Svi requestovi imaju header `Content-Type: application/json` ako nije drugacije navedeno. Ocekuje se da su datumi u ISO 8601 formatu (2022-12-15T20:29:38.270Z)

/api/auth

POST /api/auth

Request

```
headers: {
    'Content-Type': 'application/www-form-urlencoded'
}
body: {
    username: <email>,
    password: string
}
```

Komentar: mislim da OAuth zahtjeva da se polja zovu tocno username i password, ali mi cemo koristiti email.

Authentication successful response

```
{
    status: 200,
    body: {
        token: <JWTString>
    }
}
```

JWTString - JWT encoding of a User object with the following properties:

```
{
    id: number,
    firstname: string,
    lastname: string,
    role: string
}
```

Authentication failed response

```
{
    status: 401
}
```

POST /api/auth/register

Request

```
body: {
    email: string,
    firstName: string,
    lastName: string,
```

```
    password: string
}
```

Success

```
{
    status: 200
}
```

Failed

```
{
    status: 400,
    body: {
        reason: string
    }
}
```

Example of reason: "email already in use"

/api/user

GET /api/user

Returns a list of all users

```
{
    status: 200,
    body: [{
        id: Number
        firstname: String,
        lastname: String,
        role: String,
        goal: String,
        exercises: [{
            id: Number,
            name: String
        }]
    }]
}
```

GET /api/user/:id

Returns a user with the given id. trainingSessions are the session currently booked for the user.

Success response:

```
{
    status: 200,
    body: {
        id: Number
```

```
        firstname: String,
        lastname: String,
        role: String,
        goal: String,
        exercises: ExerciseID[],
        trainingSessions: TrainingSessionId[],
        trainingTypes: TrainingTypeId[]
    }
}
```

NotFound response:

```
{
    status: 404
}
```

POST /api/user

Creates a new user

Request

```
{
    body: {
        firstname: String,
        lastname: String,
        email: String,
        password: String
    }
}
```

Response

```
{
    status: 201,
    body: {
        id: Number
        firstname: String,
        lastname: String,
        email: String,
        password: String
    }
}
```

DELETE /api/user/:id

Deletes the user with the given id

Response

```
{
    status: 200
```

```
}
```

PUT /api/user

Request

```
{
    body: {
        goal: String
    }
}
```

Response

```
{
    status: 200
}
```

POST /api/user/exercise

Adds an exercise to the users list

Request

```
{
    body: {
        userId: Number,
        exerciseId: Number
    }
}
```

Reponse:

```
{
    status: 200
}
```

DELETE /api/user/exercise

Removes the exercise from the users list

Request

```
{
    body: {
        userId: Number,
        exerciseId: Number
    }
}
```

Reponse:

```
{
    status: 200
}
```

/api/trainingType

GET /api/trainingType

Returns all training types. If the jwt role is `trainer` return only their training types.

Response:

```
{
    status: 200,
    body: [
        {
            id: Number,
            name: String,
            trainerId: Number,
            exercises: [{
                id: Number,
                name: String
            }],
            trainingSessions: TrainingSessionID[]
        }
    ]
}
```

GET /api/trainingType/:id

Returns the training type with the given id

Response:

```
{
    status: 200,
    body: {
        id: Number,
        name: String,
        trainerId: Number,
        exercises: [{
            id: Number,
            name: String
        }],
        trainingSessions: TrainingSessionID[]
    }
}
```

POST /api/trainingType

Creates a new trainingType

Request:

```
{
    body: {
        name: String,
        trainerId: Number
    }
}
```

Response:

```
{
    status: 201,
    body: {
        id: Number,
        name: String,
        trainerId: Number,
        exercises: [],
        trainingSession: []
    }
}
```

DELETE /api/trainingType/:id

Deletes the training type with the given id

Success Response:

```
{
    status: 200
}
```

POST /api/trainingType/exercise/

Add the exercise to the training type

Request

```
{
    status: 200,
    body: {
        trainingTypeId: Number,
        exerciseId: Number
    }
}
```

Success Response:

```
{
    status: 200,
}
```

DELETE /api/trainingType/exercise/

Remove the exercise from the training type

Request

```
{
    status: 200,
    body: {
        trainingTypeId: Number,
        exerciseId: Number
    }
}
```

Success Response:

```
{
    status: 200
}
```

/api/exercise

GET /api/exercise

Gets the list of all the exercises

Response:

```
status: 200,
body: [{
    id: Number,
    name: String
}]
```

POST /api/exercise

Creates a new exercise

Request:

```
body: {
    name: String
}
```

Response:

```
{
    status: 201,
    body: {
        id: Number,
        name: String
```

```
    }
}
```

DELETE /api/exercise/:id

Deletes the exercise with the given id

Success response

```
{
    status: 200
}
```

NotFound response

```
{
    status: 204
}
```

/api/trainingSession

GET /api/trainingSession

Gets the list of all training sessions. If the sent token contains a user object with the *user* role, returns only those sessions the user can register for, based on their exercises. For the trainer role, all existing training sessions are returned. The results should be ordered by startTimestamp.

Response:

```
{
    status: 200,
    body: [{
        id: Number,
        startTimestamp: Date,
        duration: Number,
        capacity: int,
        trainingType: {
            id: Number,
            name: String,
            trainer: {
                id: Number
                firstname: String,
                lastname: String
            }
        },
        users: UserID[]
    }]
}
```

POST /api/trainingSession

Creates a new training session

Request:

```
{
    body: {
        startTimestamp: Date,
        duration: Number,
        trainingTypeID: Number
    }
}
```

Response:

```
{
    status: 201,
    body: {
        id: Number,
        startTimestamp: Date,
        duration: Number,
        trainingTypeId: Number,
        users: []
    }
}
```

DELETE /api/trainingSession

Deletes the given training session

Request:

```
{
    body: {
        id: trainingSessionId
    }
}
```

PUT /api/trainingSession

Updates the trainingSession

Request

```
body: {
    id?: trainingSessionId,
    startTimestamp?: Date,
    duration:? Number,
    trainingTypeId?: Number
}
```

POST /api/trainingSesssion/user

Adds a user to a training session

Request

```
{
    body: {
        trainingSessionId: Number,
        userId: Number
    }
}
```

Response:

```
{
    status: 200
}
```

DELETE /api/trainingSession/user

Removes a user from a training session

Request

```
{
    body: {
        trainingSessionId: Number,
        userId: Number
    }
}
```

Response:

```
{
    status: 201
}
```