CSCI 5308

ADVANCE TOPIC IN SOFTWARE DEVELOPMENT CONCEPTS

Brownie Point

Group 03

Project Report

1. Objective:

Brownie Point is a mobile application designed to revolutionize the way teachers and students interact in the classroom. This React Native-based app provides a seamless user experience, allowing students to register and log in, view their enrolled courses, while teachers can log in, edit courses, and award points to students by scanning QR codes of each student. With Brownie Point, educators can effortlessly monitor student progress and encourage engagement, while students can stay on top of their coursework and track their progress with ease.

2. Requirements and design considerations for the Brownie Point application:

Requirements:

1. User Roles: The application should support two user roles - *Teachers and Students.*
2. *Registration and Login*: Students should be able to register and login to the application. Teachers should be able to login to the application.
3. *Course Management*: Teachers should be able to add and delete courses.
4. *Point Management*: Teachers should be able to assign points to students for a specific course by scanning the QR code generated by the student.
5. *Profile Management*: Students should be able to view their profile.
6. *Course Enrollment:* Students should be able to see the list of courses they are enrolled in and their corresponding points.

User Interface Design:

1. The application has a simple and intuitive interface.

2. The login and registration screens are easily accessible and easy to use.
3. The course management screen should allow teachers to add and delete courses easily.
4. The point management screen should allow teachers to scan the QR code and assign points to students easily.
5. The student profile screen should allow students to view.
6. The course enrollment screen should allow students to view the list of courses they are enrolled in and their corresponding points easily.

Technology Stack:

1. Frontend: React Native.
2. Backend: JAVA Springboot
3. Database: MYSQL
4. Authentication: By Using JSON Web Token (JWT).

3. The following dependencies are essential for the functioning of this project:

React Native - React Native is an open-source mobile application framework created by Facebook. It is used to develop applications for Android, iOS, and Web using React, which is a popular JavaScript library.

Redux - Redux is a predictable state container for JavaScript apps. It helps to write applications that behave consistently, run in different environments (client, server, and native), and are easy to test.

React Navigation - React Navigation is a library that helps to handle navigation in React Native apps. It provides a consistent navigation experience across different platforms, including iOS and Android.

Axios - Axios is a popular JavaScript library used to make HTTP requests from the browser or Node.js. It provides an easy-to-use API for making AJAX requests and handling responses.

React Native Elements - React Native Elements is a library of UI components for React Native. It provides pre-designed UI elements, including buttons, forms, icons, and more, to speed up the development process.

React Native Vector Icons - React Native Vector Icons is a library that provides a set of customizable icons for React Native apps. It includes icons from popular icon sets

React Native Image Picker - React Native Image Picker is a library that provides an easy-to-use interface for selecting images from the user's device or camera. It supports both iOS and Android

Spring Boot Starter Data JPA: This dependency provides the Spring Data JPA library, which simplifies the implementation of JPA-based repositories.

Spring Boot Starter Security: This dependency provides the Spring Security framework, which allows for authentication and authorization in the application.

Spring Boot Starter Web: This dependency provides the Spring Web MVC framework, which enables building web applications and RESTful APIs.

Spring Boot Starter Mail: This dependency provides the Spring Framework's email sending capabilities.

Lombok: This is a utility library that provides annotations to simplify the implementation of common Java boilerplate code, such as getters and setters.

JWT: This dependency provides the JSON Web Token (JWT) implementation for authentication and authorization in the application.

Spring Boot Starter Test: This dependency provides testing tools for the Spring Boot application, such as JUnit and Mockito.

MySQL Connector/J: This is the MySQL JDBC driver that enables communication with a MySQL database.

JAXB API: This is the Java Architecture for XML Binding API, which provides functionality for converting XML data to and from Java objects.

JUnit: This is a testing framework for Java applications.

OpenCSV: This is a library for reading and writing CSV files in Java.

Annotations: This dependency provides JetBrains annotations that can be used to improve static code analysis and documentation generation.

4. Build/Deployment:

Here are the steps to set up a Brownie point React Native mobile application for the frontend and Spring Boot in the backend using JWT tokens for responses:

Backend:

1. Create a new Spring Boot project using your preferred IDE, we choose Intellij.
2. Add the necessary dependencies to your pom.xml file, including spring-boot-starter-data-jpa, spring-boot-starter-security, junit ,jjwt-api, spring-boot-starter-web, jjwt-impl, jjwt-jackson, lombok, spring-boot-starter-test, mysql-connector-java, jaxb-api, opencsv, annotations, spring-boot-starter-email, spring-boot-maven-plugin.

Frontend:

1. Create a new React Native project using the npx react-native init command.
2. Add the necessary dependencies to your package.json file, including react-native, react-redux, redux-saga, axios, native-base, react-navigation, and jest.
3. Create the necessary screens and components for the login and registration forms, teacher and student flow.

4. Implement the necessary functionality to send HTTP requests to the backend API endpoints for login and registration forms, teacher and student flow, including storing and retrieving JWT tokens.
5. Build and deploy our frontend application on emulator by following the steps outlined in the prompt.

CI/CD:

Thought Process of Implementing CI/CD (We have implemented complete CI/CD including deploying the code on server and running from here.)

The idea behind CI/CD is to automate the process of building, testing, and deploying software changes so that developers can focus on writing code instead of worrying about the logistics of deployment.

The file starts with some variables that will be used throughout the process, such as the Maven options and the container image. Then, it defines the different stages of the pipeline that the code will go through.

The first stage is called "smells," The idea behind this stage is to use a tool called DesigniteJava to analyze the code and detect "code smells," which are patterns that could indicate deeper issues with the code. This stage is responsible for detecting code smells and generate the terminal output as shown while running Designite.jar file in our system. The artifacts have been used to pass on the smells generated in this stage.

The next stage is called "issue," and it depends on the "smells" stage. This stage uses Python to read the csv files developed on running the smells stage and generate the code issues in the issues section of the Gitlab in the tabular form.

The third stage is called "build," and it uses Maven image described in the image tag to build the maven project from source code.

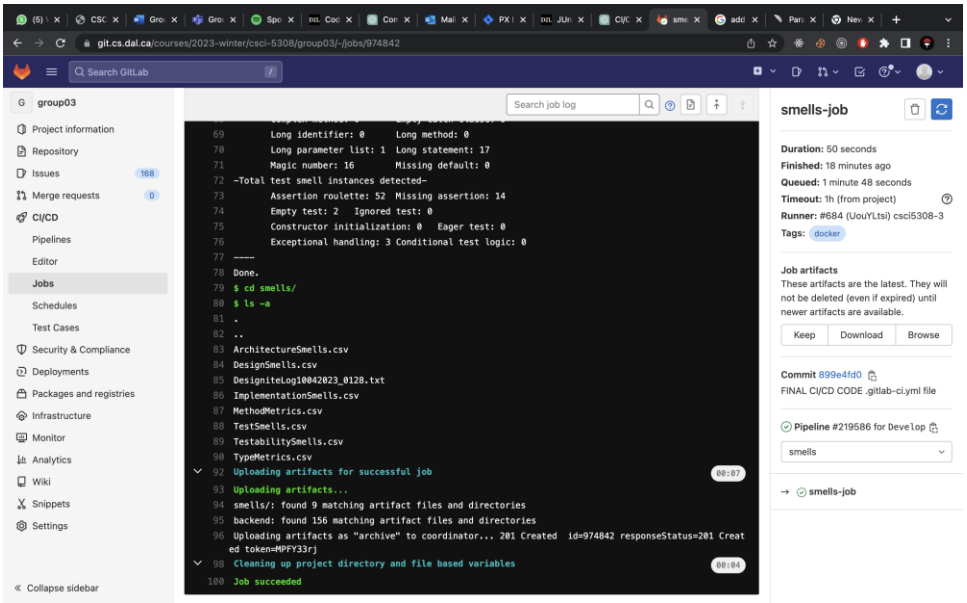
The fourth stage is called "test," and it uses Maven image described in the image tag to run the project's test suite.

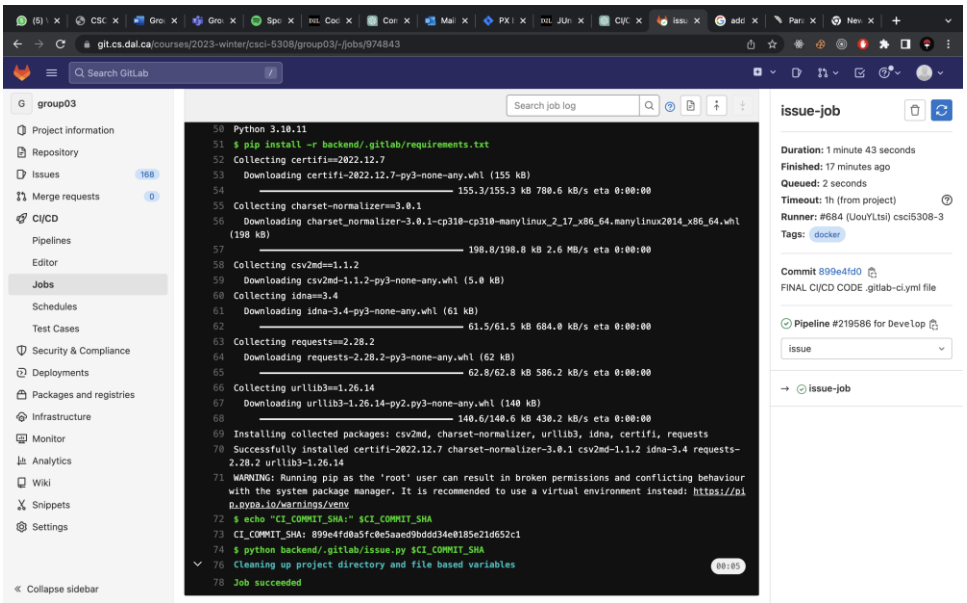
The fifth stage is called "package," and it uses Maven image to package the project into a JAR file. This JAR file is then stored as an artifact of the pipeline, meaning that it can be accessed and used later in the pipeline.

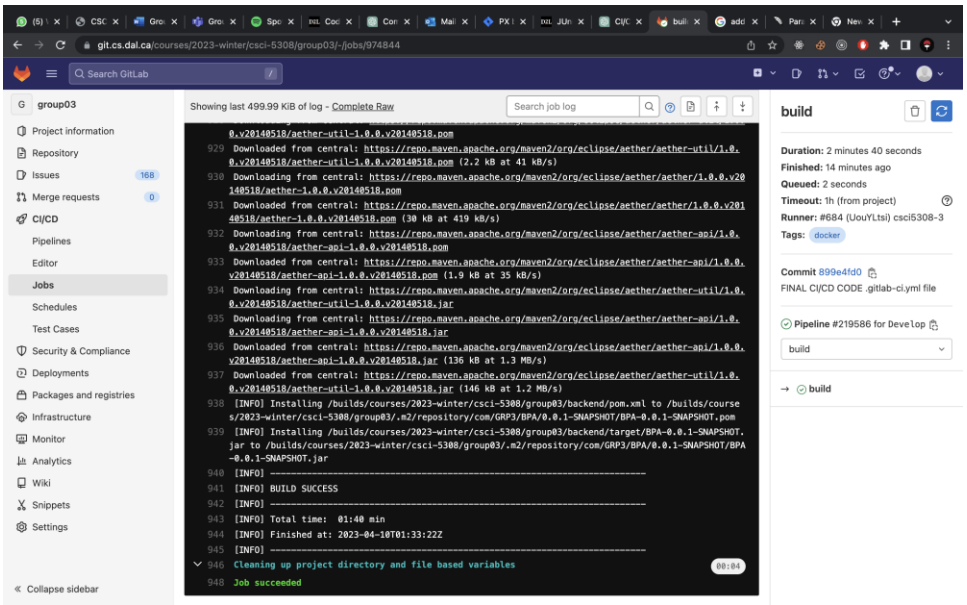
Finally, the sixth stage is called "deploy," and it uses Docker to deploy the application to a remote server. To store the docker images we had multiple options like Docker Hub, Gitlab container registry and many others but we decided to go through GitLab container registry. This stage depends on the "package" stage, since it needs the JAR file that was created in that stage. The stage starts by building a Docker container and pushing it to a Docker registry. Then, it uses SSH to copy the JAR file to the remote server or virtual machine, pull the Docker container, and run it on the remote server. The SSH key generated is of ECDSA type rather than RSA because it was mentioned in the lab that the ECDSA type of SSH key has more compatibility with the GitLab runners. But sometimes private SSH key that we copy in Gitlab CI/CD variables can be malformed and it can be any reason like extra space or missing characters, so we use base64 encoding to

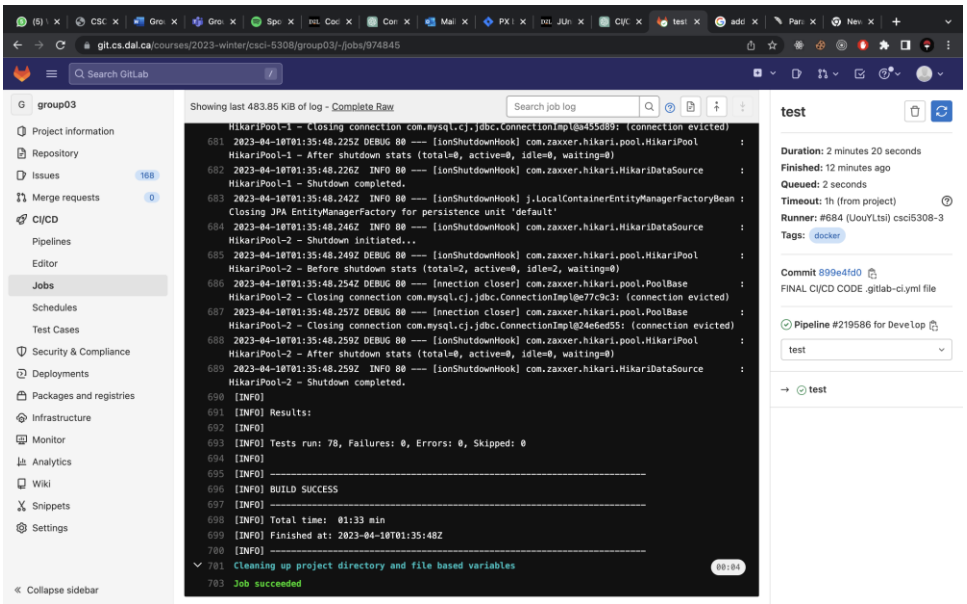
make sure the key is formatted properly. One thing to note is that while running the docker image on virtual machine, once the specific port has been assigned to the docker image, we cannot do the continuous deploy on that port again because it would be already assigned, that's why we need to remove the container running on the port and then run the new docker container.

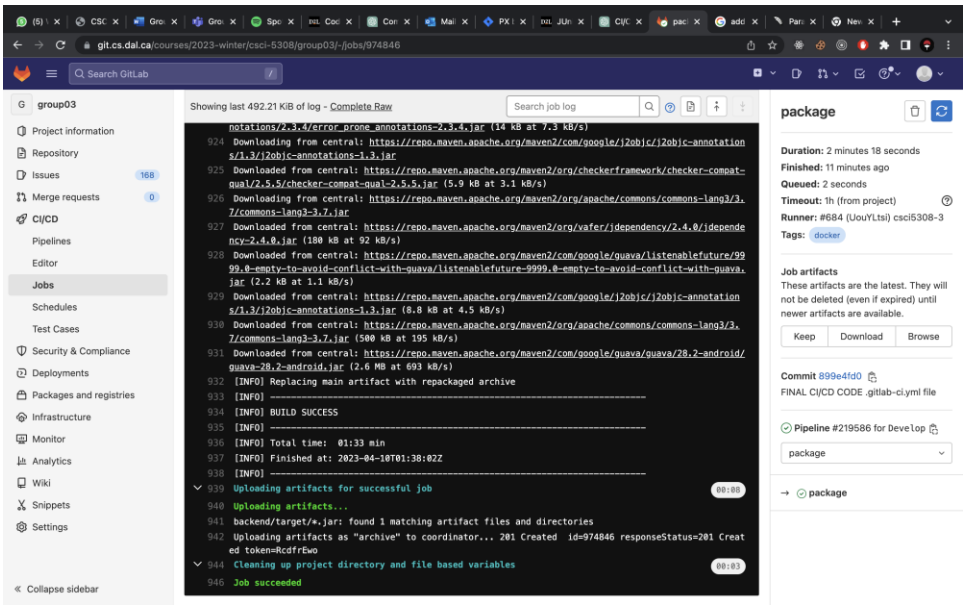
The "only" keyword is used throughout the file to specify which branches should trigger each stage. For example, the "package" stage only runs when changes are pushed to the "Develop" branch.

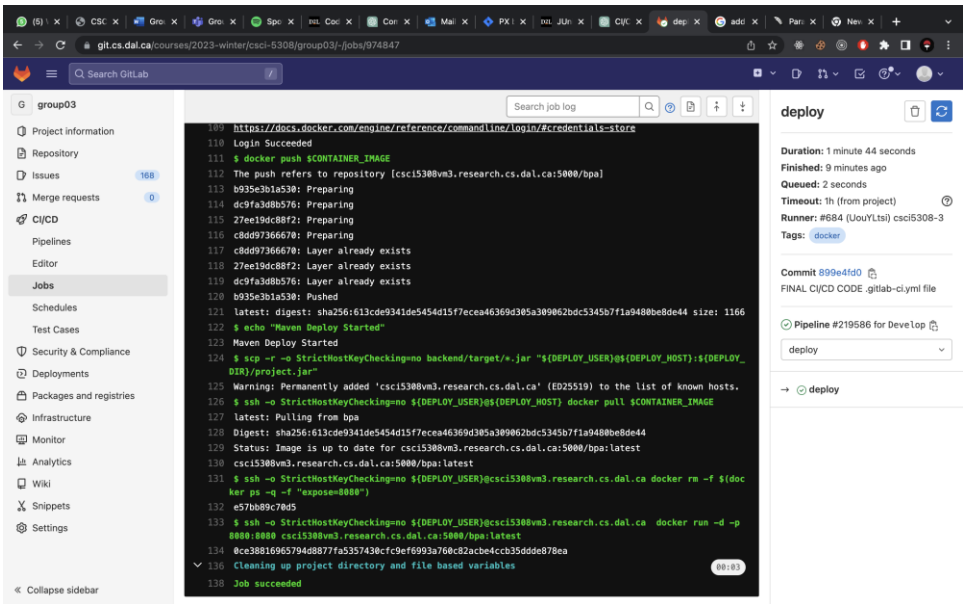


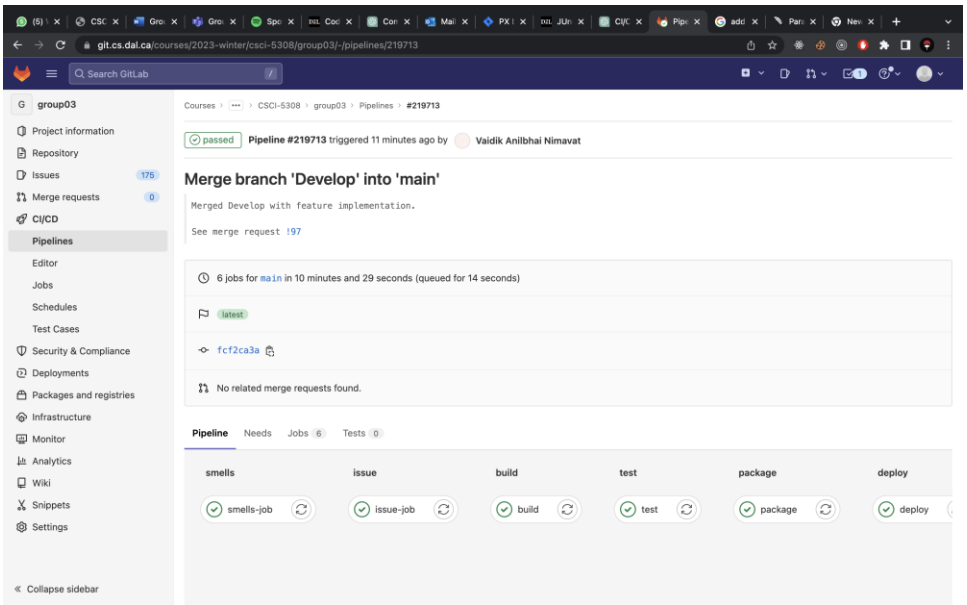


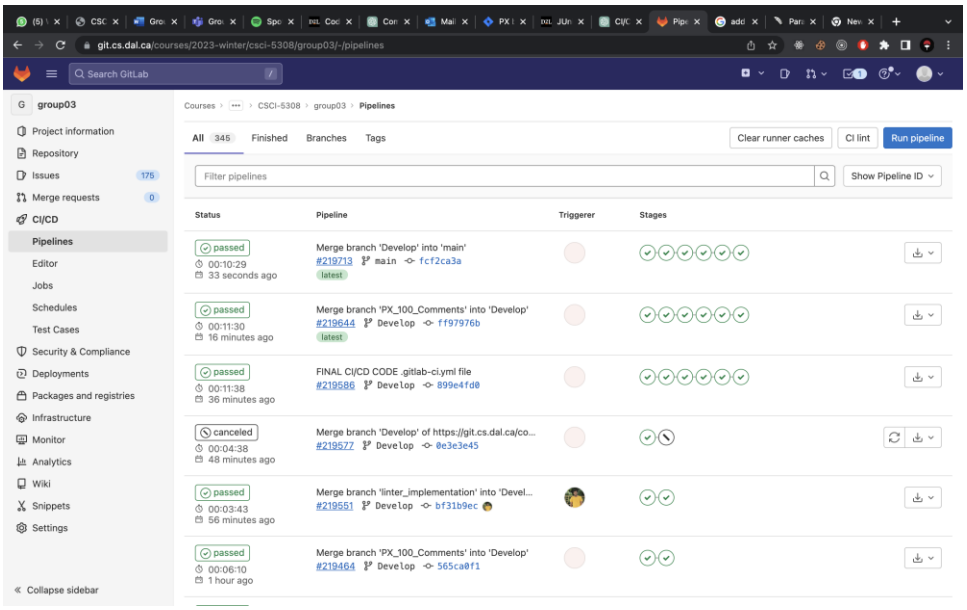

git.cs.dal.ca/courses/2023-winter/csci-5308/group03/-/jobs/974844
Search GitLab
group03
Project information
Repository
Issues 168
Merge requests 0
CI/CD
Pipelines
Editor
Jobs
Schedules
Test Cases
Security & Compliance
Deployments
Packages and registries
Infrastructure
Monitor
Analytics
Wiki
Snippets
Settings
Collapse sidebar
Showing last 499.99 KiB of log - Complete Raw
Search job log
0.v20140518/aether-util-1.0.0.v20140518.pom
929 Downloaded from central: https://repo.maven.apache.org/maven2/org/eclipse/aether/aether-util/1.0.0.v20140518/aether-util-1.0.0.v20140518.pom (2.2 kB at 41 kB/s)
930 Downloading from central: https://repo.maven.apache.org/maven2/org/eclipse/aether/aether/1.0.0.v20140518/aether-1.0.0.v20140518.pom
931 Downloaded from central: https://repo.maven.apache.org/maven2/org/eclipse/aether/aether/1.0.0.v20140518/aether-1.0.0.v20140518.pom (30 kB at 419 kB/s)
932 Downloading from central: https://repo.maven.apache.org/maven2/org/eclipse/aether/aether-api/1.0.0.v20140518/aether-api-1.0.0.v20140518.pom
933 Downloaded from central: https://repo.maven.apache.org/maven2/org/eclipse/aether/aether-api/1.0.0.v20140518/aether-api-1.0.0.v20140518.pom (1.9 kB at 35 kB/s)
934 Downloading from central: https://repo.maven.apache.org/maven2/org/eclipse/aether/aether-util/1.0.0.v20140518/aether-util-1.0.0.v20140518.jar
935 Downloading from central: https://repo.maven.apache.org/maven2/org/eclipse/aether/aether-api/1.0.0.v20140518/aether-api-1.0.0.v20140518.jar
936 Downloaded from central: https://repo.maven.apache.org/maven2/org/eclipse/aether/aether-api/1.0.0.v20140518/aether-api-1.0.0.v20140518.jar (136 kB at 1.3 MB/s)
937 Downloaded from central: https://repo.maven.apache.org/maven2/org/eclipse/aether/aether-util/1.0.0.v20140518/aether-util-1.0.0.v20140518.jar (146 kB at 1.2 MB/s)
938 [INFO] Installing /builds/courses/2023-winter/csci-5308/group03/backend/pom.xml to /builds/courses/2023-winter/csci-5308/group03/.m2/repository/com/GRP3/BPA/0.0.1-SNAPSHOT/BPA-0.0.1-SNAPSHOT.pom
939 [INFO] Installing /builds/courses/2023-winter/csci-5308/group03/backend/target/BPA-0.0.1-SNAPSHOT.jar to /builds/courses/2023-winter/csci-5308/group03/.m2/repository/com/GRP3/BPA/0.0.1-SNAPSHOT/BPA-0.0.1-SNAPSHOT.jar
940 [INFO] ----------
941 [INFO] BUILD SUCCESS
942 [INFO] ----------
943 [INFO] Total time: 01:40 min
944 [INFO] Finished at: 2023-04-10T01:33:22Z
945 [INFO] ----------
946 Cleaning up project directory and file based variables
00:04
948 Job succeeded
build
Duration: 2 minutes 40 seconds
Finished: 14 minutes ago
Queued: 2 seconds
Timeout: 1h (from project)
Runner: #684 (UouYLtsi) csci5308-3
Tags: docker
Commit 899e4fd0
FINAL CI/CD CODE .gitlab-ci.yml file
Pipeline #219586 for Develop
build
build


git.cs.dal.ca/courses/2023-winter/csci-5308/group03/-/jobs/974845
Search GitLab
group03
Project information
Repository
Issues 168
Merge requests 0
CI/CD
Pipelines
Editor
Jobs
Schedules
Test Cases
Security & Compliance
Deployments
Packages and registries
Infrastructure
Monitor
Analytics
Wiki
Snippets
Settings
Collapse sidebar
Showing last 483.85 KiB of log - Complete Raw
Search job log
HikariPool-1 - Closing connection com.mysql.cj.jdbc.ConnectionImpl@a455d69: (connection evicted)
681 2023-04-10T01:35:48.225Z DEBUG 80 --- [ionShutdownHook] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - After shutdown stats (total=0, active=0, idle=0, waiting=0)
682 2023-04-10T01:35:48.226Z INFO 80 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown completed.
683 2023-04-10T01:35:48.242Z INFO 80 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for persistence unit 'default'
684 2023-04-10T01:35:48.246Z INFO 80 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-2 - Shutdown initiated...
685 2023-04-10T01:35:48.249Z DEBUG 80 --- [ionShutdownHook] com.zaxxer.hikari.pool.HikariPool : HikariPool-2 - Before shutdown stats (total=2, active=0, idle=2, waiting=0)
686 2023-04-10T01:35:48.254Z DEBUG 80 --- [nnection closer] com.zaxxer.hikari.pool.PoolBase : HikariPool-2 - Closing connection com.mysql.cj.jdbc.ConnectionImpl@e77c9c3: (connection evicted)
687 2023-04-10T01:35:48.257Z DEBUG 80 --- [nnection closer] com.zaxxer.hikari.pool.PoolBase : HikariPool-2 - Closing connection com.mysql.cj.jdbc.ConnectionImpl@24e6ed55: (connection evicted)
688 2023-04-10T01:35:48.259Z DEBUG 80 --- [ionShutdownHook] com.zaxxer.hikari.pool.HikariPool : HikariPool-2 - After shutdown stats (total=0, active=0, idle=0, waiting=0)
689 2023-04-10T01:35:48.259Z INFO 80 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-2 - Shutdown completed.
690 [INFO]
691 [INFO] Results:
692 [INFO]
693 [INFO] Tests run: 78, Failures: 0, Errors: 0, Skipped: 0
694 [INFO]
695 [INFO] ----------
696 [INFO] BUILD SUCCESS
697 [INFO] ----------
698 [INFO] Total time: 01:33 min
699 [INFO] Finished at: 2023-04-10T01:35:48Z
700 [INFO] ----------
701 Cleaning up project directory and file based variables
00:04
703 Job succeeded
test
Duration: 2 minutes 20 seconds
Finished: 12 minutes ago
Queued: 2 seconds
Timeout: 1h (from project)
Runner: #684 (UouYLtsi) csci5308-3
Tags: docker
Commit 899e4fd0
FINAL CI/CD CODE .gitlab-ci.yml file
Pipeline #219586 for Develop
test
test


git.cs.dal.ca/courses/2023-winter/csci-5308/group03/-/jobs/974846
Search GitLab
group03
Project information
Repository
Issues 168
Merge requests 0
CI/CD
Pipelines
Editor
Jobs
Schedules
Test Cases
Security & Compliance
Deployments
Packages and registries
Infrastructure
Monitor
Analytics
Wiki
Snippets
Settings
Collapse sidebar
Showing last 492.21 KiB of log - Complete Raw
Search job log
notations/2.3.4/error_prone_annotations-2.3.4.jar (14 kB at 7.3 kB/s)
924 Downloading from central: https://repo.maven.apache.org/maven2/com/google/j2objc/j2objc-annotations/1.3/j2objc-annotations-1.3.jar
925 Downloaded from central: https://repo.maven.apache.org/maven2/org/checkerframework/checker-compat-qual/2.5.5/checker-compat-qual-2.5.5.jar (5.9 kB at 3.1 kB/s)
926 Downloading from central: https://repo.maven.apache.org/maven2/org/apache/commons/commons-lang3/3.7/commons-lang3-3.7.jar
927 Downloaded from central: https://repo.maven.apache.org/maven2/org/vafer/jdependency/2.4.0/jdependency-2.4.0.jar (180 kB at 92 kB/s)
928 Downloaded from central: https://repo.maven.apache.org/maven2/com/google/guava/listenablefuture/9999.0-empty-to-avoid-conflict-with-guava/listenablefuture-9999.0-empty-to-avoid-conflict-with-guava.jar (2.2 kB at 1.1 kB/s)
929 Downloaded from central: https://repo.maven.apache.org/maven2/com/google/j2objc/j2objc-annotations/1.3/j2objc-annotations-1.3.jar (8.8 kB at 4.5 kB/s)
930 Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/commons/commons-lang3/3.7/commons-lang3-3.7.jar (500 kB at 195 kB/s)
931 Downloaded from central: https://repo.maven.apache.org/maven2/com/google/guava/guava/28.2-android/guava-28.2-android.jar (2.6 MB at 693 kB/s)
932 [INFO] Replacing main artifact with repackaged archive
933 [INFO] --
934 [INFO] BUILD SUCCESS
935 [INFO] --
936 [INFO] Total time: 01:33 min
937 [INFO] Finished at: 2023-04-10T01:38:02Z
938 [INFO] --
939 Uploading artifacts for successful job
00:00
940 Uploading artifacts...
941 backend/target/*.jar: found 1 matching artifact files and directories
942 Uploading artifacts as "archive" to coordinator... 201 Created id=974846 responseStatus=201 Created token=RcdfrEwo
944 Cleaning up project directory and file based variables
00:03
946 Job succeeded
package
Duration: 2 minutes 18 seconds
Finished: 11 minutes ago
Queued: 2 seconds
Timeout: 1h (from project)
Runner: #684 (UouYLtsi) csci5308-3
Tags: docker
Job artifacts
These artifacts are the latest. They will not be deleted (even if expired) until newer artifacts are available.
Keep
Download
Browse
Commit 899e4fd0
FINAL CI/CD CODE .gitlab-ci.yml file
Pipeline #219586 for Develop
package
package


git.cs.dal.ca/courses/2023-winter/csci-5308/group03/-/jobs/974847
Search GitLab
group03
Project information
Repository
Issues 168
Merge requests 0
CI/CD
Pipelines
Editor
Jobs
Schedules
Test Cases
Security & Compliance
Deployments
Packages and registries
Infrastructure
Monitor
Analytics
Wiki
Snippets
Settings
Collapse sidebar
Search job log
109 https://docs.docker.com/engine/reference/commandline/login/#credentials-store
110 Login Succeeded
111 $ docker push $CONTAINER_IMAGE
112 The push refers to repository [csci5308vm3.research.cs.dal.ca:5000/bpa]
113 b935e3b1a530: Preparing
114 dc9fa3d8b576: Preparing
115 27ee19dc88f2: Preparing
116 c8dd97366670: Preparing
117 c8dd97366670: Layer already exists
118 27ee19dc88f2: Layer already exists
119 dc9fa3d8b576: Layer already exists
120 b935e3b1a530: Pushed
121 latest: digest: sha256:613cde9341de5454d15f7ecea46369d305a309062bdc5345b7f1a9480be8de44 size: 1166
122 $ echo "Maven Deploy Started"
123 Maven Deploy Started
124 $ scp -r -o StrictHostKeyChecking=no backend/target/*.jar "${DEPLOY_USER}@${DEPLOY_HOST}:${DEPLOY_DIR}/project.jar"
125 Warning: Permanently added 'csci5308vm3.research.cs.dal.ca' (ED25519) to the list of known hosts.
126 $ ssh -o StrictHostKeyChecking=no ${DEPLOY_USER}@${DEPLOY_HOST} docker pull $CONTAINER_IMAGE
127 latest: Pulling from bpa
128 Digest: sha256:613cde9341de5454d15f7ecea46369d305a309062bdc5345b7f1a9480be8de44
129 Status: Image is up to date for csci5308vm3.research.cs.dal.ca:5000/bpa:latest
130 csci5308vm3.research.cs.dal.ca:5000/bpa:latest
131 $ ssh -o StrictHostKeyChecking=no ${DEPLOY_USER}@csci5308vm3.research.cs.dal.ca docker rm -f $(docker ps -q -f "expose=8080")
132 e57bb89c70d5
133 $ ssh -o StrictHostKeyChecking=no ${DEPLOY_USER}@csci5308vm3.research.cs.dal.ca docker run -d -p 8080:8080 csci5308vm3.research.cs.dal.ca:5000/bpa:latest
134 0ce38816965794d8877fa5357430cfc9ef6993a760c82acbe4ccb35ddde878ea
136 Cleaning up project directory and file based variables
00:03
138 Job succeeded
deploy
Duration: 1 minute 44 seconds
Finished: 9 minutes ago
Queued: 2 seconds
Timeout: 1h (from project)
Runner: #684 (UouYLtsi) csci5308-3
Tags: docker
Commit 899e4fd0
FINAL CI/CD CODE .gitlab-ci.yml file
Pipeline #219586 for Develop
deploy
deploy





5. USER FLOW OF BROWNIE POINT

A teacher logs into the Brownie Point app and creates a new course for their class. They add the course details, such as the title and description.

A student downloads the Brownie Point app and creates a new account. They provide their name, email, Banner ID and password to register, and then log in to the app. Once logged in, the student can see a list of available courses and enroll in the ones they want to take.

The teacher goes to their course page in the Brownie Point app and sees a list of enrolled students. They can view each student's profile and performance on the course, as well as the number of points they have earned so far. The teacher can also delete the course details, as needed.

In class, the teacher uses the Brownie Point app to award points to students who answer questions or participate in activities. The teacher simply scans the QR code on the student's phone, which instantly awards them the designated number of points.

Outside of class, the student can log into the Brownie Point app to view their point balance and the courses they are enrolled in.

At the end of the semester, the teacher may use the Brownie Point app to generate a report of each student's performance in the course, including their final point balance and any comments or feedback. The teacher can then use this report to assign grades or assess the students' progress.

6. The Development process for Brownie Point:

Team Collaboration:

1. The team consisted of two frontend developers, three backend developers, with all being QA.
2. The team worked together including daily stand-up meetings, sprint planning, and retrospectives.
3. Communication was facilitated using tools such as TEAMS and in person Meet-ups

Project Organization:

1. The project was divided into sprints, with each sprint lasting two weeks.
2. Each sprint was organized based on the user stories and tasks that needed to be completed.
3. Tasks were assigned to team members based on their strengths and areas of expertise.

Tools and Methodologies:

1. The frontend was developed using React Native, while the backend was developed using Spring Boot.
2. Version control was managed using Git, and code reviews were conducted using GitHub pull requests.
3. Continuous integration and deployment were managed using --------?
4. In the beginning, JUNIT testing was performed on the system's backend in accordance with the Test-Driven Development (TDD) methodology. Integration test cases were added to the system once the building was finished. Combining the two situations resulted in a total of 162 test cases, all of which were passed.

5. Overall, the development process for Brownie Point was well-organized and followed best practices in software development. The team will be able to successfully deliver the project on time.

7. API:

GET API: http://localhost:8080/api/user
This API is used to retrieve information about a user in the system. It is a GET request and is expected to return user details such as Name, Banner ID, email, and other relevant information.

GET API: http://localhost:8080/status
This API is used to retrieve the status of the server. It is a GET request and is expected to return information about the server running or not.

POST API: http://localhost:8080/api/auth/login
This API is used to authenticate a user in the system. It is a POST request and is expected to take in the user's credentials such as email id and password and return an access token if the user is authenticated.

POST API: http://localhost:8080/api/auth/register
This API is used to register a new user in the system. It is a POST request and is expected to take in the user's details such as First Name, Last Name, email, BannerID and password and create a new user in the system.

POST API: http://localhost:8080/api/auth/reset-password
This API is used to initiate the process of resetting a user's password. It is a POST request and is expected to take in the user's email and send a password reset email to the user's email address containing the OTP.

POST API: http://localhost:8080/api/auth/reset-password-matchotp
This API is used to reset a user's password after they have verified their identity through an OTP. It is a POST request and is expected to take in the user's email, OTP, and new password.

POST API: http://localhost:8080/teachers/courses/addCourse
This API is used to add a new course for a teacher in the system. It is a POST request and is expected to take in the course details such as course code, Course Name and course description.

GET API: http://localhost:8080/teachers/courses

This API is used to retrieve all the courses that a teacher has created in the system. It is a GET request and is expected to return a list of courses with their details.

POST API: http://localhost:8080/teachers/courses/addCourses
This API is used to add multiple courses for a teacher in the system. It is a POST request and is expected to include a list of course details such as name and description.

DELETE API: http://localhost:8080/teachers/courses/removeCourse
This API is used to remove a course created by a teacher in the system. It is a DELETE request and is expected to take in the course ID and delete the course from the system.

PUT API: http://localhost:8080/teachers/courses/{courseId}/{studentId}
This API is used to add a student to a course created by a teacher in the system. It is a PUT request and is expected to take in the course ID and the student ID and add the student to the course.

8 .Testing and QA:

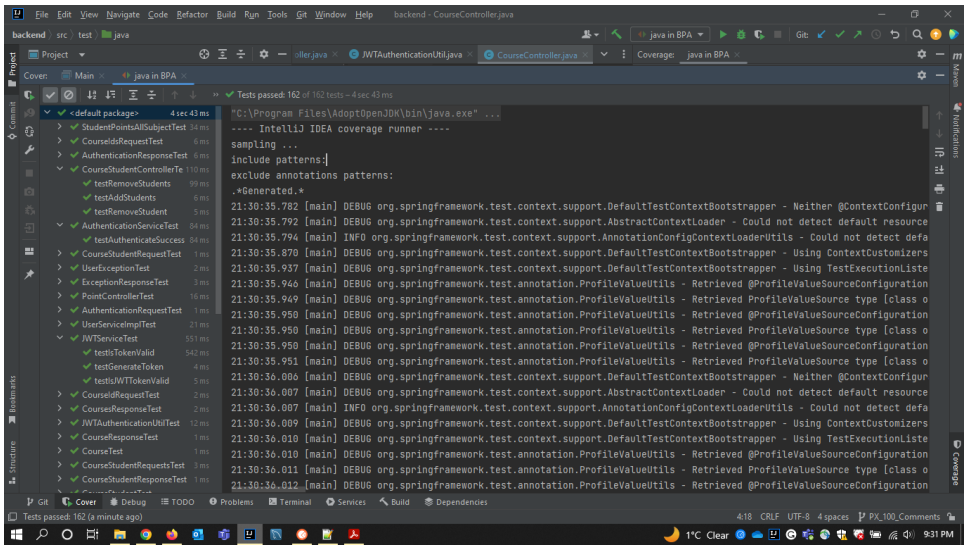


Figure 1 Test Cases

Figure 1 demonstrates that we have successfully completed 162 test cases, and Figure 2 demonstrates that our code coverage is around 70%.

Element	Class, %	Method, %	Line, %
com	94% (51...	84% (197/...	71% (489/...
GRP3	94% (51...	84% (197/...	71% (489/...
BPA	94% (51...	84% (197/...	71% (489/...
config	100% (3...	100% (8/8)	73% (31/42)
controller	100% (6...	76% (16/21)	67% (97/1...
DTO	100% (1...	100% (9/9)	100% (16/...
exceptions	100% (3...	90% (9/10)	90% (10/11)
model	100% (6...	83% (36/43)	84% (37/44)
repository	100% (0...	100% (0/0)	100% (0/0)
request	90% (9/...	91% (33/36)	91% (33/36)
response	84% (11...	84% (39/46)	86% (44/51)
service	100% (9...	80% (42/52)	63% (207/...
utils	100% (2...	83% (5/6)	86% (13/15)
BpaApplication	100% (1...	0% (0/1)	50% (1/2)

Figure 2 Code Coverage of Test Cases

9. Results and Evaluation:

The results achieved for Brownie Point were quite successful in meeting its objectives. The application was developed as per the requirements and design outlined in the project report. The application was tested rigorously, and the bugs were fixed promptly to ensure that the application is stable and reliable. The application was deployed successfully on Android.

In terms of user engagement, the application may receive positive feedback from both teachers and students who will use it. The user interface is designed keeping in mind the target audience and is found to be user-friendly, intuitive and easy to navigate. The application is successful in addressing the pain points of both teachers and students, and it will help them manage their courses and points effectively.

Overall, the project is deemed successful as it achieved its goals of developing a mobile application that would help teachers and students manage courses effectively, and the project meets the required quality standards.

10. Snapshots of the User Interface Design:

10.1 Home Page:

Figure 10.1 Home Page

10.2 Registration Page:

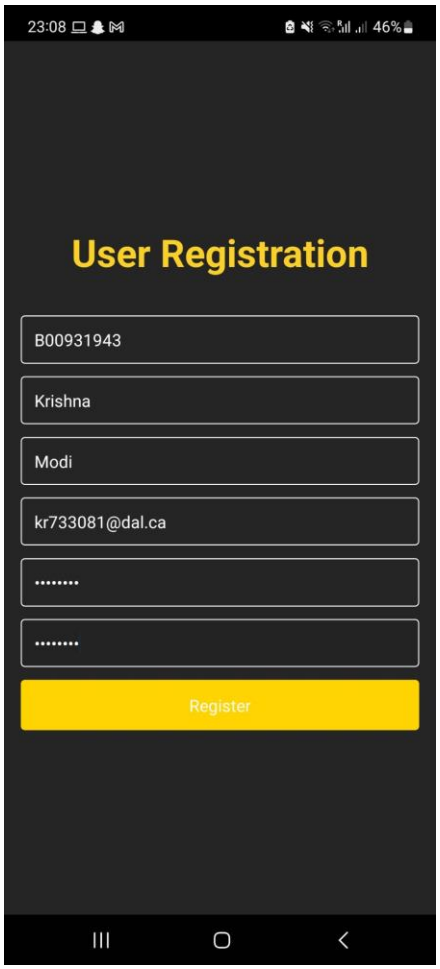


Figure 10.2 Registration Page

10.3 Login Page:

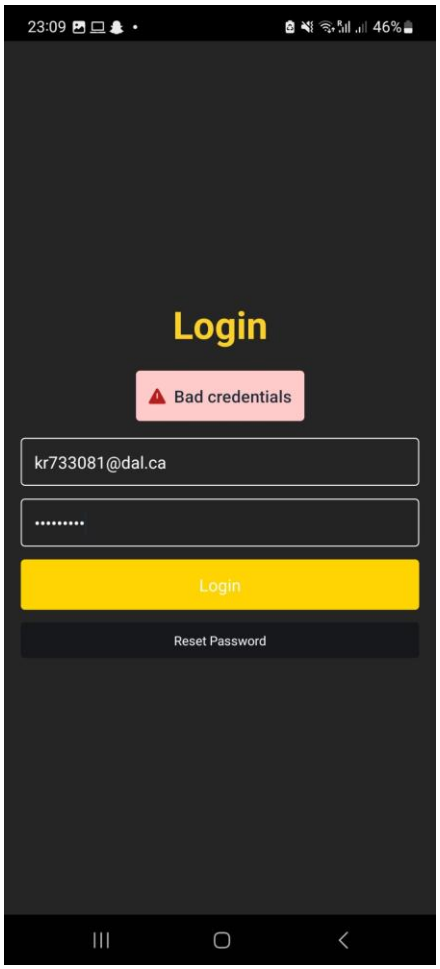


Figure 10.3 Login Page

10.4 Student User Home Page

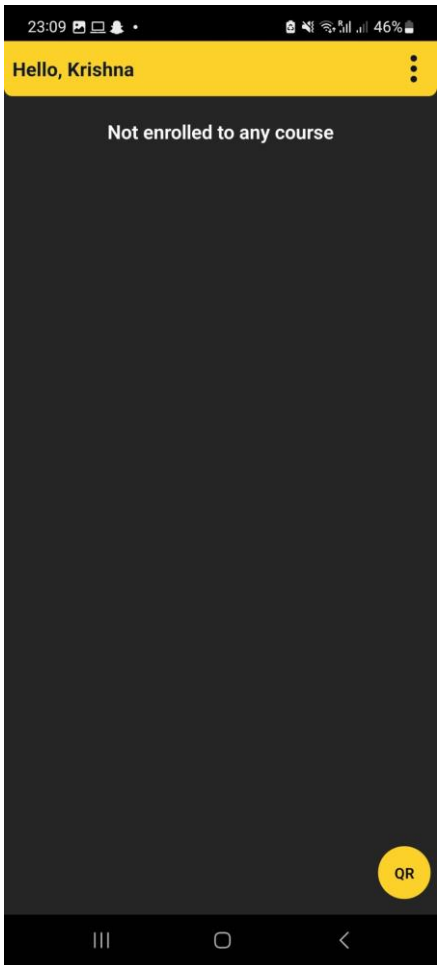


Figure 10.4 Student Home Page

10.5 User Profile

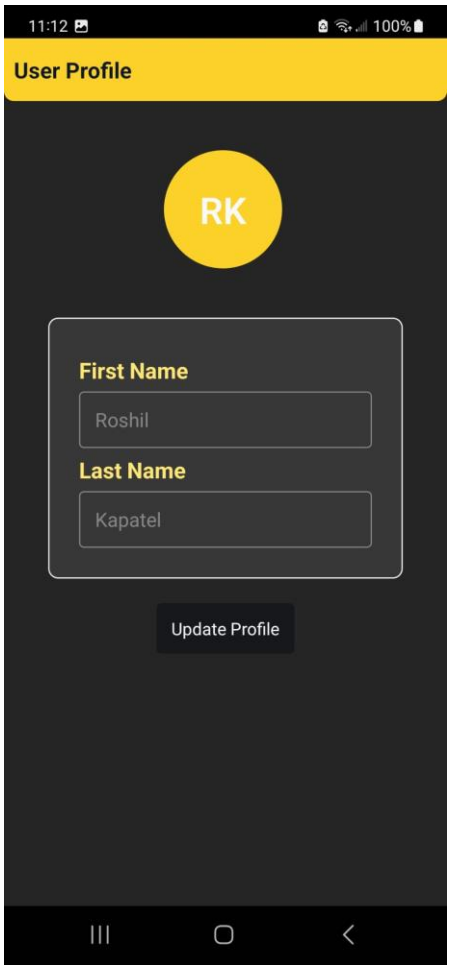


Figure 10.5 User Profile

10.6 Teacher Adding Students

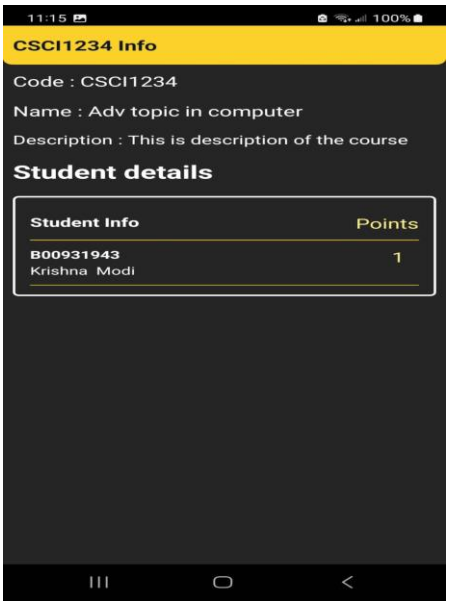


Figure 10.6 10.6 Teacher Adding Students

10.7 Teacher EDIT COURSE

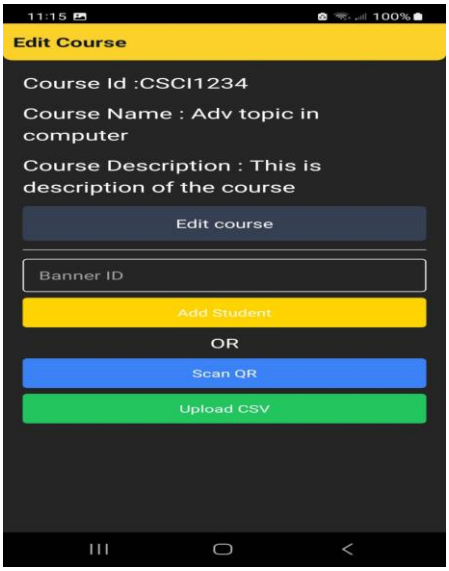


Figure 10.7 Teacher EDIT COURSE

10.8 Teacher Delete Course

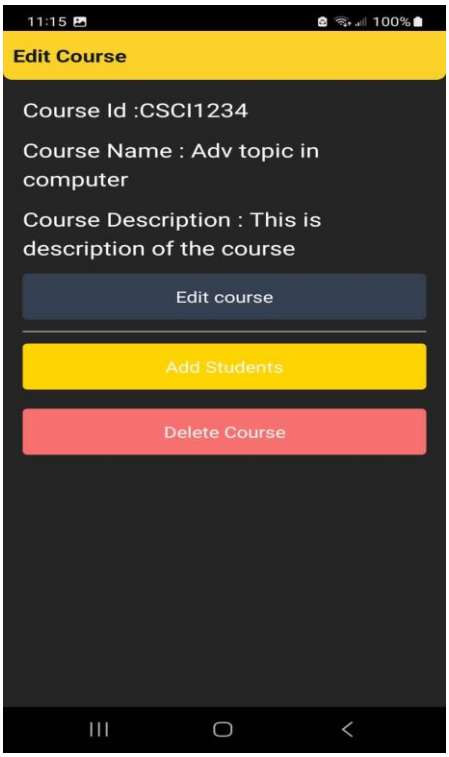


Figure 10.7 Teacher EDIT COURSE

10.8 Teacher Scanning QR

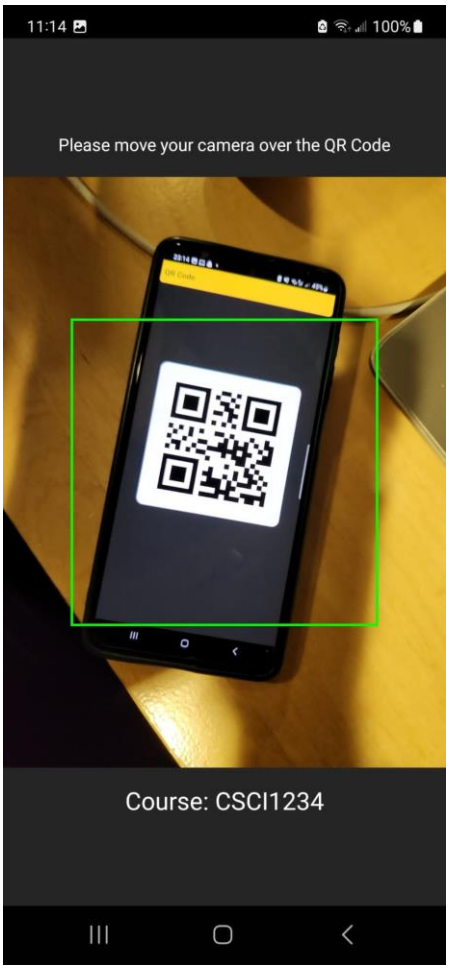


Figure 10.8 Teacher Scanning QR

10.9 Point Added

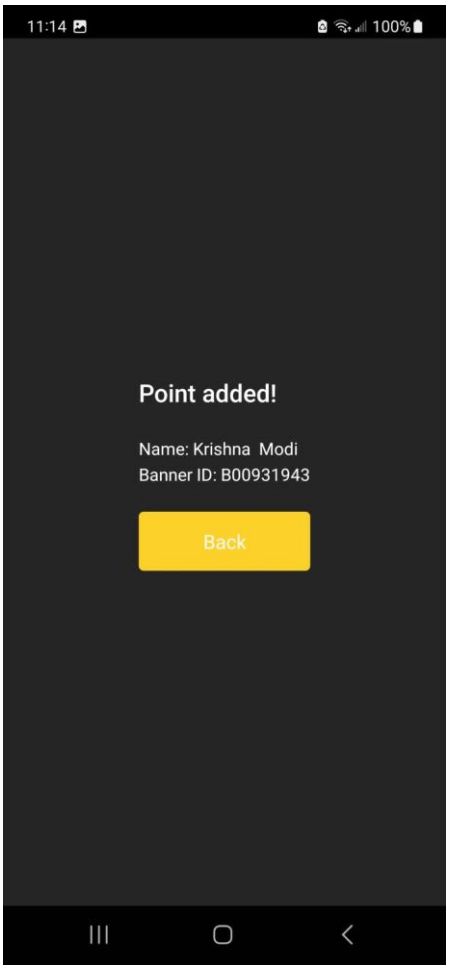


Figure 10.9 Point Added

10. 10 Teacher Home Page

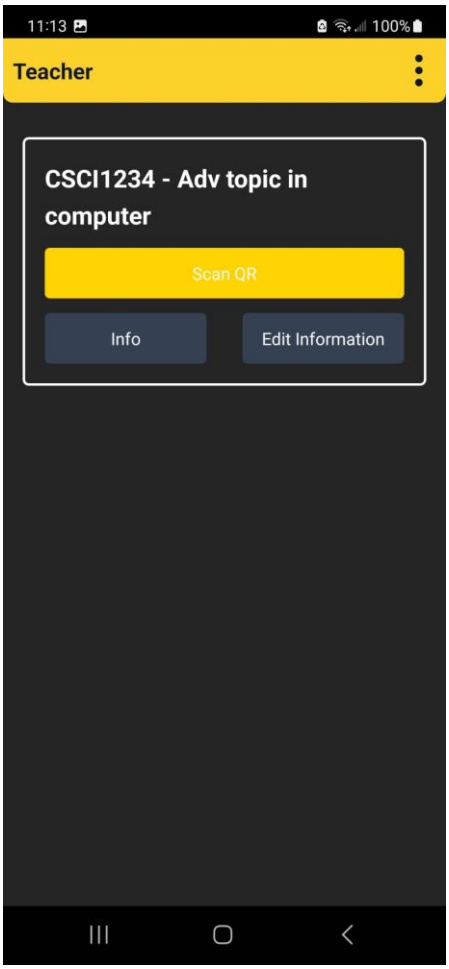


Figure 10. 10 Teacher Home Page

10.11 Teacher Add Course

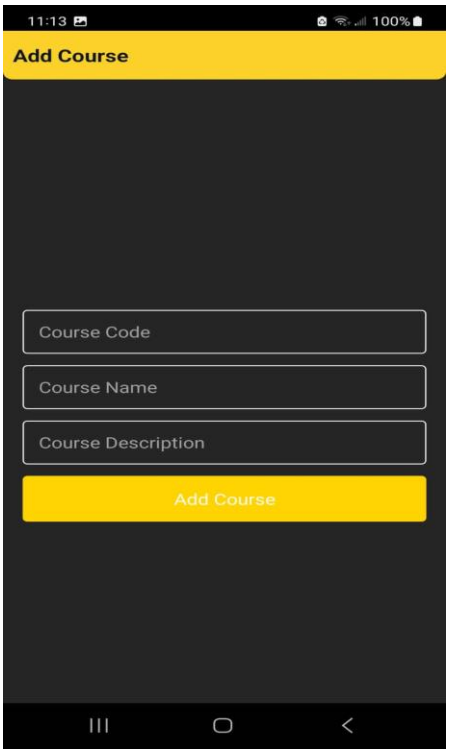


Figure 10.11 Teacher Add Course

10.12 Techer Home without subject

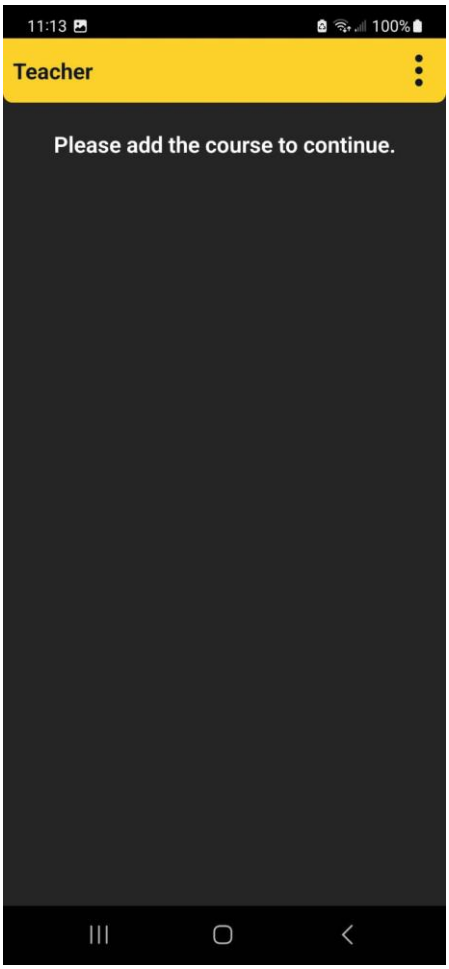


Figure 10.12 Techer Home without subject

10.13 QR Code Generation

Fig 10.10 QR code on Student's screen

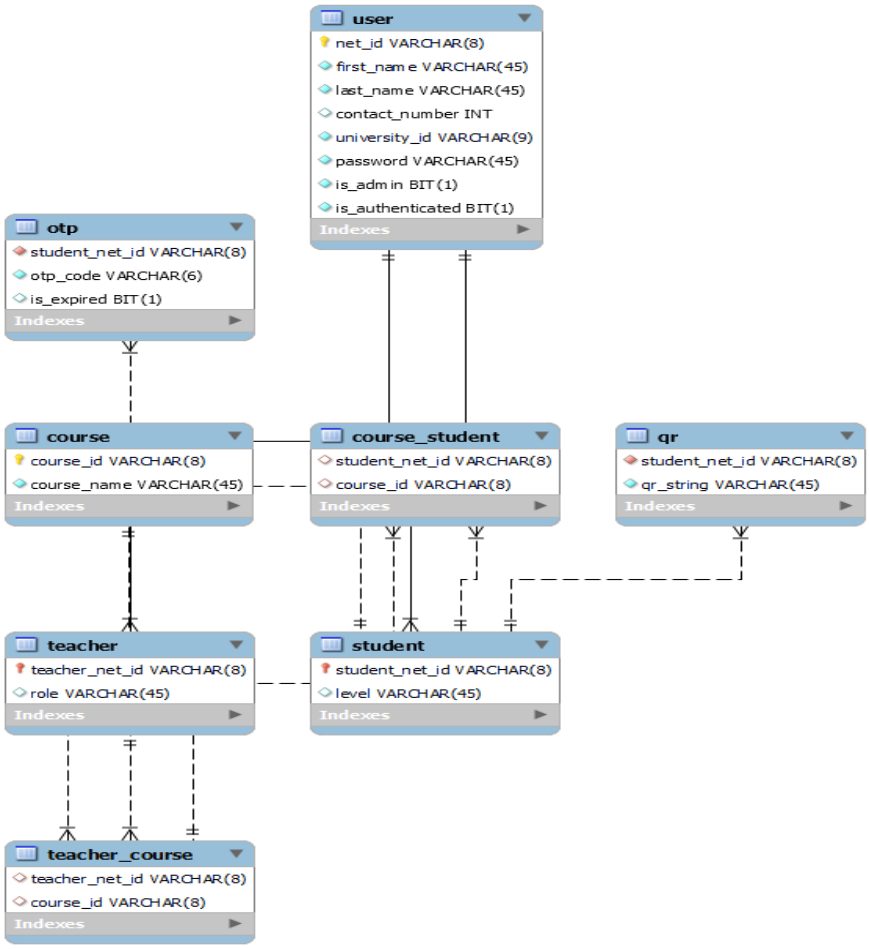
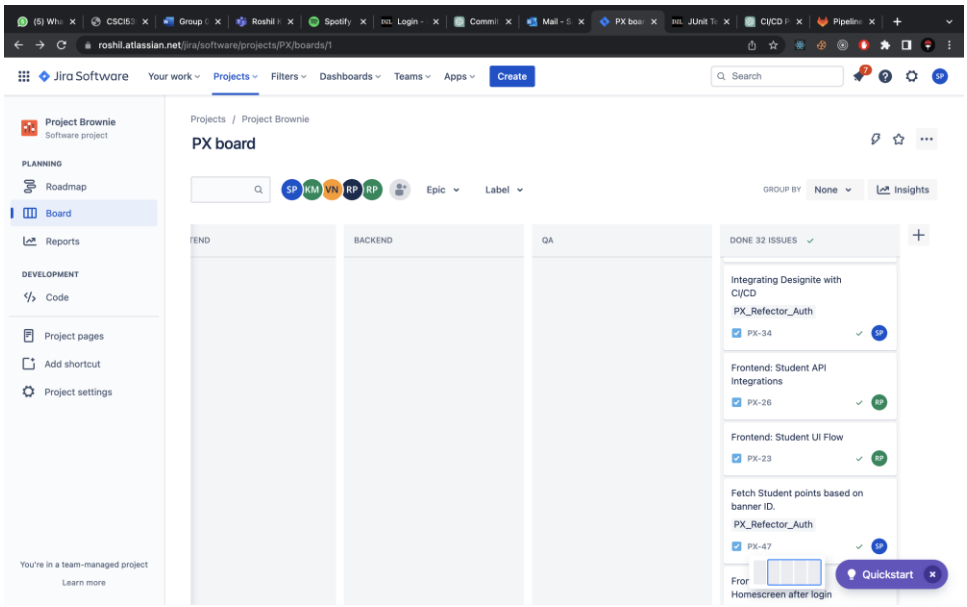
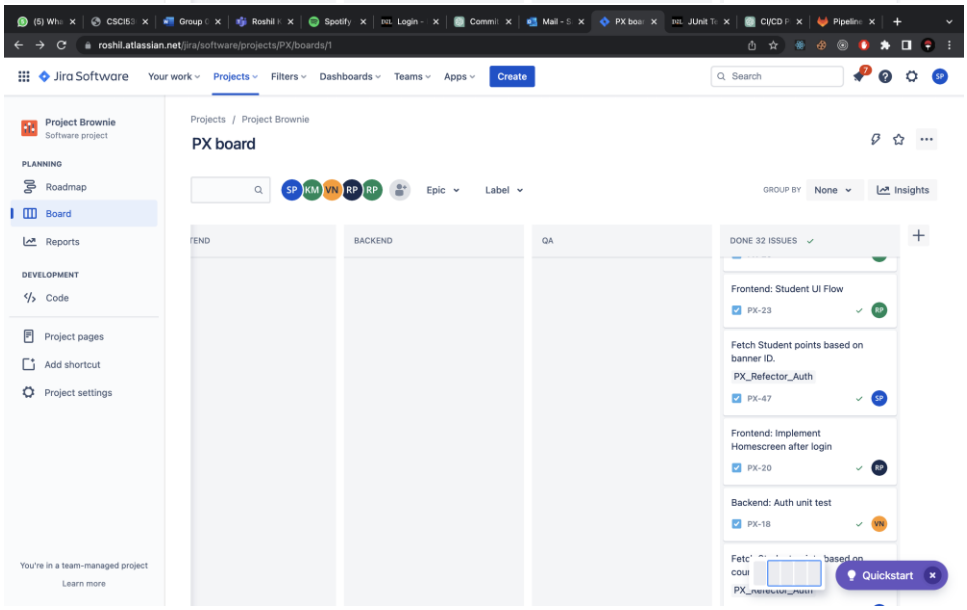
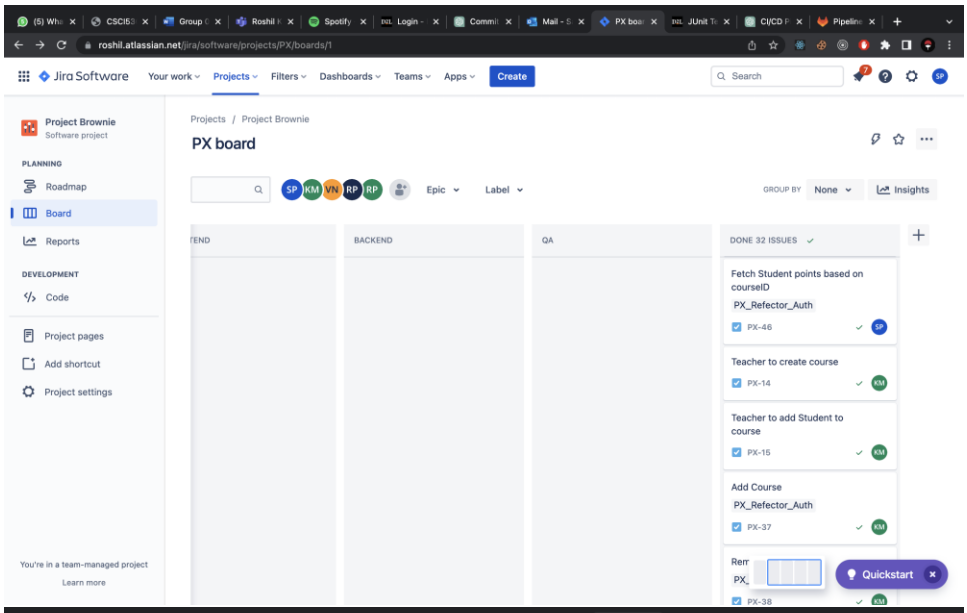


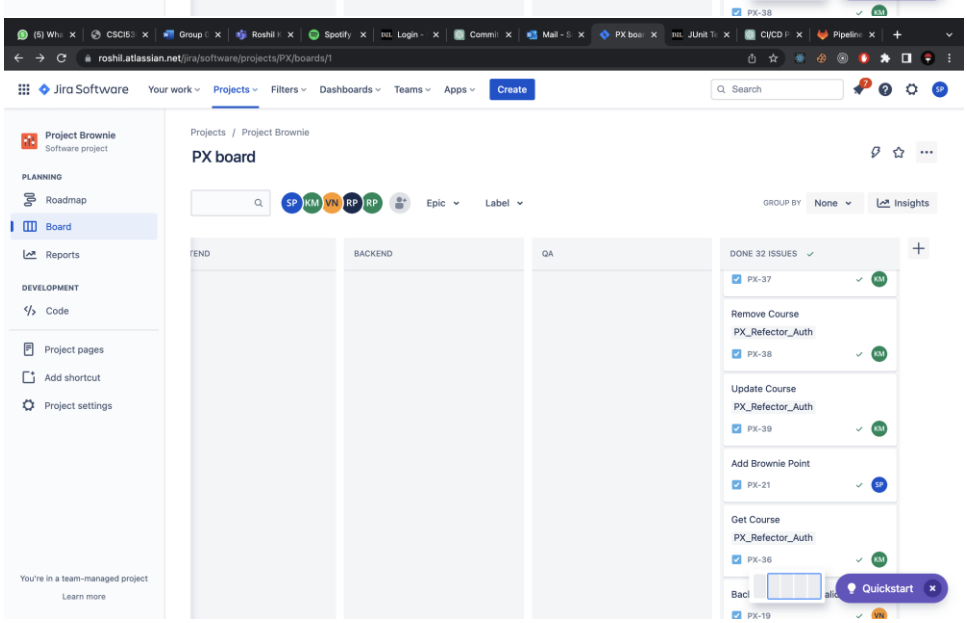
Figure 10.14 ERD for Brownie Point

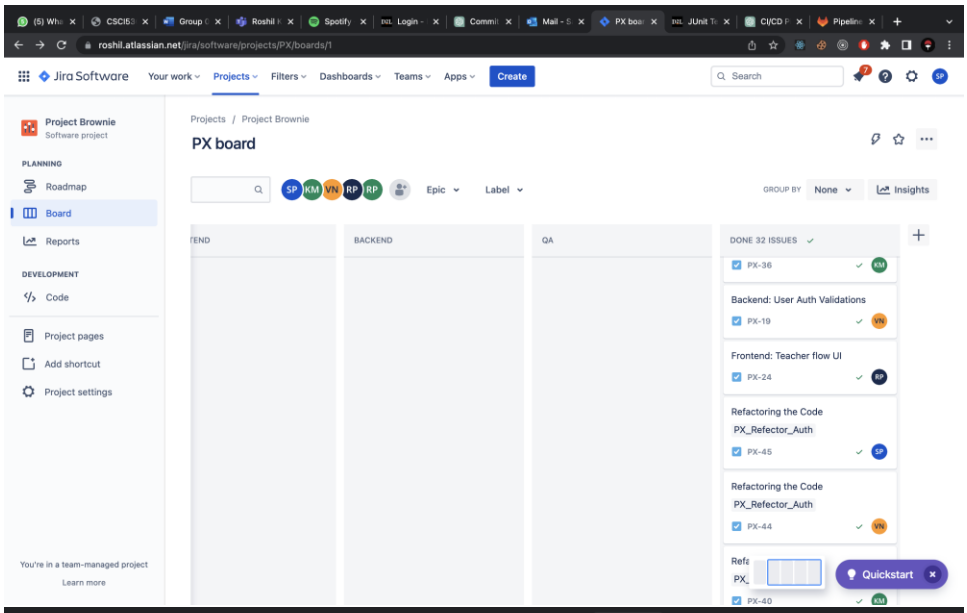
10.15 JIRA BOARD

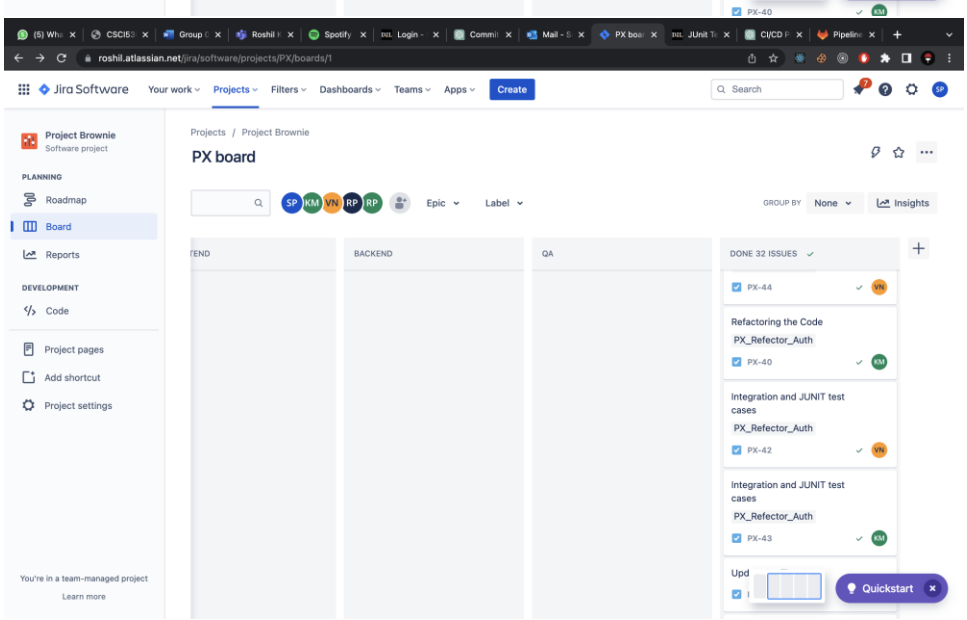

roshil.atlassian.net/jira/software/projects/PX/boards/1
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Project Brownie
Software project
PLANNING
Roadmap
Board
Reports
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Project Brownie
PX board
Epic
Label
GROUP BY
None
Insights
BACKEND
QA
DONE 32 ISSUES
Integrating Designite with CI/CD
PX_Refector_Auth
PX-34
Frontend: Student API Integrations
PX-26
Frontend: Student UI Flow
PX-23
Fetch Student points based on banner ID.
PX_Refector_Auth
PX-47
Quickstart

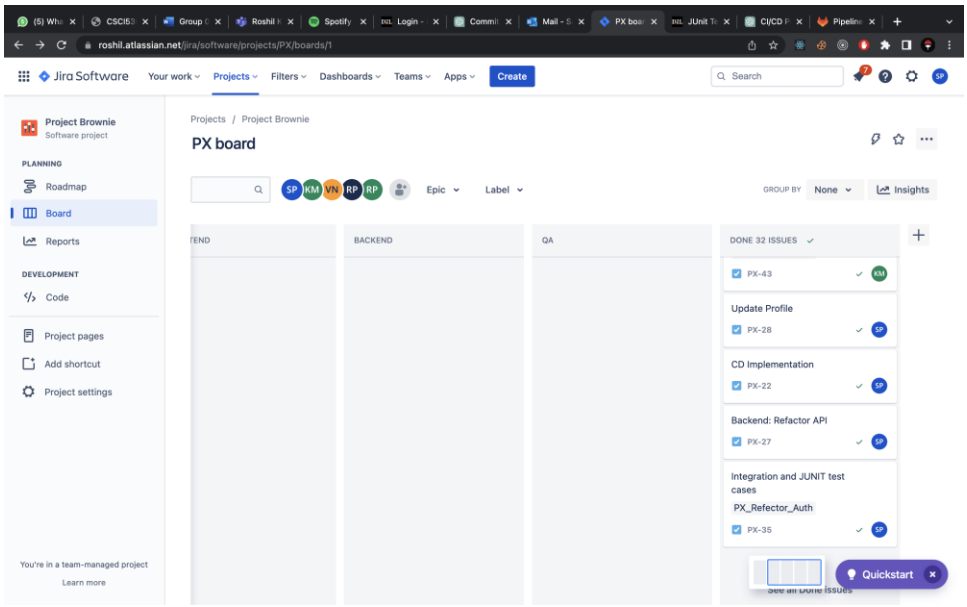

roshil.atlassian.net/jira/software/projects/PX/boards/1
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Project Brownie
Software project
PLANNING
Roadmap
Board
Reports
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Project Brownie
PX board
Epic
Label
GROUP BY
None
Insights
BACKEND
QA
DONE 32 ISSUES
Frontend: Student UI Flow
PX-23
Fetch Student points based on banner ID.
PX_Refector_Auth
PX-47
Frontend: Implement Homescreen after login
PX-20
Backend: Auth unit test
PX-18
Quickstart


roshil.atlassian.net/jira/software/projects/PX/boards/1
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Project Brownie
Software project
PLANNING
Roadmap
Board
Reports
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Project Brownie
PX board
Epic
Label
GROUP BY
None
Insights
BACKEND
QA
DONE 32 ISSUES
Fetch Student points based on courseID
PX_Refector_Auth
PX-46
Teacher to create course
PX-14
Teacher to add Student to course
PX-15
Add Course
PX_Refector_Auth
PX-37
Quickstart


roshil.atlassian.net/jira/software/projects/PX/boards/1
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Project Brownie
Software project
PLANNING
Roadmap
Board
Reports
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Project Brownie
PX board
Epic
Label
GROUP BY
None
Insights
BACKEND
QA
DONE 32 ISSUES
PX-37
Remove Course
PX_Refector_Auth
PX-38
Update Course
PX_Refector_Auth
PX-39
Add Brownie Point
PX-21
Get Course
PX_Refector_Auth
PX-36
Quickstart


roshil.atlassian.net/jira/software/projects/PX/boards/1
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Project Brownie
Software project
PLANNING
Roadmap
Board
Reports
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Project Brownie
PX board
Epic
Label
GROUP BY
None
Insights
BACKEND
QA
DONE 32 ISSUES
PX-36
Backend: User Auth Validations
PX-19
Frontend: Teacher flow UI
PX-24
Refactoring the Code
PX_Refector_Auth
PX-45
Refactoring the Code
PX_Refector_Auth
PX-44
Quickstart


roshil.atlassian.net/jira/software/projects/PX/boards/1
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Project Brownie
Software project
PLANNING
Roadmap
Board
Reports
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Project Brownie
PX board
Epic
Label
GROUP BY
None
Insights
BACKEND
QA
DONE 32 ISSUES
PX-44
Refactoring the Code
PX_Refector_Auth
PX-40
Integration and JUNIT test cases
PX_Refector_Auth
PX-42
Integration and JUNIT test cases
PX_Refector_Auth
PX-43
Quickstart



Lessons Learned:

During the development process of Brownie Point, the team learned several valuable lessons. Firstly, communication and collaboration among team members were critical for the successful completion of the project. The team used daily stand-up meetings to ensure that everyone was on the same page and that any issues were addressed promptly.

Secondly, proper planning and organization were key factors in ensuring that the project was completed on time. The team used tools such as JIRA to manage tasks and ensure that everyone was working on the right thing at the right time.

Lastly, the team learned that it is essential to gather and incorporate feedback from users (our client team) during the development process. User (Client Team) feedback helped the team to identify issues and make improvements to the application, resulting in a better user experience.

Overall, the team was able to successfully complete the project, but there were several areas for improvement. For future projects, the team could improve our documentation process, Reduce Code Smells and implement stricter testing and quality assurance measures to further enhance the application's performance and user experience

Individual Contribution:

Table 1. Feature Contribution

Sr no	Team Member	Feature Contributed / Implemented	Member's contribution to the project (in percent)	Number of meetings attended With TA WITH
01	Sarthak Patel	Backend- Ci/CD configuration, Brownie point distribution, Update Profile, Create API student point, Create API for teacher from that they can see points, Test cases, Smell handling	20	9
02	Roshil Ka Patel	Frontend- User Registration, Reset Password, Home screen for Teacher Flow – Course, Course Information, Edit Course and Scanner for Brownie Point with backend integration	21	9
03	Riya Patel	Frontend-User Login, Home Screen for Student Flow – User Profile, Qr code generation for Brownie Point With backend integration	17	9
04	Vaidik Nimavat	Backend- Authentication flow, JWT implementation, Security Configuration, Reset/Change password, Integration test cases, API Status confirmation, API refactoring, Smells handling	21	9

05	Krishna Modi	Course Service: get, add, update and delete course/courses CourseStudent Service: get, add, delete student/students from course API end points in Course and Course Student Controllers Customizable Exception, Major Refactoring to follow SRP, Unit test cases: Course and Course Student workflow, Implementation smells handling in Controller, Architecture smell handling.	21	9
	Total		100	9

Conclusion:

- In conclusion, the Brownie Point project has been a successful mobile application that meets the needs of both teachers and students. The team was able to deliver all the required features and design a user-friendly interface that was easy to navigate.
- During the development process, the team encountered some challenges, including code smells and time constraints, but we were able to overcome them and deliver a quality product.
- In terms of future development, if we got an extra time permit, the team could consider adding more features such as chat functionality between teachers and students. Additionally, the team could explore ways to optimize the application's performance and security.
- Overall, the Brownie Point project has been a success.

References:

---

[1] Ali-Bouali, "Ali-Bouali/spring-boot-3-jwt-security: Sample Project on how to implement JWT security based using Spring Boot 3 and Spring Security 6," *GitHub*. [Online]. Available: https://github.com/ali-bouali/spring-boot-3-jwt-security. [Accessed: 09-Apr-2023].

[2] S. B. Stefan Bechtold, *JUnit 5 user guide*. [Online]. Available: https://junit.org/junit5/docs/current/user-guide/#writing-tests. [Accessed: 09-Apr-2023].

[3] Baeldung.com. [Online]. Available: https://www.baeldung.com/intro-to-project-lombok. [Accessed: 10-Apr-2023].

[4] "Mockito - mockito-core 5.2.0 javadoc," Javadoc.io. [Online]. Available: https://javadoc.io/doc/org.mockito/mockito-core/latest/org/mockito/Mockito.html. [Accessed: 10-Apr-2023].

[5] "Accessing Data with JPA - Spring," Spring, [Online]. Available: https://spring.io/guides/gs/accessing-data-jpa/. [Accessed: Apr. 10, 2023]

[6] "Gitlab ci CD tutorial for beginners [crash course]," *YouTube*, 09-Jun-2022. [Online]. Available: https://www.youtube.com/watch?v=qP8kir2GUgo&t=2629s&ab_channel=TechWorldwithNana. [Accessed: 09-Apr-2023].

[7] "Npm: React-native-document-picker," npm. [Online]. Available: https://www.npmjs.com/package/react-native-document-picker. [Accessed: 10-Apr-2023].

[8] "Npm: React-native-qrcode-scanner," npm. [Online]. Available: https://www.npmjs.com/package/react-native-qrcode-scanner. [Accessed: 10-Apr-2023].

[9] "Introduction," Reactnative.dev. [Online]. Available: https://reactnative.dev/docs/getting-started. [Accessed: 10-Apr-2023].