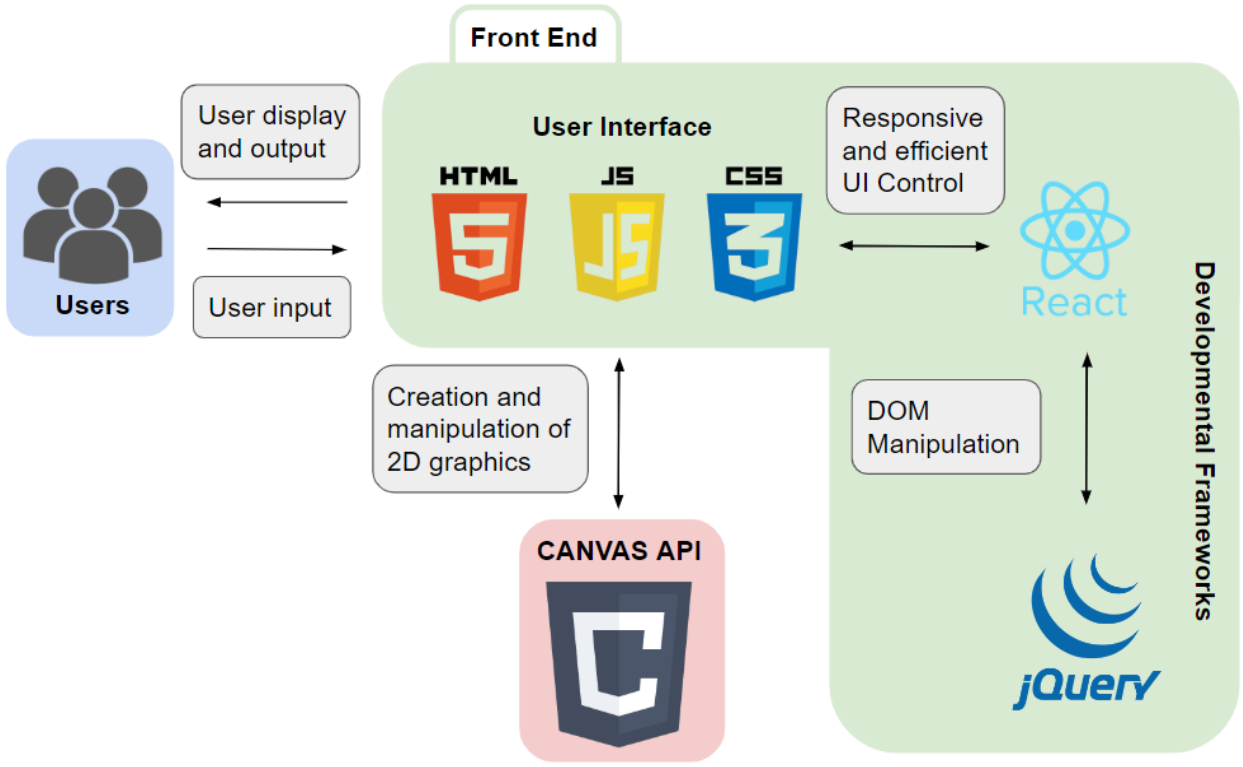
Design Document

Architecture Diagram



Code Repository

- Our GitHub repository with all our code can be found here.
- Our code is primarily written in HTML/CSS and JavaScript via React, which are used for the API and the frontend.
    - Github will say that a small amount of our project used python. This is because the motion models originally provided by the client were written that way.
- More info may be found in the README.

Detailed Data Definitions

- No external data is associated with this application.
- There are fields that do have limits within the motion models section of the application:
    - The differential motor motion model contains the fields left_radius, right_radius, distance_between_wheels, left_angular_velocity, and right_angular_velocity
    - The fields left_radius and right_radius can be any real number greater than 0 and up to 10
    - The field distance_between_wheels can be any real number greater than 0 and up to 10
    - The fields left_angular_velocity and right_angular_velocity can be any real number between 0 and 10

    - The bicycle motion model and the tricycle motion model have almost the exact same fields. Both motion models contain the fields degree, front_wheel_radius, distance_front_to_back, and angular_velocity
    - The field degree can be any real number. The number in the field is converted to radians in which any real number in degrees can be simplified into a number between 0 and 2.
    - The field front_wheel_radius can be any real number greater than 0 and up to 50
    - The field distance_front_to_back can be any real number greater than 10 and up to 100
    - The field angular_velocity can be any real number between 0 and 10
    - The only difference between bicycle and tricycle is the tricycle's distance_between_back_wheels, which is any real number greater than 30 and up to 200
- PID field limits
    - Setpoint: The Setpoint value should be within the valid range of the process being controlled. For example, if the process has a maximum value of 100, the setpoint should not exceed this limit. It is recommended to keep the setpoint within a reasonable range for the specific application to ensure smooth control and prevent excessive overshoot.
    - Kp: The proportional gain (Kp) should be a positive value. While there is no strict upper limit for Kp, extremely high values may lead to instability or oscillation. It is recommended to start with a low value and gradually increase it until an acceptable balance between response speed and overshoot is achieved.
    - Ki: The integral gain (Ki) should be a positive value. Like Kp, there is no strict upper limit for Ki. However, high values may cause overshoot or oscillation. Start with a low value and gradually increase it to minimize steady-state error without compromising system stability.
    - Kd: The derivative gain (Kd) should be a positive value. While there is no strict upper limit for Kd, high values may cause the system to respond more slowly. Adjust Kd to dampen the controller's response to rapid changes in error, providing stability without significantly sacrificing responsiveness.

Design Rationale

- Currently, the file structure is set up to where you have an initial folder called ROBOTEDU which contains the application code.
- As none of the current team members have any experience with React Hooks, we have React Class Components which share state via two-way data binding instead. This might be a longer way syntactically but it accomplishes the same purpose as React Functional Components with React Hooks.
- We have the application separated into “Motion Models”, “Pathfinding Algorithms”, and “PID” because we thought that it would be easier for the user to navigate instead of just having a list of “robotics things”.
- There is a Canvas overlaying a portion of the grid layout instead of one big canvas across the entire screen and some buttons on the bottom. This is just in case the client

needs to add more things to the application, and it simply looks better to the team and the client to do so.

- We have JQuery code that runs alongside the React portion of the code which is called to action by the React component's lifecycle methods. Again, this was not only the easiest way to inject code for DOM manipulation but also helped separate concerns better and make it easier for us to understand what was going on.
- We decided not to use other responsive UI frameworks such as Bulma because even though Bulma is more flexible with other devices, we really only plan to develop the application for a desktop.
- There are a few dependencies that need to be noted.
    - The first is snowpack, which helps build a React application. Therefore, when you run npm start, snowpack will start bundling and running the React application. This is just because the initial starting mechanism was deleted and this was an easy replacement.
    - Another two dependencies are the React Testing Library and Jest, both things we use to help test our application.
    - The final dependency is ReactDOM which helps render nodes within the test and helps render the initial node within the application itself.