Development of an Interactive Web-Based Platform for SQL Learning and Skill Enhancement

By

Ho Yin Kin

A REPORT
SUBMITTED TO
Universiti Tunku Abdul Rahman

in partial fulfillment of the

requirements for the degree

of

BACHELOR OF INFORMATION SYSTEMS
(HONOURS) INFORMATION SYSTEMS
ENGINEERING
Faculty of Information and Communication
Technology (Kampar Campus)

June 2025

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisor, Dr. Nur Syahirah Binti Roslan, for her unwavering guidance and support throughout the development of my Final Year Project entitled *"Development of an Interactive Web-Based Platform for SQL Learning and Skill Enhancement"*. This project has marked a significant milestone in my journey, allowing me to explore and build real-world applications using technologies like Node.js, MySQL, and interactive SQL execution. It has been a truly enlightening experience and a strong foundation for my future in web development and educational technology.

To a very special person in my life, Chang Zhu Zhi, for her patience, unconditional support, and love, and for standing by my side during hard times. Finally, I must say thanks to my parents and my family for their love, support, and continuous encouragement throughout the course.

COPYRIGHT STATEMENT

This Final Year Project proposal is submitted in partial fulfillment of the requirements for the degree of Bachelor of Information Systems (Honours) Information Systems Engineering at Universiti Tunku Abdul Rahman (UTAR). This Final Year Project proposal represents the work of the author, except where due acknowledgment has been made in the text. No part of this Final Year Project proposal may be reproduced, stored, or transmitted in any form or by any means, whether electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the author or UTAR, in accordance with UTAR's Intellectual Property Policy.

ABSTRACT

This project outlines the creation of SQL Quest, an interactive platform that aims to assist in learning SQL through gamification. SQL, or Structured Query Language, is essential for data management and analysis, however, most online learning platforms are not engaging, lack personalized feedback, or do not provide real-world relevance. SQL Quest seeks to provide a solution by integrating conventional content into interactive practices coupled with real-time feedback, an XP progress tracker, and challenges to motivate learners. The system is built with HTML, CSS, and JavaScript alongside web development software such as Node.js and MySQL. These components structure the platform using the Prototyping model which allows for iterative refinement. The platform boasts a few features including lesson modules with quizzes, practice SQL editors, multi-tiered challenges, dashboards, and a progress tracker. Users are also enabled to bookmark lessons, track lessons they have completed, and earn XP which unlocks badges. SQL comprehension and user engagement preliminary results indicate that user functionality and performance meet SQL Quest's objectives. The project has value in responsiveness, scalability, and user-centered design principles applied to SQL and computer science education. Further development five aims to incorporate a SQL Smart assistant for suggestion and correction tasks to enhance interactivity and incorporate industry-specific SQL application scenarios.

Area of Study (Maximum 2): Web-Based Learning, Gamified Education

Keywords (Maximum 5): SQL Learning, Gamification, Web Platform, Interactive Lessons, Progress Tracking

TABLE OF CONTENTS

TITLE PAGE ..... **i**
ACKNOWLEDGEMENTS ..... **ii**
COPYRIGHT STATEMENT ..... **iii**
ABSTRACT ..... **iv**
TABLE OF CONTENTS ..... **v-vii**
LIST OF FIGURES ..... **viii-x**
LIST OF TABLES ..... **xi**
LIST OF ABBREVIATIONS ..... **xii**
CHAPTER 1 INTRODUCTION 1-7
1.1 Problem Statement and Motivation 1-3
1.2 Objectives 3-4
1.3 Project Scope and Direction 5
1.4 Contributions 6
1.5 Report Organization 7
CHAPTER 2 LITERATURE REVIEWS 8-16
2.1 Review of the Technologies 8
2.1.1 Hardware Platform 8
2.1.2 Firmware / Operating System 8
2.1.3 Database 9
2.1.4 Programming Language 9
2.1.5 Algorithm 10
2.1.6 Conclusion of the Review of the Technologies 10
2.2 Solutions Currently Available in SQL Education 11
2.2.1 Codecademy - SQL Course 11
2.2.2 Khan Academy (Intro to SQL) 12
2.2.3 W3Schools (SQL Tutorial) 13
2.2.4 LeetCode (SQL Practice Problems) 14
2.2.5 SQLZoo 14
2.2.6 Summary of the Existing System 16
CHAPTER 3 PROPOSED METHOD 17-23
3.1 Methodologies and General Work Procedures 17-18
3.2 Technology Involved 20-21
3.2.1 Hardware 20
3.2.2 Software 21
3.3 User Requirements 22-23
3.3.1 Guest User Requirements 22
3.3.2 Registered User Requirements 22-23
3.3.3 General Requirements (All Users) 23
3.4 Timeline 224-25
CHAPTER 4 SYSTEM DESIGN /OVERVIEW 26-53
4.1 System Architecture Diagram 26-27
4.2 Block Diagram 28-29

4.3 Sitemap Diagram 30-31
4.4 Use Case Diagram 32-33
4.4.1 Login Account – Use Case Description 34
4.4.2 Register – Use Case Description 34
4.4.3 View Website- Use Case Description 35
4.4.4 View Lessons 35
4.4.5 Bookmark Lessons 35
4.4.6 Mark Lesson Completed 36
4.4.7 Track XP & Progress 36
4.4.8 Practice SQL 36
4.4.9 Submit Practice SQL 37
4.4.10 View SQL Feedback 37
4.4.11 Take Quiz 37
4.4.12 View Quiz Feedback 38
4.4.13 Open Challenges 38
4.4.14 Select Challenge Set 38
4.4.15 Select Level 39
4.4.16 Gain XP 39
4.4.17 View Real-World Projects 39
4.4.18 Use Chat Assistant (AI Tutor) 40
4.4.19 View Leaderboard 40
4.4.20 Theme Switcher (Dark/Light Mode) 40
4.5 Activity Diagram 41-52
4.5.1 Login Activity 41
4.5.2 Registration Activity 42
4.5.3 Lesson Interaction Activity 43
4.5.4 Bookmark Lesson Activity 44
4.5.5 Challenge Attempt Activity 45-46
4.5.6 XP & Badge Earning Activity 47
4.5.7 Profile & Progress Dashboard Activity Diagram 48
4.5.8 Leaderboard Activity 49
4.5.9 Theme Switcher Activity 50
4.5.10 Use Chat Assistant (AI Tutor) Activity 51
4.5.11 View Real-World Projects Activity 52
CHAPTER 5 SYSTEM IMPLEMENTATION 53-100
5.1 Hardware Setup 53-54
5.2 Software Setup 54-56
1. Web Browser 54
2. Backend Runtime Environment 55
3. Database Management System 55
4. AI Tutor and External Integrations 55
5. Front-End Development Tools 56
6. Additional Tools 56
5.3 Setting and Configuration 56-63

Step 1: Install Node.js and npm 56
Step 2: Install MySQL Database 60
Step 3: Configure SQL Quest Source Code 61
Step 4: Database Setup 61
Step 5: Start the Server 62
Step 6: Accessing SQL Quest 63
5.4 System Operation 64-99
5.4.1 Home Page 64-66
5.4.2 Register Page 67-69
5.4.3 Login Page 70-73
5.4.4 Lesson Page 74-80
5.4.5 Real-World Project Page 81-87
5.4.6 Chat Assistant (AI Tutor) Page 88-90
5.4.7 Challenge Page 91-97
5.4.8 Leaderboard Page 98
5.4.9 Profile Page 99
5.5 Implementation Issues and Challenges 100
CHAPTER 6 SYSTEM EVALUATION AND DISCUSSION 102-119
6.1 Black-Box Testing 102
6.1.1 Test Scenario: Home Page Functionality 102-103
6.1.2 Test Scenario: Register Page Functionality 103-105
6.1.3 Test Scenario: Login Page Functionality 105-107
6.1.4 Test Scenario: Lesson Page Functionality 107-109
6.1.5 Test Scenario: Real-World Project Page Functionality 109-111
6.1.6 Test Scenario: Chat Assistant (AI Tutor) Page Functionality 111-113
6.1.7 Test Scenario: Challenge Page Functionality 113-115
6.1.8 Test Scenario: Leaderboard Page Functionality 115
6.1.9 Test Scenario: Profile Page Functionality 116
6.2 Project Challenges 117
6.3 Objectives Evaluation 118
6.4 Concluding Remark 118
CHAPTER 7: CONCLUSION AND RECOMMENDATION 119-**121**
7.1 Conclusion 119
7.2 Recommendation 120
REFERENCES 121-123
APPENDIX A-1

LIST OF FIGURES

Figure Number	Title	Page
Figure 2.1	Codecademy - SQL Course Interface	11
Figure 2.2	Khan Academy - SQL Course Interface	12
Figure 2.3	W3Schools SQL Tutorial Interface	13
Figure 2.4	LeetCode SQL Practice Interface	14
Figure 2.5	SQLZoo Interactive Exercises	15
Figure 3.1	Prototyping Model	19
Figure 3.2	Timeline – FYP1	24
Figure 3.3	Timeline – FYP2	24
Figure 4.1	System Architecture Diagram of SQL Quest Platform	26
Figure 4.2	Block Diagram of SQL Quest platform	28
Figure 4.3	Sitemap Diagram of SQL Quest Platform	30
Figure 4.4	Use Case Diagram of SQL Quest Platform	32
Figure 4.5	Login Activity Diagram	41
Figure 4.6	Registration Activity Diagram	42
Figure 4.7	Lesson Interaction Activity Diagram	43
Figure 4.8	Bookmark Lesson Activity Diagram	44
Figure 4.9	Challenge Attempt Activity Diagram	45
Figure 4.10	XP & Badge Earning Activity Diagram	47
Figure 4.11	Profile & Progress Dashboard Activity Diagram	48
Figure 4.12	Leaderboard Activity Diagram	49
Figure 4.13	Theme Switcher Activity Diagram	50
Figure 4.14	Use Chat Assistant (AI Tutor) Activity Diagram	51
Figure 4.15	View Real-World Projects Activity Diagram	52
Figure 5.1	Node.js Official Website (Homepage)	57
Figure 5.2	Node.js Download Page	57
Figure 5.3	Node.js Installer Downloaded	58
Figure 5.4	Node.js Custom Setup	58

Figure 5.5	Tools for Native Modules	59
Figure 5.6	Ready to Install Node.js	59
Figure 5.7	Verifying Node.js and npm Installation	60
Figure 5.8	MySQL Community Server Download Page	60
Figure 5.9	MySQL Workbench Download Page	61
Figure 5.10	MySQL Workbench with Database Creation	62
Figure5.11	Home Page with Sidebar Navigation	64
Figure5.12	Home Page with Theme Toggle (Light Mode)	65
Figure5.13	Home Page After Login with User Statistics	66
Figure 5.14	Registration Form with Validation Errors	67
Figure 5.15	Successfully Completed Registration Form	68
Figure 5.16	User Data Stored in Database	69
Figure 5.17	Login Page Interface	70
Figure 5.18	Invalid Email Format Warning	71
Figure 5.19	Empty Field Validation Warning	72
Figure 5.20	User Not Found Error	73
Figure 5.21	Lesson Page Structure	74
Figure 5.22	Lesson Page with Login Features	75
Figure 5.23	Bookmarking a Lesson	76
Figure 5.24	Removing a Bookmark	76
Figure 5.25	Interactive SQL Query Editor	76
Figure 5.26	Query Execution Result	77
Figure 5.27	Practice Section with Error Feedback	77
Figure 5.28	Practice Section with Partial Feedback	78
Figure 5.39	Correct Practice Query Execution	78

Figure 5.30	Quiz Section with Explanation	79
Figure 5.31	Quiz Completion Feedback	79
Figure 5.32	Lesson Completion Status	80
Figure 5.33	Explorer page of the Real-World project	81
Figure 5.34	Full Example Page	82
Figure 5.35	Use Case Overview	83
Figure 5.36	Sample Queries Section	84
Figure 5.37	Entity Relationship Diagram (ERD) Access	85
Figure 5.38	Column Relationship (ERD) Visualization	85
Figure 5.39	Mock Database Viewer – Viewing a Single Table	86
Figure 5.40	Mock Database Viewer - Relations of Tables	87
Figure 5.41	Briefing of the AI Chat Box	88
Figure 5.42	Example SQL Code Assistance	89
Figure 5.43	Handling Off-Topic Queries	90
Figure 5.44	Challenge Sets Overview	91
Figure 5.45	Challenge Level Selection	92
Figure 5.46	Challenge Question Interface	93
Figure 5.47	Correct Answer Feedback	94
Figure 5.48	Incorrect Answer Feedback	95
Figure 5.49	Chapter Completion Screen	96
Figure 5.50	Path Choice on Completion	97
Figure 5.51	Leaderboard Page	98
Figure 5.52	Profile Page	99

LIST OF TABLES

Table Number	Title	Page
Table 2.1	Comparative Analysis of SQL Learning Platforms	16
Table 3.1	Hardware Requirements	20
Table 3.2	Software Requirements	21
Table 6.1	Home Page Functionality	102
Table 6.2	Register Page Functionality	103-104
Table 6.3	Login Page Functionality	105-106
Table 6.4	Lesson Page Functionality	107-108
Table 6.5	Real-World Project Page Functionality	109-110
Table 6.6	Chat Assistant (AI Tutor) Page Functionality	111
Table 6.7	Challenge Page Functionality	112-113
Table 6.8	Leaderboard Page Functionality	114
Table 6.9	Profile Page Functionality	115

LIST OF ABBREVIATIONS

SQL	Structured Query Language
AI	*Artificial Intelligence*
API	*Application Programming Interface*
CRUD	*Create, Read, Update, Delete*
CSS	*Cascading Style Sheets*
DBMS	*Database Management System*
ERD	*Entity Relationship Diagram*
FYP	*Final Year Project*
GPU	*Graphics Processing Unit*
HR	*Human Resources*
HTML	*HyperText Markup Language*
IDE	*Integrated Development Environment*
JS	*JavaScript*
JSON	*JavaScript Object Notation*
LTS	*Long-Term Support*
MVC	*Model View Controller*
MySQL	*Structured Query Language (MySQL Database System)*
NLP	*Natural Language Processing*
npm	*Node Package Manager*
OS	*Operating System*
RAM	*Random Access Memory*
RDBMS	*Relational Database Management System*
SQL	*Structured Query Language*
UI	*User Interface*
UX	*User Experience*
XP	*Experience Points*

CHAPTER 1 Introduction

This chapter introduces the project, outlining the problem it addresses, the motivation behind it, the specific objectives, the project's scope and direction, and the contributions.

1.1 Problem Statement and Motivation

Problem Statement

Organizations ranging from businesses and financial institutions to hospitals and online retailers to software firms and universities all generate data in the form of relational databases and use SQL as the language to interpret that data. The SQL language is widely considered to be the "operational" aspect of data engineering and analysts. It is used to retrieve, manipulate, report and optimize data providing an indispensable skill to those focused on data. However, learning SQL is still a barrier to entry.

Classical pedagogy like textbooks, lecture-style instruction on SQL or fixed-content online tutorials, teach to understanding of the SQL syntax, but are usually insufficient to fill the gap between theory and practice usage. Learners often find it hard to make the jump from solving simple textbook problems to solving complex and large-scale SQL problems in real world - problems such as joints, optimization, applying constraints, designing structured databases etc.

There are learning platforms for SQL which attempt to resolve this issue, however due to many constraints they cannot be used practically:

- **Non-interactive and Non-Engaging:** Many sites teach SQL as if it were any other subject, resulting in a very static training where students don’t get a chance to play around with the contents of the lessons. Learners cannot experiment interactively with queries and do not immediately see the effects of their queries on a (simulated) database.

- **Lack of feedback:** Current tools generally yield only a binary decision (correct/incorrect) with minimal explanation. This leaves students feeling confused about errors and leads to stagnation in their development of skills.

- **Lack of Customizations and Gamification:** The learner population for SQL is broad, from beginners to intermediate to even professional levels. Most platforms are one-size-fits all, not personalizing challenges to a user's skill level. And without any of that gamified bullshit- it’s hard to stay interested

- **Real-World Applications Disconnect:** A lot of Time Sometimes, a language/platform only focuses on teaching basic syntax or C.R.U.D. operations. They are incomplete without hands-on real-world problems from fields of customer relationship management, finance, inventory systems, and HR databases. This is a limitation which prevents students from verifying or see how SQL is being used in industry.
- **Lack of AI-Boosted Assistance:** Students sometimes demand in-time assistive recommendations after their input fails or concepts are unclarified. Traditional momigo systems are gap-less in that they do not provide contextual hints, validation or query implementation advice with intelligent tutoring systems being an exception.

These deficiencies are causing learners to enter the industry unprepared to work with SQL in professional environments where databases are high-stakes, complicated, and focused on performance.

Motivation

This project has been motivated by the great necessity for providing students with an adaptive, entertaining, and applied educational support for SQL that matches up to professional use-cases of SQL. The developed platform -SQL Quest- fills this gap by integrating AI driven tutoring, gamification and real-world cases in a single and integrated learning environment.

Key motivating factors include:

- **AI Based Tutor and Validator:** Leveraging AIML and ChatGPT API, the system has implemented an interactive SQL tutor which can explain, validate the query and provide the correction while the user is typing. This closes the feedback loop in traditional systems.
- **Gamified to Foster Engagement:** also encourages sustained motivation with XP Tracker, Ranks, Badges, and an Adult Leaderboard to spark friendly competition in a supportive manner.
- **Customized Progression:** Students can bookmark lessons, mark modules as completed and unlock progressive challenges (from the fundamentals to advanced SQL such as joins, subqueries, views, indexes and constraints), which cater to their custom learning pace and ability

- **Hands-On Projects:** The platform projects including CRM, Finance, HR, Inventor, Supplier databases make SQL practice real. This guarantees that students learn more than just syntactic knowledge, that is, the trainees should also be trained in domain-specific problem solving.
- **Interactive Curriculum Framework:** Inside every lesson, we have descriptions, syntax, examples with mock data, interactive editors with validators and education exercises mixing course theory with real world practice**Accessibility and Engagement:** Learn SQL in a less intimidating environment with gamified lessons that are detailed yet concise — everything you need to succeed inside a retro-themed UI that eliminates cognitive load with audio cues, dark/light themes, on-screen memes, gamified responses and more.

In conclusion, the purpose of this research is to fill the perpetual void between theoretical SQL knowledge and practical, day-to-day adeptness in usage. Through AI tutoring, gamification, and contextual case studies, the proposed platform seeks to create learners who not only can code SQL but also can solve professional database problems with confidence.

1.2 Objectives

The main aim of this project is to develop **SQL Quest**, an advanced learning service for SQL which corrects the deficits of the available solutions and enhances the learning experience, and subsequently the SQL skill of the users. The system is intended to not just teach SQL syntax, but to train the ability of students to apply SQL confidently in real-world situations.

The project is underpinned by a number of sub-objectives designed to deliver upon this overarching aim:

An Interactive Learning Environment for SQL

- **Objective:** Create an easy-to-use platform where users can learn SQL in a guided, interactive way.
- **Contribution:** This facility encourages more participation of students in learning as they can run queries on mock databases at runtime and see the results immediately. This practical aspect makes the platform much more interactive and powerful as opposed to the standard static learning methods.

Integration of Real-World Scenarios

- **Goal:** To include real-world-like exercises that simulate day-to-day functions of database systems such as Customer Service, Finance, HR, Inventory, Sales and Management Reporting systems.
- **Contribution:** Learners will be job-ready from "on the job" through exposure to realistic case studies—not just syntax. This allows students to put their learning into context and leverage SQL in the industry

A Detailed Feedback System to be put in Practice

- **Goal:** To provide feedback giving immediate, concrete, and actionable information on user requests. And the system should consider not just correctness, but also performance and best practices.
- **Contribution:** Learners will be able to identify mistakes, appreciate different approaches, and enhance problem-solving abilities. The ontological feedback loop is that enhancing so much better than the fully binary response so many systems currently provide me!

Games for Reward or Competition and their Effect on Motivation and Performance

- **Goal:** Maintains engagement by including elements familiar from games, such as XP tracking, badges, achievements, ranks, and leaderboards, throughout the learning process.
- **Contribution:** Making SQL learning a game increases and encourages practicing on a more regular basis. It also encourages competition and motivation.

AI-Powered Tutoring and Personalization

- **Aim:** Use AIML and ChatGPT APIs to generate an intelligent tutor system, offering contextual hints, query validation, and corrective guidance, and choosing exercises based on students' level of learning.
- **Contribution:** This will enable learners to get the proper amount of help at the appropriate times. New students can be steered with step-by-step instruction, or existing students can be pushed with difficult problems to create an individual path for progression.

1.3 Project Scope and Direction

This project will focus on the design, development, and deployment of **"SQL Quest"**, a web-based interactive educational environment for SQL. The system is intended to address the shortcomings of current SQL learning systems by providing a more effective, engaging, and intuitive approach to learning SQL with a focus on practice and application to real-world scenarios.

The project does the following explicit

Interactive Learning Platform: A live SQL editor that allows learners to write and run SQL queries on dummy databases in real time to promote a learn-by-doing approach.

- **Simulated Real-World Scenarios:** CRM, Finance, HR, Inventory, and Supplier databases from industry to enable learners to solve problems as faced by professionals.
- **AI Tutor and Validator:** AIML-based, ChatGPT-powered SQL tutor that validates queries, highlights error, recommends improvements, and gives hints using the context.
- **Detailed Feedback:** Unlike regular yes-or-no answers, the program provides instant, constructive feedback to see if you wrote what's correct, if it's efficient, and even shows the best way to solve the problem.
- **Gamification Elements:** Points built in teams, badges, awards, ranks, and leaderboards to motivate learners, engage them, and foster friendly competition.
- **Profile and Progress:** Personal dashboards help you keep track of your curriculum progress by showing what lessons you have completed and what you have bookmarked for later.
- **User Interface Design:** A responsive retro-style interface with dark/light modes, animations, and audio feedback helps make the system accessible, reduce cognitive load, and create a fun place to learn.
- **Lesson Modules and Quizzes:** Lessons with explanations, syntax demonstrations, for example data, interactive editors, quizzes, and real-world tasks to practice what you have learnt in practical labs.

1.4 Contributions

The project contributes significantly to SQL education and to the overall field of educational technology by integrating current pedagogical approaches into an interactive learning environment.

These include:

- Enhanced Learning Experience: The platform will allow for the creation of a more dynamic learning environment in which users can interactively practice SQL queries with real-time feedback that adjusts the learning pace. Such an approach will foster not only more active involvement but also better retention by incorporating practical, problem-based learning scenarios reflective of real-world challenges.
- Real-World Scenario Simulation: The platform will be developed to allow modules of real-world SQL competencies, including querying large datasets, planning schema changes, and optimizations in query performance. Simulations offer learners a place where they can apply academic theory into practical applications and bridge the gap in their learning to professional SQL usage.
- Adaptive Learning Paths: It shall have an adaptive learning system that will automatically tailor the difficulty of the content to fit a user's skill level. This will protect novice users from being overwhelmed while advanced learners are constantly challenged. Thus, it will satisfy both sets of users: those who started with SQL yesterday and seasoned professionals who want to fine-tune their expertise.
- Open-Access Educational Resource: Most importantly, the platform is to be an open-access tool, intended for the wide facilitation of education in SQL among students, pros, and educators alike. By offering free resources, it would be a contribution toward initiatives meant to democratize technical education so that more people get a chance to pick up this essential skill in SQL.

1.5 Report Organization

This report is organized into 7 chapters. Chapter 1 provides an introduction and overview of the project including the background, problem statement, objectives, scope, contribution, and structure of the report. In Chapter 2, the background of the work is introduced in a manner of literature review style which first gives a survey in terms of technology including hardware platform, operating system, database, programming language, and algorithms, as well as the comparison of current SQL learning platforms in aspects of features, limitation and working efficiency, and presents what this project is aimed for that have not existed to fill the gap of the market. 3 System Methodology This chapter explains the overall system methodology which includes overall architecture, use case diagrams, activity diagrams that describe workflow of the platform. System Design is presented in Chapter 4, which describes block diagram, parameters of the system's elements and the way the components interface with each other in such a way that a future reader can re-build the project. Implementation In this chapter we describe the system implementation which includes hardware and software setups, configuration, demonstration of system operations with screenshots, and problems faced and challenges we faced during development. Chapter 6 discusses evaluation and discussion, which includes system testing, performance measures, testing results, issues and evaluation of the project's objectives to determine if the project being effective. Chapter 7: Summary and conclusion of the report concludes with Chapter 7 where the findings of the report are summarized, and suggestions are given on how the SQL Quest platform could be extended in the future

CHAPTER 2 Literature Reviews

In this chapter, we consider solutions developed by other researchers and developers to address challenges in SQL education. We analyze the strengths and weaknesses of such solutions and suggest ways through which the identified limitations can be addressed. This analysis forms the basis of developing a more effective SQL learning platform.

2.1 Review of the Technologies

This subsection presents the technologies on which SQL Quest is implemented. Every technology we chose was chosen for its applicability, extensibility, and relevance when developing an interactive, gamified SQL learning platform.

2.1.1 Hardware Platform

SQL Quest is developed on a modern laptop with a CPU **12th Gen Intel(R) Core (TM) i5-12400F @ 2.50GHz**, memory **16GB DDR4 RAM (3200 MHz)**, and storage of **KLEVV CRAS C710 M.2 NVMe SSD 1TB**. These specifics made multitasking during development a breeze, allowed the example front-end animations to run silky smoothly, and the database transactions could only be described as warp speed.

An **NVIDIA GeForce RTX 4060 GPU** was also added for rendering complex, dynamic retro user interfaces and gamified SFX. Coding and testing were supported by input peripherals including a standard keyboard and a mouse to facilitate debugging.

There are minimal hardware requirements for end-users as it is a web-based solution, only needing a device with access to the internet and a recent browser.

2.1.2 Firmware / Operating System

I upgraded the host environment from **Windows 10 to Windows 11** to provide the most stable and compatible environment possible for all development tools. Both of these were tested. SQL Quest was also tested on several OSs to ensure it was truly cross-platform (Windows, Ubuntu).

And since the system is browser-based, front-end operations are not OS dependent. All you need is a modern browser like **Google Chrome, Mozilla Firefox or Microsoft Edge**. On the backend runtime side of things, the platform uses the **Node.js engine** that is both platform independent and uniform across varying OS flavors.

2.1.3 Database

We have selected **MySQL** as the backend database technology for SQL Quest, which is one of the most popular RDBMS. MySQL was chosen because of its open-source technology, easy deployment, and adherence to the SQL standard.

In SQL Quest, you will use MySQL for the two:

1. **Presentation databases:** A few domain-specific databases (CRM, Finance, HR, Inventory, and Supplier systems) were developed to imitate real-world situations for learning purposes. Learners can use these databases to query and practice, in a way that simulates real industry issues.
2. **System Database Management:** In addition, MySQL is another example for your user data (profiles, logins, tracking progress, lessons finished, bookmarks, XP, leaderboard scores).

The scalability of MySQL and the presence in professional use is a further reason why to own be a part of SQL Quest.

2.1.4 Programming Language

JavaScript is the language in which SQL Quest is programmed, mainly because it is suitable for frontend and backend work. The frontend is based on **HTML5, CSS3 and JavaScript**, with **Bootstrap 5** for responsive design and retro-styled custom CSS.

Node.js with Express.js aims to build routing, API calls, and middleware configuration more simply. Other pieces of JavaScript laying around cover gamification (XP, badges, achievements), AI tutoring integration, and progress tracking. Supporting libraries, like **bcrypt** (for secure password hashing), **dotenv** (for environment configuration), and **mysql2** (for database connectivity), add to the usability of the system.

2.1.5 Algorithm

A few algorithms are used in SQL Quest to create its special learning system:

- **SQL Validation Algorithm:** Evaluates the user queries to assure the correctness of the queries, efficiency, and syntax conformance. It provides students with instant and nuanced validation over and above binary feedback.
- **AI Feedback Algorithm:** The system uses Ci-wi AI to interpret queries and identify mistakes from query using AIML and ChatGPT APIs. NLP algorithms are employed to process queries and provide context-sensitive hints/corrections.
- **Gamification Algorithms:** With the ability to determine XP points, assign ranks, unlock badges, and keep the leaderboard live or real-time, learners are motivated thoroughly.
- **Progress Monitoring Algorithm:** The tracking technology gauges lessons completed, bookmarked modules, and quiz results, setting a personal dashboard interface on the course based on the real-time learning material.
- **Security codes:** Add password hashing (bcrypt) and session handling to protect the user account and support the secure login.

2.1.6 Conclusion of the Review of the Technologies

All in all, the technologies in the SQL Quest stack were considered enough to be succinct, functional, and easy to scale. The hardware was smooth for development and testing, but we have chosen Windows OS and Node.js runtime and have guaranteed compatibility across environments.

MySQL was employed as a teaching as well as a system management database, allowing the learners to have realistic practice. JavaScript with Node.js and Express.js is also the most practical and flexible choice for the front and back-end development. The validation, AI tutoring, gamification, and security algorithms used in SQL Quest differentiate it from the typical static SQL tutorials. Together, they make a resilient basis to build an interactive, gamified, and AI-enhanced SQL learning system.

2.2 Solutions Currently Available in SQL Education

Over the past decade, numerous mechanisms have developed online to help meet the demand for SQL education. This section reviews five leading tools to determine approaches, strengths, and weaknesses relevant to SQL education.

2.2.1 Codecademy - SQL Course

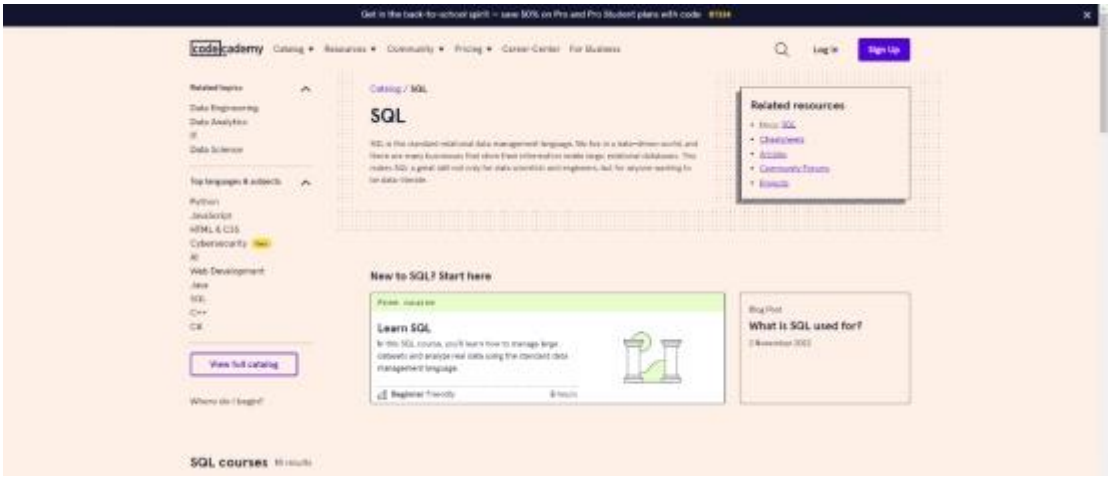


Figure 2.1Codecademy - SQL Course Interface

Codecademy is a web-based online platform to learn programming languages; SQL is no exception. It is an interactive, beginner-friendly environment where SQL queries can be practiced live in the browser. The system is designed to be interactive and accessible, thus allowing learners to get immediate feedback on the queries they build [6].

Strengths:

- **Interactivity**: The good thing with Codecademy is that it allows for hands-on practice right away, as you will write and execute SQL queries [6].
- **Beginner-friendly**: Courses are structured in such a way that they can accommodate a full newbie; all the concepts are explained step by step to lay the foundation of the learner in SQL [6].
- **Accessibility**: Because Codecademy has both free and paid content, it is accessible to the mass market [7].

Weaknesses:

- **Limited Depth**: The platform does not go in-depth into more advanced topics of SQL, leaving the experienced user under-challenged [7].
- **Real-World Application**: Most of the exercises are highly hypothetical in nature and fail to provide any simulation of real-world SQL themselves, so the practical applicability is very limited [7].

Improvement Suggestion: Add more complex, real-world scenarios to the exercises that

Codecademy creates; this would build more practical knowledge.

2.2.2 Khan Academy (Intro to SQL)

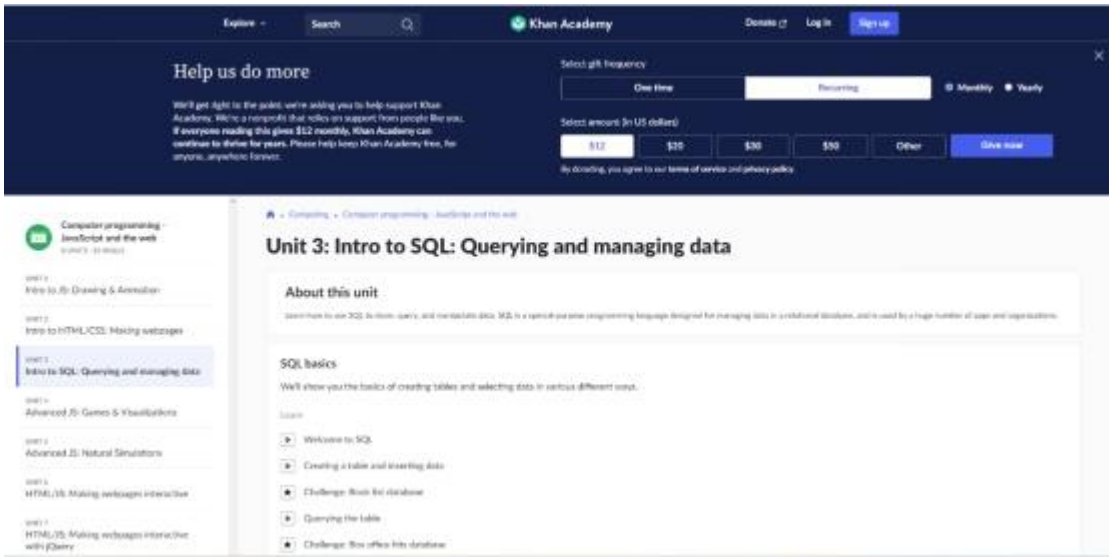


Figure 2.2Khan Academy - SQL Course Interface

One free introductory course to SQL is available on the Khan Academy website. It covers the basics, mixing video tutorials with interactive exercises. The platform is highly accessible, with high educational quality; it is thus very popular among learners new to SQL [8].

Strengths:

- **Free Access**: Everything on Khan Academy is free, which allows equal opportunity for all learners who cannot afford to pay for other courses anywhere else in the world [8].
- **Video-Based Learning**: It covers video lessons to explain SQL concepts in bite-sized chunks, assisted by interactive practice exercises [9].
- **Basics covered**: The course covers the basics of SQL; therefore, it would be recommended for beginners [8].

Weaknesses:

- **Lacking Advanced Content**: The SQL course on Khan Academy does not extend to more complicated topics, thus decreasing its helpfulness for learners beyond the basics.
- **Limited Real-World Context**: The exercises provided are straightforward and do not incorporate complex, real-world scenarios, reducing their practical application in professional settings [9].

Improvement Suggestion: Khan Academy can further develop the course by the addition of more advanced topics and inclusion of exercises connected with real situations, making it more practical.

2.2.3 W3Schools (SQL Tutorial)

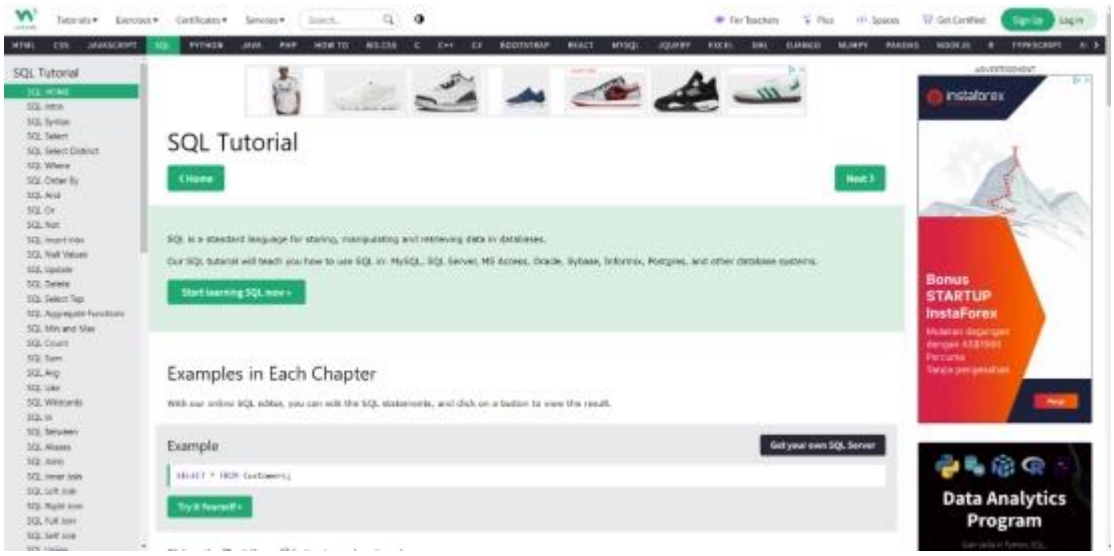


Figure 2.3 W3Schools SQL Tutorial Interface

W3Schools is an online educational portal about various web technologies, including SQL. More of a reference tool, it consists of a full SQL tutorial which covers a wide variety of topics [10].

Strengths:

- **Full-Text Content**: W3Schools covers SQL from basic queries up to some complex commands and hence, is helpful for both beginners and advanced learners [10].
- **Interactive Examples**: The reader can interactively try SQL queries using an inbuilt editor with immediate feedback [10].
- **Free and Accessible**: Because the platform is freely available to every individual on the network, its usage spanning across the globe has become predominant [11].

Weaknesses:

- **Not Guided Learning**: W3Schools is more of a reference site than it is for structured learning, which may cause confusion among beginners who need some guided learning.
- **No Adaptive Features**: It does not adapt content to the learner's skill level, which might result in less personalization in learning itself [10].
- **Lack of Real-World Scenarios**: Although the tutorial provides a lot of information, the exercises are rather elementary and do not come close to the reality of everyday tasks in SQL [10].

Improved Suggestion: W3Schools must come up with a learning tour which would lead the learner through this SQL tutorial in a more orderly manner. Besides, it will also be more practical if some exercises that are applicable to real-world scenarios can be added to it.

2.2.4 LeetCode (SQL Practice Problems)

Figure 2.4 LeetCode

LeetCode is primarily a platform for people to prepare for their coding interviews; however, it also hosts an extensive set of SQL problems. Most of these problems vary in difficulty and are targeted toward real-world scenarios and, thus, would be extremely valuable to more advanced learners looking to further develop their skills in SQL [12]

Strengths:

- **Real-World Scenarios Simulation**: LeetCode contains SQL problems similar to those encountered in real life, which provides experience for the learner in solving practical database problems [12]
- **Variety in Difficulty Scale**: The platform contains a wide variation in problems, ranging from easy to hard, to accommodate users of different skill levels [12]
- **Community Engagement**: LeetCode involves the community in active discussion of problems, and solution sharing, which eventually develops collaborative learning [13]

Weaknesses:

- **Not for Beginners**: LeetCode assumes familiarity with SQL, and therefore applies more appropriately to the intermediate to advanced learner, not a beginner at all [12]
- **Lack of Instructional Content**: This is primarily a problem-solving site with limited instructional content, so the users probably must use some other resources to understand the concepts being implemented here [13]

Improvement Suggestion: LeetCode can introduce introductory content and a beginner's track to make it friendlier for beginners. In this regard, more instructional content should be integrated to help them understand the ideas behind the queries.

2.2.5 SQLZoo

Figure 2.5 SQLZoo Interactive Exercise

SQLZoo is a free interactive SQL tutorial developed to teach advanced SQL through exercises. It is one of the older platforms available, but it continues to be popular because of the comprehensiveness of its coverage and its ease of access.

Strengths:

- **Absolutely Free**: SQL Zoo is free; hence, it is affordable for the mass public with no cost implications whatsoever from the individual's perspective [14]
- **Interactive Exercises**: The SQL queries practice is provided within the platform interactively, allowing learners to practice the material directly [15]
- **Any Topic under the Sun**: SQL Zoo covers a large spectrum of SQL topics, thus providing almost the perfect background for learners in virtually any stage of their education [14]

Weaknesses:

- **Outdated Interface**: The interface adopted by the platform is a bit dated and may be off-putting for more modern learners [14]
- **Limited Feedback**: Although immediate feedback may be provided regarding the queries, an explanation or other ways of doing something might not be given; thus, SQL Zoo provides limited feedback that restrains deep understanding from the users' side.

Improvement Suggestion: SQL Zoo will enhance its website by providing an improved UI and offering detailed feedback regarding SQL queries.

2.2.6 Summary of the Existing System

The review of existing SQL learning platforms highlights that while systems such as Codecademy, Khan Academy, W3Schools, LeetCode, and SQLZoo have each contributed to SQL education, they also present considerable limitations. Codecademy and Khan Academy are beginner-friendly but lack advanced real-world coverage. W3Schools and SQLZoo provide comprehensive SQL content but are primarily reference-based and limited in interactive feedback. LeetCode focuses on practical and challenging problems but is unsuitable for beginners.

The comparative study in **Table 2.1** summarizes the strengths and weaknesses of these platforms against the proposed system, SQL Quest.

Table 2.1 Comparative Study of SQL Learning Platforms

Feature	Codecademy	Khan Academy	W3Schools	LeetCode	SQLZoo	Proposed System (SQL Quest)
User login & profile tracking	✓	✓	✗	✓	✗	✓ (profiles, bookmarks, dashboards)
Interactive learning	✓	✓	✓	✓	✓	✓ (real-time SQL editor, quizzes)
Real-world scenario exercises	✗	✗	✗	✓	✗	✓ (CRM, Finance, HR, Inventory, Supplier)
Comprehensive SQL coverage	✗ (basic)	✗ (basic)	✓	✗ (focus on problems)	✓	✓ (basics to advanced joins, views, constraints, indexes)
Feedback mechanisms	✓ (limited)	✓ (limited)	✗	✗	✗	✓ (AI tutor, SQL validator, detailed explanations)
Beginner-friendly	✓	✓	✓	✗	✓	✓ (step-by-step guided lessons)
AI Support	✗	✗	✗	✗	✗	✓ (AIML + ChatGPT tutor with contextual hints)
Gamification	✗	✗	✗	✗	✗	✓ (XP, badges, ranks, achievements, leaderboard)
Free access	✗ (partial)	✓	✓	✗ (paid premium)	✓	✓ (open access)

CHAPTER 3 System Methodology/Approach

3.1 Methodologies and General Work Procedures

To fulfil the goals of SQL Quest, the project followed the Prototyping Model as its development approach. It focuses on creating an early prototype, involving the users to test the system and provide feedback, and refining the product with iterative iterations until it fulfills its functional and non-functional needs. The Selection of the Prototyping Model was selected, as it matches the kind of educational technology projects where the usability, interactivity and learner experience are just as important than technical accuracy.

The developmental phase commenced with the planning and analysis component of existing SQL learning sites including Codecademy, Khan Academy, W3Schools, LeetCode and SQLZoo. Comparative pre-testing results revealed several shortcomings of current systems: lack of interactivity, limited real-world practice, non-adaptive feedback, no gamification features and the absence of AI tutoring. These results were used as a starting point for SQL Quest's first requirements.

Design Low fidelity wireframes and sketches were developed during the design process for the user registration, SQL editor, lesson pages, and profile dashboard. It was a simple, retro and user-friendly design. Wireframes evolved into interactive prototypes that included a facsimile of the learning experience itself: logging in, jumping to lessons, posing questions and getting instant answers.

The prototype phase of the Available only as per terms of use Project involved the development of key features using HTML5, CSS3, JavaScript and Node. js/Express, and MySQL. User authentication, lesson module creation, submitting queries and progress were the basic functionalities implemented at this point. The dev team performed internal tests on those prototypes for accuracy, ease of use, and responsiveness.

Afterwards, the testing and feedback stage went on involving actual users (students and colleagues) who used SQL Quest to complete lessons or challenges. Feedback was gathered for interface usability, clarity of instructions, accuracy of SQL validation and engagement. In response to this feedback, iterative improvements including integrations with AI Tutor (ChatGPT + AIML APIs for contextual hints and query validation), gamification mechanics (XP tracking, badges, achievements and leaderboard), saving book lesson for personalized learning and the responsive retro-styled UI with dark/light themes and sound were made.

During the iteration phase, more modules representing physical SQL databases were included

in CRM, Finance, HR, Inventory and Supplier domains. These simulations turned SQL Quest into an educational platform that reflects real life use-cases for industry. The course also contributed to challenging problems beyond the basic CRUD operations, such as database optimization challenges, schema design tasks and query performance practices.

Lastly, the execution and deployment phase had an up-and-running SQL Quest application. The front end was developed with HTML5, CSS3, JavaScript and styled using Bootstrap 5 & Custom CSS. The backend was built with Node.js with Express.js, which handles backend API calls, gamification logic and interface with external AI services. Both mock datasets (query practices) and systems data were stored in MySQL database, including user account information, learning progress of lessons, bookmarks for favorite lessons or recommended programs, XP points and leaders' boards results. This system was tested end-to-end for functionality, speed, and security before being turned into an entire web application.

The cycle of Prototyping Model design, feedback, revision and deployment is depicted in Figure 3.1 showing how from sketches SQL Quest related into a complete gamified AI based on learning environment for SQL. Here we describe multiple iterations through which the final platform was developed so that it would be technically feasible but also engaging and user-centered in terms of learning.

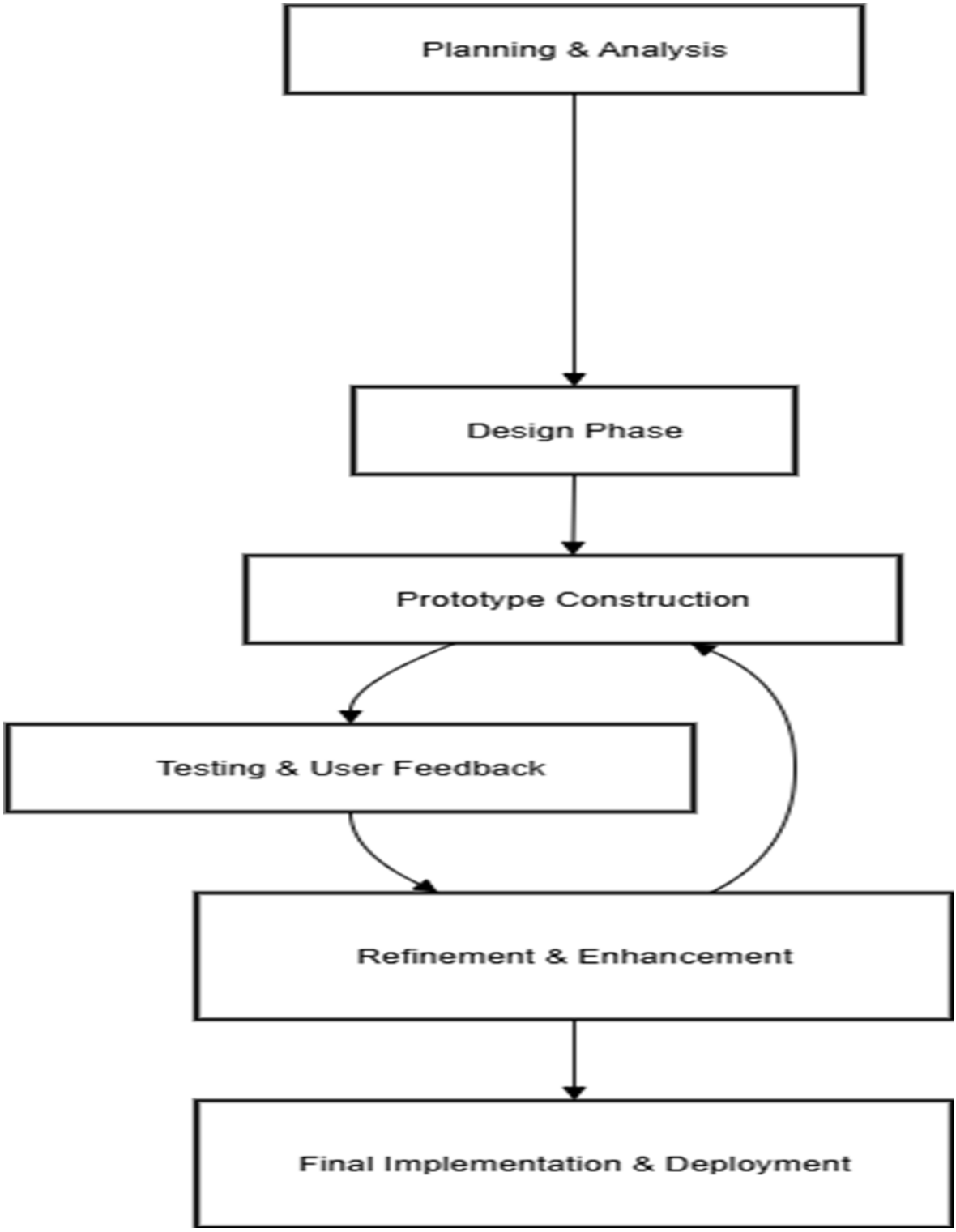


Figure 3.1 Prototyping Model

3.2 Technology Involved

3.2.1 Hardware

The SQL Quest platform is developed and implemented on a modern, custom laptop capable of full-stack web development. Development system execution and multitasking performance is sufficiently maintained by the 12th Gen Intel(R) Core (TM) i5-12400F @ 2.50GHz processor. Complex front-end designs and animations are smoothly loaded through the NVIDIA GeForce RTX 4060 graphics card enabling unparalleled complex and dynamic front-end interface DE Simultaneous execution of development environments as well as backend servers and database systems is made possible by the 16.0 GB DDR4 3200 MHz RAM with no lag on the system. Application compilation, deployment, and database transaction handling is further augmented with system storage from KLEVV CRAS C710 M.2 NVMe SSD 1TB due to rapid read/write speeds. Easy operation during the whole process of development, coding, testing, and navigation is enhanced through standard keyboards and mice peripherals throughout the system.

Table 3.1 Hardware Requirements

Hardware	**Specifications**	**Description**
Processor	12th Gen Intel(R) Core (TM) i5-12400f @ 2.50GHz	Used for executing and managing the overall performance of the system.
Graphics	NVIDIA GeForce GTX 4060	Used to present graphical content and interface.
Memory	16.0 GB DDR4 3200 MHz RAM	Used for program execution and data processing.
Storage	KLEVV CRAS C710 M.2 NVMe SSD 1TB	Provides fast storage access and efficient data management.
Input Device	Mouse and Keyboard	Used to control the system during development and program execution.

3.2.2 Software

The chosen software environment for the project provides concrete application development, compatibility, and scalability. The current operating system in use is Windows 11, updated from Windows 10, which achieves stability for all development tools and server software needed. The primary web browser employed for testing front-end applications is Google Chrome, allowing seamless verification of cross-platform rendering and responsive design.

The IDE selected for the project is Visual Studio Code (VS Code 2022), which offers numerous plugins for Node.js, MySQL, and JavaScript debugging. The SQL Quest platform is front end with HTML5, CSS3, and JavaScript, with Bootstrap 5 utilized for styling. The managed component front end is controlled by React.js (latest stable version). For managing server operations and RESTful APIs, Express.js runs on the server-side with Node.js. The relational database is managed using MySQL Workbench. It stores user profile data, lesson content, progress on challenges, XP, and feedback records.

Table 3.2 Software Requirements

Software	Specifications	Description
Operating System	Windows 10	Provides the necessary environment for running the laptop and software programs.
Web Browser	Google Chrome	Used for displaying and testing the platform's output during development and use.
IDE	Visual Studio Code (2022)	Main tool for front-end, backend, and database coding and debugging.
Front-End Tools	HTML, CSS, JavaScript, Node.js, Express.js	Used for creating and designing the graphical user interface of the platform.
Database	MySQL Database	Used to store, manage, and retrieve user data, learning modules, and progress information.

3.3 User Requirements

The SQL Learning Platform has two types of users which are: **Guest User** and **Registered User**. Both have a few basic features in common, but as Registered Users you can play for points and maintain your progress, earn XP and compete on leaderboards.

3.3.1 Guest User Requirements

Guest Users can access the platform without registration but with reduced actions and spaced in time:

- **View Website**: Visit our SQL Quest home page and navigation menu.
- **View Lessons**: Take free classes in areas like Beginner, Intermediate, Relational databases and Database Management System.
- **Try Example SQL**: Have fun with example queries in demo mode.
- **Practice SQL**: Use the practice editor to write, run, and share queries.
    - **Submit Practice SQL**: Run queries on a mock database.
    - **View SQL Feedback**: Get instant response including results or errors.
- **Take Quiz**: Attempt quizzes related to lessons.
- **View Quiz Feedback**: View score and rationale but not saved.
- **View Real-World Projects (Demo)**: Get restricted datasets of CRM, HR, Finance, Inventory and Supplier.
- **Theme Switcher (Dark/Light Mode)**: Toggle between light and dark themes (preference not saved).
- **Use Chat Assistant (AI Tutor)**: Ask for query troubleshooting or explanations, and don't save the history.

3.3.2 Registered User Requirements

Registered Users have access to **all Guest User features**, plus advanced options for tracking, saving, and gamification:

- **Register**: Create an account with username, email, and password.
- **Login**: Securely log in with credentials to access profile.
- **View Website & Lessons**: Same as Guest, but can also:
    - **Mark Lesson Completed**: Record progress after finishing lessons.
    - **Bookmark Lessons**: Save lessons for quick access later.
- **Practice SQL**: Same as Guest, but queries can be saved for progress tracking.
    - **Submit Practice SQL** and **View SQL Feedback** are linked to the user profile.
- **Take Quiz**: Same as Guest, but results are stored in MySQL.
    - **View Quiz Feedback** history can be revisited.
- **Track XP & Progress**: Personal dashboard shows XP, completed lessons, badges, and achievements.
- **Open Challenges**: Access gamified SQL challenges.
    - **Select Challenge Set** (Beginner, Intermediate, Relational, Database).
    - **Select Level** (Easy, Medium, Hard).
    - **Gain XP** by solving challenges and completing activities.
- **View Real-World Projects**: Full access to CRM, HR, Finance, Inventory, and Supplier datasets with saved progress.
- **View Leaderboard**: Compare ranks with peers based on XP and badges.
- **Use Chat Assistant (AI Tutor)**: Receive debugging help, SQL explanations, and best practices with session history stored.
- **Theme Switcher (Dark/Light Mode)**: Same as Guest, but preferences are saved to profile.

3.3.3 General Requirements (All Users)

- The platform must be accessible on **standard web browsers**.
- The design must be **responsive** for both desktop and mobile.
- **Authentication** must be secure, and session based.
- **User progress, XP, challenges, and leaderboard data** must persist in the MySQL database.
- All system feedback (query errors, invalid login, unavailable challenges) must be clear and user-friendly.
- The **dark/light mode toggle** must be available for personalization.

3.4 Timeline

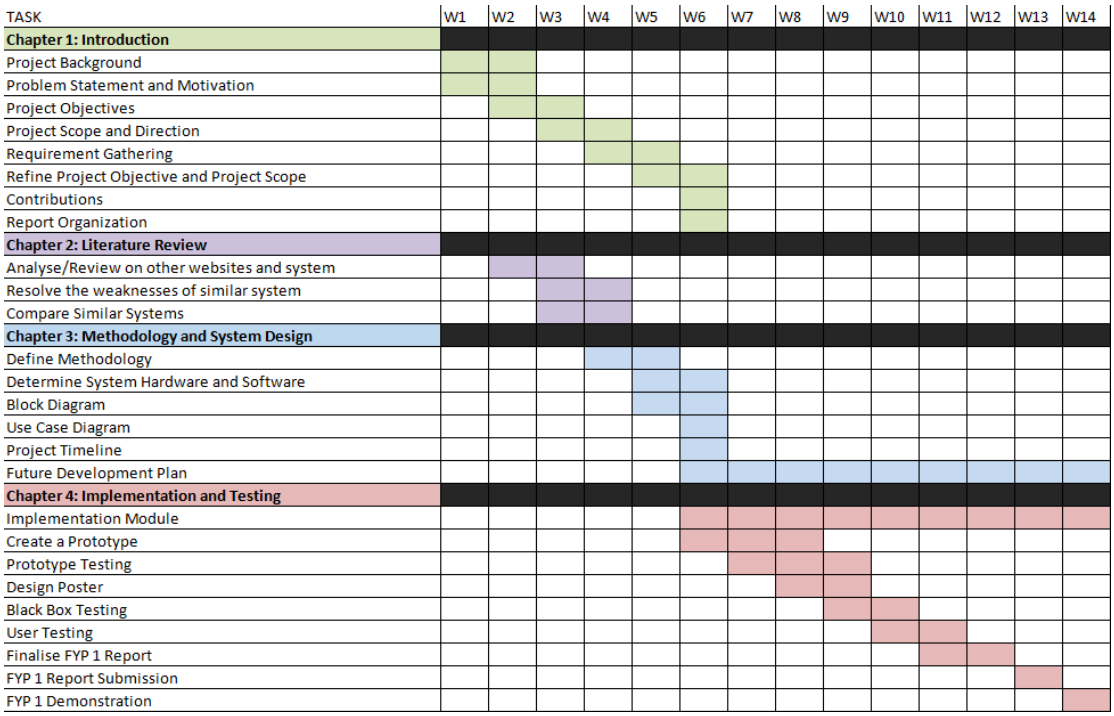


Figure 3.2 Timeline – FYP1

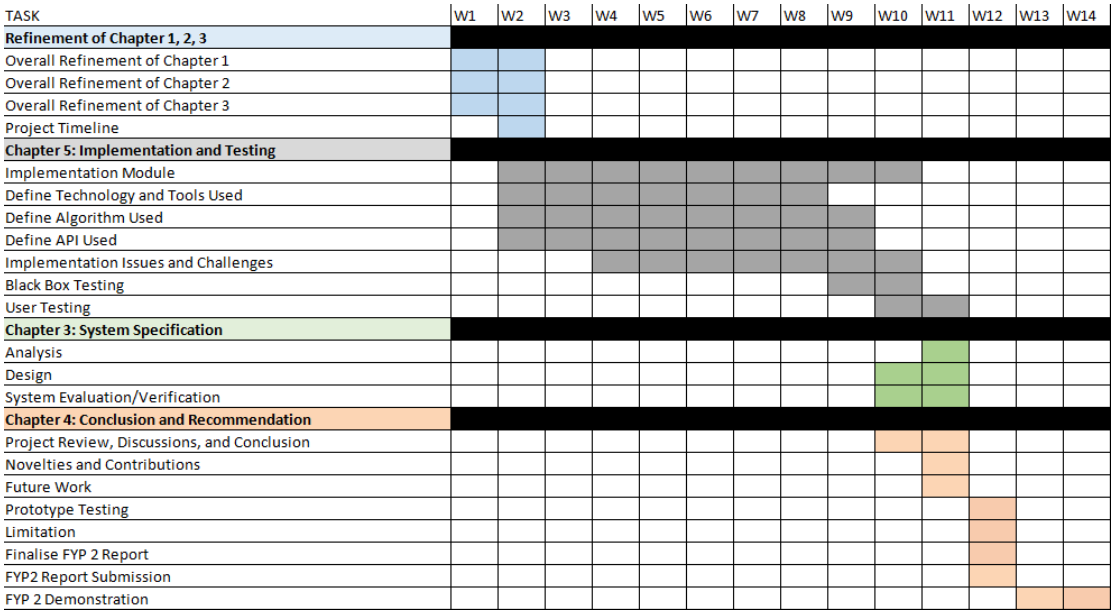


Figure 3.3 Timeline – FYP2

The project scope spans two semesters. In the first semester (FYP1), the focus is on gathering VA and system analysis, defining an appropriate methodology, and developing a design.

Important deliverables include a complete report of Chapter 1–3, the system block diagram, use case diagrams, and a simple Gantt chart timeline. During this period, prototype development for core modules (Registration/Login and Lessons) begins.

In the second semester (FYP2), primary focus moves to system development with design evaluation, testing, and system verification. Front end completion milestones (Home, Lessons, Challenges, Leaderboard, Profile), backend logic construction (authentication, lesson content serving, XP tracking, Smart SQL assistant), and backend MySQL database integration outline construction milestones. In these stages, black box testing, user feedback collection, and pre-system verification testing are done. Writing the report, designing the poster, and FYP2 final presentation are scheduled for the last month. A full Gantt chart showing all week-by-week tasks for both semesters is attached below.

CHAPTER 4 System Design /Overview

4.1 System Architecture Diagram

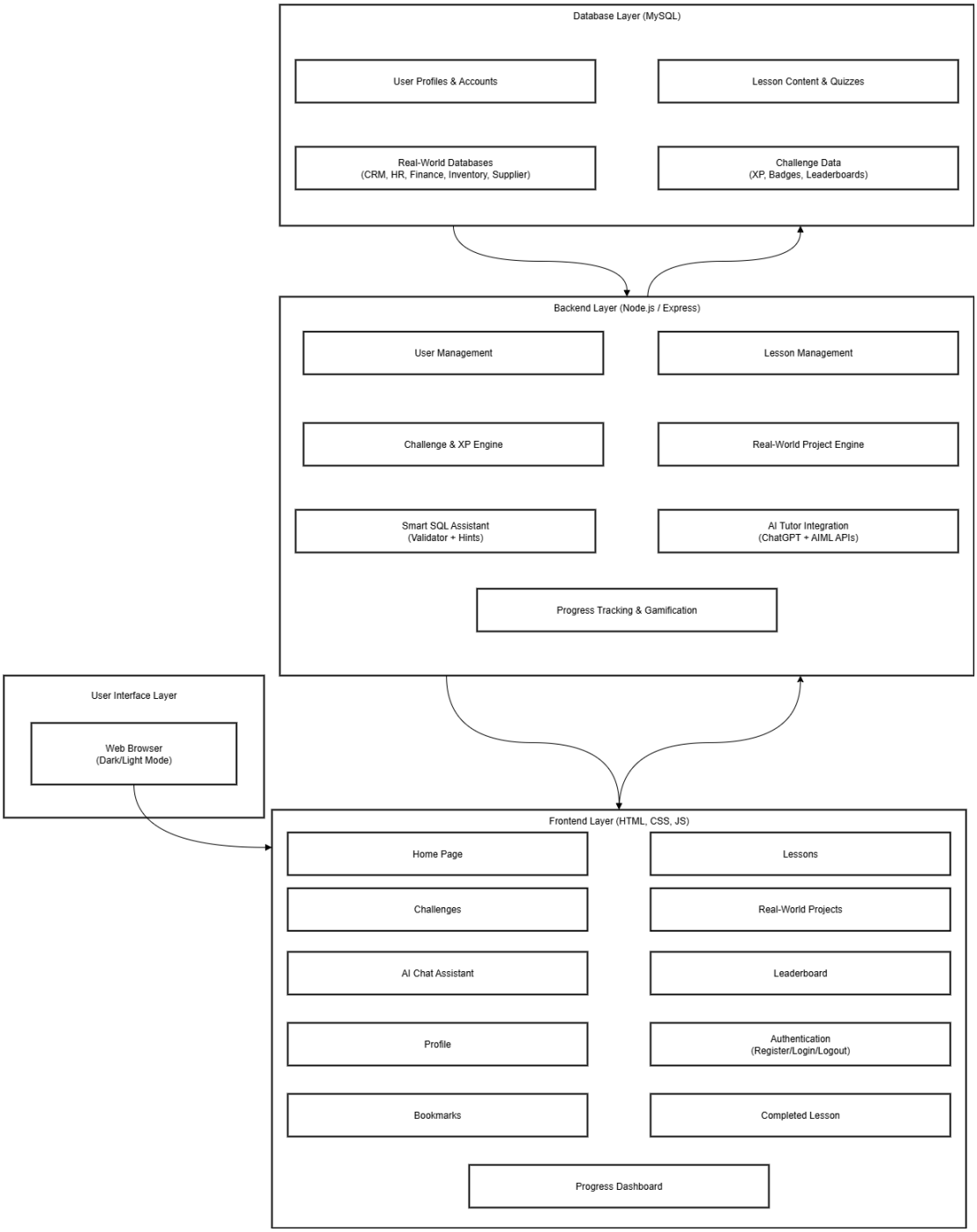


Figure 4.1 System Architecture Diagram of SQL Quest Platform

SQL Quest System Architecture Diagram The first SQL Quest system architecture diagram shows how their layered architecture is a solid conceptual foundation for achieving scalability, modularity and intuitive user interaction. Web Browser: this one supports dark and light

themes), at the "top" level of the User Interface Layer. It is the main learner interface for accessing lessons, challenges, projects and dashboards.

The Frontend Layer is constructed with HTML/CSS/JavaScript, and it is the layer responsible of showing features in an interactive form. The home page, lessons, challenges, real world projects core modules come: Ai chat assistant leaderboard profile authentication(register/login/logout) bookmarks completed lesson progress dashboard This layer provides a nimble navigation, instant feedback and gamified UI parts to make students user experience enjoyable.

Backend Layer It's built using Node. js and Express to process the business logic and interface with the frontend and database. It is responsible for handling crucial services like User Management, Lesson Management, Challenge & XP Engine, Real-World Project Engine, Smart SQL Assistant (Validator and Hints), AI Tutor Integration (ChatGPT AIML APIs) and Progress Tracking & Gamification. Together, these services handle user input to provide interactive hints on SQL queries; manage and validate the progression of gamified logic; and log users' progress and achievements.

The Database Layer The root layer is the Database Layer, which in MySQL stores and indexes all persistent tabular data. This includes User Profiles & Accounts, Lesson Content & Quizzes, Real-World Databases (e.g. CRM, HR, Finance, Inventory and Supplier systems), along with Challenge Data including XP, Badges and Leaderboards. The design guarantees all data interactions are reliable, ensuring availability and consistent access with quick retrieval that can be used to both monitor personal progress and competitive gamification.

In combination these layers provide a flexible design which connects the user in the browser to the Frontend Layer, then his request is taken care of by backend and database where all information needs to be persisted. This architecture has, in addition to the support of SQL Quest's gamified learning model, built by ensuring room for flexibility with additional real-life modules, advanced analytics as well as more AI-supported tutoring features in future releases.

4.2 Block Diagram

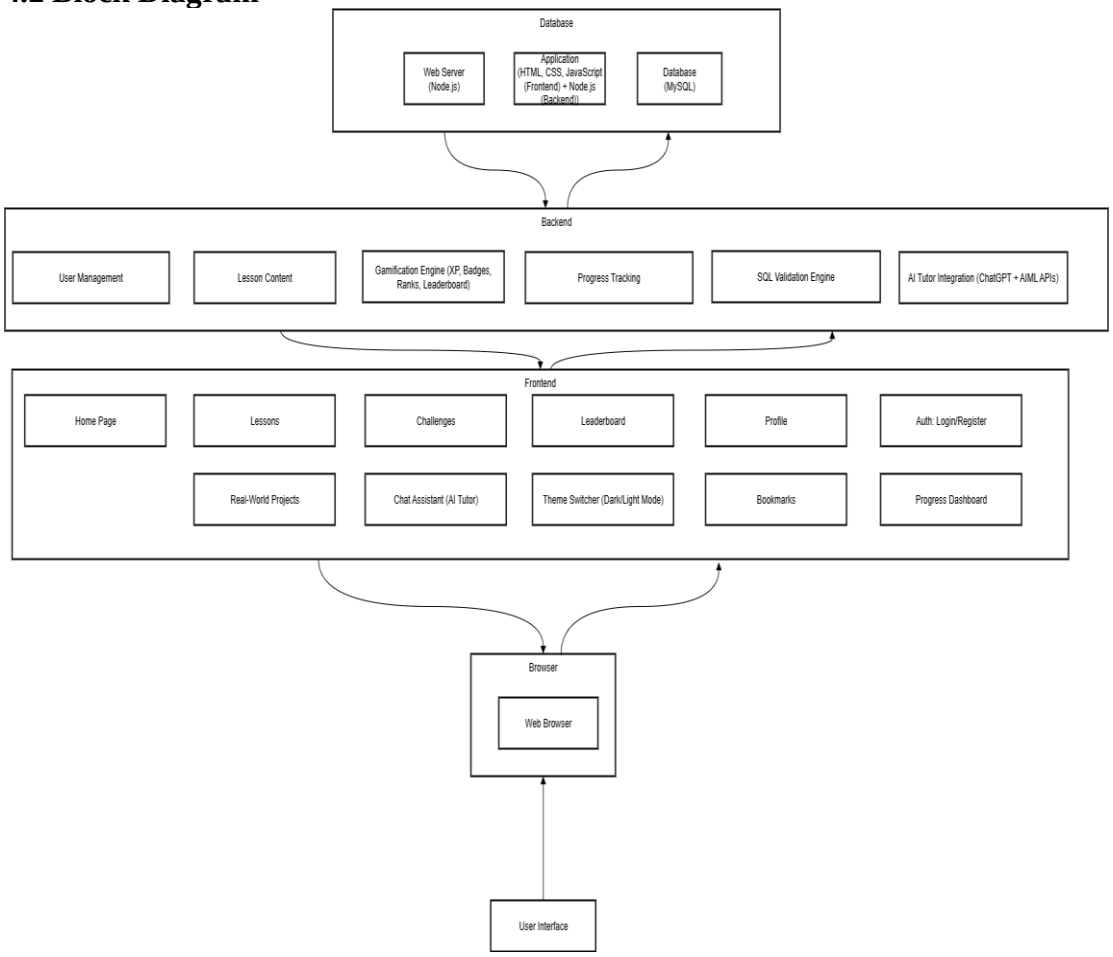


Figure 4.2 Block Diagram of SQL Quest platform

The platform structure of SQL Quest is designed in a layered way to support scalability, maintainability and interactivity. At bottom on the base, the User Interface serves as a primary interface point by which learners engage with the platform via a web browser. Frontend Layer conveys an interactive, retro-looking theme through its design and is built using HTML, CSS/React-like module components. It consists of the key things like Home Page to navigate around, Lessons for learning content in a structured way, Challenges for practice and Real-World Projects to mimic the professional scenarios with CRM, Finance, HR and Inventory databases. Front-end includes User Experience features like Profile handling, Authentication (Login/ Register), Bookmark for personalized learning, Progress Dashboard to track your performance, Leaderboard to compete with the community members, Theme Switcher (Dark/Light Mode) for accessibility and Chat Assistant backed by AI support you with any query.

The Backend Layer: The backend has been developed in the Node.js with Express.js, which serves as the computational engine for controlling data flow and orchestration. It manages user, what means account, login and role. Lesson Content Management manages lesson questions, quizzes and exercises. These levels on the Gamification Engine are used to calculate User's XP, rank, assign badges and leaderboard positions increasing engagement. Progress Tracking saves the progress of completed lessons, bookmarked modules and challenge results in real time as the SQL Validation Engine executes as well as evaluates submitted SQL queries against mock databases ensuring correctness and efficiency. The AI Tutor Integration the AI tutor APIs based on ChatGPT and AIML supports learning with contextual hints, corrective feedback and optimal suggestions that constitute the bridge between theory and practice.

The Database Layer is used as data persistence and realized in MySQL. It saves users profile, authentication details, progress history and bookmarks to inform XP scores, leaderboards and other historical data that guarantees an individual learning experience. It also includes lesson material, quizzes and dummy databases on various real-time domains (CRM, Finance, HR, Inventory, Supplier) as well. These benchmark databases are necessary for realistic problem-solving, as well as training and research contexts. The database is accessed by the back-end system via APIs, however the web server (Node.js) is responsible for the client-side application and handling requests and responses to the database.

This learning ecology is designed to be a strong rich ecosystem. The frontend provides a gamified, user-friendly experience; the backend ties logic, validation and AI tutoring together; and the database allows for persistent, structured training data straight from the real world. Such a multi-level design is employed to make SQL Quest an advanced, AI-driven and practice-based SQL learning tool that overcomes the problems with the current tools while getting learners ready for professional databases at work.

4.3 Sitemap Diagram

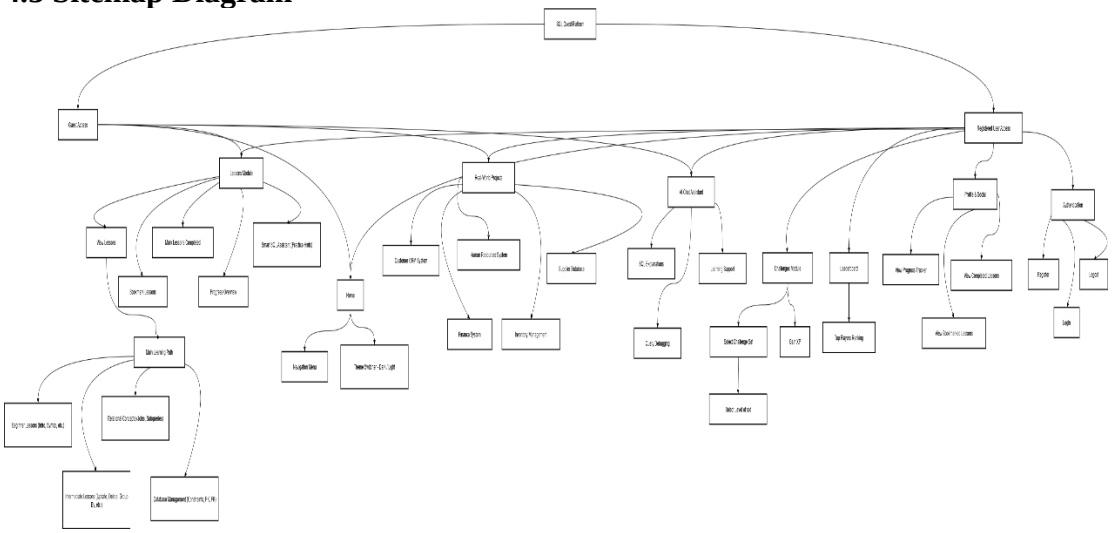


Figure 4.3 Sitemap Diagram of SQL Quest Platform

The sitemap of the SQL Quest reflects in whole the navigation structure of SQL-Quest, making a clear distinction between guest functionality and registered user actions. At the top level, there are modules for Guest Access, Registered User Access, Lesson Module Real-World Projects AI Chat Assistant Challenges Module Leaderboard Profile & Social Authentication, and Home/Navigation interface. The structure is designed so that users can also fluidly transition between basic learning, higher-level challenges and community-focused components.

Users experience initial engagement as they are walked through some of the content (lesson, leaderboards) for Guest Access. They can search the website for basic functionality, try demos in SQL and get to know the environment of the site. Since there is no stateful account, they can't do things like saving a lesson for later or keeping track of progress. This distinction provides an incentive for people to sign up for more complete access.

Access the full feature set immediately after you register for an account. This means users can have a profile/social aspect, bookmark lessons for later and the ability to go back and look at completed lessons. ON THE COUCH: Student learning progress is tracked on the Progress Dashboard which combines XP with badges, history, and achievements so they can see how far they've come. The user space is protected through authentication flows such as Registration, Login, and Logout which persisted via MySQL storage.

The heart of the pedagogical experience in SQL Quest is the Lesson Module. Lessons are split into Beginner (basic overview of SQL including syntax and SELECT query), Intermediate (update, delete and group functions), Relational Concepts (joins, subqueries, unions) and Database Management (constraints, primary/foreign keys, indexes & views). Lessons can be marked as complete, bookmarked and interacted with through ratings or comments. The lessons include an embedded Smart SQL Assistant with practice hints and instant feedback to empower self-paced exploration.

In addition to our structured classes, we offer real-world projects that simulate professional SQL environments through domain-specific datasets. Modules covering CRM, HR, Finance, Inventory and Supplier systems enable learners to sample real-world data in their exploration journey between classroom exercises and workplace problem solving. This rare combination of theory and practice will soon have your confidence in processing your data with SQL.

The AI Chat Assistant offers further interactive assistance. Learners can inquire on how to debug, explanations for difficult queries, and best-practices recommendations. The assistant operates as a steady companion, limiting how much frustration players can experience while exploring. Its contextual hints complement the validator, available to refine users' knowledge and avoid making the same mistake again.

To solidify their skills, the Challenges Module introduces gamification to learning. Users choose challenge sets by difficulty (easy, medium, hard) and advance through XP accrual. Motivation is maintained with points, badges, and ranks—awarded by the gamification engine. The XP you earn in this activity will progress your rank on the Leaderboard - showing you how much XP others have accrued and helping to create some friendly competition among Warrior's Path members.

The Home Page and Navigation Menu are the glue that holds everything up at UI level. The application comes with a theme switcher (dark/light mode) for both accessibility and personalization, so that you can learn in a refreshing interface. All functionality is easily reachable through intuitive menu entries for students of all skill levels.

4.4 Use Case Diagram

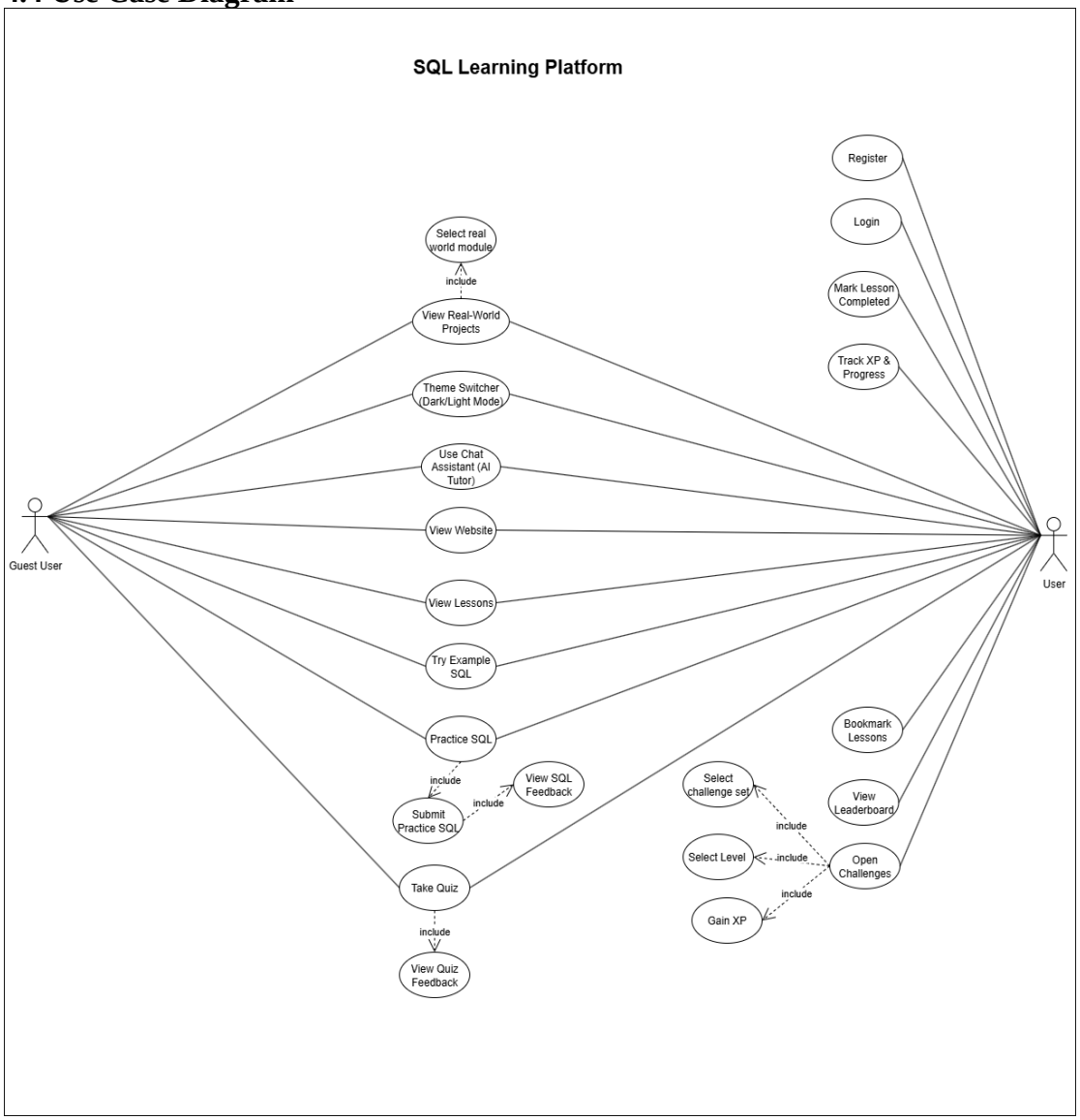


Figure 4.4 Use Case Diagram of SQL Quest Platform

The Use Case Diagram of SQL Quest, in Figure (2), shows all the potential interactions with learners regarding system and, by using stereotypes method, it separates guest access from registered user access. You, as a Guest User, can browse the website, examine lessons, execute example SQL queries, practice in SQL editor and take quizzes. These functions act as an interface, allowing new users to experience the functionality before deciding to sign up. But all guest interactions are ephemeral — progress, scores, bookmarks are not saved, and that's an incentive to sign up.

User A registered and logged in User is granted the full ability to use SQL Quest. They can take structured lessons like syntax, commands and relational concepts. Lessons can be completed, and progress is logged in an individual profile. Users can virtually "bookmark"

lessons for revisitation, and comment on/rate lessons in support of peer feedback and the quality assurance process. Other than guests, users logged in will be able to see the Leaderboard that shows XP, ranks as well as badges and achievements encouraging friendly competition.

Practice SQL includes a query validator that not only verifies the correctness of your queries, but also provides hints, explanations and alternative optimizations for your queries. This not only enables students to spot mistakes but to learn best practice. Quizzes are also auto graded with rich feedback that turns mistakes into learning moments.

Practice is gamified by letting users choose challenge sets (such as Joins, Aggregates, and Subqueries) and difficulty levels (Easy, Medium or Hard) in the Challenges Module. Students earn XP, level up, and unlock new avatars when they complete challenges. Their progress is recorded in their Progress Dashboard which integrates history, milestones and performance into one consolidated view.

The Real-World Projects Module is one of the best features, which provides learners an opportunity to reconcile real hands-on experience on simulated industry databases (CRM, HR, Finance, Inventory and Supplier Systems). This is the gap between academic learning and a production SQL workplace, students will be prepared for real use cases with SQL.

And the AI Chat Assistant gives you real-time guidance. Users can request help with debugging, ask for explanations of queries and best practices. This AI-based guide complements the learning modules, so students never feel “stuck.”

For easy reading and customization, SQL Quest offers theme switching (dark mode or light mode), to prevent visual fatigue during long learning process. The MySQL data store, operated through Node, stores all user interaction (progress tracking, challenges, quiz attempts, bookmarks and comments).js/Express APIs.

To conclude it can be stated that the new use case diagram brings forward more clearly than before the dual use of the system, as either guest access (explored only) or registered user access (full progress, leaderboards, achievements and personalization). SQL Quest isn't simply a learning tool; it's an enclosed gaming universe dedicated to helping you transform

from beginning syntax memorizer into confident, capable SQL ready to take on professional data analysis projects of the highest caliber.

4.4.1 Login Account – Use Case Description

Use Case Name	**Login**
Purpose	To allow a registered user to access their profile and learning features.
Actor	User
Normal Flow of Events	1. User selects "Login". 2. Enters email and password. 3. System checks MySQL for credentials. 4. If valid, session begins, and user is redirected to dashboard.
Alternate Flows	If invalid, error messages are displayed.

4.4.2 Register – Use Case Description

Use Case Name	**Register**
Purpose	To allow a guest to create a new SQL Quest account.
Actor	Guest User
Normal Flow of Events	1. Guest selects "Register". 2. Inputs username, email, and password. 3. System validates fields. 4. Account is created and stored in MySQL. 5. Guest is redirected to login page.
Alternate Flows	If email already exists or validation fails, system shows an error message.

4.4.3 View Website- Use Case Description

Use Case Name	**View Website**
Purpose	To allow any visitor to browse the SQL Quest homepage and access the navigation menus.
Actor	Guest User, User
Normal Flow of Events	1. Actor opens the SQL Quest website. 2. Homepage with navigation, theme switcher, and menus loads.
Alternate Flows	None.

4.4.4 View Lessons

Use Case Name	**View Lessons**
Purpose	To let actors access SQL lessons categorized into Beginner, Intermediate, Relational, and Database Management.
Actor	Guest User, User
Normal Flow of Events	1. Actor selects “Lessons”. 2. System displays list of lesson modules. 3. Actor clicks on a lesson to open explanations, syntax, and examples.
Alternate Flows	Guests cannot mark lessons as completed or bookmark them.

4.4.5 Bookmark Lessons

Use Case Name	**Bookmark Lessons**
Purpose	To allow registered users to save lessons for later reference.
Actor	User
Normal Flow of Events	1. User clicks “Bookmark” on a lesson. 2. System saves bookmark entry in MySQL under user profile. 3. Lesson is shown in bookmarked lessons tab.
Alternate Flows	The bookmark will hide if no login

4.4.6 Mark Lesson Completed

Use Case Name	**Mark Lesson Completed**
Purpose	To allow users to mark lessons as completed and track their learning progress.
Actor	User
Normal Flow of Events	1. User selects a lesson. 2. After finishing, clicks "Mark Completed". 3. System records completion in user's progress log.
Alternate Flows	If user is not logged in, the system will no mark it .

4.4.7 Track XP & Progress

Use Case Name	**Track XP & Progress**
Purpose	To track and display user's earned XP, badges, achievements, and completed lessons.
Actor	User
Normal Flow of Events	1. User performs actions (quizzes, challenges, projects). 2. XP and badges are updated in MySQL. 3. User dashboard displays updated progress statistics.
Alternate Flows	If database error occurs, system shows "Unable to fetch progress. Try again later."

4.4.8 Practice SQL

Use Case Name	**Practice SQL**
Purpose	To let learners write and run SQL queries in a live interactive editor.
Actor	Guest User, User
Normal Flow of Events	1. Actor opens practice editor. 2. Inputs SQL query. 3. System executes queries on mock database. 4. Results are displayed immediately.
Alternate Flows	If query is invalid, error message with syntax hints is shown.

4.4.9 Submit Practice SQL

Use Case Name	Submit Practice SQL
Purpose	To allow learners to submit SQL practice queries for validation.
Actor	Guest User, User
Normal Flow of Events	1. Actor executes SQL query. 2. Query is validated against expected output. 3. Feedback returned (correct, incorrect, or partially correct).
Alternate Flows	If system error occurs, query submission fails, and user is prompted to retry.

4.4.10 View SQL Feedback

Use Case Name	View SQL Feedback
Purpose	To provide immediate feedback on user-submitted SQL queries.
Actor	Guest User, User
Normal Flow of Events	1. System analyzes query. 2. Displays results, errors, and optimization tips. 3. User reads feedback to correct mistakes.
Alternate Flows	If system cannot evaluate query, it returns a generic error message.

4.4.11 Take Quiz

Use Case Name	Take Quiz
Purpose	To test learners' knowledge after completing lessons.
Actor	Guest User, User
Normal Flow of Events	1. Actor selects a quiz. 2. Answers multiple-choice or coding questions. 3. System auto-grades quiz.
Alternate Flows	Guest results are not saved (only preview). Registered users' results are stored in MySQL.

4.4.12 View Quiz Feedback

Use Case Name	**View Quiz Feedback**
Purpose	To show users their quiz performance with explanations.
Actor	Guest User, User
Normal Flow of Events	1. Quiz submitted. 2. System provides score and detailed solution explanations. 3. User reviews mistakes.
Alternate Flows	None.

4.4.13 Open Challenges

Use Case Name	**Open Challenges**
Purpose	To allow learners to attempt gamified SQL challenges.
Actor	User
Normal Flow of Events	1. User selects "Challenges". 2. System displays challenge categories. 3. User attempts selected challenge and earns XP if completed.
Alternate Flows	If challenge is unavailable, system displays "Challenge not available".

4.4.14 Select Challenge Set

Use Case Name	**Select Challenge Set**
Purpose	To allow learners to choose specific challenge sets (Beginner, Intermediate, Relational, Database).
Actor	User
Normal Flow of Events	1. User opens Challenge Module. 2. Chooses challenge set (1-12). 3. System loads appropriate problems.
Alternate Flows	If selected set is empty, error "No challenges found".

4.4.15 Select Level

Use Case Name	**Select Level**
Purpose	To allow learners to select the difficulty level of a challenge.
Actor	User
Normal Flow of Events	1. User opens a challenge set. 2. Chooses difficulty (Easy, Medium, Hard). 3. Challenge content adjusts based on chosen level.
Alternate Flows	None.

4.4.16 Gain XP

Use Case Name	**Gain XP**
Purpose	To reward users with XP points for solving quizzes, challenges, and completing projects.
Actor	User
Normal Flow of Events	1. User completes activity (quiz, challenge, project). 2. System updates XP in database. 3. XP reflected on progress dashboard.
Alternate Flows	None.

4.4.17 View Real-World Projects

Use Case Name	**View Real-World Projects**
Purpose	To allow users to access domain-specific real-world databases (CRM, HR, Finance, Inventory, Supplier).
Actor	Guest User, User
Normal Flow of Events	1. Actor selects “Real-World Projects”. 2. System loads project modules. 3. User runs SQL queries on domain datasets.
Alternate Flows	None.

4.4.18 Use Chat Assistant (AI Tutor)

Use Case Name	Use Chat Assistant (AI Tutor)
Purpose	To provide contextual query debugging, SQL explanations, and best practices.
Actor	Guest User, User
Normal Flow of Events	1.Actor opens Chat Assistant. 2.Inputs question or SQL query. 3. Assistant returns explanations, optimizations, or fixes.
Alternate Flows	If API is unreachable, fallback error message is displayed.

4.4.19 View Leaderboard

Use Case Name	View Leaderboard
Purpose	To allow users to view rankings based on XP, badges, and achievements.
Actor	User
Normal Flow of Events	1. Actor selects "Leaderboard". 2. System fetches user rankings from MySQL. 3. Rankings displayed with usernames and XP.
Alternate Flows	If database unavailable, leaderboard displays "No data available".

4.4.20 Theme Switcher (Dark/Light Mode)

Use Case Name	Theme Switcher (Dark/Light Mode)
Purpose	To allow users to toggle between light and dark UI themes.
Actor	Guest User, User
Normal Flow of Events	1. Actor selects "Toggle Theme". 2. System switches UI colors between dark/light mode. 3. Preference is saved (if logged in).
Alternate Flows	Guest changes are temporary and reset on reload.

4.5 Activity Diagram

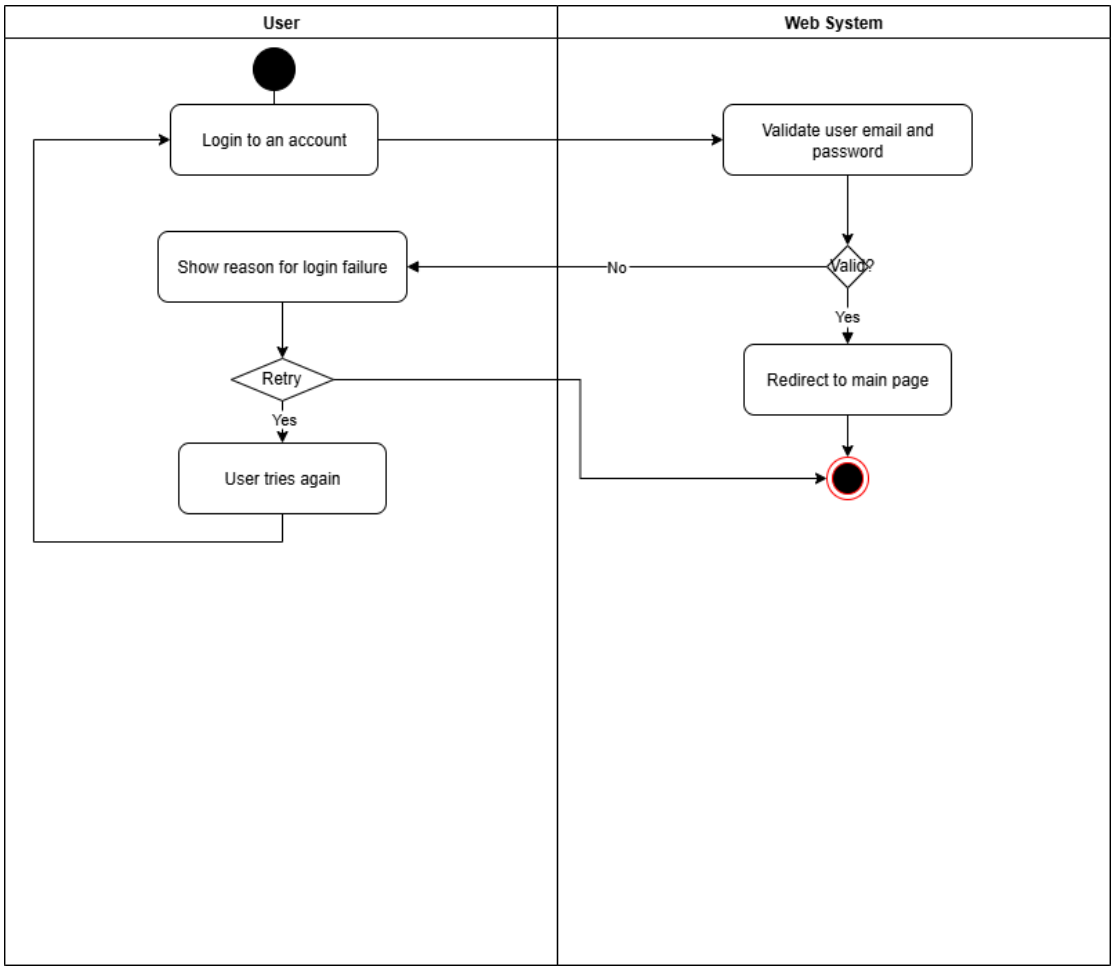


Figure 4.5 Login Activity Diagram

4.5.1 Login Activity

The login uses case diagram as shown in the figure represents how a user can interact with the web system during authentication. The flow starts with the user trying to sign-in, providing their e-mail and password in the app via a form. These credentials are then validated against the database when they submitted into the web system. If the input is valid, it passes validation and user logged redirect to main page which concludes our login process. If the login information is incorrect, the system gives you instant feedback by showing what was wrong with your login: a wrong password, or an unregistered email. The user is presented with the choice to retry at this stage. Opting to try again takes the user back to the sign-in and they can re-enter their information. If they decline a new attempt, you stop without issuing a successful login. This loop perfectly balances user experience and security by presenting meaningful validation errors, allowing for multiple failures, and only authorizing access after valid credentials

are authenticated.

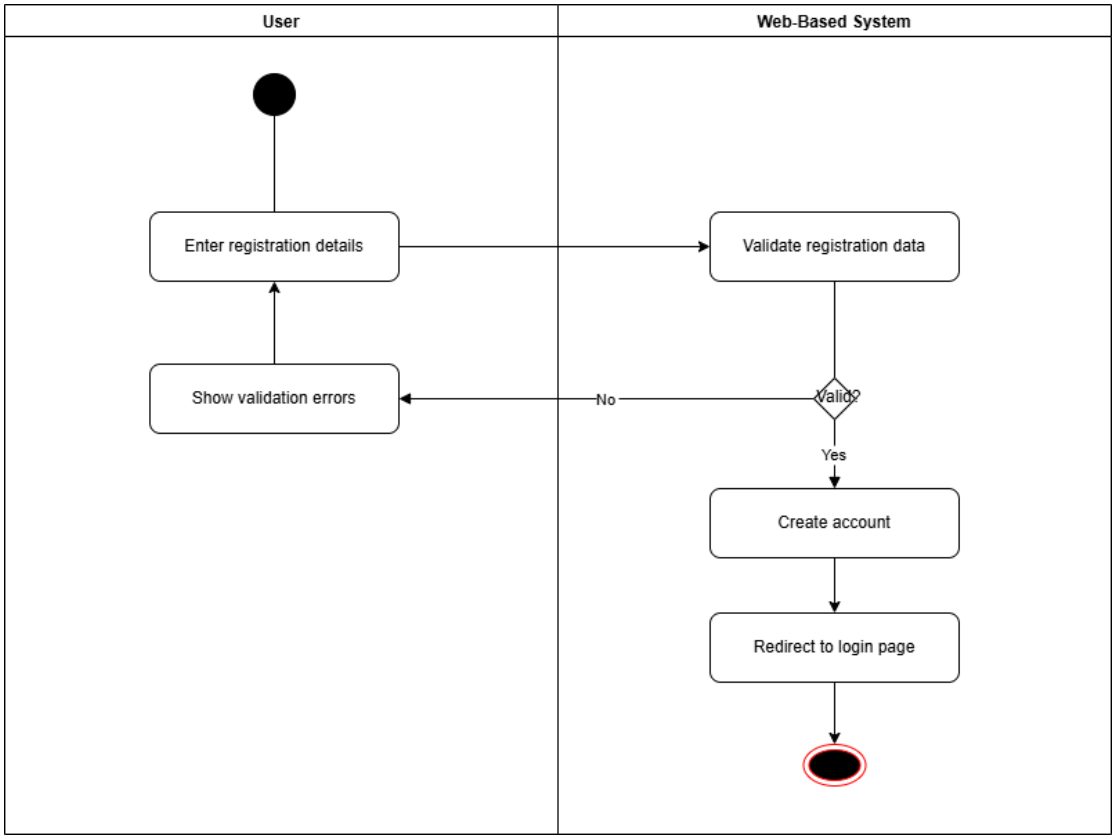


Figure 4.6 Registration Activity Diagram

4.5.2 Registration Activity

This diagram shows how a new user is onboard. After a user fills out the registration form, our online system checks the data for basic validity like format, email uniqueness, password strength, etc. The system returns errors if the validation fails, prompting the user to correct the input. Upon successful validation, the system creates a log entry in the users table and presents the login page.

This helps in maintaining a logical flow of operations on the system with a single-entry point for data and effective management of user accounts at registration. Validation restrictions ensure that system resources are gently tapped. Implementation was done with frontend validation on form filling in JavaScript and validation, and on the other side with Node.js. Such validation enhances the dependability of the system and reduces errors, erased records, or imperfections in the data storage center, aka the database, and maintains data integrity.

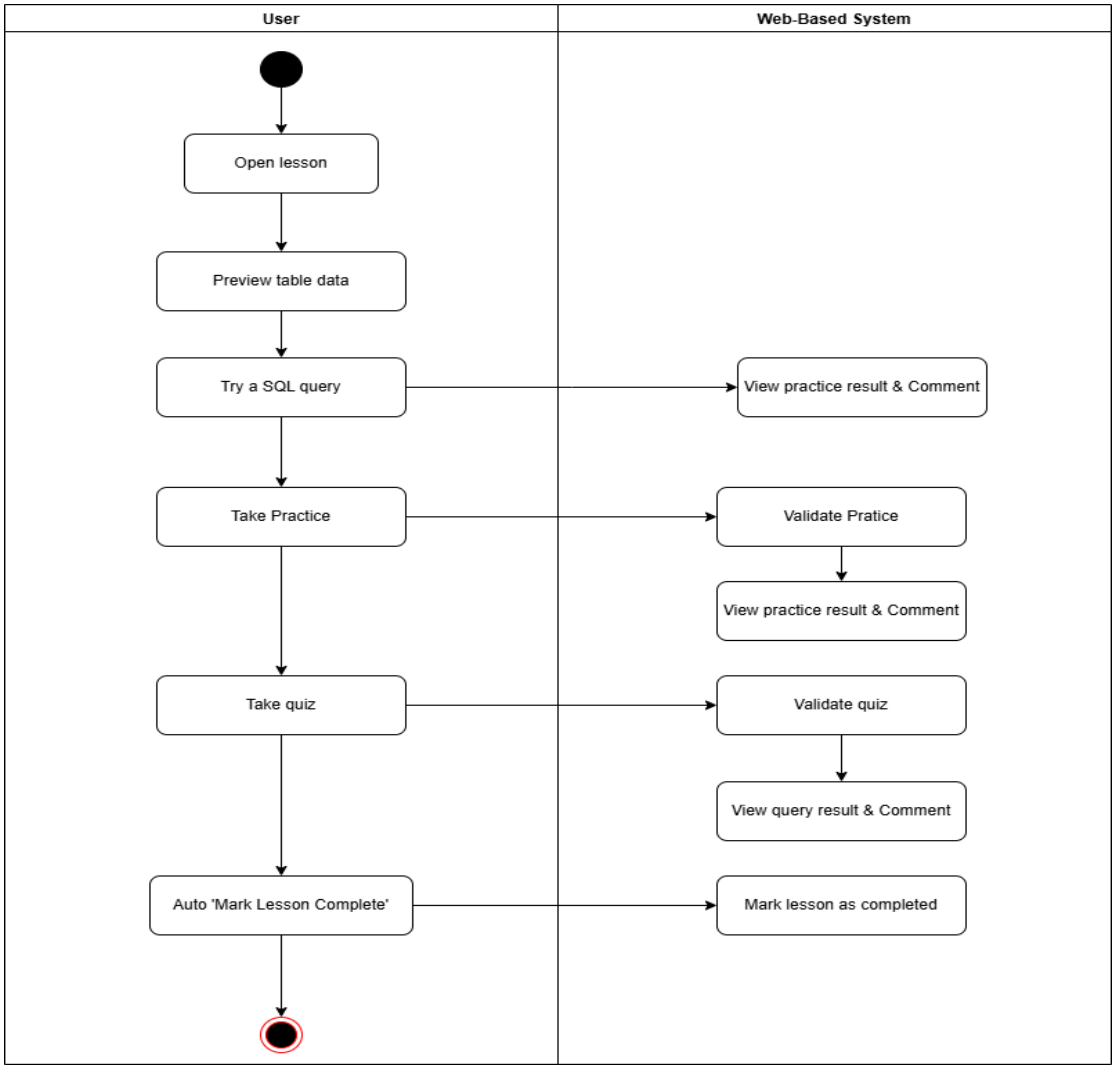


Figure 4.7 Lesson Interaction Activity Diagram

4.5.3 Lesson Interaction Activity

The lesson module activity diagram captures the logical order of learning a lesson that a user undergoes. It starts with a lesson opening that shows a related mock database table, alongside which users can optionally execute exploratory queries. Users then advance to guided practice sessions where they formulate SQL statements to solve some tiny problems. The system first checks these responses and approaches each of them with a line-by-line commentary for a remix. Upon finishing practice, users try a quiz for that topic. If they pass, the system indicates that they have completed the lesson and shuffles the record in the progress database. The pedagogical approach involving theory, practice, and evaluation simultaneously is depicted in this diagram. It shows the SQL Quest pedagogy framework that incorporates active learning. During the development, this flow was modularized to be reused across various lessons, improving codebase maintainability and scalability.

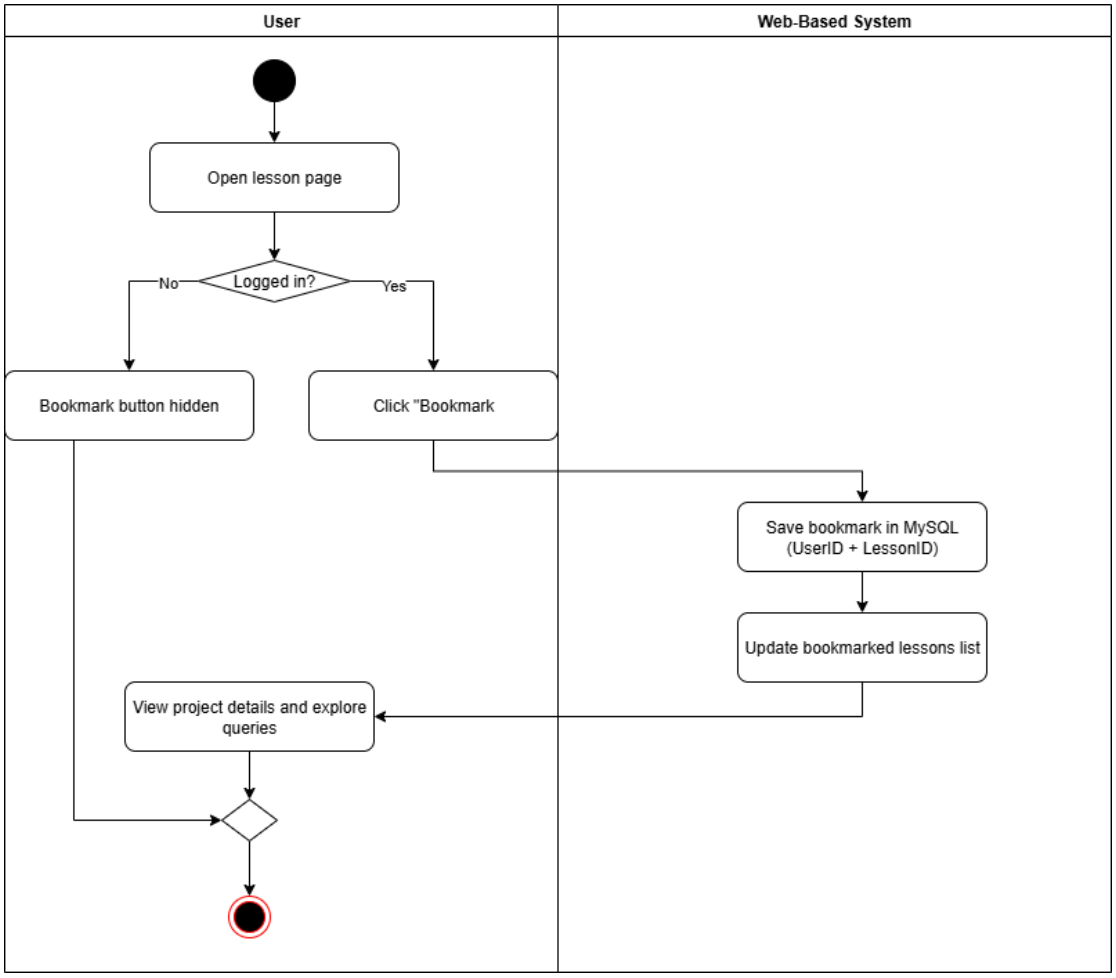


Figure 4.8 Bookmark Lesson Activity Diagram

4.5.4 Bookmark Lesson Activity

The Bookmark Lesson Activity Diagram demonstrates how users can keep lessons for future reference according to the logged in or out state. The system looks at the login status when a user enters a lesson page. If the answer is no, then we will hide the bookmark button because guests can browse and view every lesson (but they are not able to bookmark a lesson). When the user is logged in, the bookmark button appears and when clicked adds a new entry in the database that ties the UserID with the LessonID. The system then updates the user's bookmarked lessons list, making it easy for the user to quickly enter and view saved lessons while viewing lesson content.

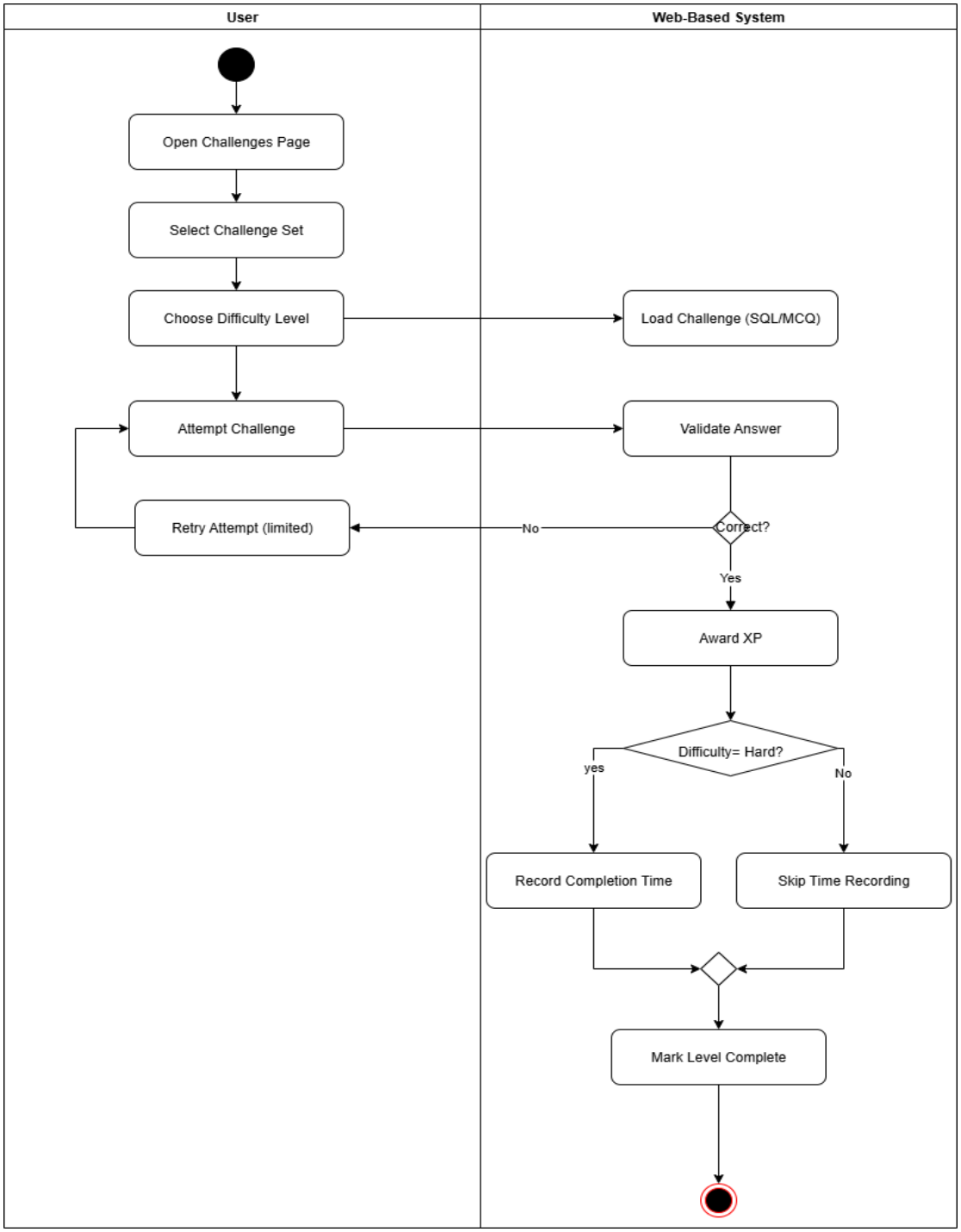


Figure 4.9 Challenge Attempt Activity Diagram

4.5.5 Challenge Attempt Activity

This activity diagram details the ultimate process of taking on challenges in SQL Quest, and a new feature focused on tracking completion times for Hard level challenges. The process starts when a participant opens the Challenges page and chooses a challenge set that they fancy. Then it asks the user to select a difficulty level and loads a desired challenge (SQL or

multiple-choice). The user tries the challenge and enters an answer, as the system checks it right away. If the response is wrong, the user receives a limited number of retries until they produce a correct answer or run out of retry attempts. If the answer is accurate, points are rewarded based on a gamified advancement system. It's at this point that the system introduces the decision: Is my difficulty Hard? If so, the system logs how long it took for them to complete the challenge and can store this data for use in comparing users against one another, e.g. sorting on leader boards or tie breaking of MMR across XP identical users. If it doesn't specify Hard difficulty, then the system just skips tracking time at all. For each condition the challenge is considered complete, and the flow stops. This flow ensures that every task type contribute to user progress and XP, but only Hard challenges introduce an added metric—completion time—that encourages the balance between accuracy and efficiency, creating a more competitive learning environment that will motivate users even in (almost) all too easy tasks.

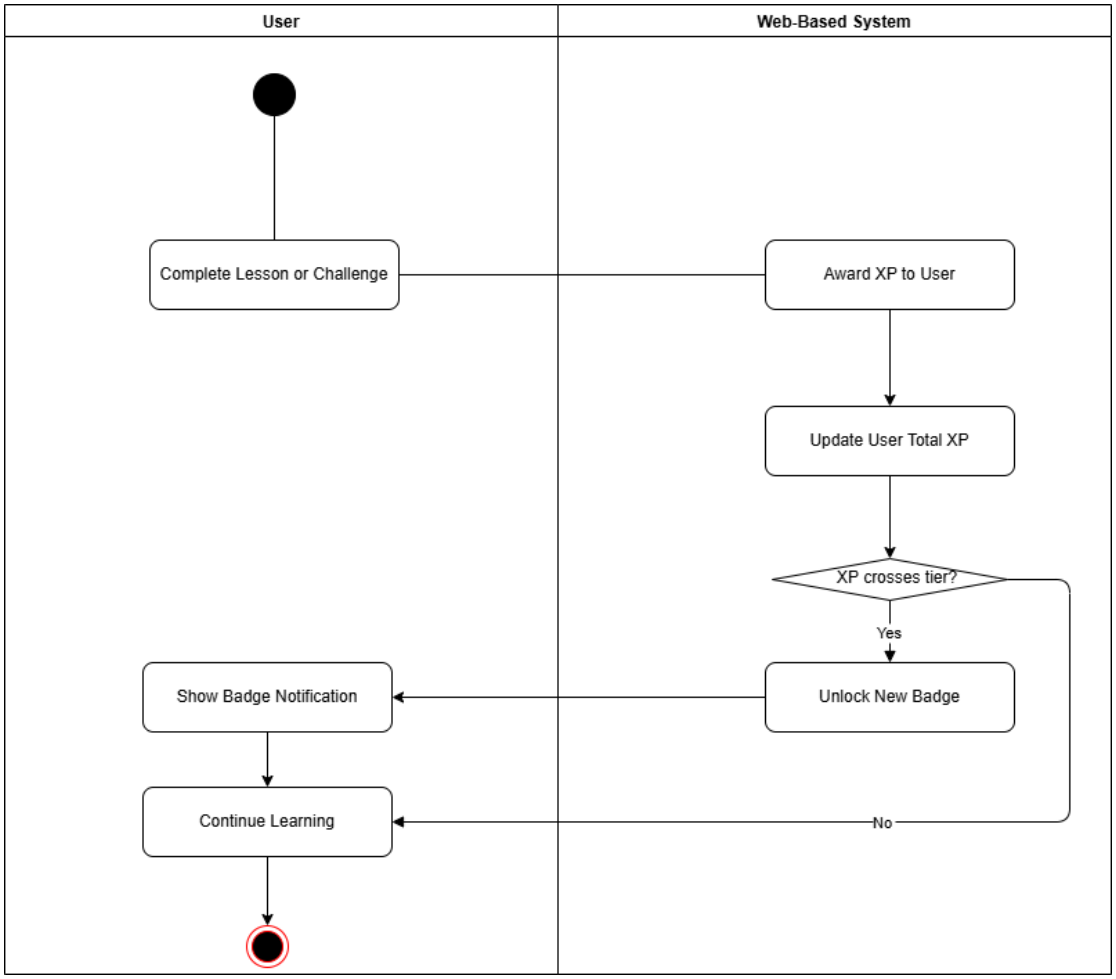


Figure 4.10 XP & Badge Earning Activity Diagram

4.5.6 XP & Badge Earning Activity

This diagram illustrates the behavior of the User System with respect to performance quantification. After a lesson or challenge is completed, the system is awarded XP and determines if the user has reached a new threshold. In case they have, a new badge is unlocked, and a notification is displayed. The thresholds for XP and badges are kept in a separate reference table in the database so they can easily be modified.

This diagram supports gamification, which increases learner engagement. The system is designed to use a reward loop for this part of the system as positive reinforcement for consistent learning. The logic in the backend turns off and updates XP instantaneously in real time which unlock achievement triggers without the need of user interaction.

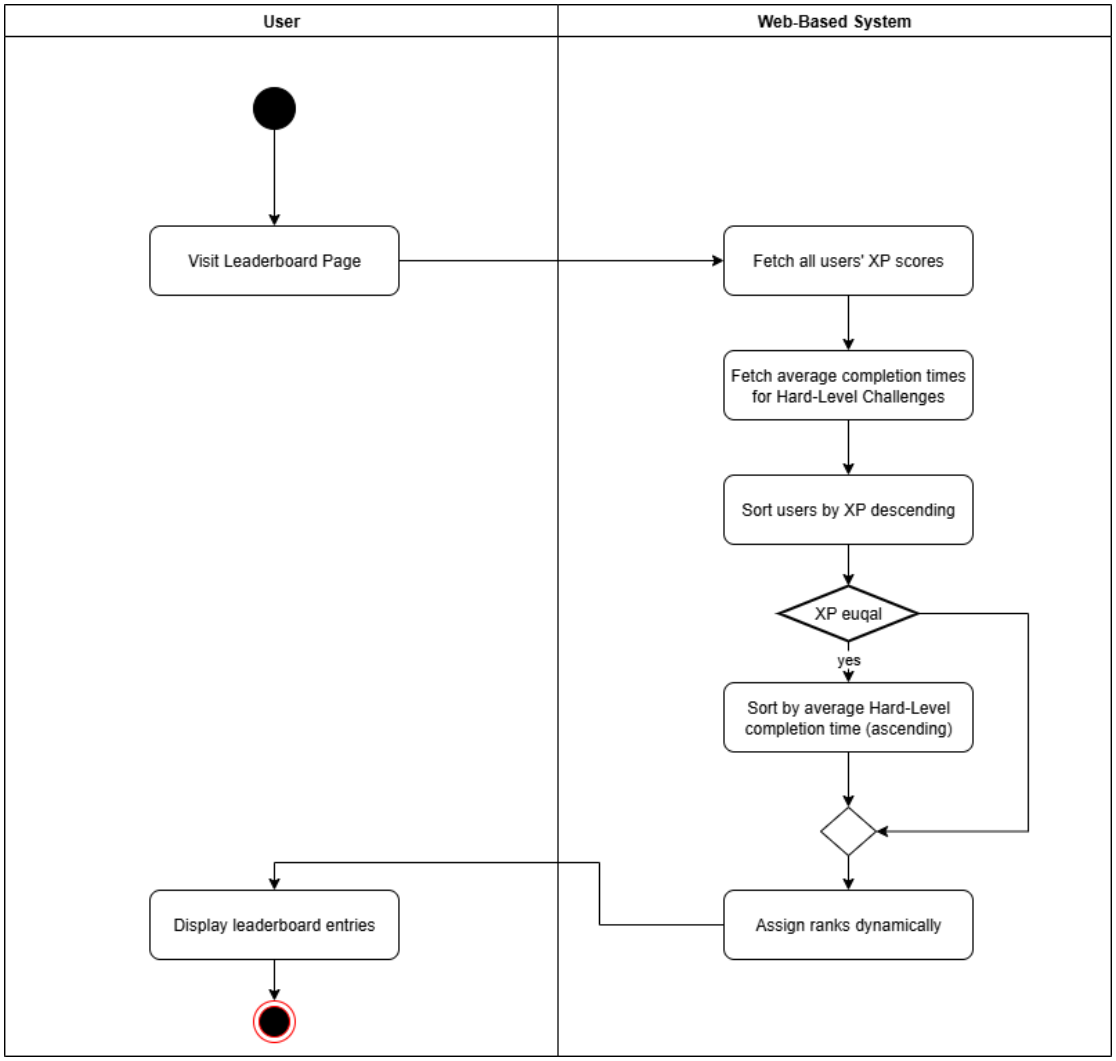


Figure 4.11 Profile & Progress Dashboard Activity Diagram

4.5.7 Profile & Progress Dashboard Activity Diagram

The system loads the user's study data after a successful login, including XP, lessons completed, badges earned, and bookmarks." The profile dashboard surfaces a progress bar showing visually milestone completion summary alongside metrics. This flow optimizes learning while helping abs cope with disengagement. Motivation is built through users seeing progress reported transparently enabling self-monitoring and boosting engagement. The information is fetched by executing optimized SQL queries to retrieve the tabulated data, then refresh the display using JavaScript DOM rendering.

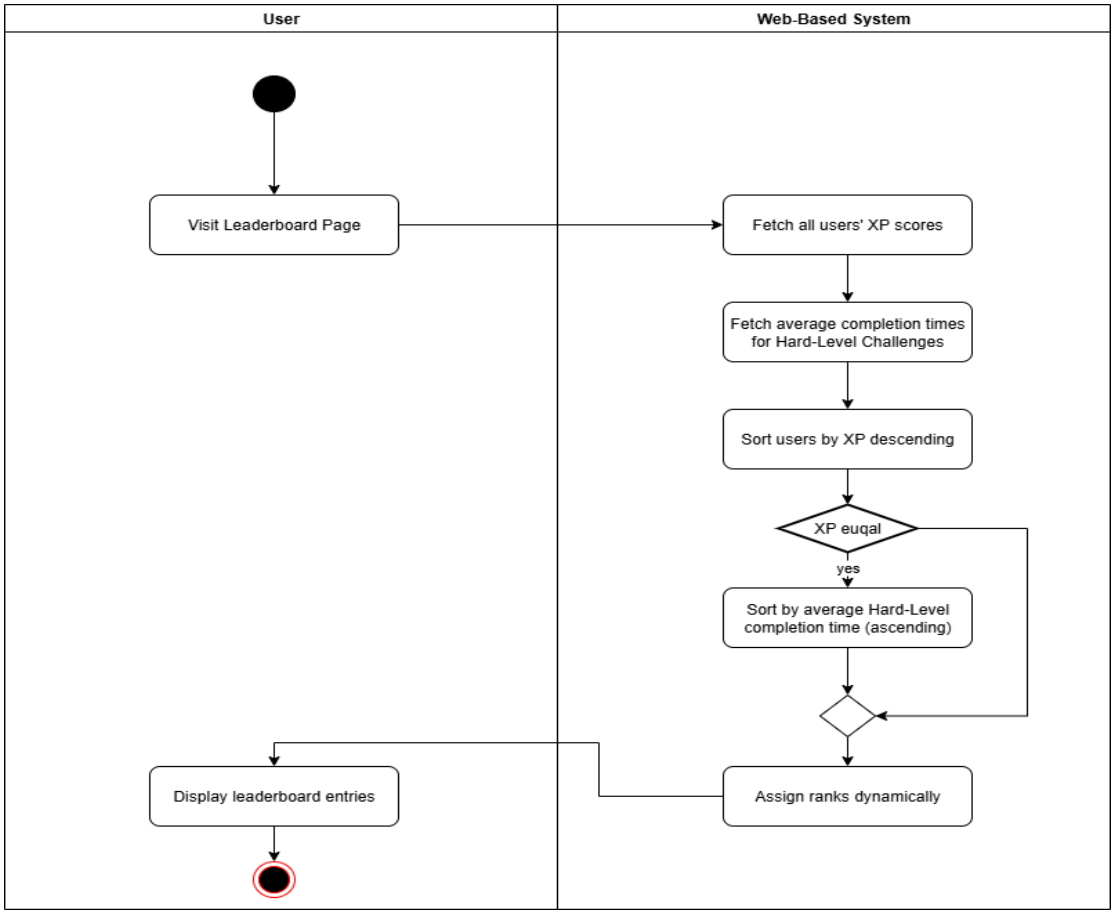


Figure 4.12 Leaderboard Activity Diagram

4.5.8 Leaderboard Activity

The activity diagram presented here -represents the order of operations for generating and displaying SQL Quest's leaderboard. It begins the moment the user loads onto the leaderboard page, this prompts the system to go out and collect everyone's XP scores from the database. To further its ranking authenticity and challenge the player base, it additionally gathers the Hard times. The ranking is performed as follows: first XP users and visitors are sorted by descending order, and they come at the top. If there is a tie—in the case of same XP—the rule for tiebreaker puts you in higher slot through their average completion time range of Hard-level challenges. Following this sorting and tie-breaking, the ranks are generated on the fly by the system and displayed as the final leaderboard to the user. This method emphasizes respect for the total learning effort and problem-solving efficiency, leading to a more balanced competitive environment.

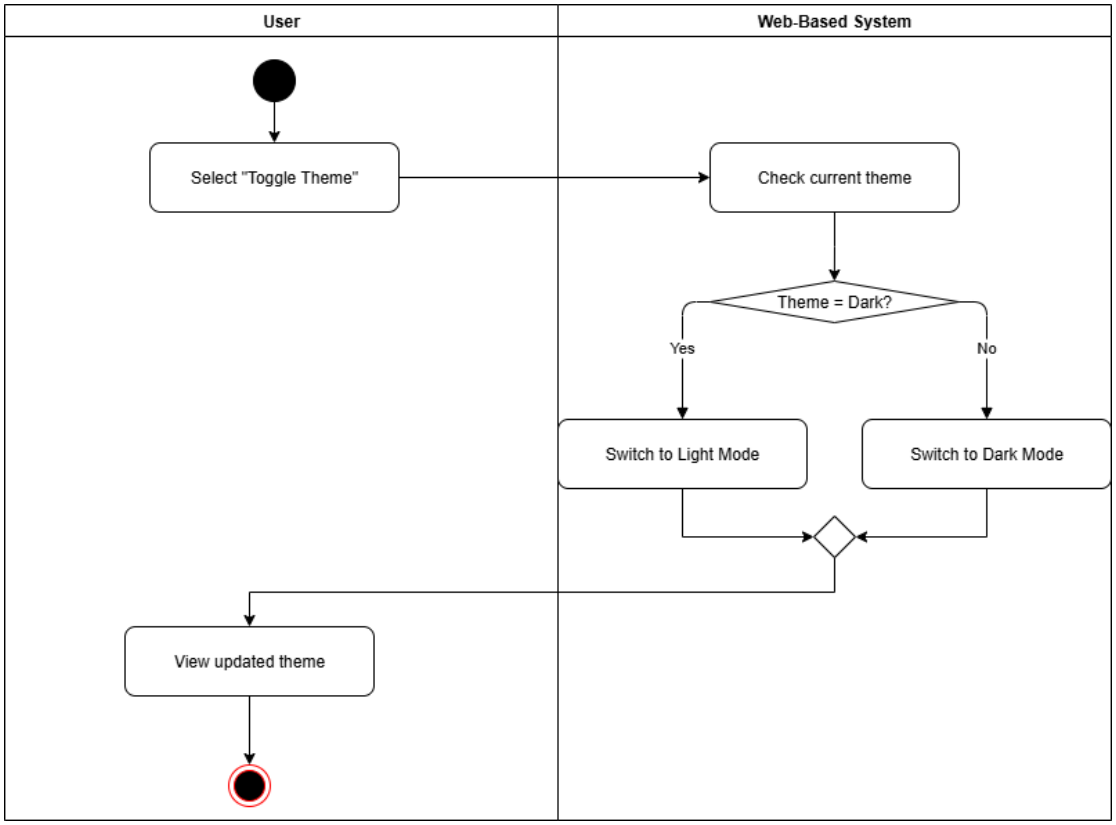


Figure 4.13 Theme Switcher Activity Diagram

4.5.9 Theme Switcher Activity

The Theme Switcher Activity Diagram explains how the users switch modes between dark and light on the SQL Quest site. The flow starts off with the user clicking on “Toggle Theme” at that point, the system checks which theme is currently set up. If the system sees that the current theme is dark, it changes the interface to light mode, otherwise it changes the interface to dark mode. After switching the mode, user will be able to see this new theme immediately without any delay. This functionality enhances personalization and accessibility, giving both guests and members the ability to engage with the platform in their preferred viewing format.

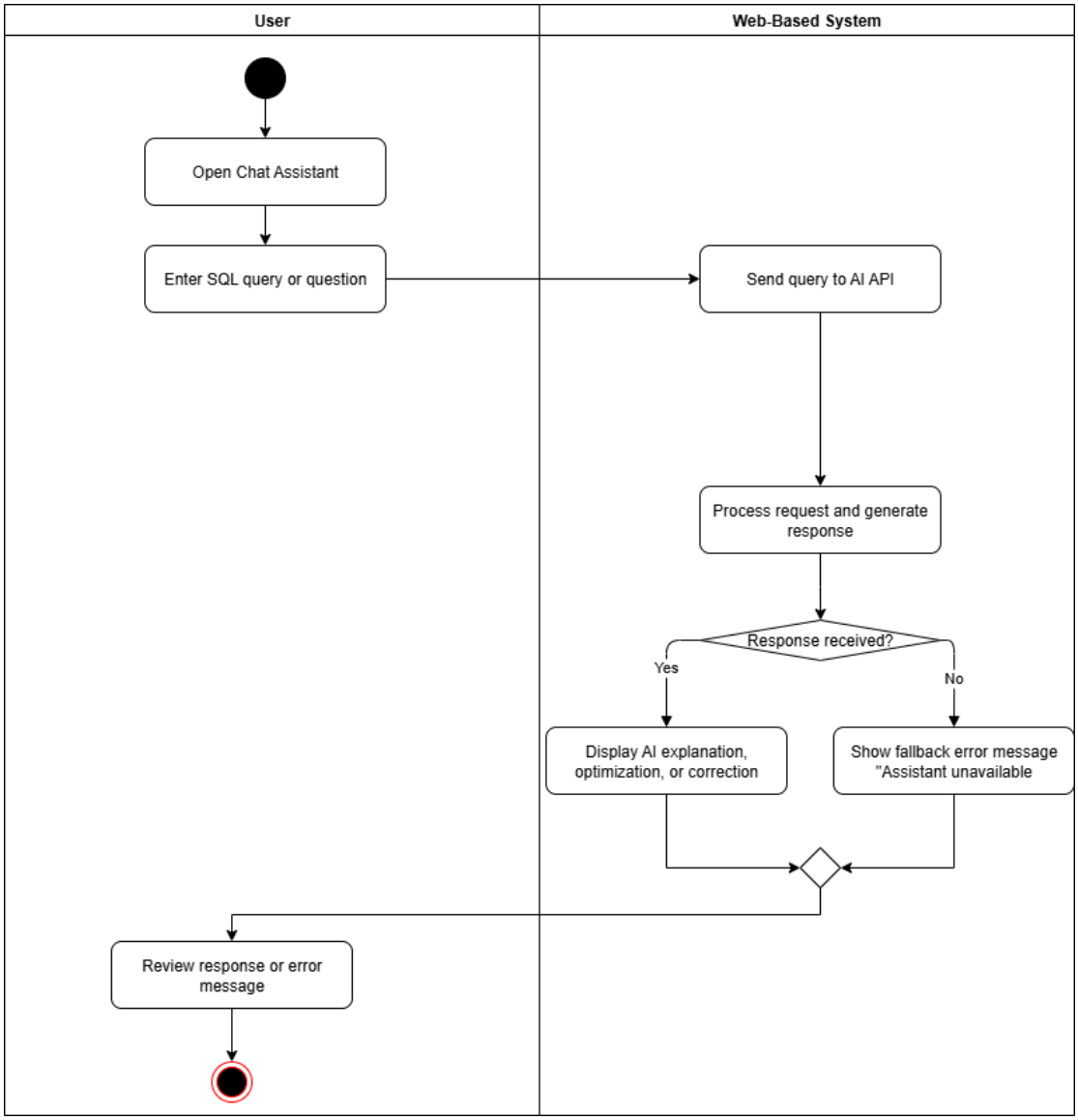


Figure 4.14 Use Chat Assistant (AI Tutor) Activity Diagram

4.5.10 Use Chat Assistant (AI Tutor) Activity

The Use Chat Assistant Activity Diagram illustrates the process flow as a user request assistant from the AI tutor in SQL Quest. To do so, the user opens Chat Assistant and types an SQL query or a text question. The system sends this request to AI API, and the latter handles the input and gives a response. If the system retrieves a response well, it provides some AI-based explanation/optimization/correction. But should the response from API be un-successful, then it will display a fallback error message displaying that the assistant is not available. Lastly, the user reviews an AI response or error message, and conversing turns to be completed. This approach guarantees users' feedback is never lost, keeping platform reliability even when API calls fail.

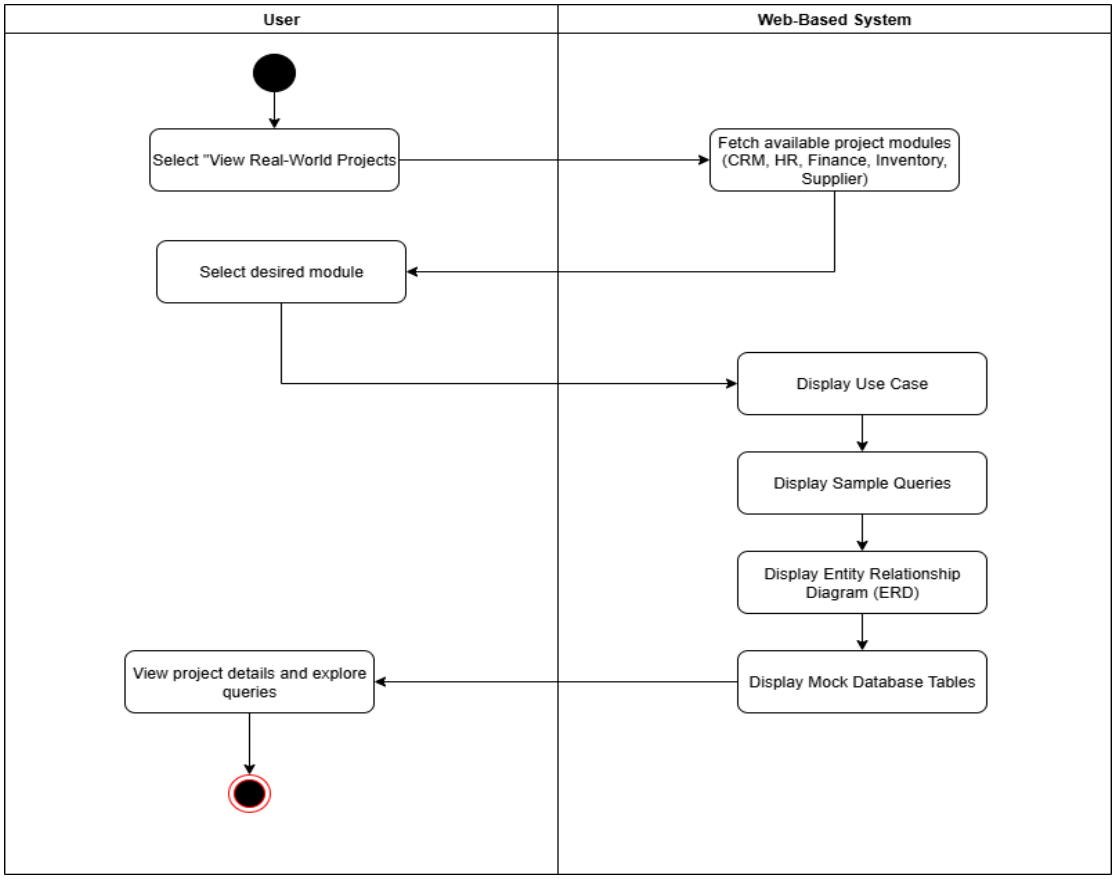


Figure 4.15 View Real-World Projects Activity Diagram

4.5.11 View Real-World Projects Activity

The real-world projects activity diagram of the View is shown in SQL Quest when a user or guest navigates through domain specific project modules. This flow will occur when the user clicks on “View Real-World Projects,” upon that click, the system should collect all available project modules like CRM, HR, Finance, Inventory and Supplier. The user can then select one of these modules. After selecting a module, the system will display in order: related use case, sample queries, ERD and mock database tables. The user can then check for the project specification and run queries interactively. This flow allows guest users and registered users alike to consume realistic data sets and scenarios in a hands-on fashion - it takes the academic out of theoretical learning and makes it real.

All in all, the sitemap shows that SQL Quest isn't merely a simple ‘tutorial’ site, but an entire ecosystem. It combines curriculum-based lessons with gamified practice, real-life projects, AI driven support and social feedback in a synchronized unit. By tracking these paths, the sitemap illuminates how SQL Quest makes learning SQL fun, successful and satisfying on the road from novice to expert.

CHAPTER 5 System Implementation

5.1 Hardware Setup

For best performance and functionality of the SQL Quest platform, some hardware requirements are recommended for the hosting server as well as end users' devices. Below is the bare minimum on hardware:

1. **Processor:**
    - Intel® Core™ i3, or AMD Ryzen™ 3 processor Such processors perform the computations needed by SQL Quest, that is query processing and communication with the database. Recommendation is an upper-tier CPU (i.e. Intel i5/i7, AMD Ryzen 5/7) for optimal multitasking and deployment of a larger number of the same or similar objects. Mesh/Capsule Colliders are used to perform optimized interaction. Note that this only is related to affecting the speed by which scripts are processed.
2. **RAM (Random Access Memory):**
    - 4 GB RAM Sufficient memory is required for smooth navigation and running of the SQL queries in the system. Users should have a minimum of 4 GB RAM to guarantee ease of use for lessons and interactive exercises. Server: To host servers, it is recommended to have higher than 8GB specification (as many learners can use at once).
3. **Hard Disk Space:**
    - 10GB of available hard disk space for installation and storage. Users need slack storage for caching, logs and data files. You will need a minimum of 250 GB storage for dealing with real-world projects (CRM, HR, Finance, Inventory, and Supplier) thanks to your server. It's recommended to have SSD-based storage for the performance advantage of executing queries faster with low latency.
4. **Display:**
    - Display: 1366 × 768 screen resolution with True Color Graphics Card: 16-bit color video display adapter. Lessons, dashboards and query results are best viewed at a display resolution of 1366 × 768 pixels or higher with a standard

graphics adapter. It's recommended to use a full HD resolution (1920 × 1080) for the best experience.

5. **Network:**
    - Internet connection High speed internet with a minimum download speed of 5MBPS. Because SQL Quest is a web application, an internet connection is needed to execute interactive queries and work with the real-life project data sets while also syncing of your progress. More bandwidth means latency should be lower for real-time responses and AI interaction.

This hardware enables learners and admins to both use SQL Quest without major performance problems. These are minimum requirements as users and hosting servers with better configurations will experience faster response times, quicker query execution and better performance.

5.2 Software Setup

For end-users as well as developers to be able to work with, understand and support the SQL Quest platform easily, it is necessary to have the following software components and configurations:

---

1. Web Browser

SQL Quest is 100% web based so all users need a modern web browser. The platform is tested and tailored for the most recent releases of:

- **Google Chrome**
- **Mozilla Firefox**
- **Microsoft Edge**
- **Apple Safari**

These browsers are responsive of course; they follow the light/dark mode system and support the interactive SQL editor. Make sure your users have an up-to-date browser for complete compatibility and best rendering of animations, quizzes, dashboards, and gamified elements.

---

2. Backend Runtime Environment

SQL Quest is powered by Node.js and Express.js, which are responsible for server-side logic, user authentication, calling APIs and interacting with the database.

- **Node.js**: v18.0 or above (LTS recommended)
- **Express.js**: Middleware framework used for routing, authentication, and handling client–server requests.
- **npm (Node Package Manager)** is required for dependency installation.

---

3. Database Management System

The site's backend is built upon MySQL as the main RDBMS:

- To Keep and store your user profiles, log-in details, bookmarks, progress, XP (experience points) and leaderboards records.
- Test project databases (CRM, HR, Finance, Inventory, Supplier) to practice SQL exercises.

Recommended tool:

- **MySQL Workbench** (latest version) for database management, ERD view and query testing.

---

4. AI Tutor and External Integrations

SQL Quest can connect to external APIs using the AI Chat Assistant (ChatGPT + AIML) for hints, explanations, and optimization feedback.

- Requires (read-only) key to OpenAI API and store it safely in a file. env file.
- **Node.js** modules such as axios and dotenv oversee constructing and managing API calls.

---

5. Front-End Development Tools

The front end is developed using:

- **HTML5, CSS3, JavaScript** for basic structure and style.
- **Bootstrap 5** is used for responsive and uniform design.
- **React.js** does (if we're doing a full implementation) to control lessons parts and reactively update the UI.

A modern IDE is recommended locally:

- **Visual Studio Code (VS Code)** and Node.js, ESLint, and MySQL plugins.

---

6. Additional Tools

- **Git** for a version controlled collaborative development.
- **Postman** (optional) to test the APIs and backend endpoints.
- **Draw.io or PlantUML** for UML, activity and architecture diagrams in documentation.

5.3 Setting and Configuration

To access and use the SQL Quest platform, users and developers must configure the software components correctly. This section provides a step-by-step guide on setting up and configuring the required tools for smooth execution of the system both locally (for development) and on the server (for deployment).

Step 1: Install Node.js and npm

1. **Download Node.js Installer**
   - Visit the official Node.js website at https://nodejs.org.

- Select the **LTS (Long-Term Support)** version for better stability.

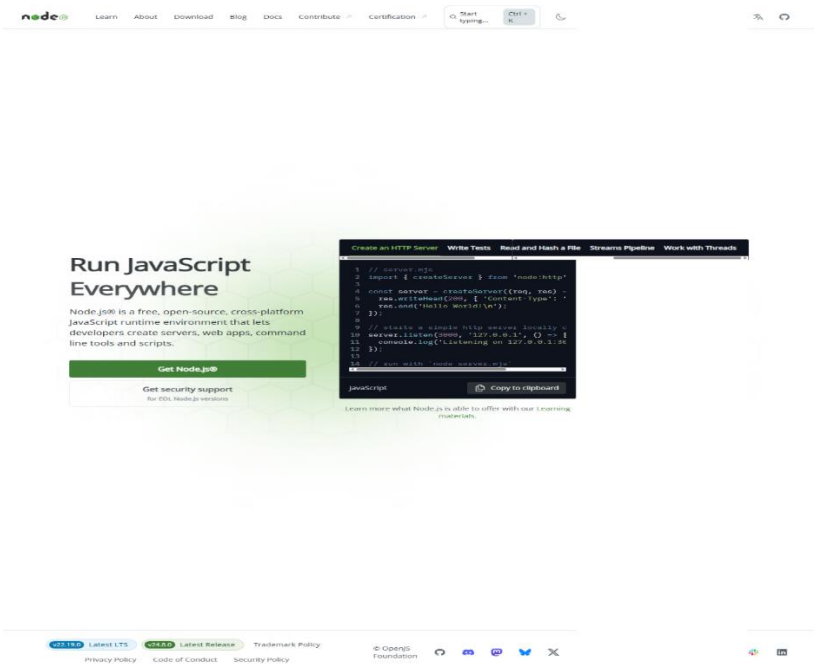


Figure 5.1: Node.js Official Website (Homepage)

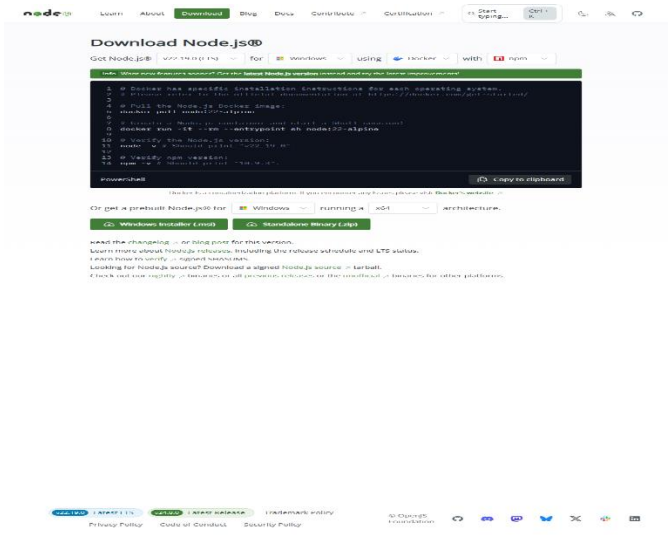


Figure 5.2: Node.js Download Page

2. **Run the Installer**

- Locate the downloaded `.msi` file in your **Downloads** folder.

- Double-click to launch the installer.

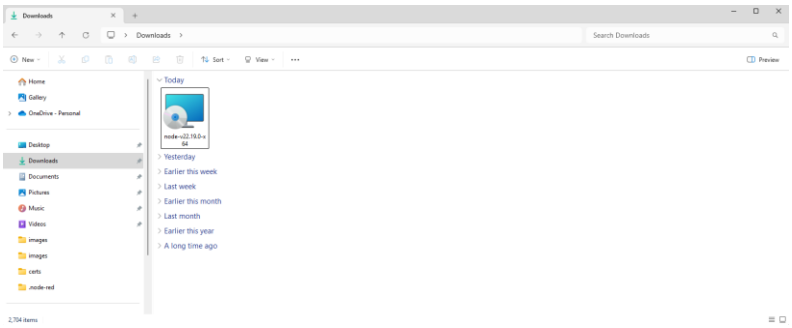


Figure 5.3: Node.js Installer Downloaded

3. **Custom Setup**
   - Ensure the following options are checked during setup:
     - **Node.js runtime**
     - **npm package manager**
     - **Add to PATH**

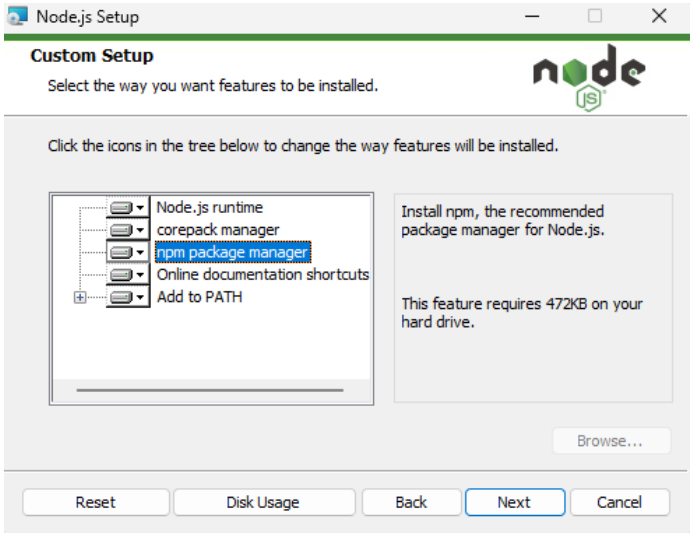


Figure 5.4: Node.js Custom Setup

4. **Install Tools for Native Modules**
   - Tick the box: *"Automatically install the necessary tools"*.
   - This ensures support for compiling C/C++ addons required by npm modules.

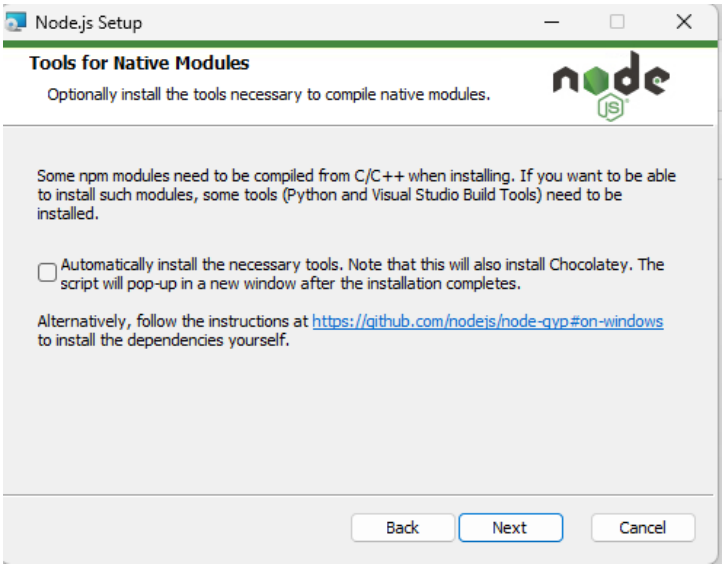


Figure 5.5: Tools for Native Modules

5. **Start Installation**
   - Click **Install** and wait until setup completes.

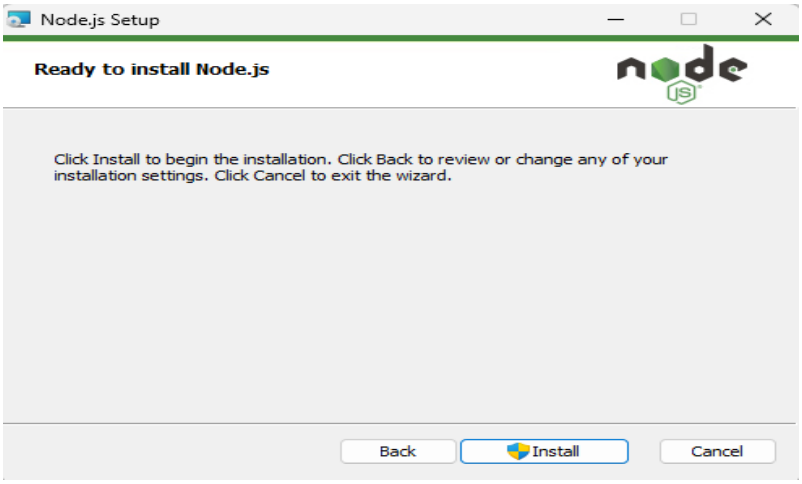


Figure 5.6: Ready to Install Node.js

6. **Verify Installation**
   - Open **Command Prompt** and run:
   - `node -v`
   - `npm -v`
   - The versions of Node.js and npm should appear.

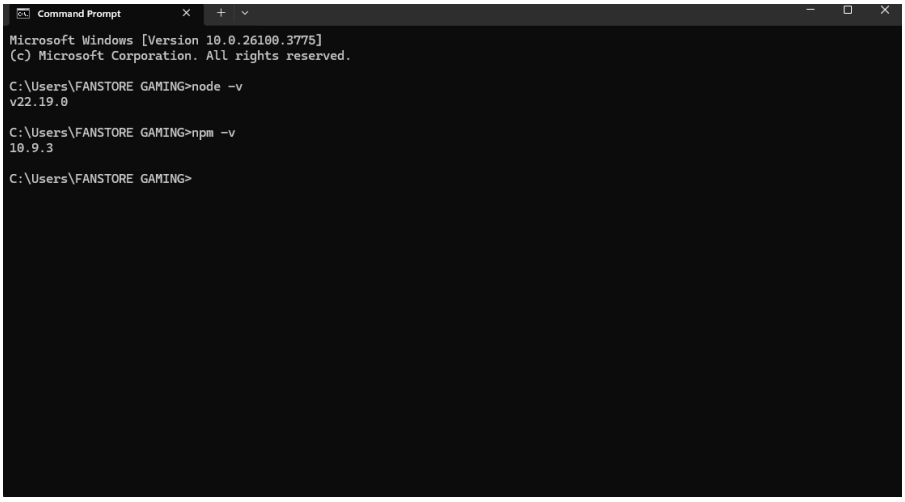


Figure 5.7: Verifying Node.js and npm Installation

Step 2: Install MySQL Database

1. Download and install **MySQL Community Server** from https://dev.mysql.com/downloads/workbench/

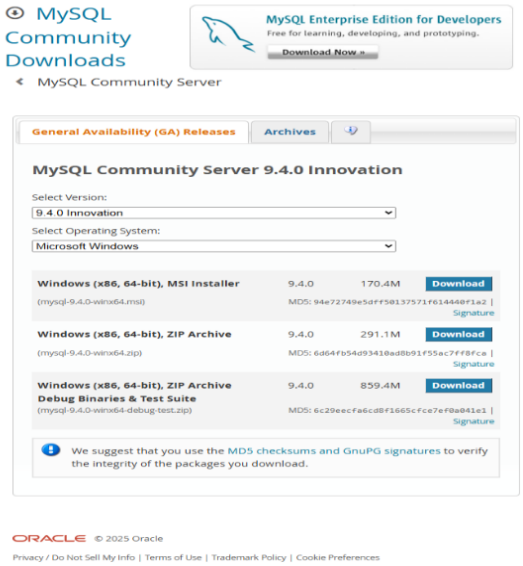


Figure 5.8: MySQL Community Server Download Page

2. Also install **MySQL Workbench** to manage and visualize databases from (https://dev.mysql.com/downloads/workbench/).

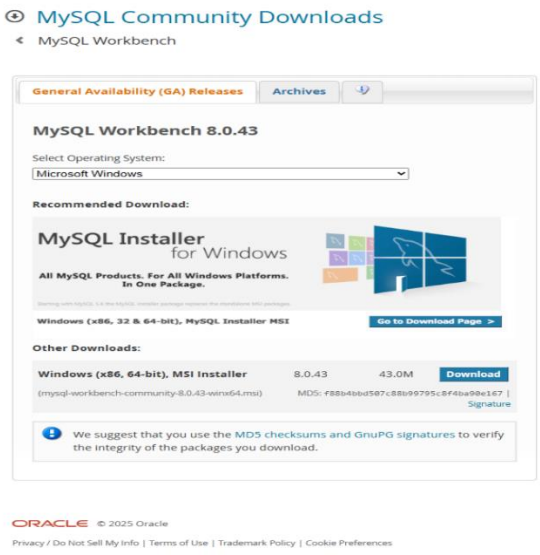


Figure 5.9: MySQL Workbench Download Page

3. During installation, configure the MySQL root user with a strong password (to be used later in `.env`).
4. Start the MySQL service.

Step 3: Configure SQL Quest Source Code

1. Download the provided **SQL Quest source code** (zip file or Git repository).
2. Extract the project to a working directory, for example:
3. `C:\Users\FANSTOREGAMING\source\repos\sql-learning-platform`
4. Open the project in **Visual Studio 2022**.
5. Install all dependencies by running:

```
6. npm install
```

Step 4: Database Setup

1. Open MySQL Workbench
    - Launch MySQL Workbench from the Start Menu or after installation.

- Connect to your MySQL Server instance using the root account (or the account you set up during installation).

2. Create a new database:
3. Type the SQL code "`CREATE DATABASE sql_learning;`" inside the MySQL workbench then click the lightning logo button, then the database will create

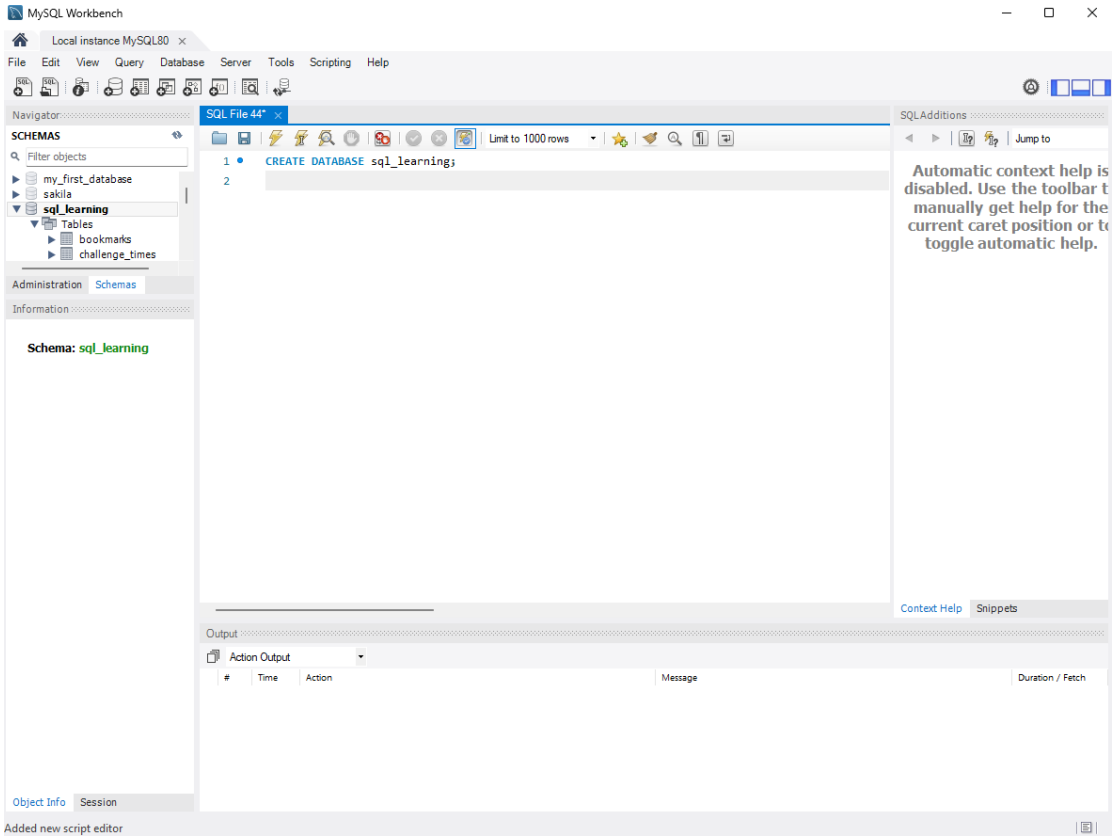


Figure 5.10: MySQL Workbench with Database Creation

4. Import the provided SQL schema (`sql_learning.sql`) located in the `/database` folder of the project.
    - This will set up tables for **users, lessons, bookmarks, progress, XP, challenges, leaderboard, and mock datasets (CRM, HR, Finance, Inventory, Supplier)**.

Step 5: Start the Server

1. In the terminal, navigate to the project directory.

2. Run the application using:

```
3. npm start
```

4. By default, the server will run at: http://localhost:5000

Step 6: Accessing SQL Quest

1. Open a modern web browser (Chrome/Firefox/Edge).
2. Enter the URL:

```
3. http://localhost:5000
```

4. The **SQL Quest homepage** will load, showing options for **Lessons, Challenges, Real-World Projects, Leaderboard, and Profile Dashboard**.
5. Guest users can explore lessons and demos, while registered users can **log in, bookmark lessons, track XP, attempt challenges, and view progress**.

5.4 System Operation

This chapter features the functionality of the SQL Quest web application using screenshots and several scenarios as to how a user operates within the system, from sign-on to browsing real-life projects, lessons, challenges and profile attributes. It has a guest and a user system, so guests can also challenge other guests to solve SQL problems in head-to-head Johnny Tables mode.

5.4.1 Home Page

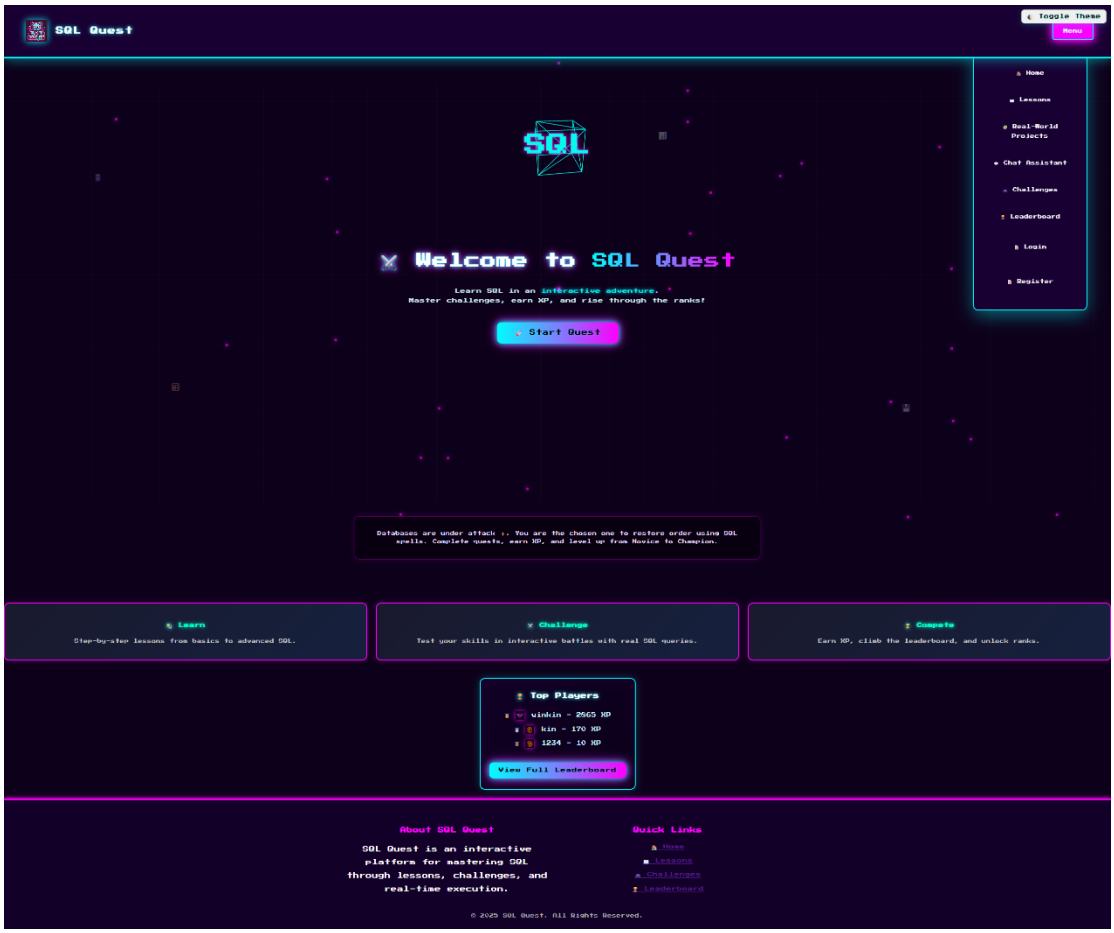


Figure5.11 Home Page with Sidebar Navigation

The Home Page of SQL Quest (see Figure 5.11) serves as the central hub for users, greeting them with a futuristic, gamified feeling design. Here, the left-hand menu is depicted on which you can quickly access important sections such as Home, Lessons, Real-World Projects, Chat Assistant, Challenges, Leaderboard, Login and Register. This feature permits users or guest

visitors to transition between the various modules of the platform. You'll also find the theme toggle button at the top for your viewing pleasure.

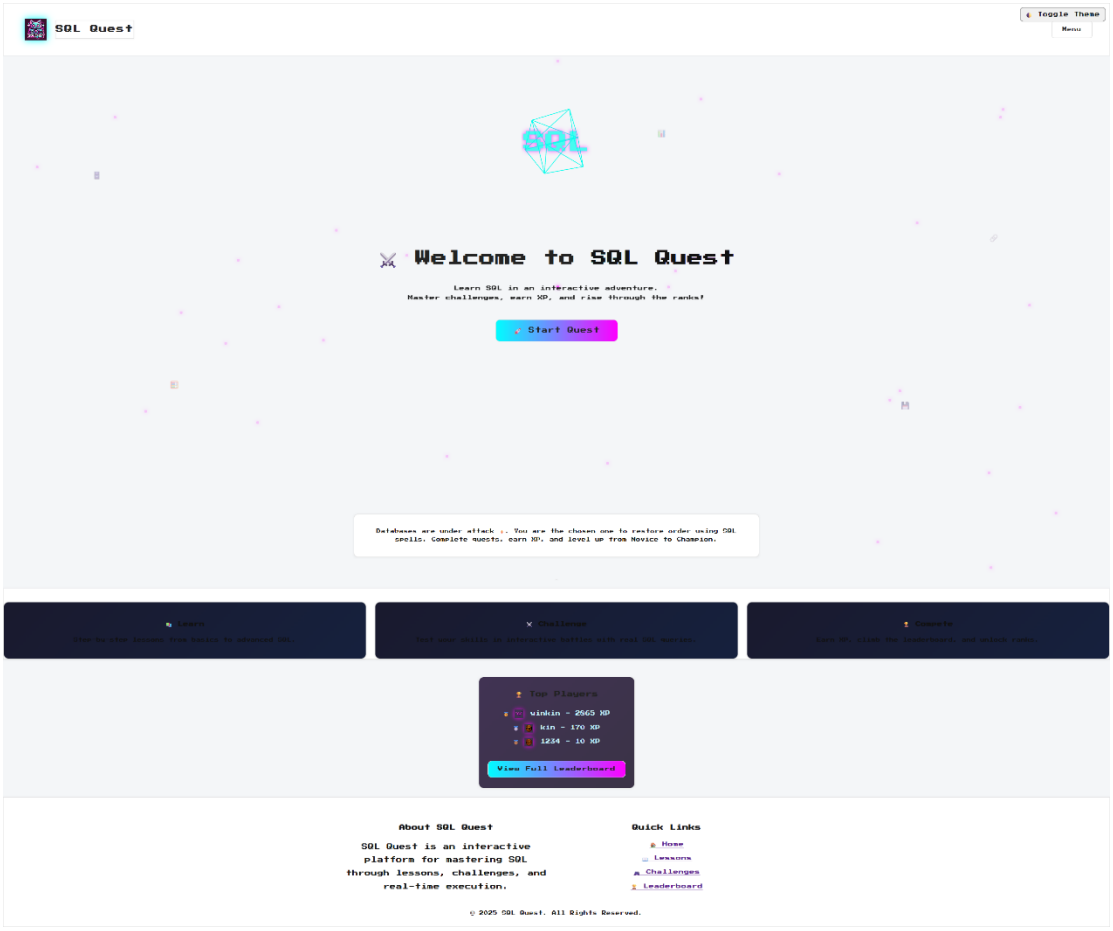


Figure5.12 Home Page with Theme Toggle (Light Mode)

The whole UI could be switched from dark mode to light mode by clicking this button, as indicated in Figure. 5.12, to meet the user's habits and barrier-free design. In the middle of the landing page, there is a welcome header phrase with a "Start Quest" button that petting users to start learning SQL.

Figure5.13 Home Page After Login with User Statistics

Upon a successful login, the interface shifts to include a personalized stats panel as presented in Figure 5.13 that displays the user's profile, experience points gained (XP), rank and progress towards next level. Underneath it, three clickable gauntlets—Learn, Challenge, and Compete—lead users to lessons, practice challenges, and the leaderboard. Moreover, competitive rankings are displayed in the Top Players area to incentivize learners. In sum, the Home Page uses accessibility and customized motivation to create a psychologically safe opening experience on SQL Quest.

5.4.2 Register Page

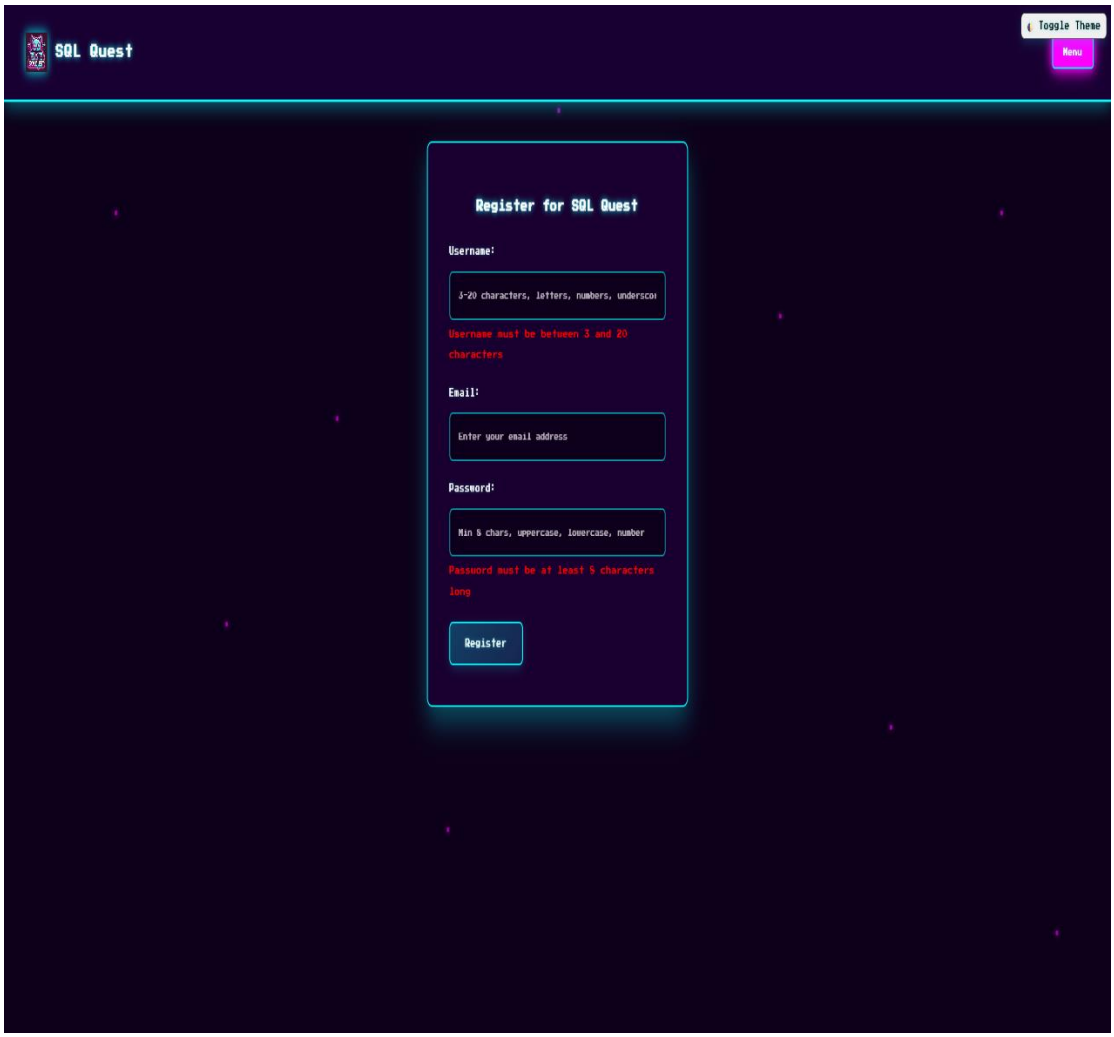


Figure 5.14 Registration Form with Validation Errors

SQL Quest Register Page is a simple and safe way for new users to sign up for an account prior to using the full features of the system. Figure 5.14 The registration form needs three mandatory fields: Username, Email and Password. The system applies a list of validation rules directly in form, e.g. username 3-20 characters and can have letters, numbers or underscores. Passwords need to be a minimum of 8 characters in length and should contain a combination of upper case, lower case & numbers. When users mistakenly input false information, error messages show right away in red underneath the field to guide correction, so that data is valid and security-standard-proof.

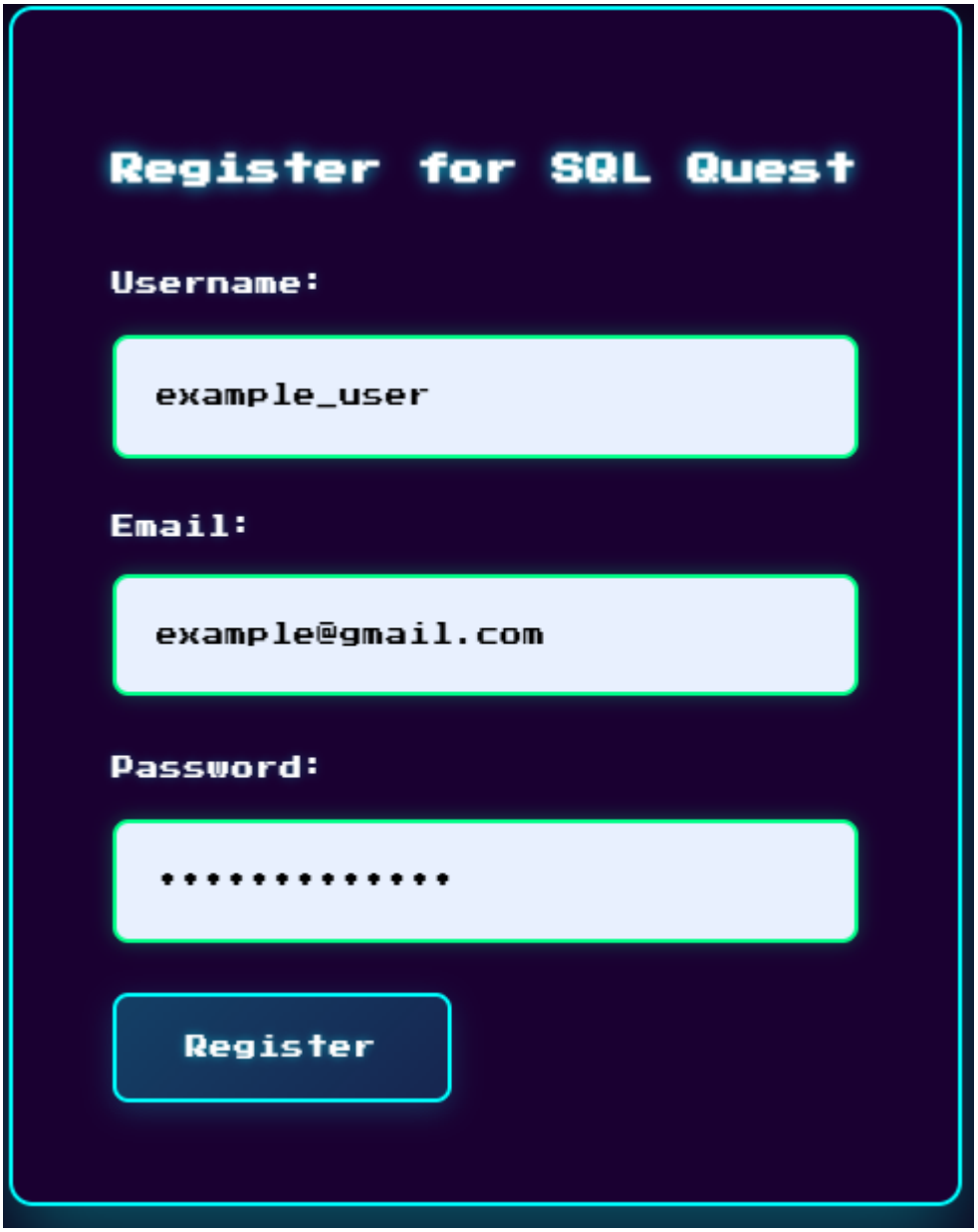


Figure 5.15 Successfully Completed Registration Form

Figure 5.15 shows a well-formed registration form, filled out with an acceptable username, e-mail address, and password. When the input is complete, you can use the Register button to send your form. Such entries are revalidated once they are submitted, and if they're legitimate then the system saves them into database.

id	username	email	password	score	created_at	completed_levels	current_chapter	current_lesson	sql_progress	overall_progress	hard_level_time	avg_hard_time
1	yinkin	yinkin26@gmail.com	$2b$10$7bW68AqUCQKksoW96MTDB.0ZMH.qS...	2865	2025-04-15 01:17:25	{"set1":["easy","medium","hard"],"set2":["easy","medium","hard"]}	NULL	Intro,DataTypes,Syntax,challenges,Table,Inse...	0	0	3	8.42
2	kin	vsdonkeyzai@live.com	$2b$10$.EshWgBEzpPCq3pfsVzy1uOR2SQfO1k...	170	2025-04-15 01:51:05	{"set1":["easy"]}	NULL	Intro	0	0	0	NULL
5	1234	yinkin23@1utar.my	$2b$10$gfsAj31YizBHnS.1KwYzzuMJpPP22EmO...	10	2025-05-01 21:57:42	NULL	NULL	NULL	0	0	0	NULL
7	yinkeong	yinkeong@gmail.com	$2b$10$RYFOUxyVoSj34Xu10fjb0.0wu2B6vJtB...	10	2025-09-20 18:07:51	NULL	NULL	intro,Syntax,DataTypes,Select,Select_Distinct,...	0	0	0	NULL
8	example_user	example@gmail.com	$2b$10$87evQP4tu8YoaFLBuDbDpOgwNqdWW...	0	2025-09-20 23:50:51	NULL	NULL	NULL	0	0	0	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 5.16 User Data Stored in Database

Figure 5.16 Shows the user details saved in the MySQL database with table name users as shown below. It includes inputs of user ID, username, email, encrypted password, account creation timestamp score scores completed levels current chapter current lesson proven indentation and average time hard level. Passwords are hashed before saving which makes impossible accessing users data without authorization. It also sets up tracking fields like XP, progress and completed challenges which will be updated as the user moves ahead with lessons and challenges.

5.4.3 Login Page

Figure 5.17 Login Page Interface

Figure 5.17 illustrates the SQL Quest login screen, which enables registered users to securely log into the system. And the page itself has an aggressively future-y, neon feels to it that fits the gamified angle of the system. There's a light switch on the top right of the page to switch between light and dark theme for your preferred viewing experience. The primary login form is in the middle with two boxes: e-mail address and password as well as a Login button. With this straightforward and user-friendly interface, users can authenticate in a jiffy and head on to their customized dashboard.

Figure 5.18 Invalid Email Format Warning

As you can see in Figure 5.18, the system pro- vides form validation out of the box. If a user logs in with an incorrect email (for example, without @) sends the system a warning message: "Please enter an '@' in the email address." This interactive validation will help prevent invalid submissions and keep you informed as to what makes a valid email.

Figure 5.19 Empty Field Validation Warning

Further, Figure (5.19) demonstrates the system validated empty input fields for login. If the user leaves the email and password field empty, when he clicks on send a message "Please fill out this form." pops up in no time. That validation is for each field on its own—I leave the email blank, it submits and opens a warning under the email; I leave password blank, a warning appears below that input. That way users shouldn't be able to proceed until they have completed credentials which satisfies security and stops them from submitting half the form.

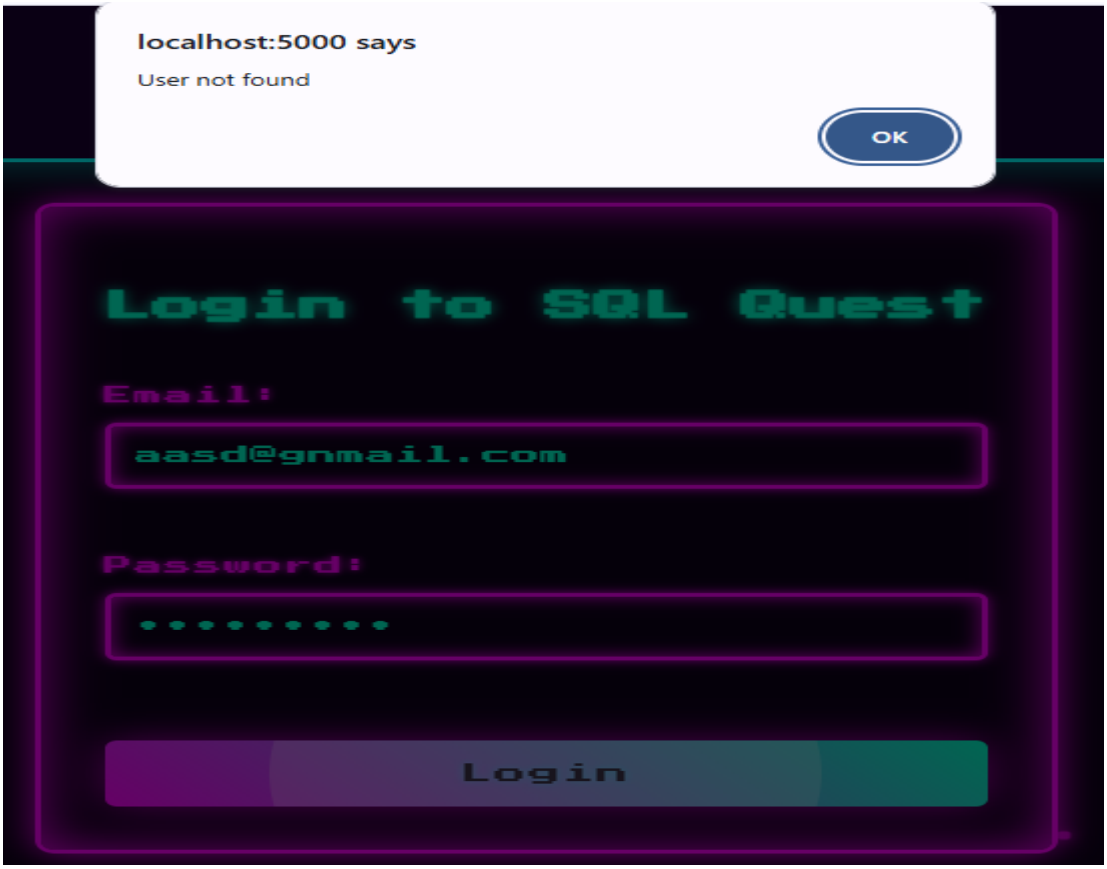


Figure 5.20 User Not Found Error

The login fails in Figure 5.20 because the email entered does not exist in the database, as shown. The system alerts “User not found” in error pop-up. This response notifies the user that either the email is wrong, or account does not exist, and that they can retry with correct details or else sign up.

5.4.4 Lesson Page

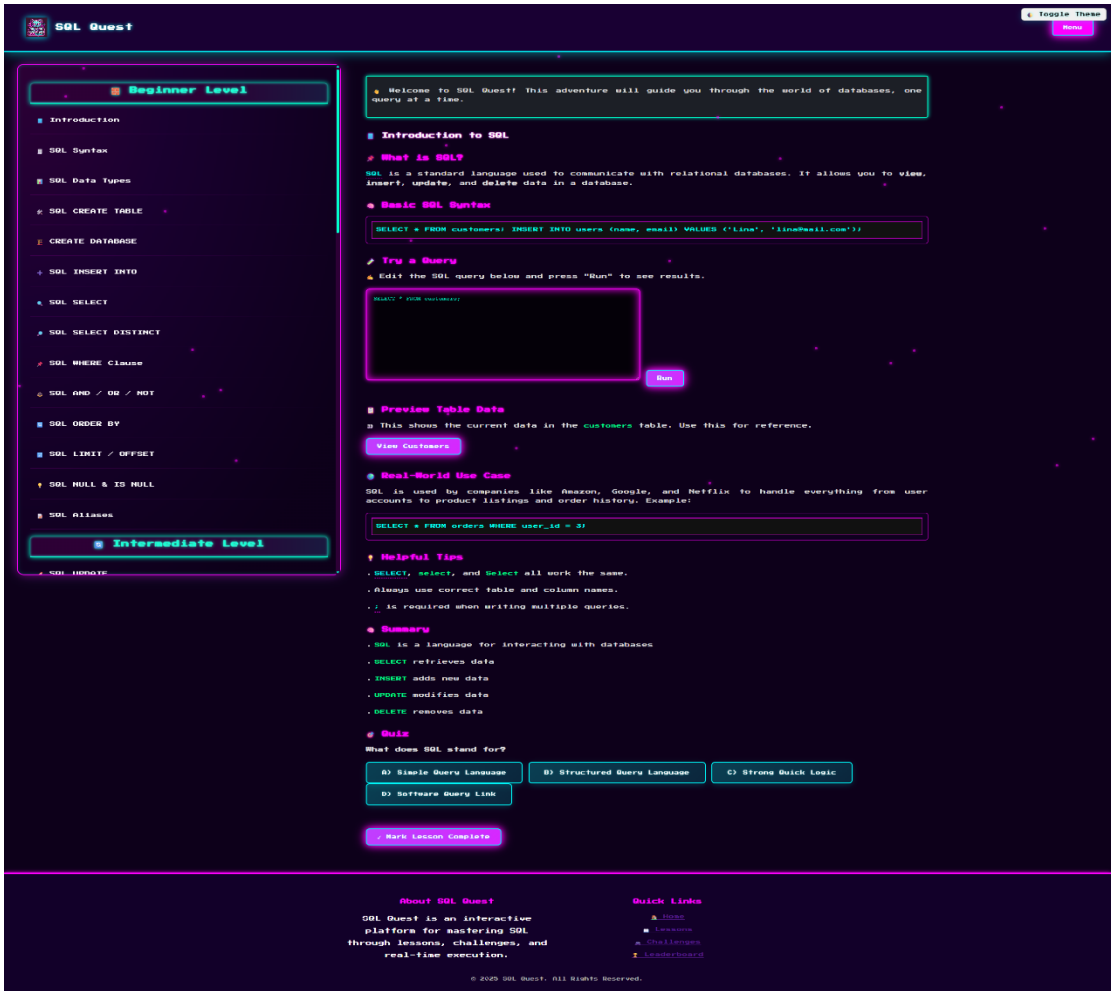


Figure 5.21 Lesson Page Structure

The lesson page layout for users viewing and not logged in is as what is shown on figure 5.21. Over on the left, to the side of the page you'll find a SQL sidebar divided into Beginner and Intermediate Levels covering such lessons as SQL Syntax, Data Types, and Create Database. The core content section features the chosen lesson (namely "Introduction to SQL"), which sections out topics such as What is SQL, Basic SQL Syntax, Try a Query (complete with code editor), Preview Table Data, Real-World Use Case, Helpful Tips, Summary and Quiz. This ascertains that there are theory parts, practical case studies and conducive testing at every lesson progressing systematically. But, pre-login, learners can view and practice content without any functions functioning (no bookmarking or tracking of completed lessons).

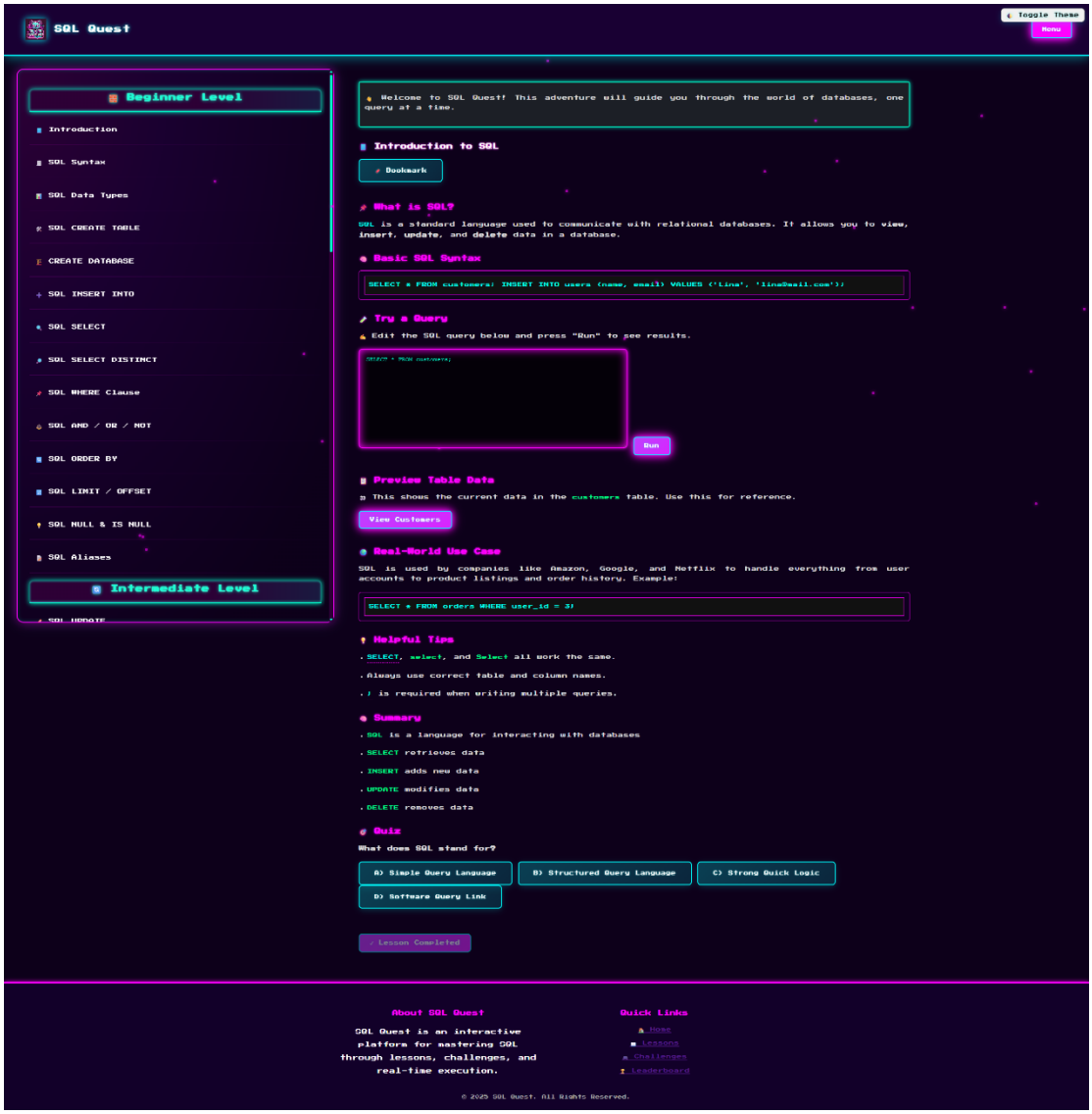


Figure 5.22 Lesson Page with Login Features

Figure 5.22 is the page lesson logged on, beside the usual present interactive elements (fig. In addition to the course content displayed in Figure 5.21, there are two new features/buttons: a Bookmark button (Screenshot 5.23 and Screenshot 5.24) so users can save the lesson for later use, and “Mark Lesson Complete” button which gives learners an opportunity to track their completion throughout the course. These new features customize learning experiences by empowering logged-in users to control their study flow and track lessons learned.

Figure 5.23 Bookmarking a Lesson

The Bookmark feature is shown in Figure 5.23. Readers can bookmark a lesson to use at any time. When you do so, the system will let you know that the counter has been saved and give you the option to remove it from your bookmarks (see Figure 5.24). This feature allows better navigation and personalization.

Figure 5.24 Removing a Bookmark

The interface in Figure 5.24 depicts a lesson that has been bookmarked, and the button now reads Remove Bookmark. This affords people the option to determine all of their frequent test prep reviews.

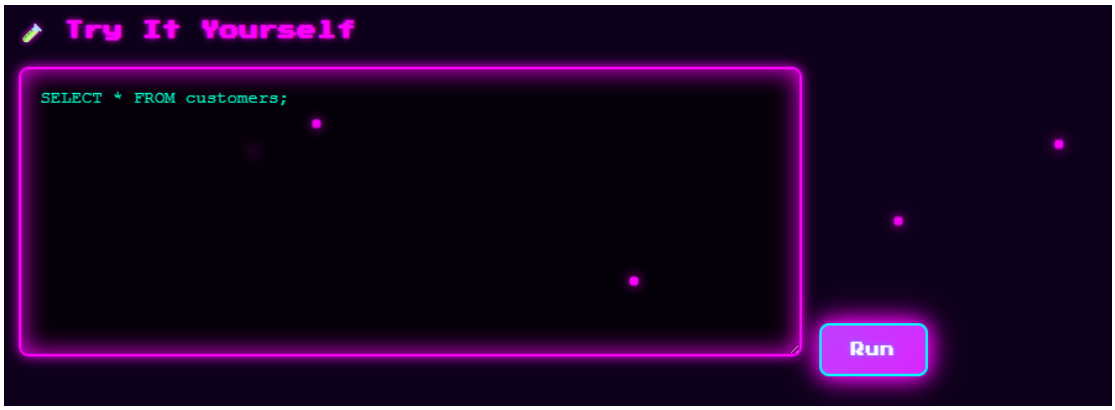
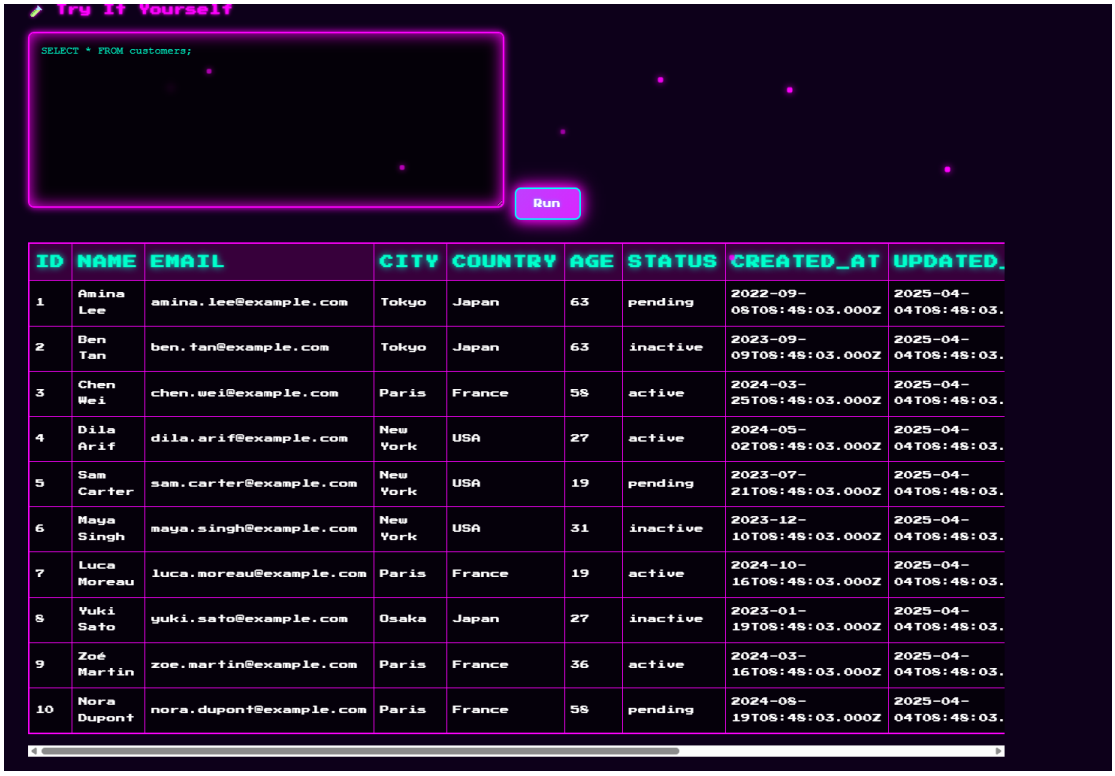


Figure 5.25 Interactive SQL Query Editor

The Try It Yourself query editor is shown in Figure 5.25, Users can enter SQL commands (e.g., SELECT * FROM customers;) and run them typing into the browser's console then clicking on "Run" button. It is instant practice in lesson setting.



ID	NAME	EMAIL	CITY	COUNTRY	AGE	STATUS	CREATED_AT	UPDATED_
1	Amina Lee	amina.lee@example.com	Tokyo	Japan	63	pending	2022-09-08T08:48:03.000Z	2025-04-04T08:48:03.
2	Ben Tan	ben.tan@example.com	Tokyo	Japan	63	inactive	2023-09-09T08:48:03.000Z	2025-04-04T08:48:03.
3	Chen Wei	chen.wei@example.com	Paris	France	58	active	2024-03-25T08:48:03.000Z	2025-04-04T08:48:03.
4	Dila Arif	dila.arif@example.com	New York	USA	27	active	2024-05-02T08:48:03.000Z	2025-04-04T08:48:03.
5	Sam Carter	sam.carter@example.com	New York	USA	19	pending	2023-07-21T08:48:03.000Z	2025-04-04T08:48:03.
6	Maya Singh	maya.singh@example.com	New York	USA	31	inactive	2023-12-10T08:48:03.000Z	2025-04-04T08:48:03.
7	Luca Moreau	luca.moreau@example.com	Paris	France	19	active	2024-10-16T08:48:03.000Z	2025-04-04T08:48:03.
8	Yuki Sato	yuki.sato@example.com	Osaka	Japan	27	inactive	2023-01-19T08:48:03.000Z	2025-04-04T08:48:03.
9	Zoé Martin	zoe.martin@example.com	Paris	France	36	active	2024-03-16T08:48:03.000Z	2025-04-04T08:48:03.
10	Nora Dupont	nora.dupont@example.com	Paris	France	58	pending	2024-08-19T08:48:03.000Z	2025-04-04T08:48:03.

Figure 5.26 Query Execution Result

Figure 5.26: After running a query in the editor, we see the results in formatted table form. Example Output from the customers Table with fields such as ID, Name, Email, City, Country, Age and Status. This is a very nice crossover from learning into real database interaction.

Figure 5.27 Practice Section with Error Feedback

Figure 5.27 shows the Practice Section which prompts users to query against a table (e.g.," Write a query to select product name, price from products"). If a bad search is entered, the system will return an error message as feedback giving clues to help realize the correct syntax.

Figure 5.28 Practice Section with Partial Feedback

Figure 5.28 The user's query is correct or close, but its syntax or structure is incorrect. The feedback of the system, for example, is "Check column names and syntax again," which offers constraining guidance not to get frustrated.

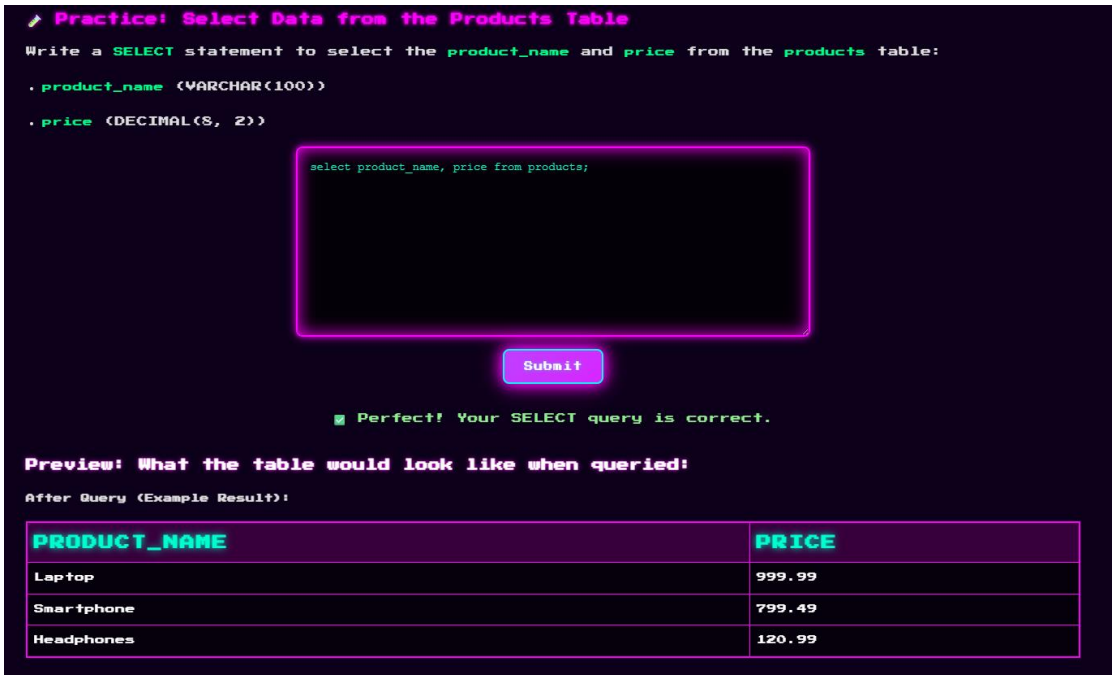


Figure 5.29 Correct Practice Query Execution

Figure 5.29 Successful attempt when the query syntax is successfully constructed. Upon successful completion the system responds with a green success message and renders the result table which demonstrates what an actual database query output looks like (Laptop, Smartphone, Headphones with their prices).

Figure 5.30 Quiz Section with Explanation

In figure 5.30 Section We show the quiz section of each lesson. Users answer multiple-choice questions (e.g., find names and cities on the customers table). Wrong tries lead to an extensive explanation, so that learners realize their mistakes and remember the correct format of a SQL query.

Figure 5.31 Quiz Completion Feedback

Figure 5.31 The quiz portion of the lesson: an interactive XQuery exercise in which users are challenged to select the SQL query that will extract values (names and cities) from a customer's table. When the appropriate selection is made, the system issues feedback in real-time and posts a confirmation message: Correct! Quiz completed!". Further down the confirmation we have a section to explain why that query is right, this way completing the learning by making sure SELECT name, city FROM customers; correctly brings back the desired columns. By doing that, it confirms the answer of the user and at the same time ensures conceptual understanding through explaining its structural syntax.

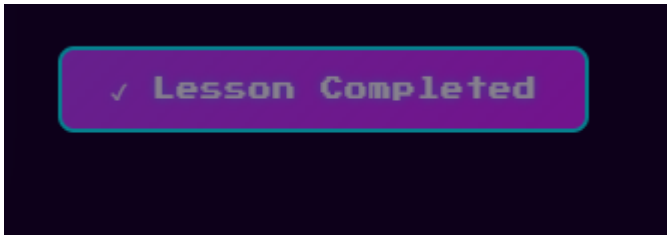


Figure 5.32 Lesson Completion Status

Figure 5.32 shows the contents page of a lesson when you have completed all its parts and allows you to use the buttons on screen to move between lessons which are handy features. A button that says "Lesson Completed" comes up telling the user that he/she has done the exercise. This button, when clicked, updates a user's progress and enables them to see which lessons they have done so far. This feature is provided with a key to help participants keep track of their progress and their commitment but also as a system for unlocking other levels/steps or sections.

5.4.5 Real-World Project Page

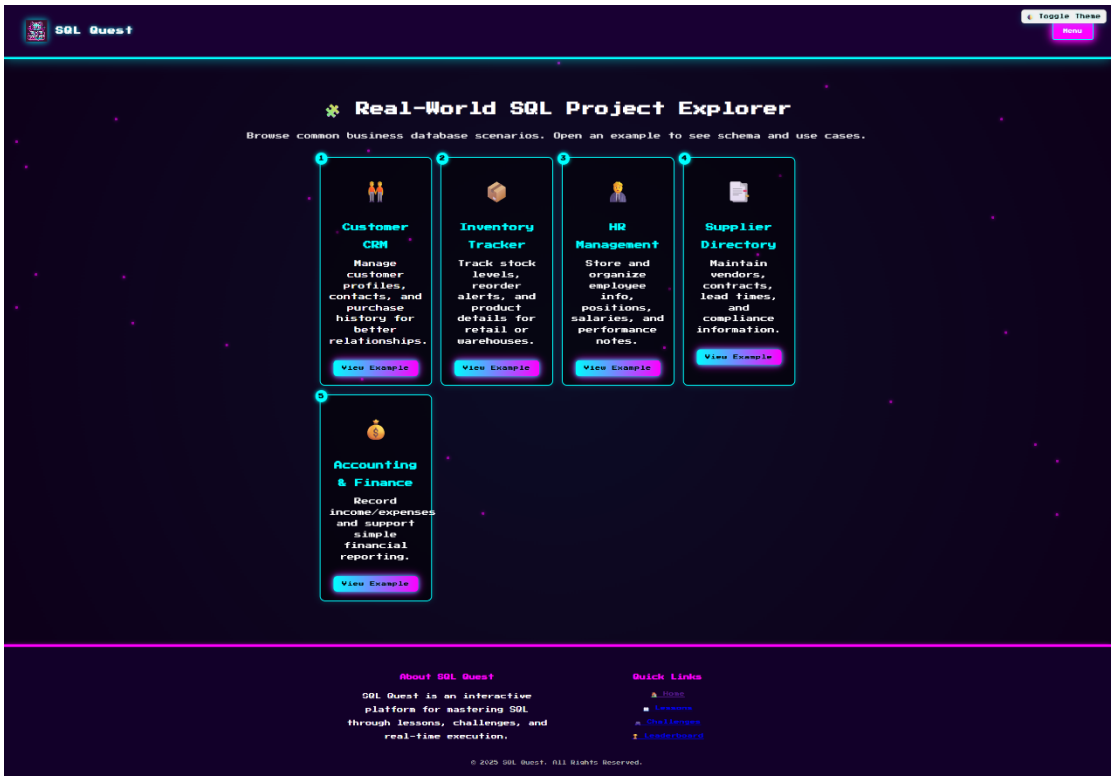


Figure 5.33 Explorer page of the Real-World project

The Real-World Project Explorer page (Figure 5.33) is the portal to applied learning modules. Here, you can look at some user-favorite business-oriented SQL project ideas like Customer CRM, Inventory Tracker, HR Management, Supplier Directory and Accounting & Finance. Each module has a brief overview and a “View Example” link that directs students to detailed use cases, sample queries, and data sets. This structure adds necessary validity, as it relates SQL with business in the real world.

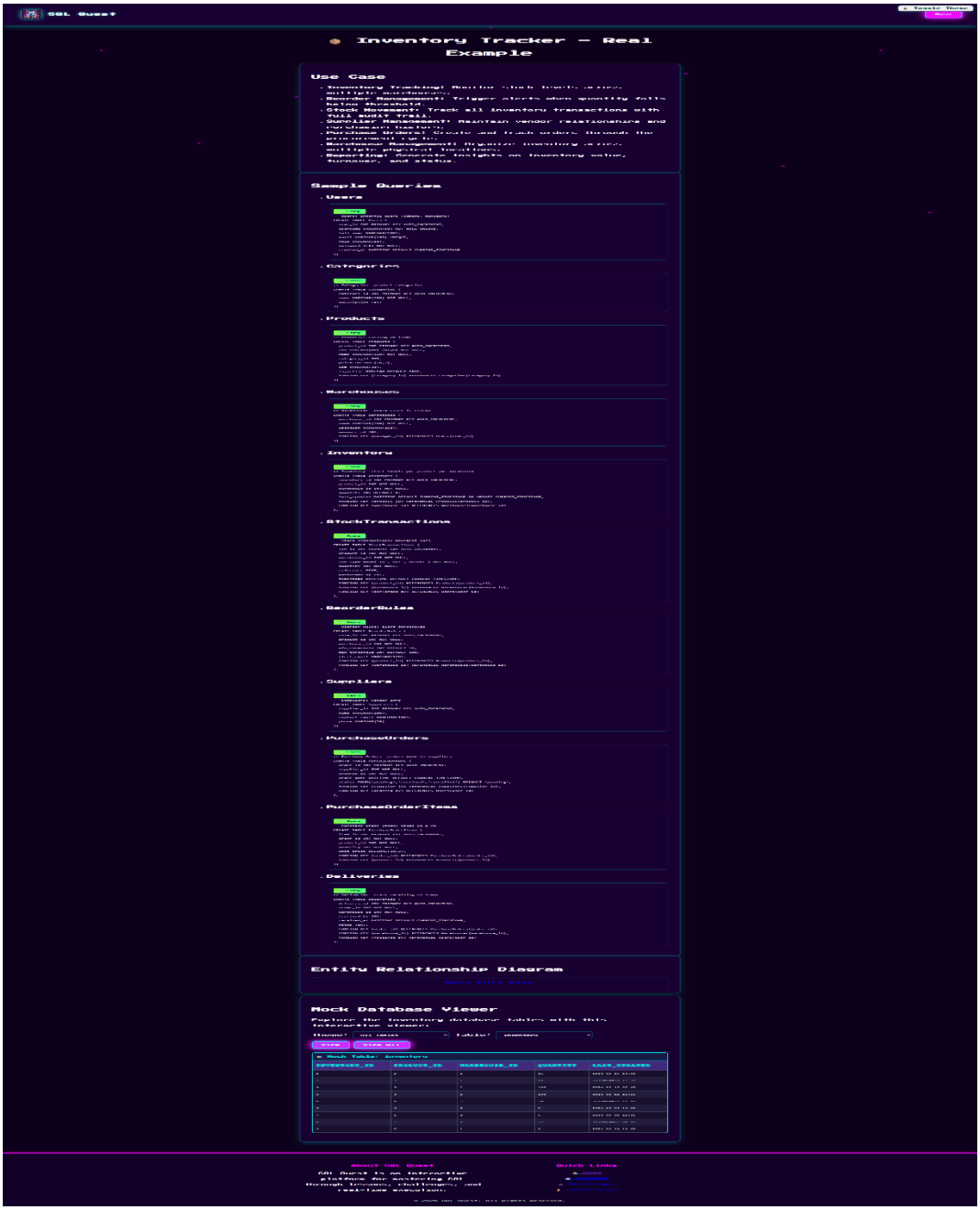


Figure 5.34 Full Example Page

Figure 5.34: Full Example Page Once a user clicks on any one of the real-world projects offered in SQL Quest, the Full Example Page appears as in Fig. The system comes with core modules like Customer CRM, Inventory Tracker, HR Management, Supplier Directory and Accounting & Finance. For illustration the Inventory Tracker module is presented. The page is broken down into subpages to provide comprehensive explanations for students about the chosen project. At the highest level of structure, the Use Case portion describes the business or other real-world context for which use of the module is intended, e.g., tracking inventory (stock levels), supplier management, or purchase order processing. Underneath that, the

Sample Queries shows organized SQLs for corresponding tables with a copy tab advocating you to be hands-on. Also, on the page there is an ERD (Entity Relationship Diagram) to help learners better understand relationships between the tables of your database. Finally, is the Mock Database Viewer that enables browsing and interacting with simulated data so when sifting through lists of tables and columns for them to test their queries. All together, these parts of the project make sure each example shows practical applications, SQL queries, relational model and real-data simulation.

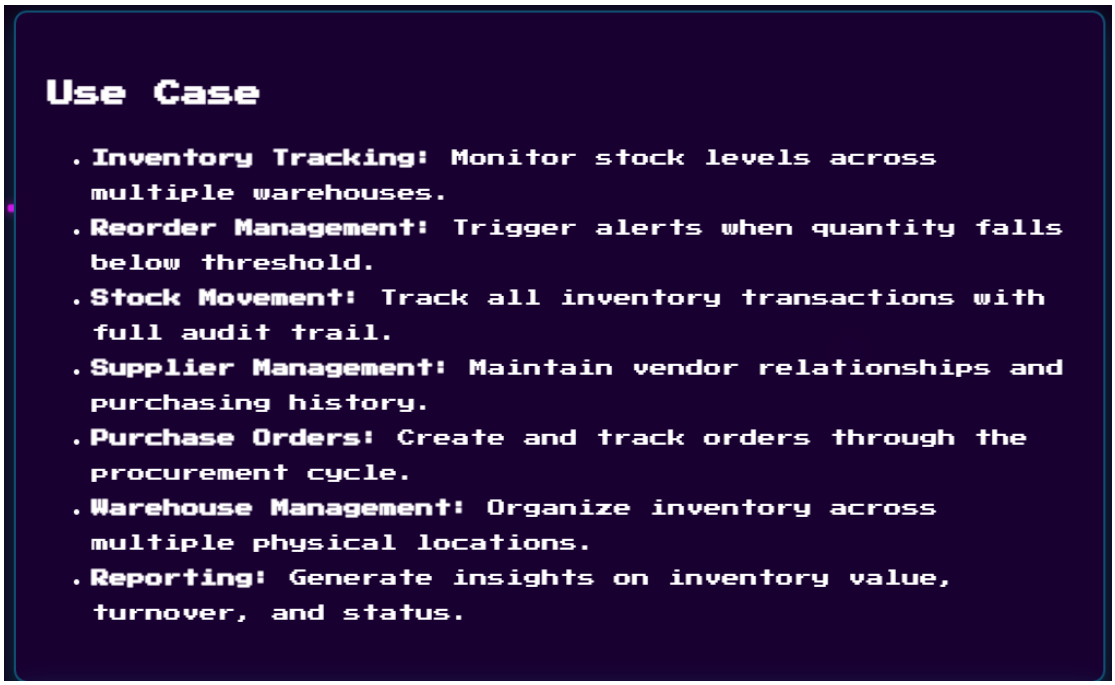


Figure 5.35 Use Case Overview

Figure 5.35 depicts the Use Case section for the chosen real-world project. This section provides an overview of the main business processes supported by the system: stock control, reordering functionality, movement and audit trails, supplier management, purchase orders (including scanning), data warehouse operations and reporting. Each point is on how SQL can be used to carry out these Functions so that students will have a working understanding of Database Applications.

Figure 5.36 Sample Queries Section

The Sample Queries section is displayed in Figure 5.36, which provides learners with predefined SQL to be used with various entities, such as User, Category and Product. These are examples of how to manipulate the table, adding a primary key, unique constraints and fields. This section assists learners in bridging the gap from abstract SQL syntax to actual data model and table creation.

Figure 5.37 Entity Relationship Diagram (ERD) Access

The text is hyperlinked to the ERD (Entity Relationship Diagram) showing all columns in a node within Figure 5.37, which readers may click on the link to browse through Figure 5.38 The Entity Relationship Diagram (ERD) link can be clicked and viewed full size similarly to images. Graphical form in a separate tab or window of their browser. This section becomes important to understand how various tables and entities are related with each other in the database schema and it provides a shape or structure-based understanding of databases.

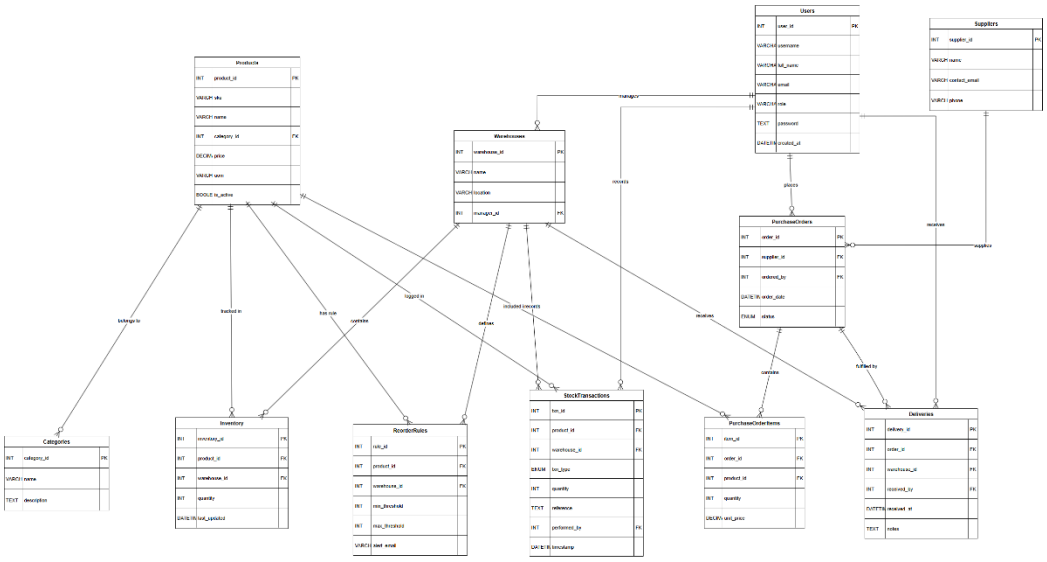


Figure 5.38 Column Relationship (ERD) Visualization

The overall ERD of the system is shown in Figure 5.38 that shows tables such as Products, Categories, Inventory, Warehouses, Stock Transactions Reorder Rules Users Suppliers Purchase Orders etc. The diagram emphasizes the links (one-of-multiple and multiple- of-one) among entities, which train students to interpret complex schemata and relational associations in use.

Figure 5.39 Mock Database Viewer – Viewing a Single Table

Figure 5.39 The Mock Database Viewer - learners can interactively work with the database tables. In a typical representation, the Inventory table could look like something like shown with columns such as Inventory_ID, Product_ID, Warehouse_ID, Quantity and Last_Updated. This functionality enables users to mimic browsing real datasets, which reinforces their testing skills of querying in a real-world context.

Figure 5.40 Mock Database Viewer - Relations of Tables

Furthermore, in Figure 5.40 the Mock Database Viewer has been enhanced to support multiple data sets and table relationships. The above schema tells us how tables Products, Inventory, Stock Transactions and Reorder Rules are related to each other which is much cleaner in the upper part of the picture and data stored in inventory management over all product items, stock transactions are shown below. The fusion of schema relationships with data views gives the learner conceptual as well practical understanding about relational databases.

5.4.6 Chat Assistant (AI Tutor) Page

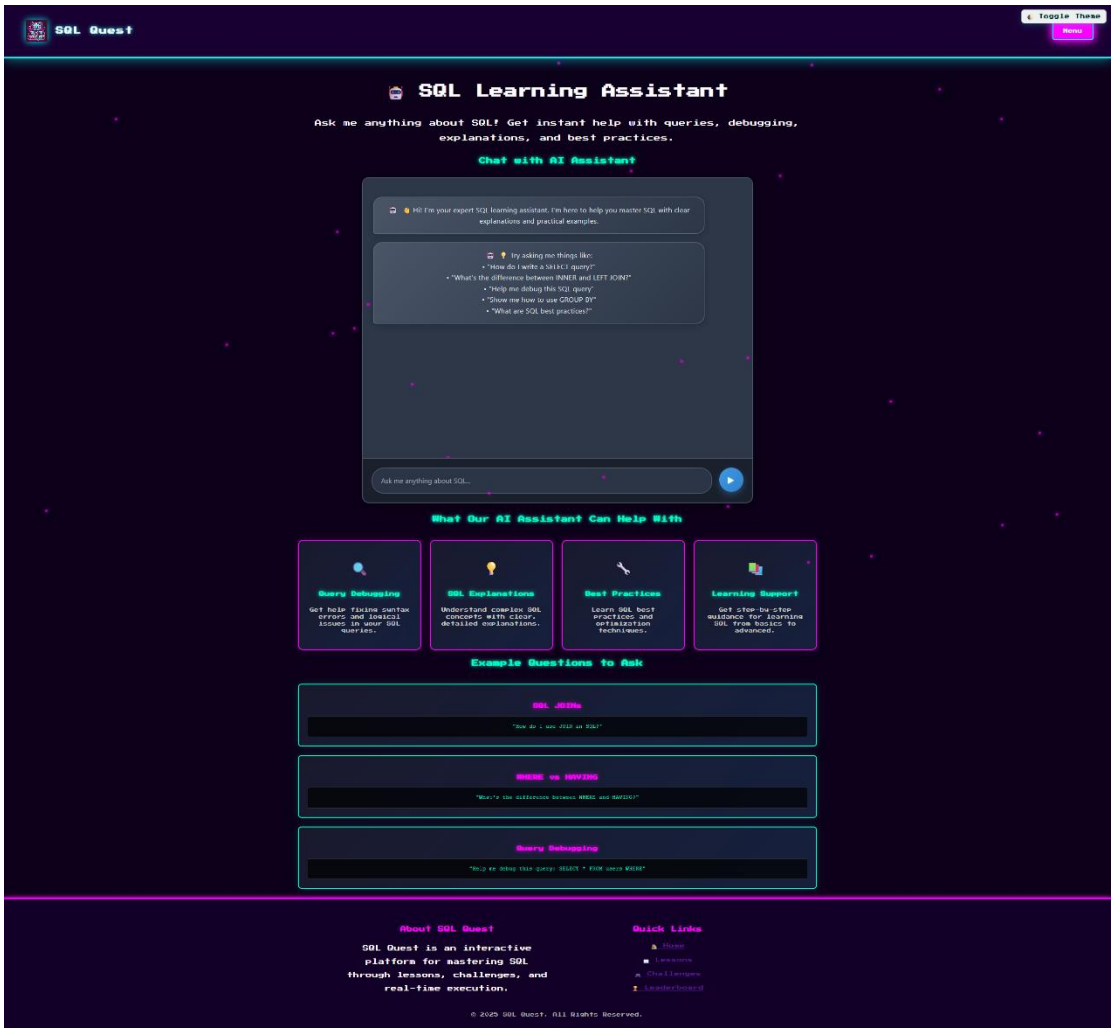


Figure 5.41 Briefing of the AI Chat Box

Figure 5.41 The SQL Learning Assistant as an AI-based chat box integrated via the ChatGPT API. This feature makes it easier for learners to ask whatever (SQL-related) questions they may have and get instant explanations, debugging tips, and best practice advice from SQL experts. In the middle of everything is the chat interface, where users can type in their questions. To support learners, recommended prompts are like: “How do I write a SELECT statement? and “What is the difference between INNER and LEFT JOIN? are provided. The lower part of the homepage features categories such as Query Debugging, SQL Explanations, Best Practices and Learning Support to reflect what its assistant can do for you. Sample questions are shown to facilitate users that they can quickly start conversations with.

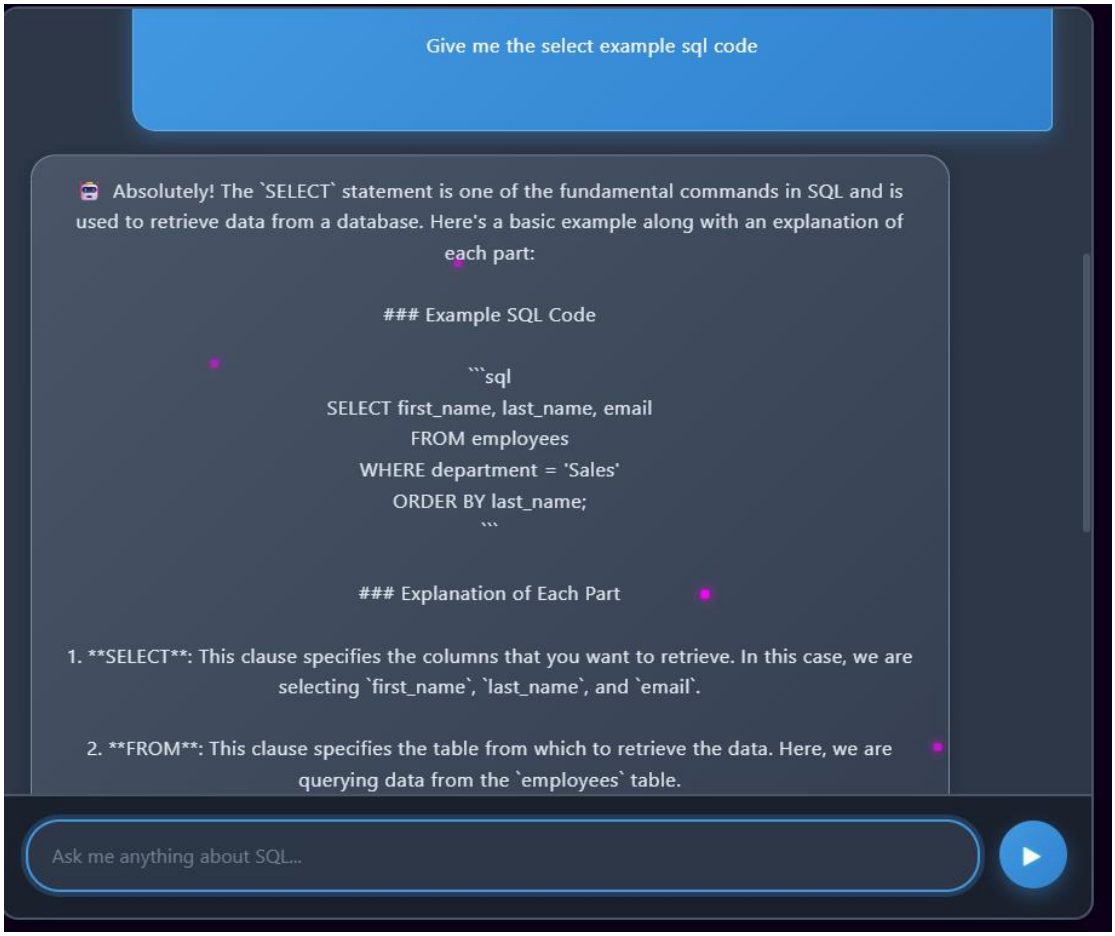


Figure 5.42: Example SQL Code Assistance

Figure 5.42 illustrates the way the AI chat box answers if a user's enters "Give me the example of SQL SELECT code" as an input to ask for the SQL sample case. The system uses the chatbot-powered ChatGPT API and returns an entire SQL query which allows the retrieval of employee names & email addresses, filtered by department and sorted by last name. Next to the code, there is an explanation of everything written which breaks down each part of -FROM, SELECT -ORDER BY and WHERE. Thus, students do not get just working examples and should grasp how each of the SQL components work.

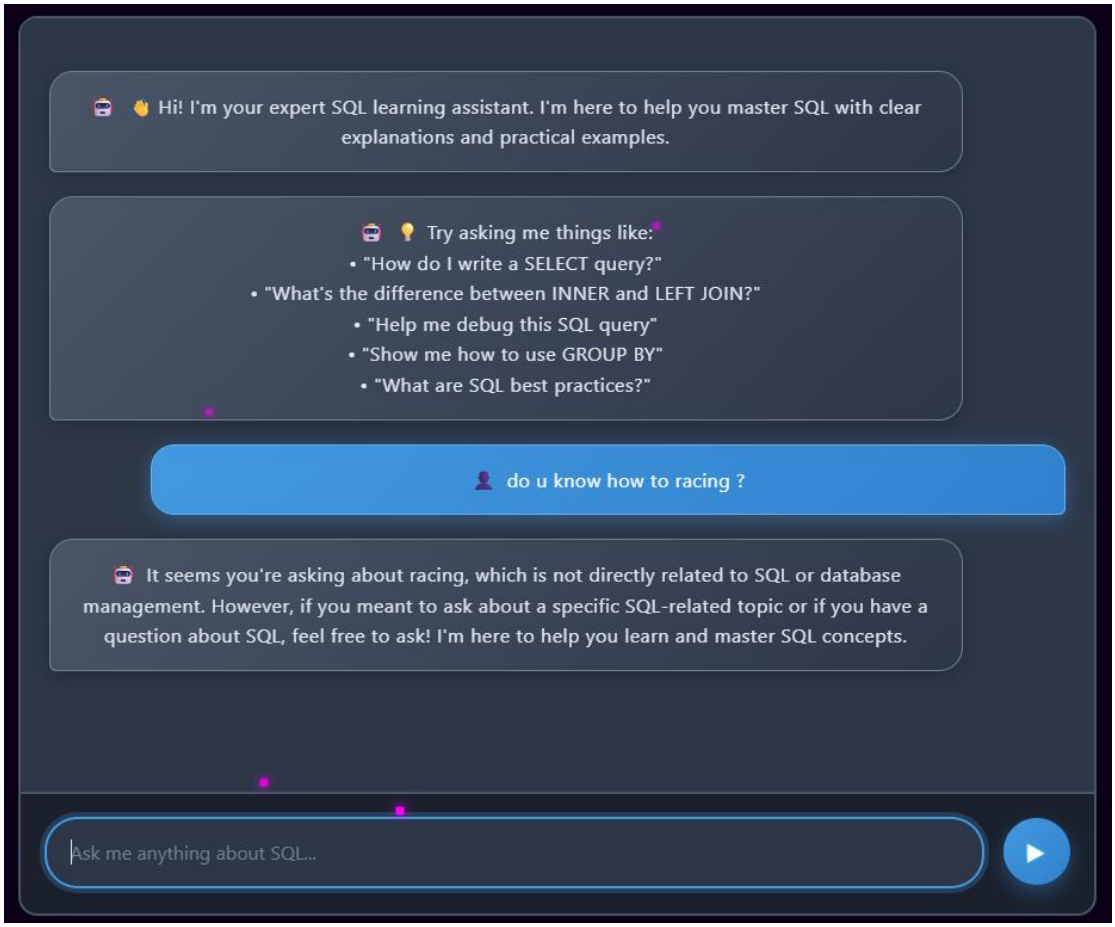


Figure 5.43: Handling Off-Topic Queries

Figure 5.43 Features that allows the AI to respond well to irrelevant or out-of-topic questions. When the user says: “Do u know how to racing? the assistant running on the ChatGPT API detects that the question is out of scope with respect to SQL. Rather than giving nonsense output, it graciously tells the reader that this is not in fact suitable input for learning SQL, but also suggests they come back with some SQL questions. This keeps the learning environment on track and shows just how flexible AI can be.

5.4.7 Challenge Page

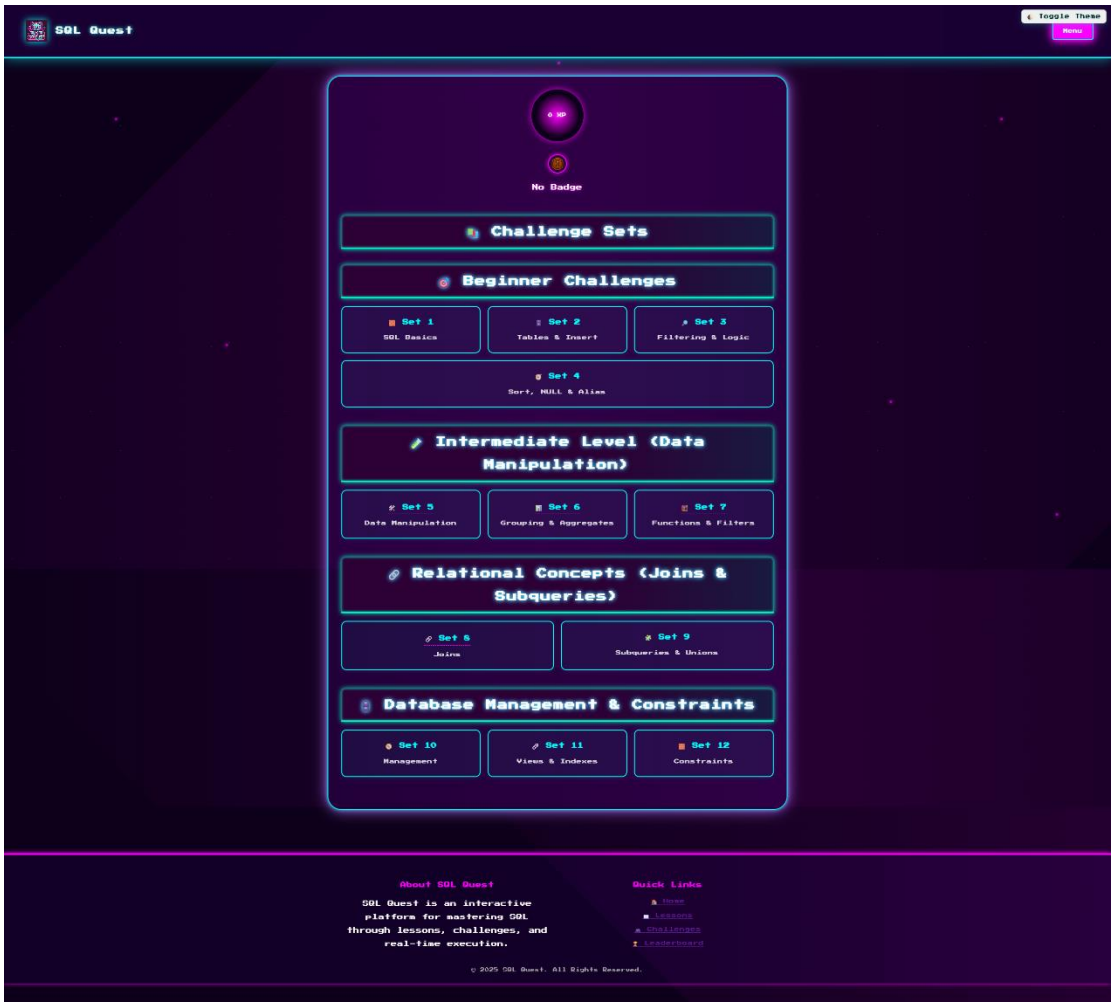


Figure 5.44 Challenge Sets Overview

Figure 5.44 presents the Challenge Sets Overview page that is utilized to illustrate how learning challenges in SQL are organized within the PLAR system. The page is organized into several sections including Beginner Challenges, Intermediate Level (Data Manipulation), Relational Concepts (Joins & Subqueries) and Database Management & Constraints. There are challenge sets available in their classes (e.g.: SQL Basics, Joins, Views & Indexes) that you can pick from. At the top, you see your XP and badge area to show player progression. In this demo, the Beginner Challenge Set 1: SQL Basics is chosen to demonstrate how the challenge feature works.

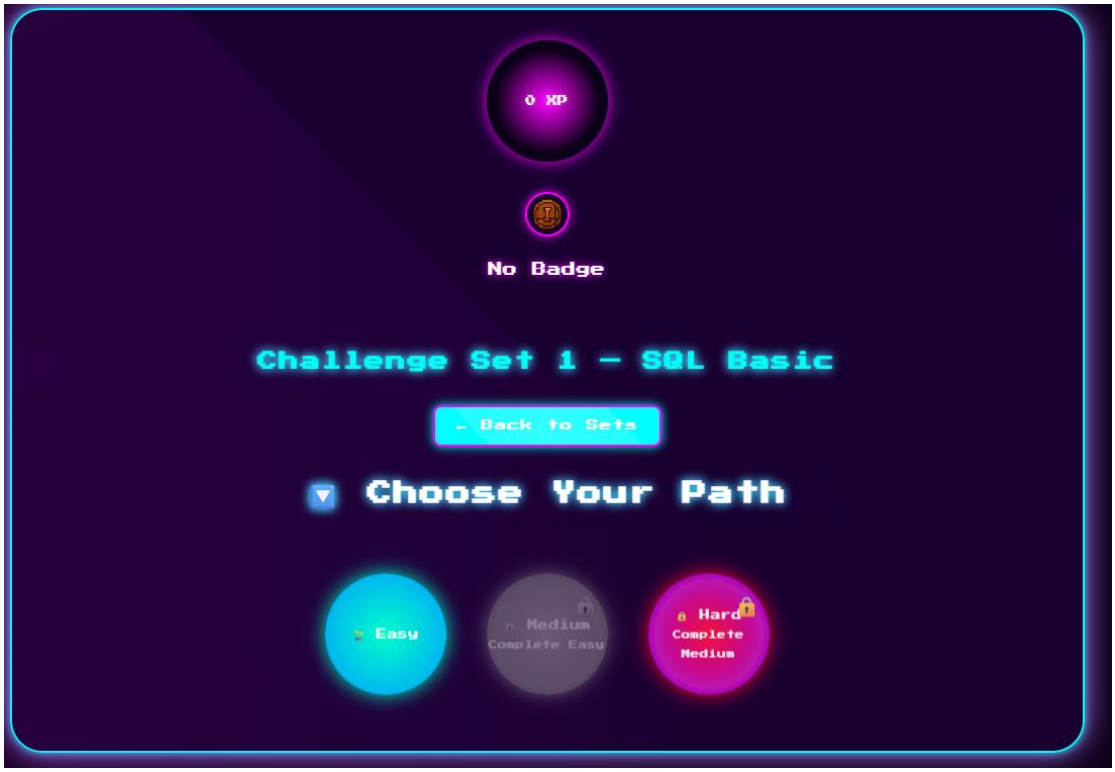


Figure 5.45 Challenge Level Selection

Figure 5.45: The screen where users can select the difficulty for a challenge set is displayed in figure 5.14. The architecture offers three categories of levels: Easy, Medium and Hard. To start with Easy is the only level accessible, by reaching a certain distance the Medium and Hard options will become unlocked when reached. This page is meant to be the starting place for learners who are ready to choose a focus area based on their current skill set.

Figure 5.46 Challenge Question Interface

Figure 5.46 shows a representation of how the system would pose questions within a challenge to the user. The platform will present both MCQ and typed input questions to learners, allowing them the chance to check not only their recognition skills but also how well they can write queries. A MCQ is shown to the learner in this demo, and photos are displayed too. This shows challenging elements embedded into an interactive and gamified design.

Figure 5.47 Correct Answer Feedback

The feedback process in case the learner is correct can be seen in Figure 5.46. The selected option is green, and the system gives 5XP for answering the question. There's a bar that fills at the top as you complete and saying that next challenge will load automatically. This demo demonstrates how the system incentivizes correct responses and engages participants with XP points.

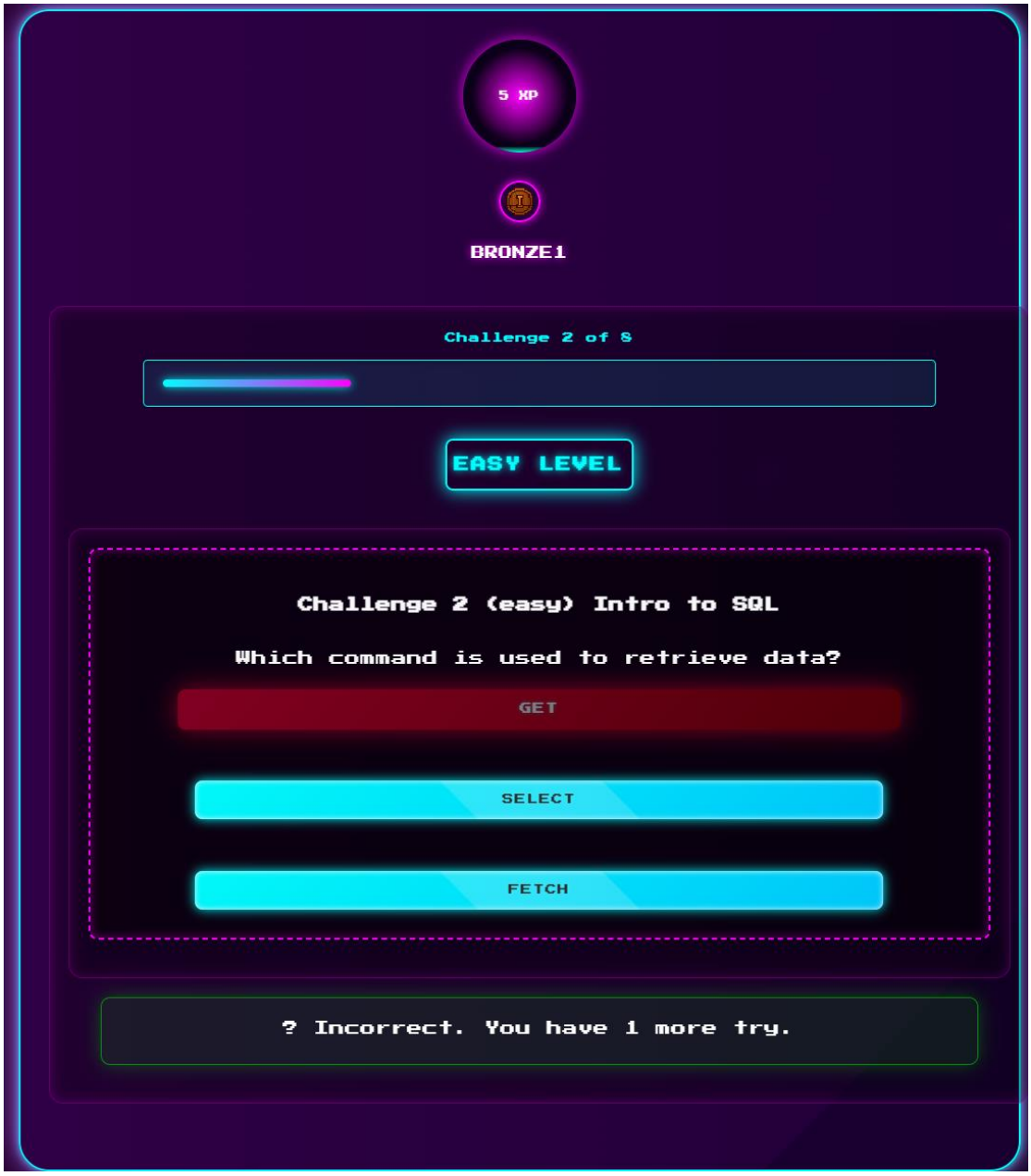


Figure 5.48 Incorrect Answer Feedback

Figure 5.48 shows what should occur when the learner chooses the wrong option. In this demonstration the command “GET” is answered instead of “SELECT”. The incorrect answer is in red with the response "Incorrect. You have 1 more try.” This really demonstrates that the system offers real-time feedback and extends to a second chance for users to remedy their error prior to continuing.

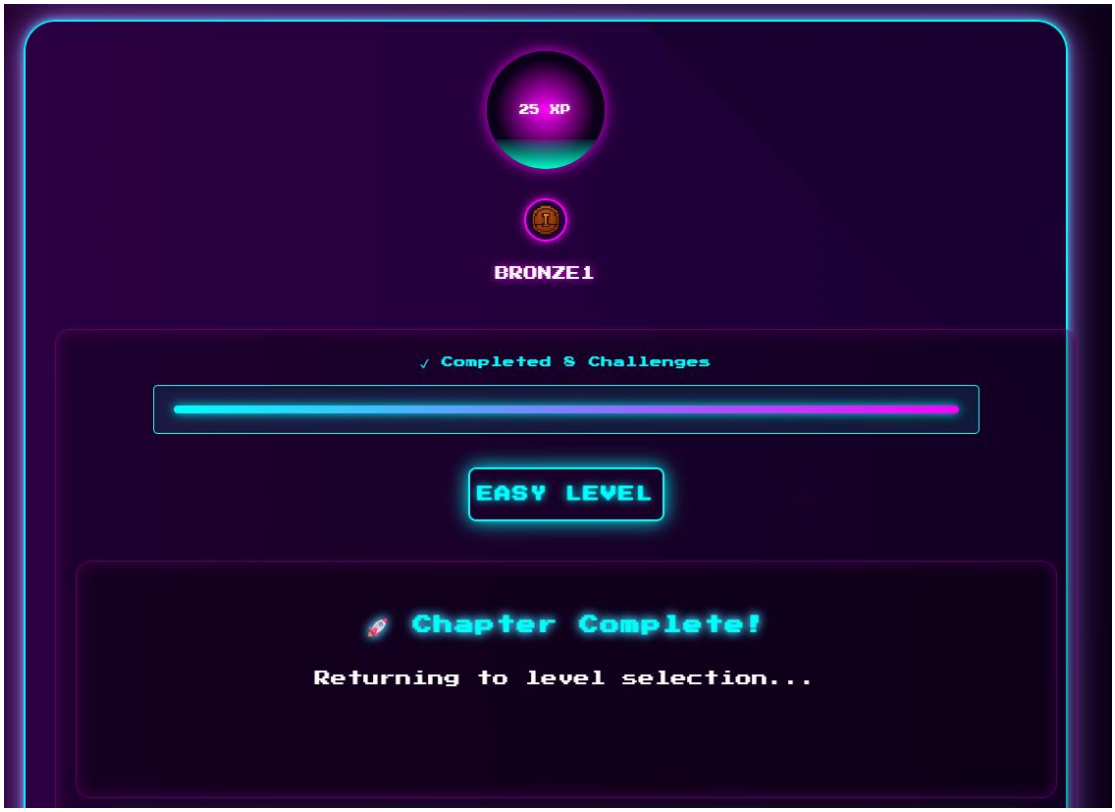


Figure 5.49 Chapter Completion Screen

Completion of Easy mode after completion of all questions in the challenge set is shown by Figure 5.49. The system will verify that the student read through the chapter, reward 25 XP, and generate a badge (Bronze1). In this demo you will see how the platform acknowledges learner progress and accomplishments at the end of each chapter.

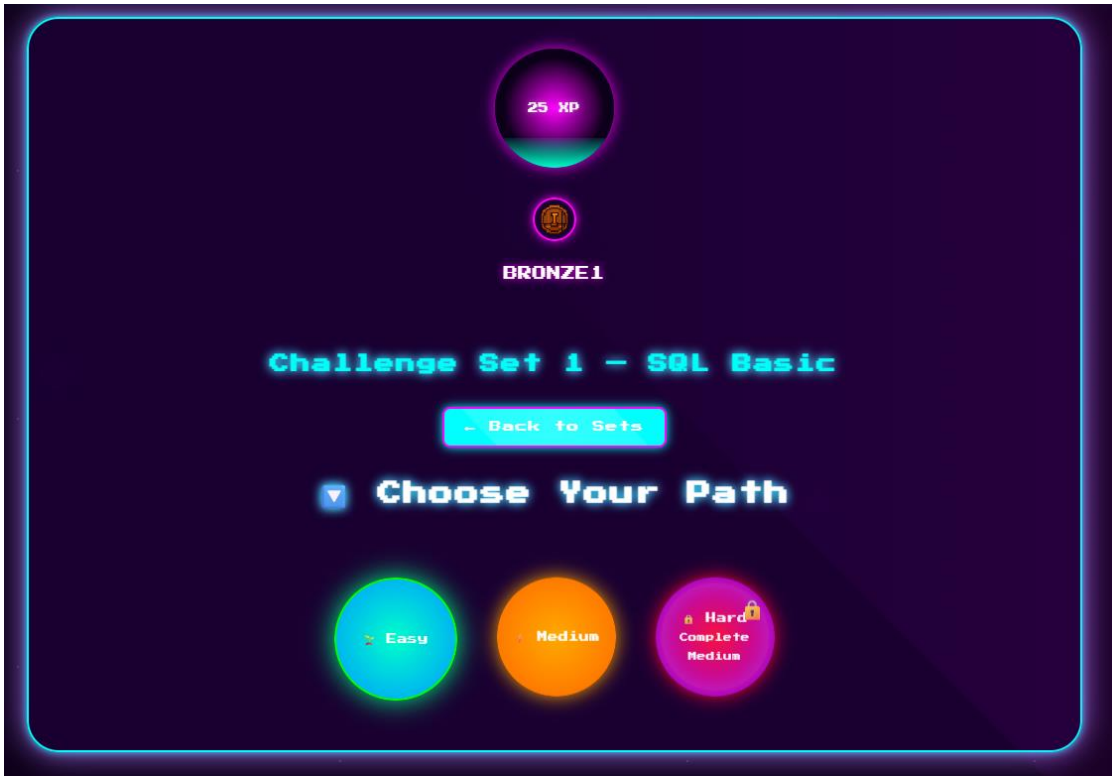


Figure 5.50 Path Choice on Completion

After the Easy Level is finished, the Path Selection Page is changed to what is shown in Figure 5.50. In this walkthrough, the medium path is unlocked, and the learner can proceed. The Hard level is locked until medium challenges are finished. This is a demonstration of how the system through progressive unlocking designs the learning path.

5.4.8 Leaderboard Page

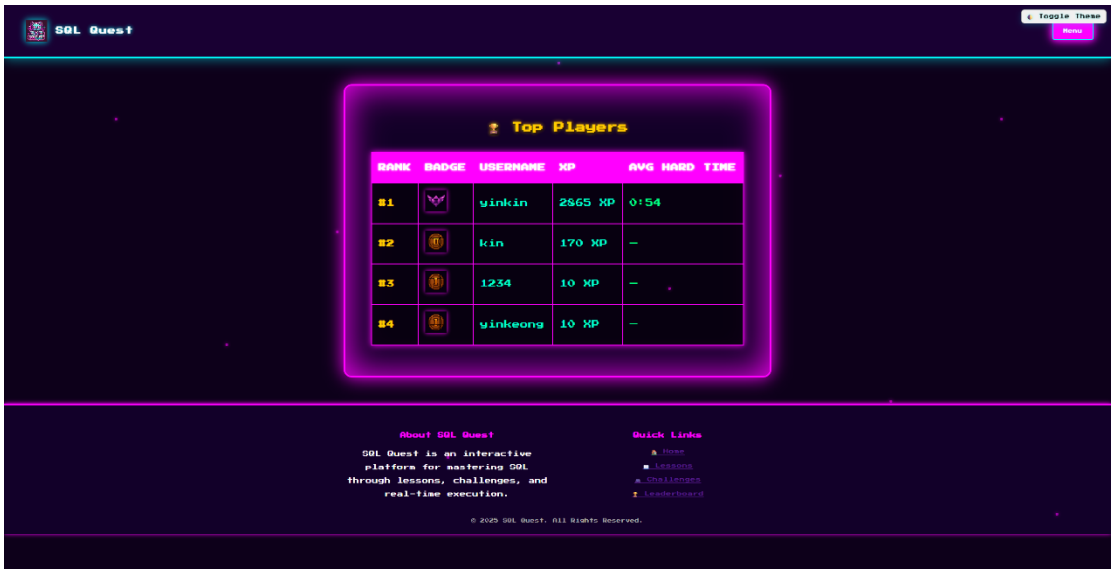


Figure 5.51 Leaderboard Page

As shown in figure 5.51, this page displays the top players who have earned the highest XP explained earlier in the user profile section. Additionally, players are placed based on ranking where the table includes the rank, badge, username, total xp, and average time taken for hard-level completion shown in the column besides it and determined on the dashboard. The ranking occurs based on the achieved total XP where the greater the XP, the higher the ranking is each hard. However, if two or more players have acquired the same XP, one's average completion time in hard-level challenges is considered to determine who ranks higher. This system rewards the achieved XP but also knowledge and efficiency; hence, it is a fair ranking system. Moreover, by displaying the top performers, it creates competition and encouragement among the users to complete more challenges to be top.

5.4.9 Profile Page

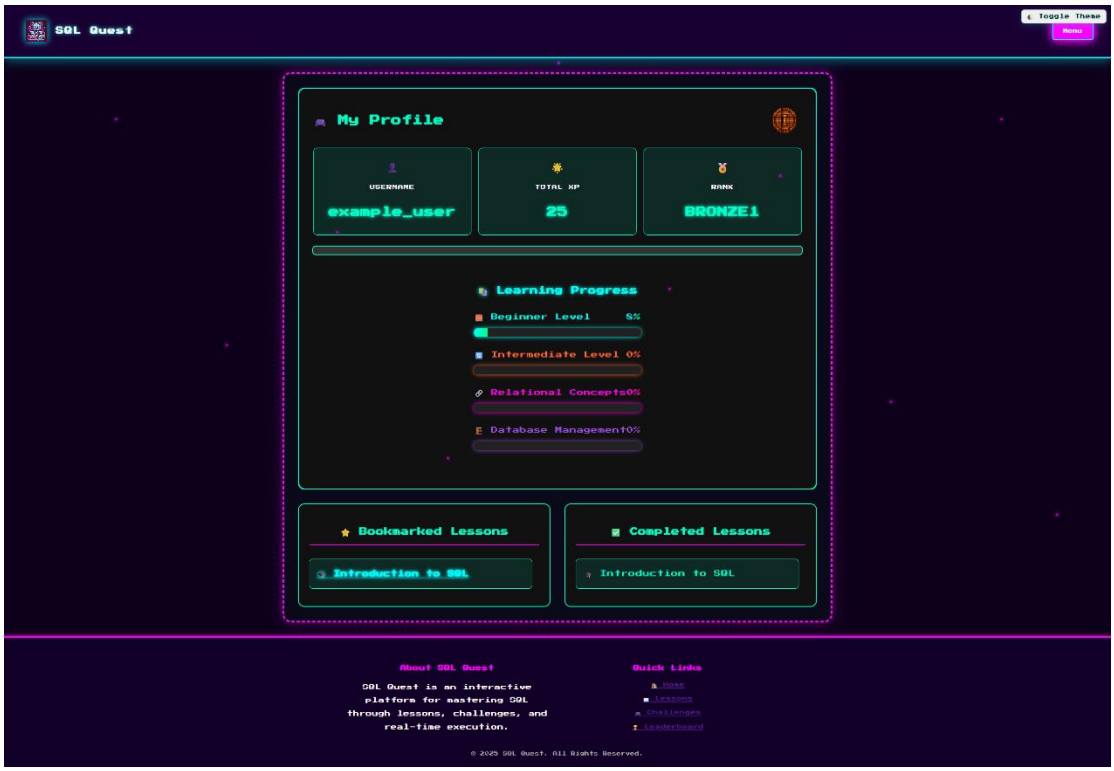


Figure 5.52 Profile Page

The Profile Page is the user-owned dashboard of the SQL Quest and is displayed in Figure 5.52. At the top of it, the page shows you some of the specific details about your profile such as your username, how much XP you've earned in total and, finally what's your current rank at that time (i.e.: Bronze, Silver or Gold). This is a nice overview of what this player has accomplished and how far they've come.

Under this, The Learning Progress section shows bars for areas such as Beginner Level, Intermediate Level, Relational Concepts and Database Management. Each progress bar shows the percentage completion of its corresponding module so that you can keep track of your progress in different areas of SQL learning. At the bottom of the page, you are given two panels: Bookmarked Lessons and Completed Lessons. The bookmarked section permits users to save the exact lessons they would like to go back to, and the completed section shows all the success stories on all the chapters (ones that you finished successfully), e.g. Introduction to SQL" Stuff. This design allows learners to easily view their accomplishments and areas they need to work on more and quickly absorb the history of learning personalized for them.

5.5 Implementation Issues and Challenges

Throughout the development, publication and release process of the SQL Quest system, numerous problems and challenges arose. It's mainly because of technical and user experience stalls that the main issues developed. They had to be solved and resolved for the application to work right, well and healthy.

Driver integration was one of the most important problems. Since SQL Quest heavily depends on mock datasets (CRM, HR, Finance, Inventory & Suppliers) and real-time queries, the database schema must be kept consistent and compatible throughout all the modules. In very rare cases, the data integrity type mismatch or missing foreign key relationship may have prevented some queries from being executed successfully. To battle this, we refined, optimized and carefully tested the schema against various example queries.

There was also an issue with user security and authentication. As it came time to implement a registration and login system, it became necessary to deal with bad inputs, securely hash and store passwords, and check email addresses. For instance, it was not obvious to encrypt passwords with hashing methods and still have a frictionless login flow. They also had to ensure that rigorous input validation, such as checks against empty fields or invalid email formats, was in place to prevent vulnerabilities or crashes.

The UI design was additionally challenging. SQL Quest was also anticipated to be a futuristic game-like environment, which could include interactive content such as quizzes, missions and bookmarks. Finding performance has been more difficult with such an immersive design, especially when certain pages, like those doing interactive querying, were hogging system resources. Polished performance was achieved through pixel-tuned CSS animations, optimizing responsiveness and layout, and by simplifying some of the features to get a faster load, both on desktop and mobile.

Real-time query execution was another substantial problem to solve. “Try It Yourself” and challenge features are implemented by having a backend that can safely handle SQL commands without sacrificing data integrity. Writing clear and informative error messages

for wrong queries was a real challenge, but it had to be both correct and useful for students. It took several iterations to fine-tune this feature, so the feedback from queries and errors contributed to learning.

Finally, the gaming and scoring logic implementation added more complexity. The platform had to perform accurate XP calculation, unlock levels (Easy, Medium and Hard), and update the leaderboard in real time. At first release, it had issues like wrong XP rewards for duels and such, a delay in updates and ranked wasn´t consistent. These issues were fixed by introducing details on the backend level and synchronizing with the database and frontend.

Conclusions: In general, SQL Quest encountered many issues in database integration, security, the design of the user interface, real-time query processing and gamification. And although these challenges initially stunted growth, at the end of the day, they became valuable lessons. By perfecting the schema, implementing security features, making changes to the design, and through iterative testing, the system was fortified and fully implemented for end users.

CHAPTER 6 System Evaluation and Discussion

6.1 Black-Box Testing

The SQL Quest system test was Va built in this section by carrying out black box testing on all features to verify if everything properly functions regarding different input and usage circumstances. The tests target system outputs, but don't look at any internal code. All modules were tested using valid and invalid input to verify that the functions are executing according to design.

The findings of the black-box testing are presented in sub-sections below.

6.1.1 Test Scenario: Home Page Functionality

No	Test Case	Value (Input/Action)	Expected Result	Actual Result
1	Access home page	Open system URL	Home Page loads with sidebar navigation and header	Same as expected result
2	Use sidebar navigation	Click on "Lessons" in sidebar	Redirected to Lessons Page	Same as expected result
3	Toggle theme	Click on theme toggle button	UI changes from Dark Mode → Light Mode (and vice versa)	Same as expected result
4	Use "Start Quest" button	Click on "Start Quest"	Redirected to first lesson page ("Introduction to SQL")	Same as expected result
5	View home page before login	Not logged in	Sidebar and Start Quest available, but no personalized stats shown	Same as expected result
6	View home page after login	Login successfully, return to Home Page	User statistics panel appears (username, XP, rank, progress, Learn/Challenge/Compete buttons)	Same as expected result
7	Check Top Players area	Logged-in user views Home Page	Leaderboard snippet with top players displayed	Same as expected result

Table 6.1 Home Page Functionality

6.1.2 Test Scenario: Register Page Functionality

No	Test Case	Value (Input/Action)	Expected Result	Actual Result
1	Register with valid details	Username: `example_user` Email: `example@gmail.com` Password: `Valid123`	User registered successfully, redirected to confirmation page, and data saved in database	Same as expected result
2	Username too short	Username: `ab` Email: `valid@gmail.com` Password: `Valid123`	Error: "Username must be between 3 and 20 characters"	Same as expected result
3	Username too long	Username: `averyveryverylongusername` Email: `valid@gmail.com` Password: `Valid123`	Error: "Username must be between 3 and 20 characters"	Same as expected result
4	Invalid email format	Username: `validuser` Email: `invalidemail` Password: `Valid123`	Error: "Please enter a valid email address"	Same as expected result
5	Password too short	Username: `validuser` Email: `valid@gmail.com` Password: `123`	Error: "Password must be at least 8 characters long and contain uppercase, lowercase, and numbers"	Same as expected result

No	Test Case	Value (Input/Action)	Expected Result	Actual Result
6	Password missing uppercase	Username: `validuser` Email: `valid@gmail.com` Password: `valid1234`	Error: "Password must contain at least one uppercase letter"	Same as expected result
7	Password missing number	Username: `validuser` Email: `valid@gmail.com` Password: `ValidPass`	Error: "Password must contain at least one number"	Same as expected result
8	Submit empty form	Username: (Blank) Email: (Blank) Password: (Blank)	Error: "All fields are required"	Same as expected result
9	Duplicate email registration	Username: `newuser` Email: `example@gmail.com` Password: `Valid123`	Error: "Email already exists"	Same as expected result
10	Database storage validation	Register new user with valid details	User data saved in MySQL table `users` with hashed password and tracking fields (XP, progress, timestamps)	Same as expected result

Table 6.2 Register Page Functionality

6.1.3 Test Scenario: Login Page Functionality

No	Test Case	Value (Input/Action)	Expected Result	Actual Result
1	Login with valid credentials	Email: `valid@gmail.com` Password: `Valid123`	User successfully logged in, redirected to dashboard	Same as expected result
2	Invalid email format	Email: `invalidemail` Password: `Valid123`	Error: "Please enter an '@' in the email address"	Same as expected result
3	Empty email field	Email: (Blank) Password: `Valid123`	Error: "Please fill out this field" (under email input)	Same as expected result
4	Empty password field	Email: `valid@gmail.com` Password: (Blank)	Error: "Please fill out this field" (under password input)	Same as expected result
5	Both fields empty	Email: (Blank) Password: (Blank)	Error messages shown for both fields: "Please fill out this field"	Same as expected result
6	Non-existent user	Email: `notfound@gmail.com` Password: `Valid123`	Error: "User not found"	Same as expected result
7	Wrong password for valid user	Email: `valid@gmail.com` Password: `WrongPass`	Error: "Incorrect password"	Same as expected result
8	Case sensitivity check	Email: `VALID@GMAIL.COM` Password: `Valid123`	Depending on system config, either login success (case-insensitive email) or error (if case-sensitive)	Same as expected result

No	Test Case	Value (Input/Action)	Expected Result	Actual Result
9	SQL injection attempt	Email: `valid@gmail.com' OR '1'='1` Password: `anything`	Input sanitized, error message returned, login prevented	Same as expected result
10	Theme toggle test	Click on theme switcher	UI switches between light and dark mode	Same as expected result

Table 6.3 Login Page Functionality

6.1.4 Test Scenario: Lesson Page Functionality

No	Test Case	Value (Input/Action)	Expected Result	Actual Result
1	Access lesson page as guest	Open **Lessons → Introduction to SQL** without login	Lesson content (theory, examples, quiz) is displayed, but no **Bookmark** or **Mark Complete** button visible	Same as expected
2	Access lesson page as logged-in user	Login → Open **Lessons → Introduction to SQL**	Lesson content displayed **with Bookmark + Mark Lesson Complete** buttons	Same as expected
3	Bookmark a lesson	Logged-in user clicks **Bookmark** button	Lesson is saved to user's profile, confirmation shown	Same as expected
4	Remove a bookmark	Logged-in user clicks **Remove Bookmark** button	Lesson removed from bookmark list, button label changes	Same as expected
5	Run valid SQL query in editor	Input: `SELECT * FROM customers;`	System executes query and shows formatted table results	Same as expected
6	Run invalid SQL query	Input: `SELCT name FROM customers;`	Error message shown with feedback	Same as expected
7	Run partially correct SQL query	Input: `SELECT names, cities FROM customers;`	Feedback: "Check column names and syntax again"	Same as expected
8	Run correct SQL query	Input: `SELECT name, city FROM customers;`	Success message + table results displayed	Same as expected
9	Attempt quiz question	Select incorrect	System marks red +	Same as

No	Test Case	Value (Input/Action)	Expected Result	Actual Result
	(wrong answer)	option	provides explanation	expected
10	Attempt quiz question (correct answer)	Select correct SQL query	Confirmation "Correct! Quiz completed!" with explanation shown	Same as expected
11	Mark lesson complete	Logged-in user clicks **Mark Complete**	Lesson marked as completed, stored in DB, progress updated	Same as expected
12	Guest user attempts to bookmark/mark complete	Guest clicks lesson page	No option to bookmark or mark complete (restricted to users only)	Same as expected

Table 6.4 Lesson Page Functionality

6.1.5 Test Scenario: Real-World Project Page Functionality

No	Test Case	Input/Action	Expected Result	Actual Result
1	Access Real-World Project Explorer	Guest or logged-in user clicks **Real-World Projects**	System displays list of modules (CRM, HR, Finance, Inventory, Supplier) with short descriptions and **View Example** links	Same as expected
2	Open a project example page	Click **View Example** for one module (e.g., Inventory Tracker)	Full Example Page appears with Use Case, Sample Queries, ERD access, and Mock Database Viewer	Same as expected
3	View Use Case details	Select **Use Case** tab	Business processes (stock control, reordering, supplier mgmt., reporting, etc.) displayed with explanations	Same as expected
4	View Sample Queries	Select **Sample Queries** tab	Predefined SQL examples (e.g., CREATE TABLE, SELECT, constraints) displayed with copy option	Same as expected
5	Access ERD diagram	Click **ERD Access** link	ERD diagram opens in new tab/window showing table relationships	Same as expected
6	Explore full ERD visualization	Navigate inside ERD view	System displays full ERD with tables (Products, Categories, Inventory, Suppliers, etc.) and their relationships	Same as expected
7	Open Mock Database Viewer (single table)	Select Inventory table in viewer	Table displayed with columns (Inventory_ID, Product_ID, Quantity, etc.) and sample data	Same as expected
8	View relational data in	Select multiple	Relationships (Products →	Same as

No	Test Case	Input/Action	Expected Result	Actual Result
	Mock Database Viewer	related tables	Inventory → Stock Transactions → Reorder Rules) displayed with sample data	expected
9	Guest vs. logged-in access	Access page without login	Both guests and logged-in users can view modules and content, no restrictions	Same as expected

Table 6.5 Real-World Project Page Functionality

6.1.6 Test Scenario: Chat Assistant (AI Tutor) Page Functionality

No	Test Case	Input/Action	Expected Result	Actual Result
1	Open Chat Assistant Page	Click on **Chat Assistant** from sidebar	System loads chat interface with input box, suggested prompts, and categories	Same as expected
2	Enter valid SQL-related query	User types: *"How do I write a SELECT statement?"*	ChatGPT API responds with example SQL syntax and explanation	Same as expected
3	Request example SQL code	User enters: *"Give me example of SQL SELECT code"*	ChatGPT API generates query (e.g., SELECT name, email FROM employees WHERE …) with explanation of clauses	Same as expected
4	Submit irrelevant question	User enters: *"Do you know how to race cars?"*	Assistant detects off-topic and replies politely that it is out of scope, suggesting SQL-related queries	Same as expected
5	Handle invalid/empty input	User submits blank input or gibberish	System returns validation message ("Please enter a valid question") without crashing	Same as expected
6	Display suggested prompts	On page load	Suggested prompts such as *"How do I write a SELECT?"* are visible for user quick start	Same as expected
7	Check learning support categories	Scroll down to categories (Query Debugging, SQL Explanations, Best Practices, Learning Support)	Categories displayed, each expandable with examples	Same as expected
8	Response formatting	Submit any SQL query question	Answer returned in structured format: code snippet + plain explanation	Same as expected
9	Guest vs.	Open Chat Assistant	Both guest and logged-in users can access	Same as

No	Test Case	Input/Action	Expected Result	Actual Result
	logged-in access	without login	AI tutor equally (no restriction)	expected

Table 6.6 Chat Assistant (AI Tutor) Page Functionality

6.1.7 Test Scenario: Challenge Page Functionality

No	Test Case	Input/Action	Expected Result	Actual Result
1	Open Challenge Page	User clicks on **Challenges** in sidebar	Challenge Sets Overview (Beginner, Intermediate, Relational, Database Management) displayed with XP/Badge info	Same as expected
2	Select challenge set	User clicks **Beginner Challenge Set 1: SQL Basics**	System navigates to challenge level selection page	Same as expected
3	Level accessibility	Try to select **Medium** or **Hard** before completing Easy	Medium/Hard are locked, only Easy available	Same as expected
4	Start Easy challenge	Click on **Easy**	System loads first challenge question interface	Same as expected
5	MCQ Question attempt	User selects correct answer	Correct option highlighted in green, +5XP awarded, progress bar updated	Same as expected
6	Incorrect answer attempt	User selects wrong option (e.g., GET instead of SELECT)	Option highlighted in red, feedback shown “Incorrect. You have 1 more try”	Same as expected
7	Retry incorrect question	User selects correct answer on second try	Progress continues, XP awarded, next challenge auto-loads	Same as expected
8	Complete Easy level	Finish all Easy challenges	System displays chapter completion screen, +25XP awarded, Bronze1 badge unlocked	Same as expected
9	Path unlocks logic	After Easy completion, go	Medium path unlocked, Hard still locked	Same as expected

No	Test Case	Input/Action	Expected Result	Actual Result
		back to level selection		
10	Unlock Hard level	Complete Medium challenges	Hard path unlocked, now selectable	Same as expected
11	Progress consistency	Check XP and badges on Profile after completing challenges	Updated XP, badge, and challenge completion reflected in profile and leaderboard	Same as expected

Table 6.7 Challenge Page Functionality

6.1.8 Test Scenario: Leaderboard Page Functionality

No	Test Case	Input/Action	Expected Result	Actual Result
1	Open Leaderboard Page	User clicks **Leaderboard** in sidebar	Leaderboard page loads showing ranking table	Same as expected
2	Display top players	System fetches players' XP & ranking info	Top players displayed with columns: Rank, Badge, Username, Total XP, Avg Hard-Level Time	Same as expected
3	Ranking by XP	Compare players with different XP values	Players with **higher XP** always ranked above lower XP	Same as expected
4	Ranking tiebreaker	Two players have equal XP	Players with **lower average Hard-level completion time** ranked higher	Same as expected
5	Badge display	Players with XP milestones reached (e.g., Bronze, Silver, Gold)	Correct badge shown beside username	Same as expected
6	Guest user access	Open leaderboard without login	Leaderboard data still visible, but no personal highlight	Same as expected
7	Logged-in user access	Open leaderboard while logged in	User's row highlighted for easy tracking	Same as expected
8	Dynamic updates	Another player completes challenges & earns XP	Leaderboard refreshes, ranks recalculated instantly	Same as expected
9	Hard-level influence	Players with no Hard-level attempts	Avg Hard-level time shows "—" or blank, rank still based on XP only	Same as expected
10	Data integrity	Database connection lost while fetching	System displays fallback message: "Unable to load leaderboard"	Same as expected

Table 6.8 Leaderboard Page Functionality

6.1.9 Test Scenario: Profile Page Functionality

No	Test Case	Input/Action	Expected Result	Actual Result
1	Open Profile Page	Logged-in user clicks **Profile** in sidebar	Profile page loads showing username, XP, and current rank (Bronze/Silver/Gold)	Same as expected
2	Display XP & Rank	User with certain XP (e.g., 150 XP)	Correct rank badge (e.g., Bronze) displayed according to XP threshold	Same as expected
3	Learning Progress Bars	User has completed part of modules	Progress bars show correct percentage for Beginner, Intermediate, Relational Concepts, and Database Management	Same as expected
4	No progress yet	New user with 0 completed lessons	All progress bars show 0%	Same as expected
5	Bookmark Lessons	User bookmarks for a lesson	Lesson title appears in Bookmarked Lessons panel	Same as expected
6	Remove Bookmark	User removes a bookmarked lesson	Lessons disappear from Bookmarked Lessons panel	Same as expected
7	Completed Lessons Tracking	User finishes a lesson	Lesson title appears in Completed Lessons panel	Same as expected
8	Multiple Completed Lessons	User completes multiple lessons	All finished lessons appear in Completed Lessons list with correct titles	Same as expected
9	Guest access	Guest tries to open Profile Page	System redirects to login/register page	Same as expected
10	Data update sync	User completes a new lesson or earns XP	Profile data updates immediately upon refresh (XP, rank, progress bar)	Same as expected

Table 6.9 Profile Page Functionality

6.2 Project Challenges

There were several challenges encountered during the creation of SQL Quest, which required consideration. One of the challenges was database consistency between all mock datasets (CRM, HR, Finance, Inventory and Supplier). Keeping foreign key relationships straight and preventing query execution errors required constant testing and schema iterations. Another challenge was real-time query execution, as the platform had to securely run SQL without putting data at risk. It took several iterations to find a user-friendly way of processing error messages while at the same time making sure that they led learners properly.

User authentication and security were also difficult, particularly hashing of passwords, input validation and unauthorized access. It was further complicated due to the request for a gamified solution, rendering tracking of progress, calculating XP, achieving badges and ranking leaderboards – without being detrimental to the performance. Finally building out a futuristic, responsive UI that performed well on both desktop and mobile involved the careful balancing of heavyweight features such as interactive in browser queries with a desire to remain optimized. These obstacles in the end supported the further development of the system: it grew by debugging, testing and gradual improvements.

6.3 Objectives Evaluation

The goals of SQL Quest were to create a gamified, interactive learning environment for SQL that incorporated tutorials, practice-based challenges, real-world data sets and AI tutors. These goals have been accomplished in practice:

- Interactive Learning: The package comes with Lessons, Guided Practice sessions and intervened Quizzes for step-by-step learning.
- Real-Time Practice: The included query editor enables learners to execute SQL commands on simulated data sets & solidify their hands-on query writing abilities.
- Gamification: XP, badges & gradual unlocking of challenges to keep you motivated to learn more.
- Real-World Context: System modules (e.g., CRM, HR, Finance, Inventory and Supplier) provide application frameworks for deploying various real-world database structures used in industrial contexts.
- Leaderboard & Challenge: Ranking by total XP and completed times at hard level brings out competition spirit.
- AI Chat Assistant: Built-in ChatGPT tutor gives immediate feedback, explanations and help.

In general, SQL Quest fulfils its objectives by providing a mix of interactive, competitive and applied way of learning SQL.

6.4 Concluding Remark

SQL Quest illustrates the successful integration of gamification with real-world application to offer an AI-inferential approach, and it provides potential benefits for database learning. The platform mixes lessons, quizzes, practice challenges and real datasets for projects to offer students both theoretical knowledge as well as practical skills in an industry-like environment. The extensibility hinderances, such as database incorporation, UI enhancement and security problems that were overcome during development, culminated in a stable and easy-to-use solution.

In summary, SQL Quest achieves its goal of making the learning of SQL easier, more interactive and entertaining. It offers a starting point to add more datasets, complex query simulations and improved AI-driven tutoring capabilities. The system is not only an academic success but a technology to be used by academics and professionals for the benefit of learners.

Chapter 7: Conclusion and Recommendation

7.1 Conclusion

The SQL Quest project has proven that applying diverse gamification, live query execution results and AI-assistant features can lead to a successful learning platform for the SQL language. Developed with the intention to convert theoretical knowledge in SQL into an application, the system consists of a few components, including structured lessons, practice queries, challenges, real-world datasets and a leaderboard competition. With XP points, badges, and progressively unlocked challenge levels, SQL Quest encourages students to interact with it and fosters a rewarding environment which makes learning fun. We added the average Hard completion time as a tiebreaker, so rankings would feature both stamina and efficiency, making the competitive model more equitable. Another major contribution of the system is in its Chat Assistant, which uses ChatGPT API to give immediate SQL explanations, debugging hints and personalised recommendations to learners like you will be taught by a tutor.

They faced several technical and design challenges when developing the system, such as database integration, authentication, query processing and front-end optimisation. They were systematically addressed by schema refactoring, increased security and performance tuning, improving the stability of the platform and scalability. Consequently, SQL Quest was successful in reaching academic learning targets as well as becoming a functional, interactive teaching/learning game that motivates and supports non-registered and registered players to gain proficiency in SQL. As a result, the project has achieved its initial objectives: to provide an accessible set of SQL lessons, interactive query execution, personalised dashboards and gamified progress tracking, with potential application beyond entertainment software within the industry.

7.2 Recommendation

While SQL Quest has achieved its initial goals, there is potential for extending and enriching it to have a larger reach and a more engaging experience. The tool would be improved by increasing dataset coverage to cover a greater diversity of industries and thus exposing learners to broader business uses of SQL. Even though the AI Chat Assistant is now effective, with context awareness and push-to-optimisation suggestions at higher levels of sophistication, it can become an ever more personalised coach. Technically, however, migrating the backend and database to a cloud-based solution would have enabled much more scalability and security (for example, supporting concurrent multiple users). Some added bells and whistles, like timed tests, digital credential verification, and collaborative team challenges, would also increase the platform's cache with users while maintaining learner engagement.

UI updates such as more general accessibility additions for different learner requirements (that are different from those of the average user), yet more mobile optimisation to make it easier to use on traditional devices in a wider community. Finally, ubiquitous feedback and analytics would enable learners and teachers to track progress in a structured manner, whilst providing valuable data for system enhancement. If implemented, these recommendations will move SQL Quest from its status as a solid foundation for learning about SQL to an advanced and scalable system for instruction that can cater not only for individual learners but also to academic institutions or professional training programs

REFERENCES

[1] Statista, "Relational Database Market Share 2023," Statista, 2023. [Online]. Available: https://www.statista.com/statistics/264810/worldwide-market-share-of- database-management-systems/.

[2] Dataversity, "Database Management Best Practices," Dataversity, 2023. [Online]. Available: https://www.dataversity.net/database-management-best-practices/.

[3] Simplilearn, "The Future of SQL: Exploring its Role in Big Data," Simplilearn, 2023. [Online]. Available: https://www.simplilearn.com/future-of-sql-article#:~:text=As%20big%20data%20grows%2C%20SQL,massive%20datasets%20s tored%20in%20Hadoop.

[4] Lonti, "Utilizing SQL in Big Data Solutions," Lonti, 2023. [Online]. Available: https://www.lonti.com/blog/utilizing-sql-in-big-data-solutions.

[5] MySCAL, "Key Reasons SQL is Vital for Relational Databases," MySCAL, 2023. [Online]. Available: https://www.myscal.com/blog/key-reasons-sql-vital-relational- databases/.

[6] Codecademy, "SQL Course," Codecademy, 2023. [Online]. Available: https://www.codecademy.com/learn/learn-sql.

[7] PCMag, "Codecademy Review," PCMag, 2023. [Online]. Available: https://www.pcmag.com/picks/the-best-online-learning-platforms-for-2021.

[8] Khan Academy, "Intro to SQL," Khan Academy, 2023. [Online]. Available: https://www.khanacademy.org/computing/computer-programming/sql.

[9] Common Sense Education, "Khan Academy Review," Common Sense Education, 2023. [Online]. Available: https://www.commonsense.org/education/website/khan- academy.

[10] W3Schools, "SQL Tutorial," W3Schools, 2023. [Online]. Available: https://www.w3schools.com/sql/.

[11] Guru99, "W3Schools Review," Guru99, 2023.

[Online]. Available: https://www.guru99.com/w3schools-tutorial.html.

[12] LeetCode, "SQL Practice," LeetCode, 2023.
[Online]. Available:
https://leetcode.com/problemset/database/.

[13] Coding Ninjas, "LeetCode Review: A Comparison Guide for Programming Interview Preparation," Coding Ninjas, 2021.
[Online]. Available:

https://www.codingninjas.com/blog/2021/12/02/leetcode-vs-interviewbit-a- comparison-guide-for-programming-interview-preparation/.

[14] SQLZoo, "SQLZoo Tutorial," SQLZoo, 2023.
[Online]. Available: https://sqlzoo.net/.

[15] LearnSQL, "SQLZoo Review," LearnSQL, 2023.
[Online]. Available: https://learnsql.com/blog/sqlzoo-com-review/.

APPENDIX

Poster