Michigan State University

Team Evolutio Group

#BIKES4ERP Tracking Initiative

Project Plan

Fall 2020

Evolutio Contacts:

Jordan Cobe

Bob Dyksen

Devin Stonecypher

#BIKES4ERP Contact:

Nicolai van der Merwe

Gert Vermeulen

Michigan State Capstone Team:

Caleb Duchan

Sam Peterson

Phillip Nguyen-Phan

Yash Sharma

Table of Contents

Executive Summary

Evolutio was founded in May 2018 and is headquartered in Chicago, Illinois. Evolutio leverages data platforms to provide businesses with services to monitor and optimize their enterprise systems. Evolutio's clients expect and receive a detailed view of their overall performance and inspired insight into how they can improve.

Elephants, Rhinos, & People (ERP) was founded with the aim to preserve and uplift the communities and wildlife of South Africa through rural poverty alleviation. Rural poverty alleviation strives to eliminate the need for illegal poaching through education and employment. #BIKES4ERP is a program that provides bicycles to students free of charge. With these bicycles, students can get to school faster and with more energy. With a complete and better education, students in the #BIKES4ERP will uplift themself and their communities.

As #BIKES4ERP evolved, ERP realized that the program had room for improvement. In that vein, #BIKES4ERP wants to make use of our team, in conjunction with Evolutio, to create a bike maintenance tracking platform. The #BIKES4ERP Tracking Initiative takes #BIKES4ERP to new digital mediums incorporating technologies to make the initiative more efficient and expandable.

#BIKES4ERP sees this as a needed step to make their philanthropic efforts more efficient. They hope to see this set of applications assist teachers, mechanics, and the organization in their collective functions. Each party's application will assist in an overarching system to improve bike maintenance, distribution, and tracking. These applications working in tandem will keep the gears turning, the students learning, and their grades rising!

Functional Specifications

This platform consists of three programs for three distinct groups of users: An Android application for teachers, an Android application for mechanics, and a web application for ERP administrators.

ERP teachers are tasked with overlooking the bikes of their students. The teacher application is a useful tool to keep track of each bike and request repairs when needed. Each bike is equipped with a unique NFC tag and can be scanned by the teacher to verify bike attendance. Once checked in, the teacher can file a maintenance report for a problematic bike. To account for varying bicycle knowledge levels, the app allows teachers to give as much or as little information as they choose with each report. The teacher can write a detailed note about the bicycle's specific issues or simply note its condition.

ERP mechanics' job is to ensure the continued functionality of the students' bicycles. Since a single mechanic handles the maintenance of hundreds of bikes, they can use all the help they can get in streamlining their workflow. The mechanic application provides a dashboard to assist in a mechanic's daily tasks. The mechanic app receives repair requests from the teachers. The mechanic application contains features for scanning bikes, recording completed repairs, and viewing previous repairs done on any bike. The app also has the inventory of the bike parts, which the mechanic can use to keep their supply of parts up to date.

There are a couple of key limitations that influence the functionality of both the teacher and mechanic applications. First, limited tech experience. There can be no assumption that either the teachers or mechanics are 'tech-savvy'. Until now, the ERP program has maintained itself primarily on pen-and-paper. To accommodate all users, the applications apps are intended to be simple and intuitive for individuals with any level of tech experience. Second, it cannot be assumed that either the teachers or mechanics will always have internet access every time they utilize the applications. The applications are intended to be fault-tolerant and functional, even when not connected to the internet.

The last group of users is administrators. The web application provides useful data and analytics using data from the teachers and mechanic applications. This application shows trends regarding bike usage and maintenance to help them make administrative decisions and evaluate the program's success.

Design Specifications

Teacher Application

Overview

The teacher application is an application designed to streamline each teachers' interactions with ERP bicycles. With this application, the teacher can check-in bicycles and alert the mechanic about needed repairs.

Bikes Screen and Bottom Navigation Tabs

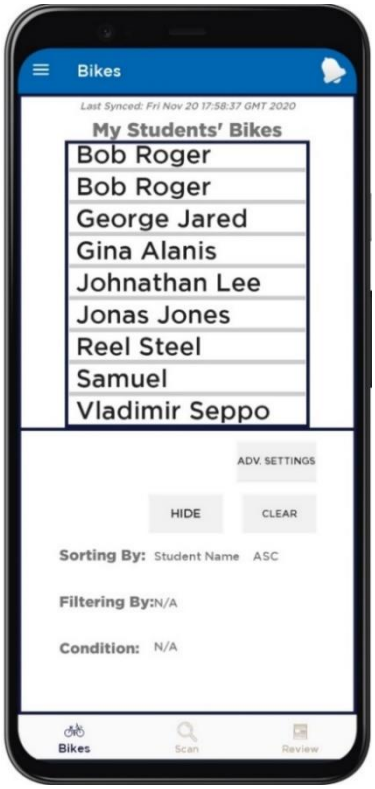


Figure 1: Bikes Screen

As seen in Figure 1, the "Bikes" screen showcases students' names that are assigned a bike. When the teacher is ready to scan a bike, they can proceed to the bike scanning screen by pressing the "Scan" button on the bottom. The bottom tab of these pages aids the user in navigating the application. The "Bikes" button always takes the user back to this screen from any page. The "Scan" button takes the user to the scanning screen. The "Review" button displays any bikes that have been checked-in so far. This bottom tab persists through all screens in the application. The teacher can narrow down the bike list through the "Advanced Search" option using the available sort fields and filters.

Scan Screen and Successful Scan

Figure 2: Scan Bike

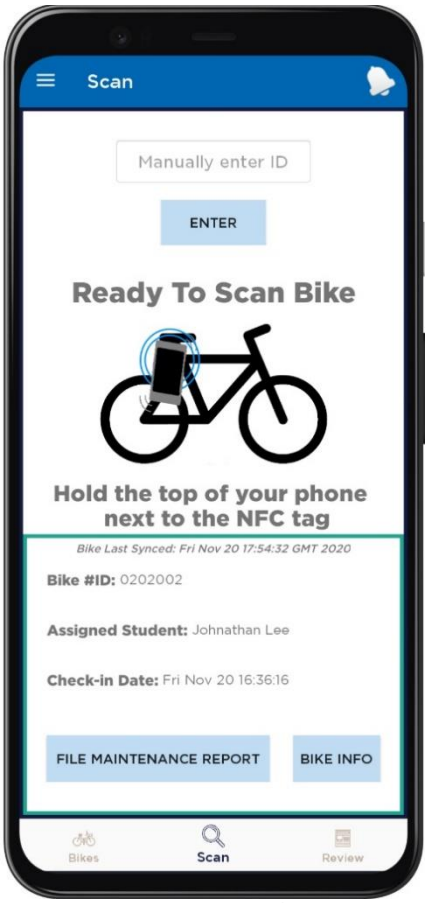


Figure 3: Successful Scan

Figures 2 and 3 show the scan screen and the successful scan screen. Once on the scan screen, the teacher is instructed to scan a bike by holding their phone close to a bicycle's NFC tag. The teacher also has the option to type in the bicycle's identification number manually. The phone vibrates on a successful scan and moves to the successful scan screen.

Once a bike is scanned, the user has multiple options. The teacher can note the bike's status, detailing any issues it has through the "File Maintenance Report" button or view the bike information by pressing the "Bike Info" button. The teacher can also scan another bike directly from this screen, following which the screen immediately updates with the newly scanned bike's information.

Bike Info, File Report, and Review and Finish Screen

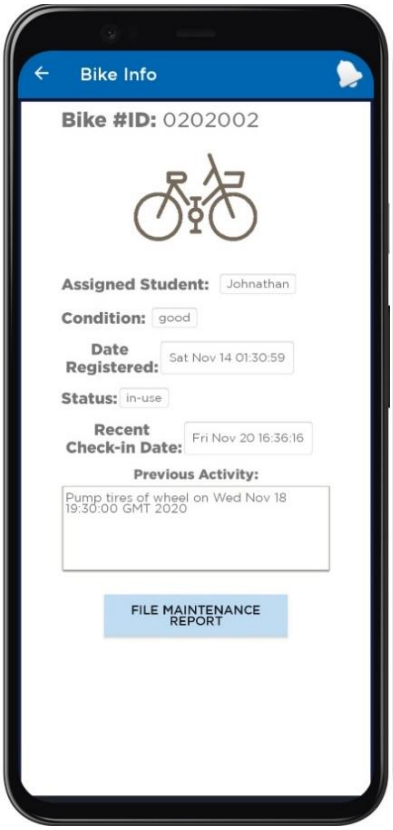


Figure 4: Bike Info

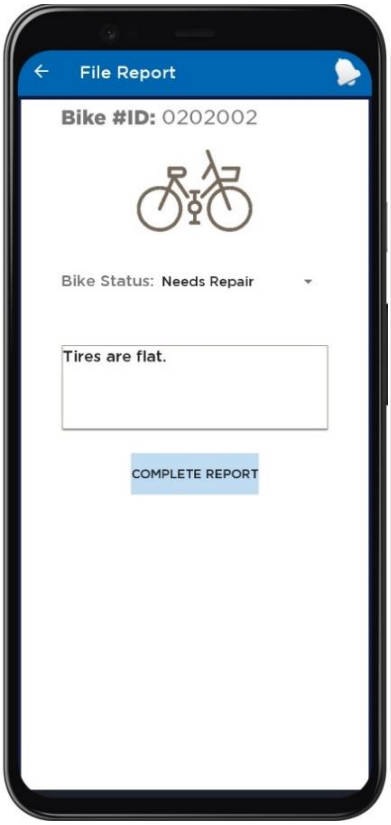


Figure 5: File Report

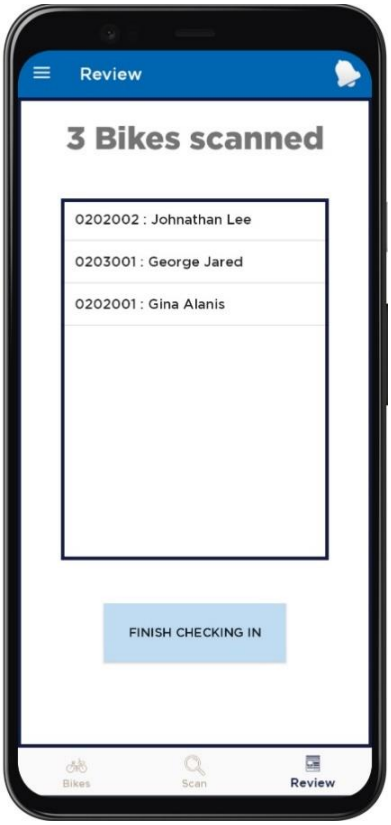


Figure 6: Review

Figures 4, 5, and 6 show the screens used to view, create, and submit bicycle data. The “Bike Info” screen lists the student to whom the bicycle is assigned and its previous activity. The “File Report” screen gives the teacher 3 status levels: All Good, Needs-Repair, and Non-Functional. The teacher also has the option to add a note to their report. The teacher can then submit their status report and return to scanning bicycles.

Once all the bicycles are scanned, the teacher can move to the “Review” screen with the navigation tab. This screen displays the number of bicycles they have scanned and a list of all the bike identification numbers with the assigned student’s name. When they are sure they have checked in all available bicycles, the teacher can press “Finish Checking In’ to submit the day’s check-ins and maintenance reports.

Mechanic Application

Overview

The mechanic application is designed to organize and optimize a mechanic's workflow. The application allows the mechanic to view needed repair requests, view repair tickets created, log their inventory, and share their data with ERP Admins. The mechanic application shares the Account Setup and Login screen with the teacher application.

Bikes Page and Navigation Tabs

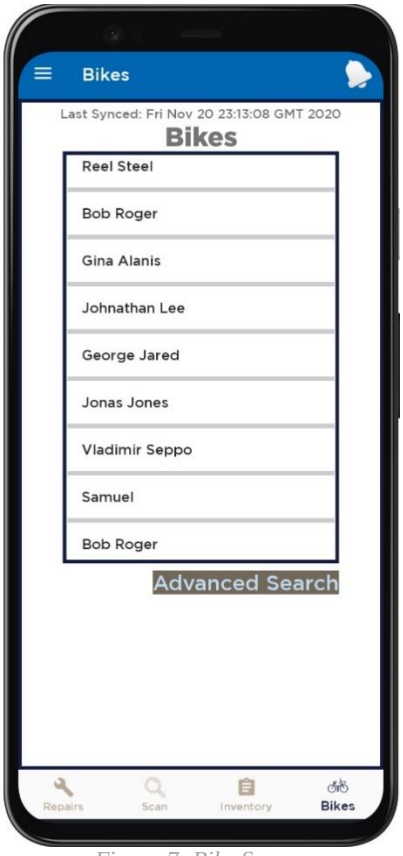


Figure 7: Bike Screen

Figure 7 shows the "Bikes" screen for the mechanic application. The "Bikes" screen displays a list of the student's bikes where the mechanic can select one of them to view detailed bike information. The mechanic can move to their open repair requests, their history of repairs, or directly to bicycle scanning. The navigations tabs stay aligned to the bottom of the application. The Bikes button always links back to this screen; the scan button directs the user to the "Scan Bikes" screen; the inventory button links to the inventory and the repairs button directs to a page that contains information on requested and ongoing repairs. Like the teacher application, the mechanic application can sort or filter through the bikes list by selecting sort fields and filters.

Register and Assign Bikes Screen

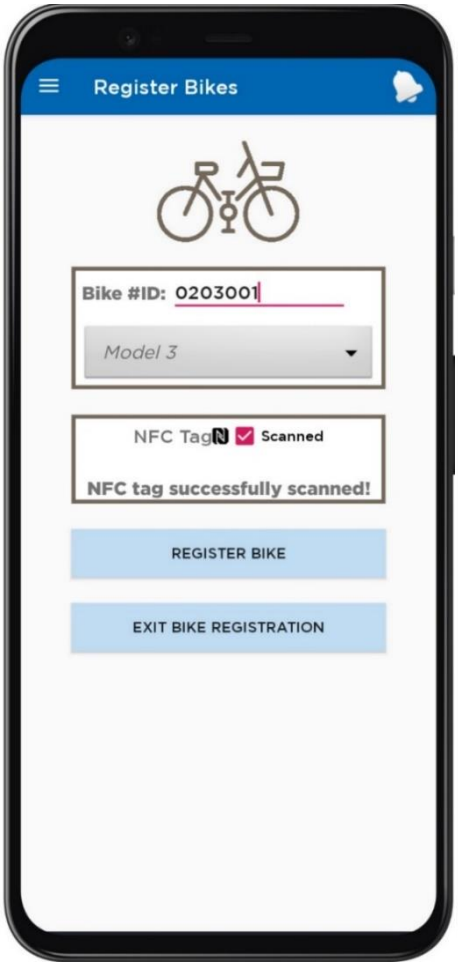


Figure 8: Register Bike Screen

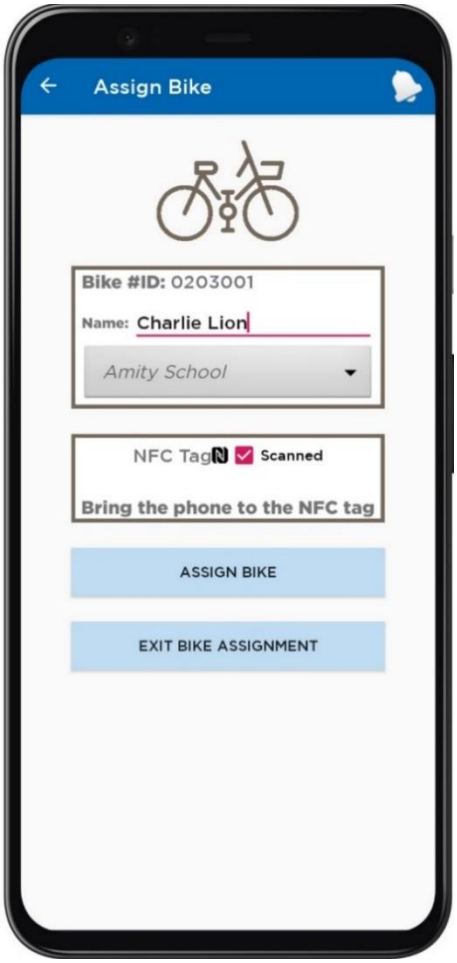
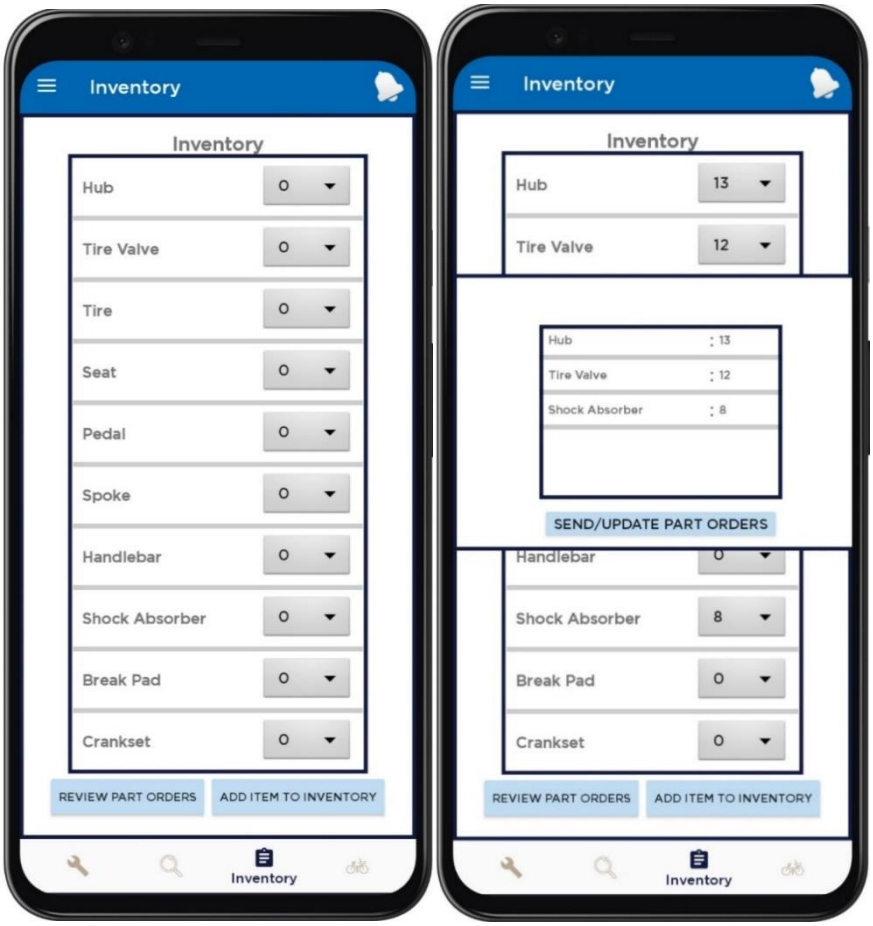


Figure 9: Assign Bike Screen

Figures 8 and 9 show the bike registration and assignment process for the mechanic application. The “Register Bikes” screen allows the mechanic to register a bicycle. To register, the mechanic inputs a new Bike ID supplied by ERP and selects the bike model. The Bike ID can then be written to the NFC tag. After this has been completed, the bike registration is completed by pressing the “Register Bike” button. On the assign page, a bike to assign is selected by scanning a registered NFC tag. Then, the student name and school is selected. Assignment is completed by pressing the “Assign Bike” button.

Inventory Screen



Figures 9 and 10: Inventory Screen with and without Review List

Figures 9 and 10 show the Inventory screen. The inventory is essential for a mechanic to stay organized and prepared for needed repairs. The inventory screen shows an itemized list of all parts the mechanic uses during repairs. The mechanic can select parts and quantities to populate an order. ERP admins can view the orders to keep track of which parts the mechanic is buying.

Repair Request and Repair Ticket Tabs

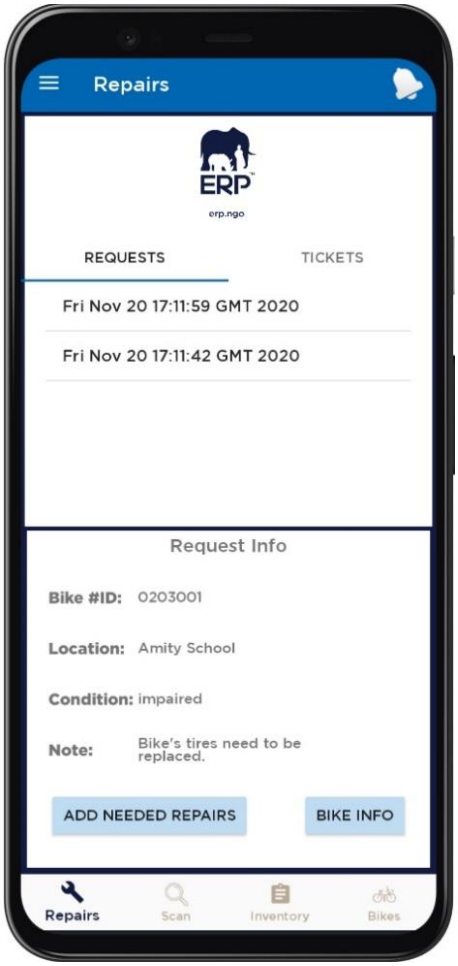


Figure 11: Repair Requests Screen

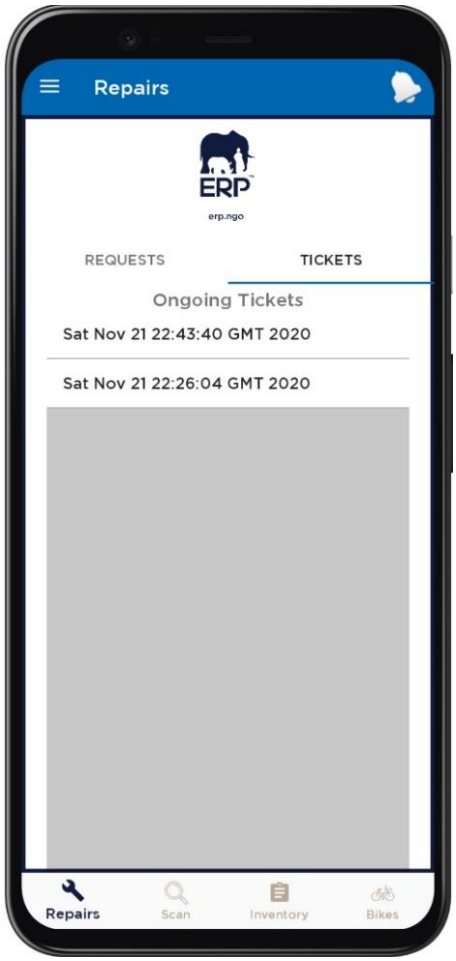


Figure 12: Repair Ticket Screen

The Repair Request and Repair Ticket tabs are shown in Figures 11 and 12, respectively. The requests tab shows maintenance requests that have been submitted by teachers. By selecting a request or scanning a bike, the mechanic can open a new repair ticket (shown in figures 16 and 17) to address a given request. The tickets tab shows the ongoing repair tickets. Through this menu, a mechanic can access any ticket to view and update as needed.

Scan Screen and Successful Scan Screen

Figure 13: Scan Screen

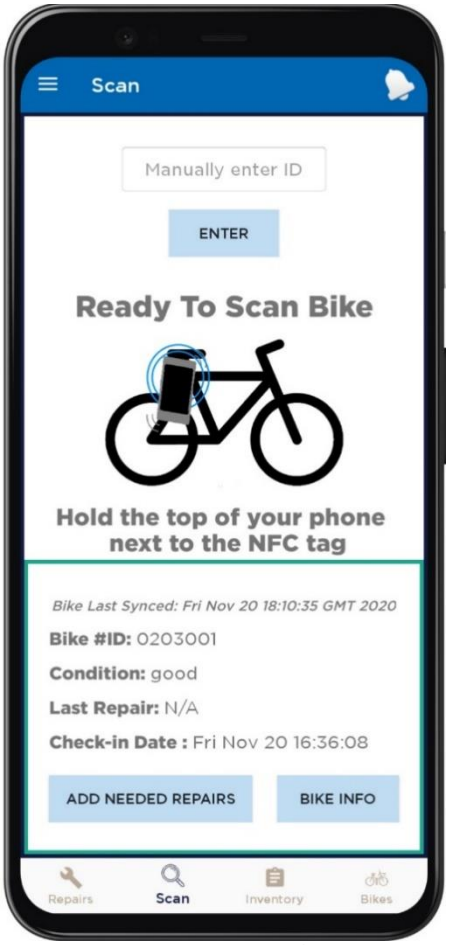


Figure:14: Successful Scan Screen

Figures 13 and 14 show the scan screen and successful scan screen. The mechanic can scan the bicycle's NFC tag to bring up an abbreviated list of important information. The mechanic is then able to open the Bike Info page for the bicycle, open a new repair ticket with the "Add Needed Repairs" button, or directly scan another bicycle to bring up that bicycle's information.

Bike Info Screen

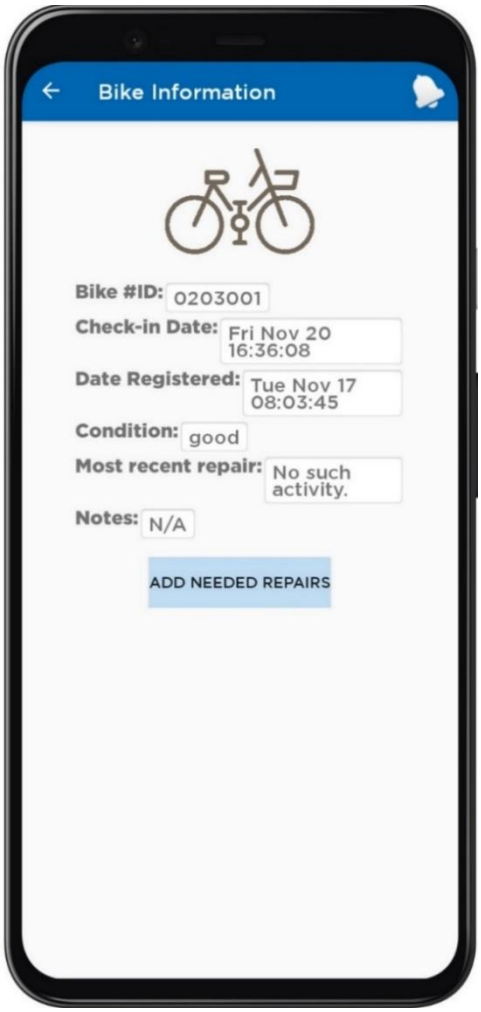
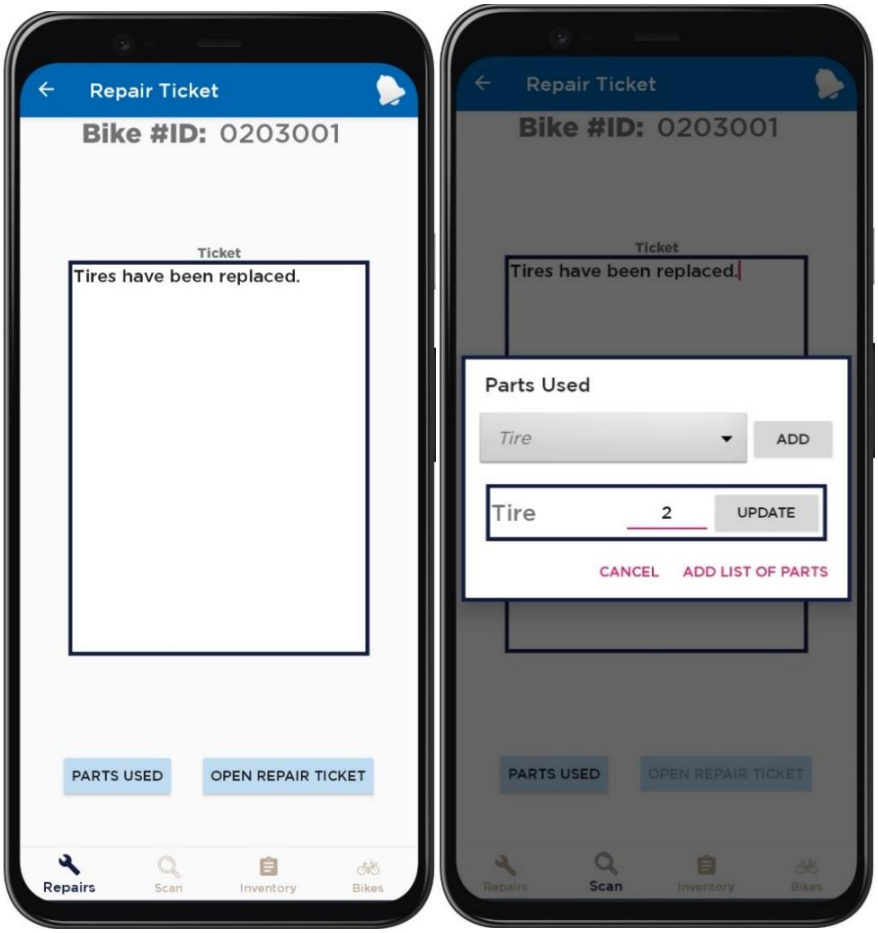


Figure 14: Bike Info Screen

Figure 14 shows the “Bike Info screen”. This screen helps the mechanic stay organized and current on a bicycle’s status. This screen contains detailed information about all previous repairs and notes about the bicycle. The “Add Needed Repairs” button can be used to create a new repair ticket from this screen.

Repair Ticket Screen



Figures 16 and 17: Repair Ticket Screen w/ and w/o Parts Used Pop up Window

Figures 16 and 17 show the “Repair Ticket” screen. The mechanic describes the maintenance done on a given bike and logs what parts are needed to complete the repair on this screen. The mechanic can close a repair to resolve it immediately or keep it open to be accessed and completed later.

ERP Web Dashboard

Overview Page

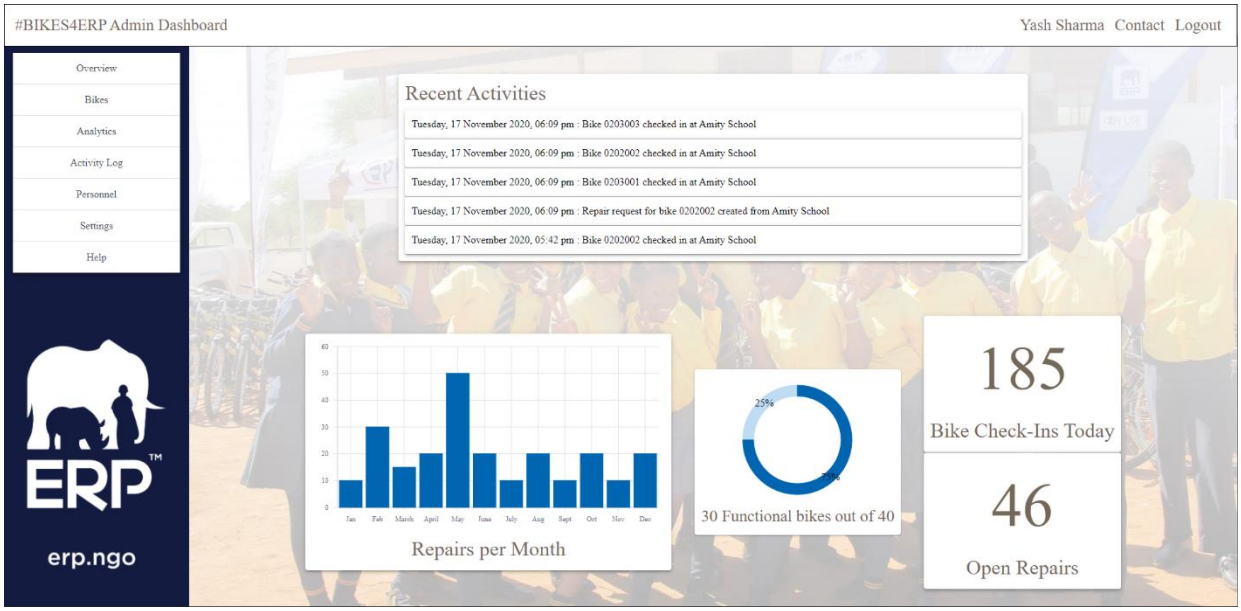


Figure 18: Overview Page

Figure 18 shows the web app overview page. The dashboard shows useful data and analytics taken from the two Android apps' activity, such as daily number of bike check-ins, open repair requests, and check-in rate for schools.

Analytics Page

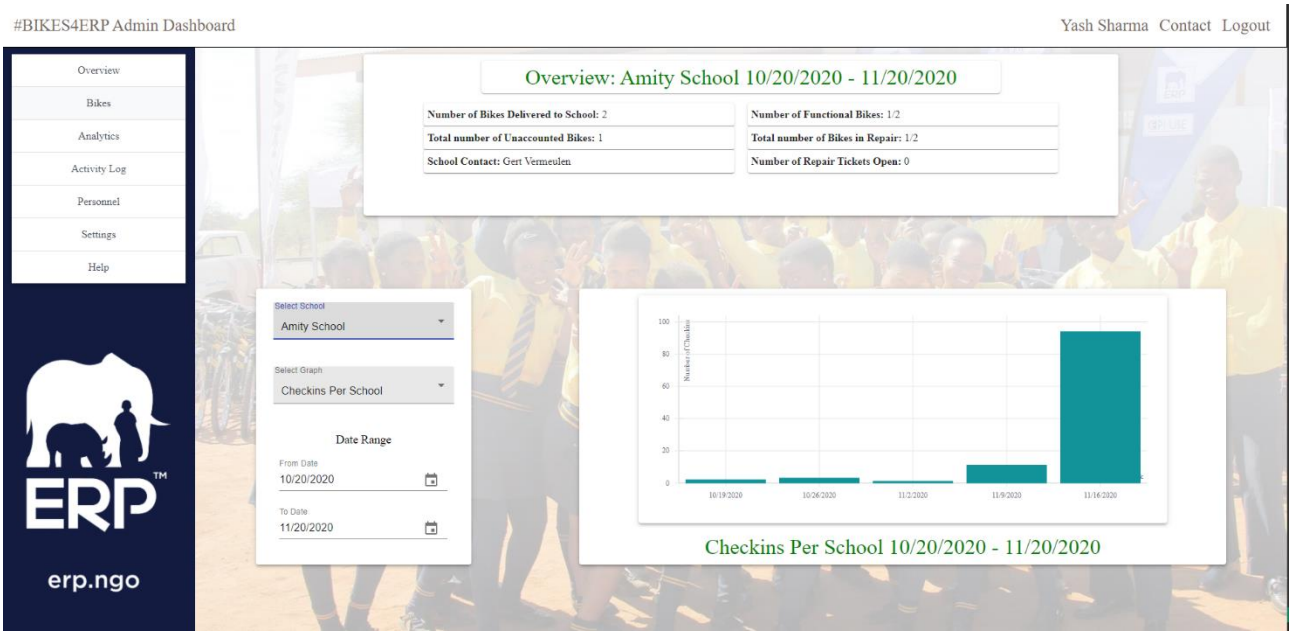


Figure 19: Analytics Page

Figure 19 shows the analytics page, which graphically represents useful data such as bike attendance, repairs, and part usage in repairs. Admins can change the graphs on the screen to show different analytics. The graphs show check- ins, repairs, and other metrics in relation to specific time intervals. This functionality gives the ERP admins detailed insight into specific aspects of the program and helps them track the program's efficiency.

Technical Specifications

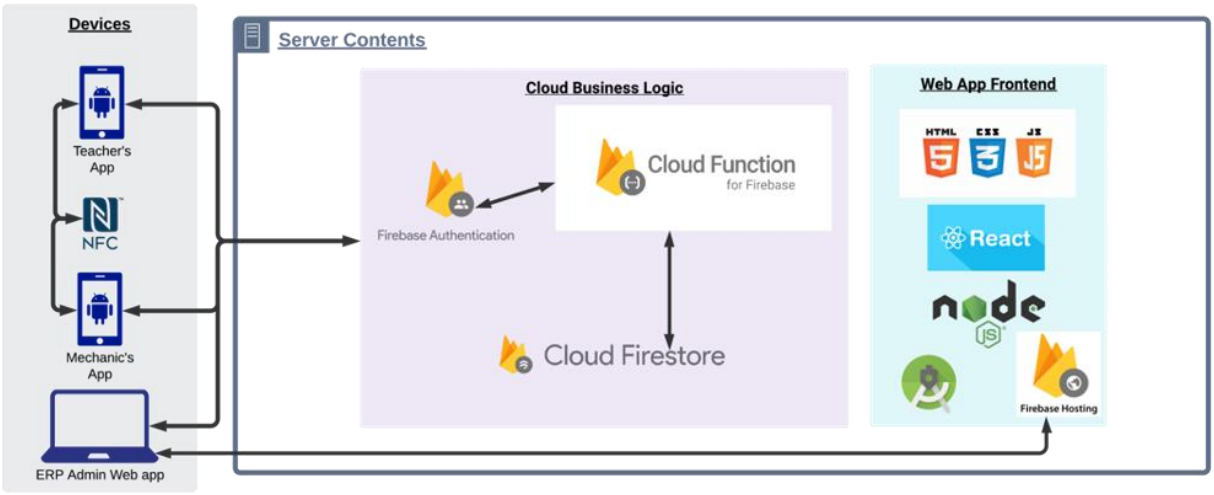


Figure 20: System Architecture

The project's software component consists of two native Android applications for the teachers and mechanics written in Java and a web application for ERP Admins built on top of HTML+CSS and ReactJS.

Various firebase services are used to service the android apps and the web app. Firebase Cloud Messaging is used to allow communication between the back end and the Android apps. Firebase Authentication is used to authenticate users such as teachers, mechanics, and admins. Firebase Cloud Firestore is used as a real-time database to store and retrieve data pertaining to the android apps and the web app. Firebase Cloud Functions is used for normalizing the Firestore collection and run heavy computations for analytics on the web app.

The project's hardware component consists of NFC tags, which are attached to the bikes in the field and allow quick interaction with the android apps.

System Components

Web Application

React

React is a front-end JavaScript library for building user interfaces. The web app talks to Firebase Firestore and Cloud Functions to collate data about the bikes such as last check-in, repair statistics, student grades (who has been allotted a bike) and shows it to the ERP admins.

Node.js

Node.js is a JavaScript runtime environment that executes JavaScript code outside a web browser.

Firebase

Firebase is a platform developed by Google for creating mobile and web applications. The system leverages the Firebase Authentication to authenticate users on the apps and Firebase Cloud Messaging (FCM) service to talk to the mobile apps and deliver updates from the back-end server to the mobile apps and vice versa. FCM has reliable data sync ability, which is crucial for users in limited network connection locations.

Mobile Application

Android Studio

Built on JetBrains's IntelliJ IDE, Android Studio is the official integrated development environment for Google's Android operating system. The user interfaces are generated through Android Studio.

Firebase SDK

Cloud Firestore supports SDKs for Android, iOS, and Web. Real-time updates and offline data persistence are supported by mobile and web SDKs. An Admin SDK is used to initialize access to Cloud Firestore and other services from a single SDK. The Firebase Admin SDKs support Cloud Firestore access in Java, Python, Node.js, and Go.

Android SDK

The Android SDK and API level 30 allow the development of software for devices running Android. The target builds for both the android apps is Android 9 up to Android 11, which will cover all the teacher and mechanic's Android phones.

Hardware

NFC tags

Near Field Communication (NFC) is a technology that allows quick transfer of data to and from NFC tags and mobile devices. The tags can store data from 1KB to 8KB. Data about the bike, such as the Bike ID, name of the student the bike is assigned to, date of bike assignment are stored on the NFC tags. The tags are readable from the teacher's app and writeable and readable from the mechanic's app.

Risk Analysis

Creation of Data

Difficulty: Hard

Description: There was no data provided to our team. The mechanic writes his repairs on pieces of paper along with the inventory. A suitable design is needed to provide an easy representation of the data for the admins to look at.

Mitigation: Build the application(s) to learn as the database grows, finding relevant data points about the bike models and repair.

Limited internet reception for end-users

Difficulty: Hard

Description: Due to potentially harsh conditions in South Africa, the internet is a scarce resource. The teachers and mechanics must rely on the school's Wi-Fi or cellular data to use the app. Only a limited amount of data can be sent to and from the apps.

Mitigation: Use Firebase Cloud Messaging's fault-tolerant service to reliably send data when connectivity is available that would deliver a 4KB message directly to our database and the apps. A sub-set of the data is going to be stored locally on the phone for the mechanic's convenience.

Unforgiving environment for bicycles and technology

Difficulty: Medium

Description: Again, with the harsh conditions in South Africa, there needs to be a way for users to scan a bike that does not deteriorate over time. Bikes rust and give the user a hard time to identify them. Technology is limited in terms of what was provided.

Mitigation: Near-field communication (NFC) provides a way for the user to use their device to read and store information. Four phones were bought specifically for this program. Bike IDs are written and stored onto the tag to pull up the latest information on a bicycle. NFC tags are robust, waterproof, and securely attached to the bicycles.

Authentication system

Difficulty: Easy

Description: There would be an authentication system for the teachers and mechanics to prevent outsiders from logging in. Our team has no confirmation if ERP has its own authorization/authentication system.

Mitigation: Google Firebase authentication is used with Firebase to handle logging in and out of the android apps and the web app.

Schedule

Week 1 (9/2 - 9/5):

- Teams were created and assigned a project proposal
- First Client Meeting
- Slack created for team communication

Week 2 (9/6 - 9/12)

- Met with EPI-USE technical contact
- First Triage meeting with James Mariani
- Start project plan rough draft
- Set up project GitLab Team and repositories

Week 3 (9/13 - 9/19)

- Status report presentation
- Contact the on-site mechanic for design insight
- Created wireframes for teacher and mechanic applications
- Work on Teacher Prototype
- Created Data Model for the project and UML for the API

Week 4 (9/20 - 9/26)

- Finalize designs for teacher and mechanic applications
- Demoed Teacher application/prototype to clients
- Finish project plan document and project plan presentation
- Finish Data Model for the project and UML for the API

Week 5 (9/27 - 10/3)

- Turn in and present Project Plan Document and Presentation
- Determine API endpoints
- Determine the authorization and authentication system

- Initialize Firebase Cloud Messaging
- Solidify Back-end architecture
- Work on Mechanic Prototype
- Contact Graphic Design Lead
- Start Webapp Design

Week 6 (10/4 - 10/10)

- Receive hardware for testing applications
- Plan NFC chip attachment procedure
- Finalize Webapp Design
- Start Implement authorization and authentication system

Week 7 (10/11 - 10/17)

- Prepare Alpha Presentation
- Working Frontend for Teacher and Mechanic Application
- Bike registration demo
- Basic check-in functionality
- Finalize authorization and authentication system

Week 8 (10/18 - 10/24)

- Present Alpha presentation
- Detailed bike check-in functionality
- Determine data we are collecting for prediction

Week 9 (10/25 - 10/31)

- Initialize React Web app front-end
- Initialize predictive model for bike data
- Cleaning/Debugging Applications and Webapp

Week 10 (11/1 - 11/7

- Finalize predictive model
- Add functionality to the Web app
- Cloud functions
- Finalize wireframes for Web app

- Implement the Overview page for the web app
- Repair Requests and Repair Ticket system

Week 11 (11/8 - 11/14)

- Prepare Beta presentation
- Finalize Inventory functionality
- Implement Bikes and Analytics page for the web app

Week 12 (11/15 - 11/21)

- Implement final functionality for both apps
- Android application notifications
- Implement the Activity Log and Personnel page for the web app
- Clean up and debug apps
- Present Beta Presentation
- Complete testing
- Complete documentation of Project Plan
- Create Storyboard for Project Video

Week 13 (11/22 - 11/28)

- Work on Video Team Status Presentation
- Work on the visual experience of Web and Android applications

Week 14 (11/29 - 12/5)

- Work on the final presentation
- Finalize Project Video
- Training Video

Week 15 (12/6 - 12/12)

- Submit project video
- Submit All deliverables
- Design Day