Vue Js (Part 1)

Setup a Vue Js Project

- Open command prompt or terminal
- Run command node -v to check node version or node is installed or not
    - If not installed [click here]() to download the node.
- To set up vue js project run below commands
    - npm install -g @vue/cli (install vue cli in your system)
    - vue create <your project name> (create vue project)
    - cd <your project name> (changing directory to your project)
    - npm run serve (run your project)

Project Structure

```
frontend/
├── node_modules
├── public/
│   ├── favicon.ico
│   └── index.html
├── src/
│   ├── assets/
│   ├── components
│   ├── App.vue
│   └── main.js
├── .gitignore
├── babel.config.js
├── jsconfig.js
├── package-lock.json
├── package.json
├── vueconfig.json
└── README.md
```

`frontend/`

This is the root directory of your Vue.js project.

`node_modules/`

This directory contains all the npm packages installed for your project. It is automatically generated by npm and should not be modified manually.

Vue Js (Part 1)

`public/`

This directory contains static assets that are not processed by Webpack. These files are copied directly to the final build output.

- **`favicon.ico`**: The favicon for your project. This small icon appears in the browser tab.
- **`index.html`**: The main HTML file for your project. Vue CLI injects the necessary build assets into this file.

`src/`

This directory contains your application's source code.

- **`assets/`**: This directory is for static assets like images, fonts, etc., that you want Webpack to process.
- **`components/`**: This directory is for your Vue components. Each component is typically a single file component with a `.vue` extension.
- **`App.vue`**: The root component of your application. This component is the top-level component that all other components are nested within.
- **`main.js`**: The entry point of your application. This is where Vue is initialized and where the root component is mounted to the DOM.

`.gitignore`

This file specifies which files and directories should be ignored by Git version control. It typically includes `node_modules/` and other build-related files to avoid committing them to the repository.

`babel.config.js`

This is the Babel configuration file. Babel is used to transpile modern JavaScript to ensure compatibility with older browsers. The configuration file defines the presets and plugins Babel should use.

`jsconfig.js`

This file is used to configure JavaScript and TypeScript settings in your project, especially for editors like Visual Studio Code. It helps in path mapping and code intelligence features like autocomplete and go-to-definition.

`package-lock.json`

This file is automatically generated by npm and includes the exact versions of installed packages. It ensures that anyone who installs dependencies for the project gets the same versions, thus maintaining consistency.

`package.json`

This file contains metadata about your project, including the project name, version, dependencies, scripts, and other configuration settings. It is essential for managing the project and its dependencies.

`vueconfig.json`

This is a Vue CLI configuration file (typically named `vue.config.js`). It allows you to modify the default configuration of the Vue CLI, such as setting up aliases, modifying webpack configuration, or configuring development server options.

`README.md`

This file provides information about your project. It is a good place to describe the project, how to set it up, and how to use it. This is particularly useful for other developers or contributors to understand the project better.