САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа 2 “Тестирование, разработка и документирование RESTful API ”

Выполнил:

До Ван Тхиен

К33402

Проверил:
Добряков Д. И.

Санкт-Петербург

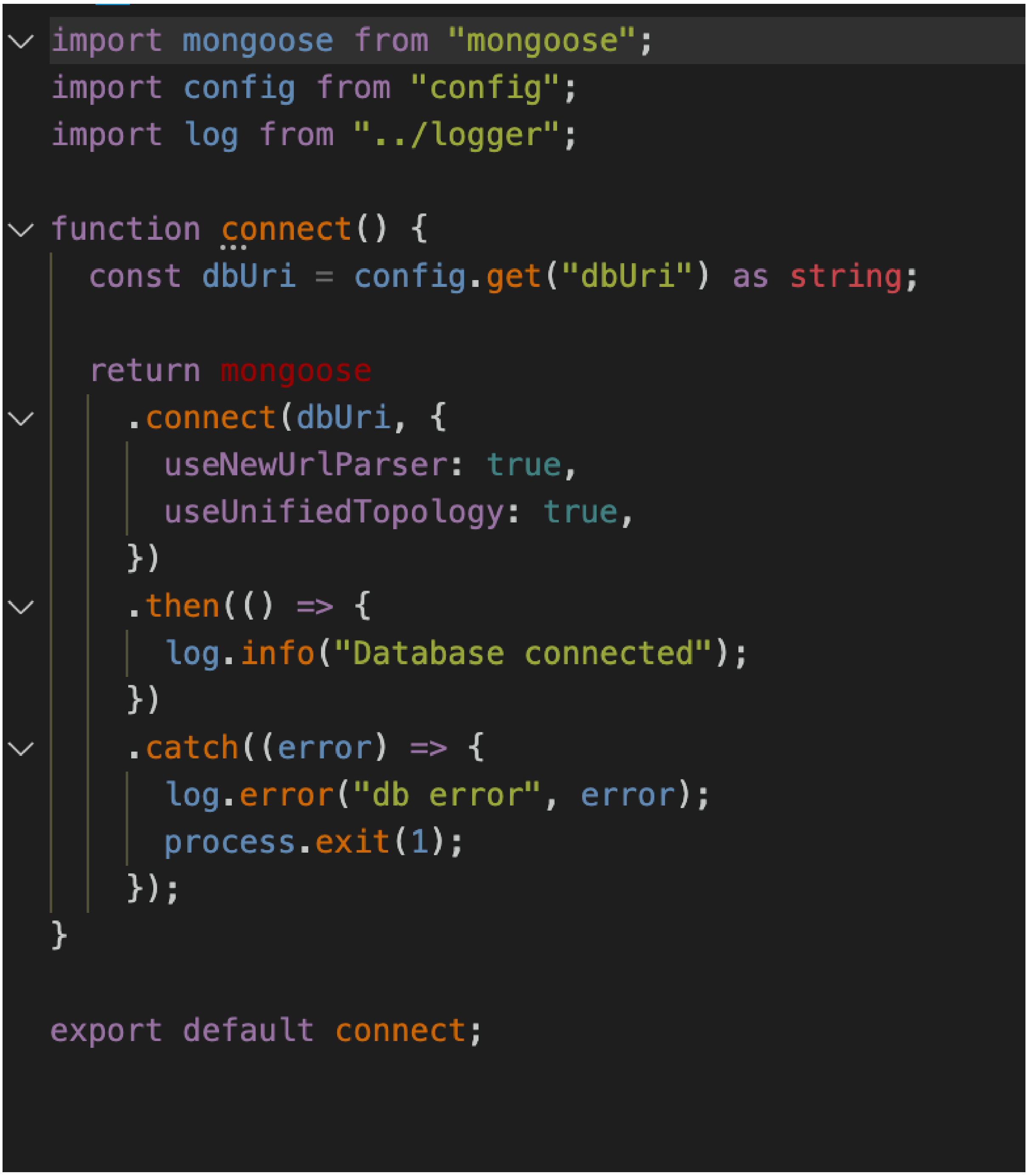
2022 г.

Задача

Реализовать RESTful API средствами express + typescript

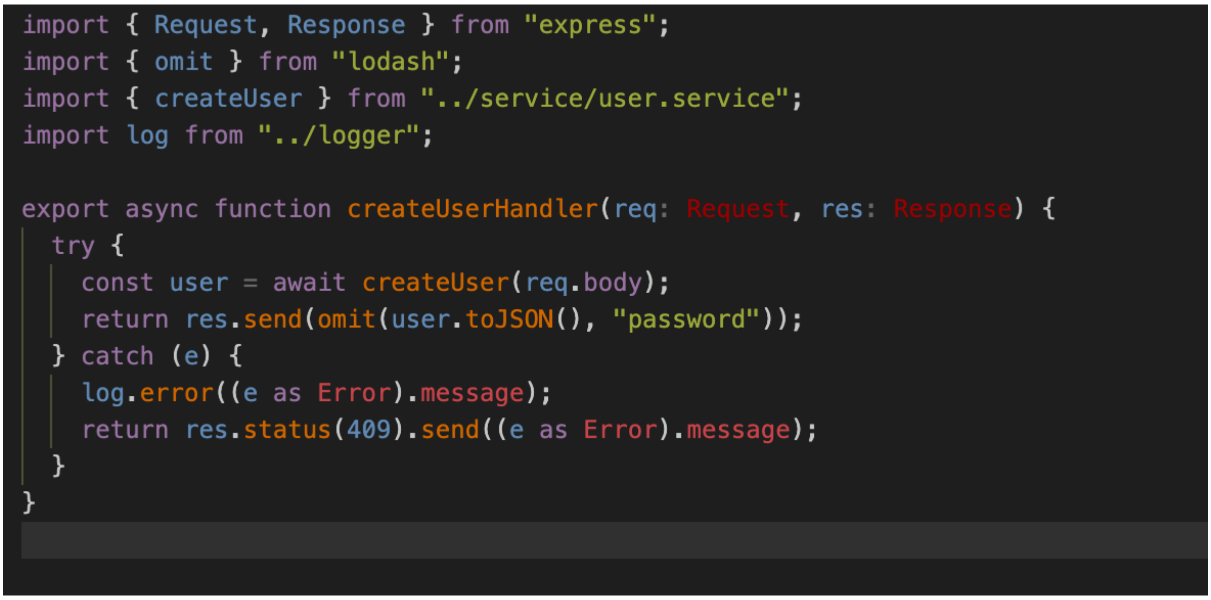
Ход работы

1. подключиться к бд



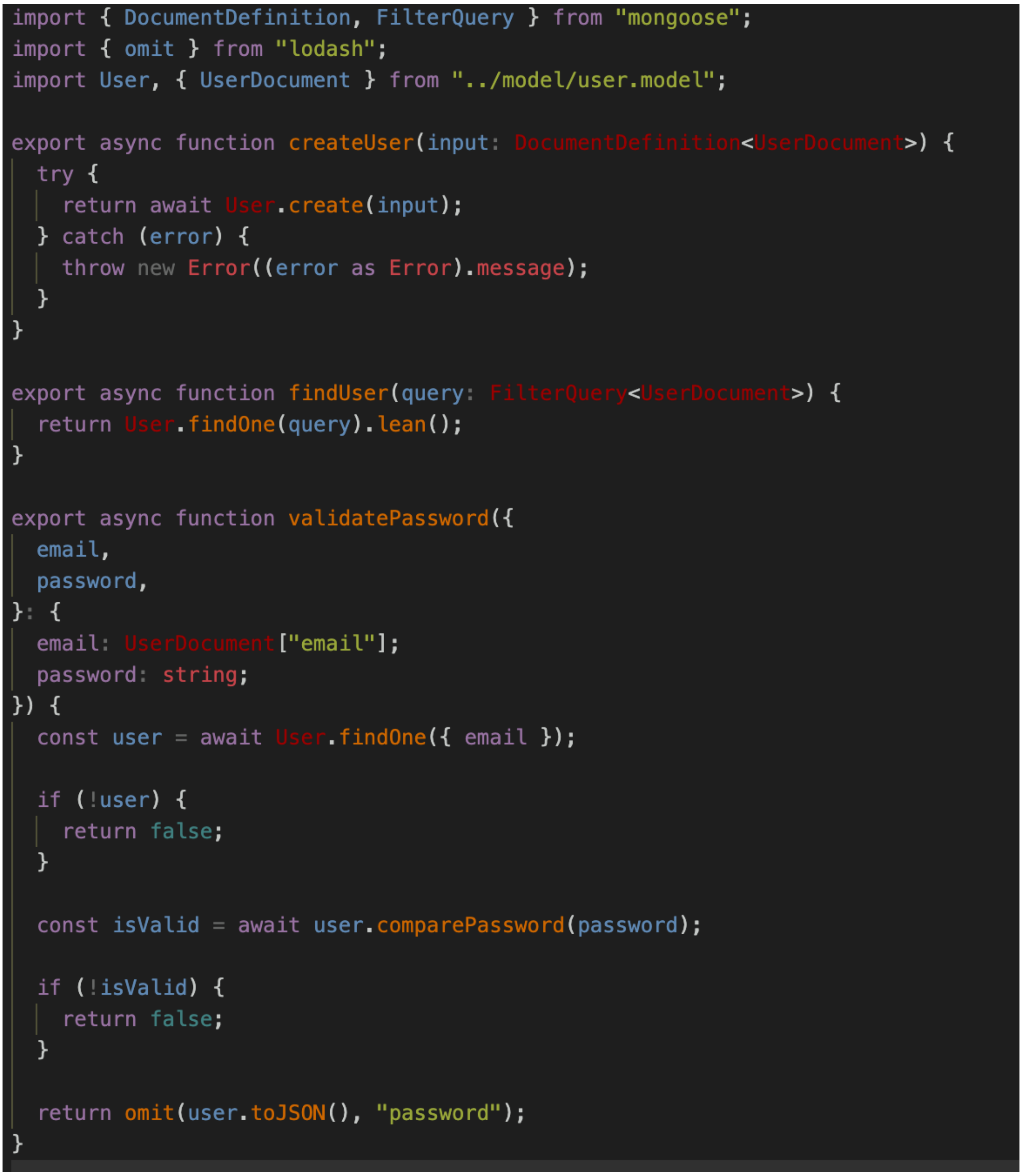
```
import mongoose from "mongoose";
import config from "config";
import log from "../logger";

function connect() {
  const dbUri = config.get("dbUri") as string;

  return mongoose
    .connect(dbUri, {
      useNewUrlParser: true,
      useUnifiedTopology: true,
    })
    .then(() => {
      log.info("Database connected");
    })
    .catch((error) => {
      log.error("db error", error);
      process.exit(1);
    });
}

export default connect;
```

2. controller..user.ts



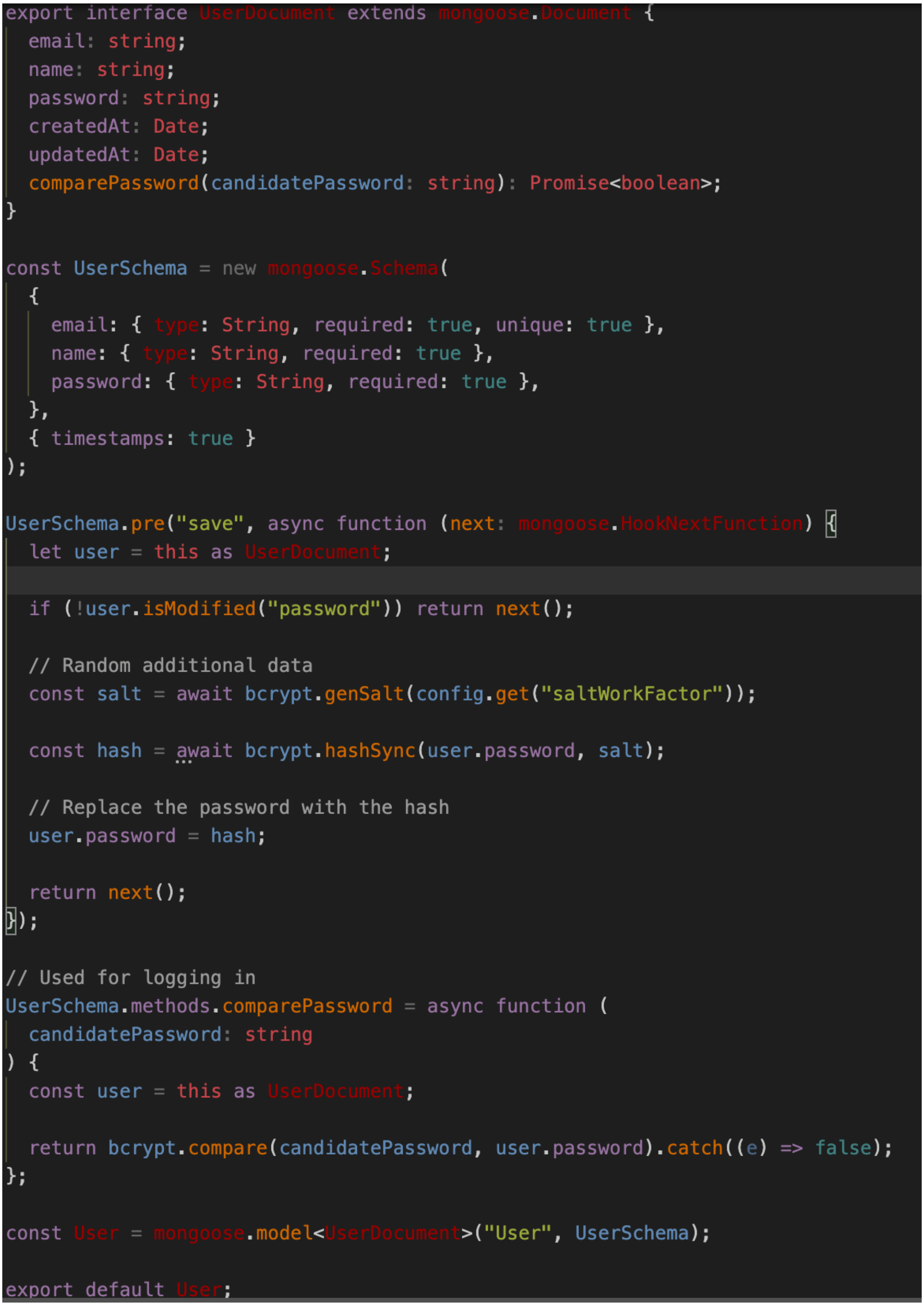
```
import { Request, Response } from "express";
import { omit } from "lodash";
import { createUser } from "../service/user.service";
import log from "../logger";

export async function createUserHandler(req: Request, res: Response) {
  try {
    const user = await createUser(req.body);
    return res.send(omit(user.toJSON(), "password"));
  } catch (e) {
    log.error((e as Error).message);
    return res.status(409).send((e as Error).message);
  }
}
```

3. service.user.ts



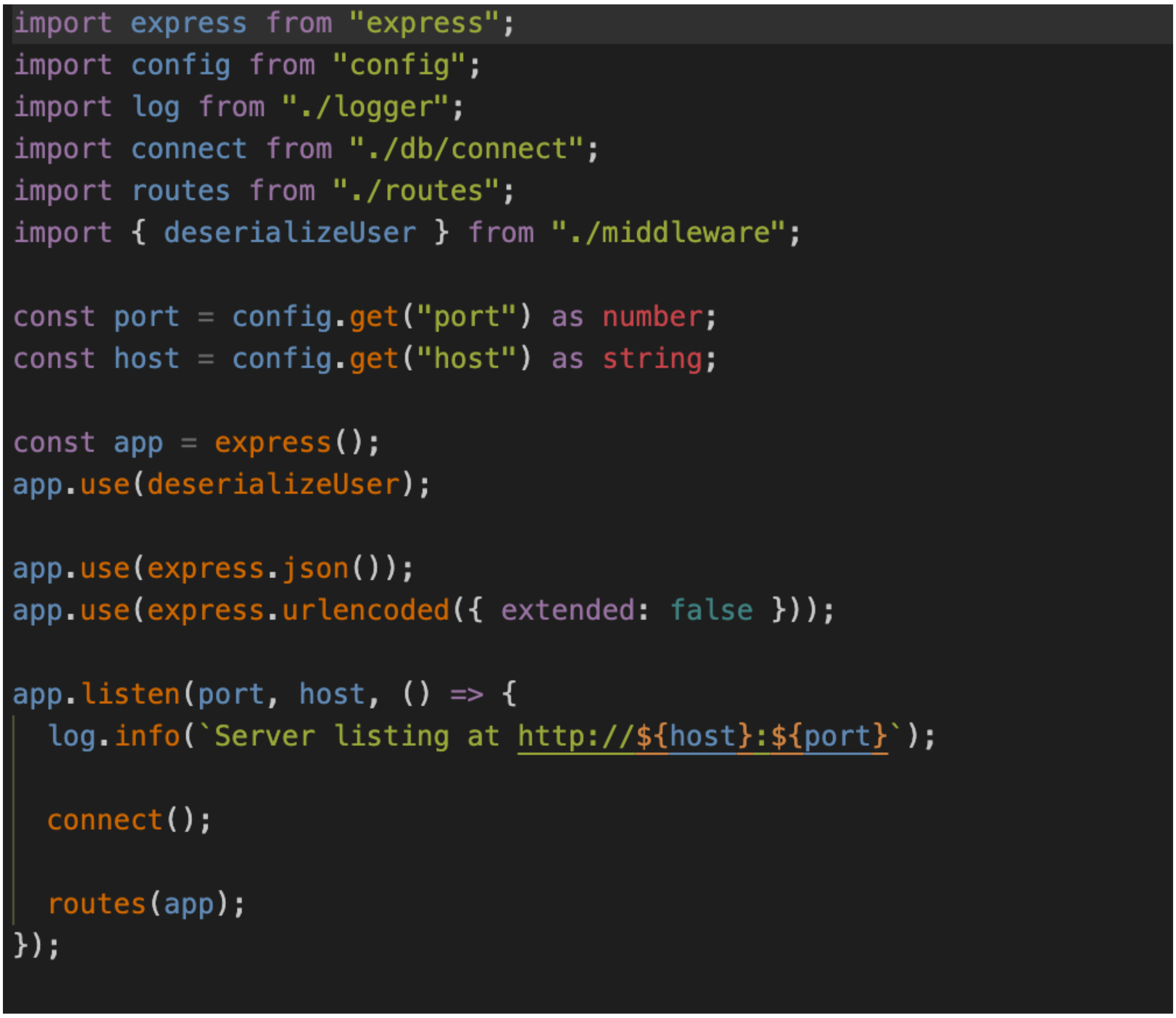
```typescript
import { DocumentDefinition, FilterQuery } from "mongoose";
import { omit } from "lodash";
import User, { UserDocument } from "../model/user.model";

export async function createUser(input: DocumentDefinition<UserDocument>) {
  try {
    return await User.create(input);
  } catch (error) {
    throw new Error((error as Error).message);
  }
}

export async function findUser(query: FilterQuery<UserDocument>) {
  return User.findOne(query).lean();
}

export async function validatePassword({
  email,
  password,
}: {
  email: UserDocument["email"];
  password: string;
}) {
  const user = await User.findOne({ email });

  if (!user) {
    return false;
  }

  const isValid = await user.comparePassword(password);

  if (!isValid) {
    return false;
  }

  return omit(user.toJSON(), "password");
}
```

4. model.user.ts



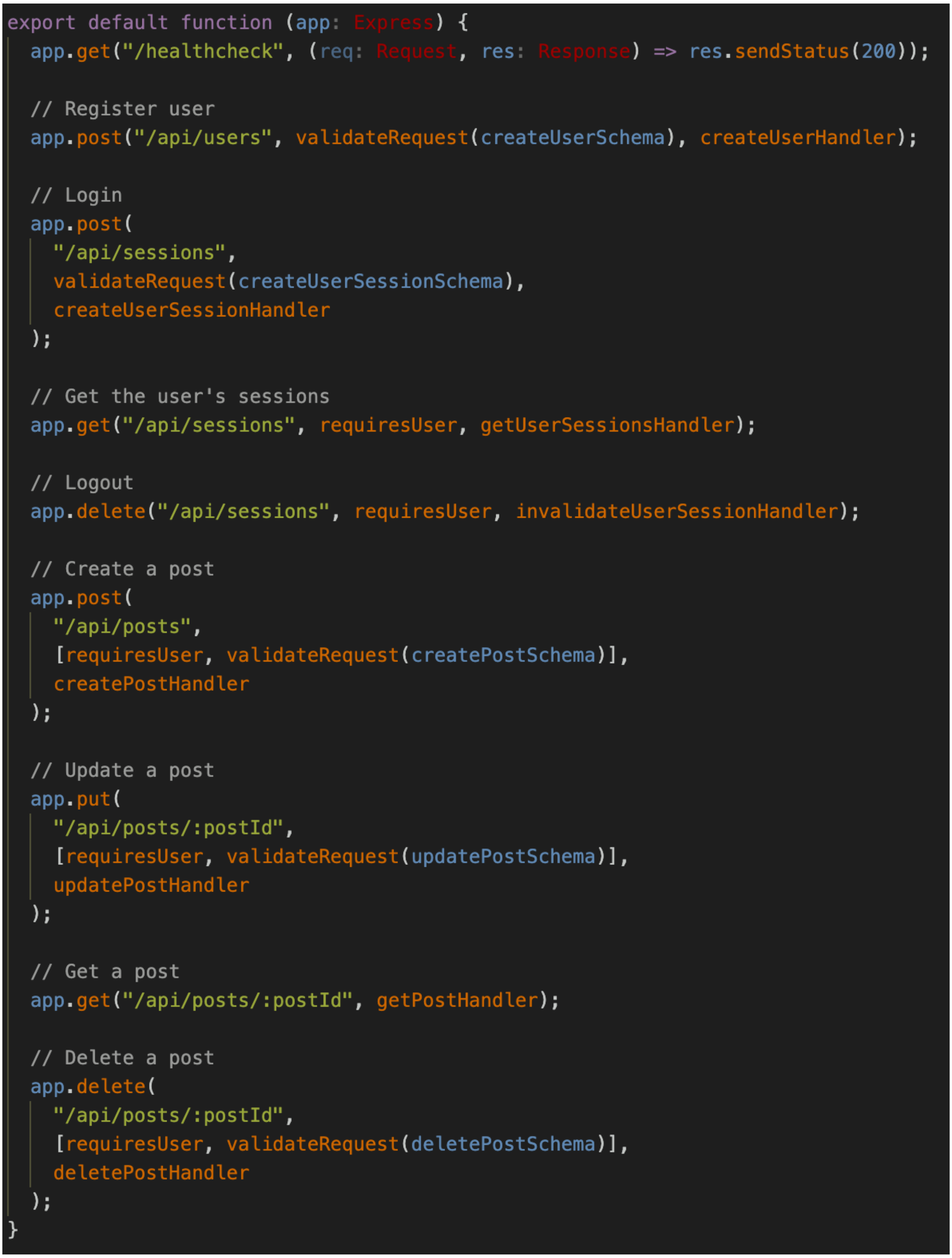
```
export interface UserDocument extends mongoose.Document {
  email: string;
  name: string;
  password: string;
  createdAt: Date;
  updatedAt: Date;
  comparePassword(candidatePassword: string): Promise<boolean>;
}

const UserSchema = new mongoose.Schema(
  {
    email: { type: String, required: true, unique: true },
    name: { type: String, required: true },
    password: { type: String, required: true },
  },
  { timestamps: true }
);

UserSchema.pre("save", async function (next: mongoose.HookNextFunction) {
  let user = this as UserDocument;

  if (!user.isModified("password")) return next();

  // Random additional data
  const salt = await bcrypt.genSalt(config.get("saltWorkFactor"));

  const hash = await bcrypt.hashSync(user.password, salt);

  // Replace the password with the hash
  user.password = hash;

  return next();
});

// Used for logging in
UserSchema.methods.comparePassword = async function (
  candidatePassword: string
) {
  const user = this as UserDocument;

  return bcrypt.compare(candidatePassword, user.password).catch((e) => false);
};

const User = mongoose.model<UserDocument>("User", UserSchema);

export default User;
```

5. app.ts



```
import express from "express";
import config from "config";
import log from "./logger";
import connect from "./db/connect";
import routes from "./routes";
import { deserializeUser } from "./middleware";

const port = config.get("port") as number;
const host = config.get("host") as string;

const app = express();
app.use(deserializeUser);

app.use(express.json());
app.use(express.urlencoded({ extended: false }));

app.listen(port, host, () => {
  log.info(`Server listing at http://${host}:${port}`);

  connect();

  routes(app);
});
```

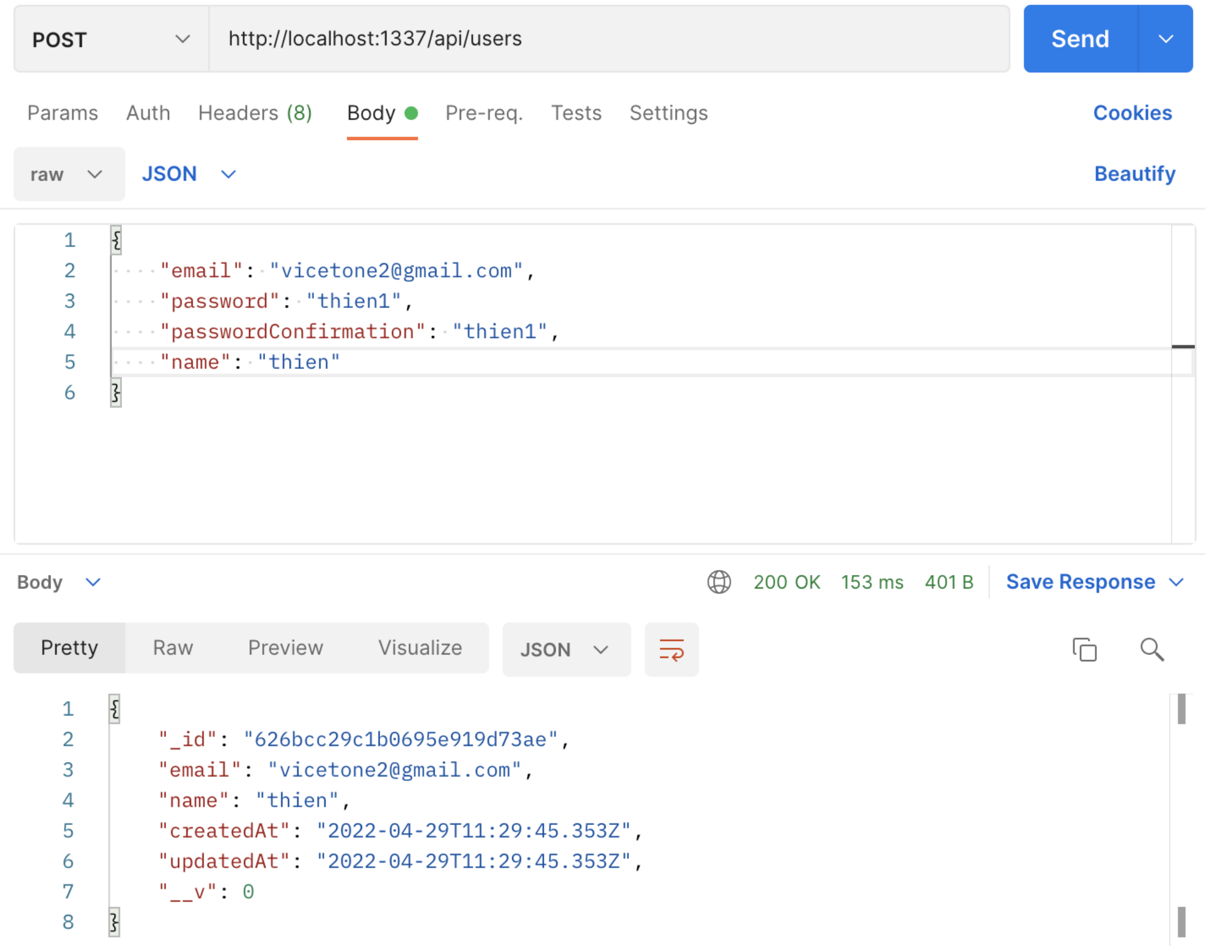
6. routes.ts



```
export default function (app: Express) {
  app.get("/healthcheck", (req: Request, res: Response) => res.sendStatus(200));

  // Register user
  app.post("/api/users", validateRequest(createUserSchema), createUserHandler);

  // Login
  app.post(
    "/api/sessions",
    validateRequest(createUserSessionSchema),
    createUserSessionHandler
  );

  // Get the user's sessions
  app.get("/api/sessions", requiresUser, getUserSessionsHandler);

  // Logout
  app.delete("/api/sessions", requiresUser, invalidateUserSessionHandler);

  // Create a post
  app.post(
    "/api/posts",
    [requiresUser, validateRequest(createPostSchema)],
    createPostHandler
  );

  // Update a post
  app.put(
    "/api/posts/:postId",
    [requiresUser, validateRequest(updatePostSchema)],
    updatePostHandler
  );

  // Get a post
  app.get("/api/posts/:postId", getPostHandler);

  // Delete a post
  app.delete(
    "/api/posts/:postId",
    [requiresUser, validateRequest(deletePostSchema)],
    deletePostHandler
  );
}
```

Postman

Create user

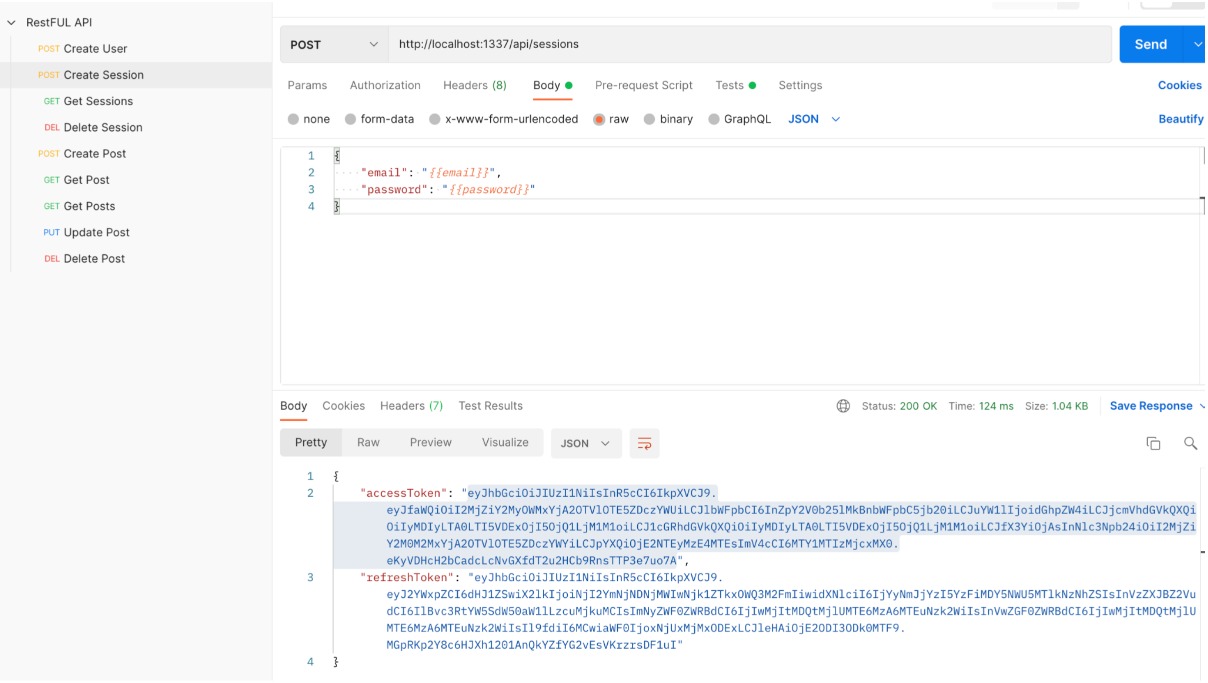


POST http://localhost:1337/api/users

```
{
    "email": "vicetone2@gmail.com",
    "password": "thien1",
    "passwordConfirmation": "thien1",
    "name": "thien"
}
```

200 OK 153 ms 401 B

```
{
    "_id": "626bcc29c1b0695e919d73ae",
    "email": "vicetone2@gmail.com",
    "name": "thien",
    "createdAt": "2022-04-29T11:29:45.353Z",
    "updatedAt": "2022-04-29T11:29:45.353Z",
    "__v": 0
}
```

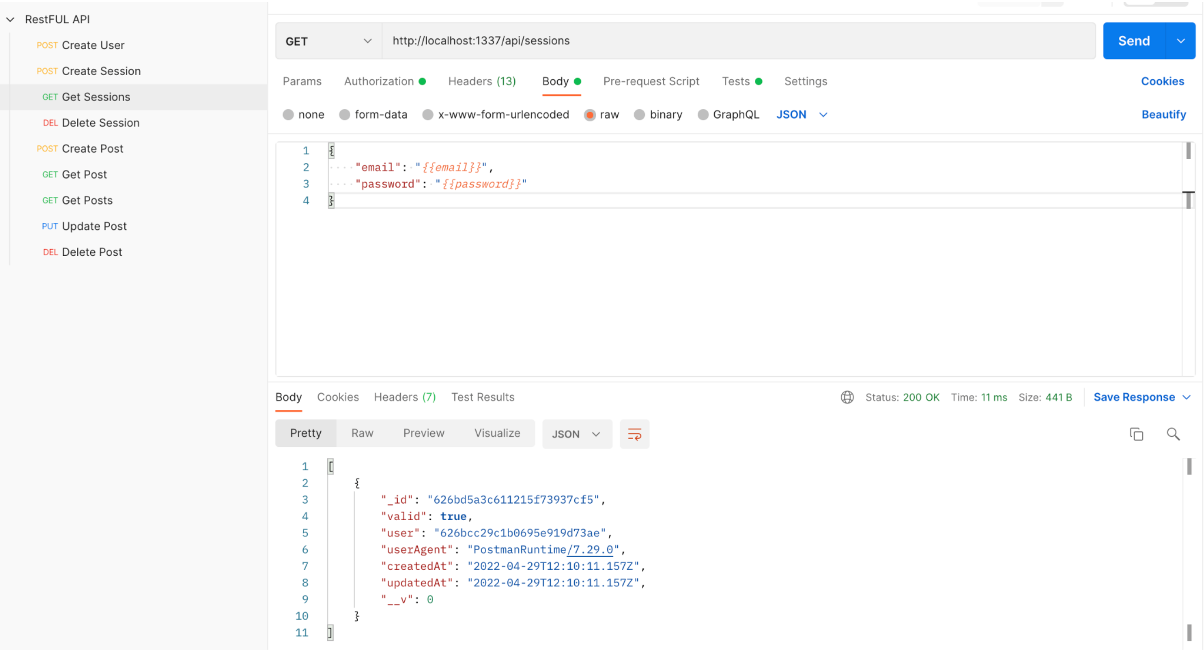


POST http://localhost:1337/api/sessions

```
{
    "email": "{{email}}",
    "password": "{{password}}"
}
```

Status: 200 OK Time: 124 ms Size: 1.04 KB

```
{
    "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
        eyJfaWQiOiI2MjZiY2MyOWMxYjA2OTVlOTE5ZDczYWUiLCJlbWFpbCI6InZpY2V0b25lMkBnbWFpbC5jb20iLCJuYW1lIjoidGhpZW4iLCJjcmVhdGVkQXQi
        OiIyMDIyLTA0LTI5VDExOjI5OjQ1LjM1M1oiLCJ1cGRhdGVkQXQiOiIyMDIyLTA0LTI5VDExOjI5OjQ1LjM1M1oiLCJfX3YiOjAsInNlc3Npb24iOiI2MjZi
        Y2M0M2MxYjA2OTVlOTE5ZDczYWYiLCJpYXQiOjE2NTEyMzE4MTEsImV4cCI6MTY1MTIzMjcxMX0.
        eKyVDHcH2bCadcLcNvGXfdT2u2HCb9RnsTTP3e7uo7A",
    "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
        eyJ2YWxpZCI6dHJ1ZSwiX2lkIjoiNjI2YmNjNDNjMWIwNjk1ZTkxOWQ3M2FmIiwidXNlciI6IjYyNmJjYzI5YzFiMDY5NWU5MTlkNzNhZSIsInVzZXJBZ2Vu
        dCI6IlBvc3RtYW5SdW50aW1lLzcuMjkuMCIsImNyZWF0ZWRBdCI6IjIwMjItMDQtMjlUMTE6MzA6MTEuNzk2WiIsInVwZGF0ZWRBdCI6IjIwMjItMDQtMjlU
        MTE6MzA6MTEuNzk2WiIsIl9fdiI6MCwiaWF0IjoxNjUxMjMxODExLCJleHAiOjE2ODI3ODk0MTF9.
        MGpRKp2Y8c6HJXh1201AnQkYZfYG2vEsVKrzrsDF1uI"
}
```



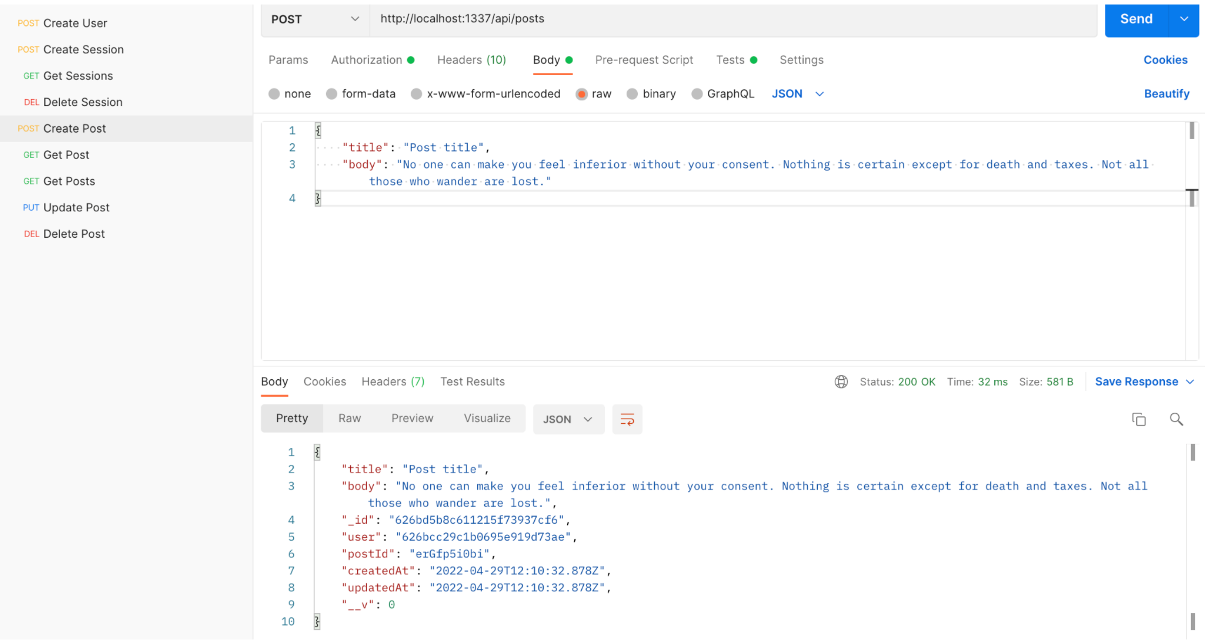
GET http://localhost:1337/api/sessions

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{
····"email":·"{{email}}",
····"password":·"{{password}}"
}
```

Body · Status: 200 OK Time: 11 ms Size: 441 B · JSON

```
[
    {
        "_id": "626bd5a3c611215f73937cf5",
        "valid": true,
        "user": "626bcc29c1b0695e919d73ae",
        "userAgent": "PostmanRuntime/7.29.0",
        "createdAt": "2022-04-29T12:10:11.157Z",
        "updatedAt": "2022-04-29T12:10:11.157Z",
        "__v": 0
    }
]
```



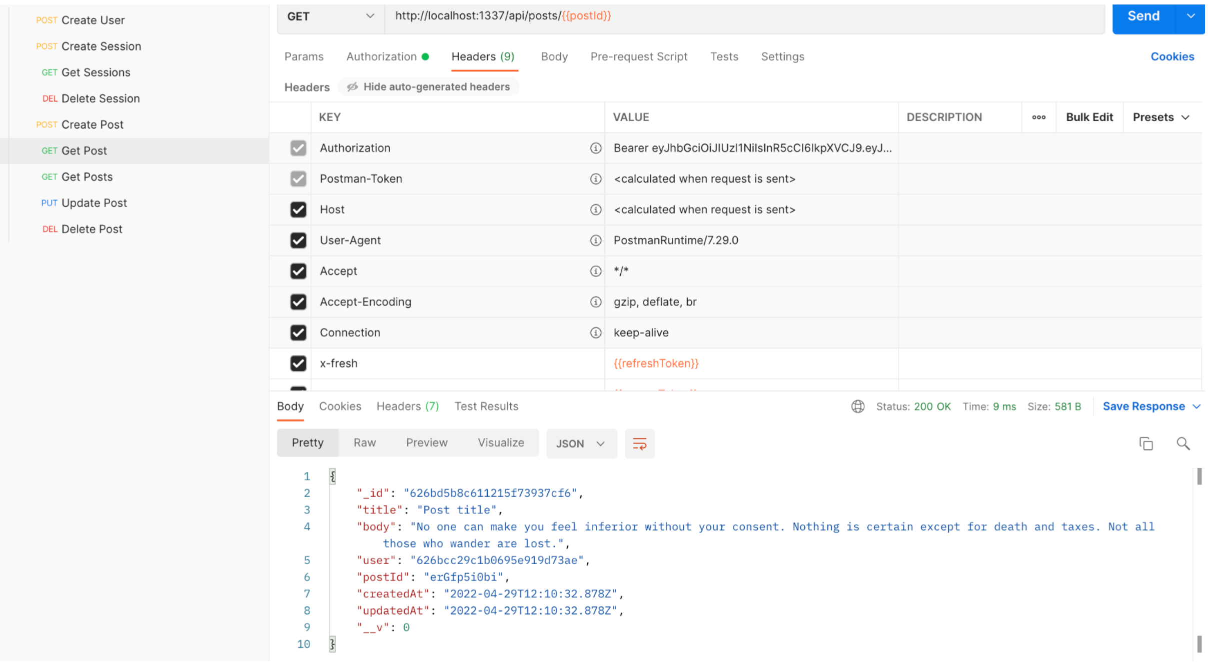
POST http://localhost:1337/api/posts

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{
····"title":·"Post·title",
····"body":·"No·one·can·make·you·feel·inferior·without·your·consent.·Nothing·is·certain·except·for·death·and·taxes.·Not·all·
    those·who·wander·are·lost."
}
```

Body · Status: 200 OK Time: 32 ms Size: 581 B · JSON

```
{
    "title": "Post title",
    "body": "No one can make you feel inferior without your consent. Nothing is certain except for death and taxes. Not all
        those who wander are lost.",
    "_id": "626bd5b8c611215f73937cf6",
    "user": "626bcc29c1b0695e919d73ae",
    "postId": "erGfp5i0bi",
    "createdAt": "2022-04-29T12:10:32.878Z",
    "updatedAt": "2022-04-29T12:10:32.878Z",
    "__v": 0
}
```



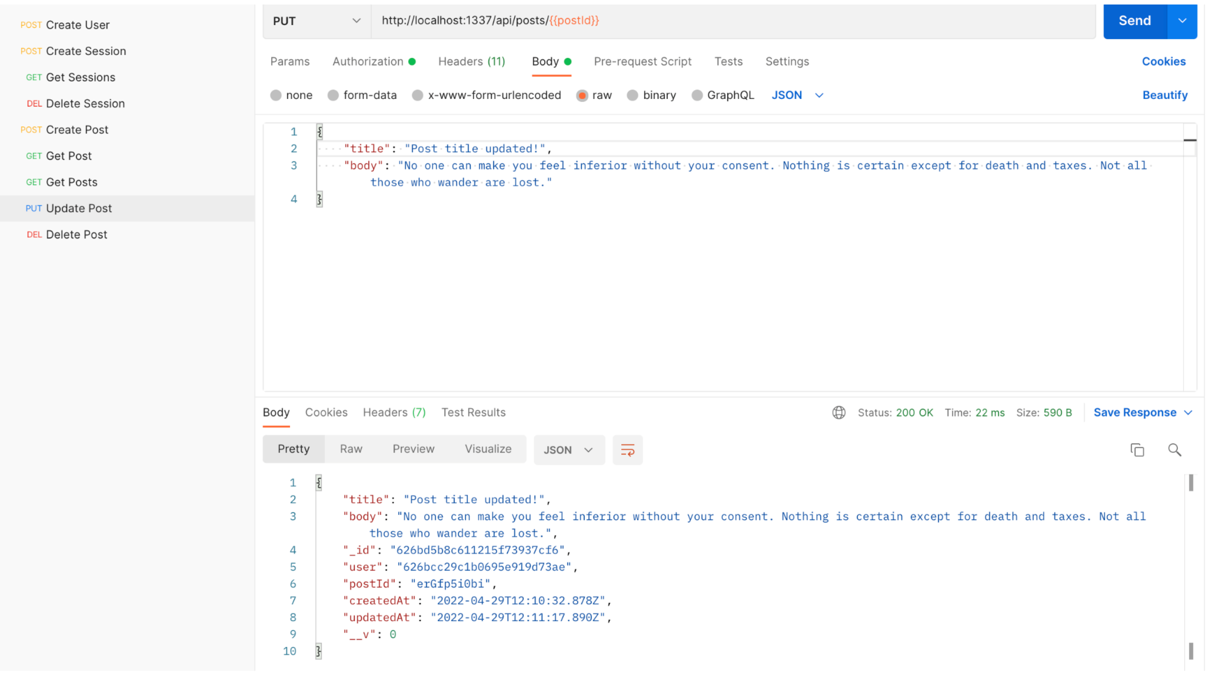
GET http://localhost:1337/api/posts/{{postId}}

Headers

	KEY	VALUE	DESCRIPTION
☑	Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ...	
☑	Postman-Token	<calculated when request is sent>	
☑	Host	<calculated when request is sent>	
☑	User-Agent	PostmanRuntime/7.29.0	
☑	Accept	*/*	
☑	Accept-Encoding	gzip, deflate, br	
☑	Connection	keep-alive	
☑	x-fresh	{{refreshToken}}	

Status: 200 OK Time: 9 ms Size: 581 B

```
{
    "_id": "626bd5b8c611215f73937cf6",
    "title": "Post title",
    "body": "No one can make you feel inferior without your consent. Nothing is certain except for death and taxes. Not all
        those who wander are lost.",
    "user": "626bcc29c1b0695e919d73ae",
    "postId": "erGfp5i0bi",
    "createdAt": "2022-04-29T12:10:32.878Z",
    "updatedAt": "2022-04-29T12:10:32.878Z",
    "__v": 0
}
```



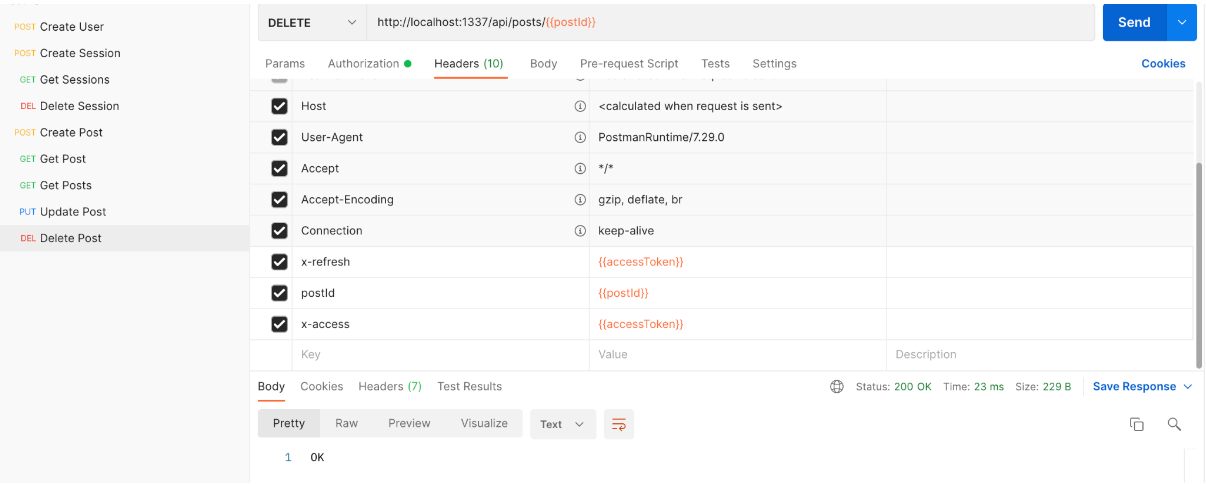
PUT http://localhost:1337/api/posts/{{postId}}

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{
    "title": "Post title updated!",
    "body": "No one can make you feel inferior without your consent. Nothing is certain except for death and taxes. Not all
        those who wander are lost."
}
```

Status: 200 OK Time: 22 ms Size: 590 B

```
{
    "title": "Post title updated!",
    "body": "No one can make you feel inferior without your consent. Nothing is certain except for death and taxes. Not all
        those who wander are lost.",
    "_id": "626bd5b8c611215f73937cf6",
    "user": "626bcc29c1b0695e919d73ae",
    "postId": "erGfp5i0bi",
    "createdAt": "2022-04-29T12:10:32.878Z",
    "updatedAt": "2022-04-29T12:11:17.890Z",
    "__v": 0
}
```

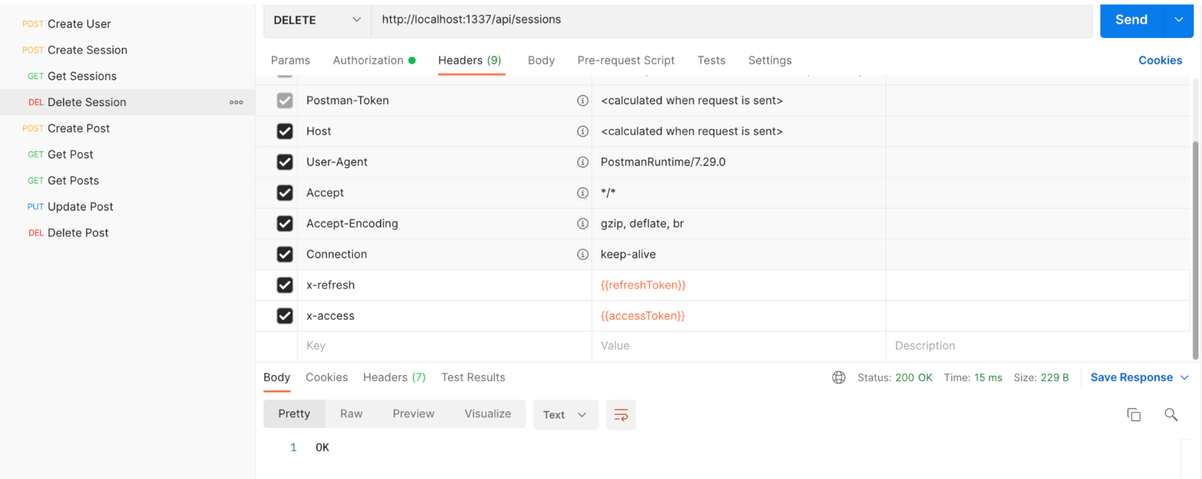


DELETE http://localhost:1337/api/posts/{{postId}}

	Key	Value	Description
☑	Host	<calculated when request is sent>	
☑	User-Agent	PostmanRuntime/7.29.0	
☑	Accept	*/*	
☑	Accept-Encoding	gzip, deflate, br	
☑	Connection	keep-alive	
☑	x-refresh	{{accessToken}}	
☑	postId	{{postId}}	
☑	x-access	{{accessToken}}	

Body Cookies Headers (7) Test Results — Status: 200 OK Time: 23 ms Size: 229 B

```
OK
```



DELETE http://localhost:1337/api/sessions

	Key	Value	Description
☑	Postman-Token	<calculated when request is sent>	
☑	Host	<calculated when request is sent>	
☑	User-Agent	PostmanRuntime/7.29.0	
☑	Accept	*/*	
☑	Accept-Encoding	gzip, deflate, br	
☑	Connection	keep-alive	
☑	x-refresh	{{refreshToken}}	
☑	x-access	{{accessToken}}	

Body Cookies Headers (7) Test Results — Status: 200 OK Time: 15 ms Size: 229 B

```
OK
```

Вывод

Узнал как создать RESTful API средствами express + typescript для создания публикацию.