Bilkent University

Department of Computer Engineering

Senior Design Project

Impartial - Group T2324

Final 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

In a world surrounded with biased and manipulated information, Impartial News emerges as a media of unbiased reporting. More than just a news aggregator, Impartial is a groundbreaking project that revolutionizes how we consume and interact with news. This dynamic platform utilizes generative AI technology to transform articles from one political perspective to another, eradicating bias and offering a balanced view of current events. Whether an article begins with a left-leaning, right-leaning, or center perspective, Impartial can adeptly reframe it to present a more comprehensive picture. By promoting fairness and providing thorough, unbiased information, Impartial aims to bridge ideological divides and elevate public discourse. Join us in reshaping the media landscape and fostering a more informed and nuanced understanding of global issues!

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.

The design goals for the app encompass usability, security, performance, supportability, and scalability. Usability is paramount due to the app's broad user base, necessitating a user-friendly interface and intuitive functionality. Security measures are crucial, given the

handling of user data, thus encryption and safety practices are implemented throughout. Performance expectations involve processing new content swiftly, with machine learning and generative AI comparable in speed to alternatives. Supportability entails optimizing the web app for various platforms to ensure widespread accessibility. Scalability is imperative to accommodate a substantial user influx, with the system designed to handle millions of users efficiently. The system automates tasks such as classifying news articles using machine learning, neutralizing political bias, and rewriting articles through a GenAI model, thus reducing manual labor.

2. Requirements Details

- The application interface must be intuitive and user-friendly, allowing the use of a diverse user base.
- Non-functional usability requirements must be met to ensure ease of use for all users globally.
- The app's functionality should be instinctive, requiring minimal learning curve for users.
- Security practices should be integrated into every aspect of the application's design and development process.
- New content processing time should be optimized, aiming for a maximum processing time of one minute for new news articles.
- The performance of machine learning and generative AI components should be comparable to industry standards and have a competitive accuracy.
- Response times for AI-generated content should be competitive with alternative solutions and should not make the user wait too much.
- The application should be compatible with various web platforms and devices to ensure widespread accessibility such as laptops and mobile platforms.

- Responsive design techniques must be employed to optimize the app's appearance and functionality across different screen sizes and resolutions.
- The system architecture must be designed to accommodate a large user base, with provisions for handling thousands of users.
- Code optimization and scalability measures should be implemented to ensure smooth operation even under high user loads.
- The application should automate tasks such as news article classification using machine learning models.
- Political bias detection and neutralization should be automated through a GenAI model trained on a large news article dataset.
- Manual labor for rewriting articles to remove bias should be eliminated through automation processes.

3. Final Architecture and Design Details

3.1 Overview

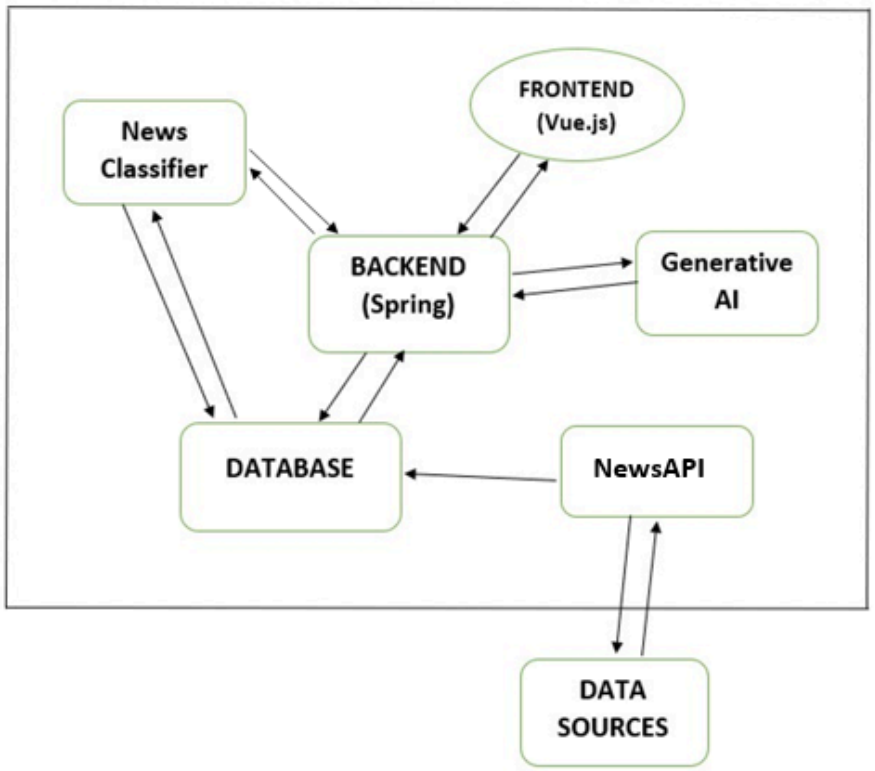

Figure 1: Basic system setup of the project

Our project's high-level system design comprises components such as the "News Classifier," "Database," "NewsAPI," "Data Sources," "Generative AI," "Backend," and "Frontend." For backend development, we leveraged "Spring," a Java Library, while the frontend was built using the "Vue.js" framework. News content sourced from NewsAPI, an external API, was sent to the classifier component. After the alignment of the news articles are determined, they are then stored in the database. Additionally, our project features a Generative AI system that generates articles with a desired alignment from a given news article text.

3.2 Subsystem Decomposition

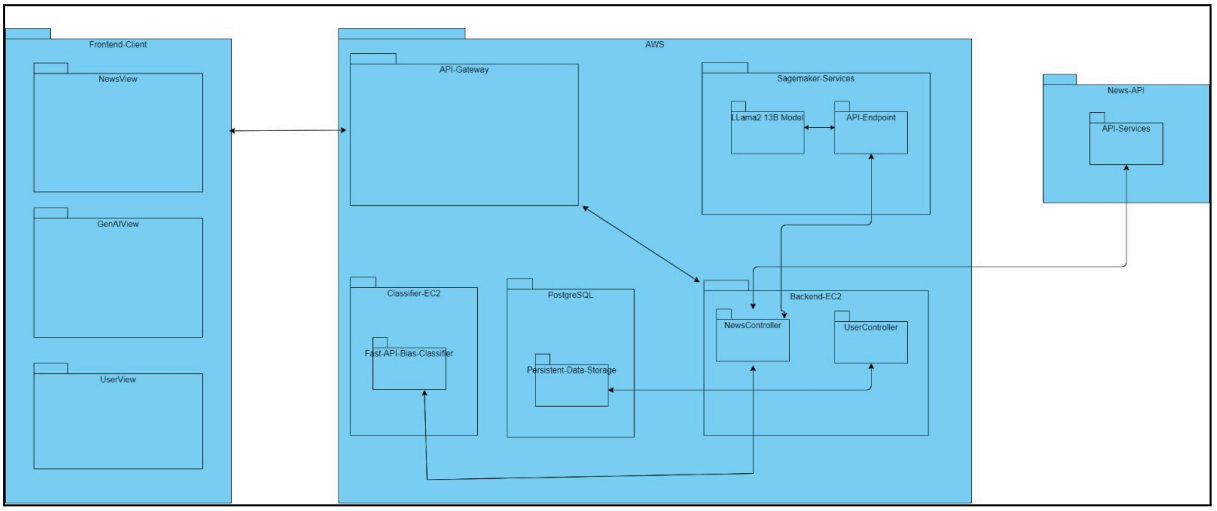

Figure 2: Subsystem decomposition diagram

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

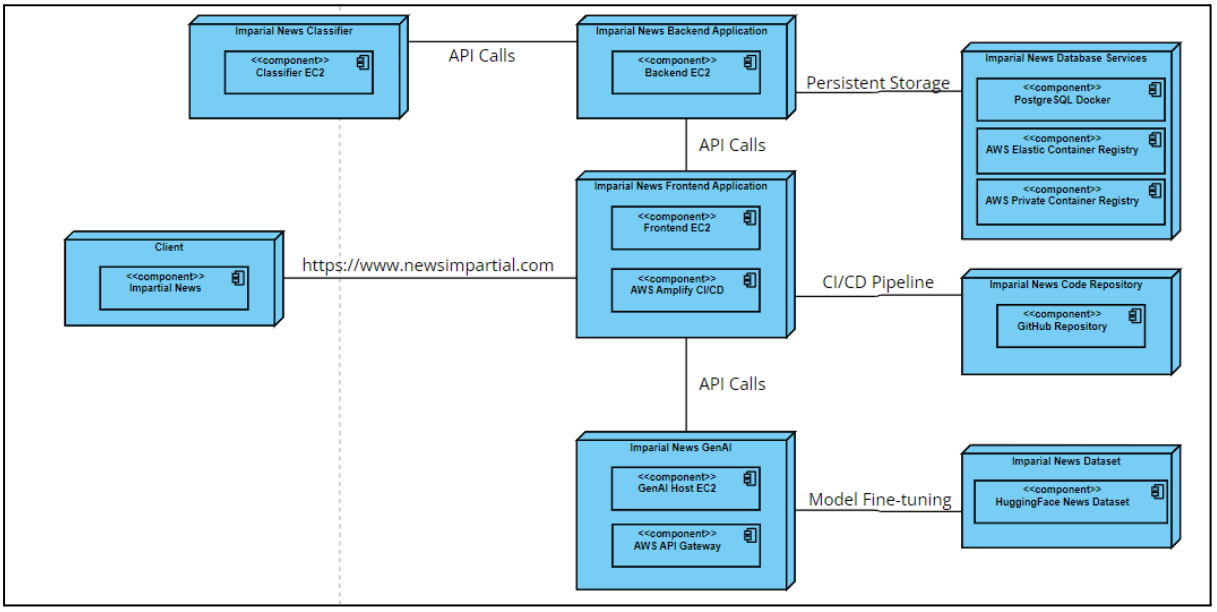

Figure 3: Hardware/software mapping diagram

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. Below is the detailed diagram of AWS structure.

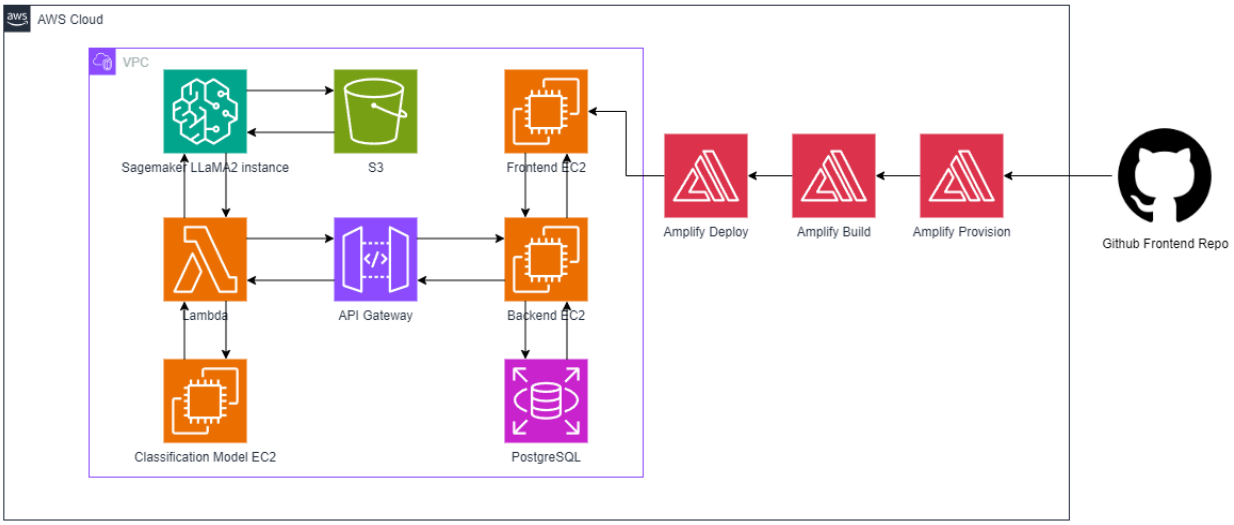

Figure 4: Detailed AWS Diagram

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 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. Development/Implementation Details

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 5.

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 indicating the alignment of the news articles, details of the article, navigating options such as genAI option and read more option and some functionalities which are classification reasoning, bookmark button and comment button.

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

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

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

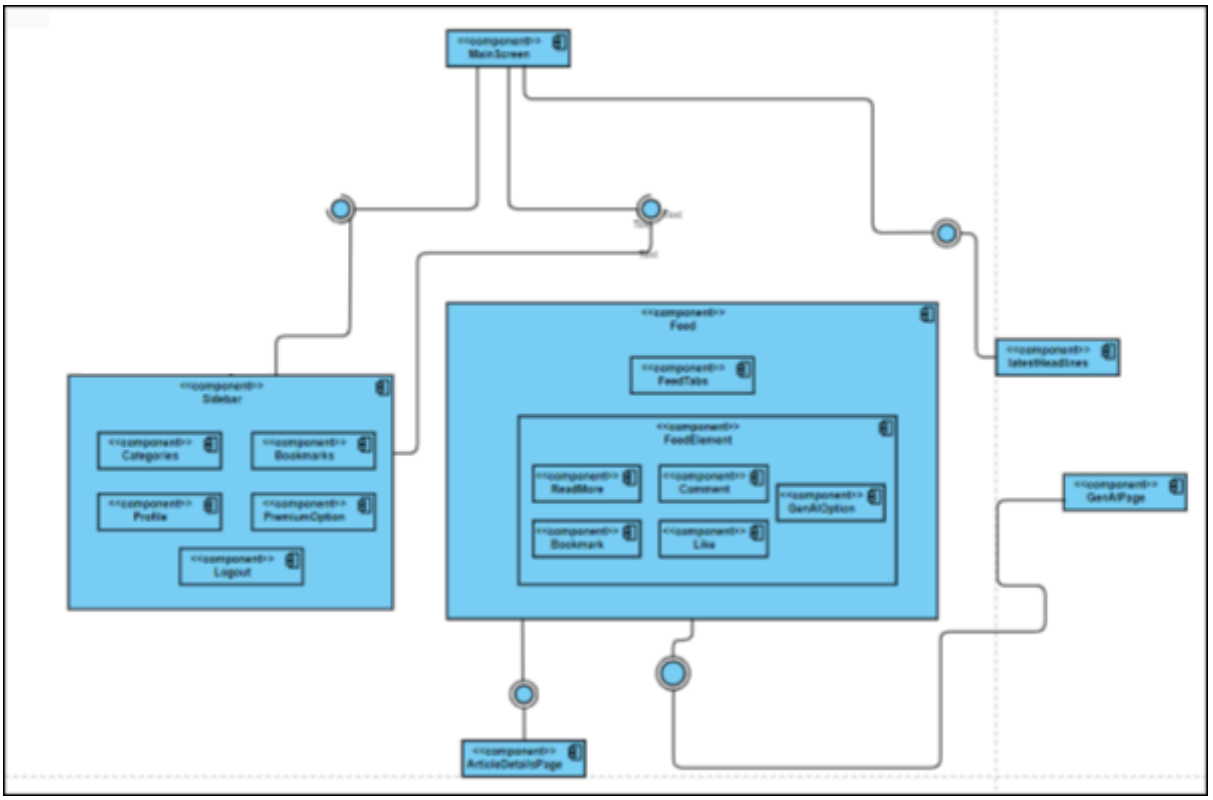

Figure 5: User interface subsystem diagram

4.2 Server

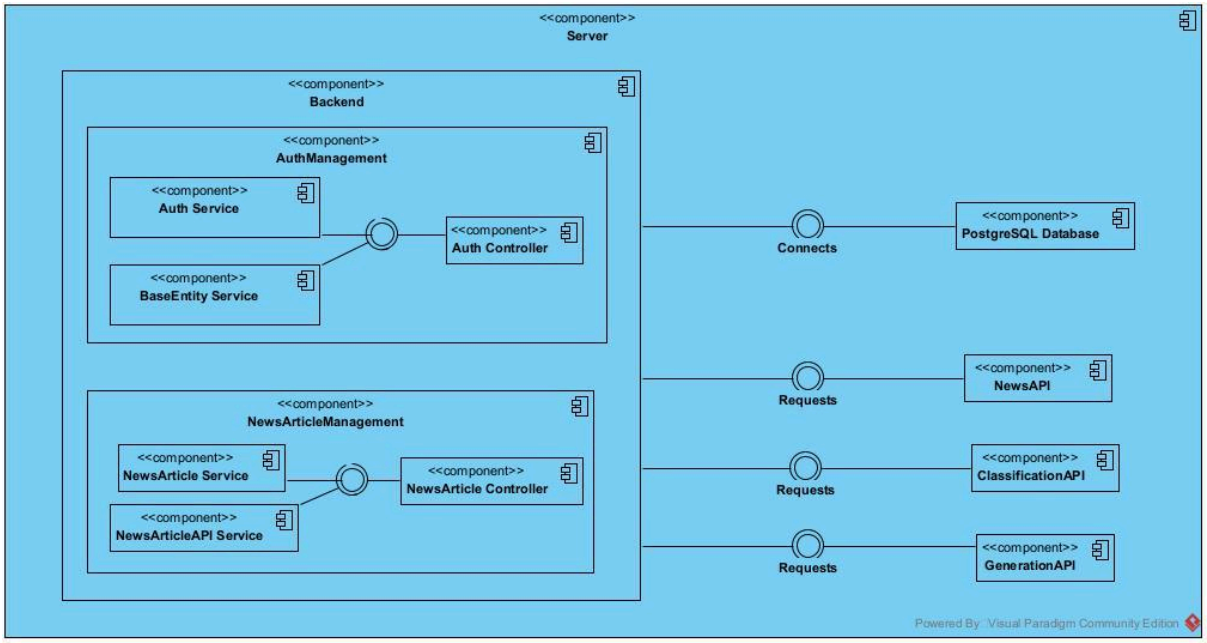

Figure 6: Server subsystem diagram

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 and Results

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	Pass
User enters their password	Password appears on the respective field	Pass
User clicks on the signup button, user is navigated to home and logged in automatically	User is registered and redirected to home page	Pass

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	Pass
User enters their password	Password appears on the respective field	Pass
User clicks on the login button, user is navigated to	User is logged in and redirected to home page	Pass

home and logged in automatically		

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 login page	Pass

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	Pass

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	Pass
User enters comments	Comments appears on the respective field	Pass
User clicks on the submit button	Comment box closes and evaluation is submitted	Pass

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	Pass

User enters the text to the text field	Test appears on the respective field	Pass
User clicks on the submit button	The text is classified and result is returned	Pass

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	Pass
User enters the text to the text field	Test appears on the respective field	Pass
User clicks on the submit button	New text is generated and returned to user	Pass

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	Pass

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	Pass

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/Removed**	**Explanation**
User clicks on profile page button (Prerequisite: User must be logged in)	User is redirected to profile page	Removed	Profile page button has been changed with bookmarks button. We decided that there was no use for a profile page on our application.
User clicks on password change button	Password change box appears	Pass	
User enters the new password 2 times in text fields	Texts appears on their respective fields	Pass	
User clicks on the submit button	User's password has been changed	Pass	

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/Removed	Explanation
User clicks on profile page button (Prerequisite: User must be logged in)	User is redirected to profile page	Removed	Profile page button has been changed with bookmarks button. We decided that there was no use for a profile page on our application.

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	Pass

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/Removed**	**Explanation**
User clicks on profile page button (Prerequisite: User must be logged in)	User is redirected to profile page	Removed	Profile page button has been changed with bookmarks button. We decided that there was no use for a profile page on our application.
User clicks on bookmark tab	User's bookmarks are displayed	Pass	

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/Removed**	**Explanation**
User clicks on profile page button (Prerequisite: User must be logged in)	User is redirected to profile page	Removed	Profile page button has been changed with bookmarks button. We decided that there was no use for a profile page on our application.

User clicks on delete the account button	The user account gets deleted	Removed	Profile page button has been changed with bookmarks button. We decided that there was no use for a profile page on our application.

Table 14: Test Case 14

T015 Usability Test Case

A survey conducted with 5 test users, all users found the usability satisfactory.

T016 Security Test Case

AWS WAF and security groups are used in our infrastructure to ensure the security of our web application. AWS guarantees the security of the cloud. [1]

T017 Performance Test Case

Performance optimized machines were used to host our application. H1Load was used to test the performance capabilities of our web application. After the experiment, on average 1500 HTTP requests per second load was handled by the classifier web server (t2.large instance) with small articles. This number fell to 1000 HTTP requests per second with larger articles. The backend web server (t2.medium instance) could handle 1200 HTTP requests per second on average.

T018 Supportability Test Case

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

T019 Scalability Test Case

AWS provides scalability, refer to AWS docs. [2]

6. Maintenance Plan and Details

6.1 Regular Updates and Patch Management

All software components, including the frontend, backend, FastAPI, SageMaker models, and AWS services, will be regularly updated with the latest patches and security fixes to mitigate potential vulnerabilities.

6.2 Monitoring and Alerting

Monitoring and alerting tools to continuously track the performance, availability, and health of the entire system are provided by AWS.

6.3 Backup and Disaster Recovery

AWS provides backup and disaster recovery services for the application.

6.4 Compliance and Regulations

Following GDPR and KVKK closely to ensure our application is compliant with the regulations.

6.5 Feedback and Improvement

Collecting feedback from users and stakeholders to identify areas for improvement and prioritize feature enhancements or bug fixes accordingly.

7. Other Project Elements

7.1. Consideration of Various Factors in Engineering Design

7.1.1 Constraints

Implementation Constraints

- Github and Jira were used to control the development process of our application.
- Vue.js was used for the frontend development.
- Java and Spring framework were used for the backend development.
- Pytorch and Huggingface transformers frameworks were used to develop the AI models.
- FastAPI and Python were used to deploy the AI models.
- PostgreSQL is the database system.
- The backend is deployed to AWS.
- Our machine learning models were trained in the cloud using Google Collab.

- NewsAPI was used to acquire new news articles.

Economic Constraints

- Backend deployment is 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 is necessary and relevant to our system.
- The collected data is not to 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 is English. More languages may be added in the future.

7.1.2 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.

7.2. Ethics and Professional Responsibilities

Ethics and professional responsibilities are essential to our project, "Impartial News." We immensely considered issues such as user privacy, data security, and impartiality during the development of our project. Additionally, one of the crucial points was ethical AI usage, as it is one of the most essential and necessary parts of our project. Some of the vital principles were transparency, accountability, and responsible content creation in the processes of our

project. Throughout our project, professional development was one of the prioritized parts for the ethical standards and best practices.

7.3. Teamwork Details

7.3.1. 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.

7.3.2. 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.

7.3.3. 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.

7.3.4. Meeting objectives

We held weekly meetings about the project, our roadmap, goals, and problems. We united to solve/overcome these. To meet the objectives required by the school and by our supervisors, we worked on an assignment-based strategy. We assigned current problems to each team member related to that line of work and expected each one to find solutions to them. Most of

the time, we did find the solutions to each assignment on our own, but sometimes, we collaborated in small groups to overcome difficulties.

7.4 New Knowledge Acquired and Applied

In this project, we learned a lot of new things. To start with, we learned about the whole AWS infrastructure because all of our systems work on AWS. Then, we explicitly learned how to use the Sagemaker Studio to train the LLama 3 model. After that, we learned how to code responsive web applications to be more device-inclusive. Then, we experimented with our classification model and found some errors. This made us research new ways to do the job, so we learned about ensemble learning and created a multi-model classifier. With all of these coding skills and knowledge, we also gained a lot in the area of soft skills. We learned how to do assignments efficiently, how to collect the results to meet the deadlines, and how to communicate more effectively.

8. Conclusion and Future Work

In conclusion, the design and development of the application have successfully addressed key requirements in usability, security, performance, supportability, scalability, and automation. By prioritizing user experience, implementing necessary security measures, optimizing performance, ensuring cross-platform support, and designing for scalability, the application stands poised to deliver a seamless and reliable experience to its diverse user base. The integration of automation technologies further enhances efficiency and reduces manual intervention, contributing to overall operational effectiveness.

While the current iteration of the application meets established requirements, there are opportunities for future enhancements and expansions. Future work could focus on:

- Enhanced User Personalization: Implementing features to personalize user experience based on individual preferences and behavior patterns.
- Advanced Security Measures: Continuously updating security protocols to adapt to evolving threats and ensure the highest level of user data protection.
- Optimization for Emerging Technologies: Adapting the application to leverage emerging technologies for more advanced GenAI and machine learning news experiences.

- Global Localization: Further refining support for different languages and cultural contexts to cater to a more diverse global audience.
- Integration of Ethical AI Practices: Incorporating ethical considerations into AI algorithms and decision-making processes to mitigate potential biases and ensure fairness.
- Community Engagement and Feedback: Actively soliciting user feedback and engaging with the community to iteratively improve the application based on user needs and preferences.

9. 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

10. References

[1] “Security and compliance - overview of Amazon Web Services,” AWS Security and Compliance, Accessed May. 7, 2024. https://docs.aws.amazon.com/whitepapers/latest/aws-overview/security-and-compliance.html

[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