Department of Computer Science
COSC 4P02 - Software Engineering II
WINTER 2025

AI-Powered Newsletter and Social Media Content Generator

Progress Report 2

Professor: Naser Ezzati Jivan

Prepared by: Group 14

Arifuzzaman (Scrum Master)	(7088214)
Hridoy Rahman	(7340334)
Jeffin Sam Joji	(7344526)
Kabir Sethi (Product Owner)	(6933782)
Sahil Rashid	(6954739)
Yuanhan Huang	(7642440)
Jayant Saini	(7331556)

Date: March 24, 2025

Contents

1 Overview

In the second stage of our project, we built upon the strong foundation laid in the initial phase, focusing on enhancing functionality, user engagement, and automation. Over the past few weeks, our team successfully completed sprints 3 and 4, delivering key features such as content summarization using LLM, social media integration, content scheduling, and newsletter sharing using templates on social media and email. These additions have greatly improved content accessibility and streamlined the sharing process, enhancing the overall user experience.

We further refined the user interface by developing dynamic preview functions for SNS platforms, implementing email-sharing options for Gmail and Outlook, and optimizing the UI for content scheduling. In the back-end, we reinforced automation, ensuring seamless content distribution based on user-defined intervals, providing greater flexibility and ease of use.

This document provides detailed insights into our design and implementation choices, sprint cycles, scrum meetings, and the testing of key components. It also highlights the challenges we encountered and the solutions we devised, along with a clear roadmap for the next steps. Several diagrams are included to visually represent workflows and functionalities, giving a comprehensive view of our progress and future direction.

By adhering to Agile principles, we have ensured a flexible and adaptive development process, allowing us to respond quickly to user needs and project requirements. Looking ahead, we will continue refining existing features and expanding the platform's functionality to further enhance user engagement and satisfaction.

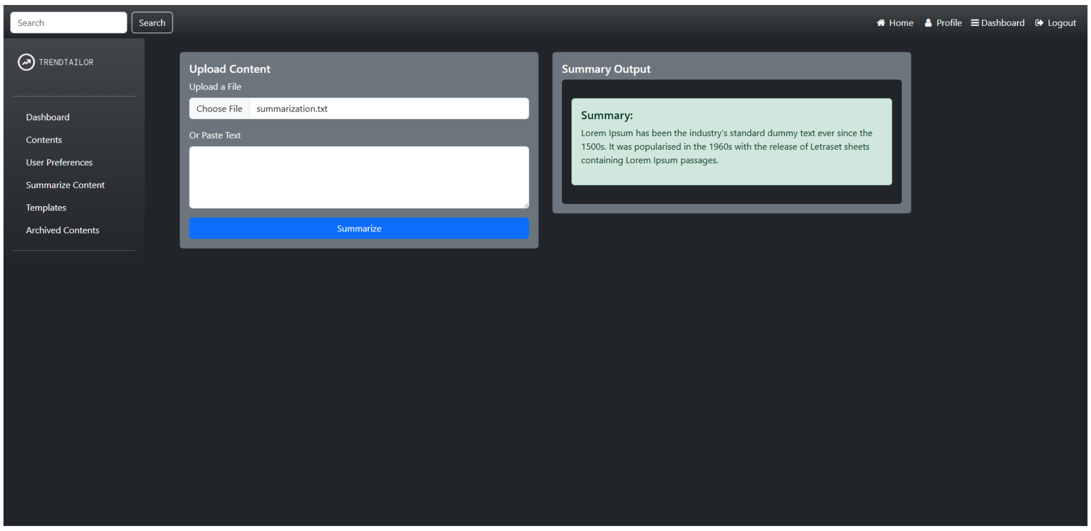
2 Design & Implementation

2.1 Sprint-3

2.1.1 SCRUM - 5

In this sprint, we implemented LLM-based summarization, allowing users to upload content via a text box or a file. The uploaded text is then processed and summarized using a large language model. We leveraged the Transformers library's pipeline functionality with Facebook's BART-Large-CNN model to generate summaries within our Trend Tailor app. The following demonstration provides the implementation details:

Figure 2.1 demonstrates the Summarization UI, where the user can upload a file or text to summarize the contents. The system takes one of the inputs from the user and summarizes the content. Furthermore, the UI was developed using HTML and Bootstrap, while Django handles the backend form and parses the user inputs, either from the file or the text input box, and summarizes them. Additionally, this functionality was tested using various sample inputs. We will conduct a proper unit testing in our next sprint.

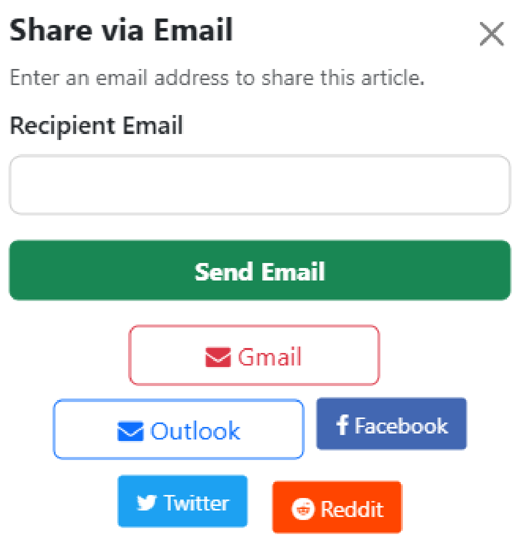


Summary:
Lorem Ipsum has been the industry's standard dummy text ever since the 1500s. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages.

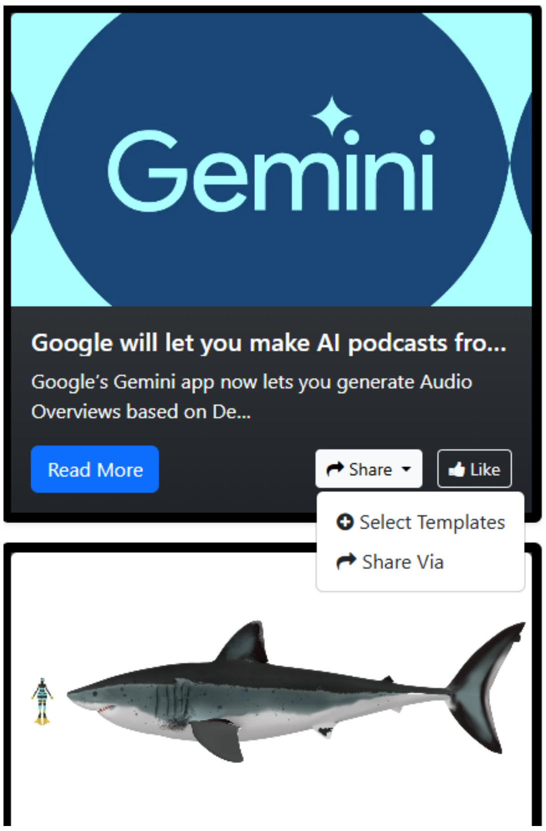
Figure 2.1: Summarization User Interface

2.1.2 SCRUM - 7

This section discusses the design and implementation of SCRUM - 7. In this phase, we focused on enhancing content distribution by integrating email and social media sharing features into the front-end.



(a) Sharing via Email Module

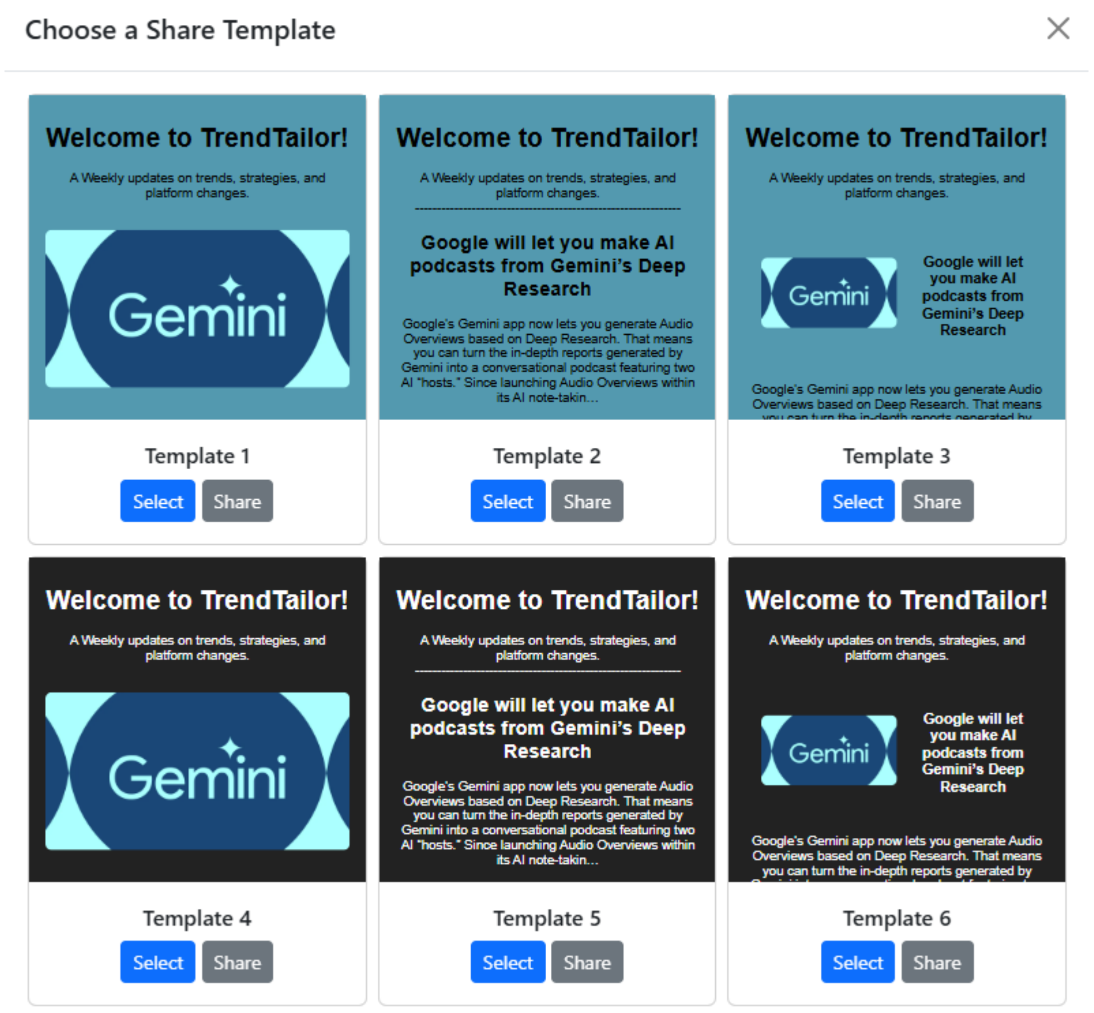


Google will let you make AI podcasts fro...
Google's Gemini app now lets you generate Audio Overviews based on De...

(b) Sharing Option UI

Figure 2.2: Sharing Workflow Screens

We implemented the email-sending feature by utilizing a "Share Via" button that opens a Bootstrap modal. Inside the modal, we created a simple form prompting the user to input an email address to share the article, as shown in Figure 2.2(a). The sharing options include email and social media platforms such as Outlook, Gmail, Facebook, Reddit, and Twitter. When the "Share Via" button is clicked, the modal opens, allowing the user to select a sharing option, as seen in Figure 2.2(a). Upon selecting a social media or email platform, the system automatically pulls articles from the database and shares the content directly to the respective platform. This feature enables users to quickly and easily share content, increasing visibility and engagement, while ensuring that the appropriate URLs and metadata are formatted correctly for each platform. The system is not limited to sharing default content. We created several template options from which the user can select, and then share the respective template via email, as shown in Figure 2.2(b). After selecting a template, as seen in Figure 2.3, an input form appears, prompting the user for an email address. Once the user provides the email and clicks "Send Email," the content, along with the selected template, is shared with the provided email address, as shown in Figure 2.4(a).


Choose a Share Template

Welcome to TrendTailor!

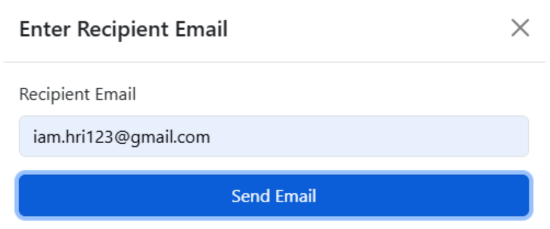
A Weekly updates on trends, strategies, and platform changes.

Google will let you make AI podcasts from Gemini's Deep Research

Google's Gemini app now lets you generate Audio Overviews based on Deep Research. That means you can turn the in-depth reports generated by Gemini into a conversational podcast featuring two AI "hosts." Since launching Audio Overviews within its AI note-takin...

Figure 2.3: Email Templates

After sending the email, the user is shown a successful or unsuccessful message using the modal, see Figure 2.4.(b). Figure 2.5 shows the email template in the user's inbox.

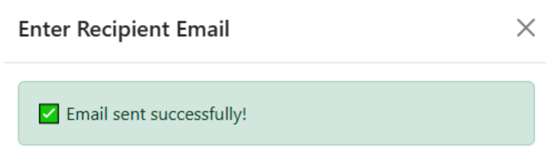


Enter Recipient Email

Recipient Email

iam.hri123@gmail.com

(a) Upon selecting Share from Figure 2.3, this email prompt is shown

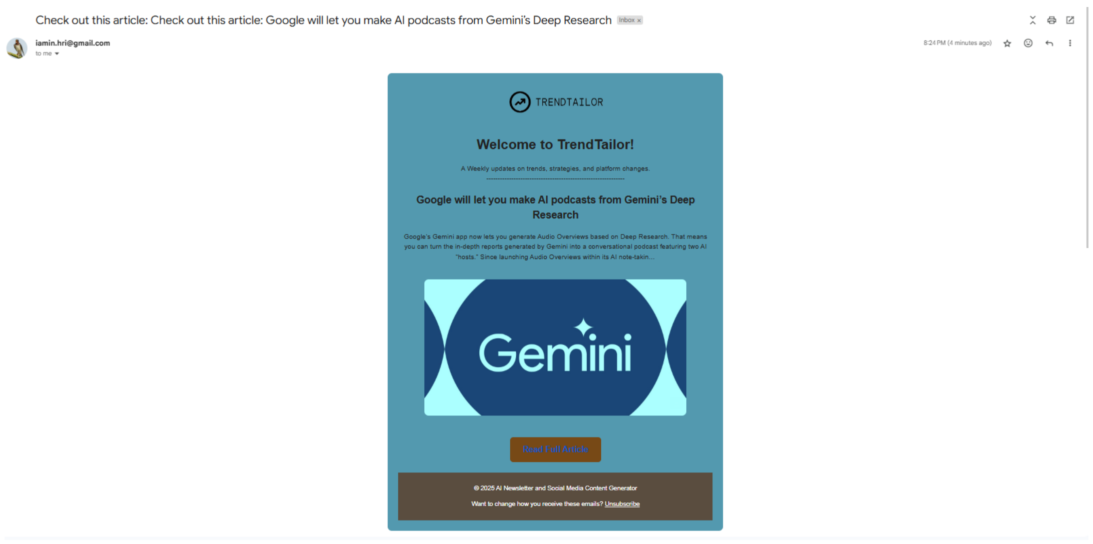


Enter Recipient Email

✓ Email sent successfully!

(b) Success Email Sent

Figure 2.4: Email Sent and Sharing UI



Check out this article: Check out this article: Google will let you make AI podcasts from Gemini's Deep Research

iamin.hri@gmail.com
to me

8:24 PM (4 minutes ago)

TRENDTAILOR

Welcome to TrendTailor!

A Weekly updates on trends, strategies, and platform changes.

Google will let you make AI podcasts from Gemini's Deep Research

Google's Gemini app now lets you generate Audio Overviews based on Deep Research. That means you can turn the in-depth reports generated by Gemini into a conversational podcast featuring two AI "hosts." Since launching Audio Overviews within its AI note-takin...

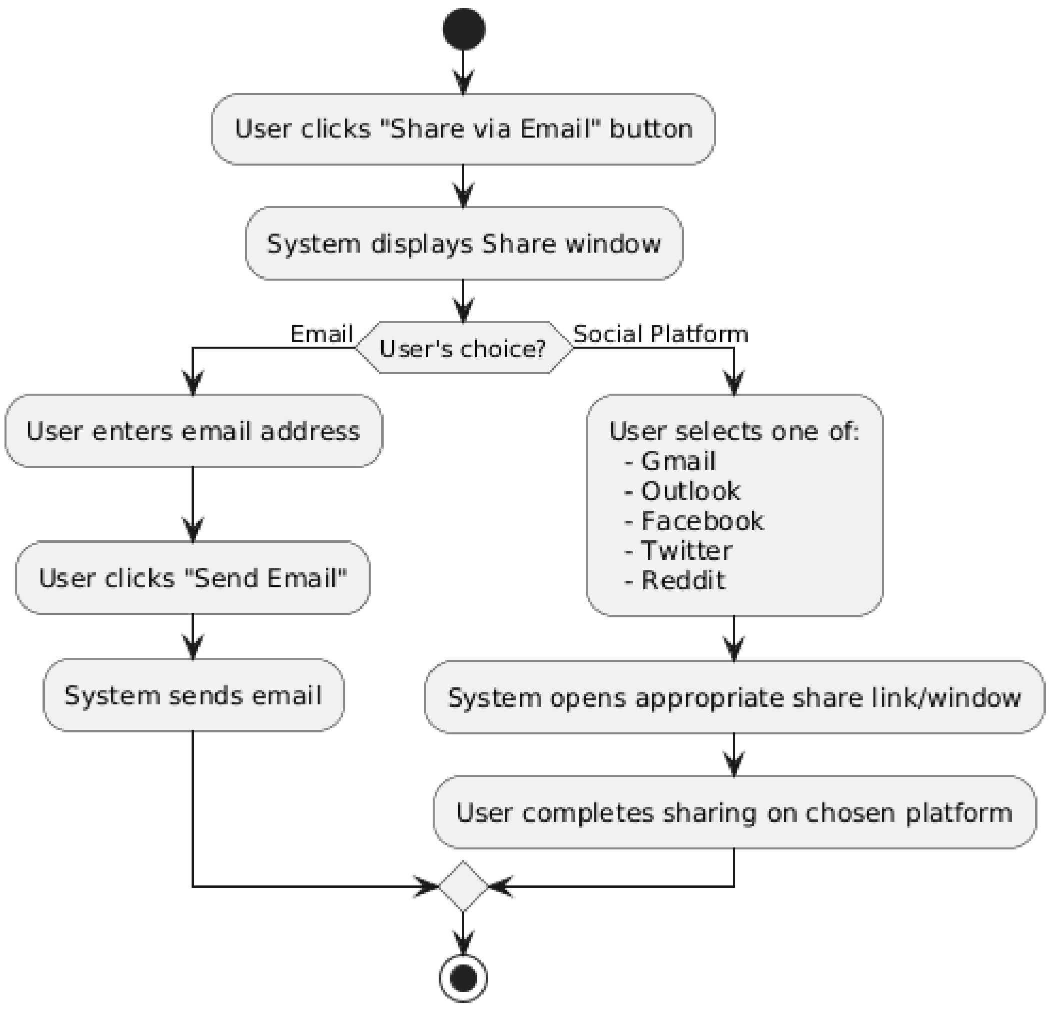
© 2025 AI Newsletter and Social Media Content Generator

Want to change how you receive these emails? Unsubscribe

Figure 2.5: Shared Using Email Templates

2.1.3 Activity Diagram

The following activity digram in Figure 2.6, depicts the back-end functionality for sharing content via email.



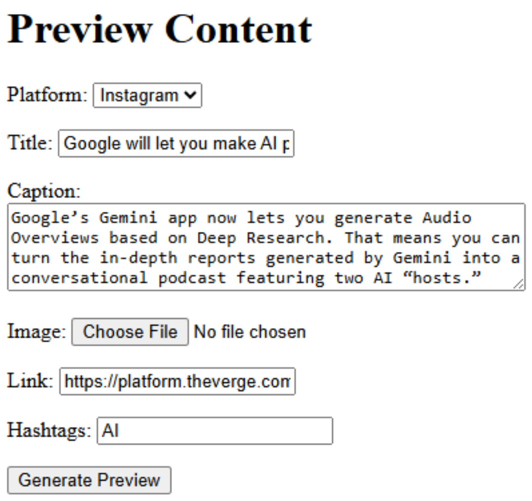
User clicks "Share via Email" button
System displays Share window
User's choice?
Email
Social Platform
User enters email address
User clicks "Send Email"
System sends email
User selects one of:
- Gmail
- Outlook
- Facebook
- Twitter
- Reddit
System opens appropriate share link/window
User completes sharing on chosen platform

Figure 2.6: Activity Diagram - Share Via Email

2.2 Sprint - 4

2.2.1 Scrum - 11

In this section, we focused on creating SNS templates and sharing functionalities. We developed a dynamic preview function that allows users to input content for SNS platforms like Instagram and Reddit and immediately see a live preview, see Figure 2.7. The following functionality was implemented using a Django view, which used prebuilt templates to capture details such as titles, captions, and images, then render a preview using platform-specific HTML, see Figure 2.7. Additionally, we implemented a file upload mechanism that saves user images to a designated media folder, ensuring that the images are properly displayed. One major challenge was handling local image uploads. Initially, we used Base64 encoding, which cluttered the HTML and resulted in inconsistent image displays. This issue was resolved by switching to Django's File System Storage to save uploaded files directly into the media/SNS content folder, retrieving a clean URL for each image, which provided a reliable solution for displaying user-uploaded images. Furthermore, we implemented the “read out loud” functionality, a newly added sub-task to our Trend Tailor platform. Although the individual functionality has been implemented, it is not yet integrated into our system. The final implementation will be shown in our final progress report.


Preview Content

Platform: Instagram

Title: Google will let you make AI p

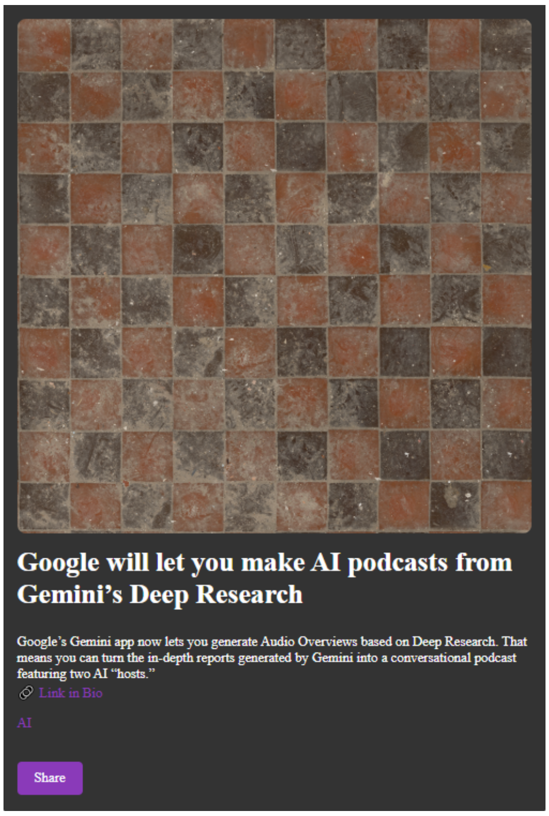
Caption: Google’s Gemini app now lets you generate Audio Overviews based on Deep Research. That means you can turn the in-depth reports generated by Gemini into a conversational podcast featuring two AI “hosts.”

Image: Choose File No file chosen

Link: https://platform.theverge.com

Hashtags: AI

(a) SNS Preview Generator


Google will let you make AI podcasts from Gemini’s Deep Research

Google’s Gemini app now lets you generate Audio Overviews based on Deep Research. That means you can turn the in-depth reports generated by Gemini into a conversational podcast featuring two AI “hosts.”

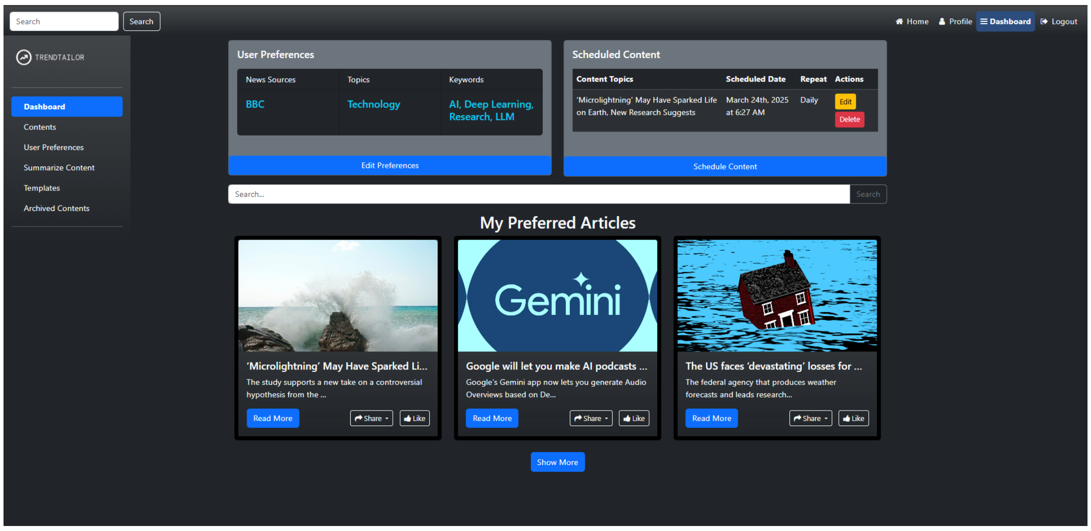
(b) Success Email Sent

Figure 2.7: SNS preview template for instagram

In Figure 2.7.(b), the share button automatically allows the user to share content directly to their respective SNS platform, similar to the functionality described in Section 2.1.2.

2.2.2 Scrum - 12

In this sprint, we focused on developing the content scheduling functionality. We designed the content scheduling user interface by adding a “Schedule” button to the dashboard, along with a “Scheduled Content” section to store scheduled items, as shown in Figure 2.8. When the ”Schedule” button is clicked, a Bootstrap modal appears, allowing users to choose the desired date, time, frequency, and article for scheduling, as illustrated in Figure 2.9. Once the information is submitted, it is displayed in the “Scheduled Content” section, showing the article, time, date, and frequency, along with edit and delete buttons, as seen in Figure 2.9(b). The edit button opens a modal that allows users to modify the time, date, and frequency, updating the scheduled content accordingly, as shown in Figure 2.10(a). Finally, the delete button allows users to remove the scheduled content if they no longer wish to post the article, as seen in Figure 2.10(b).



User Preferences

News Sources	Topics	Keywords
BBC	Technology	AI, Deep Learning, Research, LLM

Scheduled Content

Content Topics	Scheduled Date	Repeat	Actions
'Microlightning' May Have Sparked Life on Earth, New Research Suggests	March 24th, 2025 at 6:27 AM	Daily	Edit Delete

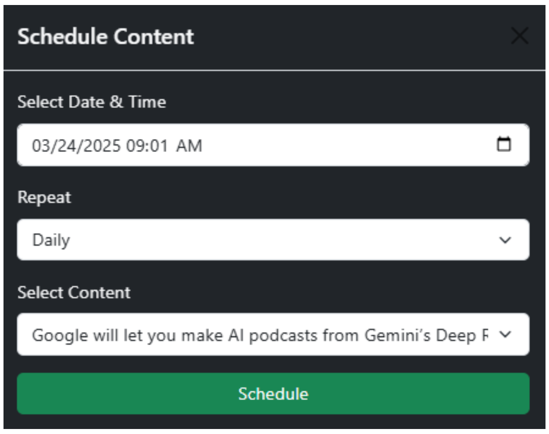
My Preferred Articles

'Microlightning' May Have Sparked Li...
The study supports a new take on a controversial hypothesis from the ...

Google will let you make AI podcasts ...
Google's Gemini app now lets you generate Audio Overviews based on De...

The US faces 'devastating' losses for ...
The federal agency that produces weather forecasts and leads research...

Figure 2.8: Dashboard with Schedule Module



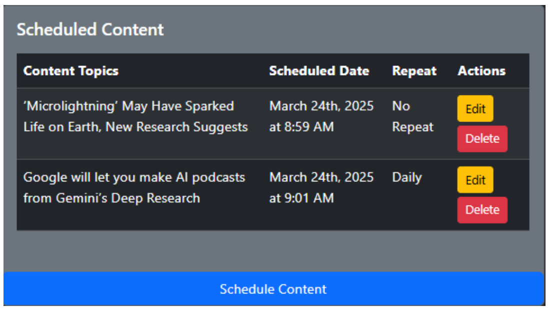
Schedule Content

Select Date & Time: 03/24/2025 09:01 AM

Repeat: Daily

Select Content: Google will let you make AI podcasts from Gemini's Deep F

(a) Scheduling Content Modal

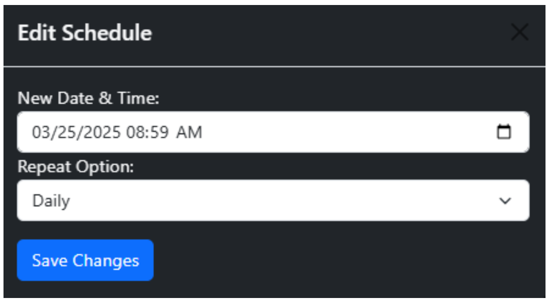


Scheduled Content

Content Topics	Scheduled Date	Repeat	Actions
'Microlightning' May Have Sparked Life on Earth, New Research Suggests	March 24th, 2025 at 8:59 AM	No Repeat	Edit Delete
Google will let you make AI podcasts from Gemini's Deep Research	March 24th, 2025 at 9:01 AM	Daily	Edit Delete

(b) New Schedule Added

Figure 2.9: Add and View Scheduling Content

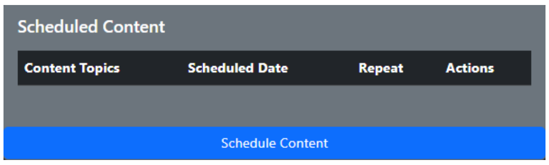


Edit Schedule

New Date & Time: 03/25/2025 08:59 AM

Repeat Option: Daily

(a) Edit Existing Schedule



Scheduled Content

Content Topics	Scheduled Date	Repeat	Actions

(b) Schedules deleted

Figure 2.10: Scheduling Content: Edit & Delete

2.2.3 Activity Diagram

The following diagram in Figure 2.10, depicts the back-end functionality of the Scheduling Component that is shown in section 2.2.2.

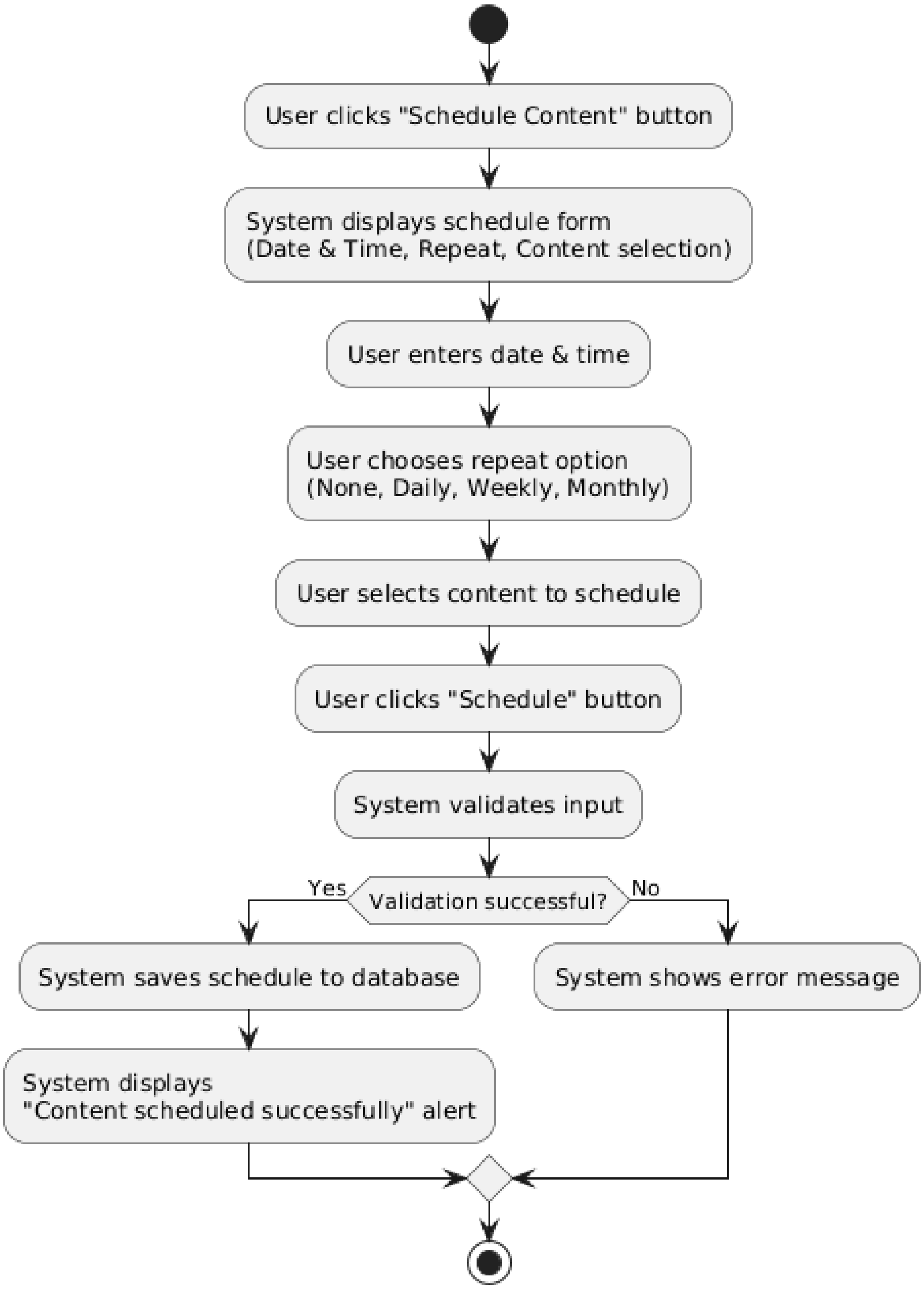
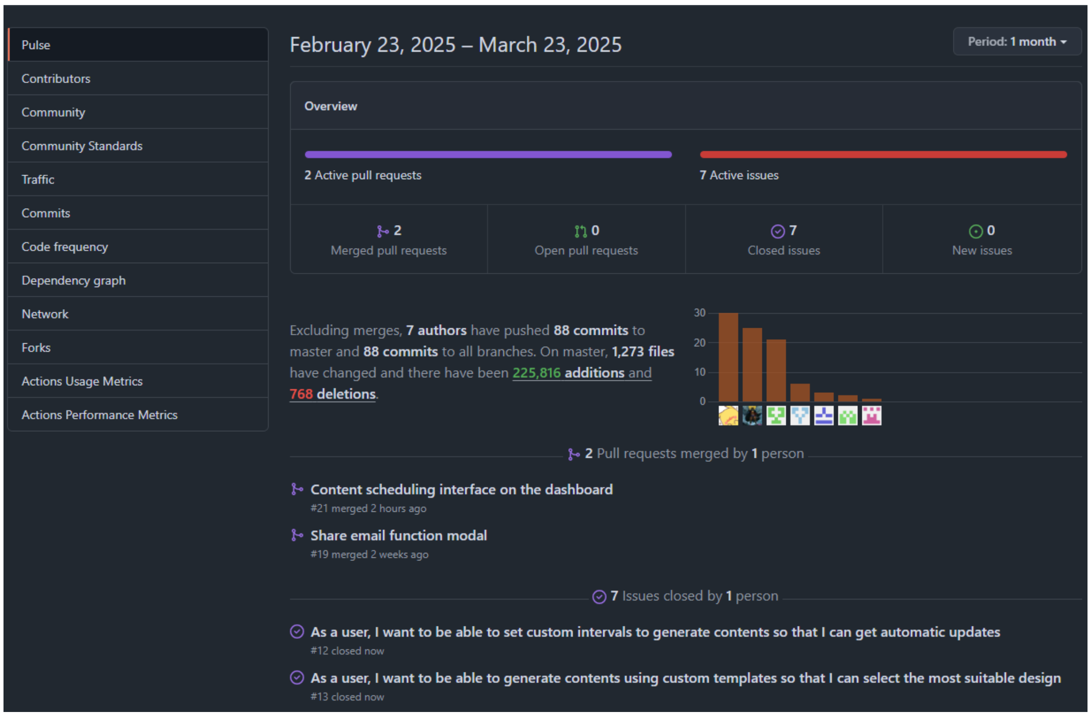

User clicks "Schedule Content" button
System displays schedule form (Date & Time, Repeat, Content selection)
User enters date & time
User chooses repeat option (None, Daily, Weekly, Monthly)
User selects content to schedule
User clicks "Schedule" button
System validates input
Validation successful?
Yes
No
System saves schedule to database
System shows error message
System displays "Content scheduled successfully" alert

Figure 2.11: Activity Diagram - Content Scheduling

2.3 GitHub

2.3.1 GitHub Progress

For project version control we utilize GitHub as mentioned in our previous report. The following GitHub link has all our commits and work. This section highlights key aspects of our work such as Pulse Overview, Commits Over Time, and Individual Contribution, as shown in Figures 2.12, 2.13, and 2.14, respectively.



Pulse
Contributors
Community
Community Standards
Traffic
Commits
Code frequency
Dependency graph
Network
Forks
Actions Usage Metrics
Actions Performance Metrics

February 23, 2025 – March 23, 2025

Period: 1 month

Overview

2 Active pull requests

7 Active issues

2 Merged pull requests

0 Open pull requests

7 Closed issues

0 New issues

Excluding merges, **7 authors** have pushed **88 commits** to master and **88 commits** to all branches. On master, **1,273 files** have changed and there have been **225,816 additions** and **768 deletions**.

2 Pull requests merged by **1** person

Content scheduling interface on the dashboard
#21 merged 2 hours ago

Share email function modal
#19 merged 2 weeks ago

7 Issues closed by **1** person

As a user, I want to be able to set custom intervals to generate contents so that I can get automatic updates
#12 closed now

As a user, I want to be able to generate contents using custom templates so that I can select the most suitable design
#13 closed now

Figure 2.12: Pulse Overview

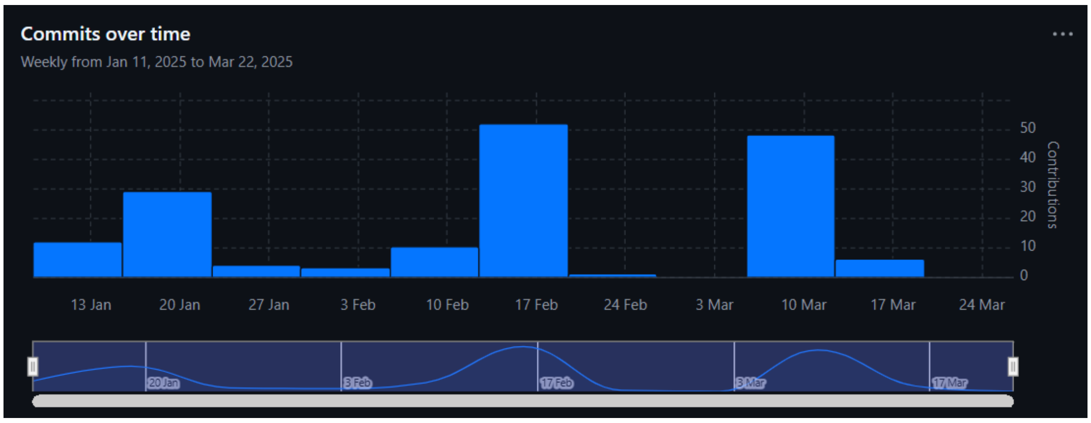
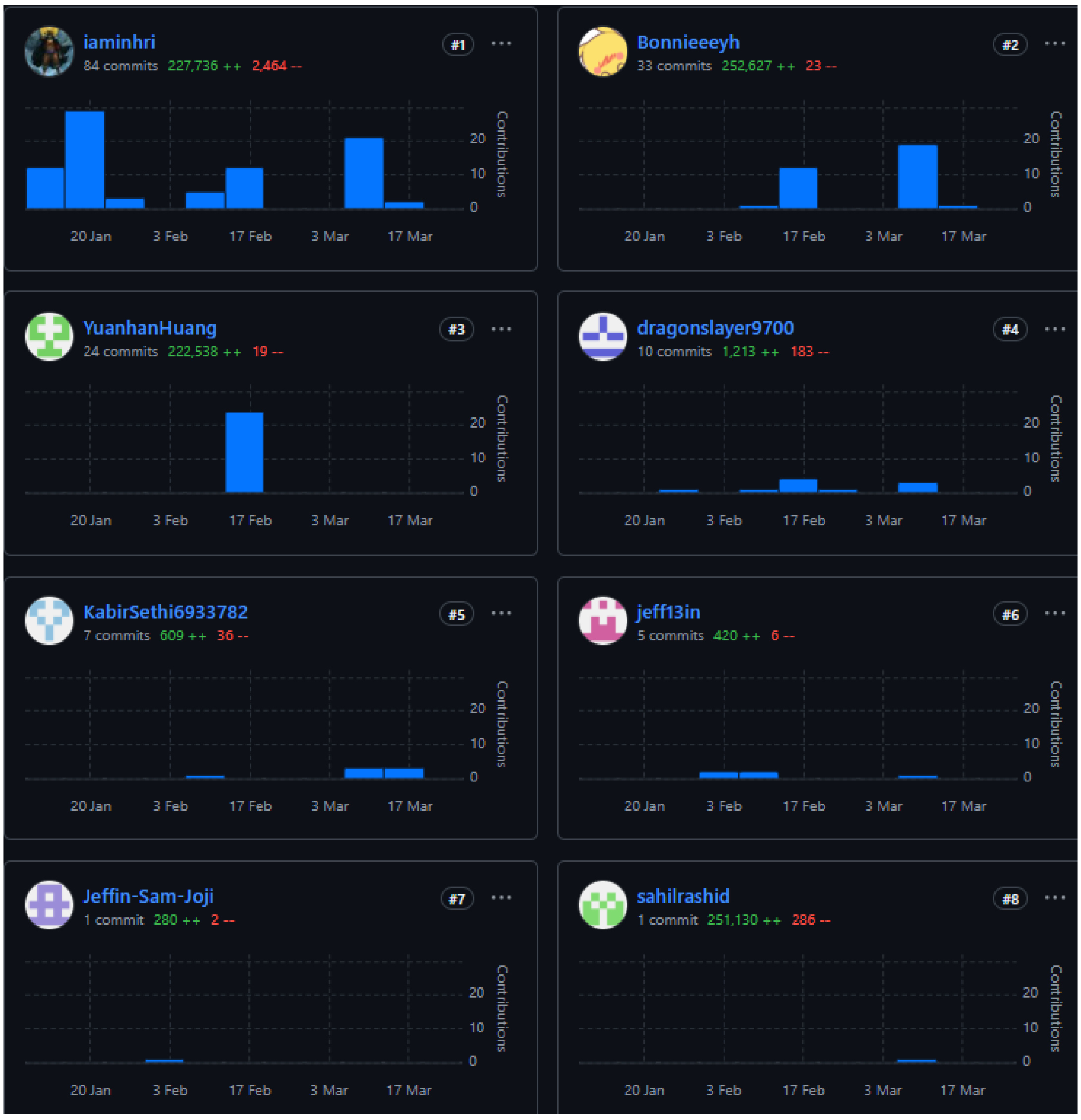


Figure 2.13: Commits over time


iaminhri #1
84 commits 227,736 ++ 2,464 --

Bonnieeeyh #2
33 commits 252,627 ++ 23 --

YuanhanHuang #3
24 commits 222,538 ++ 19 --

dragonslayer9700 #4
10 commits 1,213 ++ 183 --

KabirSethi6933782 #5
7 commits 609 ++ 36 --

jeff13in #6
5 commits 420 ++ 6 --

Jeffin-Sam-Joji #7
1 commit 280 ++ 2 --

sahilrashid #8
1 commit 251,130 ++ 286 --

Figure 2.14: Individual Contribution

3 Unit Testing

This is a complete list of all the tests we have implemented and successfully run throughout the project. These tests cover authentication, user preferences, database operations, form validation, and pagination. Before our final project report, we will do all the other unit testing.

3.1 Authentication & User Registration Tests

- `test_register_user` – Ensures that users can successfully register and their data is stored in the database.
- `test_login_user` – Confirms that users can log in with valid credentials.
- `test_invalid_login` – Ensures login fails with incorrect credentials.
- `test_authenticated_user_redirect` – Checks if logged-in users are redirected away from login/register pages.
- `test_unauthenticated_user_redirect` – Verifies that unauthenticated users are redirected to the login page when accessing restricted pages.
- `test_logout_user` – Confirms that users are successfully logged out.

3.2 Home Page Tests

- `test_homepage_access` – Ensures the home page loads successfully and contains the expected heading "Featured Articles".
- `test_homepage_shows_preferences` – Checks if the user's preferences (e.g., "Technology", "Science", "AI") are displayed on the home page.
- `test_homepage_pagination` – Ensures the pagination correctly loads multiple pages of articles.

3.3 Article Storage Tests

- `test_articles_saved_in_database` – Ensures that created articles are correctly stored and retrieved from the database.
- `test_articles_show_on_homepage` – Checks if stored articles appear on the homepage.

3.4 News Database Filtering Tests

- `test_check_articles` – Tests if articles stored in the database are correctly retrieved based on user topics and keywords.

- `test_database_article_storage` – Ensures that articles are saved correctly in the database.

- `test_article_model_fields` – Confirms that article models contain all required fields (title, description, URL, image, etc.).

3.5 Form Validation Tests

- `test_register_form_validation` – Ensures that registration fails when required fields are missing.

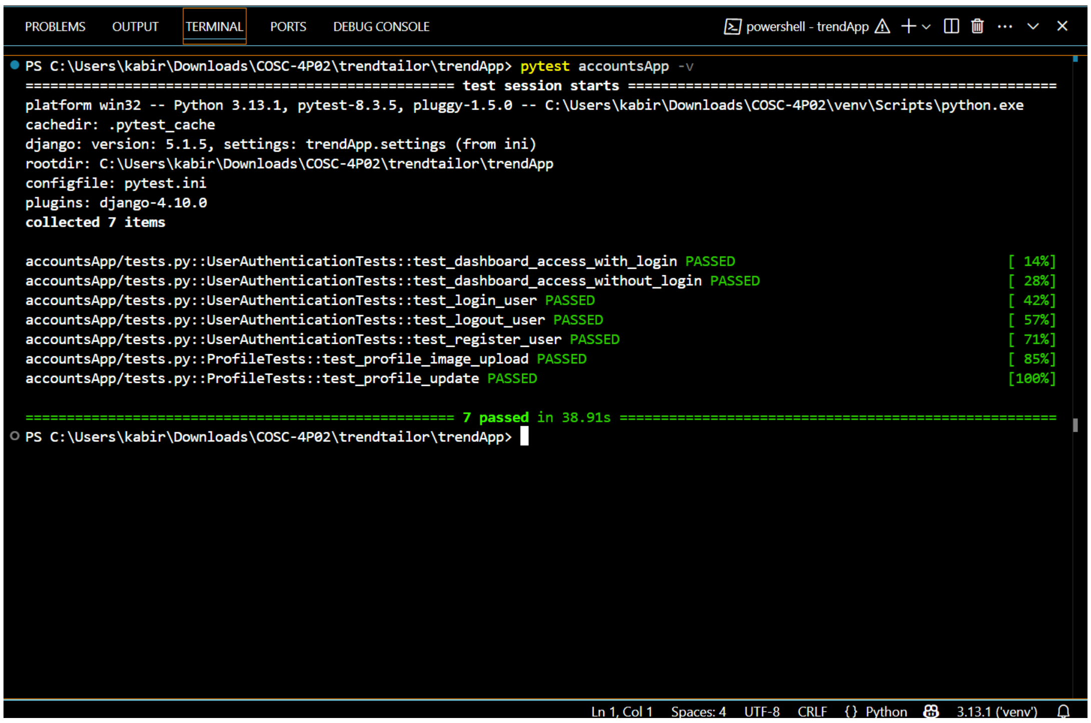
- `test_login_form_validation` – Ensures that login fails with empty fields.

- `test_preference_form_validation` – Verifies that user preference forms handle input errors correctly.

3.6 Pagination Content Display Tests

- `test_pagination` – Ensures that articles are paginated correctly and displayed in the Preferred Articles section.

- `test_pagination` – Confirms that the pagination displays the correct number of articles per page.

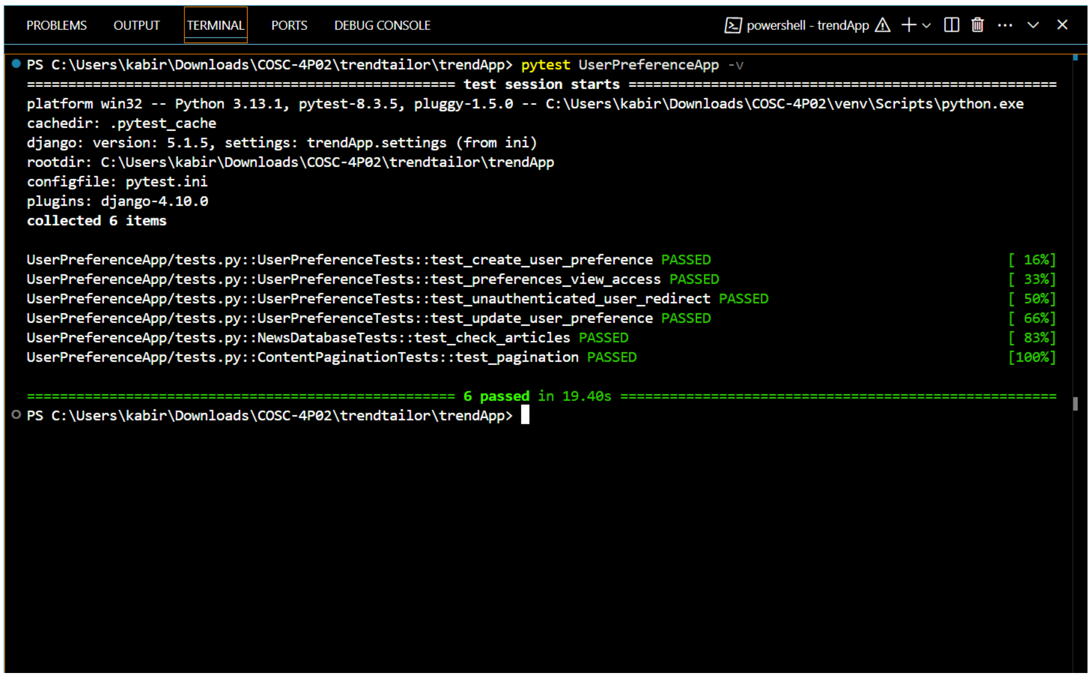


```
PS C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp> pytest accountsApp -v
==================================================== test session starts ====================================================
platform win32 -- Python 3.13.1, pytest-8.3.5, pluggy-1.5.0 -- C:\Users\kabir\Downloads\COSC-4P02\venv\Scripts\python.exe
cachedir: .pytest_cache
django: version: 5.1.5, settings: trendApp.settings (from ini)
rootdir: C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp
configfile: pytest.ini
plugins: django-4.10.0
collected 7 items

accountsApp/tests.py::UserAuthenticationTests::test_dashboard_access_with_login PASSED                                [ 14%]
accountsApp/tests.py::UserAuthenticationTests::test_dashboard_access_without_login PASSED                             [ 28%]
accountsApp/tests.py::UserAuthenticationTests::test_login_user PASSED                                                 [ 42%]
accountsApp/tests.py::UserAuthenticationTests::test_logout_user PASSED                                                [ 57%]
accountsApp/tests.py::UserAuthenticationTests::test_register_user PASSED                                              [ 71%]
accountsApp/tests.py::ProfileTests::test_profile_image_upload PASSED                                                  [ 85%]
accountsApp/tests.py::ProfileTests::test_profile_update PASSED                                                        [100%]

==================================================== 7 passed in 38.91s ====================================================
PS C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp>
```

Figure 3.1: Test 1

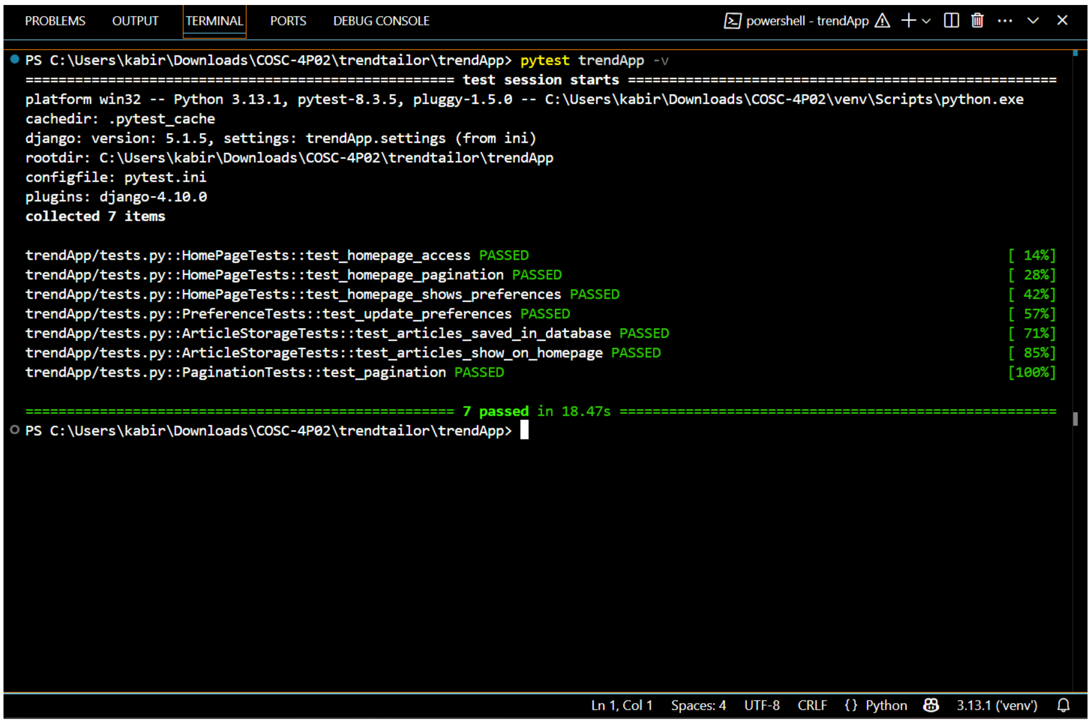


```
PS C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp> pytest UserPreferenceApp -v
=================================================== test session starts ===================================================
platform win32 -- Python 3.13.1, pytest-8.3.5, pluggy-1.5.0 -- C:\Users\kabir\Downloads\COSC-4P02\venv\Scripts\python.exe
cachedir: .pytest_cache
django: version: 5.1.5, settings: trendApp.settings (from ini)
rootdir: C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp
configfile: pytest.ini
plugins: django-4.10.0
collected 6 items

UserPreferenceApp/tests.py::UserPreferenceTests::test_create_user_preference PASSED                                 [ 16%]
UserPreferenceApp/tests.py::UserPreferenceTests::test_preferences_view_access PASSED                                [ 33%]
UserPreferenceApp/tests.py::UserPreferenceTests::test_unauthenticated_user_redirect PASSED                          [ 50%]
UserPreferenceApp/tests.py::UserPreferenceTests::test_update_user_preference PASSED                                 [ 66%]
UserPreferenceApp/tests.py::NewsDatabaseTests::test_check_articles PASSED                                           [ 83%]
UserPreferenceApp/tests.py::ContentPaginationTests::test_pagination PASSED                                          [100%]

=================================================== 6 passed in 19.40s ====================================================
PS C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp>
```

Figure 3.2: Test 2

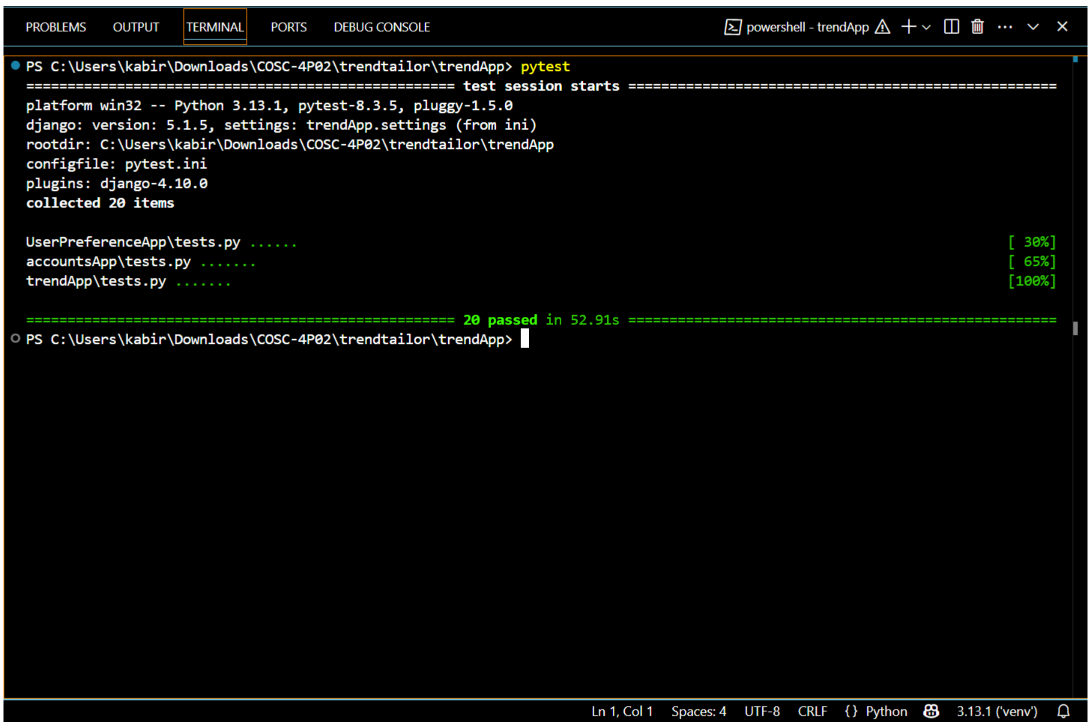


```
PS C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp> pytest trendApp -v
=================================================== test session starts ===================================================
platform win32 -- Python 3.13.1, pytest-8.3.5, pluggy-1.5.0 -- C:\Users\kabir\Downloads\COSC-4P02\venv\Scripts\python.exe
cachedir: .pytest_cache
django: version: 5.1.5, settings: trendApp.settings (from ini)
rootdir: C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp
configfile: pytest.ini
plugins: django-4.10.0
collected 7 items

trendApp/tests.py::HomePageTests::test_homepage_access PASSED                                                       [ 14%]
trendApp/tests.py::HomePageTests::test_homepage_pagination PASSED                                                   [ 28%]
trendApp/tests.py::HomePageTests::test_homepage_shows_preferences PASSED                                            [ 42%]
trendApp/tests.py::PreferenceTests::test_update_preferences PASSED                                                  [ 57%]
trendApp/tests.py::ArticleStorageTests::test_articles_saved_in_database PASSED                                      [ 71%]
trendApp/tests.py::ArticleStorageTests::test_articles_show_on_homepage PASSED                                       [ 85%]
trendApp/tests.py::PaginationTests::test_pagination PASSED                                                          [100%]

=================================================== 7 passed in 18.47s ====================================================
PS C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp>
```

Figure 3.3: Test 3

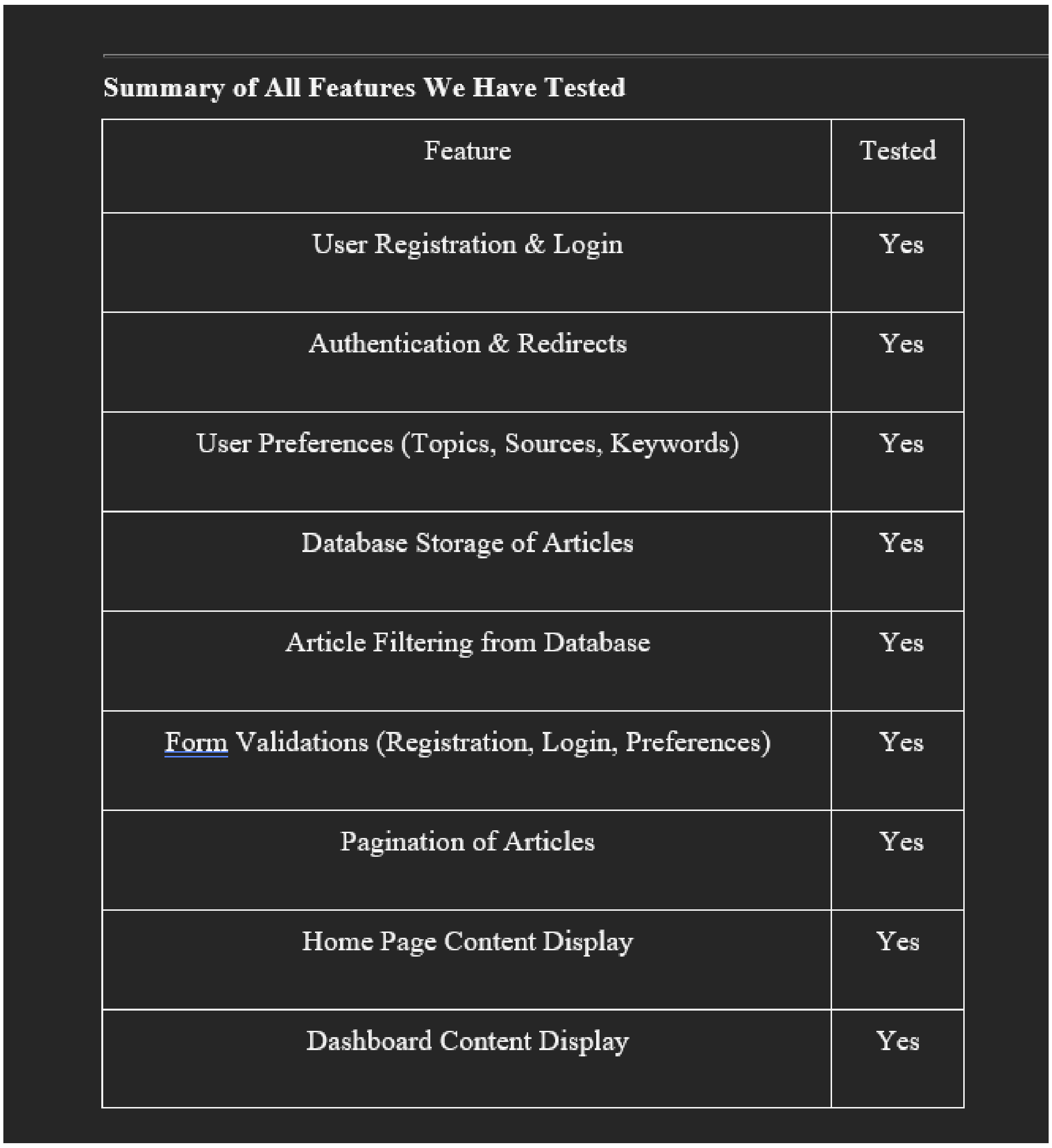


```
PS C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp> pytest
==================================================== test session starts ====================================================
platform win32 -- Python 3.13.1, pytest-8.3.5, pluggy-1.5.0
django: version: 5.1.5, settings: trendApp.settings (from ini)
rootdir: C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp
configfile: pytest.ini
plugins: django-4.10.0
collected 20 items

UserPreferenceApp\tests.py ......                                                                                      [ 30%]
accountsApp\tests.py .......                                                                                           [ 65%]
trendApp\tests.py .......                                                                                              [100%]

==================================================== 20 passed in 52.91s ====================================================
PS C:\Users\kabir\Downloads\COSC-4P02\trendtailor\trendApp>
```

Figure 3.4: Test 4



Summary of All Features We Have Tested

Feature	Tested
User Registration & Login	Yes
Authentication & Redirects	Yes
User Preferences (Topics, Sources, Keywords)	Yes
Database Storage of Articles	Yes
Article Filtering from Database	Yes
Form Validations (Registration, Login, Preferences)	Yes
Pagination of Articles	Yes
Home Page Content Display	Yes
Dashboard Content Display	Yes

Figure 3.5: Summary of Unit Testing

4 Sprints & Meetings

4.1 Weekly Scrum Meetings

In these sprints, our focus shifted towards refining and expanding the existing foundation of the project. Our weekly Scrum meetings, held every Friday and led by Arifuzzaman (Scrum Master), served as structured checkpoints where we provided updates on completed tasks, discussed ongoing work, and addressed any challenges faced by the team. We also ensured that any team members needing support received the necessary accommodations, maintaining a smooth and efficient workflow.

Beyond our scheduled meetings, we frequently held unscheduled one-on-one sessions on Discord to provide targeted assistance. These impromptu discussions allowed for quicker issue resolution, enabling team members to stay on track and maintain productivity. Alongside that, we held Sprint Retrospective meetings to reflect on our progress, identify areas for improvement, and optimize our workflow.

Additionally, we dedicated time to evaluating our adherence to the Agile-Scrum methodology, ensuring we remained aligned with best practices. We encouraged input from all team members to refine our processes. We also strategized on incorporating key elements such as diagrams, burndown charts, and formal testing plans into our project report to enhance documentation and clarity.

The project is already shaping up well. With continuous feedback, collaboration, and a commitment to delivering a high-quality solution, we are making steady progress toward building a well-structured and efficient web application.

4.2 Sprint 3

4.2.1 Scrum - 5

In this sprint, we focused on implementing article summarization using an LLM while following Agile Scrum methodologies to ensure efficiency and adaptability. The primary goal was to integrate a summarization LLM API into the backend, enabling accurate and concise text summarization. We tested the summarization with sample inputs to validate performance and developed a user-friendly UI to enhance accessibility. Rigorous testing ensured reliability and seamless integration within the platform.

Task	Associated User Stories	Priority
Implement article summarization using LLM by prompt	Scrum-5	High
Integrate summarization LLM API into the backend	Scrum-5	High
Test summarization for samples	Scrum-5	Medium
Develop UI for summarization	Scrum-5	Medium

4.2.2 Scrum - 7

Additionally, we prioritized enhancing user control over email management by allowing users to send newsletters via Gmail or Outlook, with options to easily add or remove recipients from the list. To expand the newsletter's reach, we integrated a feature for seamless sharing on Facebook and Reddit. The backend logic was developed to ensure efficient email distribution and smooth integration with these social platforms. These high-priority tasks were executed with a focus on reliability, usability, and seamless automation.

Task	Associated User Stories	Priority
Create a form to send the newsletter content using a list of emails	Scrum-7	High
Create options for users to modify emails, such as add or remove	Scrum-7	High
Create an option to share it with social media	Scrum-7	High
Develop backend logic to send emails and to share it with social media	Scrum-7	High

4.3 Sprint 4

4.3.1 Scrum - 11

In this sprint, we focused on understanding the formatting needs of different SNS and created content templates for each. We added a preview function so users can check their content before sharing, and made it easy to share across SNS. To improve accessibility, we also introduced a ”read out loud” feature. These updates aimed to make content creation, sharing, and interaction more user-friendly.

Task	Associated User Stories	Priority
Investigate SNS formatting requirements	Scrum-11	Low
Create content templates for different SNS	Scrum-11	Medium
Implement preview function for contents	Scrum-11	Medium
Create content sharing functionality for SNS using templates	Scrum-11	Medium
Implement read out loud functionality (newly added tasks)	Scrum-11	Medium

4.3.2 Scrum - 12

During this sprint, we focused on developing a scheduling interface for content generation. We implemented a user-friendly UI to allow users to easily create, edit, and delete scheduled intervals. On the backend, we developed logic to store and manage these user-defined intervals. Additionally, automation features were introduced to ensure content is generated at the specified times. We also conducted thorough testing to ensure the scheduling and content generation processes function as expected.

Task	Associated User Stories	Priority
Design the scheduling interface for content generation settings	Scrum-12	High
Implement UI to display, edit, and delete scheduled intervals	Scrum-12	High
Implement backend logic to store and manage user-defined scheduling intervals	Scrum-12	High
Develop automation logic to generate content at the specified intervals	Scrum-12	Medium
Conduct testing to ensure proper scheduling and content generation	Scrum-12	Medium

4.4 Sprint Progress

In this section, we highlight the progress of our work. By the time of our first progress report, we had completed two sprints. By the second report, we had completed two additional sprints, maintaining a consistent velocity and progress throughout. In our final report, we plan to complete Sprint 5, along with all remaining testing and bug fixing.

4.5 Sprint Burndown Charts

This section highlights the Sprint Burndown Chart, a visual tool used in Agile and Scrum project management to track the progress of a sprint.

The Sprint Burndown Chart for Scrum Sprint 3 shows delayed progress, with no issues completed until mid-sprint. This delay was caused by midterms that occurred during that period. However, we completed the sprint on time, by March 18. The updates for Sprint 4 in JIRA were delayed, but the sprint started around March 10.

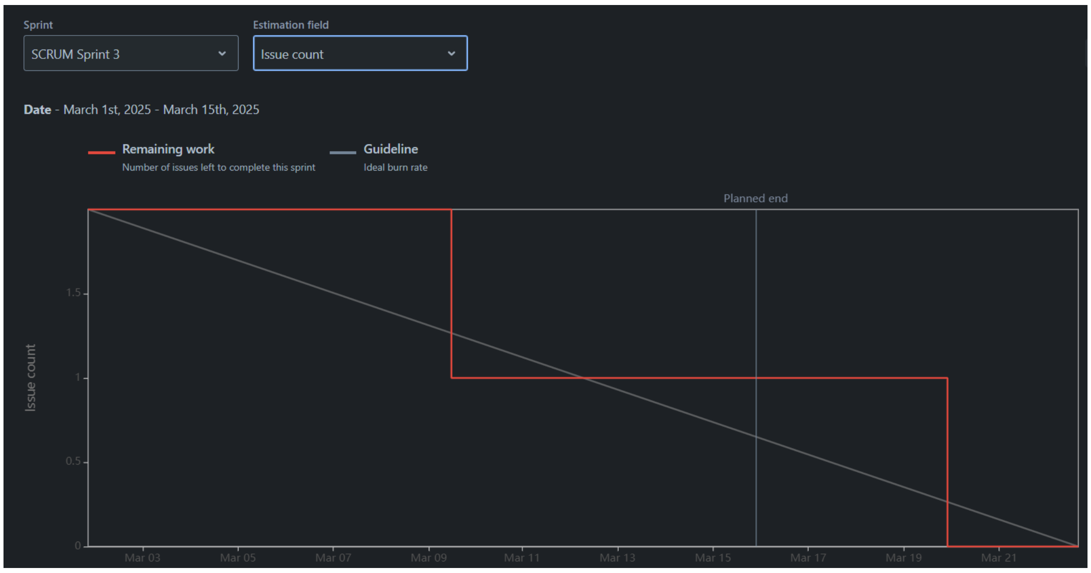


Figure 4.1: Sprint Burndown Chart - Sprint 3

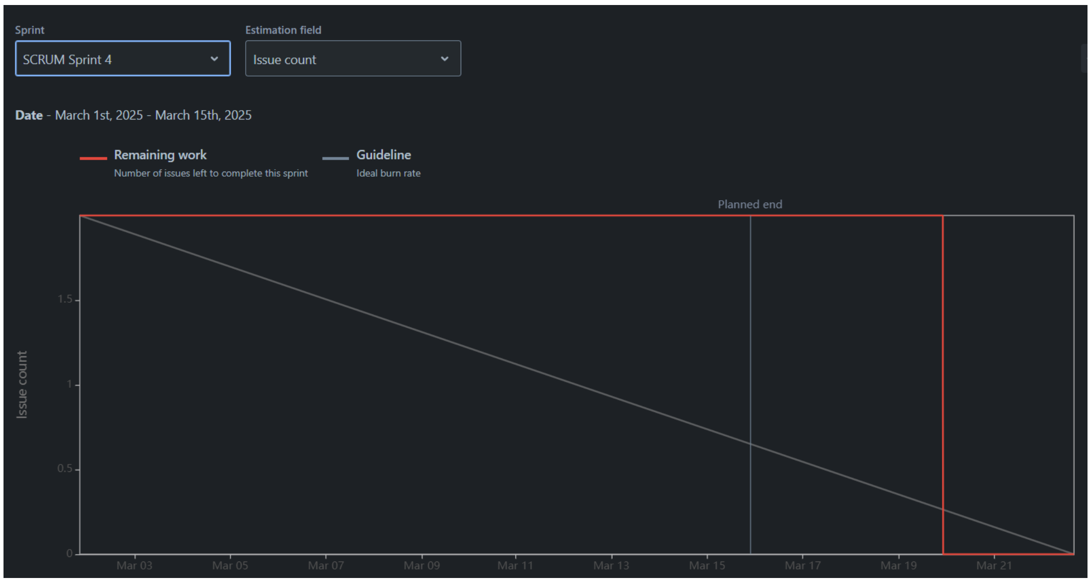


Figure 4.2: Sprint Burndown Chart - Sprint 4

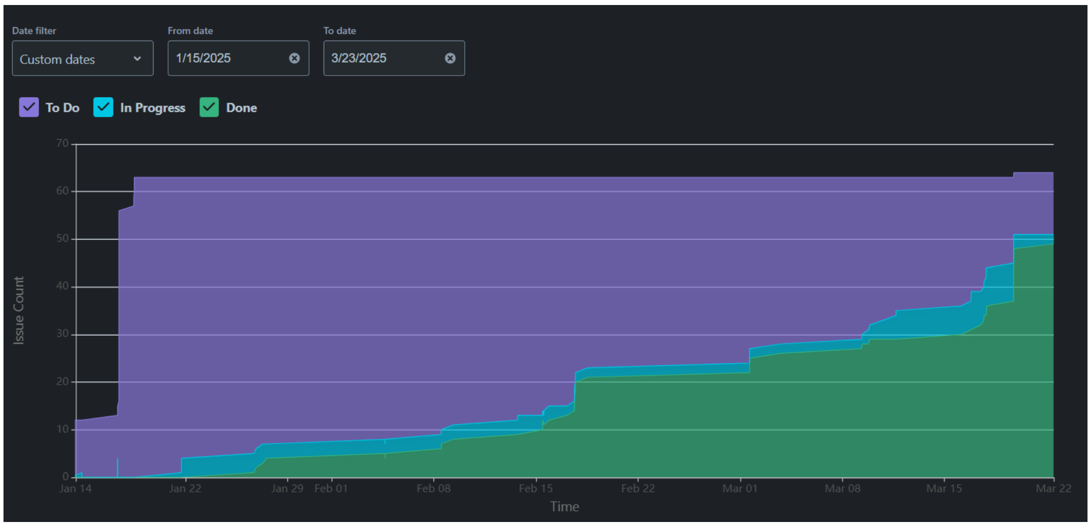

Date filter: Custom dates
From date: 1/15/2025
To date: 3/23/2025

Figure 4.3: Cumulative Flow Diagram

5 Challenges and Next Steps

As the project progressed, we encountered new challenges that required adaptability and problem-solving. One of the major issues was optimizing content sharing across multiple social media platforms. Each platform had its own formatting rules, requiring adjustments to metadata, image handling, and text formatting to ensure consistent and accurate sharing.

Handling local image uploads also presented difficulties. Initially, we used Base64 encoding, but it cluttered the HTML and caused inconsistent image display. To resolve this, we switched to Django's File System Storage, storing uploaded images in a dedicated media folder and retrieving clean URLs for reliable display.

Another challenge was implementing the scheduling functionality. Handling different time zones and ensuring content was published at the correct intervals required precise backend logic. We also had to account for editing and deleting scheduled posts dynamically without disrupting the automation process.

Additionally, integrating the "read out loud" feature for accessibility proved complex. Ensuring smooth text-to-speech conversion while maintaining compatibility across different devices and browsers required multiple iterations and testing.

Despite these challenges, we remained committed to refining our approach, collaborating effectively, and implementing solutions that improved the project's functionality and user experience.

For the next phase, we will conduct a short sprint to finalize key features and wrapping up the project. Our focus will be on implementing multilingual support, enabling content archiving for future reference, and integrating our API into various services. These additions will enhance accessibility, improve content management, and expand the project's functionality across different platforms.

6 Team Contributions

Team Member	Contributions
Arifuzzaman (Scrum Master)	Implemented article summarization using LLM by prompt, tested summarization for sample inputs to ensure accuracy, and developed a clean, user-friendly UI for summarization. Created content templates for various SNS to standardize formatting and improve presentation. Managed team communications, scheduled meetings, ensured adherence to Agile-Scrum methodology. Wrote and refined the progress report.
Hridoy Rahman	Delegated tasks. Wrote Progress Report 2. Modified and integrated LLM summarization functionality with the Django backend. Merged and refined all sharing functionalities. Redesigned the Dashboard UI. Integrated scheduling functionalities with the dashboard and resolved conflicts. Fixed bugs and resolved GitHub conflicts. Ensured seamless integration of functionalities by merging scripts for sharing and scheduling features. Improved code modularity by refactoring and enhancing the existing codebase.
Jeffin Sam Joji	Worked on both frontend and backend to implement share buttons for various social media platforms using JavaScript and CSS. Also created activity diagrams for social media sharing and content scheduling. Contributed to writing the progress report.
Yuanhan Huang	Developed a dynamic preview function for SNS platforms like Instagram and Reddit, enabling live content previews with prebuilt templates to capture titles, captions, and images. Implemented a file upload mechanism to store and display user images correctly. Contributed to writing the progress report.
Kabir Sethi (Product Owner)	Wrote test cases for key functionalities, covering unit, system, and integration testing to ensure everything works as expected. Contributed to writing the progress report.
Sahil Rashid	Implemented the email-sharing feature with a Bootstrap modal, allowing users to share articles via Gmail and Outlook. Designed and developed the content scheduling interface and functionality, enabling users to set, edit, and manage scheduled posts. Ensured automatic content sharing based on frequency settings. Contributed to report writing.
Jayant Saini	Researched the implementation of new features such as 'Read Out Loud,' and successfully implemented the 'Read Out Loud' functionality for accessibility. Investigated SNS formatting requirements, assisted with LLM Summarization, and created sequence and activity diagrams to illustrate the workflow across various project features. Contributed to writing the progress report.