ISU BikeShare
Design Document

May1619

Advisors: Mark Kargol, Mani Mina

Kaitlynn Kruse
Jeff Neel
Jorden Studer
Mohammad Darwish
Molly Schaeffer
Brian Anderson
Kyle Perkins

Contents

Project and Document Definitions

Project Information

The Iowa State Bike Share project is an initiative to provide a bike-sharing service to Iowa State University and the Ames community. The project requires considerable work from a collection of multi-disciplinary teams to create a working solution. As Electrical and Software Engineers, we must develop and expand a working program that can fit in with the work of other teams working on this project. This Bike Share project is expected to provide a solution at or below the cost of solutions that have already been created at other universities and large cities.

The electrical team has been working on redesigning and perfecting the docking system electronics that help control the distribution of and manage the bikes. During the fall semester they fixed the previous prototype stand for demonstrations, worked with the mechanical engineering team to weatherproof the stand, and began simplifying the existing stand electronics. The electrical team also temperature tested components and ordered and received hardware for a second docking station.

The software team will be working on backend administration web application and frontend, user-facing mobile and web applications. As of the end of the first semester of the project, the web application GUI and CSS have been completed and the administrative portion connects to the database. Our server is also set up for Django to continue web application development during spring semester. The Android application GUI and main functionality have been designed and the app connects to the database. The app also has completed maintenance request and contact-us forms.

This project has currently been worked on for a few years now and is set to have a deadline of June 2016.

Project Goals

Our team’s goal for this project is to deliver a reliable bike sharing system that can be used throughout Iowa State University and the city of Ames. The software team is working on developing multiple applications which can ensure the project runs smoothly by developing an Android application, user web page, admin web page. These applications will help make the bike share user friendly, reliable, and secure. The hardware team is working on improving the current hardware for the system.

Deliverables

- November 2015
  - Circuit Design & PCB Design
  - Google Maps Prototype in Android Application
  - Server Setup
- December 2015
  - Hardware Connected
  - Troubleshooting/Testing
  - Developing web application code
  - Dock interfaced over web talking to database
- January 2016
  - Working hardware
  - More advanced prototypes of Android and web applications
  - API communication from dock to server
- April 2016
  - Completed testing plan
- May/June 2016
  - Testing plan executed
  - Working web and Android applications

Document Purpose

The purpose of this document is to outline the design mechanism and resources we will be using when creating the Bike Share software and hardware for our project. What follows is a document including system level designs and detailed descriptions of our implementation we plan on creating.

System Level Design

System Requirements

Bike Stand Functional Requirements

1. When a user swipes their ID card then the system will unlock a bike for the user. The System will log the bike ID to the individual.
2. When a user puts the bike into the station then the system will lock and check in the bike identifying the user interactions as necessary.
3. When an unauthorized user attempts to check out a bike then the system will not unlock a bike.
4. When a user tries to put something in the rack that is not a bike with a valid RFID, the bike stand won't lock.

Android Application Functional Requirements

1. Keep track of statistics of the user's current ride that will start running when a user checks out a bike
2. Users can report that a bike is damaged
3. When a user checks in a bike and marks it as damaged then the system will mark the bike "in need-of-service".
4. When the user completes training, the system will be able to correctly register their account with the information they enter
5. A user will be able to view all bike stations on a map and see more information about the stations including bikes available and stations available.

User Web Application Functional Requirements

1. Keep track of statistics of the user's current ride that will start running when a user checks out a bike
2. Users can report that a bike is damaged
3. When a user checks in a bike and marks it as damaged then the system will mark the bike "in need-of-service".
4. When the user completes training, the system will be able to correctly register their account with the information they enter
5. A user will be able to view all bike stations on a map and see more information about the stations including bikes available and stations available.

Administrative Web Application Functional Requirements

1. An administrator will be able to see bikes that have been marked as damaged by users in order to perform maintenance.
2. When an administrator performs maintenance on a damaged bike then the system will mark the bike available to be checked out.
3. An admin can disable checkout of a bike through the admin web application.
4. An administrator will be able to view bike transactions and filter it by bike ID, station ID, student or faculty ID, or date.
5. View, add, edit, and remove bikes from the web admin page.
6. Filter search results from the list of all bikes and docks.
7. View a google map of the dock locations.
8. View, add, edit, and remove users and administrators

Raspberry Pi Application Functional Requirements

1. Ensure clear status readout from the raspberry pi
2. Ensure Reliable data Connection through the Transport layer

Hardware Functional Requirements

1. Use simple and recreatable wiring for the docking station
2. Ensure that all components and wiring for the system neither interferes with nor is interfered by the opening or closing of the docking station.
3. Ensure that the docking station can consistently and reliably identify bikes being checked into the station and identify if the bike is properly secured.
4. Work with the IE 409 team to ensure that the docking station meets their requirements.
5. Eliminate the use of custom designed components.

Non-Functional Requirements

1. Only students, faculty, and registered residents of Ames should be able to checkout a bike
2. The checkout process should take less than 5 seconds.
3. All bike transactions should be retained a minimum of 2 years for auditing purposes.
4. Map should load on applications within 5 seconds with all stations loaded

Functional Decomposition

Raspberry Pi Application Functional Decomposition

The Raspberry pi will effectively send and receive commands from the central server and will process the commands it receives in real time. The raspberry PI will also handle all actions performed by a user on the dock within an acceptable response time; it will also display the status of these commands in a clear and concise way to the user.

Administrative Web Application Functional Decomposition

The administrative web application will allow administrators of the system to log in and view higher level details about the system that would be hidden from normal users. Administrators will be able to view details on all bikes and bike stands in the environment. From this data, they can remotely perform actions on the docks such as disable checkout, enable checkout, put in maintenance mode, and release maintenance mode. The admin will also have the ability to modify the users on the system and add or remove them as necessary.

Android Application Functional Decomposition

The Android application will be able to register their account with the information that they enter once they complete the registration to use the bikes. Through use of the Android application the users can view all of the bike stations on a map. The user will be able to see relative information about the individual bike stations which would include information like available bikes or docks that are open to return a bike. From the Android application users can submit maintenance requests for a bike, which would mark the bike as “in need of service” until an administrator addresses the maintenance request. Another feature of the application will be the ability for the

user to track their bike ride so users can see their regular paths with the bikes. This will help analyze heavily traveled bike paths and average times and distances.

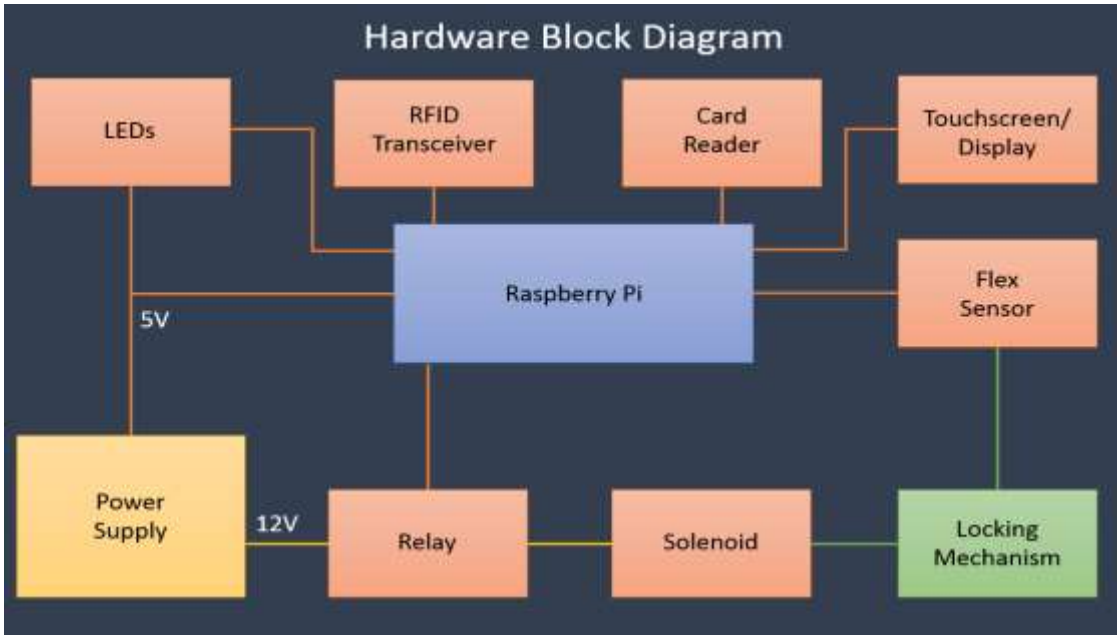
Web Application Functional Decomposition

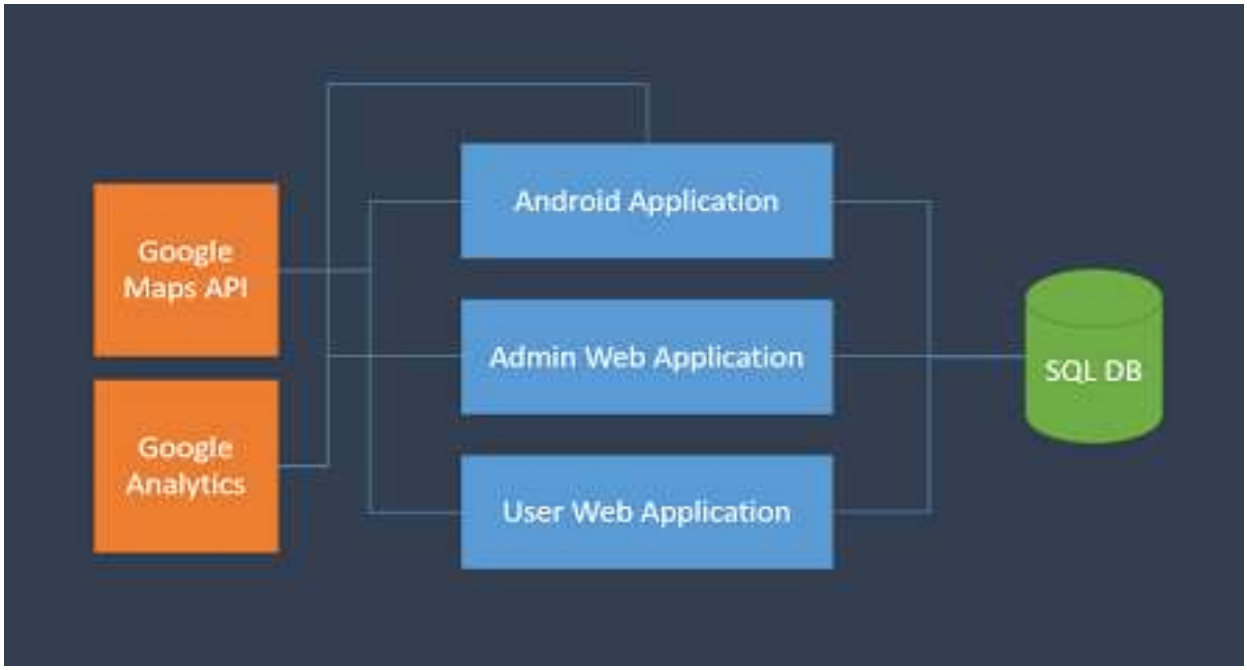
The Web Application will consist of two end points, the user and the administrator. The user-facing web application will be able to register their account with the information they enter once they complete the registration to use the bikes, this will be the same login for the Android Application. The user will be able to see relative information about the individual bike stations. The user could submit forms for requesting a bike for a special need whether it be a demo, project, or show that they need it for. The user could also submit maintenance requests for a bike that they have previous rode on. The administrative side will be able to monitor all the bikes, stands, and users and be able to see what bikes need fixing.

Hardware

In order to simplify the prior docking station, we will be using fewer components to attain the same functionality. We will also be individually routing wires, color coding them, and using wire-ties as necessary to make it clean and easy to see what's what at a glance. Most, if not all the components we will be using will come from known vendors and require little-to-no custom modifications in order to make the system easily reproduced. Reliability will be considered, and each component will be placed to ensure maximum reliability. Other bike-share teams will be discussed with concerning plausibility and component placement as needed.

Conceptual Block Diagrams





Detailed Descriptions

System Overview

Our senior design team is mostly split into two groups: the software team and the hardware team. The software team will be working on a user interface, an administrative interface, a server, and an android app. All of these things will be connected in some form to what the hardware team will be working on: the bike-stand electronics and bike electronics. These things are detailed in the sections below.

Software Specification

We will have independent software applications running at the same time. We will have an Android application, User web application, Admin web application, and Raspberry Pi application. The applications will be communicating with a server application handling all the database and server side communications. The Android application will use Java. The User and Admin web pages will use HTML and the server application is currently using Python. Our database instance is a MySQL database with tables: users, transactions, bikes, docks, and error reports. The server code is currently using a REST API implementation.

The android and web applications will use the server code's API to get information from the database and about the system. They will use HTTP requests to retrieve or post information to our server instance.

Hardware Specification

A simplified block diagram of the hardware system is shown in the conceptual block diagram section of this document. We will be powering the system with a typical PC power supply: the 5V line will run to the LEDs on the bike-stand as well as the raspberry pi; the 12V line will run to a relay (controlled by the pi) and then go to the solenoid to control the locking mechanism.

The pi will also control the LEDs on the bike-stand, the touch-screen/display, the RFID transceiver, the card reader, and take readings from a flex sensor. The LEDs will be used for indicators and visual feedback for the user, the touch-screen/display will be also be used for visual feedback, as well as possibly contain a map that tells the user where bikes are available or where bike-stands are available, the RFID transceiver will be able to scan a piece on the bike and tell if it is present, the card reader will read a ISU card or CyBike-issued card and allow the user to check out the bike, and the flex sensor will be positioned on the locking mechanism to tell the pi if a bike is in the lock.

Implementation Challenges

One of the challenges we are facing with this project so far is gaining access to Iowa State's system. We have been unable to obtain access to Iowa State's database that has information regarding students and RFID numbers. From what we have gathered so far, we are not going to get access to this and we plan to handle this information separately.

Because the bike-share system is such a large project involving many different groups of people, another main challenge in completion will be to communicate effectively with all of these different groups. There is an industrial team, a mechanical engineering team, two computer science groups, as well as our team. So communicating effectively and making sure each team is on the same-page with updates to the system will definitely be a challenge.

Our system will also require constant internet connection. Our preferred method of making this work would be to simply use wifi; however, connections on and off campus are neither great nor secure. This could be mitigated by using an ethernet connection, but the problem with this requires running a cable to each stand location. Overall, we would like to use wifi if we can get a good, secure connection in all locations.

The docking stations must be able to identify accurately when a bike is checked in. This is made difficult by the fact that the locking mechanism on the bike station must be both reliable and made difficult to 'hack'. Another challenge with the check-in system is that a short range RFID and complementing receiver must properly mate when the bike is put into the locking position. This requires the RFID reader to have a very specific read range as well as being oriented on the stand and bike perfectly.

Testing, Procedures, and Specifications

Hardware Testing

The bike stand and bike both contain components that need to be tested individually as well as put together and tested in the environment in which they will be applied. Individually, each component can be tested to make sure it is either providing or receiving the correct voltages and working as it should through use of a multi-meter and/or visual feedback of the component. Each component will then be incrementally added to the system and tested in a similar manner until the full system is up and running.

The bike stand needs to be reliable in all seasons and conditions so all of the hardware needs to be tested in the corresponding range of temperatures. In order to evaluate this the docking hardware will be tested in both high temperature conditions as well as freezing temperatures. To test the high heat requirements all of the electrical hardware will be placed into a reflow oven and tested at 100 degrees Fahrenheit. To test its performance in low temperatures the components will be tested in either a freezer or outside this winter. The stand will also need to be made to withstand all forms of precipitation but this will be the responsibility of the IE409.

Software Testing

The software applications will have a variety of aspects to be tested. The Applications will be tested on bike share environment. All functional requirements will be tested as well as ensuring that there are no unexpected actions in the environment. We will conduct a series of automated and manual tests to ensure the functional requirements work as needed. The system must function in a real world situation.

Conclusion

As Electrical and Software Engineers, we will use the methods detailed throughout this document to develop and expand a working program that can fit in with the work of other teams working on this project. The electrical team will be mostly working on redesigning the old bike-stand, and the software team will be working on backend administration web application and frontend, user-facing mobile and web applications as discussed throughout this document. This Bike Share project is expected to provide a solution at or below the cost of solutions that have already been created at other universities and large cities, as well as be more efficient than the design of last year's group. This project is set to have a deadline of June 2016.