UNIVERSITY OF SAN FRANCISCO | College of Arts and Sciences

<NAME OF YOUR PROGRAM/DEPARTMENT/MAJOR OR MINOR>

ASSESSMENT REPORT
ACADEMIC YEAR 2018 – 2019

I. LOGISTICS

1. **Please indicate the name and email of the program contact person to whom feedback should be sent (usually Chair, Program Director, or Faculty Assessment Coordinator).**

   EJ Jung, ejung2@usfca.edu, Chair and Faculty Assessment Coordinator of CS dept.

2. Please indicate if you are submitting report for (a) a Major, (b) a Minor, (c) an aggregate report for a Major & Minor (in which case, each should be explained in a separate paragraph as in this template), (d) a Graduate or (e) a Certificate Program

(d) Graduate

3. Please note that a Curricular Map should accompany every assessment report. Has there been any revisions to the Curricular Map?

No

II. MISSION STATEMENT & PROGRAM LEARNING OUTCOMES

1. Were any changes made to the program mission statement since the last assessment cycle in October 2018? Kindly state "Yes" or "No." Please provide the current mission statement below. If you are submitting an aggregate report, please provide the current mission statements of both the major and the minor program

No changes were made.

The mission of the MS in Computer Science Bridge program is:

To prepare students for Master's in Computer Science at USF who are changing fields from non-computer science backgrounds and to give students who do not have a computer science background enough knowledge to do basic software development.

2. **Were any changes made to the program learning outcomes (PLOs) since the last assessment cycle in October 2017? Kindly state "Yes" or "No." Please provide the current PLOs below. If you are submitting an aggregate report, please provide the current PLOs for both the major and the minor programs.**

Yes. Based on FDCD's feedback, we rephrased the first PLO.

Students who pass the bridge program and proceed to the MS in Computer Science will be able to:

- Application: Implement medium- and large-scale programs in a variety of programming languages.
- Theory: Explain and analyze standard computer science algorithms
- Systems: Describe the interactions between low-level hardware, operating systems, and applications

3. **State the particular Program Learning Outcome(s) you assessed for the academic year 2018-2019.**

Application: Implement medium- and large-scale programs in a variety of programming languages.

III. METHODOLOGY

Describe the methodology that you used to assess the PLO(s).

We used a direct method. CS 514 Object-Oriented Programming is a required course for the Bridge students. One of the learning outcomes of this course is:

At the end of this course, students who successfully completed the class will be able to do all of the following:

- Independently design and code medium-sized object-oriented programs in Java (assessed by homeworks, labs, a final project)

We used the final project from CS 514 to assess if students met this learning outcome. The project description and the rubric are attached.

IV. RESULTS & MAJOR FINDINGS

What are the major takeaways from your assessment exercise?

Course Syllabus: CS514 Object-Oriented Programming

An accelerated introduction to object-oriented programming using Java. Topics include classes and objects, Java collections framework, inheritance and polymorphism, multi-threaded programming, networking and the basics of web development using Jetty/servlets. We will also discuss techniques for debugging and code refactoring.
Prerequisite: Equivalent of one semester of prior programming experience.

General Course Info

Time & Place: MWF 9:55am - 11.40am, Lo Schiavo 307

Instructor: Prof. Olga Karpenko

Office: Harney 403A

Email: okarpenko@usfca.edu

Piazza: Please post your course-related questions on Piazza

Instructor's Office Hours: MWF 12.30-2pm and by appointment

Teaching Assistants:

Pragya Garg, pgarg@dons.usfca.edu, Office hours: Tuesday 2pm-3.30pm,

Pengfei (Perry) Song, psong4@dons.usfca.edu, Office hours: Friday 1.30-3pm.

Office hours will take place in Harney 4th floor, in the room next to CS labs (HR411).

You may also visit the CS Tutoring Center for help with this course.

Course Announcements

Course announcements will be posted on **Piazza**. You are also encouraged to post all your course-related questions on Piazza, where they can be answered by the instructor, the TA or one of your classmates. Please note that you may **not** post solutions (even partial) on Piazza; if in doubt, please ask the instructor. Students are responsible for checking Piazza daily.

Textbook

There is no required textbook for the class. Recommended books:

- Java Software Solutions by Lewis and Loftus (8th or 9th edition). Introductory book for those new to Java. You do **not** need the Programming Lab.

- Java How to Program (Early objects) by Deitel and Deitel (11th edition). Has more advanced topics; best used as a reference.

Links to lecture slides and code samples will be posted on the **Schedule** page. Make sure to check it often.

Level	Percentage of Students
Complete Mastery of the outcome	72% (18/25)
Mastered the outcome in most parts	20% (5/25)
Mastered some parts of the outcome	0% (0/25)
Did not master the outcome at the level intended	8% (2/25)

2 students who did not master the outcome did not pass this course and could not move onto the Master's program. In other words, all students who mastered the outcome in most parts successfully completed the assessed PLO.

V. CLOSING THE LOOP

1. Based on your results, what changes/modifications are you planning in order to achieve the desired level of mastery in the assessed learning outcome? This section could also address more long-term planning that your department/program is considering and does not require that any changes need to be implemented in the next academic year itself.

It is assuring that the students who fulfill this course's learning outcome (and pass this course) fulfills the program's learning outcome properly and continue to the Master's program. This is the second cohort of the Bridge program, and so far students who successfully finished our Bridge program are successfully making progress in their Master's Program. We will continue to track their achievements and revise the curriculum if necessary.

2. What were the most important suggestions/feedback from the FDCD on your last assessment report (for academic year 2016-2017, submitted in October 2017)? How did you incorporate or address the suggestion(s) in this report?

Based on the FDCD's feedback, we rephrased one of the PLOs to use an active verb.

Learning Outcomes

At the end of this course, students who successfully completed the class will be able to do all of the following:

- Independently design and code medium-sized object-oriented programs in Java (Assessed by homeworks, labs, a final project)

- Understand intermediate / advanced Java concepts (Assessed via quizzes and exams)

- Use external Java libraries (Assessed by labs and a final project)

- Produce professional-quality code Java (Assessed by homeworks, labs and a final project)

- Design and execute tests to identify and repair software bugs, redesigning and refactoring code when necessary (Assessed by labs and a final project)

- Analyze and critique code written by others (Assessed by exams and labs)

Exams

There will be two midterm exams held in class (Oct 3 and Nov 7) and a 2 hour final exam. The exams will be written and closed notes. **If a student gets <60% on any two out of the three exams, the student will get an automatic F for the class.**

Make-up exams will be given only in the case of a medical emergency verified by a doctor's note.

Homeworks, Labs, Projects and Quizzes

Homeworks are short/medium programming assignments given *weekly*.

Labs are programming assignments of more substantial size; each lab would take you about 2-3 weeks to complete and you should work on them in parallel with your homework assignments.. In this class, you will be given 3-4 medium-sized labs throughout the semester.

Project You will work on one large capstone **project** submitted in 2-3 releases. Project and labs are graded on both functionality (85%) and design (15%). Quality of code matters a lot in this class! A project and some labs will be graded interactively by the instructor and the TAs.

All programming assignments must be uploaded to github by 11:59pm on the due date. No email or Canvas submissions will be accepted. If your code is not in the correct github repo, you will get a 0 for the assignment even if you completed it by the deadline and can prove it using local history on your computer. Please also note that solutions that do not compile will be assigned a 0 (no partial credit). Double check that your code compiles and runs before submitting it to github!

The instructor and the TAs are happy to help you with the assignments during office hours.

Quizzes will be given weekly/bi-weekly during the semester; please note that they may be given unannounced. If you are not in class on the day of the quiz, you will receive a 0 for the quiz unless you have a verifiable medical emergency.

Grade Breakdown

The final grade for this course will depend on a mix of projects, labs, quizzes and exams. The specific breakdown is as follows:

Assignment Group	Weight
Homeworks	10%
Labs	15%
Final Project	25%
Quizzes	10%
Midterm exams	20% (10% each)
Final exam	20%

Extra credit will be given at the end of the semester for class participation (asking questions, responding to questions, leading discussions). These extra credit points will be applied to the Quizzes group.

Interactive code walk-throughs

An in-person code review will be required for all releases of the final project and may be required for any assignment. A student who is unable to explain his/her code and answer the instructor's questions about his/her code as expected will receive a 0 on the assignment, even if it was graded earlier by the TA . A student who does not meet with the instructor for a code review in a timely manner will be given a 0 on the assignment.

Letter Grades

Letter grades will be assigned according to the following APPROXIMATE straight scale:

A ≥ 94%
A– ≥ 90%
B+ ≥ 87%
B ≥ 84%
B– ≥ 80%
C+ ≥ 77%
C ≥ 74%
C– ≥ 70%
F < 70%

Please note this scale is subject to change.
See http://www.usfca.edu/catalog/regulations/student/#497495 for more information about letter grades and how they are translated into GPA.

Attendance

Students are expected to be on-time to all classes. Attendance is mandatory for all exams, quizzes and labs. Topics that are discussed in class but are not in the lecture notes, might be included in the midterms and final.

Late Policy

All deadlines and exam dates are firm. Lab, project assignments, quizzes and exams will **not** be accepted if submitted after the deadline.

Exceptions to this policy are made only in the case of *verifiable* medical or family emergency. Extensions and makeup exams must be arranged PRIOR to the original deadline unless in case of extreme emergency (such as an emergency room visit).

Academic Honesty

All assignments are to be completed individually. Cheating will **NOT** be tolerated.

All students are expected to know and adhere to the University of San Francisco's Honor Code. Go to http://myusf.usfca.edu/academic-integrity/honor-code for details. The first violation of the Honor Code will result in an automatic 0 on the offending assignment and a report to the Dean's office. Repeat violations will result in an automatic F for the course.

Any student may be asked to reproduce any of his/her work at any time. Failure to reproduce work in a timely manner will be considered academic dishonesty.

Students may:

- Receive help from the professor, the TA or the tutors in the CS Tutoring center. *However, you must always fully understand and "own" your code, and be able to reproduce it, if needed*. If you got so much help from the TAs or the tutors that you no longer understand your code, it is considered cheating.

- Discuss the code examples posted by the instructor and general requirements of the assignments with other students or outside sources. However, if as a result of your discussions with another student, your programs ended up looking very similar, it means you discussed low level implementation details and it would be considered cheating. If you have any doubt with respect to what is acceptable to discuss, speak with the professor first.

Students may NOT:

- look at another student's code.

- look at another student's solutions to assignments

- discuss implementation details of an assignment with another student

- receive unapproved help from an outside source including a tutor or a family member.

- submit code which has, in whole or in part, been copied from *any* other source (including another student, a web page, or a textbook). Even if you reference the source, it is still *not* allowed.

CS Tutoring Center

The CS department has a great resource available to CS students: CS Tutoring center, located in HR413, provides **free** tutoring/class support for students.

Check the current schedule to see when the tutors are available:

http://tutoringcenter.cs.usfca.edu/calendar/ (Links to an external site.)

Student Disability Services

If you are a student with a disability or disabling condition, or if you think you may have a disability, please contact Student Disability Services (SDS) within the first week of class to speak with a disability specialist. If you are determined eligible for reasonable accommodations, your disability specialist will send your accommodation letter to the instructor detailing your needs for the course. For more information, please visit http://www.usfca.edu/sds or call (415) 422-2613.

Note: This syllabus is subject to change.

CS514 Project: Web-based Movie Recommender

Due Date for Release 1: November 30, 11:59pm

Due Date for Final Release: Dec 7, 11:59pm

The goal of the project is to create a website for recommending movies, where users can register, login, create a profile by rating movies and specifying preferred movie genres, get movie recommendations (and anti-recommendations) based on the user profile, access movie info, and perform several other operations described later in this document.

Required Features for Release #1

For Release #1, you must complete all of the following required features for a total of 40 points.

Feature	Description	Points
Html Templates	Create html templates for the project using Bootstrap 4 components from w3schools.com to give your webpages a clean look. I recommend looking at the following components (although specific look is up to you): BS4 Tables, BS4 Inputs/BS4 Forms, Glyphicons, BS4 Basic DropDown, or BS4 Navbar. Do not use any ready-made templates from any website, including w3schools.	**10 pts**
Template Engine	Use Velocity template engine to generate html in all servlets.	**10 pts**
Improve Recommendation Algorithm	Improve the movie recommendation (and anti-recommendation) algorithm developed in lab 3 by taking into account the preferred movie genres (and possibly other information, as you see fit). Allow to pass an argument that specifies how recommendations should be sorted: by year, by "popularity" (how many users total rated this movie) etc.	**5 pts**
Create User Profile	A user should be able to create a profile, where they enter the name and the password; the number of recommendations and anti-recommendations they want to see; specify two preferred movie genres, and rate a subset of movies. User's ratings should be added to the existing data structure that holds ratings for all users.	**5 pts**

	Note: this feature is similar to lab 5, but you should use your own html template and Velocity.	
Search Movies	The user should be able to search for movies by title and rate them. User's profile and the ratings data structure for all users should be updated accordingly.	**5 pts**
View Recommendations and Anti-Recommendations	Allow a user to view movie recommendations and anti-recommendations computed based on the existing dataset and user's profile. Note: this feature is similar to lab 5, but requires you to use your own html template and Velocity.	**5 pts**

Required Features for Final Release (it should also include all features from part 1 of the project). Total 50 points.

Feature	Description	Points
User Registration	"*Register*" button that allows a user to register on the website. You need to check that the username has not already been taken, and that the password satisfies a set of reasonable requirements (ex: not too short, contains at least one special character etc.) User's info **(username, hashed salt, hashed password+salt**) should be stored in the MySQL database (you should access the database using JDBC as will be shown in class). Password should be salted and hashed.	**5 pts**
Login and Logout	Allow a user to login and logout. You need to maintain the session properly (using HttpSession class or Cookies). Your code should use JDBC to access the user account info in MySQL database.	**10 pts**
View/Edit Profile	The logged-in user should be able to view and edit her/his profile (if they are logged in). The webpage should display their name, preferred genres, and movies they rated with the corresponding ratings. The user should be able to edit genres or movie ratings.	**10 pts**
Movie Info Page	When the user clicks on View Recommendations or View Anti-Recommendations (available only when they are logged in), your website should return a table of movies, where the title of each movie is a *link*. When the user clicks on the movie link, she/he is taken to the dynamically generated *Movie Info page* for this movie that displays movie title, release year, genres, and contains a link to the imdb webpage for this movie (please refer to *Links Data File Structure section* of this README that talks about mapping movieIds to imdb ids: http://files.grouplens.org/datasets/movielens/ml-latest-small-README.html).	**10 pts**

Saved Movies	Provide an option to "save" a specific movie to a list of "Saved Movies" - the movies that the user is considering watching. Allow the user to clear that list. This feature should only be available to logged in users.	**10 pts**
Last Login	Store and display the last date&time the user logged in to the website before this time (the feature is available only for logged in users). **Example:** Consider the following situation: I login at 6am on Dec 15, 2016, then logout, then login at 8am on the same day again. When I login at 8am, the website should say "Last Login : 6:00am, 12:15:2016". If I refresh the page without logging out, it should still say "Last Login : 6:00am, 12:15:2016".	**5 pts**

General Requirements:

- You are required to implement both the front end and the back end.
- Your code should be in **Java.** You are required to use Jetty/servlets/JDBC for the project.
- You may reuse code from your labs; just keep in mind that you will most likely need to modify it for the project. You may also use code from the examples I showed in class, but you need to completely understand every line of code in your project (saying "I am not sure how it works, I just copied it from your example" will result in a 0 for the project).
- By default, you may **not** use any third-party libraries except Velocity. You are *not* allowed to use DataTables jQuery plugin or similar plugins or libraries. If you would like to use a particular language, library or tool, please make sure to get the instructor's approval first. You are not allowed to use any framework.
- MySql Database: You are required to use the database assigned to you.
- Your web pages should have a clean look. The html templates should be prepared using individual Bootstrap 4 components from https://www.w3schools.com/bootstrap4/default.asp. Please do not use ready-made html templates from any website.
- You are required to use a template engine for the project (Velocity or ThymeLeaf). The instructor will only be able to provide help with Velocity.
- Before moving onto part 2 of the project, each student should go through an interactive code review for part 1 of the project. The instructor and the TAs will also grade the final project interactively.
- During the demo & interactive code review, it is your responsibility to demonstrate to the instructor or the TA that each feature works properly. You may **not** receive credit for the features you don't demonstrate at that time. Please practice showing a demo beforehand and come prepared. Make sure to show how your features handle error cases (e.g., login of a user that does not exist).

Javadoc Comments:

You are required to add javadoc comments to your code (above each class and above each public method).

Design:

It's important for your project to have good object-oriented design. I encourage each student to discuss their design with the instructor.

Submission:

1. Your project code should be submitted to your private project-username repo on github before the deadline. Please keep pushing your code to github as you work on the project. I expect to see at least 8 commits for each part of the project made over the course of at least 4 days. No history on github except for the final submission will result in a 0 for the assignment.
2. Both releases will be graded interactively - you must make an appointment with the instructor or the TA to demo your project by the given deadline.