

# HI-ER Readme

---

This code is used for the <https://er.pumps.org/> website.

## Getting the Application

The application consists of many folders and files. In order to allow multiple people to work on the HI-ER content, and for continued application development, the HI-ER is managed through a [git](#) repository. [git](#) is a platform that allows for very powerful version control - it tracks changes and allows you to "commit" them and "push" your changes to a central repository so others can always see the latest work.

Git needs to be installed on your machine. You can download it [here](#)

We will not be using many features of [git](#), and since the number of people editing the HI-ER content will be somewhat limited, it will be rare for you to need any advanced knowledge of how git works.

The initial step is to create a Github account - this website is the central repository for the HI-ER, and it is a private repository - you will need to be granted access before moving forward. Create your account [here](#), and contact me ([sfrees@intelliquip.com](mailto:sfrees@intelliquip.com)) for access.

Once you have access, open the Command Prompt and use the `cd` command to navigate to the directory you want to put the HI-ER application in. For example, create a directory called `C:\projects\`, and navigate the command prompt there by entering `cd C:\projects` at the command prompt.

Next, clone the repository with the following command:

```
git clone https://github.com/edlpumps/hi-er.git
```

This will create a directory called `hi-er` under your `C:\projects` directory.

Later, as you begin to make changes to the HI-ER, you will need to use [git](#) to commit your changes. The details of this is covered below in the "Version control with [git](#)" section.

## Text Editor - Visual Studio Code

While any text editor is suitable for creating the HI-ER content, you should use something geared towards programming, to avoid character encoding problems. You may download and install this [here](#).

Once installed, you can open the `hi-er` directory.

Visual Studio Code allows you to open a folder - `hi-er` - which is the most efficient way of working. This will give you a side panel on the left side of the screen that you can use to navigate and open any file in the directory structure.

## Running code Locally

Initialize Node JS and NPM in your Development environment

Follow instructions at: <https://code.visualstudio.com/docs/nodejs/nodejs-tutorial>

Additional information can be found at: <https://developer.ibm.com/tutorials/learn-nodejs-installing-node-nvm-and-vscode/>

Confirm npm is installed

```
$> npm --help
```

Confirm node.js is installed

```
$> node --version
```

Just one time, install the modules for your development environment:

In your VSC app and in a **powershell** terminal window:

```
$> npm init
$> npm install
$> npm audit fix <== if needed
```

## Local Database

You must be connected to a local Mongoose database install MongoDB install MongoDB Compass create .env with:

- MONGO\_CONNECTION\_DATA = mongodb://127.0.0.1:27017/er
- ADMIN\_PASSWORD\_OVERRIDE=test

You will need to have your user information in the database.

If MongoDB Compass is running but it won't connect to the local database:

- Open Task Manager
- Click on the Services tab
- Make sure MongoDB Server is running

## Local Preview - Debugging

The easiest way to do your development is to execute the following command from the root project directory in a Terminal window:

```
$> node index.js
```

OR to set breakpoints and debug:

```
Open index.js file in VSC
Click on the Debug icon on the left
Select Launch Program
Step through code
Use the debug toolbar to step into/over code
Use the debug toolbar to restart the code and stop execution
```

To monitor for code changes while running index.js, install **supervisor**:

```
$>npm init
$>npm install
$>npm -g install supervisor
```

Then run the code using **supervisor**:

```
$>supervisor index.js
```

Connect to the localhost on port:

```
http://127.0.0.1:3003
```

## Testing Subscriber XLSX Attachments

- In your development environment, to test the generation of the subscriber Full and QPL spreadsheets, run **export-file.js**:

```
$> node export-file.js
```

- To test the **EMAIL** process, you can also log onto the portal as **admin** and type in the following URL: **https://<base\_url>/admin/export/email/<email\_address>** where **<email\_address>** is the email address to which you want to receive the spreadsheets.
  - If **<email\_address>** is omitted, email will be sent to the logged-in admin's email address.
- You will receive two emails:
  - One with the Qualified Product List listings (Pumps & Circulators)
  - One with the Full listings (Pumps, Circulators & Certificates)
- Admin endpoints for downloading the Full Listing spreadsheets to your computer are:

- [https://<base\\_url>/admin/export/pumps](https://<base_url>/admin/export/pumps)
  - [https://<base\\_url>/admin/export/circulators](https://<base_url>/admin/export/circulators)
  - [https://<base\\_url>/admin/export/certificates](https://<base_url>/admin/export/certificates)
- Admin endpoints for downloading the QPL Listing spreadsheets to your computer are:
    - [https://<base\\_url>/admin/export/pumps/qpl](https://<base_url>/admin/export/pumps/qpl)
    - [https://<base\\_url>/admin/export/circulators/qpl](https://<base_url>/admin/export/circulators/qpl)

## Deploying a GitHub branch to Heroku

First, push the branch to GitHub and create a Pull Request.

Login to Heroku using the [higladetech@gmail.com](mailto:higladetech@gmail.com) account on a browser:

```
https://heroku.com
Select the "intelliquip-hi/hi-er-beta" project (make sure that it is NOT in
Maintenance mode in the Settings)
Select Deploy
Select Deploy a GitHub branch and enter the branch name
```

## Deploying GitHub Master branch to Heroku

Make sure all branches are merged to **master** on GitHub.

Follow the steps above to Deploy a GitHub branch to Heroku, but instead, select the **intelliquip-hi/hi-er** project and deploy the **master** branch.

## ER LABELS

- There are **hard coded** values for Pump Annual Run Hours (4000), Circulator Annual Run Hours (2500) and the Cost/KwH (\$0.15). These are used in the Energy & Cost Savings calculations.  
([routes/common.js](#))
- The year of the DOE Circulator Efficiency Regulation is a variable that is passed into the label pub  
([utils/label\\_builder.js](#), [views/svg/circulator-label.pug](#))
- These values can easily become Environment Variables.
- The *Meets the 2028 DOE circulator efficiency regulation* line on the large *Circulator Pump Label* should only display if the CEI value that is displayed in the black bar on the label is less than or equal to 1.00.  
([utils/label\\_builder.js](#))

## PNG to URI

- If you add an image to the Energy Ratings label (such as a logo or icon), you need to convert the image to URI content and save that as a file.
  - Use an online coverter like <https://site24x7.com/tools/image-to-datauri.html>
  - Upload the image (PNG, JPEG, whatever) to the tool which generates the URI code.
  - Copy the content and paste into a file (no extension) in the views/svg folder
  - Pass that link into the SVG generator code for generating the label.

- The URI that is generated for the HI logos on the large label are generated from:
  - English ([views/svg/hi-title](#)): hydraulic-institute-logo.png
  - French ([views/svg/hi-title-fr](#)): HI-Energy-Rating-ID\_French\_03\_Transparent.png
- The URI that is generated for the HI logos on the small label are generated from:
  - English ([views/svg/hi-title-small](#)): hydraulic-institute-logo-sm.png
  - French ([views/svg/hi-title-small-fr](#)): HI-Energy-Rating-ID\_French\_03\_small\_Transparent.png
- The URI that is generated for the Circulator Labels with the approval check on the bottom of the large label are generated from:
  - All languages ([views/svg/hi-approval-check](#)): HI-Approval-Check-s.png

## TRANSLATIONS TO OTHER LANGUAGES

- Pump LABELS can be translated using the English/French toggle at the top right of all Pump Detail pages.
- The pump details themselves are only translated for the PUBLIC pages (not the Admin or Participant pages per design).
- The i18next (<https://i18next.com>) library is used.
- Since the requirement was to provide translations for particular pages (Public Pump details) and for labels, there are 2 "types" of language settings.
  - page\_lang
  - label\_lang
- These are handled independently in the code.

## Generate README.PDF

- Make sure the [Markdown PDF](#) extension is installed on Visual Studio Code (Author yzane).
- Open the README.md file
- Press the 'F1' key
- Select the [Markdown PDF](#) extension (or search for it). The README.pdf file will automatically be generated.