**App Structure**

**systemjs.config.js :--**

**It allows to configure SystemJS to load modules compiled using the TypeScript compiler. For anonymous modules (one module per JS file), it allows to map the name of modules to JS files that actually contains the module JavaScript code.**

Here is a sample. If I try to import the module named app/test, SystemJS will do:

* Try to find a preregistered module (with System.register('app/test', ...
* If not, it will look into its configuration to build the request to execute to load the corresponding file:
  + there is a map entry for app
  + there is a packages entry for app with defaultExtension = js
* The request will be http://localhost:3000/app/test.js. If you have map: { app: dist }, the request would be http://localhost:3000/dist/test.js

**tsconfig.json:--**

**It is the compiler configuration file which is responsible for generating Javascript file from typescript using tsc compiler.**

**bs-config.json:--**

**It deals with the light weight server which works with browserSync to detect any changes in code and reflect it in browser without manually refreshing the page.**

**package.json:--**

**Configuration file for all the node modules and primeNg modules downloading and integrating them with the app.**

**tslint.json:--**

**TSLint is an extensible static analysis tool that checks TypeScript code for readability, maintainability, and functionality errors. It is widely supported across modern editors & build systems and can be customized with your own lint rules, configurations, and formatters.**

TSLint supports:

* custom lint rules
* custom formatters (failure reporters)
* inline disabling and enabling of rules with comment flags
* configuration presets (tslint:latest, tslint-react, etc.) and plugin composition
* automatic fixing of formatting & style violations **.**