Final Course Project

Web Application: Citizens Medical Situation Record

Submitting: Yonatan Laibel | yonatanla@edu.hac.ac.il | ID: 203654710

Date: 25/07/2023

Task Description

For the final course project, the objective was to develop a web application using AI tools like ChatGPT to aid underdeveloped countries in their vaccination efforts. The application was designed to have two main pages: a registration page where citizens can provide their information and a summary page displaying a tabular view of all existing registration data with a search grid to filter information based on different parameters.

To implement this task, the project utilized PostgreSQL as the database, Java for the backend REST API, and ReactJS for the frontend framework. Additionally, the MUI Framework was employed for designing the user interface, and Docker was used to set up the PostgreSQL Database for easy sharing and deployment.

Previous knowledge

Prior to undertaking this project, I had acquired significant knowledge and experience in the field of computer science through my academic studies and professional work as a Full Stack Developer. During my studies towards a B.Sc. in Computer Science at Hadassah Academic College, I completed a Web Development course that covered HTML, CSS, JavaScript, and Java. Additionally, I supplemented my academic education by enrolling in an online course on React Framework, taught by ***Maximilian Schwarzmüller*** on Udemy. This course enhanced my proficiency in working with React.

My practical experience as a Full Stack Developer at the Israeli Public Broadcast Corporation (Kan11) further solidified my skills in developing full-stack applications using technologies such as React, NodeJS, and .Net. The exposure to real-world projects enabled me to gain valuable insights into various aspects of application development, both on the frontend and backend.

Overall, the combination of academic learning and professional experience provided a strong foundation for tackling the challenges presented in this final course project. The use of AI-powered tools like ChatGPT, coupled with my existing knowledge, allowed me to efficiently develop the web application with a focus on user-friendliness, functionality, and seamless data management.

Used Resources:

Throughout the development of this project, I leveraged various resources to overcome challenges and streamline the process. My previous knowledge and experience with the required frameworks and tools were beneficial, but I also relied significantly on ChatGPT to assist me, particularly in backend development, where my expertise with Java was limited due to minimal exposure during my studies and a prolonged gap since working with the language. Given my full-time work commitments, time was of the essence, prompting me to rely even more on ChatGPT to expedite the project's completion while ensuring its functionality and aesthetics.

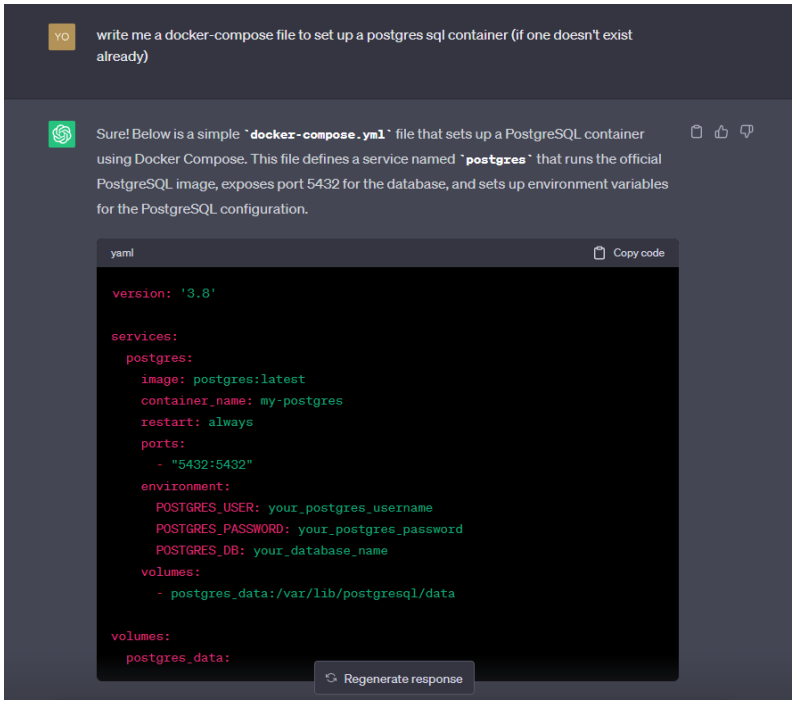
In addition to ChatGPT, I turned to the Stack Overflow website for specific guidance and troubleshooting whenever I encountered issues beyond ChatGPT's scope of assistance. The combination of these resources allowed me to effectively navigate the project's challenges and deliver a successful outcome.

Development Journey:

1. Database

During the development process, I opted to use Docker to streamline the setup of Postgres SQL. Rather than installing the database locally, I created a Postgres SQL container using Docker. To further simplify the process, I composed a docker-compose file. With a simple command like "docker-compose up -d," the container and volume were established, and the database was ready for development. It's important to note that this configuration is designed solely for the development stage. For production, a dedicated server or Kubernetes would be necessary to host the database.

To expedite my work, I sought assistance from ChatGPT for generating the docker-compose file:

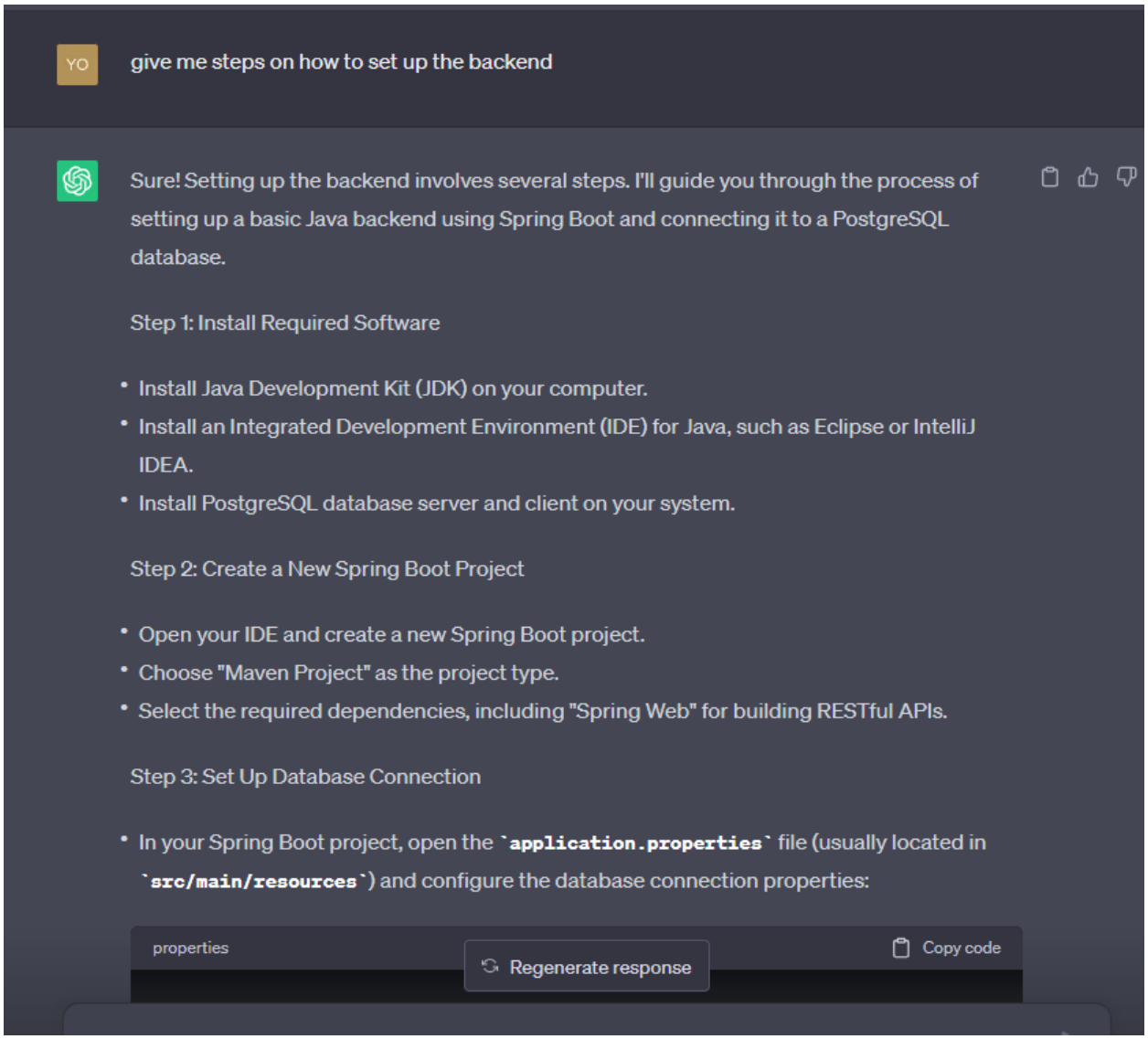


Although I had to customize some parameters in the file to align with my requirements, the solution worked flawlessly from the start.

2. Java Backend API

During the development of the Java backend for my web application, I encountered various challenges that required a blend of knowledge and guidance to overcome. As I had limited experience with Java and backend development, I heavily relied on ChatGPT to provide insights and help me build a robust backend.

The first step was to select the appropriate technology stack. With ChatGPT's assistance, I decided to use Spring Boot for its ease of use, powerful features, and extensive community support. ChatGPT guided me through the setup process and explained the fundamental concepts of Spring Boot, such as dependency injection, annotations, and inversion of control.



With the database in place, I moved on to building the RESTful API. ChatGPT advised me on defining the API endpoints, request/response formats, and error handling mechanisms. It helped me understand the importance of using HTTP methods correctly and designing RESTful APIs that adhere to best practices.

Step 4: Create Entity Classes

- Create Java classes to represent your database entities. For example, for user registration:

java — Copy code

```java
@Entity
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;
    private String email;
    private String password;

    // Constructors, getters, setters, etc.
}
```

Step 5: Create Repositories

- Create Spring Data JPA repositories to handle database operations for your entities. Create a repository for the `User` entity:

java — Copy code

```java
public interface UserRepository extends JpaRepository<User, Long> {
    // Additional custom queries can be added here if needed
}
```

Regenerate response

To implement the API endpoints, I used Spring Boot's powerful feature, the Controller class. ChatGPT provided code examples and explained how to use annotations like @RestController and @RequestMapping to map incoming HTTP requests to the appropriate controller methods.

Step 6: Create Controllers

- Create Spring MVC controllers to handle incoming HTTP requests and interact with the repositories. For example, for user registration:

java Copy code

```java
@RestController
public class UserController {
    private final UserRepository userRepository;

    @Autowired
    public UserController(UserRepository userRepository) {
        this.userRepository = userRepository;
    }

    @PostMapping("/register")
    public ResponseEntity<String> registerUser(@RequestBody User user) {
        userRepository.save(user);
        return ResponseEntity.ok("User registered successfully!");
    }
}
```

As the complexity of the project increased, I needed to handle data persistence efficiently. ChatGPT introduced me to the Java Persistence API (JPA) and guided me through the setup and configuration process. It helped me design entity classes, define relationships between entities, and use JPA annotations effectively to map Java objects to database tables.

Handling validation was another critical aspect of the backend development. ChatGPT provided guidance on using Bean Validation annotations like @NotNull, @NotBlank, and @Valid to ensure that the incoming data is valid and meets the required constraints.

In addition to technical guidance, ChatGPT was instrumental in troubleshooting and debugging. Whenever I encountered errors or issues, ChatGPT helped me analyze the stack traces and provided suggestions on potential solutions.

YO Could not autowire. No beans of 'FormResponseRepository' type found.

The error message "Could not autowire. No beans of 'FormResponseRepository' type found" typically indicates that Spring is unable to find a bean of type `FormResponseRepository` to be autowired into your class.

To resolve this issue, make sure you have followed these steps:

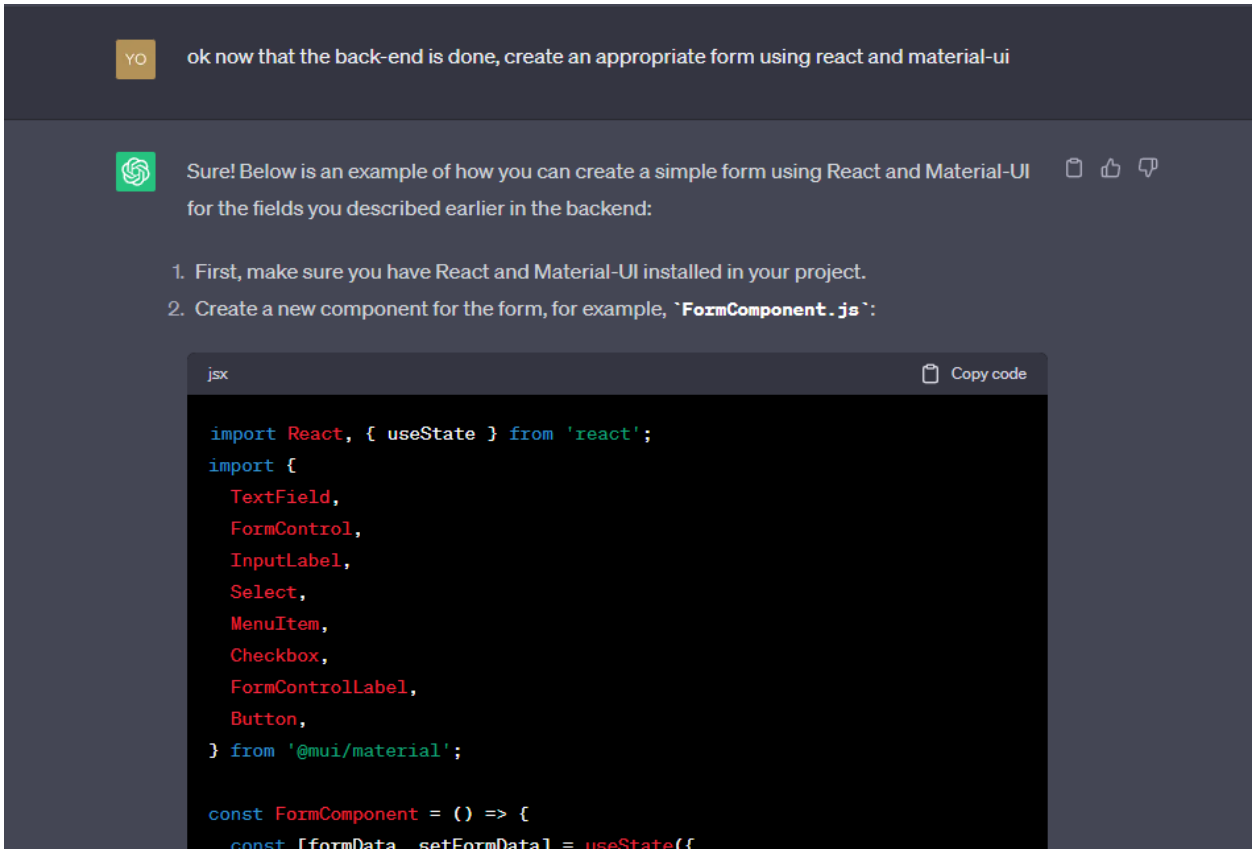
1. **Repository Annotation:**

3. React Front End

For the frontend development of the web application, I chose to use Vite as the build tool and development server. With prior knowledge in HTML, CSS, and JavaScript from my academic studies and a React course on Udemy, I had a solid foundation in frontend development. However, to expedite the process and achieve a visually appealing design, I sought guidance from ChatGPT.

Using Vite, I set up the frontend project and took advantage of its fast development and build process, enabling me to streamline the frontend development workflow. With ChatGPT's assistance, I efficiently integrated Vite with React, allowing for a smooth and productive development experience.

Firstly, I leveraged the MUI (Material-UI) Framework to enhance the overall look and feel of the application. This framework offered a vast collection of pre-designed components that could be seamlessly integrated into the app, providing a more consistent and modern user experience. ChatGPT aided in utilizing various MUI components effectively, allowing me to create responsive and aesthetically pleasing UI elements with ease.



Furthermore, I encountered challenges while implementing data validation and error handling on form inputs. With guidance from ChatGPT and using the React Hook Form library, I effectively implemented client-side validation for user input fields, ensuring that data submitted by users met the necessary requirements and preventing invalid entries.

To enhance user experience, I implemented features such as autocomplete in the city input field, allowing users to filter and select their city from a predefined list of cities. ChatGPT provided insights on integrating this feature seamlessly and making the application more user-friendly.

Lastly, I implemented pagination in the Results page to manage the display of large datasets efficiently. With ChatGPT's assistance, I incorporated the MUI DataGrid component, which allowed for dynamic pagination, enabling users to select the number of rows to be displayed per page and navigate through the dataset effortlessly. The MUI DataGrid component also allows to filter single/multiple columns, like in excel.

Throughout the frontend development process, I utilized ChatGPT as a valuable resource to overcome hurdles, gain insights, and efficiently implement various UI components and functionalities. It proved to be an invaluable tool in enhancing productivity and achieving a visually appealing and user-friendly web application.

4. Conclusion

The successful completion of the web application aimed at facilitating vaccine registration in undeveloped countries is attributed to the harmonious collaboration of my existing frontend development skills, Vite build tool, and ChatGPT's guidance. The frontend project, powered by Vite, complements the Java backend effectively, resulting in a cohesive, visually pleasing, and responsive user interface.