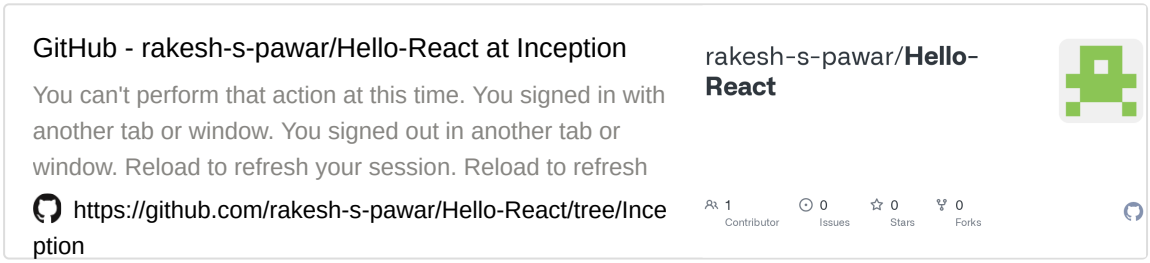
Hello React Course

▼ Inception



▼ Igniting our App

▼ Bundlers

There are lots of bundlers like Vite, Parcel, Webpack, etc. We are using Parcel in our project.

> 💡 In the original “Create-React-App” Webpack bundler was used.

Parcel is a bundler, it is a package. To use such package in our React app, we need package manager. We will be using NPM.

We need NPM, because we need to use lots of packages in our projects. To manage such packages we use NPM.

We need to minify our app, we need to bundle things, we need to remove console logs, we need to optimize our app, for these kind of tasks we need lots of helper packages. Those helper packages are managed using NPM.

```
// start with npm init in Terminal
npm init

// it will ask lots of questions. Click Enter if don't want to change
package name: (hello-react)
version: (1.0.0)
description: This is a Namaste React Tutorial course.
entry point: (App.js)
test command: jest
git repository: (https://github.com/rakesh-s-pawar/Hello-React.git)
keywords:
author: Rakesh Pawar
```

```
license: (ISC)
Is this OK? (yes)
```

💡 After this process, we will get package.json in our project. This file contains configuration which NPM needs to run.

To get “Parcel” in our App, we need to run some commands

```
// we can use
npm install parcel

// but we don't want Parcel on production, we want it on our local machine
// we use "-D", which means dev dependancies
npm install -D parcel

// Some people use "--save-dev", it is same as using "-D"
```

💡 Dependency means, all the packages that a project needs. Parcel is one of such dependency.

After running “npm install -D parcel”, “package-lock.json” file & “node_modules” folder are added in our project, and “Parcel” is added as devDependencies in package.json file as shown below.

```
"devDependencies": {
    "parcel": "^2.8.2"
  }
```

💡 **Difference Between Caret (^) & Tilde (~)**

NPM versions are written using three dots separated numbers the first number from the left shows the major release and the second number from the left shows the minor release and the third number shows the patch release of the package.

Syntax: The syntax of the npm version looks like the following.

```
Major.Minor.Patch
```

Tilde(~) notation	Caret(^) notation
Used for Approximately equivalent to version.	Used for Compatible with version.
It will update you to all future patch versions, without incrementing the minor version. ~1.2.3 will use releases from 1.2.3 to <1.3.	It will update you to all future minor/patch versions, without incrementing the major version. ^2.3.4 will use releases from 2.3.4 to <3.0.0
It gives you bug fix releases.	It gives you backwards-compatible new functionality as well.
It will update in decimals.	It will update to its latest version in numbers.
Not a default notation used by NPM.	Used by NPM as default notation.
Example: ~1.0.2	Example: ^1.0.2

💡 “package-lock.json” is very important file. It locks the version. We never have to add it in “gitignore”.

💡 “node_modules” folder is kind of database to npm. Don’t add “node_modules” folder on github. Add it in “gitignore” file. It is very heavy in size & our “package-lock.json” has sufficient information, hence we never add “node_modules” on git. “package-lock.json” helps us to generate “node_modules” folder on server.

💡 We are removing React CDN links from our project, because it is not the best way to use react. We will install React using npm, as shown below.

```
// we want React on local & server as well, hence we are not using "-D"
npm install react
```

After running “npm install react”, “React” is added as Dependency in package.json file as shown below.

```
// we havn't used "-D", hence it added as "dependencies" & not "devDependencies"
"dependencies": {
    "react": "^18.2.0"
  }
```

Now, we are installing “React-DOM” in our project

```
npm install react-dom

// we can also use "npm i react-dom" instead of "npm install react-dom"
// "npm i" is shortcut to "npm install"
```

Now, we will ignite our app, we will give entry point as index.html

```
// npx means execute using npm
npx parcel index.html


// After this import react & react-dom in App.js file
import React from "react";
import ReactDOM from "react-dom/client";
```

💡 when we run “npx parcel index.html”, “dist” creates faster development build of our project and serves it on the server.

Now, we need to edit <script> tag in index.html

```
// Change from
<script src="App.js"></script>

// Change to
<script type="module" src="App.js"></script>
```

💡 When we make any changes in any files, we don’t need to refresh our page in browser, it will automatically show changes without any need to refresh page. This concept is called as “**Hot Module Replacement (HMR)**” and it is part of “Parcel”. Parcel do this using “File Watcher Algorithm” which keeps track of changes in files.

```
// we are giving entry point as "indrx.html" in "npx parcel index.html", hence
// Remove below line from "package.json"

"main": "App.js",

// we can make production build by using below command
npx parcel build index.html
```

💡 “**Browserlist**” is used to make our app compatible with browsers. We can add it in “package.json” file, as shown below:

```
// Example
"browserslist": [
    "last 2 Chrome versions"
  ]
```