1. **Front End UI Deployment steps:**
   1. Open a cmd prompt and CD to the location of the contents of the Intel\_UI folder
   2. Enter the command < npm install >
      1. Ensure that the command returns no Errors
   3. Enter the command < bower install >
      1. Comand will prompt user to select a version for <px-datetime-common>
      2. Select the first option by inputting < !1 >
   4. Enter the command < gulp dist -f >
   5. Modify the configuration files per the instructions and examples below:
2. Modify Configuration Files

Here we have the path file which need to be modified in order to set the Predix instances (UAA, TimeSeries, Redis, Assets) and the microservices.

* /<rootname>/server/localConfig.json
* /<rootname>/server/app.js
* /<rootname>/server/routes/proxy.js
* /<rootname>/public/elements/seed-app/seed-app.html
  1. **localConfig.json File**

Here you need to modify the two base64ClientCredential values. Change the value of the tag uaaURL. If you are using Assets, enter the value for assetURL. The backend microservice is named redisServiceURL, enter the new value.



**app.js File**

The predix-webapp-starter Express web application includes these features:

* Routes to mock data files to demonstrate the UI.
* A proxy module for calling Predix services such as asset and time series
* Passport-predix-oauth for authentication, and a sample secure route.

If you need to modify the endpoints and the name of your micro-service. Then you have to change the following code:



**proxy.js File**

This module can be used to set up reverse proxying from client to Predix services. In order to modify the POST method used to request data from the microservice, you will need to modify the following code:



1. **Push the Application to the Cloud:**

Pushing (deploying) to a cloud environment requires knowledge of the commands involved and a valid user account with the environment. GE uses Cloud Foundry for its cloud platform.

**Update manifest.yml**

* 1. Change the name field in your manifest.yml to match the desired name of the backend application (Note, this is the name that will appear in Predix for the app)
  2. Edit the service section to match the named services instances deployed in your Predix space:

services:

- <Enter UAA-service>

- <Enter TimeSeries-service>

- <Enter REDIS-service>

- <Asset Service>

- <Access Control Service>

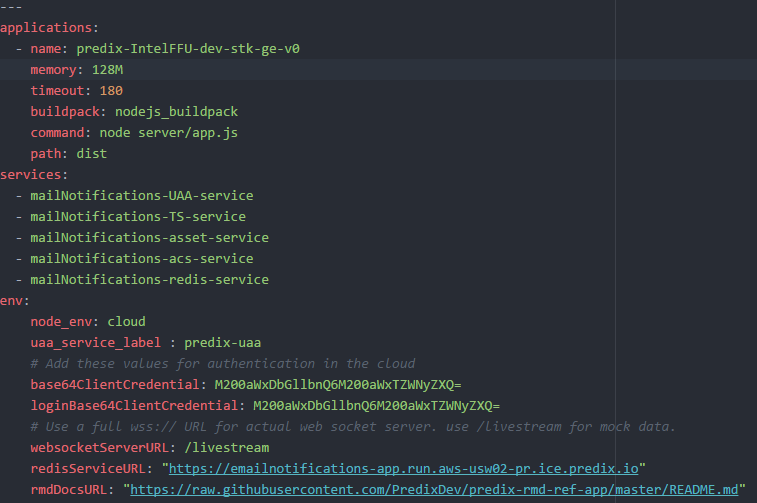
* 1. Edit the Environment (env) Section with the service instance details in your Predix space

**Base64ClientCredentials** <**bas64EncodedClientSecret** >

**loginBase64ClientCredentials** <**bas64EncodedClientSecret** >

**inteltsqueryURL** <full **URL for Timeseries** >

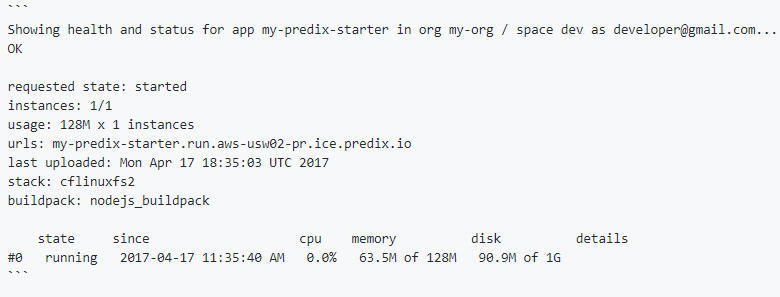
**redisServiceURL** <Backend App URL>



**Push to the cloud**

1. **RUN the Command < cf push >**
2. **Access the cloud deployment of your Starter application**

The output of the cf push command includes the URL to which your application was deployed. Below is an example:



1. Running the app locally

The default gulp task will start a local web server. Just run this command:

gulp

Browse to [http://localhost:5000](http://localhost:5000/). Initially, the app will use mock data for the views service, asset service, and time series service. Later you can connect your app to real instances of these services.