Capstone Project Wrap-Up Document – API Team

**Application Overview:**

This application is a jobs platform which is intended to provide an extensible and easy-to-use system which hiring managers can use to create and edit job listings. Likewise, users can register as candidates to apply to these listings, whereupon managers will have easy access to their cover letter and other relevant hiring information.

The technology stack utilized on the backend, our team’s area of responsibility, is as follows:

* Spring Boot using Java as the framework used for building the application’s routes and logic.
* Gradle used for dependency management and the Java build process.
* MySQL used to persist user, job and application information in a database.
* Git used for version control, alongside GitHub.
* JUnit and Postman used for unit and backend integration tests.

The implemented solution provides a variety of endpoints to create, read, update and delete users, job listings and applications, with this data being amendable and displayable from the application’s frontend. The interconnection of the various database models (such as job application and job candidate) makes it easy and quick to view all relevant data for a job listing.

**Status:**

* /registration endpoint – Implemented
* /login endpoint – Implemented
* /users
  + GET all users endpoint – Implemented
  + GET by ID endpoint – Implemented
  + POST new user endpoint – Implemented
  + PUT user update by ID endpoint – Implemented
  + DELETE by ID endpoint – Implemented
* /candidates
  + GET all candidates endpoint – Implemented
  + GET by ID endpoint – Implemented
  + POST new candidate endpoint – Implemented
  + PUT candidate update by ID endpoint – Implemented
  + DELETE by ID endpoint – Implemented
* /manager
  + GET all managers endpoint – Implemented
  + GET by ID endpoint – Implemented
  + POST new manager endpoint – Implemented
  + PUT manager update by ID endpoint – Implemented
  + DELETE by ID endpoint – Implemented
* /jobs
  + GET all jobs endpoint – Implemented
  + GET by ID endpoint – Implemented
  + GET all jobs by manager ID - Implemented
  + POST new job endpoint – Implemented
  + PUT job update by ID endpoint – Implemented
  + DELETE by ID endpoint – Implemented
* /applications
  + GET all applications endpoint – Implemented
  + GET by ID endpoint – Implemented
  + GET by Job ID endpoint – Implemented
  + GET by Manager ID endpoint - Implemented
  + POST new application endpoint – Implemented
  + PUT application update by ID endpoint – Implemented
  + DELETE by ID endpoint – Implemented

Some elements connected to a future AI-enabled application/candidate ranking system have also been implemented. Specifically, a new database table for application scores has been created, as well as endpoints for accessing the scores either via their own ID or that of the Application ID to which they are associated. Further implementation in future would necessitate the creation of a connection with an existing AI service to send the user’s submitted application for evaluation.

**Build Instructions:**

1. The application can be cloned from GitHub using the following git command:

git clone <https://uq23711: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@github.com/uq23711/ADP-Training-Group-Project-Backend.git>

1. Having cloned the project into a directory of your choice, the next step to ensure that the backend can be run is to install MySQL locally and create a new database. You will likely be required to define a username and password for MySQL. Please note these down, as well as the name of the database you created, for the next step.
2. Next, it is necessary to define environment variables so that the connection between the Spring Boot backend and the MySQL database functions. The following environment variables must be defined using a method appropriate to your operating system:
   * 1. SPRING\_DATASOURCE\_USERNAME – The username for the MySQL user that created the database which will be used for the backend.
     2. SPRING\_DATASOURCE\_PASSWORD – The password for the user which created the database.
     3. SPRING\_DATASOURCE\_URL – The URL for the database you created. For example, this may look something like: jdbc:mysql://localhost:3306/[YOUR DATABASE NAME]
3. From the command line in the project’s root directory, run the command:

gradle clean build

And then the following command:

java -jar build/libs/talent-api-0.0.1-SNAPSHOT.jar

This should start the backend server.

**Personal Comments:**

**Sam**

One thing which I think worked particularly well over the course of this project was how we divided up the work amongst ourselves. The number of endpoint groups we had to develop was perfectly matched with the number of team members, allowing all of us to take responsibility for a group of endpoints. As owner of the repository I also chose to add branch protections to avoid merging changes directly into the main branch. I decided to require 2 approvals for all pull requests into main, helping us to catch errors before they got merged in.

In my opinion we worked well together as a group overall and didn’t have any issues communicating amongst ourselves. However, earlier on in the project we could have communicated a little more frequently with the frontend team about some of the changes we made. Nevertheless, we did improve in this respect over the course of the project. We had some early technical issues in getting MySQL fully set up, particularly in the context of working across development environments (inside and outside the VM). Some small technical issues also occurred with CORS, but overall we didn’t face any technical issues that took a long time to resolve.

In terms of what I’d do differently if starting this project over again, I would redesign the SQL database models. I felt that some of the database models contained redundant information. For example, both the Job and Manager models contained information such as ‘department’, but the relationship between the two made it seem more logical that only one of these models contain that information. The use of different models for User, Candidate and Manager was also something which I felt could have been simplified.

**Ruth**

I am very happy with how the project went for our team. The way we split up the work felt fair and manageable. In addition to this, because each person had their own set of endpoints, it also meant that we were able to practice the things we learnt during the labs, such as how to create a domain, controller, and repository, as well as how to test them.

I think the way we worked together on the backend went well too. By hosting a call on Webex every day, we had the opportunity to speak to one another about how we were getting on and bring up any issues that we were having. Furthermore, it also provided a space for us to notify everyone about PRs which had been made and whether code had just been pushed to main.

As with every project, there are some things which could have been improved. While I feel the communication between the members of our team was sufficient, I believe that there was room for improvement when communicating with the frontend team. While we had a couple of meetings to discuss some endpoints, it would have been beneficial for both teams to have met earlier in the project to decide how all the endpoints would be implemented. This would have meant that both teams knew exactly what to work towards, which would have potentially saved some time and hassle later with integrating the backend with the frontend.

Overall, if I were to do this project again, I would just clarify key details from the start of the project and ensure that there would be a space where we can all communicate together easily.

**Luca**

In my opinion building the talent acquisition site was a very good practise to simulate working in a team on developing an application. We were able to split the number of Endpoints perfectly, so each group member had one to deal with. In the beginning some of us faced some issues connecting to the MySQL Database but we managed to solve the issue. The overall development was quite smooth, and team communication was well handled by Sam. I would say the most challenging part for me was thinking about logical connections between each database to store and access the data correctly. Other than that, we faced some issues running the application with docker which at the end we didn't manage to fix. With more time for working on the project we would be able to solve the issue in my opinion. Additionally, we could separate each endpoint using a dedicated service class that handles logical operations to make the architecture cleaner.

**Mathieu**

This backend development project using Spring Boot has been a very enriching experience. Since web development is not my primary field, I had to adapt quickly to new technologies and concepts. This was a challenge, but also an incredible opportunity for learning. Thanks to the training sessions I attended, I was able to effectively apply my knowledge and master tools like Spring Boot and Git.

Despite the amount of work to be done, we successfully divided the various endpoints among ourselves, which I particularly appreciated. The good rapport within the team, led by our captain Sam, was a major asset. This effective coordination allowed us to work harmoniously and overcome challenges together.

**Sokhna**

When I first began working with Spring Boot, I found many of the foundational concepts complex and difficult to apply. Understanding principles like dependency injection, configuring Spring Data JPA repositories, and structuring RESTful API endpoints was challenging. However, this project provided a hands-on experience that allowed me to reinforce these concepts, moving from theory to actual practice and helping me significantly improve my understanding and skills with Spring Boot.

One of the main challenges I faced was managing dependencies between different parts of the application, especially because my assigned task involved the application endpoints, which depended heavily on entities like job listings, candidates, and managers. Every time changes were made to another part of the system, I had to update my endpoints to ensure they remained compatible. This led to frequent debugging sessions and occasional test failures, particularly when tests that were passing one day would fail the next due to updates in other sections of the code. Debugging these issues required a deep understanding of the application's structure and close collaboration with my teammates to manage dependencies, especially when a new feature or fix affected other services.

Additionally, as the project progressed, I encountered further hurdles with tools like JUnit for testing and with configuring complex Spring Data JPA relationships. I had to adapt my endpoints continuously to fit the evolving backend logic, ensuring they worked seamlessly with database updates and other service changes. Learning to use Git in a collaborative environment was another valuable skill I developed, especially with managing pull requests, handling merge conflicts, and ensuring that my contributions didn’t disrupt others' work.

Through this project, I learned to adapt to changing requirements, debug complex issues, and manage dependencies across interconnected services. By the end, I felt a much stronger grasp on structuring, testing, and refining Spring Boot applications. This experience has boosted my confidence in backend development, making it a challenging but immensely rewarding project that I am proud to have been part of.

**Norbert**

During our talent acquisition project, we effectively planned the API structure by assigning specific endpoint groups to each team member. Clear communication was crucial : daily Webex calls allowed us to discuss progress and challenges, through earlier engagement with the frontend team could have streamlined integration. We used Github to maintain code quality, implementing branch protections and requiring two approvals for pull requests, which helped catch errors early. While we encountered initial MySQL setup issues, particularly with configuration across environments, our team resolved them swiftly, allowing us to stay on track.

It was a pleasure working with all of you, thank you for your hard work and collaboration.