Configuring RTL and JEST in our Application

1. Install RTL using cmd npm i -D @testing-library/react

2. Install Jest using cmd npm i –D jest

3. Install Babel dependencies using cmd npm install -D babel-jest @babel/core @babel/preset-env

4. Configure babel with current node version and to do that create a file babel.config.js in project root level and paste below code there.

```
module.exports = {presets: [['@babel/preset-env', {targets: {node: 'current'}}]],};
```

5.configure parcel config file to disable default babel transpiration. Up to this point Parcel bundler has been using babel. Parcel has its own babel configuration and now jest is using babel as per Jest configuration. Jest-babel configuration is overriding parcel-babel configuration. This will create a conflict between Jest and Parcel. We will have to change Parcel behavior to use babel along with jest and to do that create .parcelrc under root directory with following configuration.

```
{
 "extends": "@parcel/config-default",
 "transformers": {
  "*.{js,mjs,jsx,cjs,ts,tsx}": [
   "@parcel/transformer-js",
   "@parcel/transformer-react-refresh-wrap"
  ]
 }
}
```

6. Configure Jest using cmd npx jest –init

7. If jest version is > 28 Install JS DOM library using cmd npm i -D jest-environment-jsdom

8. Install @babel/preset-react using npm i -D @babel/preset-react to make JSX work in test cases.

9. Include @babel/preset-react inside babel config.

```
module.exports = {
 presets: [
  ["@babel/preset-env", { targets: { node: "current" } }],
  ["@babel/preset-react", { runtime: "automatic" }],
 ],
};
```

"@babel/preset-react is helping our testing library to convert JSX code to html so that it will read properly

10. Install @testing-library/jest-dom using npm i -D @testing-library/jest-dom

11. Optional - Jest code Intellisense - npm install @types/jest

We have successfully configured RTL, JEST, babel & parcel in our project