# CS476 Requirements Document
# Transit Trackers
## Version 1.0
15 November 2024


Drake Stanton, Lauren Bushman,
Alonso Jimenez Alamilla, Benjamin Griepp
NAU Shuttle Services
Paul Deasy

Accepted as baseline requirements for the project:

Client: Team:

# Table of Contents

# Introduction

Transportation is a vital aspect of any efficient modern society. Being able to reliably get to where one needs to be is a daily necessity for nearly all members of an organized community. No matter what role you fulfill in your community at a given time – worker, student, consumer, or caretaker – there is always somewhere to be, often with time restraints attached. Having reliable transportation is not just a luxury anymore, it's a necessity.

Beyond that, many community members don't have access to personal vehicles and have to rely on public transit. A city or community having reliable and safe public transportation is crucial to the wellbeing and productivity of its members. To keep these services efficient, there are many factors that must be accounted for like determining the right amount of vehicles for the number of riders, considering changes in traffic flow, and choosing efficient routes that are in close proximity to places where people often want to go. In addition, an often overlooked factor is the drivers. Every vehicle requires a driver, and these drivers must have a clear schedule and understanding of when they are meant to take over a vehicle and begin a route.

Northern Arizona University is a community that relies heavily on public transportation in the form of buses or shuttles, managed by the NAU Shuttle Services department. Thousands of students rely on Shuttle Services to get around campus and get to class and work on time. In 2023, they provided 1.3 million individual rides. Our client, Michael Seitz, and his colleagues, compose the management team for NAU Shuttle Services, and are the ones that carefully consider all the factors that were outlined above – including the scheduling of their drivers. With nearly two dozen vehicles and over 35 drivers, they require the assistance of technology to efficiently schedule the daily routes. However, their current system has many shortcomings, and

our team hopes to establish a new web application that will simplify the scheduling of drivers and save hours of manual work every week.

## Problem Statement

The current system used to create the daily driver schedule is ineffective and time-consuming. The management team must be very involved and do many tasks manually that could be automated instead. The system consists of several parts: a third-party scheduling system called Shiftboard, a spreadsheet, and another third-party software called BusGenius that tracks the physical location of the buses and when the drivers begin and end their shifts. A typical exchange goes like this:

1. A manager sets the available shifts in Shiftboard.
2. A student driver sees that a shift is available and picks it up in Shiftboard.
3. Shiftboard sends a notification via email to the management team.
4. A manager manually enters the shift and driver information into the spreadsheet, and assigns an available bus – one that is not currently undergoing maintenance.
5. The driver clocks in using the BusGenius tablet in the bus.
6. BusGenius gathers all of the clock-in/clock-out information into a graph.
7. A manager interprets the graph to ensure employees are working at the correct time.

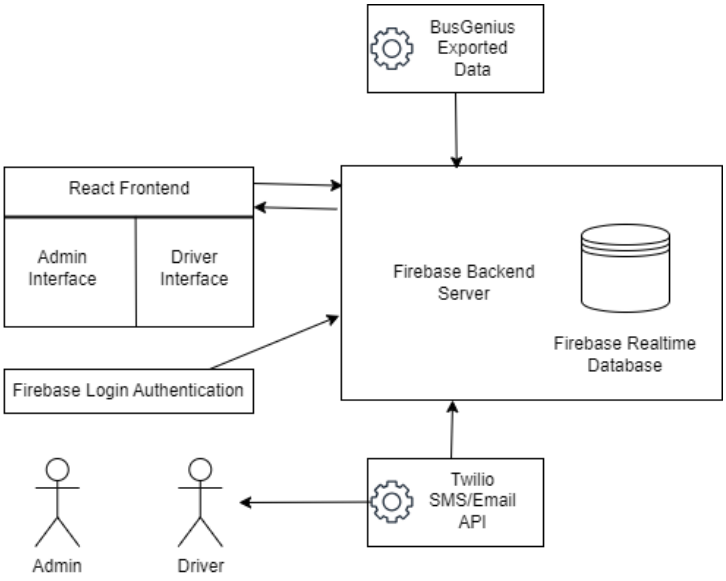
On a typical day, the first part of this exchange will occur multiple times, requiring that the spreadsheet be manually updated often. This current scheduling process causes disruptions to workflow that could be prevented by the development of a more integrated, automated system.

# Solution Vision

The new scheduling system will be an interactive web application that will replace Shiftboard and the spreadsheet, while integrating BusGenius. The system will support logins for both driver and manager/administrator users, and will have the following functionalities:

- Provide a clean Gantt-chart style display of the bus schedule, showing available shifts.
- Automatically carry over the set driving schedule for full-time employees every week.
- Allow student and part-time drivers to pick up shifts.
- Automatically assign an available bus to a recently picked up shift.
- Allow administrators to edit the available shifts and manually assign drivers and available buses when necessary.
- Allow administrators to change the status of a bus or other asset as necessary.
- Take the clock-in/clock-out data from BusGenius and provide management with valuable statistics about employee work habits.
- Allow administrators to send email and text messages to employees through the system.

These functionalities will be discussed in more detail in the Project Requirements section. The system will have an interactive frontend built with React, and a backend hosted through Firebase. Firebase also provides a real time database for securely storing user data, and an authentication system allowing us to have both driver and administrator logins. Two API's will also be used – Twilio, for email and SMS messaging, and BusGenius for comparing the schedule to real clock-in/clock-out data. The diagram below shows how all these different parts will be integrated and work together.




In the next section, we will discuss how these technologies will satisfy all the needs of both the administrators and drivers at NAU Shuttle Services.

## Project Requirements

We plan to implement and smoothly integrate all of the aforementioned technologies in order to fulfill the following client requirements, and create a new scheduling system that will positively affect the daily operations of NAU Shuttle Services and will allow the department to continue to serve NAU students and faculty and provide smoothly running and reliable transit.

**Reactive Desktop and Mobile Web App Frontend for Users and Administrators**

The application will provide a reactive frontend web application accessible for both mobile and desktop for both users and administrators. This frontend will be designed in Figma and constructed using React. The frontend will be composed of a user login interface, a real time schedule viewing interface, a user schedule management interface, an administrator schedule

management interface, an administrator user management interface, a clock in report interface and a user settings interface. The user login interface will be composed of a username/password input as well as a forgot password button for password reset. From this interface, the user will be brought either to the schedule viewing interface with either user or administrator management options depending on the user type. The schedule viewing interface will display a gantt chart depicting each bus with the current user assigned to the bus. Users will be able to click on their own shifts and be brought to the management interface to post the shift for pickup and click on available shifts to be brought to the management interface to pick up the shift. Administrators will be able to click on any shift and be brought to the administrator shift management interface. This interface will allow administrators to reassign shifts to other drivers or make shifts available for pickup as well as manage which buses are in service for the day. This interface will also allow administrators to manage automatic schedule creation to assign certain drivers the same shifts each week. The administrator user management interface will allow administrators to create/delete user accounts for new or past employees, manage the buses users are able to pick up shifts for and send messages to specific users or announcements to multiple users at once. The clock in report interface will be available for administrators displaying scheduled times and clock in times in a gantt chart with late clock ins highlighted. The user settings interface will allow users to change their email, phone number and password.

**User/Administrator Accounts**

For user login we will utilize the Firebase Auth library for managing account information. We will have three types of accounts, Drivers, Administrators and the Owner. Drivers will have the base level of access to view their schedule and pickup shifts they have

permission for as well as manage their account information. Administrators will be able to make manual overrides of driver schedules, change the status of buses for maintenance, send driver messages and announcements and manage user account information including account creation and deletion. The Owner will have all the privileges of Administrator but will be able to manage Administrator accounts. For account creation we will utilize the Auth library function createUserWithEmailAndPassword that will be accessed through the administrator panel and will generate an account with a random secure password for the administrator to provide to the driver. User login will be handled with ui.start('#firebaseui-auth-container') which creates a UI for account information entry. Drivers will then reset their password upon initial login which will utilize the sendPasswordResetEmail function. This function will also be utilized for the "Forgot Password" feature. Firebase allows rule based access control through specification of Realtime Database rules which will allow elevated privileges for Administrators and The Owner. All requests to the web application will include an authentication token which will automatically be managed by the Authentication SDK. These tokens will verify access and specify the privileges of the user to only allow intended actions.

**Automatic Schedule Creation with Manual Overriding**

The application will implement an automatic scheduling system that generates weekly schedules based on predefined driver availability and recurring shift patterns. The system will utilize Firebase Realtime Database to store schedule templates and driver preferences. Administrators can create schedule templates that specify regular shifts for drivers, which will automatically populate each week. The scheduling algorithm will consider factors such as driver qualifications for specific bus routes, required rest periods between shifts, and maximum weekly

hours. Manual override functionality will allow administrators to modify automatically generated schedules through the administrator interface, with changes being immediately reflected in the Realtime Database and triggering notifications to affected drivers. All schedule modifications will be logged with timestamps and the administrator who made the change for audit purposes.

**User Messaging via SMS/Email**

The messaging system will utilize Twilio's SMS and email services integrated with Firebase Cloud Functions for automated and manual communications. The system will have user options to automatically send notifications for schedule changes, shift availability, and clock-in reminders. Administrators can send individual messages or group announcements through the administrator interface. Messages will be stored in Firebase Realtime Database and will trigger Cloud Functions to dispatch communications via Twilio's API. The system will implement message templates for common notifications and support both SMS and email delivery based on user preferences. Message delivery status will be tracked and logged in the database, with failed delivery attempts automatically retried according to a configured retry policy.

**Automatic Clock In Report via BusGenius API**

The clock-in reporting system will integrate with the BusGenius API to automatically track and record driver clock-in times. The system will compare scheduled start times with actual clock-in times from BusGenius, storing this data in Firebase Realtime Database. A Cloud Function will run daily to fetch clock-in data from BusGenius and update the database. The administrator interface will display this information in a Gantt chart format, with color-coding to

highlight late clock-ins and missed shifts. The system will generate daily and weekly reports summarizing attendance patterns and will be exportable to PDF for administrative purposes.

## Potential Risks

There are a lot of potential risks associated with our project and if some of our systems are not developed properly it can have a real negative impact on our clients workflow and could potentially have a major impact on the universities flow of traffic as well. A major risk involved in our project is if the data is inaccurate or mismanaged. If the system incorrectly assigns drivers and buses, it could result in scheduling conflicts or employees missing their shifts. If this occurs it could lead to buses being delayed, students and staff missing their bus or connection and could also lead to a negative impact on how people feel about transit services. In the event that a driver is unavailable because our system was inaccurate or mismanaged it could leave entire routes unserved which would disrupt campus activities. A way we can mitigate this risk is to implement thorough testing for our scheduling logic and computations. Another way to mitigate this risk is to possibly implement a manual override option for administrators so they could quickly resolve these issues as they arise.

Another potential risk that could arise is if we do not integrate our platform effectively with the Twilio API for SMS or email notifications. If the SMS or email notifications fail to deliver or deliver inaccurate information, drivers could miss critical updates about their shifts. Critical updates could include: shifts being revoked, added or modified or if managers need to reach out to their employees to find coverage. If these notifications fail to be delivered or if they are delivered, they are not accurate could lead to a lack of coverage for scheduled routes, causing

delays and frustration for passengers. In addition to this, it could create confusion within the staff if the message contained information such as a shift trade or removal and it was not communicated. A way we could possibly mitigate this is to include delivery status checks for notifications or as a last resort, provide a backup communication channel to make sure that there is always a flow of communication.

There is also a risk of a potential security breach, even if it might be a rare occurrence. Unauthorized access to our systems could expose sensitive driver information such as schedules, personal data, or could potentially allow malicious actors to tamper with schedules. This could impact and disrupt transit operations and could also lead to privacy concerns since sensitive user information could have been compromised. A way we can mitigate this and take steps to prevent it from happening would be to use Firebase Authentication which uses strong encryption protocols. We could also potentially require administrators to use multi-faceted authentication. In addition to this we could also regularly audit the system for vulnerabilities.

Another risk that our system could face which a lot of other live systems face is the risk of the system going down. This could have a major impact on the workflow of our client and potentially the university as a whole. If the system becomes unavailable due to server issues or maintenance, admins and drivers would be unable to access or update schedules or contact drivers. If this happens it would force a return to manual scheduling which would defeat the whole purpose of the application. It would also create confusion and inefficiencies during the system's downtime. A way we could mitigate this is to use Firebase built-in high availability features and also implement a clear fall back plan such as exporting schedules for offline use.

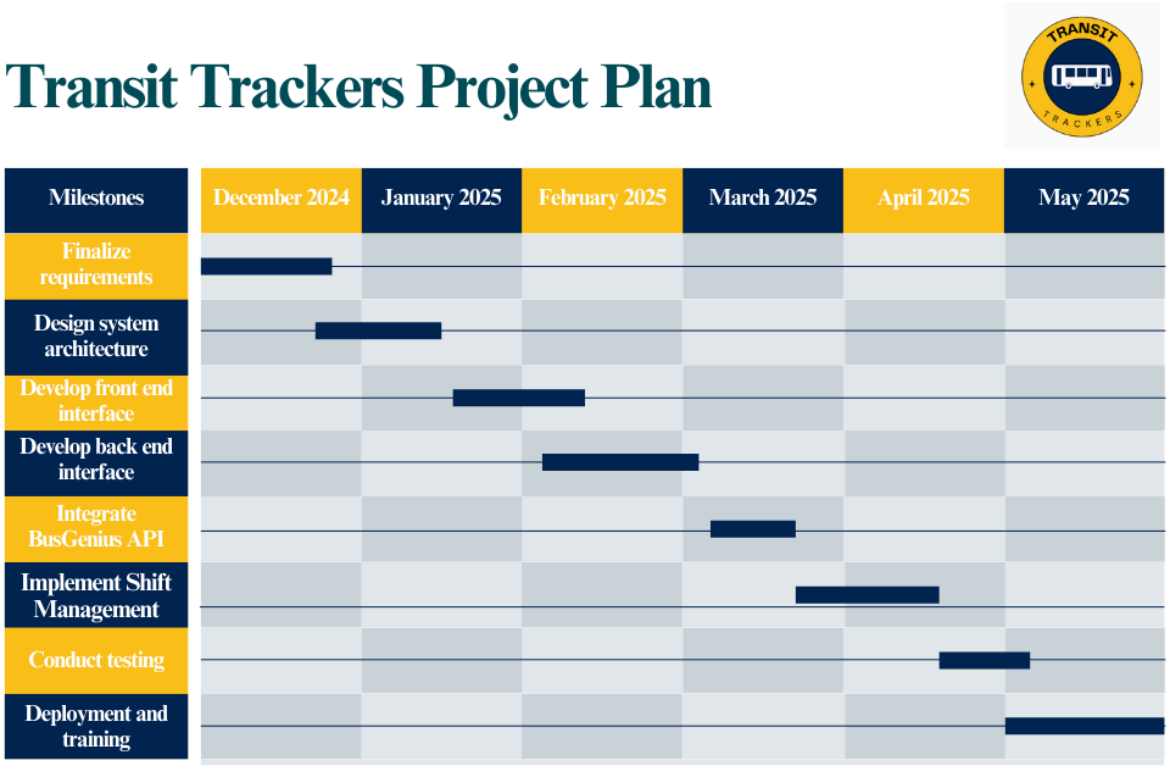
# Project Plan

The project plan we have developed is a straightforward approach focused on reaching key milestones in the development of our project. Each of these milestones would be centered around some sort of essential functionality in our application, as to ensure that we are able to build our application in an iterative and progressive manner. This plan would also streamline the workload by breaking down the project into smaller manageable phases. By structuring our project plan in this manner, we would enable the team to focus and prioritize the development and testing of each phase effectively. Breaking the project into phases (centered around key milestones, which itself is centered around a key functionality to our project) would allow us to identify any challenges we encounter in the phases early on, making it much easier to implement changes before we iterate on the phase into a more advanced phase. This approach would also enable us to be flexible towards any client feedback that might arise at any point during the development of our project. An incremental/iterative approach can help partition up the work between our team, making sure all the team members are able to evenly contribute towards the project. The following major milestones outline our basic project plan for the remainder of the project:

1) Finalize requirements for the project
2) Design system architecture and its components, through diagrams and mockups that simulate core functionalities
3) Development of front end interfaces, primarily ensuring scheduling functionality is visible on the front end
4) Development of back end interfaces
5) Integration of BusGenius API into our existing components

6) Implementation of shift management functionality
7) Conduct white and black box testing
8) Deployment of application to employees and training employees on how to utilize the service

This Gantt chart below describes what the timeframe for each of these milestones will look like:




By following this project plan, we are able to position ourselves to efficiently manage our workload, and divide up the work into equal and manageable parts for all of our team members.

# Conclusion

In summary, our project aims to address a critical need for a streamlined and automated scheduling system for our NAU Shuttle Services team. By creating a scheduling application that all bus personnel can utilize, we can assist NAU Shuttle Services and the entire university transit system in operating efficiently and providing reliable transportation to all who need it. Our system will vastly improve the scheduling process for the management team, while still offering a simple and familiar user interface. By fulfilling all of the requirements that have been discussed, and ensuring the system meets the needs of both the management team and their drivers, the system will prevent interruptions to workflow and save hours of manual work every week. It will also assist drivers in being able to know their schedule further in advance and ease the transition between vehicles and routes. This system will not only positively affect all who are employed by NAU Shuttle Services, but every student and staff member at Northern Arizona University who relies on the shuttles to get where they need to go every single day.