SX-Dashboard Documentation

MTechHub

August 23, 2019

Contents

1 Project Overview

1.1 Application Walk-through

1.1.1 Adding packages

Before the application can be run, it's dependencies must be installed. While in the project directory, run:
`npm install` or `yarn`
All the packages will be added to the folder /node_modules. A list of dependencies can be found under ./package.json.

1.1.2 Running the Application

While in the project directory, run:
`npm start` or `yarn start`
to begin running the application at http://localhost:3000.

To build the application, run:
`npm build` or `yarn build`

1.1.3 Using the Application

After running the application, visit http://localhost:3000 in your browser. You should be redirected to http://localhost:3000/main/dashboard, and your screen should look similar to figure 1.

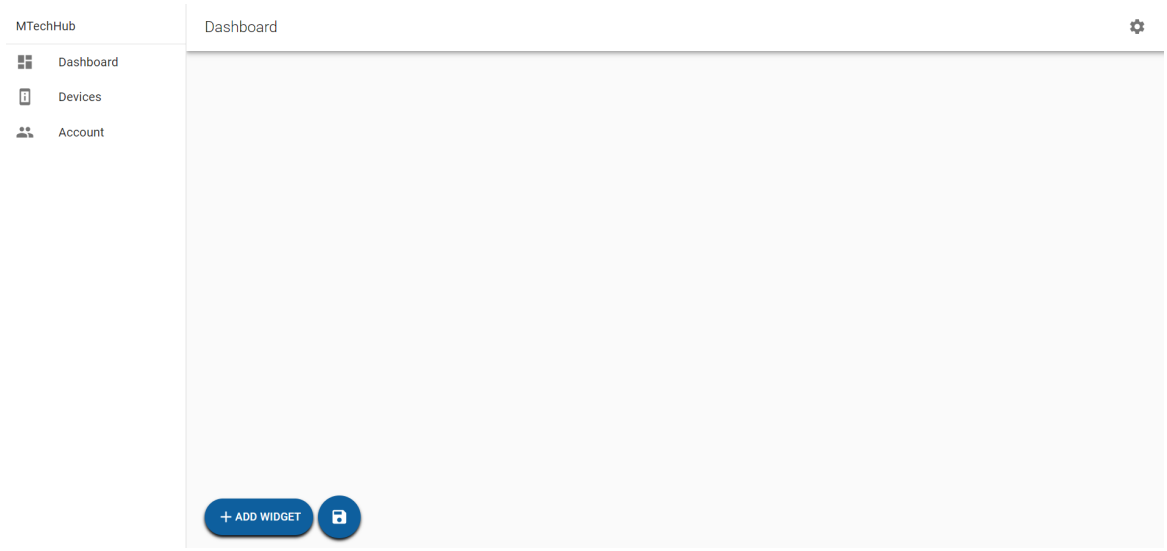

Figure 1: Starting screen

The Devices and Account tabs on the side are placeholders, and should be replaced with required tabs. Selecting Add Widget will bring up a menu with different chart types and a SQL/MQTT slider. The MQTT slider will setup a chart using the MQTT protocol to retrieve live data, whereas the SQL slider will setup a chart using a specified SQL database as the data source. Click the name of the chart you wish to create. A second menu will pop up asking for the relevant data, at the moment this is for show and the logic needs to be implemented. Fill in the required fields and select Create to create a chart.

Note that if creating a chart from SQL, at the moment it will give you a view of the last 10 records in that table, along with the column names so you can select the columns to create the chart from.

A chart should show up on the dashboard. This chart can be manipulated, click and drag to move it, drag the bottom-right corner to resize, and finally press the **X** in the top-right to delete the chart. Once you are finished with the layout of the dashboard, select the save icon to the right of Add Widget. This will save the layout to SQL, using the server API. At this point, closing the page and re-opening it should load up your last saved dashboard (assuming a successful SQL connection).

There is a cog icon located in the top right corner of the page. Currently this button does nothing, however it can be setup to contain a dropdown allowing theme selection (explained in __#REF HERE__) as well as any other options that may be required.

1.2 Brief Code Overview

Code entry point index.js. Loads layout found in main.jsx. Loads Sidebar. Sidebar loads main section based on path. Dashboard is the main section.

1.3 Code Walk-through

All of the files mentioned here are found in /src/. Each component will contain a section:

```
render() {
    return (

    );
}
```

Anything here is what will be rendered whenever this component gets called by another file. The entry point for this project is ./index.js, which contains a router that essentially redirects to ./layout/main.jsx. Main.jsx is the main layout for the project, it just renders the Sidebar component found in ./components/sidebar/sidebar.jsx.

The Sidebar Component

Sidebar is the menu component found on the left side of the dashboard. When the screen size is md and lower, the menu is hidden and opens via a button placed at the top left of the appbar. The actual items on the sidebar menu are loaded in from ./components/items/. list_items.jsx is rendered for the persistent sidebar, while mobile_list_items.jsx is rendered for the collapsible variant.

Each of the menu items has an associated path. This path is used in the Router found in the <main> section within sidebar.jsx. It essentially loads the content in the main window of the application, and allows that content to change depending on which menu item has been selected.

The Dashboard Component

The dashboard, located under ./views/dashboard/dashboard.jsx is the default rendered view. componentDidMount() is called as soon as the component is mounted, in here is where we asynchronously make a call to SQL to retrieve the last-saved dashboard layout. In the case that there is no found layout from SQL, we attempt to use the last layout stored in localstorage. Else, the dashboard will be a blank slate.

The createElement function will create a chart for the dashboard using the passed *el* object.

1.4 Project Structure

1.4.1 doc

The doc folder contains this pdf and the .tex file used to generate it. Pdflatex is required to regenerate this document locally, however there are compilers online you can use as well if you ever need to make changes. Assets contains the images used in the document.

1.4.2 /src/assets

1.4.2.1 css

This contains the style sheets used in the project.

1.4.2.2 img

Images used on the webpage will go here. (MTech Logo should be added and used on the menu)

1.4.3 /src/components

1.4.3.1 items

1.4.3.2 label

1.4.3.3 popup

1.4.3.4 settings

1.4.3.5 sidebar

1.4.3.6 widgets

1.4.4 /src/layout

1.4.4.1 main

1.4.5 /src/views

1.4.5.1 account

1.4.5.2 dashboard

1.4.5.3 devices

1.5 Libraries

2 Things to update