LibUno: A React-Based Digital Platform for Smart Library Management

Pradeep Jha
Assistant Professor, Department of CSE, Global Institute of Technology, Jaipur, Rajasthan, India
pradeep.jha@gitjaipur.com

Manju Mathur
Assistant Professor, Department of CSE, Global Institute of Technology, Jaipur, Rajasthan, India
manju.mathur@gitjaipur.com

Abhay Purohit
Assistant Professor, Department of CSE, Global Institute of Technology, Jaipur, Rajasthan, India
abhay.purohit@gitjaipur.com

Apoorva Joshi
Assistant Professor, Department of CSE, Global Institute of Technology, Jaipur, Rajasthan, India
apoorva.joshi@gitjaipur.com

Anantika Johari
Assistant Professor, Department of CSE, Global Institute of Technology, Jaipur, Rajasthan, India
anantika.johari@gitjaipur.com

ABSTRACT: LibUno represents a significant step toward modernizing library management by digitalizing the process of borrowing books. Built with React, the platform offers an intuitive and user-friendly interface that simplifies interactions between students and library administrators. By replacing manual processes with an efficient digital system, LibUno enhances the overall library experience, reduces errors, and saves time for all stakeholders.

The app's integration of features such as user authentication, real-time updates, and a centralized database ensures smooth operations and easy access to resources. By leveraging modern technology, LibUno not only streamlines library workflows but also aligns with the growing need for digital transformation in educational institutions. This platform demonstrates the potential of technology to improve traditional systems and foster a seamless, digital-first library ecosystem.

KEYWORDS: LibUno, Web Development, Library Management, React, Digital Platform.

1. INTRODUCTION

In this modern world full of modern technologies, where all the information sharing and transaction are done with the help of various applications and web services, it is necessary for the upcoming younger generations to have the knowledge of computer and its applications which are provided along with it. Over the years, the rapid improvement of hardware and software technologies made computer devices, mobiles, laptops, and other digital technologies more users friendly and affordable. Since data, information and connectivity are

rapidly increasing day by day; websites are challenged with durability, performance, security, and maintenance. Therefore, many technologies, new web applications architectures, and tools have been created to support these objectives specifically.

LibUno is a cutting-edge library management system and a platform designed for college libraries. It enables students to efficiently self-issue books through a mobile app, enhancing the book-issuing process.

Key features include:

- **Self-Issuing:** Streamlines book checkout with an intuitive mobile app.
- **Automated Reminders:** Notifies students about return deadlines to ensure timely returns.
- **Tracking and Monitoring:** Allows librarians to manage and track issued books and overdue returns.
- **Penalty Management:** Facilitates the collection of fines for overdue books.

LibUno improves library efficiency and reduces administrative workload through its comprehensive Mobile App for students and Web Admin Portal for librarians and college authorities.

According to Herbert (2022), React provides state-of-the-art user interfaces and improves the efficiency of JavaScript-driven pages. React.js is a tool that helps developers build websites and applications that users can interact with. It was created by Facebook and is like a set of building blocks that make it easier to create these websites and applications. Instead of writing a lot of complicated code from scratch, React JS lets developers reuse small pieces of code called "components." These components are like separate parts of a puzzle that can be put together to make the whole website or application. By using React, developers can build things faster and with less code than if they were just using basic JavaScript. ReactJS library helps to create a quick response and high-performance website, build productivity rather than other frameworks in the same area and reduces the cost for maintenance in the system.

The main aim of the thesis “LibUno” is to learn and represent React JS and the implementation tools used to design website design Library Management System. The focused studies and practices for the project are to present the detailed information of the topic Library Management System. Also, to give a detailed information of React JS and its core concept. To analyze the requirement and feasibility to build the project LibUno. To build a website design for the project using different implementation tools and software development tools and to use the application for better monitoring and controlling for library purposes.

It starts with an introduction of React JS and concept of the project on the first chapter which is followed by the brief explanation and introduction of React Js and its concept in chapter two. Project analysis for requirements and feasibility analysis are dis-cussed in chapter three. In chapter four, the implementation tools that are required to build the website design are discussed. Library Management System, website and its overall structure of the project are discussed in chapter five. Finally, chapter six of the thesis concludes the project Library Management System built for Library purposes.

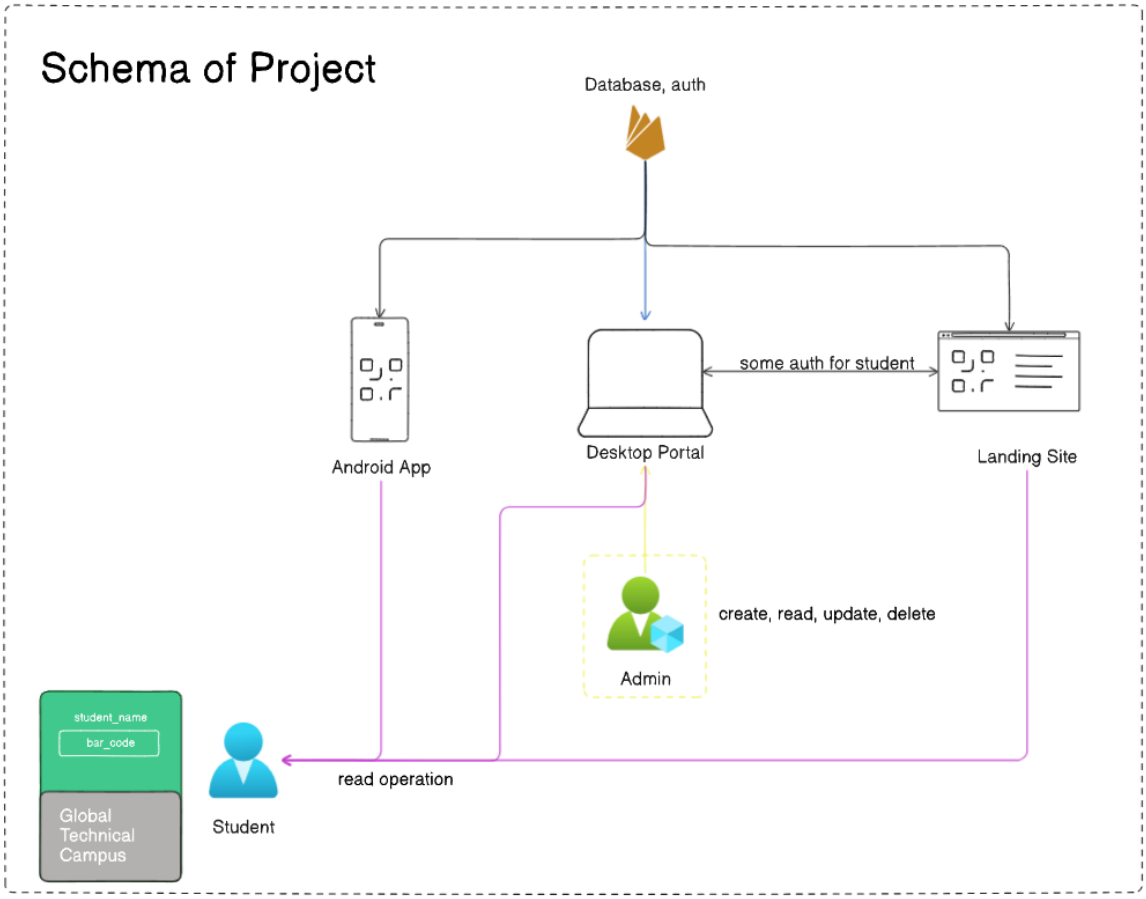

Figure 1: Schema of LibUno

2. NEED FOR THE PLATFORM

- **Inefficient Book Issuing:** Current systems are time-consuming and cumbersome for students and librarians alike. A platform like LibUno can streamline book issuing, reducing both the time and hassle involved.
- **Lack of Tracking and Management:** Without an effective tracking system, it's challenging to monitor which books are out and manage them efficiently. LibUno provides a comprehensive dashboard for administrators to track book status in real-time.
- **Poor Accountability for Overdue Books:** Many students who fail to return or misplace books are not adequately tracked, leading to unaccounted-for resources. LibUno ensures proper tracking and accountability, addressing this critical gap.
- **Manual Return Reminders:** The current system relies on manual reminders for book returns, which can be inefficient. LibUno automates these reminders, ensuring timely returns without additional effort from staff or students.

3. LIBUNO SITE DEVELOPMENT

A. Implementation Tools

During the implementation of the system, a combination of front-end and back-end development tools were utilized. The front-end development tools encompassed technologies

such as HTML, CSS, and JavaScript, which were employed to create the user interface and design elements of the system. These tools allowed for the creation of visually appealing and interactive components that enhance user experience. On the back end, programming languages and frameworks like React.js and Node.js were utilized. React.js facilitated the development of dynamic and responsive user interfaces, while Node.js provided a server-side runtime environment for executing server-side logic and handling requests.

B. Front End Tools

Front End Technologies are the set of technologies that are used to create the UI (User Interface) of web applications and web pages. With the aid of front-end technologies, developers can create the de-sign, structure, animation, and everything that you see on the screen while visiting a website, webpage, web application, or mobile app. It plays an important part in attracting users and getting them to take action. A seamless front-end will make users adore and recommend the application. Employing front-end tools and technologies to increase user interaction, efficiency, interactivity, and visual design of the application is critical for establishing user loyalty. To accomplish this, mobile and web developer need to be more efficient and precise. The use of React (JSX), CSS and Bootstrap front end technologies are implemented on the system.

- **React (JSX):** The render function of React components plays a crucial role in generating HTML for the user inter-face. By utilizing JavaScript extensions like JSX, developers can create HTML-like code using JavaScript syntax. JSX is an HTML-like syntax that is supported by ECMAScript, allowing it to coexist with JavaScript and React code seamlessly. Preprocessors utilize this syntax to convert the HTML-like code into standard JavaScript data that can be consumed by a JavaScript engine. React JSX is utilized in building the front end of the system, enabling developers to create dynamic and interactive user inter-faces.

- **CSS:** CSS, or Cascading Style Sheets, is a simple design language intended to make web pages presentable by simplifying the look and feel process. It handles the visual aspects of a webpage. Using CSS, control of the color of the text, the design of the fonts, the spacing between paragraphs, the way columns are sized and positioned, and a variety of other things. CSS is used to control the appearance of an HTML document and may be combined with HTML or XHTML to create the majority of the web ap-plication. The styling of the project has been done with CSS.

- **Tailwind:** Tailwind CSS is a utility-first CSS framework that allows developers to style web applications with pre-defined utility classes directly in the HTML. Instead of writing custom CSS or relying on traditional frameworks like Bootstrap, Tailwind provides a vast collection of low-level utility classes that can be composed to build any design. It emphasizes flexibility and customization, enabling developers to create unique designs without being confined by pre-styled components. Tailwind's configuration file allows for fine-grained control over the design system, making it easy to implement custom colors, spacing, typography, and responsive breakpoints.

  In LibUno, Tailwind CSS can streamline the design process for both the admin and student portals. Developers can use its utility classes to quickly build responsive and visually consistent components such as navigation bars, forms for book management, and tables for displaying book records. For instance, the admin dashboard can feature card components styled with Tailwind classes to showcase statistics like the number

of books, issued books, and overdue returns. On the student portal, Tailwind can help create intuitive search interfaces and attractive book-detail cards, enhancing the overall user experience. Its responsive design utilities ensure that the system is accessible across various devices, from desktops to smartphones, without additional effort.

- **shadcn/ui:** Shadcn/UI is a modern UI component library that integrates seamlessly with React and utilizes Radix UI primitives for accessibility and customization. Built with a focus on modularity and design consistency, Shadcn/UI provides pre-built components like buttons, dropdowns, modals, and tables that can be easily themed and extended. Unlike many libraries, it offers flexibility in styling, enabling developers to customize components to match their design systems or even integrate with frameworks like Tailwind CSS for a cohesive user interface.

  In LibUno, Shadcn/UI can greatly simplify the process of creating polished and functional user interfaces. For example, components like sortable tables can be used to display book records, while modal dialogs can facilitate actions like issuing or returning books. Dropdowns and select menus from Shadcn/UI are ideal for implementing filters, such as sorting books by genre, availability, or publication date. By combining Shadcn/UI with Tailwind CSS, developers can maintain a consistent aesthetic while leveraging the advanced functionality of Shadcn's components, resulting in an intuitive and professional-looking system for both admin and student users.

C. Backend Tools

The backend application of the LibUno was developed using specific tools for effective implementation. In the implementation of the LibUno specific backend tools were utilized for effective development and implementation. One of the key tools used was Node.js, which served as a communication interface between the application and the database. Another important tool utilized was FireBase, a complete solution for backend functionality including databases.

- **Node.js**

Node.js played a pivotal role as a communication interface between the application and the database in the project. It facilitated the smooth exchange of data from the database to the frontend, enabling seamless integration and transmission of information across different components of the application (Shahid, 2021). By leveraging the power of Node.js, the project achieved efficient data flow and effective communication between various parts of the application. This played a crucial role in enhancing the overall performance and functionality of the system, ensuring a seamless user experience.

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine that enables developers to execute JavaScript code server-side. Unlike traditional client-side JavaScript, which runs in the browser, Node.js allows developers to build scalable, fast, and efficient web servers and applications. It uses an event-driven, non-blocking I/O model, which makes it lightweight and ideal for handling asynchronous tasks. This non-blocking nature allows Node.js to handle a large number of simultaneous connections efficiently, making it a popular choice for real-time applications, such as chat applications, APIs, and streaming services.

One of the key features of Node.js is its extensive ecosystem of packages, available through npm (Node Package Manager). This ecosystem provides ready-to-use modules for a wide range of functionalities, such as connecting to databases, managing files, and handling HTTP requests. Node.js also supports building RESTful APIs, making it a strong candidate for building backend services in web applications. With the ability to use JavaScript on both the client and server side, Node.js simplifies development by allowing developers to use a single language across the entire stack, making it easier to share code, libraries, and tools between the frontend and backend.

- **FireBase**

Firebase is a comprehensive cloud platform by Google that provides tools and services for developing web and mobile applications. It is particularly popular for its real-time database, Firestore, and robust authentication mechanisms. Firebase allows developers to store and sync data across users in real-time, ensuring up-to-date information is always accessible. Its authentication module supports multiple sign-in methods, including email/password, social login providers like Google, and even custom authentication systems. Additionally, Firebase offers scalability, cross-platform compatibility, and built-in security rules to manage data access seamlessly.

Firebase is an excellent choice for a LibUno due to its ability to handle real-time data operations and simplified backend management. For the database, Firestore can efficiently store and manage book records, user information, and loan data, with the ability to query and sort the data dynamically. Its real-time synchronization ensures that both admins and students see the latest updates, such as newly added books or changes in availability. Firebase Authentication adds a secure and straightforward way to manage user logins for the admin and student portals, supporting features like role-based access control. Furthermore, Firebase's integration with front-end technologies like React makes it easy to implement features like search, book issuance, and notifications, streamlining the development process while delivering a seamless user experience.

Firebase integrates seamlessly with ReactJS, making it an ideal backend solution for building dynamic and responsive web applications. By using Firebase SDKs, developers can easily connect their React components to Firebase services such as Firestore, Realtime Database, and Authentication. React's state management and lifecycle methods work well with Firebase's asynchronous operations, allowing developers to handle real-time updates and fetch data efficiently. For instance, Firebase Authentication can be integrated with React's context API to manage user sessions across the application, while Firestore hooks like onSnapshot enable live updates for components displaying book data or user activity. Additionally, Firebase's compatibility with npm modules and React tools simplifies the setup process, allowing developers to focus on building features rather than managing backend complexities.

4. WEBSITE AND ITS STRUCTURE

A. Ideal IDE

When starting to build a program, choosing the right code editor is important. It should support good code quality, have the latest features, make programming easier, and be free to use. Visual Studio Code, Atom, and Sublime Text are popular options. Visual Studio Code was selected as the preferable code editor for creating the application because it is great for JavaScript and has useful extensions. There are a number of useful extensions and add-ons

that have been implemented into Visual Studio Code to improve the programming process. Snippets for ES7 React, Redux, GraphQL, and React-Native, Node.js Modules and Prettier - Code Formatter are a few of them. These extensions add helpful tools for working with React, Node.js, and formatting code neatly. Overall, Visual Studio Code helps us write better code and work more efficiently.

B. Project structure in VS Code

The library management system project is structured in Visual Studio, as illustrated in Figure 13. It encompasses various folders and components that play crucial roles in the project's implementation. Within the project, the package.json file serves as a manifest that lists the dependencies required for the Node.js project. These dependencies consist of external packages or libraries necessary for the proper functioning of the project. The SRC folder, located under the client directory, serves as the core of the project. It contains subfolders, components, content, and elements that constitute the website's structure and functionality. The Views folder contains the after-login and login subfolders. The after-login folder encompasses the code for the admin page, teacher page, and student page, each with their respective CSS files stored in the static subfolder within each main folder. The login folder, on the other hand, includes the code specifically related to the login page. The sign-up folder is responsible for handling the code related to the sign-up page.

Under the components folder within the src directory, the navbar.js and footer.js components are pre-sent. These components play a crucial role in rendering the respective pages and ensuring a consistent layout and structure. Finally, the App.js file acts as the main component, known as the App Component, in the React application. It serves as a container for all other components, providing a centralized structure and organization for the project. This well-structured project architecture enhances the development process, promotes code organization, and ensures efficient implementation of the LibUno. As shown in the project was structured in different folders and components for each page. The respective CSS was written under the static folder. Different folders named student, teacher, admin, login, signup was created to ease the project.

The displayed directory structure represents a ReactJS project organized with a modular and scalable approach. The src folder contains the core application logic, divided into multiple subfolders for better maintainability. The components directory is structured further into specific functionality areas such as auth for authentication-related components, dashboard for managing views like BorrowedBooks, Overview, and RecentActivity, and UI for user interface components like Dashboard.jsx and Login.jsx. The context folder under auth includes context-related files (firebaseContext.jsx) and route protection logic (PrivateRoute.jsx). The lib folder likely contains reusable utilities (utils.js), while the global application entry points like App.jsx and main.jsx handle the overall rendering and configuration. Configuration files like tailwind.config.js and vite.config.js suggest the use of Tailwind CSS for styling and Vite as the build tool, ensuring a fast and modern development experience. This structure promotes clean separation of concerns, making it easy to navigate and extend the application. This well-structured setup, named "React Modular Starter", reflects best practices in React development, making the application easy to scale, debug, and maintain.

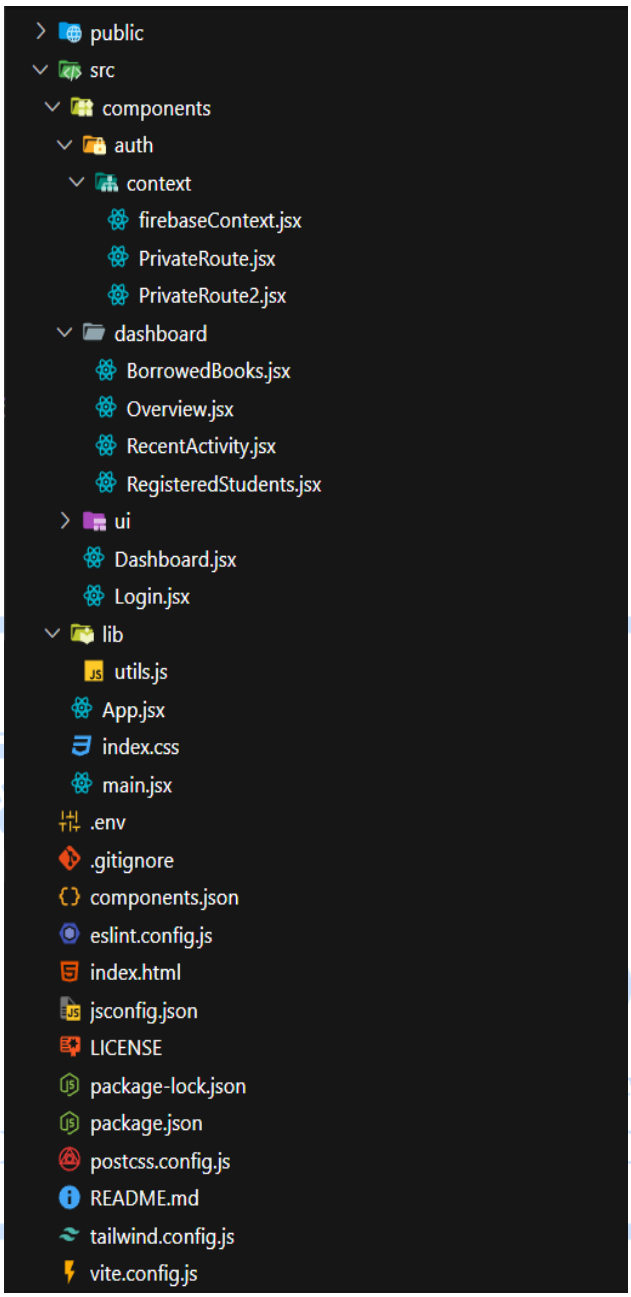

Figure 1: Website Structure in VS Code

C. Website

The designed single page application offers a user-friendly experience, enabling easy usage for both teachers and students. Teachers have the capability to effortlessly add files to the designated category, while students will encounter no difficulties accessing the library's online server. The student and teacher can view the available books and facilitate their book selection and subscription process. The application also simplifies tasks such as book renewal and deletion, further enhancing its user-friendly nature. The result of the website is further explained and shown below.

Login page

Figure 2 illustrates the login functionality of the system, showcasing the user interface where authorized users can enter their username and password. This visual representation aids in

understanding the login process and the interface design. The system validates the provided credentials against the predefined records, ensuring the authentication of the user. Once successfully logged in, users are redirected to the dashboard, granting them access to the system's features and resources. The login functionality serves as a pivotal security measure, safeguarding the system from unauthorized access and protecting sensitive information. Through the implementation of Figure 1 depicted login process, the system ensures a seamless and secure user experience, promoting the efficient usage of the LibUno.

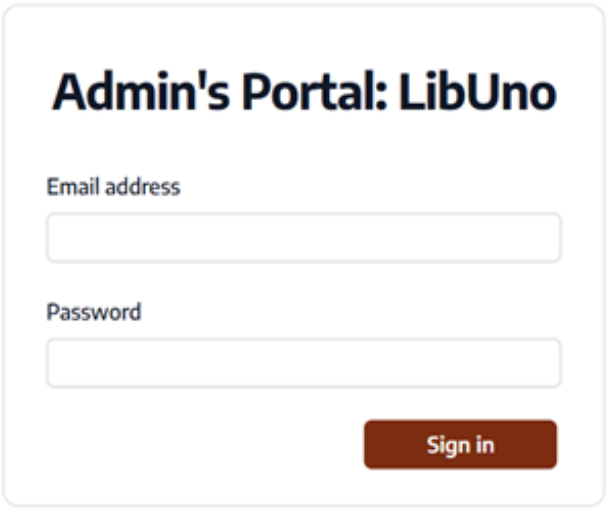

Figure 2: Login Page

Sidebar

Sidebar, which is positioned on the left side of the screen and is accessed by clicking the burger menu. The navbar serves as a crucial navigation menu, providing users with easy access to various sections and functionalities of the application. By clicking on the burger menu, users can expand the navbar and view the available options for navigation. This intuitive design ensures that users can quickly and conveniently navigate through different areas of the application. The navbar enhances the user experience by providing a clear and organized menu structure, allowing users to effortlessly explore and utilize the features and functionalities of the project.

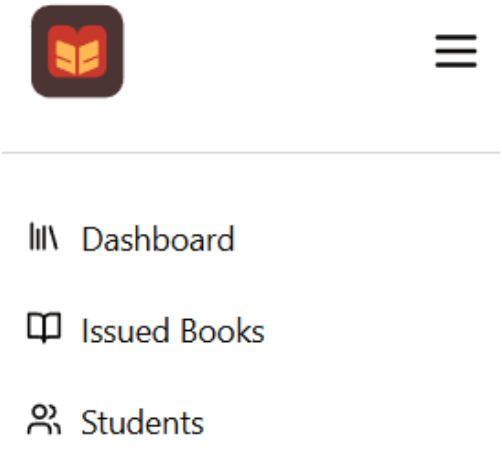

Figure 3: Sidebar

Overview

The displayed dashboard, titled "LibUno," serves as an intuitive summary interface for a library management system. With its clean and minimalistic design, it provides administrators with an at-a-glance overview of key metrics, such as the total number of books, registered students, currently issued books, and overdue books. Each metric is visually distinguished using color-coded cards, making it easy to interpret information quickly. This layout prioritizes clarity, ensuring users can identify critical library stats without navigating deeper into the system.

The dashboard also provides additional context for each statistic. For example, the "Total Books" card includes a monthly increment (+12 this month), offering insights into library growth, while the "Overdue Books" card indicates changes compared to the previous week (-2 from last week), helping admins track improvement in overdue returns. This level of detail supports better decision-making, such as identifying patterns in book borrowing or evaluating the effectiveness of library policies. Furthermore, the cards are accompanied by relevant icons, enhancing usability and accessibility.

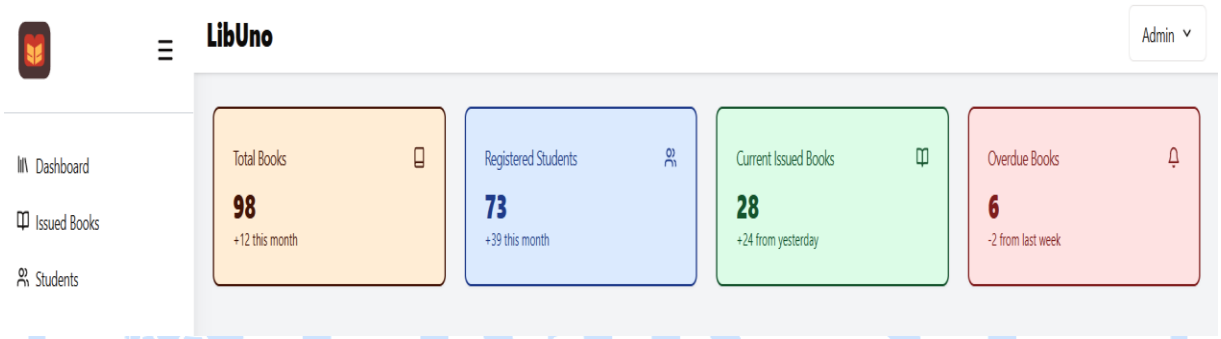

Figure 4: Overview on Website

The navigation menu on the left offers seamless access to essential sections like "Dashboard," "Issued Books," and "Students." The integration of the admin dropdown in the top-right corner facilitates user role management and settings customization. The overall structure balances simplicity and functionality, making it easy for admins to manage the library's operations efficiently. This dashboard is an effective tool for summarizing and visualizing library data while providing actionable insights for smooth day-to-day operations.

Borrowed Books Information

The "Borrowed Books" section of the library management system provides a streamlined view of the library's book lending status. This table-based layout displays key details such as the title of the book, the author, the student who has borrowed it, and the current status. The clean and simple design ensures easy readability, making it efficient for librarians or administrators to quickly assess borrowing activities.

The "Status" column uses clear labels like "Available" and "Borrowed," with color-coded tags (green for available, yellow for borrowed), offering a visually intuitive way to track book availability. This feature simplifies the identification of resources currently out of circulation and those ready for lending. The inclusion of a filter dropdown ("All") and a refresh button enhances interactivity, allowing administrators to customize and update the view instantly.

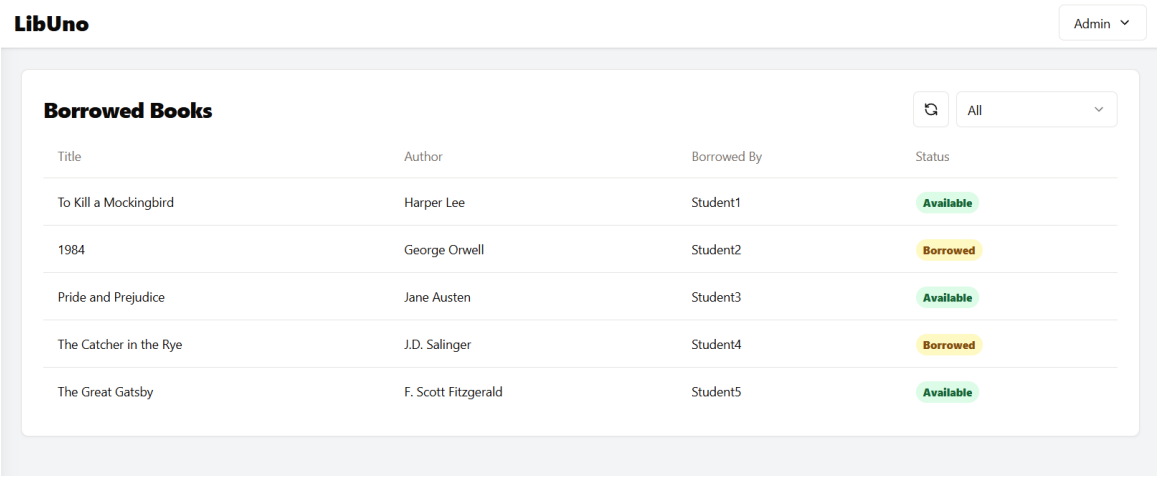

Figure 5: Borrowed Books

This section is a vital tool for managing library operations efficiently. By clearly associating borrowed books with specific students, it minimizes confusion and supports better accountability. Administrators can use this interface to identify overdue books or contact students if necessary. Overall, this feature enhances operational transparency and ensures smooth day-to-day library management.

Registered Students Information

The image showcases a section of the "LibUno" Library Management System, specifically focusing on the Registered Students module. This module provides a tabular view of students currently registered with the library. The table includes essential details such as the student's name, their unique student ID, email address, and the number of books borrowed. For example, John Doe (ST12345) has borrowed 2 books, Jane Smith (ST67890) has borrowed 1 book, while Bob Johnson (ST54321) currently has no books borrowed.

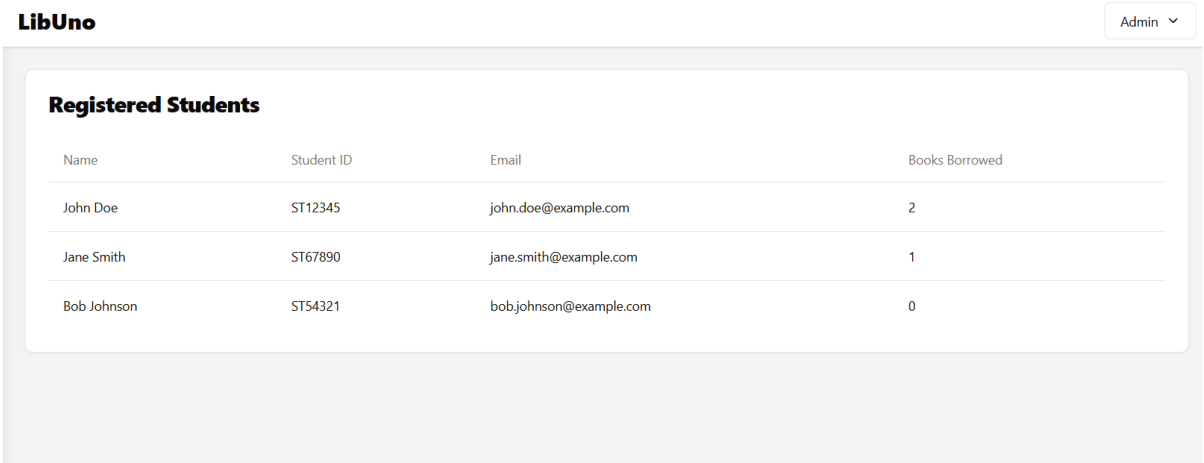

Figure 6: Registered Students

This module is crucial for library administrators to keep track of students' borrowing activity. It offers a clear and concise way to monitor the library's lending status and ensures that the library remains organized. The integration of key identifiers like student IDs and email addresses helps in maintaining accurate records and facilitates easy communication if needed.

Moreover, the system allows for efficient management, especially in scenarios involving overdue returns or inventory checks.

The clean and minimalist interface, as seen in the image, ensures user-friendliness, allowing administrators to quickly access and understand information at a glance. The presence of a dropdown menu labeled "Admin" in the top-right corner hints at additional administrative functionalities, making LibUno a comprehensive solution for library management needs. This feature aligns with modern digital solutions that prioritize efficiency and streamlined user experiences.

All Books Information

LibUno is a library management web application designed to streamline and digitalize the process of managing books in a library. The interface depicted in the image showcases the admin dashboard's "All Books" section, where administrators can view a detailed list of books available in the library. This feature simplifies library operations by allowing admins to easily monitor book details such as title, price, barcode, and availability status. The clear tabular representation ensures quick decision-making for tasks like inventory updates, book lending, and tracking book statuses.

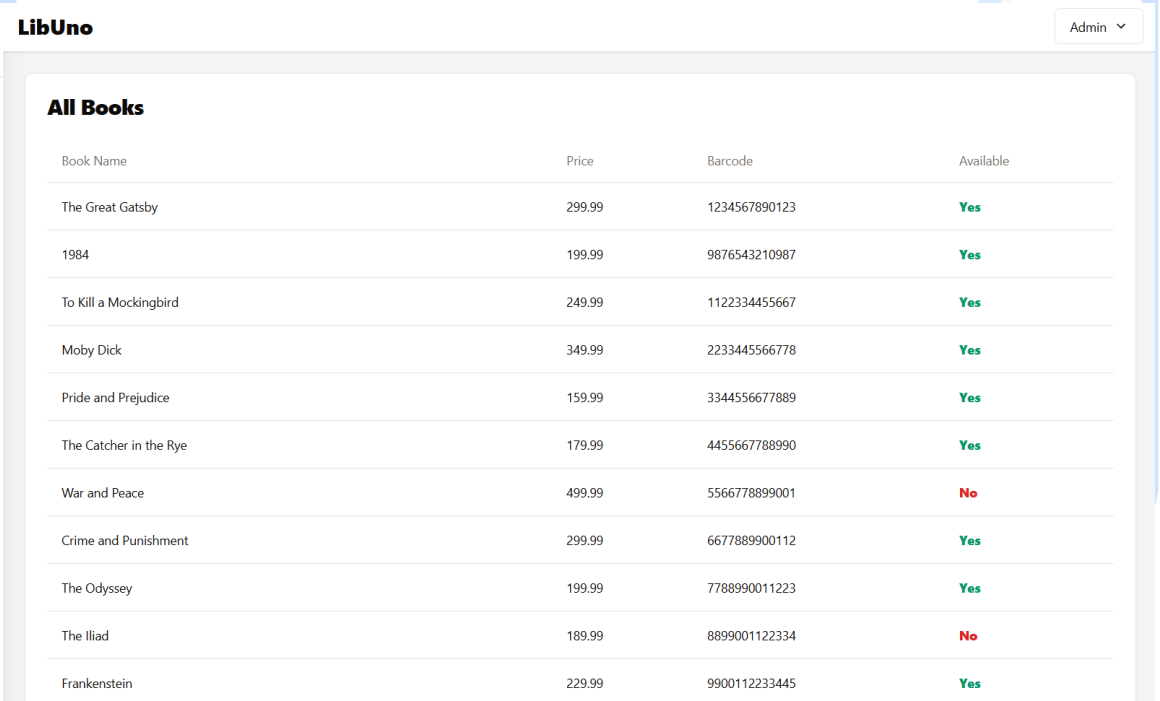

Figure 7: All books information

The "All Books" table displayed here uses dummy data to illustrate how the system organizes and presents essential book information. Each row represents a book, with columns for its name, price, barcode, and availability. The color-coded "Available" status is particularly useful, as it immediately informs the admin about which books are currently accessible for lending (marked "Yes" in green) and which are not (marked "No" in red). This visual distinction ensures minimal confusion and enhances the efficiency of day-to-day management.

By leveraging this system, admins can efficiently maintain library records and avoid manual errors common in traditional library management. Features like barcode tracking and real-time availability statuses help streamline book circulation and ensure the library's resources are optimally utilized. Although the data here is dummy, it represents a scalable and customizable approach that can adapt to libraries of varying sizes and collections.

Return Book Functionality

Following Schema outlines a book submission system designed for a library using Firebase, an admin portal, and a student-facing app. The process begins when a student decides to return a borrowed book. The student navigates to the book within the app and clicks the "Submit" button (step 3). This action triggers an onClick event (step 4), invoking a Firebase Cloud Function (step 5) that generates a unique four-digit OTP for authentication. The OTP is then pushed to the admin portal (step 6), where the librarian can access it.

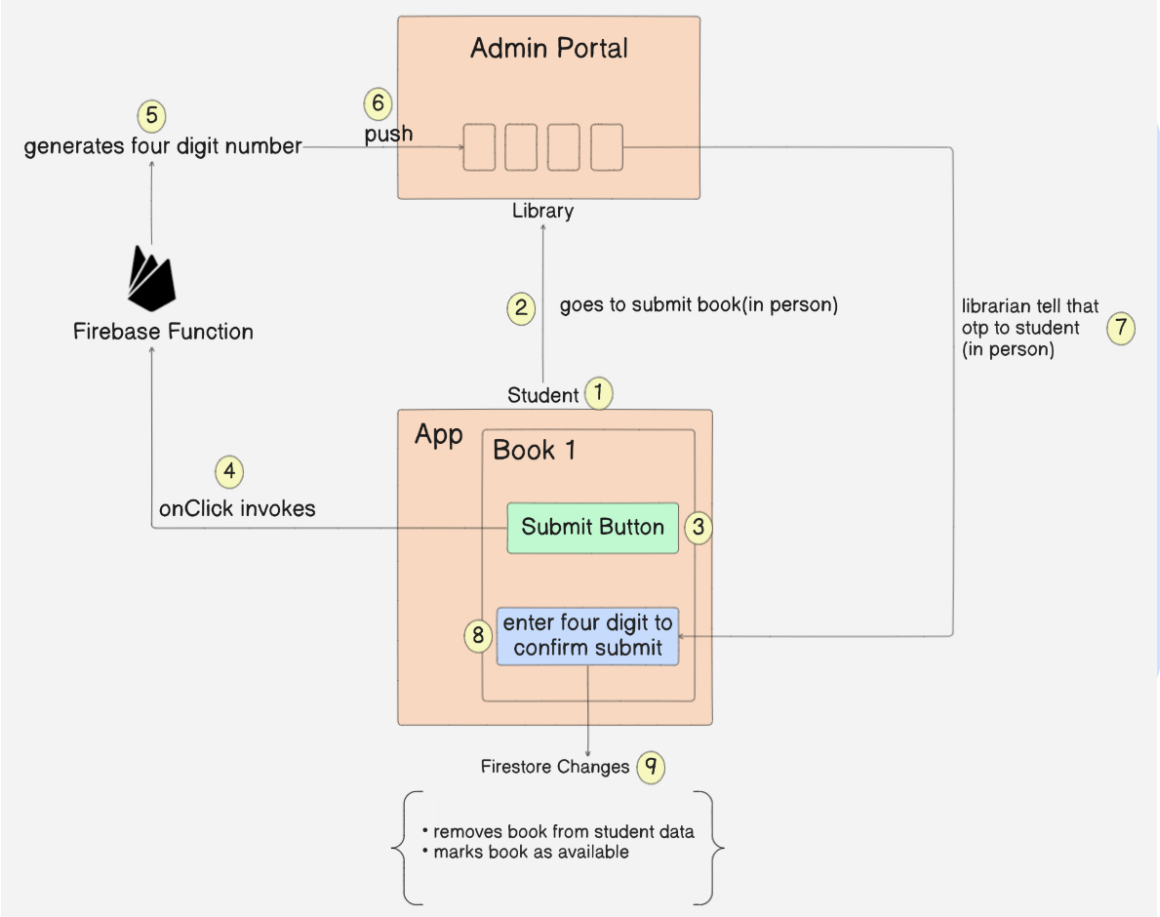

Figure 8: Schema for Returning a Book

The student physically goes to the library (step 2) to return the book. The librarian communicates the OTP generated by the system to the student in person (step 7). The student then enters this OTP in the app (step 8) to confirm the submission process. Upon successful confirmation, the app interacts with Firestore to update the database (step 9). These updates include removing the book from the student's borrowing records and marking the book as available for others to borrow.

This system ensures a seamless and secure return process by integrating both digital and physical interactions. The OTP-based verification guarantees that only the rightful borrower

can complete the submission, while Firestore automates the backend data updates, maintaining the integrity of the library's inventory.

5. CONCLUSION

LibUno represents a significant step toward modernizing library management by digitalizing the process of borrowing books. Built with React, the platform offers an intuitive and user-friendly interface that simplifies interactions between students and library administrators. By replacing manual processes with an efficient digital system, LibUno enhances the overall library experience, reduces errors, and saves time for all stakeholders.

The app's integration of features such as user authentication, real-time updates, and a centralized database ensures smooth operations and easy access to resources. By leveraging modern technology, LibUno not only streamlines library workflows but also aligns with the growing need for digital transformation in educational institutions. This platform demonstrates the potential of technology to improve traditional systems and foster a seamless, digital-first library ecosystem.

REFERENCES

[1] Borusiuk, Y. 2022. Top 10 Benefits of React.js for Application Development. Online. NCube. Available at: https://ncube.com/blog/top-10-benefits-of-react-js-for-application-development.

[2] Oracleappshelp. 2021. React JS programming tutorial for Beginners. Available at: http://oracleappshelp.com/react-js-programming-tutorial-for-beginners.

[3] R. Ajmera and D. Dharamdasani, "Comparative study of existing food delivery app," Global Research Journal, pp. 454–463, 2022.

[4] A. Kalwar, R. Ajmera, and C. S. Lamba, "An empirical study in small firms for web application development and proposed new parameters," Int. J. of Innovative Technology and Exploring Engineering, vol. 8, no. 4, Feb. 2019.

[5] R. Ajmera, A. Kalwar, and C. S. Lamba, "A modern study on progressions & issues of web applications development in small firms," Int. J. of Scientific Research in Science and Technology, vol. 3, no. 8, Nov.–Dec. 2017.

[6] A. Chauhan, R. Misra, "Outline of Web Development Life cycle in Software Engineering", International National Conference on Recent Trends in Engineering & Technology, 2023.

[7] A. Kalwar and R. Ajmera, "ARQI: Model for developing web application," Int. J. on Technical and Physical Problems of Engineering (IJTPE), vol. 13, no. 47, pp. 7–13, Jun. 2021.