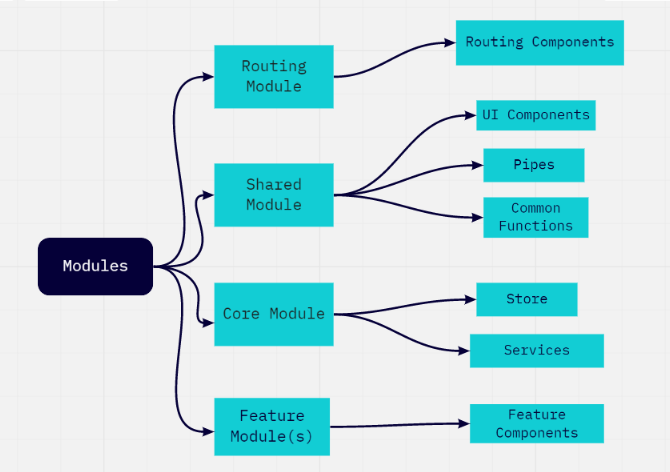
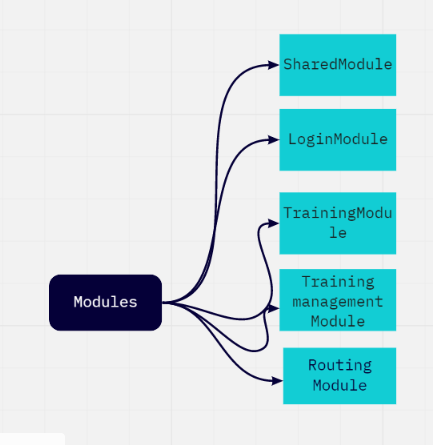
General Architecture

First let us define a general architectural diagram for an angular application.



An angular app is split into multiple modules. Let’s discuss about these modules a bit.

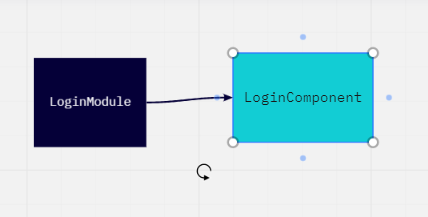
* Routing Module : Used to define all the routes in our application. Used by Routing component.
* Shared Module: Most of the shared code goes into this module. It contains common code to be used by more than 1 of the components in our application. General use cases are pipes, ui components, ui frameworks(bootstrap, material ui etc)
* Core Modules: The singletons should be defined in the core modules. This should be instantiated only once.
* Feature Modules: These are the feature modules and can be more than one based on the structure of the application.

Training Portal:  
  
Here is how the modules would look like in our training portal.  
  


Creating modules ensure that there is code separation and scalability in an application.

We shall go into each of the modules and the components individually.

LoginModule:

The login Module is one of the smaller modules and contains only one component for now  
  


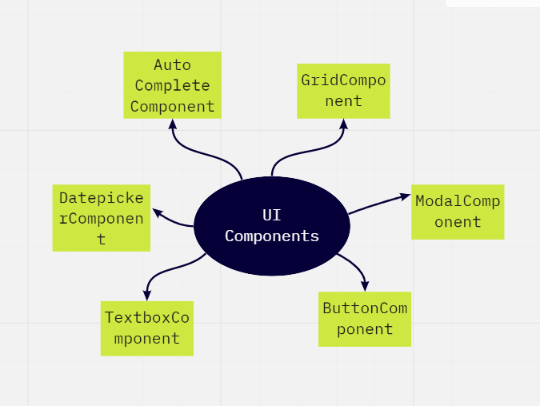
The logincomponent is responsible for authentication for either an sso or an authentication provider

SharedModule:

The Shared Module contains the common code for the application.

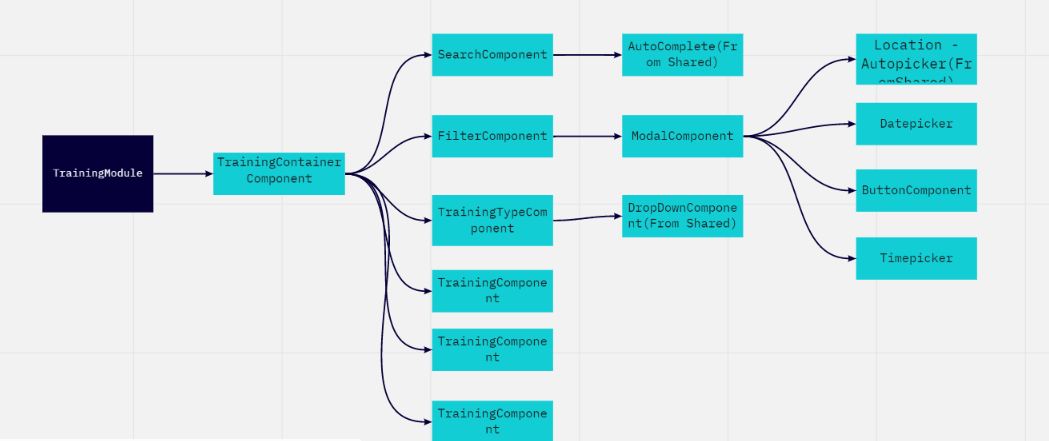


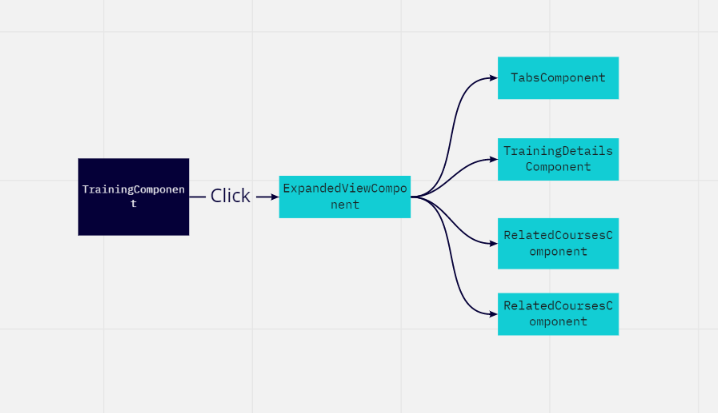
Let’s take a look at how the UI Components would look like in our application:



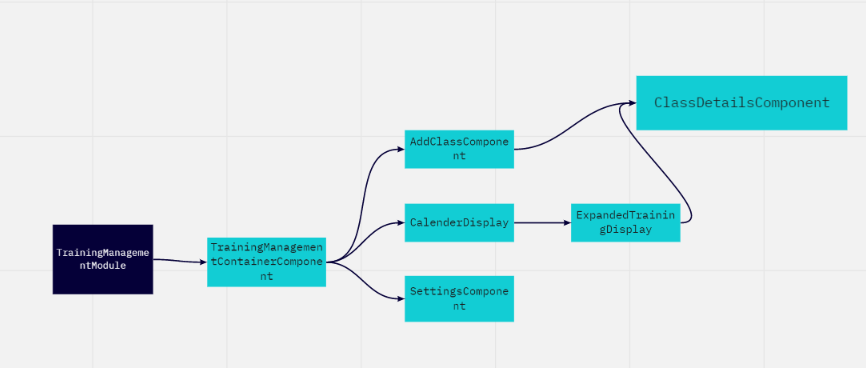
These are general UI components to be used across our application. We can also use a private npm package to host these to be used across applications.

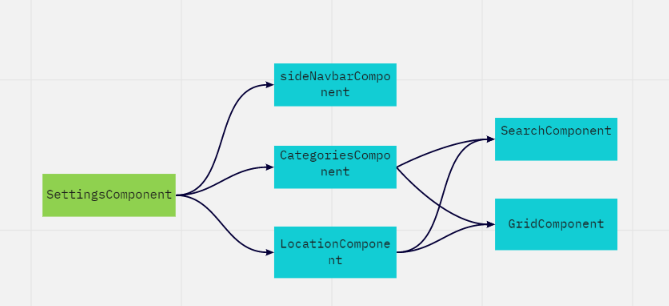
TrainingModule:

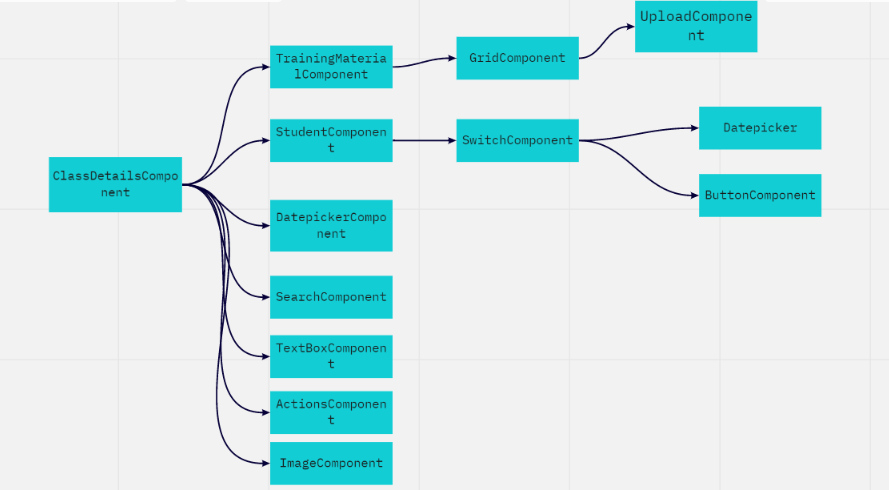




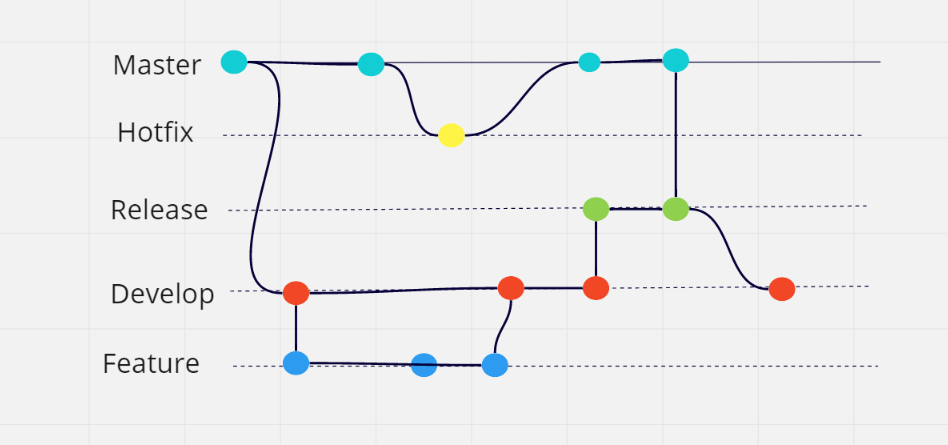
TrainingManagement Module:







Git Workflow:



This is how the git workflow would look like.

* Using 2 branches to record history of the project : Master and Develop. Develop is pulled from master at the init stage.
* Hotfix Branches are for issues which are detected into master( Prod Issues).
* When a release date approaches we fork develop into a release branch to ensure no more features can go into the release only bug fixes. Once tested it is merged to master and released and then merged to develop as well.
* Feature branches are for feature releases. They are created from develop and merged into develop.