

Table of Contents

- [Angular CI](#)
 - [TESTING LINKS](#)
 - [Pipeline](#)
 - [Adjustments](#)
 - [Pipeline Environment](#)
 - [Description](#)
- [Angular design](#)
 - [Introduction](#)
 - [Basic project structure](#)
 - [Main Views and components](#)
 - [Public area](#)

Angular CI

TESTING LINKS

refer to [page1#bookmark-a](#) for more info

refer to [page1.asciidoc#bookmark-a](#) for more info

[link:/page1](#)

[link:/page1.asciidoc](#)

[link:page1#Pipeline](#)

Xref page 1:

[xref:bookmark-a](#)

Anchor to page 1:

Click on [adjustments-id](#) to go to Adjustments

Links to page 2:

[link:/page2.asciidoc](#)

[link:page2#basic-project-structure](#)

The Angular client-side of My Thai Star is going to have some specific needs for the CI-CD Pipeline to perform mandatory operations.

1	2	A
3	4	B
5	6	C

Pipeline

The Pipeline for the Angular client-side is going to be called **MyThaiStar_FRONTEND_BUILD** . It is located in the PL instance, under the [MTS folder](#) (as previously explained). It is going to follow a process flow like this one:

angular pipeline flow

Each of those steps are called *stages* in the Jenkins context. Let's see what those steps mean in the context of the Angular application:

1. **Declarative: Checkout SCM**
2. **Declarative: Tool Install**
3. **Loading Custom Tools**
4. **Fresh Dependency Installation**
5. **Code Linting**
6. **Execute Angular tests**
7. **SonarQube code analysis**
8. **Build Application**
9. **Deliver application into Nexus**
10. **Declarative: Post Actions**

Adjustments

The Angular project Pipeline needed some "extra" features to complete all planned processes. Those features resulted in some additions to the project.

Pipeline Environment

In order to easily reuse the pipeline in other angular projects, all variables have been defined in the block environment. All variables have the default values that Production Line uses, so if you're going to work in production line you won't have to change anything. Example:

```

environment {
    // Script for build the application. Defined at package.json
    buildScript = 'build --configuration=docker'
    // Script for lint the application. Defined at package.json
    lintScript = 'lint'
    // Script for test the application. Defined at package.json
    testScript = 'test:ci'
    // Angular directory
    angularDir = 'angular'
    // SRC folder. It will be angularDir/srcDir
    srcDir = 'src'
    // Name of the custom tool for chrome stable
    chrome = 'Chrome-stable'

    // sonarQube
    // Name of the sonarQube tool
    sonarTool = 'SonarQube'
    // Name of the sonarQube environment
    sonarEnv = "SonarQube"

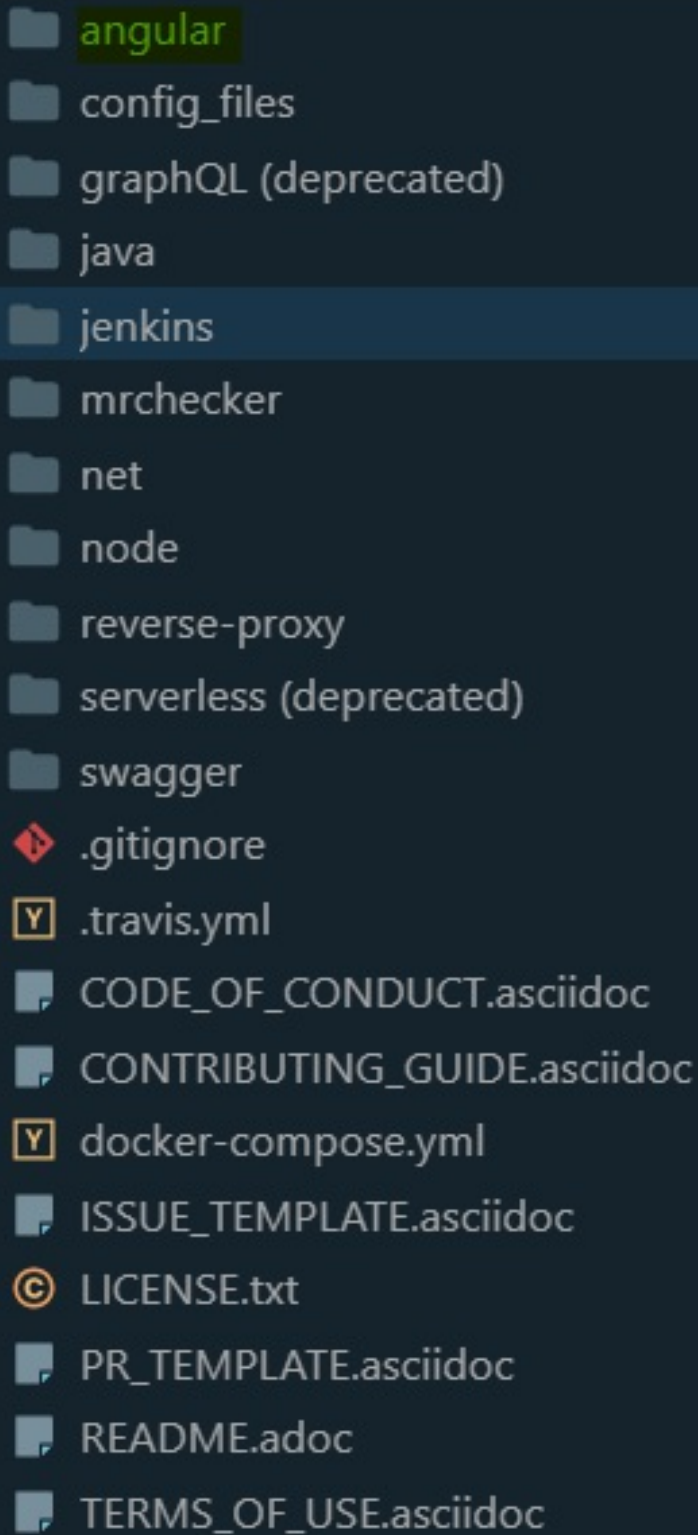
    // Nexus
    // Artifact groupId
    groupId = 'com.devonfw.mythaistar'
    // Nexus repository ID
    repositoryId = 'pl-nexus'
    // Nexus internal URL
    repositoryUrl = 'http://nexus3-core:8081/nexus3/repository/maven-snapshots'
    // Maven global settings configuration ID
    globalSettingsId = 'MavenSettings'
    // Maven tool id
    mavenInstallation = 'Maven3'
}

```

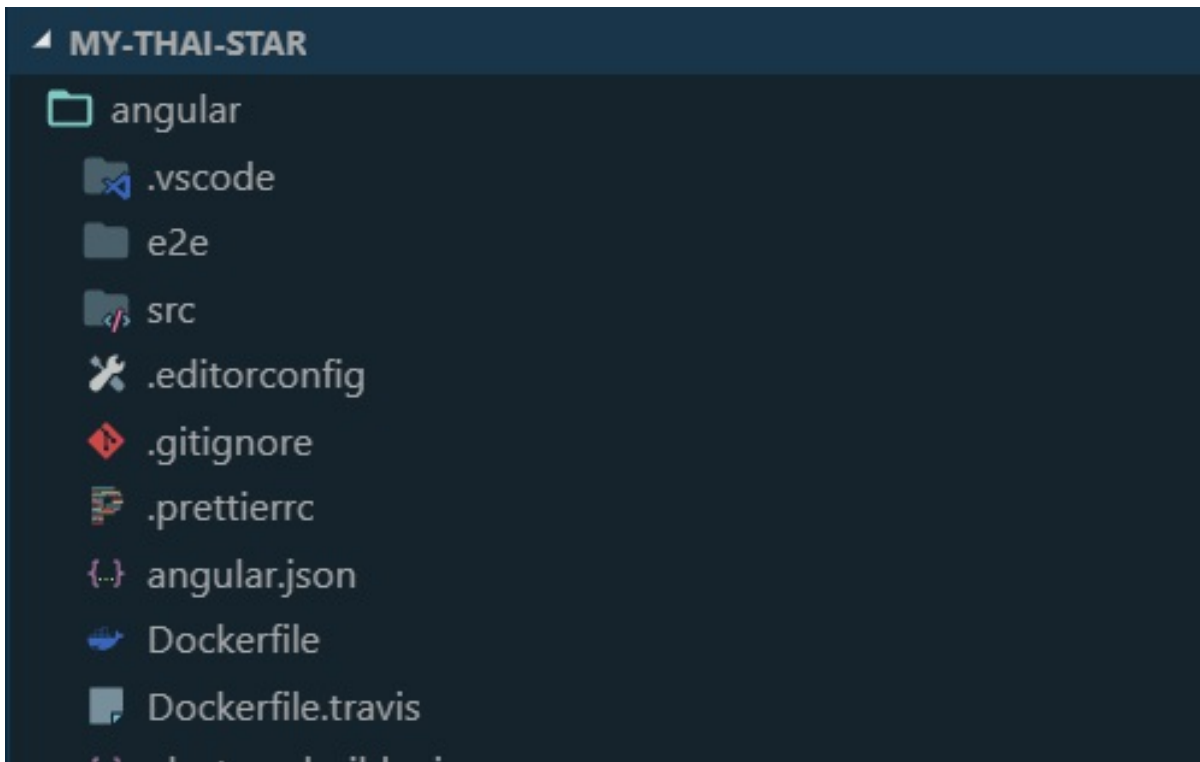
Description

- **buildScript** : script for build the application. It must be defined at package.json.
- **lintScript** : Script for lint the application. Defined at package.json
- **testScript** : Script for test the application. Defined at package.json
- **angularDir** : Relative route to angular application. In My Thai Star this is the angular folder. The actual directory (.) is also allowed.

MY-THAI-STAR



- **srcDir** : Directory where you store the source code. For angular applications the default value is `src`



- **chrome** : Since you need a browser to run your tests, we must provide one. This variable contains the name of the custom tool for google chrome.

Custom tool

Name

[Custom Tool Configuration...](#)

☒ Install automatically [?](#)

Run Shell Command [?](#)

Label

Command

```
if [ which google-chrome > /dev/null ]; then
sudo su << EOF
wget -q -O - https://dl-ssl.google.com/linux/linux_signing_key.pub | apt-key add -
echo "deb http://dl.google.com/linux/chrome/deb/ stable main" > /etc/apt/sources.list.d/google-chrome.list
apt-get update && apt-get -y install google-chrome-stable
EOF
fi
```

[?](#)

Tool Home [?](#)

[Delete Installer](#)

- **sonarTool** : Name of the sonarQube scanner installation.

SonarQube Scanner

SonarQube Scanner installations [?](#)

SonarQube Scanner

Name

☒ Install automatically [?](#)

Install from Maven Central

Version

[Delete Installer](#)

[Add Installer](#)

[Delete SonarQube Scanner](#)

- **sonarEnv** : Name of the sonarQube environment. SonarQube is the default value for PL.

- **groupId** : Group id of the application. It will be used to storage the application in nexus3
- **repositoryId** : Id of the nexus3 repository. It must be defined at maven global config file.
- **repositoryUrl** : The url of the repository.
- **globalSettingsId** : The id of the global settings file.
- **mavenInstallation**: The name of the maven tool.

Angular design

Introduction

MyThaiStar client side has been built using latest frameworks, component libraries and designs:

Angular 4 as main front-end Framework. <https://angular.io/>

Angular/CLI 1.0.5 as Angular tool helper. <https://github.com/angular/angular-cli>

Covalent Teradata 1.0.0-beta4 as Angular native component library based on Material Design. <https://teradata.github.io/covalent/>

Angular/Material2 1.0.0-beta5 used by Covalent Teradata. <https://github.com/angular/material2>

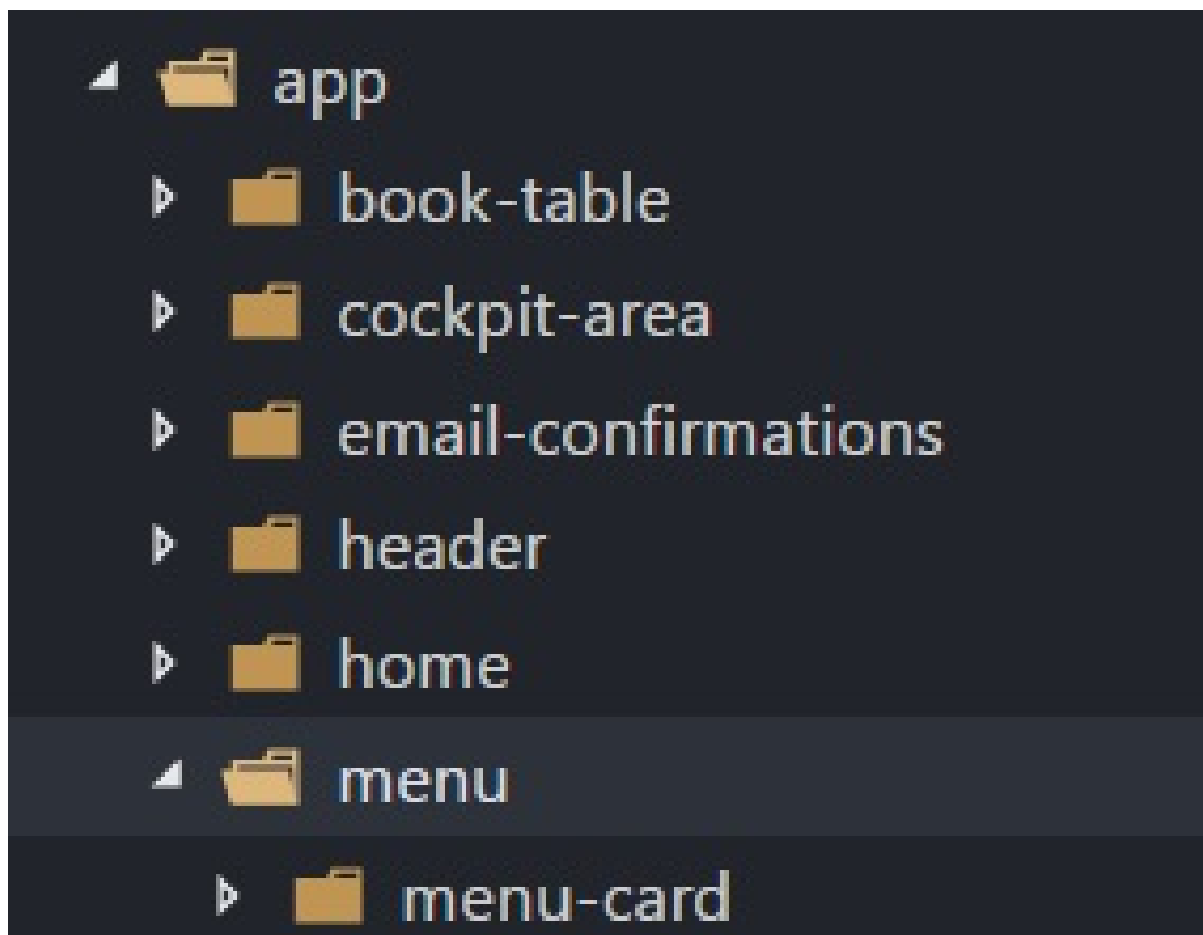
Note: this dependencies are evolving at this moment and if it is possible, we are updating it on the project.





Basic project structure







The project is using the basic project seed that Angular/CLI provides with "ng new <project name>". Then the app folder has been organized as Angular recommends and goes as follows:

- app
 - components
 - sub-components
 - shared
 - component files
 - main app component
- assets folder
- environments folder
- rest of angular files

This structure can be shown in the following example image:

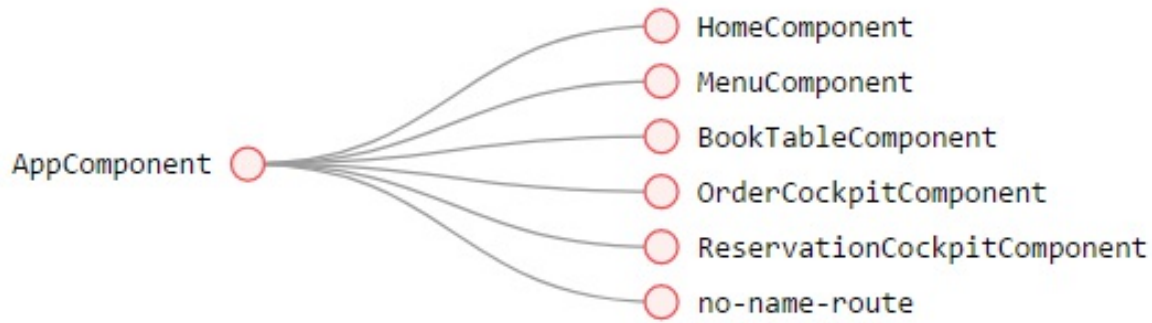


- ▶  shared
 -  menu.component.html
 -  menu.component.scss
 -  menu.component.spec.ts
 - TS** menu.component.ts
 - TS** menu.module.ts

- ▶  shared
- ▶  sidenav
- ▶  user-area
 -  app.component.html
 -  app.component.scss
 -  app.component.spec.ts
 - TS** app.component.ts
 - TS** app.module.ts
 - TS** app.routes.ts
 - TS** config.ts
 - TS** index.ts

Main Views and components

List of components that serve as a main view to navigate or components developed to make atomically a group of functionalities which given their nature, can be highly reusable through the app.



Note: no-name-route corresponds to whatever URL the user introduced and does not exist, it redirects to HomeComponent.

Public area

AppComponent

Contains the components that are on top of all views, including:

Order sidenav

Sidenav where selected orders are displayed with their total price and some comments.

