Revised List of Features in order of priority:

1. Interactive Map
   - Users will be able to interact with a map that shows their immediate surroundings as well as bus lines around their area. After selecting their route, the map will also show them their path and directions to get to their destination.
2. User Profiles
   - Implementing user profiles will allow users to customize their experience with TransitBlu. Users should be able to enter in their schedules and walking speeds. We will also use user profiles to collect data.
3. Highlighted Routes
   - Users will be able to see their routes highlighted in their map.
4. Walking Distance
   - Users will be able to find routes between specified locations, including departure location and time, destination, estimated walking distance and trip time.
5. Bus Time
   - Users will be able to find bus routes that take them from their area to a destination along with a trip time.
6. Feedback
   - Users will be able to report bugs and suggest changes to the software. Bussing agencies will also be able to give us recommendations and find and report bugs.

Architecture Diagram
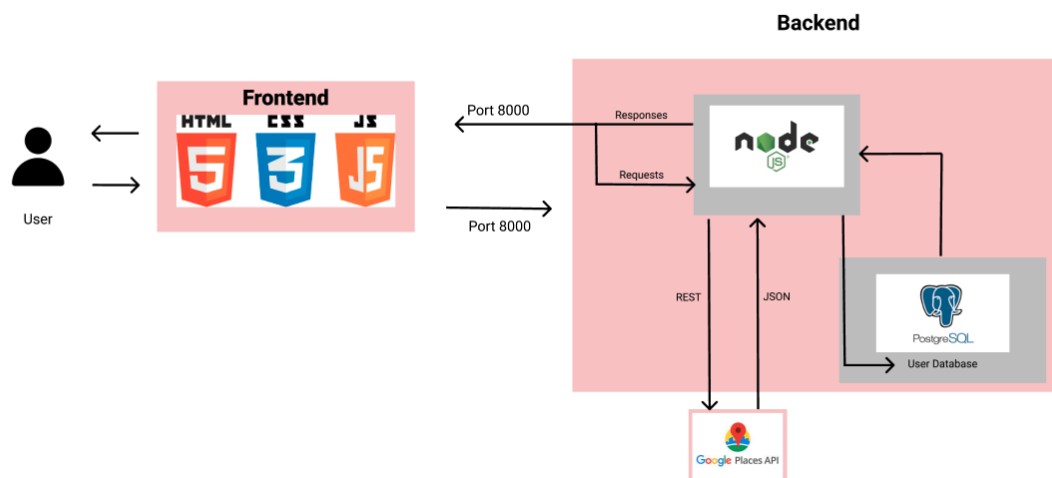
Desktop - 2

TransitBlu

Log in:

Username:

Password:

Submit

or Register

Desktop - 3

TransitBlu

Register

Enter a Username:

Enter a Password:

Submit

Desktop - 4

TransitBlu

Success!

Continue to the main app

Desktop - 1

TransitBlu   Favorite Routes   ⚙

Drop down Menu

View Favorite Routes   Log In

Add Current Route to   Report a
Favorite Routes   Problem
                                        with a Bus

Map   Sign Out

                                        Change
                                        Walking
                                        Speed

                                        Profile

Start From: (your location)

Go To: (CU Boulder)   Go

Desktop - 5

Head South

You are here

Get off at stop Broadway and Baseline

ETA: 30 minutes

Desktop - 9

Favorites:

Home to Campus   ⇄

Home to Work   ⇄

Desktop - 6

Select your Problem

Late Bus

Full Bus

Software Problem

Desktop - 7

Enter a Report Here

Go

Desktop - 8

Thank you for your feedback

Frame 2

Your Profile   Edit
                                Profile
                                Picture

Change Username

Change Password

Change Walking Speed

Favorite Routes

Frame 3

Change Username

Enter New Username

Confirm New Username

Frame 4

Change Password

Enter New Password

Confirm New Password

Frame 5

Change Walking Speed
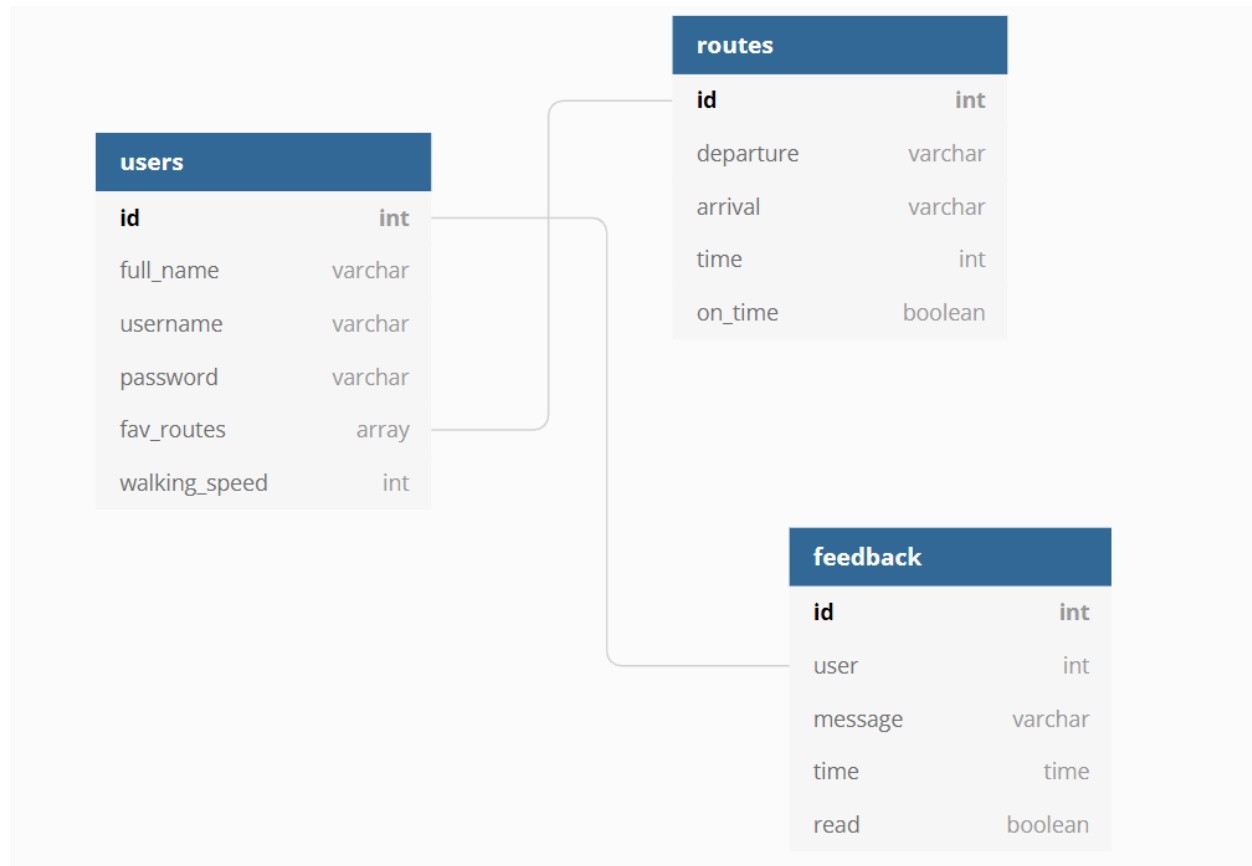
🟢 Default Speed

⚪ Custom Speed

Frame 6

Change Walking Speed

⚪ Default Speed

🟢 Custom Speed

Enter Custom Speed (Ex: 5 mph)

Web Service Design:

The main API that will be used is Google Maps Javascript API. The url call only requires a key to function. All other information can be pulled using function calls from the Javascript API library.

As an example of how it works, to display the map centered on the user's location, the application uses the navigator function getCurrentPosition to get the longitude and latitude of the user. Which are then used by the setCenter Javascript API function to change the map center. Everytime one of the Javascript API functions are used, an API call is made and returns every information that this API provides. However, the functions parse the data and automatically display the relevant information through the Map object.

For the map and routing to work properly the user only needs to input a starting address, or allow the browser to know their location, and a final destination. With these two pieces of information the API can calculate and display the route and center it on the webpage.

Challenges:

1. Not being able to implement all of our planned functions with the Google Maps API:
   a. We had many initial plans on different functions to create with the MAPS API initially, mainly dealing with the routes in the nearby area and coloring the map. If we are not able to do this due to lack of time, we will most likely just stick to the most essential functions, such as mapping a route and highlighting it on the map.
2. Not being able to convert our app to a mobile version:
   a. We were originally planning on making the app a mobile one by making it a PWA. Bus apps would most likely be used from a phone, as it would allow you to see where you are moving. However, if we are not able to figure out how to turn the app into a PWA, we will most likely just make the app a browser version on a desktop to have a proof of concept.
3. Inaccuracy of location detection:
   a. Our current location detection is a bit inaccurate (can be off by a mile), and sometimes, if given an address, will give the complete incorrect location (in a completely different state). The location detection issue is not huge, so if we can not find a solution on the API, we can most likely just leave it. However, for the incorrect detection of addresses, if it continues and we are not able to find a solution, we will likely have to just use absolute addresses.

Individual Contributions

Owen: Design, implementation of database and one endpoint to add a user.
https://github.com/CU-CSCI-3308-Fall-2021/CSCI-3308-Fall21-017-03/tree/main/Project%20Code%20Compoents/Backend
Rafael: Feature List revision and code optimization.
https://github.com/CU-CSCI-3308-Fall-2021/CSCI-3308-Fall21-017-03/tree/main/Project%20Code%20Compoents/website

Kyle Ma:
    Added challenges to milestone document,
    Cleaned up the Front End of the application:
https://github.com/CU-CSCI-3308-Fall-2021/CSCI-3308-Fall21-017-03/commit/55df8ec9210777d005369a41296cc161fc36a482
Added problem report page

Jonathan Goins: worked on the architecture diagram as well as some of the front end design
https://github.com/CU-CSCI-3308-Fall-2021/CSCI-3308-Fall21-017-03/commits?author=goinsj99