User CRUD Operations Report

Muhmmad Yaqoob
2021-CS-118

Introduction

This report documents the implementation and testing of User CRUD operations, including Create, Update, Delete, View, and a Login operation. The system stores user information, including username, password, first name, last name, date of birth (DOB), and role (Admin, Super User).

Directory Structure

The project's directory structure is as follows:

```
LAB 7 ASSIGNMENT/
     controllers/
        authenticationControllers/
           userController.js
     models/
        authenticationModel/
           userModel.js
     routes/
        authenticationRoutes/
           userRoutes.js
     utils/
        db.js
     package.json
     server.js
```

User API Operations

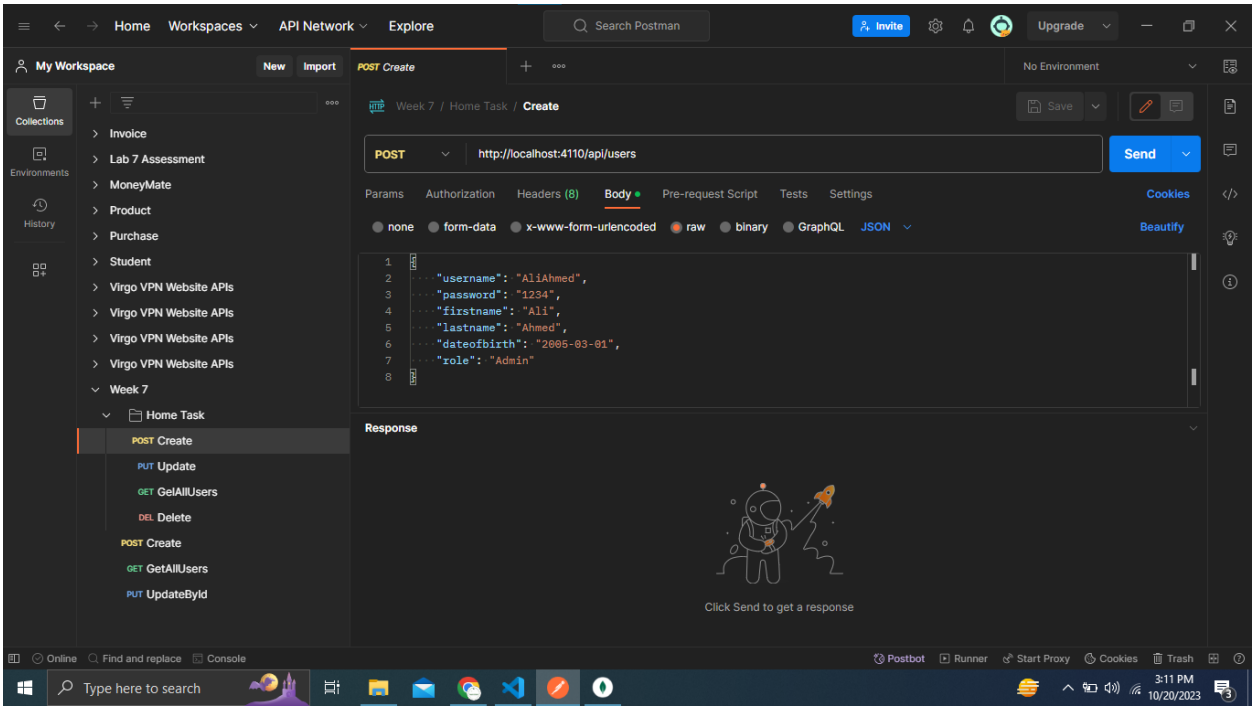

Figure 1: Screenshot of Create User API Call

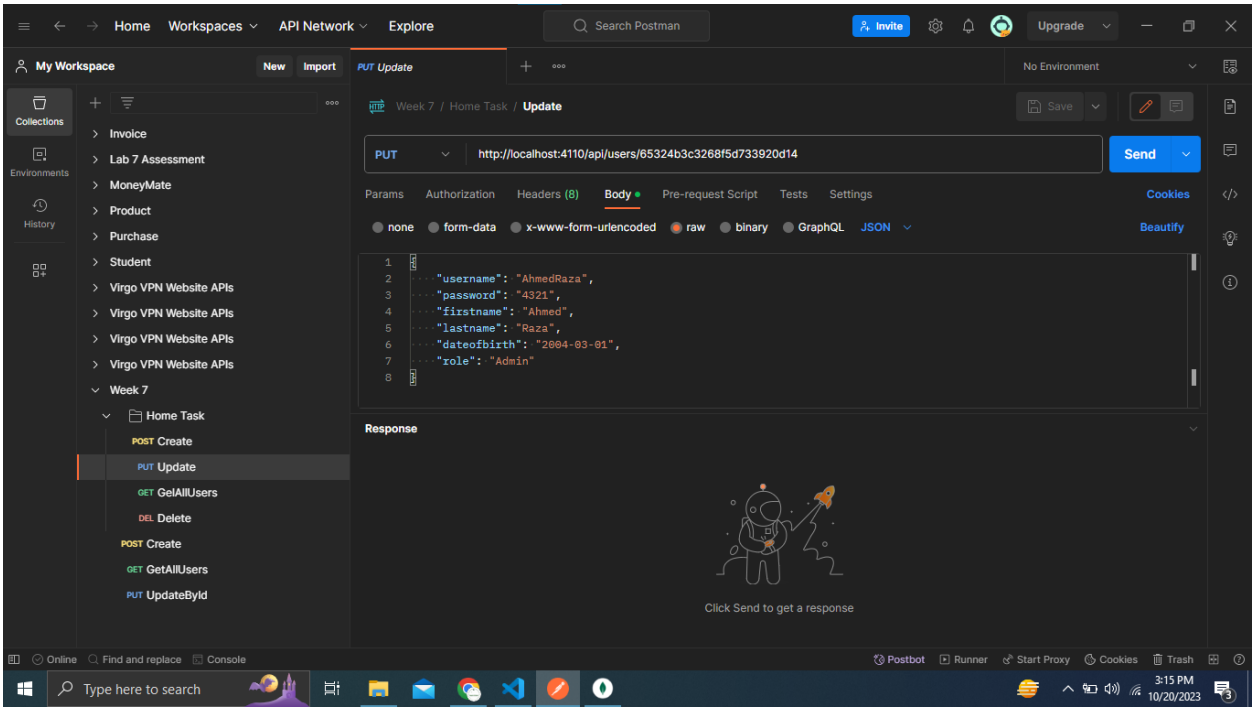

Figure 2: Screenshot of Update User API Call

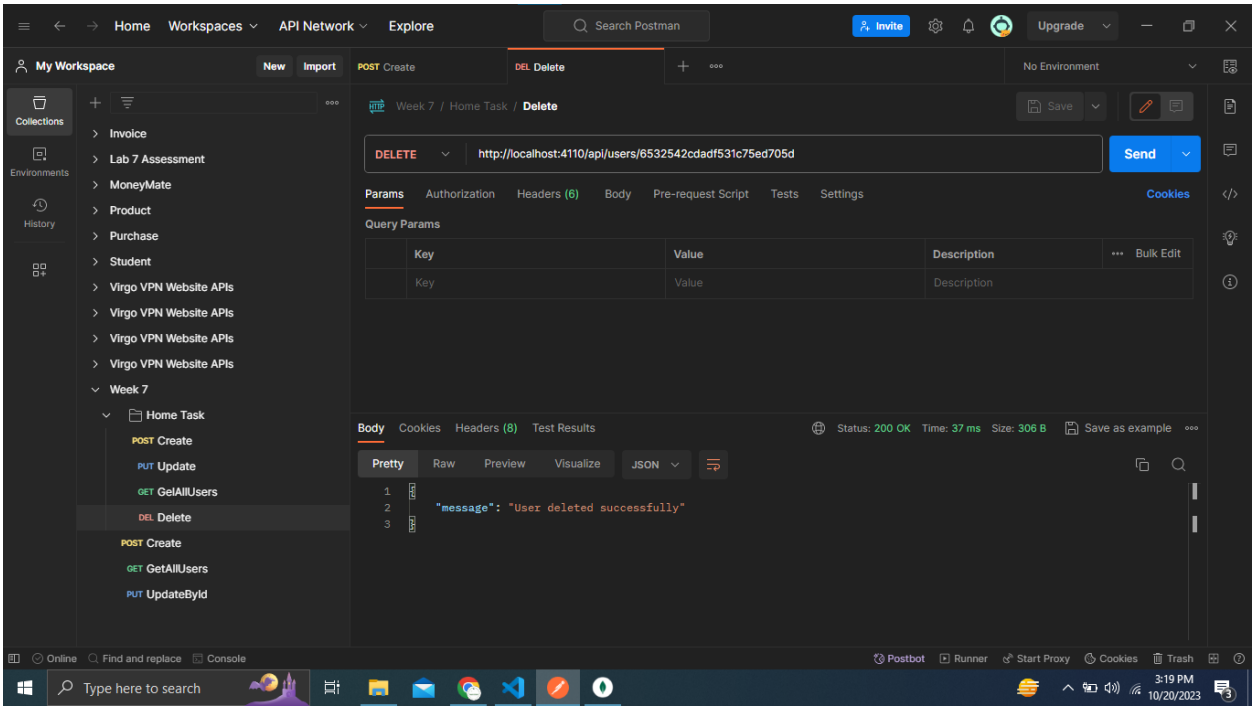

Figure 3: Screenshot of Delete User API Call

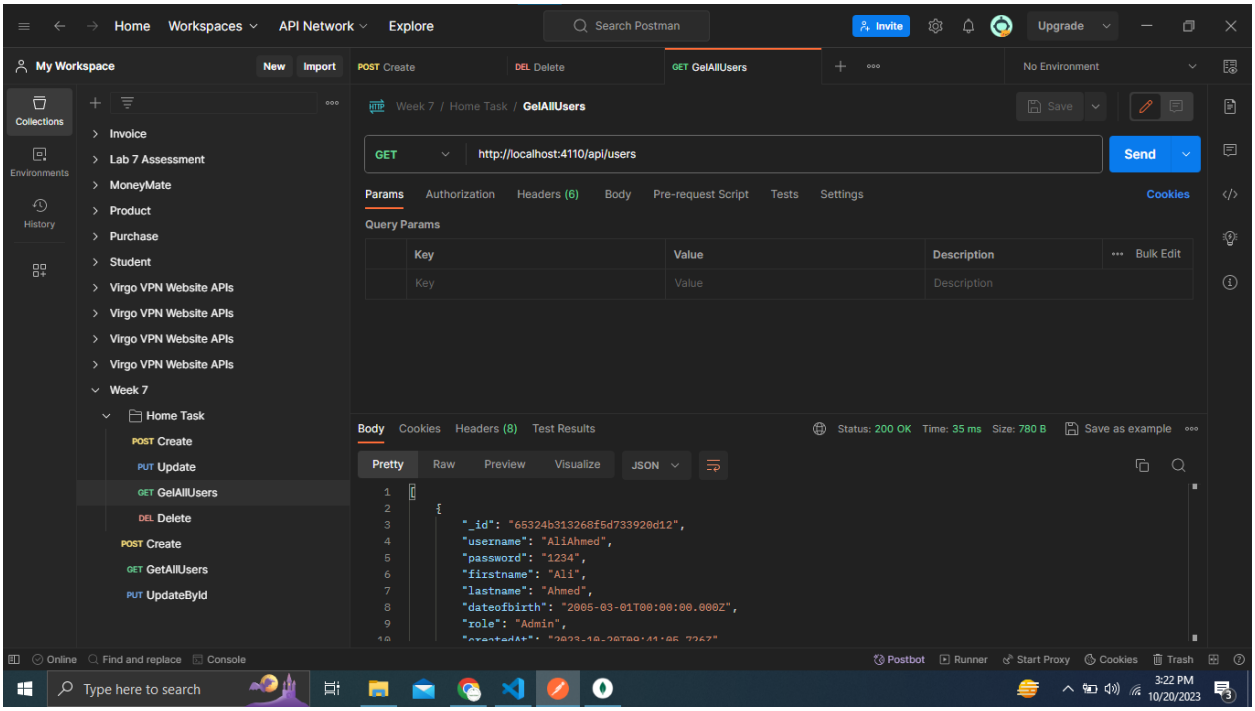

Figure 4: Screenshot of View All Users API Call

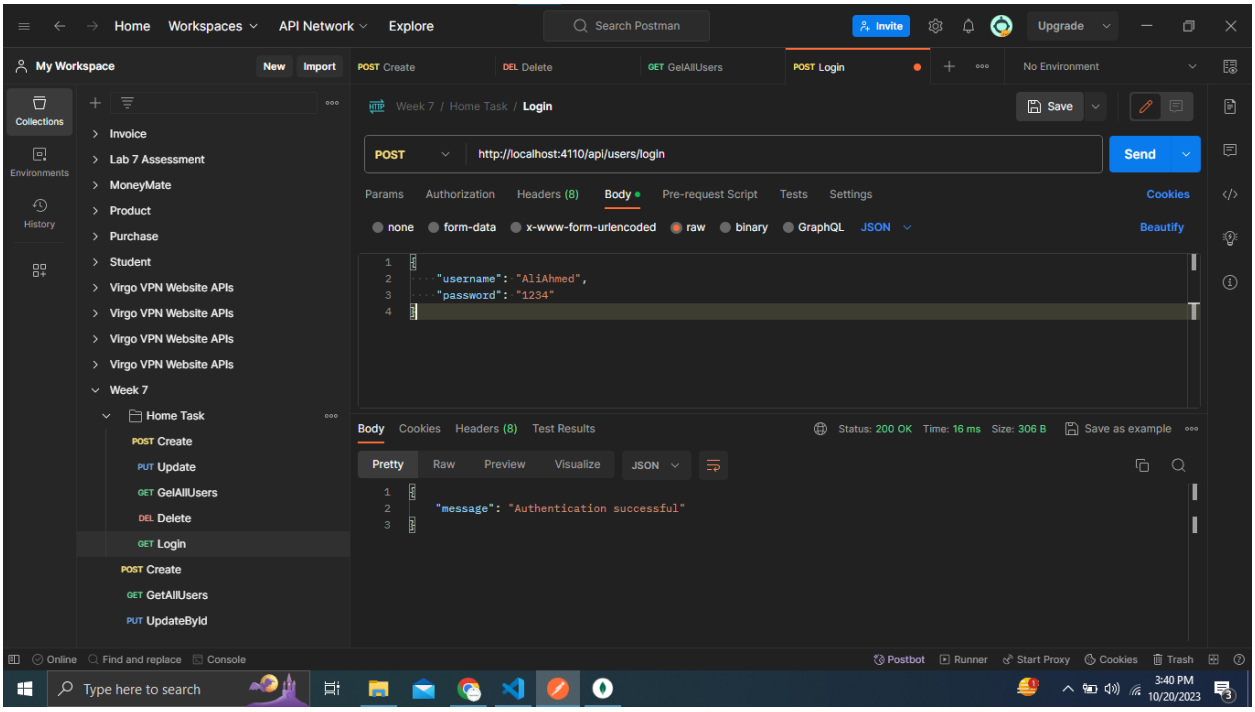

Figure 5: Screenshot of Login API Call

User Model

```
const mongoose = require("mongoose");

const userSchema = mongoose.Schema(
  {
    username: String,
    password: String,
    firstname: String,
    lastname: String,
    dateofbirth: Date,
    role: String,
  },
  { timestamps: true }
);

module.exports = mongoose.model("Users", userSchema);
```

User Controller

```
const User = require("../../models/authenticationModel/userModel");

// Create a User
async function createUser(req, res) {
  try {
    const newUser = await User.create(req.body);
    res.status(201).json({
      message: "User created successfully",
      user: newUser,
    });
  } catch (error) {
    res.status(500).json({ error: error.message });
  }
}

// Get all Users
async function getAllUsers(req, res) {
  try {
    const users = await User.find();
    res.json(users);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
}

// Update a User by ID
async function updateUser(req, res) {
  try {
    const { id } = req.params;
    const updatedUser = await User.findByIdAndUpdate(id, req.body, {
      new: true,
    });
```

```
    if (!updatedUser) {
      return res.status(404).json({ error: "User not found" });
    }

    res.json({
      message: "User information updated successfully",
      user: updatedUser,
    });
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
}

// Delete a User by ID
async function deleteUser(req, res) {
  try {
    const { id } = req.params;
    await User.findByIdAndRemove(id);
    res.json({ message: "User deleted successfully" });
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
}

// Login User
const loginUser = async (req, res) => {
  try {
    const { username, password } = req.body;
    const user = await User.findOne({ username });

    if (!user) {
      return res
        .status(401)
        .json({ message: "Authentication failed. User not found." });
    }

    if (user.password !== password) {
      return res
        .status(401)
        .json({ message: "Authentication failed. Incorrect password." });
    }

    res.status(200).json({ message: "Authentication successful" });
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
};

module.exports = {
  createUser,
  getAllUsers,
  updateUser,
  deleteUser,
```

```
  loginUser,
};
```

User Routes

```
const express = require("express");
const router = express.Router();
const userController = require("../../controllers/authenticationControllers/
    userController");

// create a product
router.post("/users", userController.createUser);

// get all users
router.get("/users", userController.getAllUsers);

// update a user by id
router.put("/users/:id", userController.updateUser);

// delete a user by id
router.delete("/users/:id", userController.deleteUser);

// Login User
router.post("/users/login", userController.loginUser);
module.exports = router;
```

Sever.js

```
const express = require("express");
const app = express();
const port = 4110;
const cors = require("cors");
const bodyparcer = require("body-parser");
require("./utils/db");
const userRouter = require("./routes/authenticationRoutes/userRoutes");

// get fun has two arguments first is end point then call back (no need to
    call. Called automatically)function
app.get("/", (req, res) => {
  res.send("Hello, World!");
});

//Middlewares
app.use(bodyparcer.json());
app.use(cors());

// Product API
app.use("/api", userRouter);

app.get("/welcome", (req, res) => {
  res.send("<h1>Welcome Ali Ahmed</h1>");
```

```
});

app.listen(port, () => {
  console.log(`Server is listening on port ${port}`);
});
// node behavious = asyncrouns
```

db.js

```
const mongoose = require("mongoose");
mongoose.set("strictQuery", true);

mongoose.connect("mongodb://127.0.0.1:27017/Week7HomeTask", {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});

const db = mongoose.connection;
db.on("error", (err) => {
  console.log("Failed to connect with db");
});
db.once("open", () => {
  console.log("Connected with db");
});
```