Objective

The objective of this workshop is to perform CRUD operations

Setup

a. Reuse the `boardgames` database from Day 26 Workshop.
b. Create a SpringBoot application with the following dependencies
   i. Spring Boot Dev Tools
   ii. Spring Web
   iii. Thymeleaf
   iv. Spring Data MongoDB
   v. JSON-P

Workshop

You are to create the following REST resources

a. Adding a new review

```
POST /review
Content-Type: application/x-www-form-urlencoded
```

All content must include the following form field

- name – name of the person that posted the rating
- rating – between 0 and 10
- comment – this can be optional
- game id – must be a valid game id from the games collection

Using the form field, the following document will be inserted into the `reviews` collection

```
{
  user: <name form field>,
  rating: <rating form field>,
  comment: <comment form field>,
  ID: <game id form field>,
  posted: <date>,
  name: <The board game's name as per ID>
}
```

Return an appropriate error code if the game id is invalid

b. Updates a given review. A review can only be updated if it has been posted previously. How to do using requestbody?

```
PUT /review/<review_id>
Content-Type: application/json
```

Only 2 fields can be updated

- rating – between 0 and 10
- comment – this can be optional

The endpoint creates the following document from the request

```
{
  comment: <name field>,
  rating: <rating field>,
  posted: <timestamp of the update>
}
```

The updates are stores the `comments` as embedded documents of the `edited` attribute; see example below

```
{
  user: <name form field>,
  rating: <rating form field>,
  comment: <comment form field>,
  ID: <game id form field>,
  posted: <date>,
  name: <The board game's name as per ID>,
  edited: [
    { comment: ..., rating: ..., posted: ... },
    { comment: ..., rating: ..., posted: ... },
    { comment: ..., rating: ..., posted: ... }
  ]
}
```

Return an appropriate error code if the comment id is invalid.

c. The following endpoint retrieves and displays the latest comment and ratings

```
GET /review/<review_id>
Accept: application/json
```

The result is a JSON document as shown below

```
{
  user: <name form field>,
  rating: <latest rating>,
  comment: <latest comment>,
  ID: <game id form field>,
  posted: <date>,
  name: <The board game's name as per ID>,
  edited: <true or false depending on edits>,
  timestamp: <result timestamp>
}
```

d. The history endpoint returns a comment and all its edits if any

```
GET /review/<review_id>/history
Accept: application/json
```

The result is a JSON document as shown below

```
{
  user: <name form field>,
  rating: <latest rating>,
  comment: <latest comment>,
  ID: <game id form field>,
  posted: <date>,
  name: <The board game's name as per ID>,
  edited: [
    { comment: ..., rating: ..., posted: ... },
    { comment: ..., rating: ..., posted: ... },
    { comment: ..., rating: ..., posted: ... }
  ],
  timestamp: <result timestamp>
}
```