

# Progress Report 2

COSC 4P02 Software Engineering II

Naser Ezzati-Jivan

Brock University

TA: Madeline Janecek

**Thomas Van Veen**

[tv15jl@brocku.ca](mailto:tv15jl@brocku.ca)

#5937123

**Yuvraj Sehgal**

[ys19rk@brocku.ca](mailto:ys19rk@brocku.ca)

#6921795

**Ralph Terte**

[rt19bu@brocku.ca](mailto:rt19bu@brocku.ca)

#6911002

**Antonio Belsito**

[ab19do@brocku.ca](mailto:ab19do@brocku.ca)

#6909543

**Duc Nguyen Minh**

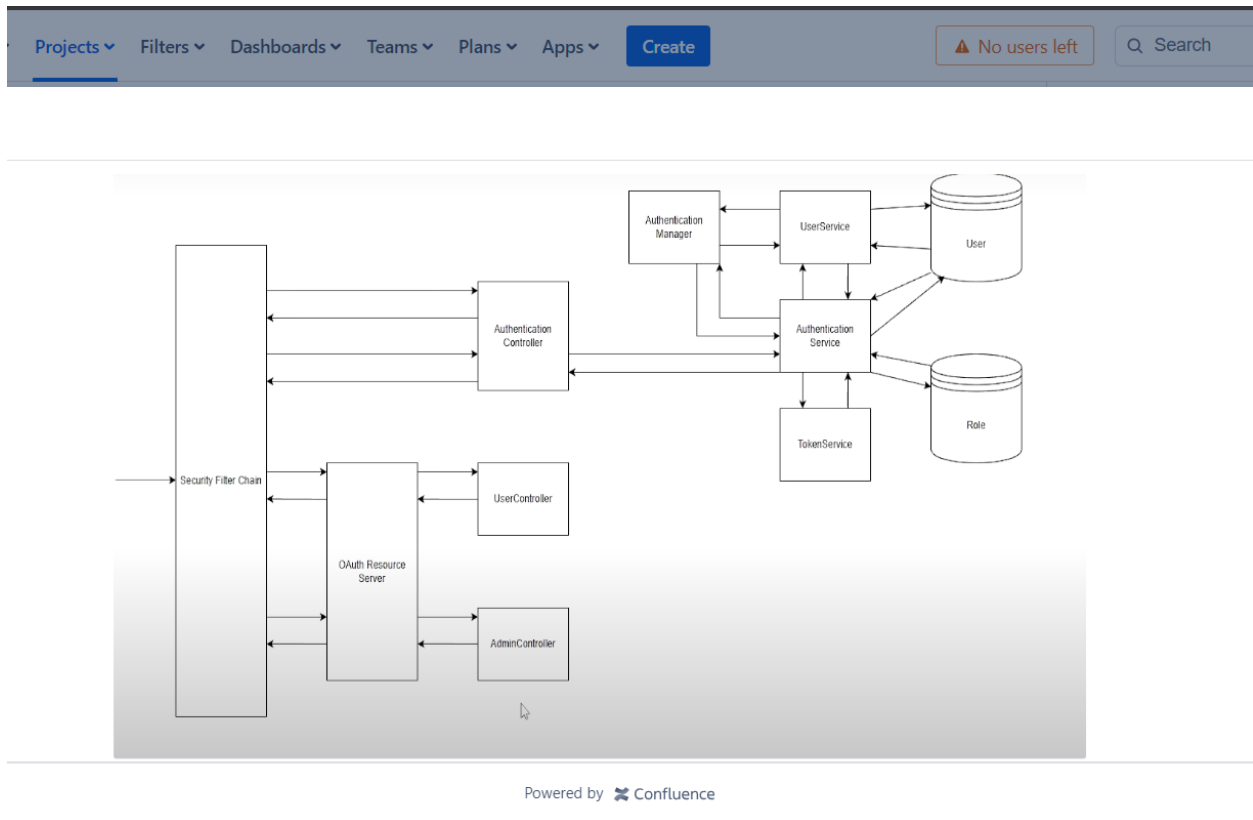
[dn19zf@brocku.ca](mailto:dn19zf@brocku.ca)

#6877765

So, the progress we made after the last progress report is as follows:

## User login and spring security

Firstly, we implemented spring security in the user management system, so now user can login/signup into the application using the spring JWT token security, which is one of the most secured ways to login into the application. Here is the screenshot of our security mechanism:



Any request made by the user will hit our Security filter chain end point, which ensures if the user has the required permission to access that end point or resource, if not permitted either user is denied access with HTTP 401, unauthorized or if they are not logged in yet, they are prompted with signin/signup page.

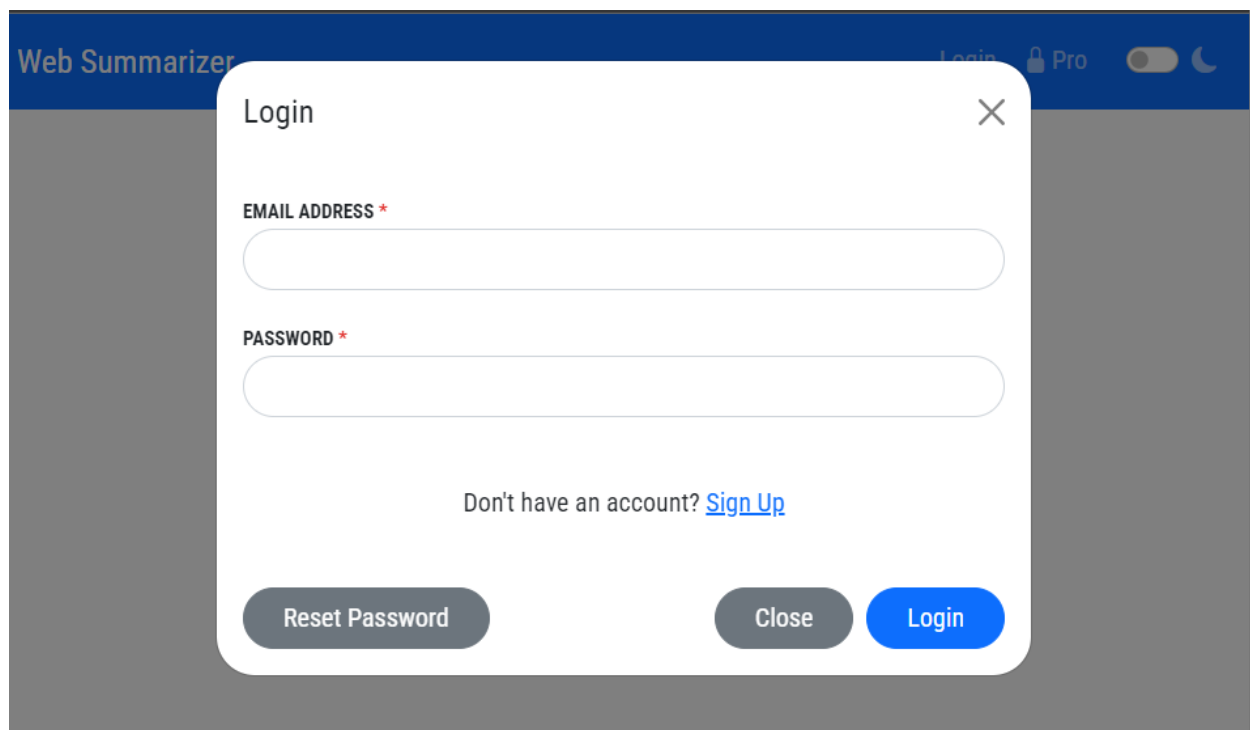
To avoid the unauthorized access to the database, we have chosen to use Authentication Service, which is responsible to communicate with the database to access the required resources. Having all the communications done through the authentication controller, it is impossible for the user to access the DB or user access of other users.

Once a request is submitted by the user, if the credentials are valid, we return a valid JWT encrypted token to the user which is valid for a session, if the credentials are invalid, then user is prompted to fill valid information.

The OAuth Resource server is responsible to access the required webpage for the user according to the validity of the request. (The administrator part of the application might not be required for this application, so we might remove it later).

Here is the sample login and registration feature:

Once the user clicks on the login, they will be prompted with the login modal:



The screenshot displays a web application interface with a dark blue header. The header contains the text 'Web Summarizer' on the left, and 'Login', a lock icon, 'Pro', a toggle switch, and a moon icon on the right. A white login modal is centered on the screen. The modal has a title 'Login' and a close button (X) in the top right corner. It features two input fields: 'EMAIL ADDRESS \*' and 'PASSWORD \*'. Below the fields is a link: 'Don't have an account? [Sign Up](#)'. At the bottom of the modal are three buttons: 'Reset Password', 'Close', and 'Login'.

Since we do not have a user account, we would like to signup as follows:

The image shows a 'Web Summarizer' application with a 'Create an Account' modal form. The modal is white with rounded corners and a close button (X) in the top right. It contains five input fields: 'FIRST NAME \*', 'LAST NAME \*', 'EMAIL ADDRESS \*', 'PASSWORD \*', and 'PHONE NUMBER'. Below the fields is a link 'Already have an account? [Login](#)'. At the bottom right of the modal are two buttons: 'Close' (grey) and 'Create' (blue). The background of the application is dark blue with a header containing 'Web Summarizer', 'Login', 'Pro', and a toggle switch. At the bottom of the application is a grey input field with the placeholder 'Enter text or a URL' and a blue 'Summarize' button.

Web Summarizer

Login Pro

Create an Account

FIRST NAME \*

LAST NAME \*

EMAIL ADDRESS \*

PASSWORD \*

PHONE NUMBER

Already have an account? [Login](#)

Close Create

Enter text or a URL

Summarize

Fill up the required information, and the front-end ensures that any invalid input is submitted (like a blank mandatory field)

The image shows a 'Create an Account' modal form overlaid on a web application. The modal has a title 'Create an Account' and a close button (X) in the top right corner. It contains five input fields, each with a label and a red asterisk indicating it is mandatory:

- FIRST NAME \***: Input field containing 'Yuvraj'.
- LAST NAME \***: Input field containing 'Sehgal'.
- EMAIL ADDRESS \***: Input field containing '17yuvraj.sehgal@gmail.com'.
- PASSWORD \***: Input field containing '.....' with a green checkmark on the right, indicating a valid password.
- PHONE NUMBER**: Input field containing '1111111111' with a green checkmark on the right, indicating a valid phone number.

Below the input fields, there is a link: 'Already have an account? [Login](#)'. At the bottom right of the modal, there are two buttons: 'Close' (grey) and 'Create' (blue).

The background of the application shows a dark blue header with the text 'Web Summarizer' on the left and 'Login Pro' on the right. At the bottom, there is a grey input field with the placeholder text 'Enter text or a URL' and a blue 'Summarize' button on the right.

Once valid user details are submitted, user will see a user creation successful message

The image shows a web application interface with a dark blue header. On the left, the text "Web Summarizer" is visible. On the right, there are links for "Login" and "Pro", a toggle switch, and a moon icon. Below the header, a modal window titled "Login" is centered. The modal has a close button (X) in the top right corner. Inside the modal, a green success message reads: "User '17yuvraj.sehgal@gmail.com' created successfully. Please login." Below this message are two input fields: "EMAIL ADDRESS \*" and "PASSWORD \*". The email field contains a single vertical bar character and has a red exclamation mark icon on its right. Below the password field is a link that says "Don't have an account? [Sign Up](#)". At the bottom of the modal are three buttons: "Reset Password", "Close", and "Login". The "Login" button is blue, while the others are grey. In the background, a list of items is partially visible, including "ry item 1" through "ry item 10".

Web Summarizer

Login Pro

Login

✓ User '17yuvraj.sehgal@gmail.com' created successfully. Please login.

EMAIL ADDRESS \*

PASSWORD \*

Don't have an account? [Sign Up](#)

Reset Password Close Login

ry item 1  
ry item 2  
ry item 3  
ry item 4  
ry item 5  
ry item 6  
ry item 7  
ry item 8  
ry item 9  
ry item 10

Now if we try to register the user using same email, it won't be allowed, since every user must have a unique email address.

The image shows a 'Create an Account' modal form. At the top, there is a title 'Create an Account' and a close button (X). Below the title, a red error message is displayed: '⚠ Registration error for '17yuvraj.sehgal@gmail.com'. Please try again.' The form contains five input fields: 'FIRST NAME \*' with the value 'Yuvraj', 'LAST NAME \*' with the value 'Sehgal', 'EMAIL ADDRESS \*' with the value '17yuvraj.sehgal@gmail.com', 'PASSWORD \*' which is empty, and 'PHONE NUMBER' with the value '4373355700'. At the bottom of the form, there is a link 'Already have an account? [Login](#)'. Two buttons are located at the bottom right: a 'Close' button and a 'Create' button.

Create an Account

⚠ Registration error for '17yuvraj.sehgal@gmail.com'. Please try again.

FIRST NAME \*

Yuvraj

LAST NAME \*

Sehgal

EMAIL ADDRESS \*

17yuvraj.sehgal@gmail.com

PASSWORD \*

PHONE NUMBER

4373355700

Already have an account? [Login](#)

Close Create

Now since we have '17yuvraj.sehgal@gmail.com' as a registered user we can login using these credentials (currently our front-end for login is under progress, so we are using the back end for demonstration using postman)

If we try to access the /user endpoint we will get a 401 unauthorized since this endpoint is protected. And the token we provided is invalid.

The screenshot shows the Postman interface for a GET request to `http://localhost:8080/user/`. The 'Auth' tab is selected, showing 'Bearer Token' as the type. A token is entered in the 'Token' field: `eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOi...`. The 'Body' tab is selected, showing a '401 Unauthorized' response. A tooltip explains that 401 Unauthorized is similar to 403 Forbidden but specifically for use when authentication is possible but has failed or not yet been provided. The response must include a `WWW-Authenticate` header field containing a challenge applicable to the requested resource.

HTTP `http://localhost:8080/user/` Save

GET `http://localhost:8080/user/` Send

Params Auth Headers (7) Body Pre-req. Tests Settings Cookies

Type: Bearer Token

Token: `eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOi...`

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Body 401 Unauthorized 68 ms 615 B Save Response

1

**401 Unauthorized**

Similar to 403 Forbidden, but specifically for use when authentication is possible but has failed or not yet been provided. The response must include a `WWW-Authenticate` header field containing a challenge applicable to the requested resource.



Now if we give this valid token to the login, we will have the required user level access.

The screenshot displays a REST client interface with the following components:

- URL Bar:** Shows the URL `http://localhost:8080/user/` and a `GET` method.
- Authorization:** Set to `Bearer ...` with a token `eyJhbGciOiJSUzI1NiJ9.eyJpc3MiOiJzZWxm...`.
- Response:** A `200 OK` status with `18 ms` and `440 B`. The response body is `1 User level access`.
- Tooltip:** A tooltip for the `200 OK` status explains that it is a standard response for successful HTTP requests, where the actual response depends on the request method (e.g., GET returns an entity, POST returns a result).

Now we try to login using valid credentials, it will return a valid JWT token which can be used for authentication.

HTTP <http://localhost:8080/auth/login> Save

**POST** <http://localhost:8080/auth/login> Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary

	Key	Value	...	Bulk Edit
<input checked="" type="checkbox"/>	login_email	17yuvraj.sehgal@gmail.com		
<input checked="" type="checkbox"/>	login_password	password		

Body Cookies Headers (14) Test Results 200 OK 202 ms 1.25 KB Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "user": {
3     "id": 2,
4     "first_name": "Yuvraj",
5     "last_name": "Sehgal",
6     "email": "17yuvraj.sehgal@gmail.com",
7     "password": "$2a$10$8W4iJSXr5.ThH2QJzLJ2a.l0mcwyfyCLNpVxWTQg94jzzYYUNF046",
8     "phone_number": "4373355700",
9     "authorities": [
10      {
11        "roleId": 2,
12        "authority": "USER"
13      }
14    ],
15     "enabled": true,
16     "username": "17yuvraj.sehgal@gmail.com",
17     "accountNonLocked": true,
18     "accountNonExpired": true,
19     "credentialsNonExpired": true
20   },
21   "jwt": "eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJzZWxmIiwic3ViIjoiaWMTd5dXZyYWouc2VoZ2FsQGdtYWlsLmNvbSIsInJvbGUiOiJlVU0VSIiwiaWF0IjoxNzExOTg1NjUyYUfQ.NEYNLq0lg4gPXoNC5mt26t4zTpQZMN-p6lQm4Emzc7osH96ZYfGhuWq20B-mP2wScB9s2ZhpFA04wGTf6799tt6qf5CP2i0Ra00jvrTCIAPlmGyjQgiake940aZWYLi36lUonQ2CJB21Pv2JS67_FT4oDQuExp9k0Y9VA_k6AZT__xX07r3vqrI1_oKaam_99SvfpXUFJ0GkXmGKfU12MwAFsvNdeYuX7yKYGGp4o8mqcycIclz_NGCnDR65T6Wcb80UHJ4yMnwSsQK32IoJR7_c5yxYhez7WlsudAeR48yE3IJu6G7massDPWvCbYYPlubf1UoIBMMULRtKKSNUQ"
22 }
```

Now since we have valid JWT, we can have User level access, which will include different features like user history, password resets etc. (some of these features are under progress).

The screenshot shows a web browser's developer tools interface. At the top, the address bar shows the URL `http://localhost:8080/user/`. Below it, the 'Send' button is visible. The 'Authorization' tab is selected, showing the 'Type' as 'Bearer...' and the 'Token' as a long JWT string. The 'Body' tab is also visible, showing the response 'User level access'.

HTTP `http://localhost:8080/user/` Save

GET `http://localhost:8080/user/` Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Type Bearer... Token

The authorization header will be automatically generated when you send the request.  
[Learn more about authorization](#)

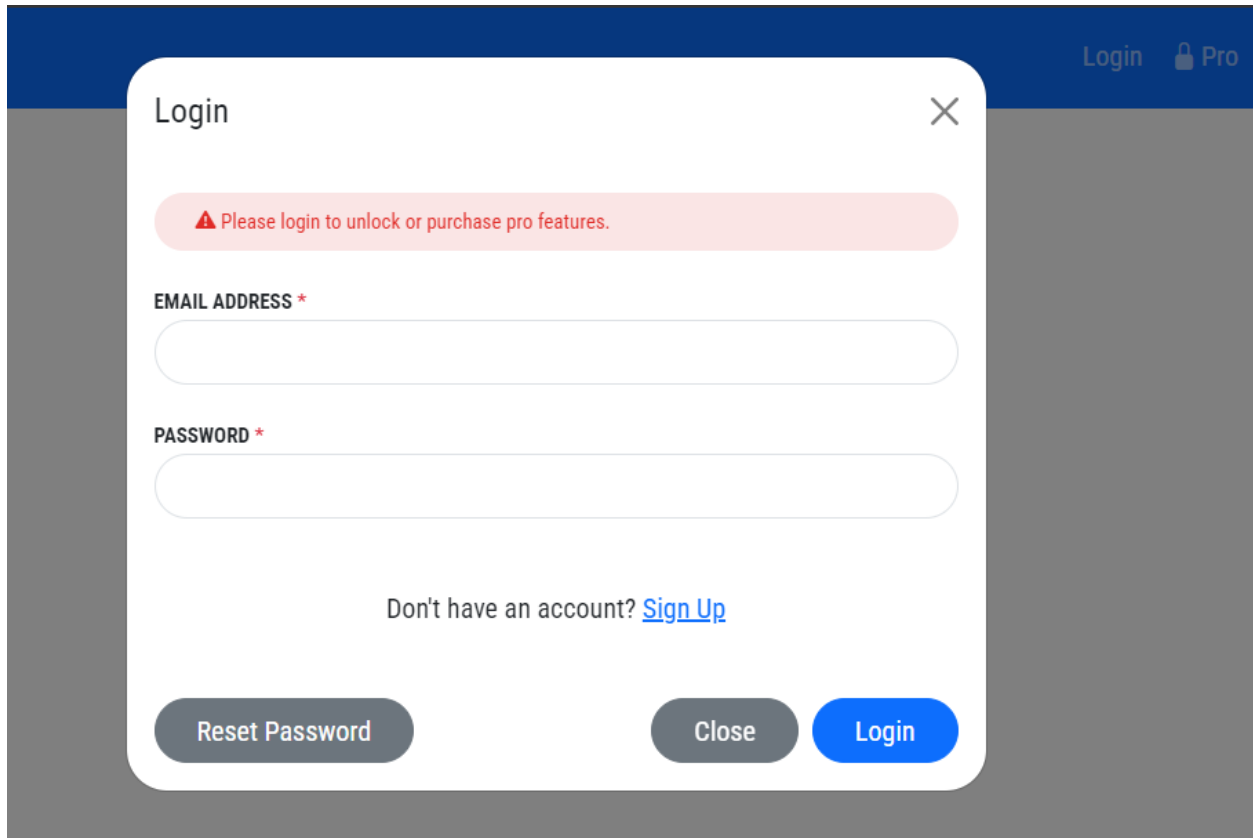
eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJzZWxmliiwic3ViljoiaMTd5dXZyYWouc2VoZ2FsQGdtYWlsLmNvbSIsInJvbmVudGUiOiJVVU0VSliwiaWF0IjoxNzExOTg1NzgxfQ.FdLVvM9sUtO41gOR0HDF6BspDGe6FZIK-mcaf4BU2EAUQOw\_I7MAAcG1w9EU\_D3PlznRXpjEblrdHUo3lu83H-Kb6BSwd2WQaA0bwn8-XZVnBJkVDMs\_O-3aADruEkWgV2w0sADhprmlY-rt1-OsrB6G83B7WnV\_RgCTlp2JolbshZgEVPS3z2PQKgawgzgO-kVMzwUgtFwwIGZLPohwdJVMuY\_b5KeKVbLF5083jNcPK9B\_bWE7fWZ3\_aOfx5BluUDK1xZQw2ldV2ZCfMOEwsY7FV3cCcFuP1gacQCLINhIQEPThed3NDIAaum

Body Cookies Headers (14) Test Results 200 OK 25 ms 440 B Save Response

Pretty Raw Preview Visualize Text

1 User level access

In order to use pro features of the application, the user must login, they are prompted to login if they haven't done so.



The image shows a login modal dialog box with a dark blue header and a grey background. The modal has a title 'Login' and a close button (X) in the top right corner. A red warning message is displayed: '⚠ Please login to unlock or purchase pro features.' Below this, there are two input fields: 'EMAIL ADDRESS \*' and 'PASSWORD \*'. At the bottom, there is a link 'Don't have an account? [Sign Up](#)' and three buttons: 'Reset Password', 'Close', and 'Login'.

Login

⚠ Please login to unlock or purchase pro features.

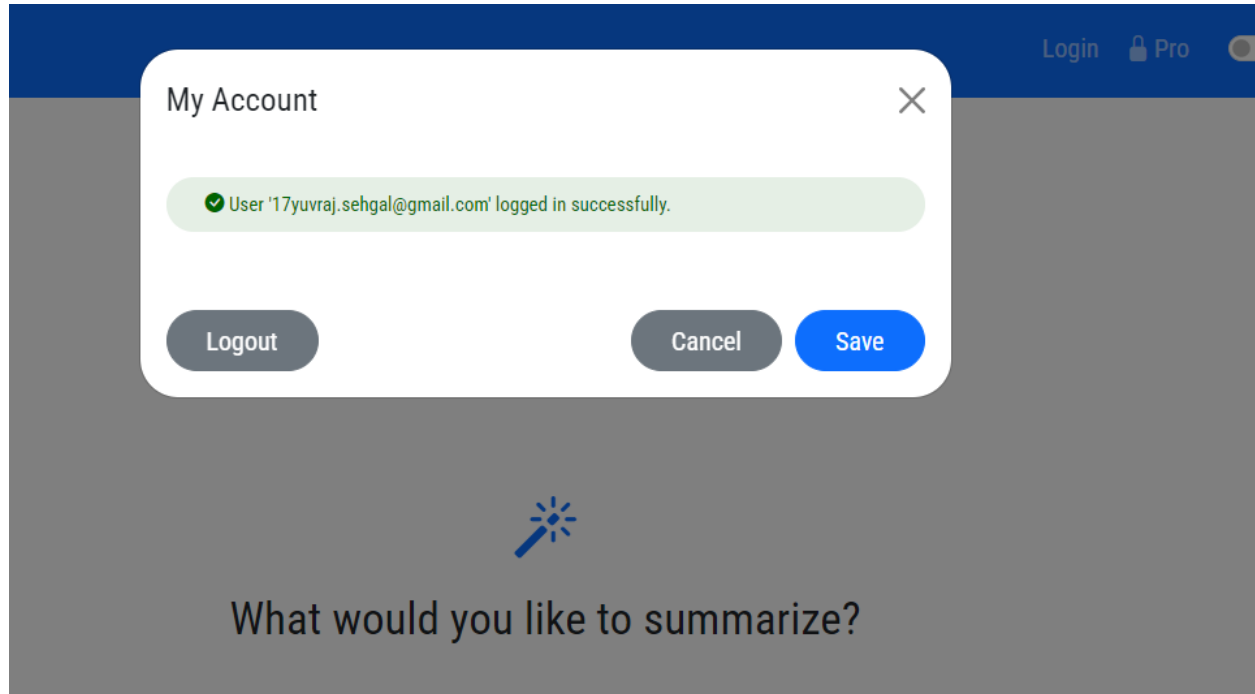
EMAIL ADDRESS \*

PASSWORD \*

Don't have an account? [Sign Up](#)

Reset Password Close Login

Front-end (in-progress)



## Premium features payment

Once user is logged in, they can buy premium features as follows:

Pro Features

SUMMARIZE TEXT OR URL

Use NLP or LLMs to summarize the content of text or a website link

Free

SHORT LINK CREATION

Create a short link to a summary

Free

CUSTOMIZATION

Manage short links, and modify account settings

Free

SUMMARY HISTORY

Save and access your summaries in your account

\$4.99 / mo

SOCIAL MEDIA

Integration with social media platforms for authentication and sharing

\$4.99 / mo

API ACCESS

Get access to use our backend in your own projects

\$4.99 / mo

Payment Details

CARD HOLDER NAME \*

Yuvraj Sehgal

✓

CARD NUMBER \*

1111111

✓

EXP \*

06/27

✓

CVV \*

111

✓

GUARANTEED SAFE CHECKOUT

VISA

MasterCard

Amex

DISCOVER

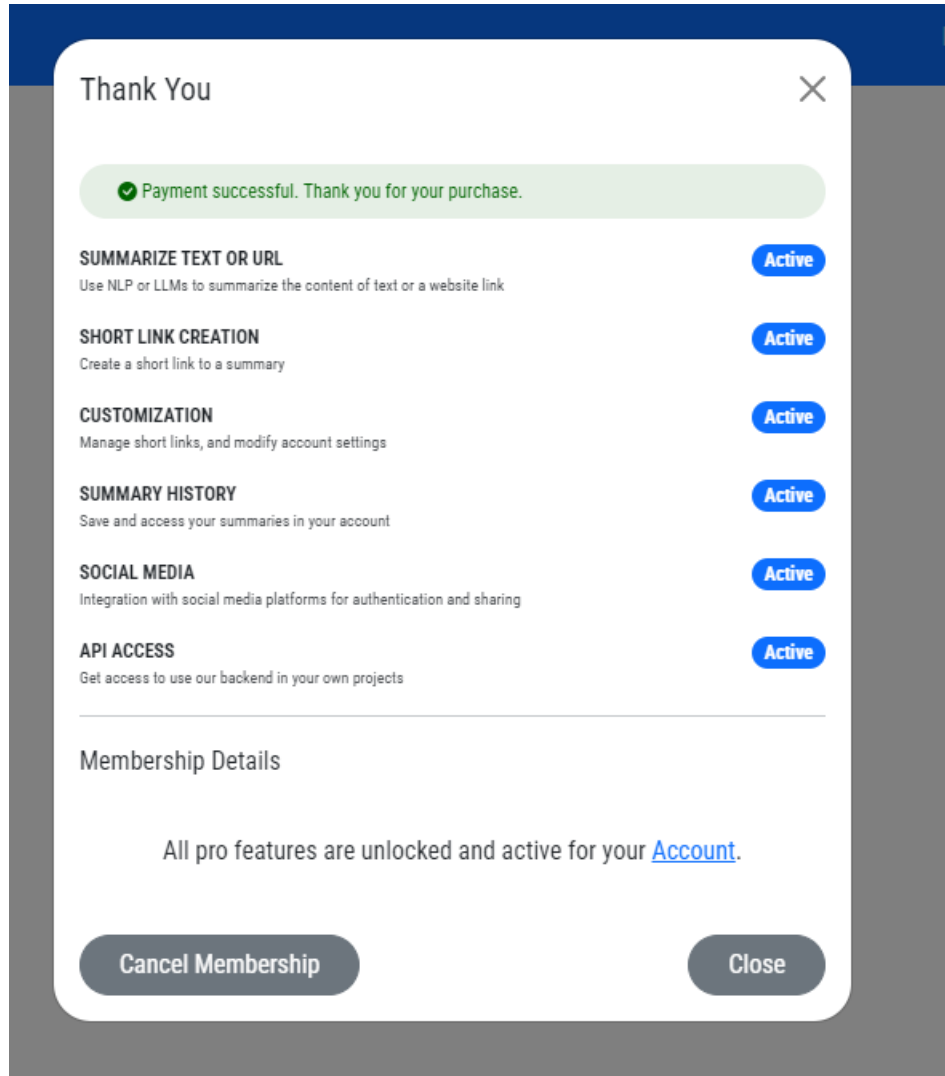
Bank of America

PayPal

Close

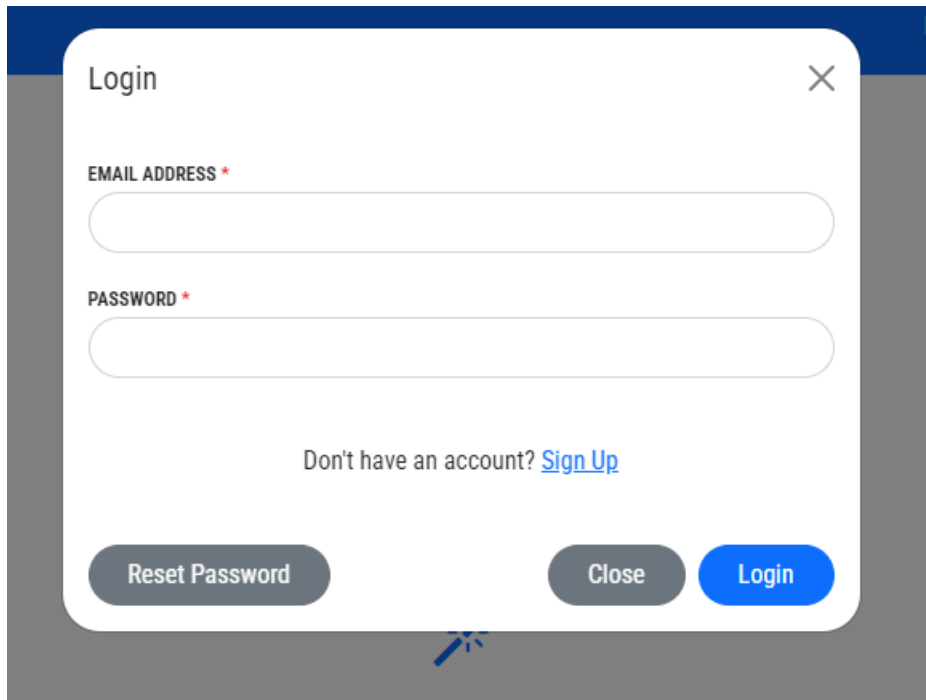
Purchase

Once the payment is made the user can access all the premium features



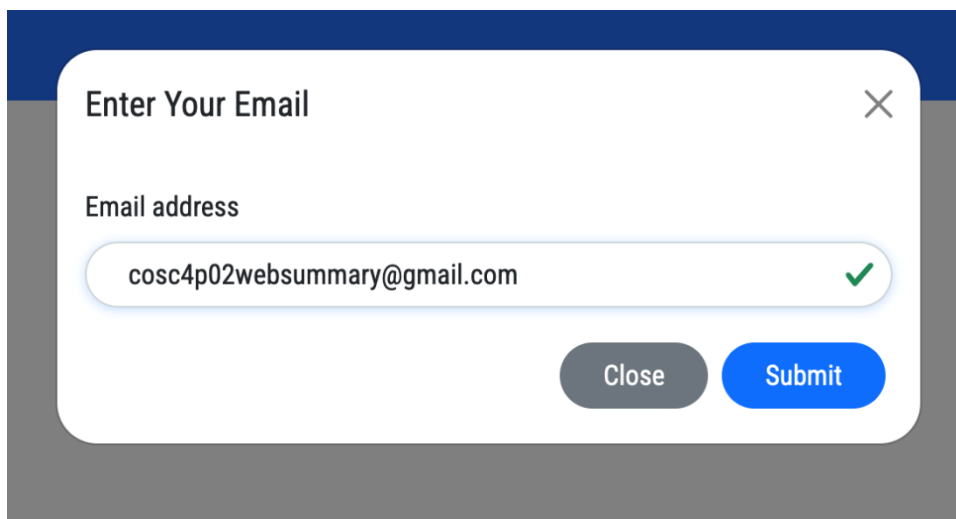
## Password reset feature.

Click on the Reset Password button.



A login modal window titled "Login" with a close button (X) in the top right corner. It contains two input fields: "EMAIL ADDRESS \*" and "PASSWORD \*". Below the password field is a link "Don't have an account? [Sign Up](#)". At the bottom, there are three buttons: "Reset Password" (grey), "Close" (grey), and "Login" (blue).

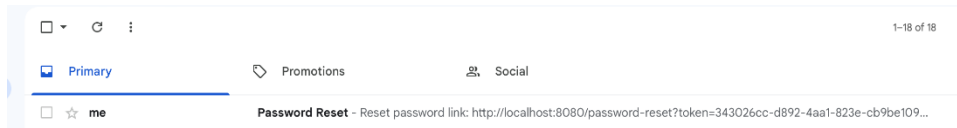
Enter email of existing user



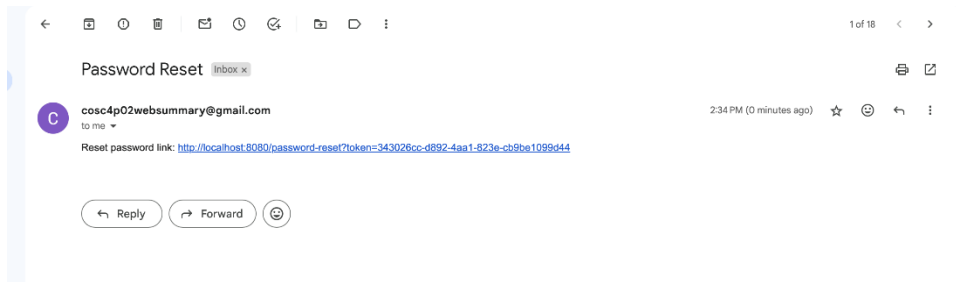
An "Enter Your Email" modal window with a close button (X) in the top right corner. It contains a label "Email address" and an input field with the text "cosc4p02websummary@gmail.com". A green checkmark is visible to the right of the input field. At the bottom, there are two buttons: "Close" (grey) and "Submit" (blue).



After submitting the request user will receive an email



The email will contain the link for the password reset.



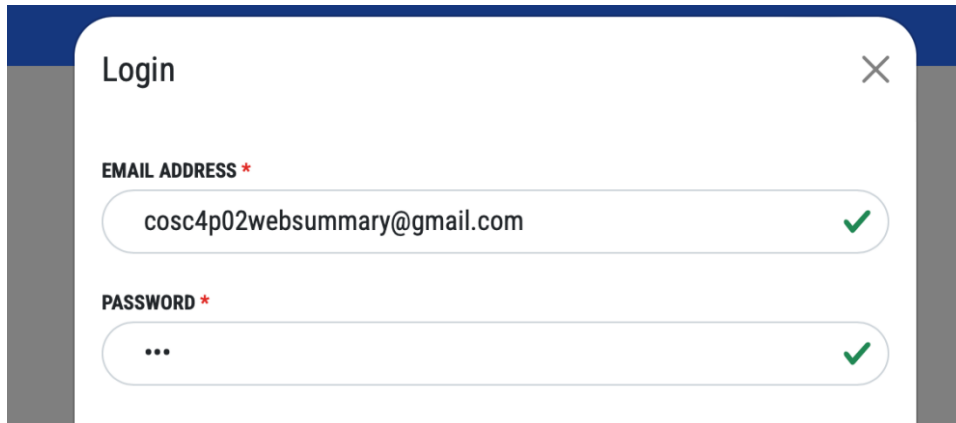
Once the user clicks on the link, they will be redirected to the enter new password prompt as follows

### Enter new password

...

Submit

After submitting user has successfully changed the password



The image shows a 'Login' dialog box with a close button (X) in the top right corner. It contains two input fields: 'EMAIL ADDRESS \*' and 'PASSWORD \*'. The email field contains 'cosc4p02websummary@gmail.com' and has a green checkmark to its right. The password field contains three dots and also has a green checkmark to its right, indicating a successful login.

Proof of new sign-in

```
where
u1_0.email=?
2024-04-01T14:37:52.642-04:00 INFO 18291 --- [nio-8080-exec-2] c.w.W.S.services.AuthenticationService : returning the following login response dto: LoginResponseDTO
(user=User(id=152, first_name=John, last_name=Smith, email=cosc4p02websummary@gmail.com, password=$2a$10$uSpKHLmD1wG3ERuP48toT0HArlWKof3zDXRnmTp0RSYv0ZW6GHRdK,
phone_number=, request_token=null, authorities=[Role(roleId=2, authority=USER)]), jwt=eyJhbGciOiJSUzI1NiJ9
.eyJpc3MiOiJzZWxmiwiic3ViIjo1Y29zYzRwMDJ3ZWJzdW1tYXJ5Q0GdtYVlsLmNvbSI6InJvbGUiOiJVV0VSIiwiaWF0IjoxNzExOTk2NjcyfQ
.py46CSHYqphqT_Uk_HSYN1601yxcoKAQJhJkXC2NKY_MqxxRx7xk9YyShGfVzAEh6Do5LnmoJ_XDlqqRvVpwqkHWAfgyccQa8R8EHdNIldctqBAzA6lbnvG7WyrEnOX8ZkIPdm_HcKcWwWT8oSQ1CsenphFXNPWMJuM57XitRI
pHXvQgf_WUCG8ayGmbL0n2PH_8Chsqbc9AgaSxDf_zBgUB3PeazHymWMQuj3a9YBav2x2UBpLkTGmM0PPoCstuIxGdebeV4GuFukdUdWDTzf9pD4yx0djLcTAuq9mbKbZQ1_TWgPwa3hLbSzpJopN6Q01R19PJA9qm3LYivg)
```

## Short Link URL for given summary feature.

Once the user log in, they get the feature of sharing their summary using a shortened URL.

For example, if we had 3 summarizes under our account:

	hid [PK] bigint	history_content character varying (100000)
1	1	Sotu: Biden's response to immigration was "unacceptable" Biden: "I don't think it was the right thing to do. It was not the right response to the immigration issue" Click here for more from CNN.com: <a href="http://www.cnn.com/2014/03/10/politics/biden-response-immigration-biden.html">http://www.cnn.com/2014/03/10/politics/biden-response-immigration-biden.html</a>
2	2	Sotu: Biden's response to immigration was "unacceptable" Biden: "I don't think it was the right thing to do. It was not the right response to the immigration issue" Click here for more from CNN.com: <a href="http://www.cnn.com/2014/03/10/politics/biden-response-immigration-biden.html">http://www.cnn.com/2014/03/10/politics/biden-response-immigration-biden.html</a>
3	3	The Academy Awards were held on March 10. The winners were announced at a ceremony in Los Angeles. The ceremony was attended by more than 100,000 people. The awards were presented by the Academy of Motion F

A short link will be associated with them as follows:

	short_link character varying (255)	upload_time timestamp without time zone (6)	uid bigint
on-biden.html	ABCDEF	2024-03-10 00:00:00	1
on-biden.html	MP16UG	2024-03-10 23:43:01.28967	1
id Foreign Press Association. For more, go to CNN.com...	W35YG0	2024-03-11 00:24:03.036088	1

(note from the developer):

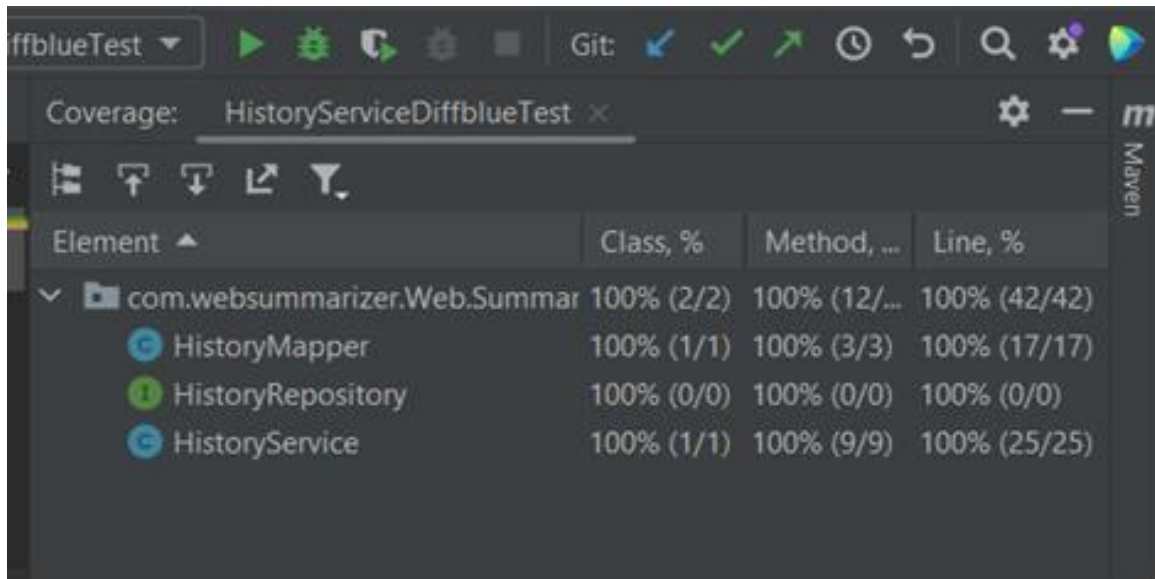
We still need a front end for this to run properly. This version of the short link is compatible with the main branch as of 03-10-2024. It requires data in the user and history tables to run. The generated short link will have six characters at the end, which can be copied and pasted into <http://localhost:8080/> to access the summarization. For example, if the short link is <http://localhost:8080/MP16UG>, MP16UG is the six characters at the end of the summarization.

```
{"history_content": "Sotu: Biden's response to immigration was\n\"unacceptable\" Biden: \"I don't think it was the right thing to do. It was\nnot the right response to the immigration issue\" Click here for more from\nCNN.com: http://www.cnn.com/2014/03/10/politics/biden-response-immigration-biden.html", "short_link": "MP16UG", "upload_time": "2024-03-10T23:43:01.28967", "uid": 1, "hid": 2}
```

Unfortunately, The API we are using in the backend for LLM is currently down, so we cannot provide the most up to date screenshots of the feature.

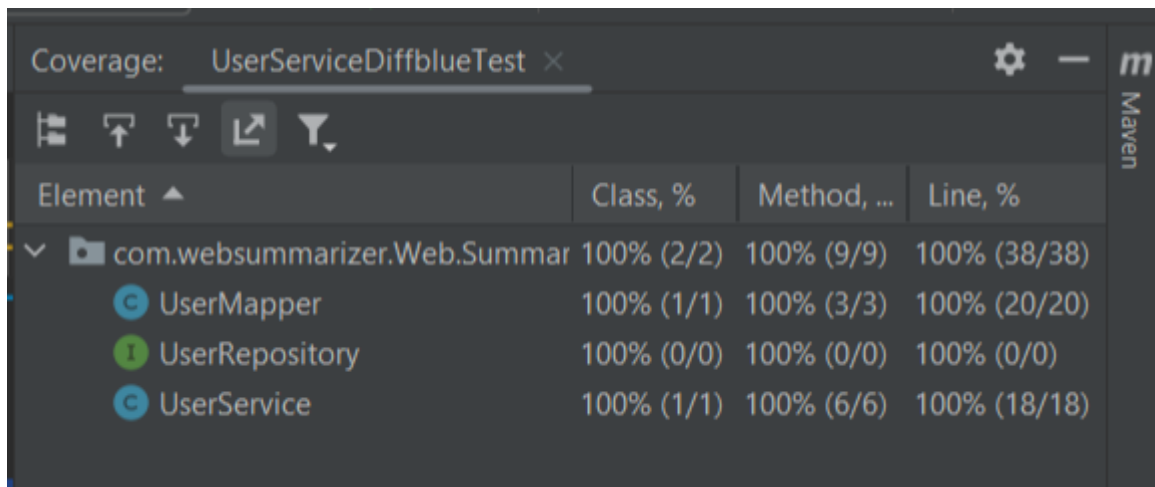
## Testing progress

The unit test coverage of our application. Some of the classes were just introduced to the application so we have less test coverage for those classes, but we are working on that currently, so it will be updated soon.



Coverage: HistoryServiceDiffblueTest ×

Element ▲	Class, %	Method, ...	Line, %
com.websummarizer.Web.Summarizer	100% (2/2)	100% (12/...)	100% (42/42)
HistoryMapper	100% (1/1)	100% (3/3)	100% (17/17)
HistoryRepository	100% (0/0)	100% (0/0)	100% (0/0)
HistoryService	100% (1/1)	100% (9/9)	100% (25/25)



Coverage: UserServiceDiffblueTest ×

Element ▲	Class, %	Method, ...	Line, %
com.websummarizer.Web.Summarizer	100% (2/2)	100% (9/9)	100% (38/38)
UserMapper	100% (1/1)	100% (3/3)	100% (20/20)
UserRepository	100% (0/0)	100% (0/0)	100% (0/0)
UserService	100% (1/1)	100% (6/6)	100% (18/18)

Maven			
Element ▲	Class, %	Method, ...	Line, %
▼ com.websummarizer.Web.Summar	46% (7/15)	44% (27/6...	32% (43/131)
> history	0% (0/6)	0% (0/20)	0% (0/22)
▼ user	100% (6/6)	95% (21/2...	91% (22/24)
UserReqAto	100% (2/2)	100% (8/8)	100% (8/8)
UserResAto	100% (2/2)	100% (9/9)	100% (9/9)
UsersResAto	100% (2/2)	80% (4/5)	71% (5/7)
HistoryController	0% (0/1)	0% (0/6)	0% (0/21)
UserController	100% (1/1)	100% (6/6)	100% (21/21)
WebController	0% (0/1)	0% (0/7)	0% (0/43)

33% classes, 20% lines covered in package 'com.websummarizer.Web.Summa				Maven	
Element	Class, %	Method, %	Line, %		
history	66% (4/6)	60% (12/20)	59% (13/22)		
user	0% (0/6)	0% (0/22)	0% (0/24)		
HistoryController	100% (1/1)	66% (4/6)	63% (14/22)		
UserController	0% (0/1)	0% (0/6)	0% (0/21)		
WebController	0% (0/1)	0% (0/7)	0% (0/44)		

# Sprint retrospective 2:

## RETROSPECTIVE MEETING

### SPRINT RETROSPECTIVE 2

Created by: yuvraj sehgal

Participants: yuvraj sehgal, Tom Van Veen, Ralph Terte, Ralph Terte, Duc Minh Nguyen, Antonio Belsito

Created at: Friday, March 15, 2024 4:58 PM

#### OVERVIEW

14  
REFLECTIONS

4  
GROUPS

6  
PARTICIPANTS

0  
NEW ISSUES

#### GROUPS

##### ADD

0 

- We can try to add screenshots of our working output in jira ticket so that others know what to expect from the output.
- Comment on pull request if the code is not properly working and how to replicate the bug/ problem.
- We should try to keep more notes on our meeting discussions (when necessary)
- Add some documentation in confluence for later use.
- Add instruction on how to test the new features.
- Work on documenting our code more

##### DROP

0 

- Update jira tickets as you go, to avoid any miscommunication

##### KEEP

0 

- Continue the consistent and friendly communication between the team
- Help others with problems in their code during a team meeting.
- Add comment for better code documentation.
- Front-end looks great and very up to date
- Good teamwork and work pace

## IMPROVE ★

0 👍

- Commit on smaller code rather than all at one in a single commit.
- We could improve our communication in some areas

## ISSUES

## ACTION LIST

The sprint retrospective for our sprint 3 is due this week, since there was a long weekend, so we delayed our meeting.

Sprint backlog:

The screenshot displays the Jira Software interface for a project named 'COSC 4P02 WEB SUMMARIZER'. The left sidebar shows navigation options: PLANNING (Timeline, Backlog, Board, Goals, Issues, Add view), DEVELOPMENT (Code, Security, Releases), and OPERATIONS (Deployments, Project pages, AgileBox, Add shortcut). The main area is titled 'Backlog' and shows a list of issues for 'Sprint 4' (25 Mar - 15 Apr, 7 issues). The issues are:

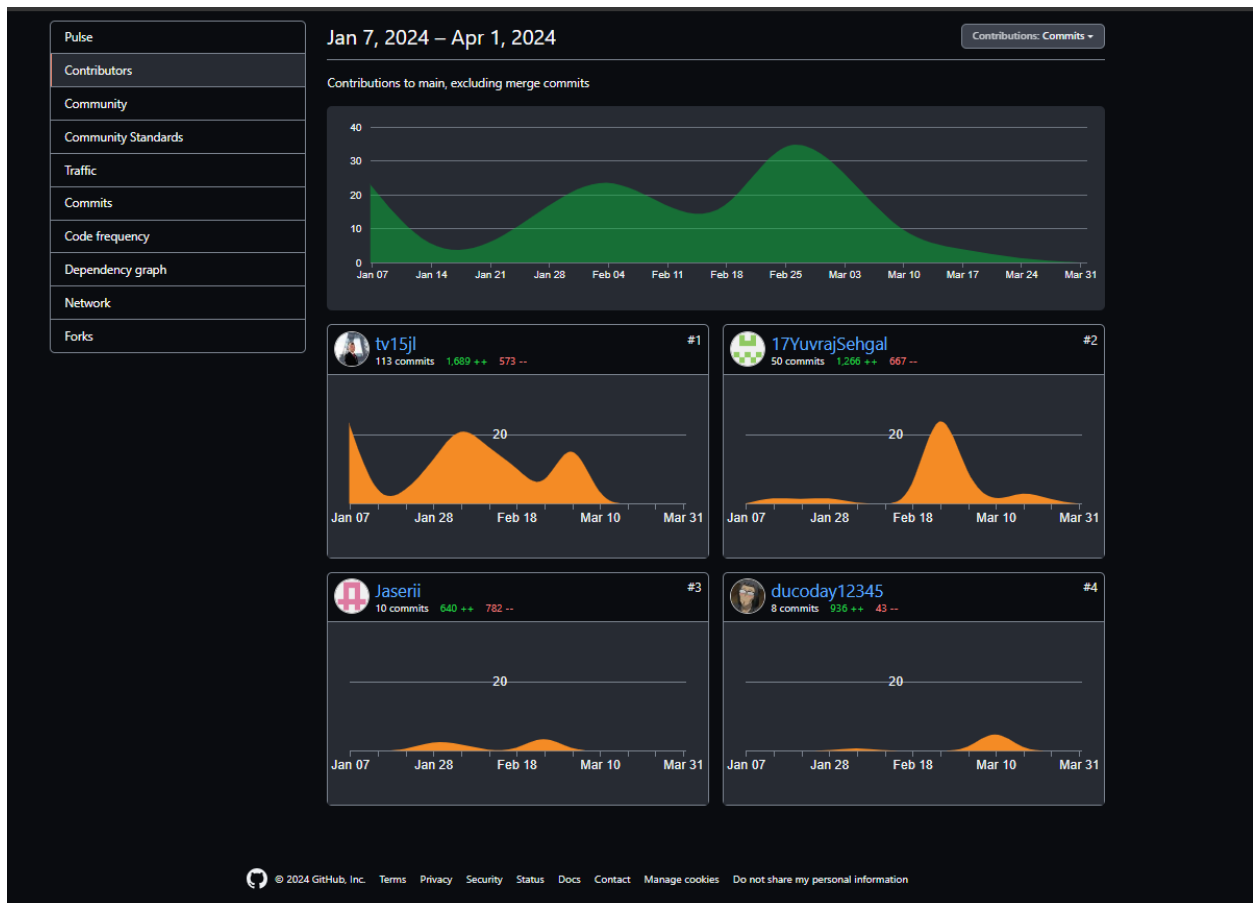
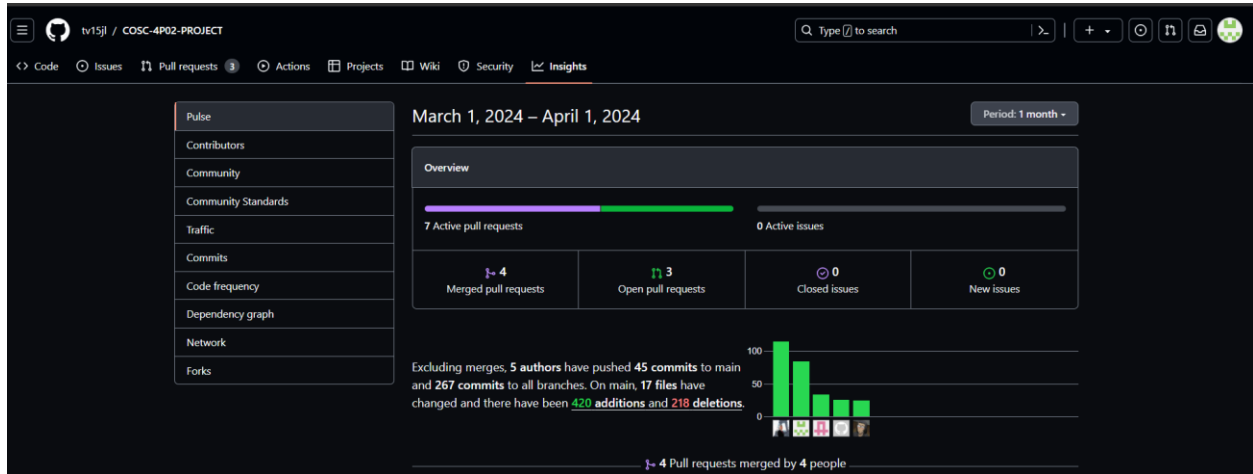
- C4WS-75 Unit Testing (IN PROGRESS)
- C4WS-43 Custom URL link generator for Pro users (IN PROGRESS)
- C4WS-2 Add URL shortener to the application (IN PROGRESS)
- C4WS-78 Create and store isLoggedIn and isProUser flags under the session (IN PROGRESS)
- C4WS-79 Store history in database once user is logged in and show at the front-end (IN PROGRESS)
- C4WS-42 Show history of past 20 summaries of Pro users (IN PROGRESS)
- C4WS-41 Admin logins for user management (TO DO)

Below the sprint backlog, there is a section for the general backlog (2 issues):

- C4WS-5 API access to Pro users (TO DO)
- C4WS-64 As an existing user I should be able to login to my account (TO DO)

The interface includes a search bar, filters, and a 'Create issue' button at the bottom of each section.

## Github logs





## Current and future work

For the current sprint we are wrapping up user management and, once it is done, we will work on the final phase of the application by adding some pro features within the application. Also, we will focus on the integrated system before the release of our application. Once the testing is done, we are planning to host our application on the server before our final presentation.