CMPS 350 Project Phase 1 – WebApp UI Design and Implementation

Conference Management System (ConfPlus)

(15% of the course grade)

The project phase 1 submission is due by 8am Tuesday 28th March 2023. Demos during the same week

1. Requirements

You are required to design and implement a Conference Management System named ***ConfPlus*** to manage the process of organizing and running an academic conference. It should be designed to meet the needs of all stakeholders involved in the conference, including **organizers**, **authors**, and **reviewers**.

You can familiarise yourself with the various notions around the management of an academic conference by visiting some important conferences such the Web Conference, the International Conference on Software Engineering (ICSE).

The app use cases include:

Use Case 1: Login

It allows users to login to use the app using their email and password.

A sample list of organizers, reviewers and authors is provided in **`users.json`** file on GitHub under the *project* subfolder. To keep the app simple, there is no need for the users to register to create an account to use the app.

DHIA

Use Case 2: Submit a paper

Precondition: User is logged in as an author and has a paper to submit.

- Author enters the paper details including the paper title and abstract and authors (including name, email and affiliation). One of the authors could be marked as the Presenter.
- Author uploads the paper pdf and submits the paper for review.
- The paper should be auto assigned randomly to 2 reviewers.

Postcondition: Paper is submitted for review.

-Paper details:
id of paper
title
abstract
authors: name, email, affiliation
presenter
reviewed: boolean

Use Case 4: Review paper

Precondition: User is logged in as a reviewer and has been assigned a paper to review.

- Reviewer gets the list of assigned papers.
- Reviewer selects the paper to review then enters and submits the review details including:

Overall evaluation:

2: strong accept
1: accept
0: borderline
-1: reject
-2: strong reject

radio

reviewd paper {
id:
...}

MOE

Paper contribution:

5: a major and significant contribution
4: a clear contribution
3: minor contribution
2: no obvious contribution
1: no obvious contribution

radio

Paper strengths: (multiline text)

Paper weakness: (multiline text)

Postcondition: Review is submitted to the system

accepted papers sum overall evaluation >=2

paper
presenter
Date time
location

- **Use Case 5: Create/update conference schedule**

  Precondition: User is logged in as an organizer and has access to the accepted papers and presenters.

  Note that accepted papers should have the sum of overall evaluation >= 2. + reviewed by 2

  - Organizer navigates to the conference schedule page. navigation
  - Organizer creates sessions and assigns accepted papers and presenters to each session. Add, drop box
  - Organizer sets session timings and locations. still in create session
  - Organizer publishes the conference schedule. ADD

  basically ADD

  Postcondition: Conference schedule is created and published and can be accessed by attendees.

Use Case 6: Get conference schedule

Allow visitors to get the conference schedule with the ability to filter the schedule data to get the presentations scheduled for a particular conference date.

use date

like the sort midterm

2. Deliverables

Seek further clarification about the requirements/deliverables during the initial progress meeting with the instructor. Note that further important clarifications maybe modified/added to the project requirements.

1. Design the App Web UI and navigation.
   You may design the UI wireframe (sketch) to decide the UI components and the layout either on paper or use a design tool such as https://www.figma.com/
2. **During the weekly office hours, you are required to present and discuss your design with the instructor and get feedback.**
3. For each use case, implement the app Web UI and navigation using HTML, CSS and JavaScript. The pages should comply with Web user interface design best practices. Also remember that 'there is elegance in simplicity'.
4. Design and implement the app navigation to allow the user to navigate from one page to another in intuitive and user-friendly way to achieve the app use cases.
5. For each use case, design and implement the server-side data access repositories using JavaScript to read/write the app data from/to a data store.
6. Also, you should initialize the data store with data from JSON files (if the data store is empty).
7. Application design documentation should include the Entities, Repositories and Services class diagrams.
8. Document the app testing using screen shots illustrating the results of testing.

- Every team member should submit a description of their project contribution. Every team member should demo their work and answer questions during the demo.
- Push your implementation and documentation to your group GitHub repository as you make progress.

Note that this phase will be focused only **a fully working client-side and server-side implementation** using data stored in json files and a simple data store.

3. Grading rubric

Criteria	%	Functionality*	Quality of the implementation
1) Design and implement the app Web UI and navigation using HTML, CSS and JavaScript. Including designing the App Web UI and navigation.	50		
2) Design and implement the server-side data access repositories to read/write the app data from/to data store. Also, initialize the data store with data from JSON files.	40		
3) Application Design: Entities, Repositories and Services class diagrams.	5		
4) Testing documentation using screen shots illustrating the testing results. - Discussion of the project contribution of each team member. Members should collaborate and contribute equally to the project.	5		
Total	100		
Important remark: *In case of copying and/or plagiarism or not being able to explain or answer questions about the implementation, you lose the whole grade.*			

* **Criteria for grading the functionality:**

- The functionality is working: you get 70% of the assigned grade.
- The functionality is not working: you lose 40% of assigned grade.
- The functionality is not implemented: you get 0.
- The remaining grade in all cases from above is assigned to the quality of the implementation,
- The grades are distributed on the various use cases, when the design/implementation is partial, you get only the grades of designed/implemented use cases.

Code quality criteria, include:

- Use of meaningful identifiers for variables and functions (e.g. using JavaScript naming conventions)
- Pages are responsive
- Clean code: simple and concise code, no redundancy
- Clean implementation without unnecessary files/code
- Use of comments where necessary
- Proper code formatting and indentation.

You lose marks for code duplication, poor/inefficient coding practices, poor naming of identifiers, unclean/untidy submission, and unnecessary complex/poor user interface design.