**AWS Tasks :**

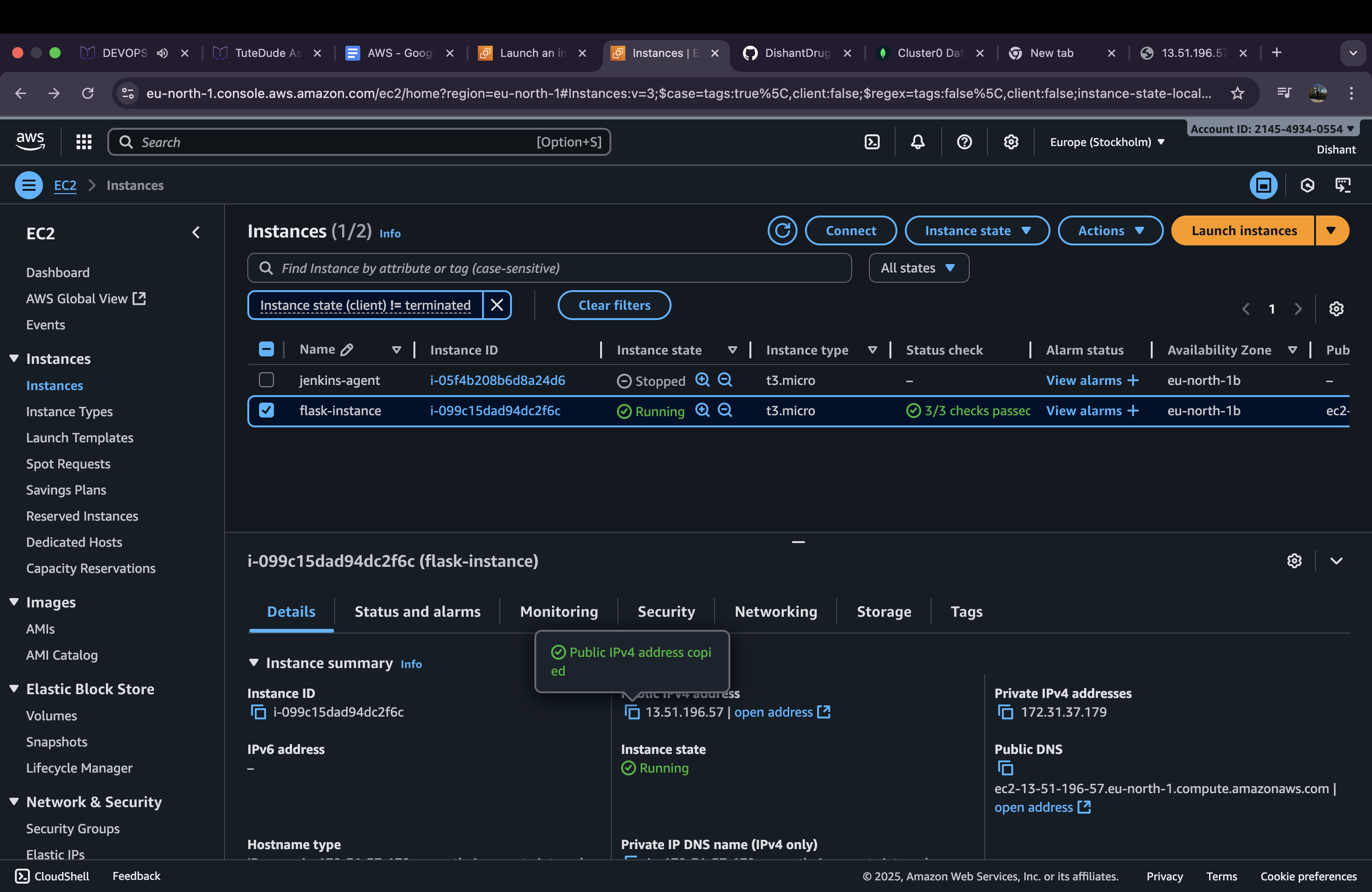
**Task 1: Deploy Flask Backend and Express Frontend on Amazon EC2**

**App URL : <http://13.51.196.57:5000>**

1. **EC2 Instance Setup**

Launched an Ubuntu EC2 instance and configured necessary ports for frontend, backend, and SSH access.

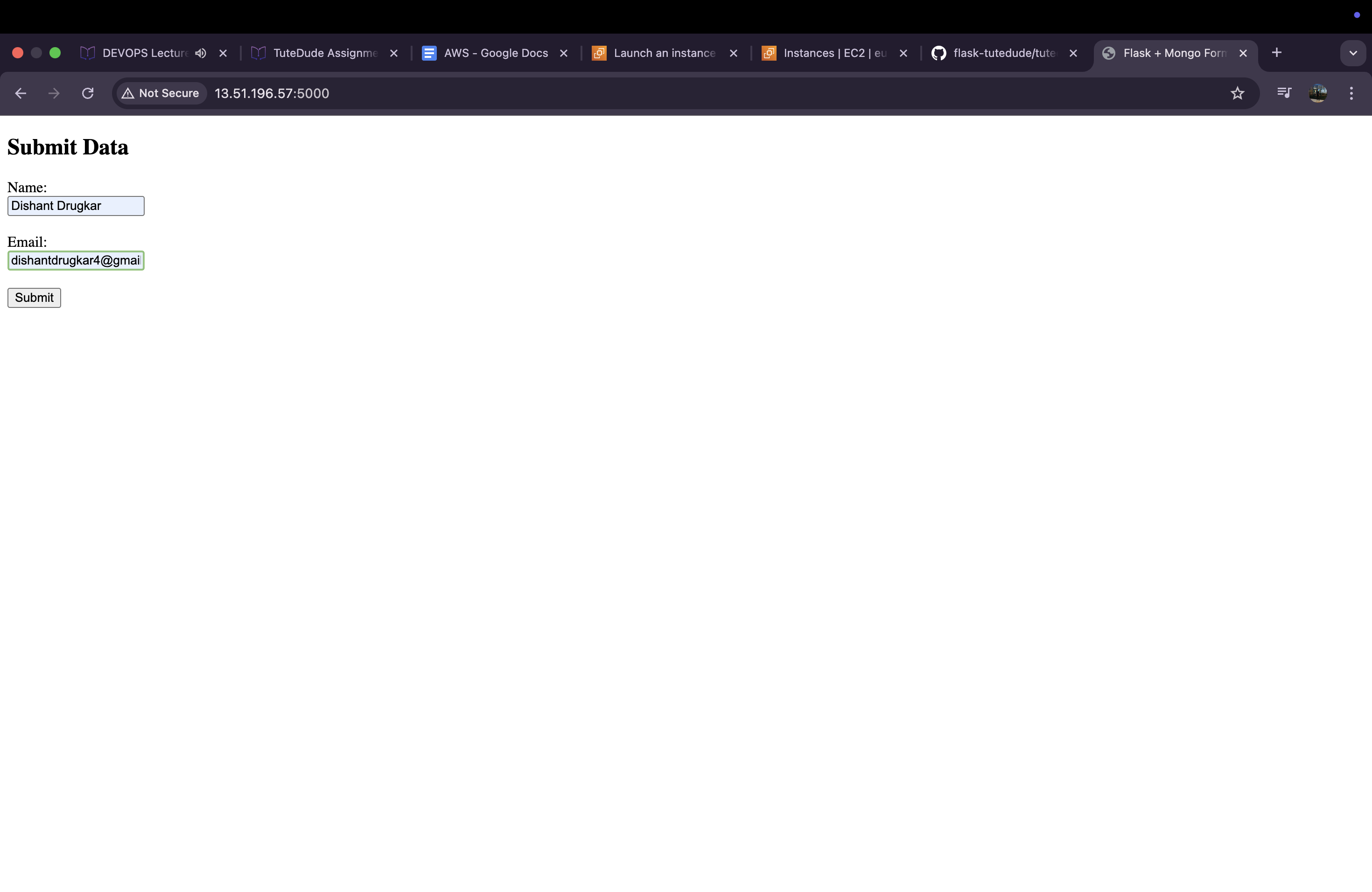
**Screenshot : Flask-instance**

****

**2. Frontend Form :**

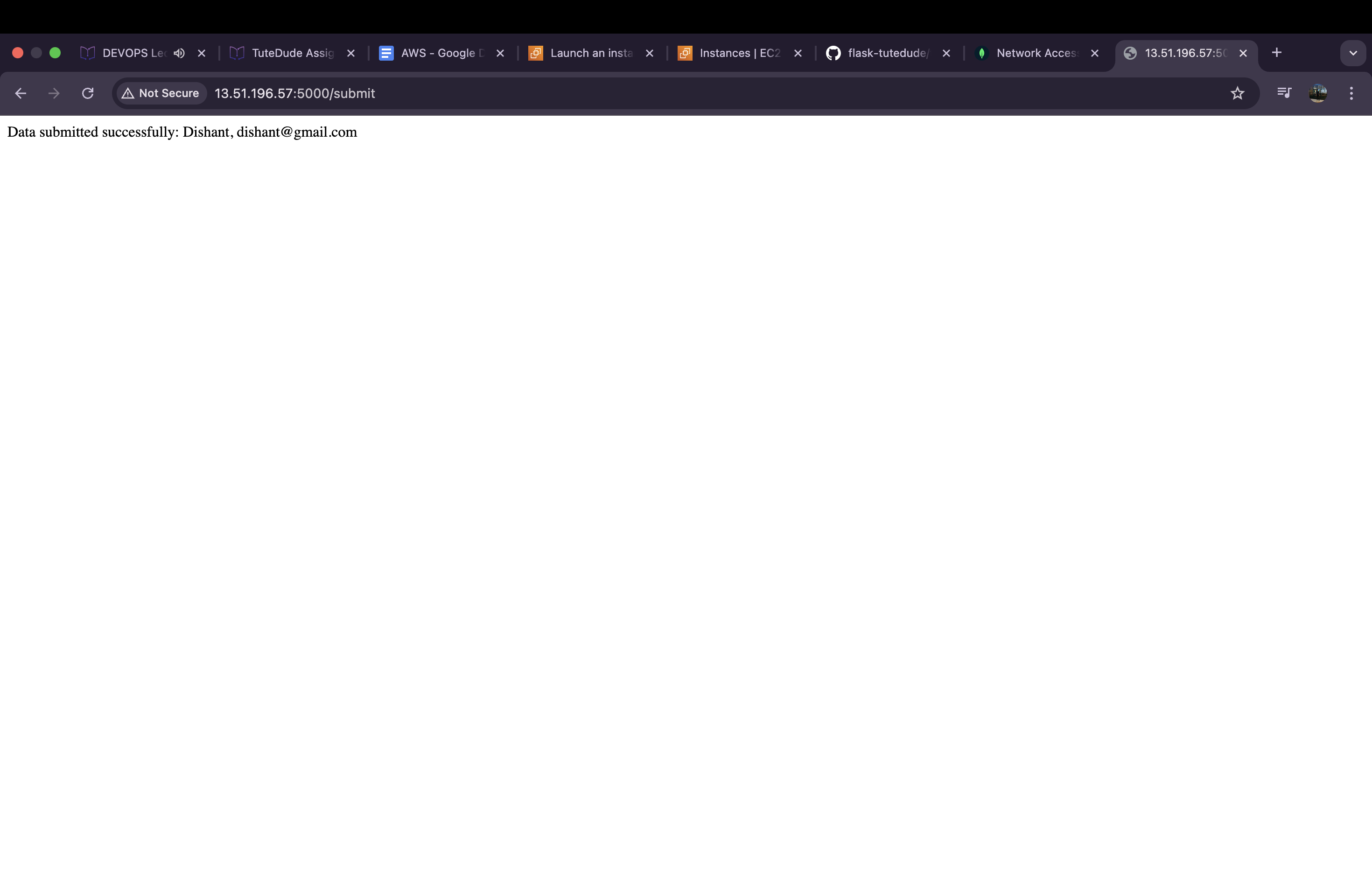
The Express frontend was deployed on the same EC2 instance. The frontend collects user input and communicates with the Flask backend.

Screenshot :



**3. Data Submission**

Submitted data from the frontend form. The backend successfully received and processed the data.

Screenshot :

**4. MongoDB Verification**

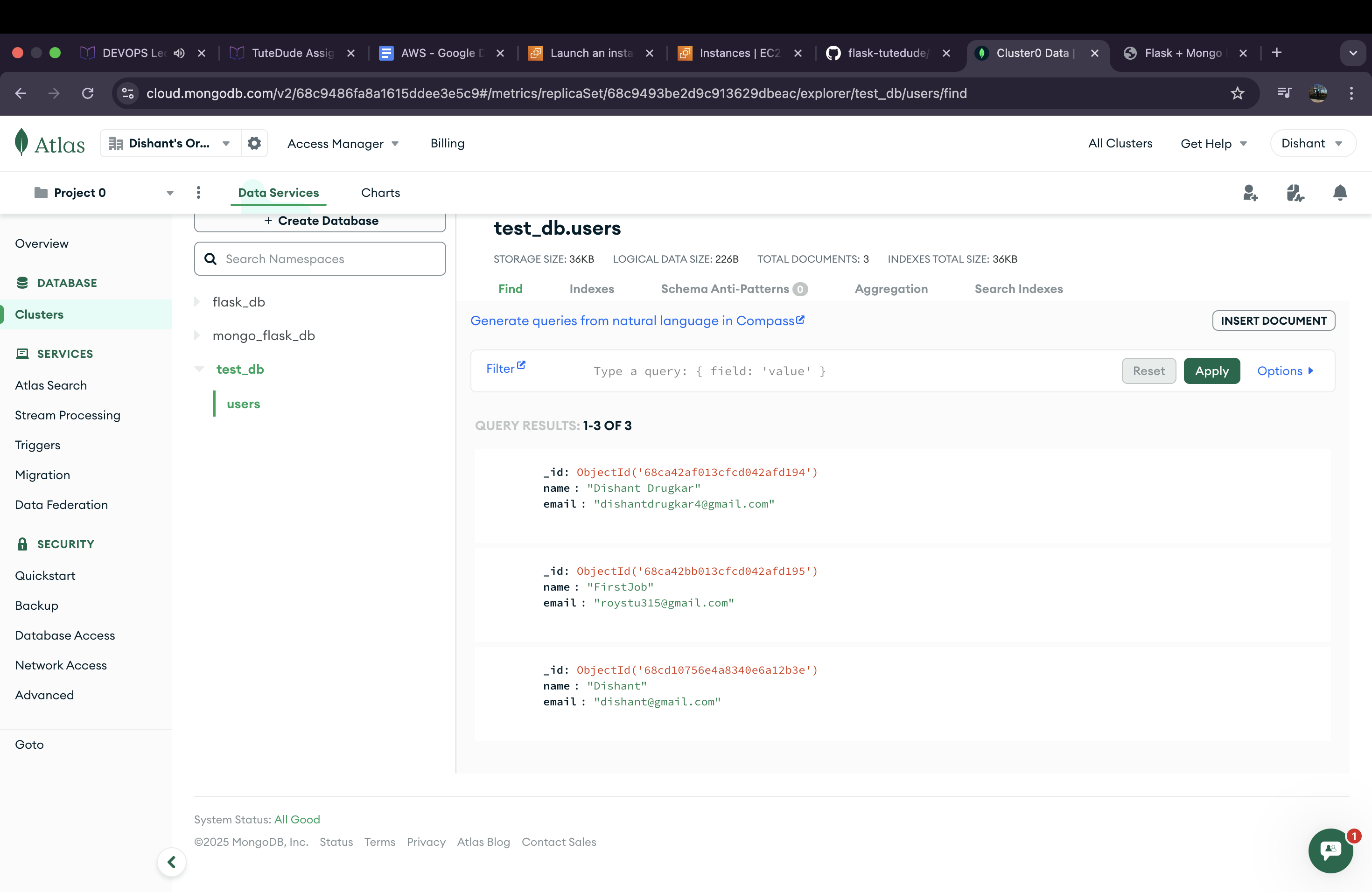
Checked MongoDB to ensure that the submitted data was inserted correctly.

**Screenshot :**

**2. Deploy Your flask backend and express frontend in separate ec2 instances.**

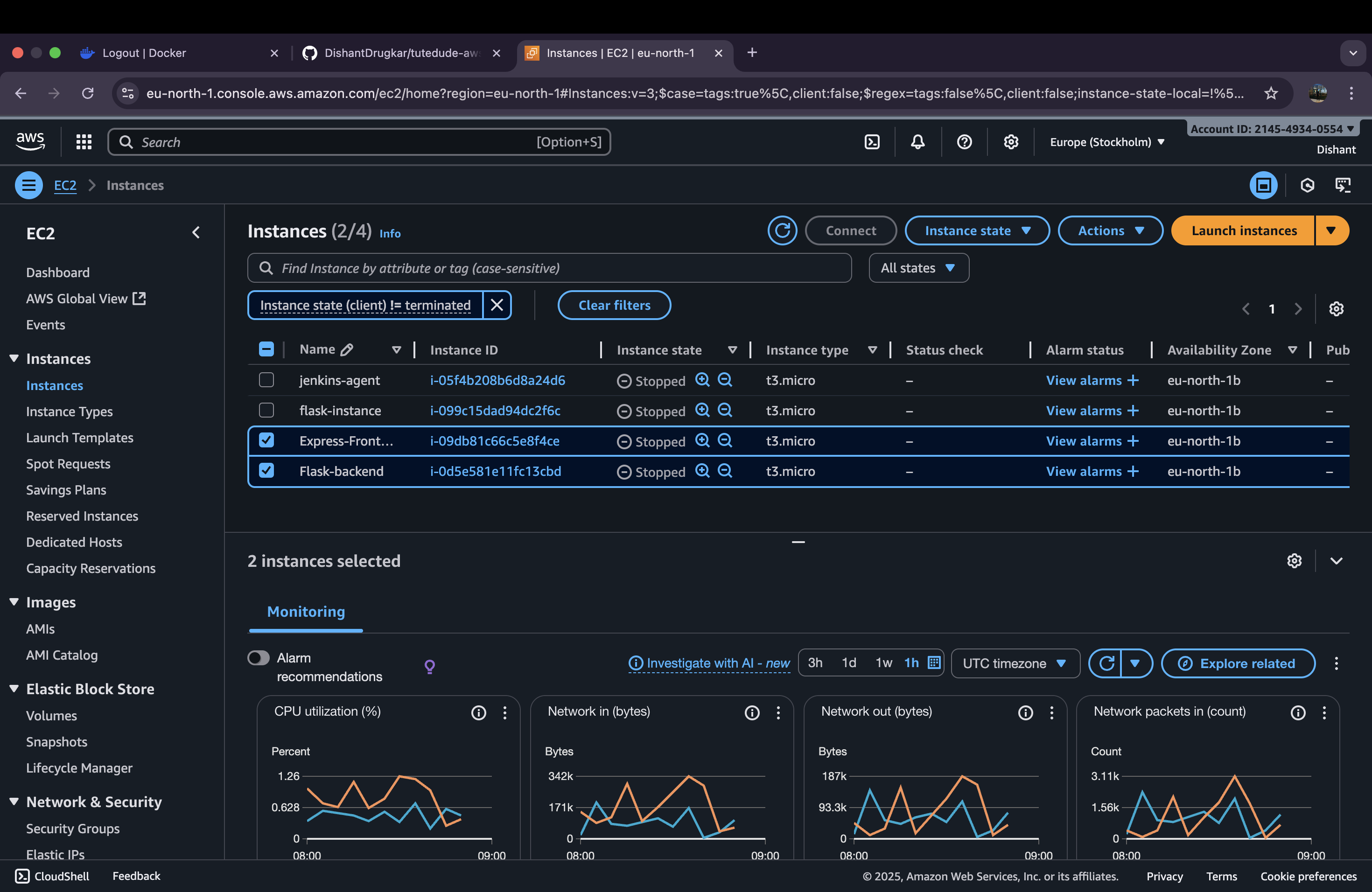
**1. EC2 Setup:**

• Launched two EC2 instances (Ubuntu).

• Configured security groups:

• Flask backend instance: inbound 5000 for HTTP, 22 for SSH.

• Express frontend instance: inbound 3000 for HTTP, 22 for SSH.

**2. Backend Deployment (Flask):**

• Flask app was uploaded to the backend EC2 instance.

• Installed dependencies using pip.

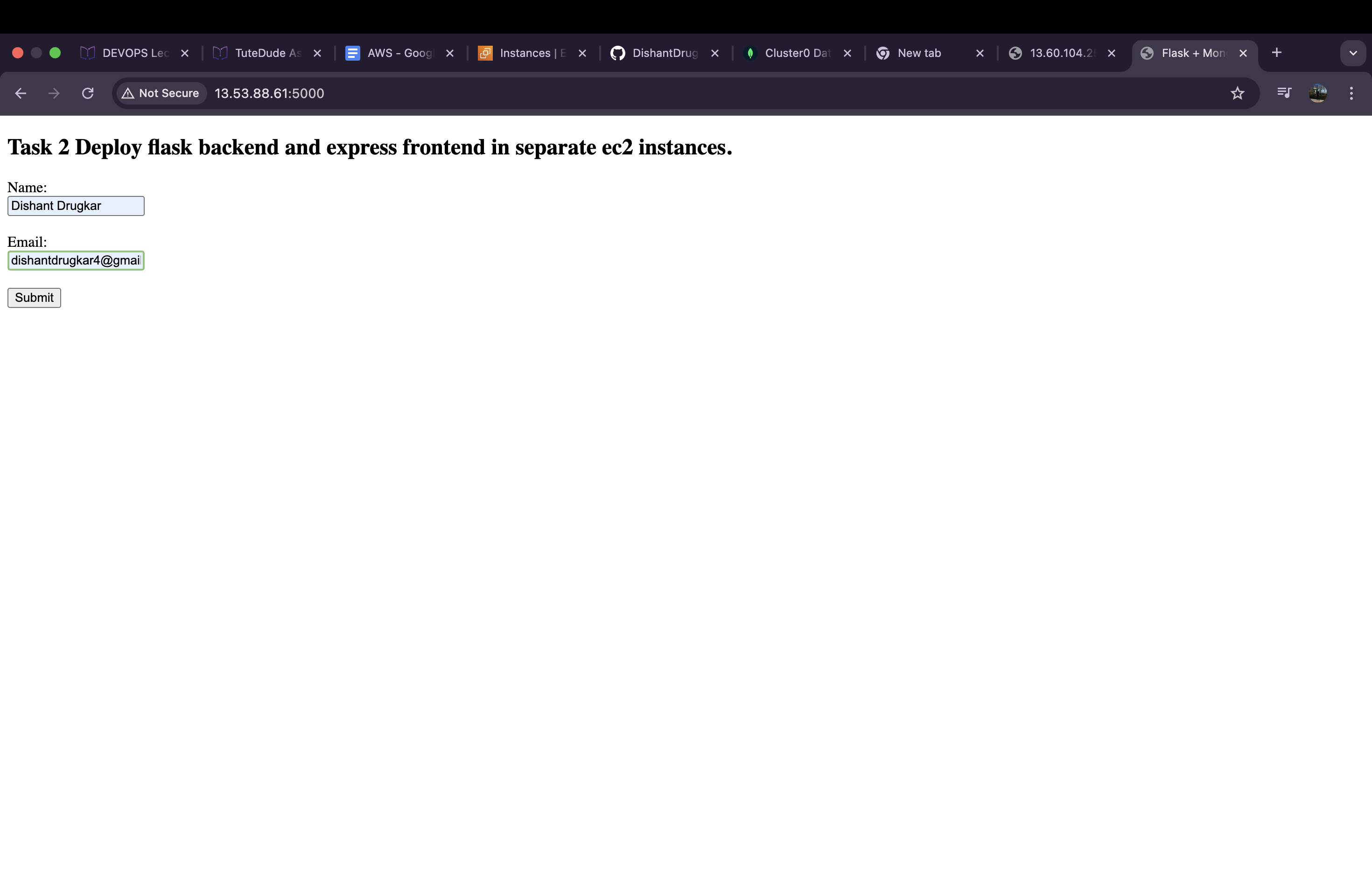
• Ran Flask on 0.0.0.0:5000 to allow external access.

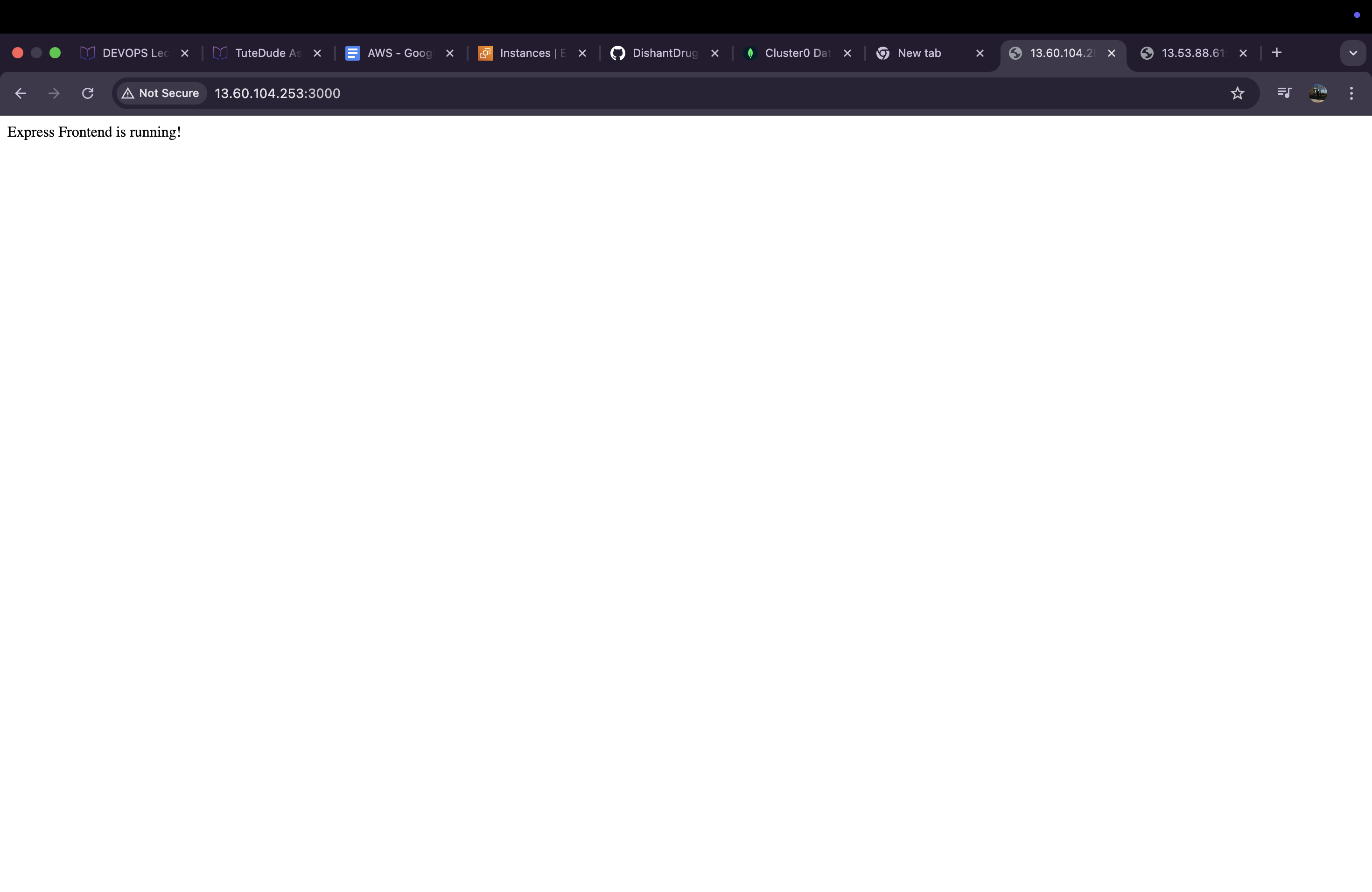
**3. Frontend Deployment (Express):**

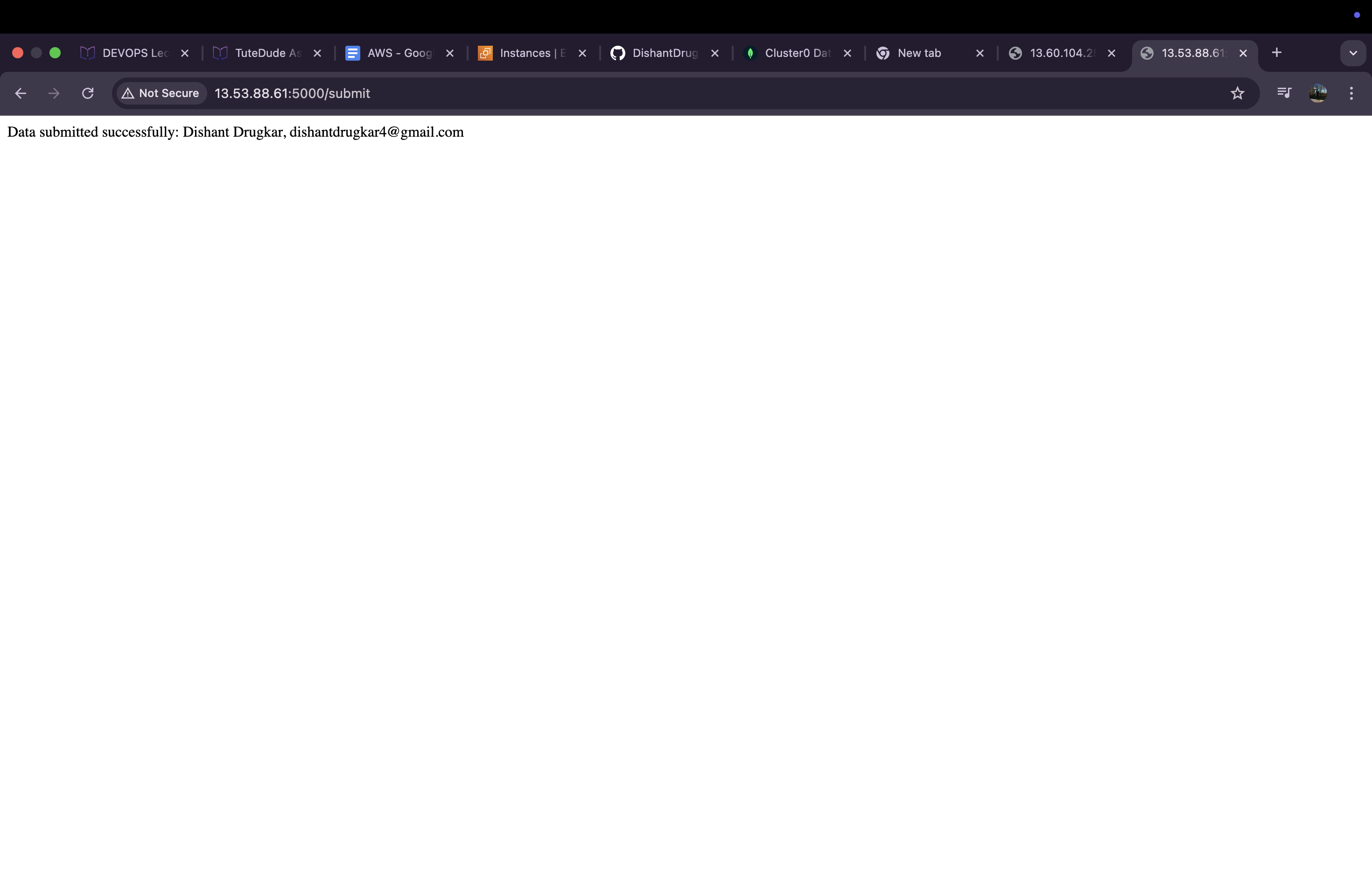
• Express app uploaded to frontend EC2 instance.

• Configured frontend to send requests to the backend EC2 public IP.

• Installed dependencies and started server on port 3000.



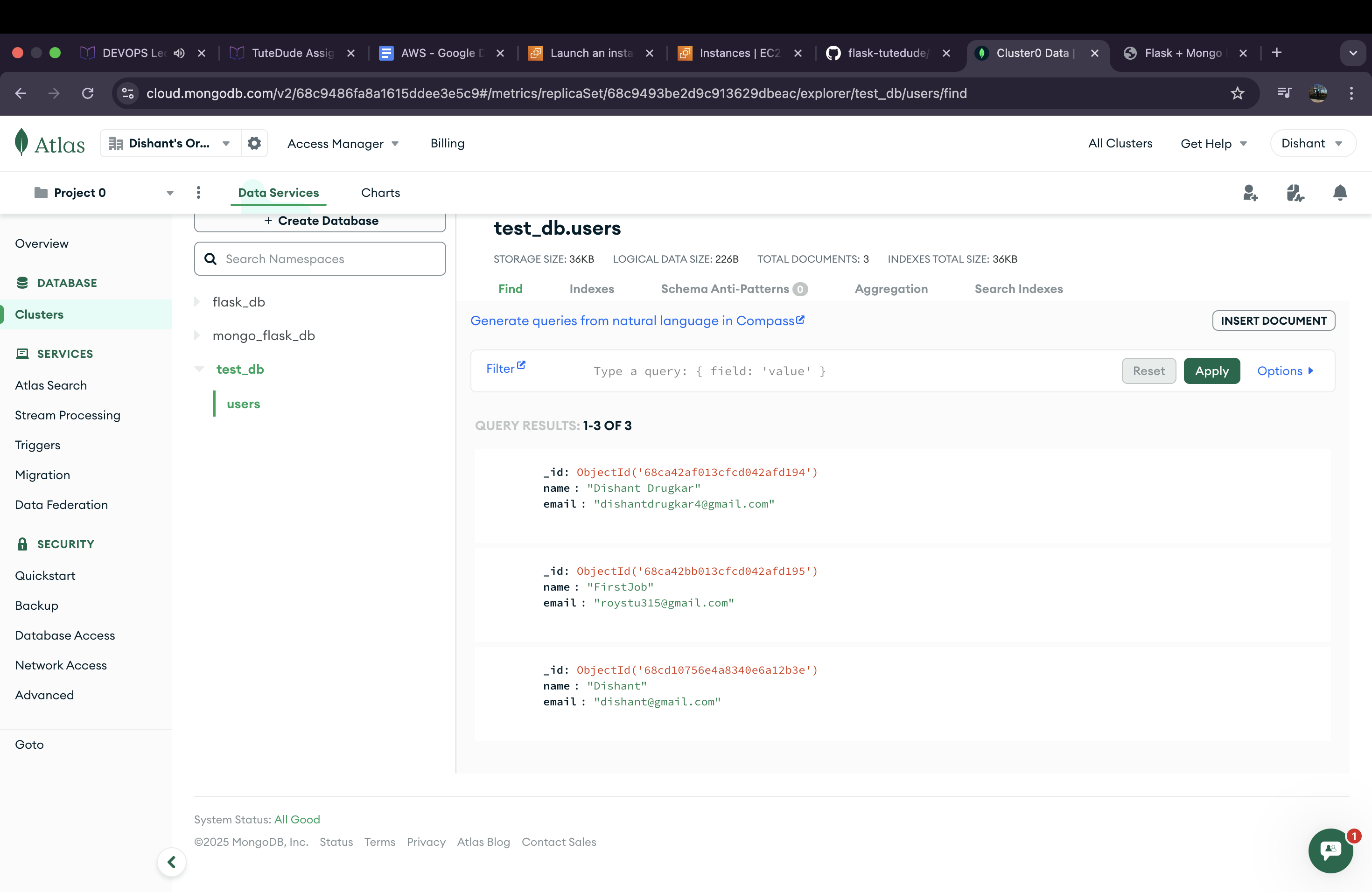




**4. Database Verification (MongoDB Atlas):**

• Submitted data via frontend.

• Verified MongoDB Atlas collection shows inserted data.

****

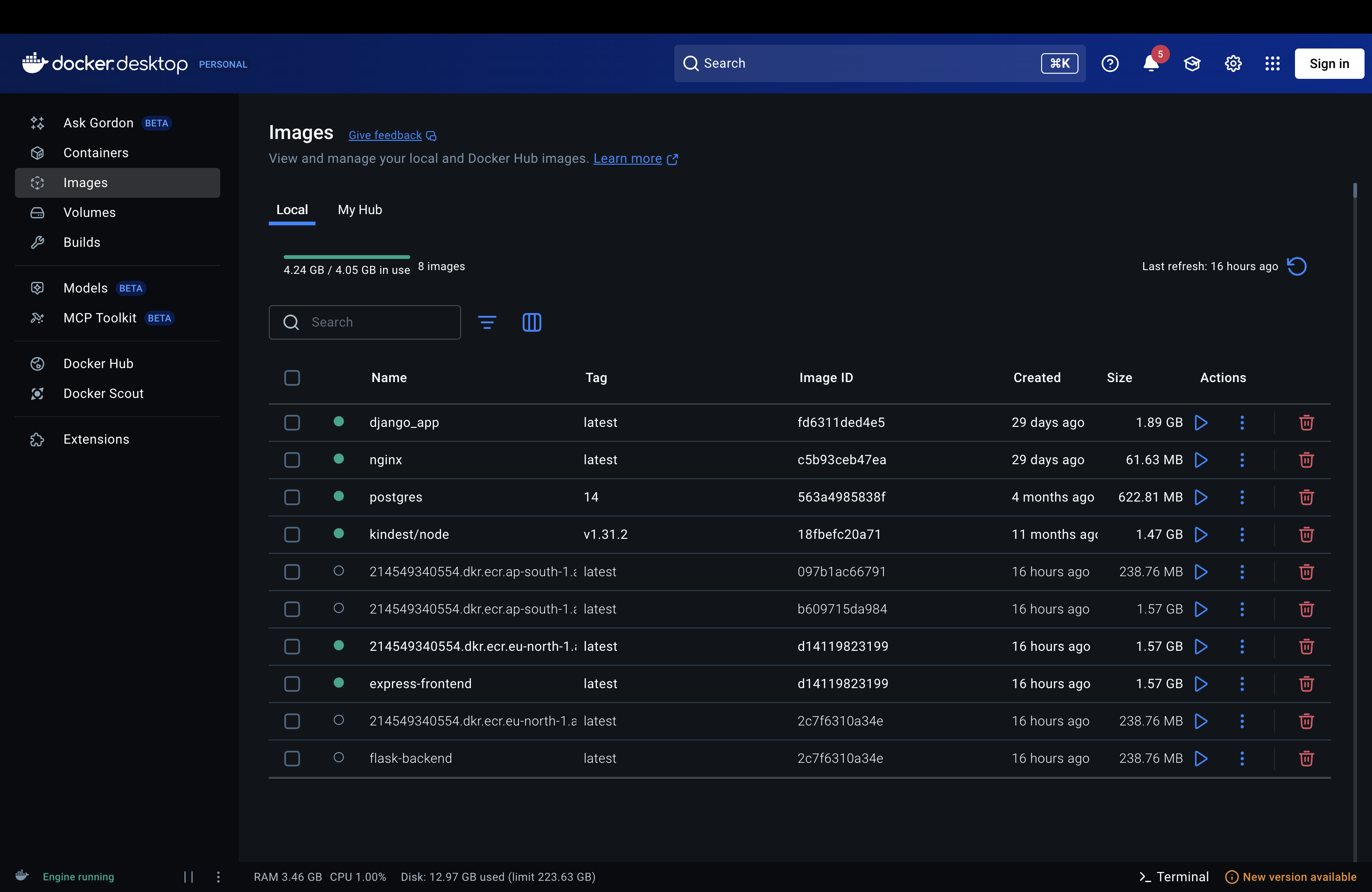
**Task 3 : Deploy Your flask backend and express frontend Docker Container using aws ecr, ecs and vpc services**

**1.Dockerization:**

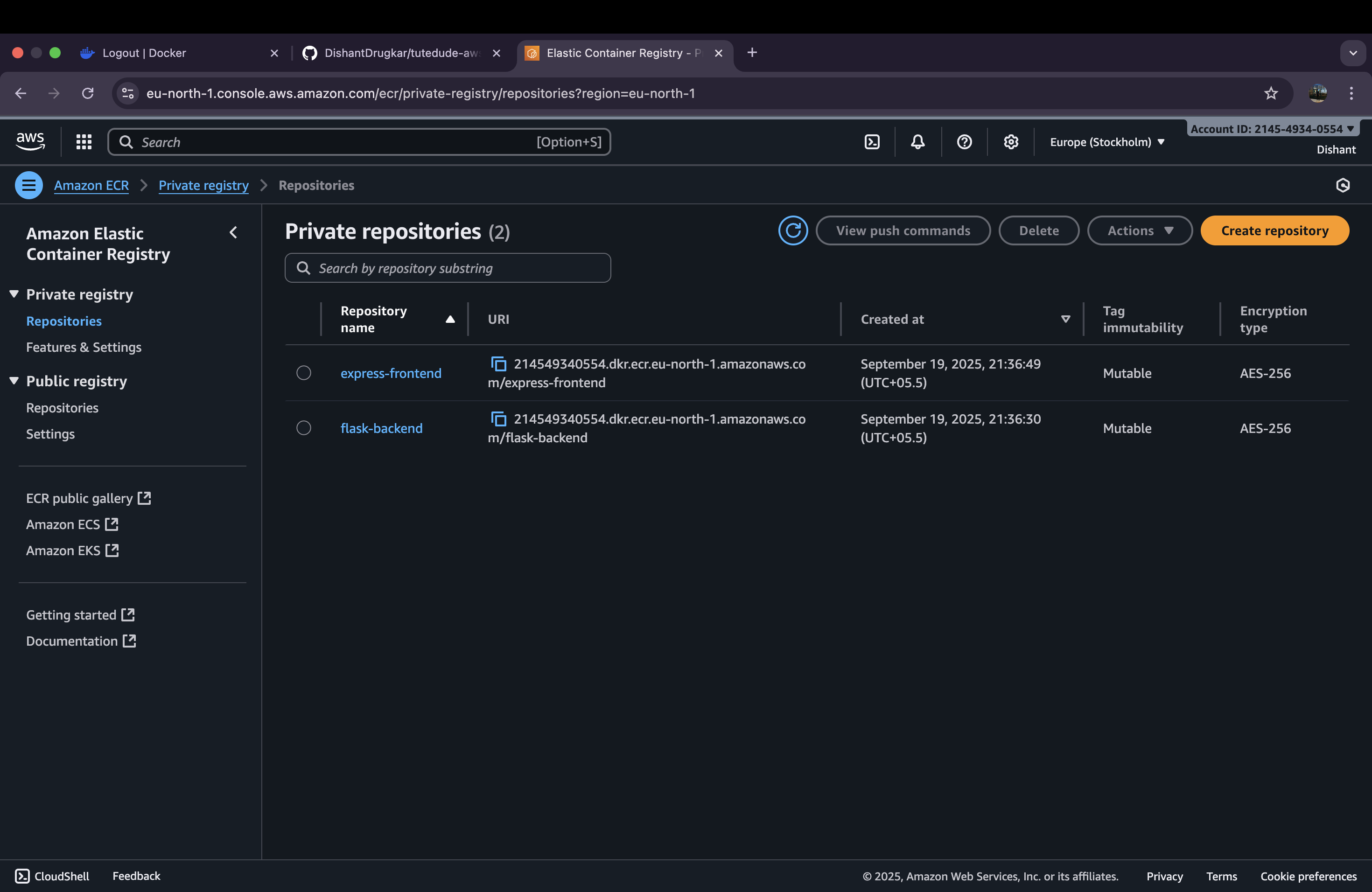
• Created Dockerfile for both Flask (backend) and Express (frontend).

• Built images locally and tested them using Docker.

• Verified container endpoints on local machine.

** 2. Push to AWS ECR:**

• Created two ECR repositories (backend & frontend).

• Tagged and pushed Docker images to ECR.

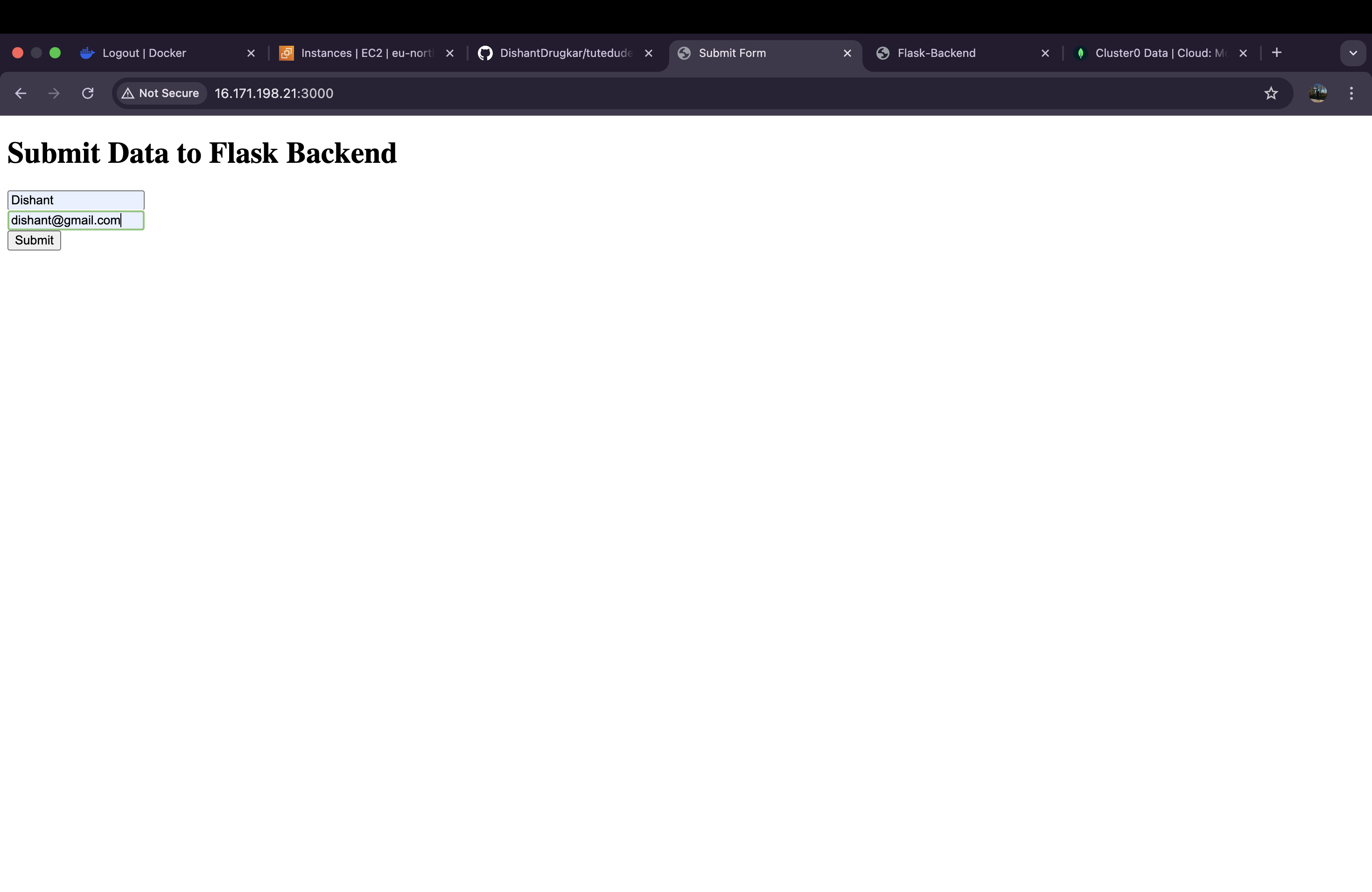
**3. ECS Deployment:**

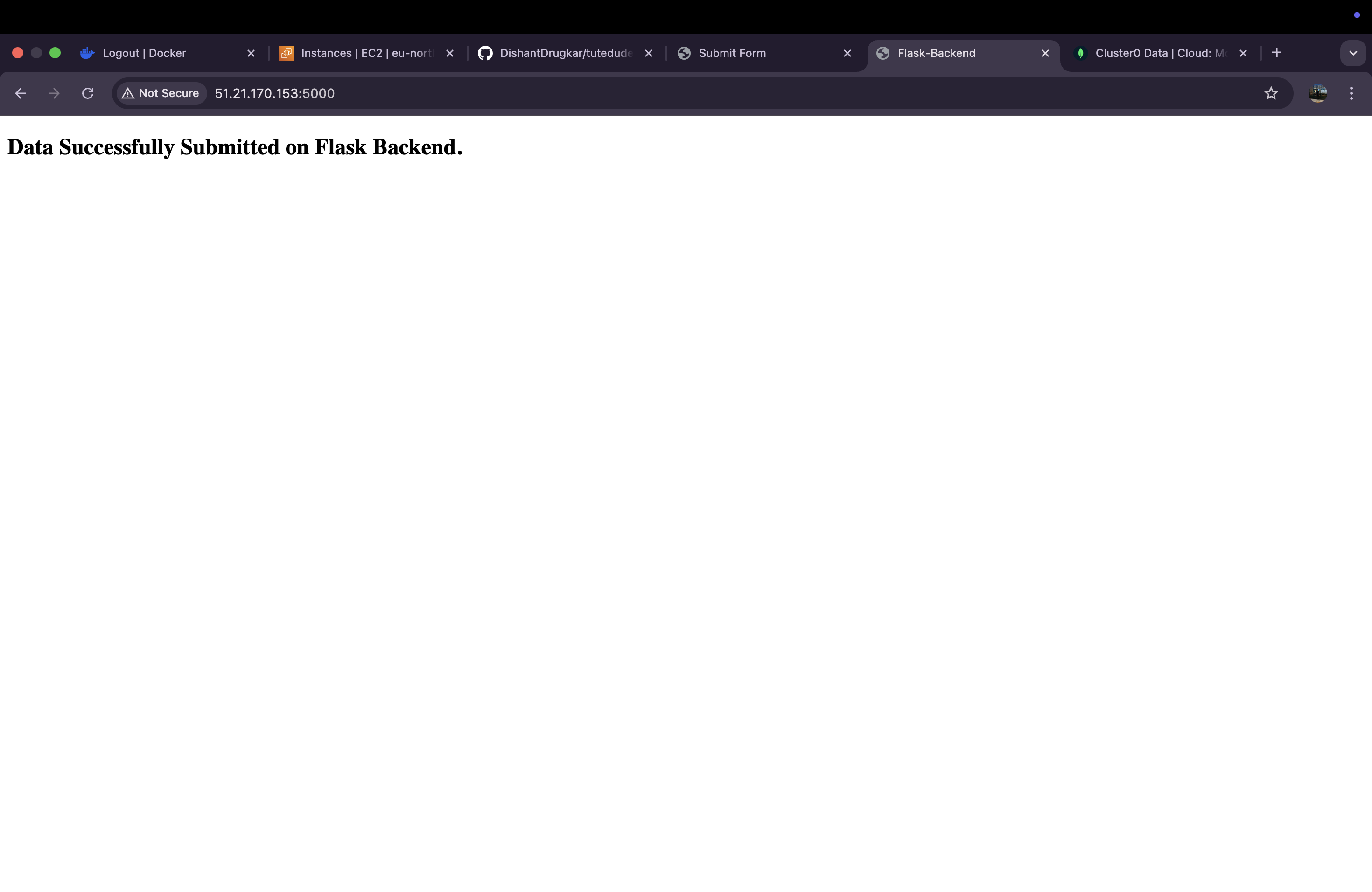
• Created a VPC with public subnets.

• Created ECS Cluster in that VPC.

• Defined Task Definitions for backend & frontend pointing to respective ECR images.

• Created ECS Services for both tasks with public IP enabled.

****

****

**4. Database Verification (MongoDB Atlas):**

• Submitted data via frontend ECS container.