HalaqaMetrash Web App

CMPS 356 Project – WebApp Design and Implementation

This is a group project worth 30%. The project submission is due by 10am Wednesday 11th January 2023.

1. Requirements

Qatar is blessed by many Quran classes for kids. You are requested to design and implement a HalaqaMetrash Web App to allow parents to follow-up the progress of their kids and help teachers engage parents and easily communicate with them. The key use cases to deliver are shown in Figure 1.

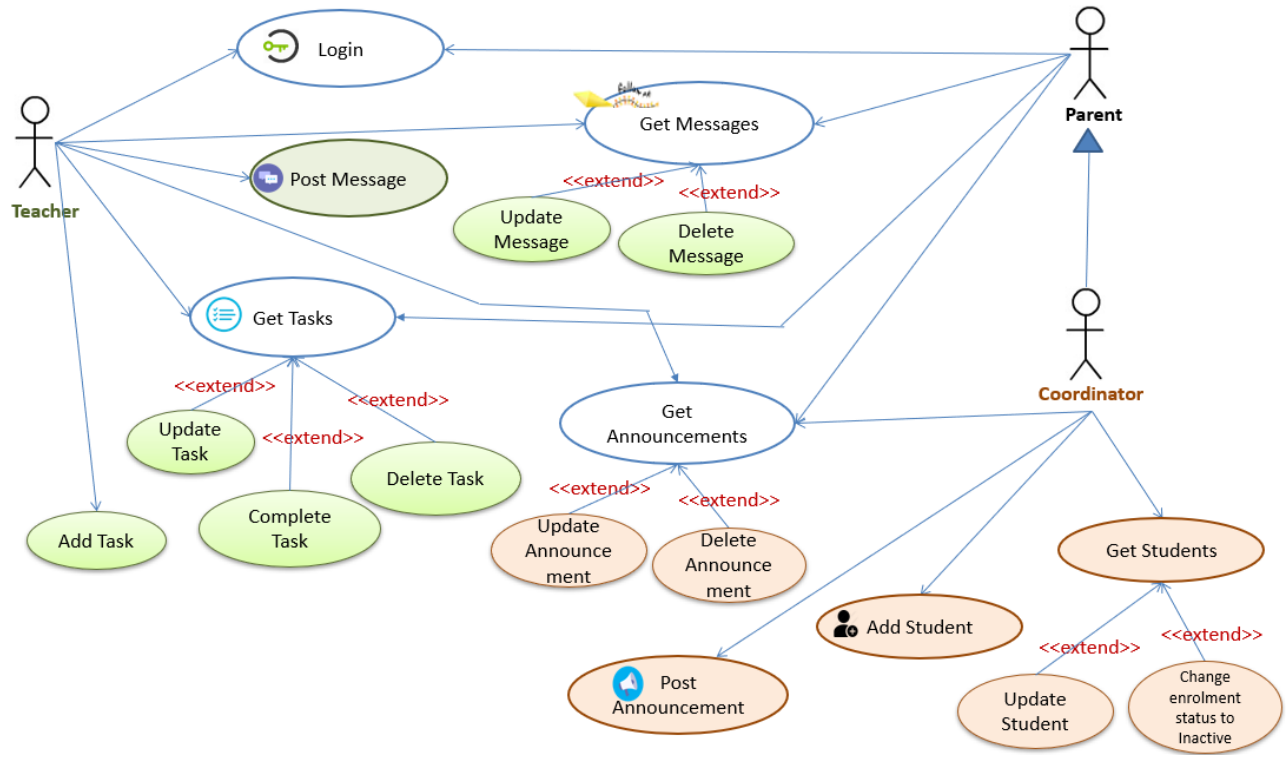

Figure 1. HalaqaMetrash Use cases

Login: Allows the user (i.e., *Coordinator*, *Teacher* and *Parent*) to login to use the application. The user should be able to Login using a custom email & password authentication developed as part of the app and 2 social media authentication providers such as Google, GitHub, Twitter, Facebook, etc.

- **Add Student**: The *Coordinator*, responsible of managing all Halaqat, can add a student. The student registration details should include the parent details: first name and last name, Qatari Id, Mobile, Email, and Password. A parent can register 1 or many children. A child details include: First name, Last name, Date of Birth, Gender, and School Grade. Note that if a parent has many registered children then they should only have 1 login account and upon login they should be able get the tasks and messages for their registered children (as explained below).

Upon registration, the child should be assigned to a Halaqa. Note that each Halaqa has a name and a teacher. The list of teachers and their Halaqa and the coordinator details are provided to you. But you still need to include these in your design and implementation.

- **Get Students**: the coordinator can get the list of registered students. As an extension to this use case, the coordinator can also update the student details or change the student's enrolment status to inactive.
- **Add Task** allows the teacher to assign a memorization or a revision task to a student in their Halaqa to keep track of their progress of memorizing new Ayats or revising previously learned ones. First the teacher selects the student to assign the task to, then selects the Sura, the Aya range, the due date (by default this should be set to Today's date + 1) and the type of task (Memorization or Revision). The Sura list and possible Aya range should be provided to the user to avoid typing them. The list of Quran Surahs is provided to you.

 **Get Tasks** for a particular student by status either completed, pending or all tasks. The parent can only get the tasks for their children, the teacher can get the tasks for a student in their Halaqa while the coordinator can get the tasks for any student.

For Pending Tasks, the teacher can either Complete, Update or Delete a task.

- **Complete task**, the teacher can mark the task as completed then specify the Completed date (by default it should be set to Today's date), the Hifz level (Excellent, Ok, Poor) and optional comment.
- **Update task** to update the task details (Sura, the Aya range, the due date and the type of task (Memorization or Revision). This use case is similar to add.
- **Delete task** to delete a task.

 **Post Message**: The *Teacher* can post a message associated to a particular student to keep the parents informed about the Halaqa learning achievements or child positive/negative behavioural or simply share that unforgettable Halaqa happy moments! The teacher can attach images to the message such as photos from the Halaqa.

 **Post Announcement**: The *Coordinator* can post announcements informing parents/teachers of important events such as Competitions and Holidays. They can also attach images to the announcement.

- **Get Messages**: The parent, teacher, and coordinator can get messages for a particular student within a date range (by default the last 30 days). The parent can only get the messages for their children, the teacher can get the messages for a student in their Halaqa while the coordinator can get the messages for any student. As an extension to this use case, the teacher can also update or delete a message.

- **Get Announcements**: The parent, teacher, and coordinator can get announcements within a date range (by default the last 30 days). As an extension to this use case, the coordinator can also update or delete an announcement.

Deliverables:

1) Architecture Design, Classes Design and the UI design to deliver HalaqaMetrash use cases.
   The design documentation should include at least the following:
   - ~~Application Architecture Diagram~~

- Class Diagram showing Entities, Repositories and Services and Controllers.
- UI Design and navigation.

During the weekly project meetings with the instructor, you are required to present and discuss your design with the instructor and get feedback.

2) Implement the client-side and the server-side Web components to deliver HalaqaMetrash use cases based on your previously developed and validated design.

The HalaqaMetrash should be fully implemented using React.js and Next.js. The application data can be managed either using *json* files or a database such as MongoDB. HalaqaMetrash web pages should use React Components, HTML 5 and CCS. The pages should comply with Web user interface design best practices. Also remember that 'there is elegance in simplicity'.

3) Bonus 5pts for a successful deployment of the app 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).

Push your implementation and documentation to your group GitHub repository as you make progress.

2. Grading rubric

Criteria	%	Functionality*	Quality of the implementation
Application Design Architecture Design, Classes Design and the UI Design to deliver HalaqaMetrash use cases. The design documentation should include at least the following: ~~• Application Architecture Diagram~~ • Class Diagram showing Entities, Repositories and Services and Controllers. • UI Design and navigation.	15%		
Complete and correct implementation of the requirements:	80%		
• Login	7		
• Student Registration (get, add, update, change enrolment status to inactive)	15		
• Get Tasks	12		
• Complete task	8		
• Delete task	6		
• Add Task	10		
• Update task	6		

• Get Messages	8		
• Get Announcements	8		
• Post/Update/Delete Message	10		
• Post/Update/Delete Announcement	10		
Testing documentation with evidence of correct execution using snapshots illustrating the results of testing.	5%		
Total	100		
Bonus - successful deployment of the app 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	-100%		

* **Possible grading for functionality**: ***Working*** (get 70% 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. Design quality includes **correct usage of MVC**, 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 and unnecessary complex/poor user interface design.

3. Ground Rules

- All assignments **must be your own original work**, not based on the work of other students, online examples/tutorials, or any other material from any other source. Any assignments found to be based on work other than your own will automatically be given a **grade of zero**, and may lead to further disciplinary action as per QU policy.
- You should push your work to Github as you make progress. Late submission policy: 10 points deduction for each late day and 0 after 3 days.

Appendix

Suggested Add/Update Task UI:

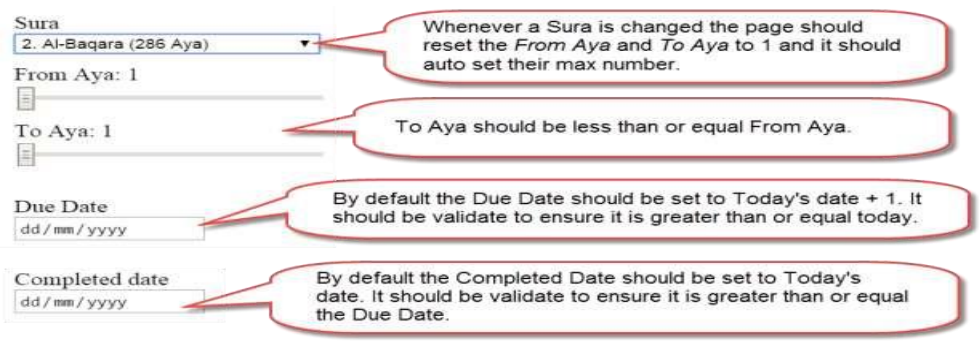

Figure 2. Add/Update Task UI Design

Resources:

- Quran Surah list http://erradi.github.io/json/surah.json
- Student list http://erradi.github.io/json/student.json
- Teacher list http://erradi.github.io/json/teacher.json
- Sample Task list http://erradi.github.io/json/task.json