ConfPlus

CMPS 350 Project Phase 2 – WebApp UI using Next.js and Data Management using Prisma

Conference Management System (ConfPlus)

(10% of the course grade)

The project phase 2 submission is due by 8am Sunday 28th May 2023. Demos during the same day.

1. Deliverables

You are required to complete the implementation of ConfPlus Web App by delivering the following.

You may use the provided model solution as a base for phase 2. Otherwise you need to address the project phase 1 feedback given during the demo session in the grading sheet.

1. **Design and implement the Data Model** using Prisma to manage the app data in a SQLite or Postgres database.
2. **DB Init:** Implement **seed.js** to populate the database with the data from the JSON files.
3. **Repository Implementation**: Implement the repository functions to read/write data from the database using Prisma Client queries. The *repositories* should offer the same functionality as phase 1. All **data queries, filtering, sorting, and aggregation should be performed by the database server rather than the application code.** To optimize the performance and minimize unnecessary data transfer, only the required data should be retrieved from the database. For instance, use aggregate query to request the database to compute the summary data for the summary report, instead of retrieving all the data and performing the computation in the app code. This will help to improve performance and simplify the app code.
4. **Web UI implementation** using Next.js and Server Actions.
5. Implement **2 additional use cases** (in addition to Phase 1 use cases):
    - **Login** using at least 2 authentication providers such as GitHub, Google, Microsoft (besides local auth using the local database).
    - Conference Statistics Report including:
        - Number of papers submitted, accepted, and rejected.
        - Average number of authors per paper.
        - Number of conference sessions and the average number of presentations per session.
6. **Design and Testing Documentation**

- Document 3 **technical** lessons learned by comparing your submitted solution with the model solution provided. You need to provide detailed reflections about the new concepts and technical lessons learnt.
- Data Model diagram.
- **UI Design** table: for each page list the page **components** along with a brief justification of your design choices.

Page	Components	Brief justification

- **Data caching** table: for each page specify the caching strategy used along with a brief justification of your choice.

Page	Caching strategy	Brief justification

- Testing document including screenshots of conducted tests illustrating a working implementation.
- Every team member should submit a description of their project contribution. Every team member should participate in the solution demo and answer questions during the demo.

7. **App Deployment** (optional 5 pts project bonus)
   A successful deployment of the app and the database to a cloud hosting service such as https://vercel.com/ . The demo should walkthrough a fully working app online to get the bonus, no partial working solution will be considered.

Important notes:

- Continue posting your questions to MS Team Q & A channel.
- Do not forget to submit your design and testing documentation (in Word format) and fill-up the Functionality column of the grading sheet using the provided phase 2 template.
- Push your implementation and documentation to your group GitHub repository as you make progress.
- You need to test as you go!
- Seek further clarification about the requirements/deliverables during office hours or by posting questions online. Note that further important clarifications maybe added to this document, and you will be notified.
- Report any team issues early, any issues reported towards the submission deadline will be ignored.

2. Grading rubric

Criteria	%	Funct Ionality*	Quality of the implementation	Grade
Design and implement the Data Model.	10			
Init DB: populate the database with the data from the json files.	5			
Repository Implementation to read/write data from the database	15			
Change Web UI to use Next.js and server actions	60			
Design and Testing Documentation *** Design documentation:** - 3 key lessons learned from Phase 1. - Data Model diagram. - UI Design table - Data caching table	10			

* **Testing documentation:** with evidence of working implementation using snapshots illustrating the results of your solution testing (you must use the provided template). * **Discussion of the project contribution** of each team member [-10pts if not done]				
Total	100			
Bonus - successful deployment of the app and the Database to a cloud hosting service such as https://vercel.com/	5			
Copying and/or plagiarism or not being able to explain or answer questions about the implementation.	0			

* **Possible grading for functionality**: ***Working*** (get 60% of the assigned grade), ***Not working*** (lose 40% of assigned grade and ***Not done*** (get 0). The remaining grade is assigned to the quality of the implementation. In case your implementation is not working then 40% of the grade will be lost and the remaining 60% will be determined based on of the code quality and how close your solution to the working implementation. Solution quality also includes meaningful naming of identifiers, no redundant code, simple and efficient design, clean code without unnecessary files/code, use of comments where necessary, proper white space and indentation.

Marks will be reduced for code duplication, poor/inefficient coding practices, poor naming of identifiers, unclean/untidy submission and unnecessary complex/poor user interface design.