**NodeJS Capstone Project**

Name : Shreyas Nadig S G

USN : 4PM21IS048

DTS Batch-4

**db.js**

const mysql = require("mysql2");

require("dotenv").config();

const db = mysql.createConnection({

    host: process.env.DB\_HOST,

    user: process.env.DB\_USER,

    password: process.env.DB\_PASSWORD,

    database: process.env.DB\_NAME,

});

db.connect((err) => {

    if (err) throw err;

    console.log(" Connected to mysql db");

});

module.exports = db;

**tradeController.js**

const Trade = require("../models/Trade");

const { v4: uuidv4 } = require("uuid");

exports.createTrade = (req, res) => {

    const { type, user\_id, symbol, shares, price, timestamp } = req.body;

    if (!['buy', 'sell'].includes(type)) {

        return res.status(400).json({ message: "Incorrect type provided" });

    }

    if (shares < 1 || shares > 100) {

        return res.status(400).json({ message: "Shares value out of range" });

    }

    const trade = { id: uuidv4(), type, user\_id, symbol, shares, price, timestamp };

    Trade.create(trade, (err) => {

        if (err) return res.status(500).json({ message: "Database error" });

        res.status(201).json(trade);

    });

};

exports.getAllTrades = (req, res) => {

    const filters = { type: req.query.type, user\_id: req.query.user\_id };

    Trade.getAll(filters, (err, results) => {

        if (err) return res.status(500).json({ message: "Database error" });

        res.status(200).json(results);

    });

};

exports.getTradeById = (req, res) => {

    Trade.getById(req.params.id, (err, results) => {

        if (err) return res.status(500).json({ message: "Database error" });

        if (results.length === 0) {

            return res.status(404).json({ message: "ID not found" });

        }

        res.status(200).json(results[0]);

    });

};

exports.methodNotAllowed = (req, res) => {

    res.status(405).json({ message: "Method not allowed" });

};

**Trade.js**

const db = require("../config/db");

const Trade = {

    create: (trade, callback) => {

        const sql = "INSERT INTO trades SET ?";

        db.query(sql, trade, callback);

    },

    getAll: (filters, callback) => {

        let sql = "SELECT \* FROM trades";

        const conditions = [];

        const values = [];

        if (filters.type) {

            conditions.push("type = ?");

            values.push(filters.type);

        }

        if (filters.user\_id) {

            conditions.push("user\_id = ?");

            values.push(filters.user\_id);

        }

      if (conditions.length) {

            sql += " WHERE " + conditions.join(" AND ");

        }

        db.query(sql, values, callback);

    },

    getById: (id, callback) => {

        const sql = "SELECT \* FROM trades WHERE id = ?";

        db.query(sql, [id], callback);

    }

};

module.exports = Trade;

**tradeRoutes.js**

const express = require("express");

const {

    createTrade,

    getAllTrades,

    getTradeById,

    methodNotAllowed

} = require("../controllers/tradeController");

const router = express.Router();

router.post("/trades", createTrade);

router.get("/trades", getAllTrades);

router.get("/trades/:id", getTradeById);

router.delete("/trades/:id", methodNotAllowed);

router.patch("/trades/:id", methodNotAllowed);

module.exports = router;

**Config.env**

PORT=5000

DB\_HOST=localhost

DB\_USER=root

DB\_PASSWORD=Snsg@1234.

DB\_NAME=stocks

JWT\_SECRET=3aa3d20131be7fa88d080e3def1769bbb9b24c4c124619fa4a9d53d109879ab4c044c8d707ed7364b9ffc2d35eb3820667a14f8cbed8ab6c544871bb678a7c5a

**Server.js**

const express = require("express");

const dotenv = require("dotenv");

const jwt = require("jsonwebtoken");

const tradeRoutes = require("./routes/tradeRoutes");

require("./config/db");

dotenv.config();

const app = express();

app.use(express.json());

const authMiddleware = (req, res, next) => {

    const token = req.headers.authorization?.split(" ")[1];

    if (!token) {

        return res.status(401).json({ message: "Unauthorized" });

    }

    try {

        const decoded = jwt.verify(token, process.env.JWT\_SECRET);

        req.user = decoded;

        next();

    } catch (error) {

        return res.status(403).json({ message: "Invalid token" });

    }

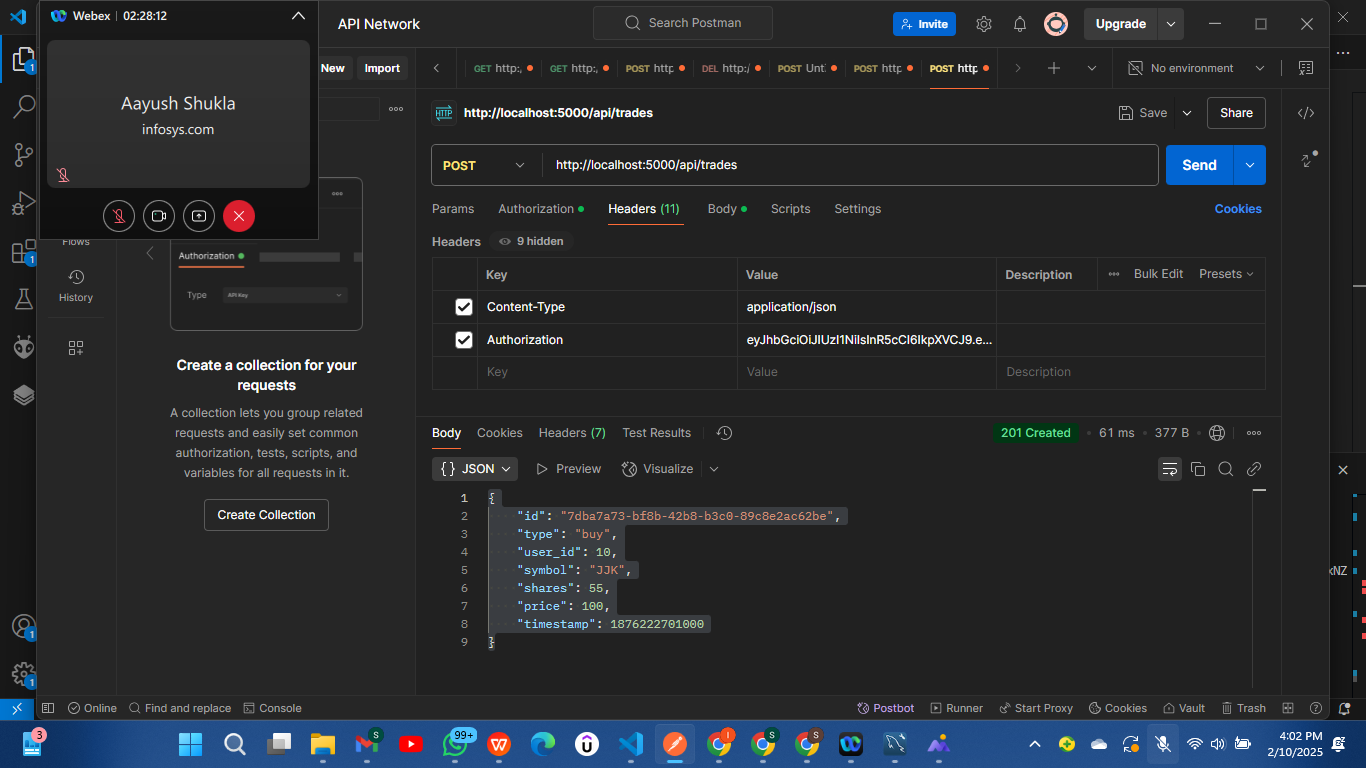
};

app.use("/api", authMiddleware, tradeRoutes);

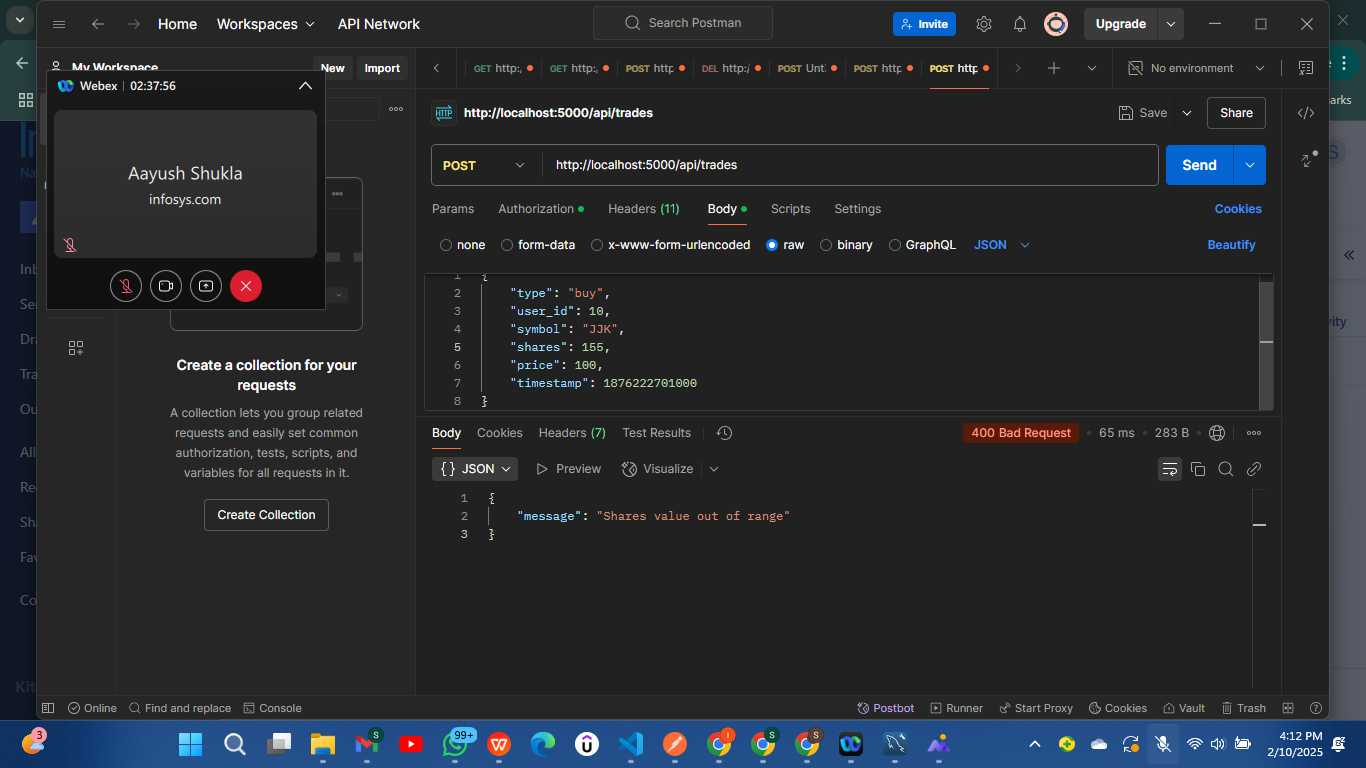
const PORT = process.env.PORT || 5000;

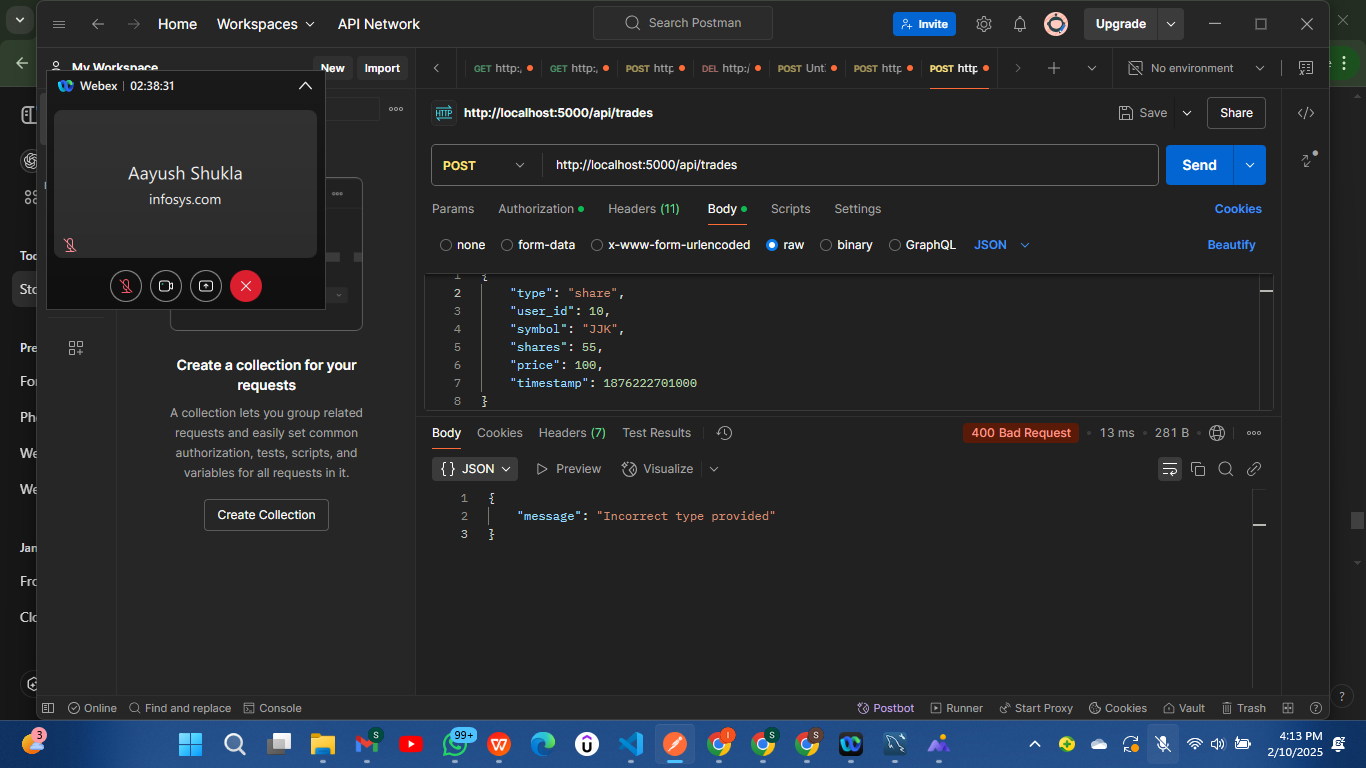
app.listen(PORT, () => console.log(` Server running on port ${PORT}`));

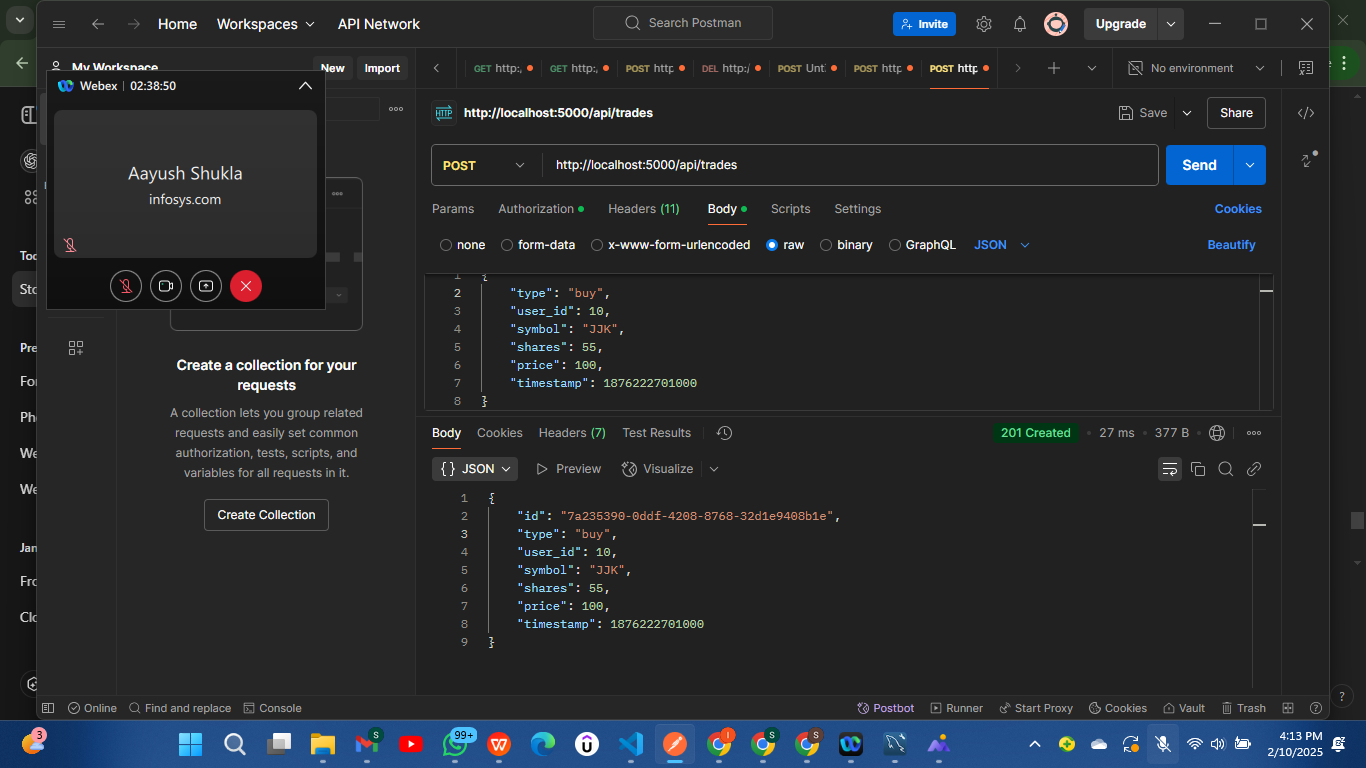
**OUTPUTS:**

****

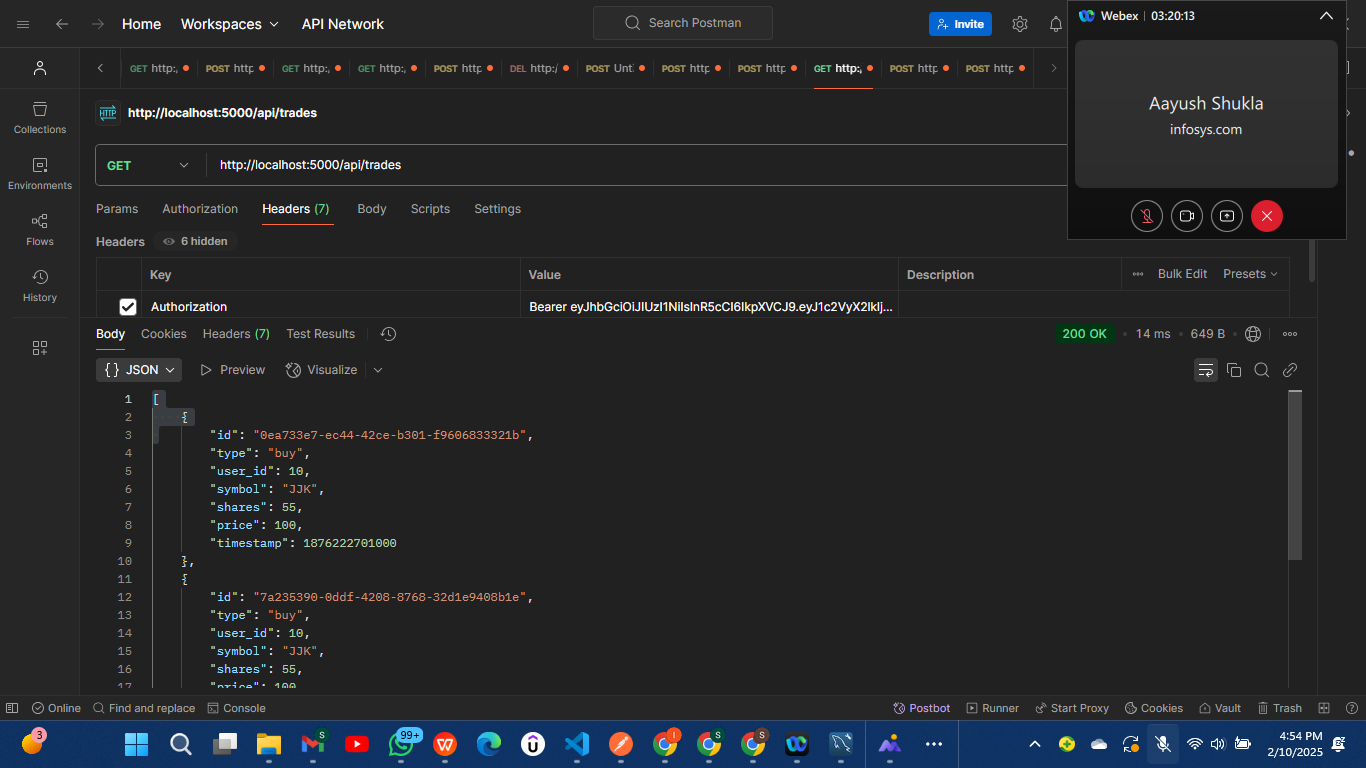
1. http://localhost:5000/api/trades

****

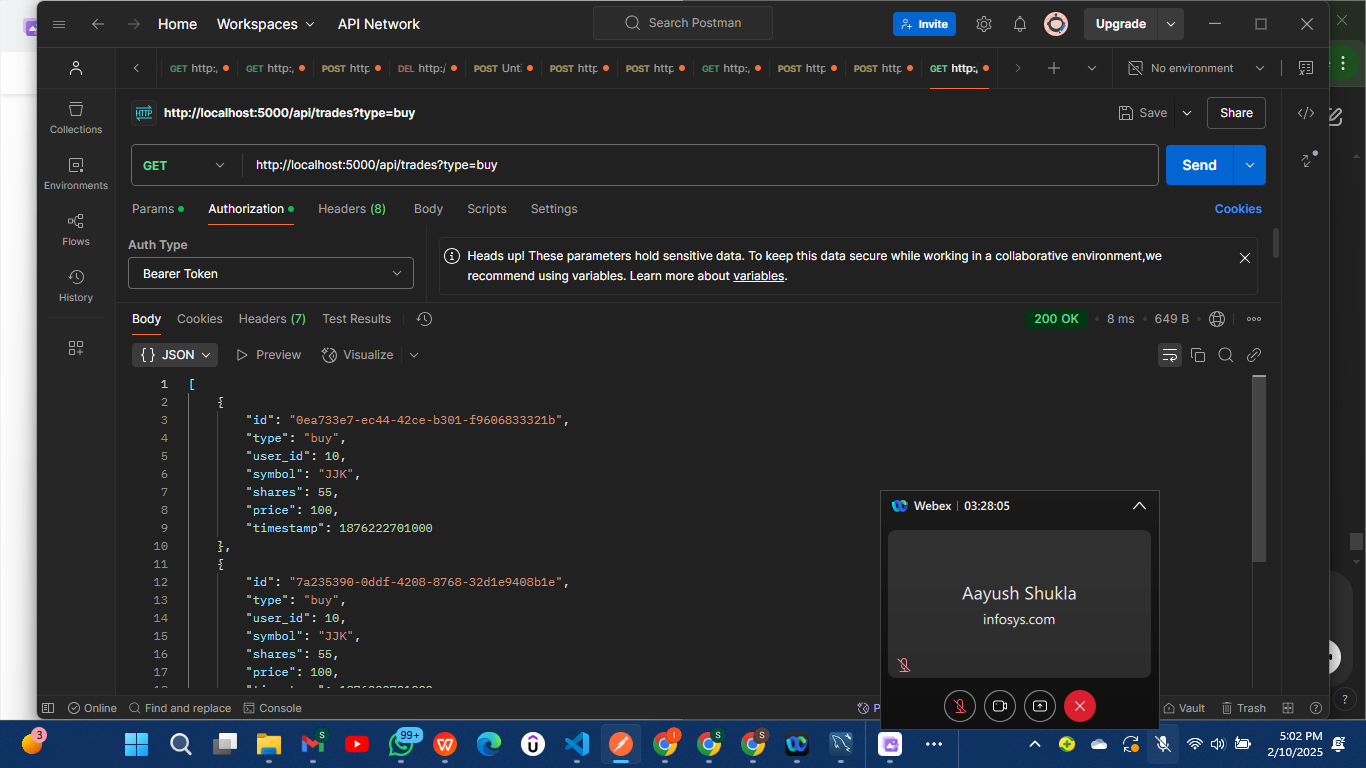
****

****

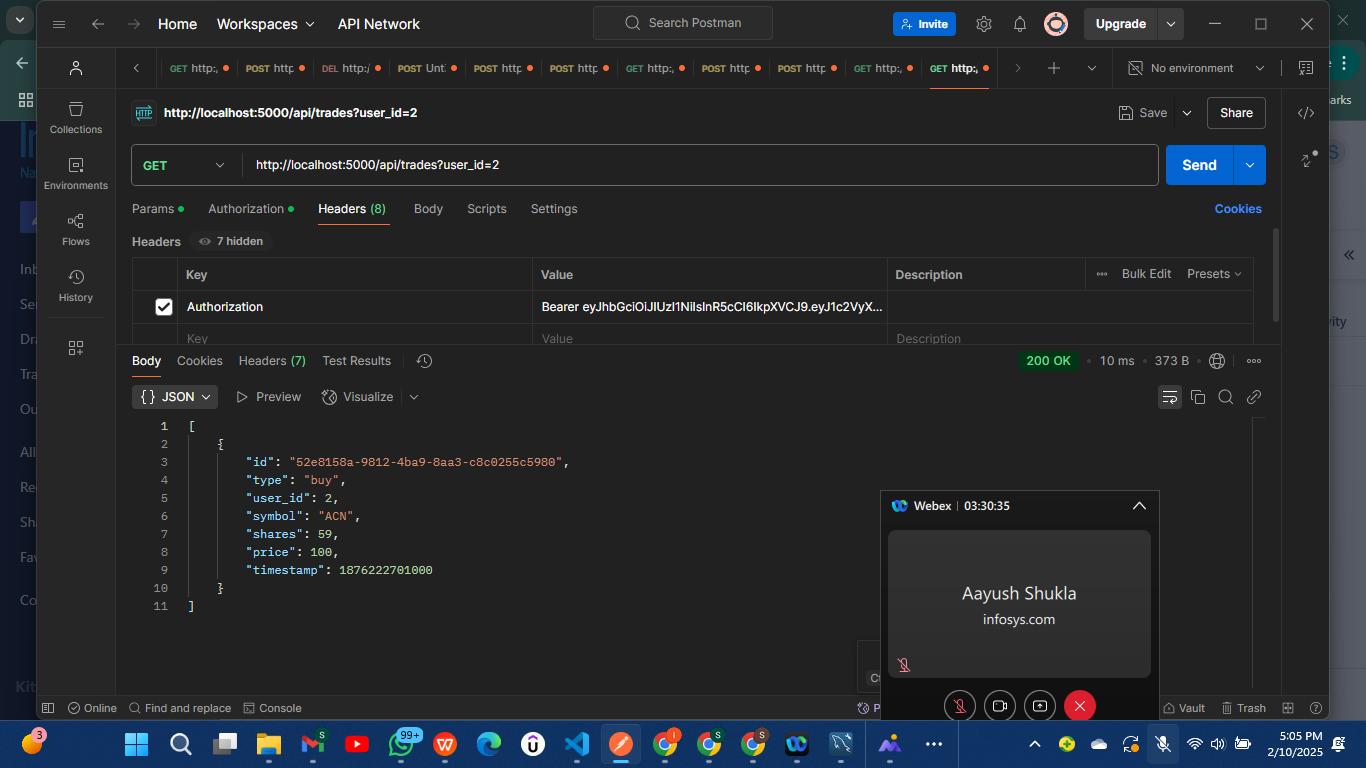
**3)**http://localhost:5000/api/trades

****

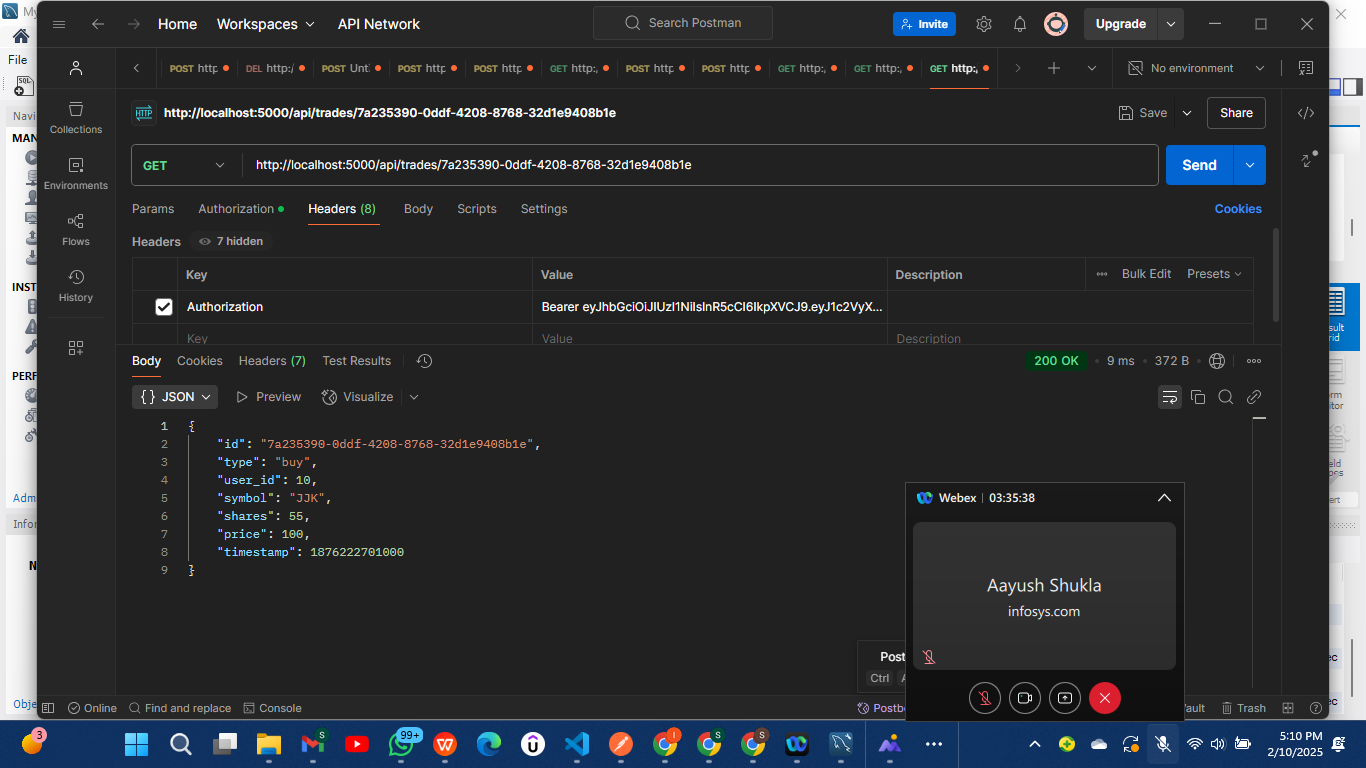
**4)** http://localhost:5000/api/trades?type=buy

****

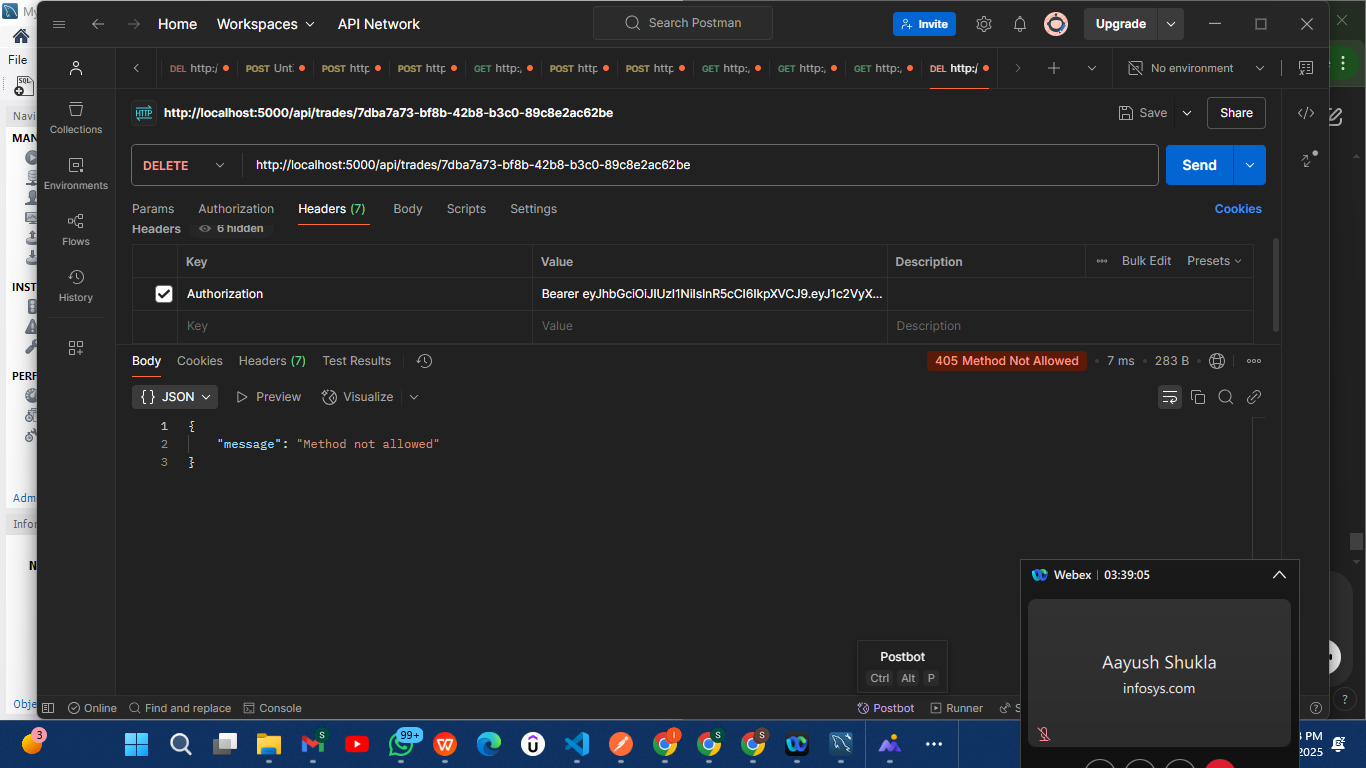
1. http://localhost:5000/api/trades?user\_id=2

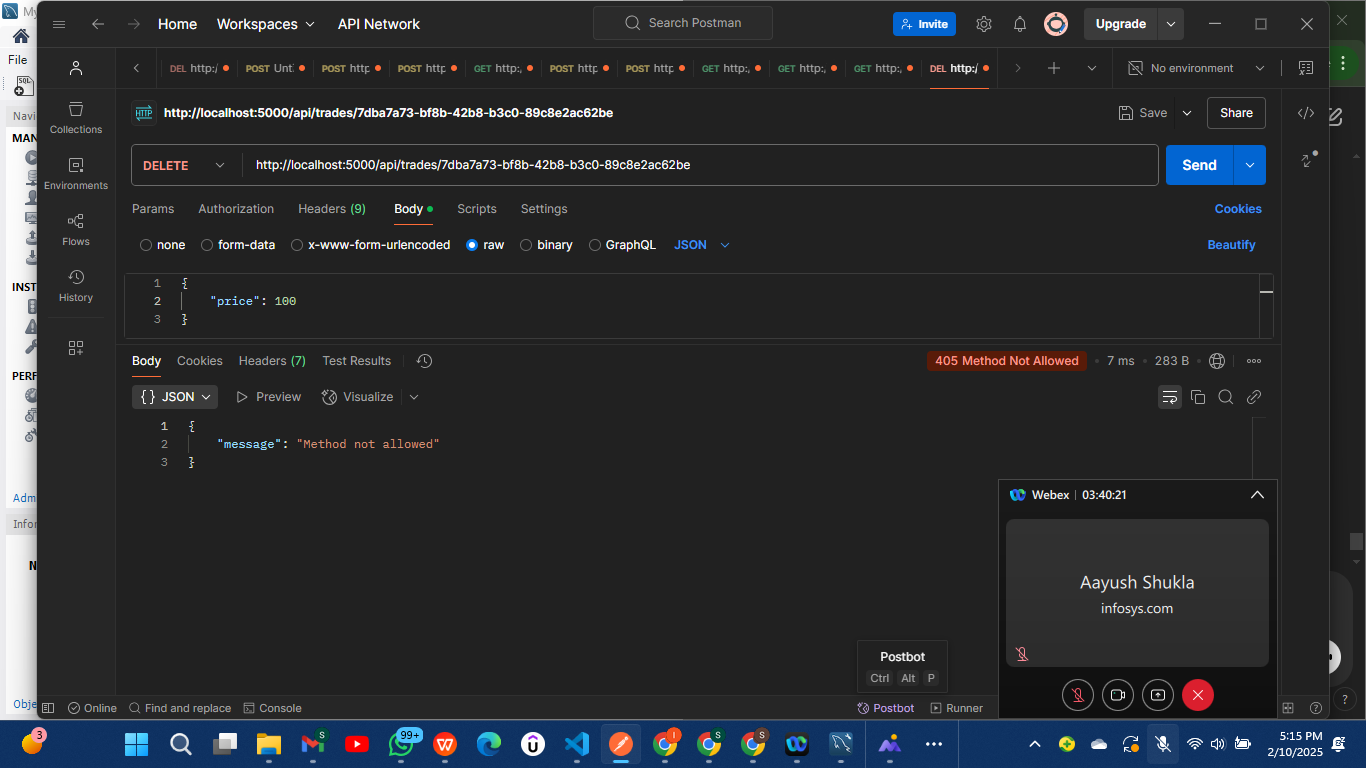
****

1. http://localhost:5000/api/trades/7a235390-0ddf-4208-8768-32d1e9408b1e

****

1. http://localhost:5000/api/trades/7dba7a73-bf8b-42b8-b3c0-89c8e2ac62be

****

****