CS3219 Task B1, B2, B3

The endpoint can be accessed at https://cs3219-taskb-postgres.herokuapp.com/ Deployment is automated through the use of heroku and travis. Each push into the master branch triggers travis, which runs both CI and CD.

Setting up

1. Create a `.env` file at the root of the directory and copy the content of `.env.example` to `.env` Change the user and password of the database accordingly.
2. Run `npm i` to install the dependencies.
3. Start postgres server
    1. For MacOS: Run `brew services start postgresql` to make sure PostgreSQL is running
4. Run `createdb movies` to create the database for development.
5. Run `createdb movie_test` to create the database for testing.
6. Run `sequelize db:migrate` to run database migration.
7. Run `npm run startdev` to start the server.
8. The deployed API can be accessed at http://localhost:3000/api/movies

Testing

Using Postman

1. Use the postman collection to run the tests in order https://www.getpostman.com/collections/7c4af0a85f99e650494a)
2. Feel free to add your own tests

Local Test

1. Run `npm run test`
2. You should see the CRUD endpoints being tested, with 12 tests in total passing.

```
  Testing the movie endpoints:
Executing (default): INSERT INTO "Movies" ("id","title","price","description","createdAt","updatedAt") VALUES (DEFAULT,$1,$2,$3,$4,$5) RETURNING "id","title","price","description","createdAt","updatedAt";
POST /api/movies 201 37.145 ms - 227
    ✔ It should create a movie (45ms)
POST /api/movies 400 0.620 ms - 62
    ✔ It should not create a movie with incomplete parameters
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM "Movies" AS "Movie";
GET /api/movies 200 2.034 ms - 233
    ✔ It should get all movies
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM "Movies" AS "Movie" WHERE "Movie"."id" = 1;
GET /api/movies/1 200 2.434 ms - 226
    ✔ It should get a particular movie
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM "Movies" AS "Movie" WHERE "Movie"."id" = 8888;
GET /api/movies/8888 404 1.007 ms - 65
    ✔ It should not get a particular movie with invalid id
GET /api/movies/aaa 400 0.150 ms - 65
    ✔ It should not get a particular movie with non-numeric id
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM "Movies" AS "Movie" WHERE "Movie"."id" = 1;
Executing (default): UPDATE "Movies" SET "id"=$1,"title"=$2,"price"=$3,"description"=$4,"updatedAt"=$5 WHERE "id" = $6
PUT /api/movies/1 200 4.238 ms - 153
    ✔ It should update a movie
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM "Movies" AS "Movie" WHERE "Movie"."id" = 9999;
PUT /api/movies/9999 404 1.371 ms - 66
    ✔ It should not update a movie with invalid id
PUT /api/movies/ggg 400 0.252 ms - 65
    ✔ It should not update a movie with non-numeric id value
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM "Movies" AS "Movie" WHERE "Movie"."id" = 1;
Executing (default): DELETE FROM "Movies" WHERE "id" = 1
DELETE /api/movies/1 200 1.720 ms - 46
    ✔ It should delete a movie
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM "Movies" AS "Movie" WHERE "Movie"."id" = 777;
DELETE /api/movies/777 404 0.927 ms - 68
    ✔ It should not delete a movie with invalid id
```

Travis

Tests for travis are triggered by:

1. Installing dependencies and building project
2. Setting environment variables
3. Running tests

```
    ✔ It should get all movies
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM
"Movies" AS "Movie" WHERE "Movie"."id" = 1;
GET /api/movies/1 200 4.853 ms - 226
    ✔ It should get a particular movie
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM
"Movies" AS "Movie" WHERE "Movie"."id" = 8888;
GET /api/movies/8888 404 2.441 ms - 65
    ✔ It should not get a particular movie with invalid id
GET /api/movies/aaa 400 0.306 ms - 65
    ✔ It should not get a particular movie with non-numeric id
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM
"Movies" AS "Movie" WHERE "Movie"."id" = 1;
Executing (default): UPDATE "Movies" SET "id"=$1,"title"=$2,"price"=$3,"description"=$4,"updatedAt"=$5
WHERE "id" = $6
PUT /api/movies/1 200 6.873 ms - 153
    ✔ It should update a movie
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM
"Movies" AS "Movie" WHERE "Movie"."id" = 9999;
PUT /api/movies/9999 404 2.379 ms - 66
    ✔ It should not update a movie with invalid id
PUT /api/movies/ggg 400 0.504 ms - 65
    ✔ It should not update a movie with non-numeric id value
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM
"Movies" AS "Movie" WHERE "Movie"."id" = 1;
Executing (default): DELETE FROM "Movies" WHERE "id" = 1
DELETE /api/movies/1 200 4.433 ms - 46
    ✔ It should delete a movie
Executing (default): SELECT "id", "title", "price", "description", "createdAt", "updatedAt" FROM
"Movies" AS "Movie" WHERE "Movie"."id" = 777;
DELETE /api/movies/777 404 2.166 ms - 68
    ✔ It should not delete a movie with invalid id
DELETE /api/movies/bbb 400 0.259 ms - 61
    ✔ It should not delete a movie with non-numeric id


  12 passing (126ms)

---------------------|---------|----------|---------|---------|-------------------
File                 | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
---------------------|---------|----------|---------|---------|-------------------
All files            |   93.27 |    73.91 |     100 |   93.22 |
 src                 |     100 |      100 |     100 |     100 |
  app.js             |     100 |      100 |     100 |     100 |
 src/config          |     100 |      100 |     100 |     100 |
  config.js          |     100 |      100 |     100 |     100 |
 src/controllers     |   85.36 |    77.08 |     100 |   85.36 |
  MovieController.js |   85.36 |    77.08 |     100 |   85.36 | 13-17,33,54,77,100
 src/models          |    93.1 |    66.66 |     100 |   92.85 |
  index.js           |    91.3 |    66.66 |     100 |    91.3 | 17-18
  movie.js           |     100 |      100 |     100 |     100 |
 src/routes          |     100 |      100 |     100 |     100 |
  index.js           |     100 |      100 |     100 |     100 |
 src/services        |     100 |    69.69 |     100 |     100 |
  MovieService.js    |     100 |    69.69 |     100 |     100 | 4-49
 src/utils           |     100 |      100 |     100 |     100 |
  Util.js            |     100 |      100 |     100 |     100 |
---------------------|---------|----------|---------|---------|-------------------
The command "npm test" exited with 0.
store build cache                                                          cache.2
                                                                                Top
```

Resources:

- https://expressjs.com/en/starter/hello-world.html
- https://medium.com/@dinyangetoh/how-to-build-simple-restful-api-with-nodejs-expressjs-and-mongodb-99348012925d
- https://www.smashingmagazine.com/2020/04/express-api-backend-project-postgresql/
- https://medium.com/@victorsteven/restful-api-with-nodejs-express-postgresql-sequelize-travis-mocha-coveralls-and-code-climate-f28715f7a014