SOEN 341-H: Software Process
Fall 2018 - Term project

Dr. Constantinos Constantinides, P.Eng.

September 4, 2018

Contents

1 Introduction

This is a group project which weighs a total of 60% of your overall grade. Please read the entire document very carefully.

In this project you will adopt an iterative process to develop a web application. The details are described in the current document.

You are currently provided with a set of requirements, described in Section 2. You are expected to form teams of 5-7 members, and assign a team leader. The position of the team leader may be rotated among team members, but not during a development iteration.

2 Requirements

This section contains the set of functional and non-functional requirements of the project.

2.1 Overview of the system

A library maintains a catalog of printed and media items. Printed media include books and magazines. Media items include music and videos. With the exception of magazines, all other items are loanable to the library's clients. Your assignment is to develop an on-line library system that supports the following:

Clients use the system to view the library's catalog and search for items. Clients may loan items from the library or return loaned items to the library.

Administrators use the system to enter new records, modify existing records or delete existing records.

2.2 The database

A database holds records of items as well as records of its users as follows:

1. A record for a book contains a title, author, format (paperback or hard cover), number of pages, publisher, year of publication, language, and two ISBNs (a unique identification code), 10- and 13-digit. For example:

```
Title:      Do Androids Dream of Electric Sheep?
Author:     Philip K. Dick
Format:     Paperback
Pages:      240
Publisher:  Del Rey; Reprint edition (Sept. 26 2017)
Language:   English
ISBN-10:    1524796972
ISBN-13:    978-1524796976
```

2. A record for a magazine contains a title, publisher, date of publication, language and ISBN's. For example:

```
Title:     TIME
Publisher: Time (May 13 2008)
Language:  English
ISBN-10:   1603200185
ISBN-13:   978-1603200189
```

3. A record for a movie item consists of a title, director, producer and actor information, language, subtitles, dubbed (can be null if not dubbed), release date and run-time. For example:

Title:	Until the End of the World
Director:	Wim Wenders
Producers:	Anatole Dauman, Ingrid Windisch, Joachim von Mengershausen, Pascale Daum.
Actors:	Bruno Ganz, Solveig Dommartin, Otto Sander, Curt Bois, Peter Falk.
Language:	German
Subtitles:	English
Dubbed:	English, French
Release Date:	Oct. 20 2009
Run Time:	127 minutes

4. A record for a music item consists of a type, release date, title, artist, label and ASIN. For example:

Type:	CD
Title:	Anastasis
Artist:	Dead Can Dance
Label:	Sony Music
Release Date:	Aug. 14 2012
ASIN:	B008F0B124

5. A client record consists of a unique id, first and last name, a physical and email address, and a phone number. No client may hold more than one account or be logged in more than once at any time. Administrators are clients to the database with privileges. There must always exist at least one registered administrator in the system.

2.3 Viewing and searching the catalogs

Clients should be able to view the contents of a catalog, either in a random order, or by creating a result set through a selection of filtering criteria, e.g. "View all movies by director Wim Wenders." Clients may additionally chose the order by which they view a result set: random order, or sort according to some criteria, e.g. "View sorted by date." From a given result set, a client may chose an item to view in detail. In this view, the system should provide an indication on whether the item (if a single copy) is available or is loaned, or, in the case of multiple copies, how many copies are available in the library. The client can subsequently proceed to the next item in detail view, or go back to the (possibly filtered) initial result set view.

2.4 Administering the database

An administrator can enter new records into the database, modify existing records, or delete existing records. An administrator can additionally view the contents of the library catalogs and perform searches, but cannot perform a loan or a return.

The system maintains a registry of its active (i.e. logged) users through a collection of pairs of id and timestamp. Administrators have access to this registry but cannot modify it.

3 Development

This section describes in detail what needs to be done: A high-level development plan, the artifacts to produce (including the running application), and finally some more technical considerations such as object-relational mapping and the deployment of formal methods.

3.1 Plan and schedule

There will be a total of six (6) iterations in the project. All iterations end at 17:00 on the last day. With the exception of the two preliminary iterations (Iterations 0 and 1), the rest are timeboxed into two weeks, by which time your iteration is concluded and your artifacts are under assessment. The development plan with respect to the iterations is shown below:

Iteration 0 / Week 2 / Mon 10 Sep - Fri 14 Sep The team leader should send an email to our teaching assistant, Claudia Della Serra, with a copy to the instructor, in plain text with full names (first, last), id's and emails of each member, indicating their position as the leader of the team.

Iteration 1 / Weeks 3 - 4 / Mon 17 Sep - Fri 28 Sep Set up your environment (See 4.1).

Iteration 2 / Weeks 5 - 6 / Mon 1 Oct - Fri 12 Oct Set up your data model as a relational database and build a database. Depending on your choice of relational database management system, use a built-in tool to define and execute queries. The system allows clients to register and administrators to view the registry of active users. Administrators cannot register themselves, but can manually register other administrators. To support this, you need to start with one hard-coded administrative account.

Iteration 3 / Weeks 7 - 8 / Mon 15 Oct - Fri 26 Oct Administrators can enter items into the catalog (in working memory) as well as modify or delete items from the catalog and may view the inventory but no viewing criteria and no search functionality.

Iteration 4 / Weeks 9 - 10 / Mon 29 Oct - Fri 9 Nov Any modifications to the catalog must now be persisted.

Iteration 5 / Weeks 11 - 12 / Mon 12 Nov - Fri 23 Nov Clients and administrators may view and search the inventory by setting searching and sorting criteria.

3.2 Artifacts and running application

Your software system will consist of the following artifacts that must be developed in line with an iterative process:

- A *Software Requirements Specification* (template available on course website),
- A *Software Architecture Document* (template available on course website),
- Code repository (contains application code, as well as testing, configuration, installation and other supporting implementation), and finally
- A running version, available as a web application.

Your code must be readable, and must follow the naming conventions of the language(s) involved. You may consider the following references:

- Top 15+ Best Practices for Writing Super Readable Code (`https://code.tutsplus.com/tutorials/top-15-best-practices-for-writing-super-readable-code--net-8118`).
- Code Conventions for the Java Programming Language (`http://www.oracle.com/technetwork/java/javase/documentation/codeconventions-135099.html`).
- PHP Coding Standards (`https://make.wordpress.org/core/handbook/best-practices/coding-standards/php/`).
- PEP 8 – Style Guide for Python Code (`https://www.python.org/dev/peps/pep-0008/`).
- JavaScript Style Guide and Coding Conventions (`https://www.w3schools.com/js/js_conventions.asp`).
- C# Coding Conventions (C# Programming Guide) (`https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions`).

3.3 Object-relational mapping

You must develop this system from scratch and in an object-oriented way. You must implement the object-relational mapping manually. Any built-in object-relational mapping technologies that come with some frameworks (see 4.2) must not be used.

4 Organization and management

This section describes the preparation of your development platforms, the expectations on the team formation, recommended technologies, and your overall responsibilities.

4.1 Development platforms

You must organize your development as follows:

Google Docs For the preparation of the SRS and SAD you must use **Google Docs** that supports **Collaborative Document Development**. Team leaders should create a folder titled `SOEN341` (make sure you include a README file with team information) which will contain the two documents. Please note that you must maintain only two documents (SRS and SAD) that will evolve over time, iteration by iteration. The folder should be kept with Link Sharing set to off; this prohibits anonymous changes. All team members should be invited to the folder via an email invite (with permission settings set to organize, add, & edit) and should be logged in to their accounts when editing to properly log all changes.

GitHub For implementation, you must deploy **Version Control** and **Issue Tracking** which can be provided by **GitHub**. Your GitHub repository must be private (GitHub provides free private repositories when you request their free student plan) and it should include a README file with team information.

Send invitations for both Google docs and GitHub to concordiasoen@gmail.com (GitHub username: concordia-soen) and to our teaching assistant, Claudia Della Serra (GitHub username: claudds).

Team formations will be posted on the course website and each team will be assigned a number. Please use this number when you contact the instructor. You may not change a team and you may not break your team or restructure teams, unless you discuss the matter and obtain permission from the instructor (also, see 5.3).

Resources

- Google Docs (`https://www.google.ca/docs/about/`
- GitHub (`https://github.com/`
- Student Developer Pack (`https://education.github.com/pack`)

4.2 Recommended technologies

At the back-end, you must maintain a relational database management system such as one of the following:

- SQLite (`https://www.sqlite.org/`

- MySQL (`https://www.mysql.com/`
- PostgreSQL (`https://www.postgresql.org/`

To develop your server-side application, you have the choice of the following languages, listed below together with recommended frameworks:

- Java with Spring (`https://spring.io/`).
- C# with ASP.NET (`https://www.asp.net/mvc`).
- Python with Flask (`http://flask.pocoo.org/`).
- PHP with Laravel (`https://laravel.com/`).
- node.js (Javascript) with Express (`https://expressjs.com/`).

4.3 Responsibilities

As this is an academic exercise (as opposed to an industrial assignment) where the objective is to learn, **you are all expected to participate in all activities (requirements analysis, architecture and design, coding, testing) on a relatively equal weight**. Failure to do so will result in penalties to the individual(s) involved as well as to the team leader.

For each member of your team, we will monitor the following: a) number of commits, b) number of assigned tasks, c) the total number of tasks, and d) the degree of importance (of committed tasks), where the latter is defined as follows:

0: Unimportant commits, e.g. comments.

1: Minor commits, e.g. changing the names of functions or variables.

2: Important commits, e.g. adding new functions into classes.

3: Very important commits, e.g. such as adding a class.

In other words, the notion of "relatively equal weight" of workload does not exclusively depend on the number of your commits, but on all four factors (a - d) described above.

5 Assessment

This section provides all necessary information about your assessment: Marking, monitoring of your progress, and dealing with slacking cases.

5.1 Marking

A marking scheme is made available to you in an accompanying document. Note that even though this is a group project, marking is not assigned collectively. Based on the criteria described in this document, individual penalties will result in different people from the same team possibly receiving a different project mark.

5.2 Monitoring your progress

Our marker will monitor your account and artifacts that you produce and at the end of each iteration, the marker will send feedback on your progress to your team leader. Please note that the marker's feedback is not meant to initiate a discussion. On the due date and time we will obtain a final image of your workspace. Please do not submit either a hard copy or an electronic copy of your artifacts by using the ENCS electronic submission system.

5.3 Slacking off and voting out

Slackers will be penalized. Accommodating a slacker will result in the team leader receiving a penalty.

As a team, you have the power to vote a team member out by majority vote. In the case of a tie vote, the team leader has the privilege of a casting vote. In the case where the alleged slacker is the current team leader, then the team leader loses their privilege of a casting vote and you (the entire team) must arrange a meeting with the instructor. Note that your vote is not a final decision but a recommendation that you (the entire team) must present to the instructor in the presence of the member you wish out. A team member can be outed only upon approval of the instructor.