Getting Started with Cloud POS Client Codebase

This is the default web application for the POS. It is a Single-Page App (SPA) written in web standard technologies like JavaScript, CSS and HTML5. It leverages common popular frameworks/libraries such as AngularJS, Bootstrap and Font Awesome.

# Codebase

We are using AngularJS and requirejs in our application so that we have proper code organization. Some of the advantages of going this way are we don’t have to worry about including script tags in the right order when building Angular app. Also, here we will manually [bootstrap](https://docs.angularjs.org/api/ng/function/angular.bootstrap) our angularjs application.

## Some Important Files

**app/index.html**

This is the first file you will start with usually. Here we have the code of login page and sidenav. So, any changes you want to do in these components you can directly do here. Also, you will see a script tag below in body of requirejs

*<script type='text/javascript' data-main="scripts/loader.js" src='bower\_components/requirejs/require.js'></script>*

**app/scripts/loader.js**

Here you will find requirejs config. In paths we set aliases for the libraries and plugins used, then we defined that angular should be shimmed. You will see a deps variable which says that specific plugins dependency. For example: webcam-directive needs angularjs library to load before that. Next you will see the angular application is manually bootstrapped rather than using ng-app syntax which you usually see it's done like below.

*angular.bootstrap(document, ['CloudPOS\_Application']);*

**app/scripts/CloudPOS.js**

Here you will find all the dependency injections for third party components. Make sure while you add any library you inject it here.

**app/scripts/CloudPOSComponents.js**

Here you will find all controllers, services, filters and directive of the application defined. If you have to add new controller, service, filter or directive just mention it here and it will be loaded in your application.

**app/scripts/CloudPOSStyles.js**

Here all the styles files are defined. For adding new css file make sure you add here to load in the application.

**app/scripts/routes.js**

As, the file name suggest all the application routes are defined here with their templateUrl.

**app/scripts/initialTasks.js**

As, the file name suggest all the application initial tasks like setting http header, setting baseurl, and localisation.

## Application Structure

/community-app

/app

/WEB-INF

/angular

/i18n → angularjs locale definitions

/bower\_components → Libraries

/fonts → Font libraries

/global-translations → locale labels file for different languages

/images → images files

/scripts → application files (controller, filters etc)

/styles → Contains css files

/views → html template files

/node\_modules → node dependencies of application

/test → Contains various test cases

## Localisation

Community app has support for multiple languages and accordingly we are using labels inside our application. For example

*{{‘'label.heading.cloudposclient' | translate}}*

In locale-en.json file you will find similar entry and its value in english. Make sure when we are adding any new label we just create an entry in locale-en file and don’t touch other files.

*"label.heading. cloudposclient": "Cloud POS Client",*