Final Report

Project Title: Post Pioneer

Course: COSC 4P02

Date: April 27, 2025

Group Number: 8

Group Members:

Karan Arora - 6226120 | ka16az/ka16az@brocku.ca

Charmvir Grewal - 7026057 | cg20gl/cg20gl@brocku.ca

Ritika Chaudhary- 7392459 | rc21mj@brocku.ca

Nadine Hoda - 7597545 | nh22dt@brocku.ca

Will Yochim - 7630924 | wy22pe@brocku.ca / yochimw@gmail.com

Mohammed Shihab Khateeb - 6867691 | mk19as@brocku.ca

Introduction

The project is a SaaS web application, utilizing JavaScript/HTML/REACT/Flask and Python, that allows users to create personalized social media posts.

The software enables users to streamline content generation by leveraging AI to produce relevant content and generate images and text.

The user can customize content for different platforms—for instance, Twitter or LinkedIn.

Users can set the frequency of the post generation bot, choosing between daily, weekly or monthly. The application will provide a centralized dashboard that provides analytical insights.

System Architecture

- Frontend
  - **React.js** – Component-based UI development
  - **Tailwind CSS -** For styling UI Components
  - **Material UI (MUI)** – Modern, responsive UI components
  - **Recharts** – Data visualization (engagement graphs)
- Backend
  - **Flask (Python)** – Lightweight API for post generation logic
  - **Deepseek (Ollama) and Stable Diffusion (Diffusers)** – AI-driven content generation
  - **Firebase Authentication** - User Authentication
  - **Firebase Realtime Database** – JSON-based NoSQL database for storing user inputs, posts, and settings
- Integration
  - **RESTful API (Flask ↔ React)** – Smooth communication between backend and frontend
  - **Firebase SDK** – Authentication and Realtime DB connection
  - **Custom Scheduling Logic** – To manage post frequency (daily/weekly/monthly)

Swe Processes + Design and implementation

Agile Methodology in Our Project

- Agile Framework
  - Adopted **Scrum** to enable iterative development and continuous improvement
- Sprint Structure
  - Followed a **2-week cycle**
  - Included **sprint planning, sprint reviews** and **retrospective** to align on goals, review progress, and reflect on improvements
- Team Collaboration
  - Initially conducted daily standups, which evolved into weekly **check-ins** and were later integrated into main team biweekly team meetings
  - Practiced **Pair Programming** to enhance code quality and knowledge sharing amongst team members

Agile Methodology in Our Project

- Task/workflow management
  - Used **JIRA** to manage user stories, track tasks, and monitor sprint progress
  - Each User Story was broken down into actionable tasks ("taskified")
- Version Control
  - Codebase has been hosted on **GitHub**
- Feedback Loop
  - Agile allowed regular meetings with stakeholders (TA and Professor) for **continuous feedback** and **iterative refinement** of the project.
- Documentation
  - Maintained **internal documentation** for development clarity
  - Provided user installation guides in the GitHub **README** for ease of use

Link to Jira : 4P02 Scrum Project
Link to Github : rc21mj/COSC4P02

Design and Implementation

Architectural Patterns

- Producer-Consumer Pattern:
  - Utilized for handling scheduled posts — the **Producer** creates scheduled content, which is then processed and displayed by the **Consumer (Dashboard)**.
- Client-server Architecture
  - Used to separate concerns — the **Client** handles the user interface and interactions, while the **Server** manages logic, data, and API communication.
- Version Control
  - **Git** was used for source control, with **GitHub** as the remote repository to enable collaboration, branching, and pull requests.
- Licensing
  - The project is open-source and released under the GNU General Public License v3.0

Features

- Account creation
- Social media account linking
- Post generation
- With image generation
- With uploaded image
- Post analytics dashboard
- Post editing functionality
- Post deletion functionality

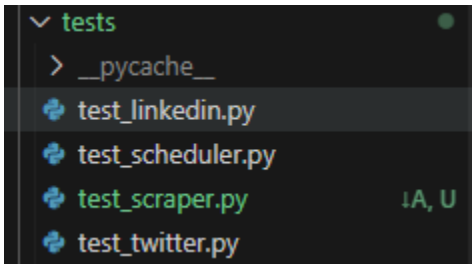
Testing

Automated Testing

- Automated testing was performed on functional requirements using the pytest library
- Focused upon testing basic functionality using mock data
- Aim to cover the parts of the code that interact with social media APIs, i.e. Twitter, LinkedIn, as well as the code responsible for interacting with Firebase db for scheduling.
- Tests are located within COSC4P02/PostPioneer/backend/tests

Manual testing

- Manual testing for functional requirements was done
- A document that details all manual test cases recorded (see below table)
- Test Description, Precondition, Test steps, Expected Results, Pass/Fail
- Manual testing for the front-end was carried out here as well. (For instance, login and register functionality)
- Testing was done by Developers of the team and third party testers
- Manual testing was done quite often
- Tested some Edge Cases as well
- Non-Functional testing was out of the scope of our testing as things like Availability and Reliability testing took too much scale to test

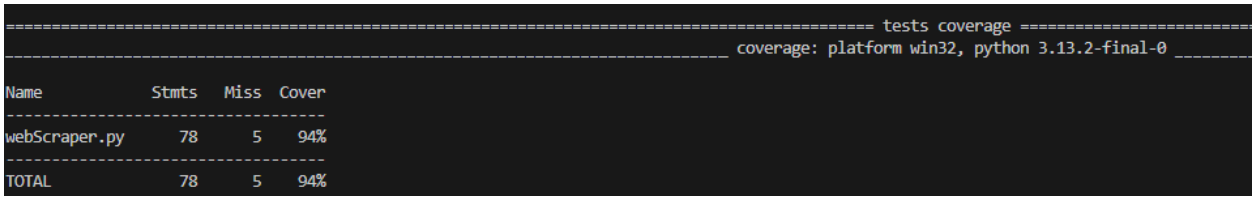



```
plugins: anyio-4.9.0, cov-6.1.1
collected 27 items

tests\test_linkedin.py ........
tests\test_scheduler.py ....
tests\test_scraper.py ......
tests\test_twitter.py .........

---------- coverage: platform win32, python 3.13.2-final-0 ----------

Name          Stmts   Miss  Cover
---------------------------------
mergedApp.py    292     66    77%
---------------------------------
TOTAL           292     66    77%
```



Test Description	Preconditions	Test Steps	Expected Result	Pass/Fail	Comments
Verify Home button navigates to homepage	N/A	Click the Home button in the navbar	User is redirected to homepage	Pass	
Verify Dashboard navigation	N/A	Click the Dashboard button	Dashboard page loads	Pass	
Verify Generate a Post page loads	User is logged in	Click on the Generate a Post! button	User is redirected to form to generate a post	Pass	
Verify Payment Plan page loads	User is logged in	Click the Payment Plan button	User is redirected to payment plan page	Pass	
Verify Settings page loads	User is logged in	Click the Settings button	User is redirected to Settings page	Pass	
Verify Sign out button logs user out	User is logged in	Click on Sign out	User is redirected to home page and session is cleared	Pass	Login session is cleared

Verify User Login button navigates to login screen	User is logged out	Click on User Login	User is redirected to the login page	Pass	
Check responsive layout on smaller screens	Device width < 768px	Resize browser window	Layout stays clean and items on web app adjust according ly	Pass	
UI consistency for navbar on all pages	User is on any page	Observe navbar layout and style	Navbar is present, buttons are spaced evenly, contrast is good, hover states work	Pass	
Submit login with empty email	User is on login page	Click "NEXT" with empty email field	Error message shown: “Enter your email address to continue”	Pass	
Submit login with invalid email format	User is on login page	Enter invalidemail and click "NEXT"	Error message shown: “That email address isn't correct”	Pass	

Submit login with valid unregistered email and valid password	User is on login page Email not in Firebase	1. Enter valid@example.com, click NEXT 2. Write password like test123, click SAVE	Create account workflow is entered and then user is navigated to homepage	Pass	User is added to firebase and user is signed in
Submit login with valid unregistered email and invalid password	User is on login page Email not in Firebase	1. Enter valid1@example.com, click NEXT 2. Write password like test1, click SAVE	Error is shown “ Strong passwords have at least 6 characters and a mix of letters and numbers”	Pass	
Submit with valid registered email and valid password	User is on login page Login details in Firebase	1. Enter valid@example.com, click NEXT 2. Write password test123, click SIGN IN	Sign in workflow is entered and user is navigated to homepage	Pass	User is signed in
Submit with valid registered email and invalid password	User is on login page Login details in Firebase	1. Enter valid@example.com, click NEXT 2. Write password test1234, click SIGN IN	Error is shown “The email and password you entered don't match”	Pass	

Submit valid support message	User is on Support page	1. Enter a topic in “Your Topic” 2. Enter a valid email in “Your Email” 3. Type a message 4. Click SEND MESSAGE	User is redirected to https://formsubmit.co/form/submission with a link back to the main website	Pass	An email should be received at the support email address, in this case rockheadrush@gmail.com
Submit with invalid email format	User is on Support page	Enter text like test@ in “Your Email” field and click SEND MESSAGE	Error shown for invalid email input	Pass	
Submit with empty fields	User is on Support page	Click SEND MESSAGE with all fields empty	Error shown for empty fields	Pass	
Submit a generate post	User is logged in User is on Generate a Post page	Click generate post with any fields (preferably default)	A post is generated and the user is redirected to the edit posting page	Pass	
Submit an edited post	User is logged in User has generated a post	Click submit a post	The user is redirected to the home page	Pass	A post is now added to the specified platform
Scheduler generates post	Hourly scheduled post is in database	1. Open backend 2. Wait for scheduler trigger	Post generation should trigger automatically	Pass	

		3. Check console logs			
Press Delete Account button	User is logged in User is on Settings page	Click the DELETE ACCOUNT button	User should be logged out	Pass	The user account is removed from Firebase
Press the Sign Out button on the settings page	User is logged in User is on Settings page	Click the sign out button	User should be logged out	Pass	
Press the Add LinkedIn button	User is logged in User is on Settings page	1. Click the Add LinkedIn button 2. Proceed through LinkedIn proprietary authorization flow 3. Submit	User should be redirected back to Settings page	Pass	LinkedIn credentials are added in database
Press the Remove LinkedIn Button	User is logged in User is on Settings page	Click the Remove LinkedIn button	User should be redirected back to Settings page	Pass	LinkedIn credentials are removed from database
Press the Add Twitter Button	User is logged in User is on Settings page	1. Click the Add Twitter button 2. Proceed through LinkedIn Twitter authorization flow 3. Submit	User should be redirected back to Settings page	Pass	Twitter credentials are added in database

Press the Remove Twitter button	User is logged in User is on Settings page	Click the Remove Twitter button	User should be redirected back to Settings page	Pass	Twitter credentials are removed from database
Generate Post without an Image	User is logged in and User is on the Generate a post page	1. Fill out the entire post form 2. Select “None” for add image 3. Click submit	Users should not see an image in the edit post form.	Pass	An image is not generated or used if not requested.
Generate Post with an Uploaded Image	User is logged in and User is on the Generate a post page	1. Fill out the entire post form 2. Select “Upload” for add image 3. Upload an image 4. Click submit	Users should be able to upload an image and see that uploaded image on the edit post form.	Pass	The uploaded image is displayed and can be posted.
Generate Post with a generated Image	User is logged in and User is on the Generate a post page	1. Fill out the entire post form 2. Select “Generate” for add image 4. Click submit	Users should see the generate d image after some time in the edit post page	Pass	The image has been generated and can be posted.
Display weekly, daily and monthly engagement stats on user dashboard	User is logged in and User is on the Dashboard page	1. Select daily, weekly and monthly on the graph.	Users should be able to see the difference in time frames.	Pass	The data is different for each timeframe.

Delete scheduled post	User is logged in and on the dashboard	1. Find post to delete 2. Select delete button	Users should see the post disappear from the list	Pass	Delete function is working

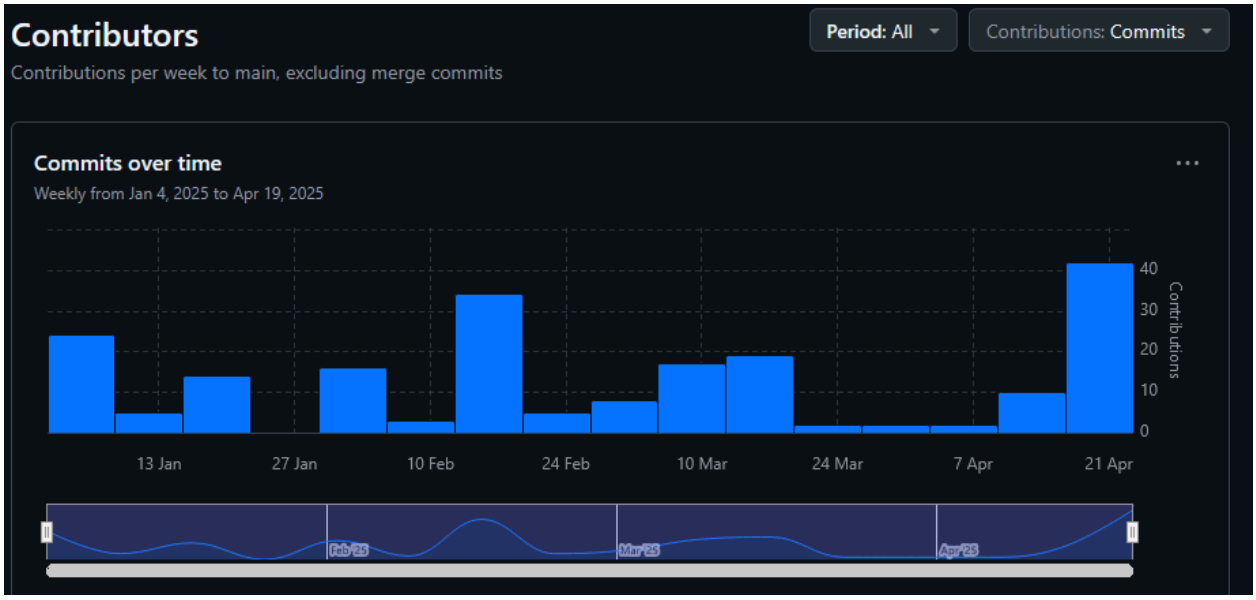
User, technical, and installation manuals

Internal Documentation
https://github.com/rc21mj/COSC4P02/blob/main/Internal%20Documentation.pdf

Instructions To Run

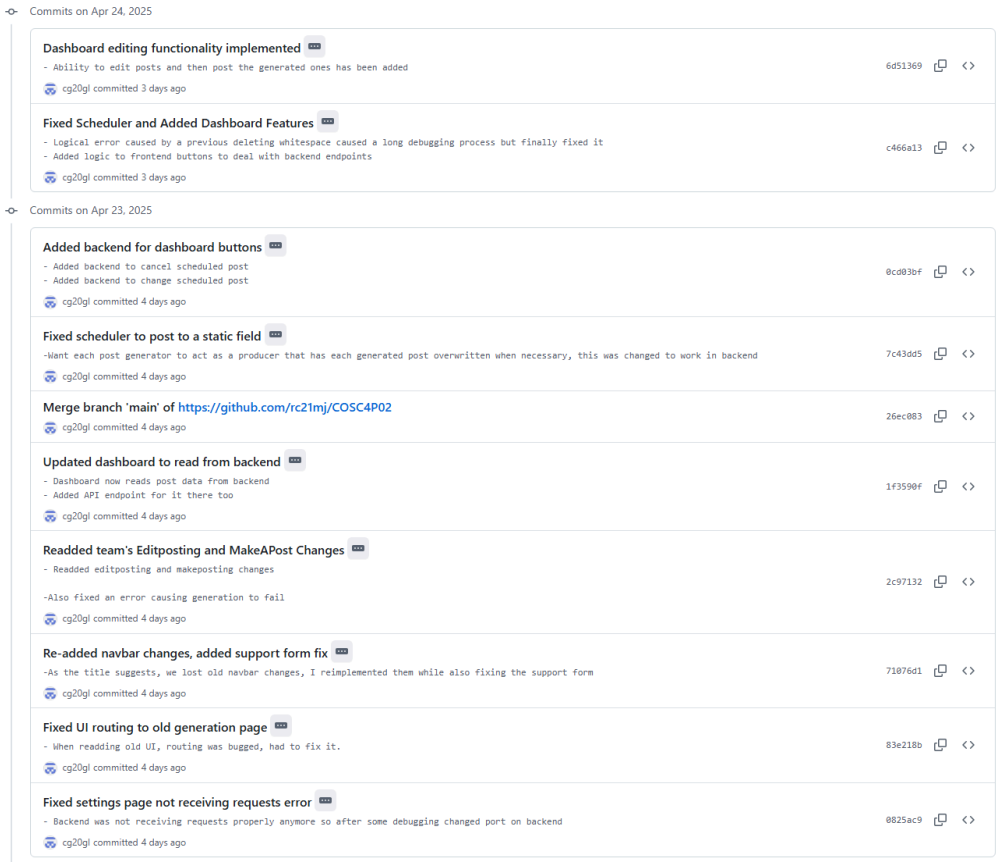
- Install Ollama
- 'pip install ollama'
- 'pip install diffusers'
- 'pip install torch'
- 'pip install transformers'
- 'ollama serve'
- Install DeepSeek (currently using 1.5b)
  - Before your first run use the command: `ollama run deepseek-r1:1.5b`
  - For other versions change the postfix in the commandline (currently 1.5b) and change the model field to that model in Save.py
  - For running timyllama, use command 'ollama run tinyllama'
- To run the front end, open your command line and navigate to `PostPioneer/frontend`
  - Have `npm` installed
  - Run `npm install react-scripts` and `npm install firebaseui`
  - Run `npm start`
- To run the back end, open your command line and navigate to `PostPioneer/4p02 testing/postpioneer`
  - Have `Python` and `pip` installed
  - Run `pip install firebase-admin`, and `pip install ollama`
  - Run `py mergedApp.py`

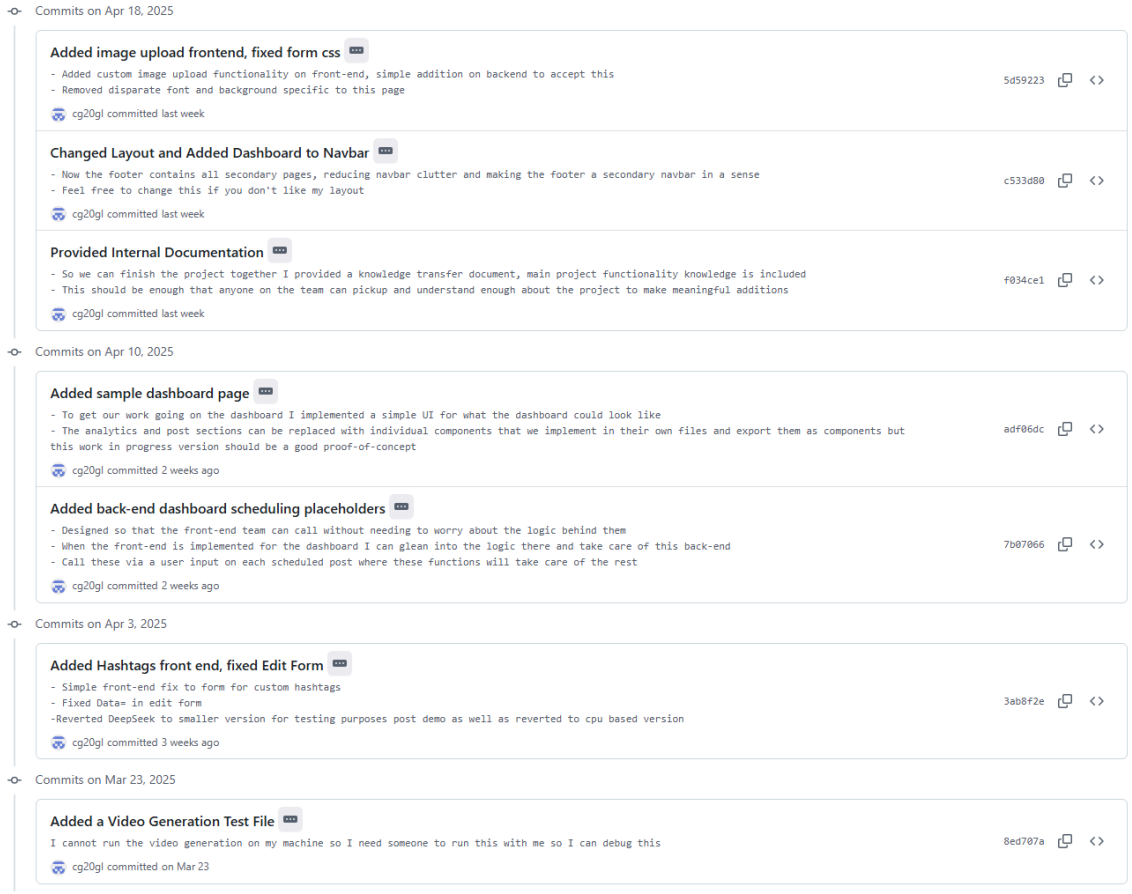
Each member’s work + GitHub logs



Charmvir:

- Text Generation (Deepseek & Ollama to run locally)
- Image Generation (Stable Diffusion, tried other models but this was the most compatible)
- Scheduling (APScheduler with Flask)
- Authentication, Account deletion (Firebase)
- Support form (formsubmit.co)
- Firebase Setup (Authentication and Realtime Database)
- Initial Database Design, Database management & rewriting reads and writes to work with database instead of CSV (Firebase Realtime Database)
- Jira Management (Setting up sprints, moving stories)
- Internal Documentation (I worked with most core features so it made sense for me to write)
- Manual Test Cases (Primarily for functionality but also checked boundary values)
- Misc front-end (Described in further Detail on GitHub)
- Video Generation (not integrated in the final product as I could not test on my AMD hardware and needed NVIDIA)



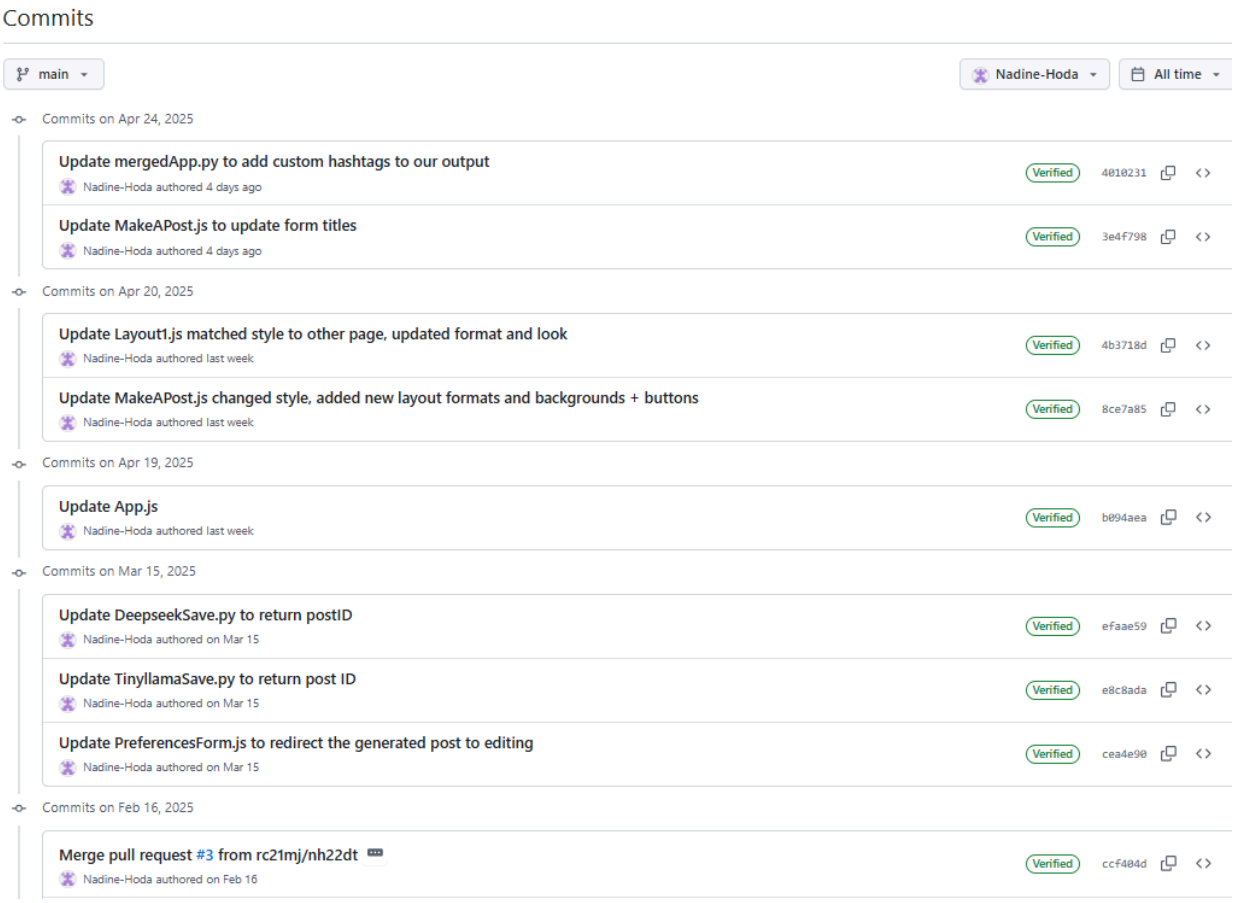


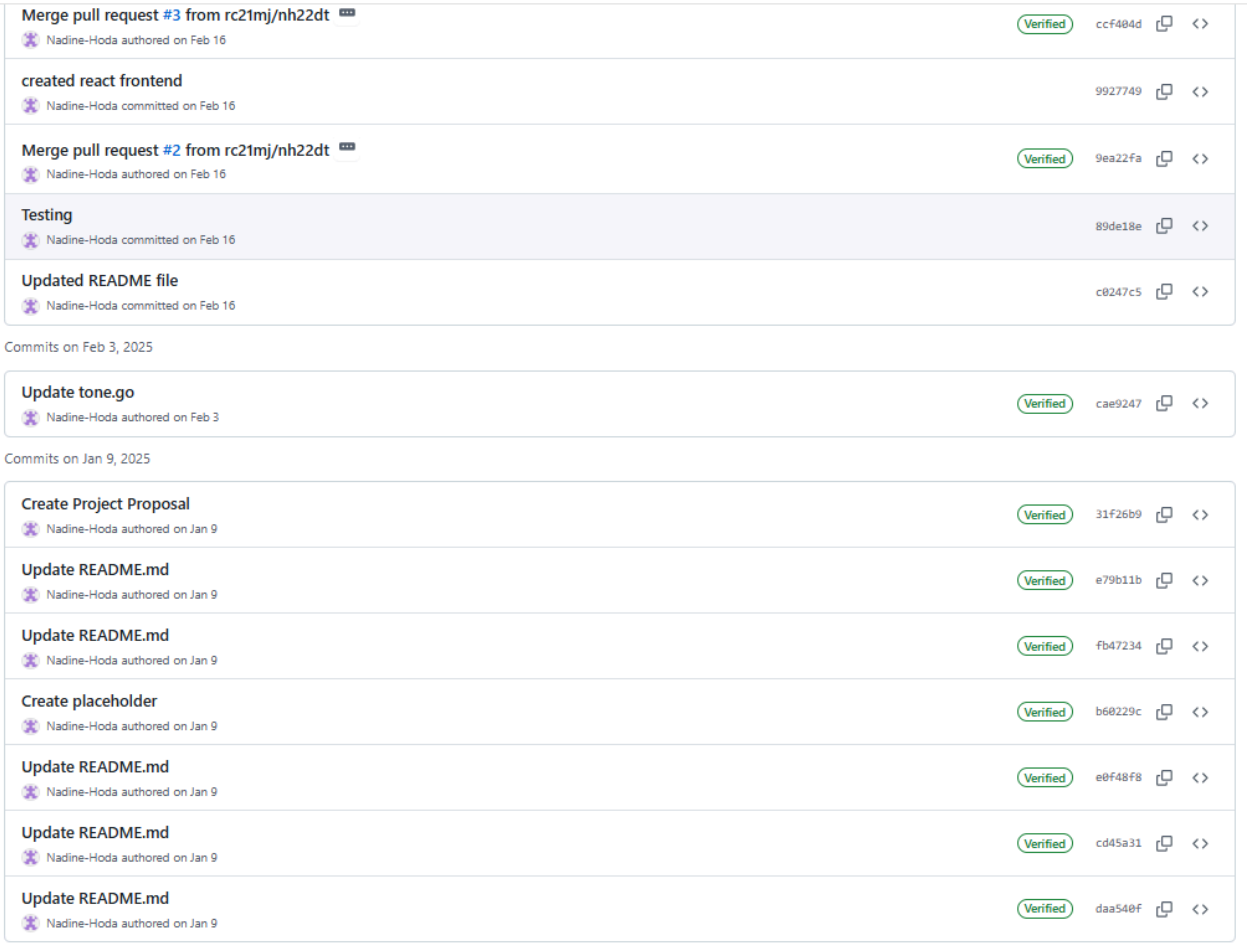
Karan

- Frontend Design and Development
- Registration to the application
- Twitter Authorization (Saving of Twitter logins)
- Twitter Posting Logic
- Twitter Analytics Logic
- Basic vs Pro User Logic
- Payment Plan Logic (Google payment Integration)
- Database Setup
- Manual Testing
- Documentation
- Repository Management

Nadine

- Frontend Design and development
- Refinement of home/landing page
- Making a uniform look across the platform
- Generate a post form
- Edit post form
- Documentation
- Repository management

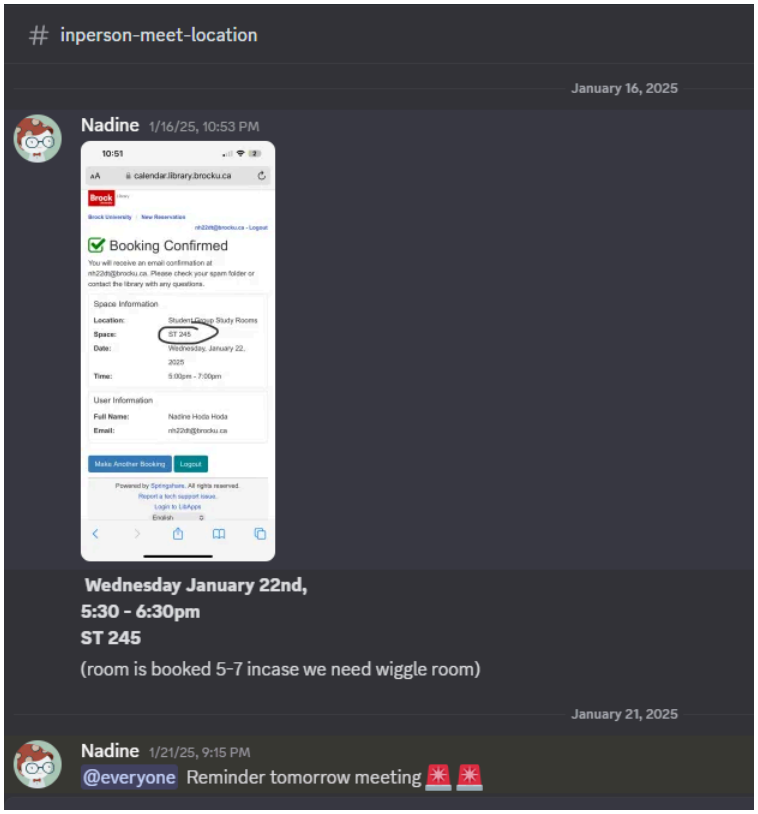


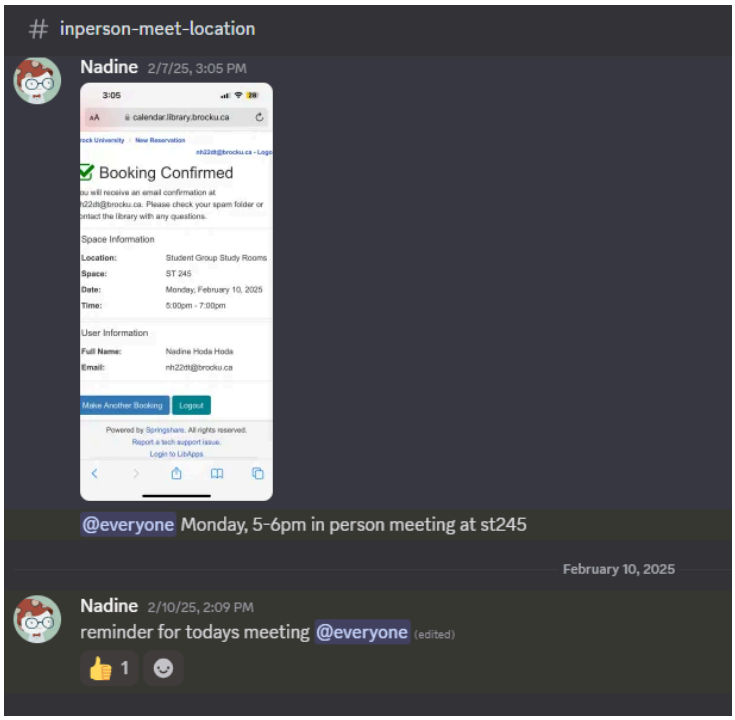


Aside From implementation I took on a key role in team management and took on the following responsibilities to help with the teams organization and time management :

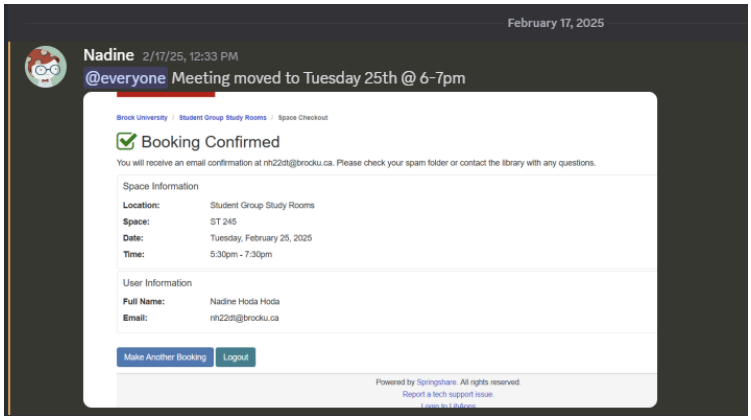
- Conducted meetings twice a week

Below is evidence of the channel I created to remind everyone of the team meetings that I took the initiative to book (time + location) and conduct

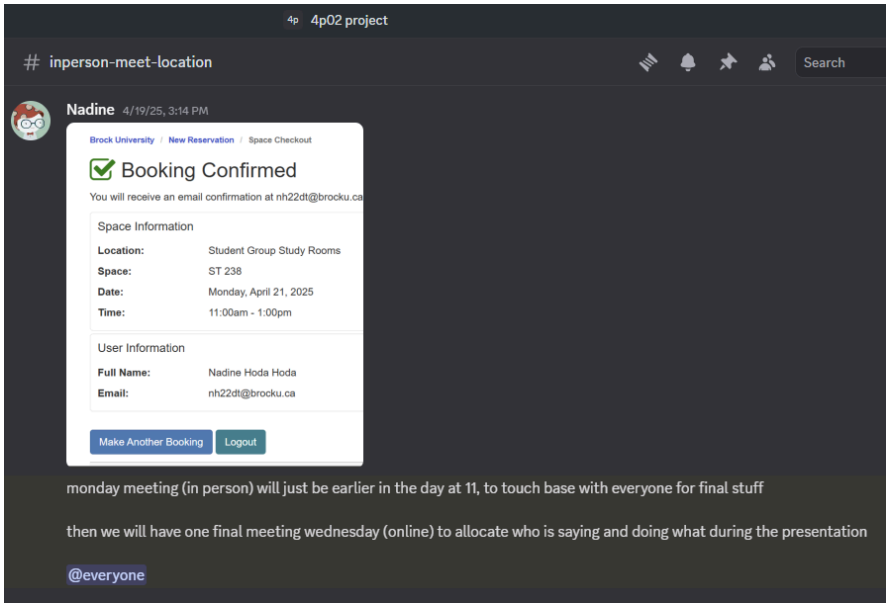




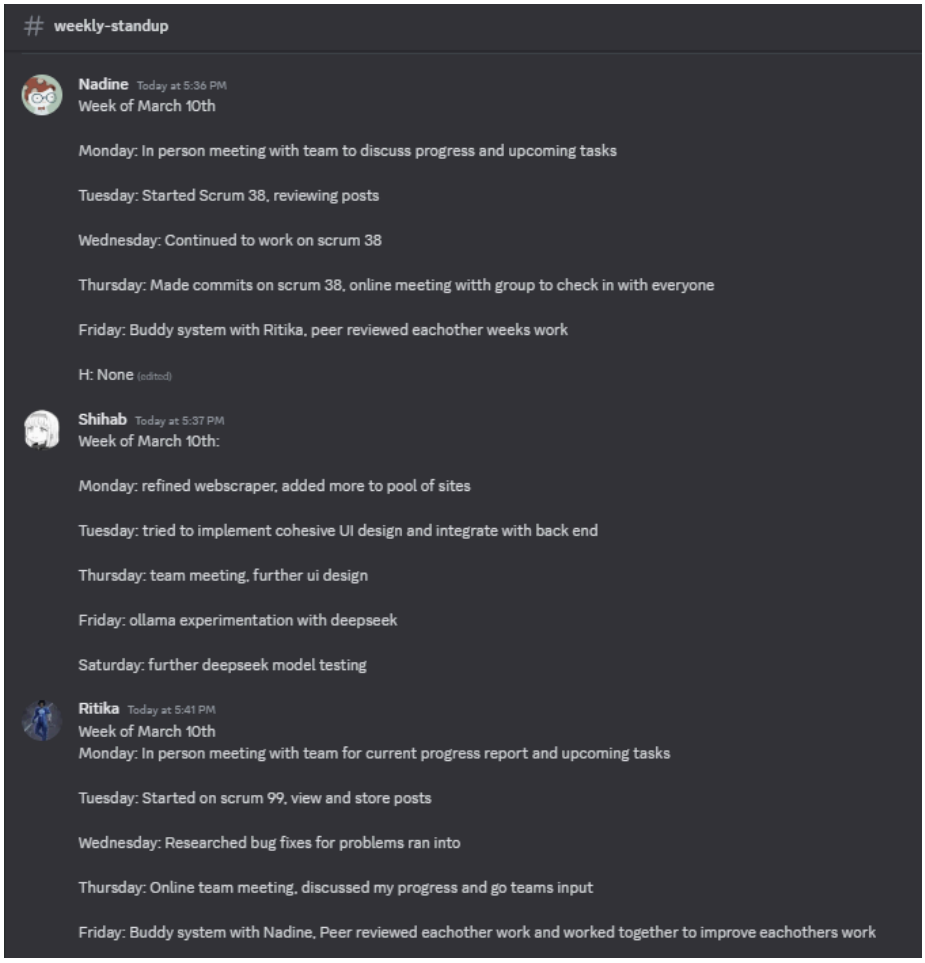
Also an example of me taking responsibility to move around the meetings and accommodate for when things came up

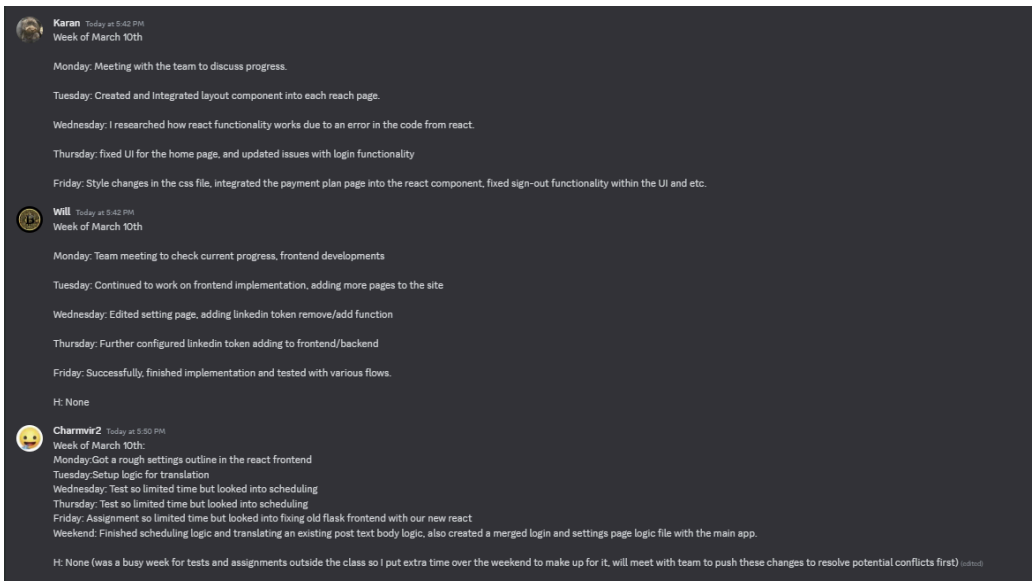


- Reviewed progress in each meeting +
- Set and adjusted goals for the team accordingly, see example below of last two points, I conducted an additional meeting where necessary



- To make sure goals that were being set in the meetings I conducted were being met I suggested the idea of a weekly stand up in our discord channel so we could watch each other progress



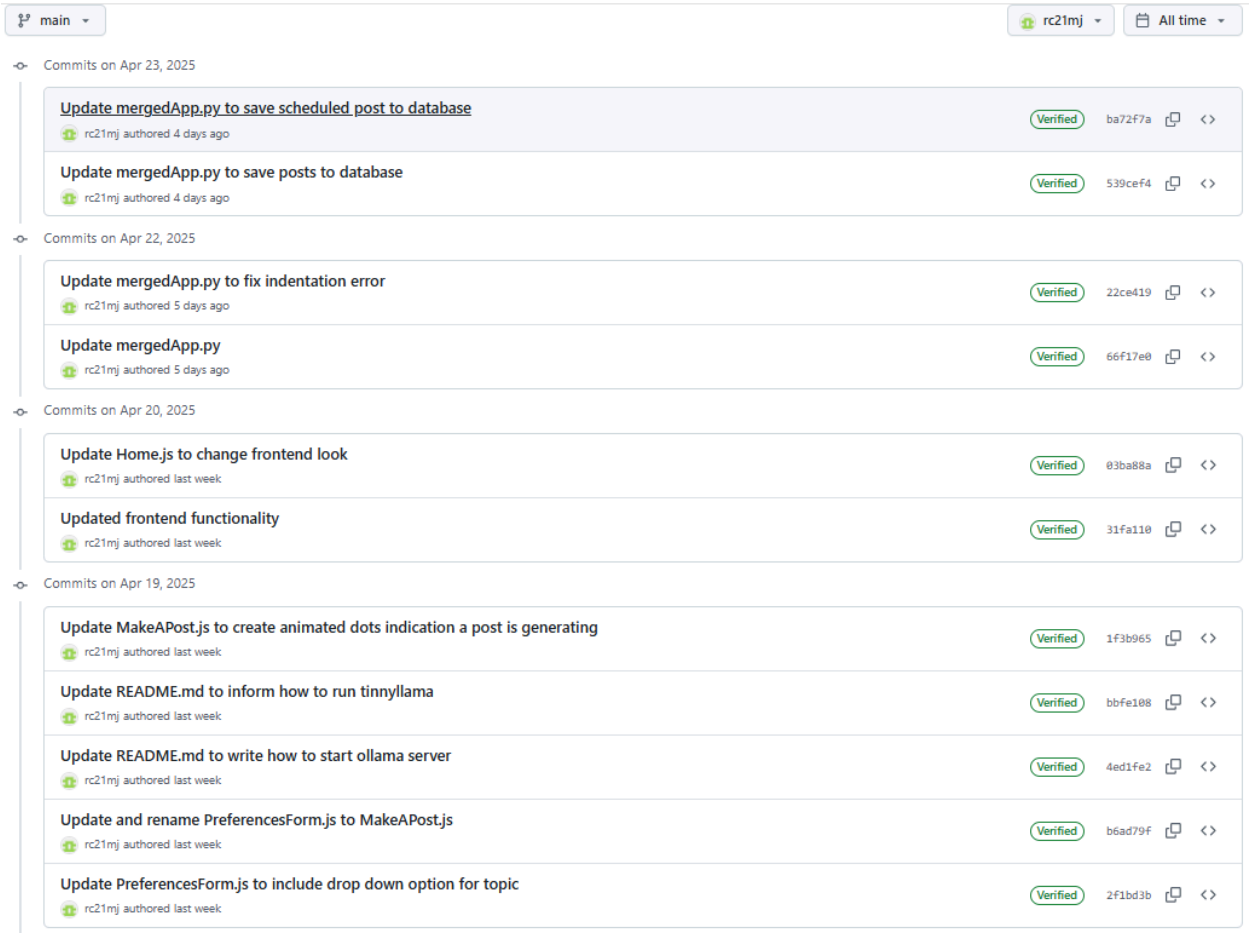


- Practicing the above structure i suggested to the team helped me see if anyone needed us to step in right away and call a meeting or if it could wait till the meeting day

- I was able to call meetings if anyone was struggling through this and helped to contribute to the team's management and success

Ritka

- Generate Post Form
- Edit Post Form
- Backend logic for handling user preferences
- Interacted with Firebase Database to save and fetch user data/posts
- Experimented with OpenAI and TinyLlama for generating text
- Documentation
- GitHub Repository Creation and Management



Commits on Mar 18, 2025

Update PreferencesForm.js to allow user to edit their post rc21mj authored on Mar 18	Verified	4bbfd89
Update EditPosting.js to allow user to edit their post rc21mj authored on Mar 18	Verified	d24eb07
Update App.js to call EditPost.js rc21mj authored on Mar 18	Verified	dfb22a5

Commits on Mar 15, 2025

Create TinyllamaSave.py doing web scraping using tinyllama rc21mj authored on Mar 15	Verified	a39dd24
Rename Save.py to DeepseekSave.py rc21mj authored on Mar 15	Verified	226afbb
Update README.md to run deepseek rc21mj authored on Mar 15	Verified	389e2c5
Update App.js rc21mj authored on Mar 15	Verified	66c8083
Add files via upload rc21mj authored on Mar 15	Verified	1f0f6eb

Commits on Feb 18, 2025

Delete frontend directory rc21mj authored on Feb 18	Verified	9b723d8
Merge pull request #4 from rc21mj/databaseConnection rc21mj authored on Feb 18	Verified	9638853
Merge branch 'main' into databaseConnection rc21mj authored on Feb 18	Verified	4753f7b

Commits on Feb 17, 2025

Update Edit function		

added edit functionality before posting
rc21mj committed on Feb 16 — dee431d

added edit functionality before posting
rc21mj committed on Feb 16 — f2fb5a0

added edit option for user preferences
rc21mj committed on Feb 16 — fe56744

Changed preferenceForm.js
rc21mj committed on Feb 16 — c08e4a0

Pushing Sprint 2, Scrum 38
rc21mj committed on Feb 16 — 7ec3cc2

Commits on Feb 15, 2025

Updating Scrum 66
rc21mj committed on Feb 15 — 689cc61

Changed frontend to React from Fiber
rc21mj committed on Feb 15 — 476a7b5

Commits on Feb 3, 2025

Update tone.go
rc21mj authored on Feb 3 — Verified — 70d99b3

Update Save.py
rc21mj authored on Feb 3 — Verified — c4ed348

Update LLMintegration.py
rc21mj authored on Feb 3 — Verified — 3a9e3e4

Commits on Feb 2, 2025

Changed tone.py to save user preferences
rc21mj committed on Feb 2 — 9237634

User topic/tone selection
rc21mj committed on Feb 2 — 03c4120

main — rc21mj — All time

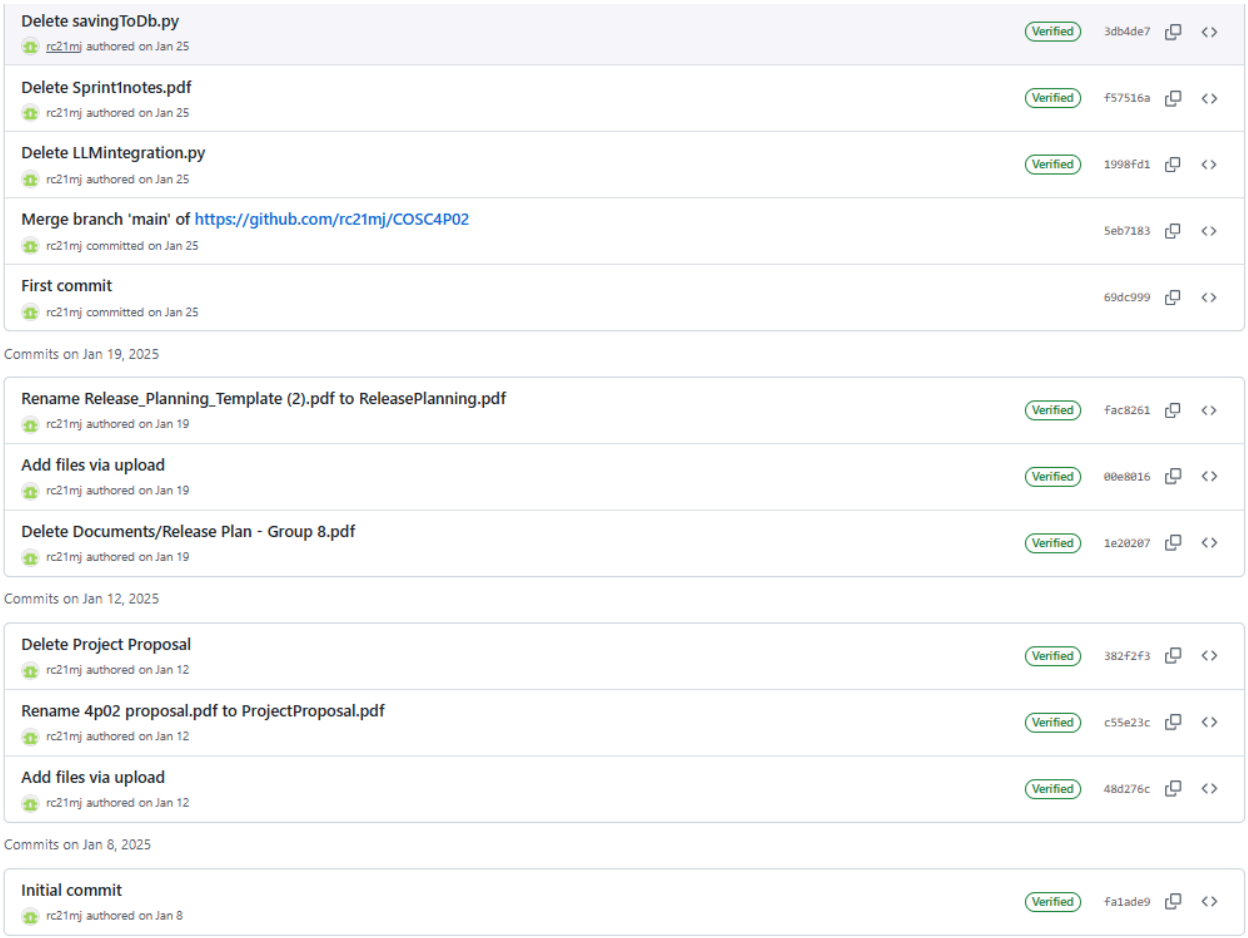
Commits on Jan 25, 2025

Delete Sprint1/Scrum66/h
rc21mj authored on Jan 25 — Verified — 5807c2b

Add files via upload
rc21mj authored on Jan 25 — Verified — 8bc3f5b

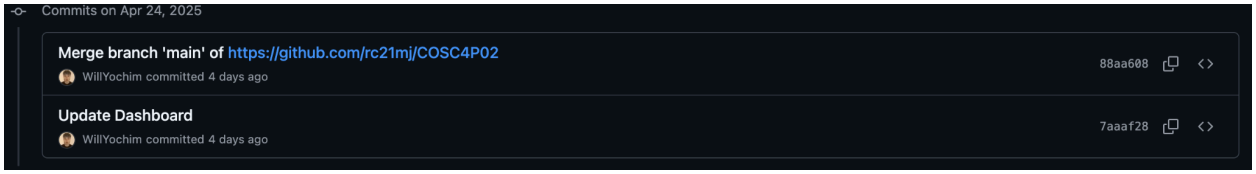
Create h
rc21mj authored on Jan 25 — Verified — 6321ad2

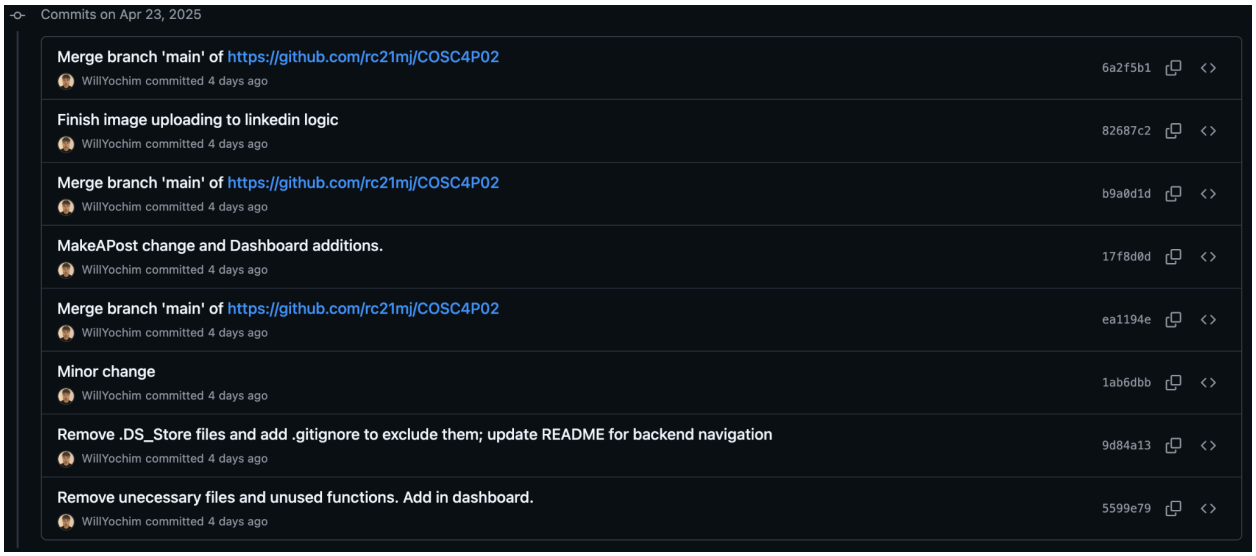
Delete userPreference.html
Verified — 377b941

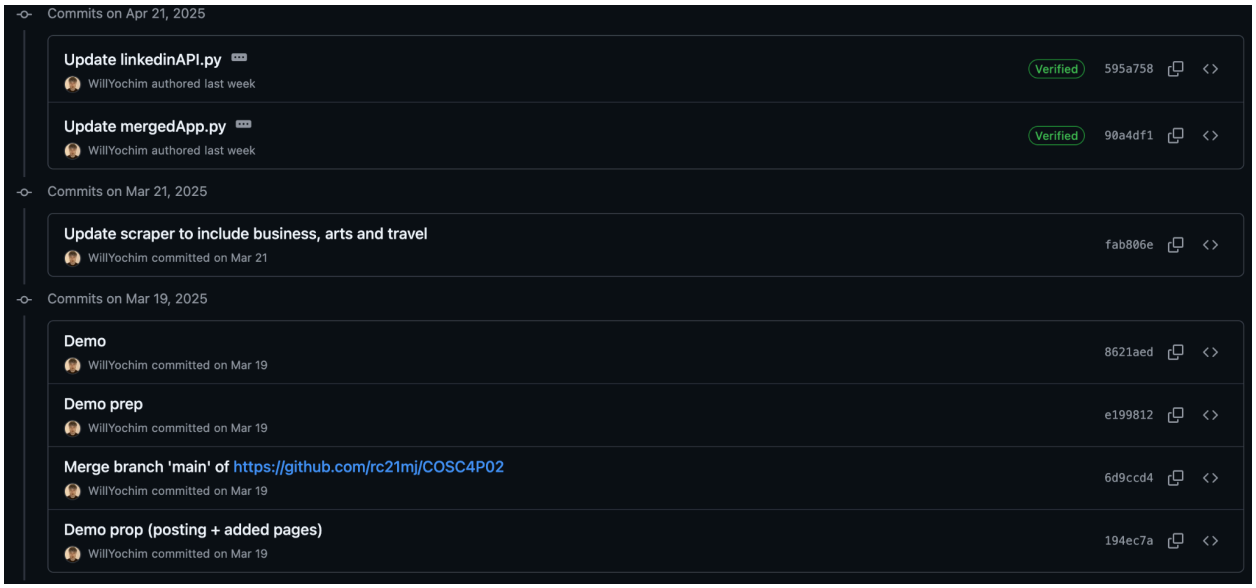


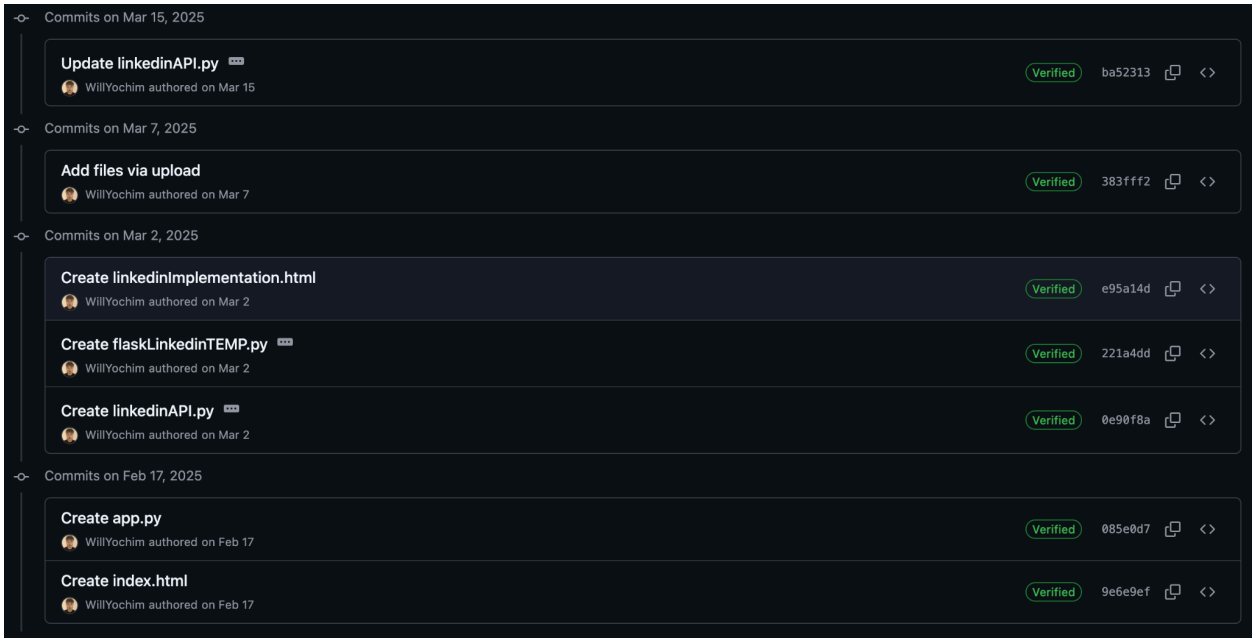
Will

- Frontend Setup
- Base frontend design
- Linkedin Posting setup (text posting, image posting)
- Oauth logic design and database storage
- User Dashboard
- Generate Post Form
- Documentation
- Repository management
- Image and text generation integration
- Web Scraper
- Manual Test Cases


Commits on Apr 24, 2025
Merge branch 'main' of https://github.com/rc21mj/COSC4P02
WillYochim committed 4 days ago
88aa608
Update Dashboard
WillYochim committed 4 days ago
7aaaf28


Commits on Apr 23, 2025
Merge branch 'main' of https://github.com/rc21mj/COSC4P02
WillYochim committed 4 days ago
6a2f5b1
Finish image uploading to linkedin logic
WillYochim committed 4 days ago
82687c2
Merge branch 'main' of https://github.com/rc21mj/COSC4P02
WillYochim committed 4 days ago
b9a0d1d
MakeAPost change and Dashboard additions.
WillYochim committed 4 days ago
17f8d0d
Merge branch 'main' of https://github.com/rc21mj/COSC4P02
WillYochim committed 4 days ago
ea1194e
Minor change
WillYochim committed 4 days ago
1ab6dbb
Remove .DS_Store files and add .gitignore to exclude them; update README for backend navigation
WillYochim committed 4 days ago
9d84a13
Remove unecessary files and unused functions. Add in dashboard.
WillYochim committed 4 days ago
5599e79

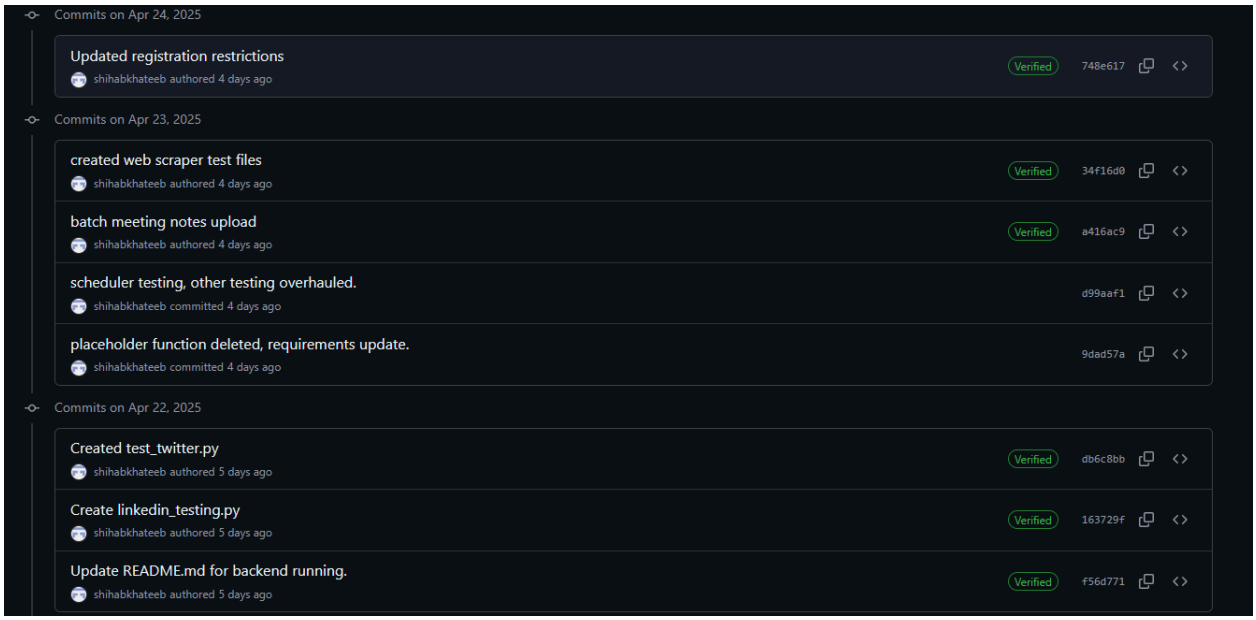

Commits on Apr 21, 2025
Update linkedinAPI.py
WillYochim authored last week
Verified
595a758
Update mergedApp.py
WillYochim authored last week
Verified
90a4df1
Commits on Mar 21, 2025
Update scraper to include business, arts and travel
WillYochim committed on Mar 21
fab806e
Commits on Mar 19, 2025
Demo
WillYochim committed on Mar 19
8621aed
Demo prep
WillYochim committed on Mar 19
e199812
Merge branch 'main' of https://github.com/rc21mj/COSC4P02
WillYochim committed on Mar 19
6d9ccd4
Demo prop (posting + added pages)
WillYochim committed on Mar 19
194ec7a

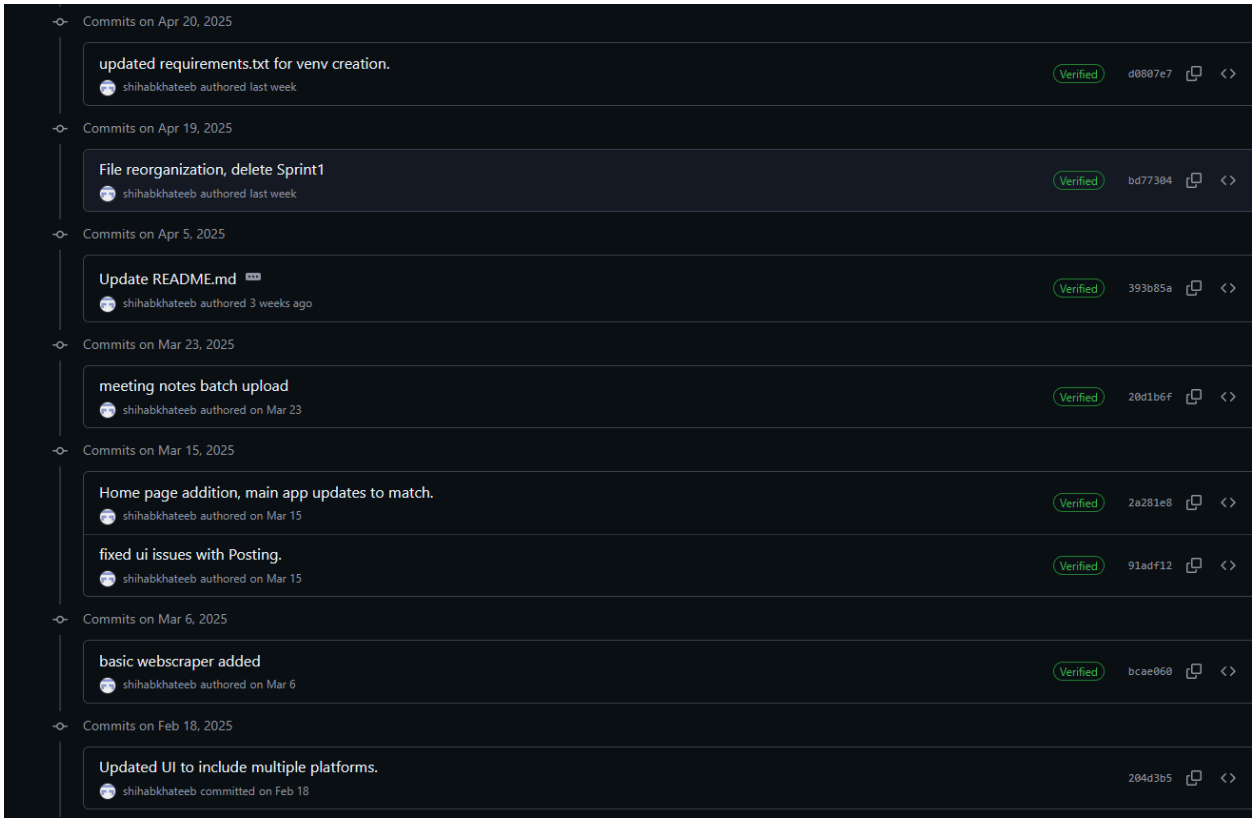

Commits on Mar 15, 2025
Update linkedinAPI.py
WillYochim authored on Mar 15
Verified
ba52313
Commits on Mar 7, 2025
Add files via upload
WillYochim authored on Mar 7
Verified
383fff2
Commits on Mar 2, 2025
Create linkedinImplementation.html
WillYochim authored on Mar 2
Verified
e95a14d
Create flaskLinkedinTEMP.py
WillYochim authored on Mar 2
Verified
221a4dd
Create linkedinAPI.py
WillYochim authored on Mar 2
Verified
0e90f8a
Commits on Feb 17, 2025
Create app.py
WillYochim authored on Feb 17
Verified
085e0d7
Create index.html
WillYochim authored on Feb 17
Verified
9e6e9ef


Commits on Feb 3, 2025
Create sampleFlaskApp.py
WillYochim authored on Feb 3
Verified
1bff263
Create linkedinAPI.py
WillYochim authored on Feb 3
Verified
6fe4534
Delete Sprint1/Scrum97
WillYochim authored on Feb 3
Verified
09ae5bc
Create Scrum97
WillYochim authored on Feb 3
Verified
820f564
Commits on Jan 10, 2025
Update README.md
WillYochim authored on Jan 10
Verified
cd5c2b3
Update README.md
WillYochim authored on Jan 10
Verified
1003610
Update README.md
WillYochim authored on Jan 10
Verified
ff2a378
Update README.md
WillYochim authored on Jan 10
Verified
d5cab32
Update README.md
WillYochim authored on Jan 10
Verified
7ff93f8
Update README.md
WillYochim authored on Jan 10
Verified
cee0512
Rename PostPinoeer.png to PostPioneer.png

Shihab

- Front end design
- Backend Ollama integration
- Automated testing using Pytest
- Focused on backend logic for API integration with LinkedIn and Twitter + model interaction through Ollama and web scraping
- Web scraping
- Prompt engineering
- Documentation
- Scrum master




Commits on Apr 20, 2025
updated requirements.txt for venv creation.
shihabkhateeb authored last week
Verified
d0807e7
Commits on Apr 19, 2025
File reorganization, delete Sprint1
shihabkhateeb authored last week
Verified
bd77304
Commits on Apr 5, 2025
Update README.md
shihabkhateeb authored 3 weeks ago
Verified
393b85a
Commits on Mar 23, 2025
meeting notes batch upload
shihabkhateeb authored on Mar 23
Verified
20d1b6f
Commits on Mar 15, 2025
Home page addition, main app updates to match.
shihabkhateeb authored on Mar 15
Verified
2a281e8
fixed ui issues with Posting.
shihabkhateeb authored on Mar 15
Verified
91adf12
Commits on Mar 6, 2025
basic webscraper added
shihabkhateeb authored on Mar 6
Verified
bcae060
Commits on Feb 18, 2025
Updated UI to include multiple platforms.
shihabkhateeb committed on Feb 18
204d3b5


Commits on Feb 3, 2025
14-1-25 Meeting with Stakeholder
shihabkhateeb authored on Feb 3
Verified
bcc21ec
18-1-25 Meeting Notes
shihabkhateeb authored on Feb 3
Verified
8382d48
16-1-25 Meeting Notes
shihabkhateeb authored on Feb 3
Verified
d31e5e4
15-1-25 Meeting Notes
shihabkhateeb authored on Feb 3
Verified
a8a1eb9
14-1-25 Meeting Notes
shihabkhateeb authored on Feb 3
Verified
ff172d6
Create 10-1-25 Meeting
shihabkhateeb authored on Feb 3
Verified
6199295
Commits on Jan 19, 2025
Add files via upload
shihabkhateeb authored on Jan 19
Verified
1e20e5b
replaced release plan file.
shihabkhateeb authored on Jan 19
Verified
0d41aed
Release planning document uploaded.
shihabkhateeb authored on Jan 19
Verified
c4ba449
Commits on Jan 12, 2025
8-1-25 Meeting Notes
shihabkhateeb authored on Jan 12
Verified
eba2ed8
Project proposal uploaded.
shihabkhateeb authored on Jan 12
Verified
643d2bf

Known issues

- Payment plan works and updates users but the features do not care about the payment plan, this could be solved through adding logic on the make a post page to check the user's payment plan and rendering corresponding elements. Furthermore, the backend would verify the user payment plan, just in case a user called the endpoint without using the UI.
- Dashboard edit schedule timing button stopped functioning so was removed for the time being, a front-end commit likely broke it as it was functioning previously.
- Rarely, uploading an image and certain form data can cause an error in the "make a post" form. This is difficult to replicate so the issue could not be fixed at the time of writing.
- Generation speed is slow if the hardware provided for the backend is weaker as the generation is locally hosted.
- Sometimes the mail endpoint that the support form is connected to can go offline, so be aware of the status of formsubmit.co

Conclusion

One of the key principles of Agile is to embrace change, expecting system requirements to evolve and designing the system to adapt accordingly.

Throughout the project, we faced several challenges that required us to rethink our sprints, rework our timeline, and adapt our communication strategy to stay aligned as a team. By accepting change as part of the process, we became more flexible and solution-focused.

This experience gave us a real-world understanding of what it means to work on a software engineering project — from collaboration and tool usage to strategic pivots. These lessons will be something we embrace as we carry forward the mindset of adaptability, teamwork, and continuous learning into our future careers.