Pet center REST API

IT325 Web Services Final Project

by

Zeineb Trimeche

January 2023

IT/BA Junior Student

Information Technology Major

Tunis Business School

Ben Arous,TUNISIA.

2022-2023

Declaration of Academic Ethics

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I declare that I have properly and accurately acknowledged all sources used in the production of this report.

I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Date: January 14th, 2023 Zeineb Trimeche

Abstract

Contents

Chapter 1

Introduction

This is a project for the IT325 Web service courses consisting of a RestAPI written in NodeJS. This project is brought by NestJS, TypeScript, Node, JWT with passportJS , MVC model with the storage layer relying on PostgreSQL as a database alongside TypeORM for database operations.

You will find in the following chapters examples of every CRUD operation, JWT authentication and authorization, MVC design pattern examples, Database relationships, Entities Validation logic and code snippets to explain some tricky parts.

This project represents a demo of a pet adoption centre where users can log in securely, browse available pets and adopt.

Chapter 2

Technical explanation

2.1 Nestjs.js

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. It allows developers to run JavaScript on the server side to build fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model, which makes it lightweight and efficient. It is commonly used for real-time applications such as chat applications, online games, and other applications that require a high amount of concurrent connections. It also has a large and active community, which provides a rich ecosystem of modules and packages to extend its functionality.

Nestjs is a backend framework based on NodeJS, it is built with TypeScript, which provides strong typing and intuitive design patterns. It is also modular, meaning that you can use only the components that you need, making your code more efficient and easier to maintain. In addition, Nestjs has a large and supportive community, making it easy to find help and resources online. It also has good documentation, making it easy to learn and use. Overall, Nestjs is a powerful and well-designed framework that can be a great choice for your web development needs.

2.2 MVC Pattern

MVC (Model-View-Controller) is a software architecture pattern that separates an application into three main components: the model (data and business logic), the view (presentation), and the controller (input and business logic). MVC aims to separate the concerns of an application,

making it easier to develop, maintain, and test. The model represents the data, the view represents the presentation, and the controller acts as an intermediary between the model and view, handling user requests and updating the view with the appropriate data.

2.3 JWT and Passport.js

Passport.js is an authentication middleware for Node.js. It simplifies the process of handling authentication in a Node.js application by providing a simple, modular, and flexible architecture. Passport.js supports many different authentication strategies, such as OAuth, LDAP, SAML, and more. It is easy to set up and use, and it integrates well with popular Node.js frameworks like Express. To use Passport.js, you simply configure the authentication strategy that you want to use and then pass incoming requests through Passport's authentication flow.

JWTStartey integration with NestJS

```
You, last week | 1 author (You)
@Injectable()
export class JwtStrategy extends PassportStrategy(Strategy) {
  constructor() {
    super({
      usernameField: 'email',
      jwtFromRequest: ExtractJwt.fromAuthHeaderAsBearerToken(),
      ignoreExpiration: false,
      secretOrKey: jwtConstants.secret,
    });
  }

  async validate(payload: User) {
    return {
      email: payload.email,
      sub: payload.uuid,
    };
  }
}      You, last week • aded hash jwt auth
```

2.4 PostgresSQL

PostgreSQL, or Postgres, is a popular open-source relational database management system that is well-suited for a wide variety of applications. It has a lot of strongpoints : Scalability: Postgres can handle high levels of concurrency and is well-suited for large-scale applications.

Flexibility: Postgres supports a wide variety of data types and can be easily extended with custom functions and data types.

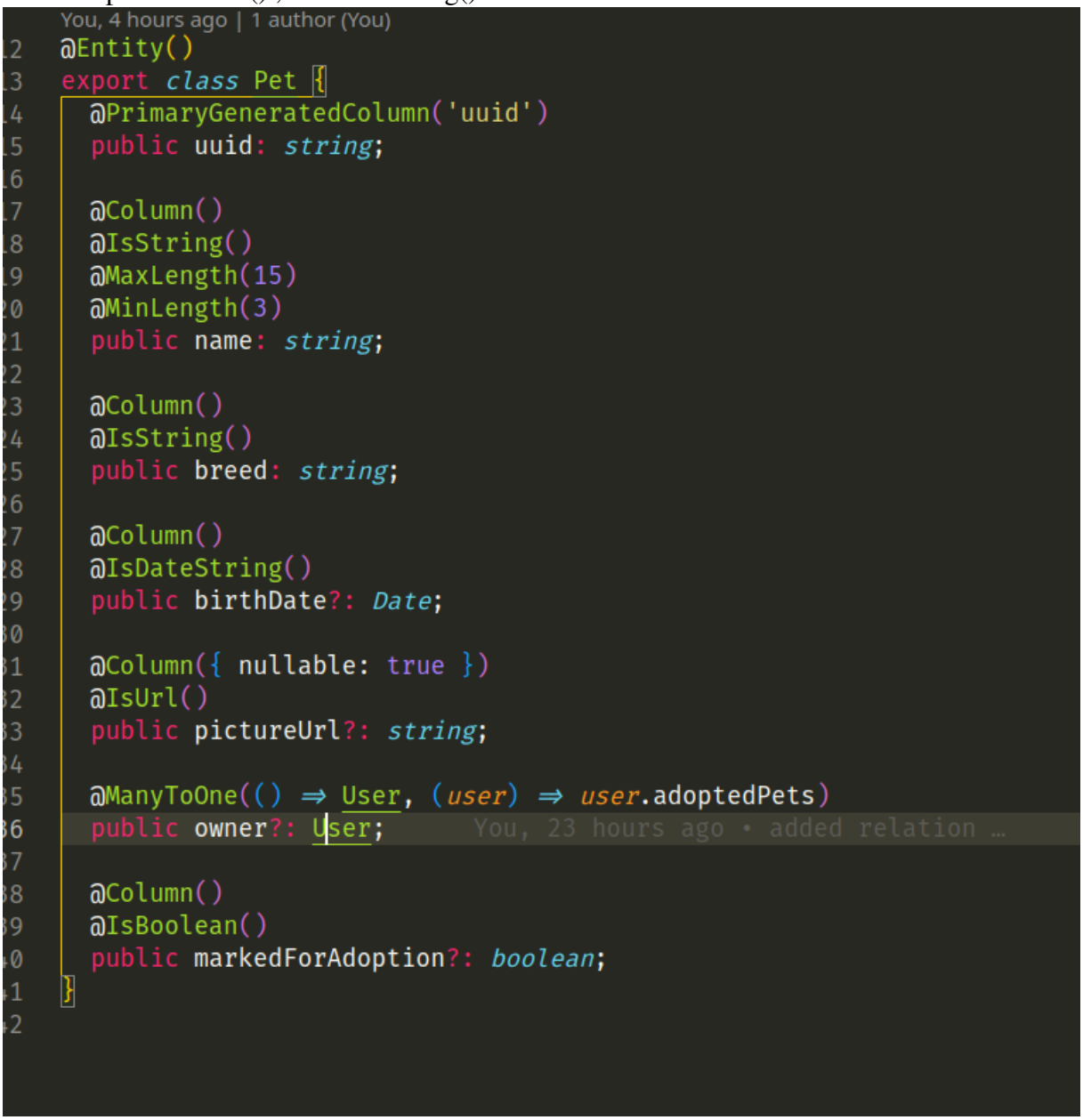
ACID compliance: Postgres ensures data integrity and consistency by supporting transactions and adhering to ACID (Atomicity, Consistency, Isolation, Durability) principles.

Strong community support: Postgres has a large and active community of developers who contribute to the project and provide support. Cross-platform compatibility: Postgres runs on many different operating systems, including Linux, Unix, and Windows. Overall, Postgres is a reliable and powerful database management system that can be a good choice for many different types of applications.

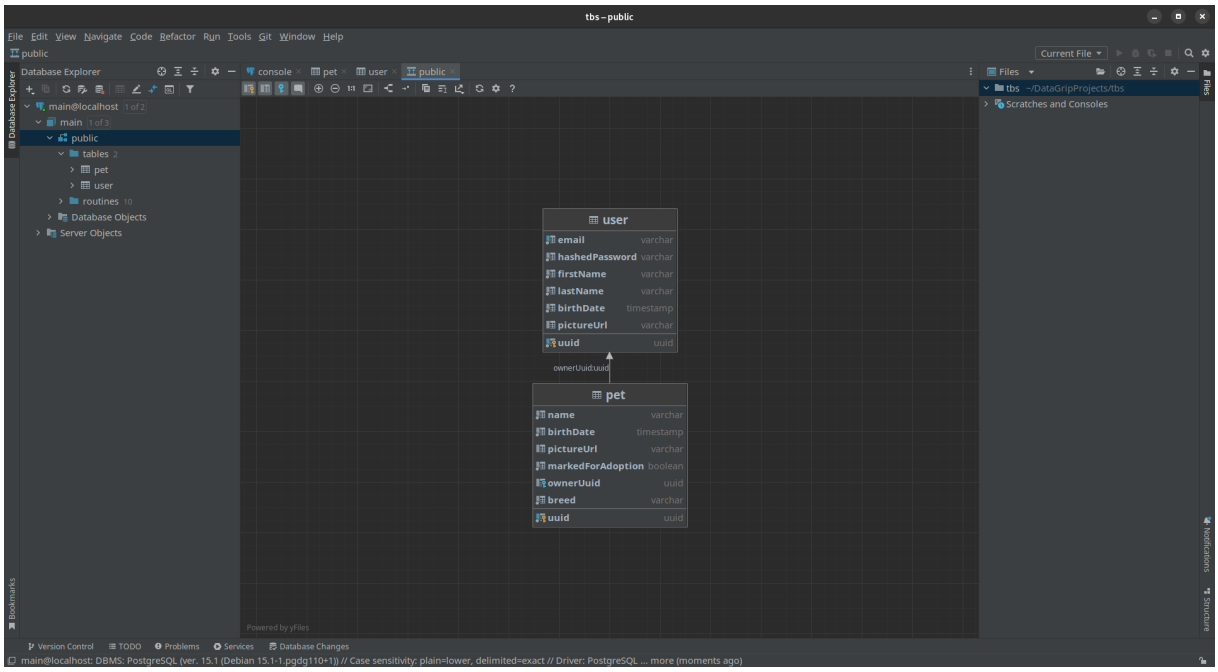
2.5 TypeORM

TypeORM is a TypeScript/JavaScript ORM tool that allows developers to interact with databases using code instead of raw SQL queries. It supports multiple databases and offers features such as migrations, connection pooling, and a query builder. It's aim is to provide high-performance and type-safe database interactions.

TypeORM also integrates with class validator library to provide validation for the defined entities. Example : @IsUrl() , @ISDatestring()



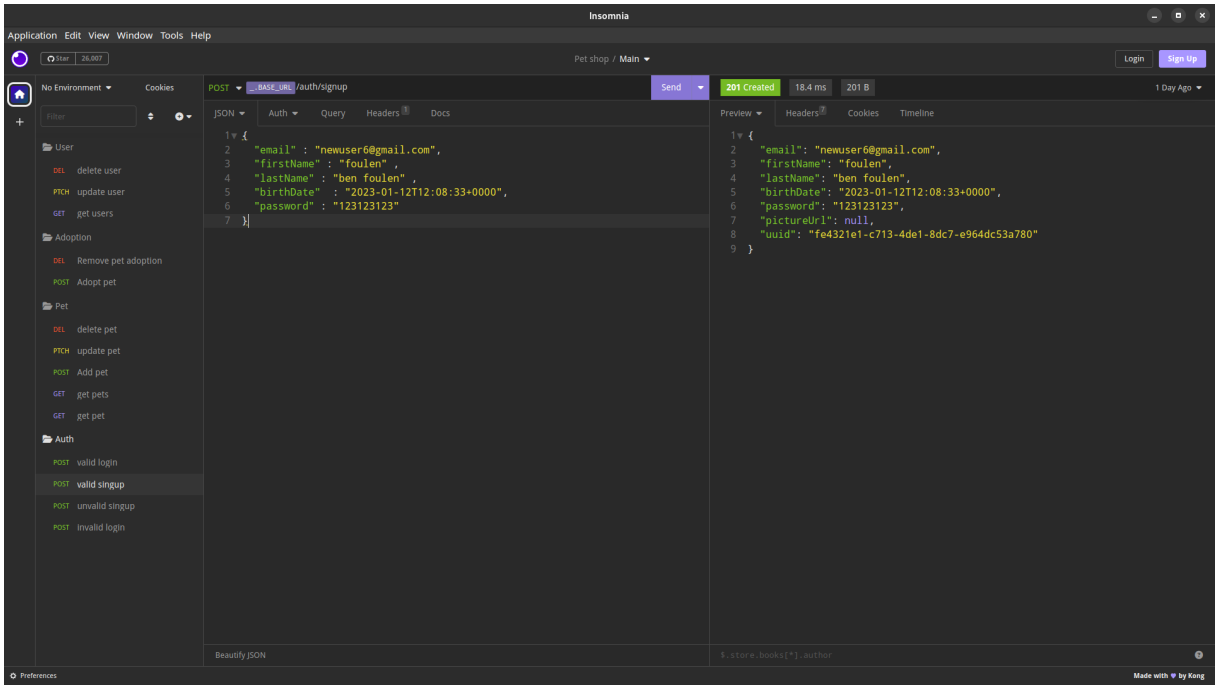
2.6 Database schema



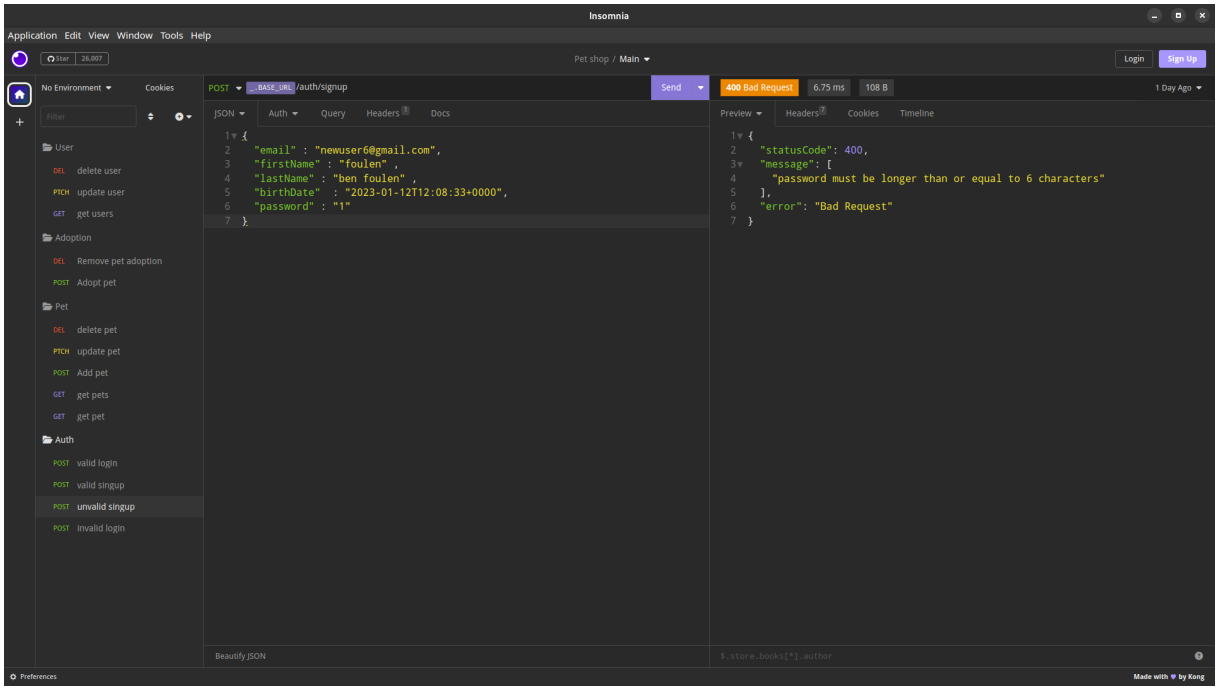
2.7 API routes

2.7.1 Authentication API

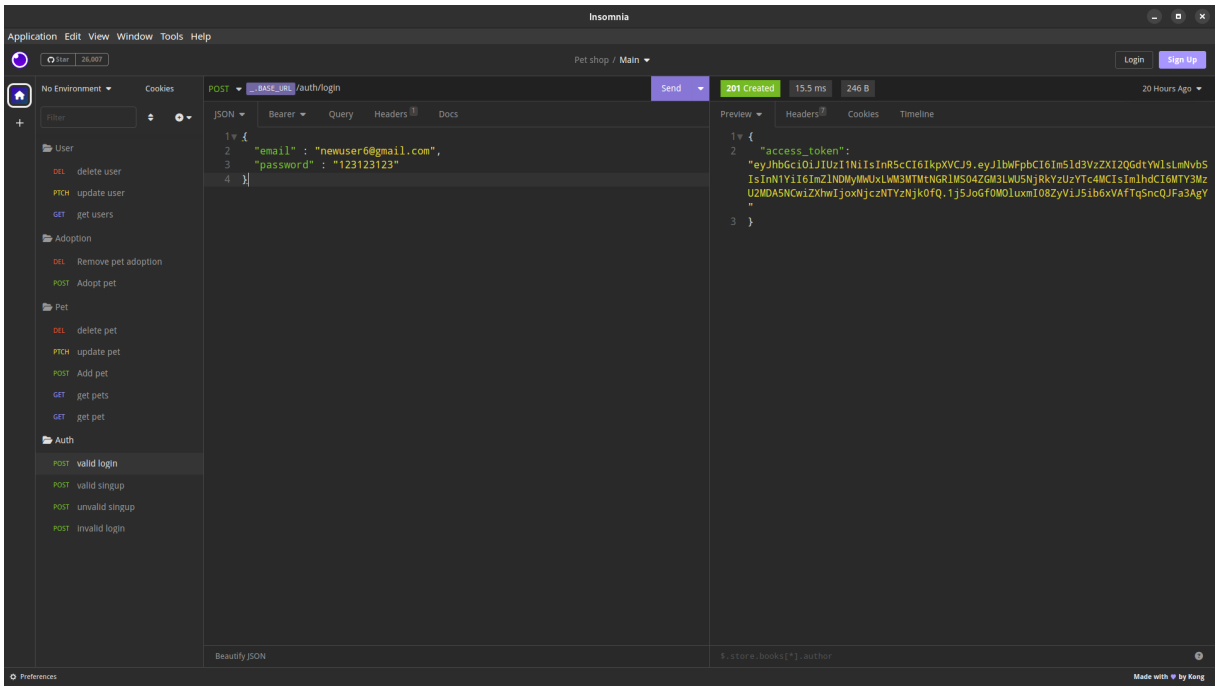
Valid sign-up



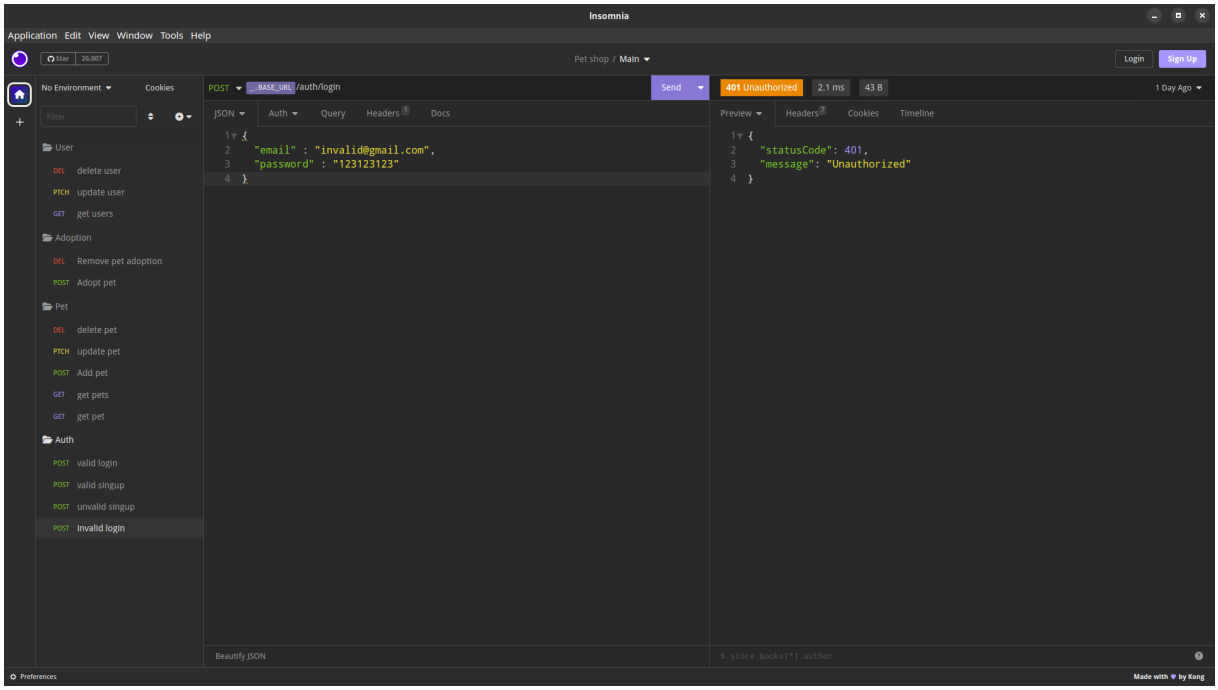
Invalid sign-up



Valid login

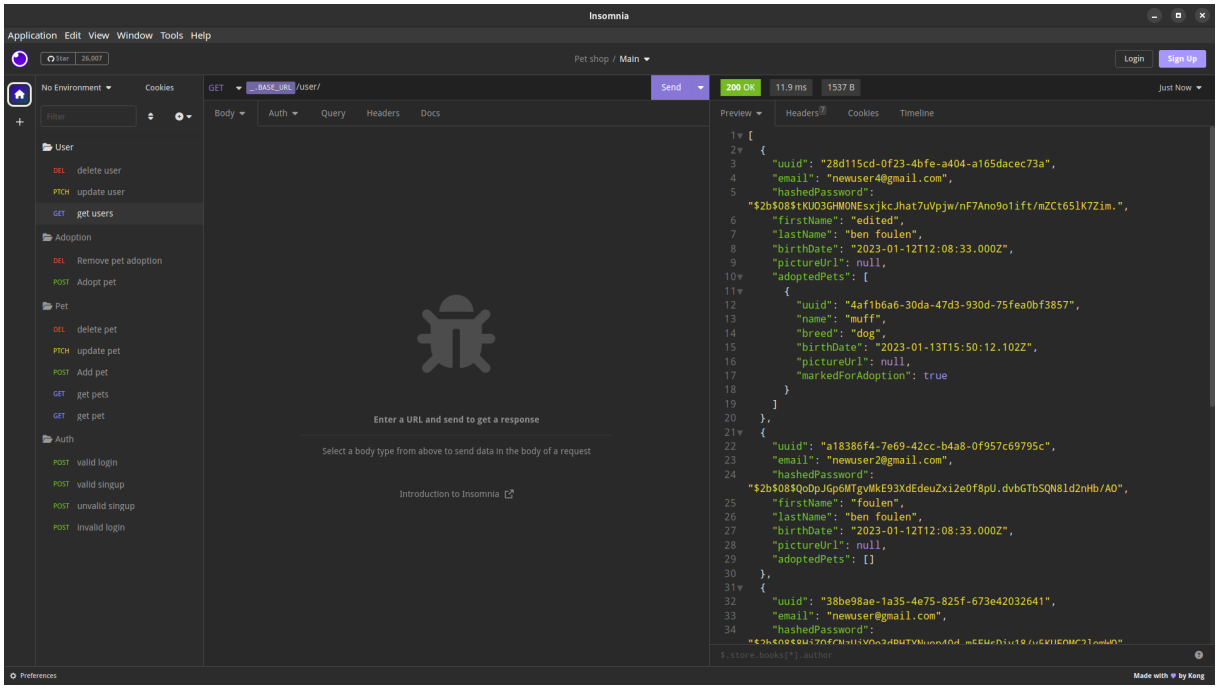


Invalid login

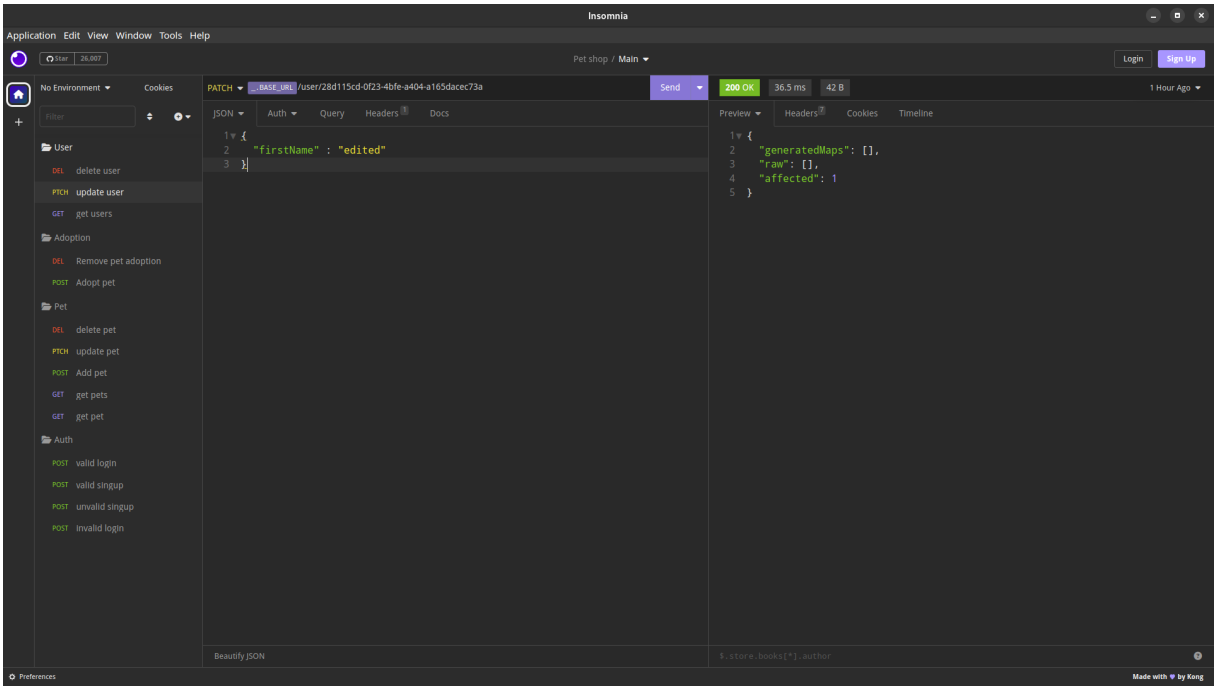


2.7.2 User API

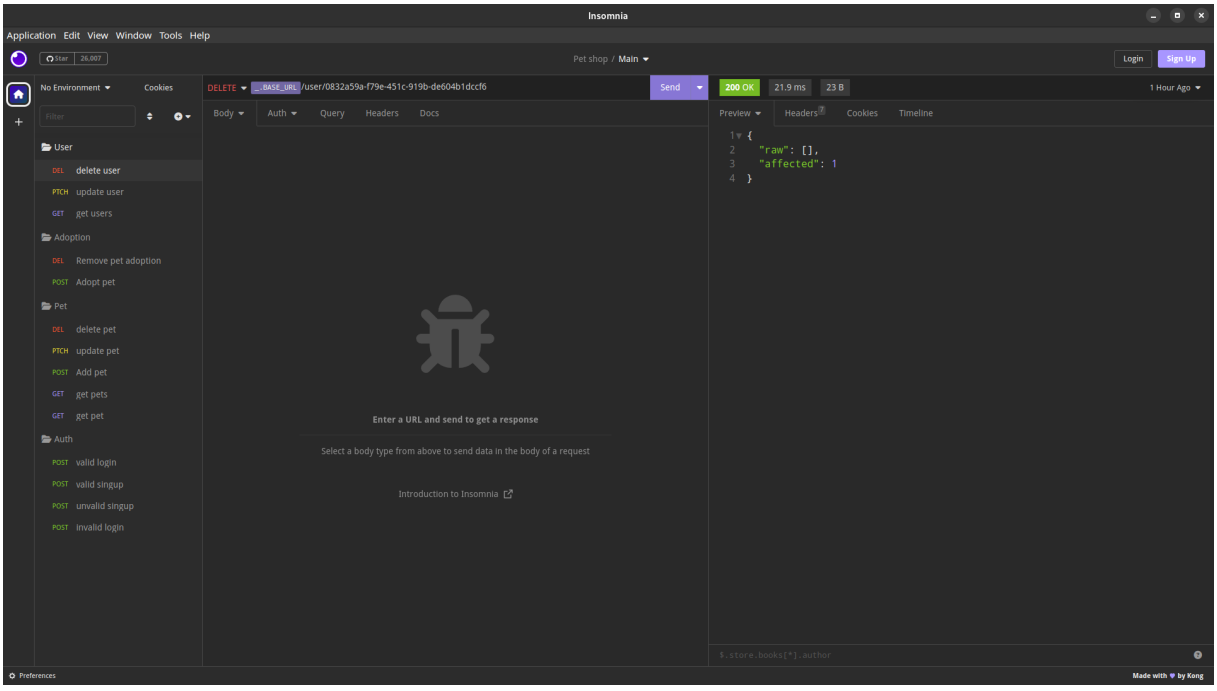
Get all users



Update User

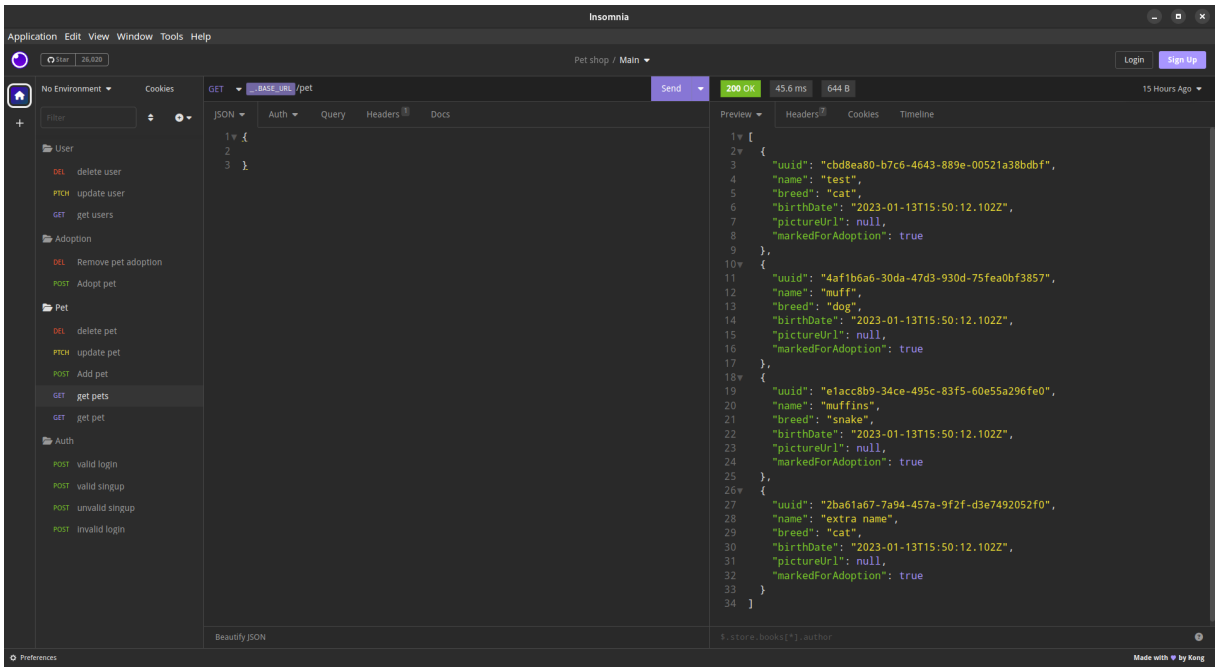


Delete User

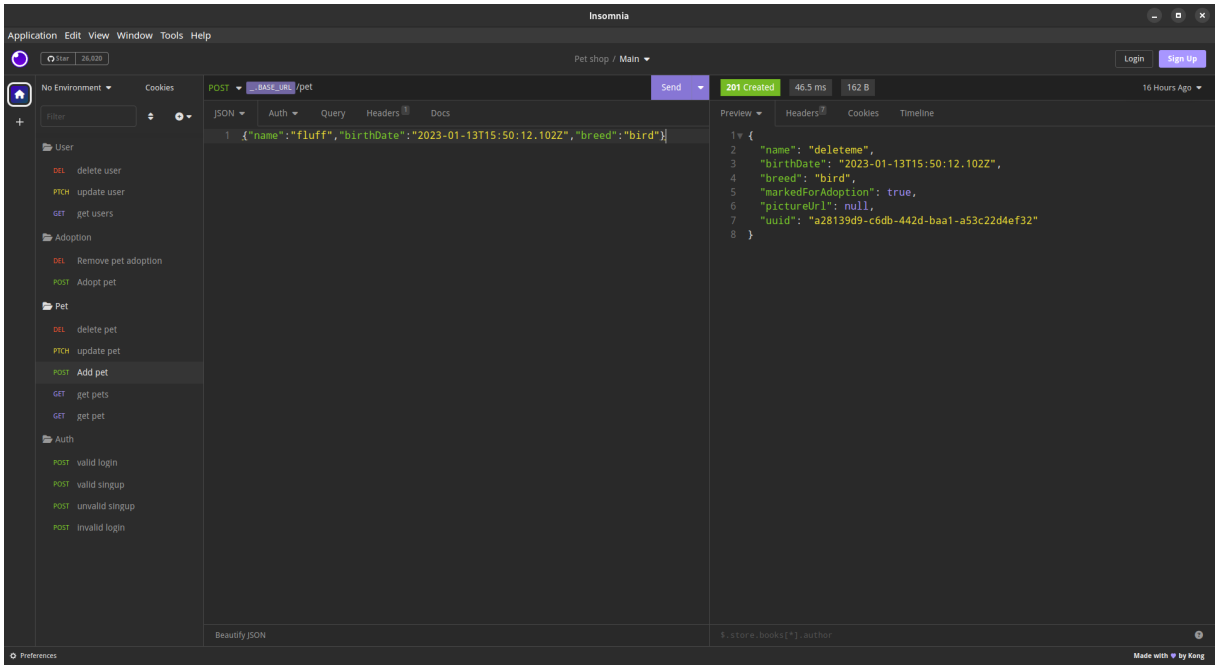


2.7.3 Pet API

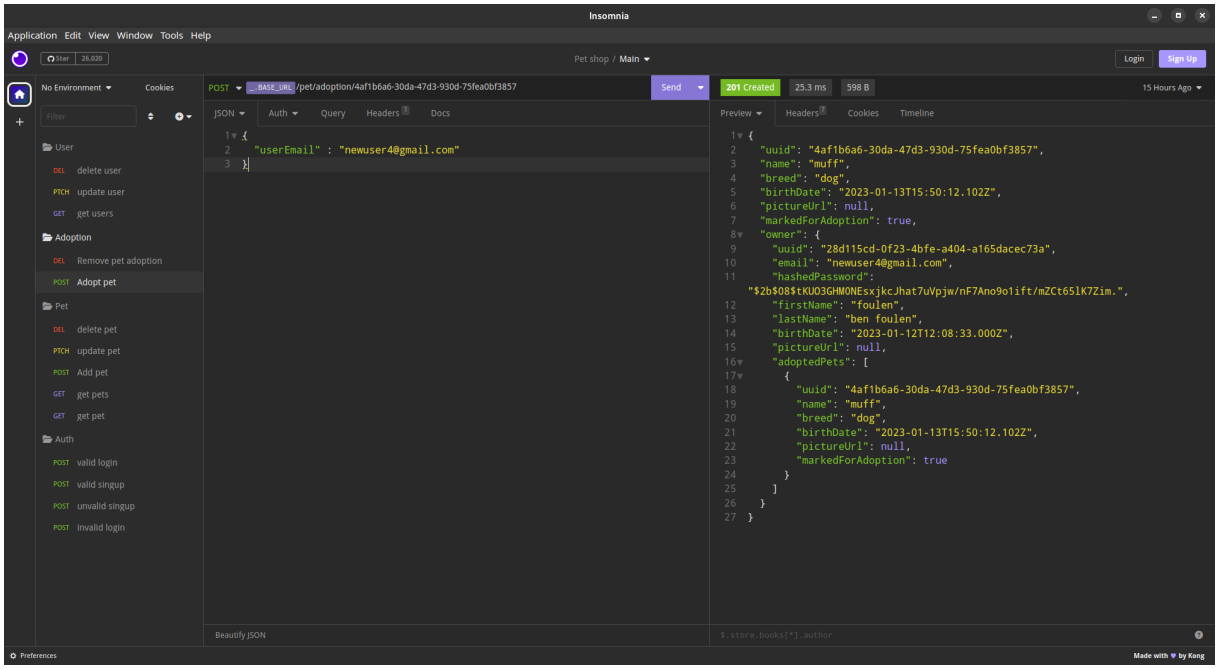
Get all pets



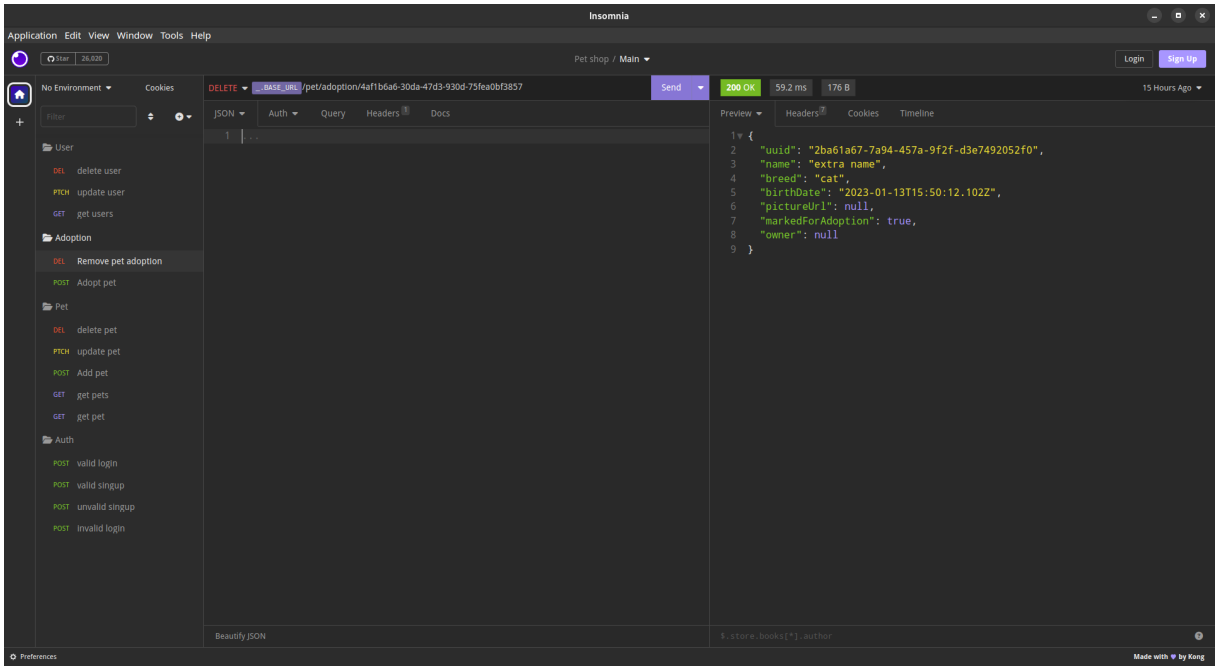
Add Pet



Adopt Pet



Mark pet for adoption



Similar to the User API all other crud operations are implemented with the same api mapping

Chapter 3

Conclusion

To sum up, the demo project on Node.js and The NestJS framework helped me learn how to use the platform and its features to create efficient and scalable web applications.

I also learned about the integration of databases and how to use it for data persistence. Additionally, I learned about creating REST APIs, which allowed me to build applications that can be easily consumed by other systems or devices. I gained a deeper understanding of Node.js and its potential, as well as the added benefits of using a database,the MVC pattern and building RESTful APIs.

Overall, it was a valuable learning experience and I'm excited to continue using and expanding upon the skills I acquired in future projects.

Bibliography