EZ Waiver

Project Outline

Speedy Waivers v 1.0

04/28/2020

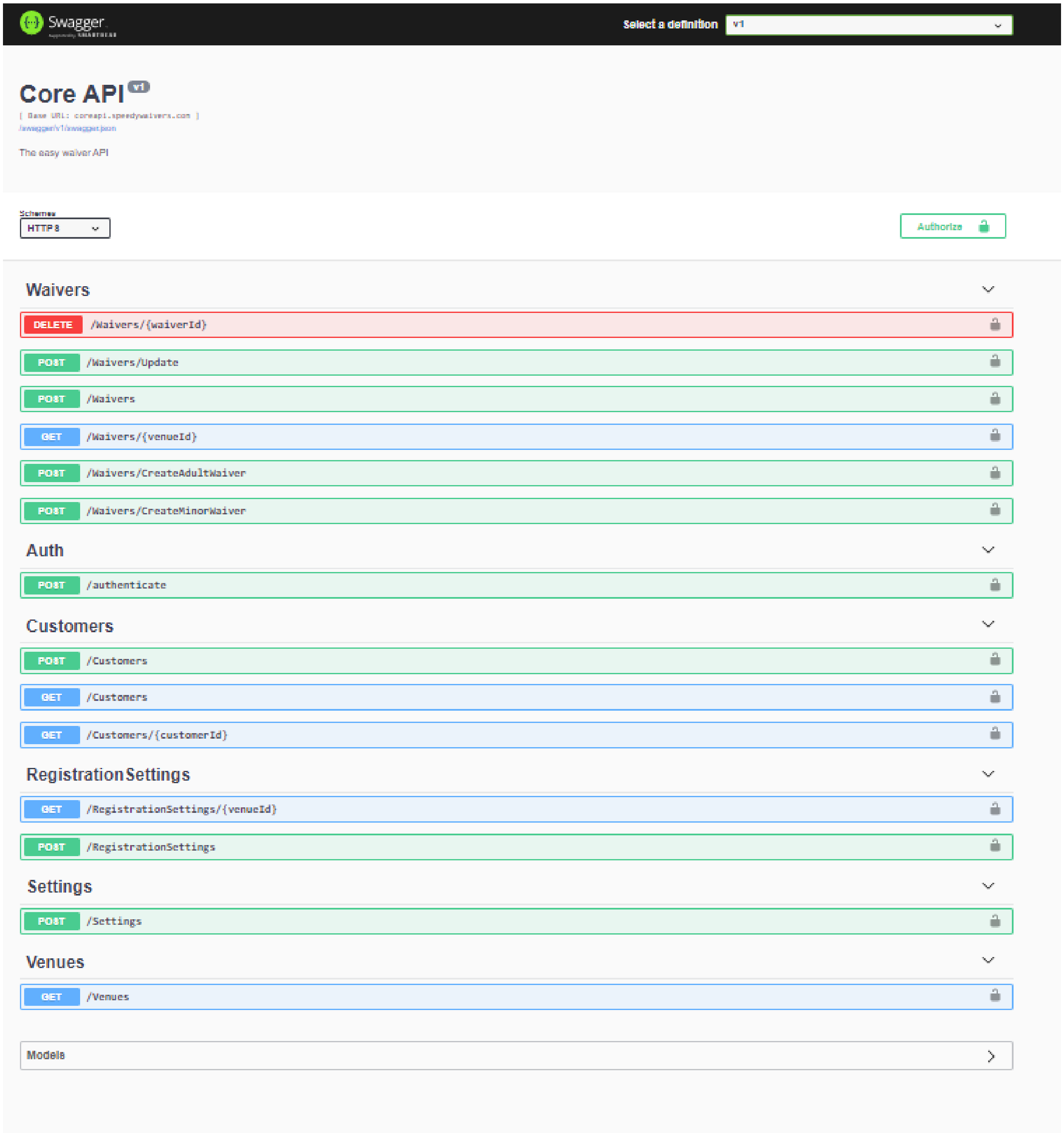
Project Outline
Speedy Waivers v. 1.0
04/28/2020

Published by:
Speedy Waivers, LLC.
123 41st B
Newport Beach, CA 92663
Telephone (404)-408-3745
Email JustinShawnGerber@gmail.com

Table of Contents

API

The API has already been built. The documentation for the API can be found at the following address: https://coreapi.speedywaivers.com/swagger/index.html. It uses JWT for authentication.



Core API v1

[Base URL: coreapi.speedywaivers.com]
/swagger/v1/swagger.json

The easy waiver API

Waivers

- DELETE /Waivers/{waiverId}
- POST /Waivers/Update
- POST /Waivers
- GET /Waivers/{venueId}
- POST /Waivers/CreateAdultWaiver
- POST /Waivers/CreateMinorWaiver

Auth

- POST /authenticate

Customers

- POST /Customers
- GET /Customers
- GET /Customers/{customerId}

Registration Settings

- GET /RegistrationSettings/{venueId}
- POST /RegistrationSettings

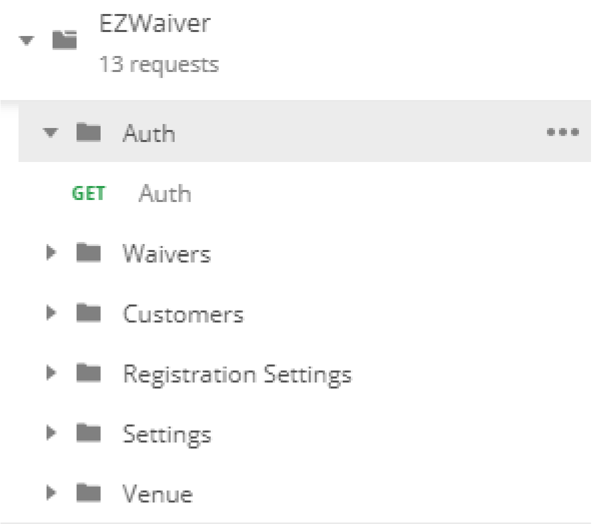
Settings

- POST /Settings

Venues

- GET /Venues

Postman Collection



EZWaiver
13 requests

Auth •••

GET Auth

Waivers

Customers

Registration Settings

Settings

Venue

The postman collection can be downloaded at the following link:
https://drive.google.com/file/d/10aRnOhBLsLOBUKx_15mKrUqaRfPQec-8/view?usp=sharing

More postman documentation can be found here:
https://documenter.getpostman.com/view/2691658/Szf24VKj?version=latest

Github

We use git and Github for source control. Our Github repository can be found here:
https://github.com/SpeedyWaivers

Please let me know your Github username and I will send you an invite to the organization. Please submit your code to the following repository:
https://github.com/SpeedyWaivers/App

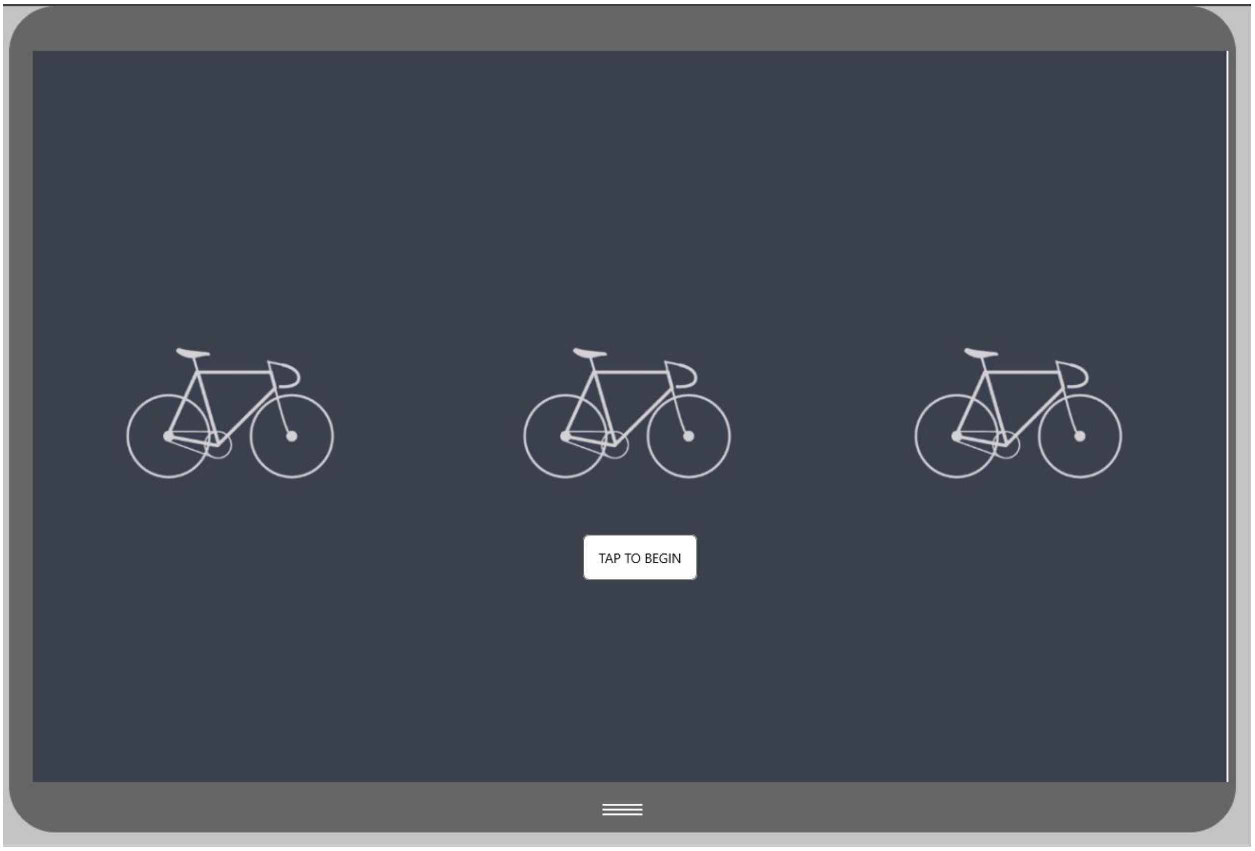
App Details

App Name: Speedy Waivers

This will be an iPad kiosk. The app will be used to create waivers for customers at venues. The application will capture basic information like first name, last name, date of birth, etc. Finally, it will display the terms and conditions and require the user to sign. This information should be submitted to the API, and then the app should return to the first screen for this process to occur again. I would like this application to work with iPads and iPad minis.

Startup Screen

When the application first loads, it should check the local storage and see if a token exists and is not expired. If the token does not exist or is expired, then it should navigate to the settings screen, seen in the next section. Otherwise, it should load the background image, seen in the settings screen. When the "tap to begin" button is clicked, it should navigate to the registration screen. Also, a secret gesture on this screen should navigate to the settings screen.



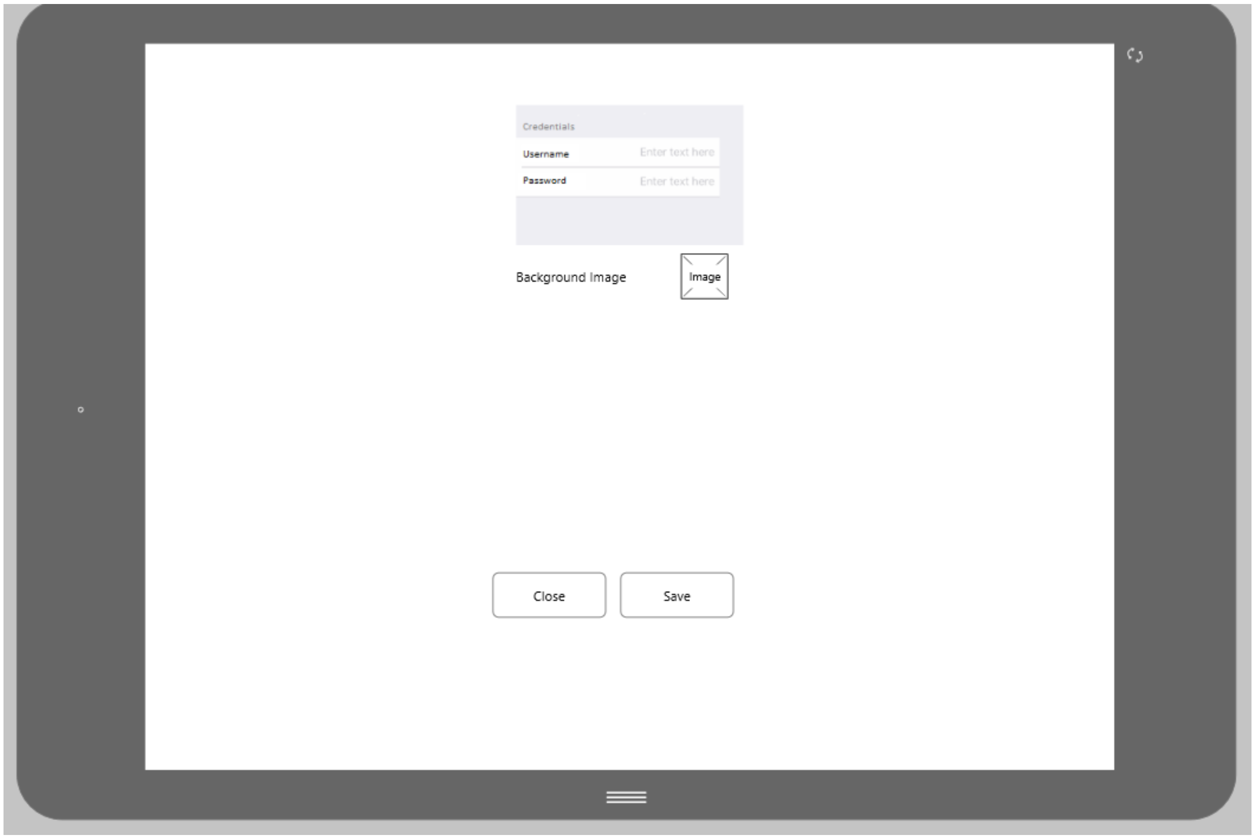
Settings Screen

When the save button is clicked, then it should make a request to the API at POST /Authenticate. The body of the request should contain the username and password from the textboxes. The response should look like this:
https://gist.github.com/Justbeingjustin/4ac84ca14cfda15531a724aadb8eb86c
Take the token from the response and save it in local storage. The token will be used in future requests to the API. It is the bearer token. Again, the API uses JWT Authentication. Valid credentials are

Username: test

Password: test

The background image should prompt the user to select from their photo library. This image should be stored in local storage. When the app is started it should display the image in the background. Also, once an image is selected, the user should be able to select a new image too.



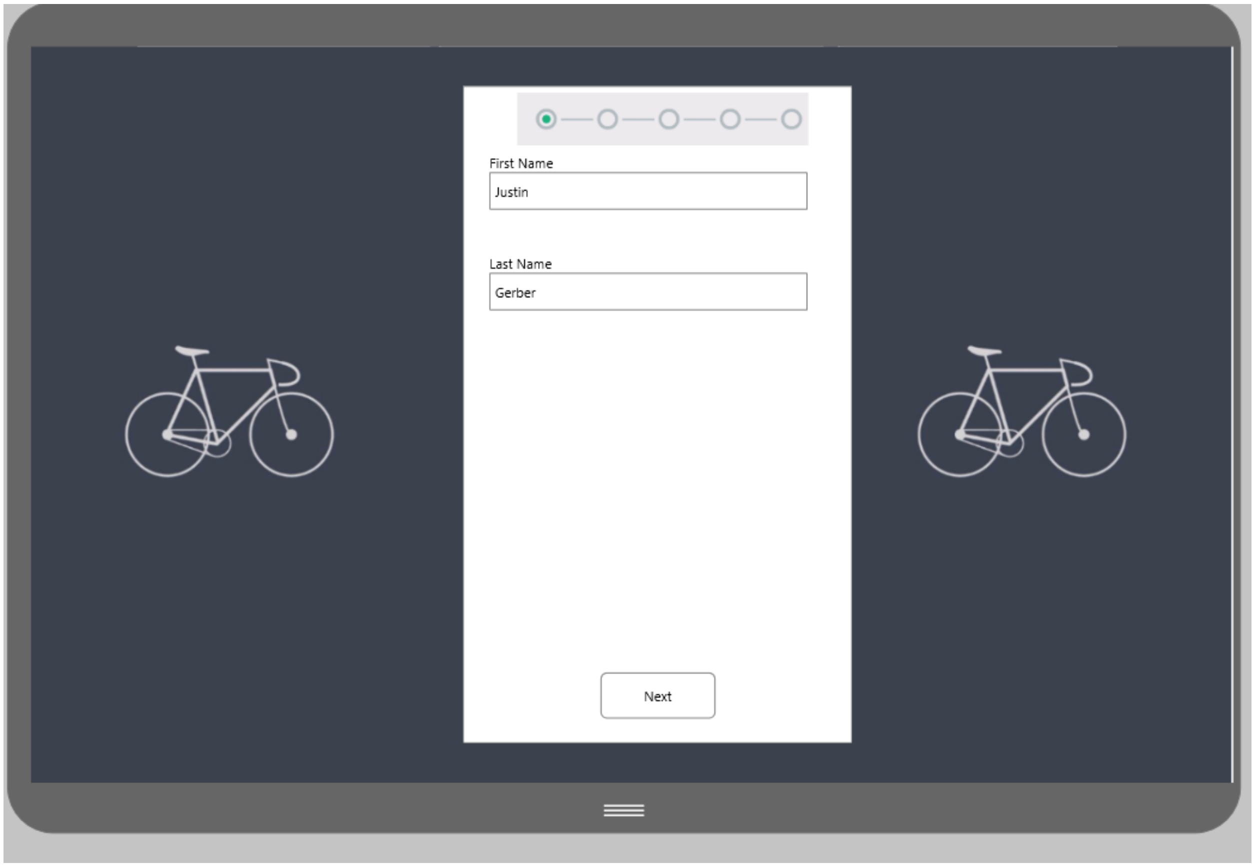
Registration Screen

Next, we need to get the registration fields. Two calls are required to get the registration fields. The first call should be made to GET /Venues. This will return an array of venues, seen here: https://gist.github.com/Justbeingjustin/066abe39fde839cf30364085b55c3366

Take the first item in the array and collect the venueId. In this example the venueId is 1. Next, make a GET request to /RegistrationSettings/1. This will return the following: https://gist.github.com/Justbeingjustin/b8b9593026aa0924d5d922d4d52a70c2

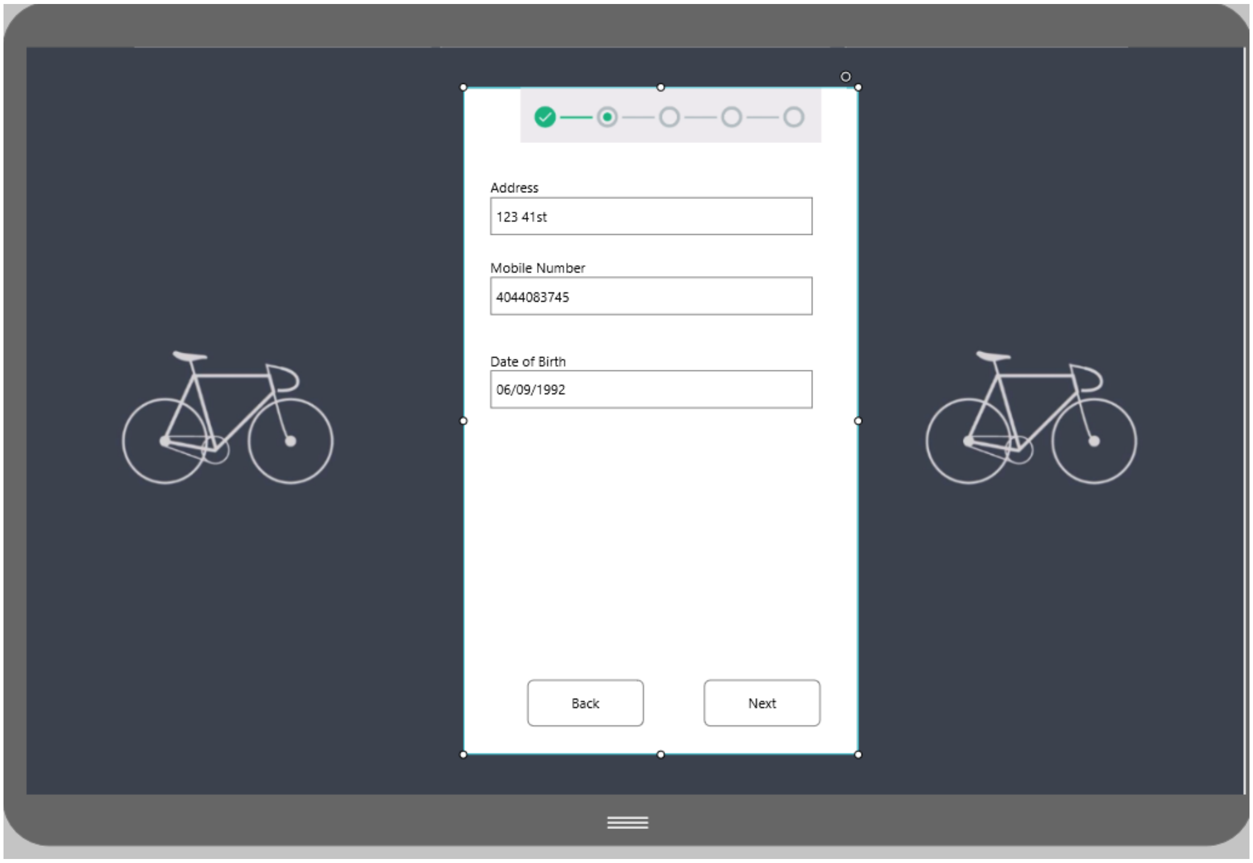
Important Note: The registration fields must NOT be hardcoded on the UI. They will need to be created dynamically because they can be configured on the website. For instance, the MobileNumber setting may not always appear in the array. Pay attention to the *settings.NumberOfFieldsPerScreen*. This is the number of fields that should be on each step of the wizard.

If the registration field has a name of “BirthDate”, then there should be a date picker. All other fields should be a textbox. If the setting.required is set to false, then it is okay to leave that field blank. If setting.required is set to true, then it should not allow the user to continue until it is filled out.



First Name: Justin

Last Name: Gerber



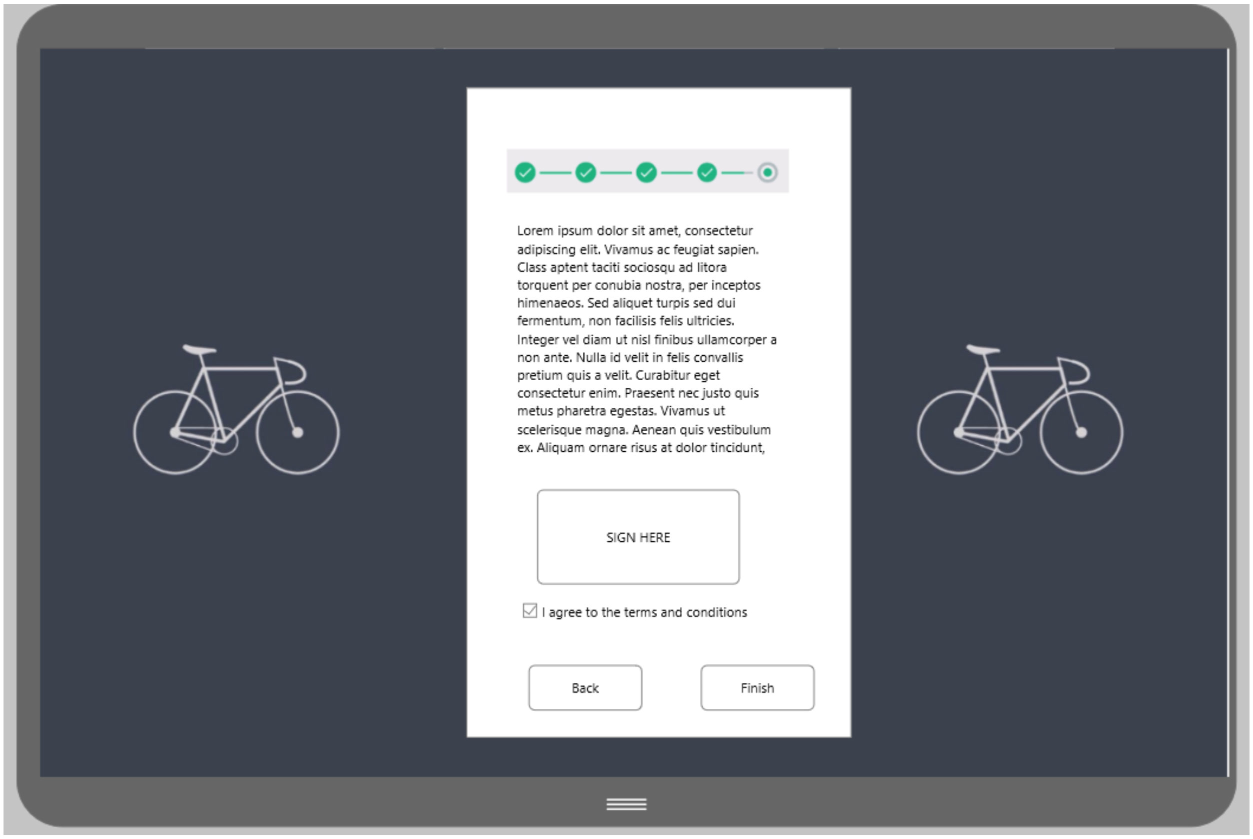
Address
123 41st

Mobile Number
4044083745

Date of Birth
06/09/1992

Signature Screen

From the previous section, a GET request to /RegistrationSettings/1 should return the following:https://gist.github.com/Justbeingjustin/b8b9593026aa0924d5d922d4d52a70c2 The waiver text should be displayed. Designate an area for a signature. Ensure the I agree checkbox is checked before submitting. Also, ensure that a signature is present before submitting. To submit the customer information, send the info to POST /Customers. Display a screen that it was successful, and then automatically navigate back to the start up page after a few seconds.



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus ac feugiat sapien. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Sed aliquet turpis sed dui fermentum, non facilisis felis ultricies. Integer vel diam ut nisl finibus ullamcorper a non ante. Nulla id velit in felis convallis pretium quis a velit. Curabitur eget consectetur enim. Praesent nec justo quis metus pharetra egestas. Vivamus ut scelerisque magna. Aenean quis vestibulum ex. Aliquam ornare risus at dolor tincidunt,