Part I
Create React App

1 Install nodejs

These instructions are meant for Linux systems. First, we need a tool called curl, if you don't have it, you can install it typing the following command in a terminal:

```
sudo apt-get install curl
```

You are going to be prompt to enter your user password to grant sudo access. After the installation is finished, now let's install Node by typing the following command:

```
curl -sL https://deb.nodesource.com/setup_6.x | sudo -E bash -
```

I'm going to install Node.js v6 LTS Version. If you want, you can install the newest version (v8) by replacing setup_6.x for setup_8.x. Either version will work fine.

The command you just type added the Node.js to your repository list, now, let's install Node.js by typing:

```
sudo apt-get install -y nodejs
```

If everything is alright, you can check that Node.js and NPM are installed (NPM is included with Node.js) by typing

```
node -v
```

and

```
npm -v,
```

and you should see something like this:

```
$ node -v
v6.11.1
npm -v
3.10.10
```

Also, let's install build-essential package, which is needed for Node to work properly:

```
sudo apt-get install -y build-essential
```

2 Create React App

Create React apps with no build configuration.

- Creating an App – How to create a new app.
- User Guide – How to develop apps bootstrapped with Create React App.

Now, let's install React by typing:

```
sudo npm install -g create-react-app
```

2.1 Quick Overview

```
npx create-react-app my-app
cd my-app
npm start
```

(npx comes with npm 5.2+ and higher, see instructions for older npm versions)

Then open http://localhost:3000/ to see your app. When you're ready to deploy to production, create a minified bundle with npm run build.

2.2 Get Started Immediately

You don't need to install or configure tools like Webpack or Babel. They are preconfigured and hidden so that you can focus on the code.

Just create a project, and you're good to go.

You'll need to have Node >= 6 on your local development machine (but it's not required on the server). You can use nvm (macOS/Linux) or nvm-windows to easily switch Node versions between different projects.

To create a new app, run a single command:

```
npx create-react-app my-app
```

(npx comes with npm 5.2+ and higher, see instructions for older npm versions)

It will create a directory called my-app inside the current folder. Inside that directory, it will generate the initial project structure and install the transitive dependencies:

```
‘‘‘
my-app
├── README.md
├── node_modules
├── package.json
├── .gitignore
├── public
│   └── favicon.ico
│   └── index.html
```

```
    └── manifest.json
└── src
    └── App.css
    └── App.js
    └── App.test.js
    └── index.css
    └── index.js
    └── logo.svg
    └── registerServiceWorker.js
```

In the left side of the screen you can see the folder structure of our application.

- **node_modules**: contains every Javascript library and configurations.
- **public**: here are the files that are going to be public to the server and the people. Like the index.html and the favicon. We'll talk more about this later.
- **src**: here are all of our **React Components**.

No configuration or complicated folder structures, just the files you need to build your app. Once the installation is done, you can open your project folder:

```
cd my-app
```

Inside the newly created project, you can run some built-in commands. This command will start a local server and will open our application in the browser, just like this:

```
npm start
```

Runs the app in development mode.

Open http://localhost:3000 to view it in the browser.

The page will automatically reload if you make changes to the code.
You will see the build errors and lint warnings in the console.

```
npm test
```

Runs the test watcher in an interactive mode. By default, runs tests related to files changed since the last commit.

Read more about testing.

```
npm run build
```

Builds the app for production to the build folder. It correctly bundles React in production mode and optimizes the build for the best performance.

The build is minified and the filenames include the hashes.
By default, it also includes a service worker so that your app loads from local cache on future visits.

Your app is ready to be deployed.

3 User Guide

The User Guide includes information on different topics, such as:

Updating to New Releases
Folder Structure
Available Scripts
Supported Browsers
Supported Language
Features and Polyfills
Syntax Highlighting in the Editor
Displaying Lint Output in the Editor
Formatting Code Automatically
Debugging in the Editor
Changing the Page <title>
Installing a Dependency
Importing a Component
Code Splitting
Adding a Stylesheet
Post-Processing CSS
Adding a CSS Preprocessor (Sass, Less etc.)
Adding Images, Fonts, and Files
Using the public Folder
Using Global Variables
Adding Bootstrap
Adding Flow
Adding a Router
Adding Custom Environment Variables
Can I Use Decorators?
Fetching Data with AJAX Requests
Integrating with an API Backend
Proxying API Requests in Development
Using HTTPS in Development
Generating Dynamic <meta> Tags on the Server
Pre-Rendering into Static HTML Files
Running Tests
Debugging Tests