

Assignment 3B

Name: Shubham Pitale

RollNo: 33352

Batch: M11

Index: server.js

```
const express = require("express");
const bodyParser = require("body-parser");

const app = express();
app.use(bodyParser.urlencoded({ extended: true }));

app.use(bodyParser.json());

const UserRoute = require("./routes/User");
app.use("/user", UserRoute);

const dbConfig = require("./config/database.config.js");
const mongoose = require("mongoose");

mongoose.Promise = global.Promise;
mongoose
  .connect(dbConfig.url, {
    useNewUrlParser: true,
  })
  .then(() => {
    console.log("Database Connected Successfully!!");
  })
  .catch((err) => {
    console.log("Could not connect to the database", err);
    process.exit();
  });

app.get("/", (req, res) => {
  res.json({ message: "Hello Crud Node Express" });
});

app.listen(3000, () => {
  console.log("Server is listening on port 3000");
});
```

Routes: User.js

```
const express = require("express");
const UserController = require("../controllers/User");
const router = express.Router();

router.get("/", UserController.findAll);
```

```
router.get("/:id", UserController.findOne);
router.post("/", UserController.create);
router.patch("/:id", UserController.update);
router.delete("/:id", UserController.destroy);
```

```
module.exports = router;
```

Model: User.js

```
var mongoose = require("mongoose");
var schema = new mongoose.Schema({
  email: { type: String, required: true, unique: true },
  firstName: { type: String, default: "" },
  lastName: { type: String, default: "" },
  phone: String,
});
var user = new mongoose.model("User", schema);
module.exports = user;
```

Output:

Create User:

The screenshot shows the Postman application interface. The top bar indicates the current workspace is "http://localhost:3000/user/ - My Workspace". The main area displays a POST request to "http://localhost:3000/user/". The request body is in the "Body" tab, showing a JSON object with the following fields: "email" (abc@gmail.com), "firstName" (ABC), "lastName" (XYZ), "phone" (85586966), "_id" (6d34eb2184665bc514a6dbf4), and "_v" (0). The response is also in the "Body" tab, showing a JSON object with "message": "User created successfully!!" and the same user object. The status bar at the bottom indicates "Status: 200 OK", "Time: 6 ms", and "Size: 405 B".

Key	Value	Description
email	abc@gmail.com	
firstName	ABC	
lastName	XYZ	
phone	85586966	
Key	Value	Description

```
1
2  "message": "User created successfully!!",
3  "user": {
4    "email": "abc@gmail.com",
5    "firstName": "ABC",
6    "lastName": "XYZ",
7    "phone": "85586966",
8    "_id": "6d34eb2184665bc514a6dbf4",
9    "_v": 0
10
11
```

Display Single User:

A screenshot of the Postman application interface. The top bar shows the URL `http://localhost:3000/user/6434eb2184665bc514a6dbf4 - My Workspace`. The main area displays a GET request to `http://localhost:3000/user/6434eb2184665bc514a6dbf4`. The response is shown in the 'Body' tab, displaying a JSON object with user details: `{ "_id": "6434eb2184665bc514a6dbf4", "email": "abc@gmail.com", "firstName": "ABC", "lastName": "XYZ", "phone": "85586966", "updatedAt": 0 }`. The status is 200 OK, Time: 8 ms, Size: 356 B.

Key	Value	Description
Key	Value	Description

```
1 {
2   "_id": "6434eb2184665bc514a6dbf4",
3   "email": "abc@gmail.com",
4   "firstName": "ABC",
5   "lastName": "XYZ",
6   "phone": "85586966",
7   "updatedAt": 0
8 }
```

Update User:

A screenshot of the Postman application interface. The top bar shows the URL `http://localhost:3000/user/6434eb2184665bc514a6dbf4 - My Workspace`. The main area displays a PATCH request to `http://localhost:3000/user/6434eb2184665bc514a6dbf4`. The request body is set to `x-www-form-urlencoded` and contains a single key-value pair: `phone=88955`. The response is shown in the 'Body' tab, displaying a JSON object: `{ "message": "User updated successfully." }`. The status is 200 OK, Time: 10 ms, Size: 275 B.

Key	Value	Description
<input checked="" type="checkbox"/> phone	88955	
Key	Value	Description

```
1 {
2   "message": "User updated successfully."
3 }
```

Delete User:

The screenshot shows the Postman application interface. At the top, the address bar displays the URL `http://localhost:3000/user/6434eb2184665bc514a6dbf4 - My Workspace`. The main workspace shows a **DELETE** request to the same URL. The request is configured with the following details:

- Method:** DELETE
- URL:** `http://localhost:3000/user/6434eb2184665bc514a6dbf4`
- Environment:** No Environment
- Params:** Params, Authorization, Headers (6), Body, Pre-request Script, Tests, Settings
- Query Params:** A table with columns Key, Value, and Description.

The response is displayed in the **Body** tab, showing a JSON object:

```
1 {
2   "message": "User deleted successfully!"
3 }
```

The status bar at the bottom indicates the response status: **Status: 200 OK**, Time: 5 ms, Size: 275 B. The bottom of the interface includes a **Find and Replace** bar, a **Console** tab, and a **Runner** tab.