

Cardiff Metropolitan University

Cardiff School of Technology

BSc Software Engineering

[Title]

Submitted in [Month]

[Year]

By

[Student Name]

(Student ID – Cardiff ID – ICBT ID)

This dissertation is submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science in Software Engineering (BSc SE)

##### DECLARATION

This work is being submitted in partial fulfillment of the requirements for the degree of

…………………………………………………………………………………………. and has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed - ............................................................ (Candidate) Date - ...................................

##### SUPERVISOR’S DECLARATION STATEMENT

Student Name – Supervisor’s Name -

I acknowledge that the above named student has regularly attended the meeting, and actively engaged in the dissertation supervision process.

Signed - ……………………………………………. (Supervisor) Date - …………………......

# Acknowledgement

I would first like to express my deepest gratitude to my parents, who have been my greatest source of strength, encouragement, and unconditional love throughout this journey. Their constant guidance, motivation, and belief in my abilities gave me the courage to overcome challenges and keep moving forward. From the very beginning, they have provided me with every possible opportunity to grow both academically and personally. Without their sacrifices, patience, and unwavering support, this project would not have been possible. I am truly blessed to have such parents who inspire me to achieve my goals and pursue my dreams with determination.

I would also like to extend my heartfelt appreciation to my teachers, whose dedication and commitment to sharing knowledge have played a crucial role in shaping my academic path. Each of them, in their own way, has contributed immensely to my understanding of the concepts that form the foundation of this project. Their lessons, guidance, and advice have been invaluable not only in the completion of this work but also in developing my overall perspective as a student. Their encouragement to think critically, ask questions, and challenge myself academically has been a constant source of motivation.

My sincere thanks also go to my friends, who stood by me during this journey. Their companionship, encouragement, and willingness to share ideas and provide feedback have been extremely valuable. Working late nights, discussing new ideas, or simply being there to motivate me when things became challenging has made this experience much more meaningful. They reminded me of the importance of teamwork, collaboration, and mutual support. Their presence made the process less stressful and more enjoyable, and for that, I will always be grateful.

Most importantly, I would like to express my special and profound gratitude to my supervisor. Their constant support, constructive feedback, and invaluable advice have been instrumental in the successful completion of this project. From the initial stages of developing the concept to the final execution, my supervisor’s guidance has been a guiding light. Their patience in addressing my queries, attention to detail, and encouragement to think beyond conventional approaches helped me to refine and improve the quality of my work. I truly appreciate the time, effort, and expertise they have invested in guiding me through this academic endeavor. Their mentorship has not only enhanced the outcome of this project but has also significantly contributed to my personal and professional growth.

Finally, I would like to acknowledge everyone who, in one way or another, contributed to the success of this project but whose names may not be mentioned here. Whether through words of encouragement, technical assistance, or moral support, their contributions have not gone unnoticed. Every bit of help, no matter how small, has been valuable in ensuring the successful completion of this work. To all of you, I extend my sincere gratitude.

# Abstract

Table of Contents

[Acknowledgement 3](#_Toc207492131)

[Abstract 4](#_Toc207492132)

[List of Figures 8](#_Toc207492133)

[List of Tables 9](#_Toc207492134)

[List of Abbreviations 10](#_Toc207492135)

[1 Introduction 11](#_Toc207492136)

[1.1 Background Studies 11](#_Toc207492137)

[1.2 Problem Statement 12](#_Toc207492138)

[1.3 Objective 13](#_Toc207492139)

[1.4 Solutions 15](#_Toc207492140)

[2. Literature Review 19](#_Toc207492141)

[2.1 Gamification in Learning 19](#_Toc207492142)

[2.1.1 Adaptive Difficulty in Game-Based Learning 21](#_Toc207492143)

[2.1.2 Comparison with existing SQL learning platforms 22](#_Toc207492144)

[2.1.3 Conclusion 22](#_Toc207492145)

[2.2 Game Life Cycle Models 23](#_Toc207492146)

[2.2.1 Game Lifecycle in Educational Games 23](#_Toc207492147)

[2.2.2 GDLC Stages 24](#_Toc207492148)

[2.2.3 Spiral Model 25](#_Toc207492149)

[2.2.4 Agile Game Development 26](#_Toc207492150)

[2.2.5 MDA Framework (Mechanics, Dynamics, Aesthetics) 27](#_Toc207492151)

[2.2.6 ADDIE-GDLC Hybrid 28](#_Toc207492152)

[2.2.7 Adaptive Frameworks 29](#_Toc207492153)

[2.3 Why ADDIE-GDLC Hybrid suite for my project 30](#_Toc207492154)

[3 Planning 32](#_Toc207492155)

[3.1 Feasibility Report 32](#_Toc207492156)

[3.1.1 Introduction 32](#_Toc207492157)

[3.1.2 Objectives 32](#_Toc207492158)

[3.2 Technical Feasibility 32](#_Toc207492159)

[3.2.1 Technology Stack 32](#_Toc207492160)

[3.2.2 Integration of Systems 33](#_Toc207492161)

[3.3 Operational Feasibility 33](#_Toc207492162)

[3.3.1 Resource Requirements 33](#_Toc207492163)

[3.3.2 Stakeholders 34](#_Toc207492164)

[3.3.3 Operational Workflow 34](#_Toc207492165)

[3.4 Economic Feasibility 34](#_Toc207492166)

[3.4.1 Cost Analysis 34](#_Toc207492167)

[3.4.2 Revenue Model 35](#_Toc207492168)

[3.4.3 Profitability 35](#_Toc207492169)

[3.5 Legal Feasibility 35](#_Toc207492170)

[3.5.1 Regulatory Compliance 35](#_Toc207492171)

[3.5.2 Intellectual Property 35](#_Toc207492172)

[3.5.3 Educational Compliance 35](#_Toc207492173)

[3.6 Risk Assessment 36](#_Toc207492174)

[3.7 SWOT 38](#_Toc207492175)

[3.8 PESTAL Analysis 40](#_Toc207492176)

[3.9 Life Cycle Model 42](#_Toc207492177)

[3.9.1 Benefits of Using the ADDIE–GDLC Hybrid Model for SQL Safari 42](#_Toc207492178)

[3.9.2 Overview of ADDIE–GDLC Hybrid Stages 42](#_Toc207492179)

[3.10 Time Plan 44](#_Toc207492180)

[4.0 Requirement Gathering and Analysis 45](#_Toc207492181)

[4.1 Requirement Gathering Technique Used for the Project 45](#_Toc207492182)

[4.2 Questionnaire 47](#_Toc207492183)

[4.3 Interview 50](#_Toc207492184)

[4.3.1 Summary of Interview Insights 51](#_Toc207492185)

[4.4 Functional and Non-Functional Requirements 52](#_Toc207492186)

[4.4.1 Functional Requirements 52](#_Toc207492187)

[4.4.2 Non-functional Requirement 55](#_Toc207492188)

[5.0 System Design 58](#_Toc207492189)

[5.1 Architecture Diagram 58](#_Toc207492190)

[5.2 ER Diagram 59](#_Toc207492191)

[5.3 UML Diagrams 60](#_Toc207492192)

[5.3.1 Class Diagram 60](#_Toc207492193)

[5.3.2 Use Case Diagram 61](#_Toc207492194)

[5.4 Wireframe Diagram 62](#_Toc207492195)

[6.0 Implementation 63](#_Toc207492196)

[6.1 Technology Stack 63](#_Toc207492197)

[6.2 Design Patterns 64](#_Toc207492198)

[6.3 Implementation of the Program 68](#_Toc207492199)

[6.3.1 Game Interfaces 68](#_Toc207492200)

[6.3.2 Province-Level Game Flow 73](#_Toc207492201)

[6.3.3 Example Scene: Jaffna – The Data Pairing Puzzle 74](#_Toc207492202)

[6.3.4 Rewards & Gamification 74](#_Toc207492203)

[6.3.5 Evaluation & Continuous Improvement 74](#_Toc207492204)

[7.0 Testing and Validation 75](#_Toc207492205)

[7.1 Test Plan 75](#_Toc207492206)

[7.2 Test Cases 76](#_Toc207492207)

[8.0 Conclusion 78](#_Toc207492208)

[8.1 Conclusion 78](#_Toc207492209)

[8.2 Future Recommendations 79](#_Toc207492210)

[8.3 Lessons Learned 81](#_Toc207492211)

[Gantt Chart 83](#_Toc207492212)

[References 83](#_Toc207492213)

[Appendix 1 86](#_Toc207492214)

[Questionnaire and answers 86](#_Toc207492215)

[Source Codes 86](#_Toc207492216)

[Appendix 2 87](#_Toc207492217)

[Supervisor meeting log sheets 87](#_Toc207492218)

**Supervision meeting log sheets**

# List of Figures

# List of Tables

# List of Abbreviations

# 1 Introduction

## 1.1 Background Studies

Learning SQL is often challenging for students, especially when faced with advanced concepts such as joins, nested queries, and normalization. Traditional teaching methods mainly rely on lectures and static exercises, which fail to provide sufficient interactivity and practical application. As a result, many learners struggle to connect theoretical knowledge with real-world problem solving, leading to low motivation and limited skill development.

In recent years, gamified learning platforms have emerged as an alternative to traditional instruction, offering points, badges, and leaderboards to increase engagement. However, most existing SQL games are limited in scope and provide only fixed sets of challenges that do not adapt to individual learners. They often focus more on entertainment than pedagogy, which means students may enjoy the activity but fail to gain a deep understanding of SQL concepts. Furthermore, these platforms generally lack integration with structured learning frameworks that guide students progressively from beginner to advanced topics.

The proposed system, **SQL Safari**, addresses these limitations by combining learning mechanisms with game-based elements. Unlike existing systems, it personalizes the difficulty level according to learner performance, ensuring that both novices and advanced students remain challenged at an appropriate level. Additionally, SQL Safari incorporates narrative-driven tasks, real-time feedback, and achievement-based progression that connects gameplay with actual learning outcomes. By blending pedagogy with engaging game mechanics, the system offers a scalable, immersive, and effective approach to mastering SQL, making it distinct from other gamified learning platforms.

The SQL Safari game is designed to maximize student engagement by combining interactive gameplay with meaningful learning experiences. Instead of passively listening to lectures or solving repetitive exercises, learners actively participate in challenges that are embedded within a storyline, earning points, badges, and rewards as they progress. The use of leaderboards encourages healthy competition, while gamified learning ensures that students remain neither bored by tasks that are too easy nor frustrated by those that are too difficult. By integrating immediate feedback and visually appealing interfaces, the game keeps learners motivated and immersed, transforming the process of mastering SQL into an enjoyable and goal-driven adventure.

## 1.2 Problem Statement

Structured Query Language (SQL) is a fundamental skill in computer science and data management, but many students encounter significant difficulties in understanding and applying its concepts effectively. Traditional teaching approaches, which rely heavily on lectures and static exercises, often fail to provide sufficient engagement or practical context. While gamified SQL platforms exist, they are often limited in scope and fail to deliver a fully integrated, interactive, and educational experience. These shortcomings create several challenges for learners:

1. **Difficulty in Understanding Complex SQL Concepts:** Students frequently struggle with abstract topics such as joins, nested queries, triggers, and normalization. Conventional learning methods do not provide enough interactive practice, leaving learners unprepared to apply SQL in real-world scenarios.
2. **Limited Engagement in Existing SQL Platforms:** Current gamified SQL systems focus heavily on entertainment aspects but lack mechanisms that sustain deep engagement. As a result, learners may play initially but quickly lose interest when tasks become repetitive or disconnected from meaningful outcomes.
3. **Absence of Narrative-Based Learning:** Most existing platforms do not integrate a storyline or thematic context, making the learning process monotonous. Without a narrative or adventure element, students may perceive SQL learning as dry and uninteresting.
4. **Lack of Progressive Learning Pathways:** Available tools often present challenges in isolation, without a structured flow that takes learners from beginner-level concepts to advanced problem solving. This absence of guided progression prevents students from building strong foundational knowledge.
5. **Minimal Real-Time Feedback:** Learners using current systems often receive only generic correctness checks rather than detailed explanations of their mistakes. Without targeted feedback, students struggle to correct errors and improve their SQL skills.
6. **Limited Customization of Tasks:** Most platforms offer fixed sets of exercises, which do not align with individual learner needs or academic syllabi. This restricts their usefulness for both self-learners and institutions aiming to adopt such platforms for coursework.
7. **Weak Connection to Real-World Applications:** Current SQL learning games rarely simulate real-world database problems or scenarios, leaving learners unprepared for practical, professional use of SQL. This limits the transferability of skills learned through the platform.
8. **Lack of Social and Collaborative Learning Features:** Although leaderboards exist in some systems, opportunities for peer-to-peer interaction, teamwork, or collaborative challenges are largely absent. This reduces the sense of community and motivation to learn.
9. **Overemphasis on Competition Without Pedagogy:** While leaderboards and badges are included, many platforms focus only on superficial competition. Without meaningful educational alignment, students may earn points without gaining a true understanding of SQL concepts.
10. **Accessibility and Scalability Issues:** Several SQL learning platforms require high technical setup or are not optimized for wider audiences, making them inaccessible to beginners, schools, or learners in developing regions who require a lightweight, scalable solution.

The proposed solution, **SQL Safari**, addresses these limitations by integrating structured learning progression, narrative-driven gameplay, real-time feedback, and engaging social elements. By balancing game mechanics with strong pedagogical design, the platform aims to enhance motivation, improve conceptual understanding, and provide a scalable, interactive environment for mastering SQL.

## 1.3 Objective

1. **Develop an Interactive SQL Learning Environment**

* Design a gamified platform where students can practice SQL through challenges, puzzles, and story-driven tasks.
* Provide a safe and controlled environment where learners can test queries without fear of damaging real databases.
* Ensure the platform supports both fundamental and advanced SQL concepts.

1. **Implement Real-Time Feedback and Explanations**

* Provide instant feedback on submitted queries, highlighting mistakes and suggesting improvements.
* Offer detailed explanations for incorrect answers to support conceptual understanding.
* Track learner progress and highlight areas requiring additional practice.

1. **Introduce a Narrative-Driven Adventure System**

* Integrate a storyline (e.g., “SQL Safari”) to guide learners through different levels and challenges.
* Unlock new scenarios and rewards as student complete tasks, maintaining motivation.
* Blend educational tasks with storytelling to make SQL practice enjoyable and immersive.

1. **Incorporate Game Mechanics to Boost Motivation**

* Use points, badges, and leaderboards to encourage consistent participation.
* Design achievements that celebrate milestones such as mastering joins or completing a level.
* Ensure competition remains healthy by balancing individual progress with peer comparison.

1. **Ensure Structured Learning Progression**

* Organize challenges in a logical sequence from beginner to advanced SQL topics.
* Align content with academic curricula and industry requirements for better applicability.
* Allow learners to revisit previous levels to strengthen weak areas.

1. **Simulate Real-World Database Scenarios**

* Provide practical tasks based on realistic datasets (e.g., hotel bookings, e-commerce, university records).
* Show students how SQL applies in professional contexts, bridging theory and practice.
* Encourage problem-solving and critical thinking beyond simple query writing.

## Solutions

1. Interactive SQL Learning Platform

**Solution:**  
Develop a gamified web-based platform where learners can practice SQL through engaging challenges, puzzles, and levels embedded within a storyline.

**How it works:**

* Students log into the platform and access challenges categorized by SQL topics (SELECT, JOIN, GROUP BY, etc.).
* Each challenge is linked to a simulated database where students can test queries safely.
* Successful completion unlocks new levels in the game’s storyline.

**Impact:**

* Provides a hands-on, risk-free environment for SQL practice.
* Builds learner confidence by connecting query writing to interactive outcomes.

1. Real-Time Query Feedback and Explanations

Solution:

Integrate a query evaluation system that provides immediate, detailed feedback on student submissions.

**How it works:**

* Submitted SQL queries are executed against a test database.
* The system compares results with expected outputs and highlights mistakes.
* Explanations and hints are provided to help learners understand errors.

Impact:

* Reduces repeated mistakes and accelerates conceptual clarity.
* Encourages self-paced learning without needing constant instructor intervention.

1. **Narrative-Driven Adventure Mode**

Solution:

Embed learning tasks within a storyline called SQL Safari to maintain motivation and immersion.

How it works:

* Each level represents a chapter in the safari journey (e.g., “Crossing the River with Joins”).
* Completing challenges unlocks new adventures, characters, or rewards.

Impact:

* Transforms SQL practice into an engaging experience rather than repetitive exercises.
* Maintains learner interest through a sense of exploration and achievement.

1. Gamification Features for Motivation

Solution:

Use points, badges, and leaderboards to increase learner engagement and competition.

How it works:

* Points are awarded for solving queries correctly.
* Badges mark milestones like mastering “Aggregations” or finishing “Level 5.”
* Leaderboards allow friendly competition among peers.

Impact:

* Encourages consistent practice and healthy competition.
* Provides recognition that motivates students to achieve more.

1. Structured Learning Progression

Solution:

Organize SQL topics into progressive levels aligned with academic and professional requirements.

How it works:

* Levels start with basics (SELECT, WHERE) and gradually move to advanced concepts (nested queries, triggers).
* Learners can revisit earlier challenges for reinforcement.

Impact:

* Ensures learners build a solid foundation before advancing.
* Provides a clear path from beginner to expert.

1. Practical Real-World Scenarios

Solution:  
Incorporate challenges modeled on real-world database applications.

How it works:

* Datasets mimic e-commerce systems, hotels, universities, and banking scenarios.
* Students solve realistic problems such as generating sales reports or managing reservations.

Impact:

* Bridges the gap between theory and practical application.
* Prepares students for professional database management tasks.

1. Social and Collaborative Learning Features

Solution:  
Promote peer interaction through discussion forums, shared challenges, and group tasks.

**How it works:**

* Students can post queries, solutions, and discuss errors in forums.
* Group challenges encourage teamwork on complex SQL problems.

**Impact:**

* Fosters collaborative learning and peer-to-peer knowledge exchange.
* Builds a learning community around SQL Safari.

1. Instructor and Admin Dashboards

**Solution:**

Provide monitoring and management tools for teachers and administrators.

**How it works:**

* Dashboards display learner progress, completion rates, and performance analytics.
* Instructors can assign specific challenges or levels as coursework.

**Impact:**

* Enables educators to track and guide learner progress.
* Provides valuable insights for curriculum planning.

1. Secure and Accessible Platform

**Solution:**  
Ensure the platform is secure, accessible, and user-friendly for a broad range of learners.

**How it works:**

* Responsive web design ensures compatibility with desktop and mobile.
* Secure login and encrypted data handling protect user information.

**Impact:**

* Increases accessibility for learners in different environments.
* Builds trust by safeguarding user data.

1. Scalable and Sustainable Architecture

Solution:  
Design the platform for long-term growth and continuous improvement.

**How it works:**

* Cloud-based infrastructure ensures the platform can support many users simultaneously.
* Modular design allows for adding new SQL concepts, datasets, and game storylines.

**Impact:**

* Ensures the platform remains relevant as SQL evolves.
* Provides sustainability and adaptability for future expansion.

# 2. Literature Review

## 2.1 Gamification in Learning

According to Seaborn and Fels (2015), gamification in learning involves the integration of game elements—such as points, badges, and leaderboards—into educational environments to enhance user motivation and engagement. The concept, while gaining popularity across domains like education and human–computer interaction, still suffers from definitional inconsistencies and underdeveloped theoretical foundations. Despite these challenges, early empirical studies have shown that gamification can positively impact learners by increasing their participation, motivation, and enjoyment, especially when designed using psychological frameworks such as self-determination theory. However, Seaborn and Fels (2015) argue that many existing studies lack strong experimental designs and theoretical coherence, calling for more rigorous, theory-driven research to establish effective and evidence-based practices for gamified learning systems(Seaborn & Fels, 2015).

Both Seaborn and Fels (2015) and the authors of the second study (*Gamification in e-learning platforms*, 2013) explore the role of gamification in enhancing educational outcomes, particularly in web-based learning environments. However, the two papers differ significantly in their approach, focus, and findings. Seaborn and Fels (2015) offer a broad theoretical and empirical literature review, synthesizing diverse sources to examine how gamification is conceptualized and applied across various domains. They emphasize that gamification is still an emerging field with inconsistent definitions and underdeveloped theoretical foundations. While they note its potential for improving user motivation and engagement, they call for more rigorous, theory-driven, and controlled empirical research to validate its impact and effectiveness(Domínguez et al., 2013).

In contrast, the second study (Domínguez et al., 2013) provides a concrete empirical case study of a gamified learning system developed as a plugin for a university e-learning platform. The experiment revealed mixed results: students in the gamified group showed higher motivation and better performance in practical assignments, but they also underperformed in written assessments and participated less in traditional classroom activities. This suggests that gamification may not uniformly improve all aspects of learning, and that it might favor practical, task-oriented activities over reflective or collaborative tasks. In terms of effectiveness, Seaborn and Fels (2015) stress the importance of aligning gamification strategies with motivational theories like Self-Determination Theory, while Domínguez et al. (2013) demonstrate the practical implementation challenges of these theories in real-world educational settings. While Seaborn and Fels provide a comprehensive landscape for understanding gamification conceptually, Domínguez et al. bridge the gap between theory and application by showing what works and what doesn’t in a controlled experiment.

Teaching SQL and databases poses a number of challenges, particularly when attempting to bridge the gap between theoretical knowledge and practical application. As noted by Seyed-Abbassi (1993), one key difficulty lies in helping students move from abstract database concepts to the concrete implementation of SQL queries. This abstract-to-practical transition is often hindered by students’ limited exposure to real-world systems, making it harder for them to visualize how relational concepts like primary keys, referential integrity, and normalization are used in actual business environments. The complexity of query formulation—especially in terms of JOIN operations and nested queries—can create substantial barriers for students, particularly when their foundational understanding of relationships between tables is weak (Seyed-Abbassi, 1993).

Traditional methods of teaching databases often rely heavily on lectures and textbook readings to convey theoretical concepts, which may not provide sufficient opportunity for students to engage with real data or practice hands-on skills. Seyed-Abbassi (1993) argues that this can result in a lack of practical understanding, particularly when it comes to applying Data Definition Language (DDL) and Data Manipulation Language (DML) commands. Without structured exercises or projects, students may struggle to understand why certain constraints are necessary, how normalization improves database design, or how referential integrity rules function during operations like deletions or updates. The static nature of traditional teaching can thus leave students underprepared for real-world tasks that require dynamic problem-solving and system interaction(Seyed-Abbassi, 1993).

In contrast, more modern approaches—such as project-based learning—have shown considerable promise. At the University of North Florida, a semester-long SQL project was introduced to provide a simulated business experience, allowing students to design databases, establish relationships, and write SQL queries collaboratively. According to Seyed-Abbassi (1993), this practical method led to increased student awareness of database structure and enhanced their problem-solving capabilities. Students were required to justify the semantics of their database relationships, apply constraints like cascading deletions, and explore how referential integrity could affect business rules. By presenting their work and evaluating one another, they also developed communication and analytical skills beyond mere syntax memorization(Seyed-Abbassi, 1993).

Interactive, visual, and collaborative tools—such as entity-relationship diagrams and hands-on SQL labs—were critical components of this modern approach. Seyed-Abbassi (1993) reports that using diagrams to visualize entity relationships helped students understand abstract dependencies more concretely. Similarly, group-based presentations reinforced understanding through peer explanation and critique, while also encouraging active engagement. This project-based model allowed students to repeatedly apply what they learned in lectures to build and refine their systems, thus reinforcing both DDL and DML in an iterative manner. The results suggest that when students work with real examples in a structured but exploratory environment, their comprehension of JOINs, constraints, and query complexity improves significantly(Seyed-Abbassi, 1993).

The advantages of serious games in education are implied through the paper’s discussion of the positive impact of GBL on student interest, engagement, and learning outcomes in computational thinking. The use of GBL in CT activities supports the development of high-level thinking skills and enables learners to apply computational concepts in practical and interdisciplinary contexts. For example, GBL activities help foster independent thinking and problem-solving by encouraging students to actively participate in designing programs or managing complex tasks, as seen in various international curricula(Hsu et al., 2018).

Although specific empirical evidence from controlled studies is not detailed, the paper indicates that GBL contributes to improved learner motivation and CT performance, particularly when integrated with project-based and cooperative learning strategies. It also suggests that GBL can facilitate cross-domain learning, allowing students to manage and analyze materials from various disciplines through computational methods(Hsu et al., 2018).

Regarding applications in computer science and coding education, the paper points out that CT and programming skills are often trained through GBL approaches. Many countries incorporate CT courses that use games or game-like activities to teach programming logic and computational problem-solving from early education stages onward. This aligns with the broader trend of embedding CT into K-12 education worldwide, where GBL forms a practical way to support the teaching of programming concepts and digital literacy(Hsu et al., 2018).

### 2.1.1 Adaptive Difficulty in Game-Based Learning

Adaptive difficulty refers to the dynamic modification of a game’s challenge level in response to a player's performance. The objective is to maintain an optimal state of "flow," avoiding both boredom and frustration. Csikszentmihalyi's (1990) flow theory emphasizes this balance as critical to engagement and intrinsic motivation.

Empirical studies have validated the efficacy of adaptive difficulty. Andersen et al. (2013), in their study published in *Computers & Education*, found that adaptive games led to significantly improved student performance and motivation. Similarly, Strieder et al. (2020) demonstrated that adaptive AI agents in educational games increased engagement while maintaining cognitive load. Further, Kickmeier-Rust et al. (2011) introduced the ELEKTRA framework, an adaptive engine designed to personalize learning in educational games.

Mechanisms of adaptive difficulty include monitoring correctness, response time, and learning trajectory. According to Darwesh (2016), such systems may utilize performance thresholds, response latency, and error rates to personalize game flow. This method is particularly effective in educational serious games where learner abilities vary widely. In the context of learning analytics, adaptive systems use real-time data to adjust game mechanics, ensuring tailored learning experiences (Ifenthaler & Yau, 2020).

### 2.1.2 Comparison with existing SQL learning platforms

A range of SQL learning platforms currently exist, each catering to different learning styles and objectives. Narrative-based games like SQL Murder Mystery and SQL Island leverage storytelling to enhance cognitive engagement (Gee, 2003; Egenfeldt-Nielsen, 2006), but their limited scope, static difficulty, and lack of progress tracking reduce long-term learning value. Platforms such as Mode Analytics and W3Schools focus on theory and real-world datasets, yet they lack adaptive mechanisms (Brusilovsky & Millán, 2007) and offer minimal gamification, which may limit motivation and user retention. More structured and gamified tools like DataCamp and Codewars provide points and ranks (Hamari et al., 2014; Papastergiou, 2009), but either require subscriptions or fail to support beginners through foundational learning stages (Anderson & Krathwohl, 2001). Coding challenge platforms like LeetCode and HackerRank are excellent for intermediate to advanced users preparing for interviews, yet offer little scaffolding or personalized progression for novices. Tools such as Kaggle Learn and SQL Fiddle support real-data experimentation but lack motivational structures (Schunk, 2012) or gameful design. In contrast, the proposed *Gamified SQL Learning Platform with Adaptive Difficulty* integrates the best aspects of these tools—story-driven missions, real-world query tasks, progressive difficulty, and immediate feedback—while addressing their limitations through adaptive learning paths, beginner-to-advanced level scaffolding, and embedded gamification features such as XP, levels, and achievements. This design is grounded in educational theories that emphasize meaningful feedback (Schunk, 2012), engagement via gameplay (Hamari et al., 2014), and structured cognitive progression (Anderson & Krathwohl, 2001), positioning the platform as both motivational and pedagogically sound.

### 2.1.3 Conclusion

In summary, the literature reveals that while gamification and game-based learning (GBL) have demonstrated considerable potential in enhancing motivation, engagement, and conceptual understanding in subjects like SQL, existing platforms fall short in key areas such as adaptivity, long-term scaffolding, and holistic cognitive development. Studies by Seaborn & Fels (2015), Domínguez et al. (2013), and Hsu et al. (2018) collectively stress the importance of integrating meaningful feedback, storytelling, and adaptive mechanisms into learning environments. However, tools like SQL Murder Mystery, Mode Analytics, LeetCode, and DataCamp often focus narrowly—either on entertainment, theory, or advanced challenges—without addressing foundational skill development and personalized progression. By adopting a hybrid ADDIE-GDLC development model and leveraging adaptive learning frameworks, the proposed Gamified SQL Learning Platform with Adaptive Difficulty directly responds to these gaps. It combines the immersive power of narrative-based GBL (Gee, 2003), the engagement benefits of gamification (Hamari et al., 2014), and the pedagogical strength of adaptive learning systems (Brusilovsky & Millán, 2007). This makes it uniquely positioned to support SQL learners from novice to advanced levels, offering a balanced, theoretically grounded, and practically effective alternative to existing platforms.

## 2.2 Game Life Cycle Models

### 2.2.1 Game Lifecycle in Educational Games

The **game lifecycle** in educational game design refers to the structured sequence of phases a learner experiences in a game-based environment, designed to foster progressive cognitive engagement and knowledge mastery. Educational games differ from purely entertainment-based games by integrating pedagogical objectives into gameplay mechanics (Plass et al., 2015).

Typical Stages in Educational Game Lifecycle

1. **Onboarding**: Introduces learners to basic mechanics and the learning context. Tutorials and visual guidance reduce cognitive load, especially important in unfamiliar domains like SQL (Van Eck, 2006).
2. **Exploration**: Learners are given freedom to try basic interactions—such as entering queries or retrieving database records—building curiosity and confidence.
3. **Engagement**: Learners face structured challenges with escalating difficulty, using SQL clauses like WHERE, ORDER BY, and GROUP BY. Progression is often incentivized with gamified elements such as scores or badges (Deterding et al., 2011).
4. **Mastery**: Learners apply complex techniques like joins, nested queries, and case statements to solve advanced database problems, demonstrating comprehensive understanding.
5. **Completion**: The learning arc concludes with a capstone assessment or cumulative game mission, reinforcing retention and signaling accomplishment.

This lifecycle maps seamlessly onto SQL education. Our platform begins with onboarding tutorials, gradually increases difficulty via adaptive exploration, engages users with real-world query challenges, and culminates in mastery-level assessments involving multi-table joins and subqueries. Completion is marked by a final boss challenge or certification system.

The Game Development Life Cycle (GDLC) mirrors software engineering cycles but includes educational design principles. It ensures games are not only technically functional but pedagogically effective (Silva et al., 2022).

### 2.2.2 GDLC Stages

1. **Initiation**: Identify learning goals (e.g., SQL fluency), target users (e.g., university students), and core game concept (e.g., gamified database mission-based learning).
2. **Pre-Production**: Create learning-aligned game design documents, define SQL modules, establish performance metrics, and assess technical feasibility.
3. **Production**: Develop backend engines (e.g., SQL execution logic), UI components, gamification mechanics (e.g., experience points), and feedback loops.
4. **Testing**: Evaluate system usability and educational impact via alpha and beta testing. *Dungeon Code*, a programming game, achieved a usability alpha coefficient of 0.82, indicating strong learner engagement (Silva et al., 2022).
5. **Beta**: Conduct limited public rollout, gather real-time learner feedback, and fine-tune gameplay and content.
6. **Release and Maintenance**: Publish the platform, monitor usage data, and release iterative updates to address learner needs and bugs.



Figure 1/Game Development Life Cycle (GDLC) for Serious Game(<https://www.topdevelopers.co/blog/mobile-game-development-process/>)

Game development for educational or serious purposes can be guided by various models that balance pedagogical integrity, technical feasibility, and engagement design. These models provide systematic approaches for addressing challenges such as curriculum alignment, adaptive learning, and user experience.

### 2.2.3 Spiral Model

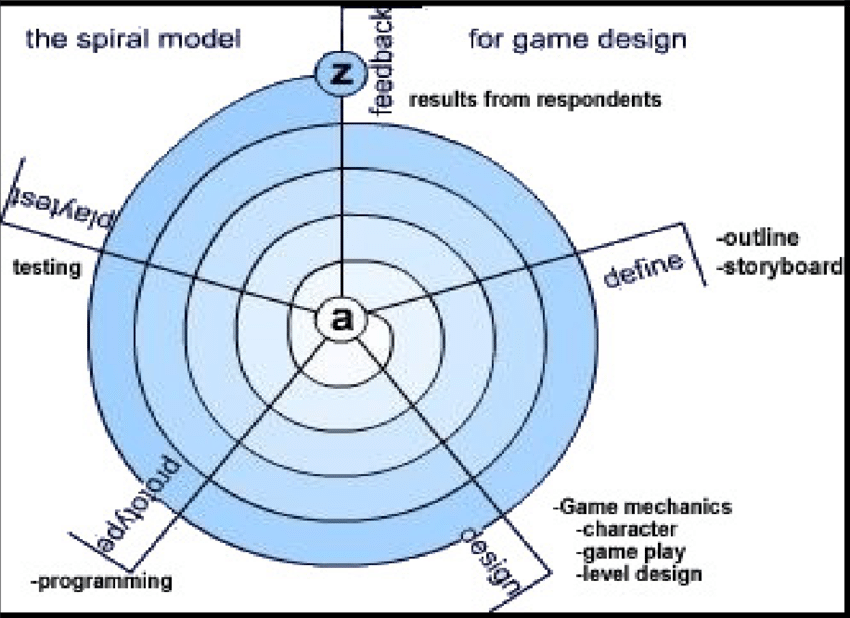
****

Figure 2/Spiral Model(https://www.researchgate.net/publication/274174154/figure/fig1/AS:669016012120067@1536517255427/Spiral-Model-for-Game-Development.png)

**Description:** The Spiral Model is a risk-driven development methodology that combines iterative prototyping with systematic aspects of the traditional waterfall model. Each loop in the spiral represents a development phase—planning, risk analysis, engineering, and evaluation—allowing incremental refinement based on feedback.

**Application to SQL Learning Platform:** In the context of gamified SQL learning, the Spiral Model supports frequent evaluation of content modules (e.g., SELECT, JOIN, SUBQUERY challenges) to ensure they remain aligned with learning goals and database standards. Iterative risk analysis helps identify potential learning bottlenecks, such as overly difficult tasks or poor feedback mechanisms.

**Drawback:** This model is resource-intensive and demands continuous user evaluation, which may not be feasible for smaller development teams or projects with limited time.

### 2.2.4 Agile Game Development

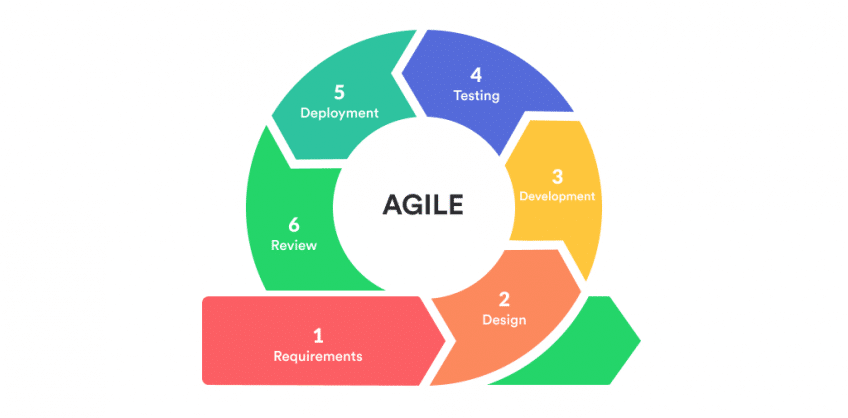
****

Figure 3/Agile Game Development(https://miro.medium.com/v2/resize:fit:1100/format:webp/0\*RziwSIdYb2capKWN.png)

**Description:** Agile Game Development adapts Agile principles—such as short development sprints, frequent iteration, and user feedback—to the game design process. It emphasizes rapid prototyping and continuous user validation.

**Application to SQL Learning Platform:** Using Agile, individual SQL levels (e.g., WHERE clause challenge) can be developed, tested, and revised within short cycles. Learner feedback can guide difficulty adjustment and UI enhancements.

**Drawback:** Agile may struggle to maintain a holistic view of long-term learning outcomes, particularly if educational goals are not explicitly tracked across sprints.

### 2.2.5 MDA Framework (Mechanics, Dynamics, Aesthetics)

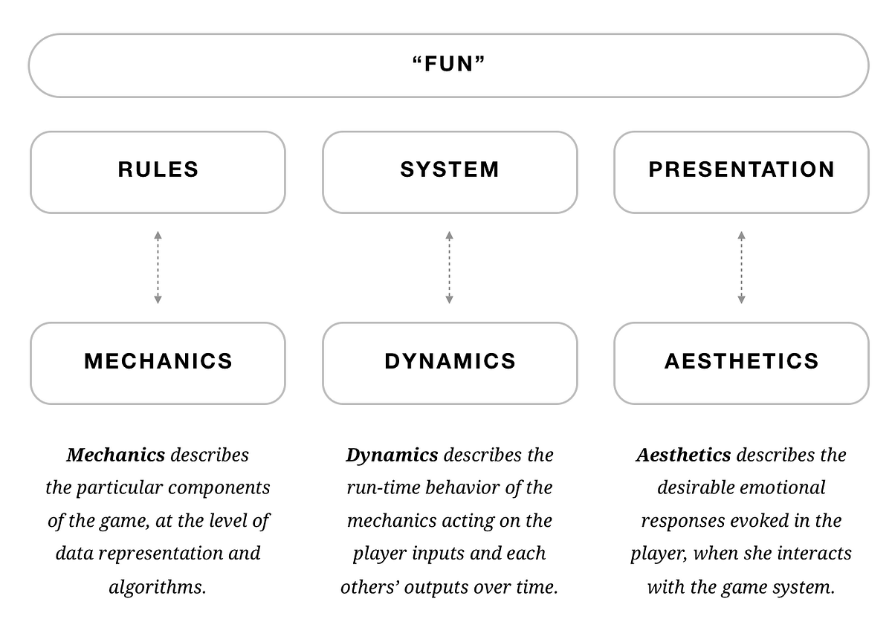
****

Figure 4/MDA Framework(https://medium.com/@jenny\_carroll/using-the-mda-framework-as-an-approach-to-game-design-9568569cb7d)

**Description:** The MDA Framework decomposes games into three interdependent components:

* Mechanics: Rules and systems (e.g., SQL question structure, scoring system)
* Dynamics: Real-time player interactions (e.g., query attempts, leaderboard response)
* Aesthetics: Emotional experiences (e.g., achievement, frustration, curiosity)

**Application to SQL Learning Platform:** MDA supports designing learning mechanics (such as point-based challenges) that produce motivating dynamics (progression, discovery) and yield aesthetic rewards (satisfaction from solving a hard SQL puzzle). It ensures gameplay remains engaging while aligned with learning intent.

**Drawback:** Requires strong design skills to balance all three elements harmoniously; a poorly designed mechanic may lead to unmotivating dynamics.

### 2.2.6 ADDIE-GDLC Hybrid

****

Figure 5/Addie GDLC Hybrid(https://www.researchgate.net/publication/378655023/figure/fig1/AS:11431281226728349@1709339784033/Development-of-the-ADDIE-Model-System.jpg)

**Description:** The ADDIE model (Analysis, Design, Development, Implementation, Evaluation) is a classic instructional design framework. When integrated with the Game Development Life Cycle (GDLC), it ensures both instructional validity and technical feasibility. The hybrid approach bridges education and game design.

**Application to SQL Learning Platform:** The hybrid model ensures that each SQL module follows a pedagogically sound sequence (ADDIE), while the GDLC ensures smooth technical execution—such as integrating feedback, adaptive challenge engines, and real-time syntax checks.

**Drawback:** This model is time-consuming due to its strong focus on planning, analysis, and iterative evaluation—making it slower compared to Agile or rapid prototyping methods.

### 2.2.7 Adaptive Frameworks

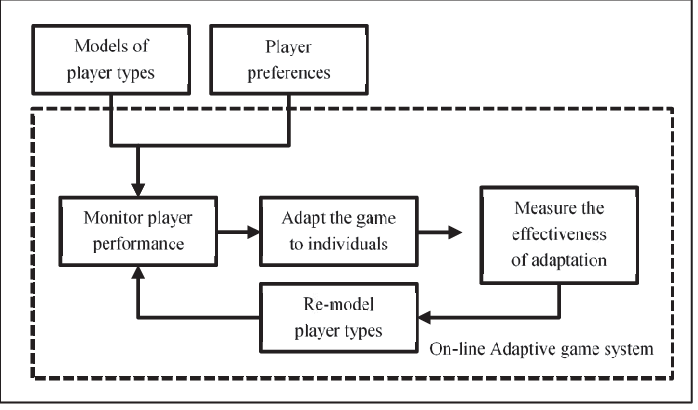
****

Figure 6/Adaptive Frameworks(https://www.researchgate.net/profile/Karl-Stringer/publication/228636408/figure/fig1/AS:669389149982721@1536606218528/A-potential-framework-for-an-adaptive-game-system.png)

**Description:** Adaptive learning frameworks employ learner modeling, real-time analytics, and AI-driven content adjustment to tailor learning paths. These systems dynamically adapt difficulty, content sequencing, and feedback based on user performance and behavior.

**Application to SQL Learning Platform:** The platform can adjust SQL query challenges based on real-time learner success rates. For instance, if a user struggles with JOIN, the system can recommend tutorials or delay advanced topics like subqueries. This personalization boosts engagement and retention.

**Drawback:** Developing adaptive systems requires complex back-end architecture, continuous data collection, and often involves ethical concerns such as learner data privacy.

## 2.3 Why ADDIE-GDLC Hybrid suite for my project

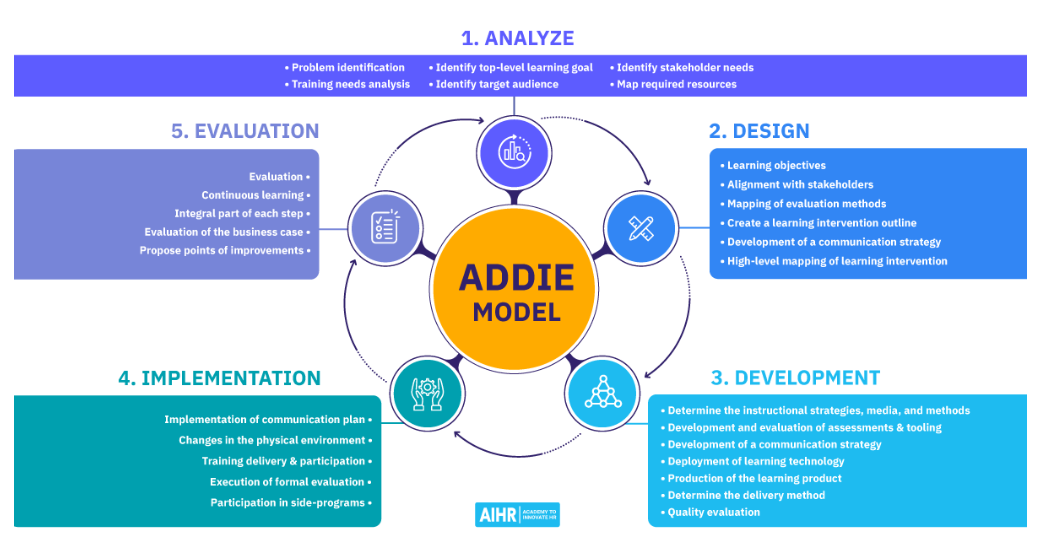
****

Figure 7/https://www.aihr.com/wp-content/uploads/ADDIE-Model-New.png

The ADDIE-GDLC Hybrid Model is a strong choice for Gamified SQL Learning Platform with Adaptive Difficulty because it blends instructional design principles (ADDIE) with game development structure (GDLC). This hybrid approach ensures that your project is:

* Educationally sound (by aligning with learning outcomes and cognitive theory),
* Technically robust (by following tested game development stages),
* And capable of engaging learners through gameplay (by introducing gamification and adaptivity).

# 3 Planning

Project planning establishes the foundation for structured development. It involves feasibility studies, risk assessment, SWOT and PESTAL analysis, and defining the life cycle model. Effective planning ensures the project is achievable, aligns with stakeholder expectations, and mitigates risks early.

## 3.1 Feasibility Report

### 3.1.1 Introduction

The SQL Safari project is a gamified learning platform designed to make SQL education engaging, interactive, and effective. By integrating real-world query challenges, gamification elements (points, badges, leaderboards), and narrative-driven missions, SQL Safari bridges the gap between theoretical concepts and practical application. This feasibility report evaluates the practicality and viability of SQL Safari by analyzing technical, operational, economic, and legal aspects.

### 3.1.2 Objectives

The objectives of this feasibility study are to assess the feasibility of:

* Developing a gamified SQL learning system that blends education with entertainment.
* Implementing query execution and real-time feedback features for SQL challenges.
* Designing interactive levels that gradually introduce SQL concepts, from beginner to advanced.
* Integrating instructor dashboards for monitoring learner progress.
* Deploying the platform as a responsive web application accessible across devices.

## 3.2 Technical Feasibility

### 3.2.1 Technology Stack

To ensure scalability, cross-platform access, and smooth user interaction, the following stack was chosen:

* **Frontend:**
  + HTML, CSS, JavaScript for interactive UI
  + Bootstrap for responsive design
  + Figma for UI/UX wireframes and mockups
* **Backend:**
  + PHP (Laravel framework) for core logic, authentication, adaptive hints, and gamification mechanics
* **Database:**
  + MySQL for storing user accounts, SQL queries, leaderboards, achievements, and datasets
* **Version Control & Tools:**
  + GitHub and Git for source code management
  + Figma for prototyping and game screen flows
  + In-game analytics system for tracking progress and engagement
* **Deployment Environment:**
  + Compatible with major web browsers (Chrome, Firefox, Safari, Edge)
  + Mobile device optimization for accessibility on phones and tablets
  + LMS (Learning Management System) integration possible for universities

### 3.2.2 Integration of Systems

The backend (Laravel) manages SQL execution logic, user progress, and gamification mechanics. Frontend interfaces built with Bootstrap and JavaScript provide interactive query editors and feedback panels. MySQL stores player profiles, progress, and achievements. APIs integrate the SQL evaluation engine with gamified progression and analytics.

## 3.3 Operational Feasibility

### 3.3.1 Resource Requirements

* **Full-Stack Developers:** Skilled in Laravel, MySQL, and frontend development.
* **Database Designers:** To create realistic SQL datasets for challenges.
* **Game/Instructional Designers:** To design levels, narratives, and reward systems.
* **UI/UX Designers:** To produce learner-friendly and visually engaging interfaces.
* **QA Testers:** To evaluate both gameplay and educational accuracy.

### 3.3.2 Stakeholders

* **Learners:** Students, professionals, and self-learners improving SQL skills.
* **Instructors:** Use dashboards to assign challenges and monitor progress.
* **Administrators:** Maintain system, manage users, and release updates.
* **Developers/Designers:** Continuously expand platform features.

### 3.3.3 Operational Workflow

1. Learner registers/logs in.
2. Learner attempts SQL challenges inside a gamified storyline.
3. Queries are executed and validated by backend logic.
4. Feedback, hints, and rewards (points, badges) are displayed instantly.
5. Progress is stored in MySQL and tracked with analytics.
6. Instructors access dashboards to monitor and guide students.

## 3.4 Economic Feasibility

### 3.4.1 Cost Analysis

**Initial Costs:**

* Development team (Laravel backend, MySQL DB, frontend dev).
* Cloud hosting & deployment environment.
* UI/UX design and prototyping (Figma).
* Marketing for academic institutions and learners.

**Ongoing Costs:**

* Server hosting and database management.
* Content expansion (new challenges, datasets).
* Platform maintenance and bug fixes.
* Continuous user support.

### 3.4.2 Revenue Model

* **Institutional Licensing:** Universities pay subscription fees.
* **Freemium Access:** Basic levels free, premium challenges/certifications for paying users.
* **Certification Fees:** Learners can purchase official SQL proficiency certificates.
* **Institution Add-ons:** Dashboards and LMS integrations offered as premium features.

### 3.4.3 Profitability

SQL Safari can achieve strong profitability due to the global demand for SQL skills. The freemium + institutional subscription model allows both individuals and universities to adopt the platform.

## 3.5 Legal Feasibility

### 3.5.1 Regulatory Compliance

* Compliance with Sri Lankan Data Protection Act and GDPR.
* Secure handling of learner data, encrypted authentication, and anonymized analytics.

### 3.5.2 Intellectual Property

* Game storyline, content, and UI designs protected under copyright.
* Original datasets and query challenges created for the platform.

### 3.5.3 Educational Compliance

* Content aligned with Bloom’s Taxonomy and database education standards.
* LMS integration ensures compatibility with existing academic systems.

**7. Conclusion**

Based on this feasibility analysis, SQL Safari is **technically, operationally, and economically feasible**. The chosen stack (Laravel, MySQL, Bootstrap, JavaScript) ensures scalability, responsiveness, and cross-platform usability. Combined with gamification and real-world datasets, the platform offers an innovative solution to SQL education. By supporting both independent learners and academic institutions, SQL Safari is positioned to become a sustainable, impactful learning tool.

## 3.6 Risk Assessment

**Technological Risks**

* *Risk: Query Execution Accuracy*
  + Incorrect validation of SQL queries could frustrate learners and reduce trust.
  + **Mitigation:** Build a robust query evaluation engine, use multiple test datasets, and provide clear error messages.
* *Risk: System Performance Under Load*
  + Large numbers of simultaneous learners could slow down query execution and leaderboard updates.
  + **Mitigation:** Optimize MySQL queries, use caching for frequent operations, and scale hosting on cloud servers.
* *Risk: Security of User Data and SQL Engine*
  + Users’ SQL submissions could be exploited for injection or malicious queries.
  + **Mitigation:** Sanitize all inputs, isolate the query execution environment, and enforce strict database permissions.

**Operational Risks**

* *Risk: Development Delays*
  + Laravel backend complexity or MySQL schema issues may extend timelines.
  + **Mitigation:** Break work into milestones, track progress in GitHub, and prioritize MVP features.
* *Risk: Skill Gaps in Gamification/UX*
  + Lack of expertise in creating engaging learning flows may weaken the game’s appeal.
  + **Mitigation:** Involve game designers early, prototype UI/UX in Figma, and test with learners.
* *Risk: Maintenance Overhead*
  + Continuous updates (new challenges, bug fixes) may strain resources.
  + **Mitigation:** Implement modular code, automate deployments, and document all systems clearly.

**Market and User Risks**

* *Risk: Low Student Engagement*
  + Learners may not find the platform engaging compared to traditional platforms (W3Schools, DataCamp, etc.).
  + **Mitigation:** Include badges, levels, storytelling missions, and leaderboards to boost motivation.
* *Risk: Institutional Resistance*
  + Universities may hesitate to adopt if integration with LMS is limited.
  + **Mitigation:** Build LTI-compatible modules for LMS integration and demonstrate academic value.
* *Risk: Accessibility Issues*
  + Students with poor internet or mobile-only access may struggle.
  + **Mitigation:** Optimize for mobile, minimize resource-heavy features, and support offline practice where possible.

**Economic Risks**

* *Risk: Higher Development and Hosting Costs*
  + Cloud hosting for real-time SQL execution may exceed budget.
  + **Mitigation:** Use scalable cloud plans, monitor usage, and control feature scope for initial versions.
* *Risk: Revenue Model Failure*
  + Institutions or learners may be unwilling to pay for premium features.
  + **Mitigation:** Adopt a freemium model—free basics, paid certifications or advanced analytics.

**Legal and Regulatory Risks**

* *Risk: Data Privacy Non-Compliance*
  + Handling user profiles and analytics may conflict with GDPR or local laws.
  + **Mitigation:** Encrypt all sensitive data, follow GDPR and Sri Lankan Data Protection laws, and publish clear policies.
* *Risk: Intellectual Property Issues*
  + Using third-party datasets or SQL tasks without rights could cause disputes.
  + **Mitigation:** Use original datasets or open-licensed ones, and copyright all SQL Safari content.

## 3.7 SWOT

**Strengths**

* Innovative use of gamification (points, badges, leaderboards, storyline) to make SQL learning engaging and motivating.
* Provides a structured progression from beginner SQL to advanced concepts like joins, subqueries, and normalization.
* Real-time query execution and instant feedback enhance practical learning compared to static textbooks.
* Built with widely used technologies (Laravel, MySQL, Bootstrap, JavaScript), ensuring scalability and maintainability.
* Potential integration with LMS systems makes it attractive for universities and training institutes.

**Weaknesses**

* High development complexity in combining game logic, SQL execution engine, and educational pedagogy.
* Requires continuous content expansion (new datasets, levels, and challenges) to maintain user interest.
* May present a learning curve for users unfamiliar with gamified learning environments.
* Limited resources or small development team could slow updates and bug fixes.
* Dependence on stable internet connection may affect accessibility for some learners.

**Opportunities**

* Growing global demand for SQL and database management skills across industries.
* Partnerships with universities, coding bootcamps, and online learning platforms to integrate SQL Safari as part of their curriculum.
* Potential to expand into related technologies (NoSQL, data visualization, cloud databases) as new modules.
* Certification opportunities can attract learners seeking career advancement and formal recognition.
* Ability to leverage learning analytics to offer personalized learning paths and recommendations.

**Threats**

* Competition from established platforms like LeetCode, HackerRank, DataCamp, and Codecademy.
* Risk of users dropping out if the difficulty curve or gameplay balance is not well designed.
* Security risks in query execution environment if not properly sandboxed.
* Educational institutions may hesitate to adopt if platform does not align with their existing syllabi.
* Rapid changes in technology trends may require frequent platform updates to remain relevant.

## 3.8 PESTAL Analysis

**Political Factors**

* Government initiatives in Sri Lanka and globally to promote ICT and digital education can support the adoption of gamified learning platforms like SQL Safari.
* Policies encouraging STEM and digital literacy in schools and universities create opportunities for integration with curricula.
* Potential dependency on government educational institutions’ approval for formal adoption may slow deployment.

**Economic Factors**

* Growing demand for IT professionals and SQL/database skills worldwide makes SQL Safari valuable in both academic and professional markets.
* Economic fluctuations could affect universities’ and learners’ ability to pay for subscriptions or certifications.
* Increasing investment in e-learning globally provides a favorable environment for gamified education businesses.

**Social Factors**

* Rising popularity of gamification and e-learning among younger learners creates a strong cultural fit for SQL Safari.
* Increasing student preference for interactive, practical learning methods over traditional lectures strengthens adoption potential.
* Some learners or educators may resist gamified learning due to unfamiliarity or preference for conventional teaching approaches.

**Technological Factors**

* Use of Laravel, MySQL, Bootstrap, and JavaScript ensures reliability, scalability, and compatibility with modern web standards.
* Growth in cloud hosting, mobile optimization, and LMS integration provides opportunities for expanding accessibility and institutional use.
* Rapid changes in educational technology trends may require continuous innovation to remain competitive with global platforms.
* Sandbox security for query execution must be carefully managed to avoid vulnerabilities.

**Environmental Factors**

* SQL Safari as a digital platform contributes to reducing paper usage and promotes sustainable education practices.
* Limited environmental footprint compared to physical training resources, but hosting on cloud servers still requires energy, making green hosting options a positive consideration.
* Universities increasingly value sustainability initiatives, and highlighting low-environmental impact could be a differentiating factor.

**Legal Factors**

* Compliance with Sri Lankan Data Protection Act and international frameworks like GDPR is necessary due to storage of learner data.
* Intellectual property protection is important for original game content, levels, datasets, and gamification mechanics.
* Licensing and academic accreditation laws may apply if the platform expands into formal certification programs.
* Security standards for online platforms (e.g., encryption, data storage compliance) must be followed to avoid legal liabilities.

## 3.9 Life Cycle Model

Given the dual nature of SQL Safari as both an educational tool and a gamified platform, the ADDIE–GDLC Hybrid model is the most suitable choice. This hybrid combines **ADDIE’s instructional design framework** (Analysis, Design, Development, Implementation, Evaluation) with the **Game Development Life Cycle (GDLC)** stages, ensuring that the platform is both **pedagogically sound** and **technically robust**.

### 3.9.1 Benefits of Using the ADDIE–GDLC Hybrid Model for SQL Safari

1. **Instructional Validity:**
   * The ADDIE model ensures that each SQL module aligns with clear learning objectives, following educational principles like Bloom’s Taxonomy.
2. **Structured Game Development:**
   * GDLC stages ensure the system is developed in a systematic, iterative manner, covering initiation, production, testing, and maintenance.
3. **Balanced Focus on Learning & Fun:**
   * The hybrid model ensures SQL Safari is not only functional but also engaging, combining query challenges with rewards and progression systems.
4. **Iterative Evaluation:**
   * Frequent evaluation cycles from ADDIE (Evaluation) and GDLC (Testing) ensure quality improvements and alignment with learner needs.
5. **Scalability and Adaptability:**
   * The model allows future expansion (new SQL challenges, datasets, or features) while maintaining core instructional integrity.

### 3.9.2 Overview of ADDIE–GDLC Hybrid Stages

1. **Analysis (ADDIE) & Initiation (GDLC)**
   * Define learning goals (e.g., SQL fluency, practical query skills).
   * Identify target learners (students, self-learners, institutions).
   * Establish scope: gamified missions, SQL challenges, leaderboards.
   * Deliverable: Project charter, user personas, initial game concept.
2. **Design (ADDIE) & Pre-Production (GDLC)**
   * Create storyboards, wireframes, and game flow (using Figma).
   * Map SQL modules to missions (SELECT, JOIN, SUBQUERY, etc.).
   * Define gamification mechanics (XP, badges, leaderboards).
   * Deliverable: Game design document, database schema, UI mockups.
3. **Development (ADDIE) & Production (GDLC)**
   * Implement backend (Laravel + MySQL) for query execution, progress tracking, and rewards system.
   * Build frontend using HTML, CSS, JavaScript, and Bootstrap.
   * Integrate in-game analytics for learner progress tracking.
   * Deliverable: Working builds of SQL levels and gameplay mechanics.
4. **Implementation (ADDIE) & Testing (GDLC)**
   * Deploy platform to a staging environment for user testing.
   * Conduct alpha/beta testing with learners and educators.
   * Validate SQL engine accuracy, gamification flow, and user experience.
   * Deliverable: Tested platform with bug fixes and refinements.
5. **Evaluation (ADDIE) & Release/Maintenance (GDLC)**
   * Gather learner and instructor feedback on SQL Safari missions.
   * Measure impact through analytics (completion rates, error patterns).
   * Release updates with new levels, datasets, and challenges.
   * Deliverable: Live platform with continuous improvements.

**Conclusion**

By applying the ADDIE–GDLC Hybrid Model, SQL Safari ensures that its **SQL learning content is structured, valid, and pedagogically effective**, while also maintaining the **technical rigor and engagement of a serious game**. This approach guarantees a balance between education and entertainment, making SQL Safari a sustainable and impactful learning platform.

## 3.10 Time Plan

The project timeline follows structured phases:

|  |  |  |  |
| --- | --- | --- | --- |
| **From Date** | **To Date** | **Work in Progress** | **For Next Week** |
| 03/07/2025 | 14/07/2025 | Project proposal submitted. | Prepare proposal literature review and game idea. |
| 14/07/2025 | 21/07/2025 | Completed proposal literature review and outlined the game idea. | Add more details to the literature review and expand on the game idea description. |
| 21/07/2025 | 28/07/2025 | Added more details to the literature review and described the game idea in depth. | Select a game development model and describe it. Add more research details on existing SQL learning platforms. |
| 28/07/2025 | 31/07/2025 | Submitted a game development model description. Added more research details on SQL learning platforms. | Summarize a different SQL learning platform. Combine both selected game development model and game idea into a document. Add a conclusion to the literature review. |
| 31/07/2025 | 04/08/2025 | Literature review submitted. | Add a storyline to the game idea. Select one level of the game scenario and describe it. |
| 04/08/2025 | 20/08/2025 | Added a storyline to the game idea. Selected one level of the game scenario and described it. | Build a prototype model for one game scenario. Add a scenario flow with outcomes. Introduce hints for different error outcomes. |
| 20/08/2025 | 24/08/2025 | Developed a prototype model for the first game scenario. | Add a scenario flow to the prototype. Ensure different error outcomes provide specific hints. |
| 24/08/2025 | 28/08/2025 | Expanded the scenario flow for the prototype. Added variations for different outcomes. | Display the SQL database at the bottom of the page. Add a “Next” button for when no SQL query appears. Create a help page with guidance for different outcomes. |
| 28/08/2025 | 31/08/2025 | Final project document submitted. | Add game reset functionality and finalize the help section. |

# 4.0 Requirement Gathering and Analysis

## 4.1 Requirement Gathering Technique Used for the Project

For the SQL Safari project, the main objective was to design an interactive gamified platform that simplifies learning SQL concepts through practical challenges, story-based missions, and gamification features such as points, badges, and leaderboards. Since the platform is both educational and entertainment-oriented, requirement gathering focused on **understanding learner needs, educator expectations, and usability concerns** to ensure the game aligns with real teaching goals while remaining engaging for students.

**Requirement Gathering Techniques**

To achieve this, a combination of questionnaires, interviews, and document analysis was used. These methods provided both broad insights from potential learners and detailed feedback from educators and database practitioners.

1. Questionnaires (Google Forms & Paper-Based Surveys)

**Technique**: Surveys were distributed among IT students, recent graduates, and beginner-level learners of databases. Both closed and open-ended questions were included to gather structured and subjective feedback.

**Objective**:

* Identify the most challenging SQL topics (e.g., JOINs, nested queries, GROUP BY).
* Understand learner expectations regarding feedback (hints, solutions, explanations).
* Measure interest in gamification features like XP, levels, and leaderboards.

**How it worked:**

* Closed questions asked learners to rank SQL topics by difficulty.
* Open-ended questions gathered suggestions on what would make learning SQL enjoyable.

**Impact**:

* Results showed high demand for step-by-step query explanations and gamified progression.
* Students expressed a need for immediate error feedback and structured difficulty progression.

3. Document Analysis

**Technique**: Reviewed existing SQL learning platforms (e.g., SQL Murder Mystery, W3Schools, LeetCode SQL) and academic papers on gamified learning.

**Objective:**

* Identify strengths and gaps in current tools.
* Understand how gamification and storytelling have been applied to learning environments.

**Impact:**

* Found that most existing platforms either lack motivation (theory-heavy) or are too advanced (industry-focused).
* Helped position SQL Safari as a bridge between theory and real-world practice, with progressive, beginner-friendly challenges.

**Integration with the ADDIE–GDLC Hybrid Model**

The requirement gathering process was embedded into the Analysis stage of ADDIE and the Initiation/Pre-Production stages of GDLC:

* Sprint Planning: Data from surveys and interviews informed user stories, particularly for level design and error feedback mechanisms.
* Game Story Development: Insights guided the creation of story-based SQL challenges to maintain learner engagement.
* Continuous Feedback: As prototypes were tested, user feedback was collected and incorporated into later iterations, ensuring the platform evolved in alignment with learner needs.

**Conclusion**

By combining surveys, interviews, and document analysis, SQL Safari gathered both learner-cantered and educator-driven requirements. This dual approach ensured that the platform was practical, engaging, and pedagogically valid. The techniques not only identified pain points in learning SQL but also helped prioritize features such as error-based hints, progressive difficulty levels, and gamified rewards, making the system effective for both novices and intermediate learners.

## 4.2 Questionnaire

A structured questionnaire was distributed among 50 undergraduate IT students. Questions

**Requirement Gathering Technique Used for the Project**

For the SQL Game project, the primary objective was to design an engaging, gamified platform that helps learners improve their SQL skills through interactive scenarios, storytelling, and practice-based challenges. To ensure the platform was both effective and user-centered, a structured requirement gathering process was conducted. This process emphasized understanding learners’ struggles with SQL, their preferences for gamified learning, and their expectations for an interactive platform.

**Requirement Gathering Techniques**

**1. Google Forms (Surveys)**

* **Technique:** Surveys were distributed among IT undergraduates and beginner programmers using Google Forms. The aim was to capture both quantitative and qualitative insights into their learning challenges and preferences for a gamified SQL learning tool.
* **Objectives:**
  + Identify the frequency with which learners practice SQL.
  + Understand the common challenges students face when learning SQL (e.g., writing queries, understanding JOINs, debugging errors).
  + Assess learners’ interest in a game-based platform for SQL practice.
  + Gather expectations regarding additional features (e.g., hints, leaderboards, interactive feedback).
* **How it worked:**
  + Closed questions focused on challenges (e.g., “What difficulties do you face most when practicing SQL queries?”).
  + Open-ended questions allowed learners to suggest useful features (e.g., “What would make a SQL learning game more engaging for you?”).
* **Impact:**
  + Responses highlighted that most learners practice SQL regularly but face significant barriers with JOINs and query debugging.
  + Many expressed that interactive feedback and a structured storyline would help them stay motivated.

**2. Semi-Structured Interviews**

* **Technique:** One-on-one interviews were conducted with selected IT students and lecturers familiar with teaching SQL.
* **Objectives:**
  + Explore difficulties students face when transitioning from theoretical database concepts to writing practical queries.
  + Understand the teaching gaps in traditional database courses and how gamification might fill those gaps.
  + Gather insights into desirable gamified elements (e.g., XP, badges, progressive difficulty).
* **Sample Questions:**
  + “What parts of SQL do students struggle with the most?”
  + “How could a game-based approach make SQL learning easier?”
  + “What game features would keep students engaged long-term?”
* **Impact:**
  + Lecturers emphasized the importance of scaffolding: starting with simple SELECT queries before moving to complex joins.
  + Students highlighted the need for instant feedback and adaptive hints.
  + This input confirmed the design decision to include progressive difficulty levels, immediate feedback, and scenario-based tasks.

**Integration with ADDIE–GDLC Hybrid Model**

In alignment with the ADDIE–GDLC Hybrid model, the requirement gathering process was iterative and user-focused:

* **Analysis Phase:** Data from surveys and interviews was used to define core learner needs (e.g., hints, leaderboard, realistic scenarios).
* **Design Phase:** User stories were created to prioritize features such as adaptive hints, scenario-based query solving, and progress tracking.
* **Development Phase:** Requirements were converted into backlog items for sprint-based implementation.

**Key Findings from Survey Data**

1. **Frequency of SQL Practice:**
   * The majority of students reported practicing SQL weekly, confirming that the game should provide structured, bite-sized levels aligned with regular practice habits.
2. **Challenges in SQL Learning:**
   * 83% of respondents indicated difficulty with **writing accurate queries and debugging errors**, especially when JOINs and nested queries were involved.
   * This reinforced the need for **error-specific hints** and a built-in SQL execution environment within the game.
3. **Interest in a Dedicated SQL Game Platform:**
   * Over 80% agreed that a gamified platform would make SQL learning more engaging compared to textbooks or static tutorials.
   * Students highlighted the motivational benefits of scores, levels, and immediate feedback.
4. **Importance of Features:**
   * 100% of participants agreed that **instant feedback and hints** were essential.
   * Similarly, all respondents valued **interactive challenges and storytelling** to sustain engagement.
5. **Desired Additional Features:**
   * Leaderboards to encourage competition.
   * Badges or achievements to mark progress.
   * Sample solutions for queries after failed attempts.
   * Scenario-based storytelling for real-world SQL applications.

## 4.3 Interview

Interviews with My Supervisor highlighted the limitations of traditional SQL teaching. Lecturers

The interviews conducted for the SQL Game project focused on understanding the expectations and challenges faced by different user groups in learning SQL. The insights gathered are crucial for shaping a gamified platform that supports students, lecturers, and self-learners in improving their SQL skills through interactive, scenario-based learning.

**Interview Focus**

**1. Undergraduate Students and Beginner Learners**

* **Challenges:** Many students reported difficulties with writing correct queries, especially when dealing with complex SQL structures such as JOINs, nested queries, and subqueries. They also noted frustration when error messages from traditional SQL environments were unclear or unhelpful.
* **Key Insights:** Students emphasized the importance of a system that provides **instant feedback and adaptive hints**. They wanted a game that doesn’t just mark answers as wrong but explains why, offering targeted guidance.
* **Game Expectations:** Students preferred scenario-based learning that connects queries to real-world examples, making it easier to understand the purpose of SQL.
* **Preferred Features:** A leaderboard, achievements, and story-based progression were seen as motivating features that could make learning more enjoyable.

**2. Lecturers and Academic Experts**

* **Challenges:** Lecturers highlighted the difficulty of ensuring that all students engage with SQL practice consistently. They noted that many students struggle to move from understanding theoretical database design to applying that knowledge in practical query writing.
* **Key Insights:** Educators stressed the importance of **scaffolding**—starting with simple SELECT queries before advancing to more complex topics like joins, grouping, and subqueries.
* **Game Expectations:** They recommended a system that tracks student progress, provides analytics on common mistakes, and integrates with learning management systems (LMS) for classroom use.
* **Preferred Features:** A reset-and-try-again option, along with detailed explanations of query logic, was considered essential for reinforcing concepts.

**3. Self-Learners and Hobbyists**

* **Challenges:** Individuals learning SQL outside of formal education often rely on static tutorials or online references, which lack interactivity and do not provide immediate feedback.
* **Key Insights:** These learners expressed interest in a game-based platform that allows them to learn at their own pace while still receiving structured guidance.
* **Game Expectations:** They wanted engaging, flexible gameplay that feels rewarding even when learning independently, with progress tracking to monitor their improvement over time.
* **Preferred Features:** Scenario-driven challenges and sample solutions after multiple failed attempts were identified as features that would help build confidence.

### 4.3.1 Summary of Interview Insights

The interviews highlighted several recurring themes directly applicable to the development of the SQL Game platform:

* **Interactive Query Feedback:** Students and lecturers agreed that a critical feature is the ability to run queries and receive real-time feedback, including hints and explanations for errors.
* **Scaffolded Progression:** Beginners need gradual progression from basic queries (SELECT, WHERE) to more advanced ones (JOINs, GROUP BY, nested queries), with checkpoints to reinforce learning at each stage.
* **Motivational Gamification:** Leaderboards, badges, and achievements are effective in sustaining learner motivation and engagement across multiple sessions.
* **Scenario-Based Contexts:** Realistic challenges tied to practical database problems help learners see the relevance of SQL beyond theory.
* **Instructor Support:** Educators value analytics and tracking features that help them monitor student performance and tailor teaching interventions.

**Conclusion**

The interviews, combined with the Google Forms questionnaire data, provided a clear picture of user needs for the SQL Game project. Key takeaways include:

* The need for **instant, meaningful feedback** on queries, with adaptive hints to support learning.
* A **scaffolded learning path** that gradually increases difficulty, making SQL accessible for beginners while still challenging for advanced learners.
* Strong demand for **gamification elements** such as achievements, leaderboards, and story-based progression to maintain engagement.
* The importance of **scenario-driven challenges** that connect SQL practice to real-world use cases.
* The opportunity to include **progress tracking and analytics**, enabling both learners and lecturers to measure improvements over time.

## 4.4 Functional and Non-Functional Requirements

The requirements are divided into functional and non-functional aspects to guide development.

### 4.4.1 Functional Requirements

1. **SQL Query Execution System**
   * **Description:** The platform must provide a built-in SQL editor where learners can practice queries against a predefined database schema in real time.
   * **Features:**
     + Users can type and execute SQL commands directly in the game interface.
     + The system must display query results immediately, showing either the expected output or detailed error messages.
     + Error feedback should include explanations and hints to guide learners toward the correct query.
     + A reset option should be available to restore the default database state.
2. **Scenario-Based Game Levels**
   * **Description:** The game must provide story-driven SQL challenges, where each level represents a problem scenario requiring SQL solutions.
   * **Features:**
     + Each level introduces tasks of increasing complexity (e.g., SELECT, WHERE, JOIN, GROUP BY, nested queries).
     + Scenarios must be connected to real-world database problems for better context.
     + The game should provide multiple possible outcomes depending on the learner’s query accuracy.
     + A “next level” button should only appear after successful query completion.
3. **Gamification Mechanics**
   * **Description:** To increase engagement, the platform must incorporate points, leaderboards, badges, and achievements.
   * **Features:**
     + Learners earn points for completing levels correctly.
     + Special achievements are unlocked for milestones (e.g., first JOIN query solved, perfect score in a scenario).
     + Leaderboards display top performers by points or progress.
     + Progress is stored so learners can resume from their last completed level.
4. **Hint and Feedback System**
   * **Description:** The platform must support adaptive hints and feedback for incorrect queries.
   * **Features:**
     + Hints vary depending on error type (e.g., syntax error, missing WHERE clause, wrong column).
     + Multiple incorrect attempts trigger step-by-step guidance.
     + A help page explains SQL syntax with examples.
     + Learners can compare their query with the correct solution after repeated failures.
5. **Database Integration**
   * **Description:** The system must run queries on a predefined MySQL database schema designed for practice.
   * **Features:**
     + Database includes sample tables (e.g., Students, Courses, Hotels, Employees).
     + Queries are executed in a safe sandbox environment to prevent schema corruption.
     + Support for both DDL (CREATE, ALTER) and DML (SELECT, INSERT, UPDATE, DELETE).
     + Reset database option per user session.
6. **Mobile-Responsive Web Design**
   * **Description:** The platform must be fully usable on both desktop and mobile devices.
   * **Features:**
     + Responsive design with Bootstrap for consistent layout.
     + SQL editor optimized for touchscreens on tablets/mobiles.
     + Compact leaderboard and score displays for mobile view.

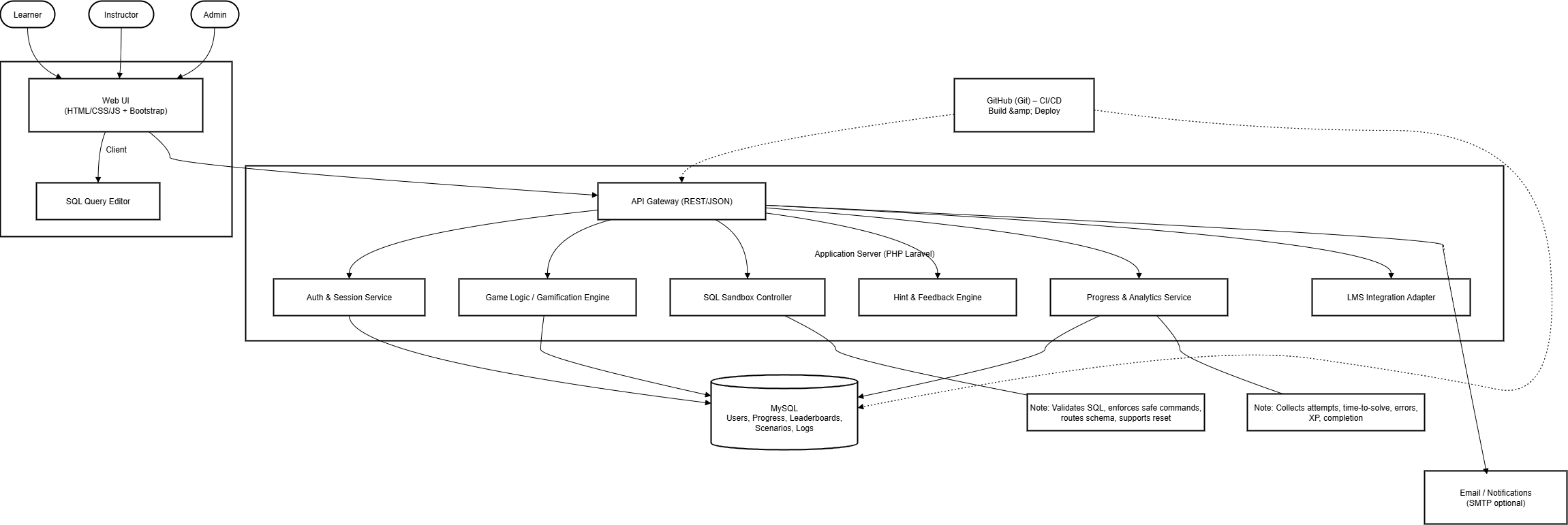
### 4.4.2 Non-functional Requirement

1. **Performance**
   * **Description:** The system must deliver smooth performance to ensure learners can interact with the SQL game seamlessly.
   * **Requirements:**
     + The platform should load within 3 seconds on both desktop and mobile devices.
     + SQL queries should execute and return results within 3–5 seconds.
     + The game must support at least 500 concurrent users in a classroom/lab setup without performance degradation.
     + Leaderboard and achievement updates must be reflected in real time with minimal latency.
2. **Usability**
   * **Description:** The platform should be intuitive and accessible to learners of different technical levels.
   * **Requirements:**
     + The user interface should be clear and easy to navigate, even for beginners with no SQL knowledge.
     + Game instructions, hints, and feedback should be understandable without requiring external tutorials.
     + The registration and login process should be simple, with clear guidance for first-time users.
     + Completing tasks (running queries, checking results, progressing to the next level) should require minimal clicks.
3. **Availability and Reliability**
   * **Description:** The platform must remain accessible and stable for continuous use.
   * **Requirements:**
     + The system should guarantee 99.9% uptime during learning sessions.
     + In case of a crash, recovery must be possible within 30 minutes to avoid disrupting gameplay.
     + Daily automatic backups of user progress, scores, and query logs should be maintained.
4. **Scalability**
   * **Description:** The platform should be capable of handling growth in terms of users, content, and game features.
   * **Requirements:**
     + The system should allow the addition of new SQL levels, scenarios, and query challenges without major rework.
     + The architecture should support scaling up for larger user groups, such as integration with university LMS systems.
     + The backend must handle expansion of the game database (new schemas, advanced query datasets) efficiently.
5. **Security**
   * **Description:** The system must protect learner data and ensure safe interactions.
   * **Requirements:**
     + All communication between clients and the server must be encrypted using SSL/TLS.
     + User authentication must be secure, with session management handled via Laravel.
     + Passwords must be hashed before storage in the MySQL database.
     + The system must comply with data privacy requirements relevant to educational institutions.
6. **Maintainability**
   * **Description:** The system should be easy to maintain and update, ensuring smooth long-term operation.
   * **Requirements:**
     + The codebase should follow modular design principles to allow updates (e.g., adding new game scenarios) without affecting existing features.
     + Regular updates, bug fixes, and enhancements should be easy to deploy via GitHub version control.
     + Documentation must be maintained for developers and instructors for smooth onboarding.
7. **Compatibility**
   * **Description:** The platform should be accessible across devices and browsers.
   * **Requirements:**
     + The website must be responsive and fully functional on desktops, laptops, tablets, and smartphones.
     + The platform must support major web browsers, including Chrome, Firefox, Safari, and Edge.
     + SQL execution and gamification features should work consistently across different screen sizes.
8. **Compliance**
   * **Description:** The platform must adhere to educational and software compliance standards.
   * **Requirements:**
     + Must comply with institutional IT security guidelines.
     + Ensure data protection in accordance with local education data policies.
     + Follow accessibility guidelines (WCAG) to support learners with disabilities.
9. **Sustainability**

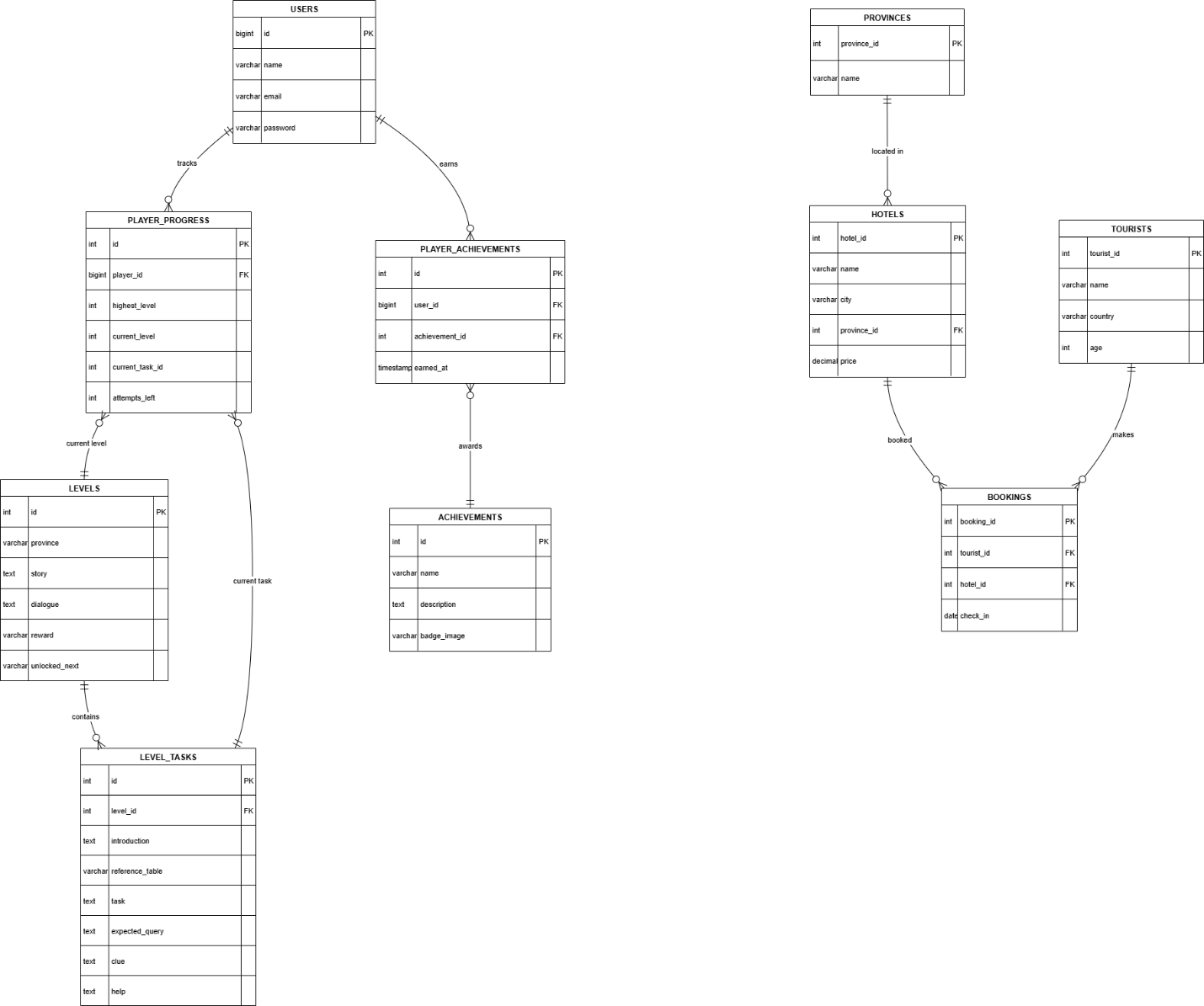
* **Description:** The platform should support sustainable use of resources.
* **Requirements:**
  + Use cloud hosting with optimized resource allocation to reduce unnecessary energy consumption.
  + Encourage digital-first practices (minimizing printed handouts by tracking progress online).
  + Provide lightweight design to reduce bandwidth usage, particularly for students with limited internet access.

# 5.0 System Design

## 5.1 Architecture Diagram

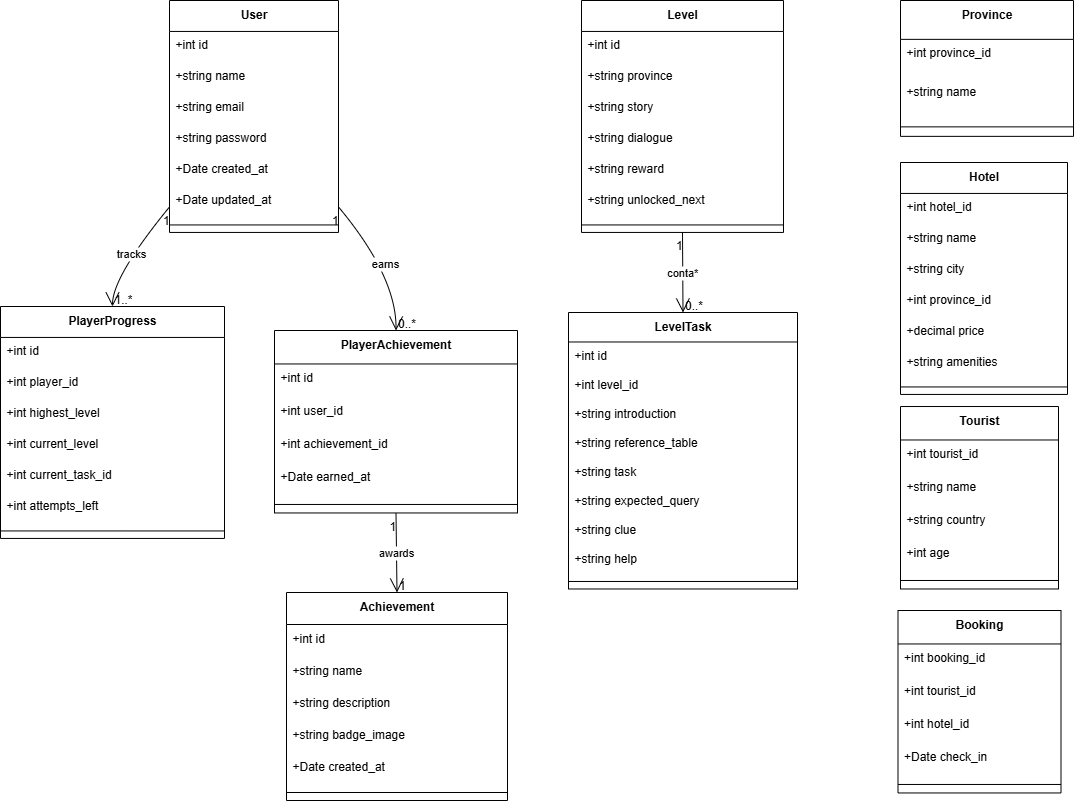


## 5.2 ER Diagram

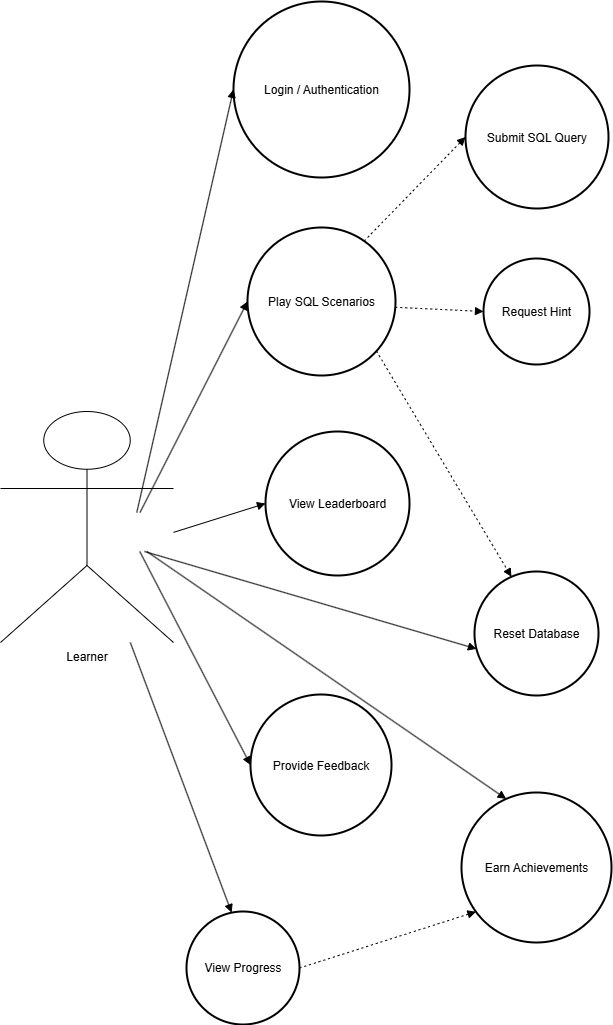


## 5.3 UML Diagrams

### 5.3.1 Class Diagram



### 5.3.2 Use Case Diagram



## 5.4 Wireframe Diagram



# 6.0 Implementation

The implementation of *SQL Travels – Explore Sri Lanka with Queries* focuses on delivering a functional gamified learning platform that combines SQL education with Sri Lankan cultural storytelling. The project is developed using the ADDIE-GDLC Hybrid Model, ensuring both instructional quality and game design rigor.

The system guides learners through Sri Lanka’s nine provinces, each province representing a level with a specific SQL concept. Learners progress by solving SQL challenges contextualized in a travel narrative, supported by characters, adaptive hints, leaderboards, and rewards.

The implementation strategy is broken into:

* Technology Stack (tools & frameworks)
* Design Patterns (architectural & gamification patterns)
* Program Implementation (game story, mechanics, launch, evaluation)

## 6.1 Technology Stack

To ensure scalability, cross-platform access, and smooth user interaction, the following stack was chosen:

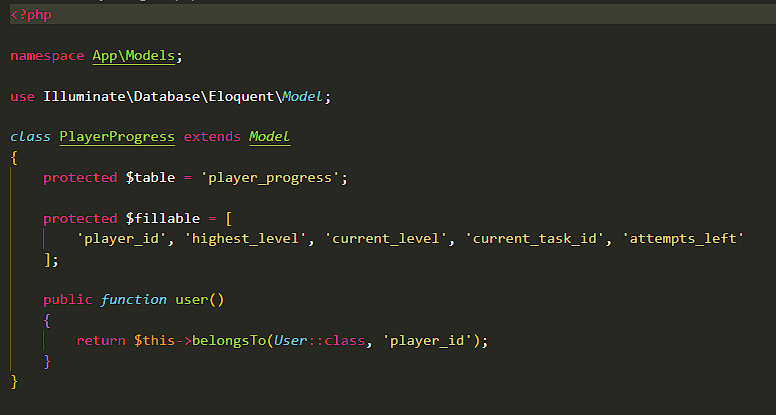
* Frontend:
  + HTML, CSS, JavaScript for interactive UI
  + Bootstrap for responsive design
  + Figma for UI/UX wireframes and mockups
* Backend:
  + PHP (Laravel framework) for core logic, authentication, adaptive hints, and gamification mechanics
* Database:
  + MySQL for storing user accounts, queries, leaderboards, achievements, and SQL datasets
* Version Control & Tools:
  + GitHub and Git for source code management
  + Figma for prototyping and game screen flows
  + In-game analytics system for progress tracking
* Deployment Environment:
  + Compatible with web browsers (Chrome, Firefox, Safari, Edge)
  + Mobile device optimization
  + LMS (Learning Management System) integration possible for universities

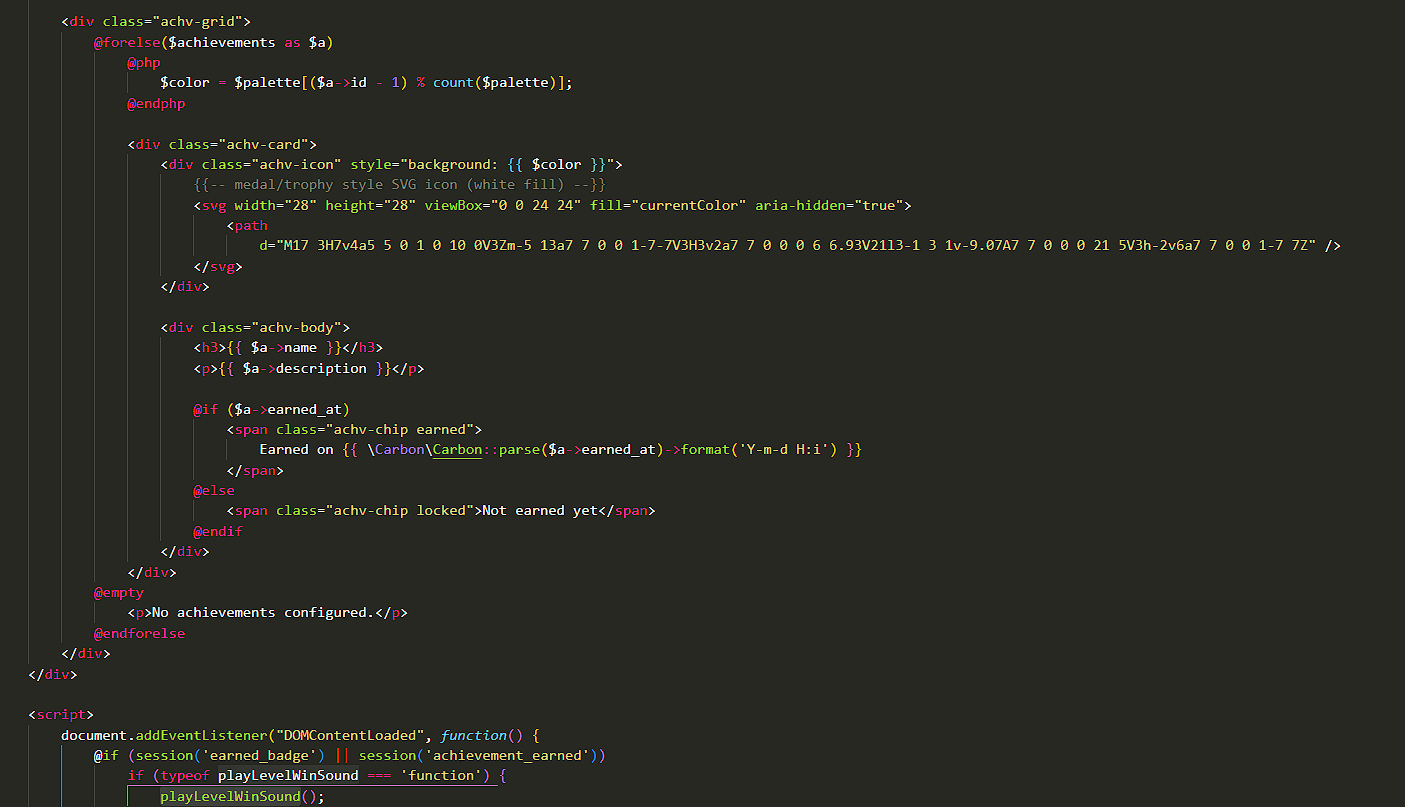
## 6.2 Design Patterns

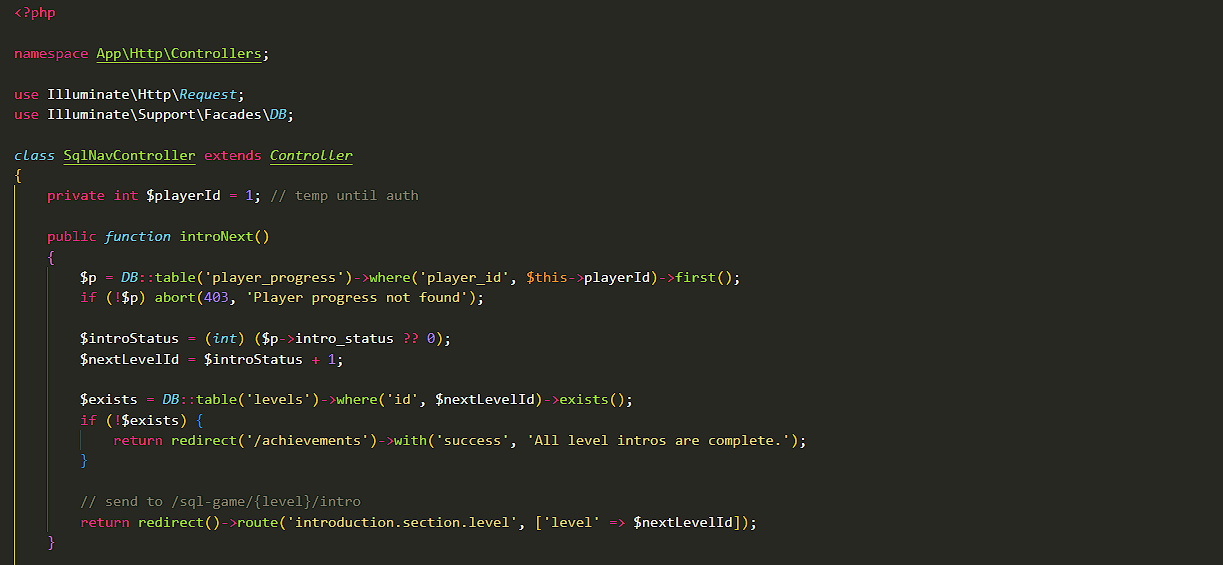
The SQL Safari project incorporates several design patterns to ensure maintainability, scalability, and efficiency in both gameplay and backend systems. Below are the design patterns applied in different aspects of the project:

Model-View-Controller (MVC) Pattern

* Model: The Laravel backend models represent entities such as Users, Levels, Tasks, Achievements, and Progress. They handle database interactions such as fetching, inserting, and updating gameplay data.
* View: The web UI (HTML/CSS/JS + Bootstrap) acts as the View, displaying SQL challenges, hints, leaderboards, and progress dashboards.
* Controller: Laravel controllers (e.g., GameController, AuthController) handle user requests, interact with the model, and update the view with results.

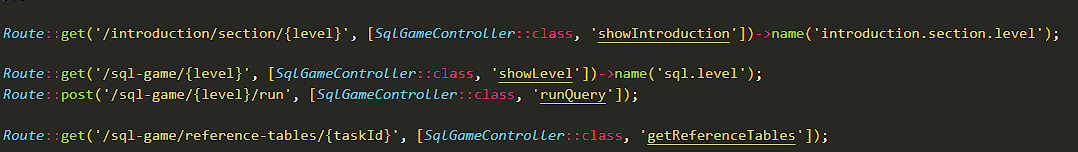
Sample Code:  
*Model example – handling database operations for levels.*  


*View example – displaying leaderboard entries with Bootstrap.*  


*Controller example – processing SQL query submissions and validating results.*  


Singleton Pattern

* Laravel Application Instance: The Laravel app instance (app()) ensures only one instance of the core application is running, centralizing configuration and service providers.
* Database Connection Pooling: Laravel uses a single database connection pool per request, which follows the Singleton principle for consistent and efficient access.



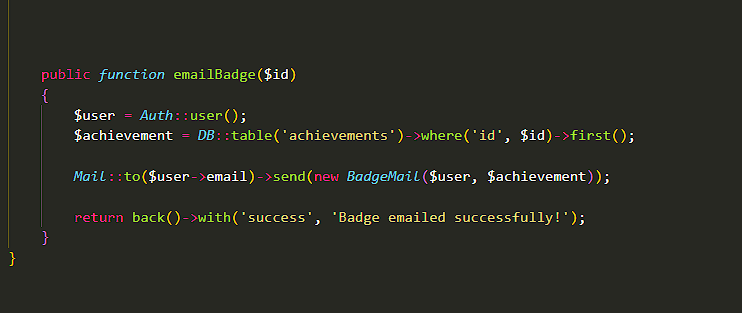
Factory Pattern

* User and Achievement Creation: When a player registers or earns an achievement, Laravel’s factory methods (e.g., User::create(), Achievement::create()) generate new objects.
* Game Tasks: Tasks for each level are created and retrieved dynamically via Eloquent factories, ensuring consistency in object creation.

****

Observer Pattern

* Player Achievements: When a player completes a task, the system “notifies” the Achievement system, awarding badges or updating the leaderboard.
* Email/Notifications: Upon registration or milestone completion, Laravel event listeners trigger email notifications or in-game messages.

****

Database Management

* Laravel Migrations: All tables (Users, Levels, Tasks, Achievements, Progress, Leaderboards) are managed via migration scripts, which follow a structured pattern to ensure maintainability and version control.

## Implementation of the Program

### 6.3.1 Game Interfaces

The SQL Safari learning game was implemented as a web-based interactive platform using Laravel (backend), MySQL (database), and HTML/CSS/Bootstrap (frontend). The system flow was designed to combine learning with gamification, ensuring students practice SQL queries in an engaging storyline environment.

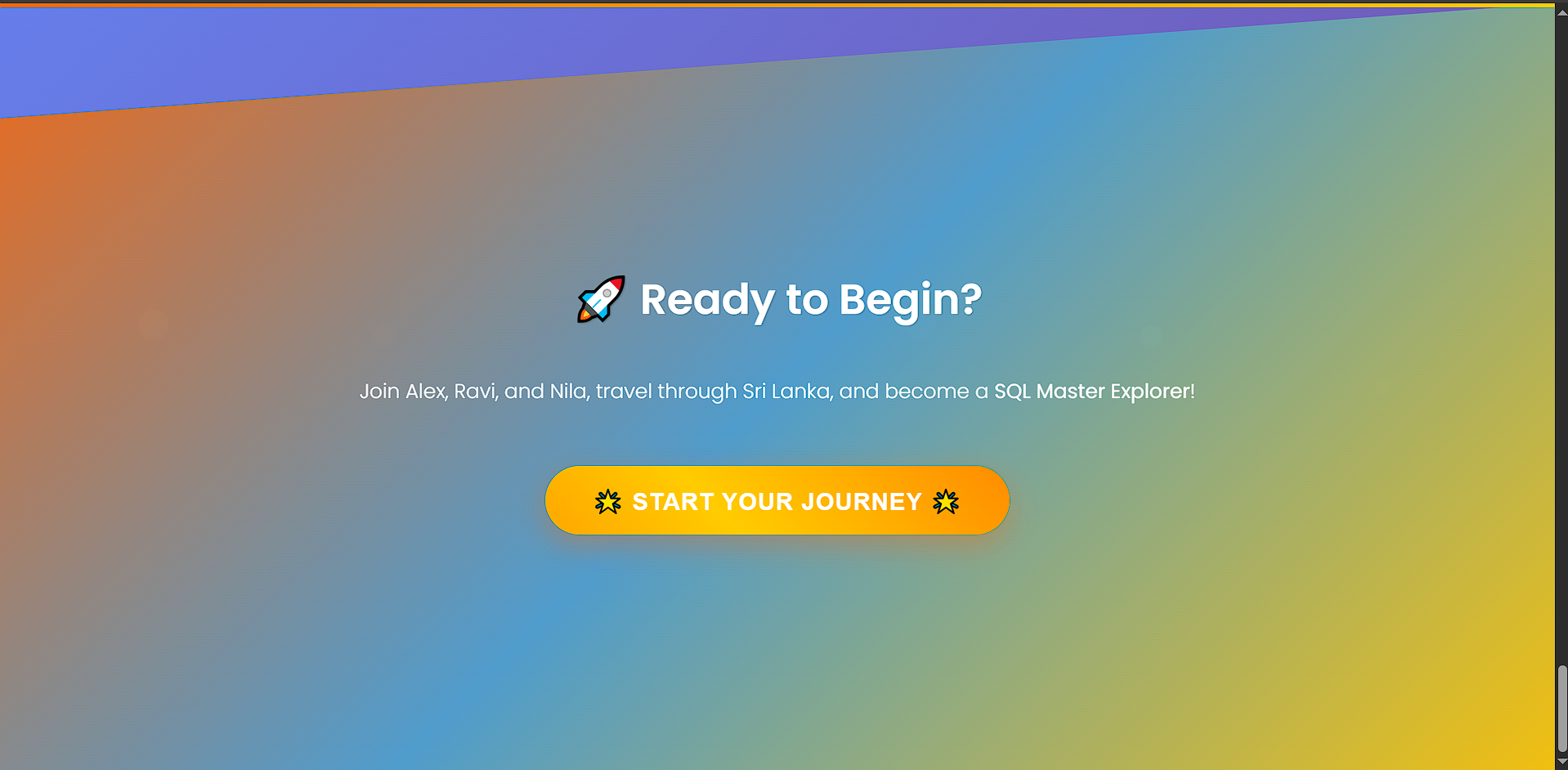
User Visit & Character Introduction

* When the user visits the website, the landing page auto-scrolls and introduces the main characters through an animated storyline.
* This creates immersion and motivates learners to continue the journey.



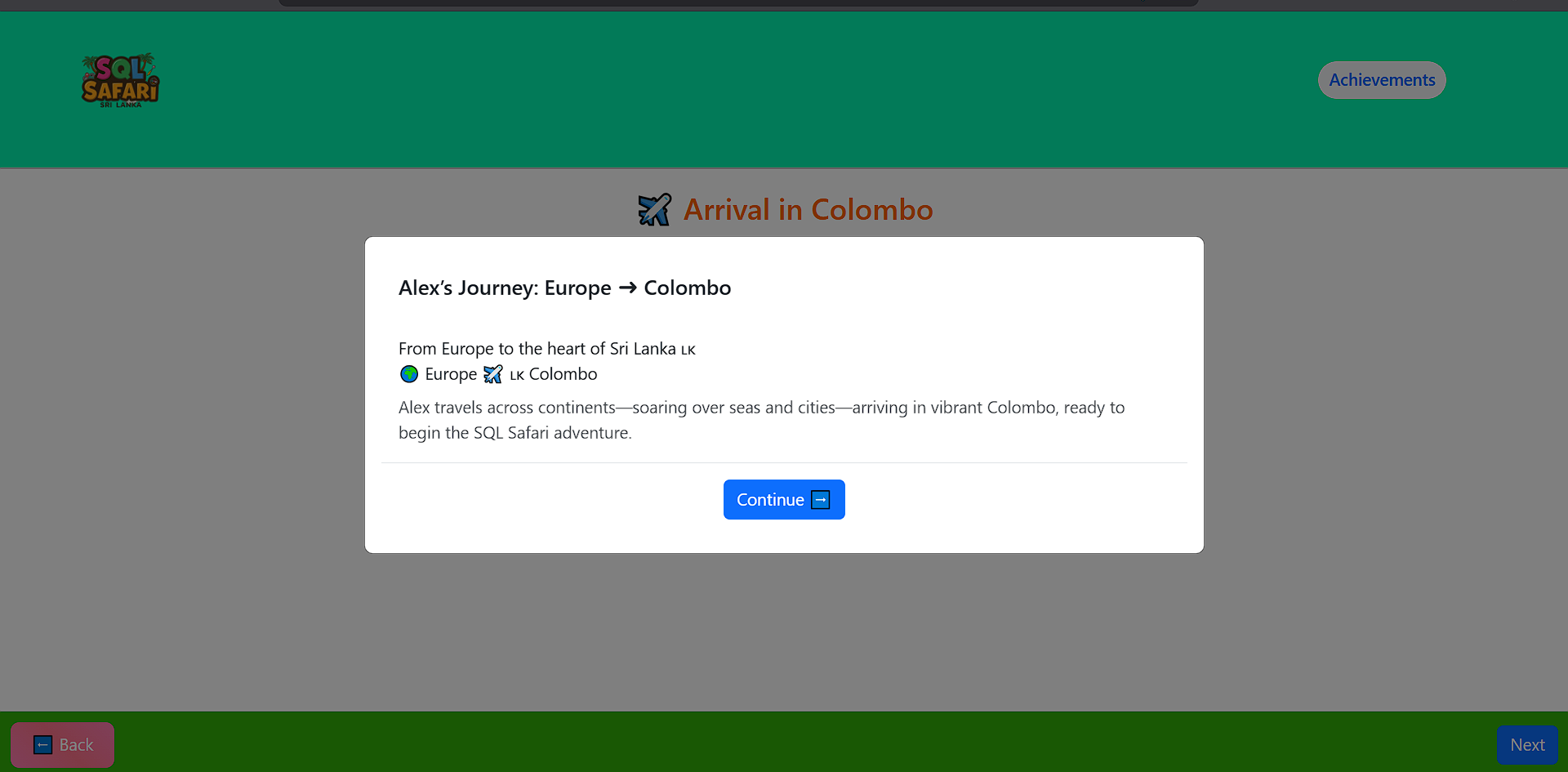
Get Started & Section 1 Introduction

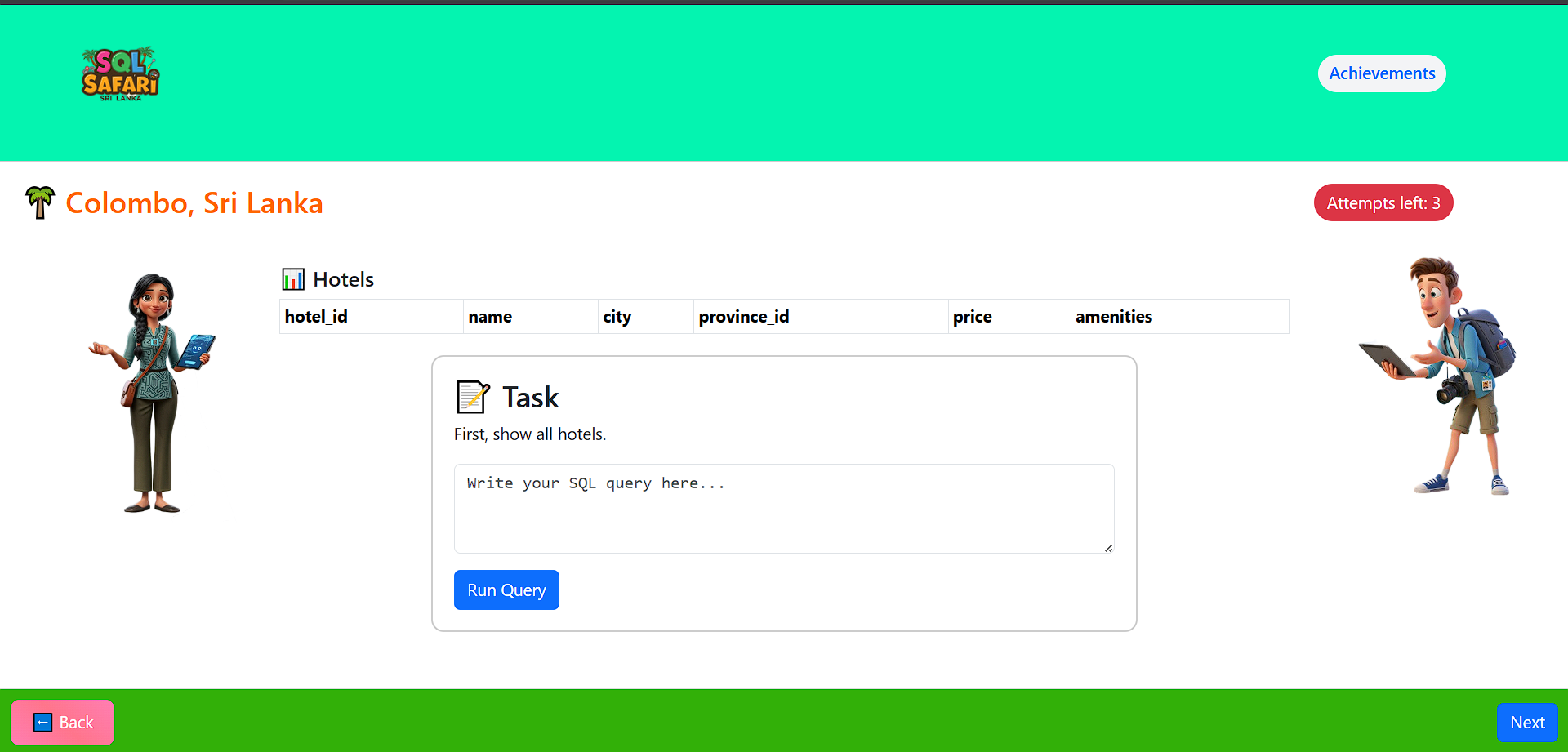
* The “Get Started” button redirects the learner to Section 1 Introduction Page.
* This page provides more details about the characters and the mission, setting the stage for SQL challenges.



SQL Challenge Environment

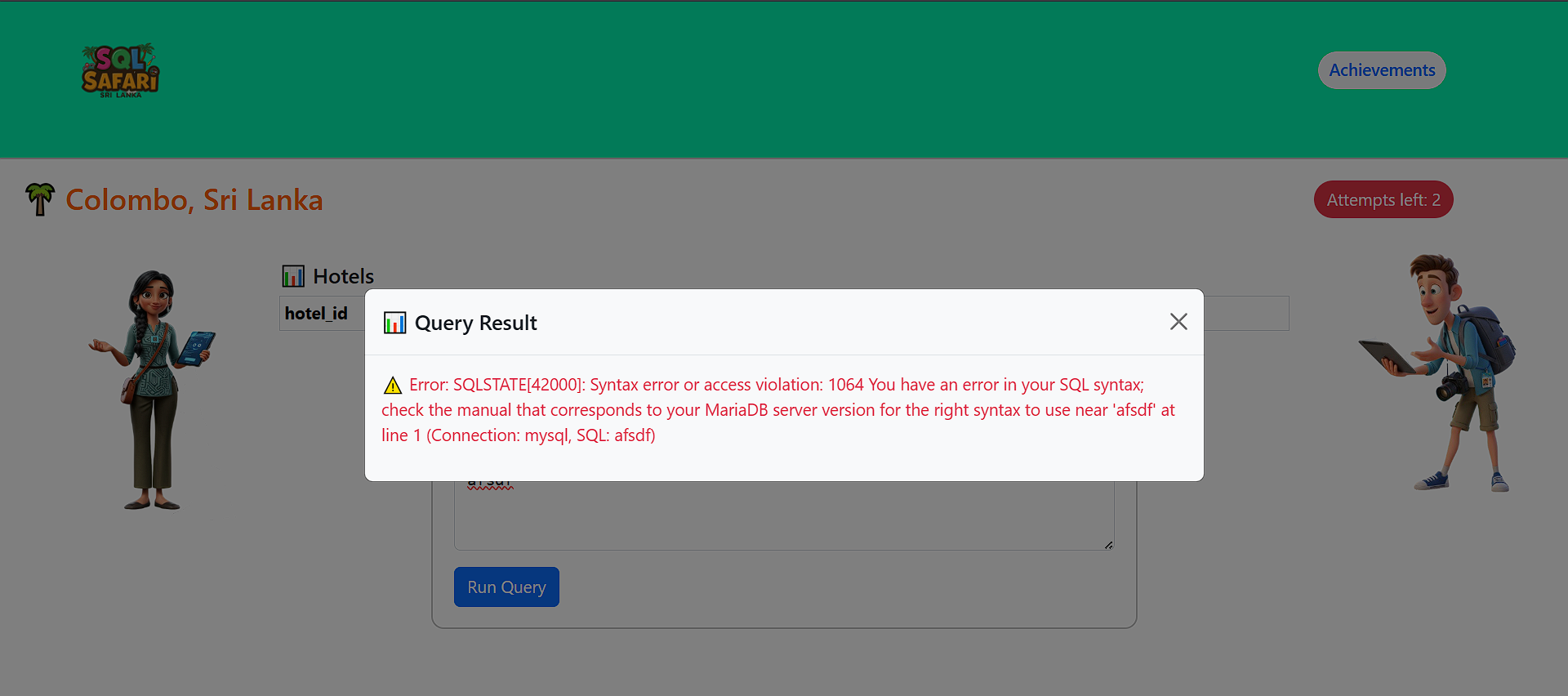
* Learners are presented with SQL tasks.
* The page contains:
  + SQL Editor (text area to enter queries).
  + Run Button to execute queries.
  + Reference Table View showing the schema/data.

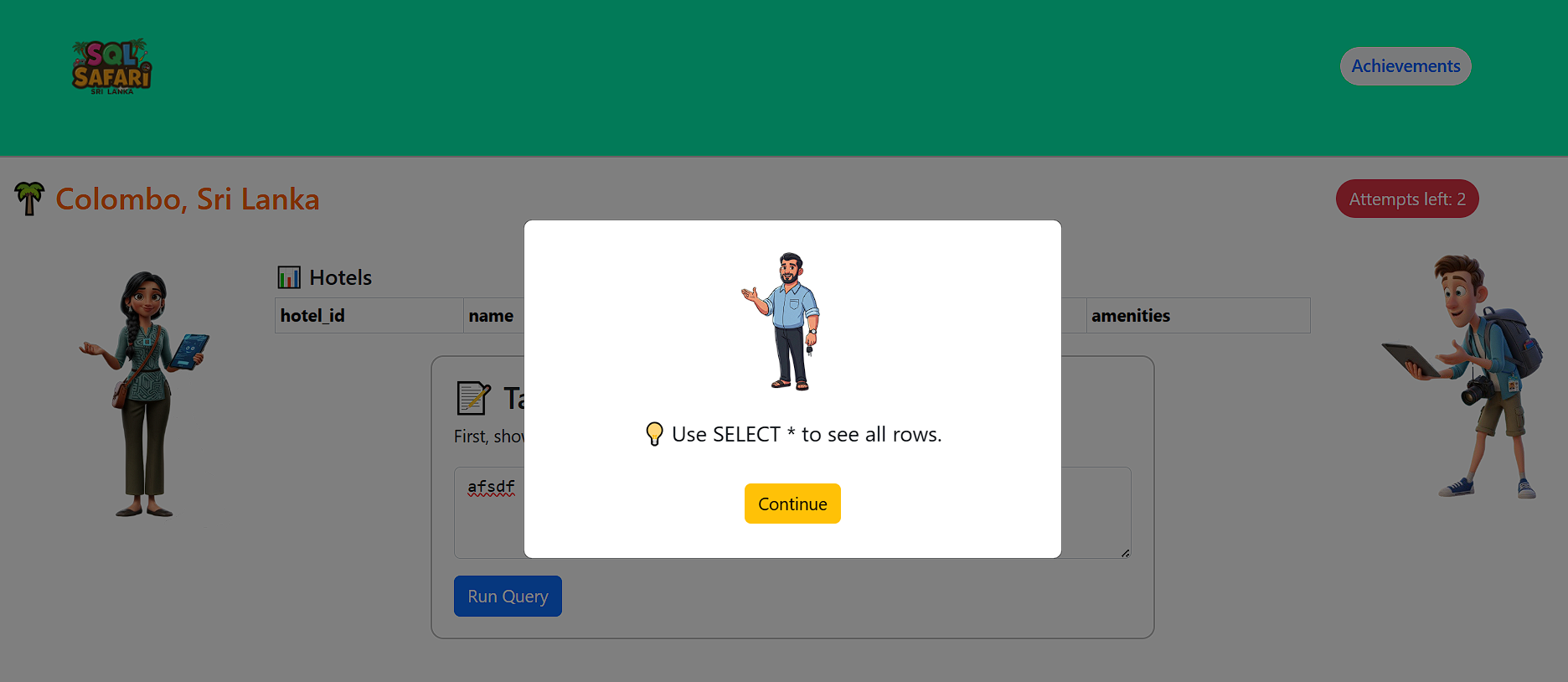
****

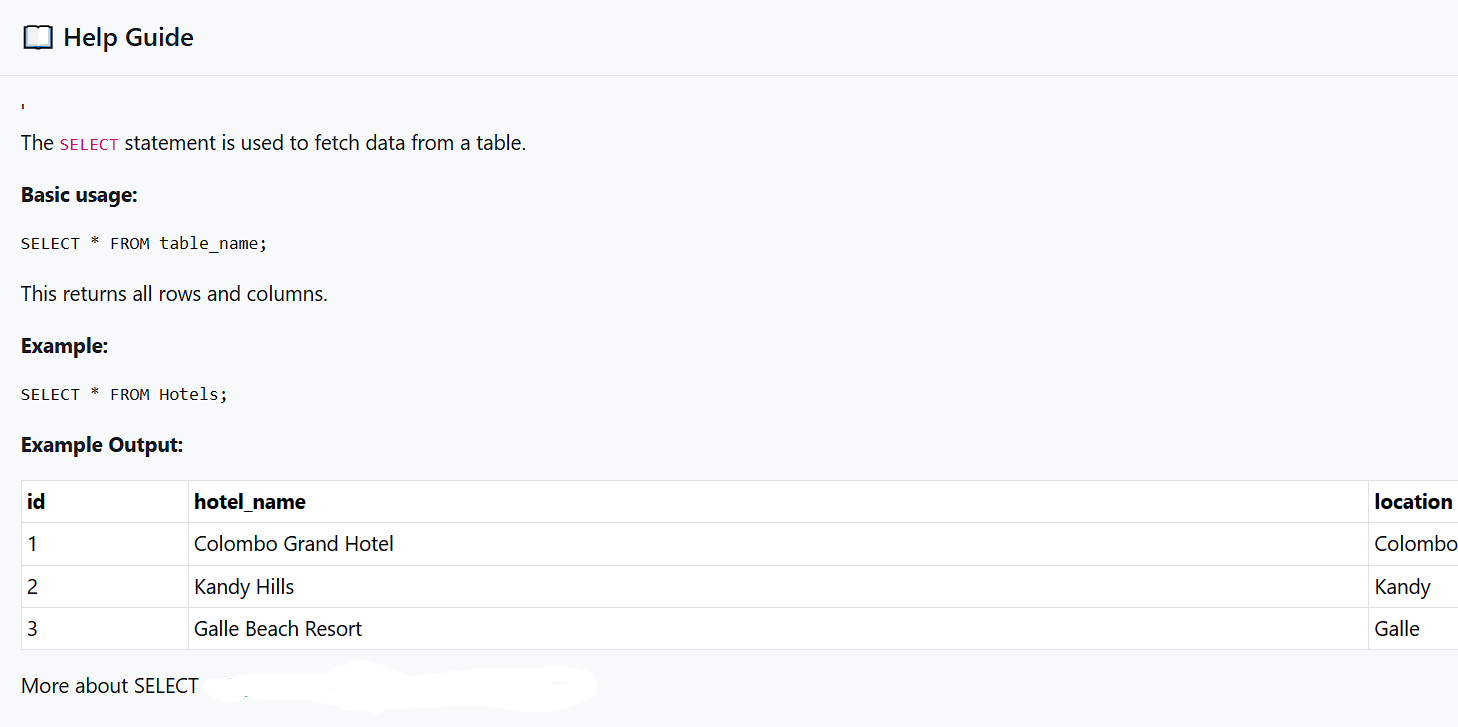


Query Validation & Feedback

* If the query is correct, a Success Popup Message is shown → the learner can move forward.
* If the query is incorrect, the system provides an Adaptive Hint.
* Learners have 3 attempts per task. After 3 failures:
  + A Help Box appears with:
    - Example queries.
    - A link to an external SQL tutorial for extra practice.



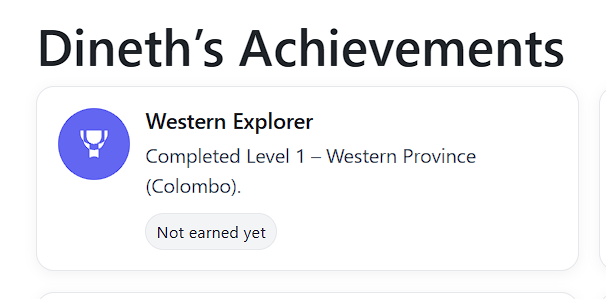




Level Completion & Rewards

* Each level contains 3 tasks.
* Upon completion:
  + A Certificate of Completion is auto-generated and emailed to the learner.
  + An Achievement Badge is unlocked and displayed in the Achievements View.

****

****

****

### 6.3.2 Province-Level Game Flow

| **Province** | **SQL Concept** | **Example Challenge** | **Game Mechanic** |
| --- | --- | --- | --- |
| Western (Colombo) | SELECT \* | View all hotels in Colombo | SQL Editor Intro |
| Central (Kandy) | WHERE | Find hotels under Rs. 5000 | Hints + Conditions |
| Southern (Galle) | ORDER BY | Sort hotels by price | Leaderboard Challenges |
| Northern (Jaffna) | JOIN | Show tourists with bookings | Schema Viewer + Puzzle |
| Eastern (Batticaloa) | GROUP BY | Count bookings per city | Pie Chart Visualization |
| North Central (Anuradhapura) | HAVING | Cities with >3 hotels | Budget Bonus System |
| Uva (Badulla) | Subqueries | Find hotels below average price | Query Debugger |
| Sabaragamuwa (Ratnapura) | DML | Add new hotel, update booking | Table Editor Sandbox |
| North Western (Kurunegala) | Views & Constraints | Final tourism report, views, constraints | Final Boss + Certification |

### 6.3.3 Example Scene: Jaffna – The Data Pairing Puzzle

* Objective: Join TOURISTS and BOOKINGS tables to match names and check-in dates.
* Adaptive Hint: Remind to use table aliases.
* Bonus Challenge: Filter tourists >30 years with bookings after July 10, 2025.
* Reward: Unlock “Jaffna Codex” with JOIN tutorials, diagrams, and a LEFT JOIN side quest.

### 6.3.4 Rewards & Gamification

* Badges: “Query Beginner”, “Sorting Explorer”, “SQL Master Explorer”
* Certificates: Completion certificate after final boss

### 6.3.5 Evaluation & Continuous Improvement

* Feedback: Google Forms
* Data Analytics: Dropout rates, most failed levels, success progression

# 7.0 Testing and Validation

## 7.1 Test Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Features** | **Objective** | **Scope** | **Test Type** |
| User Authentication | Verify login/register works correctly. | Test if users can sign up, log in, and maintain sessions securely. | Functional testing, Security testing |
| Auto-Scroll & Character Introduction | Ensure smooth introduction sequence on landing page. | Check auto-scroll animations, character display, and text narration. | Usability testing, UI testing |
| Get Started Navigation | Verify navigation to Section 1 introduction. | Test if clicking “Get Started” redirects correctly to Section 1 intro page. | Functional testing |
| SQL Editor & Query Execution | Validate SQL query input and execution. | Test if user queries are accepted, parsed, and executed against practice DB. | Functional testing, Integration testing |
| Success Feedback Popup | Ensure correct success message is displayed. | When the correct query is submitted, a popup should appear confirming success. | Functional testing, Usability testing |
| Hint System | Check hints for incorrect queries. | Test that incorrect queries trigger contextual hints after each wrong attempt. | Functional testing, Error Handling testing |
| Help Box (after 3 failures) | Ensure help content displays after 3 incorrect attempts. | Verify that the help box appears with simple queries and tutorial links. | Functional testing, Usability testing |
| Progress Tracking | Verify current level and task saving. | Ensure user’s progress (current task, attempts, completed levels) is stored correctly in DB. | Functional testing, Integration testing |
| Level Completion & Certificate | Validate reward system. | Test if certificate is emailed after completing 3 tasks, and badge appears in achievements. | Functional testing, Email testing |
| Achievements View | Ensure badges display correctly. | Verify unlocked achievements appear in the achievements view. | Functional testing, UI testing |
| Leaderboard | Validate leaderboard ranking system. | Check if scores update correctly and ranks are displayed in real-time. | Functional testing, Performance testing |
| Database Reset Option | Verify DB reset works. | Ensure that practice DB resets to initial state when reset button is clicked. | Functional testing, Integration testing |
| Error Handling – Wrong Queries | Ensure errors don’t break the game. | Verify system gracefully handles invalid SQL without crashing. | Error Handling testing |

## 7.2 Test Cases

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Feature** | **Test Case Description** | **Input Data** | **Expected Output** | **Actual Output** | **Pass/Fail** |
| TC-003 | Auto Intro Scroll | Verify homepage auto-scroll introduces characters | Visit homepage | Character intro displayed with smooth scroll | As expected | Pass |
| TC-004 | Get Started Redirect | Verify clicking **Get Started** goes to Section 1 intro | Click button | Redirected to Section 1 | As expected | Pass |
| TC-005 | Section Intro | Verify characters and story displayed in section intro | Navigate to section | Storyline + dialogues shown | As expected | Pass |
| TC-006 | Submit SQL Query | Verify correct query execution | SELECT \* FROM Hotels; | Success popup | As expected | Pass |
| TC-007 | Submit SQL Query | Verify incorrect query triggers hint | Wrong SQL query | Hint message displayed | As expected | Pass |
| TC-008 | Attempt Limit | Verify 3 incorrect attempts trigger help box | 3 wrong queries | Help box appears with tutorial link | As expected | Pass |
| TC-009 | Reset DB | Verify learner can reset practice DB | Click reset button | DB restored to default state | As expected | Pass |
| TC-010 | Achievements | Verify learner earns achievement after completing level | Complete Level 1 | Achievement unlocked, badge visible | As expected | Pass |
| TC-011 | Email Certificate | Verify certificate sent after completing 3 tasks | Complete 3 tasks | Email with certificate delivered | As expected | Pass |
| TC-012 | View Achievements | Verify learner can view unlocked achievements | Navigate to Achievements | List of earned badges displayed | As expected | Pass |
| TC-013 | Leaderboard | Verify leaderboard displays correctly | Multiple users with scores | Leaderboard sorted by XP/points | As expected | Pass |
| TC-014 | Progress Tracking | Verify progress is saved after tasks | Complete 2 tasks | Current level & task updated in DB | As expected | Pass |
| TC-018 | SQL Injection Prevention | Verify SQL injection is blocked | DROP TABLE Hotels; | Query blocked, error handled | As expected | Pass |

# 8.0 Conclusion

## 8.1 Conclusion

The SQL Safari project demonstrates an innovative approach to teaching database concepts and SQL through an interactive, gamified learning platform. By combining storytelling, character-based guidance, and practical query execution, the system provides learners with a dynamic environment that goes beyond traditional tutorials.

Key features such as task-based learning, real-time SQL validation, hints and help after limited attempts, and achievement-based rewards ensure that learners remain engaged and motivated throughout the experience. The inclusion of leaderboards, progress tracking, and certificate generation adds a competitive and rewarding dimension, making the platform suitable for both individual learners and classroom use.

From a technical perspective, the project successfully integrates a web-based interface, a Laravel-powered backend, and a sandbox database that ensures safe SQL execution. Instructors and administrators also benefit from management and reporting tools, enabling them to track learner progress, analyze performance, and maintain the platform effectively.

The SQL Safari project not only supports skill development in structured query language but also promotes experiential learning, where students learn by doing rather than rote memorization. By blending education with gamification, the project addresses common challenges in database learning, such as lack of engagement, difficulty in query understanding, and the gap between theory and practice.

Overall, SQL Safari stands as a pioneering educational tool that leverages modern web technologies and gamification strategies to enhance database education. With further development and scalability, it has the potential to be adopted in academic institutions, online learning environments, and self-study platforms as a comprehensive solution for teaching SQL in an engaging and effective manner.

## 8.2 Future Recommendations

**Expansion of Game Levels and Scenarios**

Currently, the system includes a limited number of provinces and tasks. Future work can focus on expanding the storyline to include additional provinces, more challenging SQL problems, and advanced database topics (e.g., triggers, stored procedures). This would keep learners engaged while supporting both beginners and advanced students.

**Adaptive Learning Pathways**

Introduce an adaptive mechanism that adjusts the difficulty of tasks based on learner performance. Learners who perform well can be fast-tracked to advanced queries, while those struggling can be provided with simpler challenges and more detailed hints. This would ensure personalized learning and better knowledge retention.

**Enhanced Feedback and Hints System**

Expand the feedback system to provide deeper explanations, visual query execution plans, or query correction suggestions instead of simple error prompts. This would help learners understand not only *what went wrong* but also *how to fix it*.

**Gamification Upgrades**

Additional gamification elements such as streaks, daily challenges, time-based competitions, and collaborative multiplayer quizzes can be introduced. These features would increase engagement and motivate learners to practice SQL regularly.

**Integration with Learning Management Systems (LMS)**

Provide seamless integration with platforms such as Moodle or Google Classroom via LTI standards. This would allow instructors to incorporate SQL Safari into formal academic curricula, track student progress, and grade automatically.

**Mobile Application Development**

A mobile version of SQL Safari could be developed for Android and iOS to enable learning on the go. Push notifications for daily challenges and offline practice options would further enhance accessibility.

**Certificate and Badge Verification System**

Introduce a verifiable certificate system, possibly using blockchain or digital verification, so that learners can share their achievements with universities and employers. This would add credibility and long-term value to the platform.

**Community and Collaboration Features**

Build a forum or discussion board where learners can share SQL queries, discuss alternative solutions, and help each other. A peer-to-peer support system would foster collaborative learning and create a strong SQL learning community.

**Advanced Analytics for Instructors**

Enhance instructor dashboards with detailed analytics such as common mistakes, average query-solving time, and difficulty heatmaps. This would allow instructors to identify knowledge gaps and tailor teaching strategies accordingly.

**Support for Multiple Database Engines**

While the current platform uses MySQL, future expansions can allow learners to practice on other popular systems like PostgreSQL, Oracle, and SQL Server. This would make the platform more versatile and industry relevant.

**Accessibility and Inclusivity Enhancements**

Ensure the platform is accessible to all learners, including those with disabilities. Features such as screen reader compatibility, keyboard navigation, and simplified UI modes would broaden the user base.

**Cloud Scalability and Multi-User Load Handling**

As user adoption grows, implementing cloud-based scaling with load balancing would be critical. Future work should focus on ensuring the system can handle thousands of concurrent learners without performance degradation.

## 8.3 Lessons Learned

**Importance of User-Centered Design**

From the beginning of SQL Safari, it became clear that user experience (UX) plays a critical role in learning engagement. Designing the interface to be simple, interactive, and game-like helped students stay motivated. A key lesson was that educational tools must minimize complexity and focus on clarity to support learners of all levels.

**Challenges of Query Validation and Feedback**

Implementing real-time SQL query validation was more complex than expected. Ensuring that queries were both safe (e.g., preventing DROP/DELETE misuse) and flexible enough to accept variations taught us the importance of building robust validation logic. Additionally, providing meaningful hints instead of generic error messages significantly improved learning outcomes.

**Balancing Gamification with Education**

Gamifying SQL learning added fun and motivation, but it was equally important to ensure that gameplay did not overshadow the educational purpose. Striking a balance between rewards (badges, certificates) and meaningful practice reinforced the need to design with pedagogy and motivation in mind.

**Iterative Development and Testing**

Early prototypes revealed several usability issues, such as unclear instructions and hint overuse. Iterative development with continuous testing allowed these issues to be corrected before full deployment. This experience reinforced the importance of agile practices, especially for educational applications.

**Real-Time User Feedback is Essential**

Feedback from learners highlighted areas where tasks were too difficult, hints were insufficient, or the storyline could be more engaging. Direct user feedback helped shape improvements such as providing additional help after three failed attempts and offering links to tutorials.

Scalability and Performance Considerations

As the platform grew, handling multiple learners executing queries simultaneously stressed the practice database. Optimizing SQL sandbox execution and caching common datasets were necessary to maintain smooth performance. A valuable lesson was to anticipate scalability needs early in the project lifecycle.

**The Role of Storytelling in Engagement**

Introducing characters and provincial storylines added a narrative element that made learning SQL more engaging. The lesson learned here is that storytelling can transform abstract technical content into a relatable and enjoyable learning journey.

**Collaboration Between Teams**

Coordinating between frontend (UI/UX), backend (query execution), and content creators (SQL tasks and hints) was essential. Continuous communication and documentation ensured that each component fit into the overall system seamlessly, teaching the value of interdisciplinary teamwork.

**Security and Safe Query Execution**

One major challenge was ensuring that learners could practice freely without harming the database. Implementing sandboxed databases, read-only datasets, and reset features provided a safe learning environment. This reinforced the lesson that security is as important in educational tools as in enterprise systems.

**Need for Continuous Content Updates**

Learners requested new levels, more scenarios, and updated tasks to keep engagement high. This highlighted the importance of continuously evolving content to match learner expectations, ensuring that the platform remains relevant and useful over time.

# Gantt Chart

# References

Seaborn, K. & Fels, D.I. (2015) ‘Gamification in theory and action: A survey’, *International Journal of Human-Computer Studies*, 74, pp. 14–31. doi:10.1016/j.ijhcs.2014.09.006.

Domínguez, A., Saenz-de-Navarrete, J., de-Marcos, L., Fernández-Sanz, L., Pagés, C. & Martínez-Herráiz, J.-J. (2013) ‘Gamifying learning experiences: Practical implications and outcomes’, *Computers & Education*, 63, pp. 380–392. doi:10.1016/j.compedu.2012.12.020.

Seyed-Abbassi, B. (1993) ‘A SQL project as a learning method in a database course’, *Proceedings of the 1993 ACM Conference on Computer Personnel Research (SIGCPR ’93)*. New York: ACM, pp. 291–297. doi:10.1145/158011.158238.

Hsu, T.-C., Chang, S.-C. & Hung, Y.-T. (2018) ‘How to learn and how to teach computational thinking: Suggestions based on a review of the literature’, *Computers & Education*, 126, pp. 296–310. doi:10.1016/j.compedu.2018.07.004.

Csikszentmihalyi, M. (1990) *Flow: The Psychology of Optimal Experience*. New York: Harper & Row.

Andersen, E., Liu, Y.E., Apter, E., Boucher-Genesse, F. & Popović, Z. (2013) ‘Gameplay analysis for adaptive difficulty in educational games’, *Computers & Education*, 68, pp. 570–581. doi:10.1016/j.compedu.2013.02.017.

Strieder, R.B., Zajac, M., Romero, M. & de Souza, C.R. (2020) ‘Adaptive AI agents in educational games: Engagement and cognitive load’, *Education and Information Technologies*, 25(6), pp. 5441–5460. doi:10.1007/s10639-020-10219-5.

Kickmeier-Rust, M.D., Mattheiss, E., Steiner, C.M. & Albert, D. (2011) ‘A psycho-pedagogical framework for multi-adaptive educational games’, *International Journal of Game-Based Learning*, 1(1), pp. 45–58. doi:10.4018/ijgbl.2011010104.

Darwesh, A.M. (2016) ‘Serious games in adaptive learning’, *Journal of University of Human Development*, 2(4), pp. 418–423. doi:10.21928/juhd.v2n4y2016.pp418-423.

Ifenthaler, D. & Yau, J.Y.-K. (2020) ‘Utilising learning analytics for study success: Reflections on current empirical findings’, *Interactive Technology and Smart Education*, 17(4), pp. 385–398. doi:10.1108/ITSE-07-2020-0110.

Gee, J.P. (2003) *What Video Games Have to Teach Us About Learning and Literacy*. New York: Palgrave Macmillan.

Egenfeldt-Nielsen, S. (2006) ‘Overview of research on the educational use of video games’, *Nordic Journal of Digital Literacy*, 1(3), pp. 184–213.

Brusilovsky, P. & Millán, E. (2007) ‘User models for adaptive hypermedia and adaptive educational systems’, in Brusilovsky, P., Kobsa, A. & Nejdl, W. (eds.) *The Adaptive Web: Methods and Strategies of Web Personalization*. Berlin: Springer, pp. 3–53. doi:10.1007/978-3-540-72079-9\_1.

Hamari, J., Koivisto, J. & Sarsa, H. (2014) ‘Does gamification work? – A literature review of empirical studies on gamification’, *Proceedings of the 47th Hawaii International Conference on System Sciences (HICSS)*, pp. 3025–3034. doi:10.1109/HICSS.2014.377.

Papastergiou, M. (2009) ‘Digital game-based learning in high school computer science education: Impact on educational effectiveness and student motivation’, *Computers & Education*, 52(1), pp. 1–12. doi:10.1016/j.compedu.2008.06.004.

Anderson, L.W. & Krathwohl, D.R. (eds.) (2001) *A taxonomy for learning, teaching, and assessing: A revision of Bloom’s Taxonomy of Educational Objectives*. New York: Longman.

Schunk, D.H. (2012) *Learning Theories: An Educational Perspective*. 6th edn. Boston: Pearson.

Ryan, R.M. & Deci, E.L. (2000) ‘Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being’, *American Psychologist*, 55(1), pp. 68–78. doi:10.1037/0003-066X.55.1.68.

He, C. & Park, J. (n.d.) *SQL Murder Mystery*. Northwestern University Knight Lab. Available at: [https://mystery.knightlab.com](https://mystery.knightlab.com?utm_source=chatgpt.com) (Accessed 1 August 2025).

Schildgen, J. (2014) ‘SQL Island: An adventure game to learn the database language SQL’, *Proceedings of the 8th European Conference on Games Based Learning (ECGBL 2014)*, Berlin. Available at: [https://www.researchgate.net/publication/267094856](https://www.researchgate.net/publication/267094856?utm_source=chatgpt.com) (Accessed 1 August 2025).

Mode Analytics (n.d.) *SQL Tutorial*. Available at: [https://mode.com/sql-tutorial/](https://mode.com/sql-tutorial/?utm_source=chatgpt.com) (Accessed 1 August 2025).

W3Schools (n.d.) *SQL Tutorial*. Available at: [https://www.w3schools.com/sql/](https://www.w3schools.com/sql/?utm_source=chatgpt.com) (Accessed 1 August 2025).

DataCamp (n.d.) *SQL Courses*. Available at: https://www.datacamp.com (Accessed 1 August 2025).

Codewars (n.d.) *SQL Kata*. Available at: [https://www.codewars.com/kata/search/sql](https://www.codewars.com/kata/search/sql?utm_source=chatgpt.com) (Accessed 1 August 2025).

LeetCode (n.d.) *Database (SQL) Problems*. Available at: [https://leetcode.com/problemset/database/](https://leetcode.com/problemset/database/?utm_source=chatgpt.com) (Accessed 1 August 2025).

Kaggle (n.d.) *Learn SQL Courses*. Available at: [https://www.kaggle.com/learn](https://www.kaggle.com/learn?utm_source=chatgpt.com) (Accessed 1 August 2025).

SQL Fiddle (n.d.) *SQLFiddle.com*. Available at: [http://sqlfiddle.com](http://sqlfiddle.com?utm_source=chatgpt.com) (Accessed 1 August 2025).

Plass, J.L., Homer, B.D. & Kinzer, C.K. (2015) *Foundations of game-based learning*. Educational Psychologist, 50(4), pp. 258–283. doi:10.1080/00461520.2015.1122533.

Van Eck, R. (2006) *Digital game-based learning: It’s not just the digital natives who are restless*. EDUCAUSE Review, 41(2), pp. 16–30.

Deterding, S., Dixon, D., Khaled, R. & Nacke, L. (2011) *From game design elements to gamefulness: Defining “gamification”*. Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments, pp. 9–15. doi:10.1145/2181037.2181040.

Silva, R., Teixeira, J., Fernandes, A., Oliveira, E. & Monteiro, A. (2022) *Dungeon Code: Evaluating the usability of an educational game for programming logic*. Education and Information Technologies, 27(6), pp. 8337–8360. doi:10.1007/s10639-022-10907-1.

TopDevelopers (n.d.) Mobile game development process. Available at: [https://www.topdevelopers.co/blog/mobile-game-development-process/](https://www.topdevelopers.co/blog/mobile-game-development-process/?utm_source=chatgpt.com) (Accessed 1 August 2025).

ResearchGate (2015) Spiral Model for game development. Available at: [https://www.researchgate.net/publication/274174154](https://www.researchgate.net/publication/274174154?utm_source=chatgpt.com) (Accessed 1 August 2025).

Medium (n.d.) Agile game development. Available at: [https://miro.medium.com/v2/resize:fit:1100/format:webp/0\*RziwSIdYb2capKWN.png](https://miro.medium.com/v2/resize:fit:1100/format:webp/0*RziwSIdYb2capKWN.png?utm_source=chatgpt.com) (Accessed 1 August 2025).

Carroll, J. (2019) Using the MDA framework as an approach to game design. Medium. Available at: [https://medium.com/@jenny\_carroll/using-the-mda-framework-as-an-approach-to-game-design-9568569cb7d](https://medium.com/@jenny_carroll/using-the-mda-framework-as-an-approach-to-game-design-9568569cb7d?utm_source=chatgpt.com) (Accessed 1 August 2025).

ResearchGate (2024) Development of the ADDIE model system. Available at: [https://www.researchgate.net/publication/378655023](https://www.researchgate.net/publication/378655023?utm_source=chatgpt.com) (Accessed 1 August 2025).

Stringer, K. (2008) A potential framework for an adaptive game system. ResearchGate. Available at: [https://www.researchgate.net/publication/228636408](https://www.researchgate.net/publication/228636408?utm_source=chatgpt.com) (Accessed 1 August 2025).

AIHR (n.d.) ADDIE model. Available at: [https://www.aihr.com/wp-content/uploads/ADDIE-Model-New.png](https://www.aihr.com/wp-content/uploads/ADDIE-Model-New.png?utm_source=chatgpt.com) (Accessed 1 August 2025).

# Appendix 1

## Questionnaire and answers

## Source Codes

# Appendix 2

## Supervisor meeting log sheets