Gamified website for the Computer Science e-learning environment in Hong Kong

COMP4801 Final Year Project
Final Report
Lam Chun Ho 3035685184
fyp22003

April 18, 2023
Abstract

Although access to online resources and guidance from seniors can be helpful for students in preparing for assessments and exams, this may not be enough to foster the necessary interactions and motivation in the Computer Science e-learning environment in Hong Kong. Assignments and exams often reuse questions, and there are few Computer Science forums where students can discuss with each other. For example, teachers are mostly the respondents to the questions in Moodle forum. Students may easily find solutions online and miss out on interacting with others and essential knowledge, which can harm their Computer Science knowledge and future job performance in such an unmotivated learning environment. The studies experimented with how students perform by modifying Moodle or developing a website. The studies show that students can be motivated to learn Computer Science with gamification. Based on those studies, the project team designed and built common game elements for the gamified website. This project developed a gamified website that applies game elements to non-game (learning) contexts.

The gamified website has four main features: Coding Challenge, Contest, Course, and Discussion Forum. Coding Challenge allows students to practice their coding skills and receive feedback from other students; Contests pit students against each other in timed challenges; Courses provide structured lessons and tests; Forum encourages students to collaborate, ask questions, and code review others’ code.

Students can get points (experiences) and badges in these contexts. The gamified website also provides a unique feature: code review. It provides a way to review the attempts of other students. It will also rank students into three categories: Adventurer, Learner, and Evil. LeetCode, a platform to prepare for technical interviews and expand knowledge, provides similar features to the gamified website. Unlike LeetCode, this website is for learning programming, not just practicing for interviews.

The project team designed and created one website, two Content Management Systems, and two services. The project team built the gamified website using the Next.js framework. The project team created two Content Management Systems with two different frameworks. The first uses Strapi to manage the non-game content; The second uses Express and Next.js to handle the game elements in gamification. To evaluate the coding challenge, the project team developed a simple service in Golang that checks whether the user has passed all test cases. A model from Kaggle can classify the quality of a post in the forum. The project team used Flask to calculate (experience) points for students.

Due to time constraints, this project does not evaluate the effectiveness of the gamified website and develop a better AI for classifying the quality of a post. This project provides a possible prototype for the Computer Science e-learning environment. For example, developers can modify this gamified website for various Computer Science topics. Further actions will be examining the effectiveness of the gamified website, adding more features, and implementing different AI models. This project also does not build one of the Content Management Systems and Remote Code Execution System from scratch. For example, the project team uses open-source products instead, which are Strapi and Judge0, such that the project team can focus on designing the way of applying game elements to non-game content.
Acknowledgments

I would like to thank my supervisors Dr. Choi, Yi King and Dr. Chim, Tat Wing for supervising my project. Thank you for guiding the directions of the project.
## Contents

Abstract ................................................................. i  
Acknowledgements ................................................... ii  
List of Tables ........................................................... v  
List of Figures .......................................................... vii

1 Introduction 1
   1.1 Background ...................................................... 1  
   1.2 Problem Statement ............................................... 2  
   1.3 Objectives ....................................................... 2  
   1.4 Project Scope .................................................... 2  
   1.5 Outline .......................................................... 3

2 Literature Review 4
   2.1 E-learning and Gamification ...................................... 4  
   2.2 Design Principles for Gamification .............................. 4  
   2.3 Gamified Implementations ....................................... 5  
   2.4 Gamification Experiments ...................................... 5

3 Methodology 6
   3.1 Project Structure ................................................ 6  
   3.2 Compared to Modifying Moodle .................................. 8  
   3.3 Game Elements .................................................. 8  
      3.3.1 Profile Page ............................................... 9  
   3.4 Non-game Content ............................................... 9  
      3.4.1 Forum .................................................... 9  
      3.4.2 Coding Challenges ....................................... 10  
      3.4.3 Coding Contests and Courses ............................ 10  
   3.5 Comparison to Existing Websites ............................... 11

4 Result 13
   4.1 UI Design ....................................................... 13  
   4.2 Gamified Website ............................................... 13  
      4.2.1 Coding Challenge ....................................... 14  
      4.2.2 Forum .................................................. 15  
      4.2.3 Courses ................................................ 16  
      4.2.4 Contests ................................................ 16  
      4.2.5 Leaderboard and User Profile Page ....................... 16  
   4.3 Game Elements ................................................ 17  
      4.3.1 Experience Points and Leveling ......................... 17
List of Tables

3.1 Brief comparison to Two CMS and one Strapi CMS ...................................... 7
3.2 Brief comparison to two CMS of the project ............................................. 7
3.3 Brief comparison to modifying Moodle .................................................. 8
3.4 Brief comparison between this website and LeetCode and HackerRank ...... 12

4.1 Condition of getting different points ..................................................... 18
## List of Figures

3.1 Project Structure ........................................... 6
3.2 Examples of feedback: Immediate(left); Progress(middle); Statistic(right) 8
3.3 Example Content in the User Profile, captured from LeetCode ........... 9
3.4 Use cases of the website .................................. 10
3.5 Example of Code review section .................................. 10
3.6 Flow of the coding challenge .................................. 11
4.1 Site Diagram of the gamified website .................................. 13
4.2 Lifeline diagram of judging of submission .................................. 14
4.3 Structure of the Python template .................................. 15
4.4 Flow of run() function in Coding Challenge .......................... 15
4.5 Flow of a contest ........................................... 17
4.6 The toast to indicate users got experience points. .................. 18
4.7 The dialog when users got experience points. ..................... 18
4.8 Images used for badges ....................................... 19
4.9 Example dialog when receiving a badge .................................. 19
4.10 Database Schema for Strapi CMS .................................. 20
4.11 Structure of the gamification CMS .................................. 21
4.12 Database Schema of gamification CMS .......................... 21
4.13 The state, parameter, and functions in usePlayer .................. 22
4.14 The state, parameter, and functions in useMonster .................. 22
4.15 Flow of worker ........................................... 23
C.1 Home Page of the website ..................................... 31
C.2 Post List Page of the website .................................. 32
C.3 Post Details of the website ...................................... 33
C.4 Login Page of the website ....................................... 33
C.5 Register Page of the website .................................... 34
C.6 Add Post Page of the website .................................... 34
C.7 Challenge Page of the website .................................... 35
C.8 LeaderBoard Page of the website .................................. 35
C.9 Contest list Page of the website .................................. 36
C.10 Contest Page of the website ...................................... 36
C.11 Course List Page of the website .................................. 36
C.12 User Profile Page of the website .................................. 37
D.1 The page of listing Challenges ..................................... 39
D.2 The detail page of coding challenge .................................. 39
D.3 The detail page of coding challenge .................................. 39
D.4 The pages of listing Contest ..................................... 40
List of Abbreviations

CMS - Content Management System

CS - Computer Science

CRUD - Create, Read, Update, and Delete

HK - Hong Kong

UI - User Interface
Chapter 1

Introduction

For CS students, there are many challenging school assessments. Students can find many online resources without effort. For example, it is easy for students to find tutorials on StackOverflow and YouTube. When it comes to learning, however, these resources are two-sided blades for them. Students may lack the motivation to learn and understand the materials. The following discusses the relationships between the background, problem statement, objectives, scope, and outline.

1.1 Background

There are many overseas forums for discussing different knowledge of CS. For example, students can find solutions for coding in StackOverflow and mathematics problems in Mathematics Stack Exchange. In China, many forums are in mixed languages, English and their native language, which helps different people to learn. Statistics in StackOverflow 2021 developer server have shown that HK only has 0.24% in the Geography category, which is small compared to China (1.27%) and France (3.25%) [7]. These overseas forums even have active discussions about job seeking, ranging from start-ups to famous companies.

HK students have strong English skills. They may visit course-based forums in Moodle, local large-scale forums like HKGolden and LIHKG, and overseas forums. However, there are no large-scale discussion forums specifically about CS. In contrast to overseas, there are few discussions about CS knowledge in course-based and local, compared to overseas. Students may ask questions in Moodle forums when working on the assessments and revising exam content. However, most respondents are teachers in the course-based forum.

Furthermore, the course materials and scope are sometimes similar to previous years because tutors may reference the internet or previous assessments when designing the assignments. Students can finish them effortlessly by seeking resources from seniors or the internet. They may effortlessly copy a workable solution to the questions from these resources. The attendances of lectures and tutorials may be few such that students may not fully understand the concepts. In 2022, OpenAI also introduced ChatGPT allows students to find solutions directly. With the appearance of ChatGPT, the situation becomes more serious. According to the BestColleges survey [8], 17% of students used AI to complete assignments without edits. Students may just interact with the AI to get the solutions and directions without efforts. As a result, students may not understand the materials clearly and lack interactions with others.
Nowadays, e-learning is a future trend. Many studies evaluate the effectiveness and attempt the possible implementations between e-learning and gamification. Based on the review in chapter 2, gamification can potentially motivate the learning environment of Computer Science and academic participation of students and may replace the traditional face-to-face lecture.

1.2 Problem Statement

The above shows that the CS learning environment in Hong Kong is unmotivating and inactive. Students may lack engagement in learning complex concepts and acquire the answers directly from the sources. They may also lack participation in learning and discussion in such an environment. As a result, students may be weak in future programming works and interviews after graduation. They may need more time get in famous companies like Google. It is because they do not clearly understand the materials. Thus, this project aims to design and implement a gamified website for the HK CS e-learning environment with the following objectives.

1.3 Objectives

Overall, this project creates a gamified website for university education in programming. The website applies game elements to non-game content. Students can gain motivation to learn CS through gamification. Students are leveled and ranked into three categories: Adventurer, Learner, and Evil. They can obtain points and badges by joining contests, completing coding challenges, finishing courses, and discussing in the forum. The website provides feedback to students, and students are leveled and ranked by their experiences. Chapter 3 will discuss the details. By discussion, thinking, and contributing, students can learn and reinforce their knowledge and prepare for future interviews and careers in this e-learning environment.

Furthermore, this project can be a prototype or example for the future. The approaches and structure can be a direction for the future development of other gamified websites. This project only works on the programming sections. The way of gamification progress can be an example to reduce the development process and logic for this project or other websites in the future.

1.4 Project Scope

Given the time limit, the project focuses on Programming. It is because programming is essential in future career and academic learning. This project designs a gamified website based on the findings in other studies, where it focuses on learning to code and motivating the learning environment. The website does not develop for a single course but for all CS students to use it. This project does not include research on AI or ML built into the project. For example, if the project team gives experience to students by using the quality predictor of the reply to a post. Besides, this project does not collect real-life data and research the effectiveness of the design. Moreover, the website may not contain too attractive designs on layout and all functionalities and develop some functions (for example, the chatting system [2]).
1.5 Outline

The project team implemented the gamified website. The following chapters discuss the design and implementation. The structure of this report is as follows: chapter 2, the literature review, describes the idea behind the product and explains the definitions of terms, designs, implementations, and experiences in the research. Chapter 3 is the methodology, which explains the approaches used in the project, including different tools and frameworks to develop the features for the website. In the end, chapter 4 discusses the progress result of the project.
Chapter 2

Literature Review

This chapter describes the idea behind this product with the studies on gamification in e-learning. The following discusses a review of the relationship between e-learning and gamification. There are some reviews on the designs, implementations, and experiments in gamification.

2.1 E-learning and Gamification

E-learning is a method for students to learn everywhere and every time within the time limit, which gains higher control and freedom over their learning progress [3]. According to Amriani et al. [1], e-learning should be a tool to encourage participation in the traditional class environment and for students to learn actively.

Gamification is an idea that applies game elements to non-game content [4],[6]. The existing examples are Nike+ Run Club, eBay, and Todoist, which used gamification to produce motivations and connections between customers and the company [4]. There are some implementations of gamification in education; a famous language learning app DuoLingo also implemented gamification. Studies [1]-[3],[5]-[6] also add gamification designs and strategies to the e-learning environment to examine the effects on motivating the learning environment, their performances, participation, and more. Students will gain more motivation and engagement with the given missions by combining gamification and e-learning [1].

2.2 Design Principles for Gamification

Game elements are taking an essential role in gamification. There are six common game elements: feedback, goals, badges, points, a leaderboard, and a level system [1]-[4], [5]-[6, Sec. 1]. These game elements follow the three aspects of gamification design [4] (see Appendix A). Moreover, the application should also integrate with classes and courses in education. For example, see [1]-[3],[5]-[6, Ch. 4, pp. 53-78]. Gamification implementations should also consider the self-determination theory [1],[5] (see Appendix B) because of its emphasis on their goals.
2.3 Gamified Implementations

Some researchers implemented gamification in e-learning in Moodle. For example [1],[3],[5]-[6, Ch. 11, pp. 238-260] modified Moodle and applied different game elements. There was also an implementation called PeerSpace that provides a collaborative learning environment that includes functions with online social networks [2].

2.4 Gamification Experiments

According to Poondej et al. [3], considering adding gamification strategies can motivate and engage students in learning. Most students in the gamification group had positive feedback on gamified education compared to the traditional education group [1]-[3],[5]. The gamification group had higher motivations than the control group [2],[3]. However, Amriani et al. [1] suggested that gamification does not imply evident effects on students’ participation but on their performance. The class materials had higher downloads for the experimental group, an average of 89 per week, and more than 65 downloads per week for the control group [5]. The experimental group in [6, Ch. 7, pp. 131-151] had a better academic performance than the control group, and the two groups had a similar score on motivational beliefs. Experiments [1]-[3],[5],[6, Sec. 2] showed that gamified environments gain positive effects, including motivation for learning and participating, and may also improve academic preferences. That shows the positive influence of gamification in e-learning, including better academic results and higher intention for learning. Therefore, this project will apply the game elements to the website. The project will implement common game elements to the courses, forum, coding contests, and coding challenges.
Chapter 3

Methodology

This chapter describes the methodology, including the non-game content, game elements, comparison to existing websites, and comparison to modifying Moodle. Overall, this project uses Next.js as the framework of front-end development. The project team will also use Strapi as the CMS and Judge0 as the Remote Code Executing System to reduce the development complexity and time.

3.1 Project Structure

Figure 3.1 shows the project structure. There are several services: Points Calculation, Gamification CMS, Strapi CMS, Judge Service, and Judge0. The project team uses Strapi and Judge0 as CMS for non-game content and Remote Code Execution System. The gamification CMS handles the game elements of the gamified website, which are badges, levels, and more. The judge service compiles the source code and returns the submission results of the coding challenges. Chapter 4 will discuss the details of their implementations.

Figure 3.1: Project Structure
Reasons for this structure

There are several reasons for this structure. Table 3.1 compares gamification CNS + Strapi CMS and one Strapi CMS. Strapi provides a dashboard for CRUD. With Strapi, the project team can instantly use the API without external coding which reduces the time of development. With the gamification CMS, developers can ignore how to create and use the plugins, services, and UI components of Strapi. The gamification CMS is customizable with any user interface and backend design. Moreover, there can be more services in the future. With this structure, the programming language does not restrict to JavaScript. For example, the Judge Service in Figure 3.1 uses Golang; Users can use the Python AI model via Python Flask.

<table>
<thead>
<tr>
<th>Two CMS</th>
<th>One Strapi CMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customizability</td>
<td>Yes, the gamification CMS is customizable with any user interface and backend design</td>
</tr>
<tr>
<td>Maintenance</td>
<td>Non-game content can be used when gamified CMS is under maintenance (not 100% unavailable)</td>
</tr>
</tbody>
</table>

Table 3.1: Brief comparison to Two CMS and one Strapi CMS

Table 3.2 compares two CMS. The usage of them is different: Strapi CMS to manage the non-game content so that it focuses on the ease of handling and monitoring the data of the user; the game element, which are the levels of the user, badges of users, and more, use gamification CMS to manage. Therefore, it focuses on customizability and scalability. The framework, node version, and database in gamification CMS are higher customizability and flexibility than Strapi.

<table>
<thead>
<tr>
<th>Strapi CMS</th>
<th>Gamification CMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage</td>
<td>Manage non-game content</td>
</tr>
<tr>
<td>Framework</td>
<td>100% JavaScript</td>
</tr>
<tr>
<td>Objective</td>
<td>Easy to create and maintain content like posts and user information</td>
</tr>
<tr>
<td>Node version</td>
<td>Restricted: v14, v16 for Strapi v4.0.x to v4.3.8, and v18 for Strapi v4.3.9 or above)</td>
</tr>
<tr>
<td>Database</td>
<td>MySQL / MariaDB / PostgreSQL / SQLite</td>
</tr>
</tbody>
</table>

Table 3.2: Brief comparison to two CMS of the project

The framework is customizable based on the requirements and knowledge of developers. Secondly, using two CMS can have separate maintenance. Users can
access the non-game content when gamified CMS is maintained. It provides a better user experience compared to one Strapi CMS.

3.2 Compared to Modifying Moodle

As mentioned in Chapter 2, some experiments chose to modify Moodle. However, the project team designed a gamified website from scratch. Table 3.3 compares creating this website and modifying Moodle. Creating a new website does not require specific language. With the project structure, the project website mainly uses TypeScript and not Javascript and PHP. TypeScript provides type safety for developers. With CMSs, this website is dynamic for different databases, SQL or NoSQL, while Moodle only uses SQL. This website is more than a course-based website. It aims to be a community for university CS students.

<table>
<thead>
<tr>
<th>Creating This Website</th>
<th>Modifying Moodle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language</td>
<td>Dynamic</td>
</tr>
<tr>
<td>Database</td>
<td>SQL or NoSQL</td>
</tr>
<tr>
<td>Use as</td>
<td>Community</td>
</tr>
<tr>
<td></td>
<td>Course-based</td>
</tr>
</tbody>
</table>

Table 3.3: Brief comparison to modifying Moodle

3.3 Game Elements

Levels, points, and badges can show the statistic, experience, progress, and achievements of students. The leveling of students indicates their experiences in different categories, including programming languages and more; The website can distribute the points to students, including progress points on courses, skills points on programming languages, and experience points on levels. Students can get points for their actions on the website; The badge system can create and distribute the badge to students, which can show the achievement and progress of students. For example, when students finish a course or have some contributions to the community, the system will award a badge to them.

The feedback is the visual element of this project. As shown in figure 3.2, there are three types of feedback: intermediate, progress, and statistics. Intermediate feedback shows simple feedback like earning some experience points, “You did it” and “Wrong answer.”; Progress feedback shows the progress of the course, the remaining experience for the next level, and more; Statistics feedback is a visual way to show the students’ performance. For example, students can find their rank in coding contests.

Figure 3.2: Examples of feedback: Immediate(left); Progress(middle); Statistic(right)
The leaderboard page is the second visual element of this project. Students are mainly leveled and ranked into three categories: Adventurer, Learner, and Evil. Adventurer is related to solving coding challenges; The learner is related to contributing to the community and finishing the tests in courses; Evil is related to posting coding challenges.

### 3.3.1 Profile Page

Figure 3.3 shows examples of the profile page. On the profile page of students, students can view their badges, skill points in different programming languages, and their rank in coding challenges. The Profile page shows the abilities and contributions of students. For example, the community statistics, the correctness of attempting coding challenges, the languages they used, and more.

![Community Stats](image)

Figure 3.3: Example Content in the User Profile, captured from LeetCode

### 3.4 Non-game Content

For all pages, it supports internationalization routing (i18n) for the translation of static content. It can translate non-game content with i18n. This technique provides a simple way to pick the correct language for the user. Figure 3.4 shows the use cases of the website. There are four features on the website. The following will discuss in detail.

#### 3.4.1 Forum

In general, students can discuss anything in the forum. This project also adds a new feature: code review. Students can reply to a post with a code review, which points out improvement of the post/source code (for example, figure 3.5). As a result, students can learn from others and understand their performance. The website only allows students who solved the same coding challenges to review the submissions from each other.
Students can get points by creating a new post or replying to a post. However, giving the same experience points to students is not fail because the reply content can be good or bad. Therefore, the project team uses an AI from Kaggle to classify whether it is a high-quality reply.

### 3.4.2 Coding Challenges

Figure 3.6 shows the flow of the coding challenge. There are two stages in a coding challenge, similar to having an interview. Firstly, the student should answer how they solve it in their own words, not in the programming language. This stage trains students to think about the direction and approach such that students can explain their idea clearly in interviews and exams. Secondly, the student can attempt to solve the coding challenge programmatically. After the student passes all test cases, it will automatically create a new post to the forum for reviewing by others and get some experience points on “Adventurer” and skill. If the student does not have life left, the page will refresh automatically after five seconds.

### 3.4.3 Coding Contests and Courses

The coding contests are similar to LeetCode, which provides several coding challenges to solve directly. Students are required to solve the questions within the time limits. In the end, students are ranked by their time used.
Figure 3.6: Flow of the coding challenge

Students can learn how to use different programming languages and methods. The project team decided to use multiple choice for simplicity. After finishing all questions, they should solve the specific coding challenge(s). On the course page, students can see their progress and the lessons on the course. The system will reward students with a badge after they finish a course.

### 3.5 Comparison to Existing Websites

Table 3.4 shows the comparison between this project and existing websites. Compared to LeetCode and HackerRank, this project focuses on education for university students only, not interview practice for any students. This website also restricts access to some posts mentioned in Chapter 3.4.1. For a session of attempting a coding challenge, this website has a limited chance of submission and contains several steps. In contrast, the existing websites have unlimited access to the problems and public access to all posts. Furthermore, courses in LeetCode only have questions for users to finish, but this project website also teaches users how to use different programming languages and algorithms.
<table>
<thead>
<tr>
<th></th>
<th>This Website</th>
<th>LeetCode &amp; HackerRank</th>
</tr>
</thead>
<tbody>
<tr>
<td>Focus</td>
<td>Education</td>
<td>Interview practicing</td>
</tr>
<tr>
<td>Target User</td>
<td>Students</td>
<td>Any users</td>
</tr>
<tr>
<td>Solving a coding challenge</td>
<td>Several steps</td>
<td>Direct submission</td>
</tr>
<tr>
<td>Given a chance in a session of coding challenge</td>
<td>Limited</td>
<td>Unlimited</td>
</tr>
<tr>
<td>Restricted access to some posts</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Courses</td>
<td>Teach and attempt</td>
<td>Attempt directly</td>
</tr>
</tbody>
</table>

Table 3.4: Brief comparison between this website and LeetCode and HackerRank
Chapter 4

Result

This chapter describes the result of the project, which is how the project team implements the gamified website and its services. Overall, the project team finished designing the essential pages in Figma and implementing the gamified website. The project team also focuses on scalability, usability, and customizability for future development. There are difficulties in the development process, but using open-source systems can reduce the complexity.

4.1 UI Design

The project team designed the pages of the website in Figma. The designs referenced existing products, including Reddit, StackOverflow, HKU Moodle, and LeetCode. Appendix C lists the available User Interface designs. However, the actual implementation does not the same as the design because of the time limit, better data presentation, and development.

4.2 Gamified Website

The project team implemented the Entity Relation Diagrams for the gamified website. Figure 4.1 shows the available pages of the website. The project team implemented the following pages: a user sign-in page, a user registering page, and mentioned pages in chapter 3.4 based on the database schema. The project team also implemented internalized routing for the website to translate the static content of the website. The following discusses the elements of the gamified website and its services and what and how to apply the game elements to the gamified website.

Figure 4.1: Site Diagram of the gamified website
4.2.1 Coding Challenge

Figures from D.1 to D.2 show the page views of coding challenge pages. There are two steps to solving the coding challenge. The first step is to answer how to solve the problem in their own words. The second step is to solve the problem in a programming language. For each submission, the system judges the correctness of the source code with the collaboration with the worker mentioned in chapter 4.6.1. Chapter 4.6.1 also has a detailed flow. A badge will be given to the user for every five challenges they solve.

Judge Service

The project team implemented a service in Golang to handle the judgment of submission. Figure 4.2 shows the lifeline diagram of judging of submission. With the input data from the client, the service will put the test cases and source code into the template as a string. After that, it will get and return the result from Judge0 to the client. Figure 4.3 shows the structure of the template. There are two classes in the template: `CodingChallenge` and `Solution`. `CodingChallenge` takes the parameters `cases` and `expected`. It will also create a new `Solution` object. Function `run()` will run the solution from the user. Figure 4.4 shows how the `run()` function works. It simply loops for all cases and checks whether the result equals the expected output. If not, it will raise an error. The error message is customizable in the source code. For admins who want to modify the content of output in the gamified website, they can modify the error message as an HTML string (For example: “<p>error on case i<p>”).

![Figure 4.2: Lifeline diagram of judging of submission](image)
4.2.2 Forum

Figures from D.7 to D.8 show the page views of the forum pages. It is similar to many existing forums. Users can discuss Computer Science in the forum. The project team also implements the code review method when replying to a post. Users will get points when creating posts and replies. The service points-calculation-server calculates the experience points of the content worth. It is a simple Python Flask API. Also, the system will give a badge to students for creating $5n$ posts or $5n$ replies.

Forum Post Quality Calculation Service

The project team used a model from Kaggle. The model can classify the quality of the post or reply into $LQ\_CLOSE$, $LQ\_EDIT$, and $HQ$. Admins can design the points based on the quality of that. (For example, 1 point for $LQ\_CLOSE$, 5 points for $LQ\_EDIT$, and 10 points for $HQ$.) The project team designs the base points as 0, 1, 2 for $LQ\_CLOSE$, $LQ\_EDIT$, and $HQ$. The points are given to user with this formula: where $x = LQ\_CLOSE$, $LQ\_EDIT$, or $HQ$ randomInt$(a, b)$ means random integer
from 1 to 10 (inclusive).

\[ f(x) = \text{basePoints}(x) \times 10 + \text{randInt}(1, 10) \]

**Code Review**

Code review is a unique feature of the website. Users can select the text from the content. After that, a popup menu will appear, and users can click the code review button. Then users can modify the content of that in the reply text field. (See Figure D.9.)

### 4.2.3 Courses

Figures from D.10 to D.11 and Figure D show the page views of the course and lesson pages. Users can learn different programming languages based on the content created by the admin. Users can finish the course by solving the tests and challenges of it and its lessons. Users will also get experience points after they pass a test. Students will receive a badge after they finish the course. The implementation of the website supports dynamic numbers of lessons, lesson tests, and course tests. Students can also see the progress of it with the progress bar. The following is the detailed rule of the course:

- There can be prerequisites for some lessons. Users should finish them before accessing them.
- There can be prerequisites for some courses. Users should pass them before accessing them.
- Users should solve all lesson tests correctly to finish a lesson.
- Users should solve the course tests correctly to pass a course.
- Users should pass all prerequisites before solving course tests.

### 4.2.4 Contests

Figures from D.4 to D.6 show the page views of the contests. There is a demo contest on the gamified website. Each coding contest supports any number of coding questions and any time limits. Figure 4.5 shows the flow of attempting a coding contest. Students can work on several questions on the same page. They will also get a success or fail dialog after they clear all questions or the time is up.

### 4.2.5 Leaderboard and User Profile Page

Figure D.13 shows the page view of the leaderboard page. It indicates the top 20 users in three categories: Adventurer, Learner, and Evil.

Figure D.14 shows the page view of the user profile page. It can visually show the different abilities of a user. It shows the statistic: the attempt success rate of a user, community statistics, past attempts, levels, experiences, skills, and badges they have.
4.3 Game Elements

4.3.1 Experience Points and Leveling

The curve of experience are designed as

\[ nextLevelExp(currentLevel) = 4 \times currentLevel^2 + 20 \]

The equation takes the level as input and outputs the experience needed for the next level. It can be modified in the source code of the gamified website.

There are three categories of levels, Adventurer, Learner, and Evil. Table 4.1 shows students can get experience points by different actions. Admins can define the experience points of the coding challenges in the Strapi CMS. When solving a coding challenge, students can also get experience points on the skills, which are the programming language they used, and experience points of “Adventurer”. For “Learner”, students can receive experience points by replying to a post, creating a new post, and finishing the tests of lessons or courses. For “Evil”, students can receive experience points by creating a new coding challenge. However, this is currently
unavailable in production because of the difficulties mentioned in Chapter 4.7. The following are the conditions when users get experience points. Figure 4.6 shows the toast that indicates users got experience points when replying to or creating a post. Figure 4.7 shows the dialog that tells users got experience points after users passed all test cases in a coding challenge.

<table>
<thead>
<tr>
<th>Category</th>
<th>Conditions of getting points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adventurer</td>
<td>Passed all test cases of a coding challenge.</td>
</tr>
<tr>
<td>Learner</td>
<td>Creating a new post in the forum; Replying to a post; Passed all test cases of a test in lesson or course.</td>
</tr>
<tr>
<td>Evil</td>
<td>Creating a new coding challenge for Students.</td>
</tr>
</tbody>
</table>

Table 4.1: Condition of getting different points

![Figure 4.6: The toast to indicate users got experience points.](image1)

![Figure 4.7: The dialog when users got experience points.](image2)

### 4.3.2 Badges

There are three badges for users on the website. A Metal Badge will be sent to users when they finish challenges $5n$ times. Students will receive a Discussion Badge sent when they create $5n$ posts or $5n$ replies. Students will receive a Book Badge they finish a course. Figure 4.8 shows the images used for badges. When user receive a badge, the website will show a dialog like Figure 4.9 The implementation in the source code provides an easy way to create a new badge with an image.
Repeatable badges

The project team also implemented the repeatable badges. When the user creates a repeatable badge, it will send to the user at a specific time, for example. It makes use of the node-schedule package. Admins can create and modify the logic of sending badges repeatedly in *filters* (the name of the folder in the gamification CMS) in the backend.

4.3.3 Statistic

The profile page shows the abilities and statistics of the student visually. Figure D.14 shows an example of the profile page. The project team uses a radar chart to indicate the levels of programming languages of the student. The pie chart shows the correctness of coding challenges and the demo contest. The circular and linear progress bar shows the completeness of coding challenges.

4.4 Strapi CMS

Strapi CMS manages the non-game content. Figure 4.10 shows the schema of the database. It provides all non-game content. Strapi CMS provides all CRUD operations to manage non-game content. It does not require any programming to create these operations because Strapi provided them. As mentioned in Chapter 3, developers need external knowledge to create a plugin or controllers in Strapi. It also has restrictions.
on the node version, so some packages may not work on Strapi. Therefore, the project team implemented the gamification CMS to manage the game elements that provide higher flexibility, customizability, and scalability.

![Figure 4.10: Database Schema for Strapi CMS](image)

### 4.5 Gamification CMS

Chapter 3 mentioned the basic structure of the gamification CMS. The following discusses its framework of it.

#### 4.5.1 Structure

Figure 4.11 shows the detailed structure of the gamification CMS. It uses Express.js and Socket.io for the backend; Next.js for the front end. With Express.js, the project team built the APIs on game elements like updating levels and experience and creating new player(user) info. With Socket.io, the website allows future integrations like multiplayer coding contest.

#### 4.5.2 Front-end and Back-end

Figures from D.15 to D.21 show the front end of the CMS. It provides a simple and clean User Interface. In the front end, admins can reset the experiences and levels of users, delete the badges, and stop repeatable badges.
4.6 React Hooks for Gamification

As mentioned in the previous chapter, there are two steps in a coding challenge. The project team implemented several react hooks to achieve the workflow, and the project team designed `usePlayer` and `useMonster` hooks for gamification (see Figure 4.13 and 4.14 for their construction). The idea is similar to some real-life games. Monsters
and players have their life points and actions. A player may have changes in life points. The hook will call `onLifeChange` for this situation. When the player or monster has no life remaining, it handles what to do in the `onDead` function. A monster may have several stages, and `useMonster` also manages the stage.

Figure 4.13: The state, parameter, and functions in `usePlayer`

Figure 4.14: The state, parameter, and functions in `useMonster`

### 4.6.1 Worker function

They have a `worker` function that starts some actions, attacking, for example. Figure 4.15 shows the flow of the worker given the user inputted the parameters in the hook. The worker initializes the work in the `beforeWorking` function (For example, fetching necessary data). Then the worker runs the `work` function. After that, the worker has some clean-up jobs in the `afterWorking` function (For example, updating the database). Given the result from the `beforeWork` function, `work` function, and `afterWork` function, the worker can check whether the work is successful in the `isSuccess` function. If it is a successful work, the worker will run the `onSuccess` function, otherwise the `onFail` function. For example, the website can show the dialog when successful. Users can close some work in the `onEnd` function. For example, users can disconnect the database in this function.

**Use case: solving a Coding Challenge**

Take the Coding Challenge as an example. A user is a player to kill the monster (Coding Challenge). The worker function performs as follows.

1. `beforeWorking`: Do nothing because there is no initial work.
2. **work**: Compile and judge the correctness of the source code.

3. **afterWork**: Submit a new attempt to the CMS.

4. **isSuccess**: Whether the work result is a success or not. Then run **onSuccess** when **isSuccess** is true, **onFail** otherwise.
   
   (a) **onSuccess**: Show a success dialog, create a new post, and give experience points to the user; Reduce monster life points by one.
   
   (b) **onFail**: Reduce player life points by one

5. **onEnd**: Do nothing because there is no remaining work.

For the life points of the player:

- When a player has changed in life points (**onLifeChange**), the website shows a toast to the user (inform the player that the submission does not pass all test cases).

- When a player has no life points (**onDead**), the website shows a dialog to the user (inform the player that he failed to kill the monster) and refreshes the page after five seconds.
For the life points of the monster:

- When a monster has no life points (onDead), it drops experience points to the user and updates the experience points in the CMS.

## 4.7 Difficulties

This project faces difficulties in implementation. There is only one person on the project team, so developing all services from scratch is difficult. Therefore, this project looks for existing services to reduce the complexity and time of development. For example, this project uses Strapi as the CMS and Judge0 as the Remote Code Execution System instead of developing it from scratch.

The project team also focuses on scalability, usability, and customizability. For example, the project team creates the `useMonster` hook for gamifying some new features. The gamification CMS is also customizable with various databases and requirements on gamification. For example, the gamification CMS can connect to MongoDB, add daily login events with Socket.io, and send badges to top users monthly.

Moreover, the project focuses more on the gamified website. The front end of gamification CMS only provides part of the features because of the time limit. The front end of gamification CMS does not include all CRUD features.

Furthermore, Judge0 allows different programming languages for students. However, the project team restricted it to Python only, and students could not create a new coding challenge because of difficulties in checking the user inputs, including function body, test cases, and expected outputs.
Chapter 5

Conclusion

Currently, the e-learning environment of CS in HK is not motivating. The few discussion, interactions, and motivation make students less competitive than others. This project designs a gamified website for CS students to engage them in learning. However, this website focuses on programming, an essential subset of CS with a time constraint.

5.1 Summary of works

The studies show the ability to motivate the e-learning environment with gamification. Based on the approaches of these studies, the project team designed the UI, game elements, and non-game content. The project team implemented the gamified website with the services. The services are the Strapi CMS, gamification CMS, Judge Service, and Points Calculation Service. The project team built these services with different programming languages and frameworks. As a result, this project provided an example or prototype for a gamified website for learning programming. The project team also developed reusable functions like `useMonster` for applying game elements to non-game content. The project team also focuses on customizability and scalability. Developers can easily customize and extend the gamified website with more services. The gamification CMS is also customizable with various databases and requirements on gamification.

5.2 Limitations

There are some limitations to the website. As mentioned in Chapter 1 and Chapter 3, this project implemented services with AI for the point system. However, the project team will not tune and research for the correctness of the model because of the time limit. The website is not ready for deployment. The reasons are as follows:

For the coding challenge, the project team only allows the programming language to be Python. Furthermore, the template of coding challenges is non-auto-generated. Because of the time limit, the project team cannot implement an efficient way to input test cases and their (complex) data types. Therefore, students cannot create a new challenge currently. Besides, the project team does not prettify the User Interface and implement all features like modifying a single user's experience and level for the gamification CMS.
Contests, coding challenges, courses, and lessons heavily rely on Judge0, which allows 50 daily requests only. Therefore, the possible way to unlimited requests is by creating an alternative of Judge0 or buying API request quotas.

Because of the time limit, the project team uses a package to fire a Dialog or Toast for some events. However, it does not support a Dialog and Toast in the same frame. It may cause trouble in implementing a good User Experience. For example, users may get points and a badge, but this website shows the toast and dialog with an order.

5.3 Recommendations

Experiments in studies show that gamification can motivate students to learn with gamification. However, this project will not research the effectiveness of the gamified website because it takes a long and continuous time. Researchers can use it as a prototype for learning the various knowledge of CS. Researchers can also evaluate its effectiveness and possible changes through experiments. For example, future experiments can invite some students to use the website and make evaluations by comparing it with traditional learning.

5.4 Future works

The following are some future works for converting the website from a prototype or demo to production. Building the production website like debugging for docker-compose and Dockerfile; If developers wants to use Judge0, the project team recommends to self-host Judge0 for unlimited requests; The judging service can support more programming languages; Developers can also design an alternative to Judge0; Transforming the coding challenges into challenges, which does not limit to programming; Developing a better logic for the contests with dynamic content; Provide methods to link the existing courses in Moodle or other websites; Build AI for more purpose like auto-creation of CS courses. There are more details in the source code.
Bibliography


Appendices
Appendix A

Gamification design

The following are the three section of gamification design. [4]

• **Mechanics**: Points and badges that concern data representation and algorithms.

• **Dynamics**: Players’ inputs and outputs over time are concerns of the runtime behavior of mechanics like completion and choices.

• **Aesthetics**: Users’ emotional responses after they interact with the gamified system. For example, the feeling of being challenged.
Appendix B

Self-determination theory

The following are the three sections in self-determination theory [1],[5].

- **Relatedness**: Interactions and connections with others, their goals, their interests.

- **Competence**: Challenging tasks aiming to complete their goals are what personal interests in.

- **Autonomy**: It satisfies the need to control one’s own life by participating in voluntary play to fulfill a person’s personal goals.
Appendix C

Page Designs

The following are the designs on important pages of the website. There are comments that indicate the need to change the pages. The gray rectangles are images that can be replaced in future development.

Figure C.1: Home Page of the website
Figure C.2: Post List Page of the website
Figure C.3: Post Details of the website

Figure C.4: Login Page of the website
Figure C.5: Register Page of the website

Figure C.6: Add Post Page of the website
Figure C.7: Challenge Page of the website

Figure C.8: LeaderBoard Page of the website
Figure C.9: Contest list Page of the website

Figure C.10: Contest Page of the website

Figure C.11: Course List Page of the website
Figure C.12: User Profile Page of the website
Appendix D

Images of Front-end

This Appendix shows the front-end and implementation of the project.
Figure D.1: The page of listing Challenges.

Figure D.2: The detail page of coding challenge.

Figure D.3: The detail page of coding challenge.
Figure D.4: The pages of listing Contest.

Figure D.5: Confirm Dialog.

Figure D.6: The detail page of contest.
How to write an Interview Question post

LaestCode Discuss is a community where you can share questions and experiences from interviews. Before you post, please try to follow these guidelines:

- The title should follow this format:

```
[Question type] | [Tag] | [Company] | [Location] | [Question name]
```

- Please note that we won’t accept duplicate questions or questions that have already been posted.
- If you’re including code in your post, please surround the code block with three backticks (```).
- Each question should be accompanied by at least one example:
  - The more the better.
  - Please share the company name in the title above which will help people see your question more easily.
- Each post should include only one question. If you want to share your overall interview experience, please post in the interview experience community.
- If you represent a start-up that you submit your or display the interviews does not require or violate any of the copyrights, patents, trademark, trade secrets or any other personal or proprietary rights of any third party.

Happy Discussing!

Figure D.7: Main page of Forum.

Figure D.8: Post detail page.

Figure D.9: Code review.
Figure D.10: Courses List page

Figure D.11: Course details page.

Figure D.12: Lesson page.
Figure D.13: The leaderboard page

Figure D.14: The profile page

Figure D.15: page of gamification CMS
Figure D.16: A page that shows the available badges

Figure D.17: A page that listing users’ levels and experiences
Figure D.18: A page that listing Users have what Badges

Figure D.19: Dialog when creating normal badge
Figure D.20: Dialog when creating repeatable badge

Figure D.21: Dialog when deleting the badge