ΠΑΝΕΠΙΣΤΗΜΙΟ
ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ

Σχεδιασμός και Ανάπτυξη Ψηφιακού Παιχνιδιού Μάθησης

Design and Development of Digital Learning Game

Θεοφίλου Στυλιανός
Αριθμός Μητρώου: 1072791

Επιβλέπων
Σιντόρης Χρήστος, Ε.ΔΙ.Π.

Μέλη Επιτροπής Αξιολόγησης
Σγάρμπας Κυριάκος, Καθηγητής

Πάτρα
Δεκέμβριος 2024

ΠΙΣΤΟΠΟΙΗΣΗ

Πιστοποιείται ότι η διπλωματική εργασία με θέμα

Σχεδιασμός και Ανάπτυξη Ψηφιακού Παιχνιδιού Μάθησης

του φοιτητή του τμήματος Ηλεκτρολόγων Μηχανικών & Τεχνολογίας Υπολογιστών

Θεοφίλου Στυλιανού

Αριθμός Μητρώου: 1072791

παρουσιάστηκε δημόσια και εξετάστηκε στο τμήμα Ηλεκτρολόγων Μηχανικών & Τεχνολογίας Υπολογιστών στις

.... / .... / ....

Ο Επιβλέπων

Σιντόρης Χρήστος, Ε.ΔΙ.Π.

Ο Διευθυντής του Τομέα

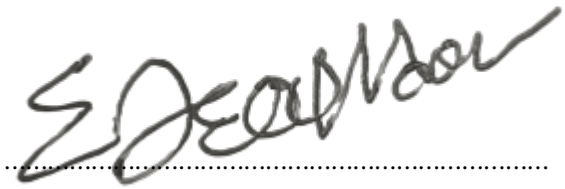
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ

Το περιεχόμενο αυτής της εργασίας δεν απηχεί απαραίτητα τις απόψεις του τμήματος, του επιβλέποντα, ή της επιτροπής που την ενέκρινε.

Υπεύθυνη Δήλωση

Βεβαιώνω ότι είμαι συγγραφέας αυτής της διπλωματικής εργασίας, και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην διπλωματική εργασία. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης, βεβαιώνω ότι αυτή η διπλωματική εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τις απαιτήσεις του προγράμματος σπουδών του τμήματος Ηλεκτρολόγων Μηχανικών & Τεχνολογίας Υπολογιστών.

(Υπογραφή)



.....

Θεοφίλου Στυλιανός

Σύνοψη

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.

Λέξεις-κλειδιά: Μάθηση βασισμένη στο παιχνίδι, Κβαντική Υπολογιστική, Κβαντική Μηχανική, Παιχνίδι για κινητά, Flutter

Abstract

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.

Keywords: Game-Based Learning, Quantum Computing, Quantum Mechanics, Mobile Game, Flutter

Ευχαριστίες

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.

Περιεχόμενα

Κατάλογος πινάκων

Κατάλογος σχημάτων

1 1 Introduction

1.1 Motivation

Quantum computing utilizes the principles of quantum mechanics to process information and solve complex problems exponentially faster that classical computers. Quantum bits can exist in multiple states at the same time, offering great computational power, beyond the limits of classical computers. The development and widespread use of quantum computing can help in fields such as pharmaceuticals, cryptography, artificial intelligence, materials science and more. (Taylor 2024), (Nagappan 2023), (Wootton 2017) Since quantum computing represents a new era for computer science, opening up new prospects for accelerating scientific discoveries, learning the basic principles of quantum computing is extremely important. The reason for my involvement with quantum computing was the course ‘Quantum Computing’, which was offered by the Electrical and Computer Engineering Department at the University of Patras.

1.2 Research Objectives

The aim of this thesis is to create an educational game that will familiarize the learners with the basic principles of quantum computing, such as quantum bits and quantum gates. Then we will define some specifications that we would like the game to meet.

First, we want to design a simple and accessible educational game with few rules, clear objectives and easy to understand mechanics. We want learners to focus on the learning object and not get distracted or tired by the complexity of the game.

The game will be aimed at university students interested in quantum computing. For this reason, it will not go into depth, but will introduce basic concepts, such as quantum registers or some basic quantum gates. This is the second requirement for the game manual.

In addition, it is desirable that the game can be played at any time and in any place, without the need for equipment or a computer, simply by using their mobile phone. The third objective is that the user should be able to play even when they have limited time (e.g. travelling, waiting for public transport, etc.), so it should be light and take a short time to play.

The fourth and final objective is to make the game suitable to be played in the context of a university lecture. The game could be distributed to students during the introductory lecture of the course, or it could be used as a tool to test their knowledge from a lecture - for example, whether they understand the operation of the basic quantum gates. For this reason, it should have features that test learners' knowledge, such as a quiz.

1.3 Thesis Structure

TODO: * ta link na einai footnote ana selida * add a diagram with thesis structure

2 Literature Review

Once the goal of creating an educational game for learning quantum computing was established, a review of the available literature began. The aim of the literature review was to evaluate Game-Based Learning (GBL) as a learning modality, to document GBL techniques, and to find games related to quantum physics and quantum computing.

In this chapter, we summarize the conclusions of the literature review on GBL as a learning approach in different sciences. We will also list games related to quantum physics and quantum computing, both educational and non-educational, which have been the inspiration for the creation of our game.

2.1 Computer Science Games for Higher Education

In order to choose a game genre, we studied the available literature to see the approach of other researchers and to choose the most appropriate genre, according to the specifications we set in section 1.3.

According to *Batisstella* (2016), educational games have to be engaging and motivate learners to keep playing. Their study looked at more than 100 games related to computer science, the majority of which were digital games. 97% of the digital educational games were computer games, while only 1 game was designed for mobile devices. This observation suggests a significant lack of games available for mobile devices.

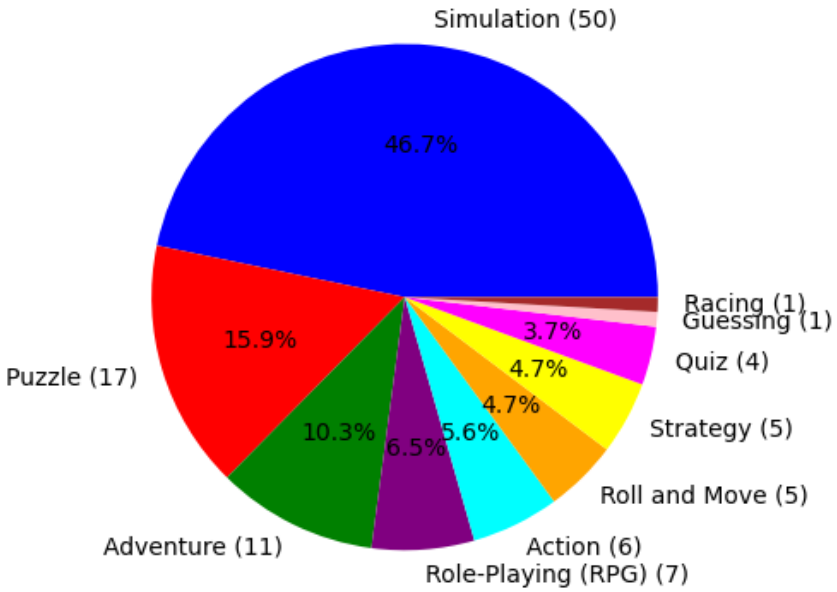


Figure 2.1. Distribution of games per genre.

The figure above, based on the aforementioned study, shows the distribution of games per genre. The most popular genre is simulation games, followed by puzzle games and adventure games. According to the researchers, most games are single-player that don' t encourage cooperation or healthy competition between learners.

2.2 Educational Puzzle Games

As we explained in σεψτιον 1.3, we want the game to be easy to use and have simple mechanics. Creating a simulation game, which is the most popular genre, may lead to complex mechanics that distract the player from the educational goal. For this reason, we decided to investigate further the second most popular genre: puzzle games.

Research by *Fakokunde* (2021) on 141 subjects showed that puzzle games improved information retention, regardless of the learners' gender and cognitive style. Two other studies on puzzle games on puzzle games, (Idika and Oluwaseyi 2024), (Srijampana, Chebrolu, and Potti 2023), confirm that

this type of game can significantly improve information retention. They also report that they increase learners' active engagement with the material and enhance their understanding of the subject.

2.3 Educational Games for Quantum Mechanics

In the literature we can find many games for quantum mechanics, which have been developed in different contexts. The earliest game for quantum mechanics that can be found in the literature is an Atari arcade game called 'Quantum' created in 1982 (Piispanen et al. 2023). To date, more than 300 games related to quantum mechanics, digital and not, have been created (Piispanen et al. 2024). There are games developed in the context of competitions, as university coursework, by research groups, by universities, by companies, by independent enthusiasts and by professors or professional engineers. By extension, there are games that have not been developed for serious purposes, competition games, games designed for research purposes, commercial games and educational games. (Piispanen et al. 2023), (Seskir et al. 2022) Due to the large number of educational games found in the literature, in this section we will focus on games developed by companies or universities with an educational purpose.

The games presented below are taken from the aforementioned studies and from the catalog on the *kiedos.art* website. (Piispanen 2024)

Πίνακας 2.1: Educational Games for Quantum Mechanics.

Game Name	Creator	Platform	Genre
Quander	Canon Lab Org (by University of Chicago)	Web	Puzzle
Collapsing Qubits	Canon Lab Org (by University of Chicago)	Board Game	Cards
Quantum Computing Playground	Google	Web	Virtual Lab**
The Qubit Game	Google (Quantum AI Lab)	Web	Simulation
Quantum Composer	IBM	Web	Virtual Lab**
Hello Quantum	IBM & University of Basel	Android, iOS	Puzzle

Game Name	Creator	Platform	Genre
Particel in a box	QPlayLearn	Web	Simulation, Puzzle
Quantum Playground	QPlayLearn	Web	Simulation
Q|Cards⟩	QPlayLearn	Board Game, Android, iOS	Cards, Puzzle
TiqTaqToe	QPlayLearn	Web	Strategy
Psi and Delta	QPlayLearn	Web	Platform, Puzzle
Quantum Solitaire	QPlayLearn	Web	Cards
Escape Quantum	QPlayLearn	Web	Puzzle
Potatoes Quest	QPlayLearn	Web	Platform, Puzzle
QWiz	QPlayLearn	Web	Puzzle
Saving Photoland	QPlayLearn	Web	Puzzle
Quantum Odyssey	Quarks Interactive	Web	Puzzle
Quantum 3	Michigan State University	Android, iOS	Puzzle
Quantum Flytrap	National University of Singapore	Web	Virtual Lab**
Quantum Pattern Matching*	Science At Home (by University of Aarhus)	-	Puzzle
Quantum Moves 2	Science At Home (by University of Aarhus)	Windows, macOS, Web	Simulation
Rydbergator	Science At Home (by University of Aarhus)	Windows, macOS, Web	Simulation
Network Game	Science At Home (by University of Aarhus)	Web	Puzzle

Game Name	Creator	Platform	Genre
Meqanic	Science At Home (by University of Aarhus)	Web, iOS	Puzzle
Quantum Kate AR	University of Southern Denmark	iOS	Simulation

[*] Under development.
[**] They are not games in the classical sense, but are treated in the literature as a game-based learning method.

2.4 Educational Mobile Games for Quantum Mechanics

As shown in the table in the previous section, there are 5 educational games for mobile devices. 4 of them are puzzles and 1 is a simulation. We will focus on the study of puzzle games to see how the researchers approached the topic.

Unfortunately, *Hello Quantum*, *Meqanic* and *Q|Cards>* are no longer available in stores, so we were unable to download them to study them. *Meqanic* was available via online simulators on Apple mobile devices (iPhone and iPad), but we were unable to find a working version; on most simulators the game wouldn' t even open, and on others it crashed on the level selection screen.

Quantum 3 is available in stores, so we downloaded it on a device to study it. The game' s home screen has 3 options, 'Play' , 'Options' and 'Tutorial. The ' Options' screen has basic setting for the game, such as volume control and language selection. There is also a button to clear game progress. The tutorial screen provides basic information about the theory presented in the game, as well as the mechanics of the game.

At the beginning of some levels there is a short presentation of the theory and instructions for that level. After the player has completed the level, the score is displayed. The player receives between one and three stars. The maximum score is achieved if the player completes the level in the minimum number of moves. When the level is completed, the player has the option of proceeding to the next level or returning to the home screen.

The game attempts to teach the player the composition of subatomic particles from quarks. Each level is a 2-dimensional match-3 puzzle. The player has to match three polygons of different colors to compose subatomic particles. There is no free choice of levels, a new level is unlocked each time the player completes the previous one. The game has very simple mechanics, as there are only 2 moves, selecting

3 polygons and swapping 2 polygons. It also offers the option of shuffling the board if no move is available.

2.5 Conclusions

As the literature review demonstrates, the number of educational games for quantum computing available and playable for mobile devices is limited. This gap in the market reinforces our desire to develop a game for mobile devices, as it will be easy to engage with the game during a lecture or at times when the user does not have access to a computer, as explained in the introductory chapter as well. Because we are not interested in improving collaboration or other teamwork skills at this stage, we will not consider creating a multiplayer game. Finally, the strong boost to information retention from puzzle games and the possibility of using simple mechanics leads us to the decision to create a puzzle game.

3 Game-Based Learning

As we discussed in the previous chapter, we want to create an educational puzzle game, so we need to consider Game-Based Learning (GBL) techniques to integrate the educational process into the game mechanics. Continuing our literature survey, we will present the benefits of GBL and some of the elements that characterize a GBL process.

3.1 What is Game-Based Learning

Game-based learning is a very old practice; it did not start with the advance of modern technology. It can be defined as the technique of being educated by playing games. It integrates the characteristics and principles of games such as elements of competition, rewards and active user engagement, into learning activities. Games can be an interactive tool that can simplify challenging concepts and help learners understand complex ideas, engaging them into educational content. (Ledda 2012), (Wirtz 2023).

3.2 Benefits of Game Based Learning

First, game-based learning is more appealing to children, as it appears to be a game on the surface, but in the background it has the ability to stimulate children' s curiosity and capture their imagination. It is a friendlier and more accessible mean of engaging young learners with a subject than traditional methods, as it is fun and motivating.

Game-based learning also has the ability to enhance critical thinking and problem-solving, as they involve human instinct to compete and desire to succeed. Because learners often compete with other players, they have to collaborate and share ideas. They must listen to and evaluate the opinions of other players and take into account the tactics of opposing teams.

Games often require users to react quicker to stimuli, make critical decisions in a short period of time and combine knowledge acquired during the game to solve complex problems. Due to their repetitive and interactive nature, they have the ability to improve retention and increase the brain' s capacity to memorize things.

Also, as games are flexible, they can be adapted to different learning styles, levels and paces, meeting individual needs and can also give instant feedback about where gaps in knowledge are or provide specific tasks for the user to help cover these areas. In this way, they can further help learners to identify their strengths and weaknesses.

Comparing games to traditional textbooks, although the latter have been used for many years with success, their revision and renewal takes a long time and is difficult and costly. The cost of reprinting, redistributing and recycling or storing old textbooks must be taken into account. Even in the case of digital textbooks, there is a significant cost of disposal and renewal. By contrast, games are very versatile, their rules can be adapted easily, and their content can be changed quickly to keep pace with technological and scientific progress.

In summary, game-based learning offers a modern, engaging and flexible approach to education. It is a great way to improve learners' critical thinking and problem-solving skills, boost their creativity and keep them engaged and motivated. Also, unlike traditional textbooks, it can be quickly and cost-effectively updated, in order to reflect new information and technological progress.

(Harding 2023), (Wirtz 2023)

3.3 Elements of GBL

The following section is based on *Plass, Homer, and Kinzer* (2015) and *Shi and Shih* (2015).

3.3.1 Game Mechanics

Mechanics determine the genre of the game and are the key element in linking the educational process to the game. They describe the type of activities the learner has to repeat throughout the game and can be of two types; learning mechanics and assessment mechanic.

3.3.2 Visual Aesthetic Design

The Visual Aesthetic Design is the process of representing information with graphic elements (information visualization). It also defines the way in which visualized feedback is provided.

3.3.3 Musical Score

The musical score includes the background sounds, the voices and haptic feedback (such as device vibration). These elements are used to direct player' s attentions to important events, evoke emotions and provide feedback on player' s actions.

3.3.4 Narrative Design

Unlike traditional textbooks, games allow for nonlinear narratives that evolve based on player's choices. A narrative can be a dialogue, a cutscene, a voice-over or other in-game action. Narratives provide information for learning, characters, events, quests and game rules. Their aim is to increase the player' s motivation to continue playing the game.

3.3.5 Incentive System

The incentive system consists of motivational elements that rewards player' s efforts and encourages them. Examples of incentives systems are point systems, leaderboards, stars, badges trophies and any other kind of reward.

3.3.6 Adaptivity

Adaptivity refers to the ability of the game to adjust to each learner' s profile, enhancing engagement and learning outcomes. To achieve adaptivity, the game should provide mechanisms to modify game elements, such as problem complexity, guidance and feedback, according to player' s needs.

3.3.7 Graceful Failure

Graceful failure refers to the idea that failure is not seen as a negative outcome, but rather as an essential and constructive part of the leadership process. The game can reduce the consequences of failure by encouraging players to take risks, explore and experiment, leading to increased motivation and engagement.

3.4 Pitfalls

There are some potential pitfalls that someone may face when designing game-based learning activities (Moore-Russo, Wiss, and Grabowski 2018). First, game elements may not be connected to learning objectives. Quite often the tasks are introduced without planning how they support the learning objectives. Also, sometimes the learning objectives are connected, but the context of the game is not. As a result, learners cannot retain knowledge or relate it to the subject. For game-based learning activities to be successful, learners must be able to retain and apply the knowledge they have been exposed to through the game. Furthermore, when students are focused on passing high stakes exams, participating in a gamified environment can be seen as an unnecessary obstacle (Berkling and Thomas 2013). Learners who are used to learning and assessment through conventional teaching methods

and are focused on achieving high grades may feel deprived in a gamified environment. They will fill uncertainty, discomfort and may find it difficult to adapt to new rules (Nicholson 2012). Finally, using leaderboards to rank learners in comparison with others, many learners may feel disappointed or face discrimination (Domínguez et al. 2013).

4 Quantum Computing

In the previous chapter we introduced the techniques and the benefits of game-based learning. We will then discuss the basic theory around quantum computing, which provided the information needed to design the theory slides presented in the game and on which the game mechanics are based.

Quantum computers are computers that perform calculations by taking advantage of quantum phenomena, such as superposition and entanglement. The quantum properties of the microcosm provide the ability to store and process larger amounts of information and perform specific calculations at higher speeds than conventional computers. Information is processed using quantum gates and quantum algorithms, analogs of which cannot exist in conventional computers. (Karafillidis 2015), (Wikipedia contributors 2024f)

4.1 Quantum Bits

Quantum computers store information as bits. A quantum bit is a two-state system based on properties of the microcosm such as spin, energy state or the way particles oscillate and is the basic unit of information storage. (Wikipedia contributors 2024f)

State 0 is represented as $|0\rangle$ and state 1 as $|1\rangle$ and are called basic or basis states and are orthogonal to each other. Because the two states belong to a vector space, Hilbert space, they can be multiplied by a number and added together, and the result will be a valid state. Superposition is based on this fact. (Wikipedia contributors 2024f)

Each valid state can be written as:

$$|q\rangle = a|0\rangle + b|1\rangle$$

where a and b are called probability amplitudes and are complex numbers. The magnitudes of a and b are always less than one, and it holds that:

$$|a|^2 + |b|^2 = 1$$

The two basic states of a qubit can be represented as matrices:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

The state $|q\rangle = a|0\rangle + b|1\rangle$ can be represented using matrices as follows:

$$|q\rangle = a \begin{pmatrix} 1 \\ 0 \end{pmatrix} + b \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}$$

The probability amplitudes, a and b, are complex numbers, so we can write the last equation in a more general form:

$$|q\rangle = e^{i\phi_a} \sin\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi_b} \cos\left(\frac{\theta}{2}\right)|1\rangle \Leftrightarrow$$

$$|q\rangle = e^{i\phi_a}\left(\sin\left(\frac{\theta}{2}\right)|0\rangle + e^{i(\phi_b - \phi_a)} \cos\left(\frac{\theta}{2}\right)|1\rangle\right)$$

If the global phase term $e^{i\phi_a}$ is omitted while the phase difference $\phi_b - \phi_a$ is called ϕ, the last equation simplifies to the following:

$$|q\rangle = \sin\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi} \cos\left(\frac{\theta}{2}\right)|1\rangle$$

The angle θ determines the magnitude of the probability amplitudes a and b, while the angle ϕ is called the phase angle and does not affect the measurement outcome. This means that two qubits differing only by a phase angle cannot be distinguished by a measurement. However, the phase angle should not be omitted, as it affects quantum computations.

The term $e^{i\phi_a}$ is called the global phase and is a mathematical artifact that can be safely ignored.

(Karafillidis 2015)

4.1.1 Superposition

Superposition is based on the addition of two states, similar to how we would add two waves, and gives quantum computers the ability to perform parallel computations.

A classical bit can have two distinct states, 0 or 1, and can be stored by any system that has two distinct states, closed or open. All information is analyzed, stored, and processed as a sequence of 0s and 1s by classical computers.

A quantum system can be in both states simultaneously. Before we measure its state, it tends to be in the state $|q\rangle$ or $|1\rangle$. The quantum state $|q\rangle = a|0\rangle + b|1\rangle$ is a superposition of the two basic states. Measuring the state of the system destroys the superposition, so a qubit can only be found in one of the two basis states.

The outcome of the measurement is impossible to predict with certainty, as all we know are the probabilities of it being in one of the two basis states, which are given by the square of the probability amplitudes, a and b.

(Amazon Web Services, Inc. 2024b), (Karafillidis 2015)

4.1.2 Quantum Entanglement

Quantum entanglement is defined as the state of two quantum systems when it cannot be written as a tensor product of their basic states. Quantum entanglement is a physical resource that can be used for the development of quantum algorithms and the execution of quantum computations. There is no classical analog of this state. When two systems are entangled, measuring the state of one reveals the state of the other, regardless of the distance between them. Essentially, the state of one quantum system depends on the state of the other. (Amazon Web Services, Inc. 2024b), (Karafillidis 2015)

4.1.3 Decoherence

Superposition states are unstable and decohere so that the system becomes stable. Decoherence is an irreversible process and can be caused by external factors such as an increase in temperature or radiation. (Amazon Web Services, Inc. 2024b)

4.2 Quantum Registers

Quantum registers are an array of qubits used as memory and represent a superposition of 2^n quantum states, where n is the number of qubits. The numbering of the qubits is done from right to left or from bottom to top. Quantum computers do not have classical circuits but perform operations by acting on bits that are within quantum registers.

The state of a quantum register with n qubits is defined as the tensor product of the states of the qubits that comprise it:

$$|q_r\rangle = |q_{n-1}\rangle \otimes ... \otimes |q_1\rangle \otimes |q_0\rangle$$

(Karafillidis 2015)

4.3 Quantum Gates

https://quantum.microsoft.com/en-us/insights/education/concepts/single-qubit-gates#:~:text=The%20Y%20gate%20perf

https://www.sharetechnote.com/html/QC/QuantumComputing_Gate_X.html

Quantum gates are not physical systems like classical gates, but physical processes applied to quantum bits (qubits) and registers that change their state. Additionally, information does not pass through quantum gates as it does with classical gates, since they are not part of any physical circuit with conductors. The information remains within the quantum registers, and the gates act on them by rotating their state vectors. A quantum gate can be a laser pulse or a magnetic field. (Karafillidis 2015), (Wikipedia contributors 2024f)

Since quantum bits are vectors in Hilbert space, quantum gates must be operators in Hilbert space. However, not all operators in Hilbert space are suitable for representing quantum gates. For an operator to be characterized as a quantum gate, it must not change the length of the state vector, only its angle, and it must not change the values of the inner products between state vectors. Thus, only unitary operators can constitute quantum gates. (Karafillidis 2015), (Wikipedia contributors 2024f)

The table below presents some of the most well-known quantum gates and the effects of their actions.

Πίνακας 4.1: Quantum Gates. (Wikipedia contributors 2024g)

Name(s)	# of qubits	Symbol(s)	Matrix
Identity, No-Op	any	I	$\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$
Pauli-X, NOT, Bit Flip	1	X, NOT, σ_x	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
Pauli-Y	1	Y, σ_y	$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$

Name(s)	# of qubits	Symbol(s)	Matrix
Pauli-Z, Phase Flip	1	Z, σ_z	$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$
Phase	1	S, P, $\sqrt{Z}$	$\begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$
Hadamard	1	H	$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$
Controlled-NOT, Controlled-X, Controlled Bit Flip	2	CNOT, XOR, CX	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$
Controlled-NOT, Controlled-X, Controlled Bit Flip	2	CNOT, XOR, CX	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$
Swap	2	SWAP	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

4.3.1 Identity

The identity quantum gate does not affect the state of the qubit. Its symbol is often omitted in quantum circuits.

Πίνακας 4.2: Identity Gate Truth Table.

$\|q_i\rangle$	$\|q_o\rangle$
$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	$\|1\rangle$

$\|q_i\rangle$	$\|q_o\rangle$
$a\|0\rangle + b\|1\rangle$	$a\|0\rangle + b\|1\rangle$

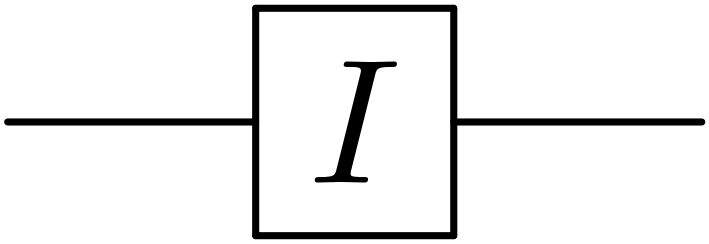


Figure 4.1. Identity Gate Circuit Diagram.

4.3.2 Pauli-X

The quantum X gate functions similarly to the classical NOT gate, rotating the state vector by 180 degrees around the X-axis. This operation changes the state from $|0\rangle$ to $|1\rangle$ and vice versa.

Πίνακας 4.3: Pauli-X Gate Truth Table.

$\|q_i\rangle$	$\|q_o\rangle$
$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$
$a\|0\rangle + b\|1\rangle$	$a\|1\rangle + b\|0\rangle$

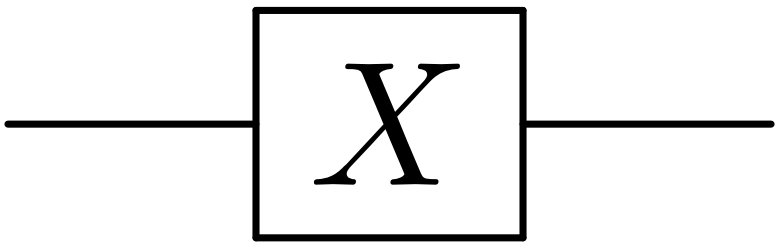


Figure 4.2. Pauli-X Gate Circuit Diagram.

4.3.3 Pauli-Y

The quantum Y gate operates similarly to the X gate but rotates the state vector by 180 degrees around the Y-axis. It changes the state from $|0\rangle$ to $|1\rangle$ and vice versa, and additionally shifts the phase of the $|0\rangle$ state by 90 degrees and the phase of the $|1\rangle$ state by -90 degrees.

Πίνακας 4.4: Pauli-Y Gate Truth Table.

$\|q_i\rangle$	$\|q_o\rangle$
$\|0\rangle$	$i\|1\rangle$
$\|1\rangle$	$-i\|0\rangle$
$a\|0\rangle + b\|1\rangle$	$ia\|1\rangle - ib\|0\rangle$

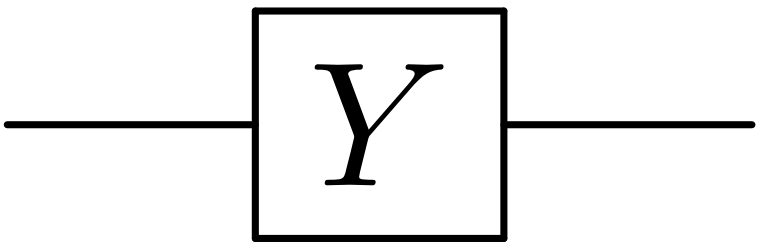


Figure 4.3. Pauli-Y Gate Circuit Diagram.

4.3.4 Pauli-Z

The quantum Z gate rotates the state vector by 180 degrees around the Z-axis. It shifts the phase of the $|1\rangle$ state by 180 degrees and does not affect the state of the $|0\rangle$ state.

Πίνακας 4.5: Pauli-Z Gate Truth Table.

$\|q_i\rangle$	$\|q_o\rangle$
$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	$-\|1\rangle$
$a\|0\rangle + b\|1\rangle$	$a\|0\rangle - b\|1\rangle$

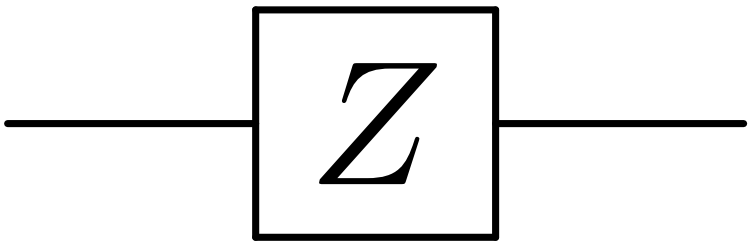


Figure 4.4. Pauli-Z Gate Circuit Diagram.

4.3.5 Phase

The S gate shifts the phase of the $|1\rangle$ state by 90 degrees.

Πίνακας 4.6: Phase Gate Truth Table.

$\|q_i\rangle$	$\|q_o\rangle$
$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	$i\|1\rangle$
$a\|0\rangle + b\|1\rangle$	$a\|0\rangle + ib\|1\rangle$

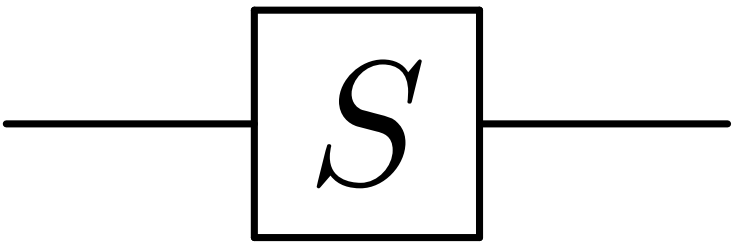


Figure 4.5. Phase Gate Circuit Diagram.

4.3.6 Hadamard

The Hadamard gate, when acting on a qubit in one of the two basis states, places it in a superposition of the two basis states. Conversely, when it acts on a qubit that is in a superposition of the two basis states, it returns it to one of the basis states.

Πίνακας 4.7: Hadamard Gate Truth Table.

$\|q_i\rangle$	$\|q_o\rangle$
$\|0\rangle$	$\frac{1}{\sqrt{2}}\|0\rangle + \frac{1}{\sqrt{2}}\|1\rangle$
$\|1\rangle$	$\frac{1}{\sqrt{2}}\|0\rangle - \frac{1}{\sqrt{2}}\|1\rangle$
$\frac{1}{\sqrt{2}}\|0\rangle + \frac{1}{\sqrt{2}}\|1\rangle$	$\|0\rangle$
$\frac{1}{\sqrt{2}}\|0\rangle - \frac{1}{\sqrt{2}}\|1\rangle$	$\|1\rangle$
$a\|0\rangle + b\|1\rangle$	$\frac{1}{\sqrt{2}}(a+b)\|0\rangle + \frac{1}{\sqrt{2}}(a-b)\|1\rangle$

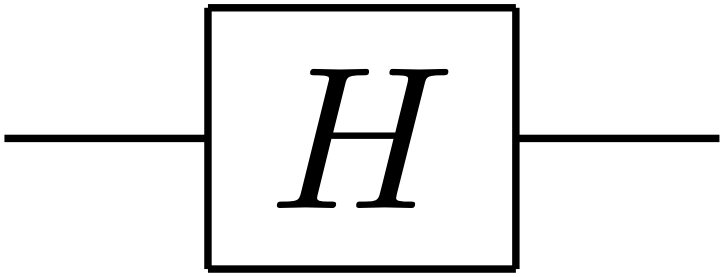


Figure 4.6. Hadamard Gate Circuit Diagram.

4.3.7 Controlled-NOT

The CNOT gate inverts the target bit when the control bit is set to 1. It functions as an X gate controlled by the control bit.

Πίνακας 4.8: Controlled-NOT Gate Truth Table - Second bit as control bit.

$\|q_{control}q_{target}\rangle$	$\|q_{control}q_{target}\rangle$
$\|00\rangle$	$\|00\rangle$
$\|01\rangle$	$\|01\rangle$
$\|10\rangle$	$\|11\rangle$
$\|11\rangle$	$\|10\rangle$
$a\|00\rangle + b\|01\rangle + c\|10\rangle + d\|11\rangle$	$a\|00\rangle + b\|01\rangle + c\|11\rangle + d\|10\rangle$

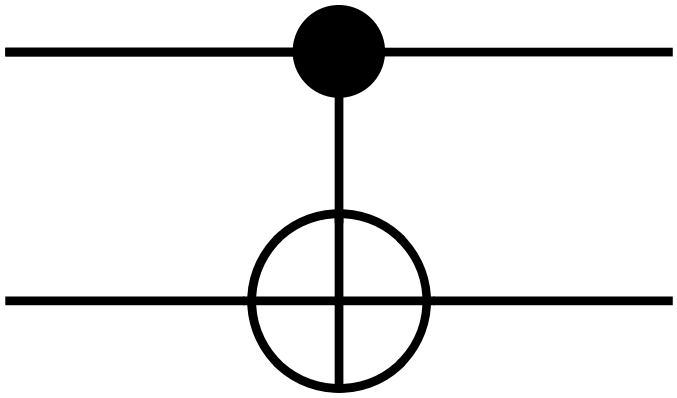


Figure 4.7. CNOT Gate Circuit Diagram - Second bit as Control bit.

Πίνακας 4.9: Controlled-NOT Gate Truth Table - First bit as control bit.

$\|q_{control}q_{target}\rangle$	$\|q_{control}q_{target}\rangle$
$\|00\rangle$	$\|00\rangle$
$\|01\rangle$	$\|11\rangle$
$\|10\rangle$	$\|10\rangle$
$\|11\rangle$	$\|00\rangle$
$a\|00\rangle + b\|01\rangle + c\|10\rangle + d\|11\rangle$	$a\|00\rangle + b\|11\rangle + c\|10\rangle + d\|11\rangle$

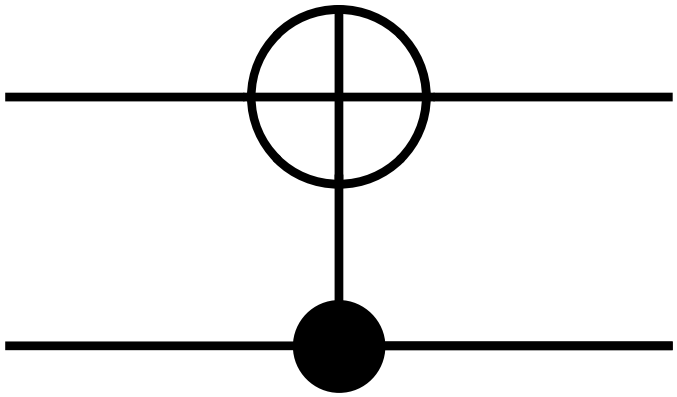


Figure 4.8. CNOT Gate Circuit Diagram - First bit as Control bit.

4.3.8 Swap

The SWAP gate exchanges the states of two qubits.

Πίνακας 4.10: SWAP Gate Truth Table.

$\|q_{i1}q_{i0}\rangle$	$\|q_{01}q_{o0}\rangle$
$\|00\rangle$	$\|00\rangle$
$\|01\rangle$	$\|10\rangle$
$\|10\rangle$	$\|01\rangle$
$\|11\rangle$	$\|11\rangle$
$a\|00\rangle + b\|01\rangle + c\|10\rangle + d\|11\rangle$	$a\|00\rangle + b\|10\rangle + c\|01\rangle + d\|11\rangle$

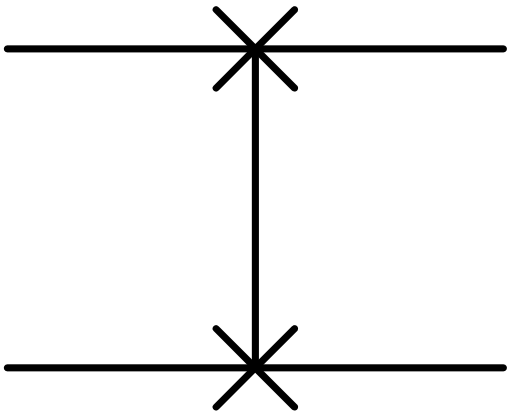


Figure 4.9. SWAP Gate Circuit Diagram.

References

The details about quantum gates and the tables are based on the following sources:

- (Karafillidis 2015)
- (Microsoft Corporation 2024a)
- (Microsoft Corporation 2024b)
- (Wikipedia contributors 2024g)

The images are sourced from (Wikipedia contributors 2024g).

5 Mobile Application Development

Having completed the presentation of game-based learning and the basic theory around quantum computing, it is time to evaluate the options available for the development of the application, that is, the programming language, the framework and the game engine. First, we will define mobile game development and present the available application types, and then we will review the different programming languages and available frameworks.

Mobile application development is the process of creating software applications that run on mobile devices. The software can be preinstalled on the device, downloaded from an app store or accessed through a web browser. (Amazon Web Services, Inc. 2024a), (IBM Corporation 2024)

Mobile games are digital games designed for mobile devices. They can utilize mobile sensors and hardware (e.g. accelerometers, GPS etc.), or even external peripherals, such as gaming controllers and AR/VR headsets.

5.1 Platforms

There are two dominant operating systems for mobile devices, Google' s Android and Apple' s iOS. iOS is used only on Apple devices while Android is used by several manufacturers.

Developing applications on each of these platforms requires the use of different software development kits (SDKs). There are four approached to mobile app development, which are compared below.

5.1.1 Native Applications

A native mobile application is a software application that is designed for a specific operating system platform (Rouse 2024). Native mobile apps can only work on the platform that they are designed for, because they use the programming languages, frameworks and interfaces that are platform-specific. They run directly on the operating system, so they tend to perform better than other applications that require interaction with the device' s operating system or hardware.

Because native apps are compiled directly into machine code, there have to be a different code base for each version of the same application (i.e. iOS or Android version). This is a requirement that significantly

increases the cost and time of development and maintenance. (Amazon Web Services, Inc. 2024a), (Rouse 2024)

Native android apps are built with the Android SDK and use Java or Kotlin. On the other hand, native iOS apps are built with the iOS SDK and use Swift or Objective-C.

5.1.2 Cross-Platform Applications

Cross-platform apps have the ability to operate on different operating systems with little to no modification. Because the use universal coding languages and frameworks, cross-platform apps can run on iOS and Android using the same codebase. These coding languages and frameworks hide from developers the underlying differences between operating systems. (Marshall 2024)

Multiplatform apps reduce the cost for building and maintaining an application that targets different platforms. On the other hand, there are some performance issues and the access to device-specific features is limited, because they are not interacting directly with the operating system or the hardware. (Marshall 2024), (Amazon Web Services, Inc. 2024a)

According to Statista, the most popular frameworks for multiplatform development are Flutter and React-Native and Kotlin Multiplatform. (Vailshery 2024)

5.1.3 Hybrid-Web Applications

A hybrid-web application combines the elements of native and web apps. They are essentially web apps that have a native app shell. They are build with standard web technologies, like JavaScript and HTML and are bundled as native app packages. Hybrid apps are executed inside a container, which wraps the applications and acts as a bridge between the application and the operating system. (Techtarget contributor 2023), (Kohout 2016)

Although these apps reduce development and maintenance cost and user experience is very good, their performance is very low, as they cannot take advantage of many native device features. (Amazon Web Services, Inc. 2024a), (Rouse 2024)

5.1.4 Progressive Web Applications

Progressive Web Apps (PWAs) skip App Store delivery and conventional installation processes - they are accessible via a URL. PWAs are web apps that use browser capabilities to provide an app-like user experience, so they are written using web technologies, such as JavaScript and HTML.

PWAs are able to overcome certain disadvantages of Hybrid Web Apps, as they have better performance and more extensive access to device features. They also have low development and maintenance cost,

but the app capabilities are restricted by the browser the use. (Amazon Web Services, Inc. 2024a), (Rouse 2024)

5.2 Languages and Frameworks

5.2.1 Android

Java

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. Java applications are compiled in byte-code that can run on any Java virtual machine, meaning that compiled Java code can run on all platforms without the need to recompile. It was initially released in 1995. Although Android is built on the Linux kernel, which is largely written in C, the Android SKD uses the Java language as the basis for its applications. (Wikipedia contributors 2024a)

Kotlin

Kotlin is a high-level, statically typed, general-purpose programming language with type interference. Kotlin is designed to interoperate fully with Java and the JVM version of Kotlin' s standard library depends on the Java class library. Kotlin, released by JetBrains in 2016, aims to address Java' s shortcomings and enhance development productivity. (Wikipedia contributors 2024b), (Fehervari 2024)

Comparison

Both Java and Kotlin compile to byte-code for the JVM, offering similar performance. Java applications tend to consume more memory, while Kotlin has more efficient memory management, with inline and extension functions that can reduce memory footprint. Kotlin has better startup time due to type interference, leading to faster initialization. Both languages support multithreading, but Kotlin' s co-routines simplify concurrent code handling. Kotlin offers modern features and capabilities for Android development while Java has stronger community support. (GeeksforGeeks contributor 2024), (Fehervari 2024), (Medium contributor 2023)

5.2.2 iOS

Objective-C

Objective-C is a high-level, general-purpose, object-oriented programming language that first appeared in 1984. It is influenced by C and Smalltalk and it was primarily selected by NeXT for NeXTSTEP operating system. Apple chose Objective-C as the main programming language for iOS and macOS, because macOS was based on NeXTSTEP. (Wikipedia contributors 2024c)

Swift

Swift is a high-level, general-purpose, multi-paradigm programming language created in 2010 by Apple. Swift is intended to support the core concepts of Objective-C, but in a safer way. It compiles to byte-code and uses an LLVM compiler. (Wikipedia contributors 2024d)

Comparison

Objective-C has more complex and verbose syntax than Swift. It uses square brackets and has longer syntax for method and property definitions, while Swift is more concise and readable, with a syntax that resembles natural language. Swift is also significantly faster than Objective-C and offers a modern framework (SwiftUI) to build user interfaces. On the other hand, Objective-C has many well-documented, third-party frameworks and is well-tested and more stable. Also, it is a superset of C, so it works smoothly with C and C++ code. (Kaur 2023), (Popko 2024)

5.2.3 Cross-Platform Frameworks

Flutter

Flutter is a user interface (UI) SDK developed and released by Google in 2017. It can be used to create natively compiled mobile, web and desktop apps from a single codebase. It uses its own rendering engine to draw widgets on the screen, unlike other UI frameworks that rely on the platform' s rendering engine or manipulate the platform' s built-in UI stack. Flutter also provides access to native APIs. The Dart programming language is used to write applications in Flutter and the applications are compiled ahead-of-time (AOT) on all platforms except the web, where the code is transpiled to JavaScript or WebAssembly. (Wikipedia contributors 2024e), (JetBrains s.r.o. 2024), (Medium contributor 2024)

React-Native

React-Native is a UI SDK released by Meta Platforms (formerly Facebook Inc.) in 2015 and can be used to develop apps for mobile devices, Android TV, tvOS, web applications and desktop applications. Its components wrap existing native code and can interact with native APIs. React-Native apps are written in JavaScript or TypeScript. (Wikipedia contributors 2024h), (JetBrains s.r.o. 2024), (Medium contributor 2024)

Comparison

React-Native is easier to learn, because it uses JavaScript as a programming language and has greater community support. On the other hand, Flutter has better documentation and its command line interface (CLI) offers tools that allow Continuous Integration (CI) and Continuous Development (CD) to be created more easily than React-Native. Flutter' s CLI also offers the ability to automate application deployment in the app stores. Both frameworks feature hot-reload functionality, which allows developers to see changes instantly while modifying their code, without having to recompile.

Flutter uses its own on widgets and libraries and its own rendering engine and compiles directly to native code, while React-Native depends on the underlying platform and uses multiple JavaScript layers before compiling to native code. Also, React-Native requires the developers to use third-party libraries, both for development and testing. These facts make Flutter significantly faster than React-Native, while allowing Flutter to use less CPU and memory, have a smaller package size and have more consistent UI across platforms. (Shah 2024), (Bat 2024)

5.3 Platform, Framework and Game Engine Choice

The application that will be developed will be a cross-platform mobile application, in order to target users using both Android and iOS. We chose to develop a cross-platform application to get optimal performance, not rely on browser limitations and have a single codebase. This choice also reduces development and maintenance costs and provides a great user experience.

Between Flutter and React-Native, we chose to use Flutter, because it is lighter, faster, produces smaller packages and automates the application deployment in the stores. Also, Flutter has its own game engine, the *Flame Engine*, which will make the development of our game easier.

In addition, Flutter has libraries (e.g. qartvm, quantools) for performing quantum calculations and simulating quantum circuits, which may be useful in the game development process, while there are no corresponding libraries for React-Native.

6 Conceptual Design & Development

Having explored the available educational games for quantum computers, studied GBL techniques and the basic theory around quantum computers, and chosen the framework and game engine we will use to develop the application, it is time to discuss the process that was followed to design the game up to the creation of the early prototype. Furthermore, in the last section of this chapter we will present the development timeline. The diagram below shows the process of conceptual design and implementation of the application prototype.

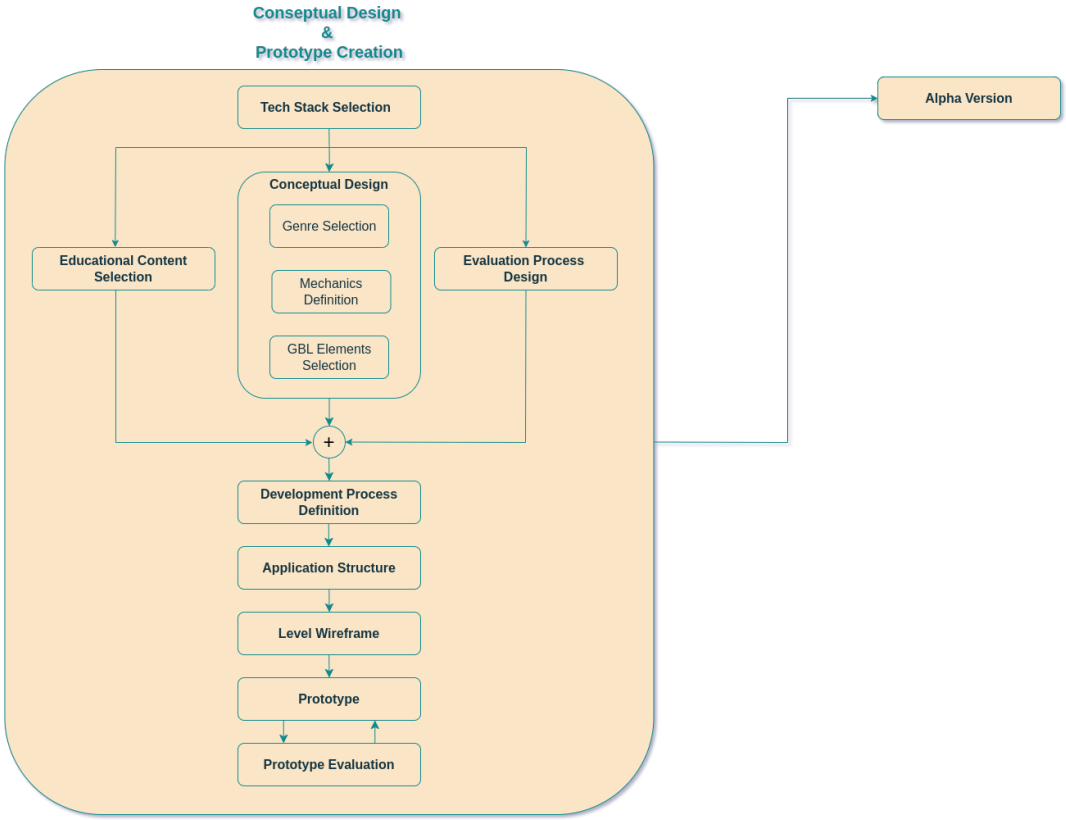


Figure 6.1. Design & Prototype Creation Flow.

6.1 Genre Selection

The first step in designing the game was to determine the gender. Based on the findings in Chapter 2, we decided to make a puzzle game, as it is suitable for increasing information retention while allowing for simple mechanisms.

6.2 Mechanics Definition

The second step was to define the game mechanics. We reasoned that a 2D shooting game could have simple mechanisms, especially if the effect of gravity or other external factors on the missiles is not taken into account. This thinking led us to the decision to create a game with missiles being launched into space from a spaceship. We immediately thought that there was a similar computer game that we could use as a blueprint: *Space Invaders*

We then had to decide which element of the game was appropriate to introduce the concepts of quantum computing. We could control 3 variables, the position of the spaceship, the position of the enemy spaceships and the state of the missiles. In terms of controlling the state of the missiles, we thought the game would be similar to Tetris, which is still a puzzle game, but it can get tedious and eventually distract the players from the learning objective, as they have a limited time to manipulate the state of the missiles, so they may randomly affect them, with no time to recall the knowledge they have gained. After rejecting the missile state control, we considered the spaceship position control and decided to control our own spaceship rather than the opposing spaceship.

Based on the above, we decided to control the spaceship not with a traditional joystick, but by acting on the spaceship with quantum gates. We decided to have distinct locations where the spaceship, and therefore the targets, could be found, and to encode each location with qubit states. In this way we match the position of the spaceship to the state of a quantum register. The player chooses the gate they want and places it at a specific location. Placing the gate at the specific location will change the value of the register, thus affecting the position of the spaceship.

Finally, once the ship is in the correct position, the player must launch missiles to destroy the targets. That requires implementing a moving graphical element and defining the equations of motion in two-dimensional space.

6.3 Game-Based Learning Elements Selection

The last step of the conceptual design was to decide which GBL elements to include in our game.

First, we decided to use narratives to introduce the theory and rules of the game, specifically overlays that appear at the beginning of certain levels and give the student the information needed to complete the level.

We then decided to create an incentive system to increase the student' s motivation. Based on the games we found in the literature, we decided to introduce a scoring system that awarded 1 to 3 stars for successfully completing a level. To get all 3 stars, the player must complete each level with the least number of moves. We also thought it would be good to give the player new spaceships as they progressed through the game to increase motivation and engagement. We also decided not to use a leaderboard to avoid frustration or discrimination, as discussed in Chapter 3.

Finally, we decided to adopt the graceful failure approach to allow users to experiment and try alternative ways of solving the puzzle. So we decided that there would be no immediate penalty if players made a wrong move, i.e. if they moved the spaceship to the wrong position or if a shot missed. At the end of the level, they will only receive the maximum number of stars if they have used the minimum number of moves to solve the puzzle and if all their missiles hit the target. The opposite scenario will not result in a complete failure, they will simply not receive the maximum number of stars.

6.4 Educational Content Selection

Together with the definition of the mechanics and the choice of the GBL elements to be used in the game, it was decided which aspects of the basic theory of quantum computing - introduced in Chapter 4 - would be presented in the game. We decided to introduce the concepts of qubit, quantum register and quantum gate at a tutorial level, as these are the most basic concepts. Next, we decided to introduce Pauli gates (Pauli-X, Pauli-Y and Pauli-Z). We decided that the first levels should have registers with one qubit. After the player has become familiar with the operation of these gates, we will introduce the concept of superposition for one-bit registers. Later, when the player also understands superposition, we will use registers with two qubits. There, the player will have to combine their previous knowledge to solve the puzzles.

6.5 Evaluation Process Design

We also wondered how we could assess the learning outcomes of the game. After reviewing the available literature ((Cooksey and Jonsson 2022), (Stratton 2019)), we decided that a pre-post test study would be appropriate to evaluate the game. For the purposes of this study, a quiz was designed and integrated into the game. At this stage, we simply collected some candidate questions as suggested in the literature. The final selection of questions was made after the construction of the game had been completed and

before the quiz screen had been created. In addition, we decided to conduct a focus group to evaluate the structure, usability and difficulty of the game. The structure and content of the two phases (pre-post test and focus group) are presented in detail in chapter 8.

6.6 Development Process Definition

After determining the genre, mechanics and GBL elements that would be used in the game, selecting the educational content and designing the evaluation process, we started designing the development process. We decided that our first step would be to define the screens of the application and the navigation flow between them. The next step would be to design a wireframe for a prototype level. The implementation of the first two steps would allow us to create a working prototype, which would then be tested by typical users. Based on their feedback, we will improve the prototype and repeat this process until we have a satisfactory result. Then the alpha version of the game will be ready, and we will be able to conduct extensive user testing.

6.7 Application Structure

The diagram below shows the structure that was chosen for the application.

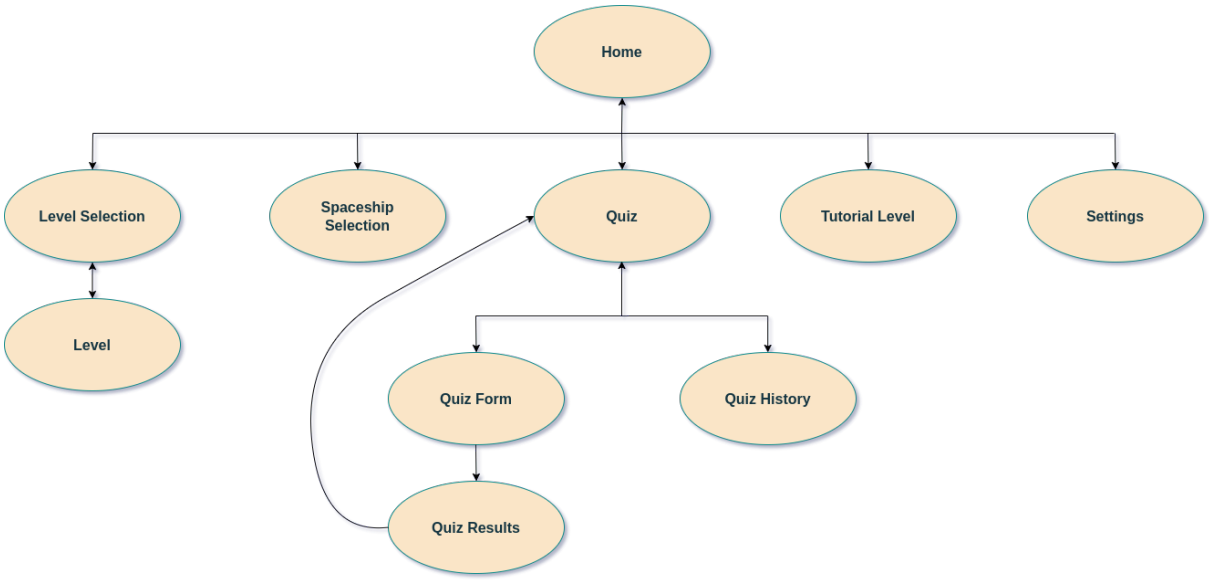


Figure 6.2. Application Structure & Navigation Flow.

The 'Home' screen will consist of 5 buttons which will navigate the user to the level selection screen, the ship selection screen, the quiz menu, the tutorial level or the settings. It will not be possible to

navigate between these screens without returning to the home menu, in order to keep navigation as simple as possible and avoid too much depth.

The 'Level Selection' screen will allow the user to select one of the available levels. Each level is preceded by a tile showing the level number, the portals available and the score the player has achieved in that level.

The 'Level' screen and the 'Tutorial Level' screen will be the only screens managed by the Flame game engine and will have Flame widgets. When the user leaves these screens, the game engine will shut down.

The 'Spaceship Selection' screen will allow the user to select which spaceship to operate. Spaceships are initially locked and rewarded to the User as described in Section 6.3.

The 'Quiz' screen will provide the ability to navigate to two sub-screens, the 'Quiz Form' screen, where the user can take the quiz we have created for the evaluation needs, and the 'Quiz History' screen, where the user can view previous quiz attempts. In this way, they can see for themselves if they have improved their knowledge. After completing a quiz, the application will proceed to the 'Quiz Results' screen, where the score for that particular attempt will be displayed.

Finally, the 'Settings' screen should allow the user to change the language and to reset the progress of the game.

The final layout and functionality of the screens will be presented in Chapter 7.

6.8 Level Wireframe

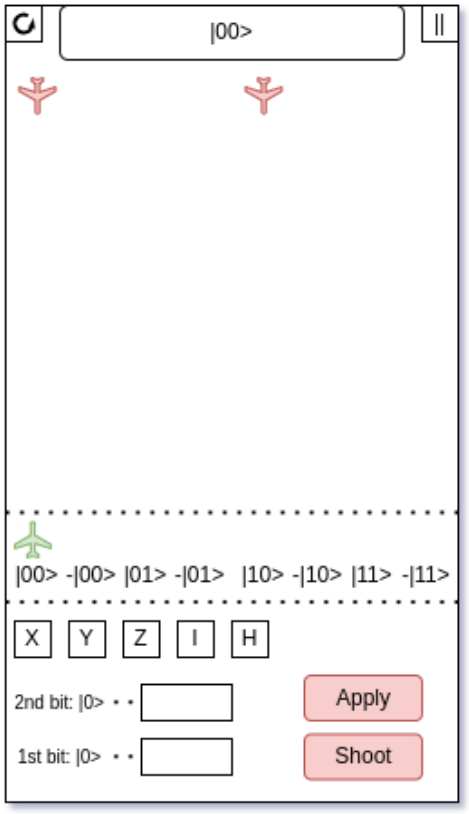


Figure 6.3. Wireframe

A prototype level (wireframe) was then designed in *diagrams.net*, which formed the foundation for creating the game' s graphical interface. As shown in the images above, the original approach did not include a graphical element for the quantum register, but rather 2 discrete qubits to which the quantum gate would be applied. There were also two buttons, one for applying the gate and one for launching the missiles. The target state was placed at the top of the screen. Finally, the targets were enemy spaceships, as in the original *Space Invaders* game.

6.9 Development Timeline

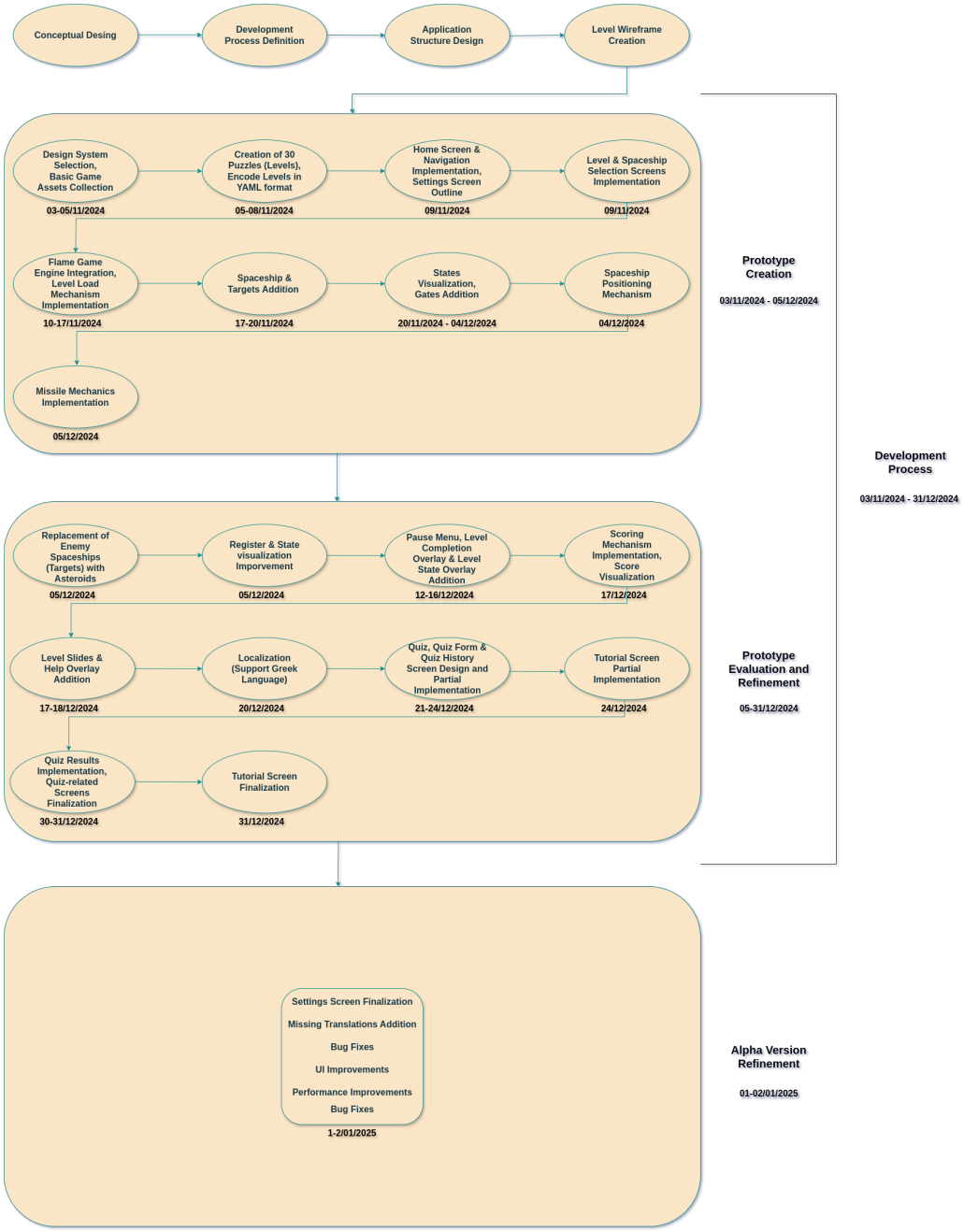


Figure 6.4. Development Timeline.

Development of the game began in November 2024. The game developed is called *Qubity* and, as mentioned before, is a variant of the arcade video game *Space Invaders*. Google' s *Material Design 3*

and *Flame Game Engine* were user to create the graphical elements of the application. The spaceships, asteroids, gates and several other bitmaps used to create the sprite components of the game are from *Freepik*. The background image was created by *ChatGPT* and was slightly modified.

The first step was the creation of 30 scenarios (puzzles) to form the levels of the game. The first 17 levels have a single qubit and the available gates are Pauli-X, Pauli-Y, Pauli-Z and Hadamard. The remaining 13 levels have a two qubits and the available gates are Pauli-X, Pauli-Z and Hadamard. In each level there are 2, 4, or 8 different states in which the spaceship can be in, depending on the number of qubits and the available gates, as described in the previous sections. Each scenario consists of the initial state of the qubits (initial position of the spaceship), the target state (positions of the target asteroids), the available gates and the minimum number of gates to be used to solve the puzzle. The scenarios are encoded in a YAML file in a manner that is human-readable while allowing for easy addition of new levels.

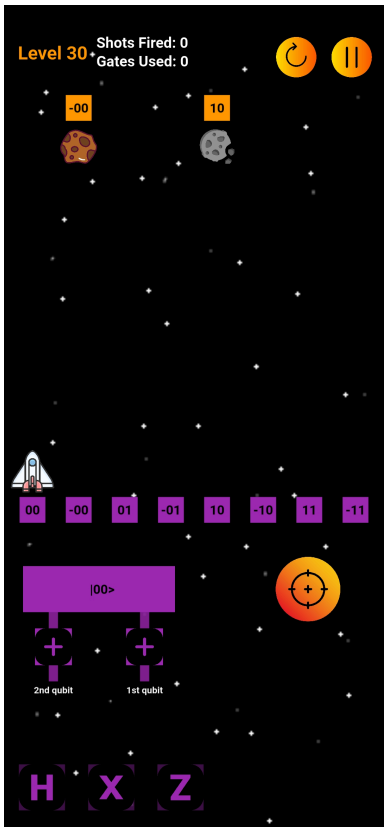


Figure 6.5. Screenshot from Level 30.

By the first week of December, the basic elements of the user interface were in place and the spaceship control mechanism was working. Then, user testing was conducted to refine the prototype, involving 5 users. Based on their feedback, the following corrections were made:

1. A graphical element was created to represent a quantum register with one or two outputs, replacing the discrete qubits. The outputs of the register represent the first or the second qubit.
2. The apply button has been removed. The gate action is applied to the register as soon as one of the register outputs is selected.
3. Targets were replaced with asteroids.
4. An overlay has been added to show the current level ID, gate used and missiles launched.
5. The target state display has been changed to make it easier of the player to understand where they need to place their spaceship.

Also, the quiz-related screens and the 'Tutorial' screen were partially implemented, theory slides were added at the beginning of some levels and help overlays were added on each level.

In the last week of December, Greek language support was added and the 'Tutorial' screen was finalized. Also, quiz questions were created, and the 'Quiz', 'Quiz Form', 'Quiz Results' and 'Quiz History' screens were finalized. The alpha version of our application was ready.

In the following days, several bugs were fixed, and UI issues were addressed after the application was tested by 2 regular users. In addition, the 'Settings' screen was finalized, and some missing translations were added. Finally, the project was restructured a bit, and some performance issues were fixed.

7 Game Screens Presentation

TODO: ADD INTRO

7.1 Home

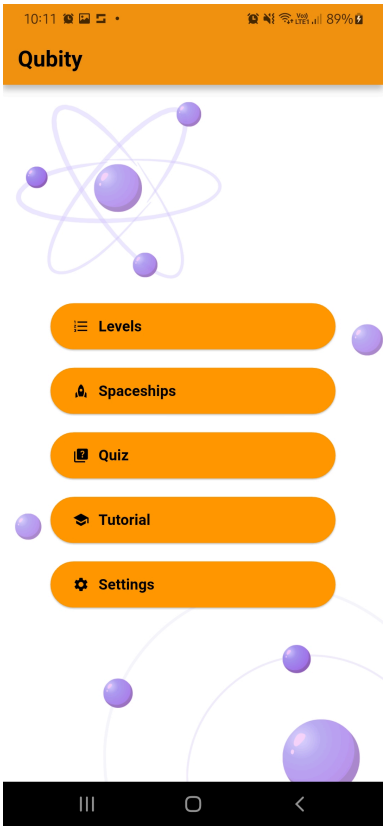


Figure 7.1. Home Screen.

From the app' s home screen, the user can access the game levels, available spaceships, the quiz, a tutorial level and game settings.

7.2 Tutorial Level

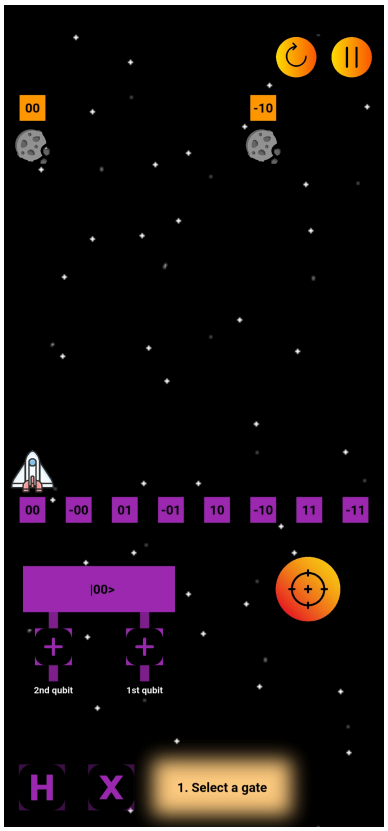


Figure 7.2. Tutorial Level Screen.

This screen loads a tutorial level, for which no score is calculated. First, a few slides are presented with basic quantum computing theory. In particular, the player is introduced to basic theory around qubits, quantum registers and quantum gates, as well as the concept of superposition. The slides consist of short sentences that summarize the theory discussed in chapter 4. Then there are 4 overlays with instructions, which demonstrate to the player how to use the gates to control the spaceship and how to destroy the asteroids.

7.3 Level Selection

Figure 7.3. Level Selection Screen.

On this screen the player can select any of the available levels. Each level' s card shows the number of qubits, the available gates and the score.

7.4 Level

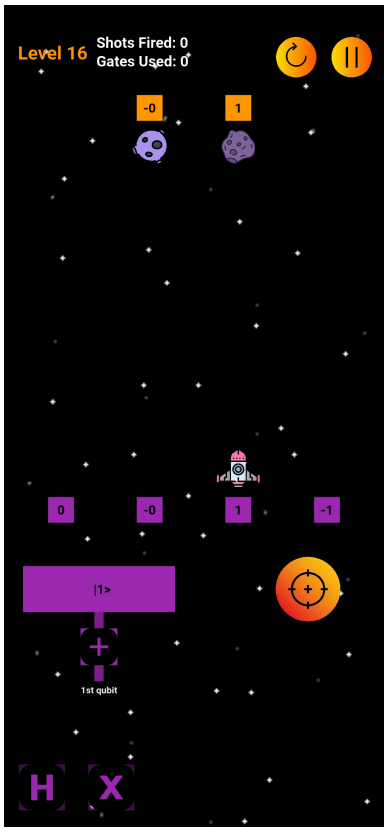


Figure 7.4. Level Screen.

In each level, there are one or more asteroid targets. The player shall move the spaceship under the targets by altering the state of the register. Then, they have to destroy the asteroid by launching missiles. Several levels have more than one solution, but to score maximum points they must use the minimum number of gates and as few missiles as possible. Each level has a different number of possible states, depending on the available gates and the number of qubits.

At the beginning of some levels, there are some slides with theory about the quantum gates available in this level. The first levels are very simple, requiring only 1 use of a gate to put the register into the target state. As the game progresses, the number of gates and combinations increases. In some levels there are gates that do not to be used. The player needs to use what they have learned to solve the puzzle using the minimum number of gates.

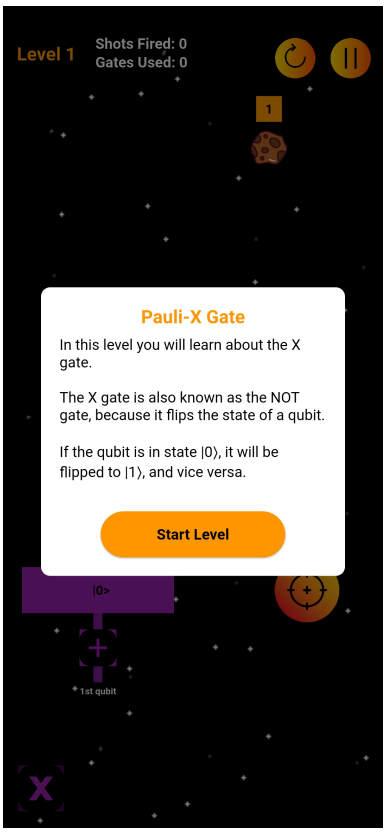


Figure 7.5. Theory Slide.

The first 9 levels introduce the Pauli Gates (X, Y and Z). In level 10, the theory of superposition is reintroduced to increase information retention. Also, Hadamard Gate (H) is introduced. In the next 7 levels, the player is asked to solve the puzzle by combining his previous knowledge of X and H gates.

From level 18, the number of qubits is increased to 2. From this level, no new theory is introduced; the player is just asked to combine their previous knowledge to solve the puzzles. In addition, the level of difficulty increases as the player has to decide which qubit to apply the gate to.

7.5 Spaceship Selection

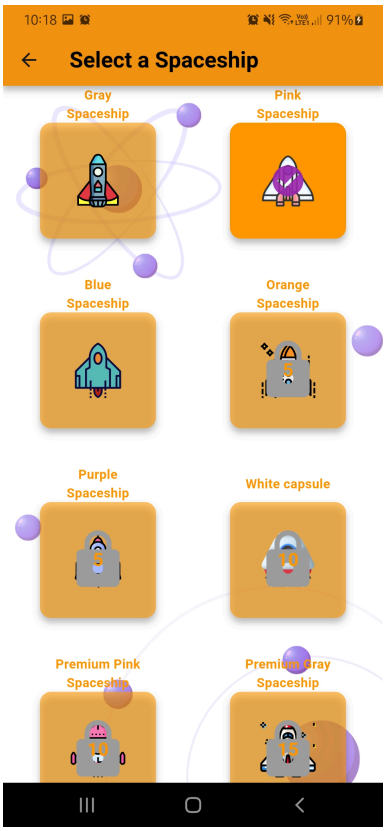


Figure 7.6. Spaceships Seection Screen.

From this screen the player can select one of the available spaceships. The spaceships are decorative in nature, as they all have the same capabilities, but are a reward for the player' s progress in the game. Initially only 3 are available, new spaceships are unlocked after completing more levels.

7.6 Quiz

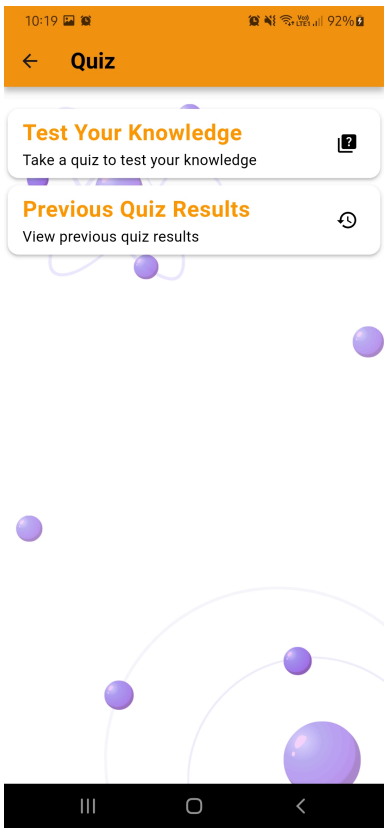


Figure 7.7. Quiz Screen.

The game is accompanied by a 12-question quiz, used for the evaluation process, as described in the previous chapter. This screen gives access to the quiz. The player can take a quiz or view their score from previous attempts.

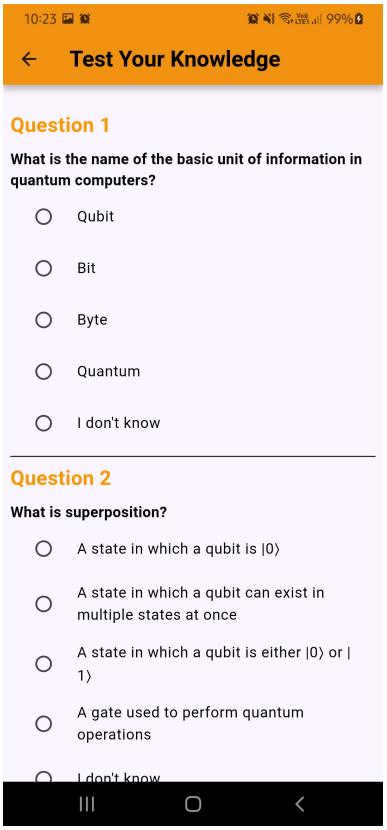


Figure 7.8. Quiz Form Screen.

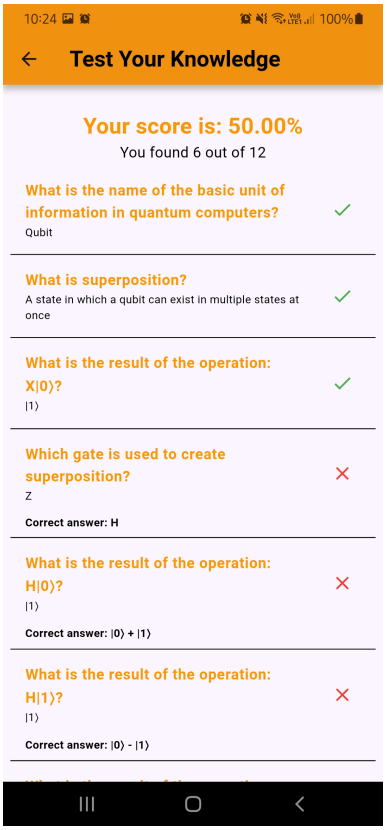


Figure 7.9. Quiz Results Screen.

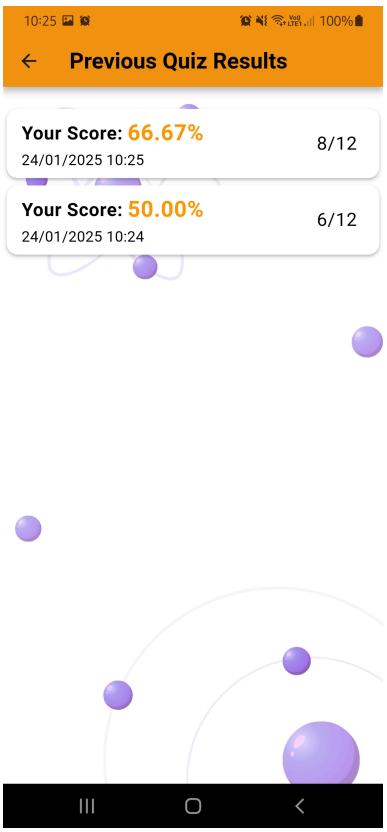


Figure 7.10. Quiz History Screen.

7.7 Settings

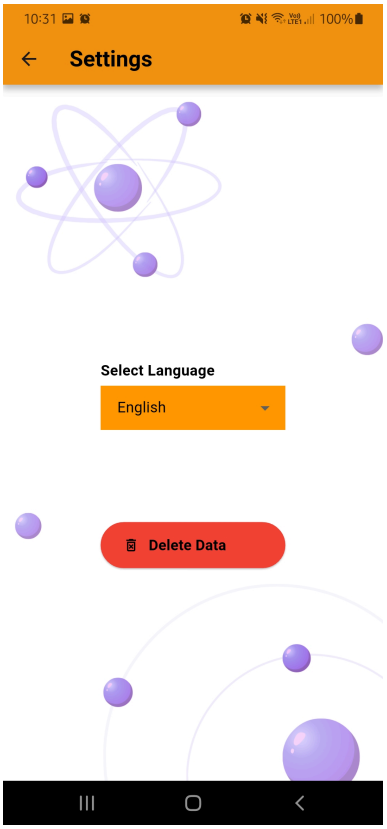


Figure 7.11. Settings Screen.

From this screen, the user can choose between the available languages and delete their progress in the game. Available languages are English and Greek, with the former being the default.

8 Evaluation

In order to measure the learning outcome of the game and to evaluate the user experience, the evaluation process described in the diagram above was designed. More specifically, this process was designed to evaluate the achievement of the objectives set out in Chapter 1, namely:

1. The game to be an effective learning tool and to successfully introduce the basic concepts of quantum computing to university students.

2. The game should be enjoyable and simple so as not to distract the student from the learning objective.

3. The game should be short and require no additional equipment other than a mobile phone, making it suitable to be played during a university lecture or when the student has limited time.

8.1 Evaluation Process Description

Figure 8.1. Evaluation Flow

10 university students volunteered to participate in the evaluation of the game, which was carried out in 2 phases. The volunteers claimed to have a strong background in mathematics, with 4 of them having knowledge of complex numbers. The first phase used the pre-post test methodology, for which a quiz was designed and incorporated into the game. The second phase was a focus group with some participants from the first phase.

To support the first phase and the implementation of the quiz, we relied on *Cooksey and Jonsson* (2022) and *Stratton* (2019). The first step was to create a repository of candidate questions to help structure the quiz. These questions were mainly taken from comprehension quizzes given in the *Quantum Computation* course at the University of Patras (academic year 2023-24, lecturers: Sgarbas, K. and Kounavis, P.). A subset of these questions was then selected and adapted to the content of the game. The quiz consists of 3 theoretical questions and 9 questions regarding quantum gates actions. Each correct answer awards one point, while each incorrect answer gives zero points. The theoretical questions relate to the theory presented to the player via slides. The quantum gate questions ask the

player to calculate the new state of a qubit after the gate has been applied to it, or ask the player to select the appropriate gate to perform a quantum calculation. The questions are designed to prove that the player has learned to use -some of- the gates introduced in chapter 4 and understands the concept of superposition.

As suggested in the literature, participants were asked to complete the pre-test before interacting with the game to get an overview of their previous knowledge. Then, they were asked to complete the tutorial level and then complete the 30 levels of the game and retake the quiz. Players were given as much time as they wanted and were asked to record how long it took to complete the game. There was no further guidance on how the game works and no external help was provided to solve the levels. The *Normalized Gain* metric was used to measure efficiency, as suggested in the literature (Cooksey and Jonsson 2022), (McKagan, Sayre, and Madsen 2022). In addition, we tried to determine whether the knowledge of complex numbers and the time it took each player to complete the game affected the learning outcome.

In the second phase, 8 out of 10 players from the fist phase participated in the focus group. The questions they were asked to discuss were related to 4 areas; *User Interface & User Experience* (UI & UX), *Game Mechanics*, *Difficulty* and *Scoring & Rewards*. They were also free to discuss with each other and comment on their overall experience of the application. The findings from the focus group will be used to improve the game in the future.

8.2 Phase 1: Statistical Analysis of Quiz Results

In this section, we analyze the quiz results using statistical methods to evaluate the effectiveness of the game as a learning tool. Key metrics used in this analysis include the p-value, Pearson correlation coefficient, and Average Normalized Gain.

- **P-value**: The null hypothesis in this context is the assumption that there is no significant difference between the initial and final quiz scores of the participants. It suggests that any observed difference in scores is due to random chance rather than the effectiveness of the game as a learning tool. The p-value is used to test this hypothesis, with a lower p-value indicating stronger evidence against the null hypothesis.
- **Pearson Correlation Coefficient (Pearson r)**: A measure of the linear correlation between two variables, ranging from -1 to 1. A value closer to 1 indicates a strong positive correlation, while a value closer to -1 indicates a strong negative correlation. A value close to 0 indicates that there is little to no linear correlation between the two variables.
- **Average Normalized Gain ($\langle g \rangle$)**: A metric used to measure the effectiveness of an educational process, calculated as the ratio of the actual average gain to the maximum possible average gain. A higher value indicates greater effectiveness.

The normalized gain for each individual player is calculated as follows:

$$g = \frac{post\% - pre\%}{100\% - pre\%}$$

The Average Normalized Gain is defined as:

$$\langle g \rangle = \frac{\langle post\% \rangle - \langle pre\% \rangle}{100\% - \langle pre\% \rangle}$$

where the brackets indicate the averages. (Cooksey and Jonsson 2022), (McKagan, Sayre, and Madsen 2022)

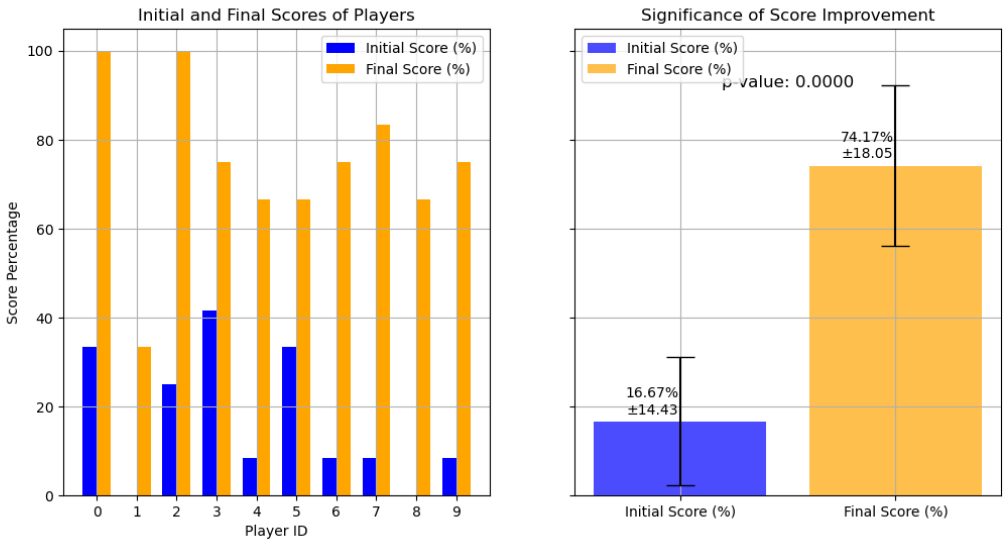


Figure 8.2. Significance of Score Improvement

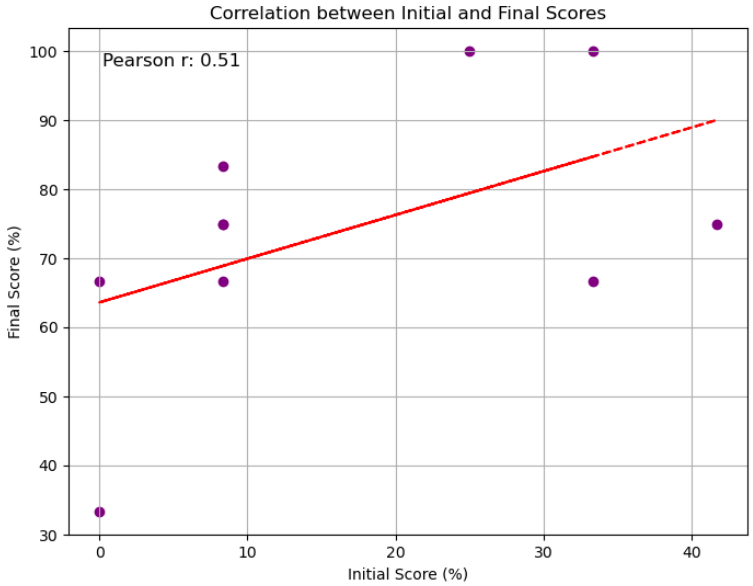


Figure 8.3. Correlation between Initial and Final Scores

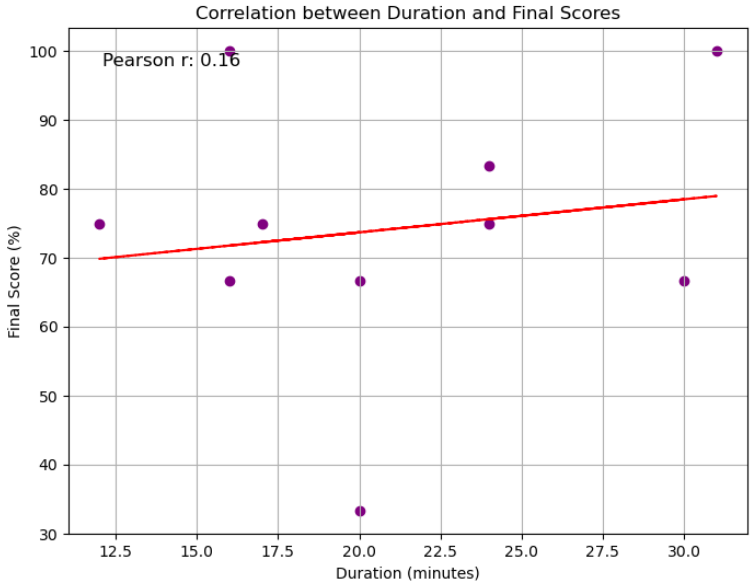


Figure 8.4. Correlation between Duration and Final Scores

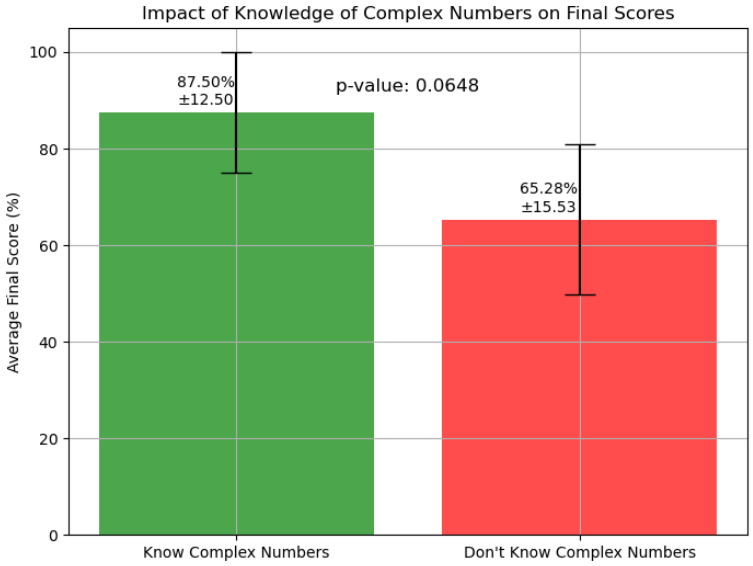


Figure 8.5. Impact of Knowledge of Complex Numbers on Final Scores

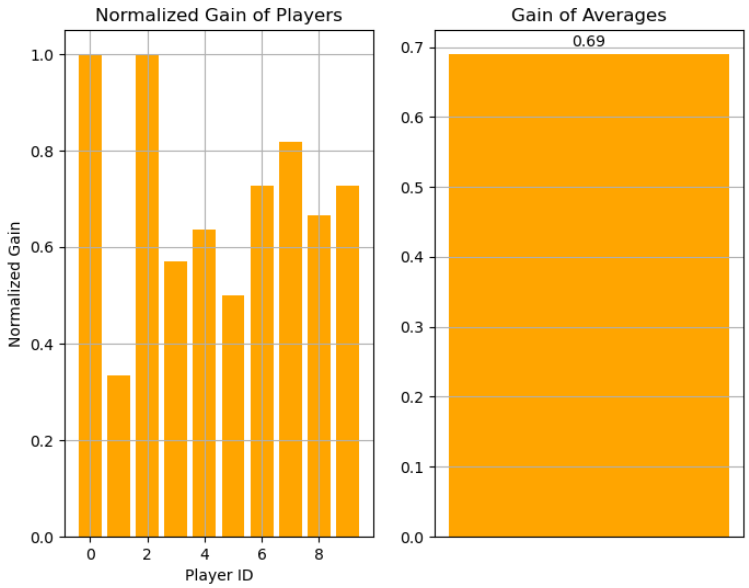


Figure 8.6. Gain of Averages

1. **Significance of Score Improvement (p-value: 0.0):** The p-value of 0.0 indicates that the difference between the initial and final scores is extremely statistically significant. This means that the

observed difference in scores is highly unlikely to have occurred by chance, and there is strong evidence to suggest that the intervention or factor being tested had a significant impact on the scores.

2. **Correlation between Initial and Final Scores (Pearson r: 0.51)**: The Pearson correlation coefficient of 0.51 indicates a moderate positive correlation between the initial and final scores. This suggests that players who had higher initial scores also tended to have higher final scores, but the relationship is not very strong. There are other factors that might be influencing the final scores.
3. **Correlation between Duration and Final Scores (Pearson r: 0.16)**: The Pearson correlation coefficient of 0.16 indicates a weak positive correlation between the duration and final scores. This suggests that the amount of time spent on the task has a minimal impact on the final scores. Other factors are likely more influential in determining the final scores.
4. **Impact of Knowledge of Complex Numbers on Final Scores (p-value: 0.0648)**: The p-value of 0.0648 is slightly above the common significance threshold of 0.05. This means that there is not enough evidence to conclude that there is a statistically significant difference in the average final scores between those who know complex numbers and those who don't at the 5% significance level.
5. **Average Normalized Gain ($\langle g \rangle$) of 0.69**: The average normalized gain ($\langle g \rangle$) of 0.69 indicates a substantial improvement in the players' knowledge after playing the game. According to the literature, an average gain above 0.25 is considered effective (Cooksey and Jonsson 2022), making this result particularly significant. This demonstrates the game's effectiveness as an educational tool and highlights its potential to significantly enhance learning outcomes.

8.3 Phase 2: Focus Group

In this section we will present the focus group findings by area.

8.3.1 User Interface & User Experience

- One player requested an extra slide in the tutorial or in the first level, explaining how each level's score is calculated.
- One player requested that the *Next Level* button should be a different color to others (on the level completion overlay).
- One player requested to group levels in the level selection menu by difficulty.
- Players agreed that the game menu was easy to use and that the navigation worked well and was predictable.

- Players agreed that the state of the application was flawless and was updating correctly.
- Some players requested to add background music and sounds when launching missiles and hitting the targets.

8.3.2 Game Mechanics

- Players found the mechanics simple and casual and commented that this simplicity helped them focus on solving the puzzles.

8.3.3 Difficulty

- Players agreed that the game is easy and intended of users with no quantum computer experience. All of them could complete all 30 levels. They said they would like to try more difficult puzzles.
- Most players said that the hardest level was level 30 (the final one), as it required a combination of 3 gates to solve the puzzle.
- One player did not understand how the Hadamard gate acts on 2 qubits. Half of the players said they would like more information about superposition, but agreed that the theory presented was sufficient to solve the puzzles.
- One player suggested that there should be hints to use in case you find it difficult to solve the puzzle.

8.3.4 Scoring & Rewards

- Players agreed that having a score system encouraged them to improve. 7 out of 8 players tried to complete all levels with 3 stars.
- Players agreed that they felt rewarded and motivated to keep playing by unlocking new spaceships as they progressed.

8.4 Final Conclusions

Based on the feedback and data collected from the initial evaluation and focus group, several conclusions can be drawn:

1. The difference between the initial and final scores is extremely statistically, as indicated by the p-value of 0.0.

2. There is a moderate positive correlation between initial and final scores, suggesting that initial performance is somewhat predictive of final performance.
3. The duration of the task has a weak correlation with final scores, indicating that time spent is not a strong predictor of performance.
4. Knowledge of complex numbers does not show a statistically significant difference in final scores at the 5% significance level, but the result is close to being significant and might be considered at a higher threshold.
5. The average time needed to complete the game was 21 minutes. This makes the game suitable to be played in the context of a college lecture to increase student engagement in the classroom or to allow students to test their knowledge in a fun and interactive way.
6. The average normalized gain ($\langle g \rangle$) of 0.69 demonstrates the game's effectiveness as an educational tool.
7. The number of participants is small, only 10 people, so a complementary evaluation phase with a larger number of participants is suggested, for more reliable conclusions. However, the result of the initial evaluation is promising.
8. The duration of the game makes it suitable to be used as a teaching tool in the context of a lecture.
9. The theory around superposition should be extended.
10. The game could be extended with more levels and quantum gates.

9 Bibliography

Amazon Web Services, Inc. 2024a. "Mobile Application Development." Amazon Web Services, Inc. https://aws.amazon.com/mobile/mobile-application-development/.

——. 2024b. "What Is Quantum Computing?" Amazon Web Services, Inc. https://aws.amazon.com/what-is/quantum-computing/.

Bat, Natesh. 2024. "Flutter Vs React Native : Performance Benchmarks You Can't Miss!" Medium. %22https://nateshmbhat.medium.com/flutter-vs-react-native-performance-benchmarks-you-cant-miss-%EF%B8%8F-2e31905df9b4%22.

Battistella, P. E., and C. G. von Wangenheim. 2016. "Games for Teaching Computing in Higher Education - a Systematic Review." *IEEE Technology and Engineering Education*. https://www.researchgate.net/publication/325046233_Games_for_Teaching_Computing_in_Higher_Education_-_A_Systematic_Review.

Berkling, Kay, and Christoph Thomas. 2013. "Gamification of a Software Engineering Course and a Detailed Analysis of the Factors That Lead to It's Failure." In *2013 International Conference on Interactive Collaborative Learning (ICL)*, 525–30. https://scholar.google.com/scholar_lookup?hl=en&publication_year=2013&author=K.+Berkling&author=C.+Thomas&title=Gamification+of+a+Software+Engineering+Course+and+a+Detailed+Analysis+of+the+Factors+that+Lead+to+its+Failure#d=gs_cit&t=1722676216480&u=%2Fscholar%3Fq%3Dinfo%3AmDp34HDvUywJ%3Ascholar.google.com%2F%26output%3Dcite%26scirp%3D0%26hl%3Den.

Cooksey, K. L., and P. Jonsson. 2022. "Using Pre-/Post-Quizzes Intentionally in Curriculum Development and Evaluation." Institution for Scientist & Engineer Educators. https://arxiv.org/pdf/2210.01823.

Domínguez, A., J. Saenz-De-Navarrete, L. De-Marcos, L. Fernández-Sanz, C. Pagés, and J. Martínez-Herráiz. 2013. "Gamifying Learning Experiences: Practical Implications and Outcomes." *Computer & Education*. Vol. 63. https://scholar.google.com/scholar_lookup?hl=en&volume=63&publication_year=2013&pages=380-92&journal=Computers+%26+Education&author=A.+Dom%C3%ADnguez&author=J.+Saenz-De-Navarrete&author=L.+De-Marcos&author=L.+Fern%C3%A1ndez-Sanz&author=C.+Pag%C3%A9s&author=J.+Mart%C3%ADnez-Herr%C3%A1iz&title=Gamifying+Learning+Experiences%3A+Practical+Implications+and+Outcomes.

Fakokunde, J. B. 2021. "Effects of Gender and Cognitive Style on Students' Learning Retention in Social Studies Using Computer-Based Instructional Puzzle." *Mediterranean Journal of Education.* https://pasithee.library.upatras.gr/mje/article/view/3580/3650.

Fehervari, Zoltan. 2024. "Another Guide on Performance: Java Vs Kotlin." Medium. https://medium.com/@fhrvri.mmxiv/another-guide-on-performance-java-vs-kotlin-40117fa93560.

GeeksforGeeks contributor. 2024. "Kotlin Vs Java - Which One Should i Choose for Android Development." https://www.geeksforgeeks.org/kotlin-vs-java/.

Harding, Eve. 2023. "The Pros and Cons of Game-Based Learning." Bedrock Learning. https://bedrocklearning.org/literacy-blogs/the-pros-and-cons-of-game-based-learning/.

IBM Corporation. 2024. "What Is Mobile Application Development?" IBM Corporation. https://www.ibm.com/topics/mobile-application-development.

Idika, M. I., and M. P. Oluwaseyi. 2024. "The Impact of Puzzle Game and Video-Based Puzzle Strategies on Students' Achievement and Retention in Periodicity." *Journal of Mathematics and Science Teacher.* https://www.mathsciteacher.com/download/the-impact-of-puzzle-game-and-video-based-puzzle-strategies-on-students-achievement-and-retention-in-14366.pdf.

JetBrains s.r.o. 2024. "The Six Most Popular Cross-Platform App Development Frameworks." JetBrains s.r.o. https://www.jetbrains.com/help/kotlin-multiplatform-dev/cross-platform-frameworks.html.

Karafillidis, Ioannis. 2015. "Quantum Computing." Kallipos - Open Academic Editions. %22https://repository.kallipos.gr/handle/11419/216?&locale=en%22.

Kaur, Baljit. 2023. "Difference Between Objective c and Swift." Medium. https://medium.com/swiftfy/difference-between-objective-c-and-swift-e53369ee2d4f.

Kohout, Jiri. 2016. "What Is a Mobile Application Containerization, or Wrapper, and Why Must It Die?" Teska Labs. https://teskalabs.com/blog/mobile-application-containerization-wrapping.

Ledda, Rosalie. 2012. "7 Tips for a Game-Based Learning Success." eLearning Industry. https://elearningindustry.com/7-tips-game-based-learning.

Marshall, Gunnell. 2024. "Cross-Platform." Techopedia. https://www.techopedia.com/definition/17056/cross-platform.

McKagan, S., E. Sayre, and A. Madsen. 2022. "Normalized Gain: What Is It and When and How Should i Use It?" PhysPort. https://www.physport.org/recommendations/Entry.cfm?ID=93334.

Medium contributor. 2023. "Kotlin Vs. Java: A Comparison of Features and Performance." Medium. https://medium.com/@midoripig1009/kotlin-vs-java-a-comparison-of-features-and-performance-fe9eaac8b2c2.

——. 2024. “Top 10 Cross-Platform App Development Frameworks for 2024.” Medium. https://medium.com/@evincedevelop/top-10-cross-platform-app-development-frameworks-for-2024-1d812fdfc776.

Microsoft Corporation. 2024a. “Explore Quantum - Multi-Qubit Gates.” Microsoft Corporation. https://quantum.microsoft.com/en-us/insights/education/concepts/multi-qubit-gates.

——. 2024b. “Explore Quantum - Single-Qubit Gates.” Microsoft Corporation. https://quantum.microsoft.com/en-us/insights/education/concepts/single-qubit-gates.

Moore-Russo, Deborah, Andrew Wiss, and Jeremiah Grabowski. 2018. “Integration of Gamification into Course Design: A Noble Endeavor with Potential Pitfalls.” *College Teaching*. Vol. 66. http://www.tandfmisc.com/doi/abs/10.1080/87567555.2017.1295016.

Nagappan, Atic. 2023. “Pros and Cons of Quantum Computing.” https://www.linkedin.com/pulse/pros-cons-quantum-computing-athik-nagappan-1nhef/.

Nicholson, Scott. 2012. “A User-Centered Theoretical Framework for Meaningful Gamification, Paper Presented at the Games+ Learning+ Society 8.0.” *8.0, Madison, USA*. https://scholar.google.com/scholar_lookup?hl=en&volume=8&publication_year=2012&pages=223-30&journal=Games%2B+Learning%2B+Society&issue=1&author=S.+Nicholson&title=A+User-Centered+Theoretical+Framework+for+Meaningful+Gamification.

Piispanen, Laura. 2024. “List of Quantum Games.” Aalto University. https://kiedos.art/quantum-games-list/.

Piispanen, Laura, Edward Morrell, Solip Park, Marcell Pfaffhauser, and Kultima Annakaisa. 2023. “The History of Quantum Games.” Aalto University.

Piispanen, Laura, Marcel Pfaffjauser, James Wootton, Julian Togelius, and Annakaisa Kultima. 2024. “Defining Quantum Games.” Aalto University. https://arxiv.org/abs/2206.00089.

Plass, J. L., B. D. Homer, and C. K. Kinzer. 2015. “Foundations of Game-Based Learning.” *Educational Psychologist*. Vol. 50. https://files.eric.ed.gov/fulltext/EJ1090277.pdf.

Popko, Aleksander. 2024. “Objective-c Vs Swift: iOS Comparison.” Netguru. https://www.netguru.com/blog/objective-c-vs-swift.

Rouse, Margaret. 2024. “Native Mobile App.” Techopedia. https://www.techopedia.com/definition/27568/native-mobile-app.

Seskir, Z. C., P. Migdal, C. Weidner, A. Anuopam, N. Case, N. Davis, C. Decaroli, et al. 2022. “Quantum Games and Interactive Tools for Quantum Technologies Outreach and Education.” *Optical Engineering*. Vol. 61. SPIE. https://doi.org/10.1117/1.OE.61.8.081809%7D.

Shah, Disha. 2024. “What Should You Choose from Flutter Vs. React Native in 2024?” Radixweb. https://radixweb.com/blog/flutter-vs-react-native.

Shi, Y. R., and J. L. Shih. 2015. "Game Factors and Game-Based Learning Design Model." *International Journal of Computer Games Technology*. Vol. 2015. https://onlinelibrary.wiley.com/doi/full/10.1155/2015/549684.

Srijampana, V. V. G. R., S. Chebrolu, and R. Potti. 2023. "Role of ' Crossword Puzzles' in Retention of Knowledge and Learning Outcomes Among Medical Students: A Meta-Analysis." *Journal of Dr. YSR University of Health Sciences*. https://journals.lww.com/jdyu/fulltext/2023/12040/role_of__crossword_puzzles__in_retention_of.8.aspx.

Stratton, S. J. 2019. "Quasi-Experimental Design (Pre-Test and Post-Test Studies) in Prehospital and Disaster Research." Cambridge University Press. https://doi.org/10.1017/S1049023X19005053.

Taylor, Eliza. 2024. "Advantages and Disadvantages of Quantum Computing." The Knowledge Academy. https://www.theknowledgeacademy.com/blog/advantages-and-disadvantages-of-quantum-computing/.

Techtarget contributor. 2023. "Hybrid Application." Techtarget. https://www.techtarget.com/searchsoftwarequality/definition/hybrid-application-hybrid-app.

Vailshery, Lionel Sujay. 2024. "Cross-Platform Mobile Frameworks Used by Software Developers Worldwide from 2019 to 2023." Statista. https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/.

Wikipedia contributors. 2024a. "Java (Programming Language)." Wikipedia. https://en.wikipedia.org/wiki/Java_(programming_language).

———. 2024b. "Kotlin (Programming Language)." Wikipedia. https://en.wikipedia.org/wiki/Kotlin_(programming_language).

———. 2024c. "Objective-c." Wikipedia. https://en.wikipedia.org/wiki/Objective-C.

———. 2024d. "Swift (Programming Language)." Wikipedia. https://en.wikipedia.org/wiki/Swift_(programming_language).

———. 2024e. "Flutter (Software)." Wikipedia. https://en.wikipedia.org/wiki/Flutter_(software).

———. 2024f. "Quantum Computing." Wikipedia. https://en.wikipedia.org/wiki/Quantum_computing.

———. 2024g. "Quantum Gate." Wikipedia. https://en.wikipedia.org/wiki/List_of_quantum_logic_gates.

———. 2024h. "React Native." Wikipedia. https://en.wikipedia.org/wiki/React_Native.

Wirtz, Bryan. 2023. "Why Game-Based Learning: Pros, Cons and How It Helps Students Retain Basic Knowledge." Game Designing. https://www.gamedesigning.org/learn/game-based-learning/.

Wootton, James. 2017. "Why We Need to Make Quantum Games." Decodoku. https://decodoku.medium.com/why-we-need-to-make-quantum-games-6f8c7bc4ace7.