# **USER MANAGEMENT PROJECT**

# **( FINAL REFLECTION DOCUMENTATION )**

**INSTRUCTOR: KEITH WILLIAMS**

**COURSE:**

**IS601: WEB SYSTEMS DEVELOPMENT**

**STUDENT DETAILS:**

**NAME: KRISHNA SATHVIKA GANNI**

**NJIT ID: 31715411**

**MY LEARNINGS FROM THIS COURSE:**

* I have learned many practical skills in this course, starting with how to use GitHub effectively for version control and collaboration.
* The course introduced me to a lot of tools and ideas that I did not know about before, and made me learn a lot more about building, testing, and deploying realistic applications.
* The midterm calculator project was a stepping block that enabled me to get acquainted with the development process and prepare myself for the challenges of this Epic User Management project.

**MY EXPERIENCE WORKING ON THIS PROJECT:**

* Throughout the User Management Project, I gained first-hand experience in debugging, clean code writing, user authentication, media upload handling, and integration with third-party storage services like Minio.
* I also understood the importance of writing thorough test cases and ensuring quality with automated tests and issue tracking.
* This experience not only improved my technical skills but also made me understand how to approach problems in a systematic manner and work as part of a team.
* Most significantly, I achieved a very high level of confidence in creating production-grade applications that are secure, sustainable, and ready for production.

**PROJECT SETUP:**

* Fork the Professor’s repository.
* Clone the forked repository locally to work.
* Create a local .env file and add your MailTrap SMTP settings. This enables testing of email features during manual testing. (Note: During automated testing with pytest, email sending is mocked and not actually performed.)
* Running pytest will delete the user table, but it does not remove the Alembic migration table, which can lead to sync issues. To resolve:
  + Drop the Alembic table manually if needed.
  + Re-run the migration using:

**docker compose exec fastapi alembic upgrade head**

* Start the project with Docker:

**docker compose up --build**

* Access the User Management website at localhost/docs
* Access PGAdmin at [localhost:5050](http://localhost:5050)
* Access the Minio at localhost:9001
* To view logs for FastAPI service:

**docker compose logs fastapi -f**

* Execute all tests inside the container:

**docker compose exec fastapi pytest**

## **QA ISSUES:**

**ISSUE 1: Mismatch of the Nickname**

**Description:**

* The issue is that the Create User API doesn't match the nickname stored in the database input to the request.
* It is also seen as inconsistent with API reaction and example values.
* The Nickname gets overwritten during processing, leading to inconsistencies between input, stored data and output.

**Unresolved:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



**Resolved:**

A screenshot of a computer

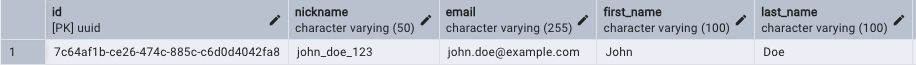
Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



**ISSUE 2: User ID passed as None in the Verification Email**

**Description:**

* At the time of user registration, the email verification link was generated with a None user ID.
* Thus the email link becomes invalid or incomplete.

**Unresolved:**

A screenshot of a computer

Description automatically generated

**Resolved:**

A screenshot of a computer

Description automatically generated

**ISSUE 3: Verification Token Missing or Invalid**

**Description:**

* User is getting a 400: Invalid or expired token for verification error.
* The verification URL is also "None," i.e., token is not being generated, not being stored, or is not being placed properly inside email verification link.

**Unresolved:**

A screenshot of a computer

Description automatically generated

**Resolved:**

A screenshot of a computer

Description automatically generated

**ISSUE 4: Role Changes from Admin to Authenticated**

**Description:**

* Whenever an admin user's email was verified, their role was automatically downgraded to "Authenticated".
* This access control violation was resolved by fixing the buggy logic to preserve the original role.

**Unresolved:**

A grey box with black text

Description automatically generated

**Resolved:**

A grey box with black text

Description automatically generated

**ISSUE 5: Confusing Login Prompt Asking for "Username" Instead of "Email"**

**Description:**

* The login prompt asked for a "username" when the system was looking for an email.
* This inconsistency caused login errors and confusion.

**Resolved:**

A screenshot of a computer

Description automatically generated

**ISSUE 6: Image Upload Endpoint Accepts All File Types Instead of Only Images**

**DESCRIPTION:**

* The image upload endpoint was allowing any file type to be uploaded.
* This was not secure and violated expected behavior.
* The validation was updated to permit only JPEG, PNG, or WEBP image file types.

**Unresolved:**

A screenshot of a computer

Description automatically generated

**Resolved:**

A screenshot of a computer

Description automatically generated

**FEATURE: 🌄 Profile Picture Upload with Minio**

**Description:**

* A new functionality was added that allows profile images to be uploaded by users, which are stored in Minio (an S3-compatible object store).
* The feature includes validations for the supported image types, UUID-based renaming to ensure uniqueness, and accessible image URLs.
* This greatly enhanced user personalization and cloud storage integration.

**Working of the Feature:**

A close-up of a blue and green rectangle

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A person wearing glasses and a suit

Description automatically generated

A screenshot of a computer

Description automatically generated

**TEST CASES:**

**TEST CASE 1: Basic Upload Functionality for Minio Image**

**Description:**

* Ensures that the core image upload to Minio is working and returns a successful image URL.

**TEST CASE 2: URL Generation for Minio Image**

**Description:**

* Ensures that the get\_image\_url\_from\_minio() function returns a valid and well-structured public image URL from a filename.

**TEST CASE 3: Long Filename Handling for Minio Image**

**Description:**

* Tests how the function behaves with extremely long filenames, to make sure they are correctly converted to UUID-based names.

**TEST CASE 4: Custom Bucket Configuration for Minio Image**

**Description:**

* Ensures that image uploads correctly utilize a custom bucket name if provided in the configuration.

**TEST CASE 5: URL Format Variations for Minio Image**

**Description:**

* Tests whether different formats of base URLs (e.g., with and without trailing slashes) are properly handled in image URL generation.

**TEST CASE 6: List Users with Pagination**

**Description:**

* Tests the endpoint for retrieving users with limit and offset to make sure there is proper pagination of results.

**TEST CASE 7: Create User with Missing Email Field**

**Description:**

* Ensures that user registration fails when the required email field is not given, ensuring proper validation.

**TEST CASE 8: Create User with Short Password**

**Description:**

* Tests that passwords shorter than the allowed length are correctly rejected when a user registers.

**TEST CASE 9: Invalid Token Access**

**Description:**

* Test that requests made with an invalid JWT token get blocked with correct unauthorized error.

**TEST CASE 10: Unauthorized User Update Attempt**

**Description:**

* Prevents a non-admin user from changing the details of another user and ensures proper access control.

**DOCKERHUB DEPLOYMENT:**

* The application was successfully containerized and deployed to DockerHub.

**DockerHub Repository:**

<https://hub.docker.com/repository/docker/krisa1329/user_management/general>

A screenshot of a computer

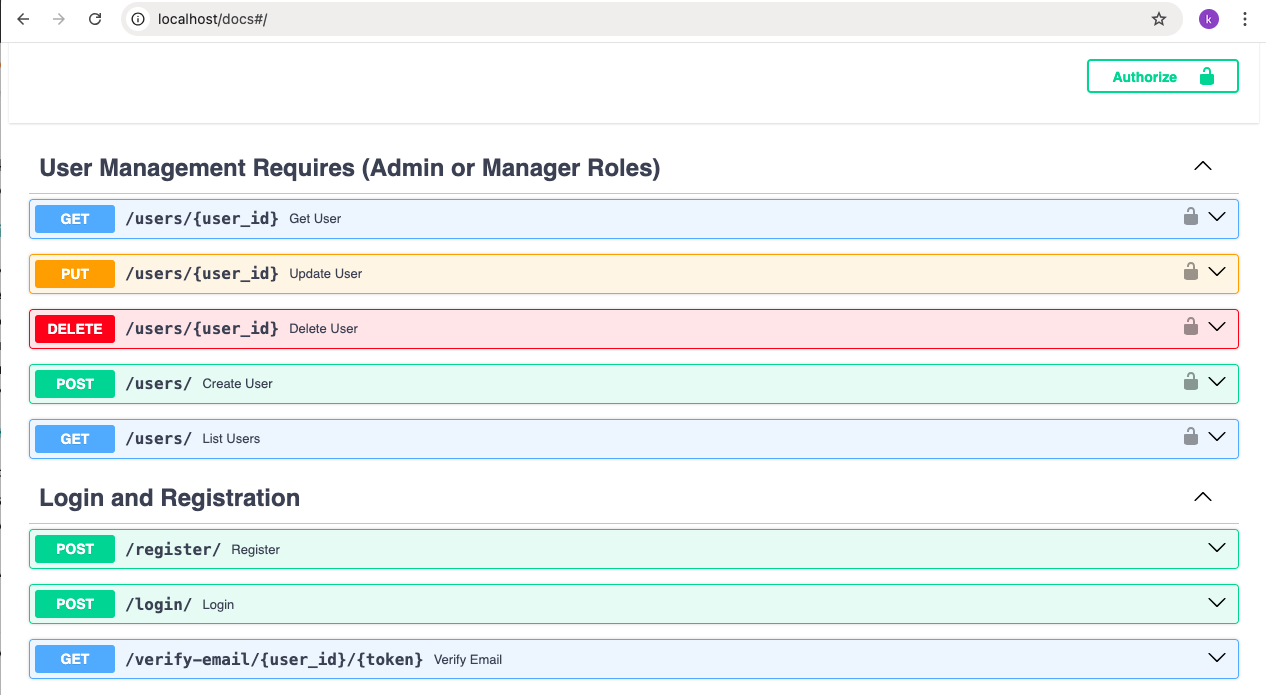
Description automatically generated

**WORKING OF THE WHOLE PROJECT:**

**USER MANAGEMENT PAGE:**

A screenshot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated

**PGADMIN PAGE:**

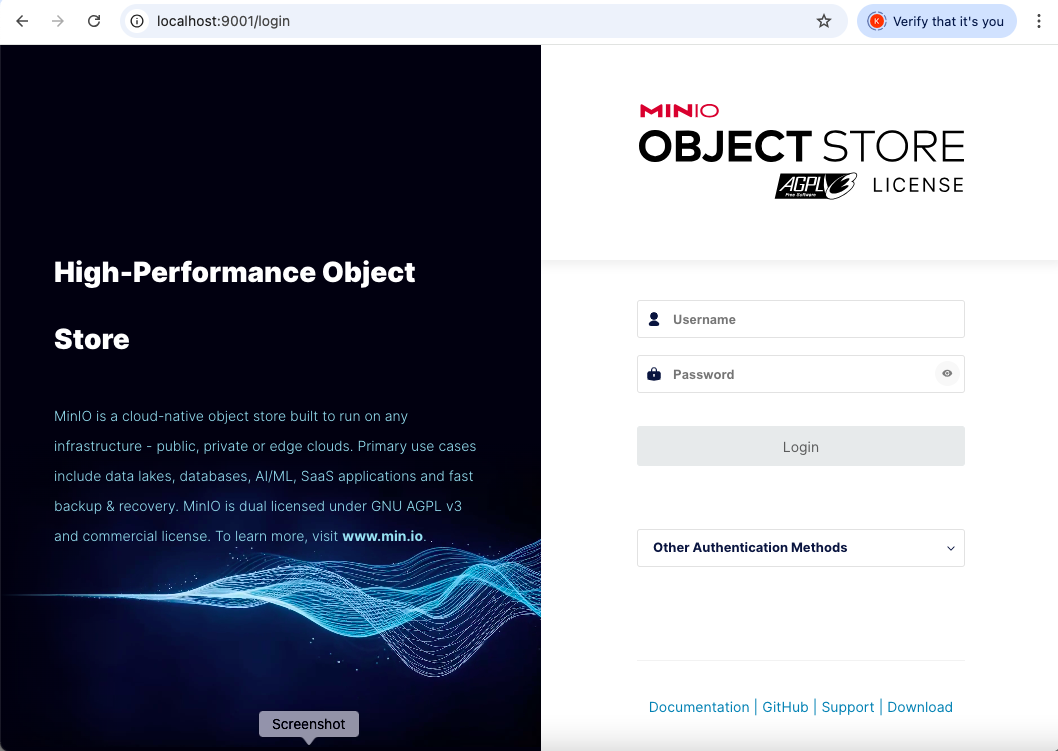
A screenshot of a computer

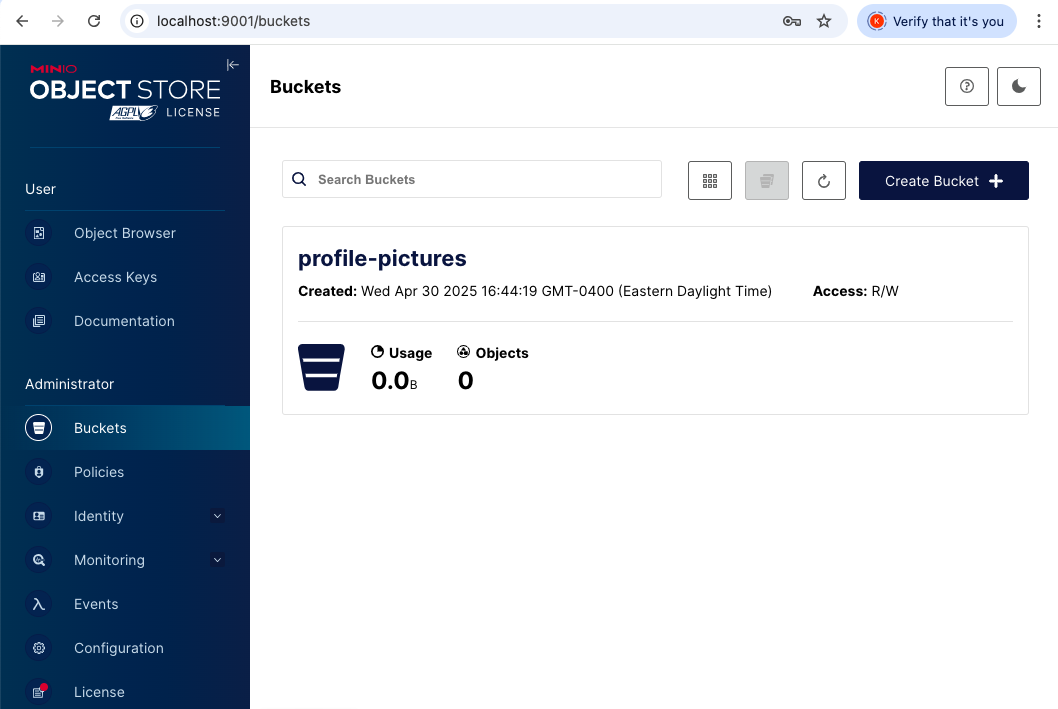
Description automatically generated

A screenshot of a computer

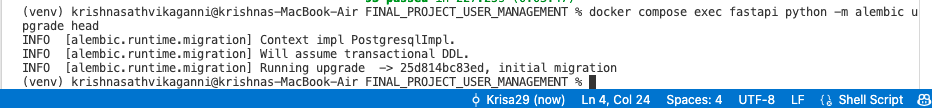
Description automatically generated

**MINIO PAGE:**

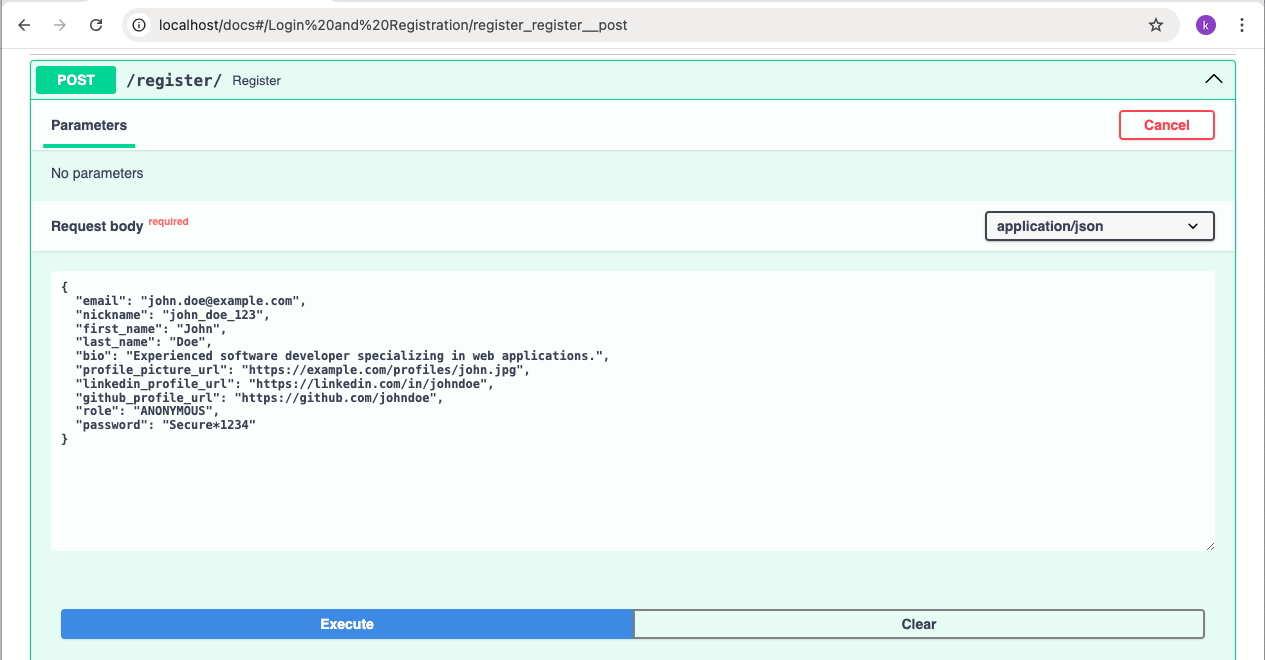


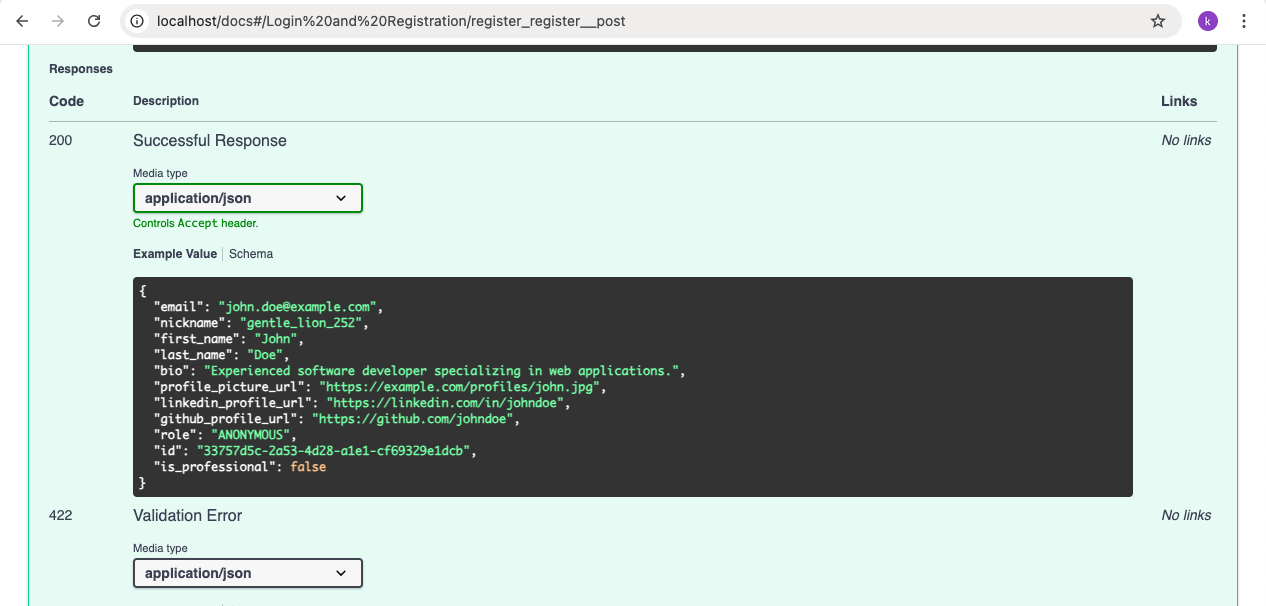


**CREATION OF TABLES IN PGADMIN WITH THE COMMAND:**

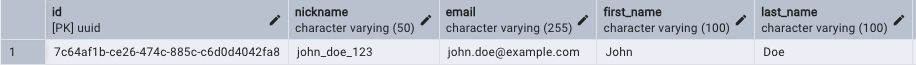


**REGISTERING OF USER:**

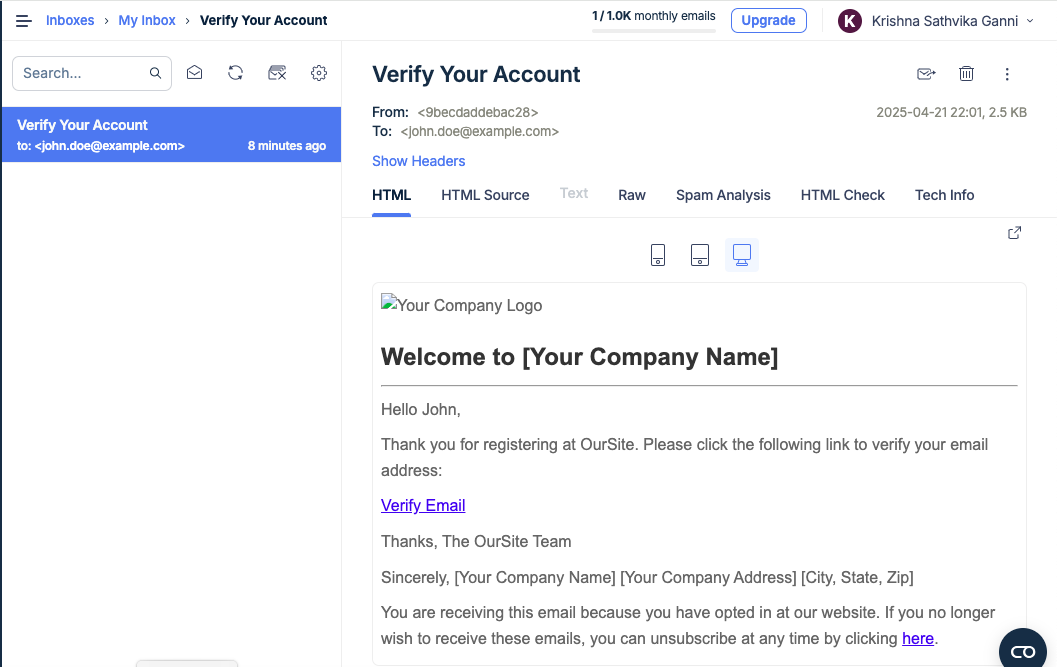




**REGISTERED USER STORED IN DATABASE:**



**VERIFICATION OF REGISTERED USER USING EMAIL:**



A screenshot of a computer

Description automatically generated

A screenshot of a phone

Description automatically generated

**LOGGING IN OF THE REGISTERED USER:**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**AUTHORIZATION OF THE USER:**

A screenshot of a computer

Description automatically generated

**PROFILE PICTURE UPLOAD OF THE REGISTERED USER:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A person wearing glasses and a suit

Description automatically generated

**REGISTERING OF ANOTHER USER:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**VERIFICATION OF ANOTHER USER:**

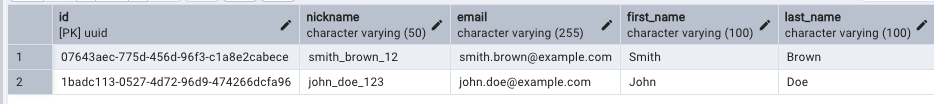
A screenshot of a computer account

Description automatically generated

A screenshot of a computer

Description automatically generated

**REGISTERED USER STORED IN DATABASE:**



**PROFILE PICTURE UPLOAD FOR ANOTHER USER:**

A screenshot of a computer

Description automatically generated

**GETTING THE USER:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**UPDATING THE USER:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated



**LISTING THE USERS:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

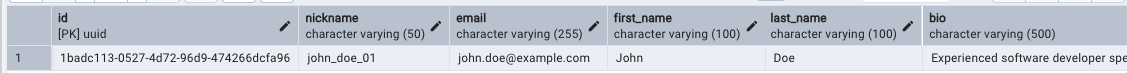
**DELETING THE USER:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



**FINAL REMARKS:**

This project was a significant milestone in my learning. It challenged me to debug, test, and apply DevOps concepts. The iterative development and QA process mimicked real-world processes and has enhanced my technical and collaboration skills.