American University Kyiv

A GOVERNMENT CONSULTATION INFORMATION PLATFORM OF DIGITAL DEMOCRACY

(КОНСУЛЬТАЦІЙНА ІНФОРМАЦІЙНА ПЛАТФОРМА САМОВРЯДУВАННЯ У СФЕРІ ЦИФРОВОЇ ДЕМОКРАТІЇ)

by Denys Lobur

A Capstone Project

Presented in Partial Fulfilment of the Requirements for the Degree

Master

APPROVED BY:

Viktor Putrenko, Dr. Habil., Prof

2025

ACKNOWLEDGEMENT

I would like to express my profound gratitude to the rapidly advancing field of Artificial Intelligence, which has not only inspired the topic of this dissertation but also provided a wealth of resources and tools critical to its completion. The continuous progress in AI technologies has been nothing short of remarkable, transforming numerous sectors and opening up new possibilities for innovation and understanding.

A special thanks to the pioneers and researchers whose relentless pursuit of knowledge and commitment to enhancing AI technologies have made significant contributions to the field. Their work has laid the groundwork for new applications and breakthroughs that empower us to envision a future intertwined with intelligent machines.

Lastly, I would like to acknowledge the community and platforms that foster collaboration and knowledge-sharing in AI, enabling individuals like me to learn, grow, and contribute to this dynamic and ever-evolving discipline. Your influence and impact are deeply appreciated.

TABLE OF CONTENTS

1. AIM OF WORK

The project, "A Government Consultation Information Platform of Digital Democracy" aims to provide an accessible platform for citizens to engage in democratic processes by reviewing, understanding, and providing feedback on proposed laws. This platform will integrate AI-based summarization of bills, voting functionalities, notifications, and analytics to streamline citizen input and governmental response.

2. OVERVIEW

In an era where technology seamlessly integrates with every aspect of human life, the intersection of human activity and artificial intelligence (AI) offers a promising pathway for enhancing civic engagement and legislative efficiency. Democracy [1], one of the cornerstones of our civilization, comes in various forms, predominantly categorized into direct and representative democracy. Direct democracy [2] empowers citizens to participate directly in decision making without intermediary representatives. This model, while fostering high levels of engagement, is often impractical in large, complex societies where the sheer volume and intricacy of issues can overwhelm individuals.

Conversely, representative democracy [3] delegates decision-making authority to elected officials. While more practical, this system often distances the electorate from the specifics of legislation, leading to a disconnect between the people's will and governmental actions. Amid growing volumes of legislative material and complex legal language, the average citizen finds it increasingly challenging to engage with the political process meaningfully.

That’s how the idea of this Capstone project was born - the concept of digital democracy, a paradigm that envisions integrating AI technologies to merge the benefits of direct and representative democracy. In this model, AI acts as an intermediary, utilizing its enormous processing power and natural language capabilities to distill complex legislative documents into concise, readable summaries. By transforming lengthy law projects into digestible bullet points, AI can empower citizens with a clearer understanding of legislative proposals, thus facilitating informed decision-making.

Digital democracy aspires to bridge the gap between the electorate and their representatives by promoting transparency and accessibility. This model could revolutionize civic participation by making the legislative process more comprehensible and immediate for the average citizen. The potential for AI-driven law summarization not only addresses the challenge of complexity in modern governance but also nurtures a more informed and engaged citizenry, thus reinforcing the foundational principles of democracy itself. In this manner, AI does not replace human judgment but rather amplifies human capacity for participation in the democratic process.

3. OBJECTIVES

1. Develop a centralized platform that allows citizens to view summarized versions of law proposals.
2. Implement AI to automatically generate short, understandable summaries of law bills.
3. Enable users to vote on bills and provide feedback with a score from 0 to 5.
4. Incorporate a backend, database, web, and mobile application to support cross-platform access.
5. Create an automated system to notify users of new bills.
6. Analyze voting data to predict public sentiment on proposed legislation, helping the government prioritize law improvements based on citizen feedback.

4. SCOPE AND FEATURES

1. **Bill Summarization and Explanation**
    a. Scrape bill information from official government sources or download it via API.
    b. Use AI to generate short, accessible summaries of bills.
    c. Present summaries in a user-friendly way on both web and mobile applications.
2. **Voting Mechanism**
    a. Allow citizens to rate each bill from 0 (absolutely disagree) to 5 (absolutely agree).
    b. Capture voting data in real-time to provide insights into public opinion.
3. **Notification System**
    a. Notify citizens of new bills through push notifications on web and mobile applications.
    b. Notifications should trigger whenever new bills are scraped or retrieved from official sources.
4. **Public Sentiment Analysis**
    a. Analyze voting data to gauge public sentiment on each bill.
    b. Based on aggregated votes, determine whether a bill should be further discussed in Parliament or sent back for improvements.
5. **Multi-Platform Availability**
    a. Backend services to support both web and mobile frontends.
    b. Database to store bill information, voting data, and user interactions.

5. **HIGH LEVEL DESIGN**

The platform consists of five main parts: namely Backend, Frontend, Data storage, AI summarizer and Sentiment analyzer. Let's consider each of these components one by one.

1. Backend

The backend is a monolith application, represented by a Spring Boot Framework [4] v.3.4.1 (the latest at the time of writing this application) written in the Kotlin programming language v1.9.25. As a build tool the Gradle v8.11.1 is used. The JVM platform (OpenJDK + JRE) v19.0.1. There are following dependencies, that help to provide the required functionality:

- Spring-boot-starter-security
- Spring-boot-starter-web
- Spring-boot-starter-data-jpa
- Jetbrains-kotlin-reflect
- Postgres-postgresql
- Jsonwebtoken-jjwt-api
- Jakarta-validation-api
- Springai-openai-spring-boot-starter

The precise versions of these dependencies are not important as they change frequently. At the time of writing this application I used the latest versions.

The topmost layer of the Backend is represented by **Controller Layer [5]**. The Controller Layer is responsible for handling incoming HTTP requests and returning appropriate HTTP responses. It acts as an interface between the client and the service layer. Controllers map requests to specific service methods and handle the input and output data transformation.

Next goes the **Service Layer.** The Service Layer contains the business logic of the application. It processes the data received from the controllers, performs necessary operations, and interacts with the repository layer if needed. Services are responsible for implementing the core functionality of the application.

Next, **Model Layer.** The Model Layer represents the data structure of the application. It includes entities that map to database tables and data transfer objects (DTOs) used for transferring data between different layers of the application.

Next, **Repository Layer.** The Repository Layer provides an abstraction over the data access layer. It contains interfaces for performing CRUD operations on the database. Repositories typically extend Spring Data JPA repositories, which provide built-in methods for common database operations.

Database. It is not a layer, but rather an integral component that stores the data between and during the application runs. In this case, PostgreSQL is used as the database management system. It holds the data in tables and allows for efficient querying and storage.

External APIs. External APIs are third-party services that the application interacts with to fetch or send data. In this case, the OpenAI API is used to generate responses based on prompts.

Configuration. The Configuration includes files and settings that configure the application. This includes **application.properties** for application-specific settings and **build.gradle.kts** for managing project dependencies and build configurations. Configuration files ensure that the application is set up correctly and can connect to necessary services and databases.

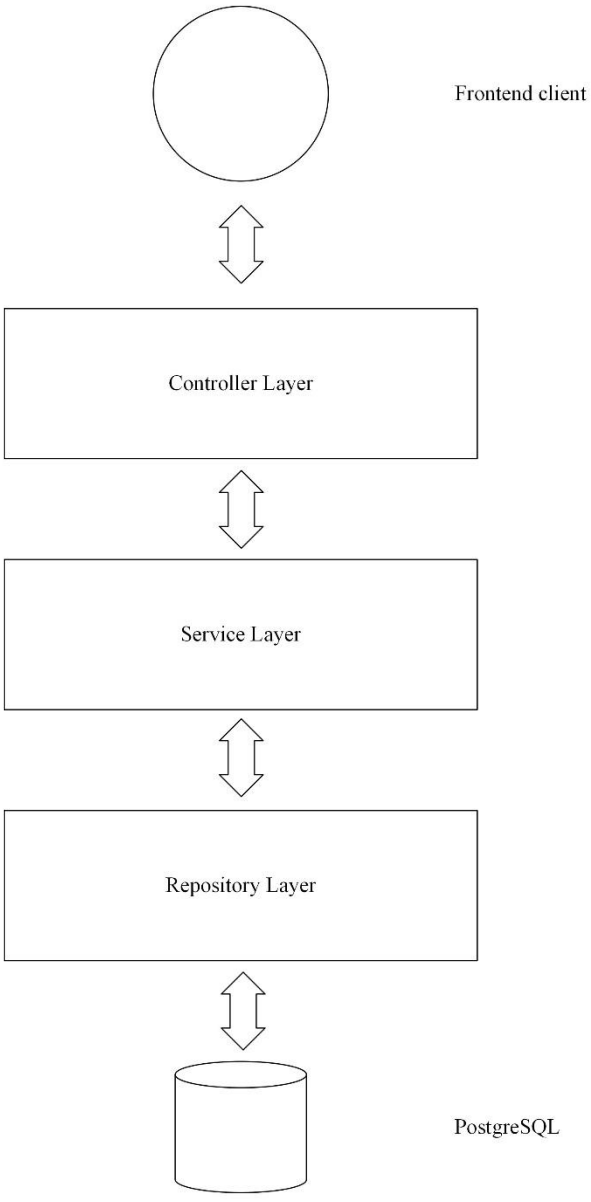

Figure 1. Backend layer diagram

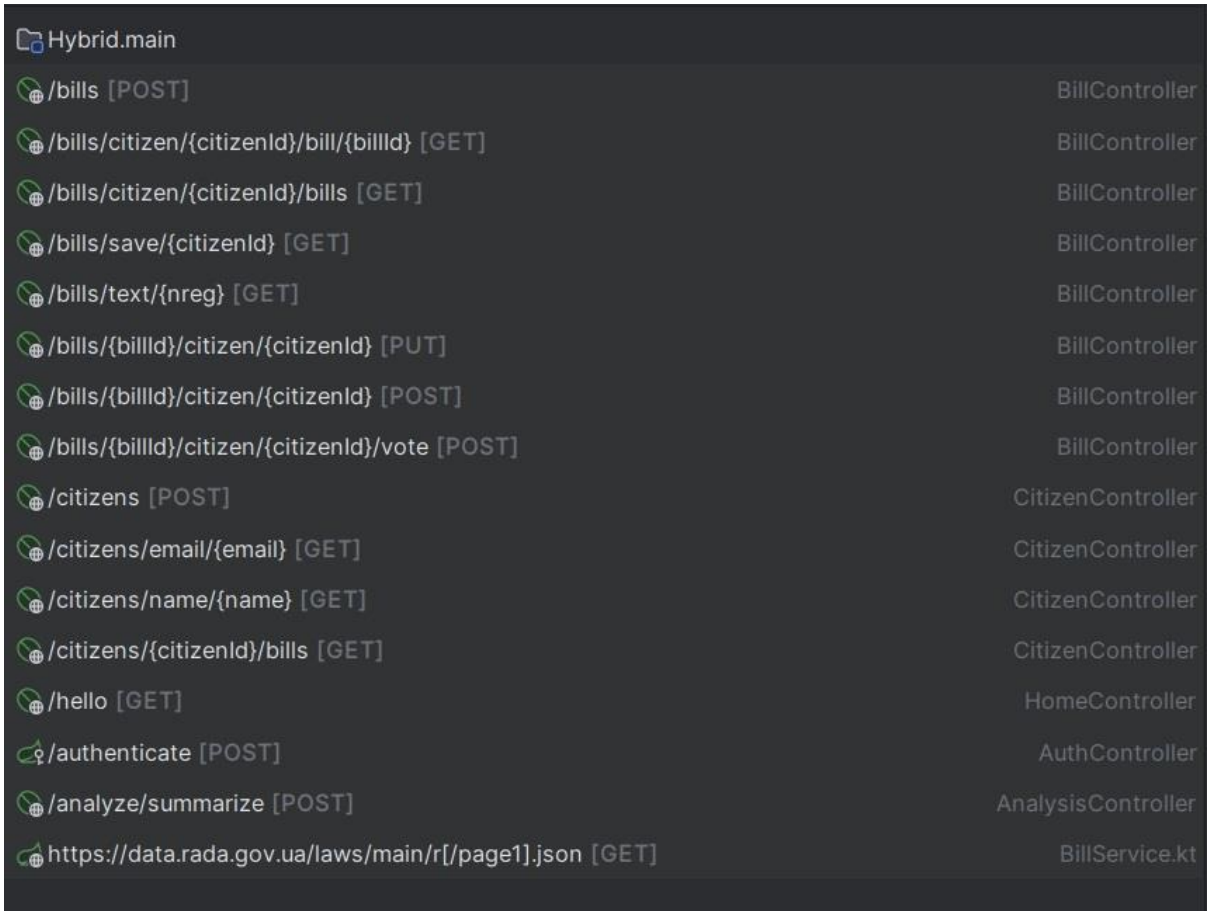

Figure 2. HTTP requests from Frontend to Backend

2. Frontend

The Frontend is represented by an Android application. It is a monolith application, designed with the help of 'Clean Architecture' [6] written in Kotlin v2.0. As a build tool Gradle v8.7.1 is used. The JVM platform (OpenJDK + JRE) v19.0.1. There are following dependencies, that help to provide the required functionality:

- Android.compose
- Androidx.material3
- Android.multidex
- Android.work-runtime
- Retrofit
- Squareup.okhttp3
- Google.code.gson
- Androidx.navigation.compose
- Android.core.spalshscreen
- Google.dagger.hilt-android
- Androidx.hilt-work

The precise versions of these dependencies are not important as they change frequently. At the time of writing this application I used the latest versions.

The Android Frontend application consists of the following layers:

UI Layer (also known as Presentation).

- Screens: Composable functions for different screens (e.g., Login screen, Home screen, etc,) [7].

- ViewModels: Handle UI-related data and business logic, using Hilt for dependency injection.

Domain layer. Defines methods for data operations. It comprises of core business logic and rules of the application.

Data layer. The Data layer is responsible for handling data from external sources, such as databases, web services, or device sensors. It consists of the following subdivisions:

- Repository Implementation: Implements the repository interface, interacts with network and local data sources.
- Network: Retrofit setup for API calls, including AuthInterceptor for adding JWT tokens to requests.
- Models: Data classes for API responses and other data entities.

Utils. Not a layer, but rather a component. IT comprises of the following modules:

- Network: DataState sealed class for representing different states of data (e.g., Success, Error, Loading).
- Security: TokenManager for managing JWT tokens, AuthInterceptor for adding tokens to HTTP requests.

Dependency Injection. Represented by Hilt Modules, which provide dependencies like Context, Retrofit, OkHttpClient, and repositories.

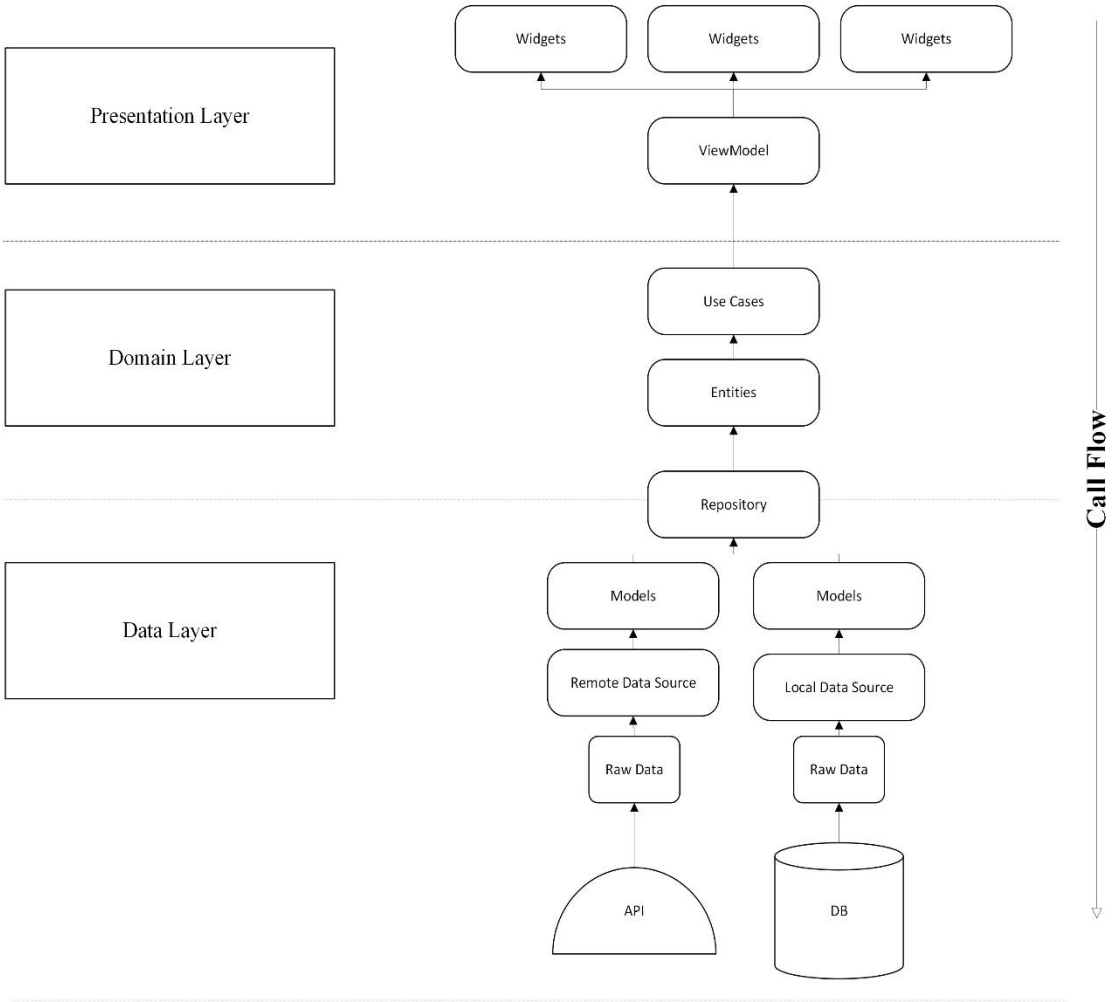

Figure 3. Frontend (Android) layer diagram

3. Data storage

The data storage used in this app is PostgreSQL [8] v15.1, a powerful, open-source object-relational database system. It is used to store application data in tables, allowing for efficient querying and storage. The app interacts with the PostgreSQL database through the **Repository Layer**, which provides an abstraction over the data access layer using Spring Data JPA repositories. This setup allows for easy CRUD operations and database management.

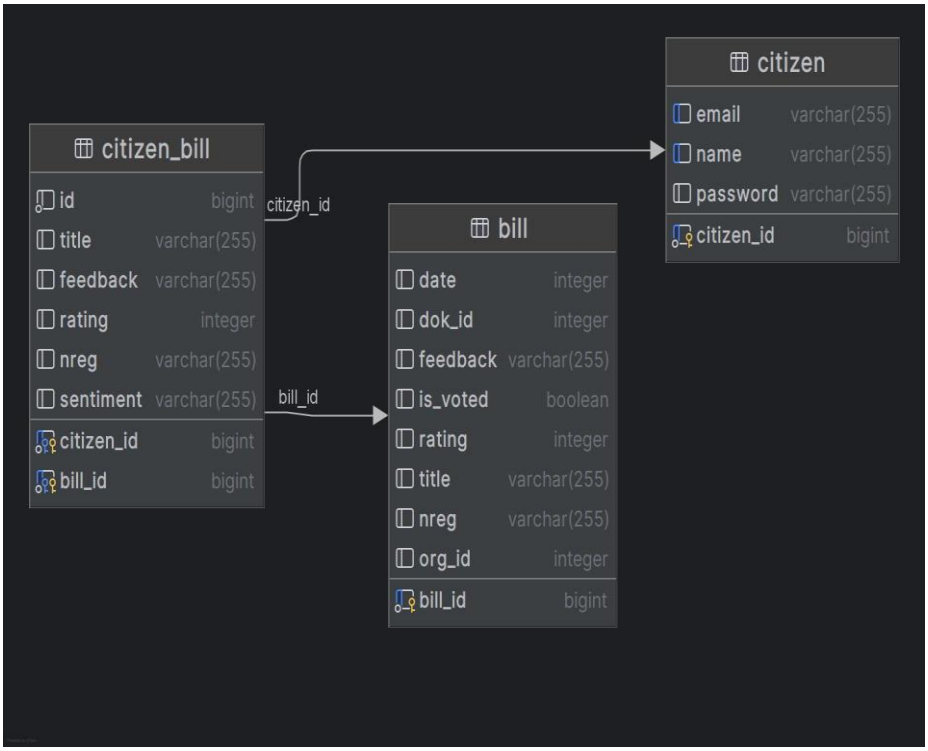

Figure 4. Database tables diagram

4. AI summarizer

The AI Summarizer is an innovative tool integrated into our Spring Kotlin backend, leveraging the capabilities of the **OpenAI API [9]** to distill and interpret complex legal documents from governmental portals. Utilizing the `org.springframework.ai:spring-ai-openai-spring-boot-starter` dependency, this sophisticated system efficiently navigates extensive legal texts, generating concise and coherent summaries while maintaining the essence and intent of the original documents. By doing so, it not only simplifies accessibility to vital legislative information but also enhances understanding of legal provisions and amendments. The AI Law Summarizer interprets the meaning behind each bill, providing users with insightful analyses that highlight the practical implications and potential impact on democratic processes and public policy. This tool empowers citizens, legal professionals, and policymakers by promoting transparency and informed decision-making in democratic governance, thereby bridging the gap between complex legal jargon and public comprehension.

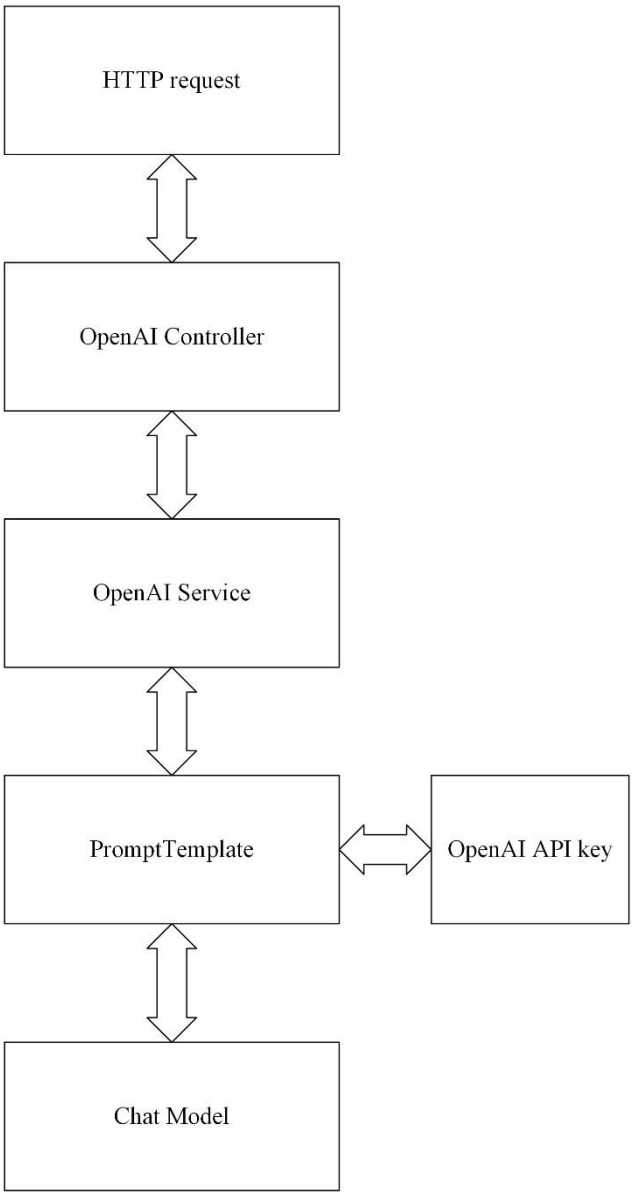

Figure 5. AI Summarizer diagram

5. Sentiment analyzer

The Sentiment Analyzer integrated into our Spring Kotlin Backend is designed to provide comprehensive insights into user feedback and ratings concerning law bills. Leveraging the capabilities of the Open AI API, this tool systematically evaluates textual feedback and star ratings ranging from 1 to 5. By employing advanced natural language processing techniques, the analyzer differentiates between positive, neutral, and negative sentiments, thereby transforming qualitative input into actionable data. This sentiment analysis enhances our understanding of public opinion on legislative matters, facilitating more informed decision-making processes. By combining AI-driven summarization of law bills with sentiment metrics, our platform offers a holistic view of stakeholder perspectives. This

integration not only improves the feedback loop for lawmakers but also strengthens democratic engagement by ensuring that public sentiments are accurately captured and represented. The seamless incorporation into the existing Spring Boot environment ensures robustness, scalability, and ease of deployment, contributing to an automated, data-driven legislative process.

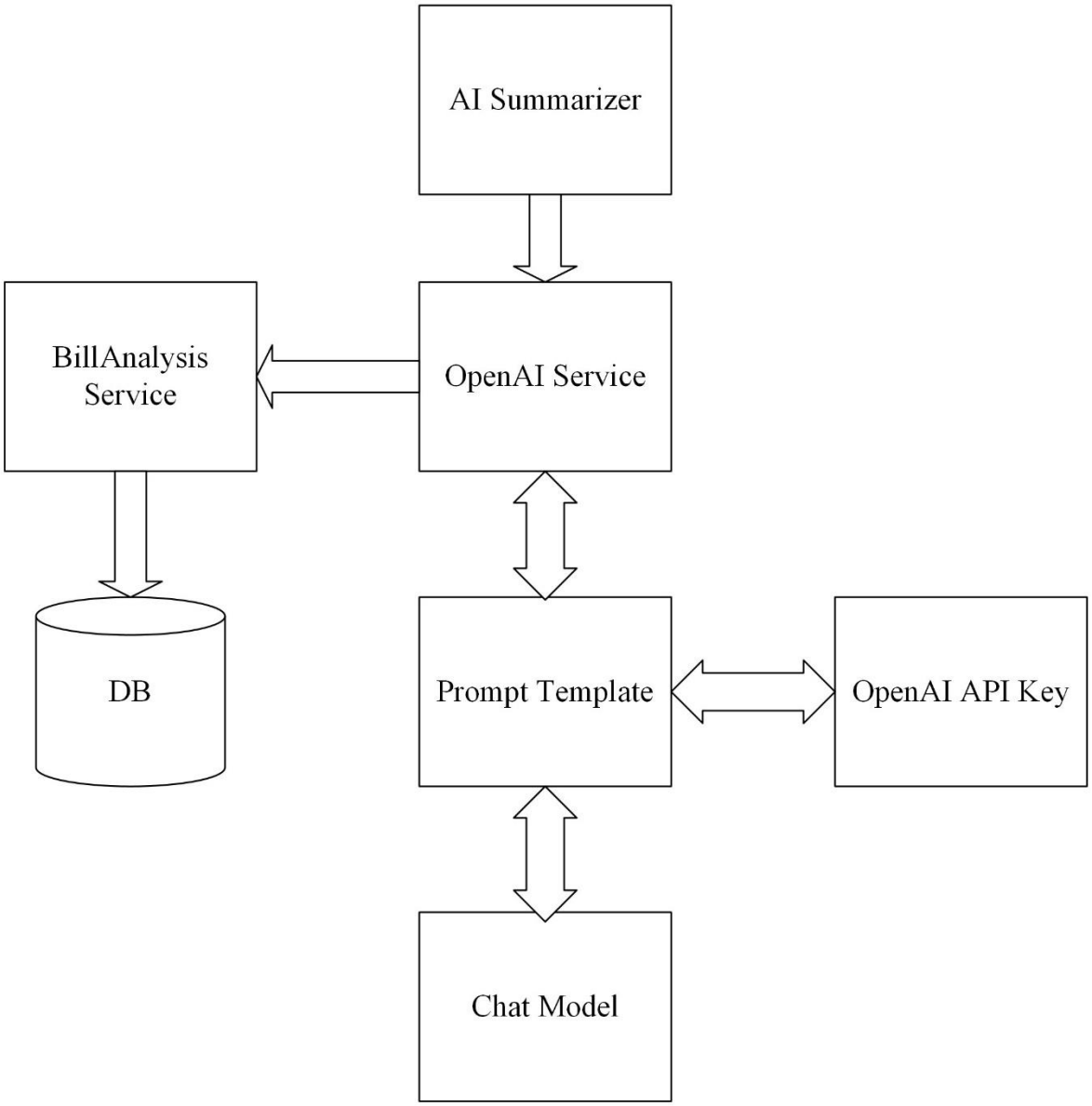

Figure 6. Sentiment Analyzer diagram

6. TECHNOLOGY STACK

- **Frontend**:
  - Web App: React.js (TBD)
  - Mobile App: Android (Ready) & IOS (TBD) native apps (for cross-platform compatibility)
  - Design Tools: Figma for UI/UX prototyping, MS Visio
- **Backend**:
  - Java. Spring Boot (Ready)
  - API integrations for data scraping and official government API handling (Ready)
- **Database**:
  - MongoDB (TBD) or PostgreSQL (Ready) (for structured data)
- **AI/ML**:
  - NLP model for summarization (e.g., OpenAI GPT, BERT) (Ready)
  - Sentiment analysis tools (VADER or TextBlob for initial testing, custom sentiment model for production) (TBD)
- **Notifications**:
  - Firebase Cloud Messaging (FCM) for mobile notifications (TBD)
  - Web push notifications via service workers (TBD)
- **Deployment & DevOps**:
  - Docker for containerization (TBD)
  - AWS or Azure for hosting (TBD)
  - CI/CD tools like Jenkins or GitHub Actions (TBD)

Next figure presents the general overview of the platform in a high-level component diagram. The data flow is shown by directed arrows:

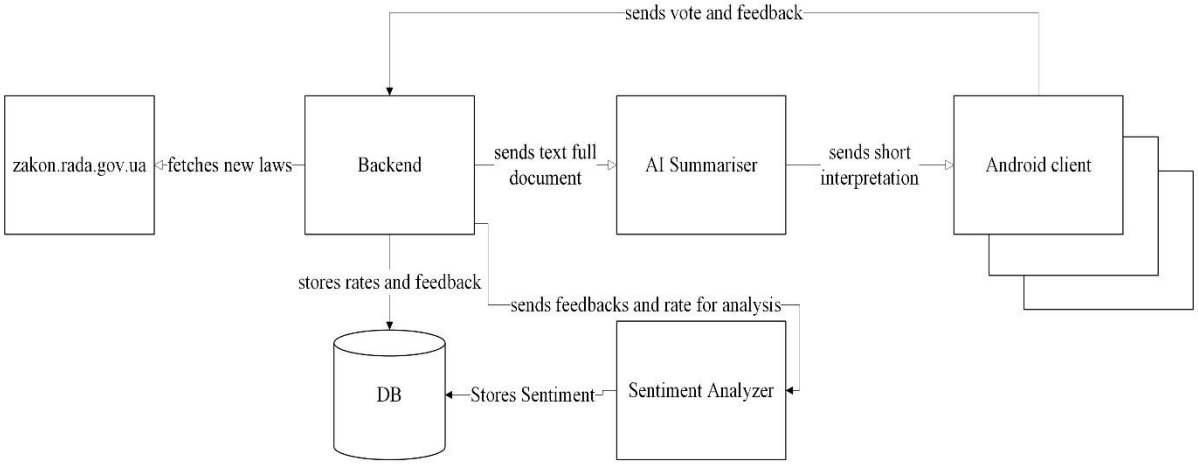

Figure 7. General system diagram

Here is presented the Android UI/UX diagram which shows the general navigation and user experience within the Android application:

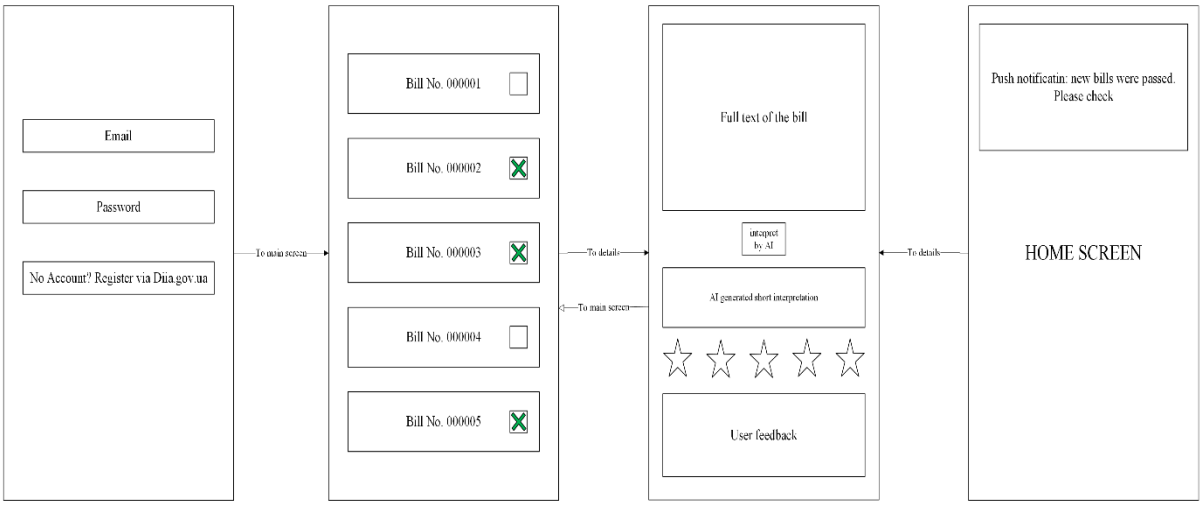

Figure 8. Mobile client (Android) diagram

7. RISK ANALYSIS AND MITIGATION

- **Risk**: Data scraping may be blocked by government websites.
    - **Mitigation**: Prioritize API use; include rate-limiting and retries for scraping.
- **Risk**: AI summarization may generate inaccurate summaries.
    - **Mitigation**: Test model thoroughly, allow for human review in early phases.
- **Risk**: High server load from real-time voting.
    - **Mitigation**: Use load balancing, optimize database queries, and implement caching.

8. EXPECTED OUTCOMES

- Citizens gain transparent access to lawmaking processes.
- The government receives structured public feedback on proposed laws.
- Simplified democratic participation, leading to a more engaged citizenry.

9. EVALUATION METRICS

- **User Engagement**: Track the number of users engaging with bill summaries and voting.
- **System Performance**: Measure response times, especially during peak voting periods.
- **AI Accuracy**: Evaluate summarization model accuracy and adjust as needed.
- **Feedback Accuracy**: Track correlation between citizen feedback and governmental actions on bills.

10. SIMULATION OF VOTING INTERPRETATION AND SENTIMENT ANALYSIS

Here's the result of Digital Democracy at work. First, I created the society, also known as Citizens.

	citizen_id	email	name	password
1	5	user@test.com	Michael Bien	123456
2	6	user2@example.com	John Smith	123456
3	7	user3@demo.com	Adam George	123456
4	8	user4@test.com	Sam Paul	123456
5	3	user6@test.com	Mich Jagger	123456
6	4	user7@test.com	Gram Dench	123456
7	1	user5@test.com	Kate Moss	123456

Figure 9. Citizens of the Digital Democracy society

Disclaimer: storing the passwords AS-IS is dangerous and should not be allowed in the production code. One needs to encode the passwords or use any other secure technics, e.g. jwt token. For simplicity reasons and time constraints I used the password in a not-encoded form. No citizen got harmed.

As soon as we get our Citizens in the database, hence in the project, we can login into the Mobile app using **email** and **password.**

Figure 10. Login screen of Digital Democracy mobile app

Now we need to pull the available law bills from the government web portal: https://zakon.rada.gov.ua/ [10].

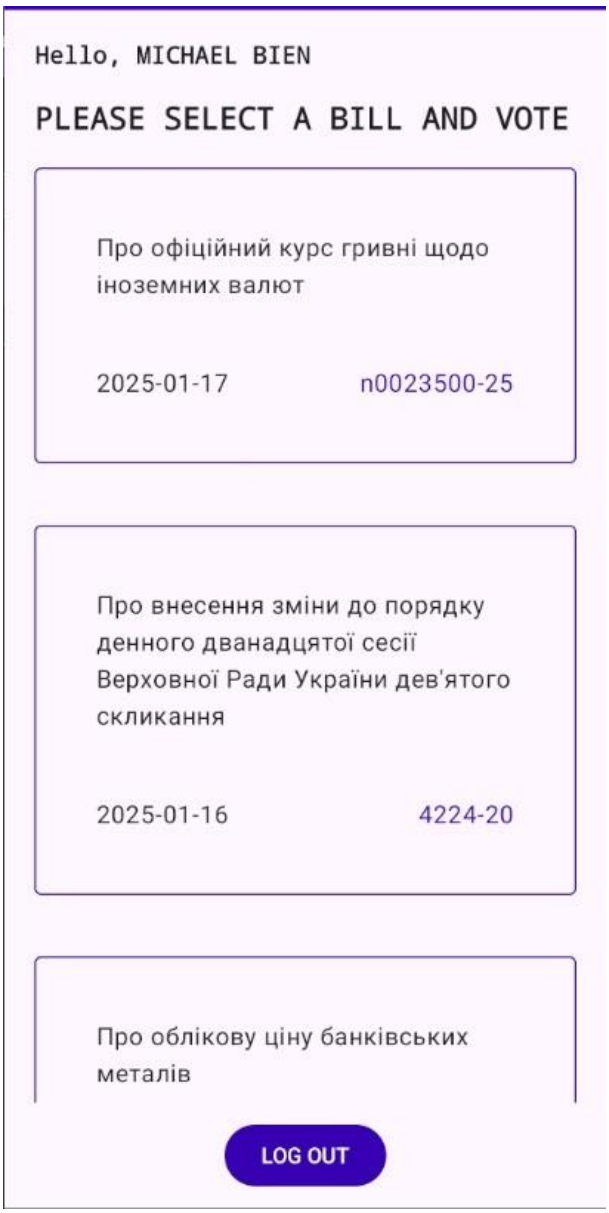

Figure 11. All Bills visible to Citizen

There are different formats, in which the portal provides information to the end users.

Документи		
Текст документа `nreg`	HTML	https://data.rada.gov.ua/laws/show/ `nreg`
Картка документа `nreg`	HTML	https://data.rada.gov.ua/laws/card/ `nreg`
Документ повністю `nreg`	JSON	https://data.rada.gov.ua/laws/show/ `nreg` **.json**
Картка документа `nreg`	JSON	https://data.rada.gov.ua/laws/card/ `nreg` **.json**
Чистий текст документа `nreg`	TXT	https://data.rada.gov.ua/laws/show/ `nreg` **.txt**
Списки документів		
Список поновлених документів	HTML	https://data.rada.gov.ua/laws/main/r
Список найновіших надходжень (за день)	HTML	https://data.rada.gov.ua/laws/main/nn
Список нових надходжень (30 днів)	HTML	https://data.rada.gov.ua/laws/main/n
Список всіх документів (по сторінках)	HTML	https://data.rada.gov.ua/laws/main/a[/page1]
Список `nreg` номерів поновлених документів	TXT	https://data.rada.gov.ua/laws/main/r**.txt**
Список `dokid` номерів поновлених документів	JSON	https://data.rada.gov.ua/laws/main/r/docs**.json**
Список поновлених документів з	JSON	https://data.rada.gov.ua/laws/main/r[/page1]**.json**

Figure 12. Allowed formats of documents in the Government portal

I used JSON format for my needs as I find it more structural for loading the list of the Bills and TXT format for loading the text of specific document. The TXT format is available without any additional requirements from the client side. So, any HTTP client can download a document (Bill) in TXT format just by making a correspondent HTTP request. But with JSON format things are a bit harder. First one needs to register their public IP in the Governmental portal (lawup@rada.gov.ua) [11] and then obtain the token (which works for one day) in order to be able to make HTTP requests in JSON format. All the rules are described here: https://data.rada.gov.ua/open/main/api/page3 [12].

After I downloaded the list of available Bills I stored them into the Bill table in the Database.

	bill_id	date	dok_id	feedback	is_voted	rati…	title	nreg	org_id
1	121	20250117	542033		false	0	Про офіційний курс гривні щодо іноземних валют	n0023500-25	70
2	122	20250116	542079		false	0	Про внесення зміни до порядку денного дванадц…	4224-20	1
3	123	20250116	541964		false	0	Про облікову ціну банківських металів	n0022500-25	70
4	124	20250116	541963		false	0	Про офіційний курс гривні щодо іноземних валют	n0021500-25	70
5	125	20250115	542077		false	0	Про внесення зміни до порядку денного дванадц…	4218-20	1
6	126	20250115	542048		false	0	Про внесення змін до додатка № 5 до Указу Пре…	33/2025	4
7	127	20250115	542047		false	0	Про внесення змін до Положення про проходженн…	32/2025	4
8	120	20250117	542034		false	0	Про облікову ціну банківських металів	n0024500-25	70
9	119	20250120	542054		false	0	Про офіційний курс гривні щодо іноземних валют	n0025500-25	70

Figure 13. Bills table with available Bills for our Citizens

Now we can vote, interpret and analyze every Bill. The voting has two options: **feedback** (textual representation) and **rating** (numerical representation, score 0 - 5). After the Citizen read through the Bill, they may ask AI to interpret it in much shorter and succinct version by pressing the “INTERPRET WITH AI” button. And vote simply pressing ‘SEND FEEDBACK’ button on Bill Details screen of the Mobile app.

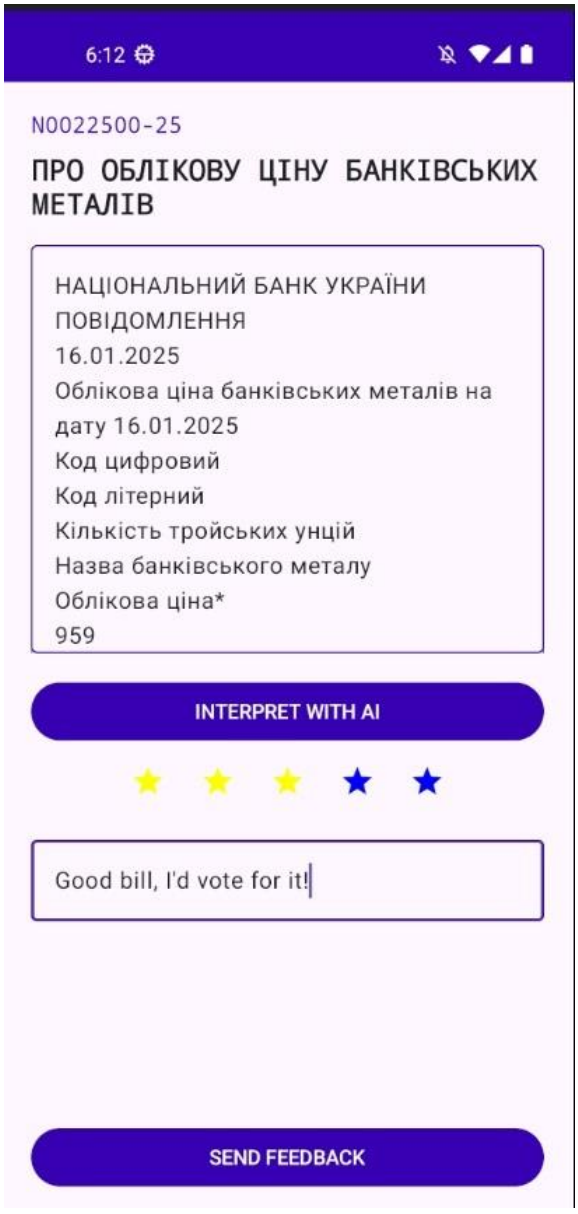

Figure 14. Mobile version of Bill Details screen

As soon as a Citizen voted for the Bill, it gets analyzed by AI and put into another table where the feedback, rating and sentiment are stored. And here I need to say a few words about sentiment analysis.

The sentiment analysis is being done by Spring AI framework (https://spring.io/projects/spring-ai). In simple schematic way it looks like this:

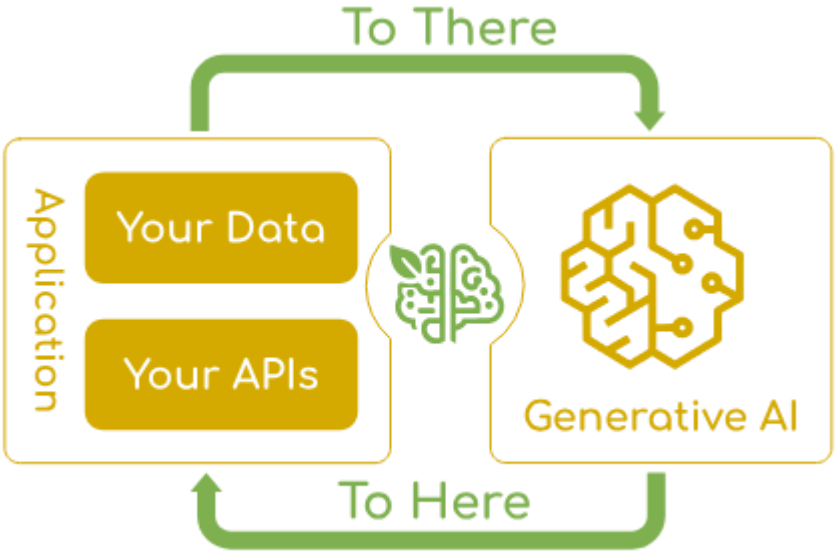

Figure 15. Spring AI diagram

Under the hood Spring AI uses models from OpenAI, Anthropic, Microsoft, Amazon, Google and Ollama. In order to use it for Bill analysis, I made a prompt which looks like this: “Based on feedback and rating, determine the sentiment of the Citizen towards the Bill. Make it one word ‘good’, ‘bad’ or ‘neutral’”. When I feed the feedback and rating to the ChatModel it responds with the correspondent sentiment word, and I write it down into the table:

ci…	bi	ic	title	feedback	rat…	nreg	sentiment
6	124	27	Про офіційний курс гривн…	seems like a good bill. I suppor…	5	n0021500-25	Good
6	121	24	Про офіційний курс гривн…	seems like just another day in p…	2	n0023500-25	Bad
6	122	25	Про внесення зміни до по…	badghsgfshfgjfhgdjfsgdhfsghfgshd…	1	4224-20	Bad
6	119	13	Про офіційний курс гривн…	bad	5	n0025500-25	Neutral
6	125	28	Про внесення зміни до по…	bad	1	4218-20	Bad
5	124	19	Про офіційний курс гривн…	good	4	n0021500-25	Neutral
6	120	23	Про облікову ціну банків…	bad	1	n0024500-25	Bad
7	120	32	Про облікову ціну банків…	I'm not sure in this bill	3	n0024500-25	Neutral
7	127	39	Про внесення змін до Пол…	don't think it is a good one	3	32/2025	Bad
7	126	38	Про внесення змін до дод…	sounds fine	4	33/2025	Good
7	125	37	Про внесення зміни до по…	horrible law	1	4218-20	Bad
6	127	30	Про внесення змін до Пол…	good	4	32/2025	Neutral
7	124	36	Про офіційний курс гривн…		0	n0021500-25	Neutral
6	126	29	Про внесення змін до дод…	bad	2	33/2025	Neutral
7	122	34	Про внесення зміни до по…		0	4224-20	Neutral
7	121	33	Про офіційний курс гривн…		0	n0023500-25	Neutral
7	119	31	Про офіційний курс гривн…	bad	2	n0025500-25	Neutral
7	123	35	Про облікову ціну банків…		0	n0022500-25	Neutral

Figure 16. Table to store feedback, rating and sentiment of each Bill

After a certain number of Citizens vote for particular Bills, we may see the sentiment picture simply filtering through the **bill_id** and **sentiment** columns. For this task I used Intellij IDEA IDE ultimate version.

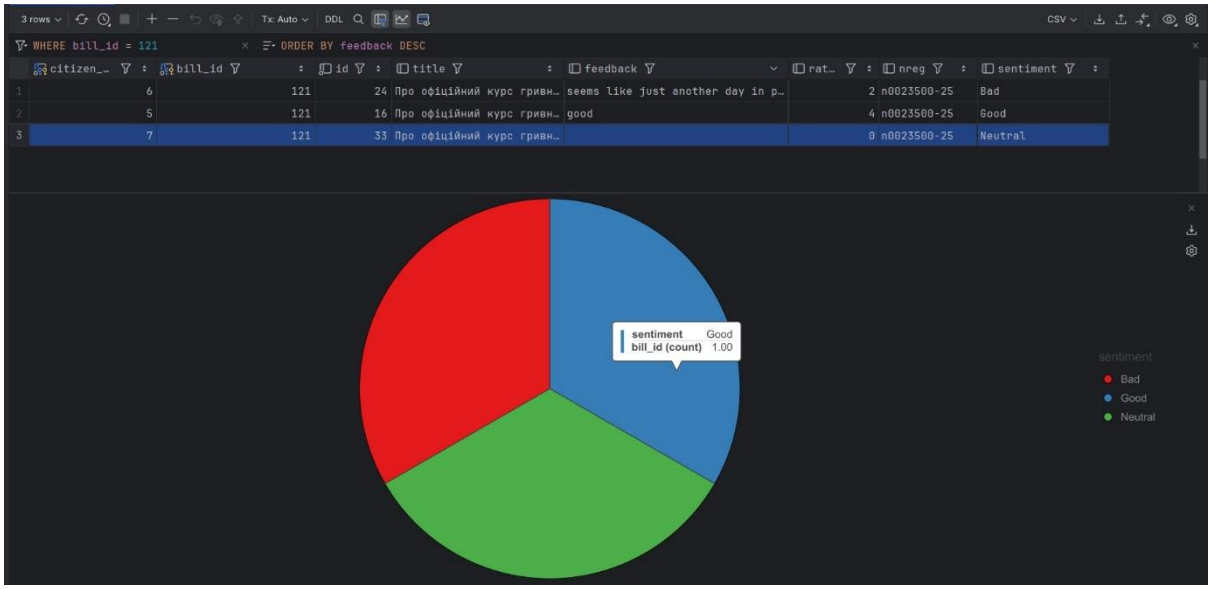

Figure 17. Sentiment analysis of Bill 121

The sentiment analysis of **Bill 121** showed that 1 Citizen out of 3 treats it as Bad, 1 Citizen treats it as Good, and 1 Citizen treats it as Neutral.

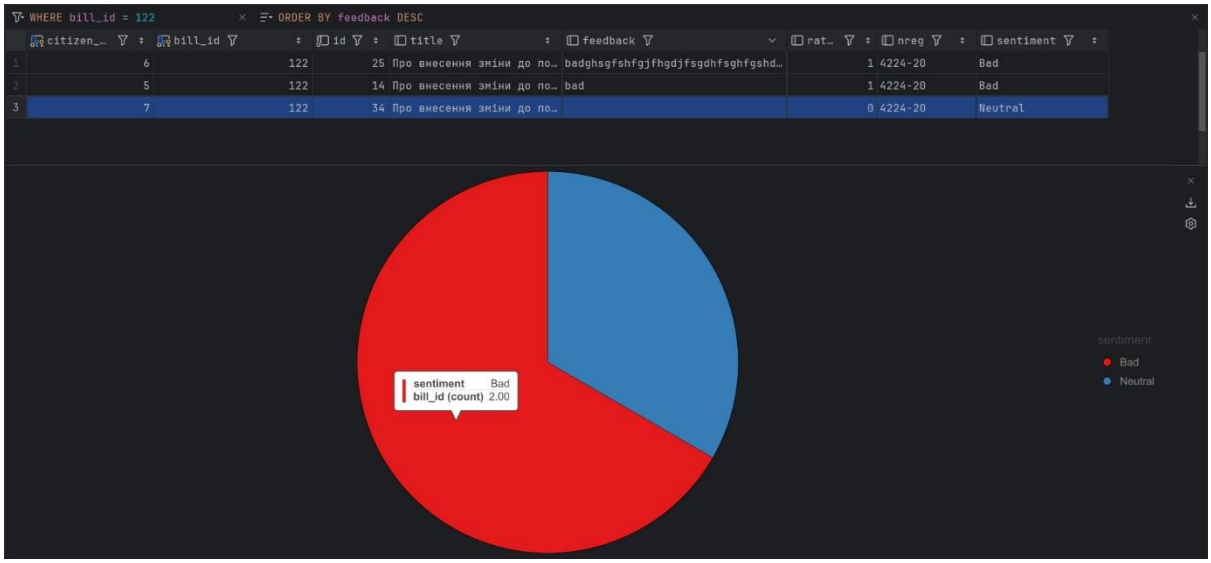

Figure 18. Sentiment analysis of Bill 122

The sentiment analysis of **Bill 122** showed that 1 Citizen out of 3 treats it as Neutral, 2 Citizens treat it as Bad.

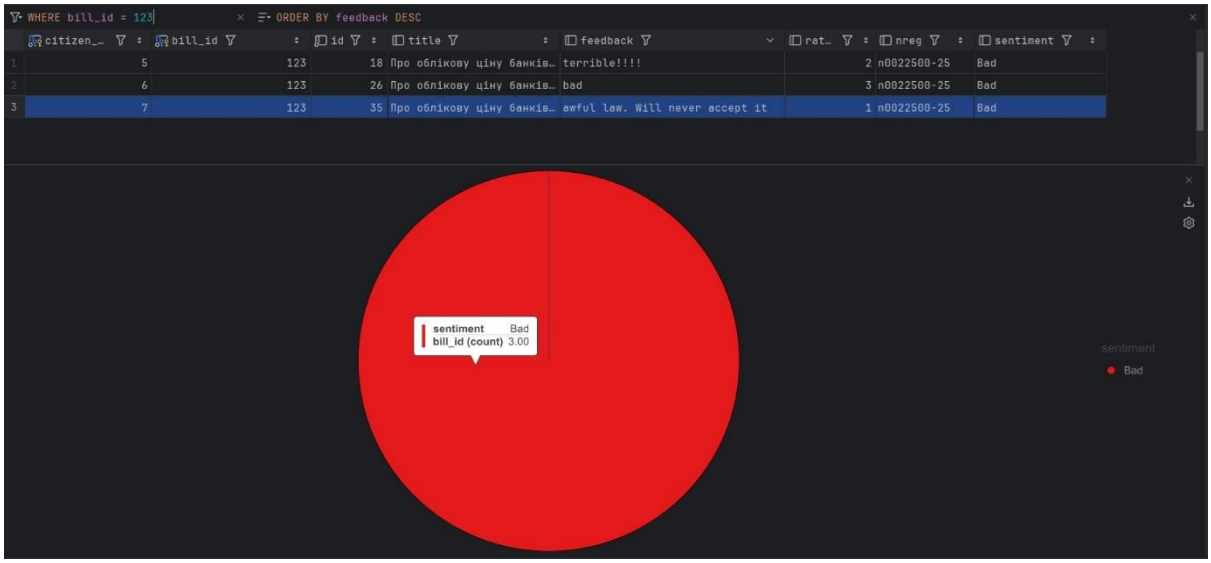

Figure 19. Sentiment analysis of Bill 123

All three Citizens treat Bill 123 as Bad. Of course, if there were more Citizens, the results would be more precise and granular. But a society of 3 Citizens is pretty enough to demonstrate the features of the platform for the Digital Democracy project.

11. LIST OF ABBREVIATIONS AND ACTORS

AI – Artificial Intelligence
DB – Database
TBD – To be done
CITIZEN – a user of the platform
BILL – a law bill which will be interpreted, voted and analyzed
SENTIMENT – generalization of Citizen treatment of the Bill

12. LIST OF FIGURES

13. REFERENCES

1. Wikipedia. https://en.wikipedia.org/wiki/Democracy
2. Wikipedia. Direct democracy. https://en.wikipedia.org/wiki/Direct_democracy
3. Wikipedia. Representative democracy. https://en.wikipedia.org/wiki/Representative_democracy
4. Walls C. - Spring in Action. 6 ed. - 2022 https://www.manning.com/books/spring-in-action-fifth-edition
5. Spring Boot CRUD Example https://rameshfadatare.medium.com/spring-boot-crud-example-with-h2-database-step-by-step-guide-921d41640bc0
6. Clean Architecture https://medium.com/@DrunknCode/clean-architecture-simplified-and-in-depth-guide-026333c54454
7. Android Compose. https://developer.android.com/compose
8. PostgreSQL documentation https://www.postgresql.org/
9. Open API https://chatgpt.com/
10. Офіційний вебпортал парламенту України https://zakon.rada.gov.ua/
11. Administrator of Governmental portal lawup@rada.gov.ua
12. General rules of pulling the official law Bills https://data.rada.gov.ua/open/main/api/page3