THOMPSON RIVERS UNIVERSITY

SENG 3210 – Applied Software Engineering

Android Voting App

Ruth Befikadu (T00696672)

Ryan Martin (T00549875)

Shaylee Broadfoot (T00551934)

March 4, 2024

Table of Contents

List of Figures

List of Tables

1 Introduction

Nowadays, things are being more digitized and the ever-growing technology makes even daily tasks more convenient. Especially with getting the youth to participate in major issues and decisions, having this method of involvement might be key in getting them to be involved [1]. In addition to that, online voting could be beneficial to those who are unable to go to a physics site to vote because of an illness or injury.

With the increase in the use of smartphones and digital platforms for communication and engagement, developing an android voting app aligns perfectly with the evolving technologies. This app aims to bridge the gap between users and the decision-making process, empowering individuals to participate in governance regardless of their physical limitations or geographical constraints. By exploiting the availability of smartphones and the accessibility of the internet, this app promises to revolutionize the way we engage with democracy and decision making, making it more inclusive and responsive to the needs of all members of society.

Online voting presents a promising avenue to address barriers faced by individuals unable to physically access polling stations due to illness or injury. By leveraging the accessibility and convenience of mobile technology, this app aims to empower all members of society, ensuring that every voice is heard and accounted for in matters affecting their communities and beyond.

Moreover, the implementation of online voting offers a solution to avoid the risks of fraud and manipulation inherent in traditional voting methods. With stringent security measures and encryption protocols in place, the app safeguards the integrity of the electoral process, preventing unauthorized access and tampering of ballots. By digitizing and automating the voting and counting procedures, human error and bias are significantly reduced, ensuring that election outcomes accurately reflect the will of the electorate.

In conclusion, the development of this android voting app represents a significant step towards modernizing democratic processes and promoting active civic engagement. By leveraging technology to facilitate accessible and user-friendly voting solutions, we aspire to bridge the gap between citizens and decision-makers, ensuring that everyone's voice is heard and valued. This will benefit society in a way that all voices are heard and accounted for when making any decision that could influence them. This solution will incorporate an easy user interface with proper documentation to help users increase their involvement in their community, workplace and country. This report is a full documentation of the process of making this system.

2 Design Problem

2.1 Problem Definition

Traditional voting methods face challenges like limited access, exclusivity, and susceptibility to manipulation. Additionally, there's a growing need for flexible voting solutions that go beyond formal elections to cover casual scenarios like class participation. Unexpected events such as illness or injury may also hinder physical voting participation. To tackle these issues, there's a demand for a user-friendly voting platform that caters to both formal and casual voting needs while ensuring accessibility, inclusivity, and security for all users. This project aims to develop such a platform, utilizing mobile technology to offer easy, fair, and convenient voting experiences across different situations.

2.2 Design Requirements

2.2.1 Functions

User Registration and Login:

- Users should be able to create accounts securely.
- Users should be able to log in using their credentials.

Poll Creation:

- Administrators should have the ability to create new polls with defined topics.
- Polls should include a title, description, and options for voting.

Poll Voting:

- Registered users should be able to vote on existing polls.
- Each user should be allowed to vote only once per poll.

Real-time Dashboard:

- Administrators should have access to a dashboard displaying real-time statistics of poll results.
- Dashboard should provide information such as total votes, vote distribution, and trending polls.

Mobile Responsiveness:

- The application should be optimized for all mobile devices to provide a quality user experience across different screen sizes.

2.2.2 Objectives

Performance:

- The application should load quickly and respond to user interactions without noticeable delays.
- Database queries and data retrieval should be efficient to minimize loading times.

Security:

- User data should be encrypted to protect sensitive information.
- Authentication mechanisms should be robust to prevent unauthorized access.

Modifiability:

- The application should be designed using modular and extensible architecture to accommodate future updates and enhancements.
- Changes to functionalities or database schema should be implemented with minimal disruption to the existing system.

Compatibility:

- The application should be compatible with Android devices of all sizes and recent software updates.

Scalability:

- The system should be built with expansion in mind. It must be able to support from thousands to millions of reported cases in the case it grows suddenly.

Maintainability:

- VoxChoice should facilitate the addition of new UI components with minimal development effort.

2.2.3 Non-functional requirements and constraints

Budget/Economical factor:

- The system should be affordable and give equal value in functionality compared to the effort and money that was invested in it.

Compatibility:

- The system must be compatible with android devices running Android API level 19 (KitKat) or higher.

Guidelines and regulations:

- The application system must work within the applicable privacy and security guidelines. It has to adhere to laws and regulations within the area, province or country it is being used in.

Reusability:

- The application must be able to be reused in many other voting scenarios, with 20% of its architecture being compatible and useful as a significant part of a new system.

Accessibility:

- The app should be accessible to users with disabilities or just inabilities. It needs to be easy to use with features like a clear font.

3 Solution

In this section, you will detail various solutions generated during your team's brainstorming sessions for project implementation. Not all solutions may encompass all desired features, and some may not fully meet the constraints. These solutions emerge as your team explores ways to implement all features within the specified constraints. Ultimately, you'll choose a solution that, in your assessment, incorporates all necessary features and adheres to the constraints. It's crucial to bear in mind that the process of engineering design is iterative!

3.1 Solution 1: Monolithic with Local Storage

Write a brief description of your first solution and provide the reasons for not selecting this one.

You can use the component diagram, sequence diagram, and class diagram.

This solution follows a modular monolithic architecture with SQLite database integration. The Android voting app is developed using Android Studio and Java, with separate modules for user management, poll creation, voting, real-time analytics, and user profile management. A single SQLite database instance is used for storing all data, providing offline support and compatibility with Android Studio.

Advantages:

- *Offline support:* Users can use the app even without an internet connection, with data synchronization occurring once connectivity is restored.
- *Compatibility:* SQLite is natively supported in Android development, ensuring compatibility with Android Studio and Java.
- *Data consistency:* SQLite database provides ACID (Atomicity, Consistency, Isolation, Durability) properties, ensuring data consistency and integrity.

Challenges:

- *Limited scalability:* Monolithic architecture may face challenges in scaling with increasing complexity and size of the application.
- *Maintenance:* Changes in one module may impact others, requiring careful coordination and testing to ensure consistency.
- *Performance:* Performance may degrade as the size of the SQLite database grows, especially on devices with limited resources.

3.2 Solution 2: Microservices with Local Storage

This solution is an improved solution but might not be the final solution that you select. Give a brief description of this solution here.

You can use the component diagram, sequence diagram, and class diagram.

This solution incorporates a microservices architecture, incorporating local storage for data management. Developed using Android Studio and Java, this solution divides the Android voting app into independent microservices for user management, poll creation, voting, real-time analytics, and user profile management. Each microservice operates with its SQLite database instance to promote data isolation and scalability. This architecture offers a balance between modularity and scalability and caters to the evolving needs of the project. Communication between microservices occurs through direct method calls within the application, ensuring simplicity and compatibility with the development environment.

Advantages:

- *Scalability:* Microservices architecture enables independent scaling of individual services, ensuring optimal resource utilization and performance.
- *Flexibility:* Decoupled services allow for independent development and deployment, providing flexibility in adopting new technologies and updating specific functionalities.
- *Fault Isolation:* Failures in one microservice do not affect the entire application, improving overall resilience and fault tolerance.

Challenges:

- *Complexity:* Managing multiple services and their interactions may introduce complexity in deployment, monitoring, and troubleshooting.
- *Communication Overhead:* Inter-service communication via RESTful APIs may introduce latency and overhead, impacting performance.
- *Operational Complexity:* Deployment and orchestration of microservices require additional operational overhead and infrastructure management.

3.3 Final Solution: Firebase Realtime Database Integration

This solution is the final solution. Explain why it is better than other solutions. You may use a table for comparison purposes. After providing the reason for selecting this solution, detail it below.

The final solution for the Android voting app project adopts a Modular Firebase Realtime Database Integration approach. This solution combines Firebase Realtime Database with a modular architecture to create a scalable, maintainable, and flexible application. The application is developed using Android Studio and Java, with separate modules for user management, poll creation, voting, analytics, and profile management. Each module operates independently, responsible for specific functionalities, while Firebase Realtime Database ensures real-time synchronization of data across devices.

Advantages:

- *Improved Modularity:* The modular architecture enhances code maintainability and extensibility. Each module operates independently, facilitating easier debugging, testing, and future updates.

- *Real-time Data Synchronization:* Firebase Realtime Database ensures instant updates to data across all connected devices, providing users with real-time information on poll results and user activities.
- *Scalability:* The combination of Firebase Realtime Database and modular architecture allows for scalable deployment. Each module can be scaled independently, ensuring optimal performance even under high user loads.
- *Flexibility:* The modular design allows for incremental updates and enhancements to the application. New features can be seamlessly integrated or existing ones modified with minimal disruption to other modules, providing flexibility in adapting to changing requirements.
- *Rapid Development:* Leveraging Firebase SDKs and libraries accelerates the development process, enabling rapid prototyping and iteration of features.

Challenges:

- *Initial Complexity:* Implementing a modular architecture with Firebase Realtime Database integration may introduce additional complexity during the initial development phase. Proper planning and design are crucial to ensure effective module interactions and data synchronization.
- *Learning Curve:* Developers may require time to familiarize themselves with Firebase Realtime Database and modular development practices, which could impact project timelines.
- *Integration Challenges:* Ensuring seamless integration between Firebase Realtime Database and the modular architecture may pose challenges, particularly in managing data synchronization and consistency across modules.

3.3.1 Features and the software architecture

Discuss all the features of your final solution. Describe the functionalities of the top-level components and how they will be used for enabling those features. The product features may be tabulated (with a title) for improved comprehension. Use component diagrams to model the internal structures (i.e., sub-components or second-level components) of two major components. Describe the functionalities of the sub-components and the interactions (e.g., the interfaces) between the sub-components. Explain the interfaces between the top-level architecture and the internal structures (i.e., explaining how the internal structures interact with other top-level components).

Modules:

Log In:

- The Log In module allows users to authenticate and access their accounts securely. Users will enter their credentials (username/email and password) to log in to the application. This module verifies the user's identity and allows access to the app's features based on their role (user or administrator).

Create an Account:

- The Create an Account module enables new users to register for the application. Users will provide necessary information such as username, email, and password to create their accounts. This module ensures secure account creation and could possibly include features like password strength validation and email verification.

Dashboard:

- The Dashboard module serves as the main interface for users and administrators to access essential information and features of the application. For users, the dashboard may display relevant polls and voting options. Administrators will have access to additional functionalities such as poll management.

Poll Management (Admin):

- The Poll Management module empowers administrators to create, edit, and delete polls within the application. Administrators can define poll topics, descriptions, and voting options. This module ensures that administrators have full control over the creation and management of polls to engage users effectively.

Poll Voting (User):

- The Poll Voting module allows registered users to participate in polls by casting their votes. Users can view available polls and select their preferred options. This module ensures a friendly voting experience for users and provides real-time feedback on their choice.

Real-time Statistics (Admin):

- The Real-time Statistics module provides administrators with statistics regarding poll activities within the application. Administrators can view real-time updates on total votes, vote distribution, and more. This module assists administrators in making informed decisions and viewing user engagement.

User Profile:

- The User Profile module enables users to view and manage their profile information within the application. Users can update their personal details, change passwords, and customize preferences. This module ensures a personalized experience for users.

Database Schema:

User Table:

- user_id (Primary Key)
- username
- email
- password
- role

Poll Table:

- poll_id (Primary Key)
- title
- description

- creator_id (Foreign Key referencing user_id)
- date_created
- options

Vote Table:

- vote_id (Primary Key)
- poll_id (Foreign Key referencing poll_id)
- user_id (Foreign Key referencing user_id)
- Voted_option

3.3.2 The system interfaces

Describe the temporal events (i.e., the time-triggered events) and the signal events (i.e., events received from external components) for the intended application. Describe the expected response of the system to each event.

3.3.3 The user interface design

Design the user interface components. Describe the user interface components, the possible business events, and the responses to the triggered events.

3.3.4 The requirements traceability matrix

List the system's requirements and map the requirements to the corresponding design component, code component (e.g., java class file or XML configuration file), and the required testing scenario.

3.3.5 Environmental, Societal, Safety, and Economic Considerations

Explain how your design project considered environmental, societal, and economic considerations. It may include how your implementation has positive contributions to the environment and society. What type of financial decisions did you make? How did you make sure that the implementation is safe to use?

3.3.5.1 Environmental considerations

Explain how your design project considered environmental considerations.

3.3.5.2 Societal considerations

Explain how your design project considered societal considerations.

3.3.5.3 Safety considerations

Explain how your design project considered safety considerations.

3.3.5.4 Economic considerations

Explain how your design project considered economic considerations.

3.3.6 Limitations

Every product has some limitations, so is the case with your design solution. Highlight some of the limitations of your implementation here.

4 Teamwork

4.1 Meeting 1

Time: February 16, 2024, 11:00 am to 1:00 pm

Agenda: Distribution of project tasks and brainstorming

Team Member	Previous Task	Completion State	Next Task
Ruth Befikadu	N/A	100%	Code main voting screen
Ryan Martin	N/A	100%	Create backend code to enable users to add a voting option
Shaylee Broadfoot	N/A	100%	Code main login screen to the app

Decisions Made:

- Team members will look into the feasibility of implementing a cloud based database and researched the best option to be Firebase Realtime Database
- The app will integrate SQLite as the local database for things that do not require syncing across devices.
- The rough schema of the desired database.
- The desired modules that the app should have.

Task Distribution:

Ruth Befikadu:

- *Task Completed:* Ruth's initial task was to create the connection between Android Studio and Github.
- Next Task: Ruth's next task is to design and implement the poll voting module for users. This screen will be the one the user uses to vote. It will consist of a topic to vote on and at least four options to select.

Ryan Martin:

- *Task Completed:* Ryan's initial task was to set up a meeting time, make sure all parties attended and to go over initial tasks and the next steps towards project completion.
- *Next Task:* Ryan's next task is to create the backend code for the administrator to add an option to vote on. This will consist of an administrator selecting the "add voting option" option which will add a cell to the screen that they can type into.

Shaylee Broadfoot:

- *Task Completed:* Shaylee's initial task was to create the GitHub repository.
- *Next Task:* Shaylee's next step is to implement the user registration and login functionality. This will include inputs for user details including role selection (admin or not), and submission.

4.2 Meeting 2

Time: February 29, 2024, 10:00 am to 11:00 am

Agenda: Preliminary coding and deliverable 2 task distribution

Team Member	Previous Task	Completion State	Next Task
Ruth Befikadu	Code main voting screen	100%	UI design
Ryan Martin	Create backend code to enable users to add a voting option	100%	UI implementation
Shaylee Broadfoot	Code main login screen to the app	100%	Layout coding

Decisions Made:

- The general color palette that will be used for the UI
- The features and interactions that each activity will contain (e.g. buttons, tables, etc.)
- The general visual design of the features and interactions for each activity

4.3 Meeting 3

Time: March

Agenda:

Team Member	Previous Task	Completion State	Next Task
Ruth Befikadu	UI design		
Ryan Martin	UI implementation		
Shaylee Broadfoot	Layout coding		

Decisions Made:

-

4.4 Meeting 4

Provide a similar description here.

5 Conclusion and Future Work

- Provide a concise summary of your accomplishments, outlining the design functions and objectives successfully achieved while adhering to the specified constraints.
- In consideration of the limitations inherent in the application design, offer recommendations for potential enhancements in future iterations of the design.

6 References

[1] “Voting app aims to get young Canadians to polls,” *CBC*, Jul. 09, 2015. https://www.cbc.ca/news/politics/voting-app-aims-to-get-young-canadians-to-polls-1.3144568 (accessed Mar. 04, 2024).

Appendix

If you want to provide any additional information, use this appendix.