



Mizān – A Workload Management Web App

CMPS 350 Project Phase 1 – Web App Design and Implementation

This is a group project worth 10%. The project submission is due by 11:59pm Sunday 6th April 2025.

1. Requirements



Students have lot on their plate! What assignment is due? Is there any in class assessment today? When is my midterm? Sometimes it is hard to keep up with everything and remember everything you must do. Moreover, some instructors tend to forget that students have other classes to study for and have a life to enjoy! On the other hand, some students tend to procrastinate, leading to unnecessary stress!

As part of the university's Road to Student Success initiative, you are tasked with designing and developing Mizān, a workload management web app that helps students and instructors manage assessments effectively. Mizān ensures students stay informed about their coursework, enabling better time management and preventing deadline congestion.

Instructors will input course assessments—including quizzes, assignments, projects, exams, presentations, and lab tests—at the start of the semester. Program Coordinators and students can then provide feedback or request workload adjustments for instructors to consider.

By promoting balanced academic planning, Mizān aims to enhance student performance, improve retention and progression, and ultimately increase graduation rates..

The key use cases to deliver are shown in Figure 1.

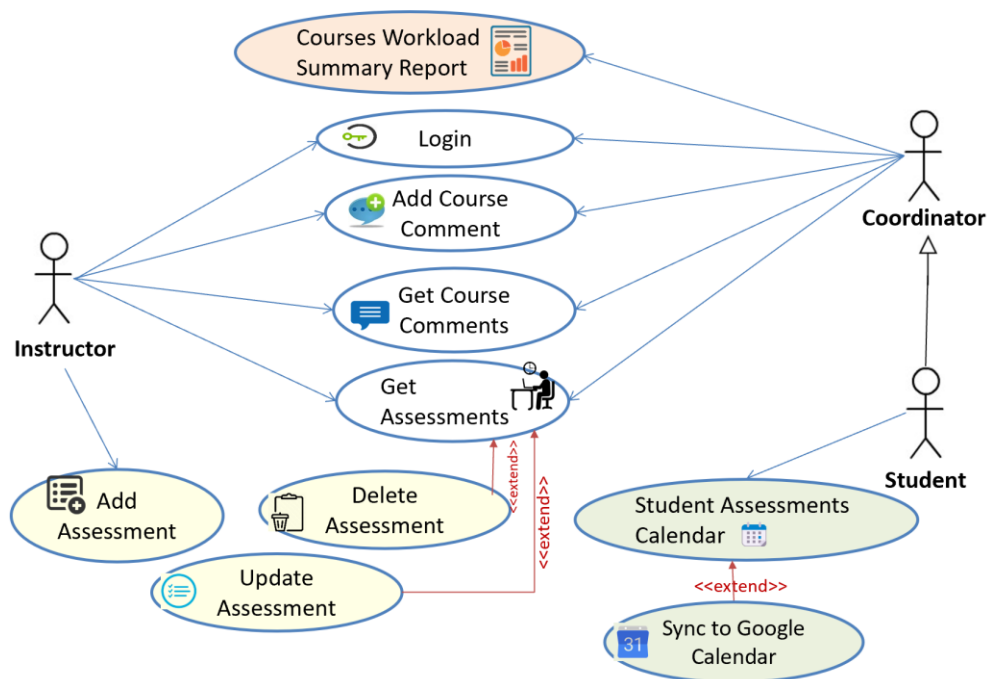










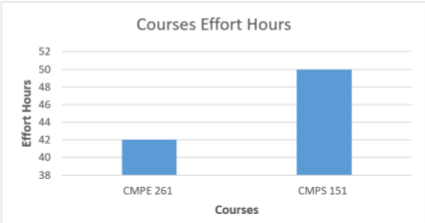


Figure 1. Mizān Use cases

Table 1. Use cases description

 Login	<p>Allows the user (i.e., <i>Coordinator, Instructor</i> and <i>Student</i>) to login to use the app.</p>
 Add Assessment	<p>Instructor can add assessments (e.g., quizzes, assignments, projects, exams) to the courses they teach during a semester.</p> <p>For each assessment, the instructor should enter the Due date, the average number of Effort Hours required to complete the assessment and the Weight (i.e., percentage towards Final Grade), Assessment Type (i.e., Homework, Quiz, Midterm exam, Final exam, Project) and Title. By default, the assessment type should be assigned as the title but the user can change it. If a previous assessment of the same type was entered before then the system should append a numeric value to the title to indicate its sequential number. For example, when entering the second midterm then its default title should be auto set to 'Midterm 2' but the user can edit it.</p> <p>Validation rules:</p> <ul style="list-style-type: none"> - Only one final exam can be added - At most 2 midterm exams can be added - No two assessments for a course can have the same due date - Project phases should be between 1 to 4 - Maximum 8 homework assignments per course
 Update Assessment	<p>Update an assessment associated with a course. Update should use the same UI as Add Assessment.</p>
 Delete Assessment	<p>Delete an assessment associated with a course.</p>
 Get Assessments	<p>Get the assessments associated with a course or set of courses:</p> <ul style="list-style-type: none"> - The instructor can only access the assessments of the courses they teach for a particular semester. They can get all the assessments or filter them by course. - The coordinator can access the assessment of any course or the assessments associated with courses belonging to a particular program (e.g., assessment for CS courses). - The student can only access the assessments associated with their courses. They can get all the assessments or filter them by course.
 Student Assessments Calendar	<p>Display the student's assessments in a calendar format.</p>
 Sync Assessments to Google Calendar	<p>Student sync their course assessments timetable to their Google Calendar to be reminded of important due dates.</p>

<div></div> <div>Add Course Comment</div>	<p>Program Coordinator and/or students can post comments to report issues or to request workload changes for the instructor’s consideration.</p> <p>The instructor can also add course comments to respond to feedback and provide justifications.</p> <p>The comment should have a title and a body. Also, the system record who created the comment and on which date.</p> <p>For simplicity, comments are associated with a course and not a assessment.</p>																												
<div></div> <div>Get Course Comments</div>	<p>Users can get comments associated with a course.</p>																												
<div></div> <div>Courses Workload Summary Report</div>	<p>This report allows the student or the coordinator to get a summary of the workload associated with their courses. The student can get the summary of their registered courses while the coordinator can get the summary of courses belonging to their program (for example CS coordinator can get the summary report for the CS courses). An example report content and format is shown below:</p> <div><div>Courses Workload Summary Report</div><table><tr><th rowspan="2">Course</th><th rowspan="2">Course Name</th><th colspan="4">Number of</th><th rowspan="2">Number of Assessments</th><th rowspan="2">Total Effort Hours</th></tr><tr><th>Homework</th><th>Quizzes</th><th>Project Phases</th><th>Exams</th></tr><tr><td>CMPE 261</td><td>Digital Logic Design</td><td>2</td><td>4</td><td>3</td><td>3</td><td>12</td><td>42</td></tr><tr><td>CMPS 151</td><td>Programming Concepts</td><td>3</td><td>6</td><td>1</td><td>2</td><td>12</td><td>50</td></tr></table><div><div>Courses Effort Hours</div></div></div>	Course	Course Name	Number of				Number of Assessments	Total Effort Hours	Homework	Quizzes	Project Phases	Exams	CMPE 261	Digital Logic Design	2	4	3	3	12	42	CMPS 151	Programming Concepts	3	6	1	2	12	50
Course	Course Name			Number of						Number of Assessments	Total Effort Hours																		
		Homework	Quizzes	Project Phases	Exams																								
CMPE 261	Digital Logic Design	2	4	3	3	12	42																						
CMPS 151	Programming Concepts	3	6	1	2	12	50																						

Deliverables:

1) Document the app design to deliver Mizān use cases including:

The design documentation should include at least the following:

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

During the weekly project meetings, 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 Mizān use cases based on your previously developed and validated design.

The HalaqaMetrash should be fully implemented using Next.js. The application data can be managed either using json files or <https://mockapi.io/>. The web pages could use, HTML, CCS and JavaScript. The pages should comply with Web user interface design best practices. Also remember that 'there is elegance in simplicity'.

3) Deploy the app to a cloud hosting service such as <https://vercel.com/>

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 <ul style="list-style-type: none"> Class Diagram showing Entities, Repositories and Services. UI Design and navigation. 	15		
Complete and correct implementation of the requirements:			
 Add Assessment	15		
 Update Assessment	10		
 Delete Assessment	3		
 Get Assessments	8		
 Student Assessments Calendar	10		
 Sync Assessments to Google Calendar	9		
 Add Course Comment	5		
 Get Course Comments	5		
 Courses Workload Summary Report	10		
Testing documentation with evidence of correct implementation using snapshots illustrating the results of testing.	5		
Deploy the app to a cloud hosting service such as https://vercel.com/	5		
Total	100		
Copying and/or plagiarism or not being able to explain or answer questions about the implementation	- 100%		

* **Possible grading for functionality:** *Complete and Working* (get 70% of the assigned grade), *Complete and 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. Quality includes **correct application 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.
- All assignments must be submitted electronically to Github. 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.