A710145025001

PYTHON LAB ASSIGNMENT

ENROLLMENT NO-A710145025001

Q1. Accepts three numbers from the user.

```
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
num3 = float(input("Enter third number: "))

if num1 >= num2 and num1 >= num3:
    largest = num1
elif num2 >= num1 and num2 >= num3:
    largest = num2
else:
    largest = num3


print(f"The largest number is: {largest}")
```

OUTPUT:

```
PS C:\Users\Harsh Parmar\PYTHON LAB> & "C:\Users\Harsh Parmar\AppData\Local\
/Users/Harsh Parmar/PYTHON LAB/accepts3numbers.py"
Enter first number: 10
Enter second number: 20
Enter third number: 22
The largest number is: 22.0
PS C:\Users\Harsh Parmar\PYTHON LAB>
```

Q2. Defines a list of fruits.

```
fruits = ["apple", "banana", "cherry", "date", "elderberry"]
fruit_lengths = {}

for fruit in fruits:
```

```
    length = len(fruit)

    length_as_string = str(length)

    fruit_lengths[fruit] = length_as_string

print(fruit_lengths)
```

OUTPUT:

```
PS C:\Users\Harsh Parmar\PYTHON LAB> & "C:\Users\Harsh Parmar\AppData\Local\Programs\
/Users/Harsh Parmar/PYTHON LAB/listoffruits.py"
{'apple': '5', 'banana': '6', 'cherry': '6', 'date': '4', 'elderberry': '10'}
PS C:\Users\Harsh Parmar\PYTHON LAB>
```

Q3. Matplotlib library to plot a simple Bar chart.

```
import matplotlib.pyplot as plt
months = ['May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct']

sales = [150, 180, 220, 190, 250, 210]

plt.figure(figsize=(8, 5))
plt.bar(months, sales, color='skyblue')

plt.xlabel('Month')
plt.ylabel('Sales (in thousands)')
plt.title('Monthly Sales Figures for the Last Six Months')

plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.xticks(rotation=0)

plt.show()
```

OUTPUT:

Q4. That takes list of integers.

Take a list of integers as input

```
# Take a list of integers as input
numbers = list(map(int, input("Enter integers separated by spaces:
").split()))

print("List of integers:", numbers)
```

OUTPUT:

```
PS C:\Users\Harsh Parmar\PYTHON LAB> python -u "c:\Users\Harsh Parmar\PYTHON LAB\ListOfIntegers"
Enter integers separated by spaces: 10 33 22 22 33 44
List of integers: [10, 33, 22, 22, 33, 44]
PS C:\Users\Harsh Parmar\PYTHON LAB>
```

Q5. Design a Python class named Book with attributes like title, author, and price. Implement a constructor, a method to display book details, and demonstrate inheritance by creating a subclass Ebook that adds an attribute file_size.

```python
class Book:
    def __init__(self, title, author, price):
        self.title = title
        self.author = author
        self.price = price

    def display_details(self):
        print(f"Title: {self.title}")
        print(f"Author: {self.author}")
        print(f"Price: ₹{self.price}")


# Define the subclass that inherits from Book
class Ebook(Book):
    def __init__(self, title, author, price, file_size):
        # Call the constructor of the parent class
        super().__init__(title, author, price)
        self.file_size = file_size

    # Override display_details to include file size
    def display_details(self):
        super().display_details()
        print(f"File Size: {self.file_size} MB")


# Demonstration
if __name__ == "__main__":
    # Create a Book object
    book1 = Book("The Alchemist", "Paulo Coelho", 499)
    print("▧ Book Details:")
    book1.display_details()

    print("\n💻 Ebook Details:")
    # Create an Ebook object
    ebook1 = Ebook("Atomic Habits", "James Clear", 299, 2.5)
    ebook1.display_details()
```

OUTPUT:

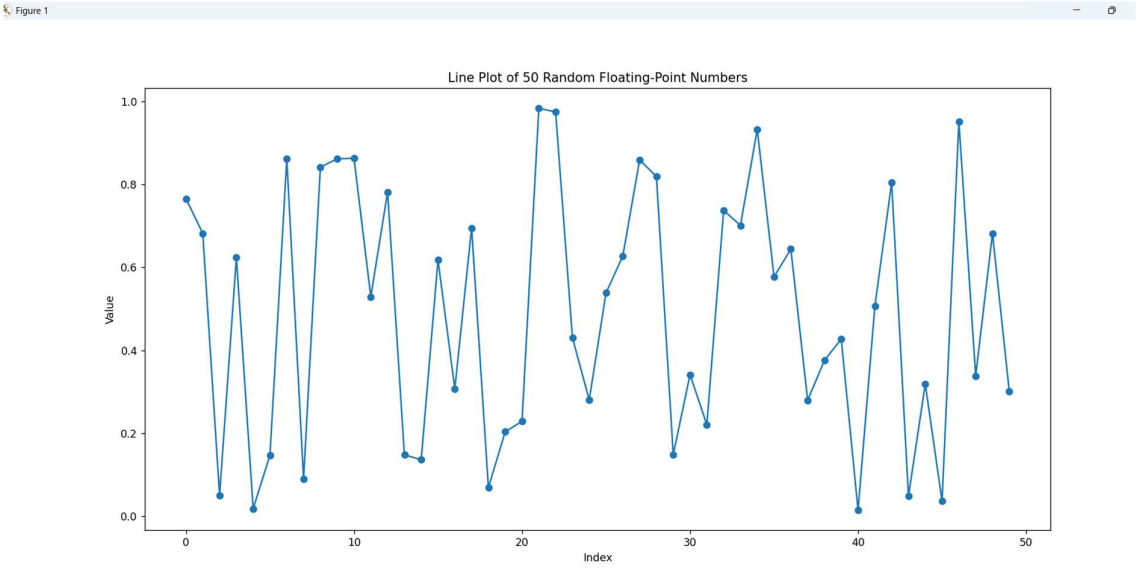
```
PS C:\Users\Harsh Parmar\PYTHON LAB> & "C:\Users\Harsh Parmar\AppData\Local
:/Users/Harsh Parmar/PYTHON LAB/Book.py"
📘 Book Details:
Title: The Alchemist
Author: Paulo Coelho
Price: ₹499

💻 Ebook Details:
Title: Atomic Habits
Author: James Clear
Price: ₹299
File Size: 2.5 MB
PS C:\Users\Harsh Parmar\PYTHON LAB>
```

Q6. Use the NumPy library to create an array of 50 random floating-point numbers. Use the Matplotlib library to generate a simple line plot of these numbers.

```python
import numpy as np
import matplotlib.pyplot as plt

# Create an array of 50 random floating-point numbers between 0 and 1
data = np.random.random(50)

# Display the generated data (optional)
print("Generated Random Numbers:\n", data)

# Create a simple line plot
plt.plot(data, marker='o')

# Add labels and title
plt.title("Line Plot of 50 Random Floating-Point Numbers")
plt.xlabel("Index")
plt.ylabel("Value")

# Show the plot
plt.show()
```

OUTPUT:

```
PS C:\Users\Harsh Parmar\PYTHON LAB> & "C:\Users\Harsh Parmar\AppData\Local\Progr
:/Users/Harsh Parmar/PYTHON LAB/NumPy.py"
Generated Random Numbers:
 [0.76544657 0.68195041 0.05062028 0.62338045 0.0183207  0.1472736
 0.8618839  0.08938318 0.8416916  0.8618071  0.86310249 0.52932391
 0.78158744 0.14815493 0.13623057 0.61814912 0.30735733 0.69462865
 0.06868198 0.20389568 0.2287779  0.9835114  0.97477219 0.43063079
 0.280369   0.53838733 0.62751883 0.85921181 0.81903511 0.14844957
 0.34143519 0.22054557 0.73722903 0.70111662 0.93245192 0.5774432
 0.64403769 0.27938368 0.37586128 0.4270258  0.0144372  0.50635661
 0.80475546 0.04875582 0.31898536 0.03633794 0.95201327 0.33822225
 0.68159494 0.30091828]
```



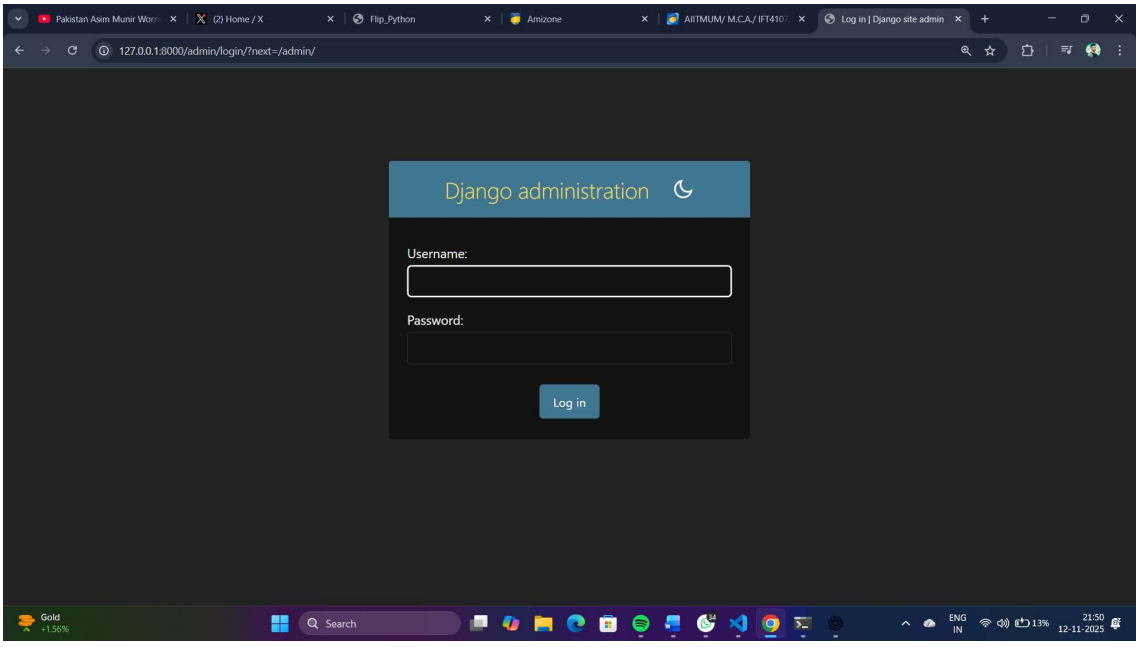
Q7. Set up a new Django project named UniversityPortal. Create an application within it named Students using manage.py. Register the Students application and verify that you can access the Django admin interface.

A710145025001

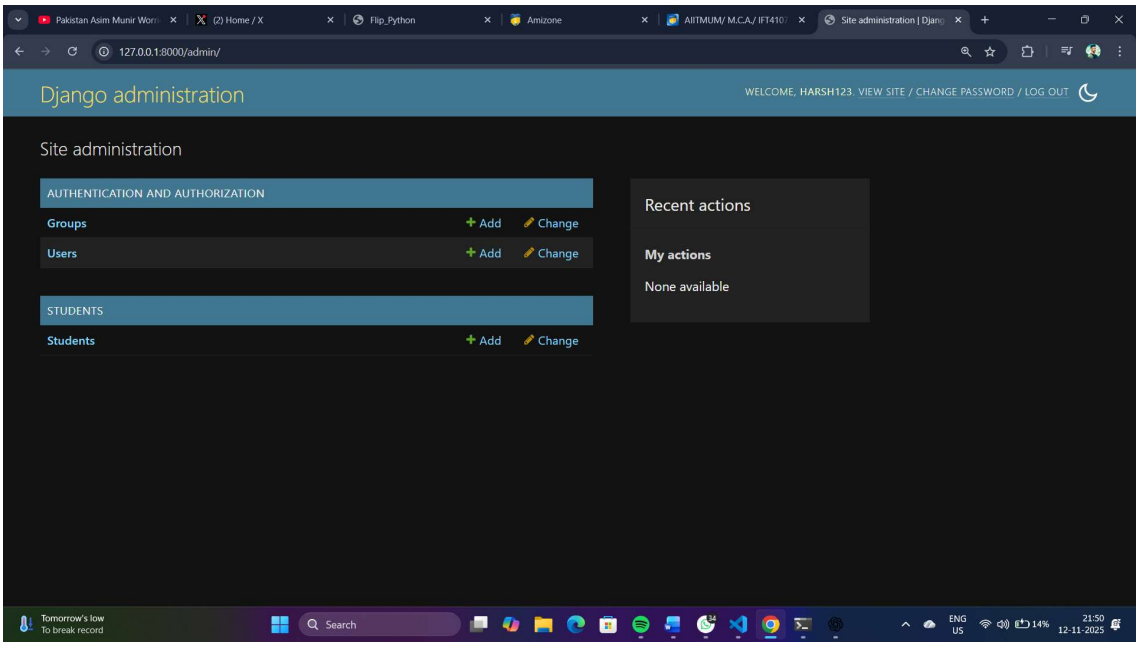
```
C:\Users\Harsh Parmar>django-admin startproject UniversityPortal
Document was last saved: Just now
C:\Users\Harsh Parmar>cd UniversityPortal

C:\Users\Harsh Parmar\UniversityPortal>python manage.py startapp Students

C:\Users\Harsh Parmar\UniversityPortal>
```

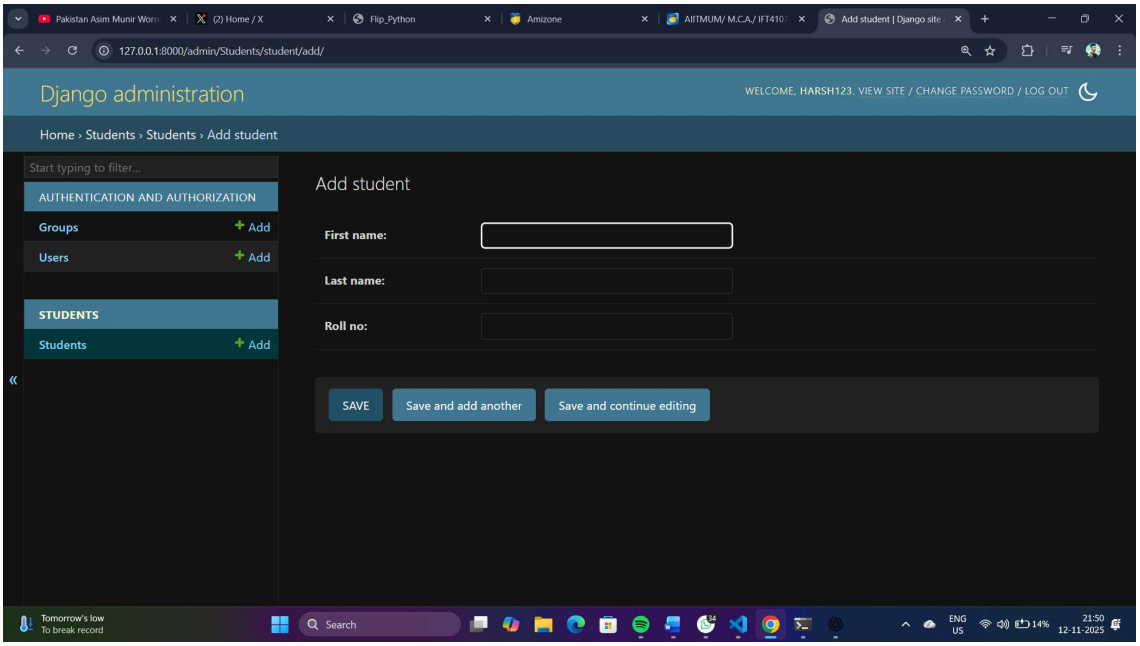
```python
# Students/models.py
from django.db import models

class Student(models.Model):
    first_name = models.CharField(max_length=50)
    last_name  = models.CharField(max_length=50)
    roll_no    = models.CharField(max_length=20, unique=True)
    enrolled_on = models.DateField(auto_now_add=True)

    def __str__(self):
        return f"{self.first_name} {self.last_name} ({self.roll_no})"
```

OUTPUT



A710145025001





Q8. Write a Python function that calculates the factorial of a given non-negative integer. Use recursion to implement this function.

```
def factorial(n):
```

```
    """
    Recursive function to calculate factorial of a non-negative integer n.

    Formula:
    factorial(n) = n * factorial(n - 1)
    Base case: factorial(0) = 1
    """

    if n < 0:
        raise ValueError("Factorial is not defined for negative numbers.")

    if n == 0 or n == 1:
        return 1

    return n * factorial(n - 1)


num = int(input("Enter a non-negative integer: "))
print(f"The factorial of {num} is: {factorial(num)}")
```

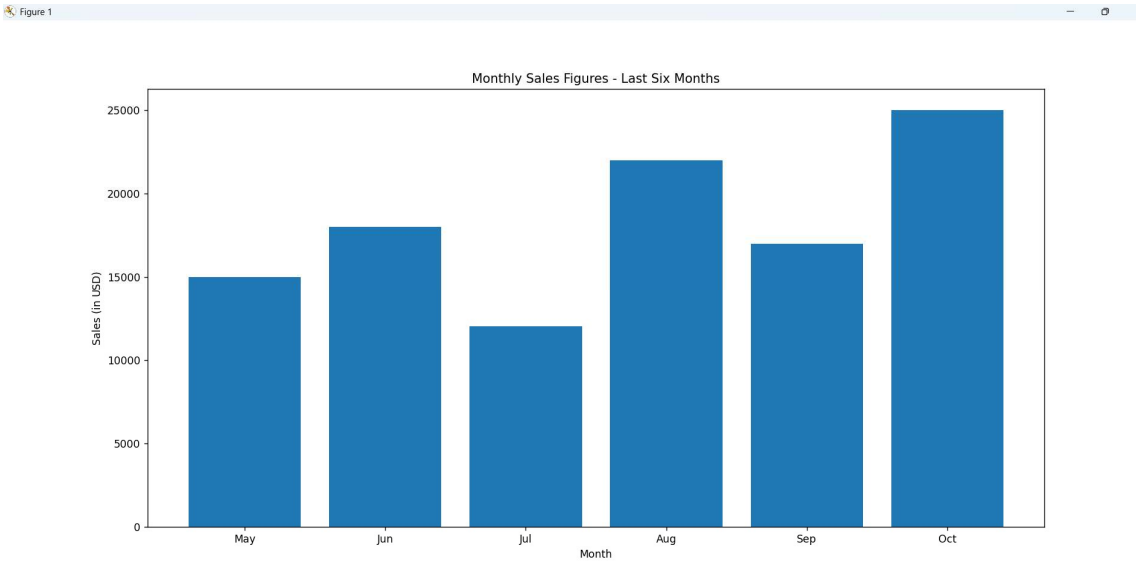
OUTPUT:

```
PS C:\Users\Harsh Parmar\PYTHON LAB> & "C:\Users\Harsh Parmar\AppData\Local\Prog
 LAB/Factorial.py"
Enter a non-negative integer: 10
The factorial of 10 is: 3628800
PS C:\Users\Harsh Parmar\PYTHON LAB>
```

Q9. Use the Matplotlib library to plot a simple Bar Chart showing the monthly sales figures (dummy data) for the last six months. Ensure the chart has labeled axes and a title.

```python
import matplotlib.pyplot as plt

# Dummy data for the last six months
months = ['May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct']
sales = [15000, 18000, 12000, 22000, 17000, 25000]

# Create a bar chart
plt.bar(months, sales)

# Add labels and title
plt.xlabel('Month')
plt.ylabel('Sales (in USD)')
plt.title('Monthly Sales Figures - Last Six Months')

# Display the chart
plt.show()
```

OUTPUT:



Q10. Write a Python program using BeautifulSoup to read a simple HTML file (provide the HTML string in the code) and extract the text content of all <a> (anchor) tags present in the file.

```python
from bs4 import BeautifulSoup
```

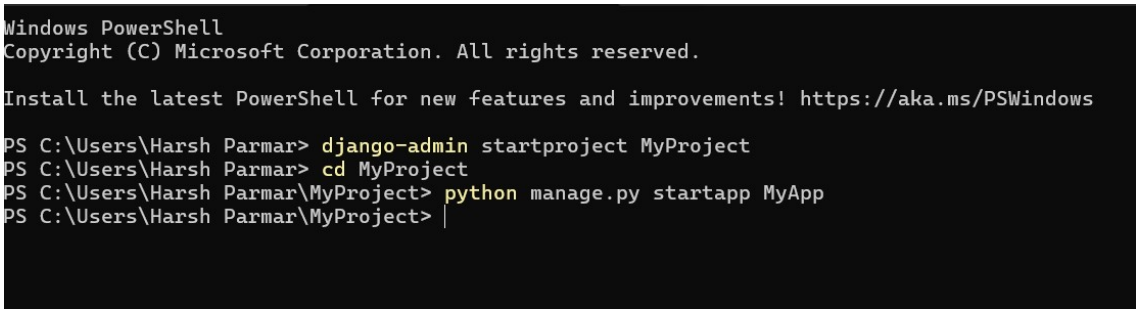
```
# Sample HTML string
html_content = """
<html>
<head><title>Sample Web Page</title></head>
<body>
    <h1>Welcome to My Website</h1>
    <p>Here are some useful links:</p>
    <a href="https://www.google.com">Google</a><br>
    <a href="https://www.github.com">GitHub</a><br>
    <a href="https://www.wikipedia.org">Wikipedia</a>
</body>
</html>
"""

# Parse the HTML content
soup = BeautifulSoup(html_content, 'html.parser')

# Extract all <a> tags
anchor_tags = soup.find_all('a')

# Display text content of all <a> tags
print("Text content of all <a> tags:")
for tag in anchor_tags:
    print(tag.text)
```

OUTPUT:

```
PS C:\Users\Harsh Parmar\PYTHON LAB> & "C:\Users\Harsh Parmar\AppData\Local\P
 LAB/BeautifulSoup.py"
PS C:\Users\Harsh Parmar\PYTHON LAB> & "C:\Users\Harsh Parmar\AppData\Local\P
 LAB/BeautifulSoup.py"
Text content of all <a> tags:
Google
GitHub
Wikipedia
PS C:\Users\Harsh Parmar\PYTHON LAB>
```

Q11. Demonstrate the steps to create a new Django project named MyProject and an application within it named MyApp using the manage.py script.

1. Install Django
2. Create a new project named MyProject



3. django-admin startproject MyProject
4. Navigate into the project directory

   cd MyProject
5. Create a new app named MyApp

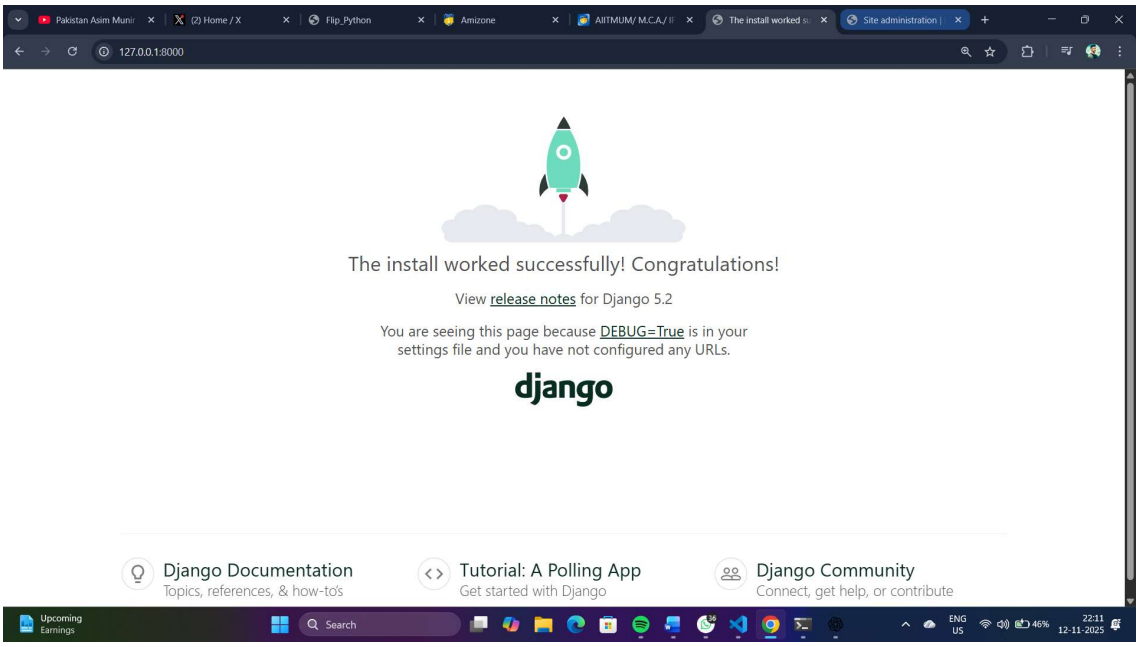
   python manage.py startapp MyApp
6. Register the app
   - Open MyProject/settings.py
   - Add 'MyApp' to the INSTALLED_APPS list.
7. Apply migrations

   python manage.py migrate
8. Run the development server

   python manage.py runserver
9. Access the site
   - Open http://127.0.0.1:8000/ in your browser.

✅ That's it — project MyProject and app MyApp are successfully created!

A710145025001



Q12. Create a Pandas DataFrame from a dictionary containing names, ages, and cities for five employees. Display the first three rows of the DataFrame and then filter and print only the employees aged 30 or above.

```
import pandas as pd

# Create a dictionary with employee data
data = {
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eva'],
    'Age': [25, 30, 35, 28, 40],
    'City': ['New York', 'Los Angeles', 'Chicago', 'Houston', 'Phoenix']
}

# Create a Pandas DataFrame
df = pd.DataFrame(data)

# Display the first three rows of the DataFrame
print("First three rows of the DataFrame:")
print(df.head(3))

# Filter employees aged 30 or above
filtered_df = df[df['Age'] >= 30]

print("\nEmployees aged 30 or above:")
```

```
print(filtered_df)
```

OUTPUT:

```
PS C:\Users\Harsh Parmar\PYTHON LAB> & "C:\Users\Harsh Parmar\AppData\Local\Programs\Python\Pytho
 LAB/Pandas-Dataframe.py"
First three rows of the DataFrame:
      Name  Age         City
0    Alice   25     New York
1      Bob   30  Los Angeles
2  Charlie   35      Chicago

Employees aged 30 or above:
      Name  Age         City
1      Bob   30  Los Angeles
2  Charlie   35      Chicago
4      Eva   40      Phoenix
PS C:\Users\Harsh Parmar\PYTHON LAB>
```