INTERNSHIP AT OCEANMTECH PVT LTD

AN INTERNSHIP REPORT

Submitted by

JAYKUMAR DHANSUKHBHAI PATEL

190200107098

In fulfillment of the award of the degree of

BACHELOR OF ENGINEERING

In

COMPUTER ENGINEERING

GOVERNMENT ENGINEERING COLLEGE, RAJKOT

GUJARAT TECHNOLOGICAL UNIVERSITY,

AHMEDABAD

[May 2023]

GOVERNMENT ENGINEERING COLLEGE, RAJKOT

Mavdi - Kankot Road, Near Hanuman Mandir

Rajkot, Gujarat, 360005

CERTIFICATE

This is to certify that the Internship report submitted along with the project entitled **Python Developer Intern** has been satisfactorily carried out by **Jay Patel** (190200107098) under my guidance in partial fulfillment for the degree of Bachelor of Engineering in **Computer Engineering**, 8th Semester of Gujarat Technological University, Ahmedabad during the academic year 2022-23.

Prof. Shyam Kotecha

Internal Guide

Dr. S.M.Shah

Head of the Department

OCEANMTECH

OCEANMTECH
To Grow Your Business

Date: 4th May 2023

CERTIFICATE OF TRAINING

This is to certify that **Mr Jaykumar Dhansukhbhai Patel**, a Student of Bachelor of Engineering at Government Engineering College Rajkot has successfully completed 3 months of training at **OCEANMTECH**. He worked as **Python Developer** from **08-02-2023 to 04-05-2023**.

The Project entitled **JETFLIX Movies Recommendation System** embodies the work done by **Mr Jaykumar Dhansukhbhai Patel** Under the guidance of **Mr Milan Bharadia** during the above mentioned project training period.

This is an original work and this work has been submitted anywhere in any form.

We found him sincere, hardworking, technically sound and result oriented. We take this opportunity to thank him and wish him all the best for his future.

Sign. OCEANMTECH PRIVATE LIMITED DIRECTOR

Place: Ahmedabad.

Date:

Oceanmtech Private Limited +91 9978686900 www.oceanmtech.com

308,309 Gopal Palace B-Wing, near Hotel MAAN, Nehru Nagar, Ahmedabad, Gujarat 380015

GOVERNMENT ENGINEERING COLLEGE, RAJKOT

Mavdi - Kankot Road, Near Hanuman Mandir

Rajkot, Gujarat, 360005

DECLARATION

We hereby declare that the Internship report submitted along with the Internship entitled **Python Developer Intern** submitted in partial fulfillment for the degree of Bachelor of Engineering in **Computer Engineering** to Gujarat Technological University, Ahmedabad, is a bonafide record of original project work carried out by me at under the supervision of Mr Milan Bharadia and that no part of this report has been directly copied from any students' reports or taken from any other source, without providing due reference.

Name of the Student	Sign of Student
Jay Patel	

ACKNOWLEDGEMENT

I would like to extend my heartly thanks with a deep sense of gratitude and respect to all those who provides me with immense help and guidance during my internship.

I would like to express our sincere thanks to my internal guide **Prof. Shyam Kotecha** who allowed me to undertake such challenging and great innovative work. We are grateful to them for their guidance, encouragement, understanding, and insightful support in the development process.

I would like to extend my heartiest thanks to **Mr. Milan Bharadiya** (Team Leader) at Oceanmtech Private Limited for supporting me during the internship period. He guided us all the time and motivated us within his busy schedule.

Last but not least we would like to mention here that we are greatly indebted to each and everybody who has been associated with my internship at any stage but whose name does not find a place in this acknowledgment.

Thank You

ABSTRACT

Internships are generally thought to be reserved for college students looking to gain experience in a particular field. However, a wide array of people can benefit from Training Internships to receive real world experience and develop their skills.

An objective for this position should emphasize the skills you already possess in the area and your interest in learning more Internships are utilized in several different career fields, including architecture, engineering, healthcare, economics, advertising, and many more.

Some internship is used to allow individuals to perform scientific research while others are specifically designed to allow people to gain first-hand experience working.

Utilizing internships is a great way to build your resume and develop skills that can be emphasized in your resume for future jobs. When you are applying for a Training Internship, make sure to highlight any special skills or talents that can make you stand apart from the rest of the applicants so that you have an improved chance of landing the position.

List of Figures

Table of Contents

Details of Chapters

1.0 Overview of the Company
- 1.1 History

2.0 Internship Management and Tools:
- 2.1 Internship Summary
  - 2.1.1 Purpose
  - 2.1.2 Objective
  - 2.1.3 Scope
  - 2.1.4 Technology
  - 2.1.5 Internship Planning
  - 2.1.6 Tools

3.0 Python
- 3.1 Basic Python
  - 3.1.1 Introduction
  - 3.1.2 Variables & Data Types
  - 3.1.3 If & Else
  - 3.1.4 For & While Loops
  - 3.1.5 Class & Functions
  - 3.1.6 Try & Except

4.0 Frontend
- 4.1 HTML
- 4.2 CSS
- 4.3 Basic Java Scripts
- 4.4 Bootstrap

CHAPTER 1

1. OVERVIEW OF THE COMPANY

- Oceanmtech was started in 2010 by a young entrepreneur Mr. Manoj Hirapara. Registered as a Private Limited Organization in 2016. Beginning of Providing IT Base Solutions. At the Oceanmtech pvt ltv started Best Digital Solution as DMT Application. Finally, the DMT App Launch in 2022

Figure 1.0 Oceanmtech Product

- Currently in India Small and Medium Scale Industries want to expand their Business on Social Media platforms but due to a Lack of Awareness of Digital and Social Media Marketing, Tools and Technology Awareness they are not able to expand.
- We have provided our app and website to help them.

Oceanmtech's Vision

- **The vision** is to Serve Digital Awareness and Digital Products for SMEs and working Professionals by providing them low cost-effective digital tools to expand their business.

CHAPTER 2

2. INTERNSHIP MANAGEMENT AND TOOLS

2.1 INTERNSHIP SUMMARY

⇨ An internship is a period of temporary work experience that allows individuals to gain practical skills and knowledge in a particular field or industry, often undertaken by students or recent graduates to gain valuable work experience and enhance their resume. It can also serve as a networking opportunity and a way to explore potential career paths.

2.1.1 Purpose

⇨ It is an excellent learning curve for me while meeting new people and making connections in the professional world. In today's job market, passing exams with high scores and getting a degree doesn't offer the much-needed work experience, you will need to succeed in a workspace.

⇨ In an internship, you will be able to gain real-life exposure, grow your knowledge, and determine if you are in the right career field.

2.1.2 Objective

⇨ The main objective of this is to learn and get Knowledge of Web Development.

2.1.3 Scope

⇨ Using this Knowledge make complex web apps.

2.1.4 Technology

⇨ Programming Languages
- Python
- Java Scripts
- HTML/CSS

⇨ Frameworks
- Django
- Flask

⇨ IDE
- VS Code
- Jupyter Notebooks

⇨ Tools
- Pandas
- Numpy
- Selenium
- Beautiful Soup
- Matplotlib

2.1.5 Internship Plannings

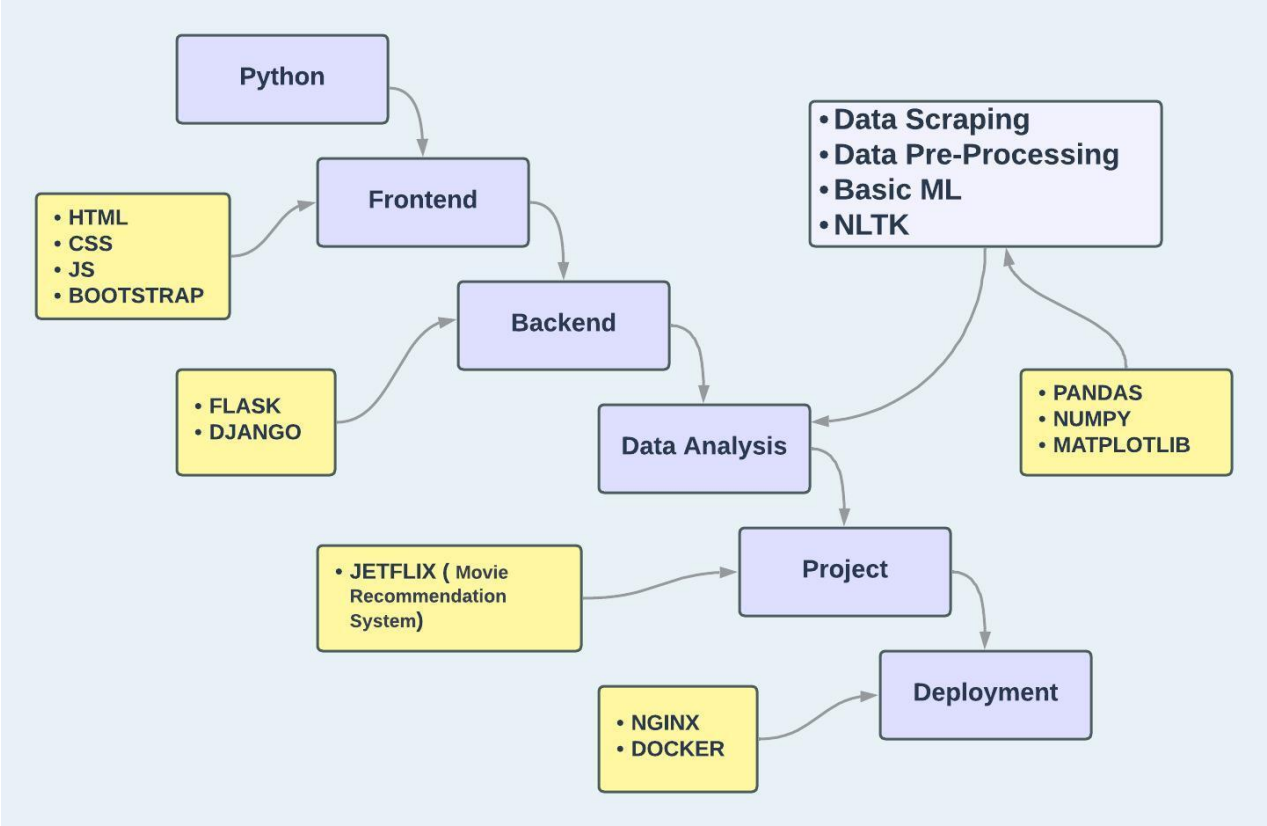


Figure 2.1.5 Learning Model

2.1.6 System Hardware

- ⇨ Hardware:
  - System: INTEL i5 2.4 GHz
  - SSD: 512 GB
  - RAM: 8GB

2.1.7 IDE

- ⇨ IDE stands for Integrated Development Environment. This software application provides comprehensive tools for software developers to write, test, and debug their code. IDEs typically include a code editor, a debugger, and a compiler or interpreter, as well as tools for managing source code, building and deploying applications and collaborating with other developers.

VS CODE

- ⇨ VS Code stands for Visual Studio Code. It is a free, open-source code editor developed by Microsoft. VS Code is a lightweight yet powerful editor that is popular among developers for its ease of use and versatility. It provides a rich set of features, such as syntax highlighting, code completion, debugging tools, and an extensive marketplace of extensions and plugins that can be used to customize and extend its functionality.
- ⇨ VS Code supports a wide range of programming languages, including C++, Java, Python, JavaScript, TypeScript, and many others. It also supports multiple operating systems, including Windows, macOS, and Linux.
- ⇨ One of the key features of VS Code is its ability to integrate with other tools and services commonly used in software development, such as Git and GitHub, Docker, and various cloud platforms. This makes it a popular choice for both individual developers and teams working on complex projects.
- ⇨ Overall, VS Code is a powerful, flexible, and customizable code editor that is well-suited for a wide range of programming tasks and workflows.

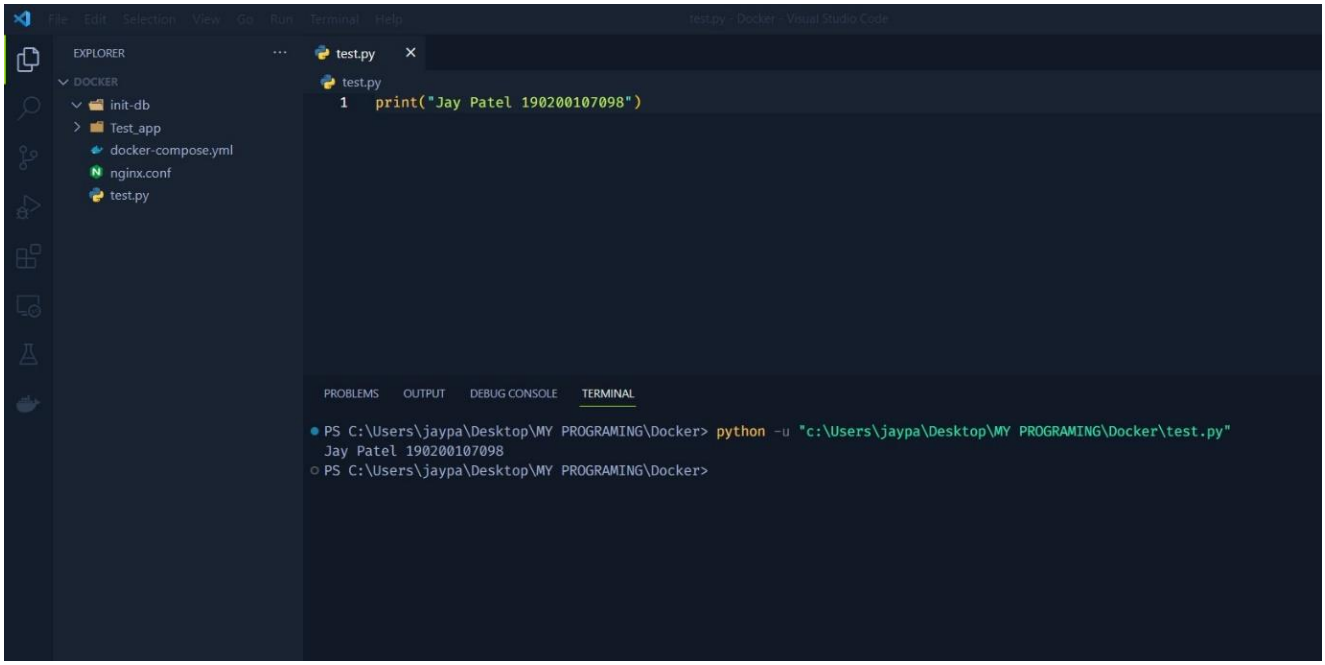


Figure 2.2.1 VS Code

JUPYTER NOTEBOOK

- Jupyter Notebook is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and narrative text. Jupyter Notebook supports over 40 programming languages, including Python, R, Julia, and MATLAB.
- The Jupyter Notebook interface consists of a web-based interactive notebook, which enables users to write and execute code in a web browser. The notebook is divided into cells containing code, text, or multimedia elements such as images or videos. Users can interact with the notebook by typing code into the cells, executing the code, and viewing the results within the notebook. They can also include visualizations and other forms of multimedia content to create rich, interactive documents.
- Jupyter Notebook is a powerful tool for data analysis, scientific computing, and research. Its interactive, web-based interface and support for multiple programming languages make it a popular choice among data scientists, researchers, and educators.

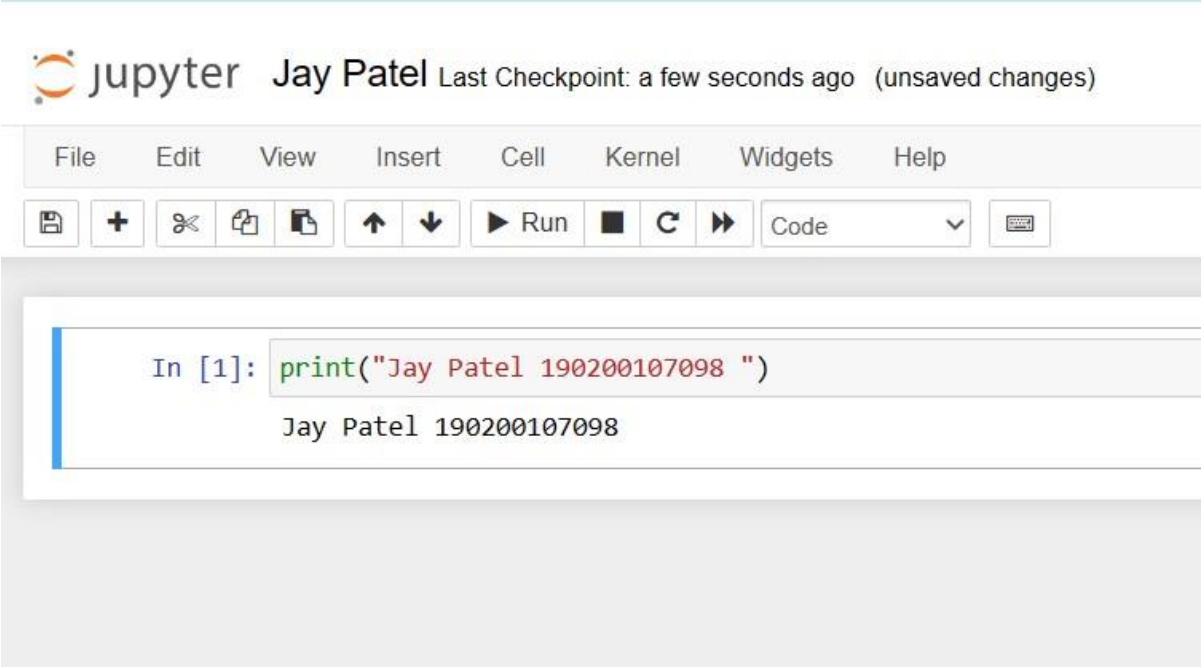


Figure 2.2.2 Jupyter Notebook

CHAPTER 3

3. PYTHON

3.1 BASIC PYTHON

3.1.1 INTRODUCTION

- ⇨ Python is a high-level, interpreted programming language that was first released in 1991 by Guido van Rossum. It is widely used for general-purpose programming, data science, artificial intelligence, web development, scientific computing, and more.
- ⇨ Python has a simple and easy-to-learn syntax, which makes it an ideal language for beginners. It emphasizes code readability and maintainability, with a design philosophy that emphasizes code clarity and simplicity.
- ⇨ Python has a vast ecosystem of libraries and tools, including the popular NumPy, Pandas, Matplotlib, and TensorFlow, making it a popular choice for data science and machine learning.
- ⇨ Overall, Python's versatility, ease of use, and large community make it a popular choice for a wide range of programming tasks.

3.1.2 VARIABLES AND DATA TYPES

- ⇨ **Variables:**
- ⇨ In Python, a variable is a named reference to a value that is stored in the computer's memory. The value can be of any type, such as an integer, floating-point number, string, boolean, or more complex data structures like lists or dictionaries.
- ⇨ Variables can be used in expressions and statements to manipulate the values they refer to.

```
Name_1 = "JAY PATEL "
roll_n_1 = 20
Mobile_num_1 = 123456

Name_2 = "SALMAN KHAN "
roll_n_1 = 21
Mobile_num_1 = 654321
```

Figure 3.1.2 Python Variables

- ⇨ Python, variables are created when you assign a value to a name using the assignment operator "="

⇨ Data Type:

Numeric Types: These are used to represent numbers in Python. There are three numeric types: integers, floating-point numbers, and complex numbers.

In [9]:
```
# Numeric types
x = 10
y = 3.14
z = 5 + 2j
print(f"x = {type(x)},y = {type(y)},z ={(type(z))}")
```

```
x = <class 'int'>,y = <class 'float'>,z =<class 'complex'>
```

Sequence Types: These are used to represent sequences of values. The most commonly used sequence types in Python are lists, tuples, and ranges.

In [10]:
```
# Sequence types
my_list = [1, 2, 3, 4]
my_tuple = ("apple", "banana", "cherry")
my_range = range(0, 10)
print(f"my_list = {type(my_list)},my_tuple = {type(my_tuple)},my_range ={(type(my_range))}")
```

```
my_list = <class 'list'>,my_tuple = <class 'tuple'>,my_range =<class 'range'>
```

Text Type: This is used to represent text in Python. The text type in Python is called a string.

In [12]:
```
# Text type
my_string = "Hello, world!"
print(f"my_string = {type(my_string)}")
```

```
my_string = <class 'str'>
```

Boolean Type: This is used to represent boolean values, which can be either True or False.

In [4]:
```
# Boolean type
my_bool = True
print(f"my_bool = {type(my_bool)}")
```

Set Types: These are used to represent unordered collections of unique elements. There are two types of set in Python: set and frozenset.

In [15]:
```
# Set types
my_set = {1, 2, 3, 4, 5}
print(f"my_set = {type(my_set)}")
```

```
my_set = <class 'set'>
```

Mapping Types: These are used to represent mappings between keys and values. The most commonly used mapping type in Python is the dictionary.

In [16]:
```
# Mapping type
my_dict = {"name": "John", "age": 25, "city": "New York"}
print(f"my_dict = {type(my_dict)}")
```

```
my_dict = <class 'dict'>
```

Binary Types: These are used to represent binary data. The two most commonly used binary types in Python are bytes and bytearray.

In [19]:
```
# Binary types
my_bytes = b"hello"
my_bytearray = bytearray(b"world")
print(f"my_bytes = {type(my_bytes)},my_bytearray = {type(my_bytearray)}")
```

```
my_bytes = <class 'bytes'>,my_bytearray = <class 'bytearray'>
```

In [20]:
```
print("@jay patel 190200107098")
```

```
@jay patel 190200107098
```

Figure 3.1.2 Python Data type

3.1.3 IF AND ELSE

```
In [2]: x = int(input())

if x > 5:
    print("x is greater than 5")
else:
    print("x is less than or equal to 5")

5
x is less than or equal to 5
```

Figure 3.1.3 Python If Else

3.1.4 FOR AND WHILE LOOPS

For Loop

```
In [ ]:
my_list = [1, 2, 3, 4, 5]
for item in my_list:
    print(item)


my_string = "hello"
for char in my_string:
    print(char)


for i in range(5):
    print(i)
```

While Loop

```
In [ ]:
i = 0
while i < 5:
    print(i)
    i += 1


while True:
    num = input("Enter a number: ")
    if num ==  5 ():
        print(" numer 5 malse tya sudhi faryu ")
        break
    else:
        print(" sacho number nathi try again ")
```

Figure 3.1.4 Python Loops

3.1.5 CLASS AND FUNCTIONS

```
In [5]: class MyClass:
            def say_hello(self):
                print("Hello, world!")

        # creating an instance of the class
        my_object = MyClass()

        # calling the function on the object
        my_object.say_hello()

Hello, world!
```

Figure 3.1.5 Python Class & Functions

3.1.6 TRY AND EXCEPT

⇨ In Python, try and except are statements used for error handling. When you write code, there's always a possibility that something unexpected might happen, such as an invalid input or a file that cannot be found. If your code encounters an error like this, it will typically stop running and print out an error message, which can be difficult for users to understand.

In Input Is True

```
In [8]: try:
            x = int(input())
            print(x)
        except:
            print(" Can Not Print This values ")
```

```
5
5
```

In Input Is False

```
In [9]: try:
            x = int(input())
            print(x)
        except:
            print(" Can Not Print This values ")
```

```
jay
 Can Not Print This values
```

Figure 3.1.6 Python Try and Except

CHAPTER 4

4. FRONTEND

- In web development, the front end refers to the part of a website that users interact with directly. This includes the design, layout, and functionality of the user interface. Front-end developers are responsible for creating the code that runs in a user's web browser and makes the website look and feel the way it does.
- The front end typically includes HTML, CSS, and JavaScript code that is executed in the user's browser. HTML provides the structure and content of the web page, CSS defines the presentation and layout, and JavaScript provides interactivity and dynamic behavior.
- In short, the front end is the visible part of a website that users see and interact with, and front-end developers are responsible for creating and maintaining the code that makes it all work.

4.1 HTML

- HTML (Hypertext Markup Language) is the standard markup language used to create web pages. It provides a way to organize and structure text, images, and other media on a web page using tags and attributes. HTML defines the content and layout of a web page.

4.2 CSS

- CSS (Cascading Style Sheets) is a style sheet language used for describing the presentation of a document written in HTML. CSS provides a way to define the appearance of text, images, and other elements on a web page, such as font size, color, and layout. It allows web designers to create visually appealing and consistent designs across multiple pages and devices.

4.3 BASIC JAVA SCRIPTS

- JavaScript is a programming language used to create interactive and dynamic web content. With JavaScript, web developers can add user interactivity and behavior to their web pages, such as form validation, animations, and user interface components. JavaScript can interact with HTML and CSS to dynamically change the content and appearance of a web page based on user actions or other events.

4.4 BOOTSTRAP5

- Bootstrap is a popular front-end framework for building responsive, mobile-first web projects. It is a collection of HTML, CSS, and JavaScript components that can be used to create user interfaces and layouts for web pages and web applications.

- By using Bootstrap, front-end developers can save time and effort by leveraging pre-built components and styles and can ensure that their projects are mobile-friendly and accessible. Additionally, Bootstrap offers extensive documentation and support, making it easy to learn and use even for developers who are new to front-end development.

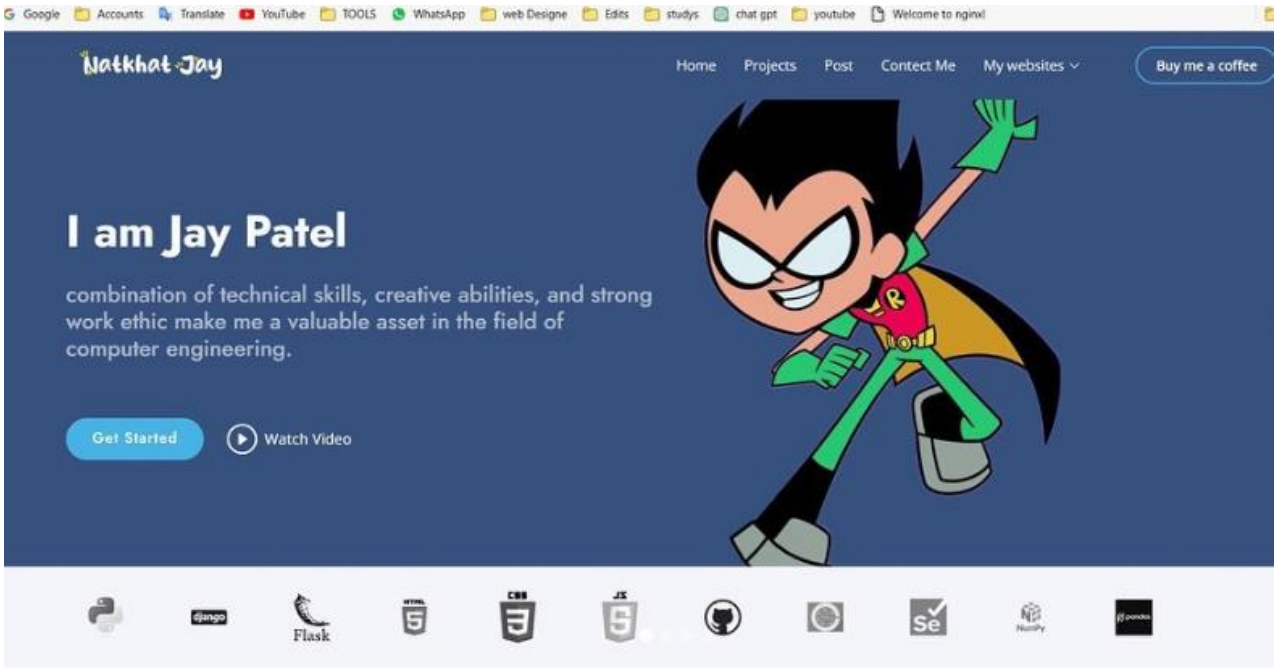


Figure 4.4 HTML CSS JS Bootstrap Page

CHAPTER 5

5. BACKEND

⇨ Backend refers to the part of a web application that runs on the server-side and is responsible for managing the application's data and functionality. It is the part of the application that is not visible to the user and consists of several components working together to provide the application's functionality. Here's a brief explanation of some of the key components of a backend:

5.1 SERVER:

- The server is a computer program that receives requests from clients (web browsers or mobile apps) and returns responses. The server runs on a physical or virtual machine, and it may use different software and hardware components depending on the application's requirements.

5.2 APPLICATION

- Application: The application is a set of programs that run on the server to provide the application's functionality. It is usually written in a programming language such as Python, Java, Ruby, or PHP. The application handles requests from clients, retrieves data from the database, and generates responses.

5.3 API

- API: An API (Application Programming Interface) is a set of protocols, routines, and tools that allows different software applications to communicate with each other. In a web application, the API is used to enable communication between the frontend and the backend. It defines the endpoints, methods, and data formats used to exchange information between the two.

5.4 DATABASE

- Database: A database is a software program that stores and manages data. In a web application, the database is used to store and retrieve data related to the application's functionality. The data may include user profiles, transactions, product catalogs, or any other data relevant to the application.

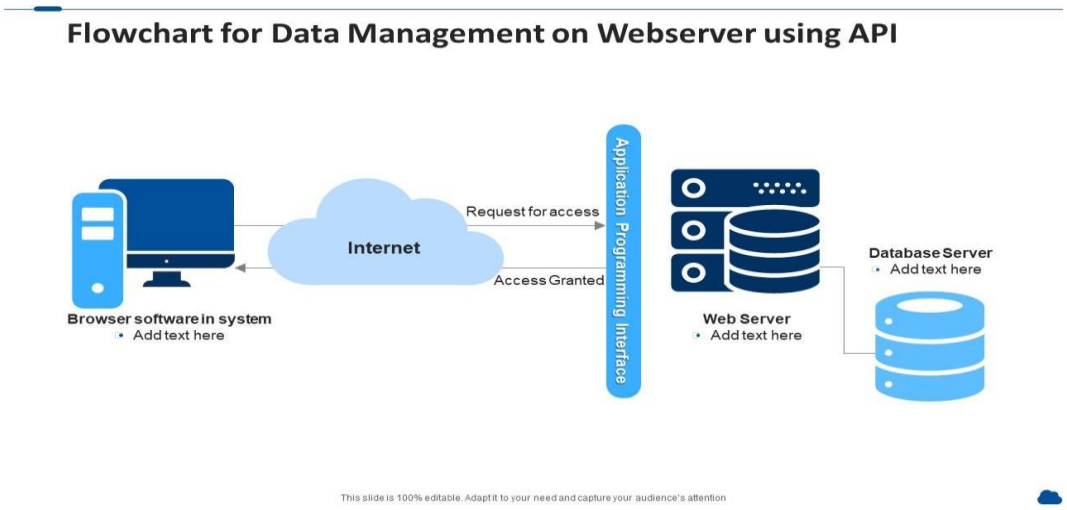


Figure 5.4 data management on Backend

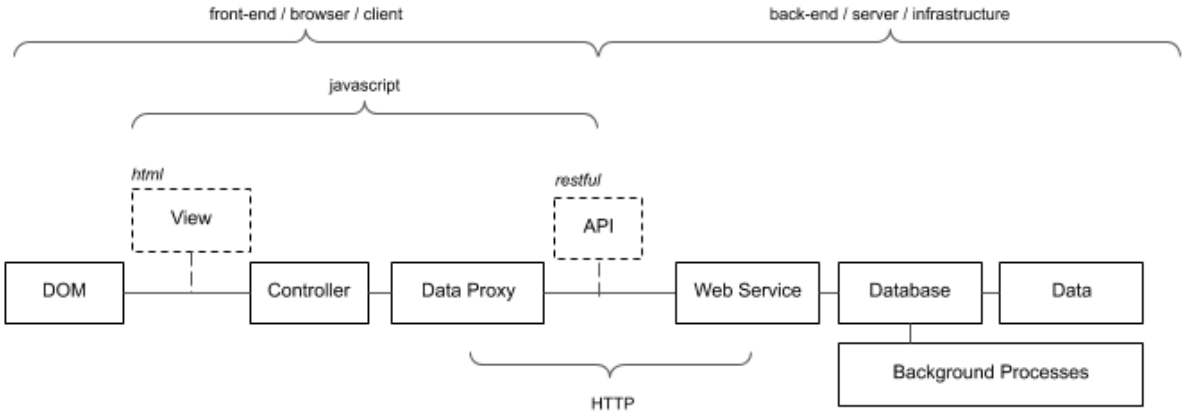


Figure 5.4 Frontend and Backend Communications Flow

⇨ In summary, the backend is the server side of a web application that manages the data and functionality of the application. It consists of different components, including the server, application, API, and database, working together to provide the application's functionality.

CHAPTER 6

6 PYTHON FRAMEWORKS

6.3DJANGO

6.3.1 INTRODUCTION

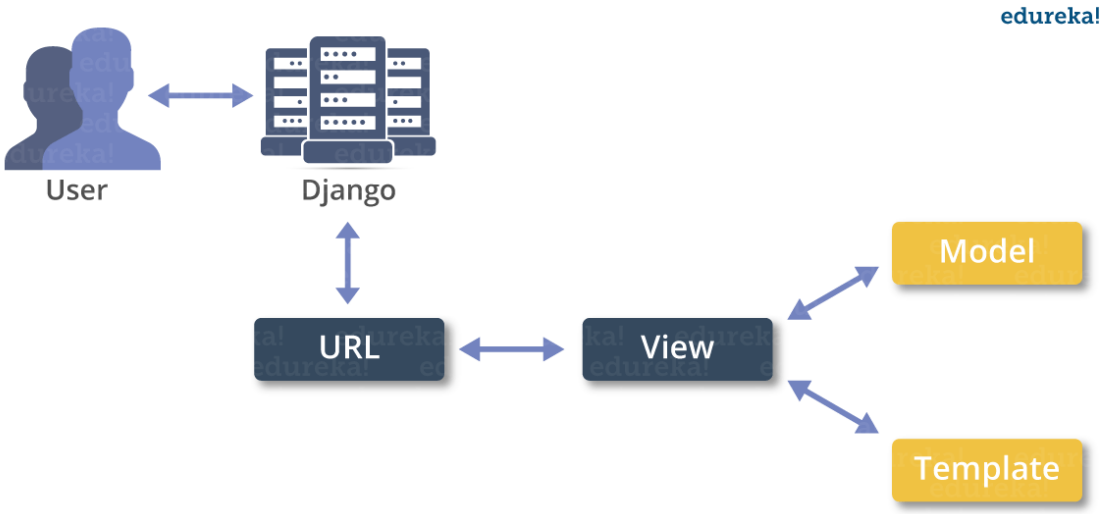


Figure 6.3.1 Django MVT Flow

⇨ Django is an open-source web application framework written in Python. The primary goal of Django is to make the development of complex, data-based websites easier. Thus, Django emphasizes the reusability and pluggability of components to ensure rapid developments. Django consists of three major parts: model, view, and template.

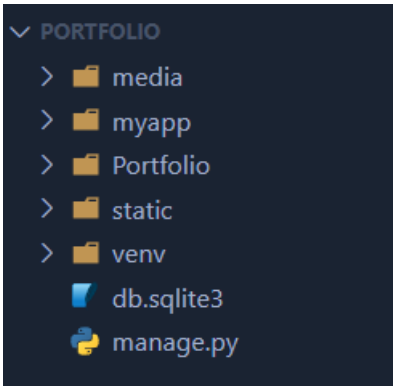


Figure 6.3.1 Django File Structure

6.3.2 MODEL

- A model is a single, definitive data source that contains the essential field and behavior of the data. Usually, one model is one table in the database. Each attribute in the model represents a field of a table in the database. Django provides a set of automatically-generated database application programming interfaces (APIs) for the convenience of users.

Figure 6.3.2 Django File Model.py

6.3.3 VIEWS

- Views is short form of the view file. It is a file containing a Python function which takes web requests and returns web responses. A response can be HTML content or XML documents or a “404 error” and so on. The logic inside the view function can be arbitrary as long as it returns the desired response. To link the view function with a particular URL we need to use a structure called URL conf which maps URLs to view functions.
- It essentially bridges the gap between a URL and the underlying business logic that processes the request and generates the response.
- Views can interact with models and templates to render dynamic content, and can also handle forms, perform database queries, and authenticate users.

models.py views.py

myapp > views.py > ...

```python
from django.shortcuts import render
from .models import *
# Create your views here.


def homepage(request):
    projects= Project.objects.all().order_by('?')[:3]
    context = {'projects':projects}
    return render(request,'base/home.html',context)


def projectpage(request):
    projects= Project.objects.all()
    language=Language.objects.all()
    categoty=Categoty.objects.all()
    id = request.GET.get('id')
    if id:
        projects=Project.objects.filter(categoty=id)
    else:
        projects = Project.objects.all()

    context = {'projects':projects,'language':language,'categoty':categoty}
    return render(request,'base/project.html',context)

def postpage(request):
    posts=Post.objects.all()
    context = {'posts':posts}
    return render(request,'base/postpage.html',context)

def aboutme(request):
    context = {}
    return render(request,'base/aboutme.html',context)
```

Figure 6.3.3 Django File View.py

6.3.4 TEMPLATES

⇨ Django's template is a simple text file that can generate a text-based format like HTML and XML. The template contains variables and tags. Variables will be replaced by the result when the template is evaluated. Tags control the logic of the template. We also can modify the variables by using filters. For example, a lowercase filter can convert the variable from uppercase into lowercase.

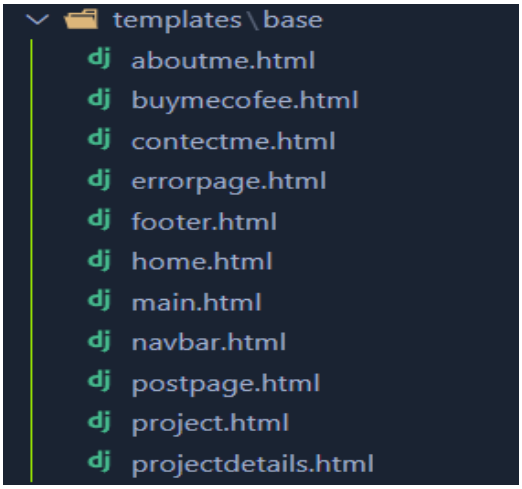


Figure 6.3.4 Django Templates File

6.4 FLASK

6.4.1 INTRODUCTION

- Flask is a lightweight web framework for Python that allows developers to quickly build web applications. It is a micro-framework, which means it provides only the essentials to create a web application and leaves the rest up to the developer. Flask is easy to learn and use, making it a popular choice for building small to medium-sized web applications.
- Flask provides several features that make it useful for web development, including:

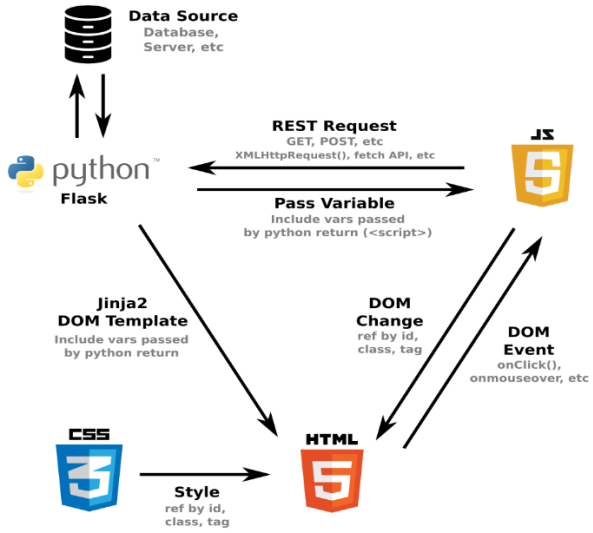


Figure 6.4.1 Flask Flow

6.4.2 ROUTING

- Flask provides a simple and intuitive way to map URLs to views, making it easy to create routes for different parts of the application.

6.4.3 VIEWS

- Views are short form of view file. It is a file containing a Python function that takes web requests and returns web responses. A response can be HTML content or XML documents or a “404 error” and so on. The logic inside the view function can be arbitrary as long as it returns the desired response.

6.4.4 TEMPLATES

- Flask uses the Jinja2 template engine, which allows developers to create dynamic HTML pages with ease. Jinja2 provides a powerful and flexible syntax for working with templates.

6.4.5 DATABASE(MongoDB)

- PyMongo can be easily integrated with Flask, a popular Python web framework, to build web applications that use MongoDB as their data store.
- PyMongo is a Python library for working with MongoDB, a popular NoSQL database. PyMongo allows Python developers to interact with MongoDB databases using a simple and intuitive API.
- With PyMongo, you can perform various database operations such as inserting, updating, deleting, and querying data in a MongoDB database using Python code. PyMongo also supports advanced features such as aggregation pipelines, grids, and indexing.
- In addition, PyMongo provides support for various MongoDB features such as shading, replication, and authentication, making it a powerful tool for building scalable and secure applications that rely on MongoDB as their data store.

Import Moduls

```
In [*]: from flask import Flask
        from pymongo import MongoClient

        app = Flask(__name__)
        client = MongoClient('mongodb://localhost:27017/')
```

Routing And Views

```
In [*]: @app.route('/')
        def home():
            db = client['mydatabase']
            collection = db['mycollection']
            return 'Welcome to my Flask app!'
```

```
In [*]: @app.route('/add')
        def add():
            db = client['mydatabase']
            collection = db['mycollection']
            document = {'name': 'Jay Patel', 'En No': 190200107098}
            collection.insert_one(document)
            return 'Data added successfully!'
```

```
In [*]: @app.route('/read')
        def read():
            db = client['mydatabase']
            collection = db['mycollection']
            documents = collection.find()
            result = ''
            for document in documents:
                result += f"Name: {document['name']}, En No: {document['En No']}<br>"
            return result
```

Run Flask

```
In [*]:
        if __name__ == '__main__':
            app.run()
```

Figure 6.4.5 Simple Flask Website Code

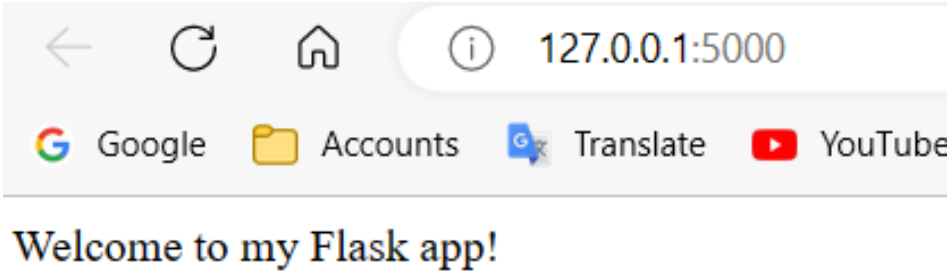


Figure 6.4.5 Flask Website Home Page

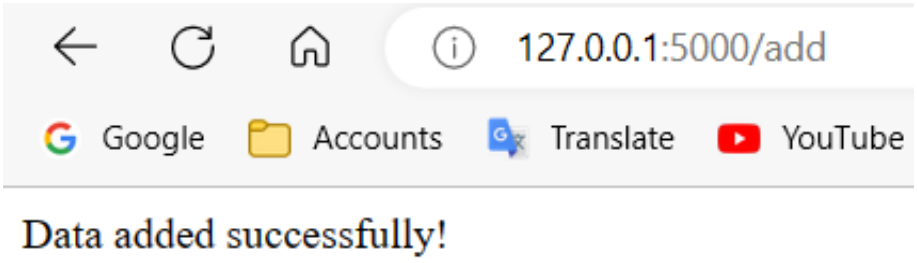


Figure 6.4.5 Flask Website Add Data Page

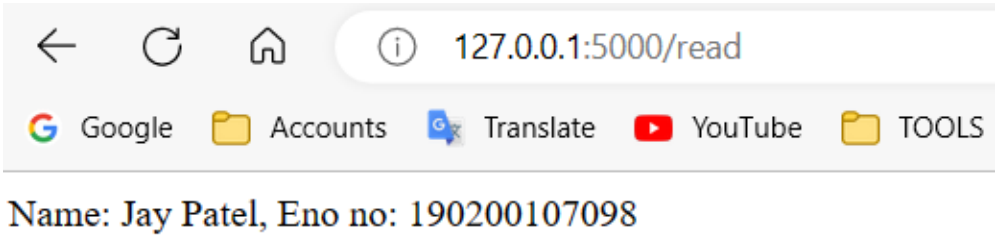


Figure 6.4.5 Flask Website Read Data Page

6.5 GUNICORN WSGI SERVER

- Gunicorn (short for Green Unicorn) is a Python Web Server Gateway Interface (WSGI) HTTP server that allows you to run multiple worker processes to handle multiple requests simultaneously. It is a pre-fork worker model, which means it forks multiple worker processes and listens to the incoming requests on a specified port.
- Gunicorn is a popular choice for deploying Python web applications, especially those built using the Flask or Django web frameworks. It can be used as a standalone server or behind a reverse proxy server such as Nginx or Apache.

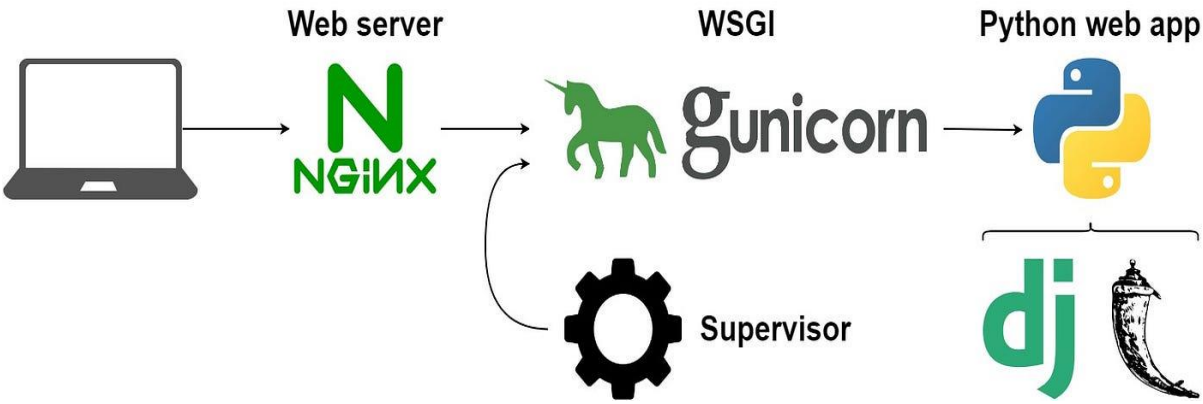


Figure 6.5 Gunicorn Webserver

- Some benefits of using Gunicorn include:

- **High concurrency**: Gunicorn's pre-fork worker model allows it to handle multiple requests concurrently, making it suitable for high-traffic websites.
- **Easy deployment**: Gunicorn is easy to set up and deploy, and it integrates well with popular web frameworks like Flask and Django.
- Graceful worker processes management: Gunicorn has built-in support for graceful restarts and stoppage of worker processes, which means it can handle new code deployments without downtime.
- **Configuration flexibility**: Gunicorn offers a range of configuration options, such as worker process count, timeouts, and logging.

⇨ Installation & Run:

```
In [*]: pip install gunicorn

        import gunicorn

In [ ]: gunicorn myapp:app
```

Figure 6.5 Gunicorn Installation Run

⇨ This tells Gunicorn to start a server listening on the default port 8000, with the WSGI callable app defined in the Python file myapp.py. You can customize various settings by bypassing command-line arguments or by using a configuration file.

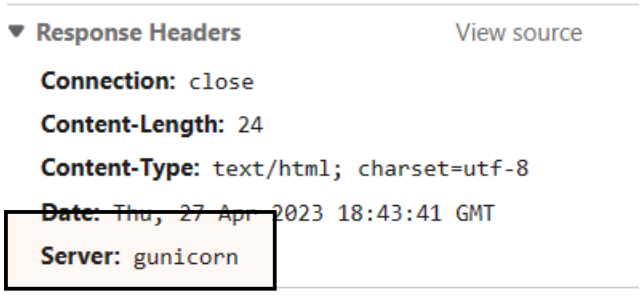


Figure 6.5 Gunicorn Server running

⇨ Overall, Gunicorn is a robust and reliable Python server that is easy to use and can handle high-traffic websites.

6.6 Flask website (Online Exam Role Base)

- **Modules I Use:**
  - Web Framework: Flask
    - Pip install flask
  - Data Base: Mongo DB
    - Pip install pymongo
  - Authentication: JWT
    - Pip install flask_jwt_extended
  - Template Rendering: Jinja
    - Default install

- Import All Modules in the main app.py file

```
from flask import Flask,render_template,request,session,redirect,url_for,flash,jsonify,Blueprint
from pymongo import MongoClient # mongodb
from flask_jwt_extended import create_access_token,get_jwt_identity,jwt_required,JWTManager,set_access_cookies,unset_jwt_cookies
from datetime import timedelta
import requests
```

Figure 6.6 Flask app Import Modules

- Flask app Configurations
- Flask app connects with database Here we use Pymongo as MongoDB

```
client = MongoClient('localhost', 27017) # connection
db = client.Website # create table
regapi = db.Userdata # triger
qna_bank = db.QNA_bank # triger
```

Figure 6.6 Flask appConnect with MongoDB

- Create All routes for each Function and add Logic for each function
- Here, @ jwt_required means pages and functions are need to authenticated by the user
- Each Function has Some Methods:
  - GET
  - POST

- **GET:** The GET method is used to retrieve data from a server. When you type a URL into a web browser, a GET request is sent to the server asking for the page or resource specified by the URL. The data is sent as part of the URL in the form of query parameters. This method is idempotent, meaning that making the same request multiple times will always yield the same result.
- **POST:** The POST method is used to submit data to a server to be processed. When you fill out a web form and click the submit button, a POST request is sent to the server with the data entered in the form as the request body. This method is not idempotent, meaning that making the same request multiple times may result in different outcomes.

```
# Page Routs
@app.route('/login',methods=['GET','POST'])
def LoginPage():...

@app.route('/logout',methods=['GET','POST'])
@jwt_required()
def Logout():...

@app.route('/register',methods=['GET','POST'])
def RegistrationPage():...

@app.route("/" ,methods=["GET"])
def HomePage():...

@app.route('/test',methods=['GET','POST'])
@jwt_required()
def TestPage():...

@app.route('/dashbord')
@jwt_required()
def Dashbord():...

@app.route('/add_qna',methods=['GET','POST'])
@jwt_required()
def add_qna():...

@app.route('/show_all',methods=['GET','POST'])
@jwt_required()
def show_all():...

@app.errorhandler(404)
def error404(error=None):
    return "<h1>Page Not Found</h1>"
```

Figure 6.6 Flask app Page Routings

- ⇨ All Routes Return Html templates
- ⇨ HTML templates are using Jinja engine to Render dynamically

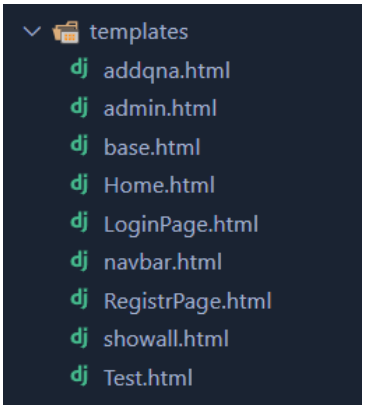


Figure 6.6 Flask app HTML Templates

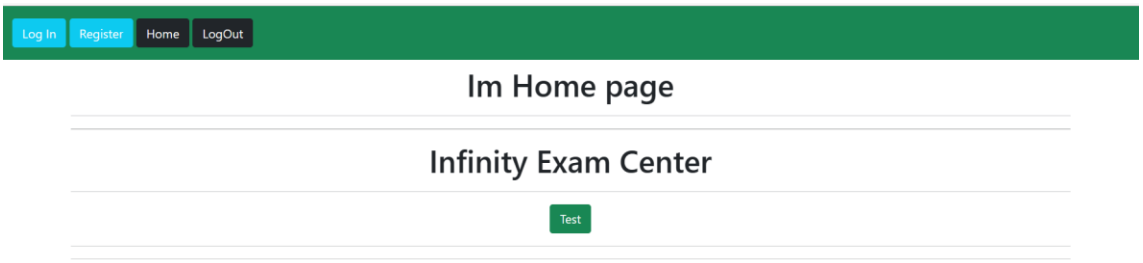


Figure 6.6 Flask app Home Page

- ⇨ If they try to Opem the Test page then redirect to login page if the user is not logged in.

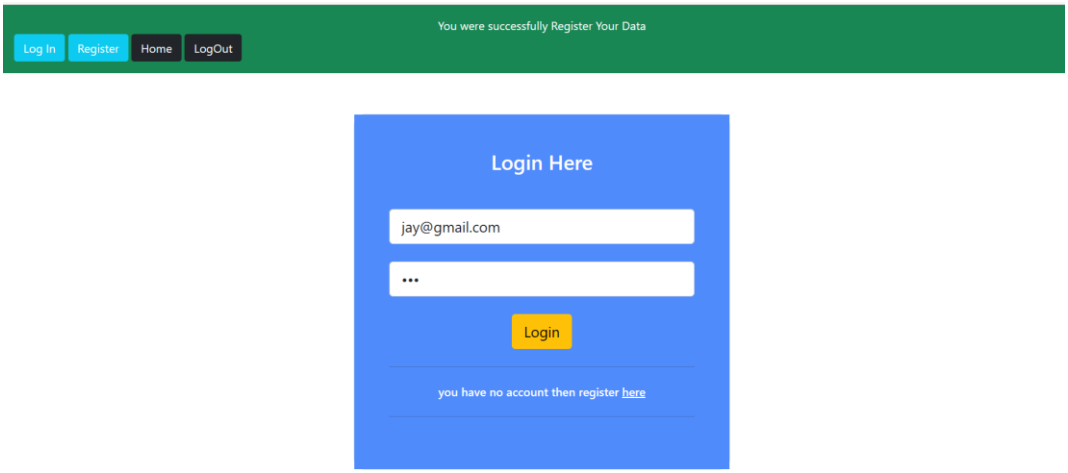


Figure 6.6 Flask app Login Page

User authentication Home Page

- ⇨ If the Login User role is Teacher, then home page display's Teacher Home Page
- ⇨ Teachers have ability to view Dashboard and Test page Both when Students are only given a test

Teachers Login:

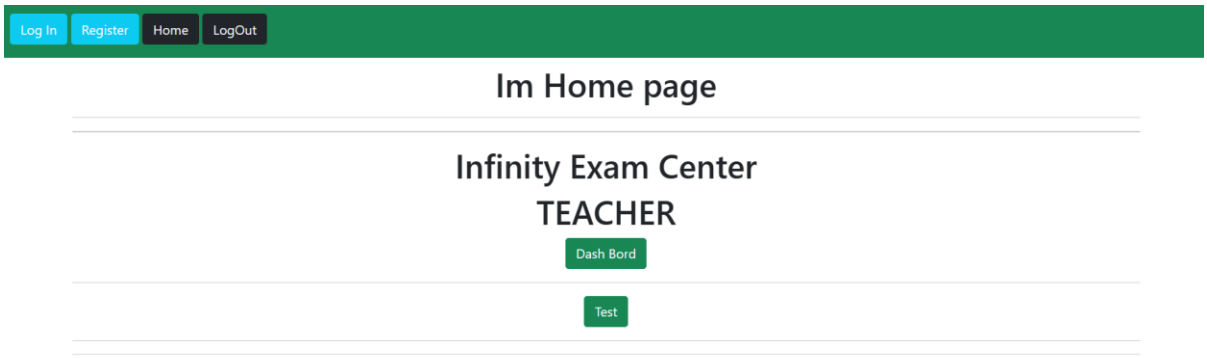


Figure 6.6 Flask App Teacher Home Page

Students Login:

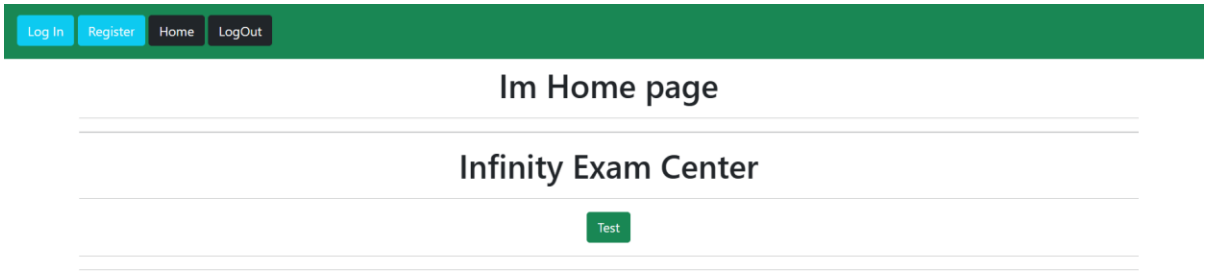


Figure 6.6 Flask app Students Home Page

- ⇨ Teacher's Ability Is They Are Creates Exerms Questions and Save In Database And Also Update and Delete Individual
- ⇨ All Teacher's Questions are show Individual Questions In their dashbord
- ⇨ Teachers can add delete and Update Test Questions:

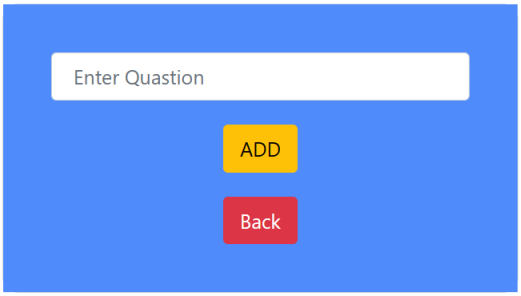


Figure 6.6 Flask app Add Questions

- ⇨ Students Are Given answers and Save them at User databases

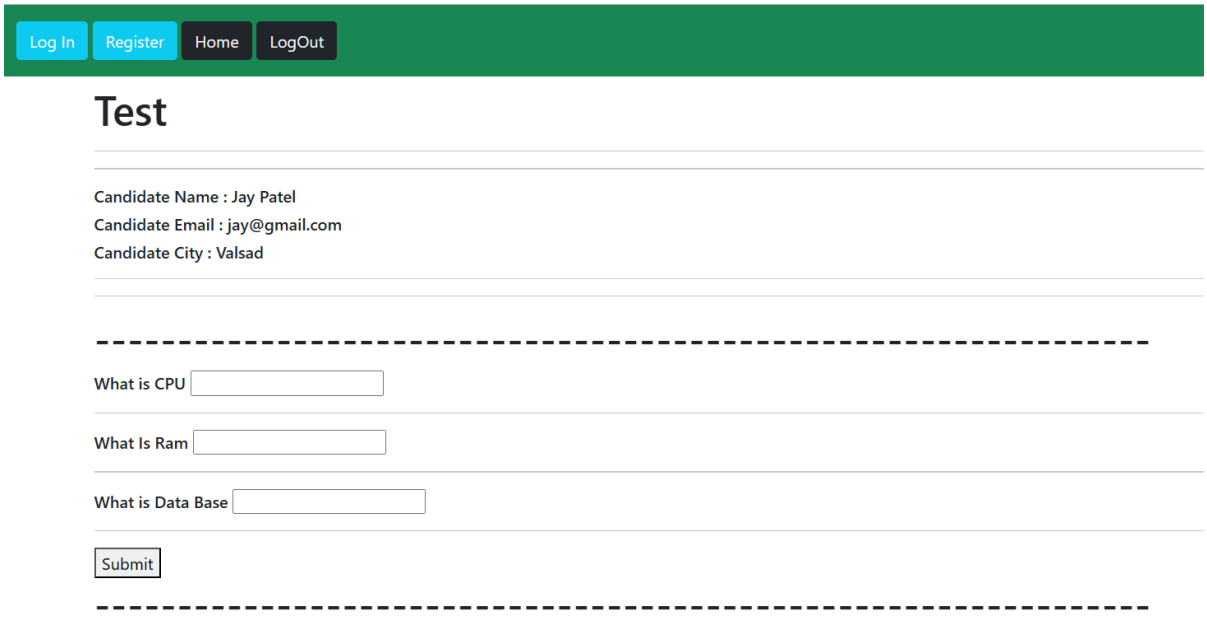


Figure 6.6 Flask app Test Page

- ⇨ This is a simple website using Bootstrap as the backend and Flask as the frontend and JWT for authentication.
- ⇨ During login and registration if the user is a teacher, then he can do some more functionality according to the teacher where the student can only take the exam.
- ⇨ This is The Basic Examples of Backend and frontend Website and I Learn Much From this Website Building

CHAPTER 7

7 DATA SCRAPING

- Data scraping, also known as web scraping, is the process of extracting data from websites. It involves the automated extraction of data from websites using software tools and techniques.
- Data scraping is used to collect information from a large number of websites quickly and efficiently. This information can then be used for various purposes, such as market research, price comparison, lead generation, content creation, and more.
- Data scraping can be done in several ways, including using specialized software tools, coding scripts to automate the process, or using browser extensions. The extracted data can be saved in various formats, such as CSV, JSON, or XML.

7.1 BEAUTIFUL SOUP

- Beautiful Soup is a popular Python library used for web scraping and parsing HTML and XML documents. It provides a simple way to navigate and search through the parse tree of an HTML or XML document and extract the data you need.
- Beautiful Soup uses a parser to parse the HTML or XML document and build a parse tree, which can then be searched and navigated using various methods and attributes. It provides a wide range of functionalities, such as searching for specific tags or attributes, extracting text, navigating the parse tree, modifying the document, and more.
- Beautiful Soup is a great choice for web scraping projects that involve extracting data from static websites that do not require user interaction.

7.2 SELENIUM

- Selenium is an open-source tool used for automating web browsers. It provides a way to automate web application testing across different browsers and platforms. Selenium supports a variety of programming languages such as Java, Python, C#, and Ruby, among others.
- Using Selenium, developers, and testers can write scripts that simulate user interactions with a web page, such as clicking buttons, filling out forms, and navigating between pages. This can help automate repetitive testing tasks, ensure that web applications work correctly across different browsers and platforms, and improve the overall quality of web applications.
- Selenium is widely used in software testing and web development, and it is considered one of the most popular automation tools for web applications.

Importing Moduls

In [12]:
```
import requests
from bs4 import BeautifulSoup
```

BeautifulSoup Class

In [2]:
```
url ="https://weather.com/en-DM/weather/tenday/l/28d595bc7ffad1010c416552a79bd576c69a3443d730c27784a0911b2e078f98"
response = requests.get(url)
soup = BeautifulSoup(response.text, 'html.parser')
```

find Each element class ¶

In [3]:
```
soup.find('span',{'class':'LocationPageTitle--PresentationName--1AMA6'}).text
```

Out[3]: 'Valsad, Gujarat, India'

In [4]:
```
soup.find('p',{'class':'DailyContent--narrative--3Ti6_'}).text
```

Out[4]: 'Generally clear. Low 26ºC. Winds SSW at 15 to 30 km/h.'

In [5]:
```
soup.find('span',{'class':'DailyContent--temp--1s3a7'}).text
```

Out[5]: '26°'

In [6]:
```
soup.find('span',{'class':'DetailsTable--value--2YD0-'}).text
```

Out[6]: '75%'

In [7]:
```
soup.find(attrs={'data-testid':'MoonriseTime'}).text
```

Out[7]: '10:57'

In [11]:
```
for wed in soup.find_all('details',{'class':'DaypartDetails--DayPartDetail--2XOOV Disclosure--themeList--1Dz21'})[:
    day = wed.find('h3',{'class':'DetailsSummary--daypartName--kbngc'})
    temp = wed.find('span',{'class':'DetailsSummary--highTempValue--3PjlX'})
    print(f' Day : {day.text}  | Temps : {temp.text}')
```

```
 Day : Tonight  | Temps : --
 Day : Thu 27  | Temps : 33°
 Day : Fri 28  | Temps : 33°
 Day : Sat 29  | Temps : 32°
```

In [16]:
```
print("@jay patel" )
```

```
@jay patel
```

Figure 7.3 Beautiful Soup Data Scrap

Importing Moduls

In [20]:
```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
```

Generate Driver for Automation

In [21]:
```
driver=webdriver.Chrome()
url = "https://www.google.com/"
driver.get(url)
```

Get Class name where Google Have Search Box

In [22]:
```
element = driver.find_element(By.NAME,"q")
```

Post Search Text

In [23]:
```
element.clear()#clear any past keys
element.send_keys("news ") # sen new keys
element.send_keys(Keys.RETURN)
```

Driver Searching elements and Text

In [24]:
```
news = driver.find_element(By.CLASS_NAME,"aUSklf")
```

Scarp Data Output

In [25]:
```
print("jay Patel News".center(30, "-"))
print(news.text)
```

```
--------jay Patel News--------
10 Cops, Driver Killed In Blast By Maoists In Chhattisgarh's Dantewada
6 mins ago
Same-Sex Marriage Hearing Live Updates: CJI criticises Centre’s reliance on controversial US Supreme Court rul
ing that took away abortion rights
LIVE18 mins ago
Delhi Public School Mathura Road receives bomb threat via email, parents rush to get kids out
5 hours ago
Contaminated syrups manufactured in India reported in WHO’s Western Pacific region
16 hours ago
AAP's Shelly Oberoi elected mayor unanimously, Aaley Md Iqbal as her deputy
2 hours ago
Huge Fire At Greater Noida Residential Apartment, No Injuries Reported
2 hours ago
```

Figure 7.4 Selenium Data Scrap

CHAPTER 8

8 DATA PRE-PROCESSING

⇨ Data preprocessing is a crucial step in any data analysis or machine learning project, as it helps to clean, transform, and prepare data for further analysis. Pandas and NumPy are popular Python libraries that provide various functions for data preprocessing. In this answer, I will explain how to handle duplication and null values using Pandas and NumPy.

8.1 PANDAS

8.1.1 HANDLING DUPLICATES

⇨ Duplicates can create bias in your analysis, and it's important to identify and remove them. Pandas provides the duplicated() and drop_duplicates() functions to handle duplicates.

⇨ duplicated() returns a Boolean Series that indicates whether each row is a duplicate of a previous row or not. You can use this function to identify duplicates in a DataFrame.

8.1.2 FEATURE SCALING

⇨ Feature scaling is the process of transforming data so that it has a similar scale. Feature scaling can improve the performance of machine learning algorithms that are sensitive to the scale of the input data, such as k-nearest neighbors and support vector machines. You can use Pandas to scale your data using different methods such as min-max scaling or standardization.

8.1.3 DATA VISUALIZATION

⇨ Data visualization using Pandas refers to the process of creating visual representations of data using the Pandas library in Python. Data visualization is an important step in data analysis, as it helps to explore and understand the data more effectively.

⇨ Pandas provides several visualization tools for exploring and analyzing data, which are built on top of the popular data visualization library, Matplotlib. Pandas' visualization functions are used to create different types of plots such as line plots, bar plots, scatter plots, histograms, and more.

⇨ Some of the advantages of data visualization using Pandas are:

⇨ **Easy to use**: Panda's visualization functions are easy to use and provide a wide range of customization options.

⇨ **Integration with Pandas data structures**: Panda's visualization functions can be directly applied to Pandas data structures such as Series and DataFrame, making it easy to visualize the data.

⇨ **Compatibility with Matplotlib**: Pandas visualization functions are built on top of Matplotlib, which means that users can use Matplotlib's functionality to customize their plots further.

- Overall, data visualization using Pandas helps to explore and analyze the data more effectively and can help to identify patterns, trends, and outliers that may be difficult to spot through other means.

8.2 NUMPY

8.2.1 HANDLING NULL VALUES

- Null values or missing values can be a common problem in datasets, and it's important to handle them properly. Pandas provides the isna() and fillna() functions to handle null values.
- isna() returns a Boolean DataFrame that indicates whether each element is NaN (not a number) or not. You can use this function to identify null values in a DataFrame.

26/04/2023, 16:31 Pandas Pre-processing - Jupyter Notebook

In [25]:
```
import pandas as pd
```

In [26]:
```
data = {'Name': ['Alice', 'Bob', 'Charlie', 'Dave', 'Eve'],
        'Age': [25, 30, None, 35, 40],
        'Salary': [50000, None, 60000, 70000, 80000]}
df = pd.DataFrame(data)
```

Row data frame

In [27]:
```
df
```

Out[27]:

	Name	Age	Salary
0	Alice	25.0	50000.0
1	Bob	30.0	NaN
2	Charlie	NaN	60000.0
3	Dave	35.0	70000.0
4	Eve	40.0	80000.0

Drop null values

In [28]:
```
df = df.dropna()
```

Convert Age and Salary columns to integers

In [ ]:
```
df['Age'] = df['Age'].astype(int)
df['Salary'] = df['Salary'].astype(int)
```

Add a new column based on Salary

In [ ]:
```
df['Salary Category'] = pd.cut(df['Salary'], bins=[0, 50000, 75000, 100000], labels=['Low', 'Medium', 'High'])
```

Cleand Processed Data

In [22]:
```
df
```

Out[22]:

	Name	Age	Salary	Salary Category
0	Alice	25	50000	Low
3	Dave	35	70000	Medium
4	Eve	40	80000	High

In [24]:
```
print("@Jay Patel")
```

```
@Jay Patel
```

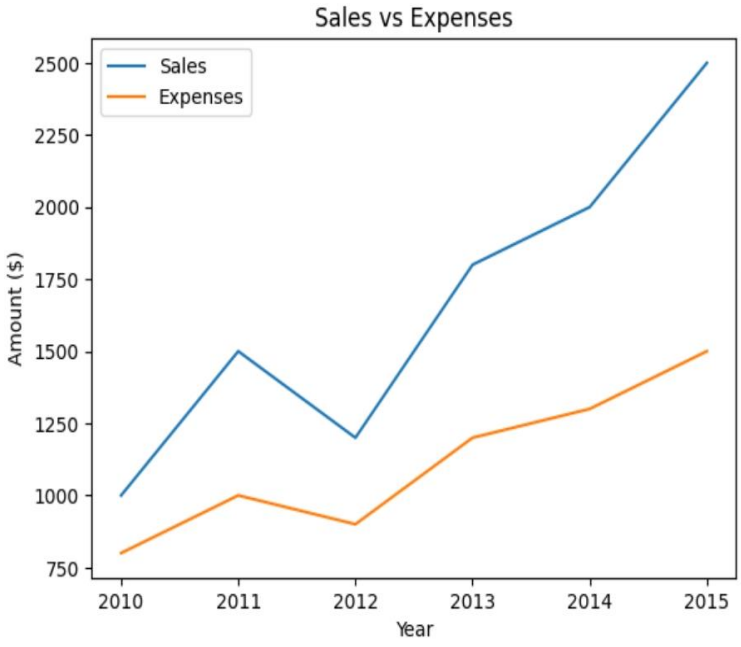
Figure 8. Data Pre-Processing

Data Visualization

In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In [3]:
```
data = {'Year': [2010, 2011, 2012, 2013, 2014, 2015],
        'Sales': [1000, 1500, 1200, 1800, 2000, 2500],
        'Expenses': [800, 1000, 900, 1200, 1300, 1500]}
df = pd.DataFrame(data)
```

In [4]:
```
plt.plot(df['Year'], df['Sales'], label='Sales')
plt.plot(df['Year'], df['Expenses'], label='Expenses')
plt.legend()
plt.title('Sales vs Expenses')
plt.xlabel('Year')
plt.ylabel('Amount ($)')
plt.show()
```



In [5]:
```
print("@jay patel")
```

```
@jay patel
```

Figure 8. Data Visualization Using Matplotlib

CHAPTER 9

9 BASIC MACHINE LEARNING

- ⇨ Machine learning is a field of artificial intelligence (AI) that focuses on creating algorithms and models that can learn from and make predictions on data. The goal of machine learning is to enable computers to learn and improve from experience without being explicitly programmed, by recognizing patterns and making decisions based on data inputs.
- ⇨ Machine learning algorithms are typically categorized into three main types: supervised learning, unsupervised learning, and reinforcement learning.
- ⇨ Supervised learning: In supervised learning, the algorithm is trained on a labeled dataset, which means that the inputs and outputs are provided to the algorithm. The goal is to learn a function that maps inputs to outputs so that it can make accurate predictions on new, unseen data.
- ⇨ Unsupervised learning: In unsupervised learning, the algorithm is trained on an unlabeled dataset, which means that only the inputs are provided to the algorithm. The goal is to find patterns or structures in the data without any specific guidance on what to look for.
- ⇨ Reinforcement learning: In reinforcement learning, the algorithm learns by interacting with an environment, and receives feedback in the form of rewards or punishments. The goal is to learn a policy that maximizes the cumulative reward over time.

9.1SCIKIT LEARN

- ⇨ Scikit-learn (also known as sklearn) is a popular machine learning library for Python. It provides a wide range of algorithms and tools for various machine learning tasks, such as classification, regression, clustering, dimensionality reduction, and model selection.
- ⇨ Scikit-learn is built on top of other popular scientific computing libraries in Python, such as NumPy, SciPy, and Matplotlib. This makes it easy to integrate with other data analysis and visualization tools.
- ⇨ Scikit-learn includes a variety of tools and functions for data preprocessing, model evaluation, and hyperparameter tuning. It also provides a consistent and easy-to-use API for fitting machine learning models and making predictions on new data.
- ⇨ Some of the popular machine learning algorithms available in scikit-learn include:

9.1.1 LINEAR REGRESSION

- Linear regression is a type of supervised machine learning algorithm that is used to model the relationship between a dependent variable (also known as the target variable) and one or more independent variables (also known as features).
- In Scikit-learn, linear regression can be implemented using the Linear Regression class. The linear regression class estimates the coefficients of a linear equation to best fit the given data. The equation is of the form:
- y = b0 + b1x1 + b2x2 + ... + bn*xn
- Linear regression is widely used for tasks such as predicting housing prices, stock prices, and sales figures based on historical data. Scikit-learn's implementation of linear regression makes it easy to apply this powerful algorithm to real-world datasets in a few simple steps.

9.1.2 LOGISTIC REGRESSION

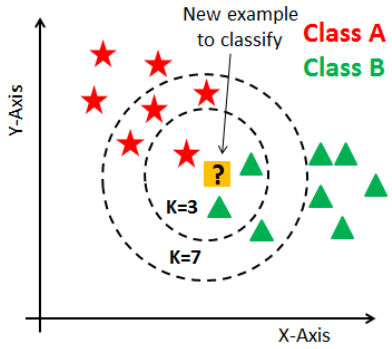
- Logistic Regression is a popular algorithm for binary classification problems. It is used to predict the probability of a binary response (e.g., whether a customer will buy a product or not) based on one or more predictor variables.
- Scikit-learn is a popular Python library for machine learning, and it provides an implementation of logistic regression in its linear_model module.
- Logistic regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes). It is used to model the probability of a certain class or event existing, such as pass/fail, win/lose, or healthy/sick.
- The logistic regression model takes the form of:
- P(y=1|X) = 1 / (1 + exp(-z))

9.1.3 SUPPORT VECTO MACHINE

- Support Vector Machines (SVM) is a machine learning algorithm used for classification and regression analysis. It works by finding the best-separating hyperplane that maximizes the margin between the two classes. In scikit-learn, SVM can be implemented using the SVC (Support Vector Classification) class for classification and SVR (Support Vector Regression) class for regression.
- The formula for SVM is based on finding the hyperplane that maximizes the margin between the two classes. In the case of binary classification, the hyperplane is a line that separates the two classes.
- The SVM algorithm finds the hyperplane that maximizes the distance between the closest points of the two classes, which are called support vectors.
- The formula for the hyperplane is:
- w^T x + b = 0

9.1.4 KNN

- K-Nearest Neighbors (KNN) is a simple and popular machine learning algorithm used for classification and regression tasks. In KNN, the class of a data point is determined by the class of its k nearest neighbors in the training dataset.
-



9.1.5 NAÏVE BAYES

- Naive Bayes is a family of probabilistic classification algorithms based on Bayes' theorem with an assumption of independence between the features. The Naive Bayes classifier is commonly used in text classification, spam filtering, sentiment analysis, and recommendation systems. It is a simple and efficient algorithm that can work well even with a small amount of training data.
- The formula for the Naive Bayes classifier is:
- $P(y|x) = P(x|y) * P(y) / P(x)$

9.1.6 DECISION TREE

- Decision Tree is a supervised machine learning algorithm that is commonly used for classification and regression tasks. The algorithm creates a tree-like model of decisions and their possible consequences. It starts with a single node, called the root, that represents the entire dataset.
- The algorithm then splits the dataset into smaller subsets by evaluating the value of a feature in each observation. This process continues recursively for each subset until a stopping criterion is met, such as a maximum depth of the tree, a minimum number of observations per leaf node, or a minimum impurity threshold.
- Decision Trees are prone to overfitting the training data, especially if the tree is deep and complex. To address this issue, various techniques can be used, such as pruning, ensemble methods, and regularization. Scikit-learn provides a robust implementation of Decision Trees and their variants, such as Random Forests, Gradient Boosted Trees, and Multi-output Trees, that can handle both categorical and continuous features and perform well on various datasets.

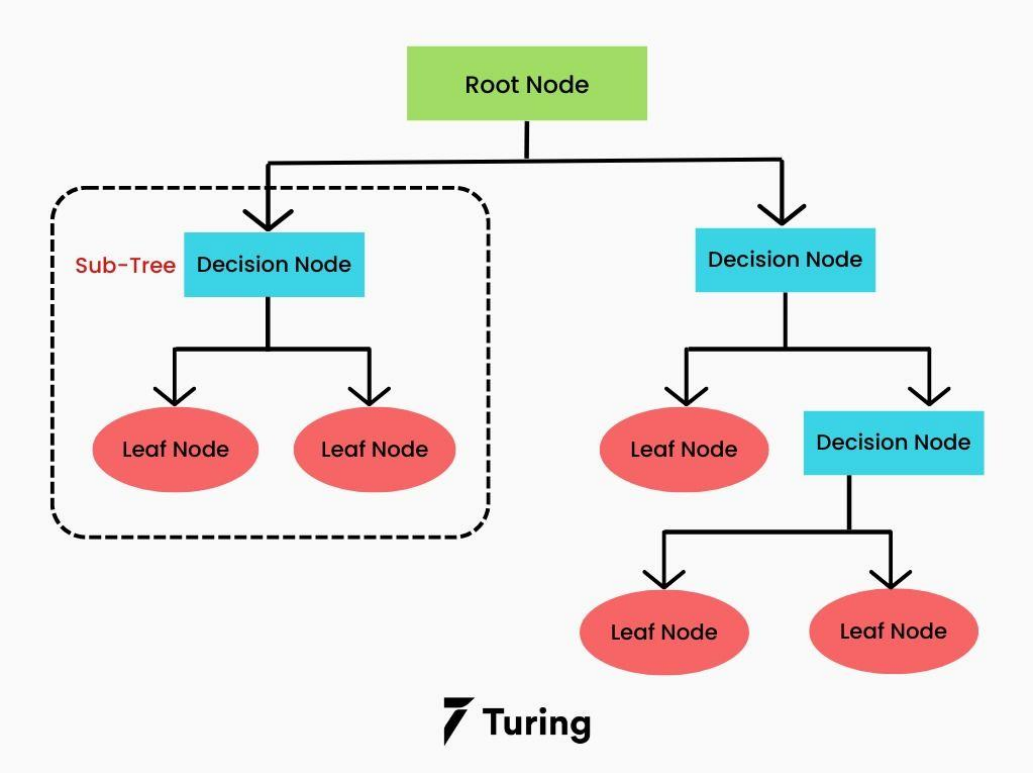


Figure 9.3.6 Decision Tree Flow Chart

9.1.7 RANDOM FOREST

⇨ Random Forest is a popular ensemble learning technique that combines multiple decision trees to improve the performance of a machine learning model. In Random Forest, multiple decision trees are trained on different subsets of the data, and the final prediction is made by averaging the predictions of all trees.

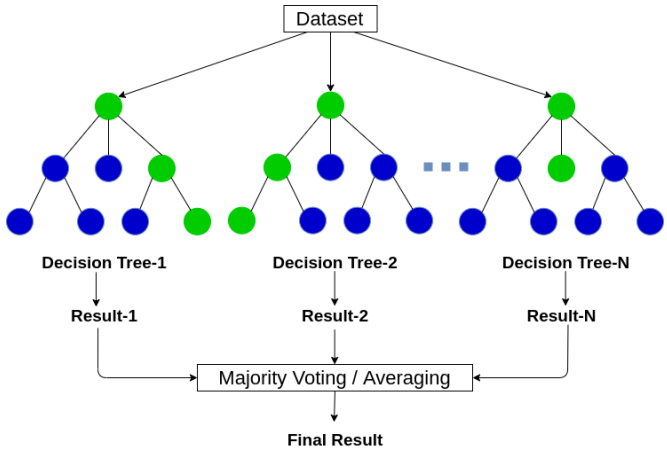


Figure 9.3.7 Random Forest Tree Flow Chart

9.2NLTK

- ⇨ NLTK stands for Natural Language Toolkit and it is a popular Python library for working with human language data. It provides a wide range of tools and resources for tasks such as tokenization, stemming, part-of-speech tagging, and sentiment analysis.
- ⇨ NLTK (Natural Language Toolkit) provides a wide range of features for natural language processing tasks. Some of the key features of NLTK are:
  - **Tokenization**: NLTK can break down the text into individual words and sentences, known as tokenization.
  - **Part-of-speech (POS) tagging**: NLTK can assign each word in a text a part-of-speech tag (e.g. noun, verb, adjective, etc.), which is helpful for many NLP tasks.
  - **Stemming and lemmatization**: NLTK can reduce words to their root forms, which is useful for tasks such as document classification and search engines.
  - **Named entity recognition (NER)**: NLTK can identify and extract named entities (such as people, organizations, and locations) from the text.
  - **Sentiment analysis**: NLTK can determine the sentiment of a text (i.e. whether it is positive, negative, or neutral).

```python
import nltk
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()

nltk.download('punkt') # download the necessary resources

sentence = "Im Going To School for Study"
tokens = nltk.word_tokenize(sentence)
print("seprate each Word ",tokens)
def mynltk(text):
    y = []
    for i in text.split():
        y.append(ps.stem(i))
    return " ".join(y)
rowword =[]
for i in tokens:
    rowword.append(mynltk(i))

print(" this is row Word ",rowword)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
PS C:\Users\jaypa> python -u "c:\Users\jaypa\Desktop\MY PROGRAMING\All_Stuff_ROW-main\Machin Learning\ML_Mo
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\jaypa\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
seprate each Word  ['Im', 'Going', 'To', 'School', 'for', 'Study']
 this is row Word  ['im', 'go', 'to', 'school', 'for', 'studi']
PS C:\Users\jaypa>
```

Figure 9.4 NLTK

CHAPTER 10

10 JETFLIX PROJECT

10.1 INTRODUCTION

- Movie recommendation systems are software applications designed to suggest movies to users based on their viewing history, preferences, and behavior. The primary goal of these systems is to personalize movie recommendations for users and improve their overall viewing experience. Movie recommendation systems are widely used by streaming platforms such as Netflix, Hulu, and Amazon Prime Video to provide users with a customized list of movies to watch.

- Another reason for the importance of movie recommendation systems is their ability to increase user engagement and retention. By suggesting personalized recommendations, these systems encourage users to spend more time on the platform and watch more movies.

- Movie recommendation systems use a variety of techniques to suggest movies to users. One of the most popular techniques is collaborative filtering, which involves analyzing the viewing behavior of users to identify patterns and similarities. This technique uses two approaches: user-based and item-based. In the user-based approach, the system identifies users with similar viewing behavior and recommends movies based on their collective behavior. In the item-based approach, the system identifies movies with similar characteristics and recommends movies based on their similarities.

- Overall, movie recommendation systems use a combination of techniques to analyze user behavior and movie features to suggest personalized recommendations to users. By doing so, these systems improve the overall viewing experience of users and increase user engagement and retention on streaming platforms.

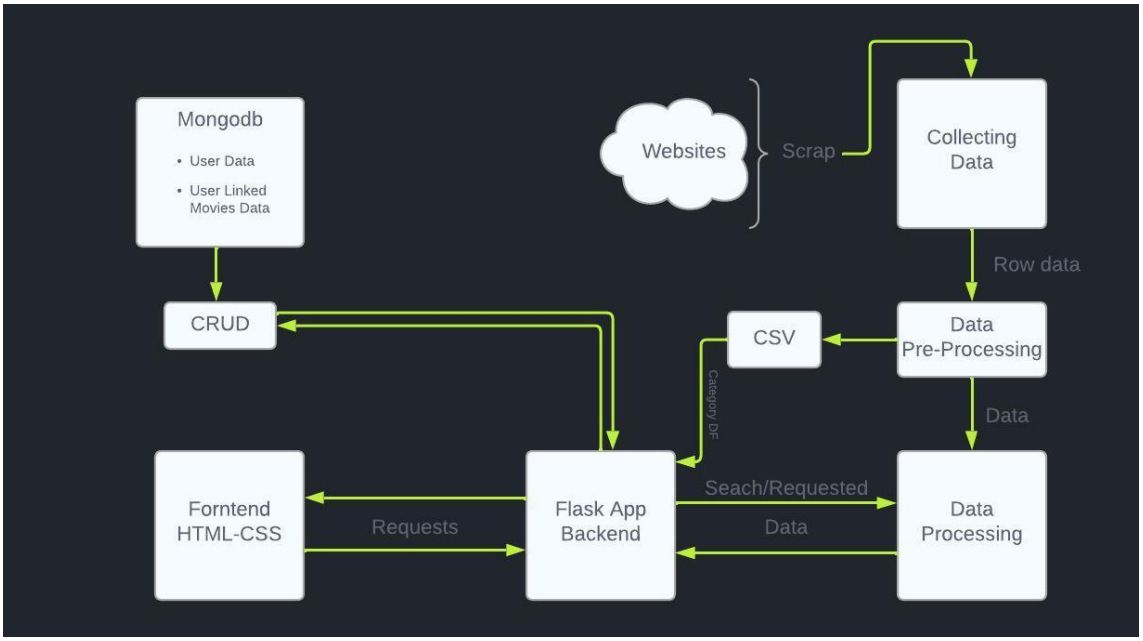


Figure 10.1 JETFLIX Flowchart

10.2 FRONTEND

⇨ For Frontend I searched Google for a suitable template A lot of changes were made to it

- Home Page
- Login Page
- Registration Page
- Movies Details Page
- Search Customization Page

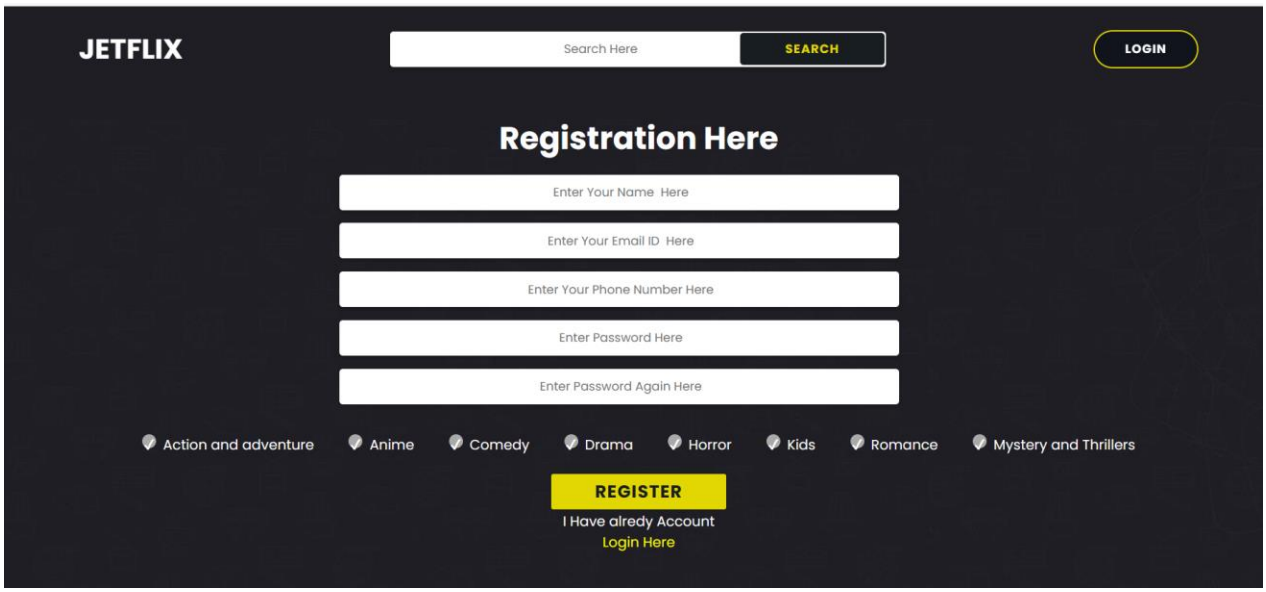


Figure 10.2 JETFLIX Registration Page

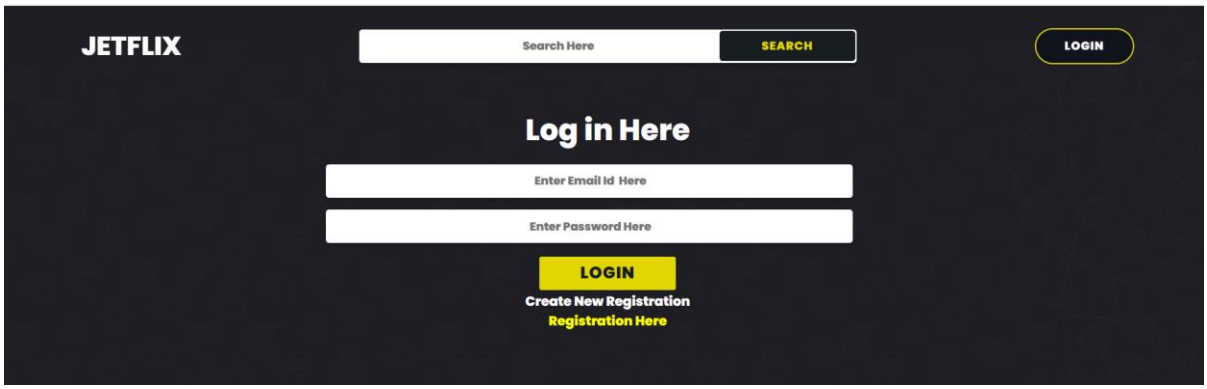


Figure 10.2 JETFLIX Login Page

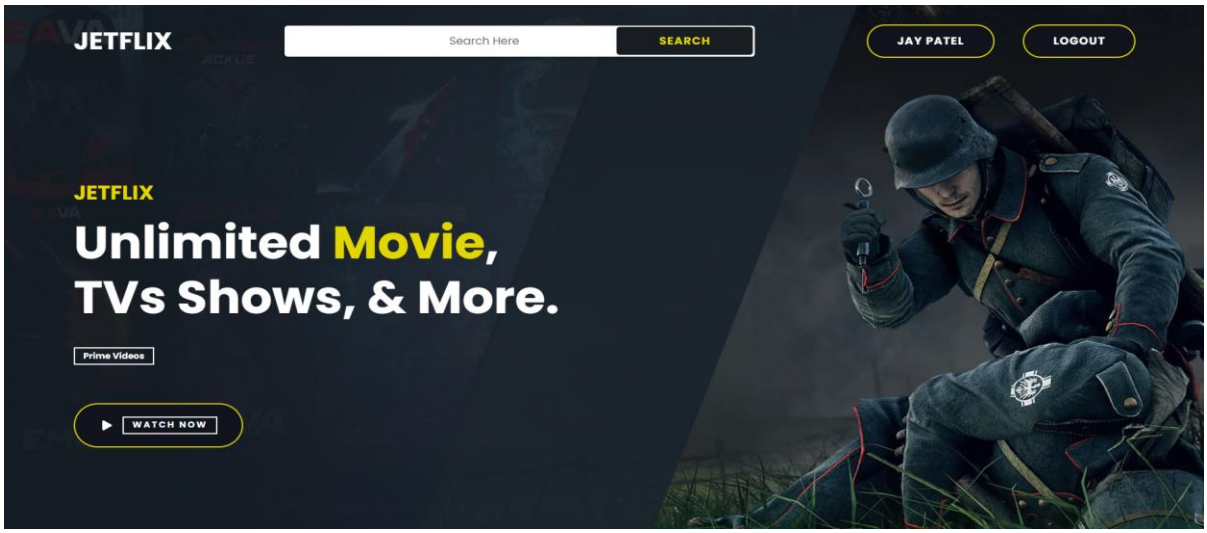


Figure 10.2 JETFLIX Home Page

- ⇨ All Pages Made by HTML Custom CSS and Boostrap with Google tamplates
- ⇨ I have for backend, I have googled the template and made a few changes in the templates and make all pages .

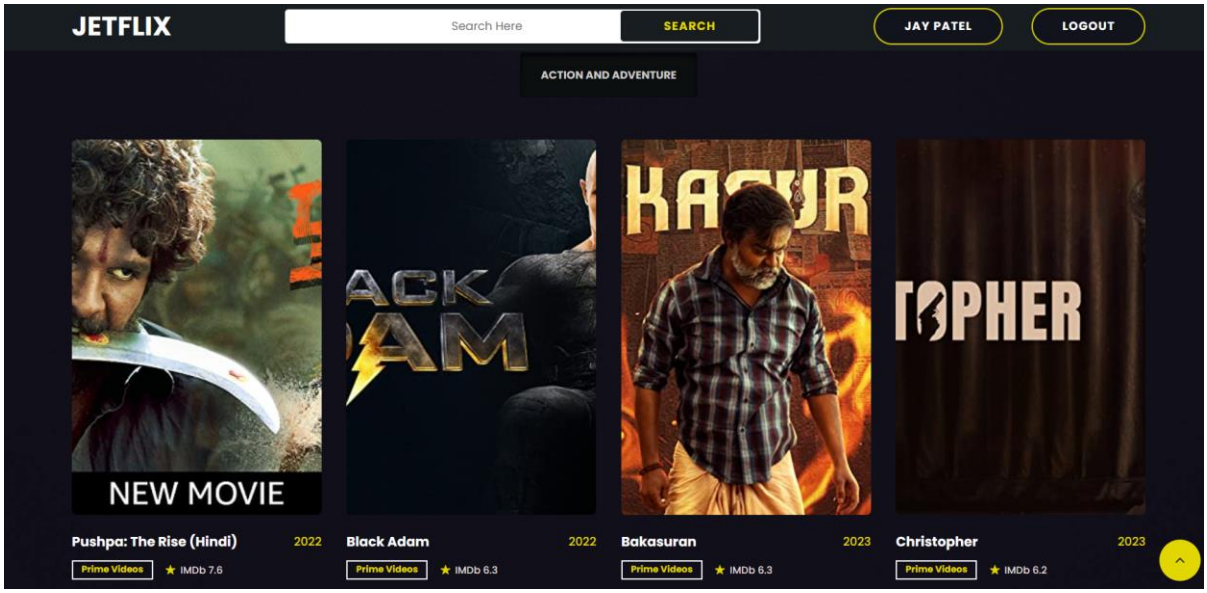


Figure 10.2 JETFLIX Home Page Movies data

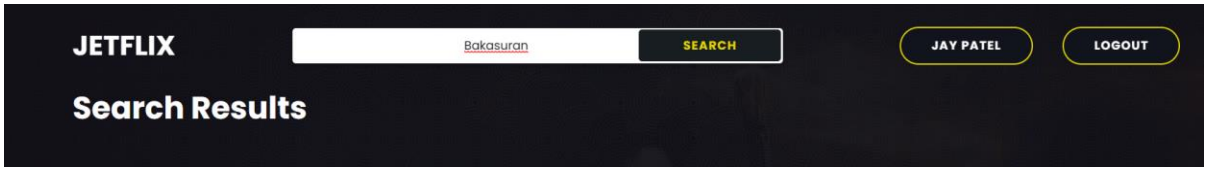


Figure 10.2 JETFLIX Home Page Search Box

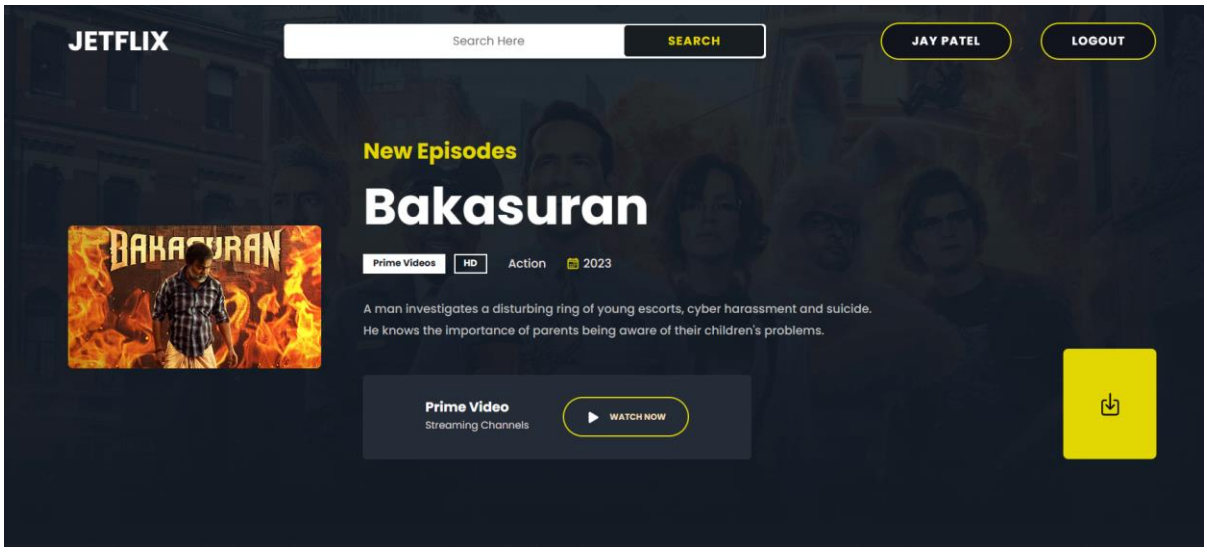


Figure 10.2 JETFLIX Movies Details Page

10.3 BACKEND

- I have used python flask as the backend and Mongo DB as Database, JWT for Authentication and Save User Information in Session, and cookies
- Also, Create my Costume Modules for Data Analysis and Data Processing
- Data gathering from modules and processes for use as user response

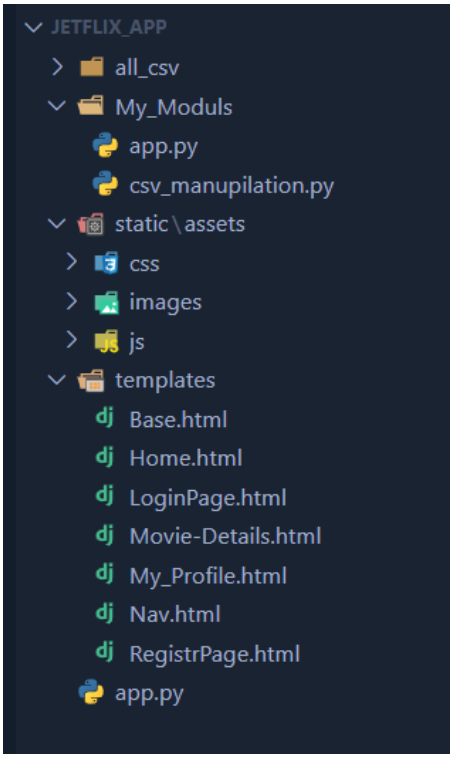


Figure 10.3 JETFLIX File Structure

10.4 DATA SCRAPING

- I did a lot of searches for movie data but I got some errors in each one so I decided to scrape the website myself I scraped the data from Amazon Prime videos.
- To scrape the data, first search for the required item on the website, then first scrape the categories of the movies. After that, takes the list of movies in the category and does data scraping of movies, including IMDB ratings and links and images of movies.
- Then this data is saved in CSV according to the category

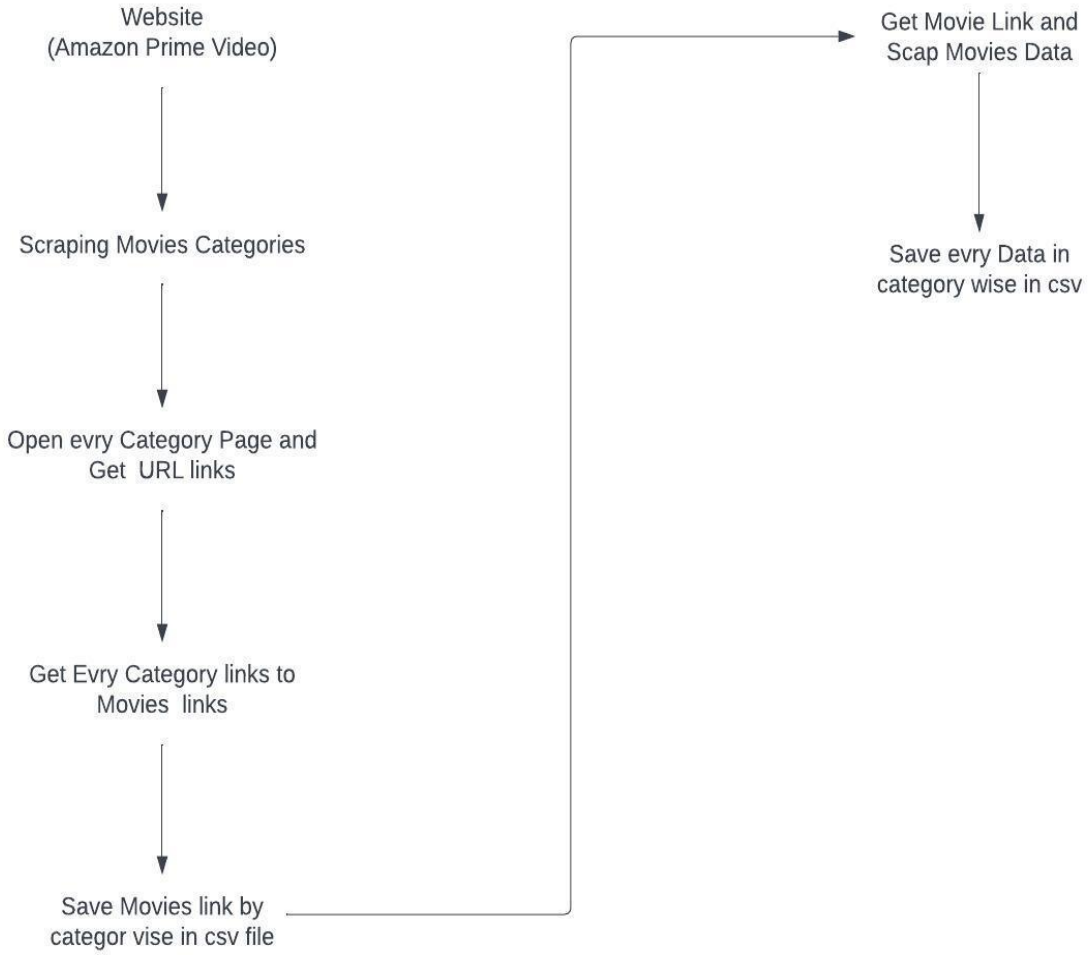


Figure 10.4 JETFLIX Data Scraping Flow

- ⇨ For the scraping, we use beautiful soup 4, and for managing data we use Pandas
- ⇨ Saving all the row Scrap data in CSV using Data Frame and Pandas
- ⇨ Try small-small test program part and In the end, combine all the programs and Create one main program to scarp all data in one click

In [1]:
```
import requests
from bs4 import BeautifulSoup
import time
```

In [2]:
```
url = "https://www.primevideo.com/categories/ref=atv_hm_pri_c_9zZ8D2_1_3"
page = requests.get(url)
soup = BeautifulSoup(page.content, "html.parser")
```

Get Category

In [3]:
```
def get_Categorys():
    Category = []
    movies = soup.find_all("h3", class_="bUIMWe")
    for movie in movies:
        category = movie.find('span',{'class':'_9qXVlg gg1gAB rnNSvV'}).text
        Category.append(category)
    #print(Category)
    return Category
```

In [4]:
```
def main_page_url(url):
    page = requests.get(url)
    soup = BeautifulSoup(page.content, "html.parser")
    page_url = soup.find('a',{'aria-label':'See more'})['href']
    return page_url
```

Get cat Page

In [5]:
```
#https://www.primevideo.com/
def get_categorys_page():
    #Category = ["Action and adventure","Anime"]
    Category = get_Categorys()
    category_page_list = []
    for i in Category:
        for link in soup.find_all("a", {"aria-label": i}):
            myurl = f"https://www.primevideo.com{link.get('href')}"
            main_link = main_page_url(myurl)
            #print(i)
            category_page = f"https://www.primevideo.com{main_link}"
            category_page_list.append(category_page)
    return category_page_list
```

Get All Movies Web_Link, Thombnil and from Home page

In [6]:
```
def get_movies_link(url_page):
    page = requests.get(url_page)
    soup = BeautifulSoup(page.content, "html.parser")
    links = soup.find_all('article',{'class':'Z9dd1d'})
    Movie_links_all = []

    for link in links:
        web_link_child= link.find('a',{'class':'_3HZFFn'})['href']
        web_link = f"https://www.primevideo.com{web_link_child}"
        thombnil_link= link.find('img',{'class':'pmtwKl u2YE+F Ah1hNY'})['src']
        Movie_links_all.append([web_link,thombnil_link])

    return Movie_links_all
     # badhi movies ni link male home page par thi
```

Save All data in CSV file

In [7]:
```
import csv
import pandas as pd
def Main_APP():
    category_page = get_categorys_page()
    all_movies_links = []
    n=1
    for megalink in category_page:#darek movies ni link malse
        Movie_links_all = get_movies_link(megalink)
        all_movies_links.append(Movie_links_all)

    return all_movies_links
```

Figure 10.4 JETFLIX Data Scraping Code

Run all Function

```
In [8]: Datas_links = Main_APP()
        import pandas as pd

        df = pd.DataFrame(Datas_links)
        new_df = df.T
        new_df.columns =get_Categorys()
        new_df.to_csv("main_links.csv")
```

Figure 10.4 JETFLIX Data Scraping Code Run

⇨ Outcome: Get 10 Movie links for Each category and save them in a CSV file.

	Action an	Anime	Comedy	Documen	Drama	Fantasy	Horror	Internatio	Kids	Music vid	Mystery a	Romance	Hindi	English	Telugu	Tamil	Malayalar	Kannada	Marathi	Punjabi	Gujarati	Bengali
0	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://www.prime
1	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://www.prime
2	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://www.prime
3	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://www.prime
4	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://www.prime
5	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://www.prime
6	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://\	['https://www.prime

Figure 10.4 JETFLIX Data Scraping Output

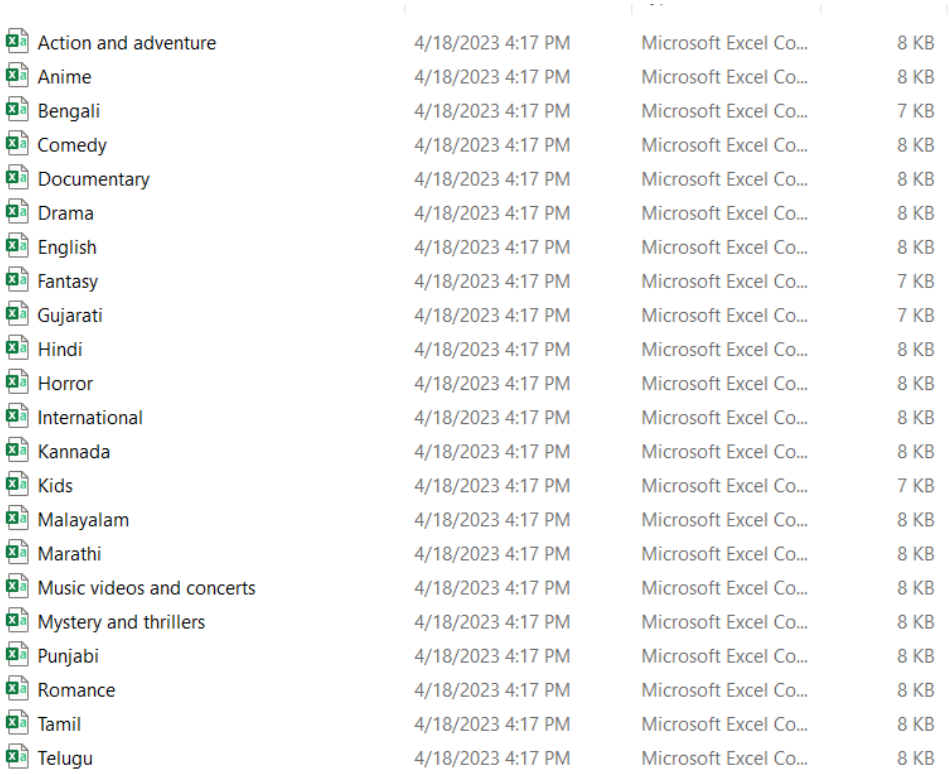
⇨ For Each Movie, Scarp data from the Movies URL link

```
In [1]: import pandas as pd
        import requests
        from bs4 import BeautifulSoup



        def checking_null(element):
            if element is not None:
                return element.text
            else:
                return "None"

        def scrap_movie_data():
            df = pd.read_csv('main_links.csv')
            df = df.dropna()
            data = {'title': [], 'disciption': [],'Imdb_rating': [], 'genres': [],'Released_date': [], 'image_url': [],'
            columsn = df.columns
            for i in columsn[1:]:
                result_df = pd.DataFrame(data)
                for link_row in df[i]:
                    link = link_row.split(", ")[0].replace("['","").replace("'","")
                    page = requests.get(link)
                    soup = BeautifulSoup(page.content, 'html.parser')

                    title = soup.find('h1',{'data-automation-id':'title'})
                    title = checking_null(title)

                    disciption = soup.find('div',{'class':'_3qsVvm e8yjMf'})
                    disciption = checking_null(disciption)

                    Imdb_rating = soup.find('span',{'data-automation-id':'imdb-rating-badge'})
                    Imdb_rating = checking_null(Imdb_rating)

                    genres = soup.find('div',{'data-testid':'genresMetadata'})
                    genres = checking_null(genres)

                    Released_date = soup.find('span',{'data-automation-id':'release-year-badge'})
                    Released_date = checking_null(Released_date)

                    image = soup.find('div',{'data-automation-id':'hero-background'})
                    image_url= image.find('img',{'class':'u2YE+F Ah1hNY'})['src']
                    thombnil_url =  image_url_2 = link_row.split(", ")[1].replace("']","").replace("'","")

                    result_df = result_df.append({'title': title, 'disciption': disciption,'Imdb_rating': Imdb_rating, '
                print(result_df)
                result_df.to_csv(f"all_csv/{i}.csv")
```

```
In [190]: scrap_movie_data()
```

Figure 10.4 JETFLIX Scraping Movies Details Code

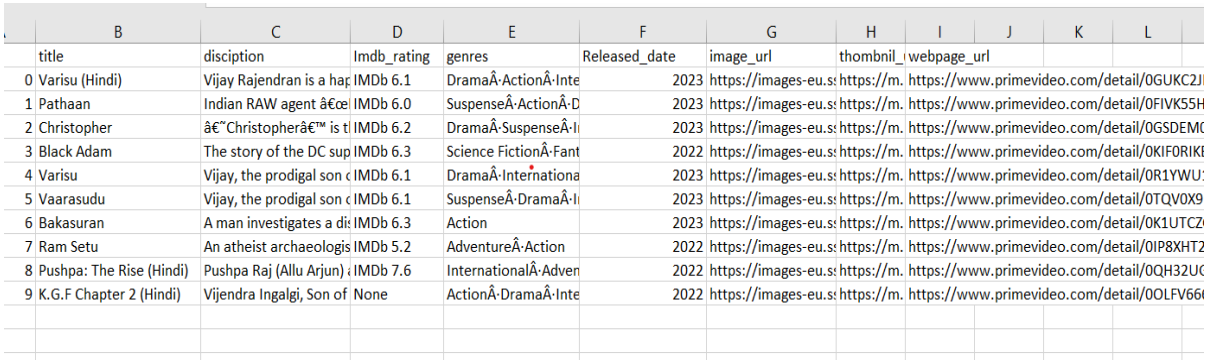
⇨ Finally, we get all movie data as categories Wise.



Action and adventure	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Anime	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Bengali	4/18/2023 4:17 PM	Microsoft Excel Co...	7 KB
Comedy	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Documentary	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Drama	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
English	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Fantasy	4/18/2023 4:17 PM	Microsoft Excel Co...	7 KB
Gujarati	4/18/2023 4:17 PM	Microsoft Excel Co...	7 KB
Hindi	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Horror	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
International	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Kannada	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Kids	4/18/2023 4:17 PM	Microsoft Excel Co...	7 KB
Malayalam	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Marathi	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Music videos and concerts	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Mystery and thrillers	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Punjabi	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Romance	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Tamil	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB
Telugu	4/18/2023 4:17 PM	Microsoft Excel Co...	8 KB

Figure 10.4 JETFLIX Scraping Movies Details Data Storing In CSV

⇨ Each Movie Data Sample:



	B	C	D	E	F	G	H	I	J	K	L
	title	disciption	Imdb_rating	genres	Released_date	image_url	thombnil_	webpage_url			
0	Varisu (Hindi)	Vijay Rajendran is a hap	IMDb 6.1	DramaÂ·ActionÂ·Inte	2023	https://images-eu.s	https://m.	https://www.primevideo.com/detail/0GUKC2J			
1	Pathaan	Indian RAW agent â€œ	IMDb 6.0	SuspenseÂ·ActionÂ·D	2023	https://images-eu.s	https://m.	https://www.primevideo.com/detail/0FIVK55H			
2	Christopher	â€™Christopherâ€™ is t	IMDb 6.2	DramaÂ·SuspenseÂ·I	2023	https://images-eu.s	https://m.	https://www.primevideo.com/detail/0GSDEMC			
3	Black Adam	The story of the DC sup	IMDb 6.3	Science FictionÂ·Fant	2022	https://images-eu.s	https://m.	https://www.primevideo.com/detail/0KIF0RIKI			
4	Varisu	Vijay, the prodigal son c	IMDb 6.1	DramaÂ·Internationa	2023	https://images-eu.s	https://m.	https://www.primevideo.com/detail/0R1YWU			
5	Vaarasudu	Vijay, the prodigal son c	IMDb 6.1	SuspenseÂ·DramaÂ·I	2023	https://images-eu.s	https://m.	https://www.primevideo.com/detail/0TQV0X9			
6	Bakasuran	A man investigates a dis	IMDb 6.3	Action	2023	https://images-eu.s	https://m.	https://www.primevideo.com/detail/0K1UTCZ			
7	Ram Setu	An atheist archaeologis	IMDb 5.2	AdventureÂ·Action	2022	https://images-eu.s	https://m.	https://www.primevideo.com/detail/0IP8XHT2			
8	Pushpa: The Rise (Hindi)	Pushpa Raj (Allu Arjun)	IMDb 7.6	InternationalÂ·Adven	2022	https://images-eu.s	https://m.	https://www.primevideo.com/detail/0QH32U			
9	K.G.F Chapter 2 (Hindi)	Vijendra Ingalgi, Son of	None	ActionÂ·DramaÂ·Inte	2022	https://images-eu.s	https://m.	https://www.primevideo.com/detail/0OLFV66			

Figure 10.4 JETFLIX Movies Data Semple

10.5 DATA PRE-PROCESSING

- ⇨ Data filtering and managing null values in all files
- ⇨ Delete Duplicate values, If data have some Null or Blank Values then Replace them with a Numpy's Null values
- ⇨ Finally, we get to use full and true Data

10.6 USER INTERACTION

- ⇨ When using Are search for some Movies Then the Function calls and fetches the data from CSV files, For Global Search We Need to merge all the in one main data frame
- ⇨ When search text is get using HTML Form, we extract the movies index from the CSV file
- ⇨ We need all the data from this searched Movie, As well as We need similar movies of searched movies. So, we need Data Processing Models
- ⇨

10.7 DATA PROCESSING OUTPUT

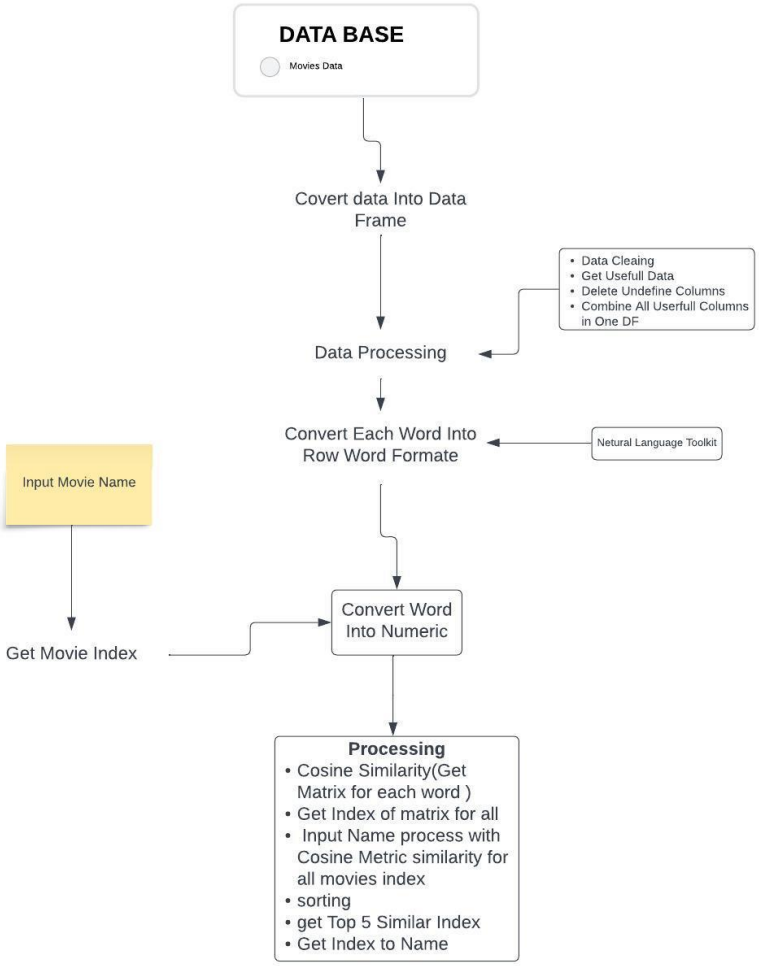


Figure 10.7 JETFLIX Data Processing Flow Chart

- ⇨ Whenever the user inputs a name, the data is retrieved from the CSV files.
- ⇨ The received data will be displayed on the home page as a Movies details Page
- ⇨ I need also similar movies so first we create one main tagline for each movie we merge the name, year, categories, and Descriptions and make one tagline using NLTK
- ⇨ therefor we cross-check all movies which is similar to the movies and get a list
- ⇨ sorting all movies and get the top 5 Movies
- ⇨ Display similar movies as the Suggested Movies List

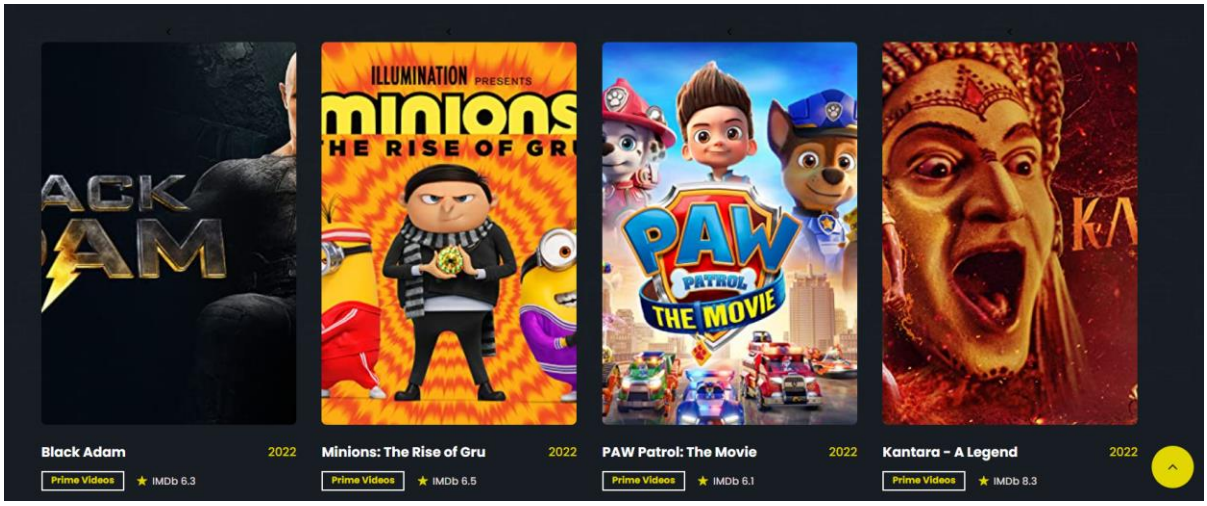


Figure 10.7 JETFLIX Suggestions

10.8 STORING DATA

- ⇨ Login User Subscribe any Movies then the movies data are store in User database
- ⇨ Ex: Subscribe Black Adam Movie

```
_id: ObjectId('644b931f48a3d2cc98e9cdf7')
name: "Jay Patel"
email: "jay@gmail.com"
phone: "123"
password: "321321123"
▾ checkbox: Array
    0: "Action and adventure"
save_genres: "Science Fiction·Fantasy·Adventure·Action"
save_title: "Black Adam"
```

Figure 10.8 JETFLIX Users data Manipulations

- ⇨ Finally, this project ends and we will learn many technical things from this project

CHAPTER 11

11 DEPLOYMENTS

11.1 DOCKER

- Docker is an open-source platform that allows developers to create, deploy, and run applications in isolated environments called "containers."
- These containers are lightweight, portable, and can run on any platform, making it easier for developers to build and test their applications in a consistent and repeatable way.
- Docker provides a way to package an application and its dependencies into a single container that can be easily distributed and deployed to various environments, from development to production.
- This makes it easier for teams to collaborate on building and deploying applications, as well as simplifying the process of scaling and managing them. Docker has become popular in recent years due to its flexibility, ease of use, and ability to streamline the application development and deployment process.

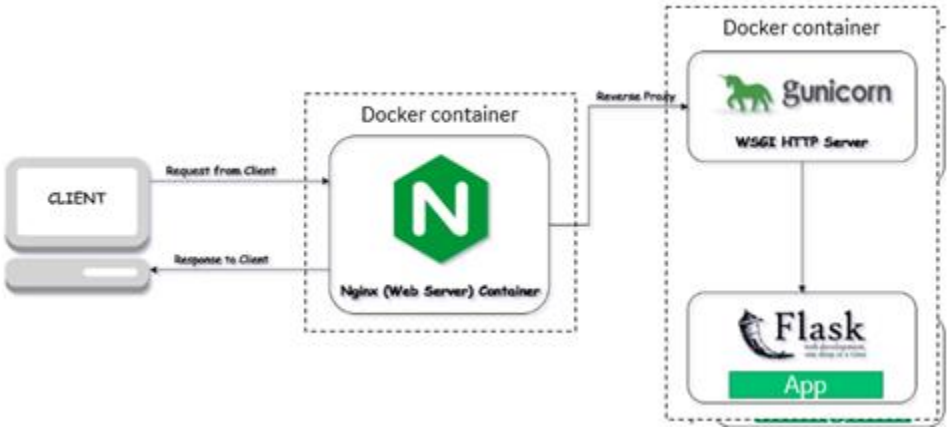


Figure 11.1 Docker flowchart

- When using Flask, Gunicorn, and Docker together, developers can build and deploy containerized web applications that are scalable, portable, and easy to manage.
- The combination of Flask, Gunicorn, and Docker provides a powerful toolset for building and deploying web applications that are both scalable and portable. By containerizing applications, developers can create a consistent environment for running the application, reducing the risk of runtime errors and making it easier to manage dependencies.
- Additionally, by using a production-grade HTTP server like Gunicorn, Flask applications can handle a high volume of requests while remaining stable and performant.

11.2 NGINX WEBSERVER

- Nginx (pronounced "engine x") is a high-performance open-source web server software that can also function as a reverse proxy, load balancer, and HTTP cache. It was developed to address some of the limitations of the traditional Apache web server and is now widely used as a server for websites, web applications, and APIs.
- Nginx is designed to handle high traffic volumes with minimal resource consumption. It can serve static and dynamic content, as well as handle SSL/TLS encryption and compression. Its modular architecture allows for easy customization and extension, with a variety of third-party modules available to add features such as authentication, caching, and security.
- As a reverse proxy, Nginx can be used to distribute incoming traffic across multiple servers, improving application availability and scalability. It can also cache frequently accessed content, reducing server load and improving performance. In addition, Nginx can be configured to serve as a load balancer, distributing traffic among multiple backend servers.

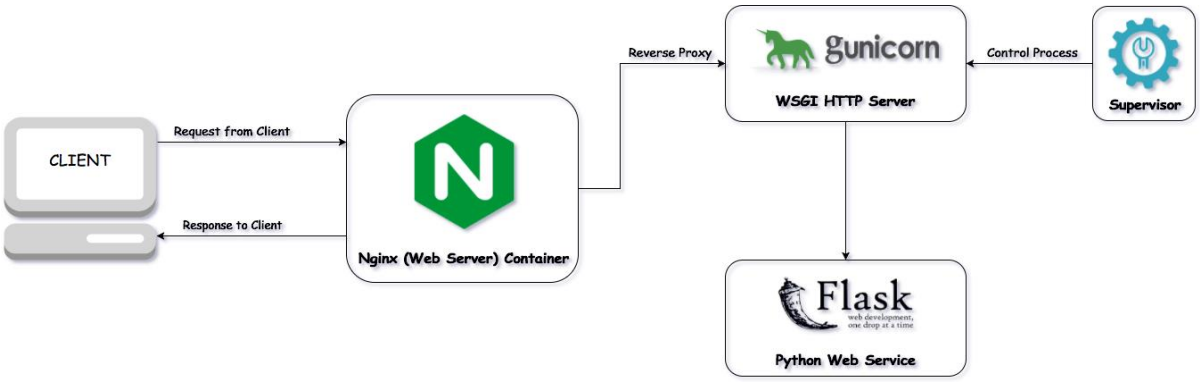


Figure 11.2 Nginx

- Nginx, Gunicorn, and Flask are commonly used together in web application deployments to provide a scalable, reliable, and high-performance architecture. Here's an overview of how they work together:
- Flask is used to build the web application or API using Python.
- Gunicorn is used to serve the Flask application, acting as a WSGI HTTP server that can handle incoming requests and responses.
- Nginx is used as a reverse proxy server that sits in front of Gunicorn and handles incoming requests, load-balancing them across multiple Gunicorn workers if necessary.
- In this setup, Nginx is responsible for handling incoming requests from clients, including static files and requests for dynamic content that are passed to Gunicorn.
- Nginx can be configured to cache frequently accessed content and handle SSL/TLS encryption, providing an additional layer of security.

- Gunicorn, in turn, is responsible for serving the Flask application, handling incoming requests, and generating responses. It can be configured to use multiple worker processes or threads, allowing it to handle a high volume of requests and improve performance.

11.3 NGINX-DOCKER WEBSITE

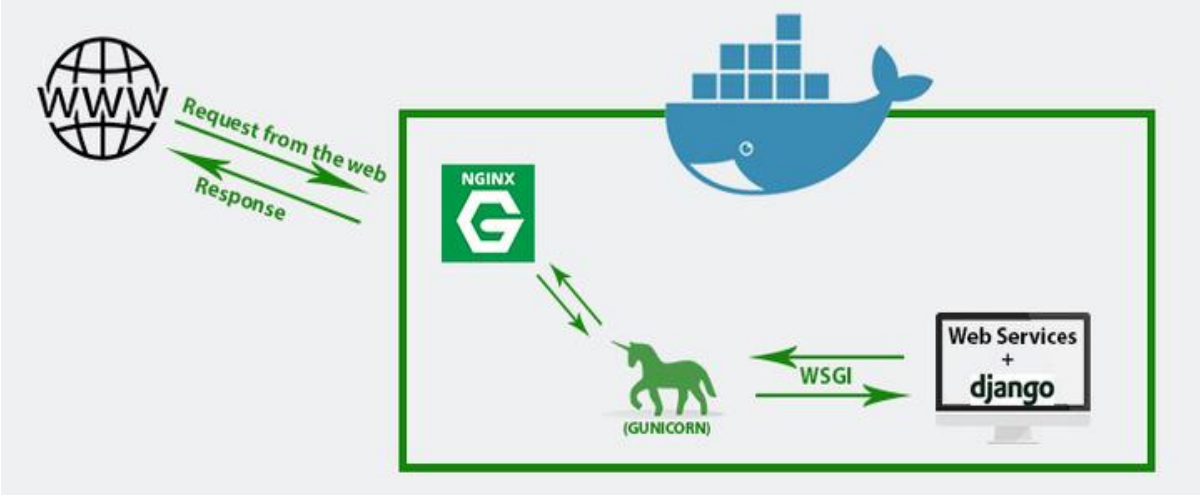


Figure 11.3 Flask Nginx in Docker

- Nginx and Gunicorn are commonly used together to deploy Python web applications in production environments. Docker can also be used to containerize these components, allowing for easier deployment and scalability.
- Tools we Need:
  - FLASK App
  - Gunicorn
  - Nginx
  - Docker

File stucture

```
myapp/
├── app/
│   ├── __init__.py
│   ├── models.py
│   ├── routes.py
│   └── templates/
│       └── index.html
├── Dockerfile
├── requirements.txt
├── docker-compose.yml
└── nginx.conf
```

Figure 11.3 Docker file Structure

⇨ Create Simple Python Code for the test :

```python
# file name : wsgi.py
    from flask import Flask
    import socket
    app = Flask(__name__)

    @app.route('/')
    def hello():
        return f" Welcome Here HOST ID :{socket.gethostbyname()}"

    if __name__ == '__main__':
        app.run(debug=True)
```

Figure 11.3 Docker flask App.py

- ⇨ Create a Dockerfile for your application. This file should include instructions for installing dependencies, copying your code into the container, and starting the Gunicorn server.
- ⇨ Create A Docker file for app.py to run in Linux to auto-run the commands:

```
#  Docker file
    FROM python:3.11.3

    WORKDIR /app

    COPY . .

    RUN pip install flask gunicorn
    CMD gunicorn --bind 0.0.0.0:5000 wsgi:app
```

Figure 11.3 Docker Config File

- ⇨ Create a docker-compose.yml for managing all containers
- ⇨ The resulting output is run at port no localhost:80

```
    version: "3"

services:
app:
    build:
    context : app
    ports:
    - "5000"

nginx:
    image: nginx:latest
    volumes:
    - ./nginx.conf:/etc/nginx/nginx.conf:ro

    depends_on:
    - app

    ports:
    - "80:80"
```

Figure 11.3 Docker-Compose.yml

- ⇨ Create Nginx configuration file for handlining servers and proxy
- ⇨ This should start the Gunicorn server and Nginx proxy in their respective containers, and allow you to access your Python web application at http://localhost.

```
    events {
    worker_connections  1024;
}

http {
    server {
        listen       80;

        location / {
            proxy_pass http://app:5000;
        }

    }
}
```

Figure 11.3 Docker Nginx. conf

```
[+] Running 5/54T22:24:33+05:30" level=war
 ✓ Network docker_default     Created
 ✓ Container docker-app-3     Started
 ✓ Container docker-app-1     Started
 ✓ Container docker-app-2     Started
 ✓ Container docker-nginx-1   Started
PS C:\Users\jaypa\Desktop\Docker>
```

Figure 11.3 Docker Run

- ⇨ **Load Balancing**
- ⇨ Every time every new request are switching servers

Welcome Here HOST ID :cc2ab9f39a1e

Welcome Here HOST ID :d872cc4918c0

Figure 11.3 Docker Outputs

REFERENCE

YouTube:

- **Code with Harry:** CodeWithHarry - YouTube
- **Dennis Ivy:** Dennis Ivy - YouTube
- **Digital Daru**: Digital Daru - YouTube
- **GeeksForGeeks:** GeeksforGeeks | A computer science portal for geeks

Documentation:

- **Flask:** Welcome to Flask — Flask Documentation (2.3.x) (palletsprojects.com)
- **Django:** Django documentation | Django documentation | Django (djangoproject.com)
- **SK Learn:** scikit-learn: machine learning in Python — scikit-learn 1.2.2 documentation
- **W3Shool:** W3Schools Online Web Tutorials
- **Tutorial Republic:** Tutorial Republic - Online Web Development Tutorials

My GitHub:

- **https://github.com/jay19patel**