Bilkent University

Department of Computer Engineering

Senior Design Project

Impartial - Group T2324

Detailed Design Report

Group Members

Nisa Yılmaz - 22001849

Mustafa Cem Gülümser - 22003430

Murat Güney Kemal - 22002692

Aslı Karaman - 21901576

Eren Duran - 22003932

Supervisor

Halil Altay Güvenir

Table of Contents

1. Introduction

1.1 Purpose of the system

In a time where information surrounds us from various sources every day, there is a growing need for a platform that rises above biases and embraces an unbiased perspective. Introducing "Impartial News" – a project that goes beyond traditional news gathering. Impartial is developed not just to collect news from a variety of sources but to completely change the way we see and interact with information.

This dynamic website has the ability to smoothly transform articles from one viewpoint to another, removing the need for manual editorial editing of news articles to eliminate or manipulate bias. Whether an article starts from a left-leaning angle, Impartial can easily reframe it to match center or right perspectives. Beyond being just a sorting tool, Impartial aims to reshape the media landscape by promoting fairness, offering thorough information, and improving our understanding of global issues. Our goal is to bridge gaps in ideologies, encouraging a more detailed and informed public conversation that goes beyond usual limits.

1.2 Design goals

Usability

The app has a non-functional usability requirement because of the diverse and vast user base. The news is something that everybody on the planet sees, and if we want to succeed in this area, the app should be user-friendly. It should have a trivial user interface and instinctive functionality.

Security

Since we will be getting requests from the users, we will be handling user data, and that is a big problem with security. We will design our app with the security constraints in mind and use the appropriate encryption and safety practices throughout our journey.

Performance

Since we will use machine learning, generative AI, microservices, and such, the new content should be processed with acceptable speed, like a minute at the top, and the generative part should be comparable to the other alternatives in the area. For example, if it is vastly slower than chatGPT's response time our application won't be preferred to better alternatives.

Supportability

We are building the app for the web because we want to support every device possible and get an extensive range of coverage across platforms, and to achieve that, we will optimize how our web app looks for every platform.

Scalability

Since all the consumers for news are a considerable number, our system should scale accordingly because of our revolutionary idea. We are expecting at least 10% of the Reuters users, and that number is in the order of millions. To accommodate that many users and handle their requests for the generative part, we need to write optimized and scalable code from the bottom up.

1.3 Definitions, acronyms, and abbreviations

- GenAI: Generative Artificial Intelligence
- AWS: Amazon Web Services
- UI: User Interface
- AI: Artificial Intelligence
- KVKK: Kişisel Verileri Koruma Kurumu
- GDPR: General Data Protection Regulation
- LLM: Large Language Model

1.4 Overview

Our system fully automizes the previously described functions. We aim to classify the incoming news articles with the machine learning model we have trained instead of manual labor alternatives of other systems. After the output from our classification AI, our system will then neutralize the left or right political bias if detected. We aim to conduct this operation by training a GenAI model using a suitable large language model. By doing this we also eliminate manual labor required to rewrite the article free from any political bias.

2. Current software architecture

Currently, no GenAI models are available specifically for neutralizing the political bias of news articles. A website named Allsides provides political bias classification for news articles, but they do not use any machine learning or artificial intelligence systems. The classifications are made by volunteer editors and community feedback.

Another existing system is a browser extension that seeks out identical news articles from various sources and assigns labels based on the political alignment of each news outlet. This extension, although helpful in identifying different perspectives on a given topic, relies solely on manual assessments by its users and lacks the sophisticated capabilities of machine learning or artificial intelligence. The classifications are subject to the interpretations of users and may not capture the nuanced and evolving nature of political bias in news reporting.

While these existing systems contribute valuable insights, they fall short in harnessing the power of technology to autonomously neutralize political bias within news articles. Recognizing this gap, "Impartial News" emerges as a project that leverages advanced GenAI models to revolutionize the landscape of news consumption and provide a more dynamic and unbiased understanding of current events.

3. Proposed software architecture

3.1 Overview

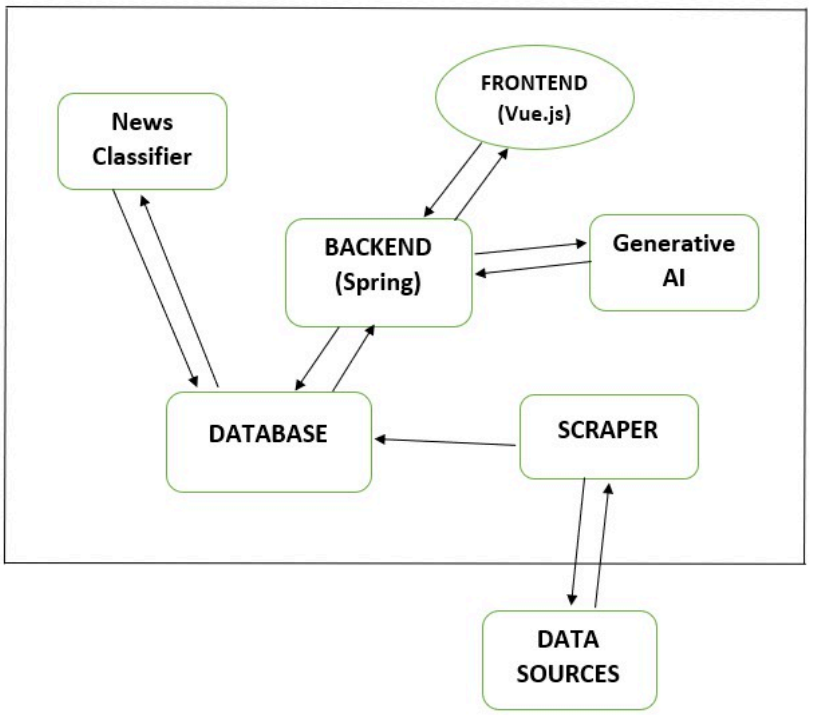

Figure 1: Basic system setup of the project

The high-level system architecture of our project will mainly consist of "News Classifier," "Database," "Scraper," "Data Sources," "Generative AI," "Backend," and "Frontend." In the backend development of our project, we plan to use "Spring," which is a Java Library, and for the frontend development of our project, we will use the "Vue.js" framework. The news will be obtained from various data sources, extracted with the scraper, and stored in our project's database. The news classifier of our architecture will receive the necessary data from our database. Later, the categorized news will also be stored in the database. Another feature of our project, which is generating articles with other categories, will be implemented with our project's Generative AI system architecture component.

3.2 Subsystem decomposition

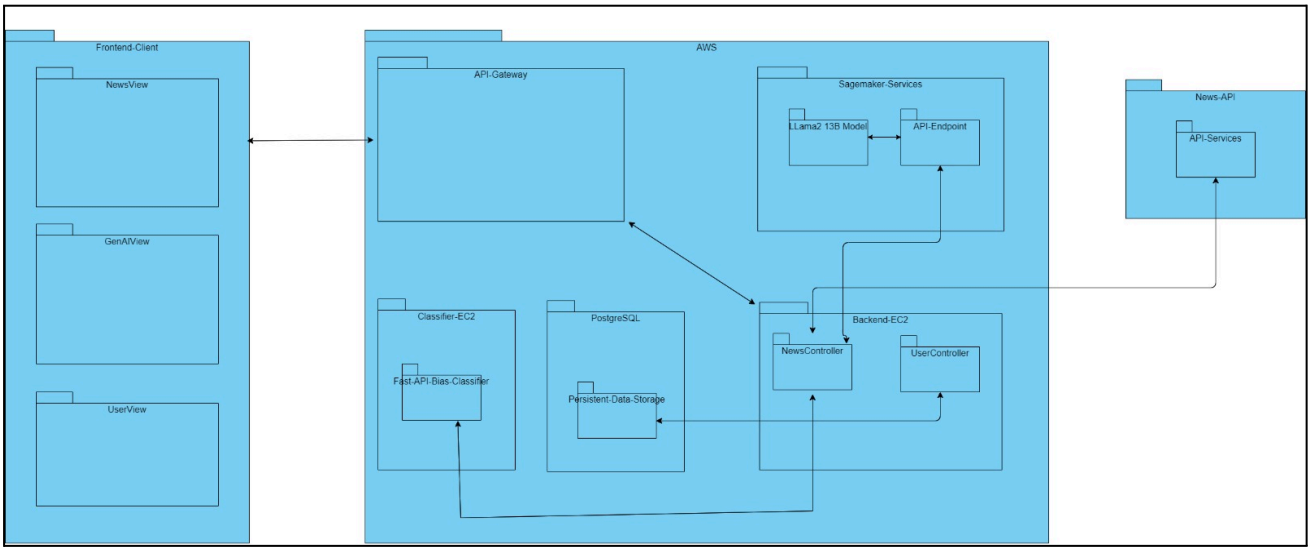

Figure 2: Subsystem decomposition diagram

We leverage the complete cloud infrastructure provided by AWS for our APIs. Specifically, we utilize the AWS API Gateway to manage our classifier, while employing an EC2 instance for our GenerativeAI tasks. For machine learning purposes, we rely on AWS SageMaker services. Both our backend and frontend are hosted on AWS Elastic Container Registry (ECR). Additionally, we fetch news data from the NewsAPI. By utilizing cloud services, we ensure scalability to efficiently handle incoming requests and optimize app performance.

3.3 Hardware/software mapping

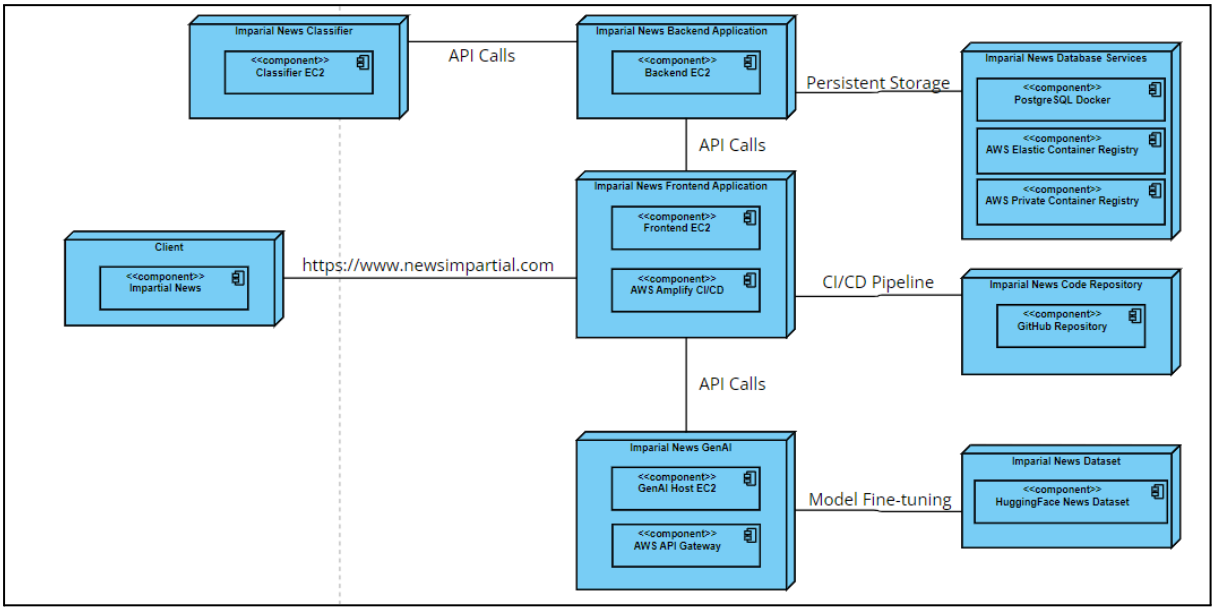

Figure 3: Hardware/software mapping diagram

The above diagram indicates the hardware-software mapping of the Impartial News application. Each box denotes a different hardware that is reserved for this application. Mostly, the hardware requirements are satisfied by the AWS services. The frontend, backend and classifier software are hosted on different EC2 machines in AWS. Similarly, the GenAI instance is hosted on a GPU-Optimized machine and is deployed in the cloud with AWS SageMaker and AWS API Gateway. For database services of the application, AWS Elastic Container Registry is used to deploy the PostgreSQL container as a docker container. Code repository is hosted in GitHub, and the GenAI fine-tuning dataset is hosted on HuggingFace.

3.4 Persistent data management

For storing persistent data, we have chosen to use PostgreSQL as it is a reliable and scalable relational database. We keep our database on AWS, as a dockerized instance. For Impartial, storing news articles is crucial as we have to display these for users. For news articles, we store the title, description, content, the link for the original article, the link for the image of the article, publisher and author information. Currently, we use an external API to collect news articles.

Another data we persist is the users' credentials. We will allow users to create accounts to use premium functionality like generating their own articles from a given article. Related to these accounts we also keep the bookmarks, comments and ratings of the user which are then can be accessed from the profile tab of the user.

3.5 Access control and security

- **Database Security and Privacy:** We are holding each password encrypted with BCrypt Password Encoder in order to ensure privacy of our customers and as a security measure in case of a data breach.
- **Web Security and Access Control:** To ensure security and access control we are using Spring Web Security, JWT authorization, and Url Based Access Policies. First of all no customer can access the secluded parts without login. And only the customers who have the right role can access the selected views.

4. Subsystem Services

4.1. User Interface Subsystem

In order to make the application more user friendly and easy to use, we worked on the design in a way that all pages can be accessed and directed from one main page which directly appears when the user logins: main page. It is indicated in Figure 2.

Main Screen: In the main page, three main components which are sidebar, feed and latest headlines shown to the user.

Feed: The feed component includes the tabs, details of the article, navigating options such as genAI option and read more option and some functionalities which are like button, bookmark button and comment button.

Sidebar: Sidebar provides the user to direct to profile page, select a category and logout from the application.

Latest Headlines: In the latest headlines component, latest news are shown to the user.

Profile Page: In the profile page, the user can see the bookmarked, liked and the commented articles by himself/herself.

Article Detail Page: In the article detail page, a more detailed version of the article is shown to the user.

GenAI Page: In the genAI page, the article generated ai technology is shown to the user.

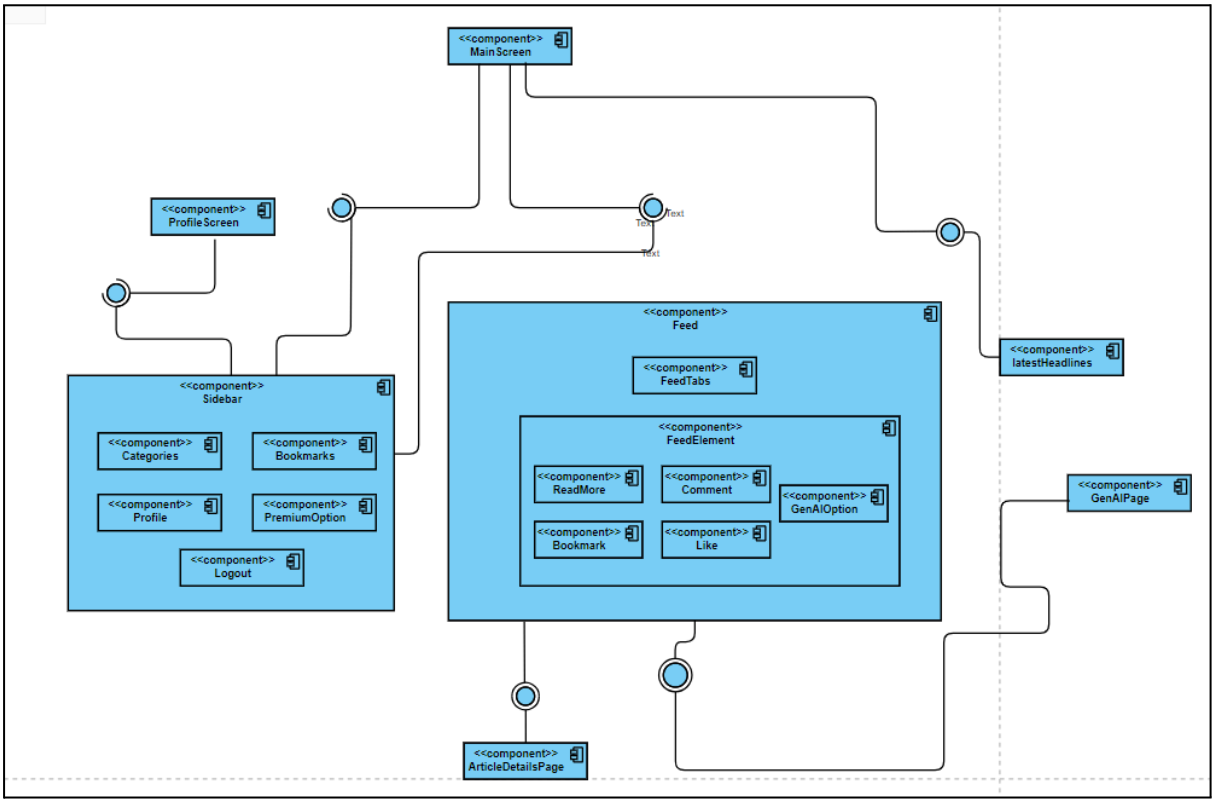

Figure 4: User interface subsystem diagram

4.2 Server

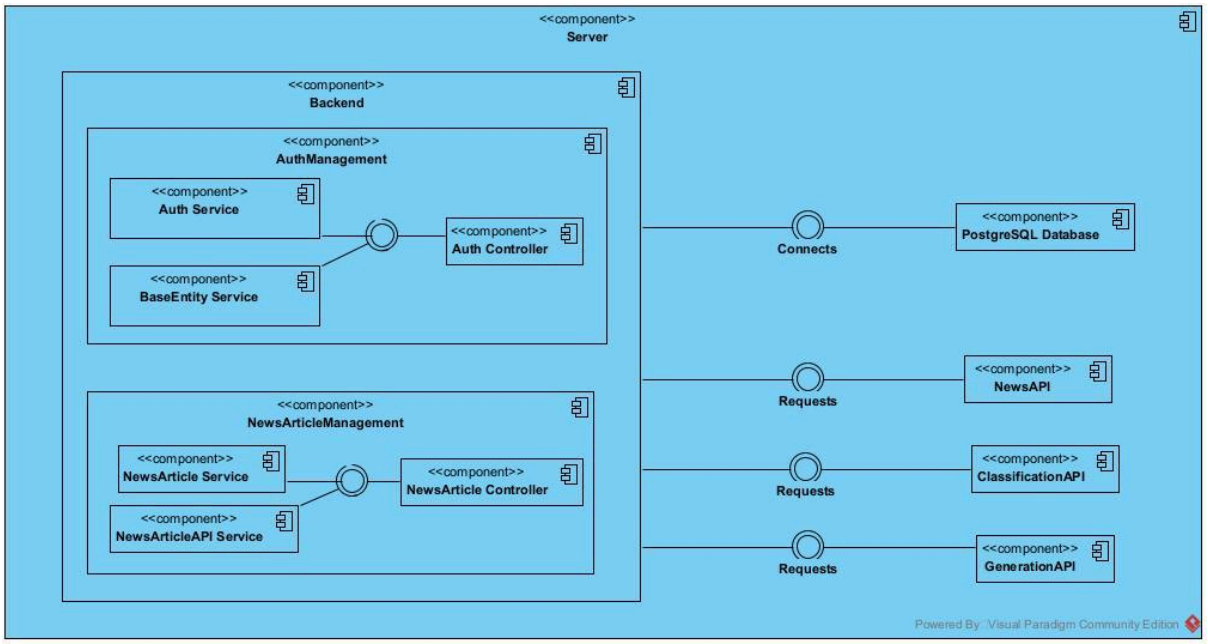

Figure 5: Server subsystem diagram

The functionality of the backend server hosted on AWS EC2 involves several steps. First, the server is accessed via SSH using a key pair for authentication. Once connected, Git is installed to retrieve the codebase from a GitHub repository. The backend code is then cloned onto the server. Docker is installed and started to containerize the application and manage dependencies. Additionally, Java is installed to execute the Spring Boot application. A PostgreSQL Docker container is pulled and launched with specific configurations for the database. The application's configuration file is updated to connect to the PostgreSQL database. Finally, the Spring Boot application is started either directly or as a daemon service using systemd. This configuration ensures that the application runs in the background and persists across server restarts. The journalctl command is used to monitor the logs of the running service for debugging and monitoring purposes. Overall, this setup leverages AWS EC2 for hosting, Git for version control, Docker for containerization, PostgreSQL for database management, Java for application execution, and systemd for managing background services.

5. Test Cases

T001 Signup Test Case

Test Requirements: Users will be able to register

Step	**Expected Result**	**Pass/Fail**
User enters their mail	Mail appears on the respective field	
User enters their password	Password appears on the respective field	
User clicks on the signup button, user is navigated to home and logged in automatically	User is registered and redirected to home page	

Table 1: Test Case 1

T002 Login Test Case

Test Requirements: Users will be able to login

Step	**Expected Result**	**Pass/Fail**
User enters their mail	Mail appears on the respective field	
User enters their password	Password appears on the respective field	
User clicks on the login button, user is navigated to home and logged in automatically	User is logged in and redirected to home page	

Table 2: Test Case 2

T003 Logout Test Case

Test Requirements: Users will be able to logout

Step	**Expected Result**	**Pass/Fail**
User clicks on the logout button (Prerequisite: User must be logged in)	User is logged out and redirected to home page	

Table 3: Test Case 3

T004 Evaluate Article Test Case

Test Requirements: Users will be able to evaluate the articles once they are logged in

Step	**Expected Result**	**Pass/Fail**
User clicks on stars to evaluate the article (Prerequisite: User must be logged in)	Evaluation registers and average is calculated again	

Table 4: Test Case 4

T005 Comment On Article Test Case

Test Requirements: Users will be able to comment on articles once they are logged in

Step	**Expected Result**	**Pass/Fail**
User clicks comment button of an article (Prerequisite: User must be logged in)	Comment box appears	
User enters comments	Comments appears on the respective field	
User clicks on the submit button	Comment box closes and evaluation is submitted	

Table 5: Test Case 5

T006 Classification API Test Case

Test Requirements: Users will be able to access our classification API through the website

Step	**Expected Result**	**Pass/Fail**
User clicks on Classification page button (Prerequisite: User must be logged in and purchased the service)	User is redirected to Classification API page	
User enters the text to the text field	Test appears on the respective field	
User clicks on the submit button	The text is classified and result is returned	

Table 6: Test Case 6

T007 GenAI API Test Case

Test Requirements: Users will be able to access our GenAI API through the website

Step	**Expected Result**	**Pass/Fail**
User clicks on GenAI page button (Prerequisite: User must be logged in and purchased the service)	User is redirected to GenAI API page	
User enters the text to the text field	Test appears on the respective field	
User clicks on the submit button	New text is generated and returned to user	

Table 7: Test Case 7

T008 View Article Details Test Case

Test Requirements: Users will be able to view article details

Step	**Expected Result**	**Pass/Fail**
User clicks on an article	App opens detailed articles Section	

Table 8: Test Case 8

T009 Change Bias Of Article Test Case

Test Requirements: Users will be able to view other biased versions of an article

Step	**Expected Result**	**Pass/Fail**
User clicks on other bias buttons (left, center, right) page button	The selected version of the article is displayed to user	

Table 9: Test Case 9

T010 Change Password Test Case

Test Requirements: Users will be able to change their passwords

Step	**Expected Result**	**Pass/Fail**
User clicks on profile page button (Prerequisite: User must be logged in)	User is redirected to profile page	
User clicks on password change button	Password change box appears	
User enters the new password 2 times in text fields	Texts appears on their respective fields	
User clicks on the submit button	User's password has been changed	

Table 10: Test Case 10

T011 View Profile Test Case

Test Requirements: Users will be able to view their profile page

Step	**Expected Result**	**Pass/Fail**
User clicks on profile page button (Prerequisite: User must be logged in)	User is redirected to profile page	

Table 11: Test Case 11

T012 Add Bookmark Test Case

Test Requirements: Users will be able to bookmark articles

Step	**Expected Result**	**Pass/Fail**
User clicks on add bookmark button of an article (Prerequisite: User must be logged in)	The article is added to the user's bookmarks	

Table 12: Test Case 12

T013 View Bookmarks Test Case

Test Requirements: Users will be able to view their bookmarked articles

Step	**Expected Result**	**Pass/Fail**
User clicks on profile page button (Prerequisite: User must be logged in)	User is redirected to profile page	
User clicks on bookmark tab	User's bookmarks are displayed	

Table 13: Test Case 13

T014 Delete Account Test Case

Test Requirements: Users will be able to delete their accounts

Step	**Expected Result**	**Pass/Fail**
User clicks on profile page button (Prerequisite: User must be logged in)	User is redirected to profile page	
User clicks on delete the account button	The user account gets deleted	

Table 14: Test Case 14

T015 Usability Test Case

A survey among users will be conducted to evaluate the usability of the application.

T016 Security Test Case

AWS WAF and security groups are used in our infrastructure to ensure the security of our web application. Certain web application attacks will be tested against these measures.

T017 Performance Test Case

Performance optimized machines were used to host our application. H1Load will be used to test the performance capabilities of our web application.

T018 Supportability Test Case

The application will be developed responsive and will be tested on different devices by our team.

T019 Scalability Test Case

Refer to AWS docs. [1]

6. Consideration of Various Factors in Engineering Design

6.1 Constraints

Implementation Constraints

- Github and Jira will be used to control the development process of our application.
- Vue.js will be used for the frontend development.
- Java and Spring framework will be used for the backend development.
- Pytorch and Huggingface transformers frameworks will be used to develop the AI models.
- FastAPI and Python will be used to deploy the AI models.
- PostgreSQL will be the database system.
- The backend will be deployed to AWS.

- Our machine learning models will be trained in the cloud using Google Collab.
- NewsAPI will be used to acquire new news articles.

Economic Constraints

- Backend deployment will be free using AWS EC2 free tier.
- Depending on our performance requirements, we might upgrade our Google Collab to a paid plan.
- All frameworks and libraries used in this project are free.
- NewsAPI is free in the development stage.

Ethical Constraints

- Any user data collected will be necessary and relevant to our system.
- The collected data will not be shared with 3rd parties without the users' explicit permission.

Sustainability Constraints

- The application and AI models will be maintained periodically.
- Any discovered bugs will be fixed in a week.

Language Constraints

- The language of the website will be English. More languages may be added in the future.

6.2 Standards

Documentation Standards:

- All sections of the report should be clearly labeled with appropriate headings and subheadings.
- Use consistent formatting throughout the document, including font style, size, and spacing.
- Ensure that all figures, tables, and diagrams are properly labeled and referenced in the text.
- Include a table of contents with accurate page numbers for easy navigation.
- Use a consistent citation style for references

.

Language and Communication Standards:

- Use clear and concise language to convey information effectively.
- Avoid jargon or technical terms that may be unfamiliar to the audience without explanation.
- Define all acronyms and abbreviations upon first use.
- Proofread the document for grammatical errors and typos.

Design Standards:

- Adhere to industry best practices and standards in software and system design.
- Ensure that the proposed software architecture meets scalability, security, performance, and supportability requirements outlined in the report.
- Use standardized frameworks and technologies where applicable (e.g., Vue.js for frontend development, Spring for backend development).
- Follow design patterns and principles to promote maintainability and extensibility of the system.

Security Standards:

- Implement appropriate security measures throughout the system architecture to protect user data and ensure confidentiality, integrity, and availability.
- Utilize encryption for sensitive data storage and transmission.
- Follow industry standards and regulations such as GDPR and KVKK for data protection and privacy.

Testing Standards:

- Develop comprehensive test cases covering all aspects of the system functionality, including usability, security, performance, and scalability.
- Conduct rigorous testing using both automated and manual techniques to identify and address any issues or vulnerabilities.
- Document test results and any deviations from expected outcomes for future reference.

Ethical Standards:

- Adhere to ethical guidelines in the development and deployment of the system, including transparency, fairness, and respect for user privacy.
- Avoid biases in algorithmic decision-making processes and ensure that the system promotes unbiased and inclusive content dissemination.
- Obtain informed consent from users for data collection and processing activities.

Teamwork Standards:

- Foster a collaborative and inclusive team environment where all members contribute effectively to project goals.
- Establish clear communication channels and project management tools (e.g., Jira) to facilitate coordination and task assignment.
- Encourage leadership and initiative among team members while promoting shared responsibility and accountability for project outcomes.

Compliance Standards:

- Ensure compliance with relevant laws, regulations, and standards governing software development, data protection, and information security.
- Regularly review and update the system to address any changes in compliance requirements or industry standards.

7. Teamwork Details

7.1 Contributing and functioning effectively on the team

To ensure proper teamwork we devised a task distribution system. In this system, we divide our work on a weekly basis. We assess the current needs of our project and assign the parts according to our needs and match the unique skills of our teammates with tasks strategically. With this system we ensure that the work is distributed equally and according to the strengths of each member. While creating reports and documentations for our project we divide the work equally among ourselves. In the other technical tasks, each member selects the tasks that fit their technical skills and the needs of our project. Tasks like research are also divided equally among all teammates. To keep track of the tasks and stay organized we use Jira.

7.2 Helping creating a collaborative and inclusive environment

We are a diverse team of engineers with expertise spanning various genders, hobbies, and professional fields. Some of us specialize in frontend development, others in backend development, some in machine learning, and still, others in DevOps. Despite our differing backgrounds and styles of working and creating, we came together in great harmony to collaborate on this project as a unified team.

7.3 Taking lead role and sharing leadership on the team

In our project each of our team members act as leaders for different tasks. For each task we appoint a leader to check the development process and organize the task. The leader changes for each task and usually depends on the strengths of the teammate and their availability to acquire extra responsibility.

8. Glossary

- GenAI: Generative Artificial Intelligence
- AWS: Amazon Web Services
- UI: User Interface
- AI: Artificial Intelligence
- KVKK: Kişisel Verileri Koruma Kurumu
- GDPR: General Data Protection Regulation
- LLM: Large Language Model

9. References

[1]

[2] Amazon Web Services, "High Availability and Scalability on AWS," Amazon Web Services Whitepapers, Accessed: Mar. 7, 2024. [Online]. Available: https://docs.aws.amazon.com/whitepapers/latest/real-time-communication-on-aws/high-availability-and-scalability-on-aws.html