



DeviceInfo Hub

¹Pratheeksha P, ²Rashmitha, ³Punyashree, ⁴Thilak Raj, ⁵Alwyn Edison Mandonca.

¹Student, ²Student, ³Student, ⁴Student, ⁵Assistant Professor,

¹Computer Science and Engineering,

¹Srinivas Institute of Technology, Mangalore, India

Abstract: This project aims to develop a cross-platform mobile application using React Native, designed to provide users with specific details about some specific devices such as mobile phones, laptops, televisions, refrigerators, washing machines in a single platform. The app will feature up-to date information on device specifications, brand, release dates, warranty durations. By aggregating data from various manufacturers and brands, the app will serve as a one-stop resource for users to stay informed about the latest devices, their features. With an intuitive and user-friendly interface, the app will be compatible with both Android and iOS platforms, making it accessible to a broad audience. This solution seeks to simplify the process of tracking device information, helps users to make correct decisions when purchasing or maintaining electronic products.

Index Terms – Device information, Specific features, Warranty tracking, React Native, Firebase, user interface, devices, front-end, data collection.

I. INTRODUCTION

In today's fast-paced world, electronic devices have become an integral part of daily life, catering to personal, professional, and household needs. With the rapid advancements in technology and the continuous influx of new devices, keeping track of their specifications, features, and service agreements can be overwhelming for users. Recognizing this challenge, Device Info Hub emerges as a comprehensive solution to simplify the way users' access and manage information about electronic devices.

Device Info Hub is a cross-platform mobile application built using React Native, designed to provide users with detailed and up-to-date information about a wide array of electronic devices, such as mobile phones, laptops, televisions, refrigerators, washing machines. The app offers users a centralized repository of device specifications, release dates, warranty periods, and service agreements.

II. PROBLEM STATEMENT

The existing ways to access the details about electronic devices are fragmented and often inefficient, relying heavily on web searches, manufacturer websites, and third-party reviews. These approaches are time-consuming and require users to visit multiple sources to gather comprehensive details about a specific device. For instance, if a user wants to compare smartphones, they need to browse through different websites to check specifications, read reviews, and find information on warranties or service agreements. This scattered approach consumes more time and also increases the chances of missing critical details necessary for making informed decisions.

Another significant limitation of the current system is the lack of centralized and up-to-date information. Data on device specifications, release dates, and warranties are often outdated or incomplete, leading to confusion for users. The shortcomings of the current system emphasize the need for a comprehensive mobile application that aggregates accurate and up-to-date information on electronic devices, while also providing features like warranty tracking. By addressing these challenges, the proposed solution aims to simplify the process of discovering, comparing, and maintaining electronic products in a seamless and efficient manner.

III. METHODOLOGY

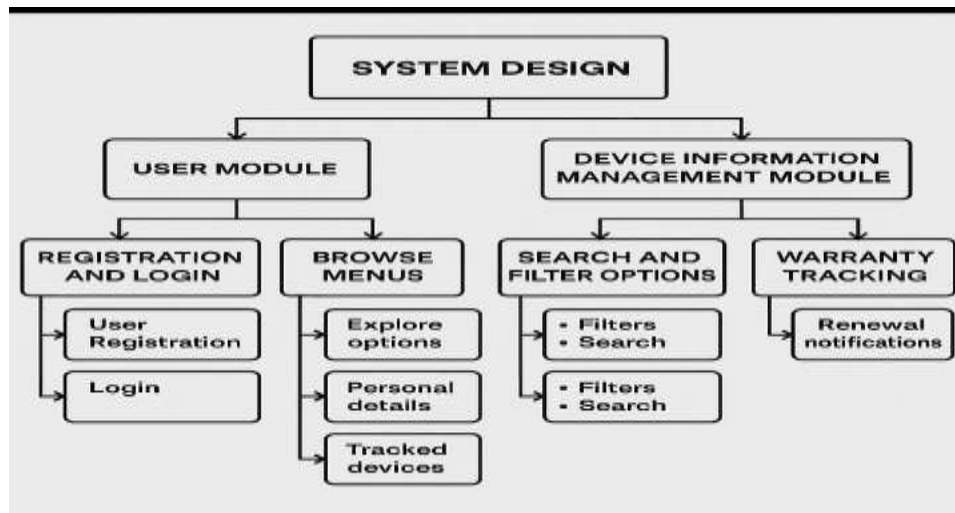
The project develops a mobile application system that enables users to interact with a device information platform through a structured and secure flow of data. The system consists of key external entities such as users, the mobile application, and the device database. The application supports several core processes including user authentication, browsing device categories, searching for devices, and exiting the application.

Data flows encompass user authentication where users input credentials that the system verifies to grant access; device exploration where users can explore device categories through the "Explore" feature without necessarily registering; search functionality where users enter specific queries and the system processes these, applies filters, configures search settings, and retrieves results based on available devices; and finally, session termination where the system logs the user out upon exiting.

Further, on these processes by detailing the login and sign-up sequence, which involves credential verification through the user database, and the explore feature, where users select from categories such as smartphones, laptops, and televisions to retrieve detailed device information. The search feature includes inputting queries, setting search configurations like brand and specifications, and displaying filtered results that show specifications, warranty, brand, and release date. Finally, it concludes

with session management, ensuring the user is logged out securely when exiting the app. This comprehensive approach ensures smooth interaction between the user, the system, and the database, offering a reliable and informative experience.

IV. SYSTEM DESIGN



A. User Interface Design

The User Module offers a user-friendly interface, enabling customers to explore device information and manage their personal accounts effectively.

Key Features of the User interface:

1. Registration and Login:

- **User Registration:** New users can sign up by providing basic information such as name, email, phone number, and address. After registration, should enter his device details, so that it can save in the database and helps the users to view more collections that user enters.
- **Login:** Registered users can log in using their credentials, which may include a username and password. The system uses secure authentication to protect user information.

2. Browse Menus:

- Customers can browse a detailed list of available Explore options, which include five device categories, and configurations. Each menu item displays important details such as price, brand name, model name, support year and warranty year information.

3. Profile Management:

- Users can check up their personal details and maintain a record of tracked devices.
- A dashboard displays a summary of warranty tracks and device details.

B. Device Information Management

It serves as the core of the application, aggregating and organizing device-related data.

Key Features of the Menu and Order Management Module:

1. Data Aggregation:

- The module collects device specifications, release dates, and warranty details from multiple manufacturers and stores them in a centralized database.

2. Search and Filter Options:

- Users can easily navigate the extensive device database using advanced filters and search options.

3. Warranty Tracking:

- The module tracks warranties and service agreements, sending timely notifications for renewals.

These two modules work in harmony to create a robust, user-friendly system that simplifies accessing, comparing, and managing information about electronic devices. By integrating intuitive features and real-time data updates, the application enhances the user experience while providing a reliable resource for electronic device management.

V. IMPLEMENTATION

The implementation of the proposed system for managing electronic device data, warranties, and service agreements integrates various technologies to ensure seamless operation, scalability, and security. The following is an overview of how each technology contributes to database management, interface design, and overall system functionality.

A. Technologies Used:

To develop a fully functional mobile application system, the following technologies are used:

1. React Native (Typescript)
2. Firebase (Backend)
3. Firestore (Database)
4. Expo go app (Development Framework)

These technologies are used to handle the back-end logic, front-end interface, data management, testing the implementation, ensuring a smooth and efficient user experience.

B. Database Design

Firestore (Database):

- It is a flexible database that stores data in documents and collections. Device data, warranty details, and user profiles are stored as documents in relevant collections.
- Collections include Users, Devices, Warranties. For example, the Devices collection stores device specifications such as model, brand, release date, and support year, while the Users collection contains all the device details that the users entered while registration.
- Firebase Cloud Functions are used for serverless backend logic, handling processes like user authentication, device data retrieval, and updating warranty details. These functions manage interactions with the Firestore database and are triggered by user actions such as placing an order, updating warranty status, or logging in. For example, when a user wants to search a device information, Cloud Functions make sure that the user's request is processed, inventory is updated.

C. Interface Design

React Native (Frontend):

- React Native is used for building cross-platform mobile applications, allowing the same codebase to be used for both iOS and Android devices. It supports the creation of visually rich and responsive user interfaces.
- TypeScript enhances the development experience by providing static typing, improving code reliability and maintainability.
- React Native's component-based architecture enables modular development, making it easy to design interactive UI components like search bars, device details pages, and home page interfaces.
- Example, the Explore page displays a collection of electronic devices with filters for categories, brands, and specifications. When a user taps on a device, they are taken to a detailed page showing information like the model, price, warranty, and service agreement options.

D. Data Management (Firestore)

- The Firestore database stores user profiles, devices, warranties. Data is structured in collections and documents.
- React Native fetches and displays data from Firestore in real-time, ensuring that the app always presents the latest information to users.

E. Interaction Between Modules

- When a user logs in, the app authenticates the user via Firebase Authentication, retrieving their data from Firestore. The user can then browse devices, check warranties, and manage information.
- Firebase Cloud Functions handle order submissions, warranty updates, and device data processing. Once a user is registered with his device the user's collection in firestore updated, and the data is reflected immediately in the app.

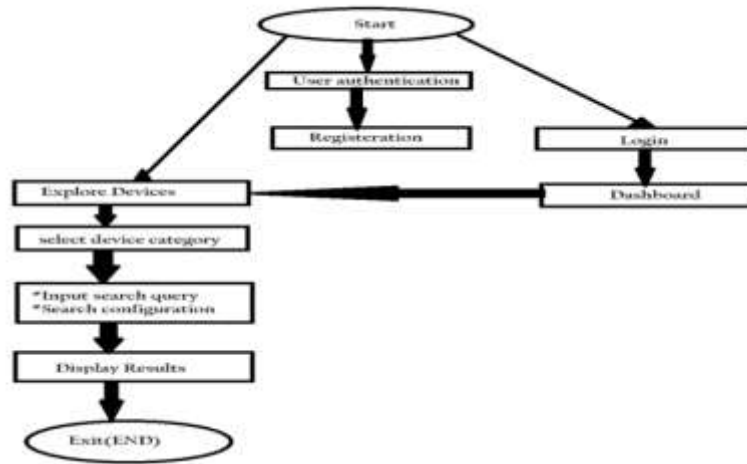


Fig.1 Data Flow Diagram

VI. Experimental Results

The implementation of the DeviceInfo Hub Application for providing comprehensive details about various electronic devices has successfully met the objectives of delivering an accessible, informative, and user-friendly resource for users. The app, built using React Native, allows users to get the information of specified devices, including device specifications, brand details, release dates, warranty expire information. Following thorough testing and integration of various components, the platform is fully functional and performs as intended.

System Performance: During testing, the application demonstrated smooth performance across both Android and iOS platforms. The integration of a centralized database allowed for efficient retrieval and display of device specifications, warranty details, and other related information in real-time.

User Experience: The user interface (UI) was designed to be intuitive and easy to navigate. The users can browse through different categories, and get details about every product, including specifications and warranty information. Feedback from usability testing confirmed that the app's layout was user-friendly, with well-organized sections and easily accessible data. The app's search functionality allowed users to filter devices making it easier to compare similar products. Moreover, the warranty tracker feature, which alerts users about expiring agreements.

Security: The application underwent extensive security testing, ensuring that sensitive data such as user profiles and the information were secured from the access which are unauthorized. Secure login methods, along with encryption of personal information, were implemented to ensure user data was kept safe. Additionally, Firebase Authentication safeguards the user accounts from the access of wrong persons.



Fig.2 Explore options of devices

VII. CONCLUSION

This paper presents a comprehensive approach to build a cross platform called DeviceInfo Hub using react native framework. The DeviceInfo Hub Application for electronic devices has achieved its goal of providing an accessible and informative resource for users to updated with the latest products. With its reliable data management, the app offers users a seamless experience for comparing devices and managing warranty details. The use of React Native for cross-platform compatibility and Firebase for real-time data handling ensures that the application can be easily maintained and scaled for future enhancements, such as the addition of more device categories or advanced user preferences.

In conclusion, the project receives the positive feedback from users. Thorough testing across multiple performance, usability, and security scenarios, the application will serve as a valuable resource to make decisions about devices. This mobile app provides a convenient solution for managing device information, simplifying the process of maintaining products details while ensuring that users stay informed and organized.

REFERENCES

- 1] TechAhead. (n.d.). Building Cross-Platform Mobile Apps with React Native. Retrieved from <https://www.techaheadcorp.com/blog/building-cross-platform-mobile-apps-using-react-native/>.
- 2] React Native Documentation: React Native Official Documentation. (n.d.). Retrieved from <https://reactnative.dev/docs/getting-started>.
- 3] Cross-Platform Mobile App Development: Baisden, A. (2021). How To Build Cross Platform Apps Using React, React Native and Redux. DEV Community. Retrieved from <https://dev.to/andrewbaisden/how-to-build-cross-platform-apps-using-react-react-native-and-redux-212o>.
- 4] Device Specifications: DeviceSpecifications. (n.d.). Retrieved from <https://www.devicespecifications.com/>.
- 5] Guide on Using React Native: Das Shuvo, S. (2023). A Complete Guide on Using React Native to Create Cross-Platform Mobile Apps. DEV Community. Retrieved from https://dev.to/sajeeb_me/a-complete-guide-on-using-react-native-to-create-cross-platform-mobile-apps-55jh.