Project Folders and Files Structure:

```
webapp-task-bitfountain
├── public
└── src
    ├── apiEndpoints
    │   └── Index.tsx
    ├── containers
    │   ├── TheContent.tsx
    │   ├── TheHeader.tsx
    │   └── TheLayout.tsx
    ├── css
    │   ├── devices
    │   │   ├── DeviceModelAddForm.css
    │   │   └── DeviceModelData.css
    │   ├── loader
    │   │   └── Index.css
    │   ├── login
    │   │   └── Index.css
    │   └── Shared.css
    ├── hoc
    │   ├── formValidation
    │   │   └── LoginFormValidation.tsx
    │   ├── CheckUserAuthentication.tsx
    │   └── SetTitle.tsx
    ├── staticData
    │   ├── devices
    │   │   └── Index.tsx
    │   └── TitleData.tsx
    ├── views
    │   ├── devices
    │   │   ├── DeviceModelAddComponent.tsx
    │   │   ├── DeviceModelInformationCard.tsx
    │   │   ├── DevicesModelComponent.tsx
    │   │   ├── DevicesModelDataComponent.tsx
    │   │   └── SelectAsyncPaginate.tsx
    │   ├── loader
    │   │   └── LoadingComponent.tsx
    │   └── user
    │       └── Login.tsx
    ├── App.css
    ├── App.tsx
    ├── Index.css
    └── routes.tsx
```
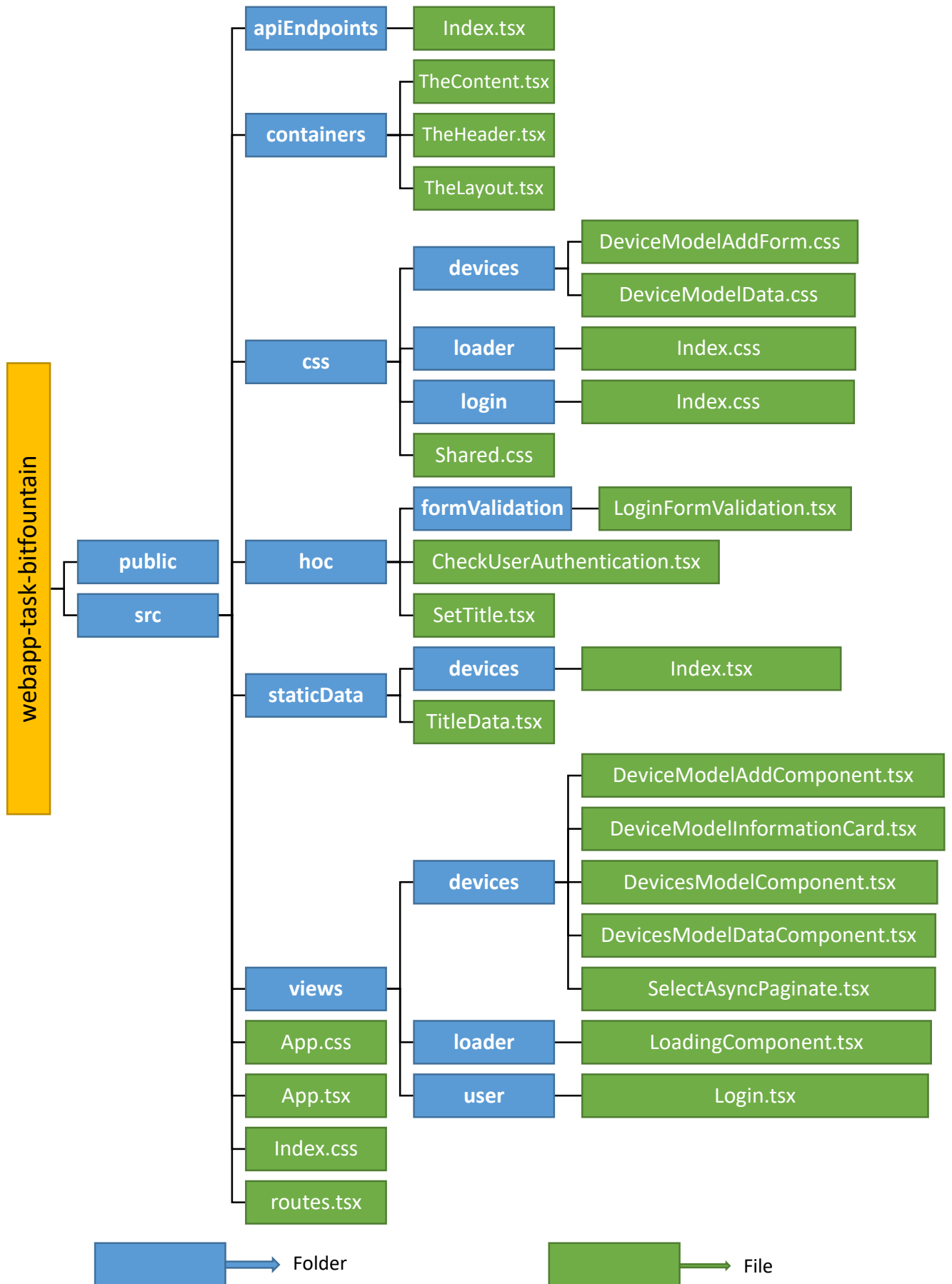
Folder

File

1

Used Packages:

1. "**react-router-dom**": "react-router" package provides the core routing functionality for React Router. Instead of installing it directly, I have installed "**react-router-dom**" and It installs "react-router" as a dependency. In my web app there are two types of routes, public routes and private routes and is declared in routes.tsx file.

2. "**axios**": "axios" is a library to make XMLHttpRequests from the browser. "axios" has wide browser support, client side support for protecting against XSRF, it automatically transforms to JSON data, it supports the promise API.

3. "**universal-cookie**": "universal-cookie" is the most popular package for managing cookies. [1]
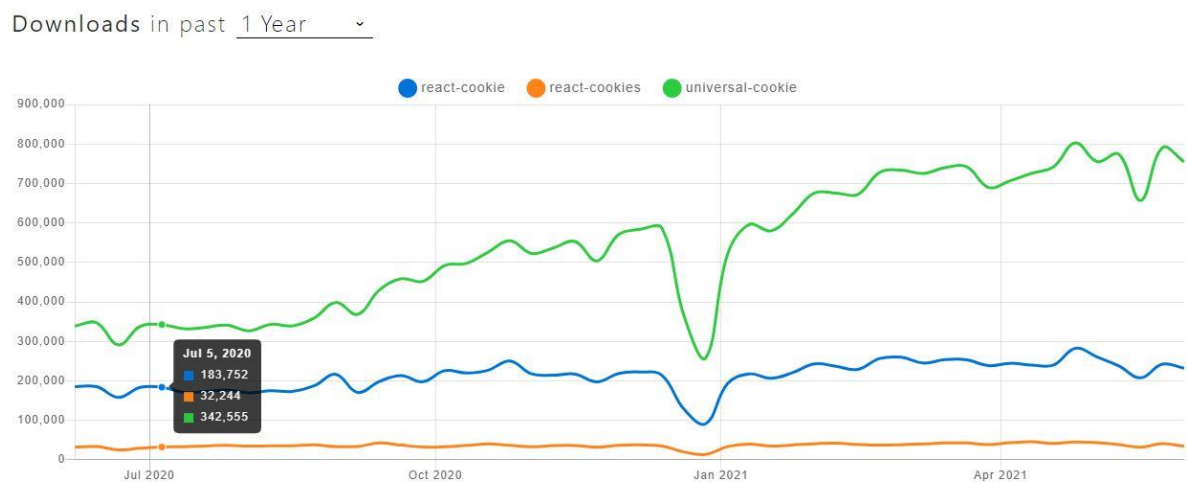
fig 1: Downloads statistics

And it's easy to use and well documented.

4. "**react-select-async-paginate**": Used to create TypeId select field, that supports pagination on options menu scroll.

5. **Material UI (**"**@material-ui/core**", "**@material-ui/icons**", "**@material-ui/lab**"**)**: Material-UI is a popular React UI framework and an open-source project that features React components which implement Google's Material Design.

UI Design and Functionality:

Login Page (Desktop View): Without logged in, user will be redirected to login page for any routes.
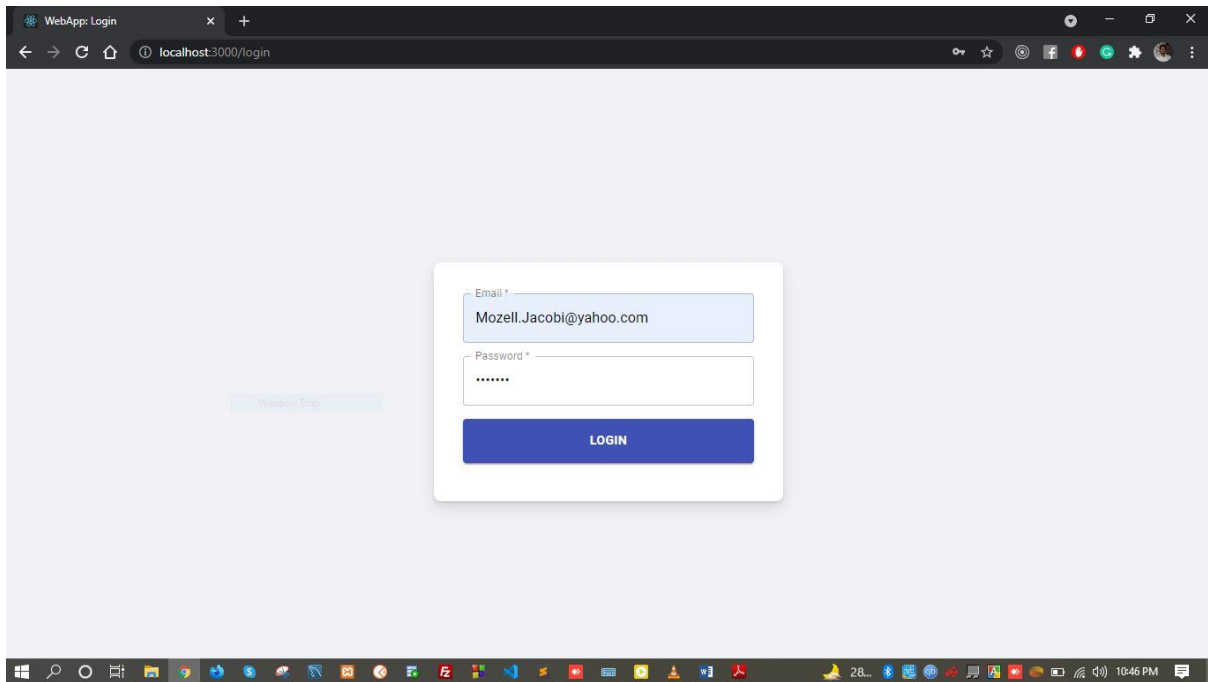

Fig2: Login page (Desktop View)

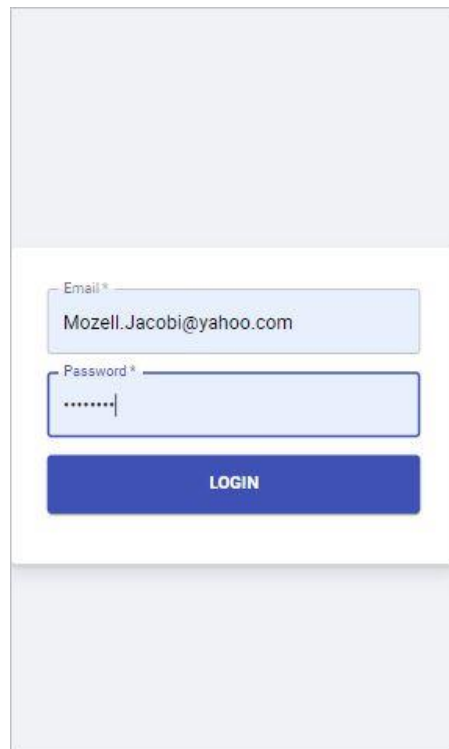**Login Page (Mobile View: Model Nexus 4 - Google Chrome emulated device)**

Fig3: Login Page (phone view)

Available Medical Devices Page (Desktop View): In this page user can select how many device model will be shown. And this page has a pagination. If user clicks on ADD NEW DEVICE MODEL button then he will see a modal dialog with form to create New Device Modal.
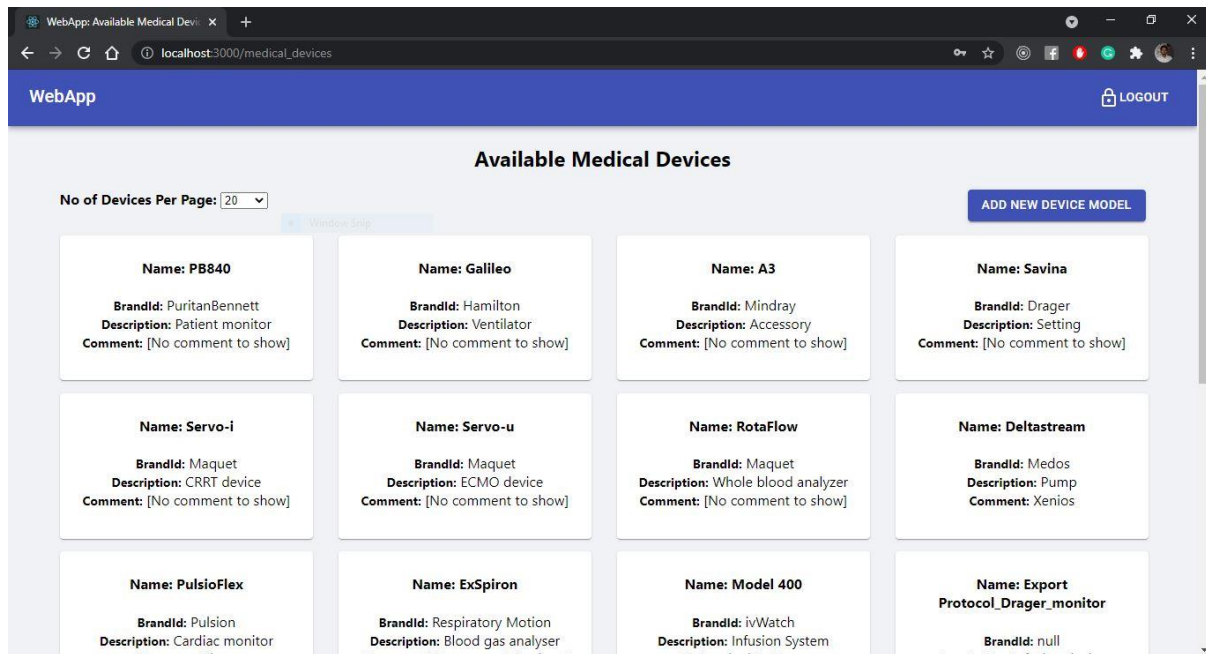
fig4: Available Medical Devices Page (Desktop View)
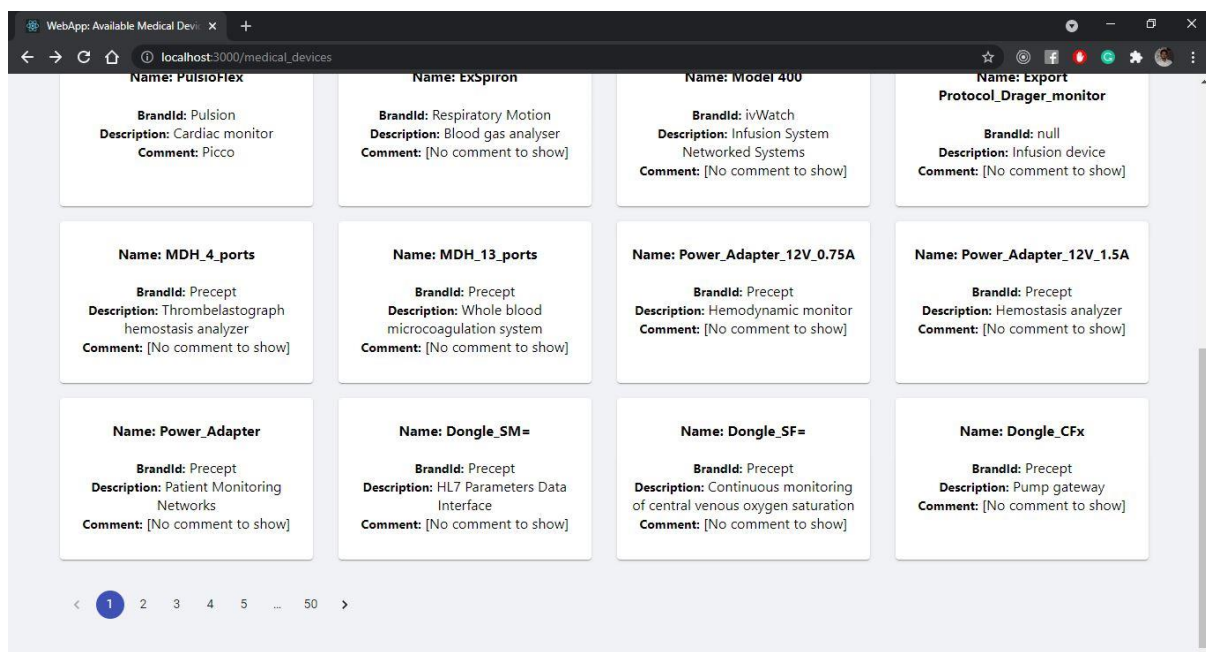
fig5: Available Medical Devices Pagination (Desktop View)

Available Medical Devices Page (Phone View):

fig6: Available Medical Devices (Phone view)

fig7: Available Medical Devices Pagination (Phone View)

Model Data Overlay Dialog (Desktop View): After click on a model device, user will see the model device data in an overlay Dialog. Before data is fully loaded user will see a **circle progress bar** on the dialog.
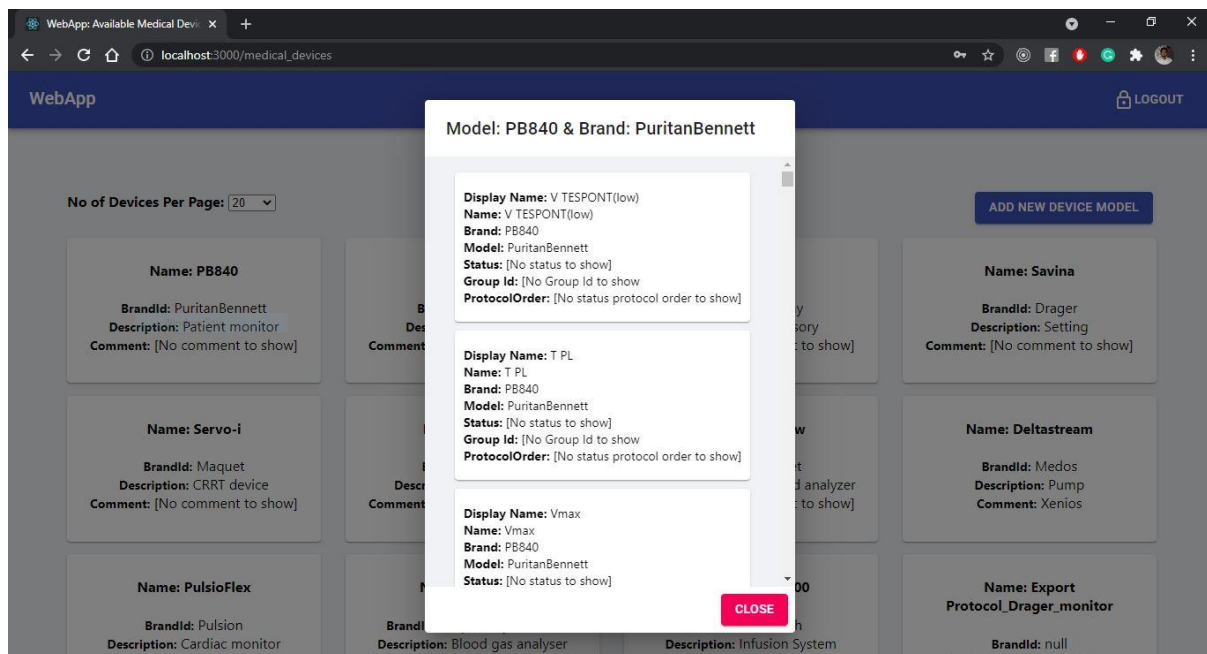
fig8: Model Data Dialog (If any Data found)
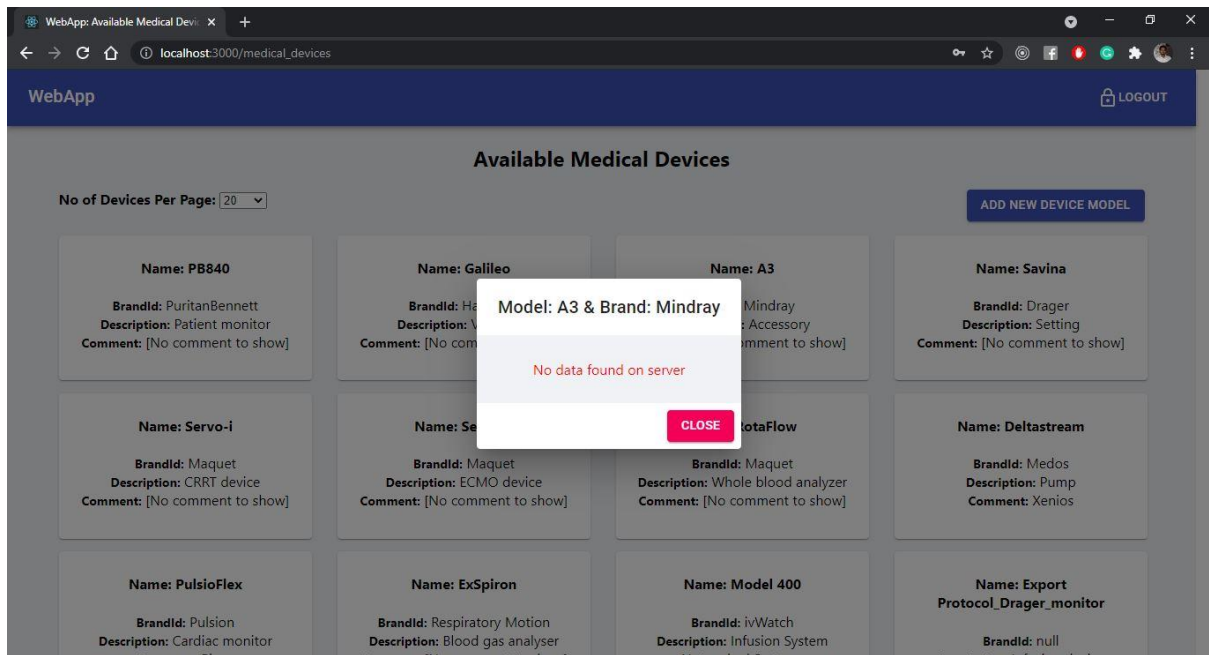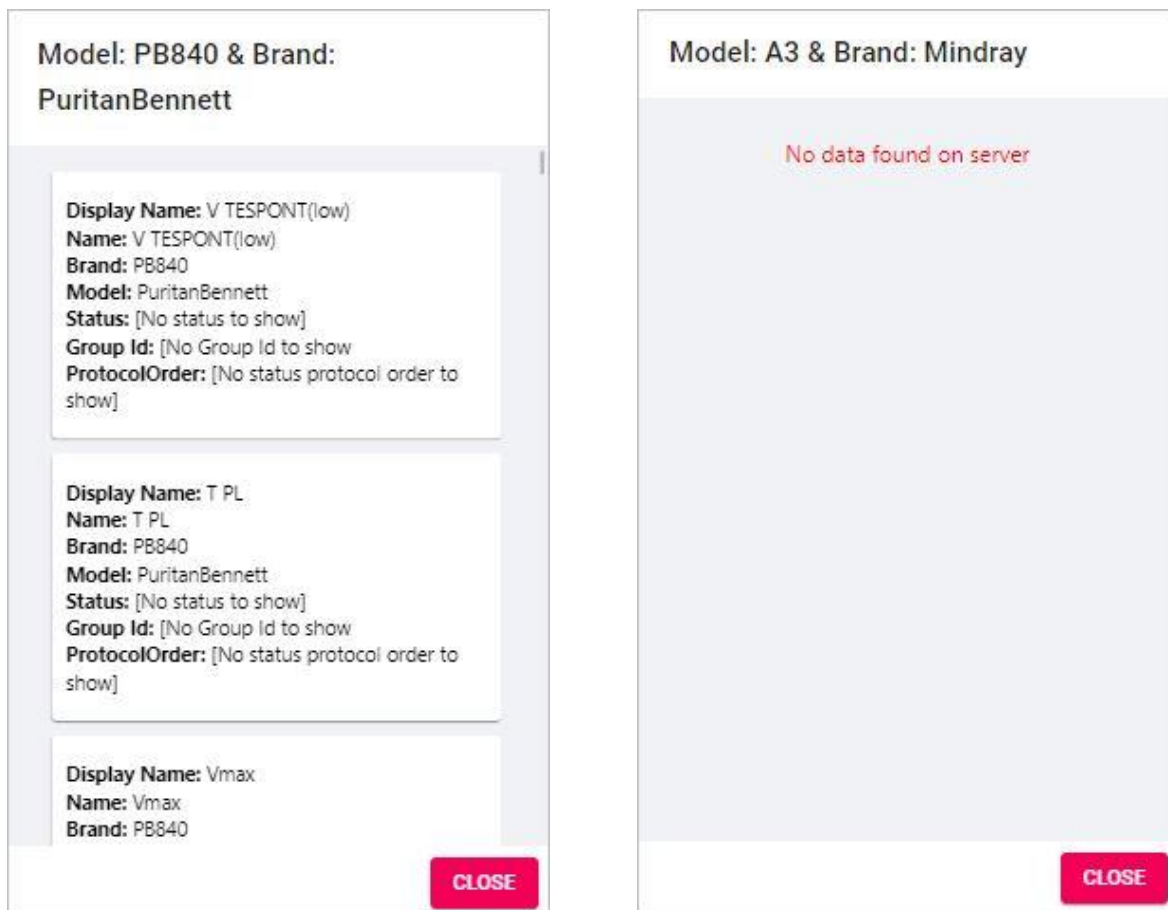
fig9: Model Data Dialog (If no data found)

Model Data Overlay Dialog (Phone View):

fig10-11: Model Data Dialog- Phone View (left- If data found, right-If no data found)

Add New Device Model (Desktop View): Create error message or success message will be shown above first input field.

fig12: Add new device model form (Desktop View)

Add New Device Model (Phone View):

Fig13: Add new device model form with dialog (Phone view)

Thank you