ConfPlus

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

Conference Management System (ConfPlus)

(15% 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 need to address the project phase 1 feedback given during the demo session.

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. The *repositories* should offer the same functionality as phase 1. All **data filtering should be done on the server** using Prisma Client queries and only the required data should be retrieved. The implementation should make use of Prisma Client query capabilities (e.g., using aggregate query to get the data for the reports).
4. **Web UI implementation using Next.js** and Server Actions.
5. Implement **2 additional use cases** (in addition to Phase 1 use cases):
   - **(Optional - with a bonus) Login** using at least 2 authentication providers such as GitHub, Google, Microsoft (besides local auth using the local database).
   - (**Mandatory**) Conference Statistics Report including:
     - Number of papers submitted, accepted, and rejected. (_count)
     - Average number of authors per paper. (_avg)
     - Number of conference sessions and the average number of presentations per session. (_count & _avg)

maybe in homepage

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.
   - Improvement actions: Detail that modifications you made to address the feedback you received for the phase 1

Feedback	Improvement action	details
Remark 1		
Remark 2..		
..		

- Data Model diagram.
- Queries: List the various **types** of queries that your application makes, and discuss concisely how your application is exploiting the power of the database server (e.g. by making data sorting and filtering inside the DB and not in the application codes).
- **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 (within the final report). 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:

- 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
Improvement over the first phase: *Depending on the implantation status of the previous phase, the following might apply:* - *Completing missing functionalities;* - *Improving the design and implementation of paper submission: paper status, etc.* - *Improving the design and implementation of paper review: distinction between reviewed papers and papers to review, etc.* - *Various filtering possibilities for the conf schedule* - *Correct interpretation of session* - *Clarity of the various UIs.*	25			

By default, if no improvement is made, the student will have the same grade of previous phase for this category. *Everything you improve will add up to your previous grade that will be used as a baseline.*				
Design and implement the Data Model. Clarity of data entities, their attributes and relations (in Prisma and the conceptual model (the diagram))	10			
Init DB: populate the database with the data from the json files.	5			
Repository Implementation to read/write data from the database	10			
Database: - The design and implementation of the statistics page - All other use-cases use the database, not JSON files or local storage. - All queries function correctly.	40			
Design and Testing Documentation *** Design documentation:** - 3 key lessons learned from Phase 1. - Data Model diagram. - UI Design table - Data caching table *** 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]	10			
Total	100			
Bonus - successful deployment of the app and the Database to a cloud hosting service such as https://vercel.com/ - successful implementation of use authentication.	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.