Requirements

Return information about students courses from in-memory DB. No need to create any frontend, everything could be tested by making API calls with some kind of "Poster" plugin for your browser. No need to implement any calls to modify data.

Create **Basic**, **Authentication** and **Data** web-services with spring-boot. All of them should be deployable on same host and running at same time. Use latest version of Spring, Java 8, create JUnit tests and Java Docs, and if possible DBUnit tests for **Data** service.

Basic service

Configuration

Configuration file should look like:

```
# Login for authentication web-service
auth.login=auth-admin
# Password for authentication web-service
auth.password=admin1
# Authentication service host
auth.host=localhost:8081

# Login for data web-service
data.login=data-admin
# Password for data web-service
data.password=admin1
# Data service host
data.host=localhost:8082

# Log file
log.file=/path/to/log/service.log
```

This service should support next RESTful requests:

Login request

```
POST /login
```

Adds X-CSRF-TOKEN header to response with generated CSRF token in case of successful authentication.

Errors

401 in case if authentication is failed.
500 in case if *auth.login*/*auth.password* properties doesn't match with **Authentication service** properties.
503 in case if **Authentication service** is unavailable.

Logout request

```
POST /logout
```

Closes session in case if CSRF token matches.

Grades request

```
GET /user/<user-id>
```

Returns information from **Data service** in case if CSRF token matches and is same as used for login.

Errors

404 in case if there is no data found in **Data service**.
500 in case if *data.login*/*data.password* properties doesn't match with **Data service** properties.
503 in case if **Data service** is unavailable.

Authentication service

Configuration

Configuration file should look like:

```
# Service login
login=auth-admin
# Service password
password=admin1

# Log file
log.file=/path/to/log/auth.log

# Users list
users.file=/path/to/users/users.properties
```

Credentials

Service should have file with stored user credentials like:

```
user-has-a-grade = aGrade
user-has-f-grade = fGrade
user-not-in-db = noDB
```

This service should support ONLY request secured by basic authentication.

Authentication request

```
POST /auth
```

```
{
    "user": <user-id>,
    "password": <password>
}
```

Returns 200 if *user*/*password* pair was found in **users.properties** file.

Errors

401 in case if there is no match in **users.properties** file or *password* is wrong.
403 in case if basic authentication was not passed.

Data service

Configuration

Configuration file should look like:

```
# Service login
login=data-admin
# Service password
password=admin1

# Log file
log.file=/path/to/log/auth.log

# Data access
data.access=
```

Data storage

Use any in-memory DB.
Add *user-has-A-grade* and *user-has-F-grade* with couple courses for each one.
Don't add *user-not-in-DB*.

This service should support ONLY request secured by basic authentication.

Data request

```
GET /data/<user-id>
```

Response

```
{
    "user": <user-id>,
    "total": <size of courses list>,
    "courses": [
        {
            "name": <course name>,
            "grade": <A-F>
        }, …
    ]
}
```

Errors

403 in case if basic authentication was not passed.
404 in case if there is no data with given *user-id*.