**Prospect Lead Predictor**

We are planning to implement a model that predicts prospect converted to Leased or Unqualified. this will help the marketing people and as well to the property owners to see in depth what features he opted, for example if the price range between so and so are mostly predicted as active lease etc, ...

**Features:**

Functional:

1) Lead form, where consumer provide his basic information as we currently have in OneSite.

2) Build complete workflow for creating Model, API to predict based on created model

3) Notification for Site leasing agent.

4) Dashboard for verifying lease status with their key feature

Non-functional:

1) Dockerizing the python application.

2) Continuous Integration using GitHub actions.

3) MLFlow to track the ML model results and artifacts.

## TechStack:

Backend: - Below are the framework / libraries we have used to build this application

[Python]: For general purpose coding

Python Flask: For building API

[MySQL]: Database

Kafka: for sending messages.

MLFlow: Managing the end-to-end machine learning lifecycle.

Docker: to dockize application for easy installations.

UI:

Angular: UI/UX actions / events on HTML documents, DOM transformation and accessing resources.

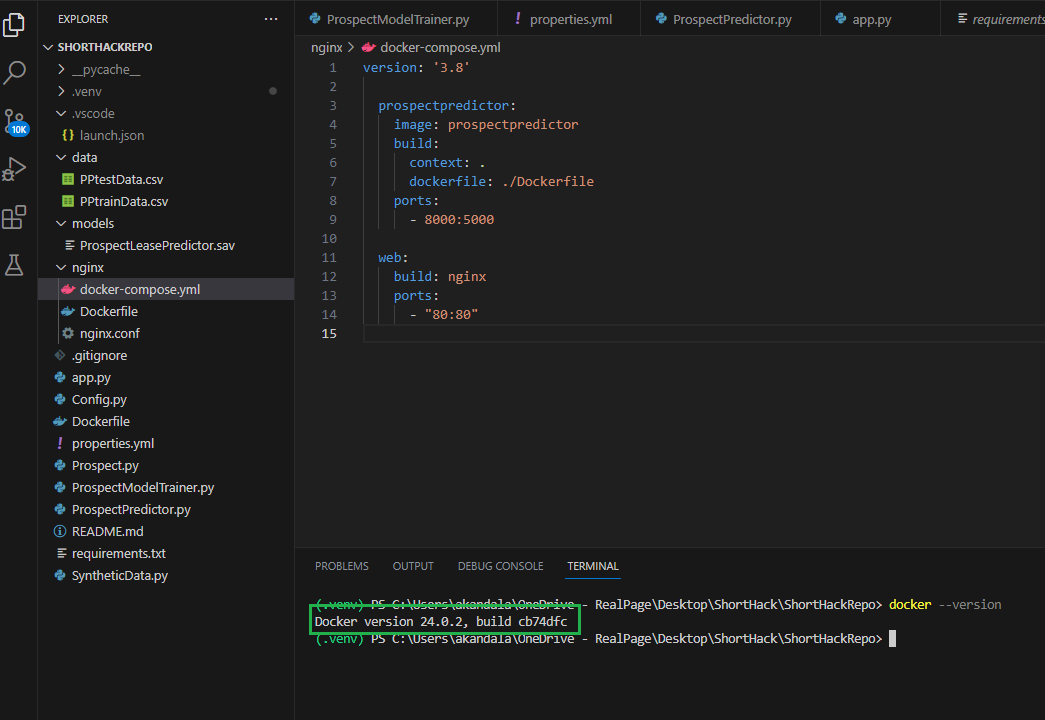
Bootstrap: UI experience look and feel.

Tool:

1. Visual Studio Code.
2. Jupyter notebook
3. MS SQL management studio
4. Docker desktop
5. Git Bash

## Installations:

1. Make sure docker is installed and docker engine is running on your system:



1. On the terminal create a image “docker build -t prospectpredictor .”

A screenshot of a computer

Description automatically generated

1. Now run the container “docker-compose up -d”

A screenshot of a computer program

Description automatically generated with medium confidence

1. Verify logs if the web server is running or not ”docker-compose logs”.

A screenshot of a computer program

Description automatically generated with medium confidence

1. Check if app is running using HealthCheck API - <http://localhost/healthcheck>

A screenshot of a computer

Description automatically generated with medium confidence

1. Curl info below:

curl --location --request POST 'http://localhost/Predict' \

--header 'Content-Type: application/json' \

--data-raw '{

"BrokenLease": 0,

"County": "UA",

"CriminalQuestion": 0,

"DOB": "01/01/2002",

"Evicted": 0,

"Gender": "M",

"IsInternationalApplicant": 0,

"MaritalStatus": "M",

"PrefCommunicationType": "",

"ResidentDesignation": "",

"SuedForDamage": 0,

"SuedForRent": 0,

"city": "Dadfllas",

"countryCode": "UA",

"gcardId": 37,

"gcardOccupantCount": 2,

"gcardPreferedFloorplanGroupId": "2Bed",

"jobTypeID": "10",

"phoneType1": "M",

"phoneType2": "J",

"relCode": "H",

"rentOwnFlag": "N",

"state": "Test",

"status": 1,

"trfsrcName": "internet",

"zip": "654321"

}

Response: {

     "Predict": 1

}

## Workflow model:

Consumer

Leasing agent

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UI** | **API** | **ML Model** | **Database** | **Kafka** |
|  | Kafka send message to consumer topic.  Resource created response.  Consumer call API | DB to persist data.  Model prediction | Kafka send message to producer topic. |  |