Simple Express Routes Project

This is a simple Express.js application that demonstrates how to set up basic routes and middleware in an Express app. The app includes various types of routes: **GET** and **POST** requests, and showcases path parameters.

Table of Contents

Installation

To get started with this project, follow these steps:

Clone the repository:

bash
Copy

```
git clone https://github.com/your-username/simple-express-routes.git
cd simple-express-routes
```

1. Install the dependencies:

   bash
   Copy
   ```
   npm install
   ```
2. Run the server:

   bash
   Copy
   ```
   npm start
   ```

3. This will start the Express server on port 3000.

Usage

Once the server is running, you can interact with the app by making requests to the routes listed below. You can use tools like **Postman** or **curl** to test the routes.

Routes

GET Routes

GET `/`: Sends a simple "Hello, Express!" message.

javascript
Copy
```
app.get("/", (req, res) => {
  res.send("Hello, Express!");
});
```

GET `/express`: Sends a message stating that creating routes with Express is simple.

javascript
Copy
```
app.get("/express", (req, res) => {
  res.send("Creating routes with Express is simple!");
});
```

GET `/user`: Sends a message stating a GET request for the user has been received.

javascript
Copy

```javascript
app.get("/user", (req, res) => {
  res.send("Received a GET request for user!");
});
```

GET `/user/:userID`: Sends a message with the user ID that was provided in the URL.

javascript
Copy

```javascript
app.get("/user/:userID", (req, res) => {
  res.send(`Navigated to the user page for: ${req.params.userID}.`);
});
```

GET `/user/:userID/profile`: Sends a message with the user profile for the provided user ID.

javascript
Copy

```javascript
app.get("/user/:userID/profile", (req, res) => {
  res.send(`Navigated to the user profile page for:
${req.params.userID}.`);
});
```

GET `/user/:userID/profile/:data`: Sends a message with both the user ID and the data parameter in the URL.

javascript
Copy

```javascript
app.get("/user/:userID/profile/:data", (req, res) => {
  res.send(`Navigated to the user profile page for:
${req.params.userID}, with the data: ${req.params.data}.`);
});
```

POST Routes

POST `/user`: Receives a POST request for creating or modifying a user.

javascript
Copy

```
app.post("/user", (req, res) => {
  res.send("Received a POST request for user!");
});
```

Middleware

In this project, a simple logging middleware is used to log the incoming requests before any routes are handled. This is done using the following code:

javascript
Copy

```
const logReg = function (req, res, next) {
  console.log("request Received")
  next();
}
app.use(logReg); // Middleware applied globally, before any routes
```

Port and Server

The server is set to listen on port 3000. To run the server, the following code is used:

javascript
Copy

```
app.listen(port, () => {
  console.log(`Jigglypuff, always listening on port: ${port}`);
});
```

This will output a confirmation message to the console once the server starts successfully.

Contributing

We welcome contributions! If you'd like to contribute to the project, please follow these steps:

1. Fork the repository.
2. Create a new branch (`git checkout -b feature-branch`).
3. Commit your changes (`git commit -am 'Add new feature'`).
4. Push to the branch (`git push origin feature-branch`).
5. Create a pull request.