JOHN ABBOTT
CEGEP/COLLEGE

Centre for Continuing Education

420-P43-AB

Project: EJBs and Frameworks

Project: 100 Marks; Worth: 20% **Due: July 10th, 2018**

Part I. Main login and steps 35 Marks
Part II. User Login and steps 15 Marks
Part III. Database tables and formation 10 Marks
Part IV. Extra 15 marks for MVC, GitHub and Coding conventions as mentioned in instructions
Part V. Presentation and demo 25 Marks

Late projects are not accepted.

Note:

1. You can add more information if you want to learn more but there are no extra marks for extra work done. That is for your learning.
2. Use any database of your choice but always keep a backup of database queries that you will use for the first time like create table queries and insert queries for adding some data.
3. You have to present your application by giving a demo to the Client. Make sure to impress the client.
4. Your page should have clear information displayed to the Client. For example: Page should display clear information, don't show everything in a single line. Also, you can use color coding in your HTML pages but again no extra marks.

Presentation:(25 Marks; presentation and demo)

Time allocated for presentation is no more than 20Mins and for demo of your project is no more than 15Mins.

You are supposed to create a presentation for the project. Every team member is supposed to take part in the presentation. Presentations can have

a. your team name, team leader and team information
b. any important information
c. slides on standard that you followed, MVC, GitHub
d. critical problems you have faced,
e. lessons/learnings etc. **Add more slides** as you feel so.

Extra marks: (15 Marks; 5 each)

- ✓ Follow MVC Architecture as much as possible. Your code should be clean enough to understand.
- ✓ Create GitHub account and submit your code on GitHub. Submission on GitHub should not be 1 time. Purpose for GitHub use is to create a team, add team members as collaborators on GitHub, work parallelly on the project by doing GitHub checking and check-out.
- ✓ Follow coding conventions and naming scheme. This should be same throughout your project. Add a slide in your presentation for the standard you are going to follow.

Submission: Submit your project online on Léa. Only one team member from the team will submit the project. File name should be teamname.zip. This zip file will contain presentation, exported project in archive format. **Deadline to submit the project is July 10th, 2018.**

1. Library Management

First page or welcome page as for login. Admin and users can login from this page.

Part I. For admin: default login admin/admin (35 Marks)

- Create a page in which admin can perform multiple(CRUD) operations on a book by connecting it to the database. For example, CRUD for book as create, read, update and delete a book.

  For example: You can have list of books and four buttons next to it, or you can have 4 pages for each operation. It is up to you to decide. You can search a book by its ID and then perform operation on that. After clicking submit it should get updated in the database.

- Create an inventory page which will show all the items in a table format. (HTML table tags) which will have a table header tag Book ID, Book Name and Book Author.

- Create a page where admin can add/read/delete/modify(CRUD) users to the database. User information stored is username, password, first, last name, city and postal code. Similar to the book crud operations.

- Create an inventory page which will show all the items in a table format. (HTML table tags) which will have a table header tag for User ID, username, first and last name combined, city and postal code.

Part II. For user (15 Marks)

- Same login page can be used for the user as admin, but user login should be made successful by validating user information from the database
- Create a page where user search for a book with bookname, if book is present then show the information of the book but if not display error message saying no book found, try again. (Searching should provide results even partial match case)

Your pages should have navigation to each other. For example, from list of users there can be a button/link for logout that sends back to home page where user can login again.

You will probably need database tables for the following:(10 Marks)

user: create attributes called as userID (possibly Auto Incremented), username, password, firstname, lastname, city, postal code

library: create attributes called as bookID (possibly Auto Incremented), bookname, authorname.

2. School Management

First page or welcome page as for login. Teacher and student can login from this page.

Part I. For teacher: default login teacher/teacher (35 Marks)

- Create a page in which teacher can perform multiple(CRUD) operations on a book by connecting it to the database. For example, CRUD for book as create, read, update and delete a book.

  For example: You can have list of books and four buttons next to it, or you can have 4 pages for each operation. It is up to you to decide. You can search a book by its ID and then perform operation on that. After clicking submit it should get updated in the database.

- Create an inventory page which will show all the items in a table format. (HTML table tags) which will have a table header tag Book ID, Book Name and Book Author.

- Create a page where teacher can add/read/delete/modify(CRUD) students to the database. Student information stored is studentname, password, first, last name, city and postal code. Similar to the book crud operations.

- Create an inventory page which will show all the items in a table format. (HTML table tags) which will have a table header tag for Student ID, studentname, first and last name combined, city and postal code.

Part II. For student (15 Marks)

- Same login page can be used for the student as teacher, but student login should be made successful by validating student information from the database
- Create a page where student search for a book with bookname, if book is present then show the information of the book but if not display error message saying no book found, try again. (Searching should provide results even partial match case)

Your pages should have navigation to each other. For example, from list of students there can be a button/link for logout that sends back to home page where student can login again.

You will probably need database tables for the following:(10 Marks)
student: create attributes called as studentID (possibly Auto Incremented), studentname, password, firstname, lastname, city, postal code
library: create attributes called as bookID (possibly Auto Incremented), bookname, authorname.

3. Office Management

First page or welcome page as for login. Boss and employees can login from this page.

Part I. For boss: default login boss/boss (35 Marks)

- Create a page in which boss can perform multiple(CRUD) operations on a book by connecting it to the database. For example, CRUD for book as create, read, update and delete a book.

  For example: You can have list of books and four buttons next to it, or you can have 4 pages for each operation. It is up to you to decide. You can search a book by its ID and then perform operation on that. After clicking submit it should get updated in the database.

- Create an inventory page which will show all the items in a table format. (HTML table tags) which will have a table header tag Book ID, Book Name and Book Author.

- Create a page where boss can add/read/delete/modify(CRUD) employees to the database. Employee information stored is employeename, password, first, last name, city and postal code. Similar to the book crud operations.

- Create an inventory page which will show all the items in a table format. (HTML table tags) which will have a table header tag for Employee ID, employeename, first and last name combined, city and postal code.

Part II. For employee (15 Marks)

- Same login page can be used for the employee as boss, but employee login should be made successful by validating employee information from the database
- Create a page where employee search for a book with bookname, if book is present then show the information of the book but if not display error message saying no book found, try again. (Searching should provide results even partial match case)

Your pages should have navigation to each other. For example, from list of employees there can be a button/link for logout that sends back to home page where employee can login again.

You will probably need database tables for the following:(10 Marks)
employee: create attributes called as employeeID (possibly Auto Incremented), employeename, password, firstname, lastname, city, postal code
library: create attributes called as bookID (possibly Auto Incremented), bookname, authorname.