Understand ReactJS library and directory

Topics covered:

- What is directory and Library?
- What is File Structure?
- What are modules in React?
  - ❖ Node module
  - ❖ exporting and importing modules

1. **Directories and library:**
   - Directories are the collection of files, subdirectories or both.
     Represented like:

     ```
     PS C:\Users\hp\Desktop\newproject>
     ```

     To change the directories we use commands:
     cd folder_name

     ```
     PS C:\Users\hp\Desktop\newproject> cd .\firstproject\
     PS C:\Users\hp\Desktop\newproject\firstproject>
     ```

     To move back to the previous directory:
     cd..

     ```
     PS C:\Users\hp\Desktop\newproject\firstproject> cd..
     PS C:\Users\hp\Desktop\newproject>
     ```

     To add a new directory:
     mkdir directory_name

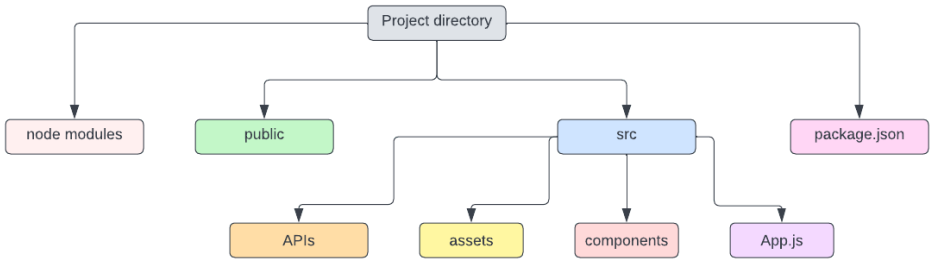
     ```
     PS C:\Users\hp\Desktop\newproject> mkdir directory1
     PS C:\Users\hp\Desktop\newproject> cd .\directory1\
     PS C:\Users\hp\Desktop\newproject\directory1>
     ```

     (note: Whenever you run a react app you must be in the right directory before running the command “npm start”).

   - Library is a combination of multiple directories and has multiple directories.

2. **File structure:**
   When you look at the project structure, you'll notice a /public and /src directory, as well as the standard node modules,.gitignore, README.md, and package.json files.
   Our important file in /public is index.html, which is quite identical to the static index.html file we created before — just a root div.
   No libraries or scripts are being loaded at this time. All of our React code will be stored in the /src directory.
   Find the following line in /src/App.js to show how the environment automatically compiles and updates your React code:

   - **Classification based on features or routes**
     One common method for organizing projects is to group CSS, JS, and tests into folders organized by feature or route. For example:



   - **File type identification**
     Another frequent method for organizing tasks is to put related files together.
   - **Avoid excessive nesting**
     There are numerous drawbacks to deep directory nesting in JavaScript programs. When the files are moved, it becomes more difficult to write relative imports between them or to update those imports.

3. **Modules:**
   - JavaScript modules allow you to split your code into individual files.
   - This simplifies code maintenance.
   - The import and export statements are used by ES Modules.

❖ **Node modules:**

➔ **Package:** A file or directory that is represented by a package.json file is referred to as a package. A package cannot be published to the npm registry without a package.json file.

➔ **Node module:** Node module is the online directory that contains the various already registered open-source packages.
NPM modules consume the various functions as a third-party package when installed into an app using the NPM command npm install.

❖ **Exporting:**

➔ Type of Export:
- Default export
- Named export

➔ **Default export**: Every module is said to have a maximum of one default export, as we already know. We must adhere to the syntax outlined below in order to export the default export from a file:

```
export default App;
```

➔ **Named export:** A named parameter can be exported using the following syntax. Each module may have several named parameters.

```
export { Classes };
```

❖ **Importing:**

➔ Type of Import:
- Import default export
- Import named export

➔ **Import default export:** Each module is said to have a single default export at most. It is possible to import the default export from a file by using merely the address and the term import before it, or by naming the import and using the syntax shown below:

import NAME from ADDRESS

Example;

```
import React from "react";
```

➔ **Import named export:** Every module has a number of named arguments, and we should use the following syntax to import one of them:

import { NAME } from ADDRESS

Example;

```
import { Classes } from "./components/Classes";
```