Lab 5 Report and Documentation

This is a tutorial style writeup documenting the process to install and build the codebase as well as all the steps it took to get the environment working with Electron. In converting the codebase from Lab 3 (multithreading) into a native application using Electron, the official documentation for Electron (reference: *https://www.electronjs.org/docs/latest/*) was used extensively.

The first step was to initialize an npm package using the command *npm init*. When initializing, it was important to set up the entry point of the application to be *main.js*. Then, I had to install the electron package as a dependency, using the command *npm install –save-dev electron*. Finally, I had to set up the start script within package.json to be “electron .” in order to invoke an Electron execution upon *npm start*. A screenshot of the complete package.json is shown below.

Text

Description automatically generated

Next, I had to create a window and load the root html file inside main.js as shown in screenshot below. In the case of this codebase, the root html file was *‘./frontend/sort.html’*.

Graphical user interface, text, application

Description automatically generated

It should be noted that sort.html uses a multithreaded library created using Web Assembly and Emscripten. At this point, it was possible to spin up a native application using the command *npm start*.

A screenshot of a computer

Description automatically generated with medium confidence

With some minor changes to CSS using a stylesheet and employing bootstrap, the window shown in screenshot below was able to be spun up.

Graphical user interface, text, application

Description automatically generated

The base features of this native application remain very similar to the one in Lab 3. There are 3 sorting functions, insertion, bubble, and merge, that are executed upon button click. In this native application, they are available as nav bar elements. As an example, when *Insertion Sort* is clicked from the nav bar, this is shown as the result.

Text

Description automatically generated

1000 random integers are sorted using insertion sort and are printed out in order along with the timing information. A feature specific to this native application is its ability to toggle between light, dark, and system default themes. As default, a system theme (depending on whether the OS is using a light or dark theme) is set and is be shown in the nav bar.



The gray ‘Toggle Theme’ button allows users to toggle between light and dark themes as shown below in the screenshot.

Graphical user interface, text, application

Description automatically generated

A picture containing text

Description automatically generated

Dark theme is shown below.

Graphical user interface, text

Description automatically generated

Text

Description automatically generated with medium confidence

And as implied above, clicking the ‘Reset to System Theme’ button will revert the application appearance back to system default.