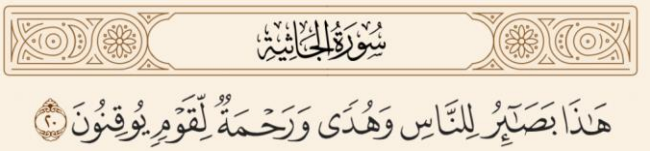
# CMPS 350 Project Phase 1 – WebApp UI Design and Implementation (15% of the course grade)




**The project phase 1 submission is due by 8am Sunday 6th March 2022. Demos during the same week.**

## 1. Requirements

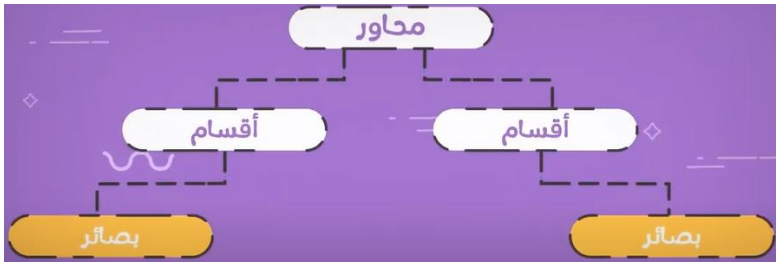
You are requested to design and implement بَصَائِر Web app for a **novel interactive** Tafsir using **Arabic** user interface. The project will be in collaboration with بصائر المعرفة القرآنية https://quranok.com/ lead by Prof. Abdulsalam Almajeedy. Tafsir will use سورة النساء as an example.

The key goal is to provide an **interactive خريطة محاور السورة** to ease access to the associated Quran verses and the associated 2 levels of Tafsir: **الوجيز** و **الوسيط**


More info @ https://www.youtube.com/watch?v=nzN2O0CaTJU

The unique characteristic of this Tafsir is

"يعتمد التفسير الموضوعي للقرآن على تدبر السورة وتقسيمها إلى محاور مترابطة، ينقسم كل محور بدوره إلى أقسام يمكن تقسيم كل قسمٍ عند الحاجة إلى فصول، ويعتمد هذا التقسيم الدقيق على ترتيب الآيات دون قفز عليها كما هو حال التفسير الموضوعي التقليدي ويظهر ذلك مدى الإحكام المذهل لآيات القرآن الكريم".

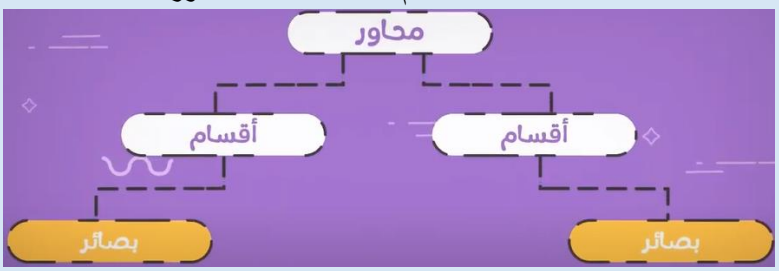


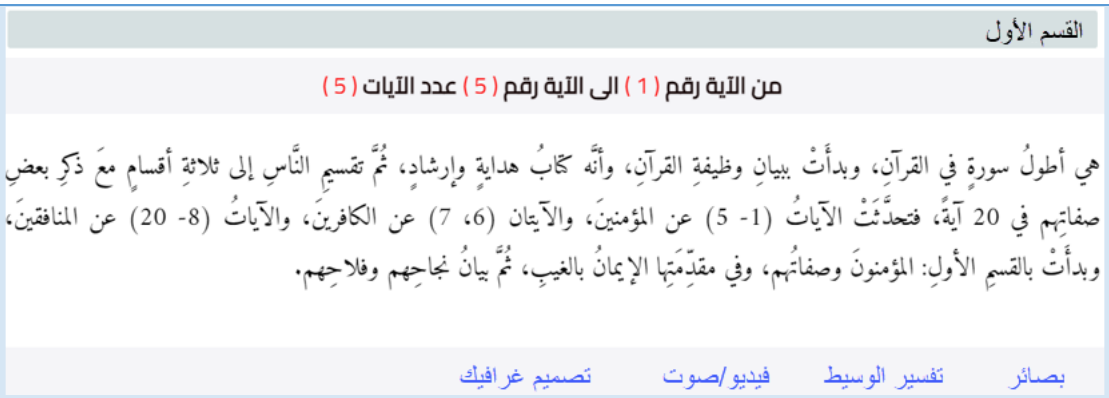

Project goal:

- تيسير التدبر و الفهم والعمل بالقرآن الكريم و ابراز جمال و روعة و اعجاز الأسلوب القرآني
- تيسير حفظ القرآن الكريم باستخدام الروابط اللفظية والمعنوية والموضوعية
- قُرْأَنٌ يُتْلَى لانسانية ترقى

The main بَصَائِر use cases are described Table 1.

**Table 1. Use cases description**

| Use case | Brief description |
|---|---|
| **1. Quran Navigator** | Navigate Quran enables to user to access a Surah, a page by number, the start of Juz جُزْءْ , or the start of ربع. In addition to the search by a word or part of a word.<br>Some of kind of UI like this:<br>  |
| **2. Quran Viewer** | Display Quran verses as requested by the user using the Quran Navigator. The viewer should display one page at time with the ability to navigate back and forth between pages. |
| 3. **خريطة المحاور للسورة** (أي ***الروابط الموضوعية***) | • For each surah provide the ability to access an interactive خريطة المحاورللسورة with ability to view for each محور the associated أقسام .<br>  <br>• For each **قسم** view the associated ayat in the Quran Viewer.<br>Color highlight the ayat of the **قسم** in the Quran Viewer and display the **قسم** summary text.<br>Also allow access to the **قسم** associated resources (if any) including:<br>- بصائر<br>- تفسير الوسيط<br>- Video/audio link<br>- Graphical illustration |

| | |
|---|---|
| |  <br>Surah *An-Nisa* will be used as an example. |
| **4. Verse Recitation and Tafsir** | From the Quran Viewer the user can click on the Verse number علامة الآية ۞ to access the verse Recitation (from everyayah.com) and Tafsir (using the provided `tafsir.json`). |

Needed Json files are provided on GitHub. Further Quran data resources -if needed- can be provided / contributed by students in json format from sources such as

https://tanzil.net/docs/home
https://globalquran.com/download/data/
https://github.com/risan/quran-json
https://github.com/quran/quran.com-api

## 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://balsamiq.com/

   **During the weekly office hours, you are required to present and discuss your design with the instructor and get feedback**.

2) 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'.

   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.

3) For each use case, implement the client-side data access repositories using JavaScript to read/write the app data from/to IndexedDB.

4) Create test data JSON files for the app entities. Then initialize the IndexedDB with data from JSON files.

   First test the implementation using a main function that displays the results to the console before using them in the UI.

5) Application design documentation including the Entities Class Diagram and the Repositories Class diagram.

6) 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 implementation** using data stored in json files and local browser database. In phase 2 you will implement the server-side to move some of the computation and data management to the server-side.

## 3. Grading rubric

| Criteria | % | Functionality* | Quality of the implementation |
|---|---|---|---|
| 1) Implement the app Web UI and navigation using HTML, CSS and JavaScript. Including designing the App Web UI and navigation. | 45 | | |
| 2) Implement the client-side data access repositories to read/write the app data from/to IndexedDB.<br>Also, create test data JSON files for the app entities. Then initialize the IndexedDB with data from JSON files. | 45 | | |
| **3) Application Design:** Entities Class Diagram and Repositories Class diagram. | 5 | | |
| **4) Testing documentation** using screen shots illustrating the testing results.<br>- Discussion of the project contribution of each team member. | 5 | | |
| **Total** | 100 | | |
| 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.

Solution quality also includes meaningful naming of identifiers (according to Android naming conventions), no redundant code, simple and efficient design, clean implementation without unnecessary files/code, use of comments where necessary, proper code formatting 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.