Spring-Boot Student Grade Management Program

Joshua Reyes

101042544

Joshua.Reyes.2023@live.rhul.ac.uk

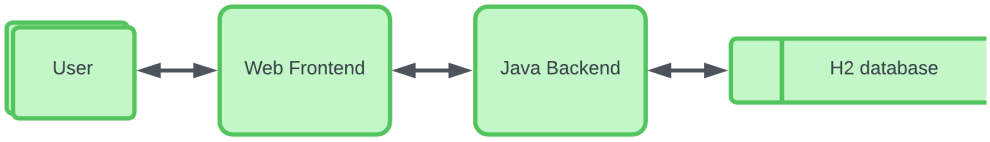
https://gitlab.cim.rhul.ac.uk/zmac312/CS2800

Section 1. Technical Description

The Student Grade Management system was built to house student records, grades and modules related to those modules.

This System was built using multi-level architecture with Java-script and CSS used to create the Web Frontend, and Spring-Boot, Maven and Java to create a working backend that utilised a h2 memory SQL database. All of the backend code was pushed through a continuous development cycle that is tracked on GitLab.

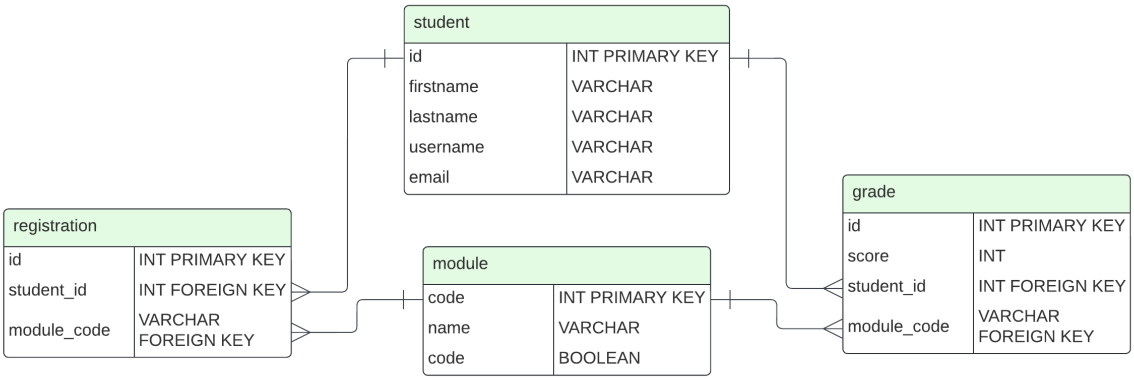
Data Flow:



Above the Data flow of the system is modelled. It Illustrates that when A user enters a set of data into the system it will send all requests through the java backend that models the CRUD repositories and REST operations, these operations will send requests to the H2 database. Then the information is saved back to the CRUD repos and then to the frontend to be viewed.

An example of one of these REST operations would be a Student GET, which would get the student records. Another example of this would be Grade POST (adding a Grade record), this required a Java dependency injection as to post a students grade it would need its related Module and related Student, so the java backend handles that specific request.

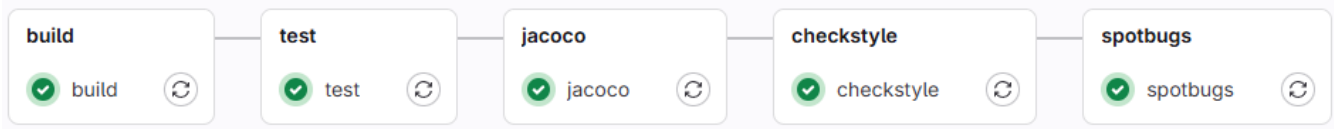
The Database Model:



The Database model above shows the logical relations between each object, these objects are modelled by objects coded in java within the first part of the project (Coursework 1). Although the registration object cannot be utilised on the web frontend it is still modelled here as it was a key part of the system before development into a full web system.

These Java Objects are instantiated in the system not by the backend but as beans used within the Spring Framework to actually represent the H2 database.

Deployment and Maintenance



The system is deployed through a CI/CD pipeline that is active on GitLab, this ensures that no bugs or errors, and that all code is covered by a test. None of which will make it to the safe build that has been deployed on the main branch on Gitlab.

All work issued and then was branched remotely: coded, tested and built there, then pushed safely to Gitlab where in which all changes where then put through the CI/CD pipeline. All Issues that track development are on the Gitlab.

Section 2. Technical Challenges

This section will show some of the technical difficulties I had when implementing this system and joining It to the frontend web service

Challenge 1: Early Mapping error (related Git Issues: #4, #10)

Situation: My code was throwing a somewhat random ApplicationContext error in my application class which resulted in an IllegalStateException

Task: I had to figure out this error as this was throwing me off completely

Action: I searched through the online labs to see if this error was encountered there, I also looked through other sources both, within the online Q&A forum and Stack Overflow to find the solution to this error, I didn't find anything that really helped me but some sources still guided me to the right course of action to take. I eventually figured out it was a mapping error between student and registration I made the change to it creating a "module_reference" variable that would house the ID of the module (little did I know that was an error in itself that would pop up later, in *Challenge 2*).

Result: When building it would no longer had an IllegalStateException

Challenge 2: Major Refactor (related Git Issues: #4,#9,#10)

Situation: When Implementing the GradeController class in #9 I ran into a fundamental error with the architecture of the system as a whole. The aggregation between: module and grade and module and registration were completely wrong. this was proved even further when the feedback from the first part of the project had released.

Task: Rework System architecture and add back the correct aggregation

Action: This called for a complete rework of the systems architecture, this meant firstly changing registration and grade to store a Module object instead of a "module_reference". Next, I had to refactor all related setters and getters, this also meant changing the tests created in the first part of the project. This is all shown in issue #10

Result: this all resulted in me being able to actually make the Grade Controller class properly.

Challenge 3: configuring CI/CD (related Git Issue: #12)

Situation: Near the end of the project, I realised whilst checking up on a few things I realised my CI/CD wasn't correctly configured, this could mean the whole project could be improperly built within Gitlab.

Task: correctly fix Gitlab CI/CD

Action: As I configured it the first part of the project the CI/CD was configured to CW1. after supposedly fixing the issue, I had to double check the build by pushing a commit to see if it would build correctly.

Result: CI/CD was correctly configured.

Section 3. Retrospective

This section will reflect on the project as a whole.

What went well

I overall think this project has given me a good grasp as to what its like in a real company environment, and what it is like to actually software engineer, as well as the continuous development life-cycle. I overall enjoyed working with Gitlab and maven projects. I also think this enabled me to learn a lot about Spring-boot and databases. This project also helped teach me on some greater Software engineering concepts.

Reflection on Challenges

I think some challenges such as challenge 3 where due to me not considering some things when initially setting up the maven project, whereas some other challenges like the major rework and refactor are just a simple factor of engineering a system like this. It was inevitable that there would have to be some things I would have to refactor within the system, here it was the mapping and a great aggregation error that was made within the first part of the Coursework.

Areas for development

If I where to further develop this system I would definitely add the post mapping/ controller for Registration section of the database, and next maybe add login and some sort of authentication scheme. However, there are any other ways this system could develop into something much greater.

Conclusion

Overall, I feel I have greatly improved as a programmer and learned a lot about life cycle of projects with this project, as well as learning a lot about the spring framework and maven projects and especially Gitlab.