Assignment Report

# Task 1 – FastAPI and MongoDB Setup

In Task 1, a RESTful web API was developed using FastAPI and connected to MongoDB Atlas using the Motor async driver.   
The following endpoints were created:  
- `POST /upload\_sprite` to upload sprite images  
- `POST /upload\_audio` to upload audio files  
- `POST /player\_score` to record player scores

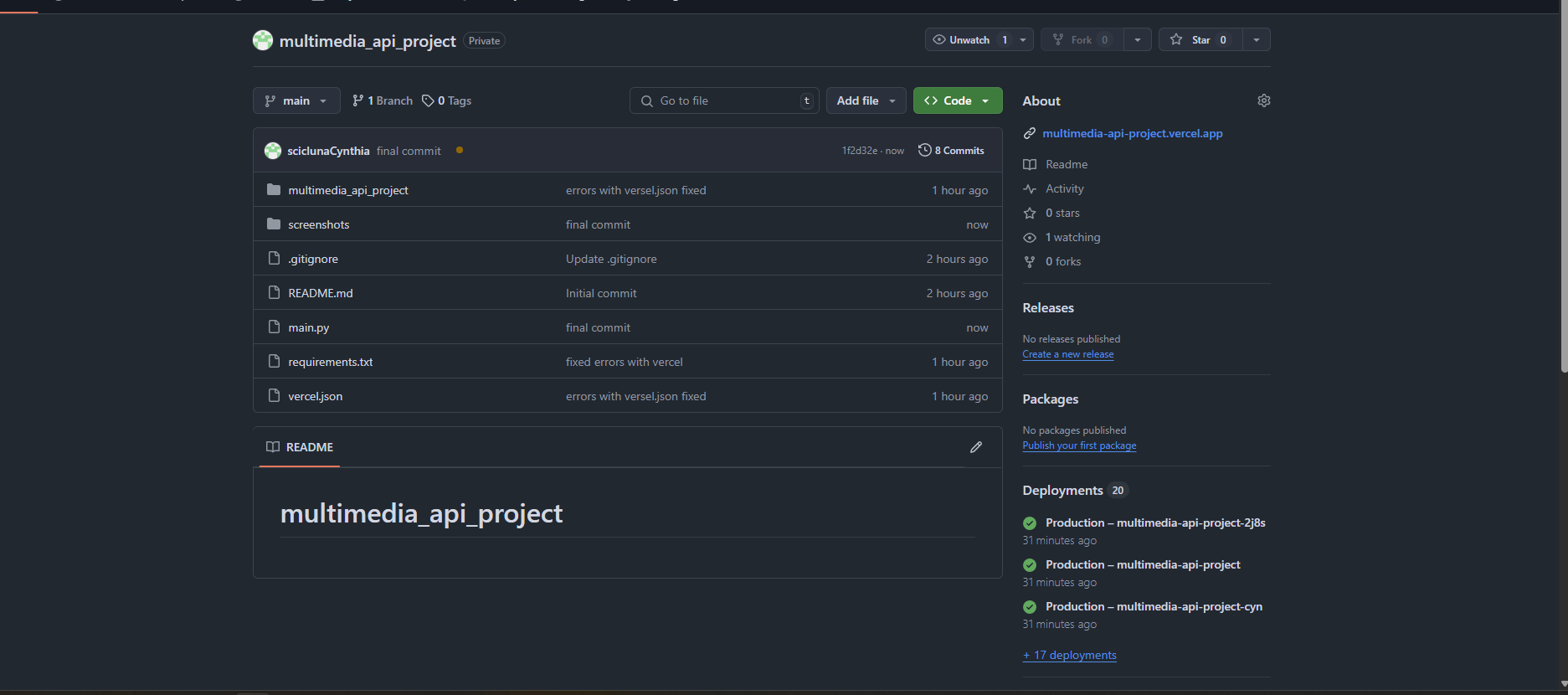
GitHub Repository link:   
  


Figure shows github repositry project files

A screenshot of a computer

AI-generated content may be incorrect.

Figure shows MongoDB Atlas cluster dashboard

A screenshot of a computer

AI-generated content may be incorrect.

Figure Successful deployment of vercel app live

# Task 2 – Mock Data Upload to MongoDB

Mock data was manually inserted into the MongoDB Atlas cluster via the web interface to simulate expected content.   
The collections were structured as follows:  
- `sprites`: Stored binary image content and filenames (e.g., `player\_idle.png`)  
- `audio`: Stored audio files with names and placeholder content (e.g., `background\_music.wav`)  
- `scores`: Stored player name and score values (e.g., `{"player\_name": "Cynthia", "score": 9999}`)

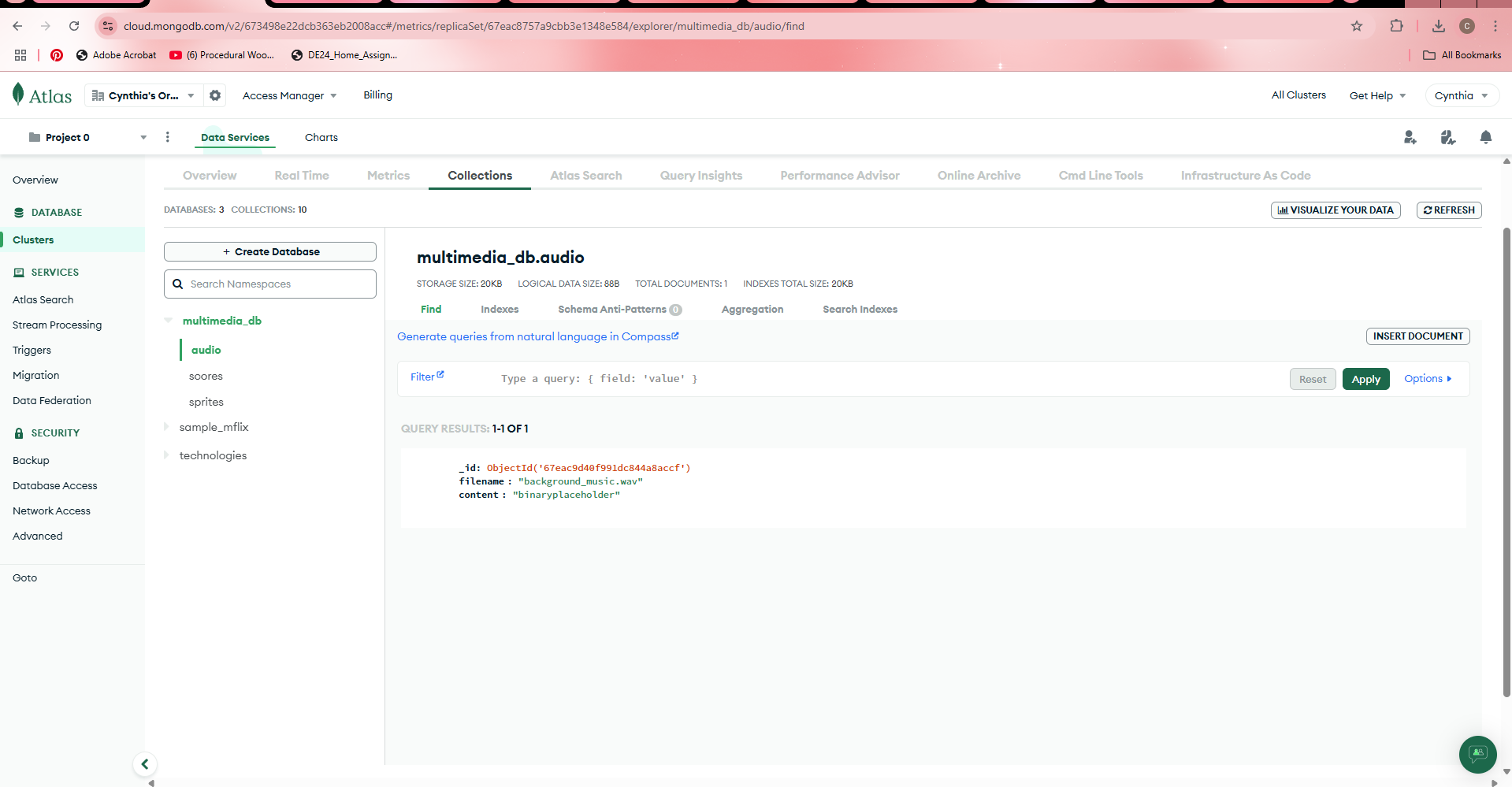


Figure shows audios

A screenshot of a computer

AI-generated content may be incorrect.

Figure shows scores

A screenshot of a computer

AI-generated content may be incorrect.

Figure shows sprites

# Task 3 – API Testing via Swagger UI

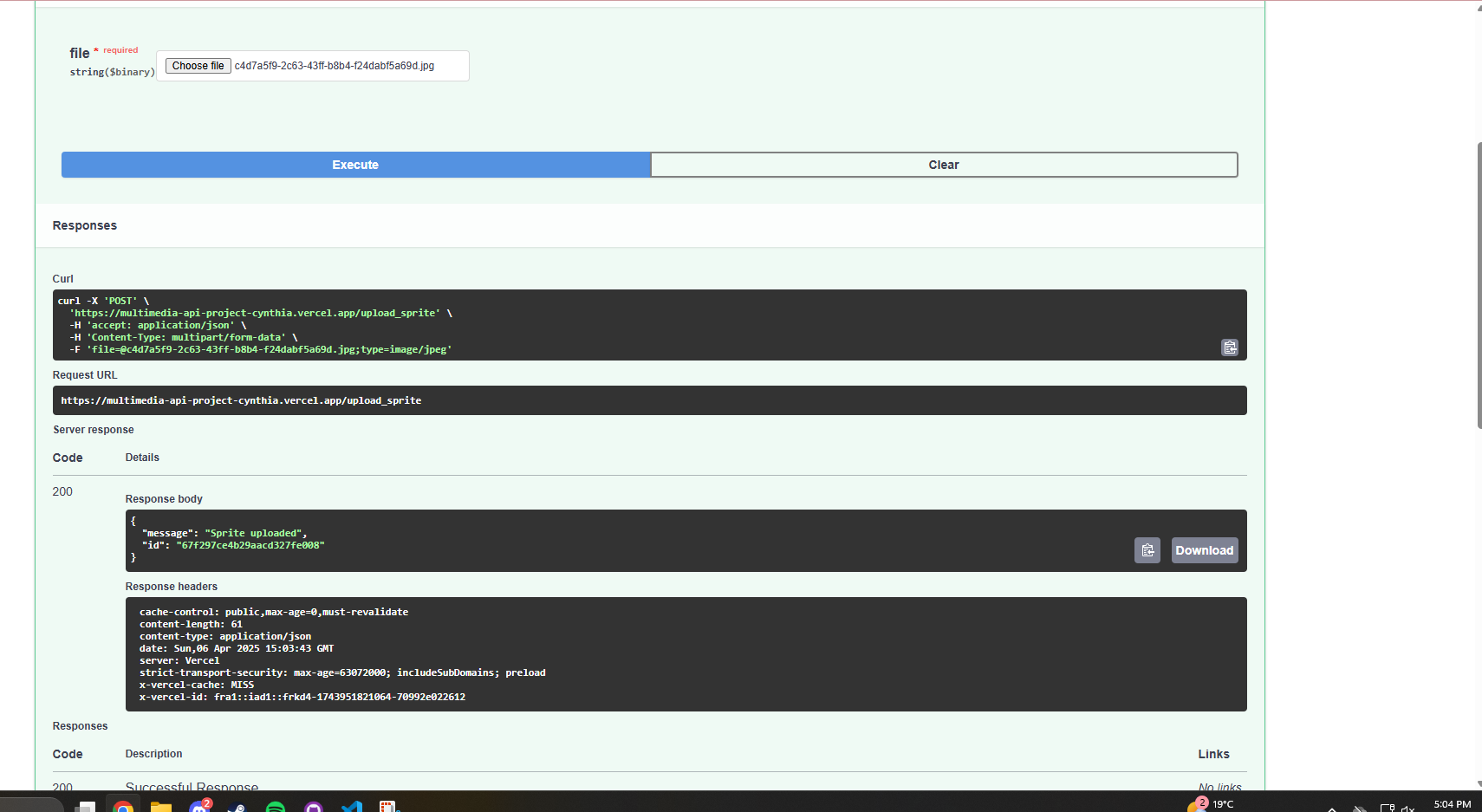
The API was successfully tested through the built-in Swagger UI provided by FastAPI. This allowed file uploads and score submissions to be tested interactively.  


Figure shows successful sprites test

A screenshot of a computer

AI-generated content may be incorrect.

Figure shows successful audios test

A screenshot of a computer

AI-generated content may be incorrect.

Figure shows successful scores test

## Live API Documentation URL

https://multimedia-api-project-cynthia.vercel.app/docs

# Task 4 – Configuring Database Security

## 4a.

A MongoDB user was created using SCRAM authentication, which is the default and secure method in MongoDB Atlas.   
The user was assigned a strong password and granted appropriate privileges using built-in roles.

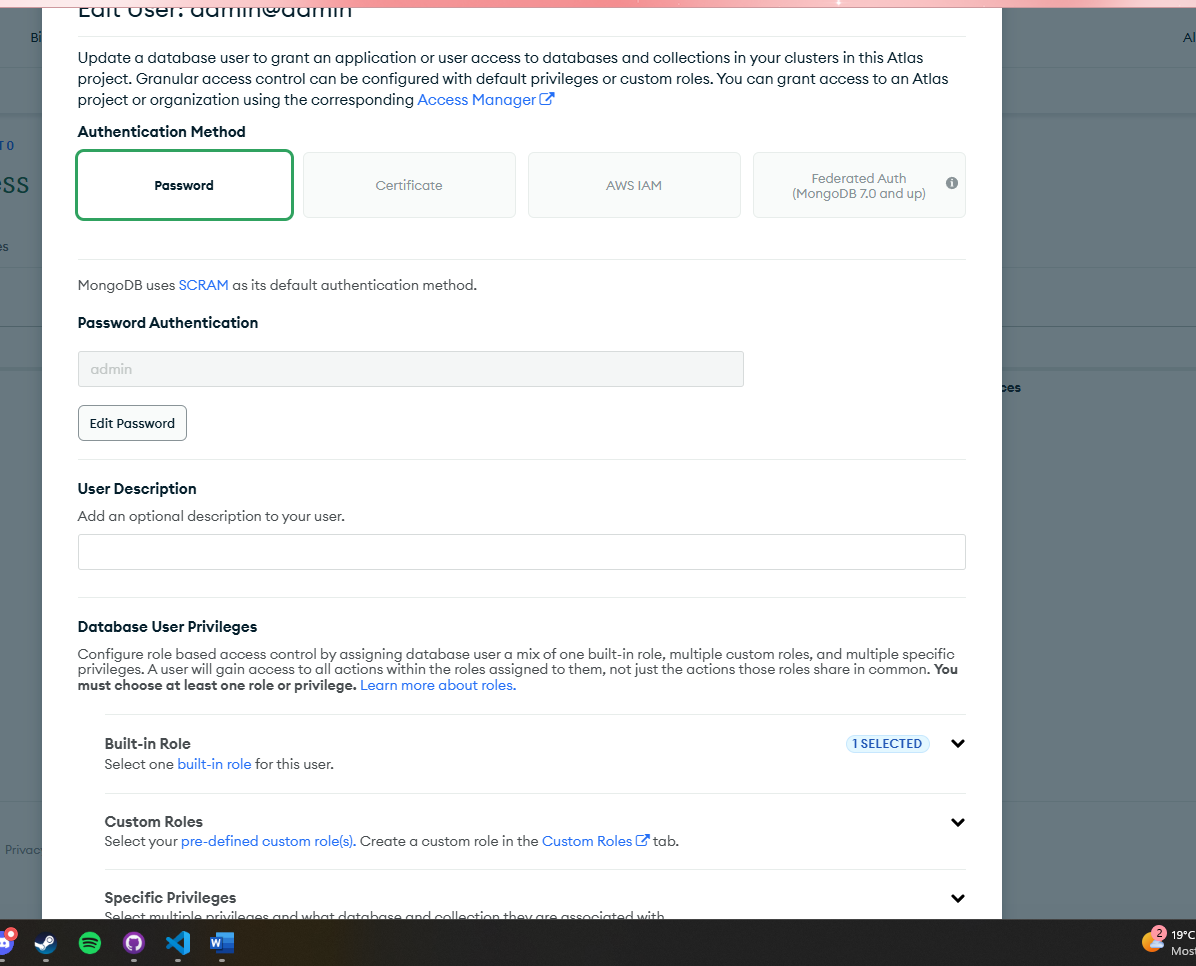


Figure shows MongoDB user credentials setup (admin@admin)

## 4b.

The cluster network access was configured by whitelisting the IP address range `0.0.0.0/0`, which allows connections from any IP address.  
This is appropriate for development and Vercel deployment. In production, this would be restricted to known server IPs only.

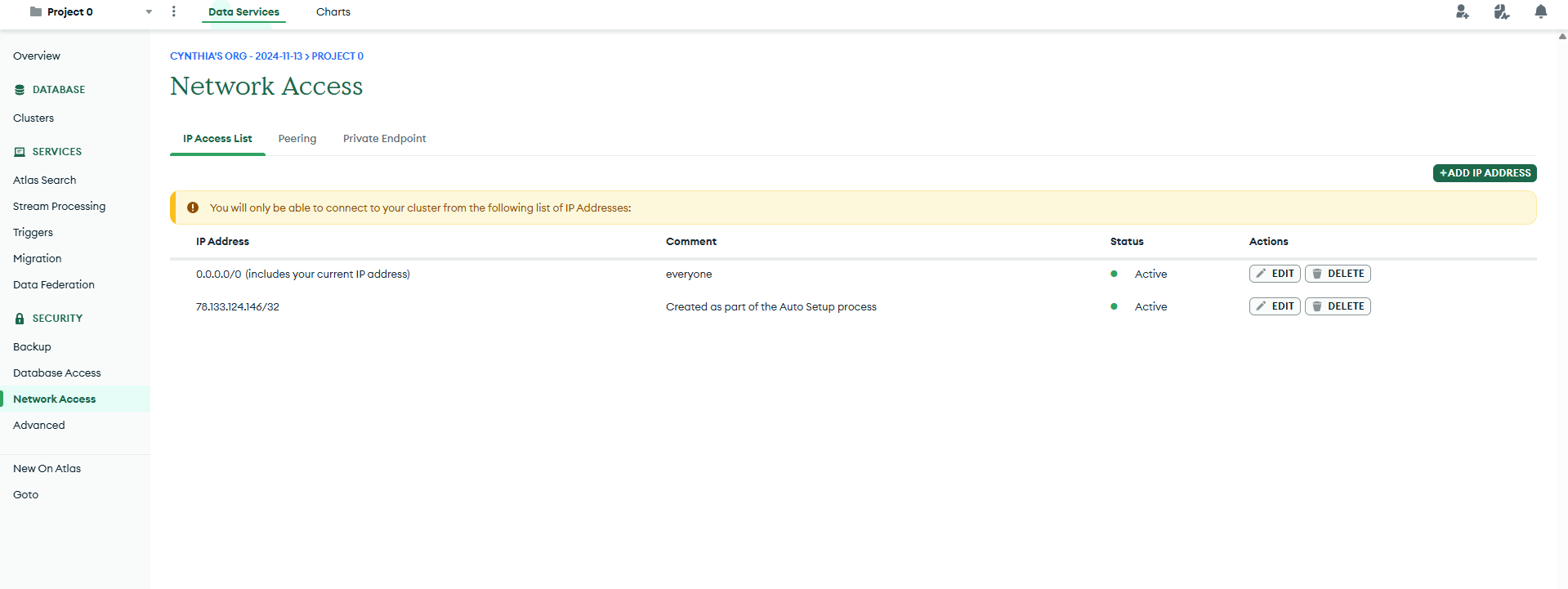


Figure shows IP Whitelist configuration in MongoDB Atlas

## 4c.

Input validation is enforced using Pydantic models, which ensure the types of inputs are strictly validated (e.g., ‘player\_name` as a string)  
To enhance security, a manual string check was added to prevent characters commonly used in injection attacks such as `$`, `{`, and `}`.   
This protects against NoSQL injection attempts.  
  
A black background with white text

AI-generated content may be incorrect.

Figure shows how I added custom input validation

A computer screen shot of code

AI-generated content may be incorrect.

Figure shows that before inserting the player’s name and score into MongoDB, I validate that the player\_name is safe

A black screen with white text

AI-generated content may be incorrect.

Figure shows that I ensured that the correct data types are being sent