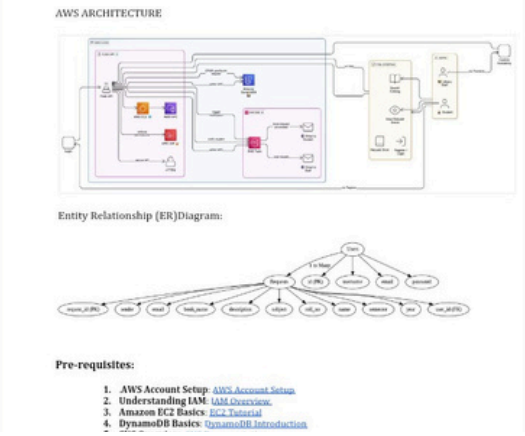
CHANGES IN THE CODES MENTIONED AT THE END OF THE DOCUMENT AND MOVE TO STEP 1

STEP 1 : UPLOAD ALL FILES ON GITHUB

1. Create a repository named capturemoments in our personal github account.
2. Once created, you can see the screen with commands on how to upload the files in that repository.
   Snapshot attached for reference.



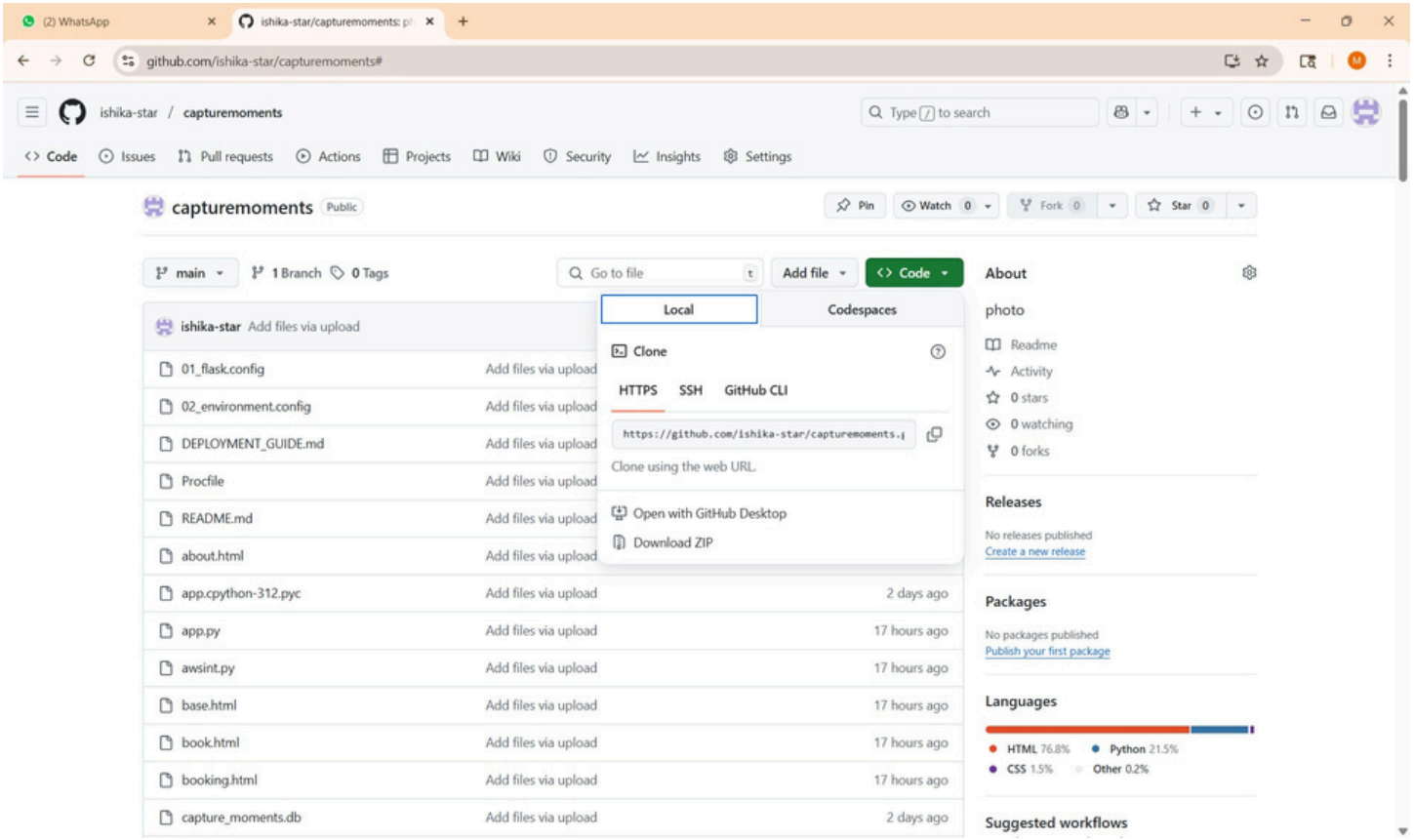


Fig 1

1. Once you see this screen Open you **GitBash tool.**
2. Type cd “Capturemoment file path” this file path should be of your system project Folder.

For example cd "C:\Users\mouni\VS Code\Capture-Moments" make sure it your project file path don’t paste this it wont work. Snapshot attached for reference.

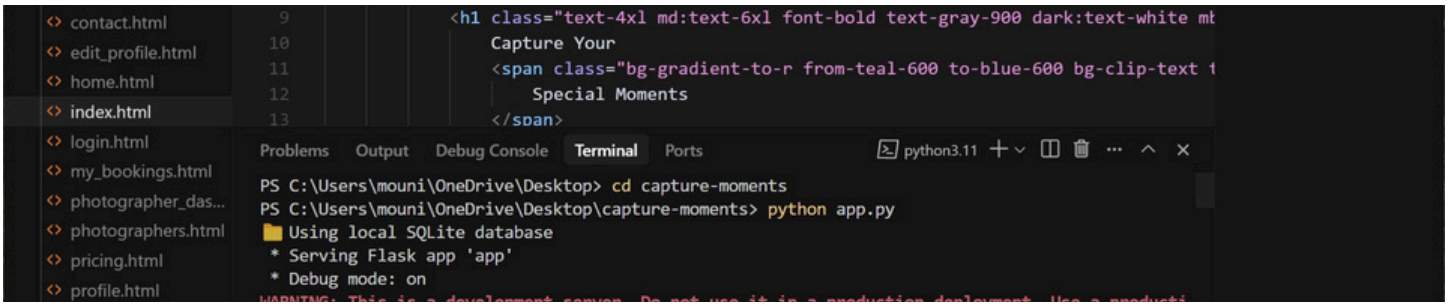


Fig 2

1. Once Inside the Folder. Type the commands line by line as in **Fig 1.**
2. Once Uploaded, reload the page and check if your GitHub Repository has all files loaded.

STEP 2: NOW BEGIN WITH YOUR TROVEN LOGIN AND ACCESS

1. Log in to your Troven and begin with your AWS process.

STEP 3: INITIALISE THE EC2 INSTANCE

Go to AWS Console → Search and open **EC2**.

Click **"Launch Instance"**.

Fill in:

- **Name**: CaptureMoments-Instance
- **AMI**: Choose **Amazon Linux 2**
- **Instance type**: t2.micro (Free Tier eligible)

Under **Key pair (login)**:

- Select **Create new key pair** if you don't have one.
- Download the .pem file and **keep it safe** (needed for login via Git Bash).

Under **Network settings**:

- Click **Edit** → Add a rule:
  - Type: **Custom TCP**
  - Port Range: 5000
  - Source: 0.0.0.0/0 (for public access)
- Also make sure **SSH (22)** is open.

Leave the rest as default → Click **Launch Instance**.

Once it's running, **copy the Public IPv4 address** (you'll need it to open your site later).

STEP 4: CREATE THE IAM ROLE

Go to AWS Console → Search **IAM** → Go to **Roles**.

Click **Create Role**.

Select:

- **Trusted entity type**: AWS service
- **Use case**: EC2 → Click **Next**

Attach permission:

- Search for and check ✅ AmazonDynamoDBFullAccess

Name it EC2DynamoDBAccessRole and **create**.

Go back to **EC2 Instances** → Select your instance → Click **Actions** → **Security** → **Modify IAM role**.

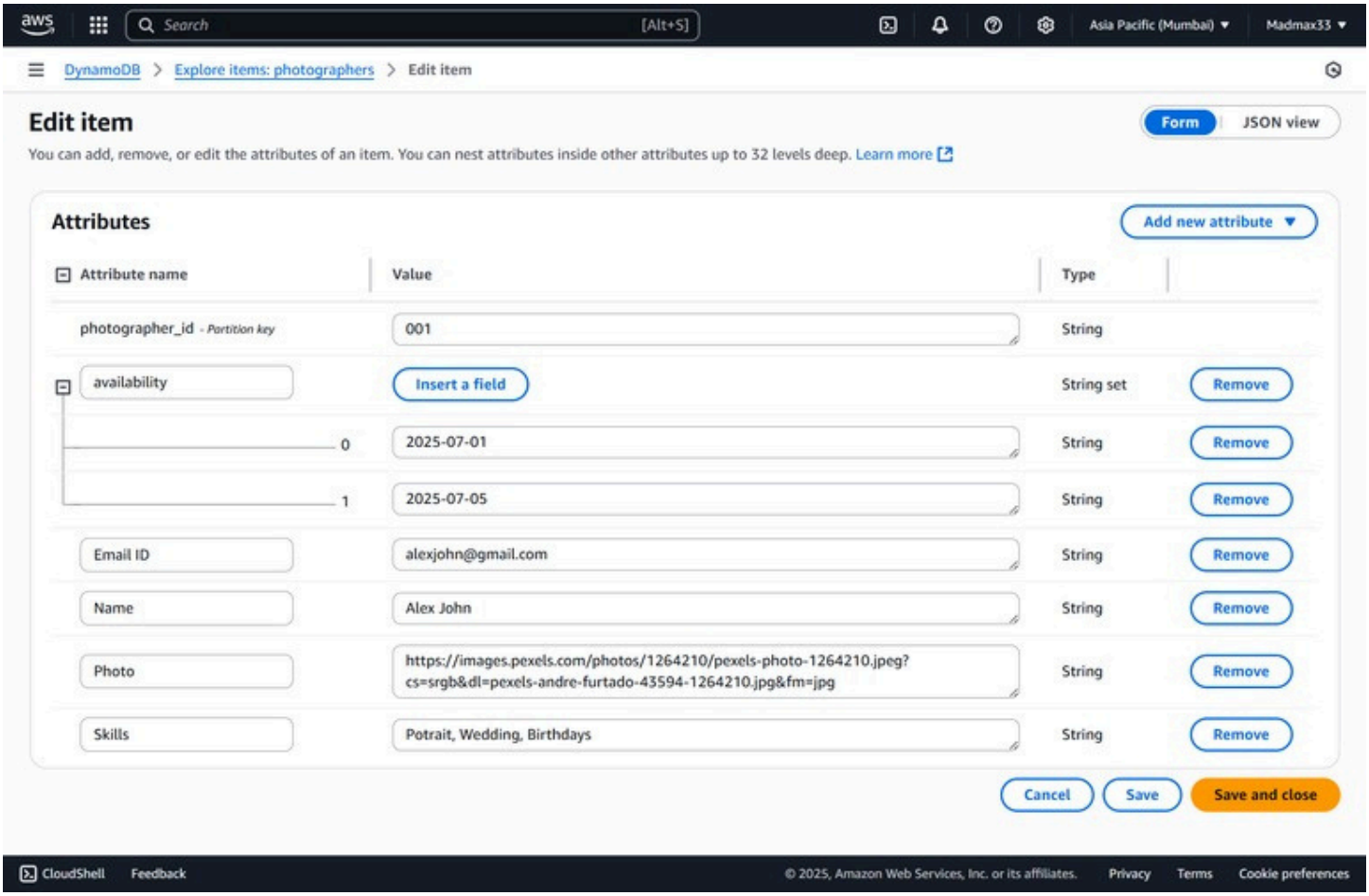
Attach the new role: EC2DynamoDBAccessRole.

STEP 5: CREATE DYNAMODB TABLES

Go to AWS Console → Search **DynamoDB** → Click **Create table**

Create a photographers table:

- Table name: photographers
- Partition key: photographer_id (String)
- Leave defaults → Click **Create**
- **Make sure the attributes are all the same as in the snapshot below.**



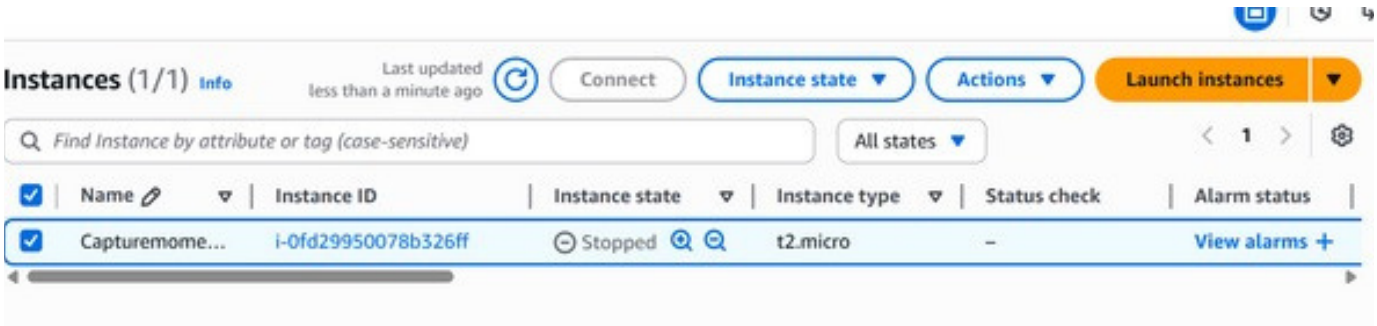
Create Bookings table:

- Table name: booking
- Partition key: booking_id (String)
- Click **Create**

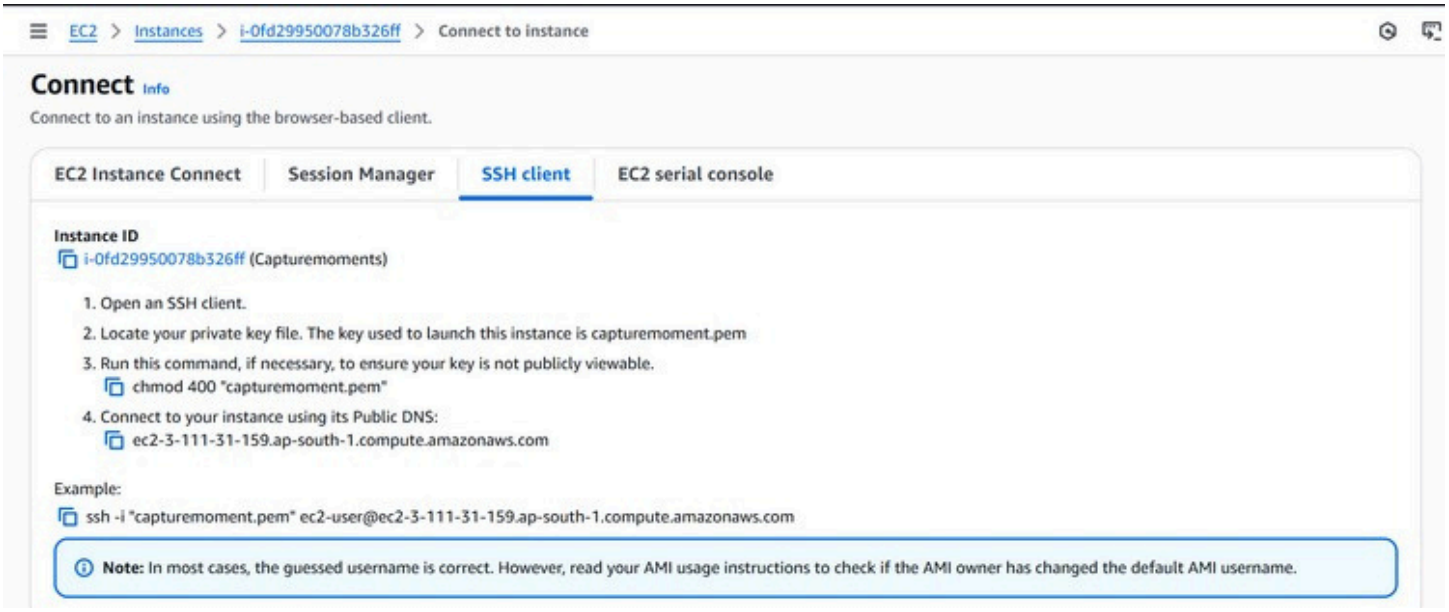
Later, use the **AWS Console** or your Flask app to insert data.

STEP 6: START INSATANCE IN GITBASH AND INSTALL NECESSARY DEPENDENCIES

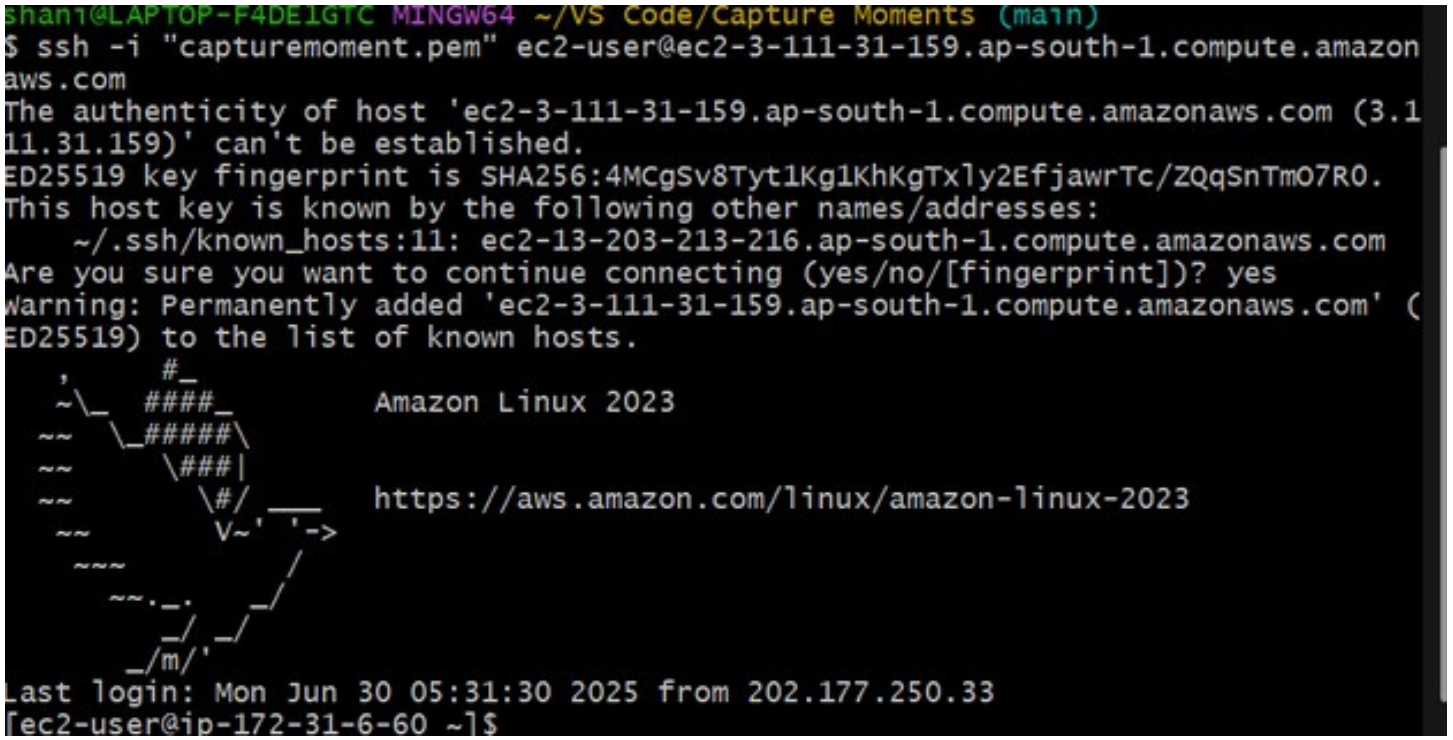
1. Shift your .pem key pair into your Capture Moment project folder.
2. After that to connect to EC2 you type the command as in your EC2 instance page. Snapshot attached for your reference. You choose the Instance and Click on “Connect” button.



This will lead your to the below page copy the command circled in the snapshot



Paste it in your Gitbash and the instance shall start running. It should look like the snapshot attached below.



1. Now install dependencies, so type following commands.
   a. sudo yum update -y
   b. sudo yum install python3 git -y
   c. pip3 install --user flask boto3
   d. git clone https://github.com/<your-username>/<repo-name>.git
   e. cd <repository-name>

STEP 7: RUN THE APP AND VISIT THE WEBSITE

1. Run

python3 awsint.py

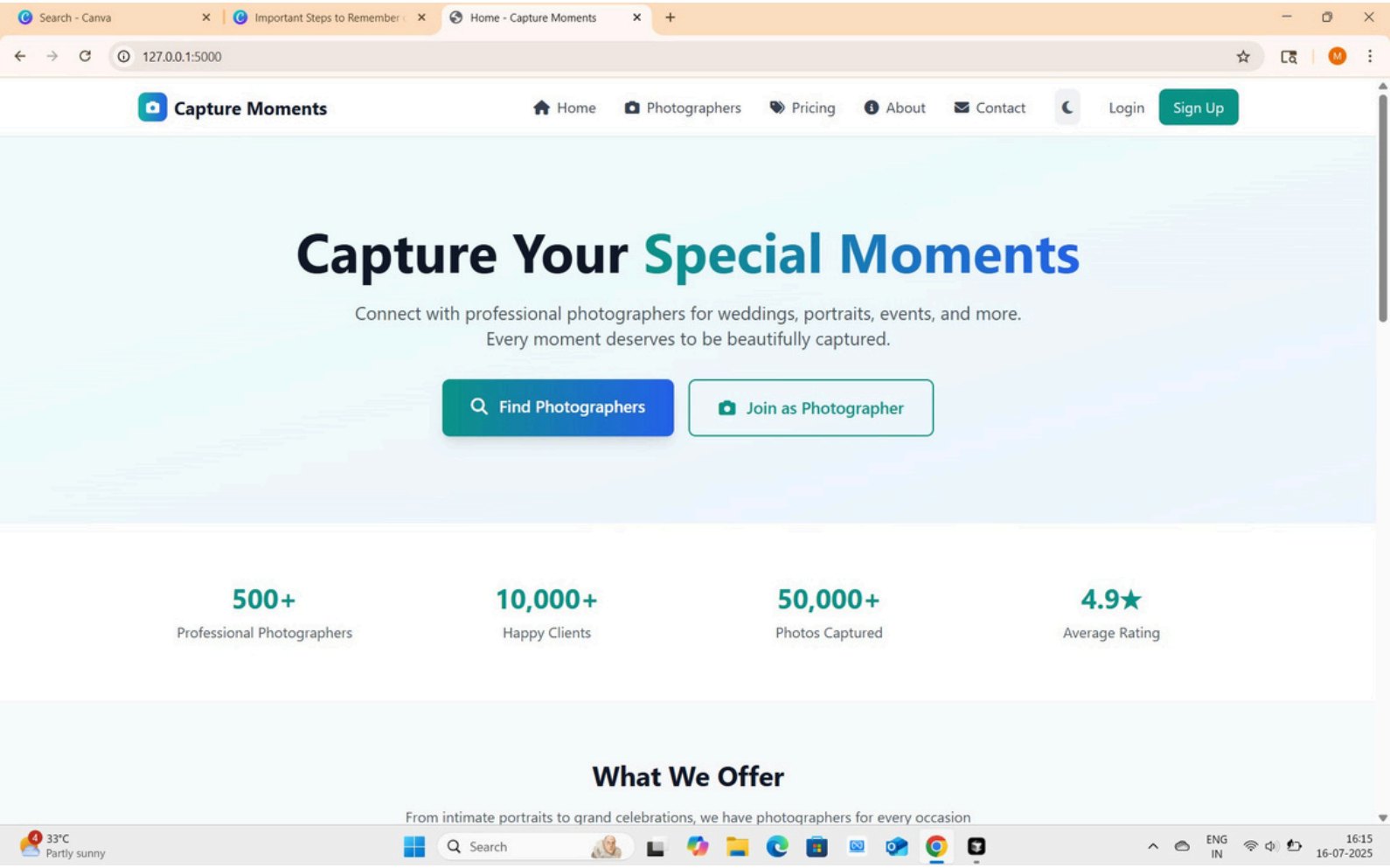
1. Go to:

http://<your-ec2-public-ip>:5000

1. If you make changes in the GitHub code and want to update the Flask app. Stop the currently runny flask app and type the below command

git pull origin main

1. Run step 1 again and check if you are still facing any error.



UPDATED CODES FOR ALL FILES FOR AWS INTEGRATION

awsint.py

```
from flask import Flask, render_template, request, jsonify
import boto3
import uuid
from datetime import datetime

# Step 1: Create the Flask app instance
app = Flask(__name__)

# Step 2: Connect to DynamoDB
dynamodb = boto3.resource('dynamodb', region_name='ap-south-1') # Replace with your region
```

```
# Tables
photographers_table = dynamodb.Table('photographers')
bookings_table = dynamodb.Table('booking')

# Home Page
@app.route('/')
def home():
 return render_template('home.html')

# Booking form route
@app.route('/book', methods=['GET', 'POST'])
def book():
 if request.method == 'POST':
        photographer_id = request.form.get('photographer_id')
        user_id = request.form.get('user_id')
        date = request.form.get('date')

        # Create unique booking ID
        booking_id = str(uuid.uuid4())

        #Store booking in DynamoDB Bookings table
        bookings_table.put_item(Item={
            'booking_id': booking_id,
            'photographer_id': photographer_id,
            'user_id': user_id,
            'date': date,
            'timestamp': datetime.now().isoformat()
        })
```

```
        return f"<h2 style='color:green;'>Booking Confirmed! For {photographer_id} on {date}.</h2><a href='/'>Back to Home</a>"


    return render_template('book.html')


# Display photographers from DynamoDB
@app.route('/show-photographers')
def show_photographers():
    response = photographers_table.scan()
 photographers = response.get('Items', [])

    #✅ FIXED: use correct DynamoDB key - 'photographer_id'
    availability_data = {
        p['photographer_id']: p.get('availability', []) for p in photographers
    }

    return render_template('photographers.html',
                           photographers=photographers,
                           availability_data=availability_data)

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=True)
```

photographers.html

```
<!DOCTYPE html> <html>
<head>
 <title>Photographers</title>
 <style>
```

```
    body {
      background-color: #f7f7f7;
      font-family: Arial;
      text-align: center;
      padding: 50px;
    }
    .card {
      background: white;
      margin: 20px auto;
      padding: 20px;
      width: 300px;
      border-radius: 8px;
      box-shadow: 0px 4px 10px rgba(0,0,0,0.1);
    }
    h3 {
      margin-bottom: 10px;
      color: #2c3e50;
    }
    p {
      color: #555;
    }
  img {
      margin-top: 10px;
      border-radius: 4px;
      width: 200px;
      height: auto;
    }
  </style>
</head>
```

Sign Up - Capture Moments

Home Photographers Pricing About Co

Create your account

Join Capture Moments and start your journey

Username

ch.pravalika

Email Address

cherukuripravalika21@gm

Password

Create a strong password

I want to join as

Client

Photographer

```
<body>
<h2>Available Photographers</h2>

 {% for p in photographers %}
   <div class="card">
     <h3>{{ p['Name'] }}</h3>
     <p><strong>ID:</strong> {{ p['photographer_id'] }}</p>
     <p><strong>Skills:</strong> {{ p['Skills'] }}</p>
     <p><strong>Availability:</strong> {{ p['availability'] | join(', ') }}</p>

     {% if p['Photo'] %}
       <img src="{{ p['Photo'] }}" alt="{{ p['Name'] }}">
     {% else %}
       <p>No image available</p>
     {% endif %}
   </div>
 {% endfor %}

 <a href="/" style="display: inline-block; margin-top: 20px;">← Back to Home</a>
</body>
</html>

 <a href="/" style="display: inline-block; margin-top: 20px;">← Back to Home</a>
</body>
</html>
```

REST ALL OTHER FILES REMAIN SAME.