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

# AI-Powered Newsletter and Social Media Content Generator

# Progress Report 1

**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:** February 23, 2025

# Contents

# 1 Overview

Over the past few weeks, our team has made significant progress in developing our project, following Agile methodology to ensure iterative improvements and continuous feedback. We have successfully completed our first two sprints, implementing key functionalities such as Login, Registration, Dashboard, Content Aggregation, and User Preferences. Our goal is to create a robust, interactive, and user-friendly platform that prioritizes ease of use and accessibility.

In the initial stages, we focused on setting up the development environment, configuring essential frameworks and dependencies, and establishing the foundational components. This phase ensured that we had a solid base to build upon, allowing for seamless integration of new features.

This report outlines our progress across four key areas. The design section provides insights into our front-end components. The implementation section discusses the tools and technologies chosen, along with justifications for their use. The sprints section highlights our sprint cycles, including scrum meetings, iterative testing, and feature rollouts. Finally, the challenges and next steps section addresses obstacles encountered, lessons learned, and our strategy moving forward.

By adhering to Agile principles, we are ensuring a flexible and adaptable development process that allows us to respond effectively to user needs and project requirements. Moving forward, we will continue refining our existing features while expanding the functionality of our platform to enhance user experience.

# 2 Design

## 2.1 UI Mock-up

When we started the project, we began by creating a Figma mockup to visualize the design and streamline the development process. This helped us plan the layout, user flow, and overall structure, making implementation more efficient.

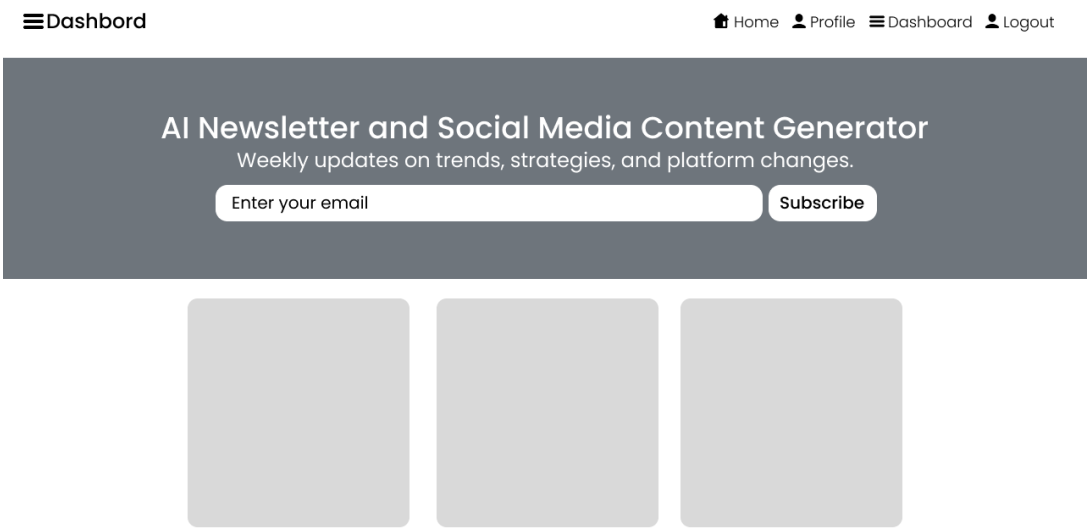


Figure 2.1: Home page

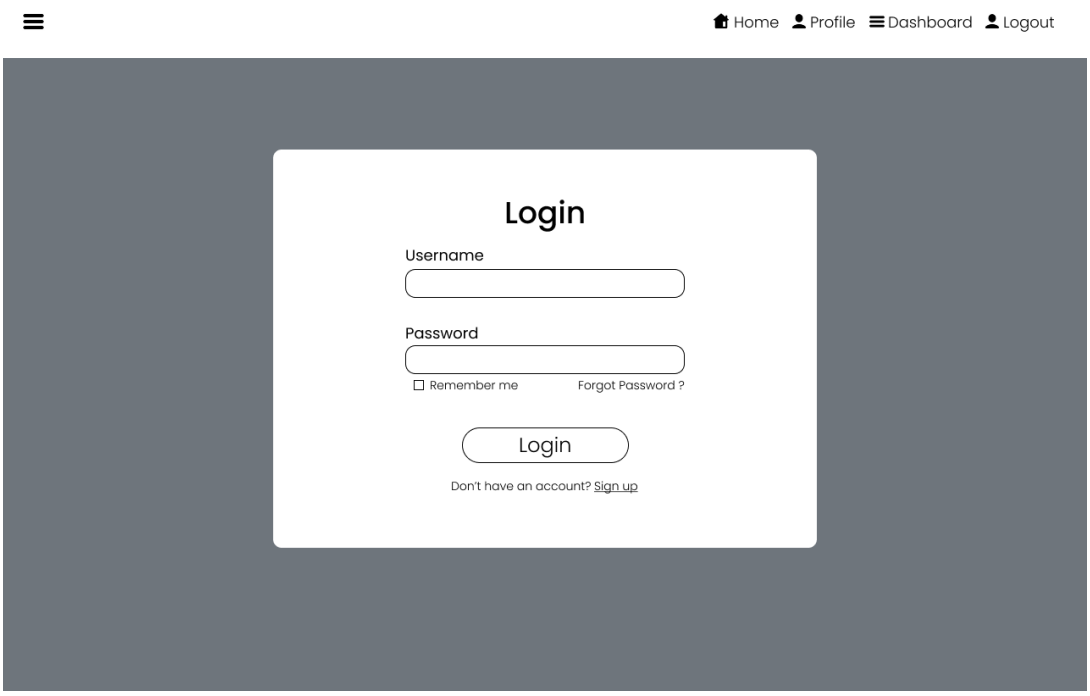


Figure 2.2: Login page

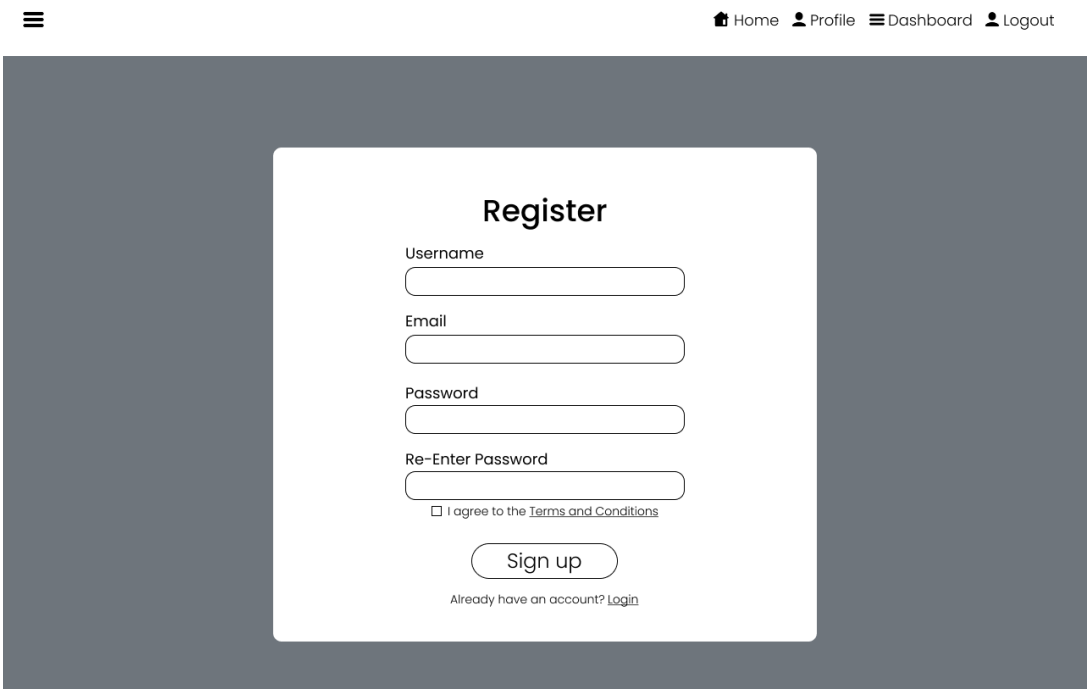


Figure 2.3: Registration page

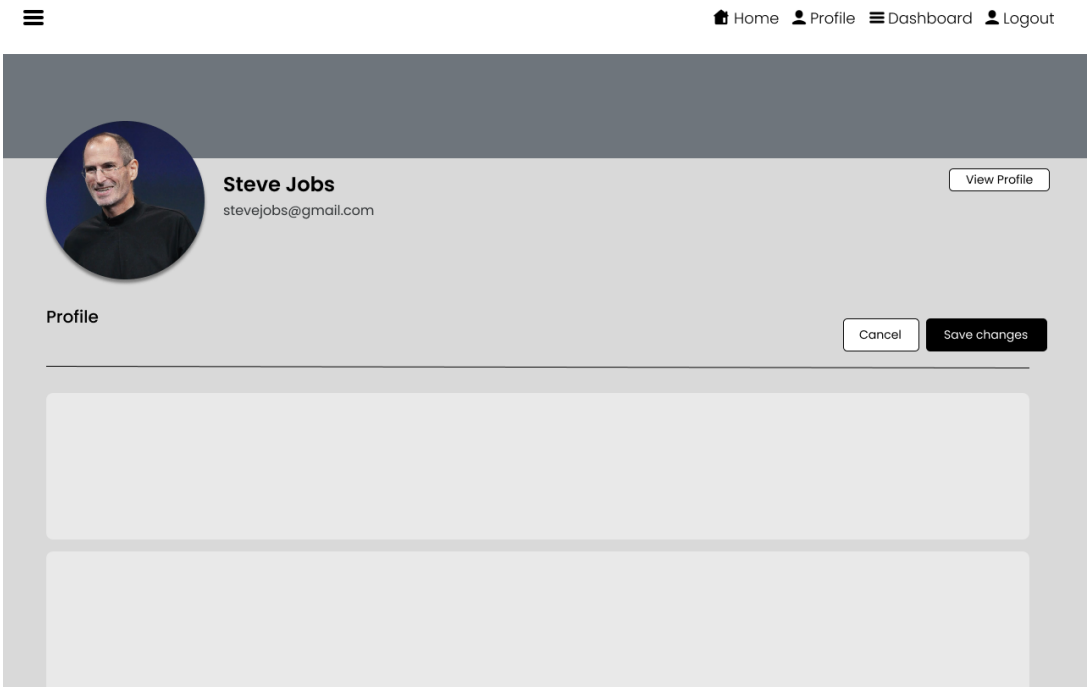


Figure 2.4: Profile page

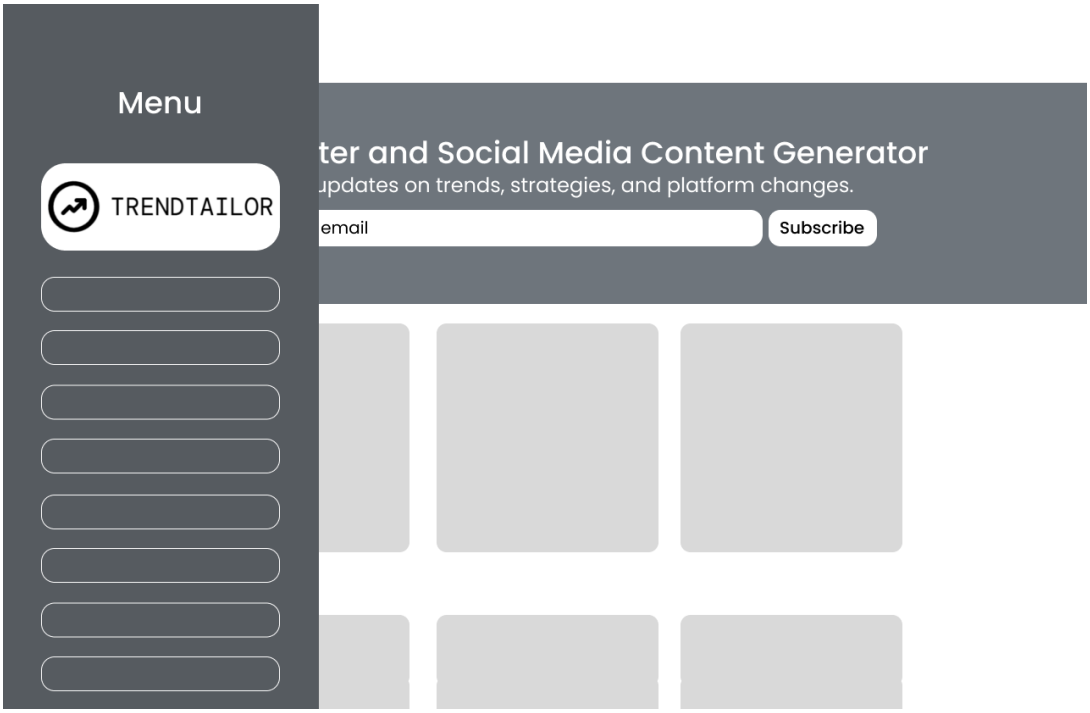


Figure 2.5: Sidebar/Menu section

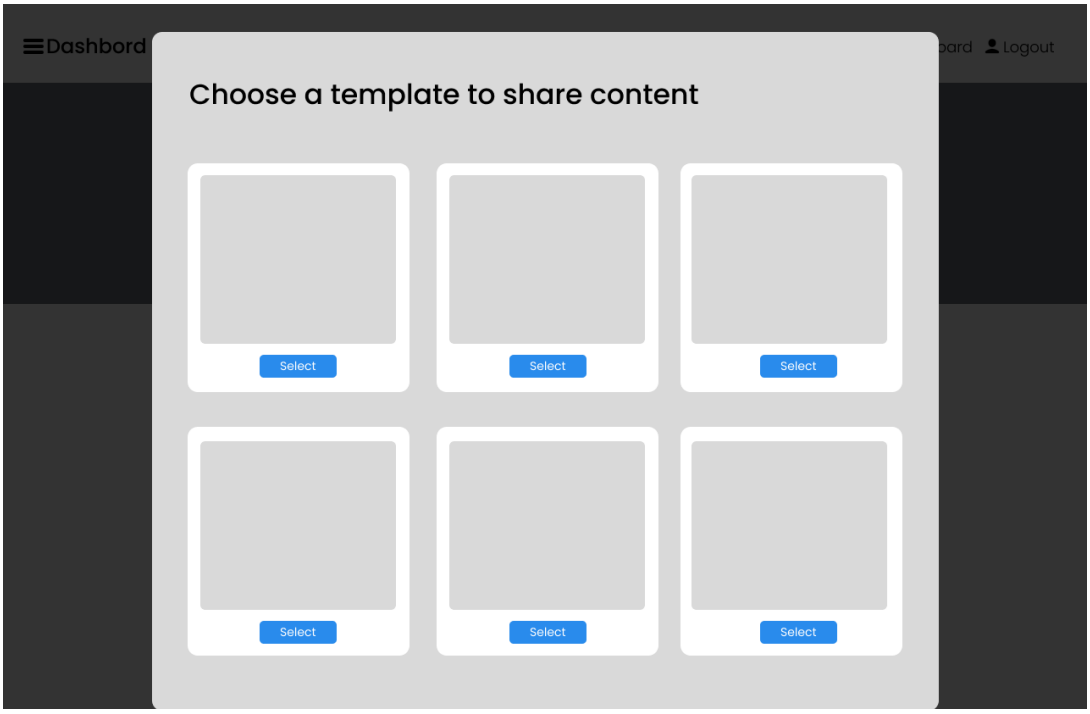


Figure 2.6: Templates selection page

# 3 Implementation

## 3.1 HTML & Bootstrap

In the world of web development, time is precious, and efficiency is key. To streamline our front-end development, we are utilizing HTML and Bootstrap, two essential technologies that work together to create a well-structured, visually appealing, and responsive platform.

HTML, often regarded as the foundation of the web, provides the structural framework for our website. On the styling and layout side, Bootstrap serves as our go-to front-end framework, which is free and open source, designed for building responsive websites and web applications. Bootstrap provides ready-to-use components, such as navigation bars, forms, tables, responsive grids etc ensuring a uniform and polished appearance across the site. Additionally, since Bootstrap is lightweight and optimized, it enhances the overall performance and maintainability of our project. This allows us to save development time while maintaining consistency and ease of use.

By using the combination of HTML and Bootstrap we ensure a modern, consistent, and future-proof web experience for our users.

## 3.2 Python & Django

We chose Python and Django as the core technologies for our backend due to their simplicity, efficiency, and flexibility. Python's readability makes it ideal for rapid development, while Django, a high-level web framework, is perfect for building highly customizable applications, including social media websites. It comes with built-in features like authentication, database management, and URL routing, allowing us to focus on key functionality without reinventing the wheel.

Django follows the Model-View-Template (MVT) architecture, ensuring our code is organized and easy to maintain. The models define the data, views handle the logic, and templates render the content. Additionally, Django provides a user-friendly admin interface for managing data and ensuring security. With Django's robust features, we can build a secure, scalable, and maintainable web application with minimal manual effort. Therefore, Python and Django together work like a charm and provide a powerful and efficient foundation for building and maintaining our web application.

## 3.3 Database Management

For our project, we selected SQLite as our primary database, definitely due to its simplicity and seamless integration with Django. SQLite is a lightweight, file-based database that comes bundled with Django by default, making it incredibly easy to set up and get started. This was an ideal choice for our needs, as we were looking for a quick and efficient way to store and retrieve data during the initial stages of development. SQLite's minimal configuration requirements allowed us to focus on building features rather than spending time on complex database setup, which was a huge benefit in the early stages of our project.

Using Django's ORM (Object-Relational Mapping), we were able to easily create and manage models such as the User Preference model, which stores each user's selected topics, keywords, and news sources. The ORM makes it simple to insert, update, and retrieve data without needing to write raw SQL queries. However, we also have the flexibility to run manual queries if we need more control over the database. As our application grows, SQLite's simplicity allows us to easily transition to a more powerful database if we need better scalability or concurrent access. Overall, SQLite's ease of use has been crucial in helping us quickly develop and test our features.

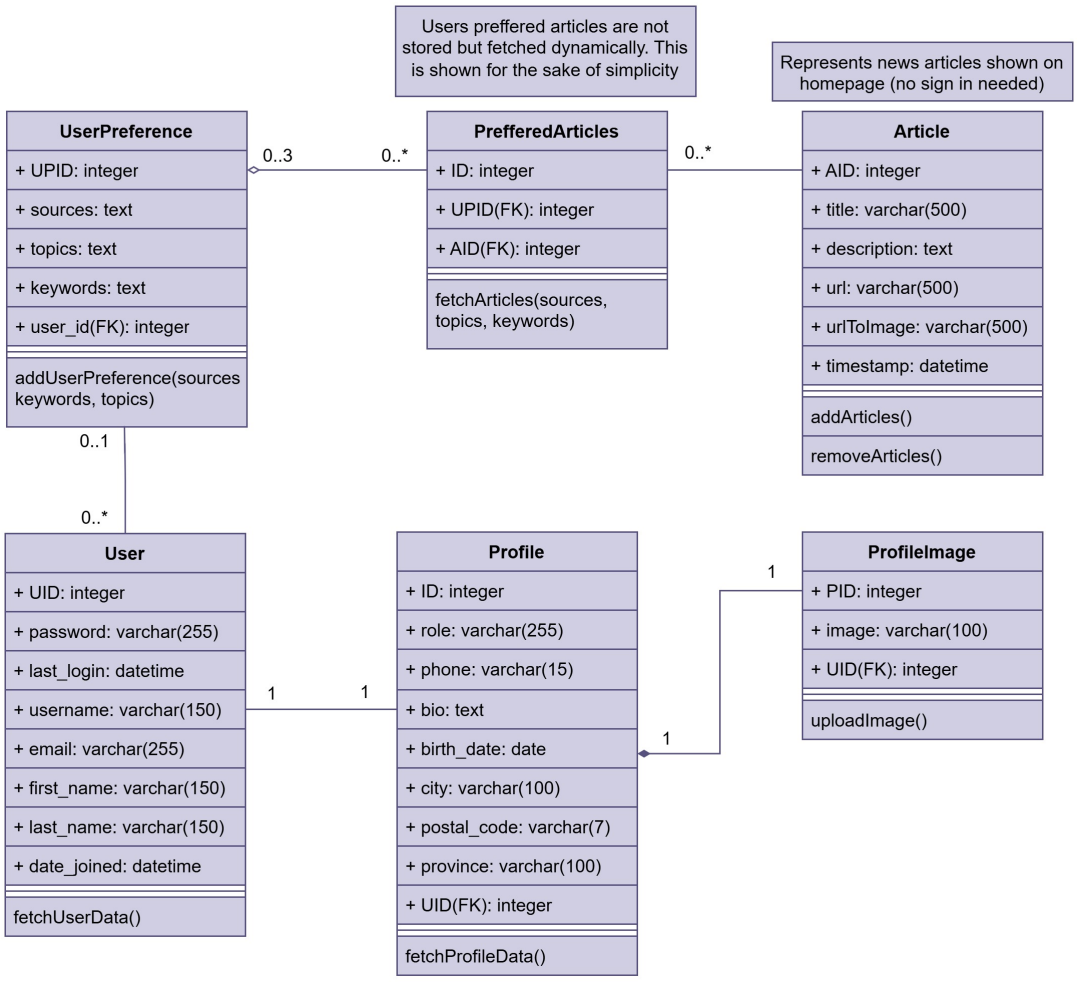


Figure 3.1: Class Diagram

## 3.4 GitHub & JIRA

For project management, we use JIRA to efficiently track progress, manage tasks, and delegate responsibilities within the team. As an excellent tool for Agile Scrum methodology, JIRA helps us manage sprints, track user stories, and ensure smooth collaboration across the team. It allows us to stay organized, adapt to changing requirements, and meet deadlines. Meanwhile, GitHub handles our version control, enabling us to work on different branches independently. This approach allows team members to develop features without conflicts, and GitHub's merge capabilities ensure smooth integration of changes, keeping the codebase organized and up-to-date.

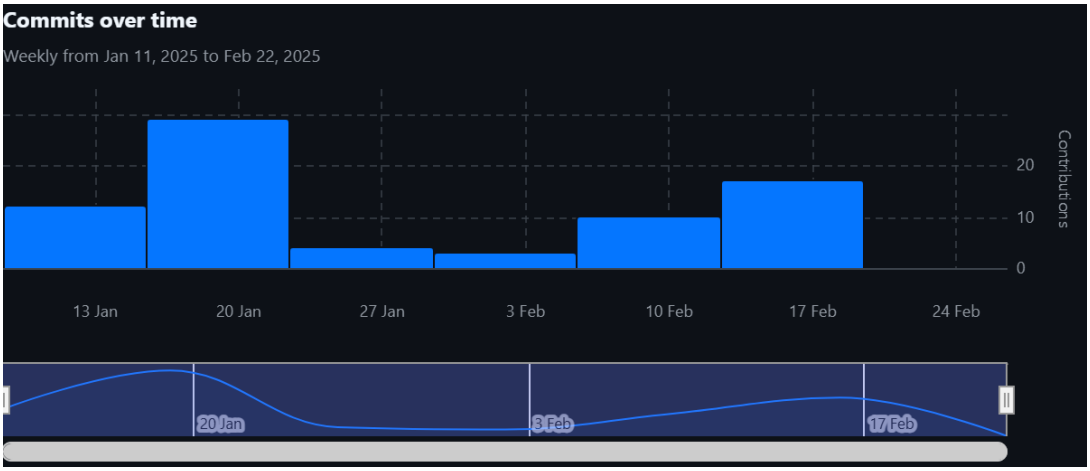


Figure 3.2: Commits over time


Figure 3.3: Individual Contribution

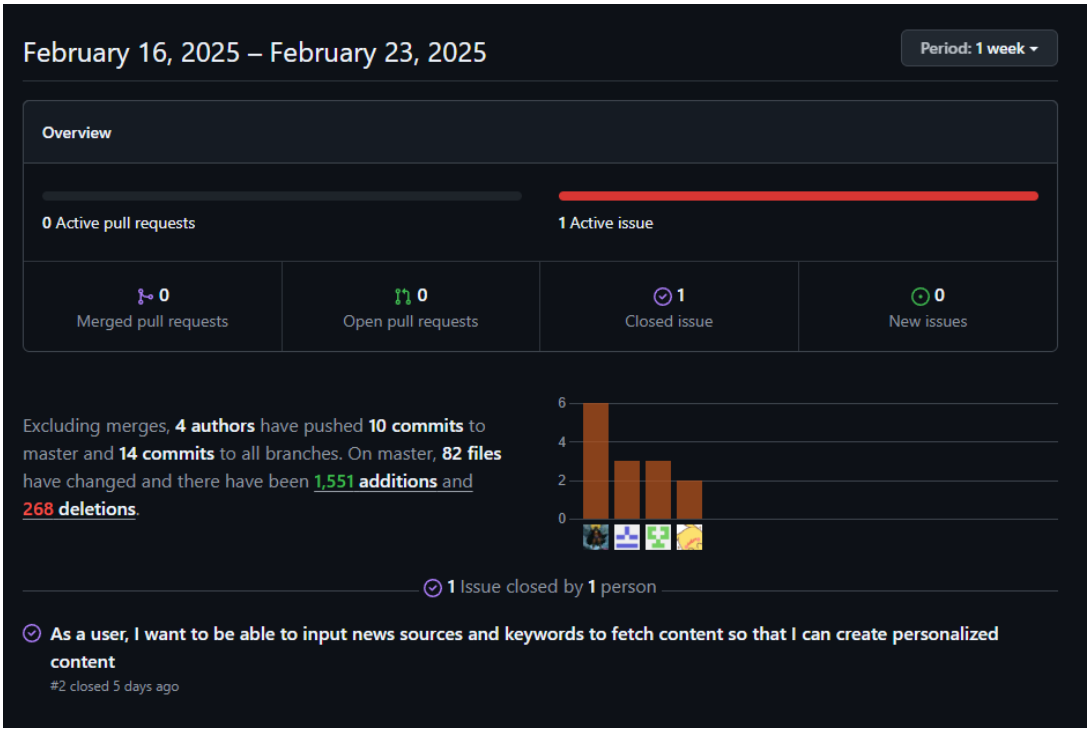


Figure 3.4: Github Pulse

# 4 Sprints & Meetings

## 4.1 Weekly Scrum Meetings

Before development began, we conducted a Sprint Planning Meeting to outline tasks, set priorities, and assign responsibilities. We divided the project into smaller sections and formed groups to focus on different aspects. Using user stories, we broke down requirements into manageable tasks and added them to our product backlog. The first week was crucial as we set up the Django development environment, ensuring everyone was comfortable with the framework. Our weekly meetings, held every Friday and led by Arifuzzaman (Scrum Master), allowed us to track progress, discuss updates, plan next steps, and offer support where needed.

Alongside our scheduled meetings, we maintained constant collaboration through pair programming sessions to enhance code quality. For any roadblocks, we held impromptu discussions on Discord to brainstorm solutions. At the end of each sprint, we conducted a Sprint Retrospective Meeting to reflect on what went well, what could be improved, and how we could enhance efficiency in the next sprint. This continuous feedback loop kept us aligned and motivated.

As we progress, we are thrilled to see our project take shape into a fully functional web application. The foundation is solid, and we are eager to refine and polish our work. With a shared commitment to delivering a high-quality, efficient, and user-friendly solution, we look forward to the final product with excitement and confidence.

## 4.2 Sprint 1

In our first sprint, we focused on laying the foundation of our platform while following Agile Scrum methodologies to ensure timely progress and adaptability. The primary goal was to implement user authentication, allowing users to register, log in, and securely access their accounts. We validated inputs, stored user data in the database, and ensured proper authentication mechanisms were in place, with rigorous testing to guarantee security and reliability.

| Task | Associated User Stories | Priority |
|---|---|---|
| Create a registration and login web interface | Scrum-9 | High |
| Create form fields for login and registration | Scrum-9 | High |
| Validate inputs for the forms | Scrum-9 | High |
| Create database to store users data | Scrum-9 | High |
| Create authentication system for users to login | Scrum-9 | High |

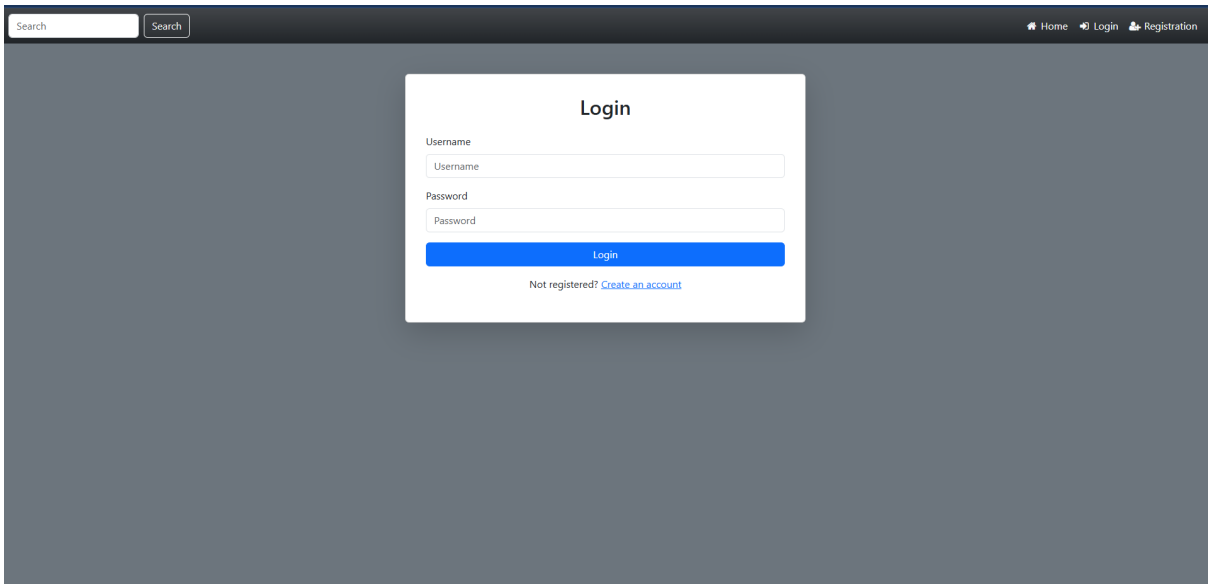


Figure 4.1: Login Page

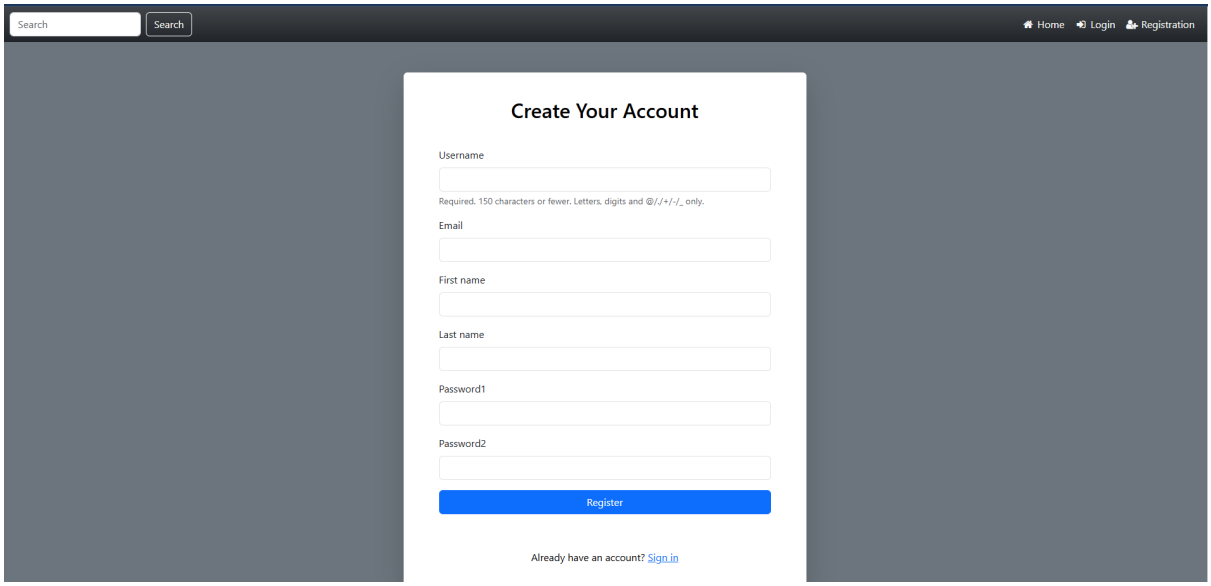


Figure 4.2: Registration Page

Next, we developed the dashboard and profile management system, enabling users to view and manage generated content, track performance, and update preferences. We aimed to give the interface an aesthetic, modern, and futuristic look, ensuring a visually appealing and user-friendly experience. We stored user preferences, including selected topics and sources in the database, making it easy for users to access and modify their settings seamlessly.

| Task | Associated User Stories | Priority |
|---|---|---|
| Create Profile page, Home page, Dashboard view, Archive News (HTML Pages) | Scrum-1 | High |
| Create backend logic for managing generated content | Scrum-1 | High |
| Create backend logic for updating preferences | Scrum-1 | High |
| Store preferences to the database | Scrum-1 | High |
| Store generated content to database for later use | Scrum-1 | Medium |

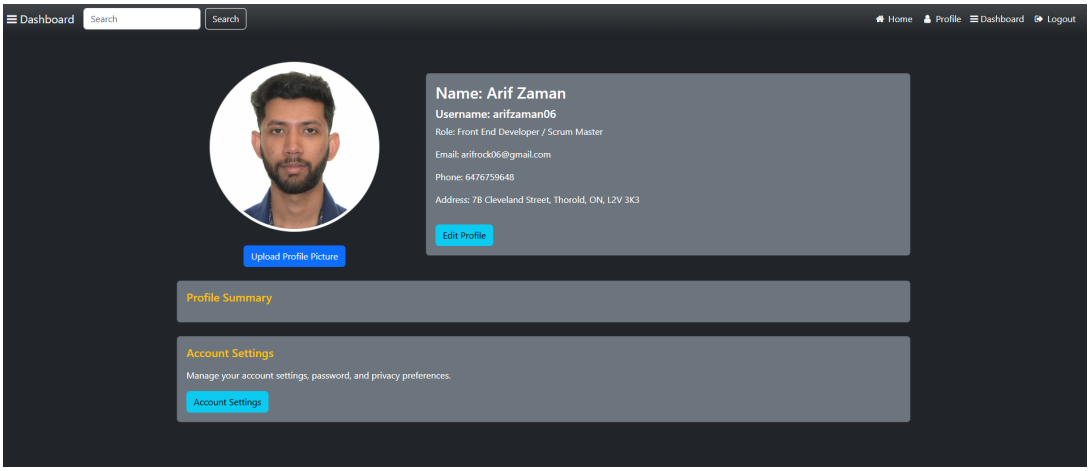


Figure 4.3: Profile Page

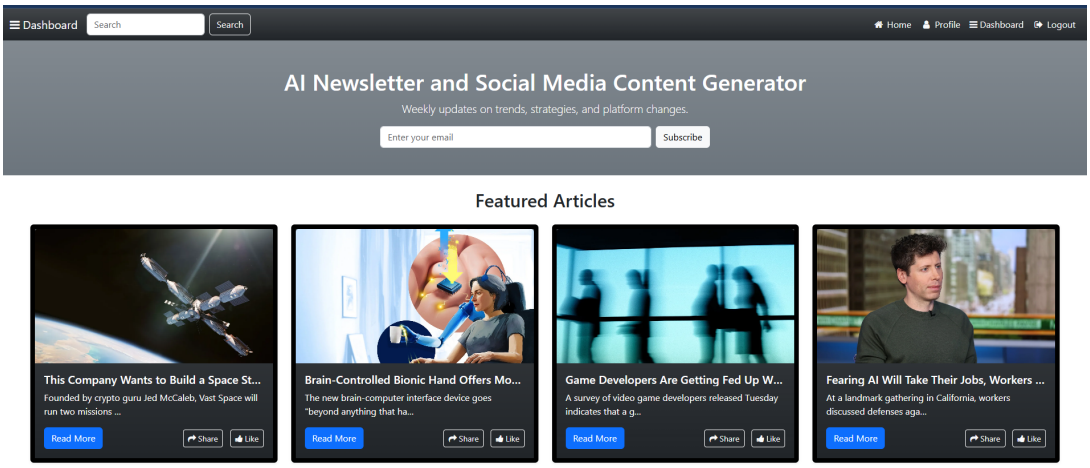


Figure 4.4: Home Page

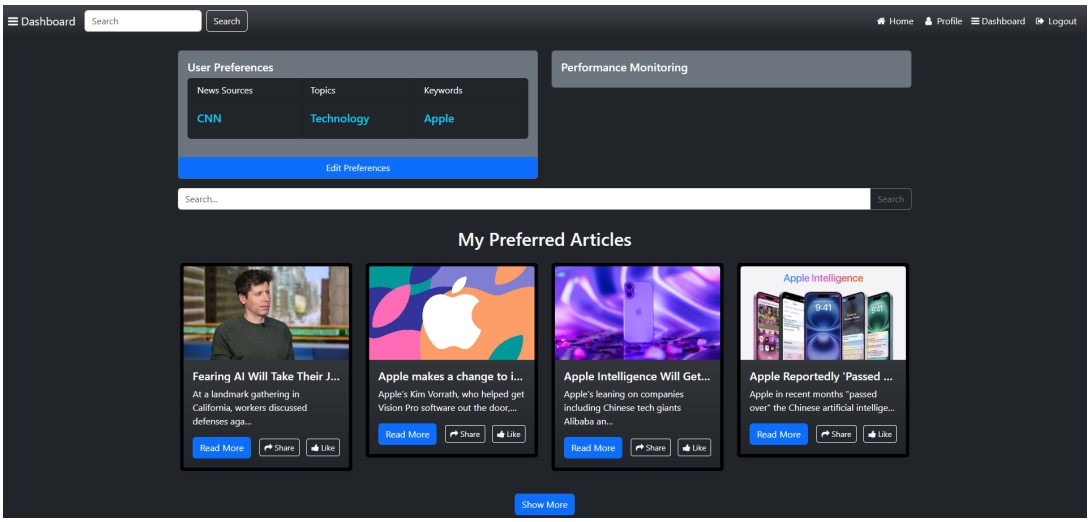


Figure 4.5: Dashboard Page

Lastly, we implemented content aggregation, allowing users to input preferred news sources, topics, and keywords. These preferences were stored and utilized by our backend to scrape relevant content dynamically, ensuring personalized and meaningful results. By the end of Sprint 1, we had successfully built the foundation of our application, keeping development aligned with Agile principles through iterative improvements and structured sprint planning.

| Task | Associated User Stories | Priority |
|---|---|---|
| Create input fields for sources, topics, and keywords | Scrum-2 | High |
| Scrap relevant keywords and topics-based news from the news sources or default sources | Scrum-2 | High |
| Develop backend logic for content aggregation | Scrum-2 | High |
| Implement a database to store user preferences | Scrum-2 | Medium |

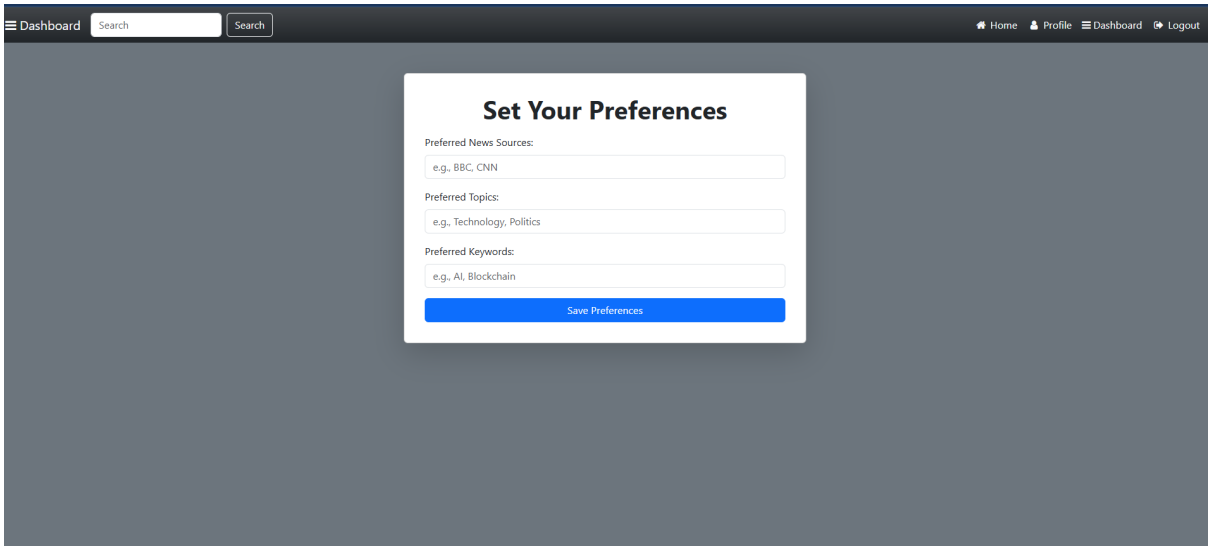


Figure 4.6: Preferences Page

## 4.3 Sprint 2

In Sprint Two, we focused on personalization and content optimization while maintaining Agile Scrum principles. We introduced a customizable email newsletter system, giving users access to a set of predesigned email templates. These templates come in different styles and designs, allowing users to select one that best suits their preferences. The templates dynamically fetch content based on user preferences, ensuring each newsletter is personalized and relevant. Users could also save and reuse templates for consistency. Testing was conducted to ensure that the templates displayed content correctly and functioned smoothly.

| Task | Associated User Stories | Priority |
|---|---|---|
| Design and develop the template selection interface (grid/list view with preview options) | Scrum-3 | High |
| Allow users to select content preferences, including topics, tone, and layout styles | Scrum-3 | Medium |
| Implement functionality to apply selected templates to generated content dynamically | Scrum-3 | Medium |
| Create an option to save and reuse custom templates | Scrum-3 | Low |
| Conduct testing to ensure templates are correctly applied and formatted | Scrum-3 | High |
| Display personalized content within the templates | Scrum-3 | Medium |

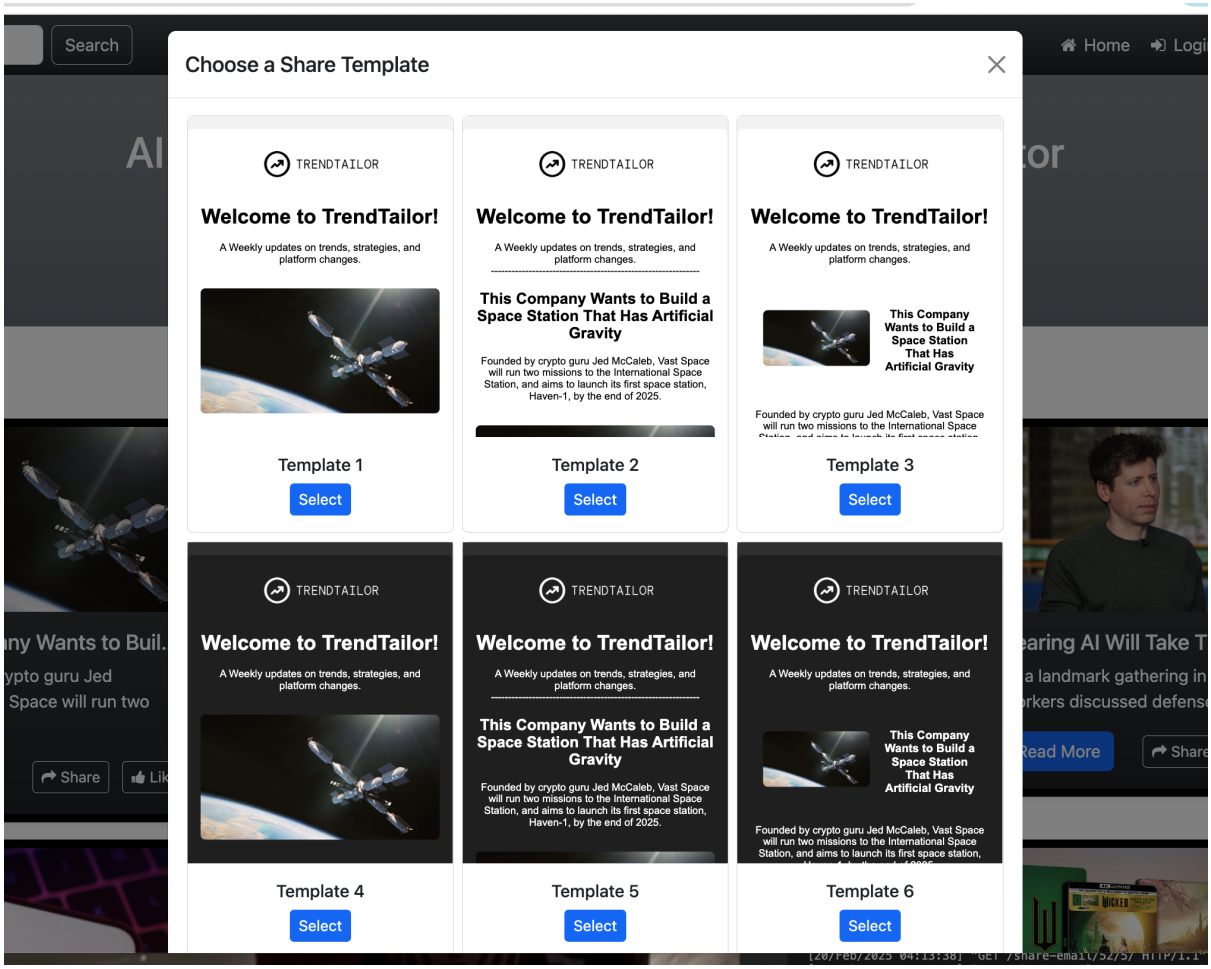


Figure 4.7: Templates Overview Diagram

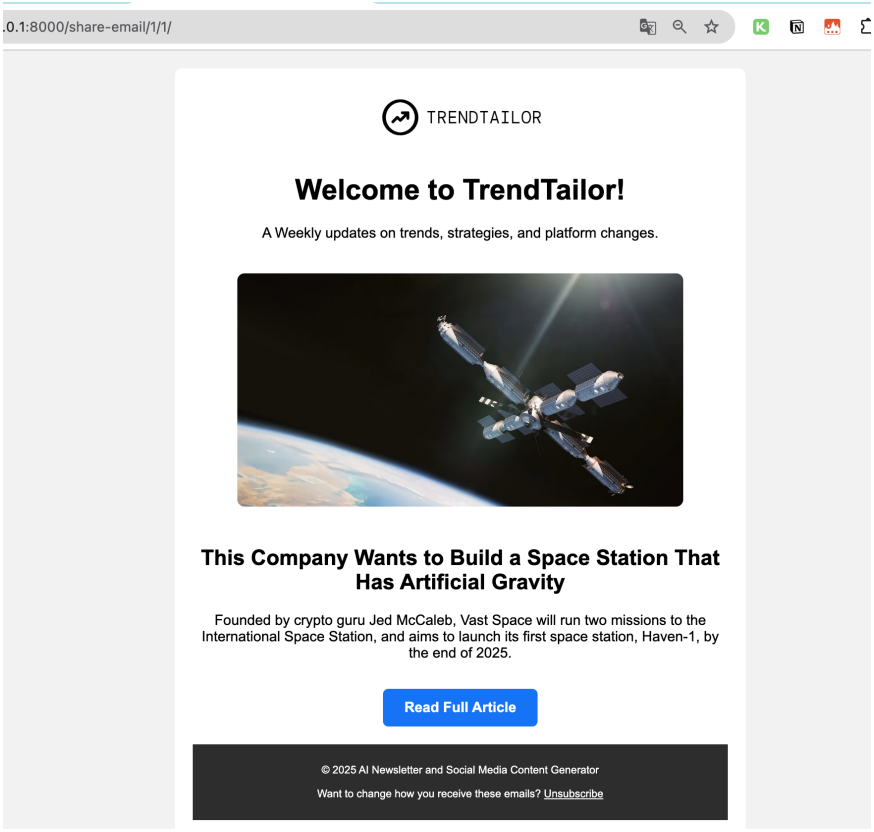


Figure 4.8: Template 1

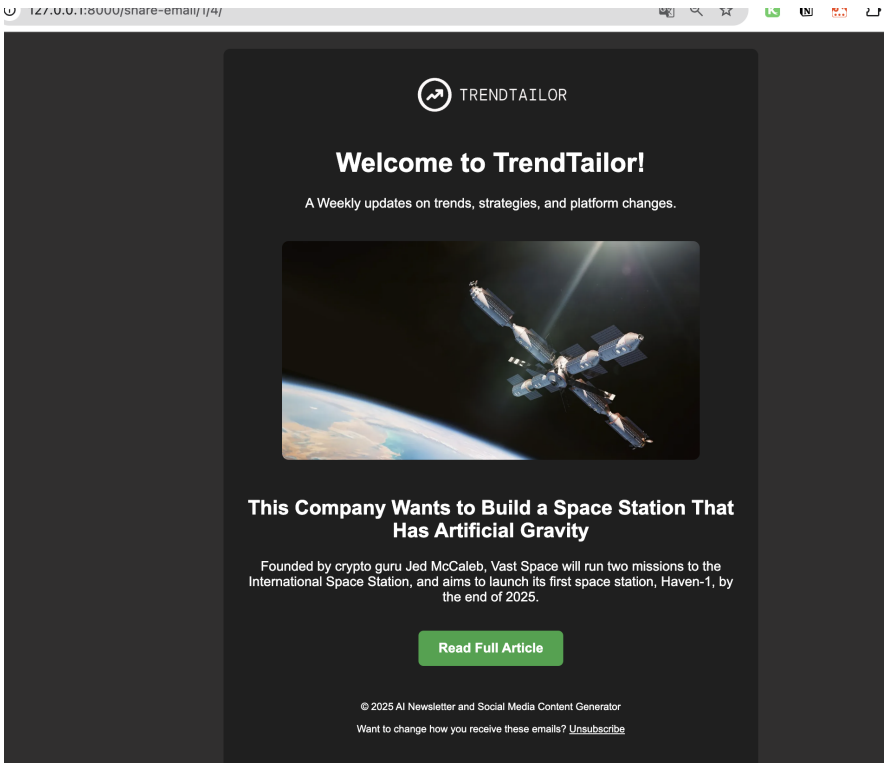


Figure 4.9: Template 2

To improve content accessibility, we implemented news article summarization using Large Language Model (LLM)-based prompts. By integrating a summarization API into our backend, we enabled the platform to condense lengthy articles into concise, easy-to-read summaries. A clean and intuitive UI was developed for seamless interaction, ensuring users could easily generate and review summaries. These enhancements brought greater customization and content efficiency to our platform while keeping the interface modern and user-friendly.

## 4.4 Sprint Progress : Activity Diagram

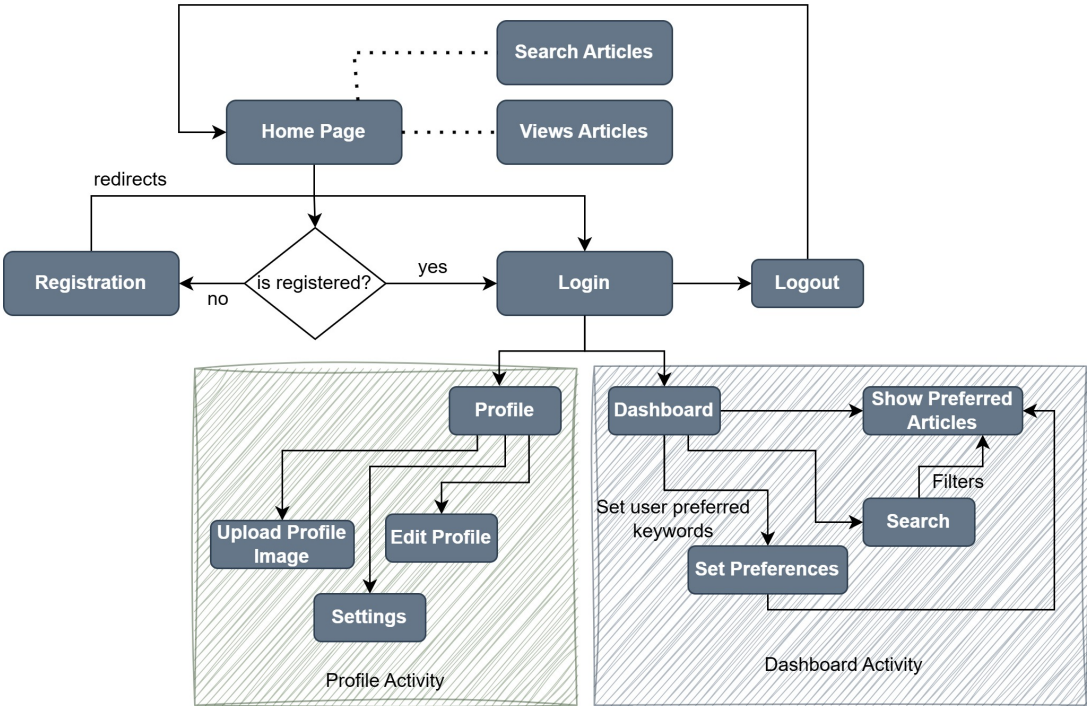


Figure 4.10: Activity Diagram

# 5 Challenges and Next Steps

During the project, we faced several challenges that required problem-solving, teamwork, and analytical abilities to resolve. Setting up the project in Django was the first hurdle, as for some of us, it was our first time using the framework. Configuring views, URLs, and templates required extra research, and we encountered server issues due to missing dependencies, which delayed the initial setup. GitHub also posed some challenges as we worked on different branches, leading to merge conflicts and synchronization problems. Additionally, we faced issues with our .gitignore file, which caused unnecessary personal files to be committed. After some research, we resolved the issue and ensured steady progress of the project.

On the API side, integrating free news APIs came with limitations. We were restricted to 100 requests per day, and some APIs limited us to only 10 articles per request, preventing real-time data fetching. To work around this, we implemented multiple free APIs and fetched data based on keywords, storing the results in our database. This ensured that when API request limits were reached, we could still retrieve content from our database, preventing the homepage from appearing empty. We also implemented a solution to normalize the data, allowing seamless aggregation from multiple sources. Despite these challenges, the team worked together to find effective solutions and keep the project moving forward.

For the next steps, we plan to enhance the project with key features, including the ability to publish content directly to popular social media platforms. We aim to introduce custom intervals for automated updates based on user preferences and support additional languages, such as French. Additionally, we will implement a system to save past newsletters and social media posts for future reference. Another potential enhancement is providing an API, allowing users to integrate our service into their own applications to extend its functionality.

# 6 Team Contributions

| Team Member | Contributions |
|---|---|
| Arifuzzaman (Scrum Master) | Worked on front-end development, creating the homepage, dashboard, profile, and email templates with a focus on a modern and intuitive user experience. Implemented article summarization using LLMs and conducted testing for accuracy. Managed team communications, scheduled meetings, and ensured adherence to Agile-Scrum methodology. Wrote and refined the progress report. |
| Hridoy Rahman | Worked on setting up the Django project foundation, managing GitHub repositories, resolving conflicts, and assisting teammates with version control issues. Developed backend functionalities, including managing generated content, updating user preferences, and database storage. Ensured seamless integration of all components and conducted final system testing. Also contributed to writing and proofreading the progress report. |
| Jeffin Sam Joji | Worked on the backend system for aggregating news content from multiple free APIs, including data scraping and standardization. Implemented error handling, secure API key management, and a reusable function for fetching and processing articles. Contributed to writing the progress report. |
| Yuanhan Huang | Focused on front-end development, designing and styling input forms for user news preferences. Developed dynamic and responsive email templates and assisted in managing the UI/UX design elements. Also helped with database management for storing user preferences and contributed to writing the progress report. |
| Kabir Sethi (Product Owner) | Worked on developing the front-end for user registration and login, ensuring an intuitive user experience. Implemented form validation mechanisms to maintain data integrity. Assisted in setting up the database for authentication and contributed to the progress report. |
| Sahil Rashid | Developed the user authentication system, enabling secure registration, login, and account management for users while ensuring security and data integrity. Additionally, assisted in creating Figma mockups to visualize the UI design and improve user experience. Also contributed to writing the progress report. |
| Jayant Saini | Helped with article summarization using Large Language Model. |