08 T2 Accessible Interactive Map Software Test Plan and Report

CS 4850, Section 03

Spring Semester 2025

3/30/2025

Sharon Perry

Justin Connick

Derrick Novack

Megan Ingram

Emily Zhu

Spencer Williams

Table of Contents

Software Test Plan

Introduction

Our project is an accessible map for KSU students that uses React, Flask, Google Maps API, and Geolocation API. As a web application, it's available on major browsers on devices such as smartphones and desktops. Upon receiving the desired features from the project document, we started analyzing the requirements and used the MoSCoW Model to refine our scope. We also made user stories to determine our target audience which helped decide the application's primary features. After finalizing the requirements, we created a mockup in Penpot that focused on a monochromatic color palette and easily identifiable elements for the visually impaired. As we continue the development of this application, we are now working on testing the functional requirements.

Tester Selection

We will test the project by loading the project on our IDEs and passing it to peers. We were selected from class members, family members, and people we know with various disabilities. We will instruct them to look for defined test cases. As they move through the application, we will check whether the requirements passed or failed. We will also note the severity of the failure: low, medium, or high. The peers will be diverse in technical backgrounds, as some will, and some won't have programming knowledge or experience. In addition to this, we will ask peers that have special accessibility needs to take part in testing. We are testing the functional requirements, which are essentially the working features that make up the application. It will be tested on our local desktops using our IDEs since the project isn't available on browser yet due to it not being hosted on the server.

Software Test Report

Test Case Results

Requirement #	Description of Requirements	Pass/Fail	Severity of failure
1	Live tracking – The user changes their location from one building to the next. The indicator should be updated to reflect the change in location in near real-time.	Fail	Medium
2	Route creation – The user will set their location and set a destination and request a route. After their request a route should be created depending on the accessibility options.	Pass	High
3	Search bar with autofill – Users will start typing the name of several buildings. The search bar should automatically bring up names that have what the user is typing.	Pass	Low
4	Elevation tracking – Once the user creates a route, we are looking for	Pass	High

	changes in color along the route that accurately depicts the changes in elevation during the route.		
5	Accessible doors icon/route – After the creation of the route the user will click on the accessible door option. There should be icons that represent accessible doors for the destination building.	Pass	Medium/High
6	Database connection – We will test the connection between app and database with things like account information.	Pass	High
7	Building-to-building routes – Users will go to a building on campus and create a route to a different building.	Fail	Medium
8	Route leg instructions – Once a route is created under the map should show each part or “leg” of the route showing instructions.	Pass	High

9	Zoom functions – When looking at the map the user should be able to press the zoom in and out buttons to zoom in and out, respectfully, on the map or pressing “Ctrl” and scrolling in and out.	Pass	Low
10	User account registration – Users should be able to click on the top right section and create an account and have their information stored.	Fail	Low
11	User account login – Once a user creates an account, they will try logging in.	Fail	Low
12	Anonymous access – Full functionality remains even if the user isn’t signed in.	Pass	Medium
13	Account page – The user will be able to see their account and but able information regarding it.	Fail	Low
14	ETA approximation – Once the user creates a route, there should be an	Pass	Medium

	estimation of how long it will take the user to arrive at their destination.		
15	User location – The app should be able to accurately obtain the users location.	Pass	High
16	Side menu with campus map links – Users will click on the side menu to ensure that it is working and the links lead to the correct campus maps.	Pass	Low
17	Map display – The map should display as the same size and in the same place on the website.	Pass	High
18	Identifiable buttons – Users should accurately be able to determine which button does what.	Fail	High
19	Mobile and desktop friendly – Our UI should look good for both desktops and mobile devices.	Fail	High