

COS60010 - Technology Inquiry Project**Semester 1 - 2024****Deliverable 2****Group Project Concept Report****Group Name:** GROUP 1

Student members: Arun Ragavendhar Arunachalam Palaniyappan	104837257
Christopher Opie	105380202
Layan Buddhimal Weerasingha Arachchige	104758921
Amirajsinh Pradhyumansinh Sonagara	104801333
Henil Mukeshbhai Pistolwala	105065800

Workshop: Thursday – 14:30-16:30 - Room EN207**Facilitator:** Dr. Alfandi Yahya

Table of Contents

1. Introduction.....	5
2. Proposed Features	6
2.1 Main functionality of the quiz (displaying questions and collecting answer inputs) for chemical structure-based chemistry questions.....	6
2.2 Displaying a list of correct and incorrect answers at the end of the quiz	6
2.3 Incorporating username/password-based login & authentication for students	6
2.4 Recording and retrieving player statistics and scores via the database	6
2.5 Deploying the project to the web	6
2.6 Providing admin access to teachers for database modifications	6
2.7 Implement difficulties levels for questions.....	7
2.8 Extending the quiz with additional question types	7
3. Key Technical Decisions	7
3.1 User-Interface: HTML/CSS/JavaScript	7
3.2 Backend: PHP	7
3.3 Database: MariaDB	8
3.4 Deployment.....	8
4. Design and in-depth view of the proposed features	8
4.1 User-facing pages and their functionalities	8
4.1.1 Login Page	8
4.1.2 Welcome Page.....	9
4.1.3 Questions Pages.....	9
4.1.4 Results Page	10
4.1.5 Admin Page	10
4.2 Program Execution Flow Diagrams	11
4.2.1 Program execution flow diagram: Student.....	11
4.2.2 Program execution flow diagram: Admin.....	12
4.3 Databases	13
4.3.1 Users Table.....	13
4.3.2 Questions Tables	13
4.3.3 Scores Table	14
5. Timeline for the Project Execution	15
5.1 Table and Gantt chart detailing the execution plan.....	15
5.2 Task allotment for team members	17
6. Risk Management	17
6.1 General Risks	17
6.1.1 Security and Data Integrity	17
6.1.2 Performance	17

6.1.3 Cross-Platform Integration.....	18
6.1.4 Scalability.....	18
6.1.5 Usability	18
6.1.6 Deployment	18
6.1.7 Data Loss.....	18
6.2 Group Risks.....	19
6.2.1 Knowledge Gaps	19
6.2.2 Group Member Exit	19
6.2.3 Technology Difficulties	19
7. Summary.....	19
8. Appendix	20
8.1 Abbreviations	20
8.2 List of figures and tables:	20
9. References	20

Total Word Count: 2856

(excludes Title Page, Table of Contents, Acknowledgement to Country, Captions, Appendix, References

Acknowledgement to Country

Today's modern-day Melbourne and Swinburne University of Technology is situated in what was originally the Kulin Nation of the past. As Swinburne students, who are truly grateful and proud to study at this esteemed institution, we would like to humbly pay my deepest respects to the Wurundjeri People of this nation who are the traditional owners of these lands.

Additionally, we would also like to sincerely thank the students, alumni, partners, and guests of Swinburne who are from Aboriginal and Torres Strait Islander backgrounds.

It is our honour and we feel proud to recognise and acknowledge the link of spirituality, history and culture of this place to the Wurundjeri country.

1. Introduction

After conducting a detailed technical business analysis and user requirements study with the client Instatute Learning Pty Ltd., we have decided to create an interactive application to support the preparation of their students for university entrance exams for both Australian and international institutions. A web-based Chemistry quiz application has been designed to make the learning easier and more engaging for students struggling with challenging chemistry concepts.

The quiz will be implemented as a support application available within the main website of Instatute Learning Pty Ltd. The core idea is that every enrolled student at Instatute Learning Pty Ltd have access to this application. A student user can login using his/her credentials and can play the game an unlimited number of times. They can modify the difficulty level and view their historical results.

The application also provides an admin login for a staff, allowing them to can create new questions and modify student records.

The goal of the report is to discuss in depth both “what” is going to be built, and “how” it is going to be built.

2. Proposed Features

Based on the requirements purpose discussed in the Introduction, the following user stories have been designed and their respective priorities designated.

2.1 Main functionality of the quiz (displaying questions and collecting answer inputs) for chemical structure-based chemistry questions

Priority: 1 (Highest)

- Students should be given a series of chemistry-based quiz questions displayed to them, and they can submit their answers to the application.
- The focus is on chemical structure-related questions because it is fundamental to many branches of chemistry and is a challenging topic for students.
- This implementation will help distinguish the program from existing chemistry-based quizzes.

2.2 Displaying a list of correct and incorrect answers at the end of the quiz

Priority: 2

- For an effective self-assessment, players should be given feedback at the end of a game which displays the questions asked, their responses, and the correct answers.
- Students will be encouraged to revisit their textbooks or notes, to clear up any misunderstandings they may have.

2.3 Incorporating username/password-based login & authentication for students

Priority: 2

- Students should be able to access the “Chem Quiz” lobby by entering their username and password which has been provided to them by the Instatute Learning Pty Ltd.

2.4 Recording and retrieving player statistics and scores via the database

Priority: 2

- Students should have their scores saved into the database after they attempt a quiz.
- Statistics of their previous attempts should be provided to them after they log into the application, such as their highest score or their previous attempt scores, showing their improvement over time and providing a sense of accomplishment.
- At the end of each session their score should be immediately updated with the latest attempt, and the updated statistics displayed.

2.5 Deploying the project to the web

Priority: 3

- The application should be deployed to the web as soon as Priority 1 and Priority 2 features have been implemented.
- Since the development environment is different to the production environment, an early deployment will provide ample time to sort out any bugs that arise.
- **At this stage, the project is recognized to be the minimum viable product.**

2.6 Providing admin access to teachers for database modifications

Priority: 4

- Teachers and staff should have access to an administrative area of the application.

- They should be provided access to perform tasks such as the addition, deletion or updating of quiz questions and student information.
- The addition of questions feature specifically would allow teachers to provide a more customised experience for their students, resulting in questions being directly relevant to the students' current studies.

2.7 Implement difficulties levels for questions

Priority: 4

- Students should be able to modify the difficulty level of the quiz questions, since this application is meant to be used by at least high school students from a range of year levels.

2.8 Extending the quiz with additional question types

Priority: 5 (Lowest)

- Students should be tasked with answering diverse types of chemistry questions, to keep them engaged and allowing for a wider subject scope.
- Some additional question types include:
 - Naming the type of chemical reaction shown
 - Naming the highlighted functionality on a chemical structure
 - Counting the number of a particular type of atom on a provided structure **(Fryhle & Snyder, 2017)**.

3. Key Technical Decisions

3.1 User-Interface: HTML/CSS/JavaScript

- For an interactive study tool, the best method to engage with users is through a graphical or visual medium.
- When combined, HTML, CSS and JavaScript provide a developer-friendly method of creating user interfaces.
- A Web-based project will be highly accessible to users, as a properly deployed program can be accessed by any device with an internet connection. **(Full, 2020)**.
- Due to the constraints of the knowledge of the team, we have decided to refrain from using a frontend library/framework such as React or Vue, and will use JavaScript instead of TypeScript.
- Additionally, a frontend library/framework would provide a lot of features that we have no use for and would add a lot of unnecessary code and complexity.

3.2 Backend: PHP

- A range of backend languages would be suitable for our application, with the primary concern to provide an interface with our database.
- PHP is a strong candidate because it can provide this interface, it is lightweight, and it doesn't have complicated syntax.
- Hence, PHP has been chosen over the complicated syntax of Java, and over Python, which would require the use of a backend framework such as Django or Flask. Using these frameworks would add unnecessary code and bulk, but PHP can be used as-is.

3.3 Database: MariaDB

- We have decided to store our data in a relational database using SQL, since the entries to the database will either be curated (e.g. quiz questions, specific information about chemicals) or at least structured (scores, users).
- Additionally, the volume of data is not expected to be not great enough to require a NoSQL approach.
- The relational database management systems MySQL and MariaDB integrate with PHP easily, making it the ideal choice. **(Andrew et al., 2003)**.

3.4 Deployment

- Since we are developing a web-based project, the intention is to deploy the project to an online private server, before integrating it with the Instatute online platform.
- If the project needs to run on a local machine, the source code and a database dump file will be provided, in addition to documentation on how to set it up.

4. Design and in-depth view of the proposed features

4.1 User-facing pages and their functionalities

4.1.1 Login Page

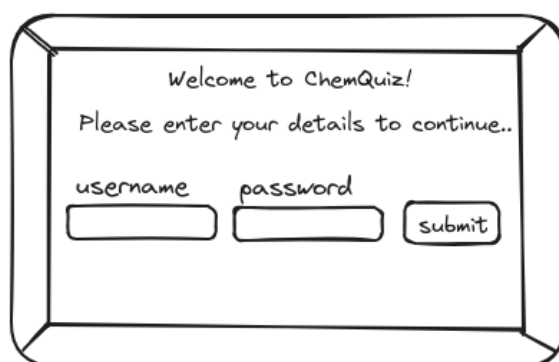


Fig. 1: Example user interface sketch for the login page

- Upon clicking the submit button, and after client-side validation, the entered username and password are sent to the server.
- The server sanitises the input, then queries the database using the username, and finally retrieves a hashed version of the user's password. The server hashes the entered password and compares it with the stored version.
- If the two don't match, or if the username was not located on the database, a failure response will be sent to the client, and the user will be prevented from proceeding. They will be prompted retry. **(Prettyman, 2020)**.
- If the credentials match, then the user has been validated. If the user is an admin, they will be directed to the administrative area. However, if the user is a student, the server will retrieve the complete record of the student's quiz attempts, as well as the top scores of other students, via SQL queries. This data is sent to the client as JSON for rendering the welcome page.

4.1.2 Welcome Page

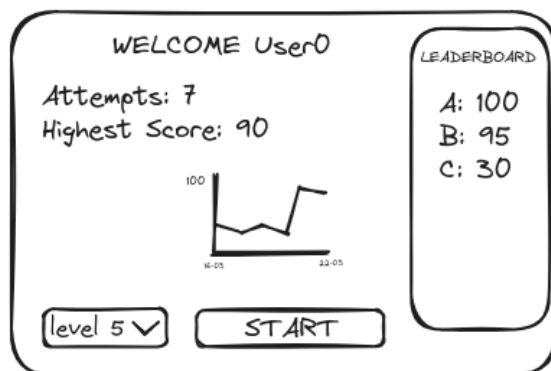


Fig. 2: Example user interface sketch for the welcome page

- After clicking the start button, the selected difficulty is sent to the server where is incorporated in a SQL query to the database, retrieving 10 questions of appropriate difficulty.
- The questions are sent to the client in one batch as JSON for rendering the questions pages, preventing the need for network requests after each response. (Smith, 2015).

4.1.3 Questions Pages

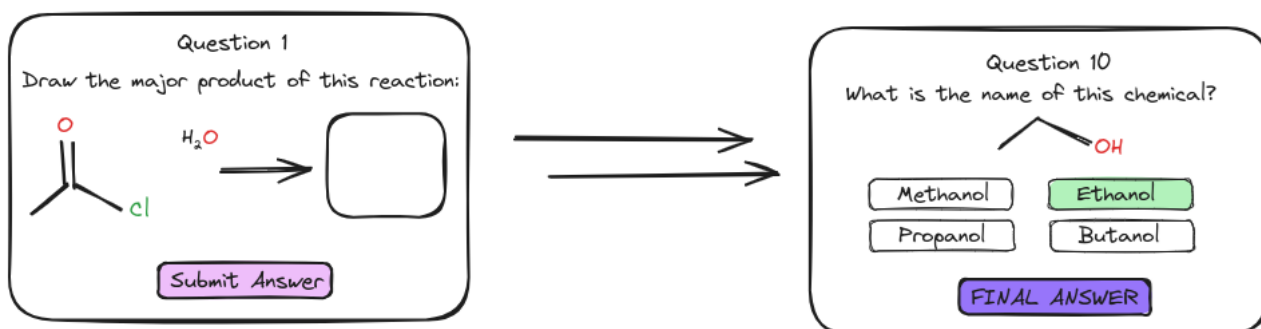
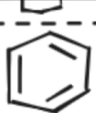



Fig. 3: Example user interface sketch for question pages 1 and 10

- Once a question has been answered, it gets stored as an array, and the next question is dynamically rendered to the screen using JavaScript. (Simpson, 2023).
- Once an answer to the final question is submitted, all responses in the array are sent to the server, alongside an identifier for the student.
- The server sends a SQL query to the database, retrieving the correct answers to all questions asked. It determines the student's score, and stores on the database using a SQL update query.
- The questions, answers and scores are sent to the client as JSON.

4.1.4 Results Page

#	question	correct answer:	you answered:	marks
9		benzene	toluene X	0
10		ethanol	ethanol ✓	10

Your Score: 90 / 100
That's a new record! Congratulations!

return

Fig. 4: Example user interface sketch for the results page

- Clicking the return button will cause a re-render of the screen to the welcome page, using the updated scores information that was retrieved earlier.

4.1.5 Admin Page

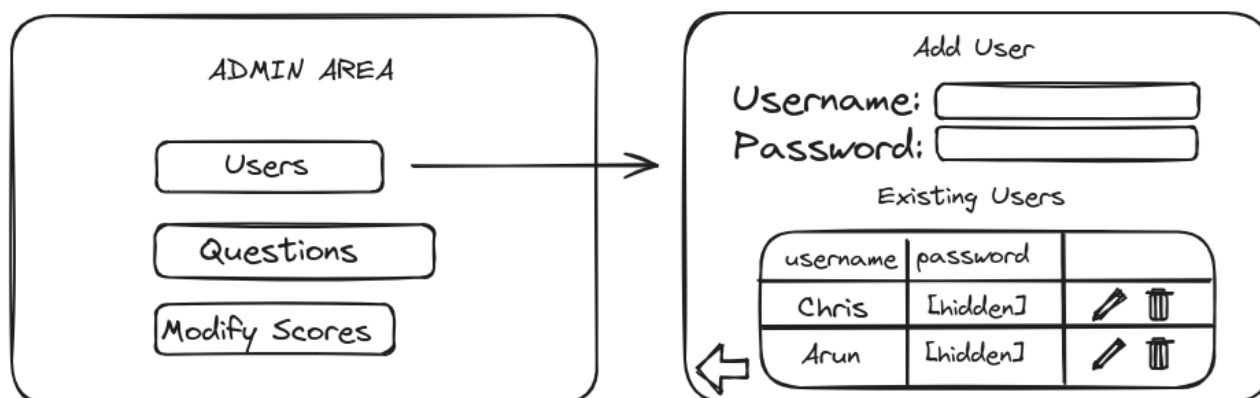


Fig. 5: User interface sketch for the results page, with the outcome of clicking the “Users” button

- Clicking a button corresponding to a table in the database triggers the server to retrieve all records of that particular table. These are sent to the client as JSON, the user interface is rendered to the screen consisting of an “add item” section and “existing items” section.
- Filling in the input fields in the “add item” section and attempting to submit the data will cause client-side validation, and if successful, the data will be passed to the server which sanitises it. A SQL update query is performed, and a success (or failure) message in the form of JSON is sent to the client based on the result of the procedure.
- Clicking a button to modify the item will provide an interface similar to the add item section, however the fields will be rendered containing the existing information. The user may make changes and submit the data, in which case a procedure resembling “add item” is performed. (Gehani, 2011)
- Clicking the button to delete a record on the database will send the unique identifier for the record to the server, and a SQL delete query will remove it from the appropriate table.

4.2 Program Execution Flow Diagrams

4.2.1 Program execution flow diagram: Student



Fig. 6: Step-by-step description of the flow of execution of the program for students

4.2.2 Program execution flow diagram: Admin

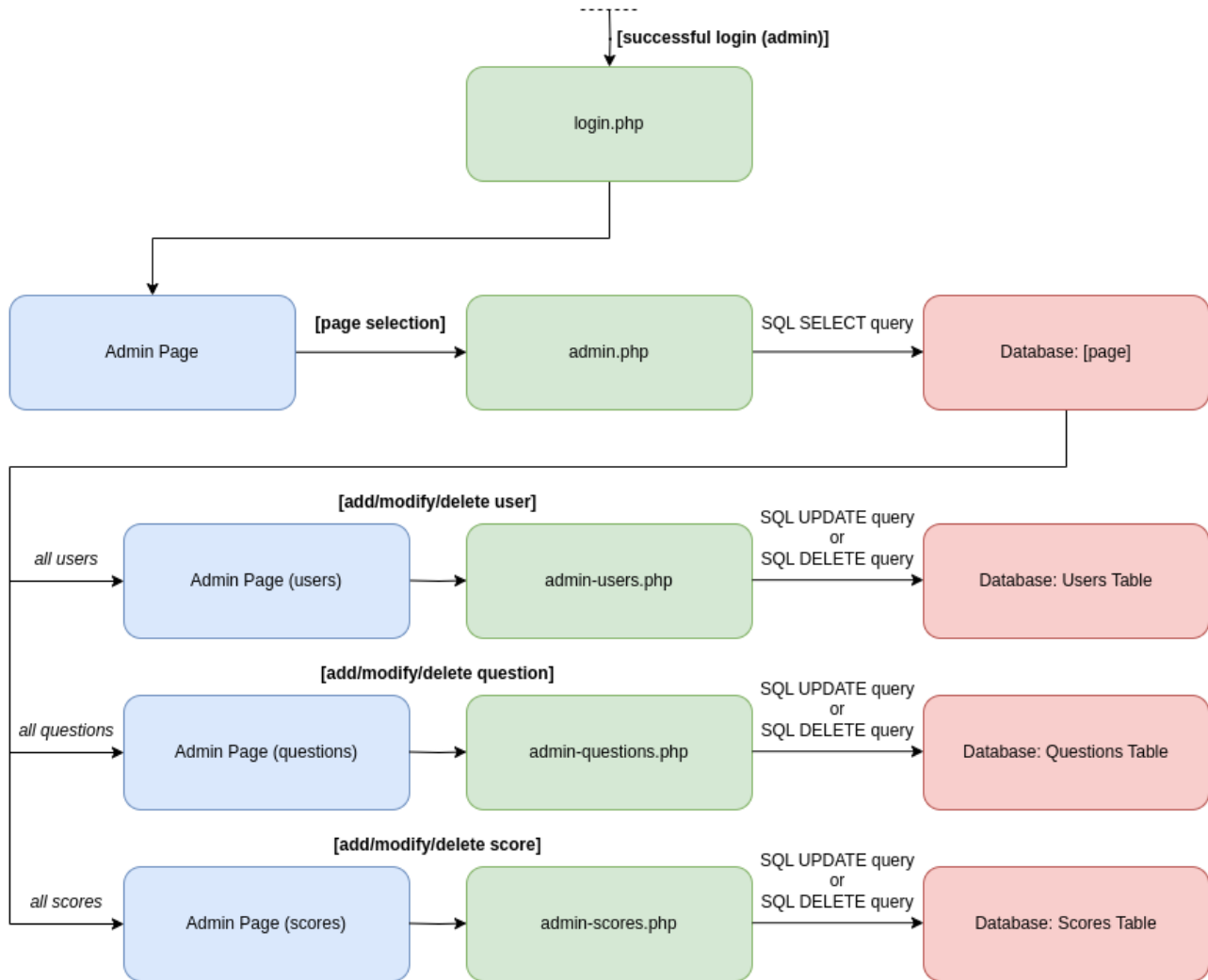


Fig. 7: Step-by-step description of the flow of execution of the program for administrators

4.3 Databases

A detailed database system with a list of associated tables is necessary to store, manage, and manipulate all essential data. To aid in this process, the following entity - relationship diagram was created. (Delisle, 2006).

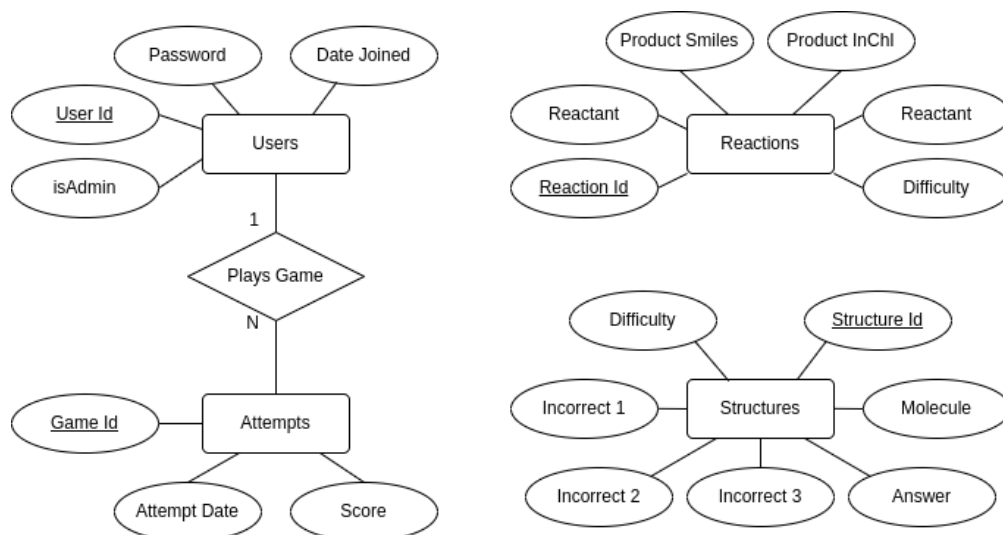


Fig. 8: Entity-relationship diagram describing the data involved in the proposed database

4.3.1 Users Table

- Stores the username, hashed password and creation date of each user.
- Each record has a user ID which uniquely identifies that user.

User Id	User Name	Password	Date-Joined	isAdmin
5	Arun	\$2y\$10\$od4/W8	23.02.2024	false

Table 1: DB schema and an example record of USERS table. User Id is the primary key

4.3.2 Questions Tables

Each type of question will require its own table, containing information about the question to be asked, the correct answer, and a difficulty rating (see **Table 2** and **Table 3** below).

- Multiple-choice questions will include pre-chosen decoy answers.
- Each question will contain a unique ID to identify each question on each table.

Structural information about chemicals will be stored in two established formats:

- SMILES (simplified molecular-input line-entry system): a way to represent chemicals in string format.
 - For example, sodium chloride could be represented as "[Na+].[Cl-]".
 - SMILES can be transformed into SVGs (image format) on the client using a library called RDKit.js. (Landrum, 2013).
- InChI (International Chemical Identifier): a more recent method than SMILES to represent chemicals in string format. Unlike SMILES, chemicals in the InChI format are represented in a completely unambiguous manner.

- For example, sodium chloride is "InChI=1S/ClH.Na/h1H;/q;+1/p-1" in the InChI format, whereas both "[Na+].[Cl-]" and "[Cl-].[Na+]" are valid SMILES.
- To compare with the stored InChI string answer to a quiz question, a user's inputted chemical structure input needs to be converted the InChI format using RDKit.js.

Reaction Id	Reactant	Reagent	Product Smiles	Product Inchi	Difficulty
5	<chem>CC(=O)Cl</chem>	<chem>CCCCN</chem>	<chem>CCCCNC(C)=O</chem>	InChI=1S/C6H13NO/c1-3-4-5-7-6(2)8/h3-5H2,1-2H3,(H,7,8)	2

Table 2: DB schema and an example record of REACTIONS table. Reaction Id is the primary key

Structure Id	Molecule	Answer	Incorrect 1	Incorrect 2	Incorrect 3	Difficulty
3	<chem>Cn1cnc2c1c(=O)n(C)c(=O)n2C</chem>	caffeine	ethanol	morphine	aspirin	4

Table 3: DB schema and an example record of STRUCTURES table. Structure Id is the primary key

4.3.3 Scores Table

- Contains details about the scores a specific user makes for each of their quiz attempts – the user Id corresponding to an entry in the Users table, the total score and the attempt date.
- Every record has a Game Id to uniquely identify each quiz attempt. (Delisle, 2006).

Game Id	User Id	Score	Attempt Date
1	5	8	02.04.2024
2	5	9	04.04.2024

Table 4: DB schema and an example record of SCORES table. Game Id is a primary key, User Id is a foreign key referencing User Id in the USERS table

5. Timeline for the Project Execution

5.1 Table and Gantt chart detailing the execution plan

The tasks required for the prompt delivery of the project deliverables are described in the following table and figure:

No	Tasks	Start Date	End Date	Week no
Task 1	Create the Database and the tables	08.04.2024	14.04.2024	6
Task 2	Use PHP to query the database and send the data as JSON	08.04.2024	14.04.2024	6
Task 3	Design the UI on paper	08.04.2024	14.04.2024	6
Task 4	Create an un styled static UI using HTML and dummy data	15.04.2024	21.04.2024	7
Task 5	Convert the static UI to dynamic JavaScript	15.04.2024	21.04.2024	7
Task 6	Use JavaScript to fetch data from PHP and display that data on the DOM	22.04.2024	28.04.2024	8
Task 7	Add JavaScript events to link all the pages together	22.04.2024	28.04.2024	8
Task 8	Deploy to a private server	29.04.2024	05.05.2024	9
Task 9	Styling of all the pages to make it aesthetic as well as usable and accessible	29.04.2024	05.05.2024	9
Task 10	Transfer the styling to the deployed project	06.05.2024	12.05.2024	10
Task 11	Final Testing, debugging and proof reading of the project	06.05.2024	12.05.2024	10
Task 12	Create a video with the project demonstration	13.05.2024	17.05.2024	11

Table 5: Proposed timeline for executing the tasks during project development

Project Timeline: Gantt Chart

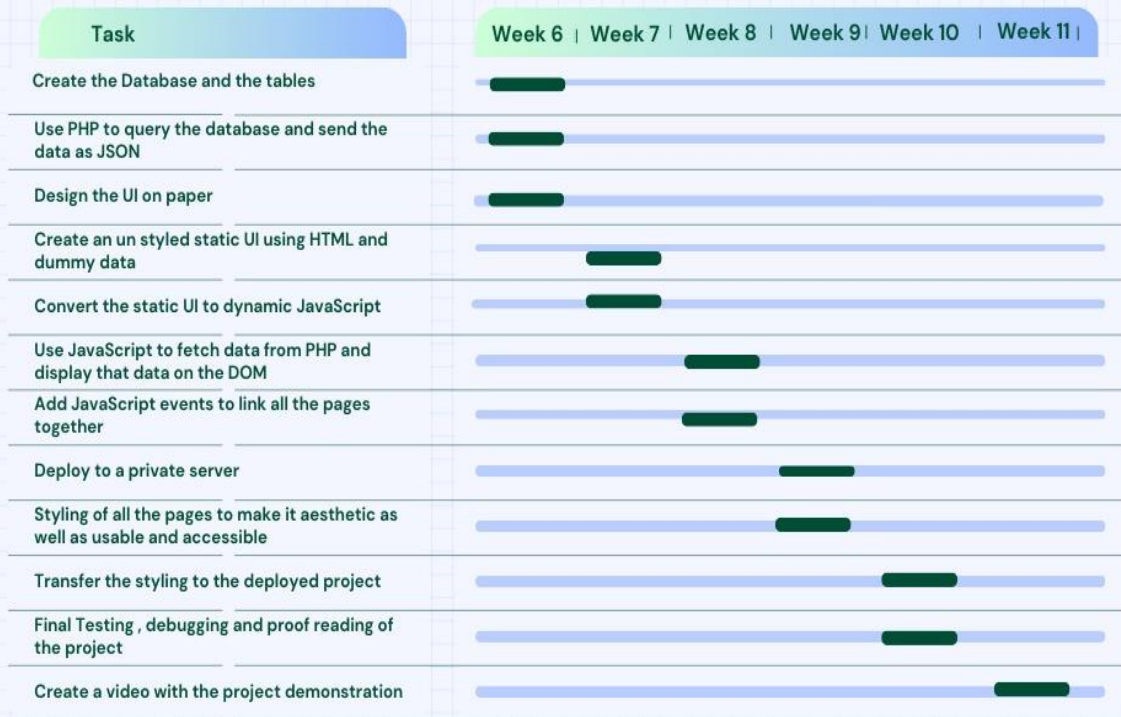


Figure 9: Visual representation of the project timeline via a Gantt chart

5.2 Task allotment for team members

The proposed division of labour for the project has been described in the following Trello board.

Trello Board			
Group 1			
1 TEAM MEMBER →	2 WEEK 6,7,8 →	3 WEEK 9,10 →	4 WEEK 11 ✓
ARUN	Results Page – Database and PHP file Creation , UI layout	Results Page – page interaction using JavaScript , Styling,Deployment	Results Page – final testing and debugging,video demonstration
CHRIS	Questions Page – Database and PHP file Creation , UI layout	Questions Page – page interaction using JavaScript , Styling,Deployment	Questions Page – final testing and debugging,video demonstration
LAYAN	Admin Page – Database and PHP file Creation , UI layout	Admin Page – page interaction using JavaScript , Styling,Deployment	Admin Page – final testing and debugging,video demonstration
AMIRAJ	Login Page – Database and PHP file Creation , UI layout	Login Page – page interaction using JavaScript , Styling,Deployment	Login Page – final testing and debugging,video demonstration
HENIL	Welcome Page– Database and PHP file Creation , UI layout	Welcome Page – page interaction using JavaScript , Styling,Deployment	Welcome Page – final testing and debugging,video demonstration

Figure 10: A Trello board detailing individual team member tasks

6. Risk Management

6.1 General Risks

These risks are generic concerns to be addressed when developing the project.

6.1.1 Security and Data Integrity

Issue: The exposure of user data may be the result SQL injection attacks through PHP if the user input is not properly sanitized before being used in database queries. **(Nixon, 2021).**

Impact: High

Likelihood: Medium

Mitigation: It's crucial to ensure that the user's input is inspected before it is sent to the database, via client-side and server-side validation, as well as server-side sanitisation. If serious, high-risk threats occur, an automated system should be in place to isolate the quiz application from the main Instatute platform, preventing further attacks.

6.1.2 Performance

Issue: Students would not tolerate applications with unacceptable performance.

Impact: Medium

Likelihood: Medium

Mitigation: The program should have neat and efficient code to avoid any sluggishness, and the database design must be efficient with minimal redundant or null records to improve query processing speed.

6.1.3 Cross-Platform Integration

Issue: Students will be frustrated if they are unable to application their device of choice.

Impact: Low

Likelihood: Medium

Mitigation: Testing should be performed on a variety of browsers and devices to ensure that the user interface is consistent and continues to function as expected.

6.1.4 Scalability

Issue: Poor performance will result from a system which is unable maintain its functionality in the event of growth.

Impact: Low

Likelihood: Low

Mitigation: A scalable database management system should be selected to accommodate large numbers of concurrent users is essential for a web-based application. It should also be able to store and retrieve increasing numbers of records with only negligible drops in efficiency.

6.1.5 Usability

Issue: Users with disabilities will be frustrated if they cannot run the application as intended.

Impact: Low

Likelihood: Medium

Mitigation: The user interface of the application should be created with accessibility standards in mind, to ensure that users with disabilities are not prevented or disadvantaged when accessing the program's features.

6.1.6 Deployment

Issue: Since the development environment is different to the production environment, unforeseen bugs or errors may occur in a deployed product.

Impact: High

Likelihood: Low

Mitigation: It is essential to deploy the project as soon as a minimum viable product has been achieved, to give ample time to address any resulting issues.

6.1.7 Data Loss

Issue: Hardware failure or other sources of data loss would frustrate teachers who have added custom questions, and students that lost their score history.

Impact: High

Likelihood: Low

Mitigation: Frequent backing up, either on physical devices or using cloud-based services would minimise the damage caused.

6.2 Group Risks

These risks are specific to our development team

6.2.1 Knowledge Gaps

Issue: Since different members of the team have different programming skills, members of the team may find an aspect of their assignment to be too difficult to perform in the proposed timeframe.

Impact: Medium

Likelihood: High

Mitigation: Being honest and open about difficulties and requesting assistance other members of the team.

6.2.2 Group Member Exit

Issue: A group member leaving would increase the workload of other members, slowing development.

Impact: Medium

Likelihood: Low

Mitigation: The tasks of the former group member would be split among the remaining members to minimise the impact.

6.2.3 Technology Difficulties

Issue: The technologies chosen for development may turn out to be inappropriate for the task.

Impact: Low

Likelihood: Medium

Mitigation: A more developer-friendly front-end framework like Svelte, or a different backend language to PHP, could be introduced if required.

7. Summary

The proposed app is an innovative and genuine attempt to change chemistry education by utilizing the interactive advancements available in the digital age. The app seeks to empower students to improve their understanding of chemical science by offering an engaging, hands-on learning platform, helping them to grow their critical thinking capabilities as well as developing a greater fascination of chemistry.

Even though this is our core design concept and plan, our approach using Agile methodology can accommodate changes and updates where required, and there remains scope for further improvement.

We thoroughly believe that we can transform this plan into a working project and look forward to demonstrating its functionality to Instatute Learning Pty Ltd in due time.

8. Appendix

8.1 Abbreviations

DB: Database

InChI: International Chemical Identifier

JSON: JavaScript Object Notation

PHP: PHP Hypertext Preprocessor / Personal Home Page

SMILES: Simplified Molecular-Input Line-Entry System

SQL: Structured Query Language

SVG: Scalable Vector Graphics

8.2 List of figures and tables:

Figure 1: Example user interface sketch for the login page

Figure 2: Example user interface sketch for the welcome page

Figure 3: Example user interface sketch for question pages 1 and 10

Figure 4: Example user interface sketch for the results page

Figure 5: Fig. 5: User interface sketch for the results page, with the outcome of clicking the “Users” button

Figure 6: Fig. 6: Step-by-step description of the flow of execution of the program for students

Figure 7: Step-by-step description of the flow of execution of the program for administrators

Figure 8: Entity-relationship diagram describing the tables involved in the proposed database

Figure 9: Visual representation of the project timeline via a Gantt chart

Figure 10: A Trello board detailing individual team member tasks

Table 1: DB schema and an example record of USERS table. User Id is the primary key

Table 2: DB schema and an example record of REACTIONS table. Reaction Id is the primary key

Table 3: DB schema and an example record of STRUCTURES table. Structure Id is the primary key

Table 4: DB schema and an example record of SCORES table. Game Id is a primary key, User Id is a foreign key referencing User Id in the USERS table

Table 5: Proposed timeline for executing the tasks during project development

9. References

Andrew, R., Ullman, C., & Waters, C. (2003). Fundamental Web Design and Development Skills. *Glasshouse*.

Bienfait, B., Ertl, P. J. Cheminformatics, L. (2013). JSME: a free molecule editor in JavaScript. *SpringerNature*.

Delisle, M. (2006). Creating Your MySQL Database. *Packt Publishing Ltd*.

Full, M. (2020). How the Internet Works and the Web Development Process. *Independently Published*.

Gehani, N. (2011). The Database Application Book Using the MySQL Database System. *Apress Media, Llc*.

Landrum, G. (2013). RDKit.js. <https://www.rdkitjs.com/>

Nixon, R. (2021). Learning PHP, MySQL, and JavaScript. *O'Reilly Media, Inc*.

Prettyman, S. (2020). Learn PHP 8: using MySQL, JavaScript, CSS3, and HTML5. *Apress Media, Llc*.

Simpson, J. (2023). How JavaScript Works. *Apress Media Llc*.

Smith, B. (2015). Beginning JSON: [learn the preferred data format of the web]. *Apress Media, Llc*.

Solomons, T. W. G., Fryhle, C. B., Snyder, S. A. (2017) Organic chemistry, 12th ed. *John Wiley & Sons, Inc*.