**Dependencies:**

Dependencies are modules which are required during development and runtime as well. To save a dependency as a devDependency on installation we need to do a : npm install –save

Some good examples of dependencies which would be required at runtime include React, Redux, Express, and Axios.

**Dev Dependencies:**

Dev Dependencies are modules which are only required during development. To save a dependency as a devDependency on installation we need to do a : npm install --save-dev

Some good examples of when to install devDependencies would be Nodemon, Babel, ESLint, and testing frameworks like Chai, Mocha, Enzyme, etc…

**Local Installation:**

Local packages are installed in the directory where you run npm install <package-name>, and they are put in the node\_modules folder under this directory.

If you’re installing something that you want to use *in* your program, using require('whatever'), then install it locally, at the root of your project.

**Global Installation:**

Global packages are all put in a single place in your system (exactly where depends on your setup), regardless of where you run:

npm install -g <package-name>

If you’re installing something that you want to use in your *shell*, on the command line or something, install it globally, so that its binaries end up in your PATH environment variable.

**Angular Material:**

Angular Material is both a UI Component framework and a reference implementation of Google's Material Design Specification. The project provides a set of reusable, well-tested, and accessible UI components based on Material Design. It will help you to create a rich, interactive and device-oriented UI for your Web app. You don't need to buy any license for using Angular Material.

Benefits:

1. Accomplish high responsiveness of the layout.
2. Cross-platform user interface components, so that it can work on Web, Mobile, and desktop app as well.
3. You can select any theme using multiple layouts and can integrate into Angular apps.

**Animation:**

Animation provides the illusion of motion: HTML elements change styling over time. Well-designed animations can make your application more fun and easier to use, but they aren't just cosmetic.

Typically, animations involve multiple style *transformations* over time. An HTML element can move, change colour, grow or shrink, fade, or slide off the page. These changes can occur simultaneously or sequentially. You can control the timing of each transformation.

Angular's animation system is built on CSS functionality, which means you can animate any property that the browser considers animatable.

**Routing Guards:**

The Angular router’s navigation guards allow to grant or remove access to certain parts of the navigation. Another route guard, the CanDeactivate guard, even allows you to prevent a user from accidentally leaving a component with unsaved changes.

Here are the 4 types of routing guards available:

* CanActivate: Controls if a route can be activated.
* CanActivateChild: Controls if children of a route can be activated.
* CanLoad: Controls if a route can even be loaded. This becomes useful for feature modules that are lazy loaded. They won’t even load if the guard returns false.
* CanDeactivate: Controls if the user can leave a route. Note that this guard doesn’t prevent the user from closing the browser tab or navigating to a different address. It only prevents actions from within the application itself.