Videotapes Galore: Project #1

The first project in this course is meant to practice design and implementation of a reasonably sized system, including API design and architecture.

Due: 26.09.2018 (submit before 23:59 in Canvas)
Weight: 20% of final grade

What to hand in?
The solution to the project (i.e. code and design documentation) should be submitted in Canvas as a single ZIP file. It's important that all files are included so the submission can compile and run without requiring external files since the TAs will compile and run the submission as part of determining the grade.

For parts 2 and 3 it's necessary to use the ideas and approaches covered in the lectures. Part of the grade is code structure and code comments, as well as choice of algorithms and data structures.

Note that this project is meant to be an individual project.

We expect students to know university rules, in particular rules regarding plagiarism: **students are expected to submit their own solution**.

1. Videotapes Galore: A Prototype (40%)

Your friend, Áslákur, has a sizable collection of VHS and Beta videotapes that's very popular with his hipster friends and they frequently borrow a tape or two. Since it's important to know who has what, Áslákur wants to create a fancy computerized management system. And, as his best friend, **you** have agreed to help him out.

As a first step you've decided to create a prototype without any up-front design process, to make sure you fully understand the problem domain.

The supported features of the prototype should be the following:

1. Register Áslákur's friends by:
    a. Name
    b. Address
    c. Email
    d. Telephone number
2. Register videotapes by:
    a. Title
    b. Director
    c. Type (VHS or Betamax)
    d. Release Date
    e. EIDR number
3. Register when a friend borrows a videotape:
    a. Identity of the borrower
    b. Identity of the tape
    c. Date when the tape is borrowed
4. Output a report that shows who's borrowing what:
    a. For a given date, output a **list of tapes** that are on loan and who has them
    b. For a given date, output a **list of friends** that have tapes on loan and which tapes
    c. For a given date, output a **list of friends** that have had a tape on loan for more than a month

Note that the prototype should be an interactive command line application, so no real UI is needed. And, since this is a prototype, you can make some simplifying assumptions:

1. The system does not need to store the data once the execution finishes.

2. Once a tape or friend has been registered, there's no requirement to be able to modify the registration.
3. Only a single version of each videotape exists

Initialization

Áslákur already has a JSON file with all his tapes which he gracioiusly has given you access to: https://api.mockaroo.com/api/f34abd10?count=1000&key=e79a3650

He likewise has a JSON file that lists his friends and the videotapes they have borrowed: https://api.mockaroo.com/api/c463f270?count=100&key=e79a3650

Reading this information should be considered as part of system initialization.

2. Programmable Interface (30%)

The prototype that you delivered was quite successful and Áslákur sees potential in continuing development of the system. After writing the prototype you believe you fully understand the problem domain and have decided that the next step is to properly design the system.

The first thing you realize is that you should define a programmable interface (API) for services. The next step is to define the following **services** and provide a REST API for each of them:

1. User management
    a. Create, Read, Update and Delete (CRUD)
2. Videotape management
    a. Create, Read, Update and Delete (CRUD)
3. Management of borrowing
    a. Create, Read, Update and Delete (CRUD)
4. Output a report as specified in the prototype section

Áslákur is a big fan of Netflix and also wants to provide a recommendation service:

5. Review management:
    a. Users can give a videotape a review, ranging from 1 star to 5 stars
    b. Users have full CRUD access to their reviews
6. Recommendation service:
    a. Users can request a real-time recommendation for a videotape they haven't seen already, based on ratings.

Please use Swagger to define the REST API and hand in a **YAML document** that runs and can be tested in Swagger.

3. Overall Design (30%)

Áslákur is happy with the REST API and wants you to deliver the overall architecture and design documents for the system. Áslákur also realizes that three types of users should be supported:

1. Anonymous users (can only view the overall videotape library)
2. Authenticated users that authenticate using the Facebook authentication service (can create, read and modify personal information)
3. Admin users (full access)

For this part deliver the following:

- General description of the system (what, why, how, etc.)
- *Context*, *Container* and *Component Diagram* for the entire system
- *Class Diagram* for the recommendation service
- Schema for the database

The documents should be PDFs and note that all images and diagrams need to have the proper descriptions and references in the text.