SNS
Inventory Management System with Barcode and QR Code Integration

Autors: Adam Klementowski · Adam Rudnicki · Adam Skowron · Wojciech Skrzypiec

Supervisor: Rafał Palak

Abstract

The project aimed to improve the inventory management experience for small and large scale enterprises by implementing the application as a Progressive Web App (PWA) with functionalities such as scanning of bar and QR codes or pin point the warehouse based on the user's geolocation. We successfully designed a responsive and intuitive application with the most vital functionalities that is accessible on multiple devices. We learned about the restrictions and strengths of PWA as well as the process of integrating into a complete application. It will help future researchers to better design and integrate this technology in their projects.

1 INTRODUCTION

1.1 Problem characteristics

Inventory management is a key element in the efficient operation of many companies, especially in industries related to logistics, trade and manufacturing. This issue becomes particularly relevant in the context of growing expectations of process automation and minimization of errors resulting from manual management.

1.2 Goal

Our goal is to create a tool that will help companies track and manage resources more efficiently. The main technical assumption was to create an MVP (Minimal Viable Product) version of the PWA application, i.e. one that has CRUD functions to manage warehouses, inventory, product availability and assigned product categories. The application's functionality has been expanded to include a QR code and barcode scanning module. Another feature is the implementation of geolocation, which is designed to speed up warehouse identification based on user location.

1.3 Business and technical benefits

In the business context, the project aimed to increase the efficiency of inventory management, reduce the time required to handle warehouse processes and minimize errors resulting from manual operations. On the other hand, the key technical task was to create an intuitive and functional tool that could be easily integrated with existing enterprise systems, as well as used on both mobile devices and computers.

2 RELATED WORKS

2.1 Description

Enterprise Resource Planning (ERP) systems, such as SAP [2], Oracle NetSuite [9], and Odoo [11], are popular solutions in this domain offering comprehensive features ranging from inventory tracking to financial management. However, these platforms often require significant resources for deployment and maintenance, making them less accessible for smaller enterprises.

For small- and medium-scale enterprises, SaaS (Software as a Service) solutions like Zoho Inventory [13] and TradeGecko [1] are popular alternatives. These tools are user-friendly, lightweight, and integrate well with other software but often lack scalability and deep customization for larger operations.

In the context of technology, barcode and QR code integration has become a cornerstone of modern inventory systems. Libraries such as ZXing [12] offer reliable and efficient methods for decoding these codes, making product identification and tracking much easier.

Progressive Web Apps (PWAs) [10] are gaining traction for their ability to combine the accessibility of web applications with the native performance of mobile apps. Their features include offline functionality, simple installation, and access to device capabilities such as cameras and geolocation. PWAs also enable unified development across platforms, reducing costs and maintenance efforts.

Our project builds upon these insights to deliver a PWA-based inventory management system tailored for enterprises of varying sizes. This project represents an initial exploration of PWA technology in the context of inventory management. Rather than aiming to create a superior alternative to existing, professional systems, our goal is to assess the potential of PWAs for such applications. We are focused on experimenting with and testing the unique capabilities of PWAs to determine their viability for small and medium-sized enterprises in real-world business applications scenarios. By integrating technologies such as Angular [4], Spring Boot [8], PostgreSQL [5], and AWS [3], we aim to create a scalable, cost-effective, and accessible solution that combines the simplicity of SaaS tools with the power of ERP platforms.

2.2 Tech Stack

Our project uses a modern and robust tech stack designed to deliver both scalability and reliability:

- Angular: A dynamic frontend framework for building interactive and responsive web applications. [4]
- Spring Framework: A backend framework that simplifies the creation of scalable and reliable APIs. [8]
- AWS (Amazon Web Services): A suite of cloud services for hosting, databases, and scaling applications etc. [3]
- Docker: A containerization tool for consistent and efficient deployment. [7]
- Hashicorp Terraform: Infrastructure as Code (IaC) to automate and manage cloud resources. [6]
- ZXing: A library for barcode and QR code processing. [12]

During the selection process, we considered other technologies:

- Frontend: Alternatives like Next.js [14] (for server-side rendering) and Vue.js [15] (for simplicity and flexibility) were evaluated.
- Backend: While we chose Java with Spring Boot for its maturity and scalability, Kotlin was considered for its modern syntax and null safety.

2.3 Project assumptions

At the beginning of the project we set multiple restrictions and assumptions to limit the scope and guide the development in the right direction.

We set out to create a simple and intuitive application to allow users across all skill levels to successfully use our application. Thanks to the PWA technology we wanted to achieve an application that is available on multiple devices with simple installation and a native feel.

We decided to limit the scope and support only a single company owning multiple warehouses. Those warehouses can contain items shared between them and have individual stocks. There will not be a registration form - each employee in the company will have to be manually added by the administrator.

3 RESULTS

3.1 Summary

The implementation of the project proceeded as planned and was successful. We managed to include all important functionalities, and comprehensively tested the usability and ease of use of PWA technology.

3.2 Functionalities

- CRUD operations, i.e. Create, Read, Update, Delete, allowing full modification of data stored in the database,
- Ability to scan barcodes and QR codes in an intuitive way to quickly access a specific product in stock,
- Inventory management across multiple warehouses,
- Management of users and their access to functionalities offered by the system layer,

3.3 Business goals

1. Enhanced Inventory Management: The system integrates barcode and QR code scanning to improve asset tracking, streamline inventory control, and provide quick access to details such as stock levels and locations.
2. Improved Efficiency: Simplified management of incoming and outgoing deliveries enhances organization, reduces processing times, and boosts overall productivity.

3.4 Technical goals

1. Cross-Platform Compatibility: Using Progressive Web App (PWA) technology ensures the app works seamlessly across devices while maintaining a single codebase, reducing development and maintenance efforts.
2. Scalable Architecture: The backend, designed with Spring Boot and deployed on AWS using Terraform, is built to handle increasing data loads and user demands.
3. Secure Data Management: With Amazon Cognito for authentication and other security measures, the app protects user data and ensures resource access is authorized.
4. Modern Development Practices Technologies like Terraform, Docker and Nginx highlight the use of efficient, modern tools for development, deployment, and infrastructure management.

Although this project is part of an academic course and lacks real-world deployment or metrics, it successfully demonstrates the feasibility of effective inventory management solutions while showcasing the educational value of applying industry-standard technologies.

3.5 Additional highlights

- Feasibility Demonstration: The app proves the potential of PWAs for inventory management, levaraging their core features and capabilities.
- Technical Design: Built with scalability and security in mind, the system employs modern tools such as AWS and Terraform, aligning with enterprise-level best practices and conventions.
- Practical Value: Offers hands-on experience with technologies like Angular and Spring Boot, preparing us for real-world challenges.

4 CONCLUSIONS

In summary, the application we designed is intuitive to use and provides the basic functionality required by warehouse workers. It will allow for more efficient and simpler inventory management.

Thanks to PWA technology, it is accessible on both mobile devices and computers, supports installation and quick and easy access in conditions of limited internet access.

What matters to a technological audience is that we have tested the capabilities of PWA technology and proved that it is viable for developing applications for multiple devices while maintaining a single code base.

5 FURTHER RESEARCH

For further development of the project, it would be important to consider what other functionality might be useful for warehouse workers and managers, such as:

- optimization of the layout of items in the warehouse,
- use of geolocation to determine the shortest route to the desired section in the warehouse,
- calculation of the route and cart capacity requirements to complete an order from the warehouse,
- expansion of the application to a commercial version that can support multiple companies in a comprehensive way,
- and much more.

6 ACKNOWLEDGMENTS

TBD

REFERENCES

[1] Cameron Priest (CEO). Tradegecko website. `http://www.tradegecko.com`. Accessed: 2024-11-30.

[2] Christian Klein (CEO). Sap website. `https://www.sap.com/cmp/dg/na-corporate-brand/index.html`. Accessed: 2024-11-30.

[3] Matt Garman (CEO). Aws documentation. `https://docs.aws.amazon.com`. Accessed: 2024-11-30.

[4] Google. Angular documentation. `https://angular.dev`. Accessed: 2024-11-30.

[5] PostgreSQL Global Development Group. Postgresql website. `https://www.postgresql.org`. Accessed: 2024-11-30.

[6] HashiCorp. Hashicorp terraform documentation. `https://developer.hashicorp.com/terraform/docs`. Accessed: 2024-11-30.

[7] Docker Inc. Docker documentation. `https://docs.docker.com`. Accessed: 2024-11-30.

[8] Rod Johnson. Spring boot documentation. `https://docs.spring.io/spring-boot/index.html`. Accessed: 2024-11-30.

[9] Oracle. Oracle netsuite website. `https://www.netsuite.com/portal/home.shtml`. Accessed: 2024-11-30.

[10] Alex Russell. Pwa website. `https://web.dev/explore/progressive-web-apps`. Accessed: 2024-11-30.

[11] Odoo SA. Odoo website. `www.odoo.com/`. Accessed: 2024-11-30.

[12] ZXing Team Sean Owen, Daniel Switkin. Zxing repository. `https://github.com/zxing/zxing`. Accessed: 2024-11-30.

[13] Tony G. Thomas Sridhar Vembu, Sreenivas Kanumuru. Zoho inventory website. `https://www.zoho.com/inventory/`. Accessed: 2024-11-30.

[14] Vercel. Nextjs website. `https://nextjs.org`. Accessed: 2024-11-30.

[15] Evan You and the Core Team. Vuejs website. `https://vuejs.org`. Accessed: 2024-11-30.