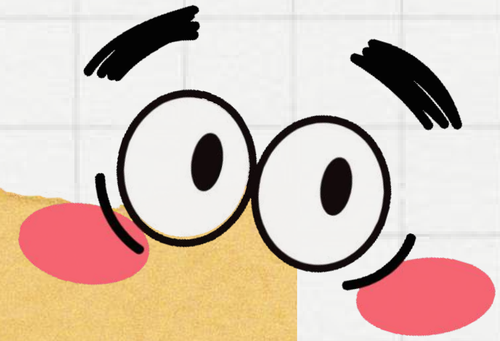
AI for Developer Productivity



Basic Details of the Team

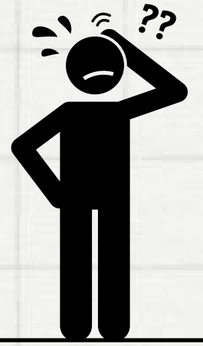
Team Name: TheBlockHeads

Problem Statement Title: AI for Developer Productivity
Porting an Invoice and Billing Platform for Government Universities from Ionic to React Native (can use AI developer tools like Github copilot)

University Name: Netaji Subhas University of Technology

Kushagra Kiyawat - 2023UCB6008
Anshuman Garg - 2023UCB6010
Love Kumar - 2023UCB6017
Vivekanand Rai - 2023UCB6018
Vatsal Veerwal - 2023UCB6055
Aryya Lohia - 2023UCB8061

Background of Problem:



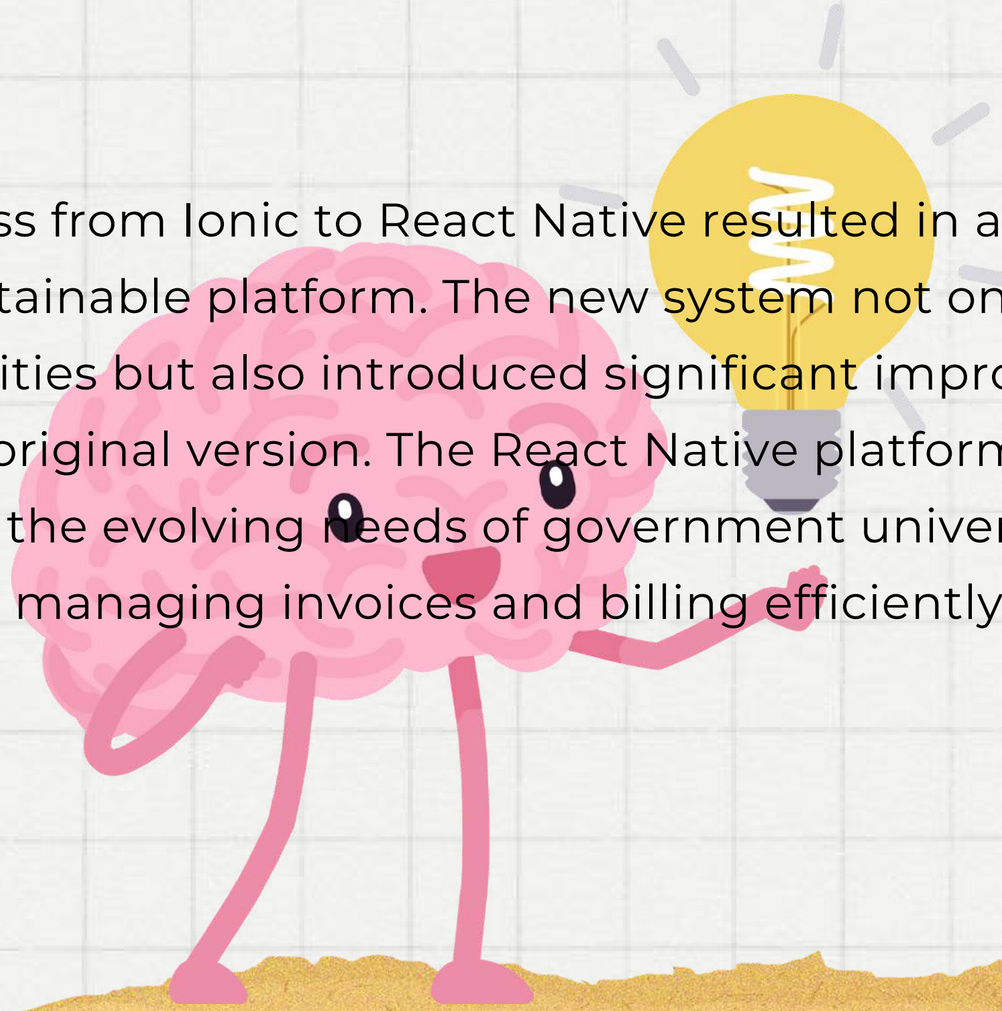
Porting an Invoice and Billing Platform for Government Universities from Ionic to React Native (can use AI developer tools like Github copilot)

Government universities require efficient management of financial transactions such as tuition fees, research grants, and vendor payments. The existing platform built using Ionic has served its purpose but now requires a transition to a more performant, native-like experience provided by React Native. This transition will enable a smoother user experience, better performance, and greater maintainability across platforms.

Solution:

.

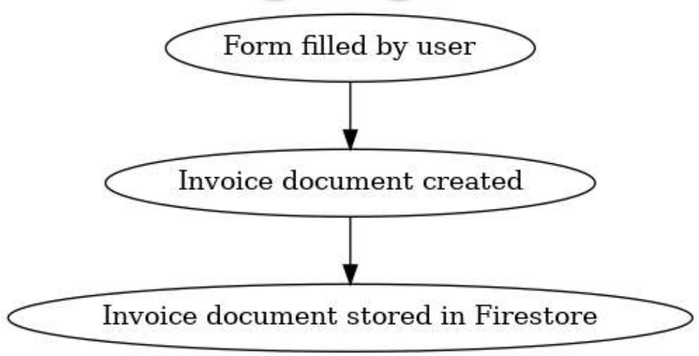
The porting process from Ionic to React Native resulted in a more performant, user-friendly, and maintainable platform. The new system not only replicated the original functionalities but also introduced significant improvements that added value beyond the original version. The React Native platform is now better equipped to meet the evolving needs of government universities, providing a robust solution for managing invoices and billing efficiently.

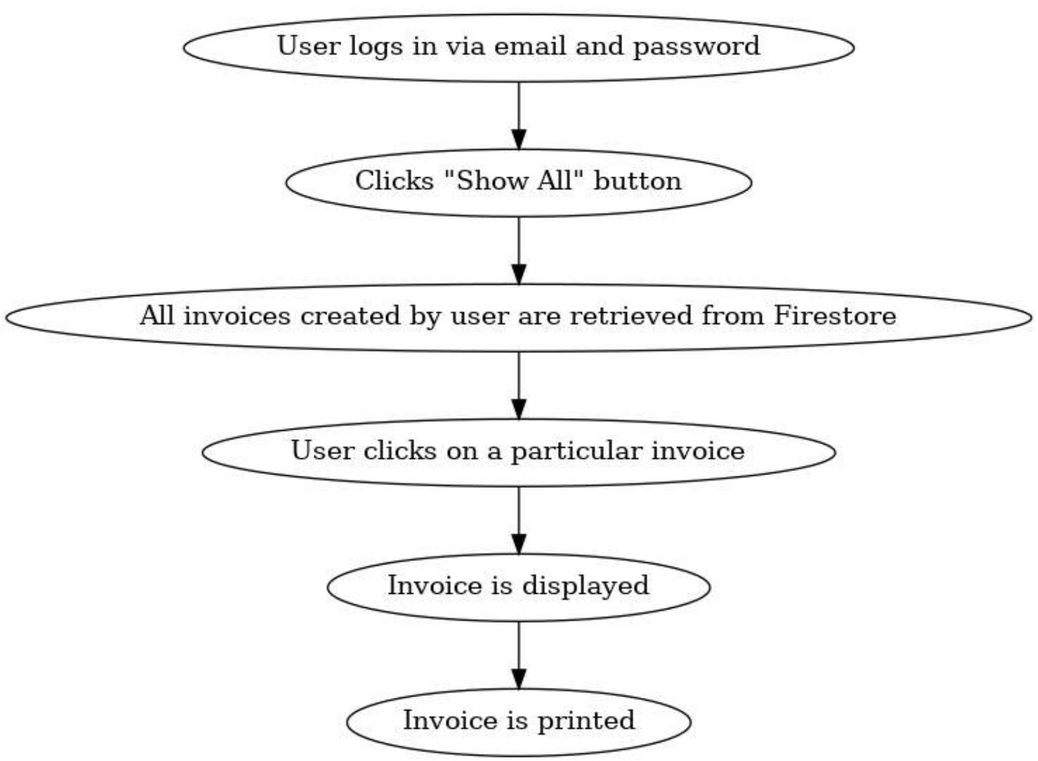


USP

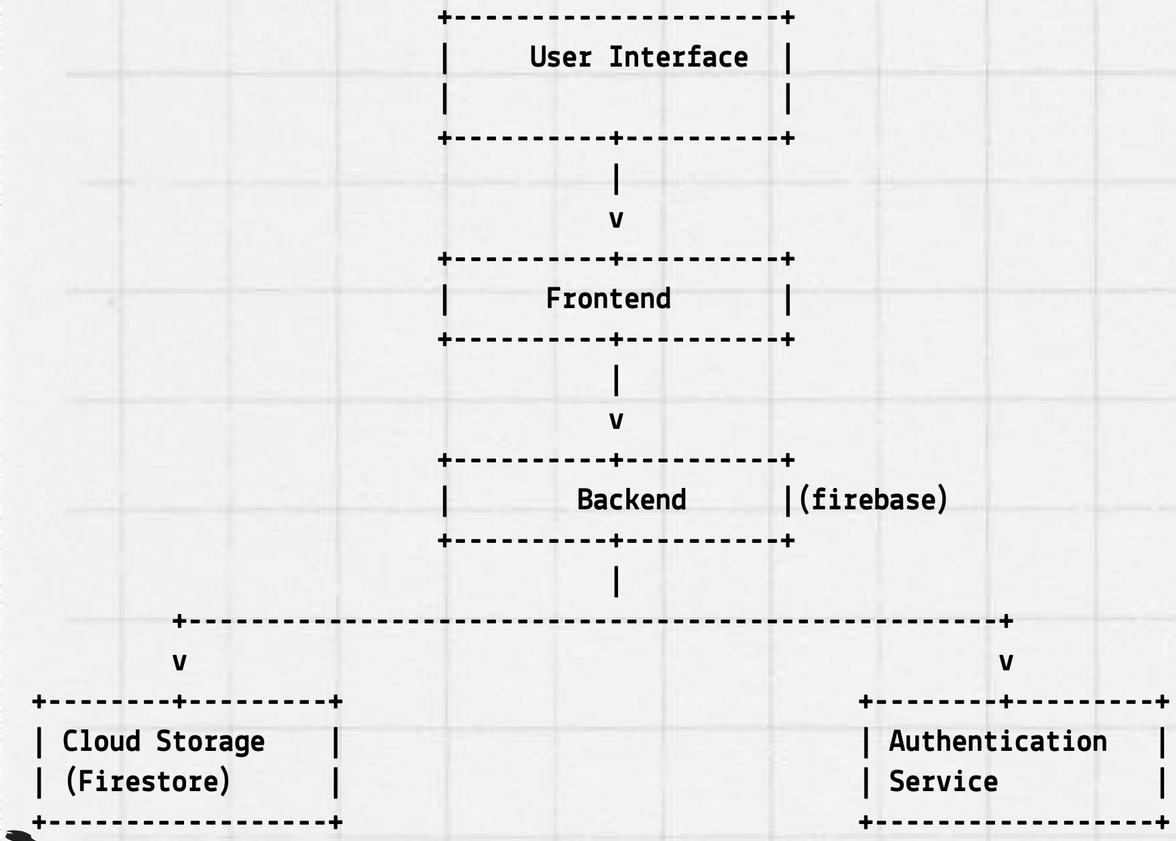
- Cross-Platform Excellence: Seamlessly transition to React Native, leveraged the cross-platform capabilities for a consistent and enhanced user experience on both Android and iOS.
- Performance Optimization: React Native’s near-native performance ensures faster load times and smoother interactions for end-users.
- Scalable Architecture: Adopting a modular and scalable architecture facilitates future expansions, feature additions, and maintenance with ease.
- Enhanced User Interface: Upgrade to a more responsive and intuitive UI, taking full advantage of React Native’s flexibility in design and interaction.
- Improved Developer Productivity: Experience increased development efficiency with React Native’s hot-reloading and rich ecosystem, reducing time-to-market for updates and new features.

Flowchart


Form filled by user
Invoice document created
Invoice document stored in Firestore


User logs in via email and password
Clicks "Show All" button
All invoices created by user are retrieved from Firestore
User clicks on a particular invoice
Invoice is displayed
Invoice is printed

Infra Diagram



Tech Stack

node.js

Firebase

React Native

Expo

Business Model

Value Proposition

- Efficient and Secure Invoicing: Provides an easy-to-use platform for creating, managing, and securing invoices, streamlining financial operations.

Target Customers

- Mainly our platform is designed for Universities but it also provides solutions tailored to small and medium-sized businesses, independent professionals, and accounting experts needing reliable invoicing tools.

Revenue Streams

- Subscriptions and Pay-Per-Use: Generates revenue through recurring subscription plans and charges based on invoice generation or usage.

Key Activities

- Software Development and Support: Focus on continuous enhancement of the platform and providing responsive customer support.

Cost Structure

- Development and Operations: Includes expenses for software development, cloud storage, marketing, and customer support

Future Scope

1. Load Distribution
   - Approach: Balance incoming traffic across multiple servers to prevent overloading any single server.
   - Implementation: Employ load balancers to evenly distribute user requests and enhance system stability.
2. Data Caching
   - Approach: Store frequently accessed data temporarily to reduce the need for repeated database queries.
   - Implementation: Utilize in-memory caching solutions (e.g., Redis, Memcached) for quick access to common data.
3. Database Optimization
   - Approach: Design and maintain the database to perform efficiently, including proper indexing and query optimization.
   - Implementation: Regularly review and improve database schema, queries, and indexing to enhance performance.
4. Background Processing
   - Approach: Offload resource-intensive tasks to background processes to keep the main application responsive.
   - Implementation: Use job queues (e.g., RabbitMQ, AWS SQS) for tasks like invoice generation and processing.
5. Performance Monitoring
   - Approach: Continuously observe system performance and set up alerts for potential issues.
   - Implementation: Deploy monitoring tools (e.g., New Relic, Datadog) to track key performance indicators and receive notifications of any anomalies.
6. Automatic Scaling
   - Approach: Adjust system resources automatically based on current demand to handle traffic fluctuations efficiently.
   - Implementation: Utilize cloud services with auto-scaling features (e.g., AWS Auto Scaling, Google Cloud Autoscaler) to manage resources dynamically.

Market Readiness Plan

- Market Research: Understand user needs, analyze competition, and identify trends.
- Product Development: Complete development, testing, and gather feedback from beta users.
- Go-to-Market Strategy: Plan launch activities, create marketing materials, and engage with influencers and partners.
- Sales and Distribution: Set up online sales platforms, train sales teams, and implement a customer onboarding process.
- Customer Support: Provide support through helpdesk and documentation, and implement a customer success program.
- Performance Monitoring: Track metrics, gather feedback, and update the product regularly.
- Compliance and Security: Ensure legal and regulatory compliance and implement robust security measures.