**Final Project: Pokémon TeamBuilder**

**A cartoon character with a black background

Description automatically generated**

**Course: SE/COMS 319**

**Ponciano Ramirez - pram1347@iastate.edu**

**Bo Oo - bhoo@iastate.edu**

**Instructor: Ali Jannesari**

**Date: May 07, 2024**

**Index**

1. Project Description
2. Software Functionality Diagram
3. Project Structure/Architecture
4. Installation Manual
5. Code file:
   1. Frontend
      1. index.js
         1. frontend.js
            1. loginPage.js
            2. signUp.js
            3. pokemonStats.js
            4. teamList.js
            5. pokemonInfoPage.js
            6. setting.js
   2. Backend
      1. index.js

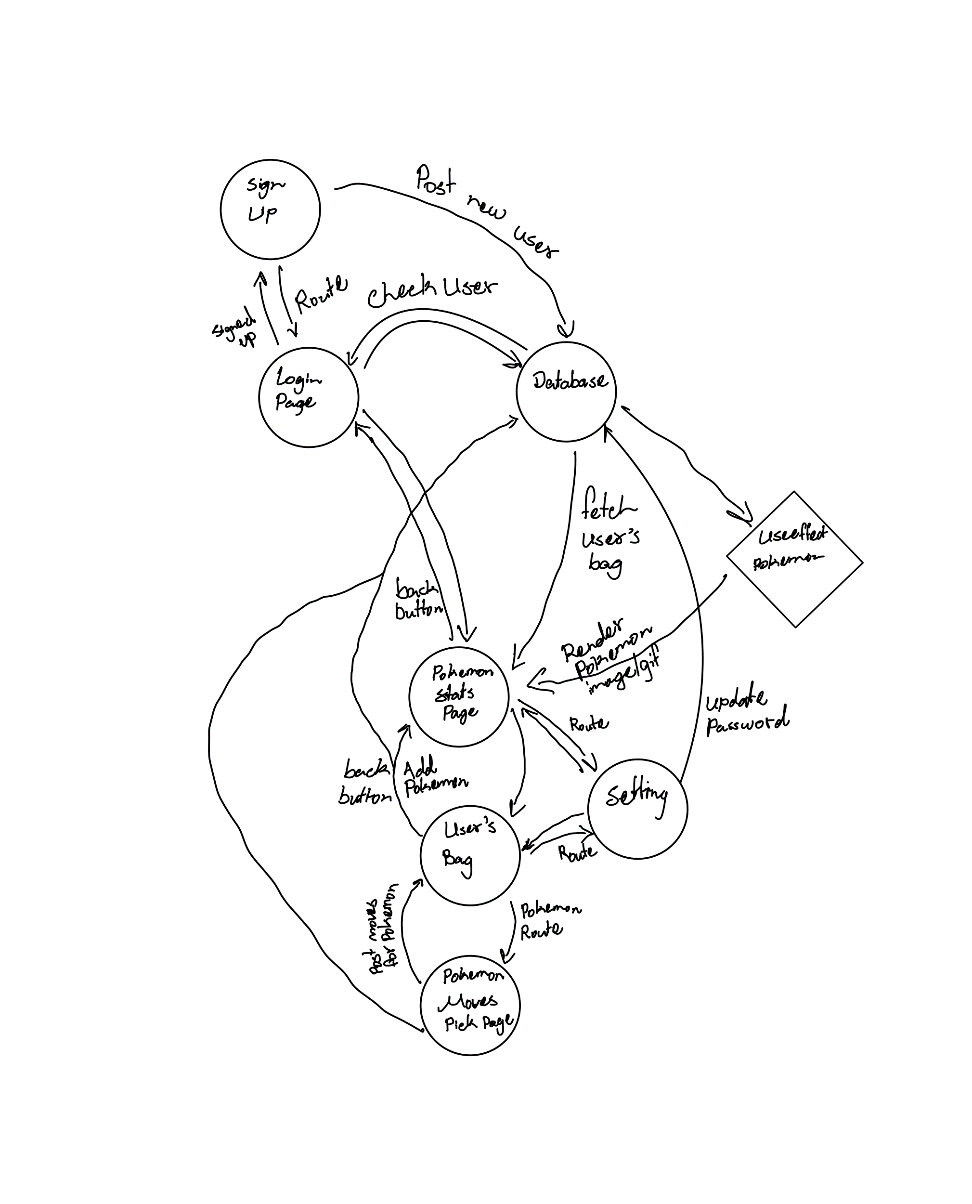
**Project Description**

The purpose of this project is to create Pokémon TeamBuilder which allows the user to create, read, update, and delete a Pokémon of their choice to their team. This project is built using Node.JS, Express, MongoDB, Mongoose, React, Tailwind CSS.

Users can create their own account to keep track of their own Pokémon with as well as being able to personalize their team with various variety of Pokémon’s. It is a user-friendly website with smooth UI that is easy to navigate and easy to keep track of.

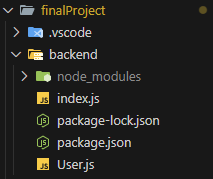
This project is created through our passion and nostalgic feelings for Pokémon. We wanted to allow users to experience being a Pokémon Trainer that can customize and build the ultimate Pokémon team for adventure.

Project Structure/Architecture



A blue and orange rectangular boxes with black text

Description automatically generated



A screenshot of a computer

Description automatically generated

Installation Manual

For the installation, you will need MongoDB, Node.JS, Express, React, Tailwind CSS.

**Database**:

For MongoDB, the recommendation is to download from this website: <https://www.mongodb.com/try/download/shell>

Where you can choose the application depending on OS of your choice.

Afterward, create a Database named FinalProject and a collection named users.

Before you get started on either the frontend or backend, create a directory named after your choice and write **npm init** inside your directory to initialize Node.JS.

**Frontend**:

And use **npx create-react-app frontend** to create a React app. After the React app is created, use **npm install -D tailwindcss** to install Tailwind CSS, and then use **npx tailwindcss init -p** to initialize it. Afterward add,

**@tailwind base;**

**@tailwind components;**

**@tailwind utilities;**

to your index.css. And you are set to work on your frontend. After setting up your own React, use npm start to run the React application. The application should open up in <http://localhost:3000/> to view your website.

**Backend**:

And for the backend, create a directory manually named backend and go inside the directory, in here use,

**npm install body-parser cors express mongodb**

**npm install -D nodemon**

to install all the dependencies needed for the backend. Afterward you are ready to start writing your code. To run the backend application, use **nodemon “app name”** to have it up and running.

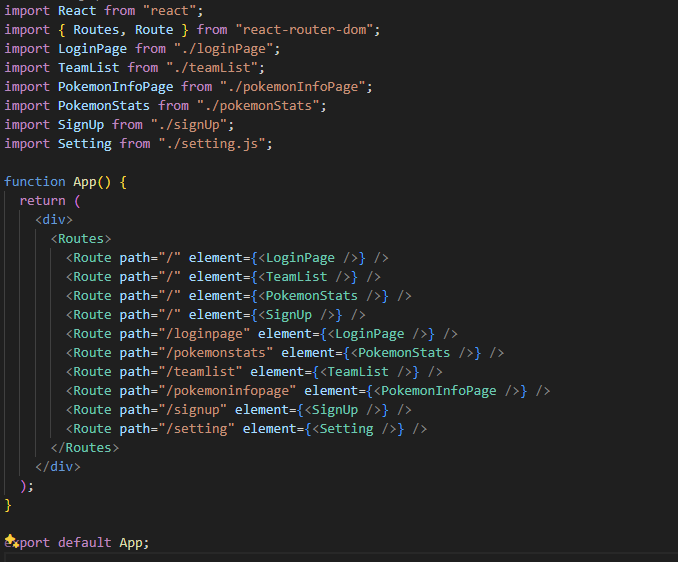
Code files:

Index.js

A screen shot of a computer program

Description automatically generated

frontend.js



pokemonStats.js