Objective:

Code Labeller:
This is an application which can aid a researcher can share code snippets publically whereupon interested personals can annotate code snippets following the annotations provided by said researcher.
Researchers can visit the uploaded code snippets later and see the annotations completed till that point.

Goal:
The goal here is to provide the admin with an ability to create surveys, provide the ability to annotator to "mark" or annotate those surveys and then admin can see whatever those annotators have annotated or "mark".

Branches:
- Frontend: “main_frontend”
- Backend: “main”

Members:
1. Harjot Singh
    a. Usernames: hr744527@dal.ca
2. Margin
    a. Usernames: mr353045@dal.ca
3. Aanandi Pankhania
    a. Usernames: aanandi2802@users.noreply.github.com
4. Sourav
    a. Usernames:
        i. souravghai96@gmail.com
        ii. = (usernames got registered as "=" by mistake).
        iii. sr734546@dal.ca

Dependency:
Backend:

- Java:
    - Version: 17
- Maven
    - Version: 3.8.7
- Spring Boot Starter Parent:
    - Version: 3.0.2
    - For MVC architecture in spring boot.
- Spring Security
    - Version: 6.0.1
    - For authentication, authorization and jwt token.
- Jwt Jackson
    - Version: 0.11.5
    - For generating and validating jwt token.
- MySQL Connector.
    - Version: 8.0.32
    - For making connection to MySql.
- Lombok
    - Version: 1.18.24
    - For getter, setters, constructor
- Spring Boot Starter Test Testing/Junit/Junit-jupiter plugin
    - Version: 3.0.2
    - For writing unit and integration test.
- Junit Jupiter plugin
    - Version: 3.0.2
    - For writing unit and integration test.
- Spring boot starter JPA
    - Version: 3.0.2

Frontend:
- **React.js**
    - **Version:** 18.2.0
    - React uses a declarative approach to building user interfaces, where developers describe what should be displayed based on the state of the application, and React takes care of updating the UI when the state changes. This makes it easier to build complex, interactive user interfaces, so we used it.
- **React-Bootstrap**
    - **Version:** 2.7.2

    - React Bootstrap is a popular front-end framework that allows developers to build responsive and mobile-first web applications using the React JavaScript library. It is a collection of reusable UI components built on top of the Bootstrap CSS framework.
- **React-dom**
    - **Version:** 18.2.0
    - React DOM is a package in the React library that provides an interface for working with the Document Object Model (DOM). The DOM is a programming interface that represents the structure and content of a web page as a tree-like structure, and allows developers to manipulate its elements, attributes, and styles using code.
- **React-prism**
    - **Version:** 4.3.2
    - React-prism is a package that help to show the code in a syntax highlighted form.
- **Bootstrap**
    - **Version:** 4.6.2
    - Bootstrap is used for making responsive and attractive UI.
- **Axios**
    - **Version:** 1.3.2
    - Axios is a popular JavaScript library that is used to make HTTP requests from a web application. It provides an easy-to-use interface for sending asynchronous HTTP requests to a server and handling the responses.
- **Jwt-Decode**
    - **Version:** 3.1.2
    - This is used to decode the jwt token on frontend.
- **React-paginate**
    - **Version:** 8.1.5
    - This is used to make the pagination on frontend for viewing different pages.

CI/CD:

Backend Pipeline:
1. Docker:latest
2. Docker:dind
3. mysql:latest
4. OpenJDK:17
5. maven:3.8.7
6. python:3.10
7. DesigniteJava.jar.

Frontend Pipeline:
1. Docker:latest
2. Dokcer:dind
3. Node:18.14.0

Build and Deployment Instructions:

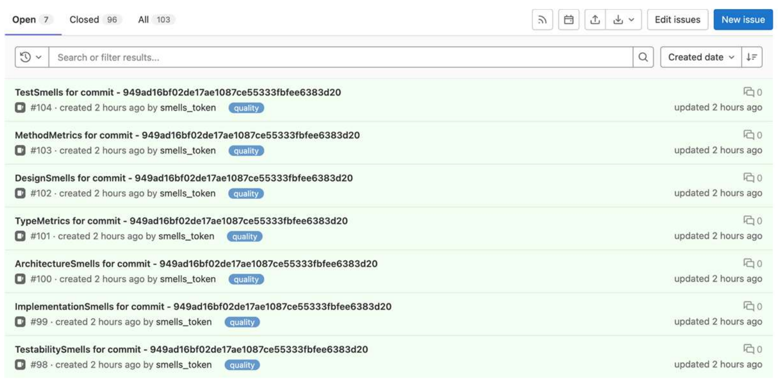
1. Identify the technology for which you want to build the application (e.g., Java, Maven, and Node.js for frontend).
2. Decide on the database you plan to use, check the connection with it, and determine the version of the database dependency you want to use in your code.
3. Set the approach for building, testing, and deploying the code. In our case, we have decided to use Docker for building and deploying the code.
4. Finalize the server on which you want to deploy the code.
5. Set up the connection with the server, create private and public SSH keys, configure the private key on the server, and install the required libraries on the server to ensure easy deployment.
6. As we have decided to use Docker, we have created a Docker registry on our server.
7. Next, make sure you create your own GitLab Runner and configure GitLab Runner on your server.
8. Then start writing CI/CD instructions for the backend code.
    - Make sure to create a local repository for Maven to save artifacts and downloaded dependencies.
    - Add default images that you want to use in each job, which can be overwritten in each job.
    - Set stages for your jobs to put them in sequence.
    - Define services and their versions that your code or pipeline will use during deployment.

    Code smells Job:

    In this job, we are running DesigniteJava.jar, which generates smells present in the code, stores the smells over a specified path, and generates artifacts for the next job.

    Issues Job:

    This job runs a Python script that has access to our GitLab boards. It pushes all the smells to the GitLab board and creates issues on the board (see Figure 1).



TestSmells for commit - 949ad16bf02de17ae1087ce55333fbfee6383d20
#104 · created 2 hours ago by smells_token quality — 0 — updated 2 hours ago

MethodMetrics for commit - 949ad16bf02de17ae1087ce55333fbfee6383d20
#103 · created 2 hours ago by smells_token quality — 0 — updated 2 hours ago

DesignSmells for commit - 949ad16bf02de17ae1087ce55333fbfee6383d20
#102 · created 2 hours ago by smells_token quality — 0 — updated 2 hours ago

TypeMetrics for commit - 949ad16bf02de17ae1087ce55333fbfee6383d20
#101 · created 2 hours ago by smells_token quality — 0 — updated 2 hours ago

ArchitectureSmells for commit - 949ad16bf02de17ae1087ce55333fbfee6383d20
#100 · created 2 hours ago by smells_token quality — 0 — updated 2 hours ago

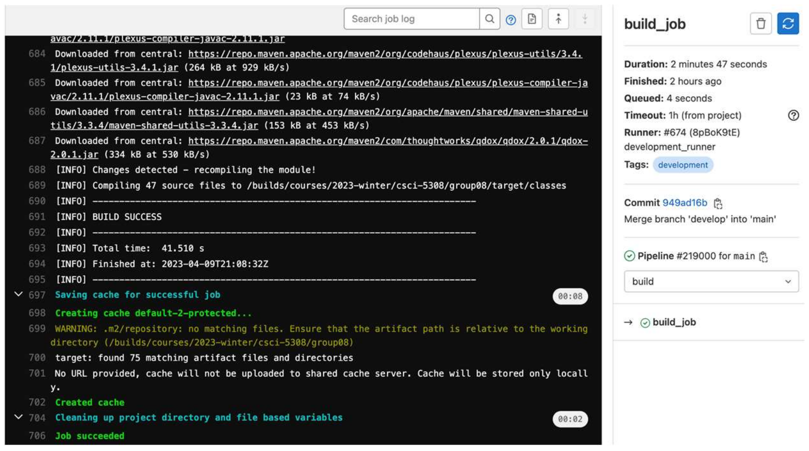
ImplementationSmells for commit - 949ad16bf02de17ae1087ce55333fbfee6383d20
#99 · created 2 hours ago by smells_token quality — 0 — updated 2 hours ago

TestabilitySmells for commit - 949ad16bf02de17ae1087ce55333fbfee6383d20
#98 · created 2 hours ago by smells_token quality — 0 — updated 2 hours ago

Figure1: Code smells created by pipeline

Build Job:

In build job we are compiling the java code (see Figure 2)



```
avac/2.11.1/plexus-compiler-javac-2.11.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/3.4.1/plexus-utils-3.4.1.jar (264 kB at 929 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-javac/2.11.1/plexus-compiler-javac-2.11.1.jar (23 kB at 74 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-utils/3.3.4/maven-shared-utils-3.3.4.jar (153 kB at 453 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/thoughtworks/qdox/qdox/2.0.1/qdox-2.0.1.jar (334 kB at 530 kB/s)
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 47 source files to /builds/courses/2023-winter/csci-5308/group08/target/classes
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  41.510 s
[INFO] Finished at: 2023-04-09T21:08:32Z
[INFO] ------------------------------------------------------------------------
Saving cache for successful job                                          00:08
Creating cache default-2-protected...
WARNING: .m2/repository: no matching files. Ensure that the artifact path is relative to the working directory (/builds/courses/2023-winter/csci-5308/group08)
target: found 75 matching artifact files and directories
No URL provided, cache will not be uploaded to shared cache server. Cache will be stored only locally.
Created cache
Cleaning up project directory and file based variables                   00:02
Job succeeded
```

build_job

Duration: 2 minutes 47 seconds
Finished: 2 hours ago
Queued: 4 seconds
Timeout: 1h (from project)
Runner: #674 (8pBoK9tE) development_runner
Tags: development

Commit 949ad16b
Merge branch 'develop' into 'main'

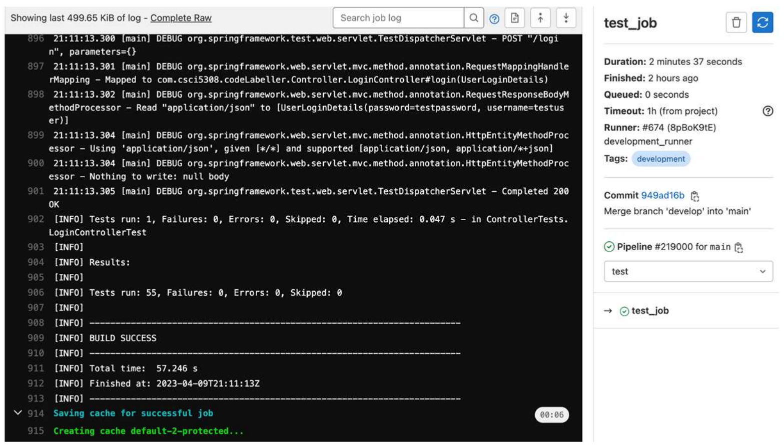
Pipeline #219000 for main
build

→ build_job

Figure 2: Build job done by pipeline

Test Job:

In this job we are running all the test cases present in the code **(see Figure 3).**



Showing last 499.65 KiB of log - Complete Raw

```
21:11:13.300 [main] DEBUG org.springframework.test.web.servlet.TestDispatcherServlet - POST "/login", parameters={}
21:11:13.301 [main] DEBUG org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerMapping - Mapped to com.csci5308.codeLabeller.Controller.LoginController#login(UserLoginDetails)
21:11:13.302 [main] DEBUG org.springframework.web.servlet.mvc.method.annotation.RequestResponseBodyMethodProcessor - Read "application/json" to [UserLoginDetails(password=testpassword, username=testuser)]
21:11:13.304 [main] DEBUG org.springframework.web.servlet.mvc.method.annotation.HttpEntityMethodProcessor - Using 'application/json', given [*/*] and supported [application/json, application/*+json]
21:11:13.304 [main] DEBUG org.springframework.web.servlet.mvc.method.annotation.HttpEntityMethodProcessor - Nothing to write: null body
21:11:13.305 [main] DEBUG org.springframework.test.web.servlet.TestDispatcherServlet - Completed 200 OK
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.047 s - in ControllerTests.LoginControllerTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 55, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  57.246 s
[INFO] Finished at: 2023-04-09T21:11:13Z
[INFO] ------------------------------------------------------------------------
Saving cache for successful job                                          00:06
Creating cache default-2-protected...
```

test_job

Duration: 2 minutes 37 seconds
Finished: 2 hours ago
Queued: 0 seconds
Timeout: 1h (from project)
Runner: #674 (8pBoK9tE) development_runner
Tags: development

Commit 949ad16b
Merge branch 'develop' into 'main'

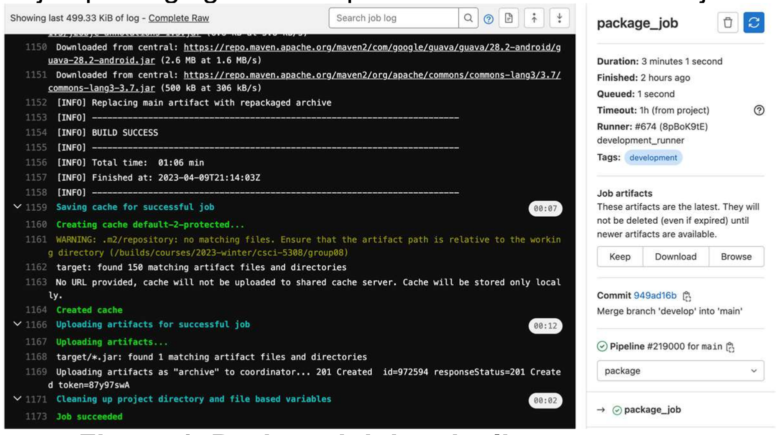
Pipeline #219000 for main
test

→ test_job

Figure 3: Test job by pipeline

Package Job:

This job packaging all the dependencies and create the jar file **(see Figure 4).**



Showing last 499.33 KiB of log - Complete Raw

```
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 1.6 MB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/commons/commons-lang3/3.7/commons-lang3-3.7.jar (500 kB at 306 kB/s)
[INFO] Replacing main artifact with repackaged archive
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  01:06 min
[INFO] Finished at: 2023-04-09T21:14:03Z
[INFO] ------------------------------------------------------------------------
Saving cache for successful job                                          00:07
Creating cache default-2-protected...
WARNING: .m2/repository: no matching files. Ensure that the artifact path is relative to the working directory (/builds/courses/2023-winter/csci-5308/group08)
target: found 150 matching artifact files and directories
No URL provided, cache will not be uploaded to shared cache server. Cache will be stored only locally.
Created cache
Uploading artifacts for successful job                                   00:12
Uploading artifacts...
target/*.jar: found 1 matching artifact files and directories
Uploading artifacts as "archive" to coordinator... 201 Created  id=972594 responseStatus=201 Created token=87y97swA
Cleaning up project directory and file based variables                   00:02
Job succeeded
```

package_job

Duration: 3 minutes 1 second
Finished: 2 hours ago
Queued: 1 second
Timeout: 1h (from project)
Runner: #674 (8pBoK9tE) development_runner
Tags: development

Job artifacts
These artifacts are the latest. They will not be deleted (even if expired) until newer artifacts are available.

Commit 949ad16b
Merge branch 'develop' into 'main'

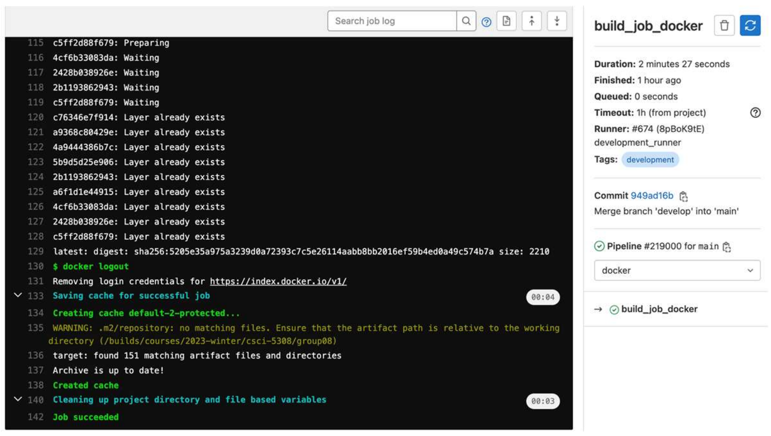
Pipeline #219000 for main
package

→ package_job

Figure 4: Package job by pipeline

Docker job:

Here, we are logging into the Docker registry. After that, we build a Docker image of the Java code. Docker uses the Dockerfile present in the code to build the image according to the instructions present in the Dockerfile (see Figure).



```
c5ff2d88f679: Preparing
4cf6b33083da: Waiting
2428b038926e: Waiting
2b1193862943: Waiting
c5ff2d88f679: Waiting
c76346e7f914: Layer already exists
a9368c80429e: Layer already exists
4a9444386b7c: Layer already exists
5b9d5d25e906: Layer already exists
2b1193862943: Layer already exists
a6f1d1e44915: Layer already exists
4cf6b33083da: Layer already exists
2428b038926e: Layer already exists
c5ff2d88f679: Layer already exists
latest: digest: sha256:5205e35a975a3239d0a72393c7c5e26114aabb8bb2016ef59b4ed0a49c574b7a size: 2210
$ docker logout
Removing login credentials for https://index.docker.io/v1/
Saving cache for successful job                                                          00:04
Creating cache default-2-protected...
WARNING: .m2/repository: no matching files. Ensure that the artifact path is relative to the working
directory (/builds/courses/2023-winter/csci-5308/group08)
target: found 151 matching artifact files and directories
Archive is up to date!
Created cache
Cleaning up project directory and file based variables                                   00:03
Job succeeded
```

build_job_docker

Duration: 2 minutes 27 seconds
Finished: 1 hour ago
Queued: 0 seconds
Timeout: 1h (from project)
Runner: #674 (8pBoK9tE) development_runner
Tags: development

Commit 949ad16b
Merge branch 'develop' into 'main'

Pipeline #219000 for main

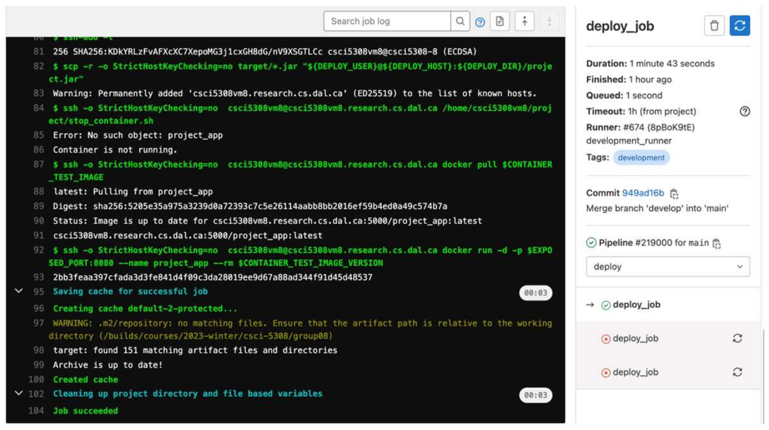
docker

→ build_job_docker

Figure 5: Code smells created by pipeline

Deploy Job:

This job basically performs three actions. Firstly, it creates a connection with our server using the SSH key. We have stored the Base64-encoded SSH key in a GitLab variable. Secondly, it copies the JAR file from artifacts to the server for backup. Thirdly, it reruns the Docker image already running on the server and reflects the changes on the server (see Figure 6). Our backend runs on port 8081. We are handling dangling images in Docker.



```
256 SHA256:KDkYRLzFvAFXcXC7XepoMG3j1cxGH8dG/nV9XSGTLCc csci5308vm8@csci5308-8 (ECDSA)
$ scp -r -o StrictHostKeyChecking=no target/*.jar "${DEPLOY_USER}@${DEPLOY_HOST}:${DEPLOY_DIR}/proje
ct.jar"
Warning: Permanently added 'csci5308vm8.research.cs.dal.ca' (ED25519) to the list of known hosts.
$ ssh -o StrictHostKeyChecking=no  csci5308vm8@csci5308vm8.research.cs.dal.ca /home/csci5308vm8/proj
ect/stop_container.sh
Error: No such object: project_app
Container is not running.
$ ssh -o StrictHostKeyChecking=no  csci5308vm8@csci5308vm8.research.cs.dal.ca docker pull $CONTAINER
_TEST_IMAGE
latest: Pulling from project_app
Digest: sha256:5205e35a975a3239d0a72393c7c5e26114aabb8bb2016ef59b4ed0a49c574b7a
Status: Image is up to date for csci5308vm8.research.cs.dal.ca:5000/project_app:latest
csci5308vm8.research.cs.dal.ca:5000/project_app:latest
$ ssh -o StrictHostKeyChecking=no  csci5308vm8@csci5308vm8.research.cs.dal.ca docker run -d -p $EXPO
SED_PORT:8080 --name project_app --rm $CONTAINER_TEST_IMAGE_VERSION
2bb3feaa397cfada3d3fe841d4f09c3da28019ee9d67a88ad344f91d45d48537
Saving cache for successful job                                                          00:03
Creating cache default-2-protected...
WARNING: .m2/repository: no matching files. Ensure that the artifact path is relative to the working
directory (/builds/courses/2023-winter/csci-5308/group08)
target: found 151 matching artifact files and directories
Archive is up to date!
Created cache
Cleaning up project directory and file based variables                                   00:03
Job succeeded
```

deploy_job

Duration: 1 minute 43 seconds
Finished: 1 hour ago
Queued: 1 second
Timeout: 1h (from project)
Runner: #674 (8pBoK9tE) development_runner
Tags: development

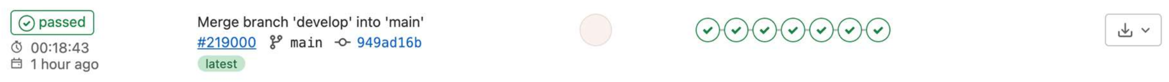
Commit 949ad16b
Merge branch 'develop' into 'main'

Pipeline #219000 for main

deploy

→ deploy_job
deploy_job
deploy_job

Figure 6: Code deployed successfully.



passed
00:18:43
1 hour ago

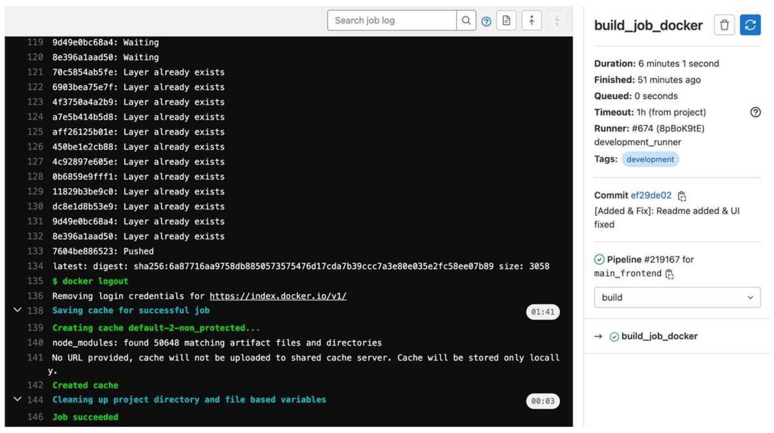
Merge branch 'develop' into 'main'
#219000 main 949ad16b
latest

Figure2: Whole pipeline run successfully.

9. Now write frontend pipeline

Build Job:

In the frontend build job, we first log in to the Docker registry, build an image on GitLab Runner, and then push that image to the Docker registry. The Docker image will be created according to the instructions present in the Dockerfile.



```
9d49e0bc68a4: Waiting
8e396a1aad50: Waiting
70c5854ab5fe: Layer already exists
6903bea75e7f: Layer already exists
4f3750a4a2b9: Layer already exists
a7e5b414b5d8: Layer already exists
aff26125b01e: Layer already exists
450be1e2cb88: Layer already exists
4c92897e605e: Layer already exists
0b6859e9fff1: Layer already exists
11829b3be9c0: Layer already exists
dc8e1d8b53e9: Layer already exists
9d49e0bc68a4: Layer already exists
8e396a1aad50: Layer already exists
7604be886523: Pushed
latest: digest: sha256:6a87716aa9758db8850573575476d17cda7b39ccc7a3e80e035e2fc58ee07b89 size: 3058
$ docker logout
Removing login credentials for https://index.docker.io/v1/
Saving cache for successful job                                                          01:41
Creating cache default-2-non_protected...
node_modules: found 50648 matching artifact files and directories
No URL provided, cache will not be uploaded to shared cache server. Cache will be stored only locall
y.
Created cache
Cleaning up project directory and file based variables                                   00:03
Job succeeded
```

build_job_docker

Duration: 6 minutes 1 second
Finished: 51 minutes ago
Queued: 0 seconds
Timeout: 1h (from project)
Runner: #674 (8pBoK9tE) development_runner
Tags: development

Commit ef29de02
[Added & Fix]: Readme added & UI fixed

Pipeline #219167 for main_frontend

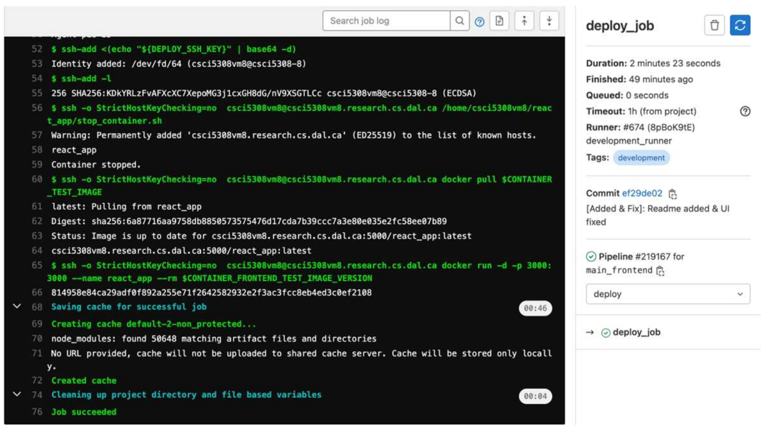
build

→ build_job_docker

Figure 7: Code smells created by pipeline.

Deploy Job:

In front-end deploy we either rerun the running container or we create a new container and run the image inside it. We are also handling dangling images.



```
$ ssh-add <(echo "${DEPLOY_SSH_KEY}" | base64 -d)
Identity added: /dev/fd/64 (csci5308vm8@csci5308-8)
$ ssh-add -l
256 SHA256:KDkYRLzFvAFXcXC7XepoMG3j1cxGH8dG/nV9XSGTLCc csci5308vm8@csci5308-8 (ECDSA)
$ ssh -o StrictHostKeyChecking=no  csci5308vm8@csci5308vm8.research.cs.dal.ca /home/csci5308vm8/reac
t_app/stop_container.sh
Warning: Permanently added 'csci5308vm8.research.cs.dal.ca' (ED25519) to the list of known hosts.
react_app
Container stopped.
$ ssh -o StrictHostKeyChecking=no  csci5308vm8@csci5308vm8.research.cs.dal.ca docker pull $CONTAINER
_TEST_IMAGE
latest: Pulling from react_app
Digest: sha256:6a87716aa9758db8850573575476d17cda7b39ccc7a3e80e035e2fc58ee07b89
Status: Image is up to date for csci5308vm8.research.cs.dal.ca:5000/react_app:latest
csci5308vm8.research.cs.dal.ca:5000/react_app:latest
$ ssh -o StrictHostKeyChecking=no  csci5308vm8@csci5308vm8.research.cs.dal.ca docker run -d -p 3000:
3000 --name react_app --rm $CONTAINER_FRONTEND_TEST_IMAGE_VERSION
814958e84ca29adf0f892a255e71f2642582932e2f3ac3fcc8eb4ed3c0ef2108
Saving cache for successful job                                                         00:46
Creating cache default-2-non_protected...
node_modules: found 50648 matching artifact files and directories
No URL provided, cache will not be uploaded to shared cache server. Cache will be stored only locall
y.
Created cache
Cleaning up project directory and file based variables                                  00:04
Job succeeded
```

deploy_job

Duration: 2 minutes 23 seconds
Finished: 49 minutes ago
Queued: 0 seconds
Timeout: 1h (from project)
Runner: #674 (8pBoK9tE) development_runner
Tags: development

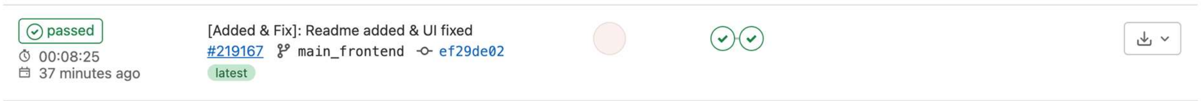
Commit ef29de02
[Added & Fix]: Readme added & UI fixed

Pipeline #219167 for `main_frontend`

deploy

→ deploy_job

Figure 8: Docker image creation



passed
00:08:25
37 minutes ago

[Added & Fix]: Readme added & UI fixed
#219167 main_frontend ef29de02
latest

Figure 9: Frontend pipeline ran successfully.

Actors:

Annotator:

1. Can Login/Signup.
2. Can select the survey from the bunch.
3. Can start a survey according to their wish.
4. After the survey is started, can go through each snippet of that particular survey and start annotating the snippet (via highlighting + tagging).

Admin:

1. Can Login/Signup.
2. Can create a survey.
3. Can upload snippets for that survey and create annotations for that.
4. Can view surveys previously created.
5. Can open a survey and then see tags and highlighted snippets under that particular survey.

Survey:

1. It contains multiple snippets.
2. It contains multiple annotations.
3. It also has a language specified.
4. It has a threshold specified.

Snippet:

1. It has code uploaded here.

Annotations:

1. Annotations are created by admin.
2. They are used by an annotator to tag or highlight a snippet.

Highlight:

1. It has an annotation, range of code highlighted along with snippet id and survey id.

Use Cases: Flow:

The general flow here will demonstrate a complete and most general use case for both annotator and admin:

Step 1:
Admin Signs Up:
Admin chooses a acceptable username and password, also they chooses their role as admin to sign up.

Step 2:
Admin starts creating a survey:
Admin lands on their home page where they can see two options, to see or to create surveys.
Admin chooses to create survey.

Step 3:
Admin creates a survey:
Admin upload snippets(bunch of .java files), chooses language as .java, and upload snippets from their machine.
Admin adds annotations to the survey and submit and logs out.

Step 4:

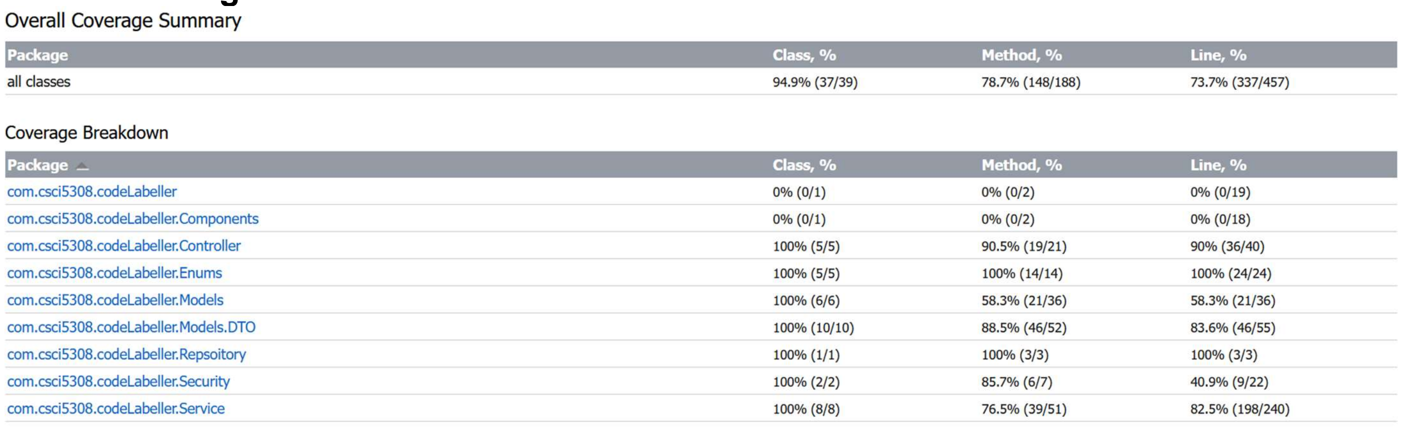
Annotator signs up/log in:
Annotator lands on their home page where they can see list of surveys created.
Annotator chooses a survey and start the pagination to start annotating the survey.

Step 5:
Annotator goes through each snippet of the chosen survey:
Annotator sees one snippet at a time.
Annotator highlights the code and marks it with an annotation.
Annotator chooses a tag for the snippet and click on "next" button to move to next snippet.

Step 6:
Annotator submits:
Annotator after annotating all the snippets, submits.

Step 7:
Admin logs in adn see the annotated snippets:
Admin logs in.
Admin lands on home page.
Admin chooses "View Surveys".
Admin clicks on the desired survey.
Admin can see the annoated tags for each snippet under that survey.
Admin can start the pagination for each snippet to see the highlighted part.
Admin logs out.

Code Coverage:



Overall Coverage Summary

Package	Class, %	Method, %	Line, %
all classes	94.9% (37/39)	78.7% (148/188)	73.7% (337/457)

Coverage Breakdown

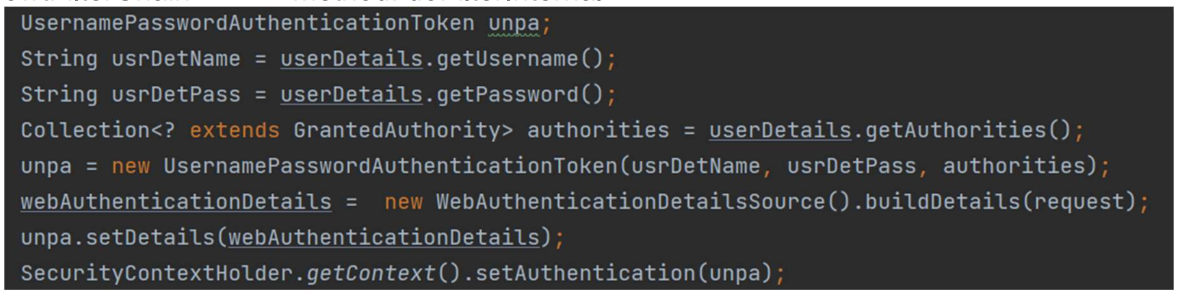
Package	Class, %	Method, %	Line, %
com.csci5308.codeLabeller	0% (0/1)	0% (0/2)	0% (0/19)
com.csci5308.codeLabeller.Components	0% (0/1)	0% (0/2)	0% (0/18)
com.csci5308.codeLabeller.Controller	100% (5/5)	90.5% (19/21)	90% (36/40)
com.csci5308.codeLabeller.Enums	100% (5/5)	100% (14/14)	100% (24/24)
com.csci5308.codeLabeller.Models	100% (6/6)	58.3% (21/36)	58.3% (21/36)
com.csci5308.codeLabeller.Models.DTO	100% (10/10)	88.5% (46/52)	83.6% (46/55)
com.csci5308.codeLabeller.Repsoitory	100% (1/1)	100% (3/3)	100% (3/3)
com.csci5308.codeLabeller.Security	100% (2/2)	85.7% (6/7)	40.9% (9/22)
com.csci5308.codeLabeller.Service	100% (8/8)	76.5% (39/51)	82.5% (198/240)

Figure 10: Code Coverage.

Code Smells:

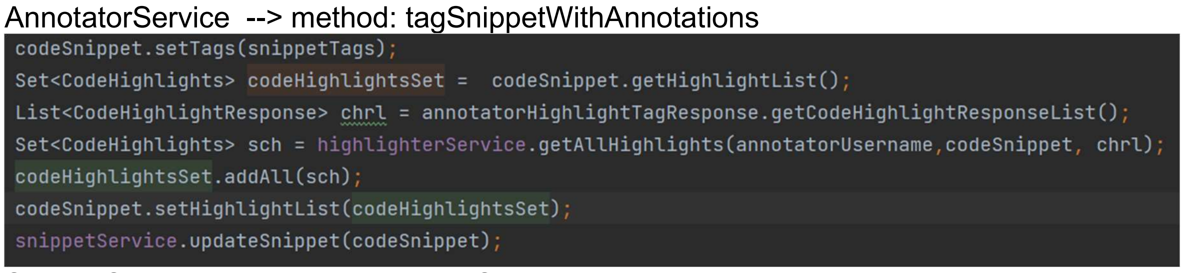
Implementation Smell:
Long Statement:

- JwtFilterChain, AnnotatorService, SnippetService
    - To shorten the statement length, some arguments were extracted out in separate variables
    - JwtFilterChain --> method: doFilterInternal



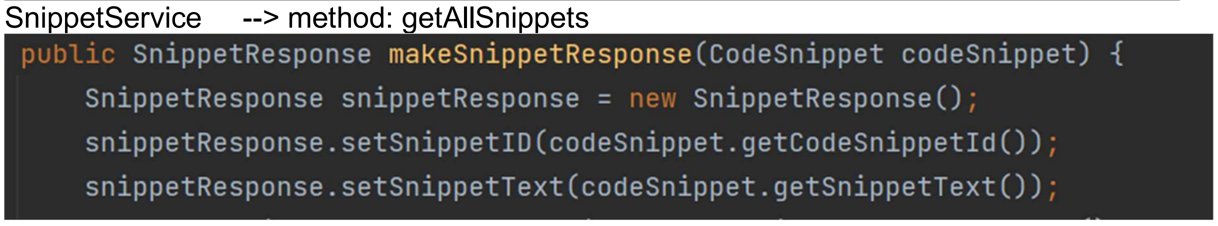
```
UsernamePasswordAuthenticationToken unpa;
String usrDetName = userDetails.getUsername();
String usrDetPass = userDetails.getPassword();
Collection<? extends GrantedAuthority> authorities = userDetails.getAuthorities();
unpa = new UsernamePasswordAuthenticationToken(usrDetName, usrDetPass, authorities);
webAuthenticationDetails =  new WebAuthenticationDetailsSource().buildDetails(request);
unpa.setDetails(webAuthenticationDetails);
SecurityContextHolder.getContext().setAuthentication(unpa);
```

    - AnnotatorService --> method: tagSnippetWithAnnotations



```
codeSnippet.setTags(snippetTags);
Set<CodeHighlights> codeHighlightsSet =  codeSnippet.getHighlightList();
List<CodeHighlightResponse> chrl = annotatorHighlightTagResponse.getCodeHighlightResponseList();
Set<CodeHighlights> sch = highlighterService.getAllHighlights(annotatorUsername,codeSnippet, chrl);
codeHighlightsSet.addAll(sch);
codeSnippet.setHighlightList(codeHighlightsSet);
snippetService.updateSnippet(codeSnippet);
```

    - SnippetService --> method: getAllSnippets



```
public SnippetResponse makeSnippetResponse(CodeSnippet codeSnippet) {
    SnippetResponse snippetResponse = new SnippetResponse();
    snippetResponse.setSnippetID(codeSnippet.getCodeSnippetId());
    snippetResponse.setSnippetText(codeSnippet.getSnippetText());
```

- Magic Number:
  - Extracted out all numbers and used them via Enums.
  - SnippetService   --> method: getSnippetPage

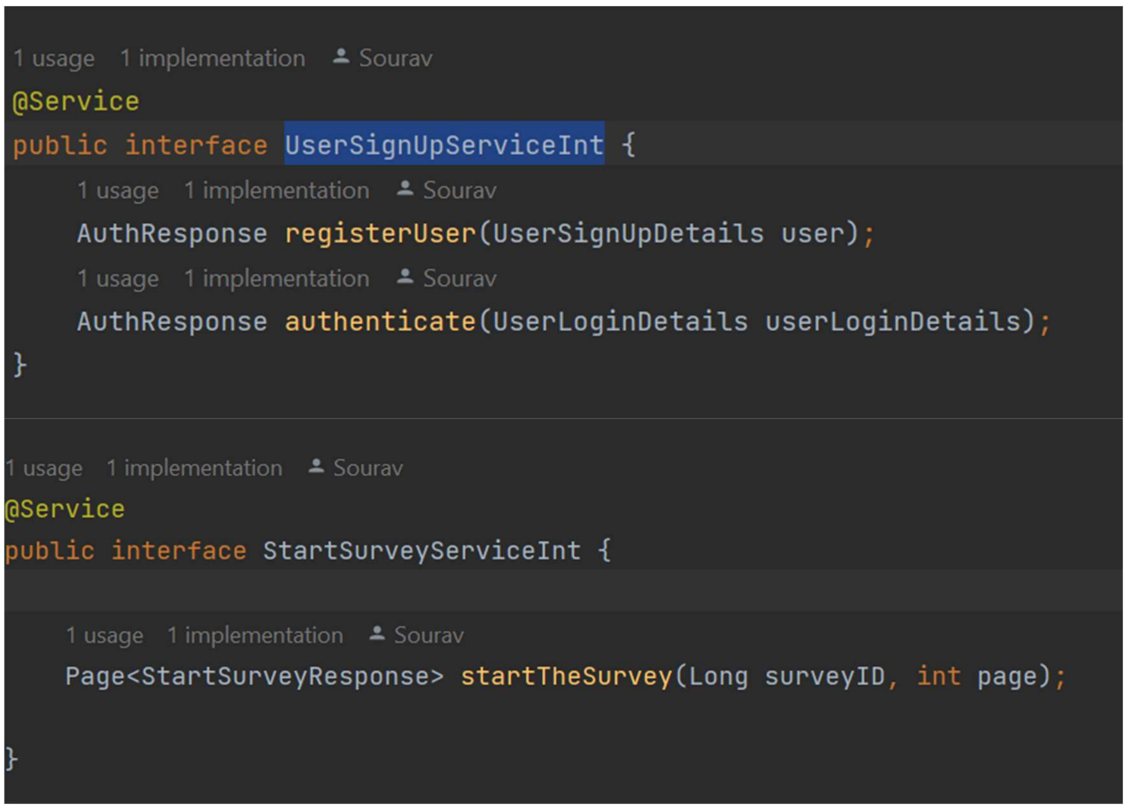
Design Smell:
- Unnecessary Abstraction, Broken Modularization
  - All the models are being marked as unnecessary abstraction, broken modularization, hence it is non-actionable.

Architecture Smell:
- Unstable Dependency
  - Model, Repositoty, Package
  - Components package
  - This package contains 'JwtFilterChain' which inherently depends on many components cause it needs to extract out user information from jwt token ad authenticate.
- Feature Concentration
  - The MVC pattern of Spring boot framework forces to put all models in one package and same for repositories and services.

S.O.L.I.D Principles

Single Responsibility Principle:
All classes in the packages are handling a single feature. There are controllers, services, repositories for each and every model created. Making the code modular.

Open/Closed Principle:
Interfaces has been used on multiple locations in the code.
For example:
The Package:
1. Repository has almost all interfaces and all the services using their respective repositories are dependent on interfaces instead of implementations.
2. The services has interfaces whioch they implement, it gives the developer felxibility to add different kind of servies in future.



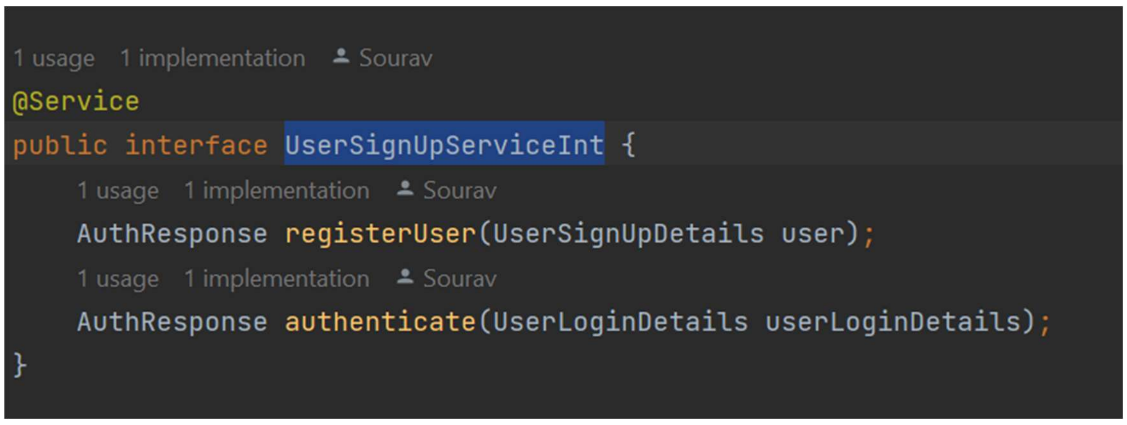
```
1 usage   1 implementation   Sourav
@Service
public interface UserSignUpServiceInt {
    1 usage   1 implementation   Sourav
    AuthResponse registerUser(UserSignUpDetails user);
    1 usage   1 implementation   Sourav
    AuthResponse authenticate(UserLoginDetails userLoginDetails);
}
```

```
1 usage   1 implementation   Sourav
@Service
public interface StartSurveyServiceInt {

    1 usage   1 implementation   Sourav
    Page<StartSurveyResponse> startTheSurvey(Long surveyID, int page);

}
```

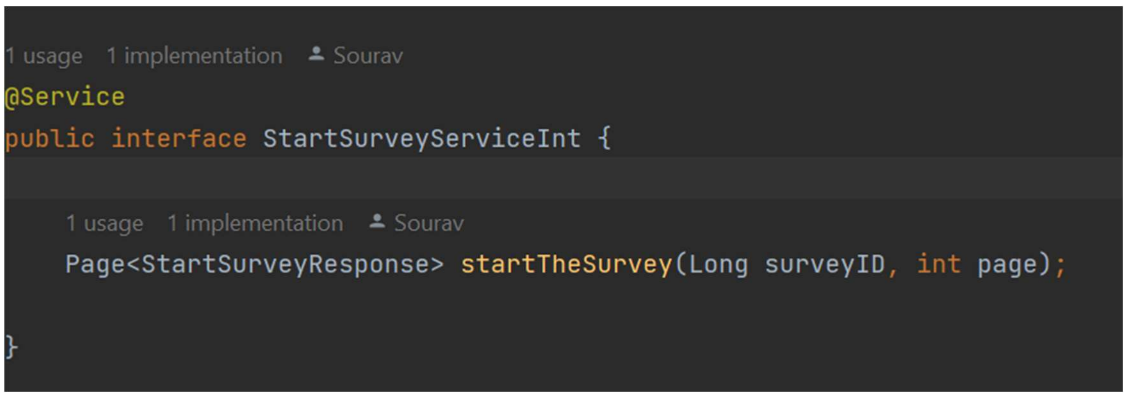
Liskov Substitution Principle:
Instances where inheritance (is-a relationship) exists, it has been made sure that the child doesn't override and completely changes the implementation of methods as its parent.
For example:
1. CodeHighlightResponse extends CodeHighlights, where it child class isnt going rebellious against the parent.
2. Intefaces created in services are following LSP.

Interface Segregation Principle:
As mentioned earlier, for all the interfaces created, none of their implementation has a method which is no use to them.

This princple is followed in:
1. Interfaces have been created for repsistories and services.
2. All the implementations of these interfaces are not overriding any methods that are not of their use.



```
1 usage   1 implementation   Sourav
@Service
public interface UserSignUpServiceInt {
    1 usage   1 implementation   Sourav
    AuthResponse registerUser(UserSignUpDetails user);
    1 usage   1 implementation   Sourav
    AuthResponse authenticate(UserLoginDetails userLoginDetails);
}
```

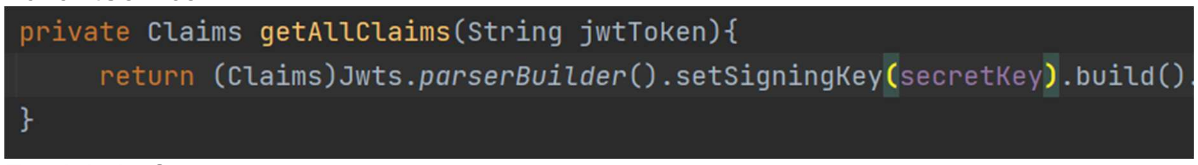


```
1 usage   1 implementation   Sourav
@Service
public interface StartSurveyServiceInt {

    1 usage   1 implementation   Sourav
    Page<StartSurveyResponse> startTheSurvey(Long surveyID, int page);

}
```

Dependency Injection Principle:
Autowire functionality and annotations like @Service and @Respository and @Controller has been used in this spring boot code. Which enables the dependency injection and and makes code more more loosely coupled.

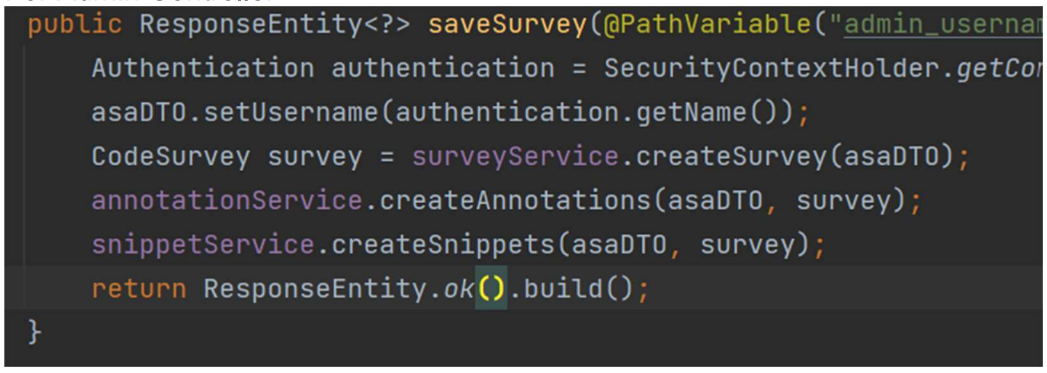
Design Principles:

Builder Pattern
It has been implemented at multiple instances.
1. For JwtService



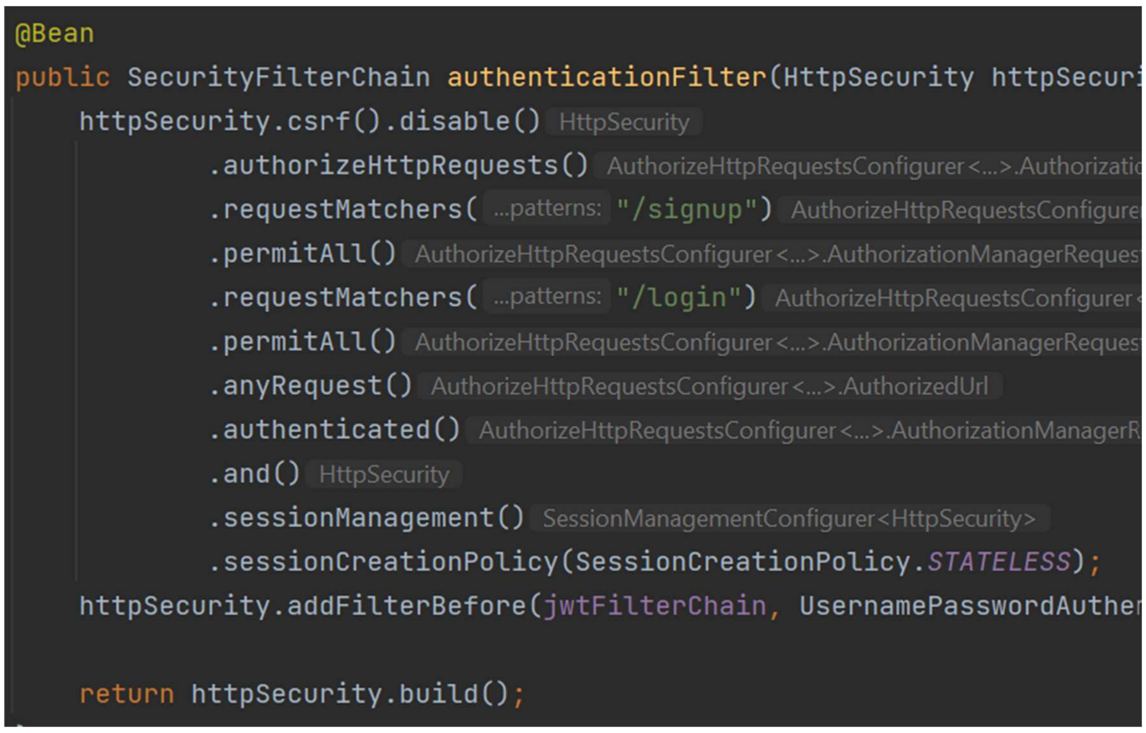
```
private Claims getAllClaims(String jwtToken){
    return (Claims)Jwts.parserBuilder().setSigningKey(secretKey).build().
}
```

2.
3. For Admin Controller.



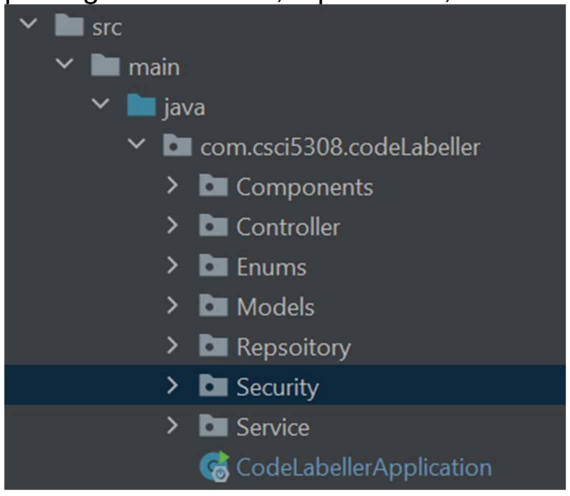
```
public ResponseEntity<?> saveSurvey(@PathVariable("admin_usernam
    Authentication authentication = SecurityContextHolder.getCo
    asaDTO.setUsername(authentication.getName());
    CodeSurvey survey = surveyService.createSurvey(asaDTO);
    annotationService.createAnnotations(asaDTO, survey);
    snippetService.createSnippets(asaDTO, survey);
    return ResponseEntity.ok().build();
}
```

4.
5. For security configuration.

6. 

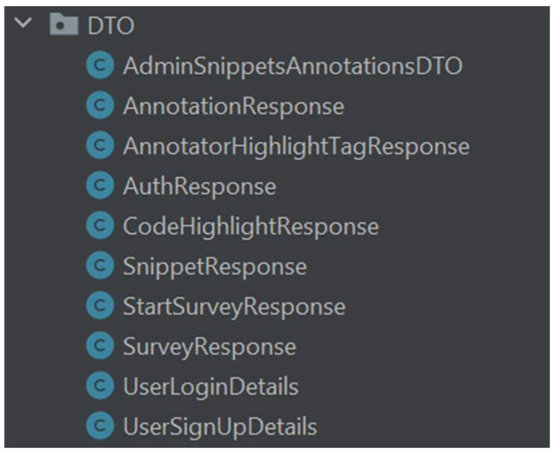
```
@Bean
public SecurityFilterChain authenticationFilter(HttpSecurity httpSecuri
    httpSecurity.csrf().disable() HttpSecurity
            .authorizeHttpRequests() AuthorizeHttpRequestsConfigurer<...>.Authorizatic
            .requestMatchers( ...patterns: "/signup") AuthorizeHttpRequestsConfigure
            .permitAll() AuthorizeHttpRequestsConfigurer<...>.AuthorizationManagerReques
            .requestMatchers( ...patterns: "/login") AuthorizeHttpRequestsConfigurer<
            .permitAll() AuthorizeHttpRequestsConfigurer<...>.AuthorizationManagerReques
            .anyRequest() AuthorizeHttpRequestsConfigurer<...>.AuthorizedUrl
            .authenticated() AuthorizeHttpRequestsConfigurer<...>.AuthorizationManagerR
            .and() HttpSecurity
            .sessionManagement() SessionManagementConfigurer<HttpSecurity>
            .sessionCreationPolicy(SessionCreationPolicy.STATELESS);
    httpSecurity.addFilterBefore(jwtFilterChain, UsernamePasswordAuther

    return httpSecurity.build();
```

MVC Pattern:
It is a pattern that has been implemented, because this is a spring boot web application, we have separate packages for models, repositories, controllers and services.



```
src
  main
    java
      com.csci5308.codeLabeller
        Components
        Controller
        Enums
        Models
        Repsoitory
        Security
        Service
        CodeLabellerApplication
```

DTO (Data Transfer Object) Pattern:
Since this is a web application, that needs to dialog back-forth with the frontend, it needs to consolidate data from multiple models and send it back to frontend or receive it from frontend.
The best pattern to perform it is DTO pattern , which has been implemented alongside many controllers.



```
DTO
  AdminSnippetsAnnotationsDTO
  AnnotationResponse
  AnnotatorHighlightTagResponse
  AuthResponse
  CodeHighlightResponse
  SnippetResponse
  StartSurveyResponse
  SurveyResponse
  UserLoginDetails
  UserSignUpDetails
```


APIs

Hostname csci5308vm8.research.cs.dal.ca:8081/

Admin Controller: Common End Point: /admin/

- API 1:
  - Method: GET
  - URI: {admin_username}/survey/{survey_id}/annotation/all
  - Description:

    This url returns all the annotations created by Admin * It expects admin username and surveyId as path variables for the call * @param username

* @param username
* @param surveyId
* @return List of AnnotationResponse

- API 2:
  - Method: GET
  - URI: "{admin_username}/survey/{survey_id}/annotation/{id}/
  - Description:

    This url returns annotation based on the Id passed by Admin * It expects admin username, annotationId and surveyId as path variables for the call * @param username
    * @param surveyId
    * @param id annotationId
    * @return AnnotationResponse

- API 3:
  - Method: GET
  - URI: {admin_username}/survey/{survey_id}/snippet/{id}/highlightResponses/start/
  - Description:

    This url returns all the responses done by different annotators, all annotations by single annotator clubs on single page. * It expects admin username, surveyId, snippetId and page number as path variables for the call
    * @param username
    * @param surveyId
    * @param snippetId
    * @param page
    * @return Page with list of CodeHighlightResponse
- API 4:
  - Method: GET
  - URI: {admin_username}/survey/{survey_id}/snippet/{id}/taggedAnnotations/all/
  - Description:

    This url returns all the annotations tagged to a particular snippet * It expects admin username and surveyId as path variables for the call * @param username
    * @param surveyId
    * @param id snippetId
    * @return List of AnnotationResponse

- API 5:
  - Method: GET
  - URI: "{admin_username}/survey/{survey_id}/snippet/all/
  - Description:

    This url returns all the snippets created by Admin * It expects admin username and surveyId as path variables for the call * @param username
    * @param surveyId
    * @return List of SnippetResponse

- API 6:
  - Method: GET
  - URI: {admin_username}/survey/{survey_id}/snippet/{id}/
  - Description:

    This url returns snippet * It expects admin username, snippetId and surveyId as path variables for the call * @param username
    * @param surveyId
    * @param id snippetId
    * @return SnippetResponse

- API 7:
  - Method: GET
  - URI: "{admin_username}/survey/all/
  - Description:

    This url returns all the Surveys created by Admin * It expects admin username as path variables for the call
    * @param username
    * @return List of SurveyResponse

- API 8:
  - Method: GET
  - URI: "{admin_username}/survey/{id}/
  - Description:

This url returns survey by surveyId * It expects admin username and surveyId as path variables for the call * @param username
 * @param id surveyId
 * @return SurveyReponse

- API 9:
    - Method: POST
    - URI: {admin_username}/survey/
    - Description:

      This url returns status of saving of survey * It expects admin username and object of AdminSnippetsAnnotationsDTO as payload for call * @param username
       * @param asaDTO
       * @return Status of saving survey

Annotator Controller: /annotator/

- API 10:
    - Method: GET
    - URI: {annotator_username}/survey/all/
    - Description:

      This url returns all surveys * It expects annotator's username as payload for call * @param username
       * @return List of SurveyReaponse

- API 11:
    - Method: POST
    - URI: {annotator_username}/survey/{survey_id}/start/
    - Description:

      This url returns a page of StartSurveyResponse * It expects annotator's username as payload for call * @param username annotator username
       * @param surveyId
       * @param page
       * @param snippetId
       * @param annotatorHighlightTagResponse Method expect this object in body.
       * @return Page of StartSurveyResponse

- API 12:
    - Method: GET
    - URI: {annotator_username}/survey/approved/all/
    - Description:

      This url returns list of SurveyResponse * It expects annotator's username as path variable for call
       * @param username
       * @return List of SurveyResponse

- API 13:
    - Method: GET
    - URI: {annotator_username}/survey/pending/all
    - Description:

      This url returns list of pending surveys * It expects annotator's username as payload for call * @param username
       * @return List of SurveyResponse

Login Controller:

- API 14:
    - Method: POST
    - URI: /login
    - Description:

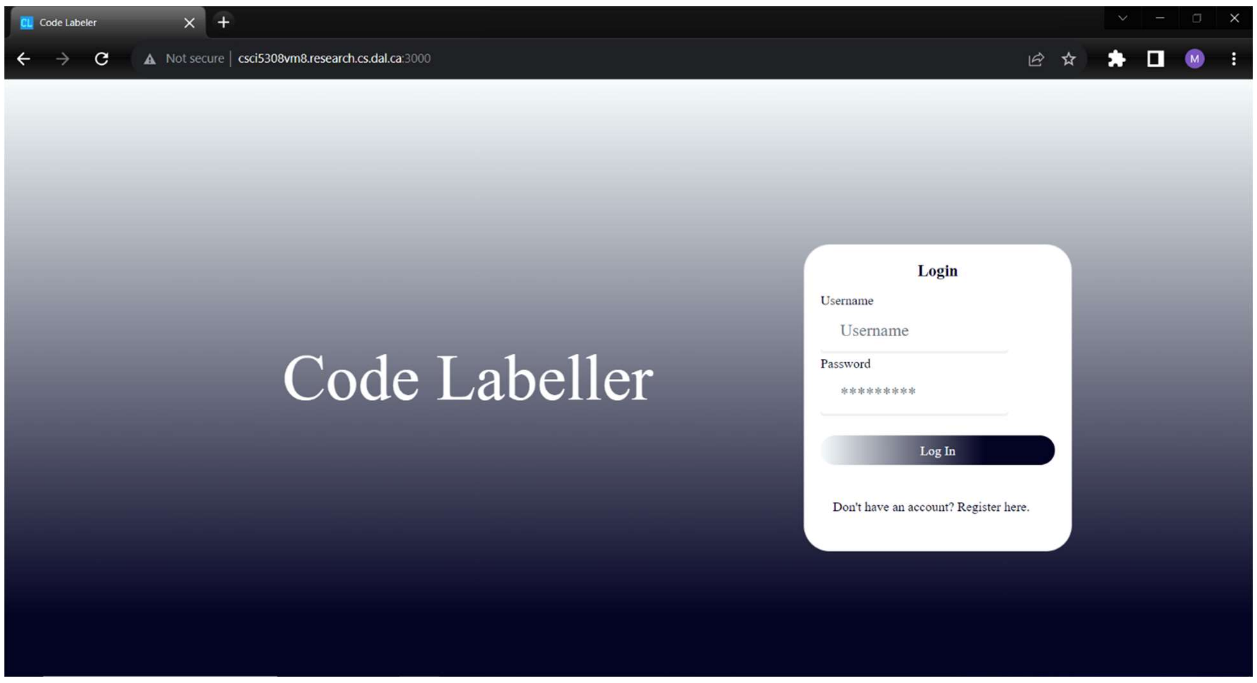
      This url returns list of pending surveys * It expects annotator's username as payload for call * @param user This is object of UserLoginDetails class
       * @return AuthResponse this returns jwtToken

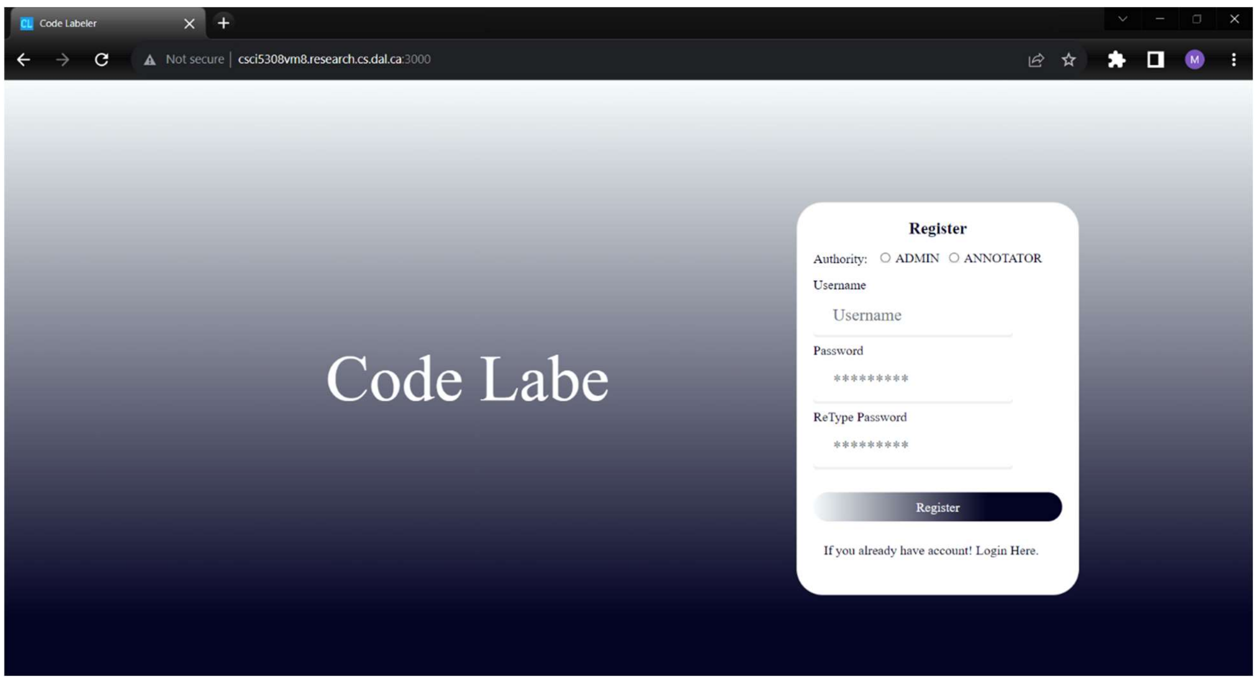
Sign up Controller:

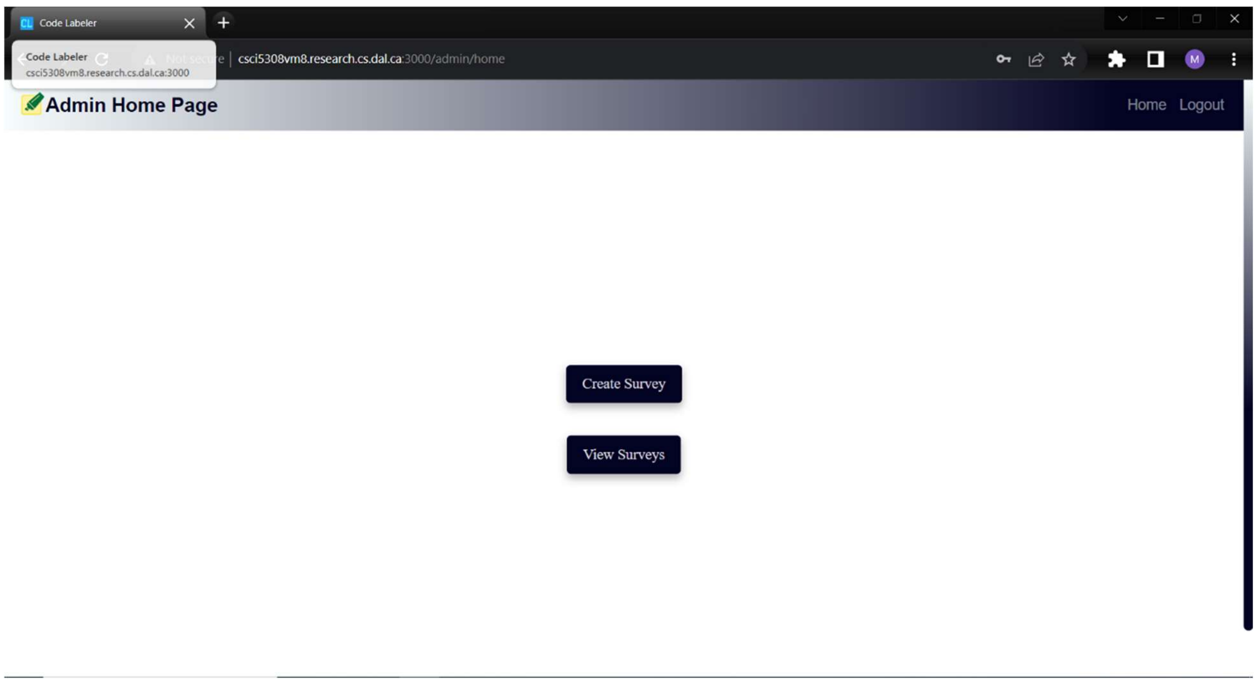
- API 15:
    - Method: POST
    - URI: /signup
    - Description:

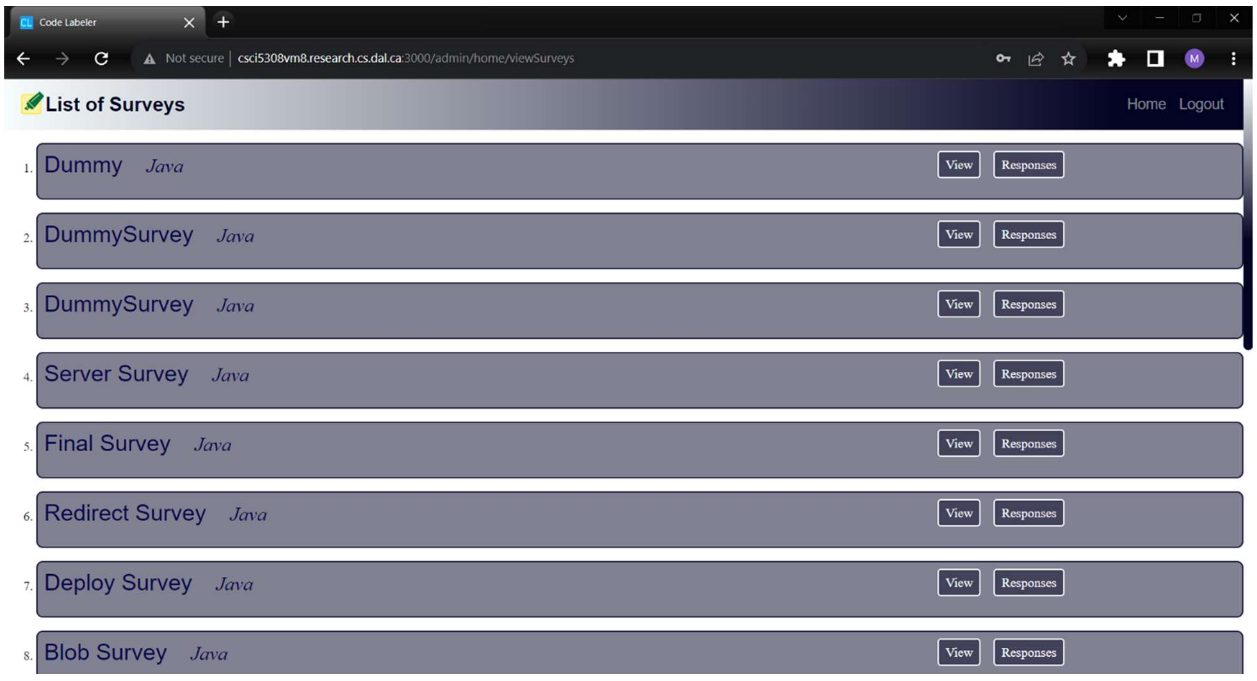
This url returns list of pending surveys * It expects annotator's username as payload for call * @param user This is object of UserSignUpDetails class
 * @return AuthResponse this returns jwtToken

Screenshots:


Code Labeler
Not secure | csci5308vm8.research.cs.dal.ca:3000
Code Labeller
Login
Username
Username
Password
****
Log In
Don't have an account? Register here.

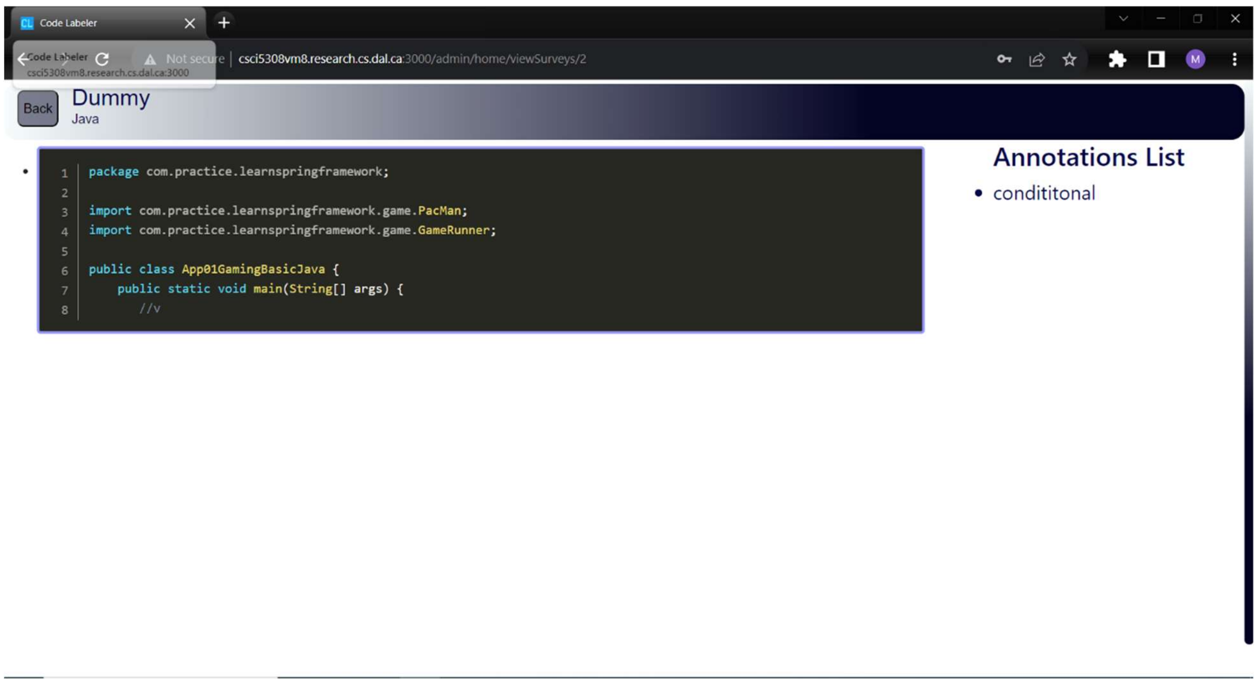

Code Labeler
Not secure | csci5308vm8.research.cs.dal.ca:3000
Code Labe
Register
Authority: ADMIN ANNOTATOR
Username
Username
Password
****
ReType Password
****
Register
If you already have account! Login Here.




Code Labeler
Not secure | csci5308vm8.research.cs.dal.ca:3000/admin/home/viewSurveys
Home
Logout

List of Surveys

1. Dummy *Java* View Responses
2. DummySurvey *Java* View Responses
3. DummySurvey *Java* View Responses
4. Server Survey *Java* View Responses
5. Final Survey *Java* View Responses
6. Redirect Survey *Java* View Responses
7. Deploy Survey *Java* View Responses
8. Blob Survey *Java* View Responses

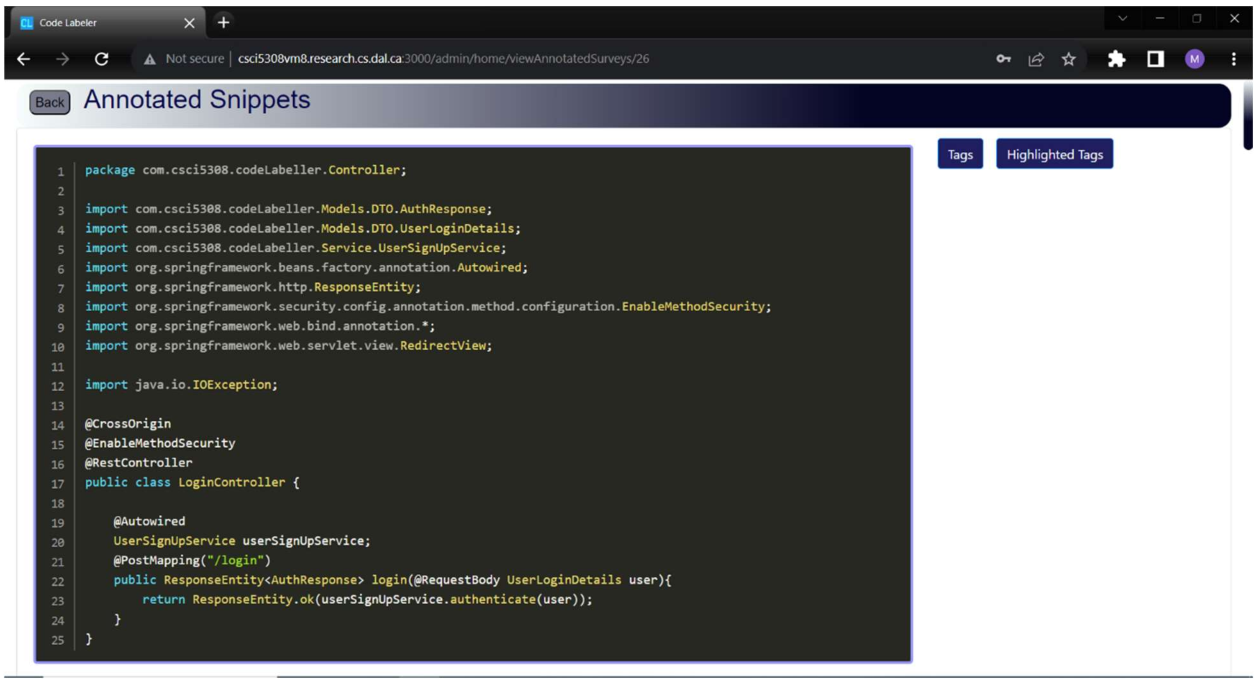

Code Labeler
Not secure | csci5308vm8.research.cs.dal.ca:3000/admin/home/viewSurveys/2
Back

Dummy
Java

```java
package com.practice.learnspringframework;

import com.practice.learnspringframework.game.PacMan;
import com.practice.learnspringframework.game.GameRunner;

public class App01GamingBasicJava {
    public static void main(String[] args) {
        //v
```

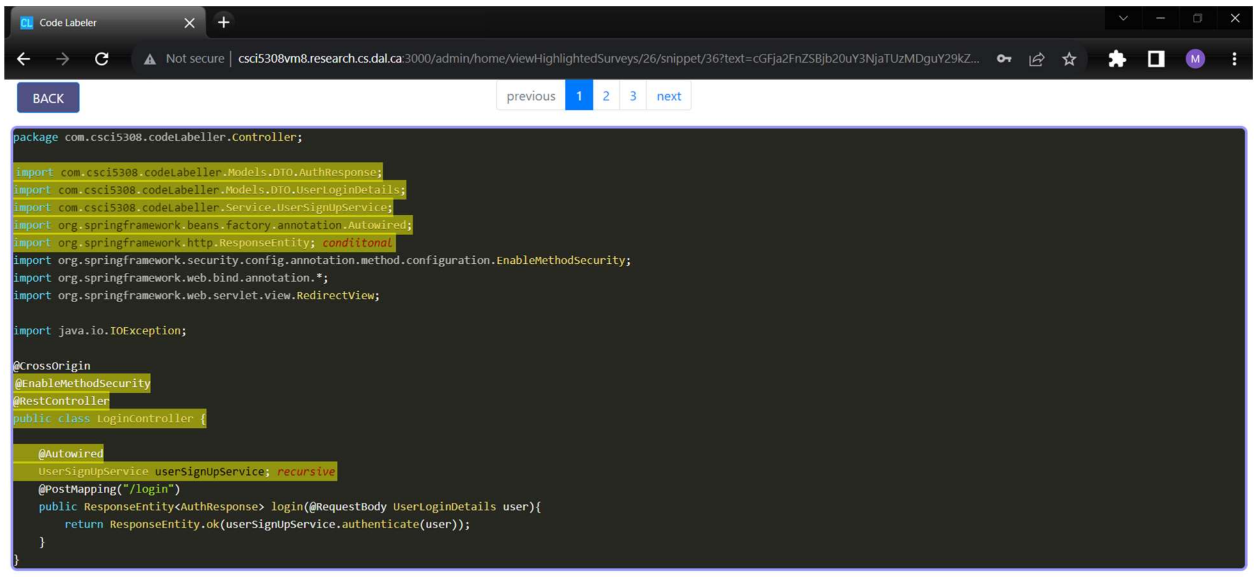
Annotations List

- condititonal

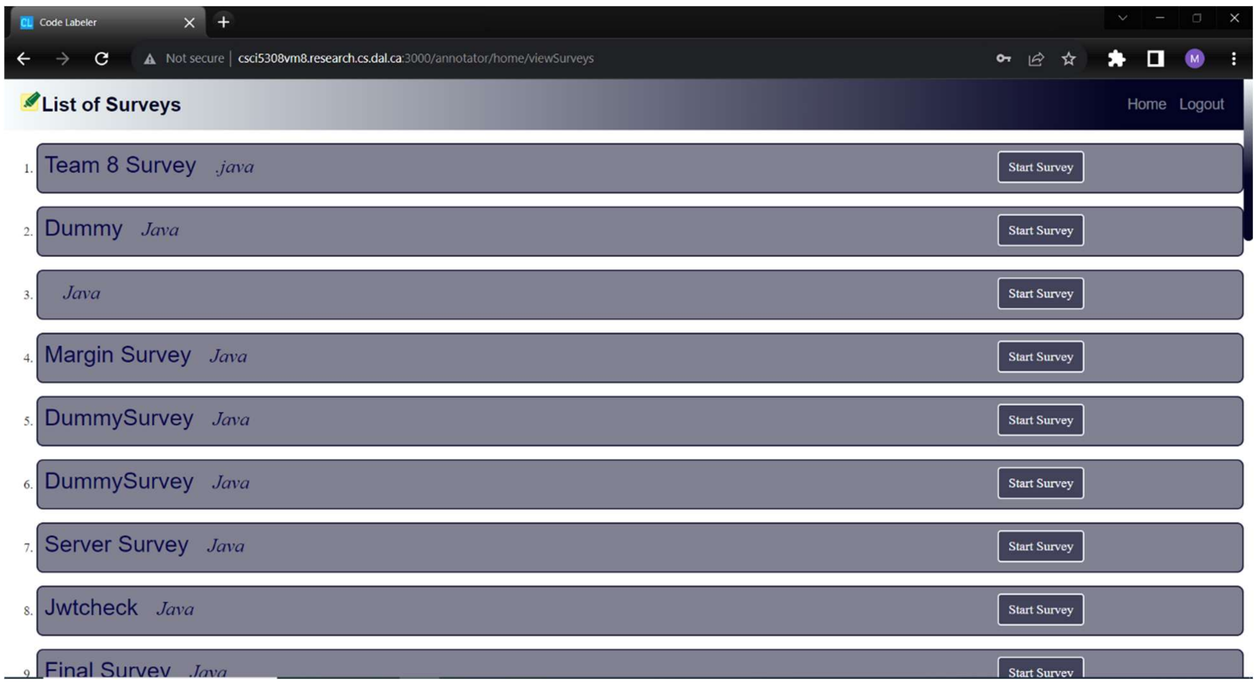

Code Labeler
Not secure | csci5308vm8.research.cs.dal.ca:3000/admin/home/viewAnnotatedSurveys/26
Back
Tags
Highlighted Tags

Annotated Snippets

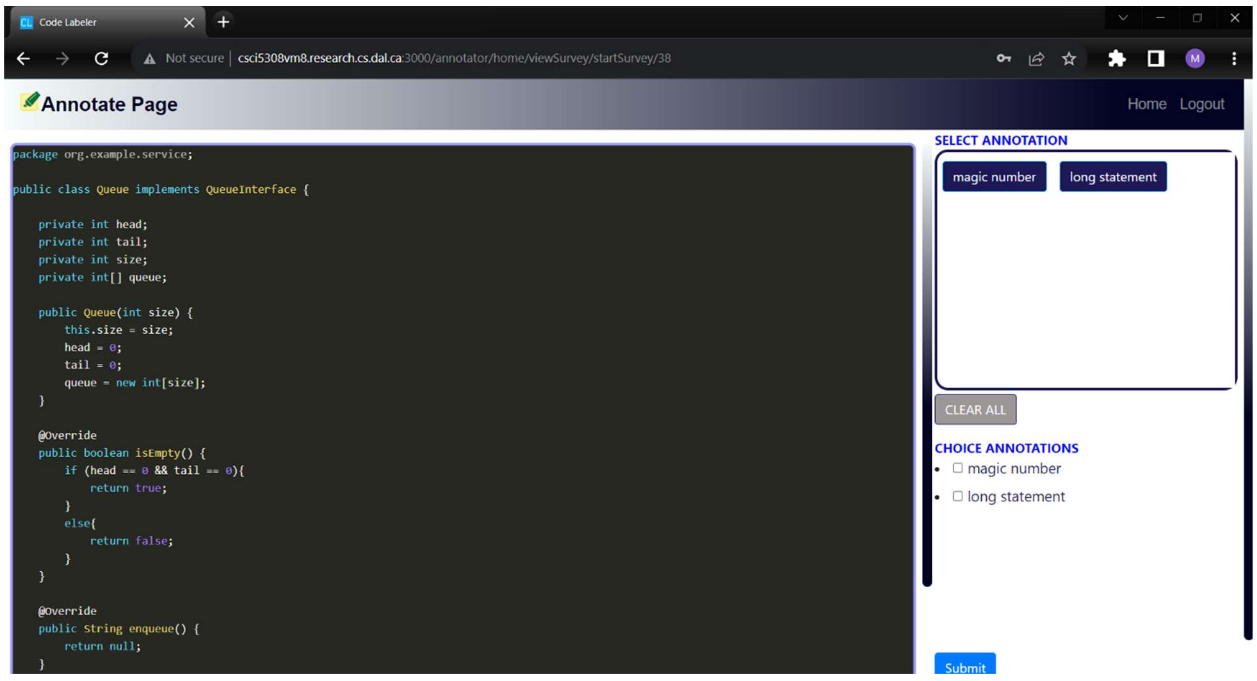
```java
package com.csci5308.codeLabeller.Controller;

import com.csci5308.codeLabeller.Models.DTO.AuthResponse;
import com.csci5308.codeLabeller.Models.DTO.UserLoginDetails;
import com.csci5308.codeLabeller.Service.UserSignUpService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.security.config.annotation.method.configuration.EnableMethodSecurity;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.servlet.view.RedirectView;

import java.io.IOException;

@CrossOrigin
@EnableMethodSecurity
@RestController
public class LoginController {

    @Autowired
    UserSignUpService userSignUpService;
    @PostMapping("/login")
    public ResponseEntity<AuthResponse> login(@RequestBody UserLoginDetails user){
        return ResponseEntity.ok(userSignUpService.authenticate(user));
    }
}
```


Code Labeler
Not secure | csci5308vm8.research.cs.dal.ca:3000/admin/home/viewHighlightedSurveys/26/snippet/36?text=cGFja2FnZSBjb20uY3NjaTUzMDguY29kZ...
BACK
previous 1 2 3 next

```
package com.csci5308.codeLabeller.Controller;

import com.csci5308.codeLabeller.Models.DTO.AuthResponse;
import com.csci5308.codeLabeller.Models.DTO.UserLoginDetails;
import com.csci5308.codeLabeller.Service.UserSignUpService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity; condiitonal
import org.springframework.security.config.annotation.method.configuration.EnableMethodSecurity;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.servlet.view.RedirectView;

import java.io.IOException;

@CrossOrigin
@EnableMethodSecurity
@RestController
public class LoginController {

    @Autowired
    UserSignUpService userSignUpService; recursive
    @PostMapping("/login")
    public ResponseEntity<AuthResponse> login(@RequestBody UserLoginDetails user){
        return ResponseEntity.ok(userSignUpService.authenticate(user));
    }
}
```


Code Labeler
Not secure | csci5308vm8.research.cs.dal.ca:3000/annotator/home/viewSurveys
List of Surveys
Home Logout

1. Team 8 Survey *.java* Start Survey
2. Dummy *Java* Start Survey
3. *Java* Start Survey
4. Margin Survey *Java* Start Survey
5. DummySurvey *Java* Start Survey
6. DummySurvey *Java* Start Survey
7. Server Survey *Java* Start Survey
8. Jwtcheck *Java* Start Survey
9. Final Survey *Java* Start Survey



```
package org.example.service;

public class Queue implements QueueInterface {

    private int head;
    private int tail;
    private int size;
    private int[] queue;

    public Queue(int size) {
        this.size = size;
        head = 0;
        tail = 0;
        queue = new int[size];
    }

    @Override
    public boolean isEmpty() {
        if (head == 0 && tail == 0){
            return true;
        }
        else{
            return false;
        }
    }

    @Override
    public String enqueue() {
        return null;
    }
```

- ☐ magic number
- ☐ long statement