HECTOR DANIEL VARGAS GONZALES

35-5218 INGENIERIA DE SOFTWARE

LAB-013

Desplegar app en RENDER

Index.js

```
import express from 'express'
import medicRouter from './routers/MedicRouter.js'
const app = express()
app.use(express.json())
app.use('/routers',medicRouter)
app.listen(5000)
console.log('Server on port : ',5000)
```

MedicRouter.js

```
import { Router } from "express";
import {
  getMedics,
  createMedic,
  updateMedic,
  getMedic,
  deleteMedic,
  //getMedicOffices,
} from "../controllers/MedicController.js";

const router = Router();

// Routes
router.post("/", createMedic);
router.get("/", getMedics);
router.put("/:id", updateMedic);
router.delete("/:id", deleteMedic);
router.get("/:id", getMedic);

//router.get("/:id/offices", getMedicOffices);

export default router;
```

MedicController.js

```
import { Medic } from "../models/Medic.js";
//import { Office } from "../models/Office.js";

export async function getMedics(req, res) {
  try {
    const medics = await Medic.findAll({
```

```
      atributes: ["id", "name", "speciality", "phone", "image"],
    });
    res.json(medics);
  } catch (error) {
    res.status(500).json({
      message: error.message,
    });
  }
}

export async function createMedic(req, res) {
  const { name, speciality, phone, image } = req.body;
  try {
    let newMedic = await Medic.create(
      {
        name,
        speciality,
        phone,
//        image: new Date(deliveryDate).getTime(),
        image,
      },
      {
        fields: ["name", "speciality", "phone", "image"],
      }
    );
    return res.json(newMedic);
  } catch (error) {
    res.status(500).json({
      message: error.message,
    });
  }
  res.json("received");
}

export async function getMedic(req, res) {
  const { id } = req.params;
  try {
    const medic = await Medic.findOne({
      where: {
        id,
      },
    });
    res.json(medic);
  } catch (error) {
    res.status(500).json({
      message: error.message,
    });
  }
}
```

```
export const updateMedic = async (req, res) => {
  try {
    const { id } = req.params;
    const { name, speciality, phone } = req.body;

    const medic = await Medic.findByPk(id);
    medic.name = name;
    medic.speciality = speciality;
    medic.phone = phone;
    await medic.save();

    res.json(medic);
  } catch (error) {
    return res.status(500).json({ message: error.message });
  }
};

export async function deleteMedic(req, res) {
  const { id } = req.params;
  try {
    // Primero, eliminamos las tareas asociadas con el médico
   // const deletedTasks = await Task.destroy({
     // where: {
       // medicId: id,
      //},
    //});

    // Luego, eliminamos al médico
    const deletedMedic = await Medic.destroy({
      where: {
        id,
      },
    });

    // Comprobamos si el médico existía y fue eliminado
    if (deletedMedic === 0) {
      return res.status(404).json({ message: 'Medic not found' });
    }

    return res.sendStatus(204);
  } catch (error) {
    console.error(`Error deleting medic with id ${id}:`, error);
    return res.status(500).json({ message: error.message });
  }
}

/*
export async function getMedicOffices(req, res) {
  const { id } = req.params;
  try {
```

```
    const offices = await Office.findAll({
      attributes: ["id", "medicId", "name"],
      where: { medicId: id },
    });
    res.json(offices);
  } catch (e) {
    return res.status(500).json({ message: e.message });
  }
}*/
```

Medic.js

```
import { DataTypes } from 'sequelize';
import { sequelize } from '../database/database.js'; // Ajusta la ruta según la ubicación de
este archivo

export const Medic = sequelize.define(
  'medics',
  {
    id: {
      type: DataTypes.INTEGER,
      primaryKey: true,
      autoIncrement: true,
    },
    name: {
      type: DataTypes.STRING,
    },
    speciality: {
      type: DataTypes.STRING,
    },
    phone: {
      type: DataTypes.STRING,
    },
    email: {
      type: DataTypes.STRING,
      defaultValue: 'xxx@gmail.com' // No recomendable usar un valor predeterminado para
email
    },
    image: {
      type: DataTypes.STRING,
      defaultValue: ''
    },
    services: {
      type: DataTypes.STRING,
      defaultValue: ''
    },
    certifications: {
      type: DataTypes.STRING,
      defaultValue: ''
    },
```

```
    state: {
      type: DataTypes.STRING,
      defaultValue: 'activo'
    }
  },
  {
    timestamps: false,
  }
);
```

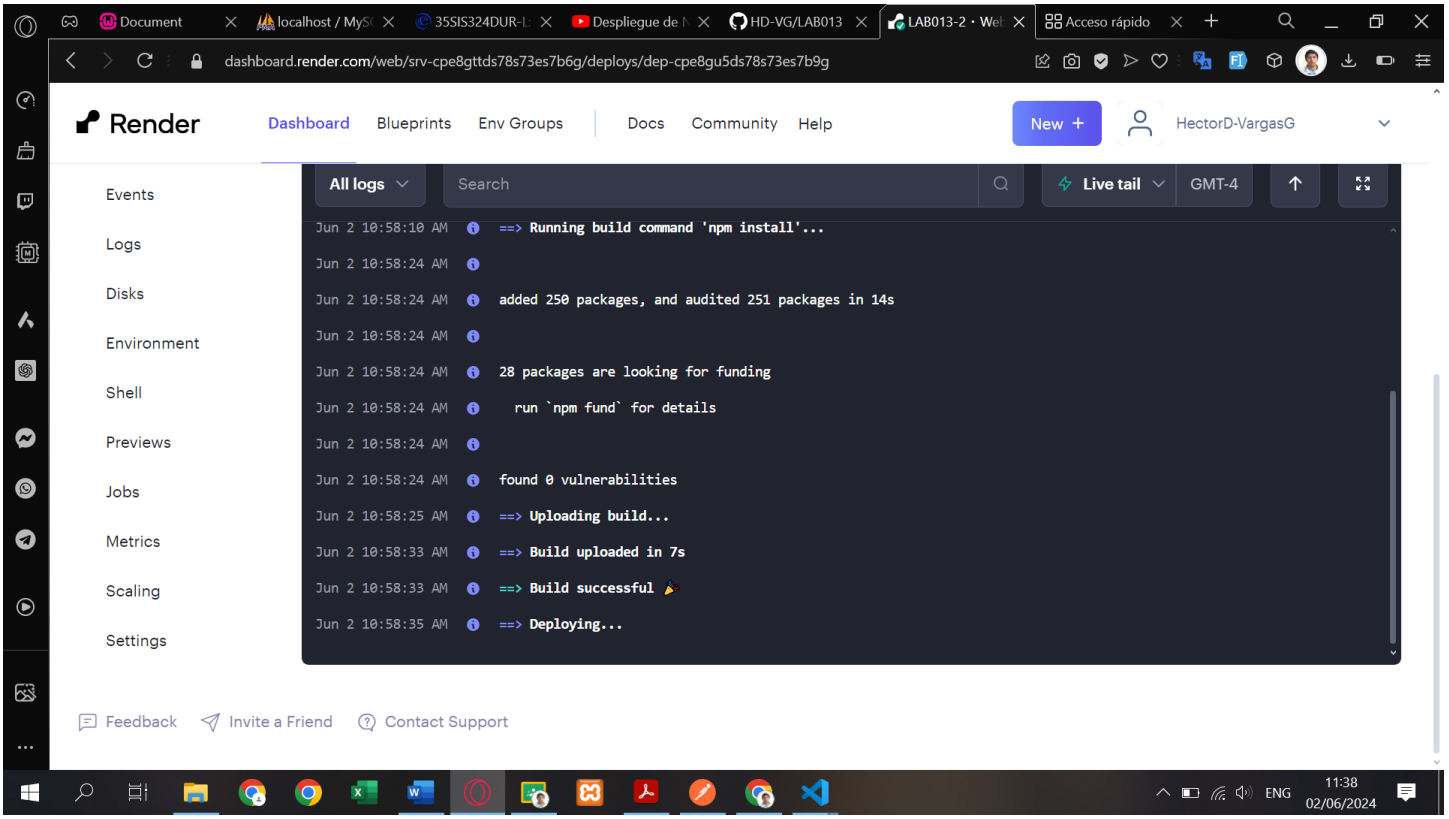
database.js

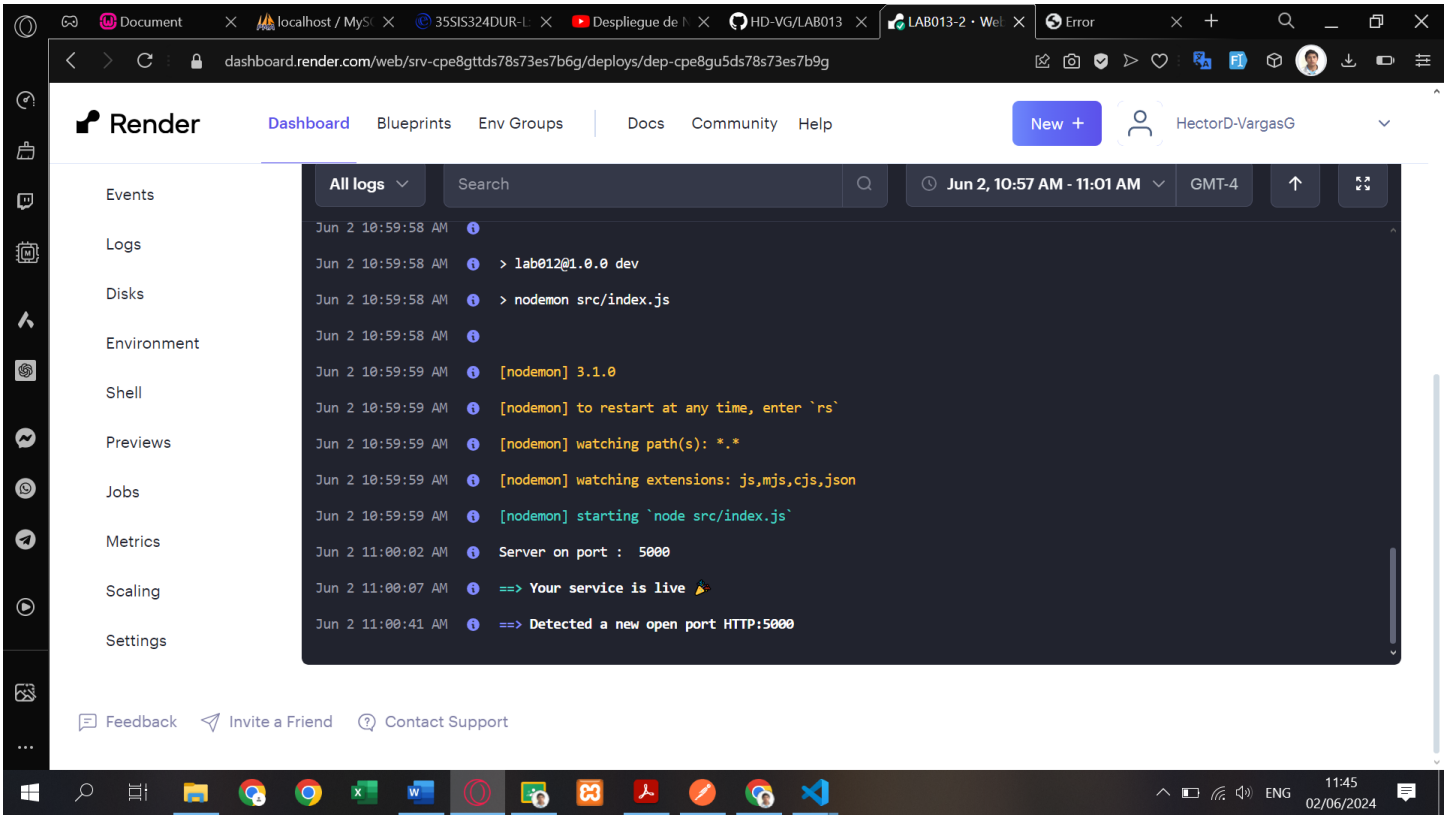
```
import Sequelize from "sequelize";

export const sequelize = new Sequelize({
    dialect: "sqlite",
    storage: '/database/database.sqlite',
  });
```

Sync.js

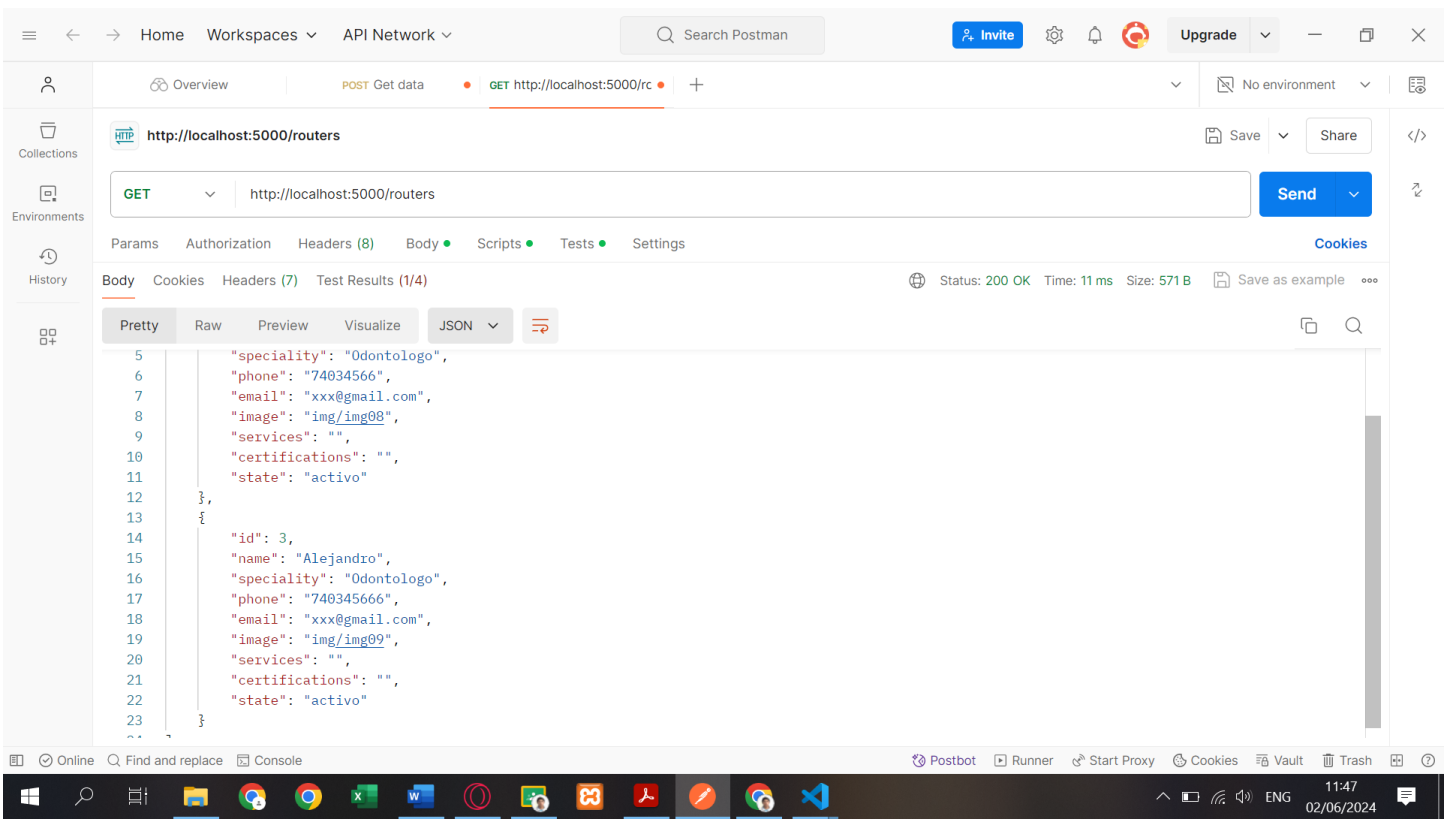
```
import { sequelize } from '../database/database.js'; // Ajusta la ruta según la ubicación de este archivo
import { Medic } from '../models/Medic.js'; // Ajusta la ruta según la ubicación de este archivo

const syncDatabase = async () => {
  try {
    await sequelize.sync({ force: true }); // El parámetro force: true recrea la tabla si ya existe
    console.log("La base de datos y el modelo 'Medic' han sido sincronizados.");
  } catch (error) {
    console.error('Error al sincronizar la base de datos:', error);
  } finally {
    await sequelize.close();
  }
};

syncDatabase();
```

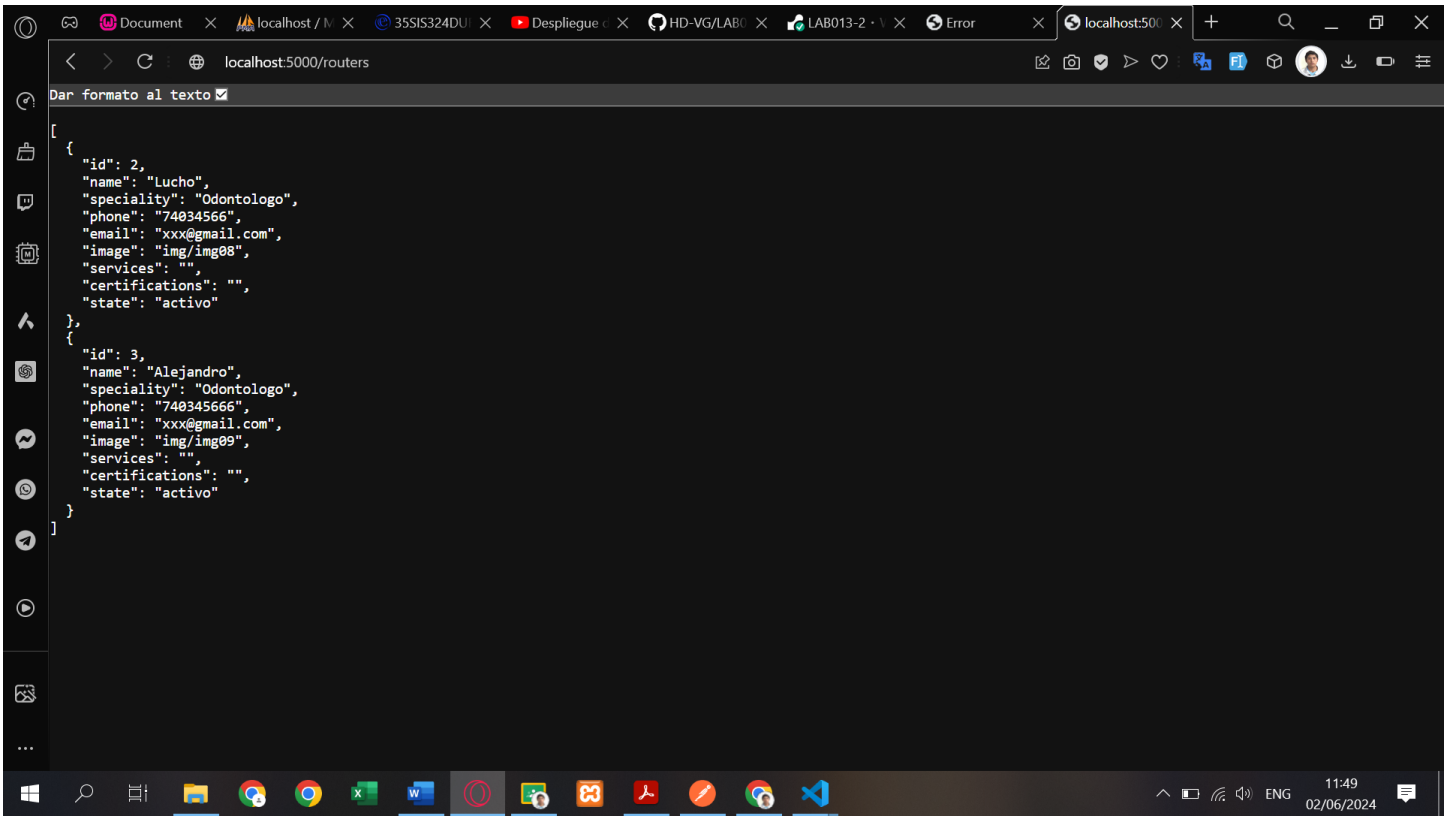
desplegando



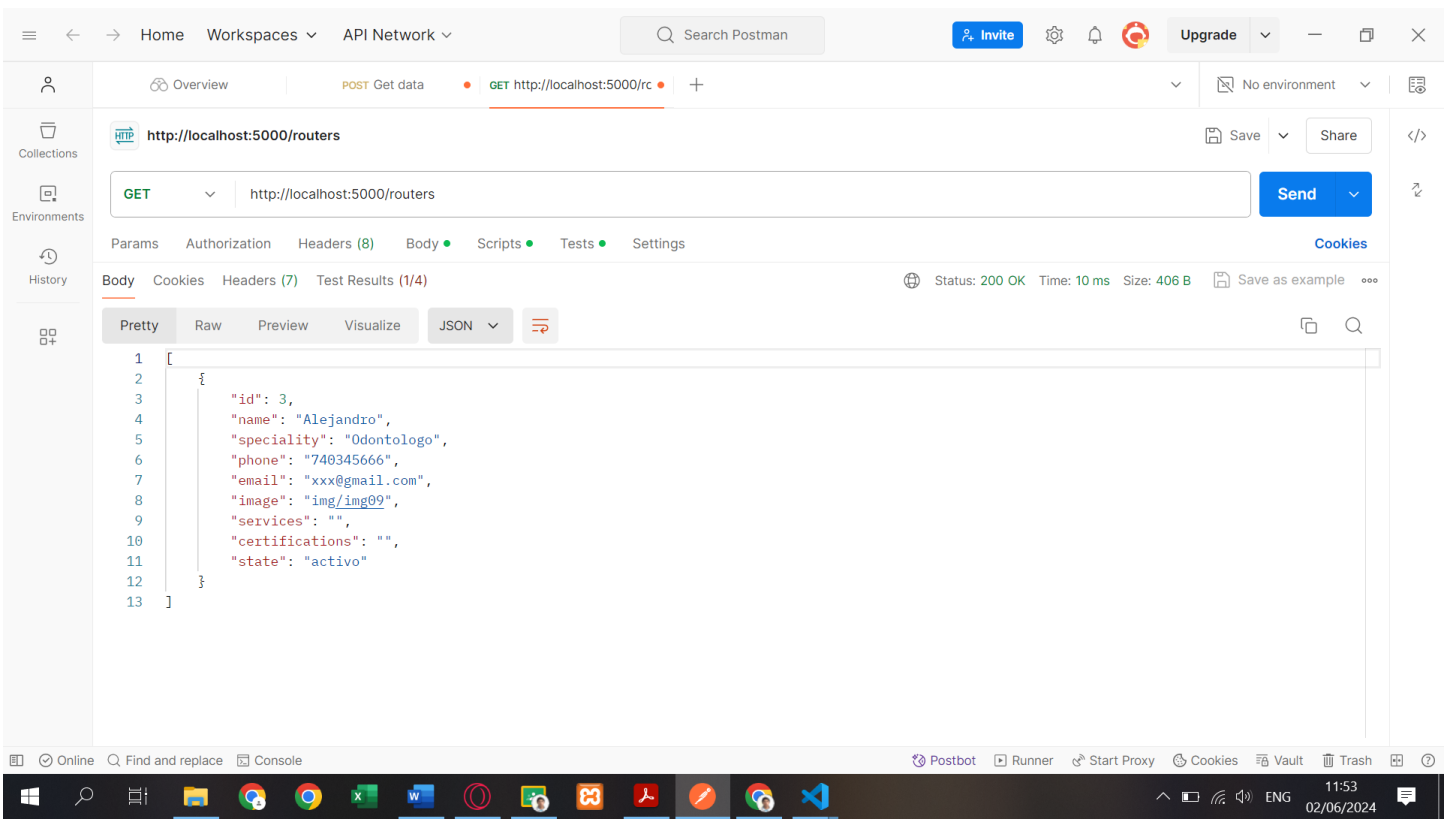


Create medic





Delete medic


Home
Workspaces
API Network
Search Postman
Invite
Upgrade
Overview
POST Get data
GET http://localhost:5000/rc
No environment
Collections
Environments
History
http://localhost:5000/routers
Save
Share
GET
http://localhost:5000/routers
Send
Params
Authorization
Headers (8)
Body
Scripts
Tests
Settings
Cookies
Body
Cookies
Headers (7)
Test Results (1/4)
Status: 200 OK
Time: 10 ms
Size: 406 B
Save as example
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": 3,
"name": "Alejandro",
"speciality": "Odontologo",
"phone": "740345666",
"email": "xxx@gmail.com",
"image": "img/img09",
"services": "",
"certifications": "",
"state": "activo"
}
]
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash
ENG
11:53
02/06/2024