WEB DEVELOPMENT (MINOR 2)

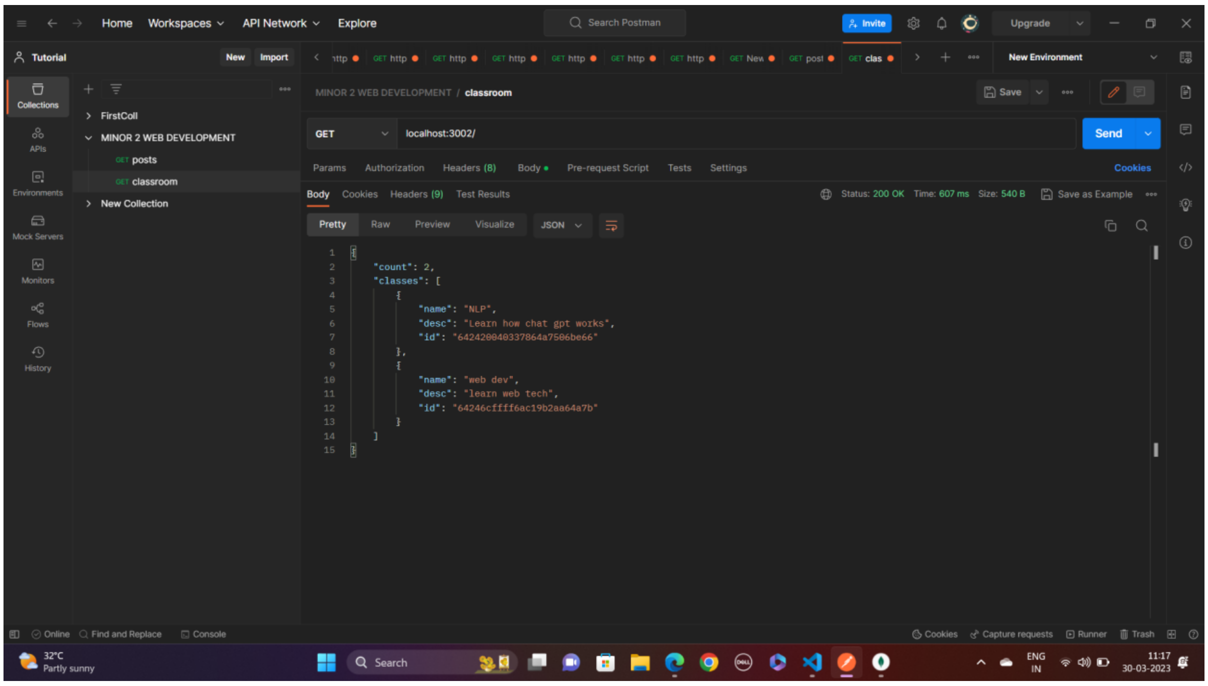
Here I am creating a rest API for a simple classroom app. Our classroom app will have following features:

1) Each classroom will have a name and a description.
2) Each classroom will have some posts.
3) Each post will contain a heading and details.

The Routes of our REST API will be as follows:

ROUTE	Request Type	Action
"/"	get	Get all the classrooms from database
"/"	post	Add a new classroom to database
"/<:classID>/posts/"	get	Get all the posts from the classroom with the specified class ID
"/<:classID>/posts/"	put	Add a post to a classroom
"/<:classID>/<:postID>/"	delete	Delete the post with the specified post ID from the classroom with the specified class ID

Let us send a get request to get all classrooms in db.



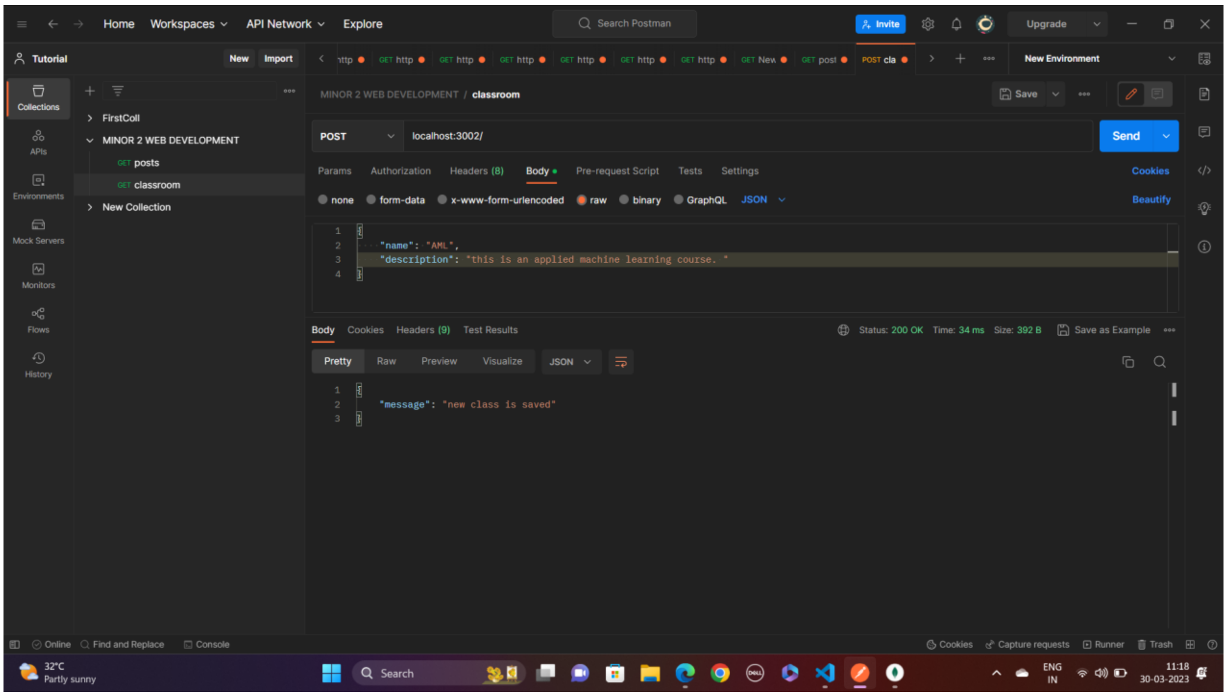
MINOR 2 WEB DEVELOPMENT / classroom

GET localhost:3002/

Status: 200 OK Time: 607 ms Size: 540 B

```
{
    "count": 2,
    "classes": [
        {
            "name": "NLP",
            "desc": "Learn how chat gpt works",
            "id": "642420040337864a7506be66"
        },
        {
            "name": "web dev",
            "desc": "learn web tech",
            "id": "64246cffff6ac19b2aa64a7b"
        }
    ]
}
```

Here we can see that we have 2 classes. Now let us add a new class using a post request



MINOR 2 WEB DEVELOPMENT / classroom

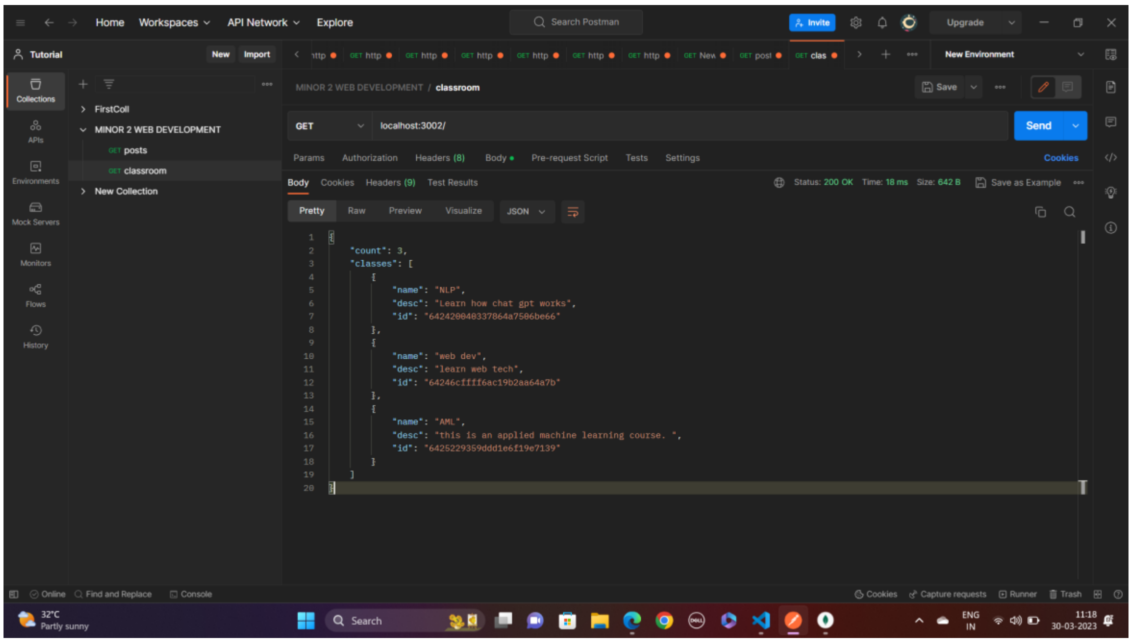
POST localhost:3002/

```
{
    "name": "AML",
    "description": "this is an applied machine learning course. "
}
```

Status: 200 OK Time: 34 ms Size: 392 B

```
{
    "message": "new class is saved"
}
```

Now let us send get request again to see all the classrooms in our db, now it must contain the new class that we have created.


MINOR 2 WEB DEVELOPMENT / classroom

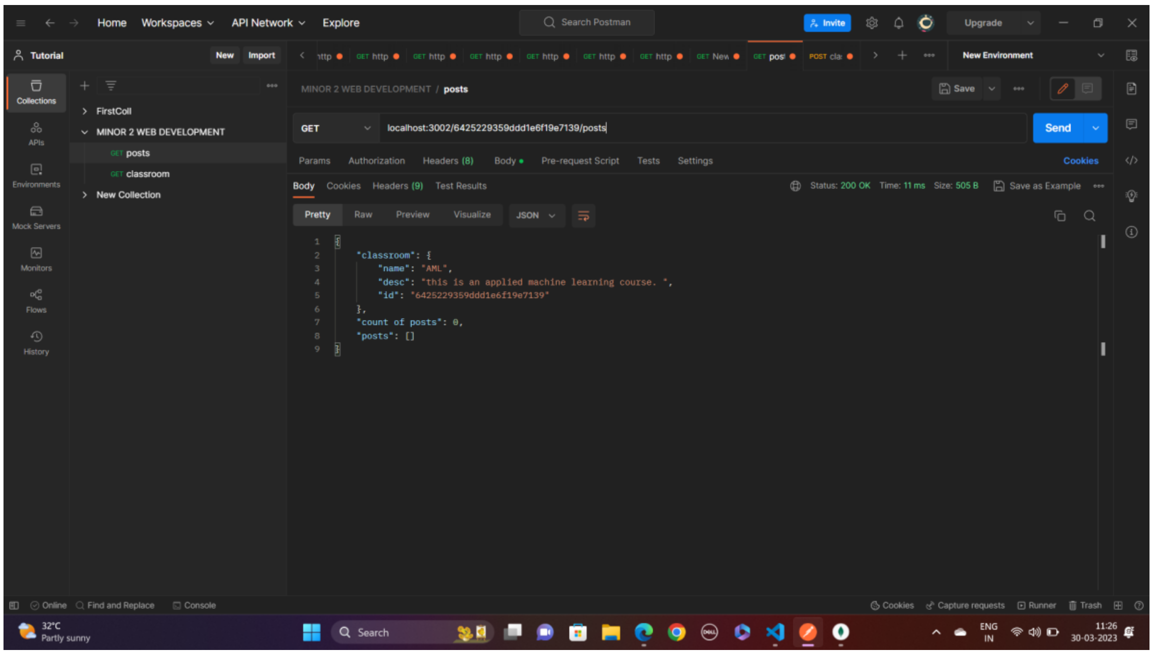
GET localhost:3002/

Status: 200 OK Time: 18 ms Size: 642 B

```
{
    "count": 3,
    "classes": [
        {
            "name": "NLP",
            "desc": "Learn how chat gpt works",
            "id": "642420040337864a7506be66"
        },
        {
            "name": "web dev",
            "desc": "learn web tech",
            "id": "64246cffff6ac19b2aa64a7b"
        },
        {
            "name": "AML",
            "desc": "this is an applied machine learning course. ",
            "id": "6425229359ddd1e6f19e7139"
        }
    ]
}
```

Now we can see the newly created class.

Now let us see all the posts in newly added classroom. We should see no posts, as if it is newly created.

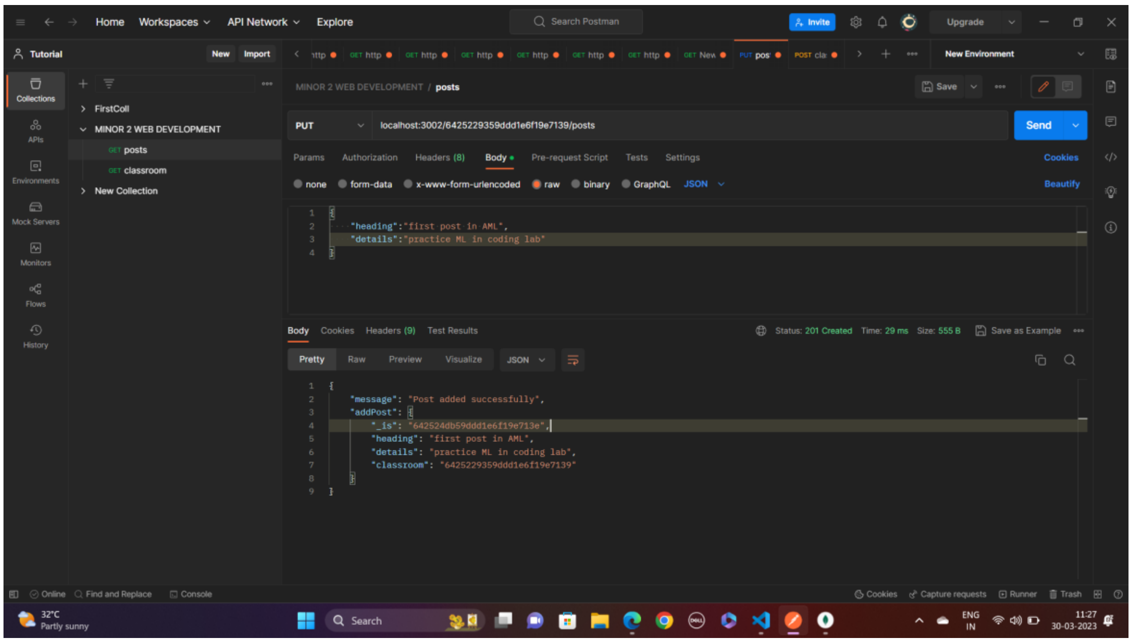

MINOR 2 WEB DEVELOPMENT / posts

GET localhost:3002/6425229359ddd1e6f19e7139/posts

Status: 200 OK Time: 11 ms Size: 505 B

```
{
    "classroom": {
        "name": "AML",
        "desc": "this is an applied machine learning course. ",
        "id": "6425229359ddd1e6f19e7139"
    },
    "count of posts": 0,
    "posts": []
}
```

Now let us add a new post to our classroom with put method.


MINOR 2 WEB DEVELOPMENT / posts

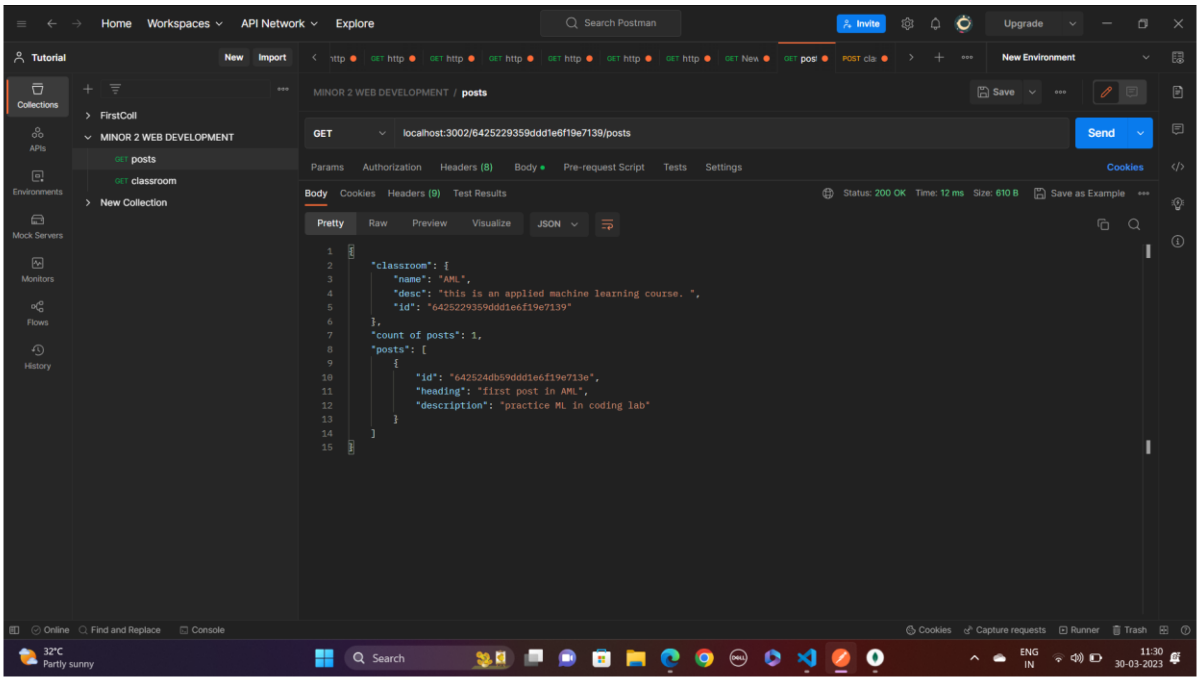
PUT localhost:3002/6425229359ddd1e6f19e7139/posts

```
{
    "heading":"first post in AML",
    "details":"practice ML in coding lab"
}
```

Status: 201 Created Time: 29 ms Size: 555 B

```
{
    "message": "Post added successfully",
    "addPost": {
        "_is": "642524db59ddd1e6f19e713e",
        "heading": "first post in AML",
        "details": "practice ML in coding lab",
        "classroom": "6425229359ddd1e6f19e7139"
    }
}
```

Now, let us view all the posts in our newly created classroom, now it must contain the newly added post.

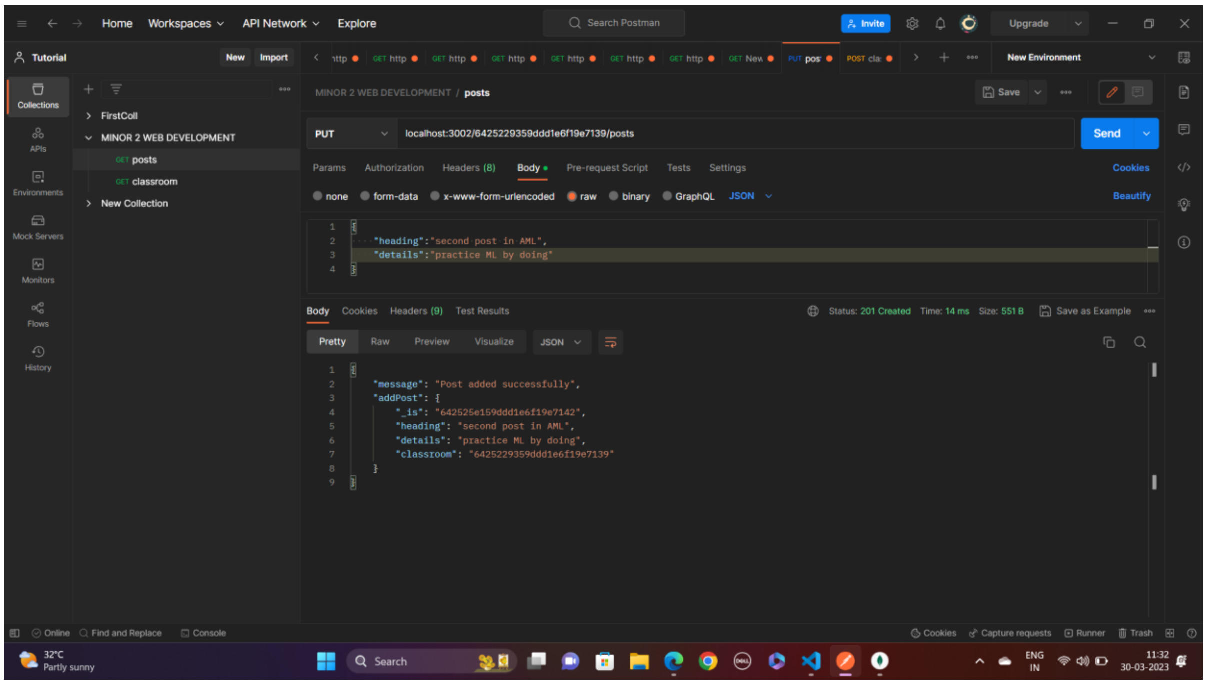

MINOR 2 WEB DEVELOPMENT / posts

GET localhost:3002/6425229359ddd1e6f19e7139/posts

Status: 200 OK Time: 12 ms Size: 610 B

```
{
    "classroom": {
        "name": "AML",
        "desc": "this is an applied machine learning course. ",
        "id": "6425229359ddd1e6f19e7139"
    },
    "count of posts": 1,
    "posts": [
        {
            "id": "642524db59ddd1e6f19e713e",
            "heading": "first post in AML",
            "description": "practice ML in coding lab"
        }
    ]
}
```

Now let us add another post in our created classroom


MINOR 2 WEB DEVELOPMENT / posts

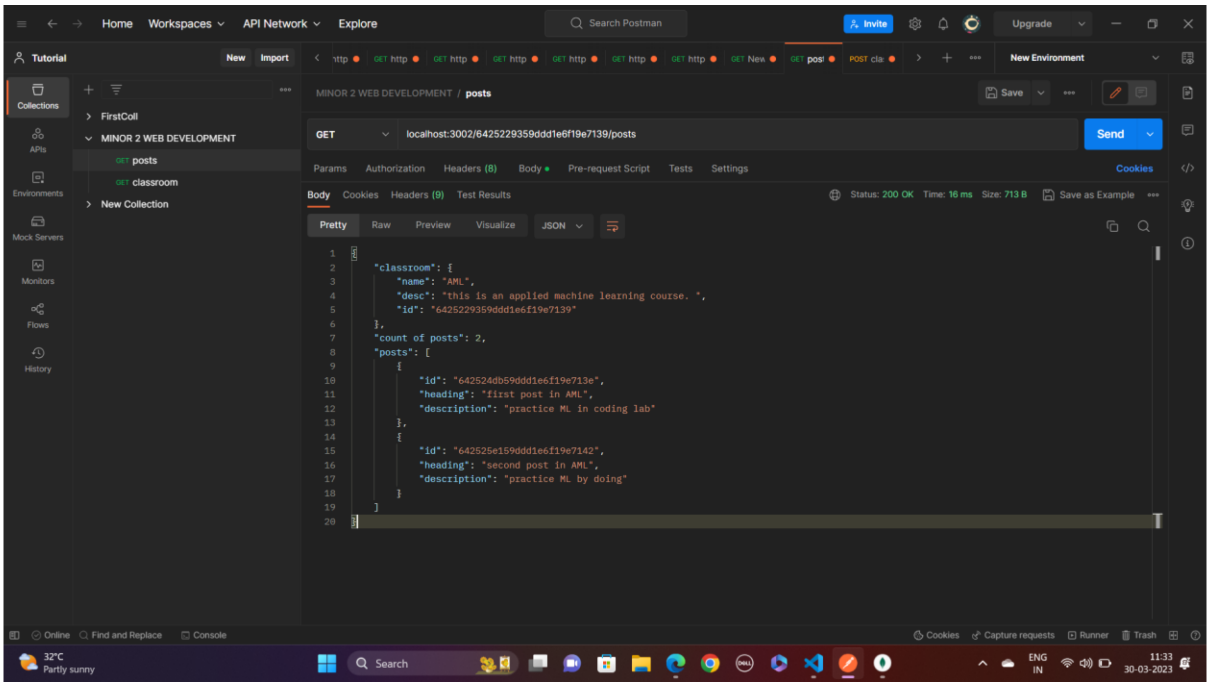
PUT localhost:3002/6425229359ddd1e6f19e7139/posts

```
{
    "heading":"second post in AML",
    "details":"practice ML by doing"
}
```

Status: 201 Created Time: 14 ms Size: 551 B

```
{
    "message": "Post added successfully",
    "addPost": {
        "_is": "642525e159ddd1e6f19e7142",
        "heading": "second post in AML",
        "details": "practice ML by doing",
        "classroom": "6425229359ddd1e6f19e7139"
    }
}
```

Now let us look at all posts in our classroom:

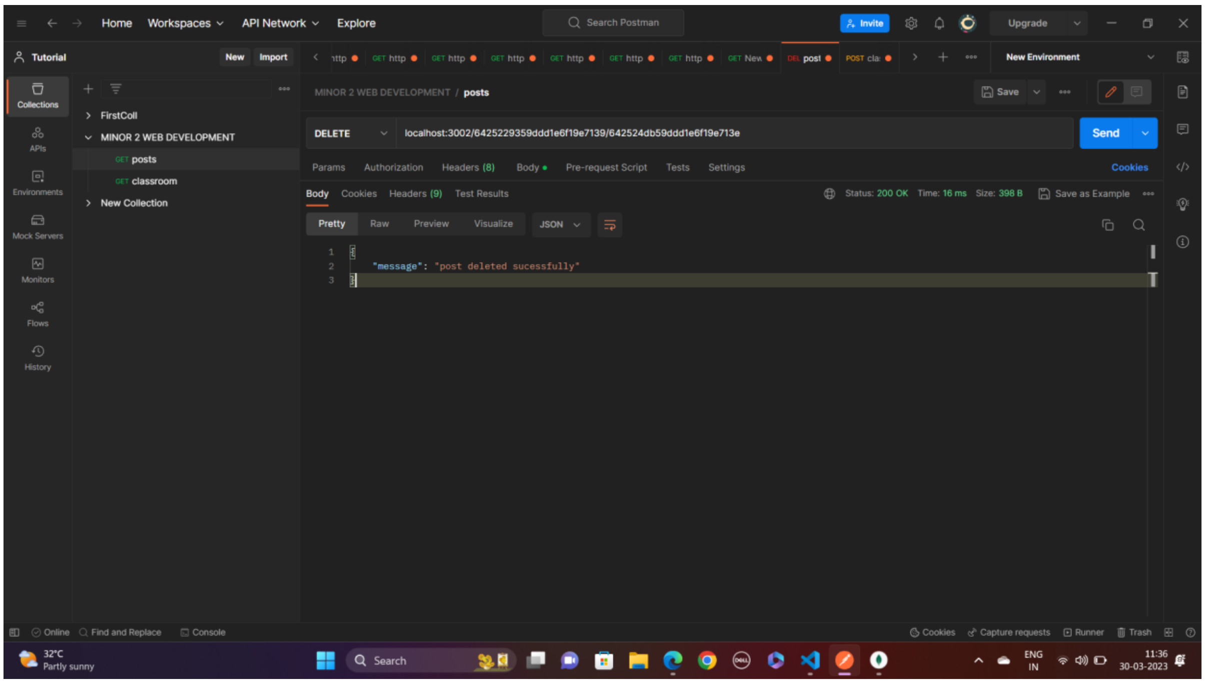

MINOR 2 WEB DEVELOPMENT / posts

GET localhost:3002/6425229359ddd1e6f19e7139/posts

Status: 200 OK Time: 16 ms Size: 713 B

```
{
    "classroom": {
        "name": "AML",
        "desc": "this is an applied machine learning course. ",
        "id": "6425229359ddd1e6f19e7139"
    },
    "count of posts": 2,
    "posts": [
        {
            "id": "642524db59ddd1e6f19e713e",
            "heading": "first post in AML",
            "description": "practice ML in coding lab"
        },
        {
            "id": "642525e159ddd1e6f19e7142",
            "heading": "second post in AML",
            "description": "practice ML by doing"
        }
    ]
}
```

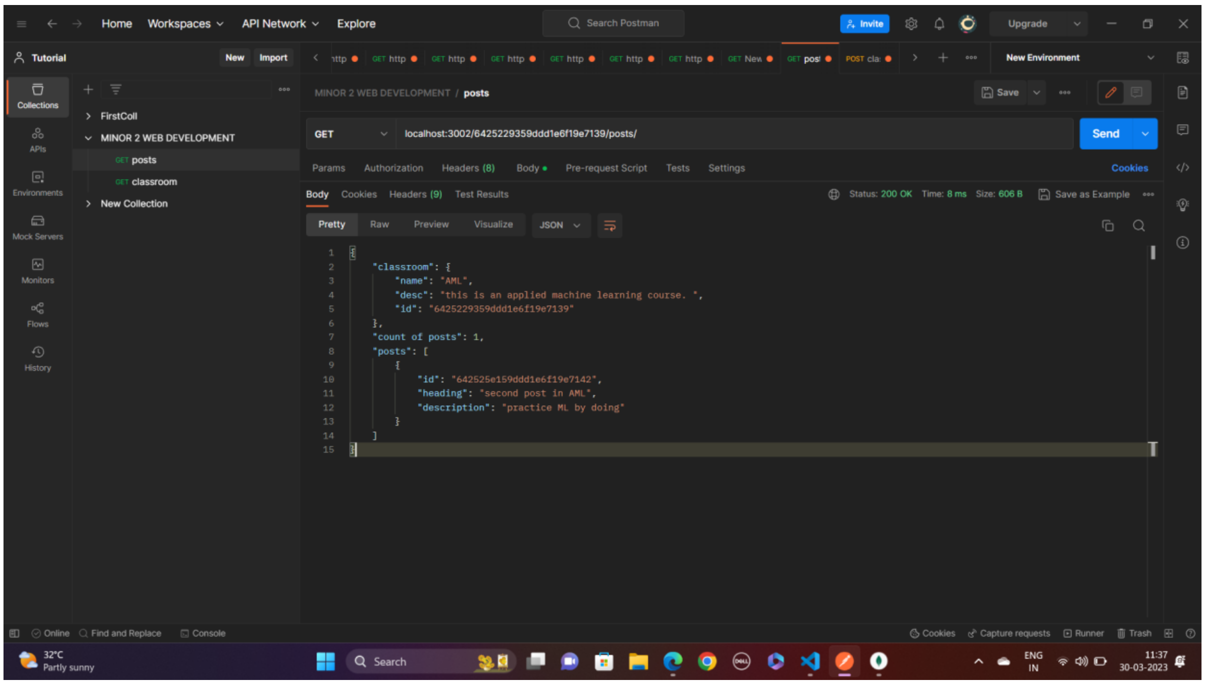
Now let us delete first post in our created classroom


MINOR 2 WEB DEVELOPMENT / posts
Status: 200 OK Time: 16 ms Size: 398 B

```
DELETE localhost:3002/6425229359ddd1e6f19e7139/642524db59ddd1e6f19e713e
```

```
{
    "message": "post deleted sucessfully"
}
```

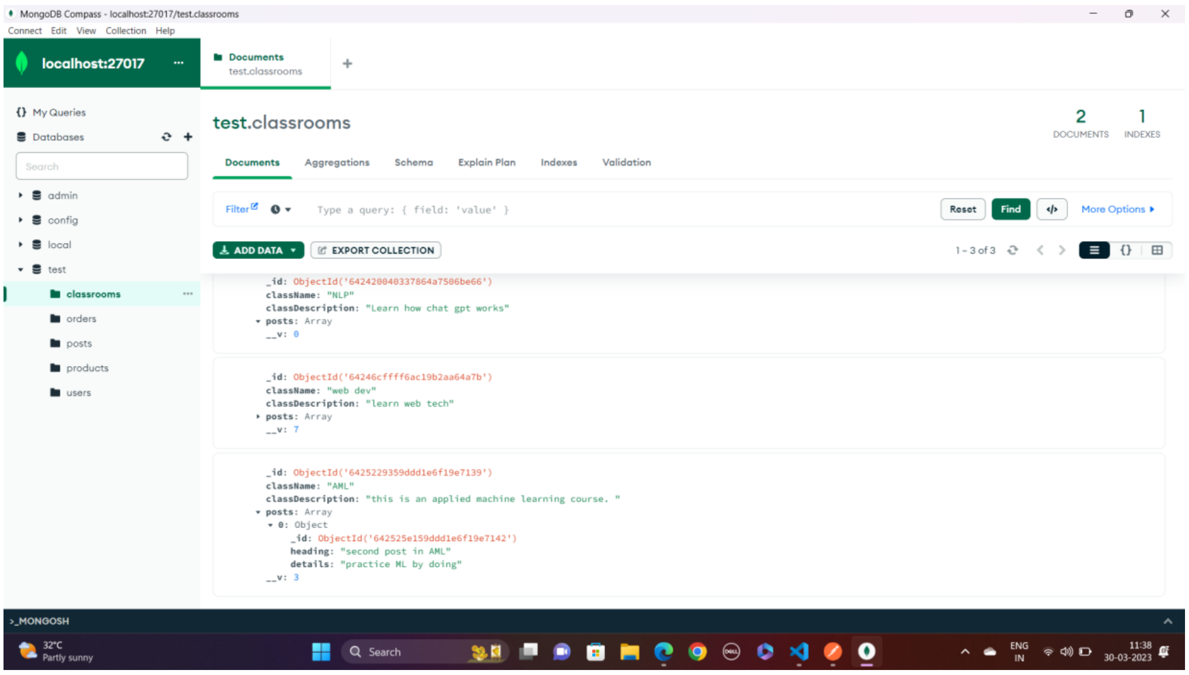
Now let us see all posts in our classroom


MINOR 2 WEB DEVELOPMENT / posts
Status: 200 OK Time: 8 ms Size: 606 B

```
GET localhost:3002/6425229359ddd1e6f19e7139/posts/
```

```
{
    "classroom": {
        "name": "AML",
        "desc": "this is an applied machine learning course. ",
        "id": "6425229359ddd1e6f19e7139"
    },
    "count of posts": 1,
    "posts": [
        {
            "id": "642525e159ddd1e6f19e7142",
            "heading": "second post in AML",
            "description": "practice ML by doing"
        }
    ]
}
```

Screenshot of mongodb compass:



```
_id: ObjectId('642420840337864a7506be66')
className: "NLP"
classDescription: "Learn how chat gpt works"
posts: Array
__v: 0

_id: ObjectId('64246cffff6ac19b2aa64a7b')
className: "web dev"
classDescription: "learn web tech"
posts: Array
__v: 7

_id: ObjectId('6425229359ddd1e6f19e7139')
className: "AML"
classDescription: "this is an applied machine learning course. "
posts: Array
  0: Object
    _id: ObjectId('642525e159ddd1e6f19e7142')
    heading: "second post in AML"
    details: "practice ML by doing"
__v: 3
```

CODE:

Server.js:

```
const http = require('http');
const app = require('./app');
const port = process.env.PORT || 3002;
const server = http.createServer(app);
server.listen(port);
```

app.js:

```
const express = require('express');
const app = express();
const morgan = require("morgan");
const bodyParser = require("body-parser");
const mongoose = require("mongoose");
mongoose.connect("mongodb://0.0.0.0:27017/");
const classRoute = require("./api/routes/classroom");
app.use(morgan("dev"));
app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json());

app.use((req, res, next) => {
    // res.header('Access-Control-Allow-Origin',"https://www.google.com")
    res.header("Access-Control-Allow-Origin", "*");
    // res.header('Access-Control-Allow-Headers',"*" );
    res.header(
      "Access-Control-Allow-Headers",
      "Origin, X-Requested-With, Content-Type, Accept, Authorization"
    );
    if (req.method === "OPTIONS") {
      res.header(
        "Access-Control-Allow-Methods",
        "PUT, POST, PATCH, DELETE, GET "
      );
      return res.status(200).json({});
    }
    next();
});
app.use("/",classRoute);
app.use((req, res, next)=>{
    res.status(200).json({
        "err":"no paths match your route"
    })
})
module.exports=app;
```

Mongoose Schema:

```
const mongoose = require("mongoose");
const postsSchema = mongoose.Schema({
  _id: mongoose.Schema.Types.ObjectId,
  heading: String,
  details: {type: String, required: false},
});
const classroomSchema = mongoose.Schema({
  _id: mongoose.Schema.Types.ObjectId,
  className: { type: String, required: true },
  classDescription: { type: String, required: true },
  posts: [postsSchema],
});
module.exports = {
    Classroom: mongoose.model("Classroom", classroomSchema),
    Posts : mongoose.model('Posts', postsSchema),};
```

Routes file:

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

const mongoose = require("mongoose");

const router = express.Router();

const { Classroom: classroom, Posts } = require("../models/classroom");

const { Posts: post } = require("../models/classroom");



router.get("/", (req, res, next) => {

  classroom

    .find()

    .exec()

    .then((docs) => {

      const response = {

        count: docs.length,

        classes: docs.map((cls) => {

          return {

            name: cls.className,

            desc: cls.classDescription,

            id: cls._id,

          };

        }),

      };

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

    })

    .catch((err) => {

      console.log("error occured at get");

      res.status(500).json({ error: err });

    });

});
```

```
router.post("/", (req, res, next) => {
  console.log("name:" + req.body.name);
  console.log("descrip:" + req.body.description);
  const cls = new classroom({
    _id: new mongoose.Types.ObjectId(),
    className: req.body.name,
    classDescription: req.body.description,
  });
  console.log("here at after cls");
  cls
    .save()
    .then((result) => {
      console.log("result:");
      console.log(result);
      console.log("saving completed");
      res.status(200).json({
        message: "new class is saved",
      });
    })
    .catch((err) => {
      console.log("error happened");
      console.log(err);
      res.status(500).json({ error: err });
    });
});

router.get("/:classroomId/posts", (req, res, next) => {
  const id = req.params.classroomId;
  classroom
    .findById(id)
    .exec()
    .then((resulingClassroom) => {
      if (!res) {
        return res.status(500).json({
          message: "cant find the classroom",
        });
      }
      console.log("uff");
      const responses = {
        classroom: {
          name: resulingClassroom.className,
          desc: resulingClassroom.classDescription,
          id: resulingClassroom._id,
        },
        "count of posts": resulingClassroom.posts.length,
        posts: resulingClassroom.posts.map((post) => {
          return {
            id: post._id,
            heading: post.heading,
            description: post.details,
          };
        }),
      };
      res.status(200).json(responses);
    })
    .catch((err) => {
      console.log(err);
      res.status(500).json({
        error: err,
      });
    });});
router.put("/:classroomId/posts", (req, res, next) => {
  const id = req.params.classroomId;
  const pst = new post({
    _id: new mongoose.Types.ObjectId(),
    heading: req.body.heading,
    details: req.body.details,
  });
  classroom
    .findById(id)
    .exec()
    .then((classroom) => {
      if (!classroom) {
        return res.status(404).json({
          message: "Classroom not found",
        });
      }
      classroom.posts.push(pst);
      classroom
        .save()
        .then(() => {
          res.status(201).json({
            message: "Post added successfully",
            addPost: {
              _is: pst._id,
              heading: pst.heading,
              details: pst.details,
              classroom: classroom._id,
            },
          });
        })
        .catch((err) => {
          console.error(err);
          res.status(500).json({
            error: err,
          });
        });
    })
    .catch((err) => {
      console.log(err);
      res.status(500).json({
        error: err,
      });
    });
});
```

```
router.delete("/:classroomId/:postid", (req, res, next) => {
  const classId = req.params.classroomId;
  const postId = req.params.postid;
  classroom
    .findById(classId)
    .exec()
    .then((clas) => {
      if (!clas) {
        return res.status(500).json({ err: "classroom not found" });
      }
      // const pos = clas.post._id(postId);
      // if(!pos){return res.status(404).json({message: "post not found"})}
      // pos.remove();
      for (var i = clas.posts.length - 1; i >= 0; i--) {
        if (clas.posts[i]._id == postId) {
          clas.posts.splice(i, 1);
        }
      }
      clas
        .save()
        .then(() => {
          res.status(200).json({
            message: "post deleted sucessfully",
          });
        })
        .catch((err) => {
          console.error(err);
          res.status(500).json({
            error: err,
          });
        });
    })
    .catch((err) => {
      console.error(err);
      res.status(500).json({
        error: err,
      });
    });
});

module.exports = router;
```

PADURU VIGNA TEJ REDDY(420217)