# LMSv3 React App Documentation

## External Packages

* moment
* react-calendar
* react-circular-input
* react-router-dom
* recharts
* react-helmet
* three
* @react-three/fiber
* @react-three/drei
* axios
* react-colorful
* react-redux, @reduxjs/toolkit

## Design Specs

<https://xd.adobe.com/view/49322a84-66bd-4035-9f38-c26e3980fee5-1c61/specs/>

## Backend API

Main link:

<https://sls.aztech.com:8888/api/v1>

Endpoints (append behind link):

List of areas and their blocks: /areas

List of floor and light data of block: /block/(id)

Hourly energy consumption: /energy/block/(id)/from/(yyyymmdd)/to/(yyyymmdd)

Daily energy consumption: /energy/block/(id)/from/(yyyymm)/to/(yyyymm)

Monthly energy consumption: /energy/block/(id)/from/(yyyy)/to/(yyyy)

## Current Status

Most UI functionality is complete, except for pages/tabs which are not present in the design specs. The ThreeJS scene is implemented in the “Add” page, found on the bottom of the column of sidebar icons. Backend API integration is limited to the dashboard page, where the light data is queried based on the location and blocks selected.

## Incomplete Features

All backend-related and configuration features such as:

* Light configuration
* User management
* Login authentication
* Gateway information

Frontend features

* Search bar
* Dark mode
* Web page responsiveness (CSS, media queries)
* Exporting data

## Source Directory Organisation

**Entry point:**

src/index.js

* Renders Main (in src/Main.js) component.

src/Main.js

* Contains the Helmet (page title), default routes and wraps the app with the Redux store provider.

**Folders:**

src/components

* Contains all the components used in the app.

src/components/three

* Contains the components used for the three.js scene.

src/redux

* Contains the Redux store.

src/resources

* Contains all image and font resources. (sorted by components or pages)

src/resources/css

* Contains all CSS files used in the app.

## Redux

### Files

**store.js**

Contains the Redux store and its list of variables/reducers.

**create.js**

Contains a helper function used to initialise a variable in the store, which is used in the slice files in this folder.

## Components

### Login/Landing Page

**LoginForm**

The initial page of the site which contains a link to the sign-up form.

**SignUpForm**

Renders the sign-up form on the initial page.

**Dashboard**

Renders the landing page after login, as well as the dashboard after the user selects "block" and "area". Persistent elements of the page such as the location selectors, sidebar, notifications, and user profile dropdown lists are held here.

**SelectorDropdown**

The component used to create the dropdown lists for location selection.

**EditProfile**

The pop-up panel for editing the user profile.

**SearchBar**

Search bar on header of page.

**Notification**

Notification dropdown list on upper right of page. Will be disabled if no blocks are selected.

**UserDropdown**

Dropdown menu when clicking on user on top right, includes link to EditProfile, user management and logout.

**Sidebar**

Contains the icons used to navigate between the pages, as well as the dark mode toggle button, clock, and version information.

### Dashboard Main Page

Note: Level dropdown selector is not needed for operation.

**DashboardView**

The main dashboard page. Contains the cards displaying the block data.

**BlockLights**

Displays overview of lights in the block.

**ActiveLights**

Displays the 10 most active (most motions detected) lights in the block.

**EnergyConsumption**

Displays energy consumption history of lights in the block. (options 5D and 3Y use 1D and 1Y respectively)

**LightControl**

Quick configuration of lights for this block. (not implemented)

**ActivityLog**

Displays the activity log of lights for this block.

**GatewayInfo**

Displays the gateway information for this block. (not implemented)

**LightStatus**

Displays the status of lights for this block, as well as allows the user to edit their location description. (Location editing not implemented)

**Relocation**

The pop-up panel for editing the light’s location description.

### General Configuration Page

### Misc. and common components or files

**MockAPI.js**

Contains functions for obtaining data from either the backend API or test values.

**Utility.js**

Contains helper and utility functions.

**RouteManager**

Contains the paths for all the pages in the app.

**SidebarIcon**

Component for sidebar icon.

**Timestamp**

Used to print the clock at the bottom of the sidebar.

**EnergyConsumptionOption**

The group of buttons used to select the display range of the energy consumption graphs.

**EnergyConsumptionGraph**

Wrapper for the recharts package used to create the energy consumption graphs.

**GenericDropdown**

A general-purpose dropdown box mainly used in the card components.

**TableSortButton**

Used for the tables shown in the main dashboard page.

## Three.js Scene

### Usage

Scene data is saved in a JSON file which contains:

* Floor plan image name
* Array of light data
* Array of group colour definitions

For example, a sample saved scene might look like:

{

“img”: “default”,

“lights”: [

{

“name”: “1.1.1”,

“pos”: [0, 0, 0],

“selected”: false,

“highlight”: false,

“mode”: “ON”,

“group”: “1”,

“triggerers”: [],

“triggerees”: [“1.1.2”],

}

{

“name”: “1.1.2”,

“pos”: [0, 0, 1],

“selected”: false,

“highlight”: false,

“mode”: “ON”,

“group”: “0”,

“triggerers”: [“1.1.1”],

“triggerees”: [],

}

],

“groupColours”: {“0”, “#ff0000”}

}

The img property defines the name of the floor plan used in this scene.

The array of light data contains data for each light in the scene.

The array of group colour definitions keeps track of the custom colours assigned to each group. If none are created, the array will be empty, and all groups would use a default grey colour.

Currently, the scene is loaded via the loadData(name) function, which can be found in the ThreeJsScene component in ThreeJsScene.js. When called with the “default” argument, it loads a blank scene without the use of any JSON files. The current “default” would load the c1basement1 image for testing purposes. In an actual implementation, this function would fetch the image defined in the JSON file from a predefined endpoint.

### Controls

**General:**

LMB: Pan/Select light

RMB: Rotate

CTRL + LMB: Multiselect (drag selection box or single select)

Scroll Wheel: Zoom

Space: Toggle add mode

G: Toggle group view

T: Toggle trigger view

1/2/3: Select Groups 0/1/2 (for testing purposes)

Q/W: Load predefined data (for testing purposes, can be modified in ThreeJsScene.js)

N: Toggle light name display

R: Reset camera position

S: Save scene (prompts save location)

**Add Light/Edit Trigger Mode:**

LMB: Add

RMB: Remove

All control functionalities can be found in the ThreeJsScene component.

### ThreeJs Components