International Journal of Religion
2024
Volume: 5| Number 11 | pp. 1938 – 1953
ISSN: 2633-352X (Print) | ISSN: 2633-3538 (Online)
ijor.co.uk
DOI: https://doi.org/10.61707/xsp9af02

HoQ - Based Learning Game Development. Case Study: Travelix Application for Learning Traffic Law

Ma Thi Chau[1]

Abstract

This research article explores the application of game-based learning in the field of education, specifically focusing on proposing a process for the development of game-based learning using the House of Quality framework. The article also introduces a game called Travelix, which is designed for educational purposes. Travelix aims to assist individuals in preparing for the theoretical section of the driver's license examination in Vietnam, featuring 600 questions. The author conducts a test scenario to evaluate the effectiveness of enhancing knowledge and understanding of driving theory at the B2 level, as well as assessing the game's usability and appeal to players. By implementing the proposed game development process, Travelix significantly reduces the time required for modifications and garners prompt and positive feedback upon its initial release, thereby enhancing the overall user experience.

Keywords: *Learning Games, House of Quality, Traffic Law, Traffic Law Education, Traffic Law Learning Application.*

INTRODUCTION

The game-based learning trend is gaining popularity and being widely embraced as it incorporates gaming elements into the learning process. This trend signifies a shift towards interactive, engaging, and learner-centered educational approaches. It acknowledges the effectiveness of games in capturing learners' attention, promoting active participation, and enhancing knowledge retention. By harnessing the power of games, educators can create stimulating learning experiences that facilitate the acquisition of knowledge, development of skills, and enjoyment for learners. Consequently, the development of learning games is increasingly crucial in both the education and game industries. It entails the creation of games with educational objectives to support the learning process and foster skill development among players. According to Linkedin's Educational Games Market Research Report 2023, "The Global Educational Games market is anticipated to rise at a considerable rate during the forecast period, between 2023 and 2031.

The House of Quality (HoQ) [1], is an important tool utilized in quality management and product design processes. It is employed to analyze and evaluate customer requirements, and subsequently translate them into specific product features and attributes according to their prioritization. The HoQ tool is commonly integrated into the Quality Function Deployment (QFD) process, which ensures that customer requirements are comprehended and fulfilled throughout the design and production stages. HoQ facilitates the establishment of connections between various quality factors and assists in determining their significance to customers. By assessing the level of relevance and prioritizing these factors, HoQ aids in identifying quality objectives and setting criteria to satisfy customer requirements in the design and production of the product.

The objective of this article is to investigate the application of HoQ in the development of game-based learning applications. Furthermore, we will support these discussions with a specific case study focused on the development of a game-based learning application for traffic law education.

According to [2], road accidents occur daily, with only a small percentage of these accidents being attributed to mechanical or technical failures. The majority of accidents are caused by factors such as drunk driving or health issues. However, if we eliminate these causes, the primary factor contributing to the majority of accidents is the lack of adequate knowledge of traffic laws. Therefore, it is crucial to enhance understanding of traffic laws and promote awareness of law compliance when participating in traffic. As a result, this article focuses on the

[1] Faculty of Information Technology, VNU University of Engineering and Technology,No. 144 Xuan Thuy Street, Dich Vong Ward, Cau Giay District, Hanoi, Vietnam, E-mail: chaumt@vnu.edu.vn

development of a game-based learning application for traffic law education and compliance during traffic participation. This approach utilizes the advantages of game-based learning theory and the HoQ-based design method. Our contributions to this article are as follows:

Proposing a development process forgame-based applications based on HoQ.

Presenting a case study on the development of the Travelix game application, which supports theoretical driving education. The rest of the article is organized as follows: Firstly, we introduce knowledge about HoQ and the development of products based on HoQ, as well as the current status of application development for learning traffic laws in Vietnam in Section 2. The proposal for developing a learning game based on HoQ is described in Section 3. We present a case study: Travelix - an application for learning traffic laws in Section 4. Evaluation and discussion are provided in Section 5.

Background And Related Works

Key considerations in game-based learning Development

Game-based learning is driven by several factors, including advancements in technology that have made educational games more accessible on various platforms. This accessibility has created opportunities to

integrate game-based learning into both formal and informal education. Research has shown that game-based learning enhances learner outcomes by increasing engagement, motivation, and active participation, leading to improved knowledge acquisition and retention. Games also foster the development of critical thinking, problem-solving, collaboration, and creativity skills. Additionally, the trend aligns with the preferences of digital-native learners who are familiar with interactive and immersive digital experiences. By incorporating game elements, educators can harness learners' natural affinity for games to facilitate effective learning. Game-based learning utilizes interactive and immersive game mechanics, such as challenges,

rewards, competition, and collaboration, to create an engaging and motivating learning environment. It leverages the inherent characteristics of games, such as interactivity, problem-solving, decision-making, and immediate feedback, to enhance the learning experience and make it more enjoyable. The factors that contribute to successful educational game development and provide educational benefits to players include:

Clear Educational Objectives

Educational games should have well-defined educational objectives that focus on delivering knowledge and developing specific skills for the players.

Engaging Game Design

Game design should create an interactive and captivating learning experience. This involves designing elements such as storyline, gameplay structure, challenges, and rewards to create an enjoyable and stimulating learning environment.

Active player interaction

Educational games should encourage active participation from players. This can involve asking questions, engaging in interactive activities, exploring, and solving problems within the game.

Learning Diversity

Educational games should cater to the diverse learning needs of players. They should offer different difficulty levels and allow customization of learning content to suit individual preferences and abilities.

Continuous Feedback and Assessment*:* Educational games should provide ongoing feedback and assess the player's progress. This helps players understand their strengths and weaknesses and provides guidance for improving their learning skills.

Real-world Connections*:* Educational games can establish connections with the real world and apply acquired knowledge to real-life situations. This helps players recognize the relevance and application of what they learn in their daily lives.

Technology and Innovation*:* Developing educational games requires familiarity with relevant technologies and tools. This includes utilizing online platforms, virtual reality technology, real-time online interactions, and artificial intelligence to create diverse and unique learning experiences.

Continuous Updates and Improvements*:* Educational games should be regularly updated and improved to meet evolving learning needs and changes in the field of education. Gathering feedback from players and implementing improvements based on research are vital for maintaining the competitiveness and effectiveness of educational games.

Implementing the House of Quality: A Step-by-Step Process

HoQ helps in designing and developing customer-centric products. The implementation process involves the following steps:

Identify customer requirements

Gather information through surveys, interviews, or market analysis to understand customer needs, expectations, and goals.

Determine Quality Attributes

Identify specific quality attributes like performance, reliability, sustainability, ease of use, etc., that the product must possess to meet customer requirements.

Construct the HoQ Model*:* Create a House of Quality matrix that maps customer requirements to quality attributes, using prioritization techniques to determine their importance.

Establish Quality Targets*:* Define precise quality targets based on the HoQ model to guide the design and development process.

Design and Develop the Product*:* Utilize the identified quality targets, considering the requirements and quality attributes determined earlier.

Test and Evaluate*:* Conduct thorough testing and evaluation to ensure the product meets the specified requirements and quality targets. Gather feedback from customers and real-world data for future enhancements.

Implementing HoQ enables product developers to create high-quality products and services that align with customer desires.

Related Works

Before acquiring a driver's license, individuals are required to participate in a training program aimed at educating them about traffic rules, driver rights and responsibilities, and safety measures. The 2024 driver's license examination includes categories such as A1 motorcycle, B11 automatic transmission car, B2 manual transmission car, C, D, and E. For the driving license exam preparation, there are applications available for theoretical driving license exam preparation with a set of 600 questions. The questions cover various topics including road traffic concepts, regulations, transport operations, car driving techniques, road sign systems, and practical exercises. Each category of driver's license has a different number of questions. To pass the exam, learners must answer a certain number of questions correctly, including critical questions. The set of 200 questions for the motorcycle driving test includes sections on Law Knowledge, Road Signs, and Practical Exercises.

We have observed that individuals generally show a genuine interest in these applications only when they are preparing for their driving license exams. Only those who are actively studying for the exam tend to use these

applications as a valuable resource for exam preparation. However, it is crucial for the entire population to have knowledge of traffic laws while driving, as it ensures personal safety and the safety of others. Therefore, it is important to explore diverse approaches to enhance the understanding of traffic laws and encourage compliance with regulations when engaging in traffic activities. Learning traffic laws through gaming is one effective and engaging method that promotes learning while providing enjoyment, which makes it particularly suitable for adolescents. According to the authors in [3], drivers primarily rely on study materials and the official driver's license examination to obtain their driver's license. In addition, drivers enhance their understanding, application, and breadth of knowledge regarding traffic laws by referring to instructional signs while on the road. They also acquire essential knowledge about traffic from well-informed friends and relatives who possess a comprehensive understanding of traffic rules and regulations. Personal observations and experiences play a significant role in expanding their understanding. However, the majority of drivers do not actively seek out seminars or utilize social media or internet resources to enhance their comprehension of traffic laws and road safety while driving.

Our aim is to develop a puzzle game application that is compatible across multiple platforms, facilitating the acquisition of theoretical knowledge for the driving license exam. We strive to create an immersive and captivating learning experience, enabling users to acquire knowledge through enjoyable gameplay. The application will provide users with the convenience of participating at their own pace, irrespective of time or location, and will cater to individuals from diverse backgrounds.

Proposal: Development of a HoQ-Based Learning Game

Developing learning games requires a learner-centric approach and careful consideration of various technical aspects. In this regard, the HoQ model can be employed to guide the development process. The HoQ model focuses on meeting the needs and expectations of end-users, specifically the players of the learning games. By identifying user requirements and expectations, the HoQ model ensures that the games are designed to fulfill these needs effectively. Additionally, the HoQ model provides a framework for integrating different elements involved in the development of learning games, establishing connections between factors such as user requirements, features, technologies, and other crucial elements.

One of the key benefits of using the HoQ model is its ability to prioritize significant factors during the development process. By evaluating the importance and impact level of each factor, the HoQ model helps developers identify and prioritize elements that require special attention. Moreover, the HoQ model contributes to maintaining the quality of learning games by defining the desired quality factors and providing evaluation criteria. It enables developers to measure and assess the extent to which these quality factors have been achieved, ensuring that the educational games meet the specified quality requirements.

Therefore, employing the HoQ model in the development of learning games offers several advantages, including a user-centric focus, effective integration of important factors, task prioritization, and assurance of quality. With this in mind, our proposal suggests a solution for developing learning games based on the HoQ model, which involves the following steps (Fig. 1): Identifying customer requirements; Buildin requirement-based HoQ and identifying priority requirements; Determining game solutions to address the priority requirements; Developing the game using the proposed solutions; and Evaluating the game. Customer Requirements focus on identifying and understanding the specific needs and expectations of the end-users. This process involves gathering feedback and input from the target audience to ensure that the game effectively addresses their requirements. Requirement-based HoQ involves constructing a visual model that depicts the relationship between customer requirements and the technical aspects of the game. The HoQ helps establish a clear understanding of how different requirements align with various development elements. Once the customer requirements are identified, it is crucial to prioritize them based on their importance and impact.

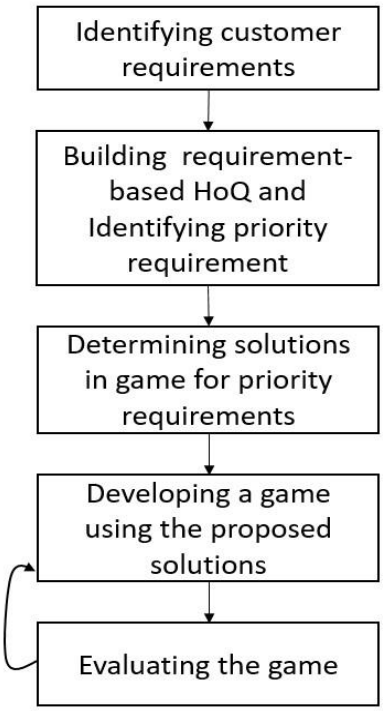


Fig. 1. HoQ-based learning game development.

This step entails evaluating and selecting the most critical requirements that should be addressed as a priority during the development process. Next, Game Solutions for Priority Requirements focuses on determining suitable solutions within the game design and development process to meet the priority requirements. This stage involves brainstorming and exploring creative ideas to incorporate features, mechanics, and content that align with the identified priority requirements. Game Development encompasses the actual implementation of the proposed game solutions. It involves handling the technical aspects of programming, design, artwork, audio, and other components necessary to bring the learning game to life. Once the game development is complete, the evaluation phase comes into play. This involves conducting a comprehensive assessment of the game, including functionality, usability, engagement, and alignment with the identified customer requirements. Evaluation criteria and metrics are utilized to measure the game's performance and ensure that it meets the specified quality standards.

By following the proposal, time can be saved in the analysis, design, and implementation of the product. To illustrate the effectiveness of this approach, we will perform the proposed evaluation in the specific development of a traffic law learning game application, which will be presented in the section 4.

Case Study: Travelix - Application for Learning Traffic Law

As mentioned earlier, our goal is to develop a learning game focused on theoretical traffic law. To achieve this, developers need to possess a comprehensive understanding and a solid grasp of traffic regulations and rules, which is a crucial factor. Researching traffic laws and real-life scenarios will contribute to creating accurate and valuable content for the game. Considering that the content is theoretical and the driver's license exam primarily consists of multiple-choice questions, we have devised a quiz-based game format. The questions, exercises, and scenarios pertaining to traffic laws have been sourced from a questionnaire consisting of 600 theoretical driving examination questions. We aim to incorporate game elements such as scoring, ranking, rewards, and levels to generate motivation and enhance players' enjoyment. This approach will incentivize players to actively engage and sustain their learning throughout the game. Therefore, we will proceed with implementing the detailed measures proposed in Section 3 to create Travelix, an interactive game specifically designed to assist players in learning the theoretical aspects necessary for passing the driver's license examination.

Identifying Customer Requirements

By studying the methodology for developing a learning game on traffic law in [2], we conducted a focus group interview with a group of 24 individuals to gather insights into the requirements for applications that support the learning of traffic rules. These requirements serve as a valuable resource for methodological research and the development of applications that facilitate the learning of traffic laws. The specific requirements obtained from the interview are presented in Tab. 1. The last column (*PriN*) in Tab. 1 represents the customers' rating of the importance of the requirements based on the preferences of the interviewees.

Tab. 1. Customer requirements

No	***Requirements***	***PriN***
R1	Use simple and clear examples	1
R2	Be up to date	1
R3	Have good quality illustrations and animation	2
R4	Have On/Offline version	2
R5	Be easy to handle	2
R6	Have small modules (compact)	3
R7	Should be quick	4
R8	Be free	5

Building A Requirement-Based Hoq and Identifying Priority Requirement

Following the guidelines provided by [4], we have built HoQ. Firstly, from the developer's side, we have outlined the technical features that the Travelix needs to ensure, as described in Tab. 2. We have classified the relationship between customer requirements and technical features into three levels: *low*, *medium*, and *high*, corresponding to point values of 1, 3, and 9, respectively (Tab. 3).

Tab. 2. Technical feature descriptions

No	**Technical features**
T1	It should be easy to use on any device
T2	It should be accessible any where
T3	It should be accessible any time
T4	It should be designed for wide range of learners
T5	It should be possibility of flexibility
T6	Its content is about learning traffic law
T7	It should feature a user-friendly interface
T8	It should have attractive background
T9	It should have suitable audio
T10	It should have openness

Tab. 3. Relationship matrix

Relationship	***Score***
Strong	9
Medium	3
Weak	1
No assign	0

Based on the provided information, we have created a correlation matrix in the HoQ model Tab. 2. Technical feature descriptions as shown in Tab. 4. The first row represents the technical features, the first column represents the customer requirements, the second column corresponds to the customer important rating of the customer requirements, and the third column is filled with percentage of customer important rating (Per_i) as formula (1). The relationships (in numeric form as shown in Tab. 3) are then filled in the cells between the specific customer requirement and corresponding technical characteristic.

$$Per_i = \frac{PriN_i}{\sum_i PriN_i} \quad (1)$$

where $PriN_i$ and Per_i are customer important rating and percentage of customer important rating respectively.

Finally, the last row, techniqual feature importance rating (*TechImp*), is fill follow the formula (2)

$$TechImp_j = \sum_i Per_i \; x \; Score_{ij} \quad (2)$$

As indicated in Tab. 4, the technical feature importance ratings are calculated in the last row. Based on these results, we conclude that in the development process, it is important to prioritize the technical features *T4*, *T8*, *T10*, *T7*, *T5*, and *T1*.

Determining Solution in Game

As mentioned in section 4.2, we will focus on the technical features T4, T8, T10, T7, T5, and T1 during the development process of Travelix. Specifically, our approach will be as follows.

T4: With this requirement, regarding the game content as mentioned above, we will transmit knowledge through 600 theoretical driving test questions during the gameplay to provide intangible benefits to the players. The game will be designed for both PC and mobile platforms. In particular, to engage players and effectively deliver information, we will focus on an immersive graphical interface that is realistic and features user-friendly and efficient interactive buttons suitable for different age groups to stimulate and enhance knowledge of traffic laws for players.

T8: With this requirement, we will design dynamic environment resembling a map, simulating traffic maps, incorporating roadside landscapes to mimic real-life scenarios, and varying maps for diversity akin to reality.

T10: With this requirement, we focus on dababase organization. Database is organized in the form of database-driven content storage, standardized and updatable modules integrated into the database for easy updating of new content.

Tab. 4. HoQ table

Reqs	***PriN***	***Per***	***T1***	***T2***	***T3***	***T4***	***T5***	***T6***	***T7***	***T8***	***T9***	***T10***
R1	1	18	3	0	0	9	3	1	0	1	0	0
R2	1	18	0	0	0	9	3	1	0	1	0	9
R3	2	14	0	0	0	3	0	0	0	9	3	0
R4	2	14	0	3	3	3	0	0	1	0	0	0
R5	2	14	3	0	0	3	0	0	9	1	0	0
R6	3	11	0	3	3	3	0	0	1	0	0	0
R7	4	7	1	1	1	3	0	0	1	0	0	0
R8	5	4	0	0	0	3	0	0	0	0	0	0
			1.04	0.82	0.82	***5.14***	***1.07***	0.36	***1.60***	***1.79***	0.43	***1.61***

T5: With this requirement, we have designed the game with a ranking system that enables players to pause in the middle, resume playing, or restart from the beginning based on their preference.

T1: With this requirement, we develop mobile app, compatible with commonly used devices, with a web-based version available.

Developing Travelix Game

We have developed the Travelix game to support players in enhancing their knowledge of traffic laws and gaining confidence in their ability to travel. After identifying the key technical features to focus on in game development (Section 4.2 and 4.3), we proceed with the following specific steps (Fig. 2): (i) suggesting the game stage concept, (ii) defining game mechanics, (iii) designing the game, and (iv) implementing the game. The prioritized technical features will receive attention during these steps.

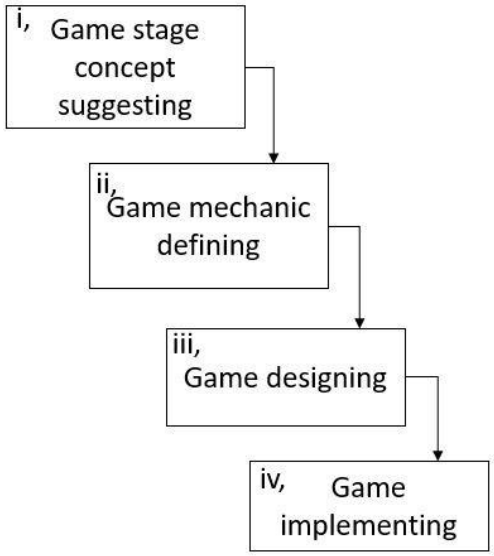


Fig. 2. Game development flow.

Technical feature *T*7 mentioned earlier will be given priority by proposing *Game stage concept suggesting*. To effectively meet technical features *T*4 and *T*5, we focus on *the game mechanic defining*. During the game designing step, we emphasize designing collision handling to fulfill the requirements of technical feature *T*4, creating in-game objects that meet the criteria of technical features *T*4, *T*7, and *T*8, designing a visually appealing and user-friendly UI to meet technical features *T*7 and *T*8, managing data to fulfill technical feature *T*10, and optimizing data loading to meet the specifications of technical features *T*8 and *T*10.

Gameplay Stage Suggesting

Our main focus is on designing gameplay stages that are both suitable and captivating. When conducting a theoretical driving tests, users are typically required to complete a test consisting of *n* questions, where *n* >= 20. However, if there are a large number of questions in a single gameplay stage, it can be challenging for users to remember all of them. Therefore, we divide each gameplay into 2 tests. So, in each game round, players can complete 1-2 tests of theoretical driving questions, with each test containing *n* questions. Each test consists of *m* game screen of two stages. With each game screen, the first stage, called *route finding*, involves placing directional markers (left, right, or U-turn) at intersections to guide the vehicle's movement towards the destination. The map provides additional markers for guidance. The number of available and given markers determines the difficulty. The objective is to create a route using the markers to collect stars and reach the destination. The second stage is *obstacle crossing*. After setting the route, players start the vehicle's movement. Stars and obstacles appear along the way. Obstacles are represented by rocks or barriers, while stars and obstacles are presented as questions. Touching a star or obstacle triggers a question from a pool of 600 traffic-

related questions. Correctly answering a star question collects the star, while answering an obstacle question correctly allows passage. Incorrectly answering a star question skips it, and the vehicle continues. However, failing an obstacle question requires restarting the stage. Answering questions imparts knowledge and aids memorization. The stage ends when the vehicle reaches the destination. In our approach, we have chosen to include 5 questions in each second game stage. For instance, if someone is practicing for a B2 driving theory exam, which comprises 35 questions, we divide it into $m = 7$ gameplay screens, with each second stage of each game play screen requiring the player to answer 5 questions. These 35 questions are randomly selected from a pool of 600 questions. As a result, these screens offer distinct variations that help users progress through the game. The selection of questions is randomized, ensuring that there is one critical question included. If a player fails to answer this critical question correctly, they have to start the game again from the beginning, with an accumulated score of 0. This method not only facilitates better memorization but also allows for the creation of visually striking images that can enhance users' visual memory of the corresponding questions.

To enhance the immersive gaming experience for players, we have designed a map-like interface for the test (Fig. 3 and Fig. 4). The interface prominently features a text-based UI positioned at the top center of the screen, providing players with information about the current stage they are engaged in. Located in the bottom corner of the screen is a Start button, which initiates the second stage of gameplay upon being clicked. Once the Start button is pressed, it, along with the adjacent directional selection panel, will be hidden from view. Simultaneously, a blue button will emerge in the top left corner of the screen. This blue button acts as a restart button for the current stage. When pressed, it triggers an event that resets all objects to their initial state prior to the start of the stage. The Start button and directional selection panel remain hidden while the restart button takes its place on the screen.

Fig. 3. Map for test 1.

Fig. 4. Map for test 2.

Game Mechanic Defining

Each question in the game has only one correct answer and is designed to be reasonably challenging for the player. The game displays the questions along with answer choices, similar to traditional multiple-choice questions on article. However, the key difference is that players immediately receive feedback on their answer without having to wait until the end of the test. The questions cover various knowledge areas such as

transportation operations, vehicle structure, and repair, or critical topics, so they are categorized and organized accordingly.

To ensure clarity, the game provides feedback to the player when they answer correctly or incorrectly. This feedback is presented through both sound and visuals. Additionally, to engage users and encourage them to answer more questions correctly, the game includes a scoring system. Each correct answer accumulates points for the player, and the results are announced at the end of each test. The questions are displayed on the question-answer screen, positioned above the gameplay area. The game uses images extracted from the official test software, which include the question, relevant examples like road signs or maneuvers, and corresponding question numbers. As it is a multiple-choice test, the answers are presented in labeled answer boxes, each labeled with a number corresponding to the order of the answer options. After the player selects an answer, the color of the answer box changes accompanied by a sound effect. There is a brief waiting period before the question screen hides to announce the player's answer result.

Each answer is represented as a button with a square white button image, accompanied by the corresponding answer number. When a player taps on an answer, the game checks if it matches the question and disables tapping on other answer options using a boolean variable. If the answer is correct, the question background turns green, and the player's score increases. If the answer is incorrect, the background turns red (Fig. 5). These stages trigger the appropriate events for a correct or incorrect answer.

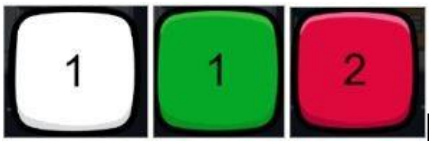


Fig. 5. Answer states.

Game Object Designing

In this game, various objects need to be created, including a car that represents the player and can move on the screen, stars that serve as rewards for correct answers, obstacles where questions appear to challenge the player, road edges, finish lines, and directional signs (Fig. 6). Regarding the features, the car object includes functionalities for movement, performing actions such as turning left, turning right, or making a U-turn, displaying visual effects, and responding to collision events with other objects like stars and road obstacles. Stars and obstacles are positioned at different locations on the road and interact with the car. When the car collides with a star, it disappears if the player answers correctly or fades away if the player answers incorrectly. On the other hand, obstacles only disappear when the player answers a question correctly. Answering incorrectly immediately results in a failed outcome in the level. The road edge object represents the visuals surrounding the buildings, and when the car collides with these objects, it simulates the car hitting the curb, potentially causing an accident. Therefore, collision events between the car and road edges, similar to answering a question incorrectly at an obstacle, lead to failure in the level. The finish line object acts as the target that the player needs to make the car touch. When the car collides with the finish line, it signifies the completion of the level. The directional signs are categorized into three types: left turn, right turn, and U-turn. These signs are placed before intersections. When the car comes into contact with these objects, it has the ability to change direction accordingly. Predefined signs will be visible on the map, while additional signs are accessible in a sign selection menu. The sign menu provides three types of signs and specifies the quantity available for each type in each level. When the available quantity of a particular sign type runs out, the corresponding selection in the sign menu will be disabled or grayed out.

Fig. 6. Directional signs.

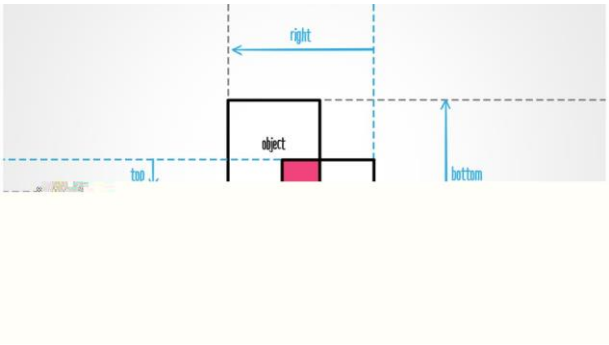
Collision Checking

The physics system plays a crucial role in ensuring collision detection between objects throughout the game. In this physics system, each graphical object is represented by a collider, also known as a collision box. These colliders are rectangular shapes with sizes and positions that are appropriate for each object. When colliders intersect, it indicates that two objects are colliding. While the colliders move along with the objects during gameplay, the actual collision detection is handled by a separate class called *CollisionDetector*.

The *CollisionDetector* class maintains a list of colliders and continuously performs collision checks between them based on predefined collision configurations. Since not all objects need to interact with each other, collision checks between certain colliders are unnecessary. Therefore, when checking collisions between colliders, each collider is assigned a tag to determine the type of object it represents. Additionally, a filter is applied to identify the specific types of objects that a collider needs to check for appropriate handling. This flexible approach allows for the construction of a simple and customizable physics system that fulfills the game's requirements, without relying on pre-existing physics libraries. The *CollisionDetector* class mentioned above is designed specifically for checking collisions between colliders. It is implemented as a Singleton pattern, ensuring that only one instance is created throughout the application's lifetime.

The collision detection between two colliders is performed using an Axis-Aligned Bounding Box (*AABB*) and triggers the appropriate collision events if necessary (Fig. 7). *AABB* is an algorithm that checks collisions between the edges of rectangles, where the edges are parallel to the coordinate axes. Essentially, it determines if two rectangles overlap by evaluating each pair of edges of the two rectangles using the following conditions: *other.left* <= *object.right*, *other.right* >= *object.left*, *other.top* >= *object.bottom*, *other.bottom* <= *object.top*.

Fig. 7. Collision.



GAME UI

The game consists of four main interfaces: *the main screen interface*, *the gameplay interface*, *the victory interface*, and *the failure interface*. Each interface contains UI objects that are essential for smooth operation. There are nine possible positions to place a UI object, which are combinations of the top, center, and bottom sides of the screen, along with the left, center, and right sides.

The main screen interface (Fig. 8) features three functional buttons: one to display the gameplay, one to show the instructions, and one to enter the gameplay screen. This interface focuses on the game's functions. The background of the main screen is a stretched PureTransform that fills the gameplay screen. Rounded orange buttons with text labels indicating their purpose are positioned in the bottom-left corner of the screen. These buttons include the play button, the instructions button, and the level selection buttons. The level button opens the level interface (Fig. 9), while the instructions button opens the player instructions interface (Fig. 10). The play button facilitates the transition from the main screen to the gameplay screen.

Fig. 8. The main interface.

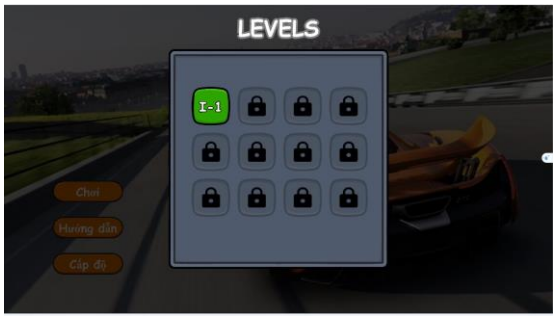


Fig. 9. The level interface.

Fig. 10. The instruction interface.

The gameplay interface, as depicted in Fig. 3 and Fig. 4, features a text UI positioned at the center of the top of the screen. This text informs the player about the current stage they are solving. The Start button is positioned at the center of the bottom corner of the screen. Pressing the Start button initiates the second stage of gameplay. Upon pressing Start, the button and the adjacent direction selection panel are hidden, and a green restart button appears in the top-left corner of the screen. The restart button allows the player to reload objects to their initial state before starting the stage. When the restart button is pressed, the Start button and direction selection panel are displayed again, while the restart button becomes hidden.

The victory interface (Fig. 11) is displayed when the vehicle reaches the finish line. Typically, a centered orange panel appears, displaying various messages. This interface incorporates the player's score or the number of questions they answered correctly, represented by gold stars. The number of stars corresponds to the number of questions in the gameplay. Each star is fixedly positioned on the orange panel. The first and fifth stars, as well as the second and fourth stars, have the same size and are symmetrical across the middle axis of the screen. The third star is the largest and positioned horizontally at the center. When the victory screen appears, the stars animate in a sequence from the first star to the fifth star, creating a starburst effect. Below the orange panel, three buttons with different functions are displayed: one to return to the main screen, one to replay the gameplay, and one to proceed to the next gameplay. In the seventh stage of each city, a white text line appears, prompting the DataManager to calculate the total score of the gameplay, including the previous six stages. After the stars have appeared, the white text line announces the total score on a thirty-five-point scale. The buttons

are lowered, and their centers are positioned at the bottom of the orange panel to accommodate the total score information.

Fig. 11. The victory interface.

The failure interface (Fig. 12) is displayed when the player fails to answer a penalty question or collides with the roadside. It features a simple screen with a "Level

Failed!!"message displayed in the center. The screen has a white background with a black border and buttons. Since the player has failed this stage, only two buttons are displayed: one to return to the main screen and one to replay the gameplay.

Fig. 12. The failure interface.

Data Management

In a game, particularly a platform or side-scrolling game, there is a significant amount of data that needs to be stored. This data includes the user's current level, the character's costume, and their score. It is crucial to retain this data because users don't want to start from the beginning every time they exit the game. Additionally, if users have spent money on purchasing costumes, they expect to see those costumes in their inventory. To meet these requirements, we have implemented appropriate mechanisms for storing user data.

To manage game data, we utilize the browser's database, specifically *IndexedDB*, as the storage solution. The data that is loaded includes the user's current level, the number of stars collected in each level, and the list of unlocked levels. To simplify data management, we have developed a *DataLocal* class that handles reading, writing, and initializing the database. The data read from the database is then recorded by a UserData class to avoid redundant reads. Accessing the data within the game is facilitated by a *DataManager* class, which is responsible for retrieving data, including read-only data, and does not allow writing.

The player's game data is stored locally using *IndexedDB*, which is a web API that enables structured data storage on the client-side within the browser. It follows a key-value format. The storage and retrieval of game data from *IndexedDB* are performed through the *DataLocal* object. Initially, *DataLocal* checks if the browser supports *IndexedDB* and sets it up with a specific name and version. If the storage does not exist, a new storage is created. Subsequently, *DataLocal* executes a method to load the user data, including the current level, unlocked levels, and the number of stars collected in each level. For new players, *DataLocal* generates initial data and stores it in *IndexedDB*. In addition to the mentioned methods, *DataLocal* provides a *getData*() method for retrieving data

from the *IndexedDB* storage. It returns a promise that is resolved with the requested data. Furthermore, *DataLocal* includes an *UpdateData*() method for updating data in the data store and a *checkLoad*() method to monitor the data loading process. Once the data loading is complete, a *DataManager* object is initialized. This object is responsible for managing player data and facilitating the game's access to the data.

Game Data Loading

To create a game that appeals to a wide audience, it is not necessary to rely solely on stunning high-resolution 3D graphics. The crucial aspect lies in ensuring that the images synchronize with each other and effectively convey the game's content. Similarly, sound plays a vital role in captivating players. The quality of a game, or a significant part of it, hinges on the audio elements. Well-suited music has the power to guide players' emotions and immerse them in the game's narrative.

There are two approaches to loading game data. The first approach involves loading images and sounds during gameplay. While this method is straightforward, it can occasionally disrupt the user experience by causing lag, as the game simultaneously handles logic and resource loading. The second approach is to preload images and sounds before starting the game. This method requires users to invest some upfront time to load the necessary resources. Although users may experience an initial delay, it minimizes interruptions during gameplay and enhances overall comfort.

In this particular game, we have opted for the second approach. The images comprise textures for in-game objects in PNG or JPG format, while the sounds are in MP3 or WAV format. Additionally, we utilize JSON to store question data and level information.

The resource loading process is initiated by an object called *AssetManager* when the game commences. The *AssetManager* takes charge of loading images and sounds during initialization. Prior to introducing the *AssetManager*, we undergo a process of converting resources into data URIs. The identified resources are transformed into data URIs and aggregated within an *assetData* object. This approach optimizes resources by embedding them as data URIs within the application or game's source code, eliminating the need to load resource files (such as images or sounds) from the hard disk each time they are required. This optimization facilitates faster resource loading by reducing disk access. Moreover, converting resources into data URIs and packaging them helps reduce the overall size of the game.

The aforementioned *AssetManager* is responsible for loading the resources and notifying the game, represented by a Game object, when the resources are ready to commence the game.

EVALUATION AND DISCUSSION

Evaluating the Travelix Game

With this approach, we have shortened the spiral process: design - execute customer experience - iterate, by directly focusing on the relevant technical features aligned with customer requirements from the outset. We assess the effectiveness of the Travelix application as an indirect measure of the effectiveness of the proposal approach. To evaluate the effectiveness of Travelix, we conduct assessments of the achieved effectiveness and gather customer feedback during product usage. As for the first objective, we observe the result of players' theoretical driving tests before and after experiencing the Travelix product. For the second objective, we design a set of five questions to gather player feedback after experiencing the product. The questions are described in Tab. 5. In addition, overall game assessment by collecting star feedback.

Tab. 5. Questionaire

No	***Questions***
1	Do you find the game to be beneficial?
2	Do you find the game captivating?
3	Is the level of interaction satisfactory?
4	Have you gained any knowledge or learned anything from playing the game?

Test Scenario:

Do B2 examination

Experiment Travelix game

Do B2 examination again

Complete the questionnaire

DISCUSSION

We collected feedback from 30 players.

For these players, 89% of them improved their scores when taking the B2 test after experiencing the Travelix. Based on a 5-point scale (ranging from 1 to 5), the feedback ratings regarding the game's usefulness are exceptionally high (for the first three questions). Every single response, accounting for 100% of the total (Fig. 13 left), indicated that the product is helpful, with a score of 3 or higher. Among these responses, more than 60% of the participants gave a perfect rating of 5 out of 5. Moreover, over 80% of the feedback emphasized the game's appeal and good interaction with *score* >= 3 (Fig. 13 middle and right, respectively).

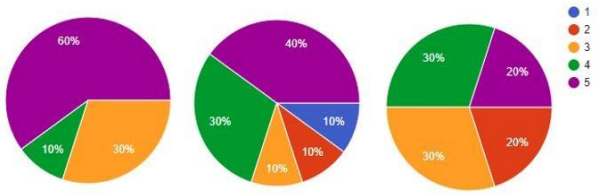


Fig. 13. Players' feedback.

In general, the game received widespread acclaim from players, with its incorporation of useful features, convenience, ease of play, and appealing design. This positive feedback has instilled a sense of confidence among the majority of players that the game would be successful if implemented in real life. Remarkabely, 43% of the evaluations for Travelix awarded it a 5-star rating (Fig. 14), highlighting a substantial level of satisfaction and a highly positive reception.

For question number 4, we received feedbacks such as: "Preparing for the B2 exam, so I find this test preparation method not boring; "It is useful because it provides fundamental driving theory", "It supports

the process of studying for the B2 license exam.

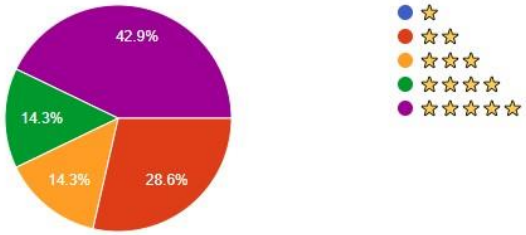


Fig. 14. Star Score.

CONCLUSION

In conclusion, learning through games has proven to be a highly effective method for individuals of all ages seeking to expand their knowledge. The interactive and engaging nature of games provides an immersive learning experience that facilitates better retention and understanding of information.

When it comes to developing digital games, particularly those with educational purposes, utilizing the HoQ approach serves as a valuable framework. By employing HoQ principles, the product design process becomes

more streamlined, enabling developers to focus on the essential features that contribute to the excellence of the final product. This approach ensures that the game meets the highest standards of quality and effectiveness in delivering educational content.

One notable example of the successful implementation of learning games is the Travelix game. Designed to assist individuals in preparing for theoretical driving license exams, the Travelix game has received positive feedback from its user community. The game's refinement and ongoing development have validated its effectiveness in helping users acquire the necessary knowledge and skills for passing their driving exams. This positive reception highlights the potential impact and value of well-designed learning games in supporting educational objectives.

In summary, learning through games offers a promising avenue for knowledge expansion across various age groups. Applying a HoQ approach to game development enables efficient product design while prioritizing crucial elements that contribute to a game's excellence. The positive feedback received for the Travelix game further emphasizes the significance of well-crafted learning games in assisting users in achieving their educational goals. As technology continues to advance, the continued exploration and refinement of learning games hold immense potential for enhancing the learning experience and promoting knowledge acquisition in an enjoyable and interactive manner.

ACKNOWLEDGEMENTS

This work has been supported by VNU University of Engineering and Technology under project number CN24.14 "Solution of 3D floor plan generation".

REFERENCES

J. R. Hauser, D. Clausing, The house of quality, Analytics And Data Science. Z. Horváth, L. Buics, P. Foldesi, B. Eisingerné Balassa, The role of hungarian traffic rules education and examination system – a quality function deployment approach, Acta Polytechnica Hungarica 19 (2022) 7–26. doi:10.12700/APH.19.7.2022.7.1.

J. Ningal, C. Oños, Traffic education, publicity and training in road safety: Basis for road safety and accident awareness program, International Journal of Current Research Vol. 13 (2021) 17870–17875. doi:10.24941/ijcr.41750.06.2021.

J. Terninko, Step-by-step qfd: Customer-driven product design, second edition 2nd edition, CRP Press.ISBN-10: 1138440574 ISBN-13 : 978-1138440579 (2017) 17870–17875.