
   ~~~

2. **Tag your Docker image** for Amazon Elastic Container Registry (ECR):
   ```sh
   docker tag proteomics:latest ASIATUYJP7SUEWGXB6WN.dkr.ecr.us-east-
1.amazonaws.com/proteomics:latest
   ```
   Replace `<aws_account_id>` with your AWS account ID and `<region>` with
your AWS region.

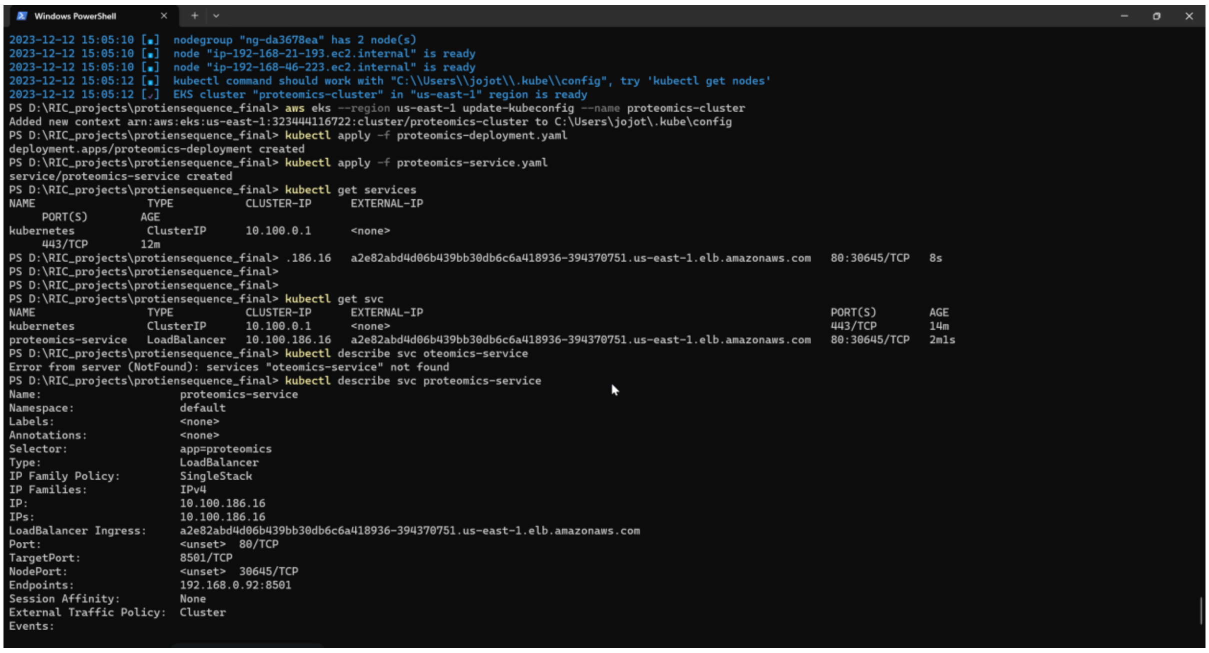
Step 2: Push Your Image to Amazon ECR

1. **Authenticate Docker to your default ECR registry**:
   ```sh
   aws ecr get-login-password --region eu-east-1 | docker login --username AWS
--password-stdin ASIATUYJP7SUEWGXB6WN.dkr.ecr.eu-east-1.amazonaws.com
   ```
```
```

~~~
2. **Create an ECR repository** (if you haven't already):
   ```sh
   aws ecr create-repository --repository-name proteomicsRepo --region us-
east-1
   ```

3. **Push your Docker image to ECR**:
   ```sh
   docker push ASIATUYJP7SUEWGXB6WN.dkr.ecr.eu-east-
1.amazonaws.com/proteomics-app:latest
   ```

### Step 3: Set Up Amazon EKS

1. **Create an EKS cluster**. This can be done via the AWS Management Console
or using AWS CLI. The CLI command is:
   ```sh
   eksctl create cluster --name proteomics-cluster --version 1.28 --region us-
east-1 --nodegroup-name standard-workers --node-type t3.medium --nodes 1 --
nodes-min 1 --nodes-max 1 --managed
   ```

2. **Configure `kubectl` to communicate with your cluster**:
   ```sh
   aws eks --region us-east-1 update-kubeconfig --name proteomics-cluster
   ```

### Step 4: Deploy Your Application on EKS

1. **Create a Kubernetes deployment**. You need a deployment YAML file (e.g.,
`proteomics-deployment.yaml`) that references your ECR image. Here's an
example of what this file might look like:

   ```yaml
   apiVersion: apps/v1
   kind: Deployment
   metadata:
     name: proteomics
   spec:
     replicas: 2
     selector:
       matchLabels:
         app: proteomics
     template:
       metadata:
         labels:
           app: proteomics
       spec:
~~~

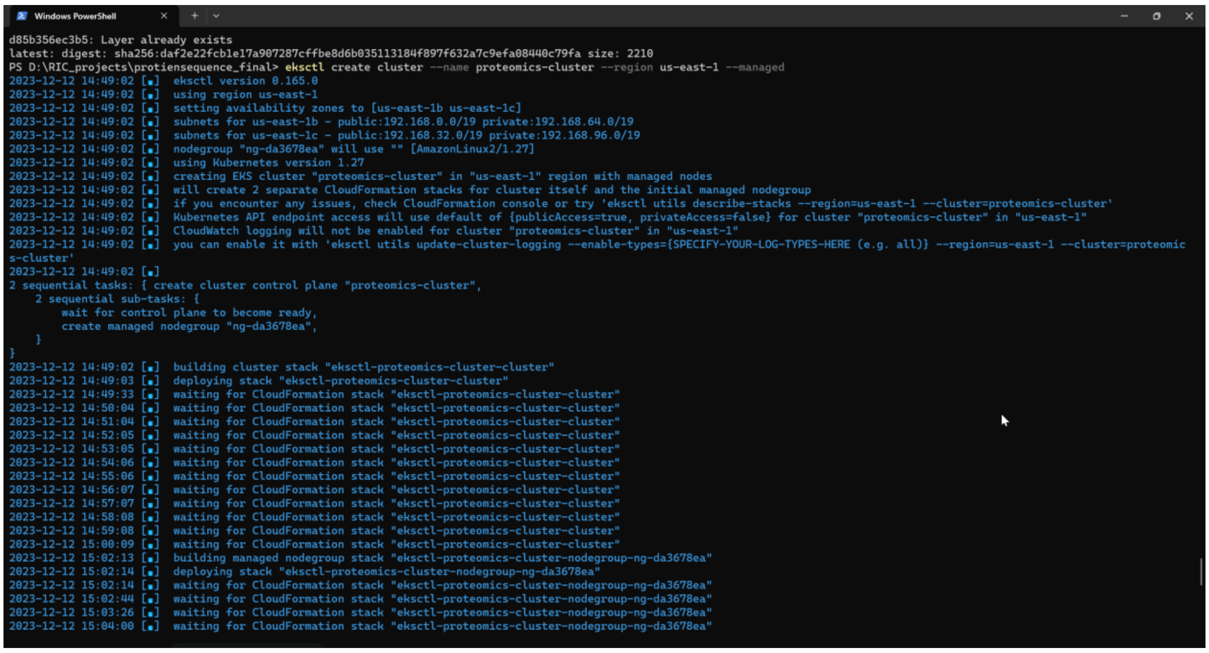
~~~
        containers:
        - name: proteomics
          image: ASIATUYJP7SUEWGXB6WN.dkr.ecr.us-east-
1.amazonaws.com/proteomics-app:latest
          ports:
          - containerPort: 8501
   ```

2. **Deploy the application**:
   ```sh
   kubectl apply -f proteomics-deployment.yaml
   ```

3. **Expose the application** (e.g., using a LoadBalancer service):
   ```yaml
   apiVersion: v1
   kind: Service
   metadata:
     name: proteomics-app-service
   spec:
     type: LoadBalancer
     ports:
     - port: 80
       targetPort: 8501
     selector:
       app: proteomics-app
   ```

   Then apply this configuration:
   ```sh
   kubectl apply -f proteomics-service.yaml
   ```

4. **Access your application**. After a few minutes, get the LoadBalancer URL:
   ```sh
   kubectl get services
   ```
   Look for the `EXTERNAL-IP` of your `proteomics-app-service`.

### Step 5: Monitoring and Management

- Use Kubernetes Dashboard or AWS CloudWatch for monitoring.
- Set up autoscaling if needed.
- Regularly update your application with security patches.




working ECR:
~~~

```
1. Retrieve an authentication token and authenticate your Docker client to
your registry.

    Use the AWS CLI:

    aws ecr-public get-login-password --region us-east-1 | docker login --
username AWS --password-stdin public.ecr.aws/y3d9e0t2

    Note: If you receive an error using the AWS CLI, make sure that you have
the latest version of the AWS CLI and Docker installed.
2. Build your Docker image using the following command. For information on
building a Docker file from scratch see the
instructions [here](http://docs.aws.amazon.com/AmazonECS/latest/developerguide
/docker-basics.html) . You can skip this step if your image is already built:

    docker build -t x21156239_proteomics-ecr .

3. After the build completes, tag your image so you can push the image to this
repository:

    docker tag x21156239_proteomics-ecr:latest
public.ecr.aws/y3d9e0t2/x21156239_proteomics-ecr:latest

4. Run the following command to push this image to your newly created AWS
repository:

    docker push public.ecr.aws/y3d9e0t2/x21156239_proteomics-ecr:latest
```



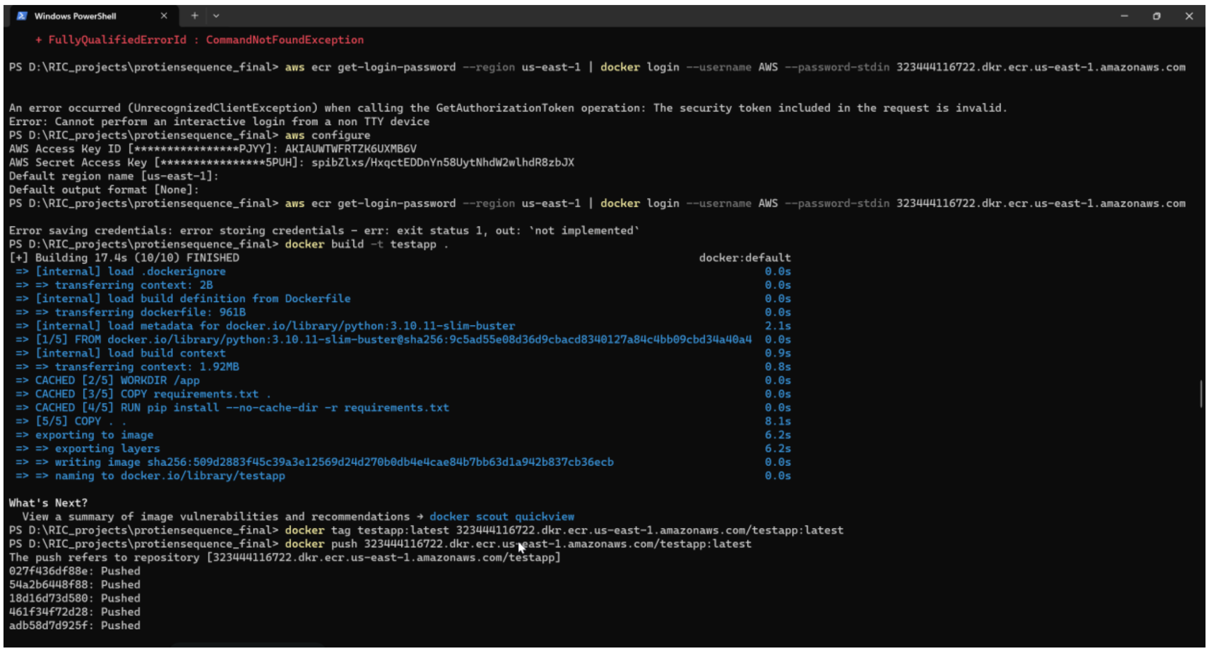
```
2023-12-12 15:05:10 [▪]  nodegroup "ng-da3678ea" has 2 node(s)
2023-12-12 15:05:10 [▪]  node "ip-192-168-21-193.ec2.internal" is ready
2023-12-12 15:05:10 [▪]  node "ip-192-168-46-223.ec2.internal" is ready
2023-12-12 15:05:12 [▪]  kubectl command should work with "C:\\Users\\jojot\\.kube\\config", try 'kubectl get nodes'
2023-12-12 15:05:12 [✓]  EKS cluster "proteomics-cluster" in "us-east-1" region is ready
PS D:\RIC_projects\protiensequence_final> aws eks --region us-east-1 update-kubeconfig --name proteomics-cluster
Added new context arn:aws:eks:us-east-1:323444116722:cluster/proteomics-cluster to C:\Users\jojot\.kube\config
PS D:\RIC_projects\protiensequence_final> kubectl apply -f proteomics-deployment.yaml
deployment.apps/proteomics-deployment created
PS D:\RIC_projects\protiensequence_final> kubectl apply -f proteomics-service.yaml
service/proteomics-service created
PS D:\RIC_projects\protiensequence_final> kubectl get services
NAME                 TYPE           CLUSTER-IP      EXTERNAL-IP
     PORT(S)        AGE
kubernetes           ClusterIP      10.100.0.1      <none>
     443/TCP        12m
PS D:\RIC_projects\protiensequence_final> .186.16   a2e82abd4d06b439bb30db6c6a418936-394370751.us-east-1.elb.amazonaws.com   80:30645/TCP   8s
PS D:\RIC_projects\protiensequence_final>
PS D:\RIC_projects\protiensequence_final>
PS D:\RIC_projects\protiensequence_final> kubectl get svc
NAME                 TYPE           CLUSTER-IP      EXTERNAL-IP                                                               PORT(S)        AGE
kubernetes           ClusterIP      10.100.0.1      <none>                                                                    443/TCP        14m
proteomics-service   LoadBalancer   10.100.186.16   a2e82abd4d06b439bb30db6c6a418936-394370751.us-east-1.elb.amazonaws.com   80:30645/TCP   2m1s
PS D:\RIC_projects\protiensequence_final> kubectl describe svc oteomics-service
Error from server (NotFound): services "oteomics-service" not found
PS D:\RIC_projects\protiensequence_final> kubectl describe svc proteomics-service
Name:                     proteomics-service
Namespace:                default
Labels:                   <none>
Annotations:              <none>
Selector:                 app=proteomics
Type:                     LoadBalancer
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.100.186.16
IPs:                      10.100.186.16
LoadBalancer Ingress:     a2e82abd4d06b439bb30db6c6a418936-394370751.us-east-1.elb.amazonaws.com
Port:                     <unset>  80/TCP
TargetPort:               8501/TCP
NodePort:                 <unset>  30645/TCP
Endpoints:                192.168.0.92:8501
Session Affinity:         None
External Traffic Policy:  Cluster
Events:
```

13. Kubernetes setup



```
d85b356ec3b5: Layer already exists
latest: digest: sha256:daf2e22fcb1e17a907287cffbe8d6b035113184f897f632a7c9efa08440c79fa size: 2210
PS D:\RIC_projects\protiensequence_final> eksctl create cluster --name proteomics-cluster --region us-east-1 --managed
2023-12-12 14:49:02 [▪]  eksctl version 0.165.0
2023-12-12 14:49:02 [▪]  using region us-east-1
2023-12-12 14:49:02 [▪]  setting availability zones to [us-east-1b us-east-1c]
2023-12-12 14:49:02 [▪]  subnets for us-east-1b - public:192.168.0.0/19 private:192.168.64.0/19
2023-12-12 14:49:02 [▪]  subnets for us-east-1c - public:192.168.32.0/19 private:192.168.96.0/19
2023-12-12 14:49:02 [▪]  nodegroup "ng-da3678ea" will use "" [AmazonLinux2/1.27]
2023-12-12 14:49:02 [▪]  using Kubernetes version 1.27
2023-12-12 14:49:02 [▪]  creating EKS cluster "proteomics-cluster" in "us-east-1" region with managed nodes
2023-12-12 14:49:02 [▪]  will create 2 separate CloudFormation stacks for cluster itself and the initial managed nodegroup
2023-12-12 14:49:02 [▪]  if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=us-east-1 --cluster=proteomics-cluster'
2023-12-12 14:49:02 [▪]  Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false} for cluster "proteomics-cluster" in "us-east-1"
2023-12-12 14:49:02 [▪]  CloudWatch logging will not be enabled for cluster "proteomics-cluster" in "us-east-1"
2023-12-12 14:49:02 [▪]  you can enable it with 'eksctl utils update-cluster-logging --enable-types={SPECIFY-YOUR-LOG-TYPES-HERE (e.g. all)} --region=us-east-1 --cluster=proteomic
s-cluster'
2023-12-12 14:49:02 [▪]
2 sequential tasks: { create cluster control plane "proteomics-cluster",
    2 sequential sub-tasks: {
        wait for control plane to become ready,
        create managed nodegroup "ng-da3678ea",
    }
}
2023-12-12 14:49:02 [▪]  building cluster stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:49:03 [▪]  deploying stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:49:33 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:50:04 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:51:04 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:52:05 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:53:05 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:54:06 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:55:06 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:56:07 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:57:07 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:58:08 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 14:59:08 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 15:00:09 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-cluster"
2023-12-12 15:02:13 [▪]  building managed nodegroup stack "eksctl-proteomics-cluster-nodegroup-ng-da3678ea"
2023-12-12 15:02:14 [▪]  deploying stack "eksctl-proteomics-cluster-nodegroup-ng-da3678ea"
2023-12-12 15:02:14 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-nodegroup-ng-da3678ea"
2023-12-12 15:02:44 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-nodegroup-ng-da3678ea"
2023-12-12 15:03:26 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-nodegroup-ng-da3678ea"
2023-12-12 15:04:00 [▪]  waiting for CloudFormation stack "eksctl-proteomics-cluster-nodegroup-ng-da3678ea"
```

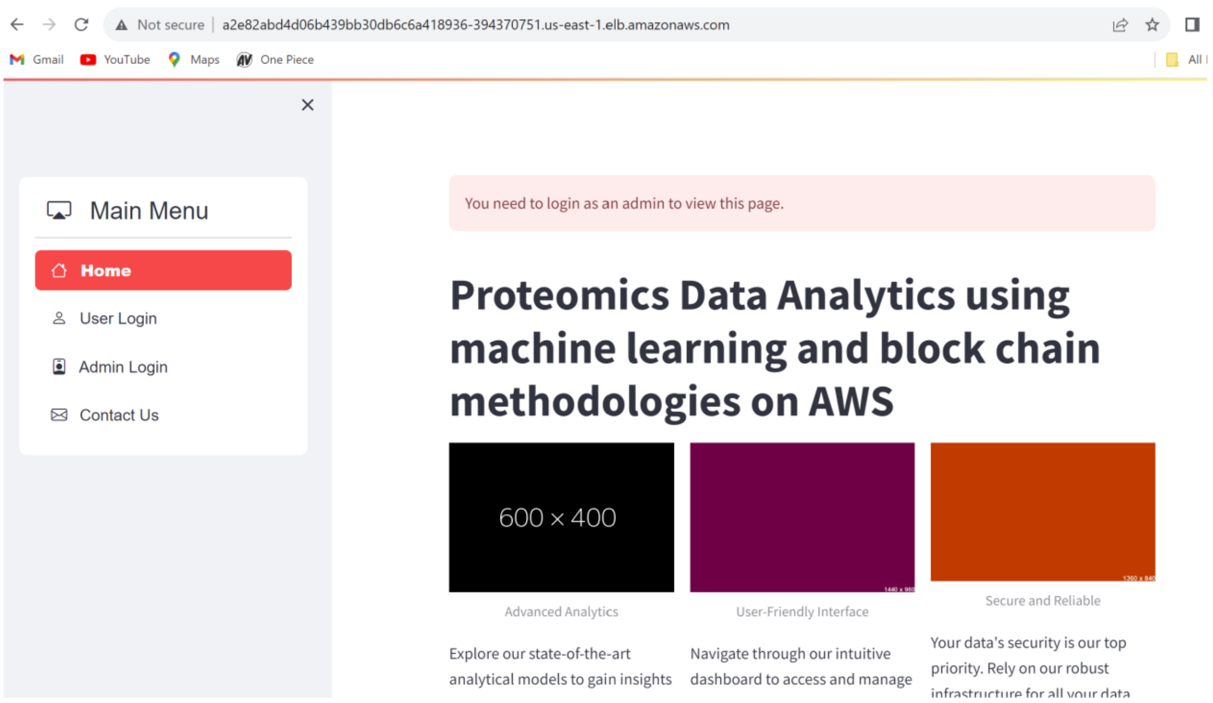
14. ECR repository creation



```
    + FullyQualifiedErrorId : CommandNotFoundException

PS D:\RIC_projects\protiensequence_final> aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 323444116722.dkr.ecr.us-east-1.amazonaws.com


An error occurred (UnrecognizedClientException) when calling the GetAuthorizationToken operation: The security token included in the request is invalid.
Error: Cannot perform an interactive login from a non TTY device
PS D:\RIC_projects\protiensequence_final> aws configure
AWS Access Key ID [****************PJYY]: AKIAUWTWFRTZK6UXMB6V
AWS Secret Access Key [****************5PUH]: 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
Default region name [us-east-1]:
Default output format [None]:
PS D:\RIC_projects\protiensequence_final> aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 323444116722.dkr.ecr.us-east-1.amazonaws.com

Error saving credentials: error storing credentials - err: exit status 1, out: `not implemented`
PS D:\RIC_projects\protiensequence_final> docker build -t testapp .
[+] Building 17.4s (10/10) FINISHED                                                                      docker:default
 => [internal] load .dockerignore                                                                                  0.0s
 => => transferring context: 2B                                                                                    0.0s
 => [internal] load build definition from Dockerfile                                                               0.0s
 => => transferring dockerfile: 961B                                                                               0.0s
 => [internal] load metadata for docker.io/library/python:3.10.11-slim-buster                                      2.1s
 => [1/5] FROM docker.io/library/python:3.10.11-slim-buster@sha256:9c5ad55e08d36d9cbacd8340127a84c4bb09cbd34a40a4  0.0s
 => [internal] load build context                                                                                  0.9s
 => => transferring context: 1.92MB                                                                                0.8s
 => CACHED [2/5] WORKDIR /app                                                                                      0.0s
 => CACHED [3/5] COPY requirements.txt .                                                                           0.0s
 => CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt                                                0.0s
 => [5/5] COPY . .                                                                                                 8.1s
 => exporting to image                                                                                             6.2s
 => => exporting layers                                                                                            6.2s
 => => writing image sha256:509d2883f45c39a3e12569d24d270b0db4e4cae84b7bb63d1a942b837cb36ecb                       0.0s
 => => naming to docker.io/library/testapp                                                                         0.0s

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
PS D:\RIC_projects\protiensequence_final> docker tag testapp:latest 323444116722.dkr.ecr.us-east-1.amazonaws.com/testapp:latest
PS D:\RIC_projects\protiensequence_final> docker push 323444116722.dkr.ecr.us-east-1.amazonaws.com/testapp:latest
The push refers to repository [323444116722.dkr.ecr.us-east-1.amazonaws.com/testapp]
027f436df88e: Pushed
54a2b6448f88: Pushed
18d16d73d580: Pushed
461f34f72d28: Pushed
adb58d7d925f: Pushed
```

15. After this all is done we have to create EC2 instance in AWS.

16. Open the deployed link. UI will be like below.



You need to login as an admin to view this page.

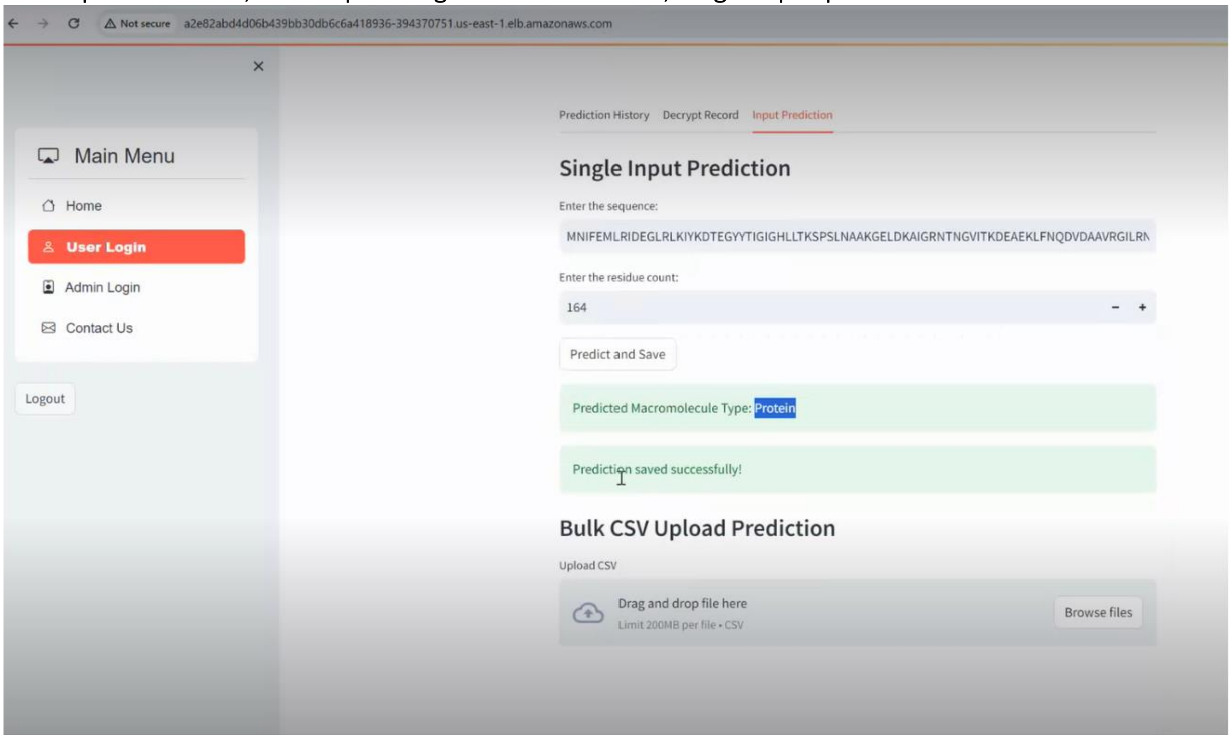
Proteomics Data Analytics using machine learning and block chain methodologies on AWS

Explore our state-of-the-art analytical models to gain insights

Navigate through our intuitive dashboard to access and manage

Your data's security is our top priority. Rely on our robust infrastructure for all your data

Now upload CSV file, After Uploading the bulk CSV data, Single input prediction is done.



Single Input Prediction

Enter the sequence:

MNIFEMLRIDEGLRLKIYKDTEGYYTIGIGHLLTKSPSLNAAKGELDKAIGRNTNGVITKDEAEKLFNQDVDAAVRGILRN

Enter the residue count:

164

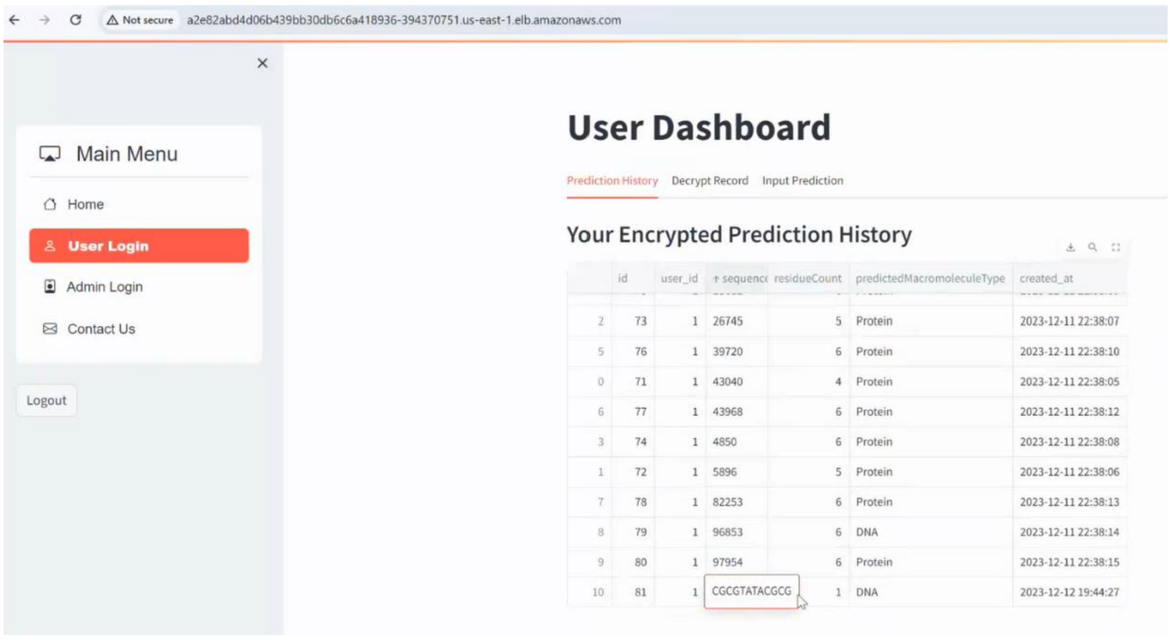
Predicted Macromolecule Type: Protein

Prediction saved successfully!

Bulk CSV Upload Prediction

Upload CSV

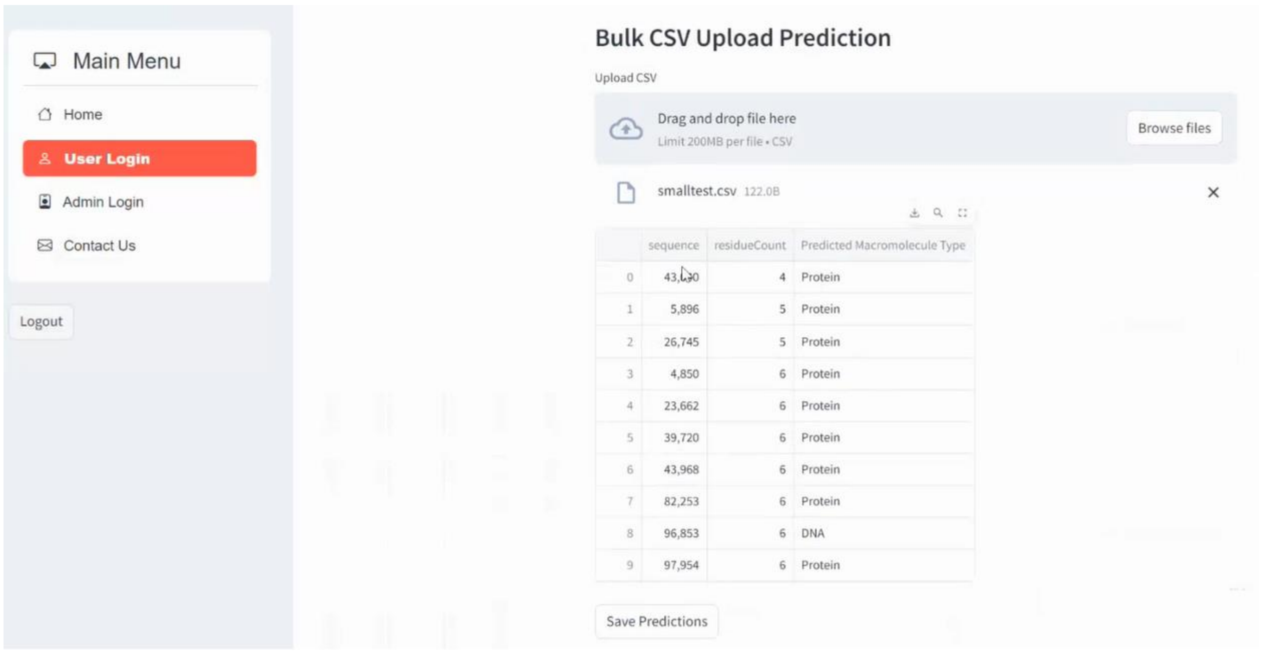
The predicted sequence will be stored in prediction history tab



Your Encrypted Prediction History

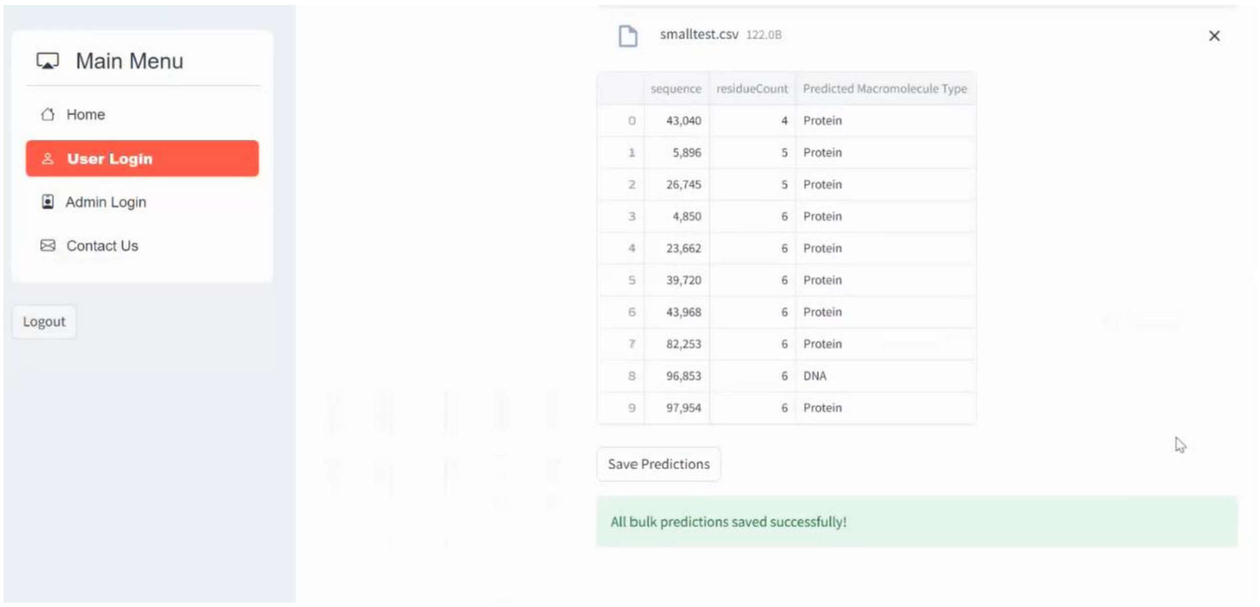
	id	user_id	sequence	residueCount	predictedMacromoleculeType	created_at
2	73	1	26745	5	Protein	2023-12-11 22:38:07
5	76	1	39720	6	Protein	2023-12-11 22:38:10
0	71	1	43040	4	Protein	2023-12-11 22:38:05
6	77	1	43968	6	Protein	2023-12-11 22:38:12
3	74	1	4850	6	Protein	2023-12-11 22:38:08
1	72	1	5896	5	Protein	2023-12-11 22:38:06
7	78	1	82253	6	Protein	2023-12-11 22:38:13
8	79	1	96853	6	DNA	2023-12-11 22:38:14
9	80	1	97954	6	Protein	2023-12-11 22:38:15
10	81	1	CGCGTATACGCG	1	DNA	2023-12-12 19:44:27

Now we will upload a small CSV file for testing, after uploading below is the view



	sequence	residueCount	Predicted Macromolecule Type
0	43,040	4	Protein
1	5,896	5	Protein
2	26,745	5	Protein
3	4,850	6	Protein
4	23,662	6	Protein
5	39,720	6	Protein
6	43,968	6	Protein
7	82,253	6	Protein
8	96,853	6	DNA
9	97,954	6	Protein

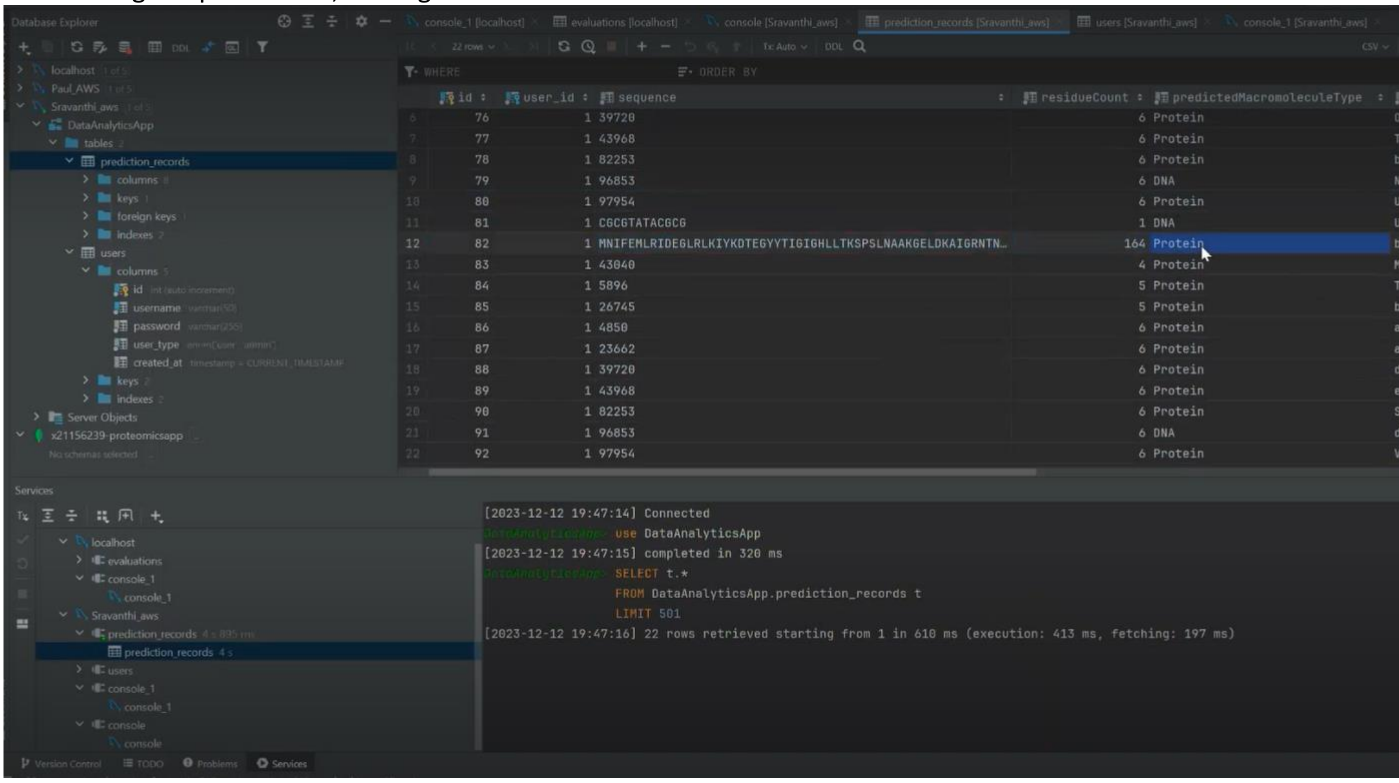
After saving upload save prediction



	sequence	residueCount	Predicted Macromolecule Type
0	43,040	4	Protein
1	5,896	5	Protein
2	26,745	5	Protein
3	4,850	6	Protein
4	23,662	6	Protein
5	39,720	6	Protein
6	43,968	6	Protein
7	82,253	6	Protein
8	96,853	6	DNA
9	97,954	6	Protein

All bulk predictions saved successfully!

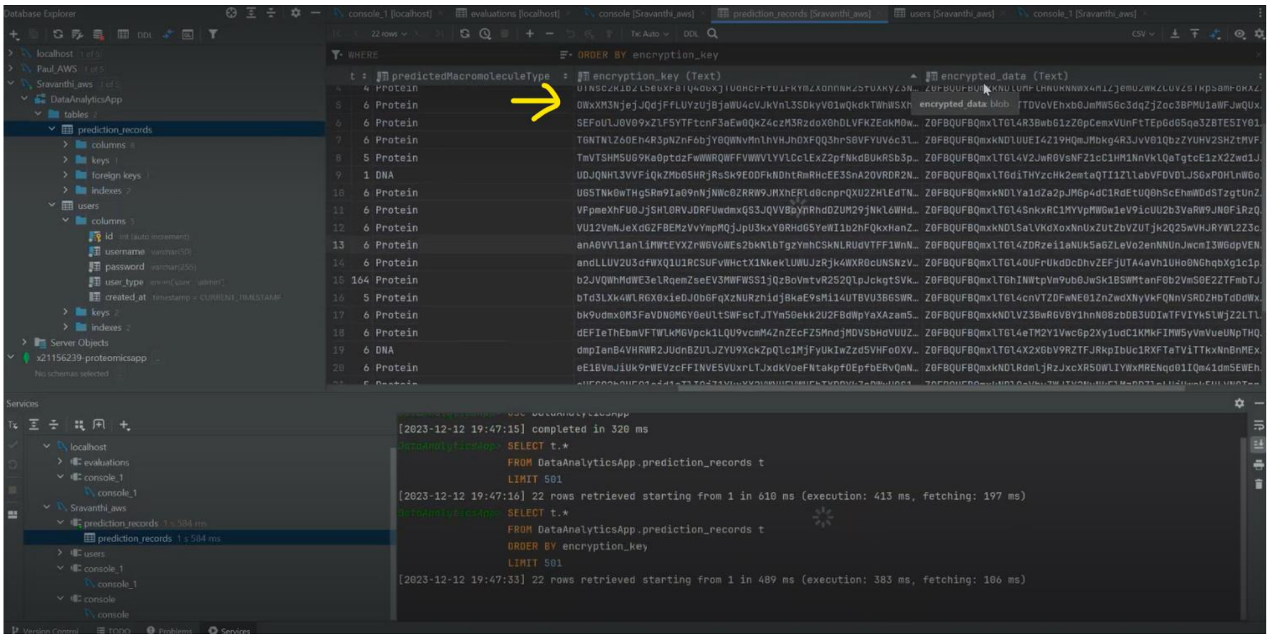
After saving the predictions, it will get stored in database



	id	user_id	sequence	residueCount	predictedMacromoleculeType
6	76	1	39720	6	Protein
7	77	1	43968	6	Protein
8	78	1	82253	6	Protein
9	79	1	96853	6	DNA
10	80	1	97954	6	Protein
11	81	1	CGCGTATACGCG	1	DNA
12	82	1	MNIFEMLRIDEGLRLKIYKDTEGYYTIGIGHLLTKSPSLNAAKGELDKAIGRNTN…	164	Protein
13	83	1	43040	4	Protein
14	84	1	5896	5	Protein
15	85	1	26745	5	Protein
16	86	1	4850	6	Protein
17	87	1	23662	6	Protein
18	88	1	39720	6	Protein
19	89	1	43968	6	Protein
20	90	1	82253	6	Protein
21	91	1	96853	6	DNA
22	92	1	97954	6	Protein

```
[2023-12-12 19:47:14] Connected
DataAnalyticsApp> use DataAnalyticsApp
[2023-12-12 19:47:15] completed in 320 ms
DataAnalyticsApp> SELECT t.*
                  FROM DataAnalyticsApp.prediction_records t
                  LIMIT 501
[2023-12-12 19:47:16] 22 rows retrieved starting from 1 in 610 ms (execution: 413 ms, fetching: 197 ms)
```

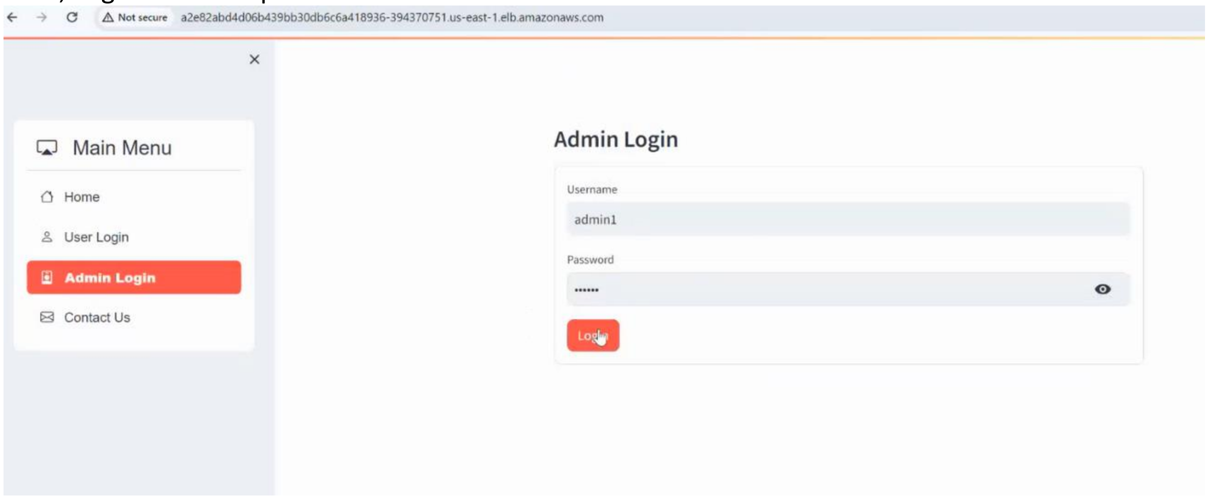
While the prediction is getting saved in database, also the data is getting encrypted due to PBFT blockchain algorithm.
Only Admin will be having access to the encrypted key, Only when User request admin for the key, then only it can accessed by user.



```
WHERE                                  ORDER BY encryption_key
 t  predictedMacromoleculeType   encryption_key (Text)                                  encrypted_data (Text)
 5  6 Protein   OWxXM3NjejJQdjFfLUYzUjBjaWU4cVJkVnl3SDkyV01wQkdkTWhWSXh…   encrypted_data: blob  TTDVoVEhxbOJmMW5Gc3dqZjZoc3BPMU1aWFJwQUx.
 6  6 Protein   SEFoUlJ0V09xZlF5YTFtcnF3aEw0QkZ4czM3RzdoX0hDLVFKZEdkM0w…   Z0FBQUFBQmxlTGl4R3BwbG1zZ0pCemxVUnFtTEpGdG5qa3ZBTE5IY01.
 7  6 Protein   TGNTNlZ6OEh4R3pNZnF6bjY0QWNvMnlhVHJhOXFQQ3hrS0VFYUV6c3l…   Z0FBQUFBQmxkNDlUUEI4Z19HQmJMbkg4R3JvV01QbzZYUHV2SHZtMVF.
 8  5 Protein   TmVTSHM5UG9Ka0ptdzFwWWRQWFFVWWVlYVlCclExZ2pfNkdBUkRSb3p…   Z0FBQUFBQmxlTGl4V2JwR0VsNFZ1cC1HM1NnVklQaTgtcE1zX2Zwd1J.
 9  1 DNA       UDJQNHl3VVFiQkZMb05HRjRsSk9EODFkNDhtRmRHcEE3SnA2OVRDR2N…   Z0FBQUFBQmxlTGdiTHYzcHk2emtaQTI1ZllabVFDVDlJSGxPOHlnWGo.
10  6 Protein   U65TNk0wTHg5Rm9Ia09nNjNWc0ZRRW9JMXhERld0cnprQXU2ZHlEdTN…   Z0FBQUFBQmxkNDlYa1dZa2pJMGp4dC1RdEtUQ0hScEhmWDdSTzgtUnZ.
11  6 Protein   VFpmeXhFU0JjSHl0RVJDRFUwdmxQS3JQVVBoYnRhdDZUM29jNkl6WHd…   Z0FBQUFBQmxlTGl4SnkxRC1MYVpMWGw1eV9icUU2b3VaRW9JN0FiRzQ.
12  6 Protein   VU12VmNJeXdGZFBEMzVvYmpMQjJpU3kxY0RHdG5YeWI1b2hFQkxHanZ…   Z0FBQUFBQmxkNDlSalVKdXoxNnUxZUtZbVZUTjk2Q25wVHJRYWl2Z3c.
13  6 Protein   anA0VVl1anliMWtEYXZrWGV6WEs2bkNlbTgzYmhCSkNLRUdVTFF1WnN…   Z0FBQUFBQmxlTGl4ZDRzei1aNUk5aGZLeVo2enNNUnJwcmI3WGdpVEN.
14  6 Protein   andLLUV2U3dfWXQ1U1RCSUFvWHctX1NkeklUWUJzRjk4WXR0cUNSNzV…   Z0FBQUFBQmxlTGl4OUFrUkdDcDhvZEFjUTA4aVh1UHo0NGhqbXg1c1p.
15  164 Protein b2JVQWhMdWE3elRqemZseEV3MWFWSS1jQzBoVmtvR252QlpJckgtSVk…   Z0FBQUFBQmxlTGhINWtpVm9ub0JwSk1BSWMtanF0b2VmS0E2ZTFmbTJ.
16  5 Protein   bTd3LXk4WlRGX0xieDJObGFqXzNURzhidjBkaE9sMi14UTBVU3BGSWR…   Z0FBQUFBQmxlTGl4cnVTZDFwNE01ZnZwdXNyVkFQNnVSRDZHbTdDdWx.
17  6 Protein   bk9udmx0M3FaVDN0MGY0eUltSWFscTJTYm50ekk2U2FBdWpYaXAzam5…   Z0FBQUFBQmxkNDlVZ3BwRGVBY1hnN08zbDB3UDIwTFVIYk5lWjZ2LTl.
18  6 Protein   dEFIeThEbmVFTWlkMGVpck1LQU9vcmM4ZnZEcFZ5MndjMDVSbHdVUUZ…   Z0FBQUFBQmxlTGl4eTM2Y1VwcGp2Xy1udC1KMkFIMW5yVmVueUNpTHQ.
19  6 DNA       dmpIanB4VHRWR2JUdnBZUlJZYU9XckZpQlc1MjFyUkIwZzd5VHFoOXV…   Z0FBQUFBQmxlTGl4X2xGbV9RZTFJRkpIbUc1RXFTaTViTTkxNnBnMEx.
20  6 Protein   eE1BVmJiUk9rWEVzcFFINVE5VUxrLTJxdkVoeFNtakpfOEpfbERvQmN…   Z0FBQUFBQmxkNDlRdmljRzJxcXR5OWlIYWxMRENqd01IQm41dm5EWEh.
```

```
[2023-12-12 19:47:15] completed in 320 ms
DataAnalyticsApp> SELECT t.*
                  FROM DataAnalyticsApp.prediction_records t
                  LIMIT 501
[2023-12-12 19:47:16] 22 rows retrieved starting from 1 in 610 ms (execution: 413 ms, fetching: 197 ms)
DataAnalyticsApp> SELECT t.*
                  FROM DataAnalyticsApp.prediction_records t
                  ORDER BY encryption_key
                  LIMIT 501
[2023-12-12 19:47:33] 22 rows retrieved starting from 1 in 489 ms (execution: 383 ms, fetching: 106 ms)
```

Now, Login to Admin panel.


Not secure
a2e82abd4d06b439bb30db6c6a418936-394370751.us-east-1.elb.amazonaws.com
Main Menu
Home
User Login
Admin Login
Contact Us
Login

Admin Login

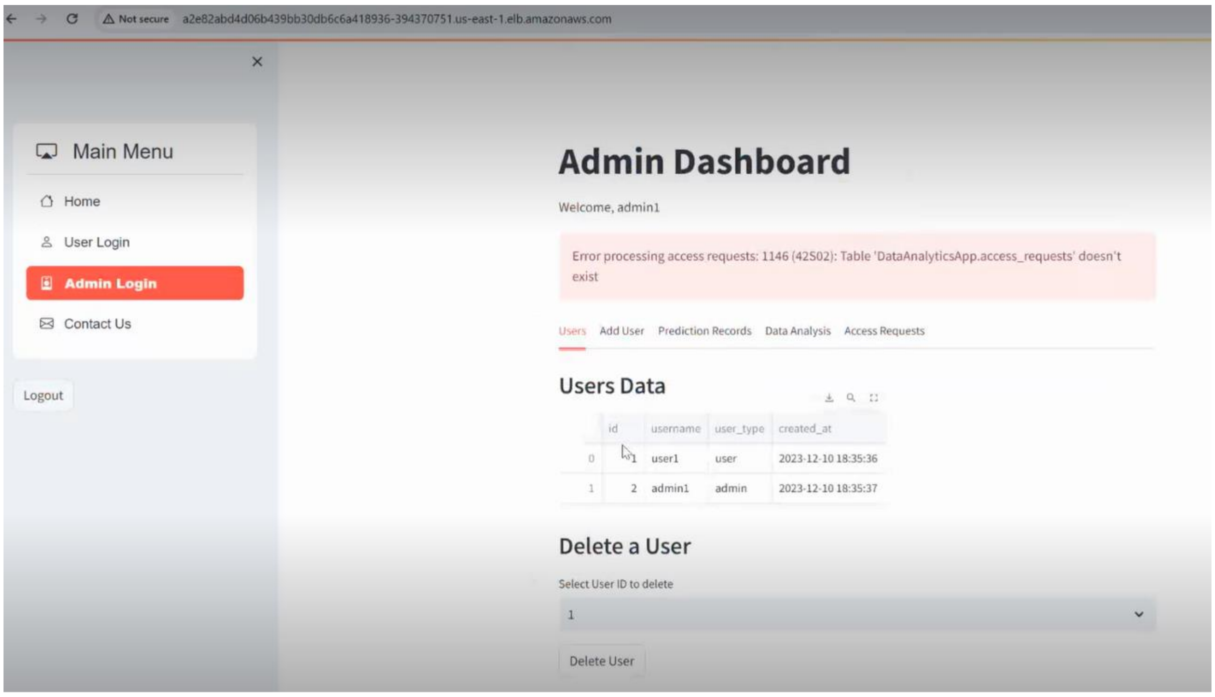
Username

admin1

Password

••••••

Here, we can create and delete user



Not secure a2e82abd4d06b439bb30db6c6a418936-394370751.us-east-1.elb.amazonaws.com
Main Menu
Home
User Login
Admin Login
Contact Us
Logout

Admin Dashboard

Welcome, admin1

Error processing access requests: 1146 (42S02): Table 'DataAnalyticsApp.access_requests' doesn't exist

Users | Add User | Prediction Records | Data Analysis | Access Requests

Users Data

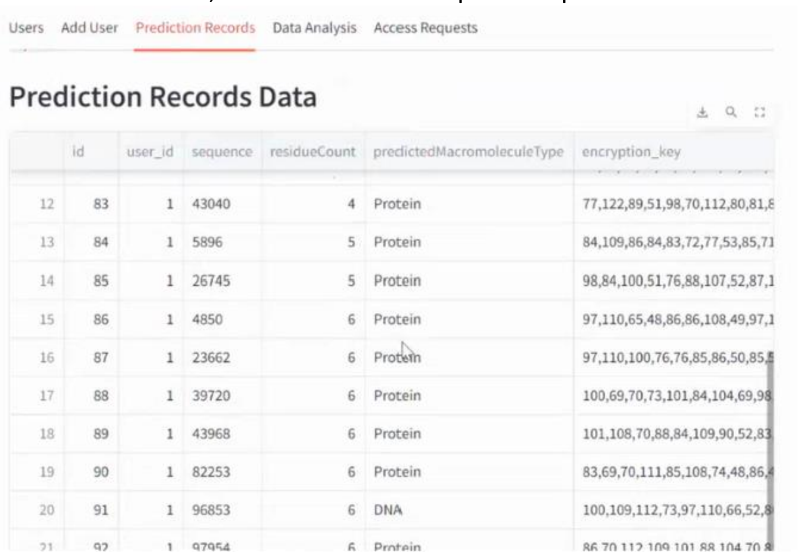
	id	username	user_type	created_at
0	1	user1	user	2023-12-10 18:35:36
1	2	admin1	admin	2023-12-10 18:35:37

Delete a User

Select User ID to delete: 1

Delete User

On Prediction tab, we can see all the uploaded prediction data.

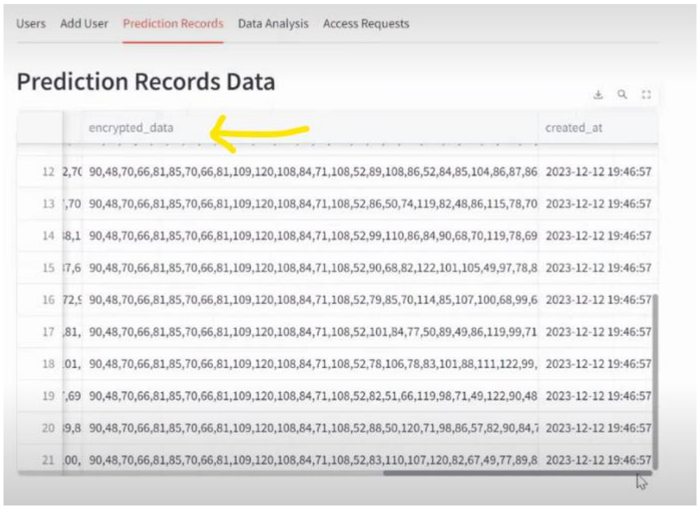


Users | Add User | Prediction Records | Data Analysis | Access Requests

Prediction Records Data

	id	user_id	sequence	residueCount	predictedMacromoleculeType	encryption_key
12	83	1	43040	4	Protein	77,122,89,51,98,70,112,80,81,
13	84	1	5896	5	Protein	84,109,86,84,83,72,77,53,85,71
14	85	1	26745	5	Protein	98,84,100,51,76,88,107,52,87,1
15	86	1	4850	6	Protein	97,110,65,48,86,86,108,49,97,1
16	87	1	23662	6	Protein	97,110,100,76,76,85,86,50,85,
17	88	1	39720	6	Protein	100,69,70,73,101,84,104,69,98
18	89	1	43968	6	Protein	101,108,70,88,84,109,90,52,83
19	90	1	82253	6	Protein	83,69,70,111,85,108,74,48,86,
20	91	1	96853	6	DNA	100,109,112,73,97,110,66,52,
21	92	1	97954	6	Protein	86,70,112,109,101,88,104,70,

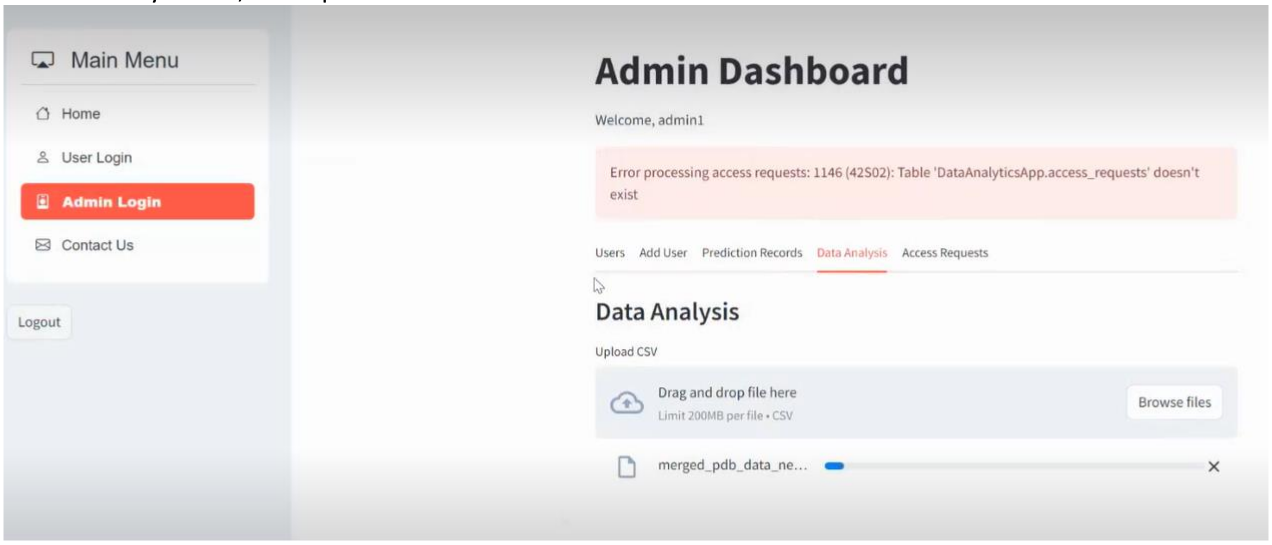
Even in UI Admin can see the Encryption key. Move the table to right to view encrypted key



Prediction Records Data

		encrypted_data	created_at
12	2,7(	90,48,70,66,81,85,70,66,81,109,120,108,84,71,108,52,89,108,86,52,84,85,104,86,87,86	2023-12-12 19:46:57
13	,70	90,48,70,66,81,85,70,66,81,109,120,108,84,71,108,52,86,50,74,119,82,48,86,115,78,70	2023-12-12 19:46:57
14	8,1	90,48,70,66,81,85,70,66,81,109,120,108,84,71,108,52,99,110,86,84,90,68,70,119,78,69	2023-12-12 19:46:57
15	7,6	90,48,70,66,81,85,70,66,81,109,120,108,84,71,108,52,90,68,82,122,101,105,49,97,78,8	2023-12-12 19:46:57
16	72,9	90,48,70,66,81,85,70,66,81,109,120,108,84,71,108,52,79,85,70,114,85,107,100,68,99,6	2023-12-12 19:46:57
17	,81,	90,48,70,66,81,85,70,66,81,109,120,108,84,71,108,52,101,84,77,50,89,49,86,119,99,71	2023-12-12 19:46:57
18	01,	90,48,70,66,81,85,70,66,81,109,120,108,84,71,108,52,78,106,78,83,101,88,111,122,99,	2023-12-12 19:46:57
19	,69	90,48,70,66,81,85,70,66,81,109,120,108,84,71,108,52,82,51,66,119,98,71,49,122,90,48	2023-12-12 19:46:57
20	9,8	90,48,70,66,81,85,70,66,81,109,120,108,84,71,108,52,88,50,120,71,98,86,57,82,90,84,7	2023-12-12 19:46:57
21	00,	90,48,70,66,81,85,70,66,81,109,120,108,84,71,108,52,83,110,107,120,82,67,49,77,89,8	2023-12-12 19:46:57

In Data Analysis tab, first upload the CSV file.



Admin Dashboard

Welcome, admin1

Error processing access requests: 1146 (42S02): Table 'DataAnalyticsApp.access_requests' doesn't exist

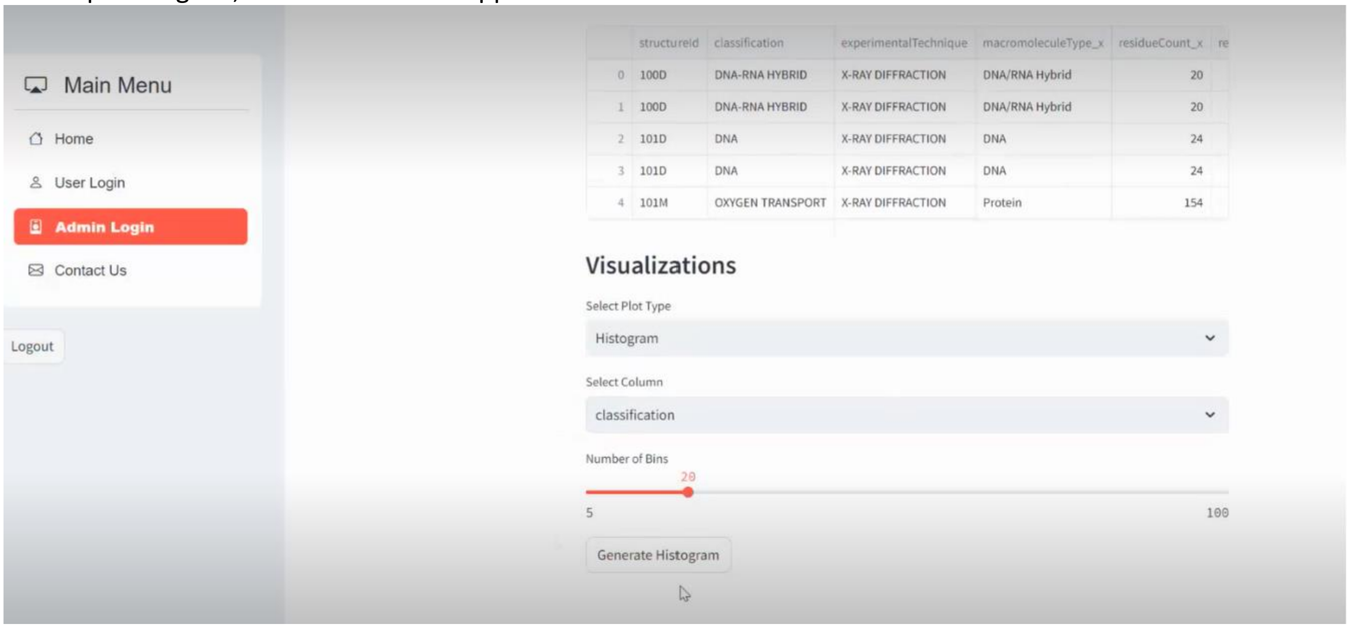
Data Analysis

Upload CSV

Drag and drop file here
Limit 200MB per file • CSV

merged_pdb_data_ne...

After uploading file, Visualizations will appear



	structureId	classification	experimentalTechnique	macromoleculeType_x	residueCount_x	re
0	100D	DNA-RNA HYBRID	X-RAY DIFFRACTION	DNA/RNA Hybrid	20	
1	100D	DNA-RNA HYBRID	X-RAY DIFFRACTION	DNA/RNA Hybrid	20	
2	101D	DNA	X-RAY DIFFRACTION	DNA	24	
3	101D	DNA	X-RAY DIFFRACTION	DNA	24	
4	101M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	154	

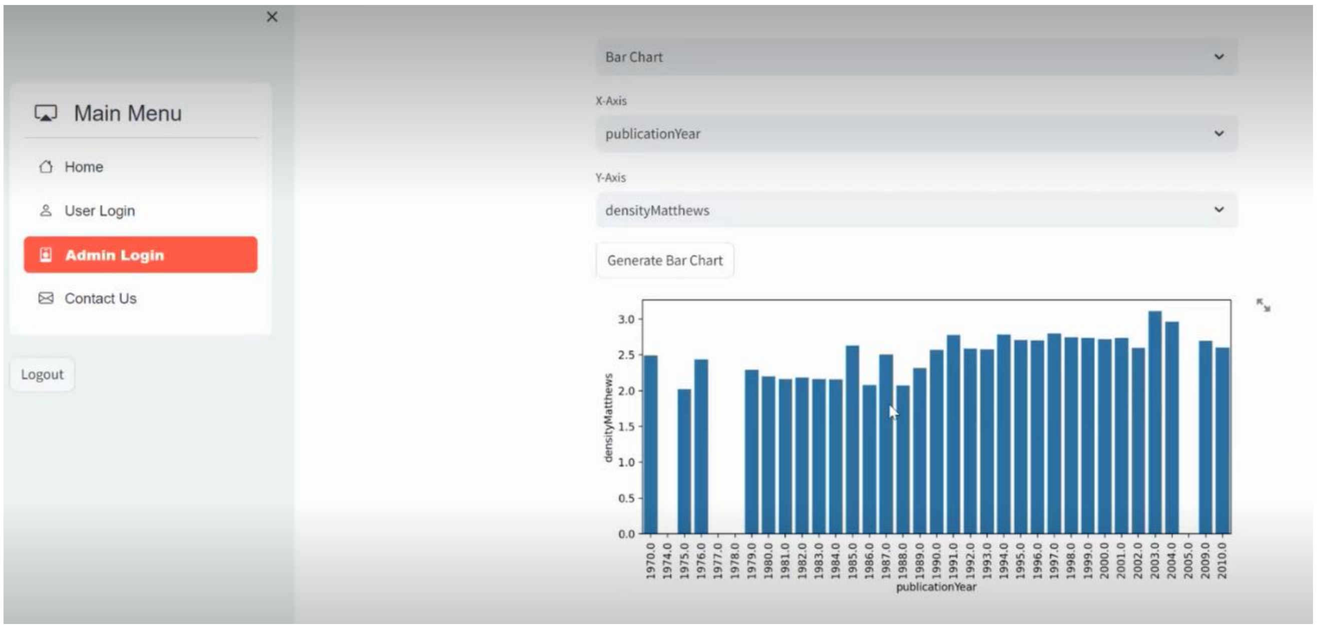
Visualizations

Select Plot Type: Histogram

Select Column: classification

Number of Bins: 20 (5–100)

Bar Chart



Main Menu
Home
User Login
Admin Login
Contact Us
Logout
Bar Chart
X-Axis
publicationYear
Y-Axis
densityMatthews
Generate Bar Chart
densityMatthews
publicationYear