

*Faculty of Computers and Artificial intelligence,*

*Beni-Suef University*

# **Edutropolis Project**

A Self Education Game for the Treatment of Learning Disorders for Children Diagnosed with ADHD

**Supervisor**

Dr. Hossam Moftah

**Project Team**

George Zaky 180045

Khaled Ali 180053

Beshoy William 180041

Fatma Nagim Eldeen 180096

Amira Adel 180033

Omayma Hamada 180035

Rawan Mohsen 180061

# **Acknowledgment**

We would like to express our special thanks of gratitude to our academic project supervisor Dr. Hossam Moftah who gave us this great opportunity to do this wonderful project on the topic (Treatment of Learning Difficulties for Children Diagnosed with ADHD), and without his assistance throughout the whole working process, this project would have never been accomplished with these results. We want to thank you very much for your support and understanding over these past months during the project working timeline.

**Contents**

**Chapter 1 : Introduction**

1.1 Introduction ............................................................................................ 10

1.2 Problem Statement ................................................................................. 10

1.3 Objectives .............................................................................................. 10

1.4 Impact on Business ................................................................................ 10

1.5 Ways of Marketing ................................................................................. 10

1.6 Conclusion ............................................................................................. 10

**Chapter 2 : Related Works**

2.1 Introduction ............................................................................................ 10

2.2 Related Works ........................................................................................ 10

2.3 System Architecture ............................................................................... 10

2.4 Features ................................................................................................... 10

2.5 Benefits ................................................................................................... 10

2.6 Project Risk Management ....................................................................... 10

**Chapter 3 : Project Development Methodology**

3.1 Software Development Life cycle ......................................................... 10

3.2 Chosen Methodology ............................................................................ 10

3.3 Applications of Chosen Methodology .................................................. 10

**Chapter 4 : Game Design**

4.1 Introduction ............................................................................................ 10

4.2 User Interface (UI) ................................................................................. 10

4.3 User Experience (UX) ............................................................................ 10

**Chapter 5 : Software Analysis and Design**

5.1 Project Software Requirement Specification ........................................ 10

5.2 Project Components Design .................................................................. 10

5.3 Entity Relationship Diagram (Class Diagram) ...................................... 10

5.4 Use Case Diagram .................................................................................. 10

5.5 Data Flow Diagram ................................................................................ 10

5.6 Flow Chart .............................................................................................. 10

**Chapter 6 : Project Software Tools and Technologies**

6.1 Unity Game Engine ................................................................................ 10

6.2 Microsoft Visual Studio & C# ............................................................... 10

6.3 Blender Software .................................................................................... 10

**Chapter 7 : Testing**

7.1 Testing User Interface ............................................................................ 10

7.2 Unit Testing ............................................................................................ 10

7.3 Integration Test ....................................................................................... 10

7.4 System Test User Acceptance ................................................................. 10

7.5 Reference ................................................................................................ 10

**Abstract**

Attention-deficit hyperactivity disorder (ADHD) is described as the most common neurobehavioral condition of childhood. We raise the concern that ADHD is not a disease per se but rather a group of symptoms representing a final common behavioral pathway for a gamut of emotional, psychological, and/or learning problems. “Core” ADHD symptoms of inattentiveness, hyperactivity and impulsivity are not unique to ADHD. Rates of “comorbid” psychiatric and learning problems, including depression and anxiety, range from 12 to 60%, with significant symptom overlap with ADHD. In this project, we have worked on two main parts, the first part, which is to help children with ADHD to overcome learning difficulties, improve the level of concentration, reduce their level of distraction and hyperactivity, by exploiting their love for play and movement, and also by exploiting technological progress. Smartphones and Internet have been growing rapidly along with the modernization

of technology. It has impacted almost all walks of human life in the prominent areas, such as business, education, health and social life. Despite the negative impacts, the use of smartphones should emphasize the positive impacts so that the children can take advantage. Education through a game is a premium project for the education industry. Therefore, here we designed a game application that supports fun learning for children especially those who are diagnosed with ADHD. Gaming, learning, and motivation are three dimensions that are based on our educational game. Learning in interesting ways can be difficult, but many researchers believe that presentations, videos and fun elements can keep the children who are diagnosed with ADHD learn through playing games. In the second part of the project, we focused on something very important, which is to educate the parents of the child with ADHD about their child's condition and symptoms and also about how to deal with their child in the right way and practice positive parenting in order to help their child improve whether Educationally, psychologically or socially, knowing his condition, accepting it and adapting to it in the right way, and also helping him to accept it and not feel anxious, depressed or stressed because of his or her difference. We called our project as Edutropolis Project which stands for Education town. It was designed by using many and different tools and through the following scrum approach in Agile methodology.

**Chapter 1: Introduction**

**1.1 Introduction**

**1.1.1 What is ADHD?**

ADHD is one of the most common neurodevelopmental disorders of childhood. It is usually first diagnosed in childhood and often lasts into adulthood. Children with ADHD may have trouble paying attention, controlling impulsive behaviors (may act without thinking about what the result will be), or be overly active.

Attention-deficit/hyperactivity disorder (ADHD) is among the most common neurobehavioral disorders presenting for treatment in children,It carries a high rate of comorbid psychiatric problems such as oppositional defiant disorder (ODD), conduct disorder, mood and anxiety disorders, and cigarette and substance use disorders. Across the lifespan, the social and societal costs of untreated ADHD are considerable, including academic and occupational underachievement, delinquency, motor vehicle safety, and difficulties with personal relationships.

ADHD affects an estimated 4% to 12% of school-aged children worldwide with survey and epidemiologically derived data showing that 4 to 5% of college aged students and adults have ADHD . In more recent years, the recognition and diagnosis of ADHD in adults have been increasing although treatment of adults with ADHD continues to lag substantially behind that of children. In contrast to a disproportionate rate of boys diagnosed with ADHD relative to girls in childhood, in adults, an equal number of men and women with ADHD are presenting for diagnosis and treatment.

**1.1.2 Causes**

* Blood relatives, such as a parent or sibling, with ADHD or another mental health disorder
* Exposure to environmental toxins — such as lead, found mainly in paint and pipes in older buildings
* Maternal drug use, alcohol use or smoking during pregnancy
* Premature birth

**1.1.3 Symptoms**

It is normal for children to have trouble focusing and behaving at one time or another. However, children with ADHD do not just grow out of these behaviors. The symptoms continue, can be severe, and can cause difficulty at school, at home, or with friends.

**Common Symptoms of ADHD in Children :** A Person with ADHD will exhibit six of these symptoms. Although symptoms may differ depending on the type of ADHD diagnosed, some common symptoms of ADHD in kids are:

1. Self-focused behavior
2. Trouble waiting turn
3. Emotional turmoil
4. Fidgeting
5. Problems playing quietly
6. Trouble finishing tasks
7. Lack of focus
8. Forgetfulness
9. Have difficulty getting along with others

## **1.1.5 How to Improve Concentration in Children with ADHD**

### Get the medication right.

### Establish eye contact.

### Practice skills step by step

### Play attention-boosting games

### Fit the task to the child

1. Make learning active.

### Make a suitable work space.

### Take frequent breaks.

### Use self-monitoring and positive self-talk

**1.2 Problem Statement**

There are many children who are unable to absorb and benefit from educational abilities because they have a learning disability due to ADHD.

Attention deficit hyperactivity disorder (ADHD) has a significant impact on a child's absorptive capacity, which can thus weaken the child's part of the work effectively, both socially and academically.

**1.2.1. How does awareness of hyperactivity and attention deficit disorder (ADHD) affect the future of a child with this condition?**

Knowledge and awareness of hyperactivity and attention deficit disorder (ADHD) is very important because it helps in the early diagnosis of the condition and thus determines the appropriate treatment method.

The step of choosing the appropriate treatment for a child with attention deficit and hyperactivity(ADHD) is very sensitive and important; The proper follow-up and management of the situation helps develop the child's compensatory skills, which the child effectively uses to deal with areas of disability.

But one of the problems with the diagnosis of ADHD is that it cannot be diagnosed and known by a laboratory examination, and there are no precise criteria for confirming the diagnosis of the child with ADHD.

**1.2.2 Criteria for Attention Deficit Hyperactivity Disorder (ADHD) :**

At present, the diagnosis of ADHD is based on criteria from the Diagnostic and Statistical Manual of Mental Disorders (DSM-IV). The Diagnostic and Statistical Manual of Mental Disorders (DSM-IV) classifies ADHD into two types: the inattentive type and the hyperactive-impulsive type. Each type has nine behavioral characteristics that indicate them. To diagnose a child as having ADHD, he/she must have at least six characteristics of inattentive type, hyperactivity-impulse, or mixture of the two types.

So from what we talked about earlier we conclude that there are three types of ADHD and they are :

**1. Hyperactive/Impulsive ADHD:**

It’s the classic ADHD in people's minds that it affects the patient in a way which made him/her overly energetic,

for example as the disruptive -9 year old- child who loves to jump off dangerously high things and seems to be driven by a motor.

A person with this Hyperactive/Impulsive ADHD will exhibit six of these nine symptoms:

1. Often fidgets with or taps hands or feet or squirms in their seat.
2. Often leaves a seat in situations when remaining seated is expected for example leaving his or her place in the classroom.
3. Often runs about or climbs in situations where it is inappropriate.
4. Often unable to play or engage in leisure activities quietly.
5. Is often “on the go,” acting as if “driven by a motor” (others may say the patient is restless, fidgety, or difficult to keep up with).
6. Often talks excessively.
7. Often blurts out an answer before a question has been completed.
8. Often has difficulty waiting his or her turn.
9. Often interrupts or intrudes on others (e.g., butts into conversations, games, or activities; may start using other people’s things without asking or receiving permission).

**2. Inattentive ADHD :**

In this type the patient suffers from lack of focus and attention which are regarded as the primary symptoms, not hyperactivity.

The next story will explain the meaning of inattentive ADHD in a better way, it’s a story about a child called Tammy that has ADHD and specifically the inattentive type.

**3. Combined Type ADHD:**

The last type is a mixture of the two previous types, it happens when inattention, hyperactivity, and impulsivity go hand-in-hand.

**the most diagnosed type of attention deficit hyperactivity disorder**

**is the Inattentive type**

This type has trouble paying attention to details, is easily distracted, often people with it have trouble organizing or finishing tasks. often forget routine chores and difficulty following instructions or completing tasks.

1. Doesn’t pay close attention to details or makes careless mistakes in school or job tasks.
2. Has problems staying focused on tasks or activities, such as during lectures, conversations or long reading.
3. Does not seem to listen when spoken to (i.e., seems to be elsewhere).
4. Does not follow through on instructions and doesn’t complete schoolwork, chores or job duties (may start tasks but quickly loses focus).
5. Has problems organizing tasks and work (for instance, does not manage time well; has messy, disorganized work; misses deadlines).
6. Often fails to give close attention to details in schoolwork, or during other activities.
7. Has difficulty remaining focused during tasks, classes, conversations, or lengthy reading.
8. Often children’s mind seems elsewhere, even in the absence of any obvious distraction.
9. starts tasks but quickly loses focus and fails to finish it and does not follow through on instructions.
10. Often avoids and dislikes engaging in tasks that require sustained mental effort.
11. Often loses things necessary for tasks or activities such as school materials, pencils, books or tools.
12. Often easily distracted by extraneous stimuli.
13. Often forgetful in daily activities.

The next story is a true story which will illustrate more the Inattentive type of ADHD.

**1.2.3 Tammy’s Story**

Tammy, an 8-year-old third grader, was halfway through the second grading period when her parents asked for another conference with her teacher. Her grades were very low with failure to complete class assignments and inconsistent performance on homework.

This wasn't the first time the school had raised concerns. Tammy was struggling with reading at the end of first grade in spite of testing by the school guidance counselor which showed very good ability. At the time Tammy complained that all the noises in the classroom were distracting. Further evaluation at that time through her pediatrician had suggested a possible diagnosis of ADHD, primarily inattentive type. Since there were no problems with hyperactivity her parents, especially her father, were reluctant to consider any medications. They got some self-help books to learn skills to help Tammy with her school assignments and address her "smart mouth" responses to adults. There was progress but her father complained that Tammy's mother was not consistent with discipline. Her mother argued that it didn't seem right that Tammy was always grounded and prohibited from all of her favorite activities.

To help identify Tammy’s difficulties and to address the disagreement between the parents regarding the correct approach, Tammy was referred to a child psychiatrist. That evaluation confirmed the diagnosis of ADHD. The parents’ efforts at providing Tammy with structure for her schoolwork were helpful, but they all agreed this was not enough. Her father's position had softened after he talked with his sister whose son also has ADHD and had a good response to treatment with medication. She also reminded him of the struggles he had in elementary school and all the times he was sent to the office.

In a session that included her parents Tammy talked about how tired she was of being in trouble all of the time and about how discouraged she was about her school efforts. She cried herself to sleep many nights.

A decision was made for treatment with stimulant medication. Tammy had an expected response. She was especially pleased when she was moved to a higher reading group and she was picked for her school basketball team. Her medication was tailored for school time since her evening and weekend behaviors were not a problem.

**1.3 Objectives**

**1.3.1 the technical side:**

1. Create as desktop app with all its capabilities
2. It has all the capabilities of the Desktop Apps.
3. Fast playing performance.
4. Good and stable storage.
5. Do not depend on the Internet.
6. The game starts with a learn & test phase for ADHD basic information for the Adults.
7. Contain 6 different Games, with different learning goals.
8. Contain 6 different learning materials for Math & Alphapts.
9. The 6 Games must have to be infinite playing games with different Goal and Difficult level at every single time.
10. The game contains a video tutorial before each level
11. At the end of the playing, there will be a Final Report that contains the Result Analysis of the child’s play and learning.
12. The game contains a shop, to buy items with game coins gained on every level.

**1. Create as desktop app with all its capabilities**

the benefit of using unity to create the Adhd game, Unity now supports

over 20 different target platforms for deploying, while its most popular

platform is the PC so the game is created as a desktop app.

**2. Fast playing performance**

The game performance is fast and there are no malfunctions or suspensions

**3. Good and stable storage**

Easy to store and does not take up much space.

**4. Do not depend on the Internet**

The child will not need an internet connection to be able to open the game, so he can open the game anywhere and anytime.

**5. The game starts with a learn & test phase for ADHD basic information for the Adults**

So that the child can log into the game a parent needs to read

information about ADHD to understand what the disease is, its

symptoms and whether or not their child has it, and then reading each

screen there are mcq questions that the parent must answer and they

exceed 80% From answering questions to be able to create an account

on the game.

**6. Contain 6 different Games, with different learning goals**

Our game contains 6 different educational games, the child will learn math and letters from the game.

**7. Contain 6 different learning materials for Math & Alphapts**

Our game contains 6 games with different content, four of them are for

Math, which are teaching numbers, addition, subtraction and

multiplication, and a game to teach English letters.

**8. The 6 Games must have to be infinite, playing games with different Goal and Difficult level at every single time**

Our game is infinite and continues with the child until the age of 15

years and with each level the difficulty of the game increases and thus

the child's concentration increases so that he can overcome the levels.

**9. The game contains a video tutorial before each level**

Before each level, the child must watch a video in order to learn the

content of the level before starting the game, and the child will not be

able to skip the video or start the game before watching the full video.

**10. At the end of the playing, there will be a Final Report that contains the Result Analysis of the child’s play and learning.**

The report contains the child’s score at each level, and thus it becomes

clear to us the child’s focus in each educational content, and whether

there is a level he should repeat or not, and whether he came out with

the expected result for him according to his educational level and it will

also become clear to us whether the game helped him or not.

**11. The game contains a shop, to buy items with game coins gained on every level.**

During the game the child can collect coins if he manages to exceed the

required during each level and with these coins he can buy the items he

needs from the store for each level such as changing the character.

**1.3.2 User Experiences Objectives:**

1. To be an interactive game.
2. A child from 5 to 15 years old can use it.
3. The child can use it on his own without difficulty.
4. Helps children reduce attention deficit.

**1.3.3 the conditions:**

1. The child has attention deficit hyperactivity disorder.
2. Not less than 5 years old, not older than 15.
3. He must be accompanied by a supervisor to register the account.
4. The supervisor must be one of the parents or a teacher of his own.
5. The supervisor must pass 80% of the test to pass the account registration process.

**1.4. Impact in business**

**1.4.1. impacts on:**

1. Business (Us)
2. Child (Main user)
3. Parents
4. Education
5. Society

**1.4.2. What effects will happen if our project is 1. implemented in reality?**

1. The academic level of ADHD Children will improve.
2. Parents will understand and know more about ADHD and how to deal with their children.
3. An agreement will be made with training centers to treat ADHD and how to benefit from our project.
4. There will be a new generation that will understand more about ADHD.
5. We will follow how users can get the most benefit.
6. At the end of the game a parent will take a report of how his child plays the game and what are the strengths & weakness points.

**1.5. Ways of Marketing**

**1.5.1. How to make profit?**

1. **Create Video Content**

If the target audience does not know about the game, how can they play it? This is one of the reasons why they made a video of its gameplay along with a promotional video.

1. **Build An Online Presence**

It means to make a social media profile on different profiles and share your content with the readers. to reach out to the target audience.

1. **Join Social Media Blogs**

Along with making social media profiles on different platforms, we need to join different gaming blogs as well. Most of the blog sites will allow us to promote our game, which will increase our game’s exposure.

1. **Cross-Promote the Game**

We will promote our project via different apps. It could imply that a free application without a clear income model can be made into a channel for cross-advertisement for money generating games.

1. **Get Reviews**

Getting reviews about the game will also encourage people to buy and download the game.

1. **Online Marketing**

by sharing our project or some of our games,

These links will help us to share our marketing of projects.

1. **Offline Marketing**

Awareness marketing by making offline events with faculty of science specialists.

1. **Feasibility Study**

Whatever we think need not be feasible .It is wise to think about the feasibility of any problem we undertake. Feasibility is the study of impact, which happens in the organization by the development of a system. The impact can be either positive or negative. When the positives nominate the negatives, then the system is considered feasible. Here the feasibility study can be performed in two ways such as technical feasibility and Economical Feasibility.

1. **Technical Feasibility.**

We can strongly say that it is technically feasible, since there will not be much difficulty in getting required resources for the development and maintaining the system as well. All the resources needed for the development of the software as well as the maintenance of the same is available in the organization here. We are utilizing the resources which are available already.

1. **Economic Feasibility**

Development of this application is highly economically feasible .The organization needed not spend much money for the development of the system already available. The only thing to be done is making an environment for development with effective supervision. If we are doing so, we can attain the maximum usability of the corresponding resources .

**1.6 Conclusion**

**ADHD**

1. ADHD stands for “attention deficit hyperactivity disorder”. It is a medical condition in which a person has differences in brain development and brain activity.
2. People with ADHD may also have trouble focusing their attention on a single task or sitting still for long periods of time.

**ADHD Type**

1. Inattentive: This type has trouble paying attention to details
2. Hyperactive: tendency toward impulsiveness
3. Combined: It is a mix between ADHD intention & hyperactive

**Our game contains**

1. learning phase and testing basic information on ADHD for adults.
2. a video tutorial before each level
3. 6 different learning materials for Math & Alphapts.
4. a Final Report that contains the Result Analysis of the child’s play and learning.
5. a shop, to buy items with game coins gained on every level.

**Who can use our game**

1. A child who is not less than 5 years old and not more than 15 years old
2. A child with an attention deficit.

**Ways of Marketing**

1. Create Video Content.
2. Build An Online Presence.
3. Join Social Media Blogs.
4. Get Reviews.
5. Cross-Promote the Game.

**Chapter 2: Related Works**

**2.1 Introduction**

This chapter includes much detailed information about the Edutropolis game project and its features that create a suitable environment for the child to learn. The Edutropolis game project helps you turn ADHD kids’ free time into a learning session, and it is designed to make children enjoy the fast-paced graphics and success they feel after completing the program's exciting levels and it blends learning, entertainment, and research-based cognitive stimuli to teach ADHD children so that the child does not find it difficult to learn.

Thus, we achieve learning with luxury based on increasing the child’s focus and awareness. It also includes the benefits that accrue to both the ADHD child and the parents after going through the experience in the Edutropolis game project and discovering it, depending on the main usage to teach the skills the ADHD kid needs to be successful in the classroom or workplace.

**2.2 Related Works of ADHD Applications**

**2.2.1 C8 Science**

C8 Sciences is the only company to offer a program to help kids increase their cognitive abilities and overall executive function. The program was originally intended specifically for kids with ADHD and other attention deficit disorders and their parents, but after it was found to make meaningful improvements with children that do not have these disorders we are now making the program available to everyone.

The program is flexible and easy to navigate and can help you enhance your child’s ability to feel a sense of pride and accomplishment both at home and in the classroom. It can be used on your electronic device or home computer. The program consists of a series of brain training games and activities that are challenging, educational, and fun! Kids love to play games that are fine-tuned to help them learn things like sequencing and categorizing.

The program recognizes when the child makes an error and coaches him until he successfully corrects the error and moves on to the next level. The games have delightful characters and graphics, just like other video games, but our games are designed to build your child’s confidence and self-esteem. The physical exercise component can be enjoyed by the whole family.

Playing games, exercising, and building athletic skills contribute to the success of the program, and to the well-being of everyone in the family! You will see improvements in your child’s ability to concentrate, complete tasks, and follow directions soon after you sign him up for the program.

**C8 Science Features**

1. Measures and improves working memory, self-control, sustained attention, and other Executive Function skills needed for daily life and to succeed in school.
2. Utilizes neuroscience-based computer games with exciting and colorful characters along with physical exercises the whole family can engage in and enjoy.
3. Easy-to-understand reporting using data collected from cognitive assessments based on cognitive science research literature with customized email alerts.

**2.2.2 Play Attention**

Play Attention is a customized program that integrates NASA inspired technology with cognitive skill training and behavior shaping to improve executive function and self-regulation.

They take attributes like hyperfocus, high energy, creative thinking, strong emotions, and risk-taking and harness them to take control of life. They become your superpowers to improve future outcomes at school, work, and home.

**Play Attention Features**

The program begins with games for:

1. Working memory
2. Spatial memory
3. Short-term memory
4. Planning
5. Finishing tasks
6. Overall focus and attention.

**2.2.3 BrainBeat**

BrainBeat is a therapeutic, at-home training system designed to improve a child’s focus, attention, grades, coordination, behavior, and more. It specially designed video games guide your child through fun & engaging physical & cognitive exercises shown to improve focus, brain synchronization, memory, and mental speed.BrainBeat is based on the research-backed medical technology called [Interactive Metronome (IM)](https://brainbeat.com/the-science-behind-brain-beat/).

IM has been studied by the nation’s leading institutions and used clinically for 20+ years by over 40,000 professional clinicians around the world. And this powerful brain-body technology is also used to help rehabilitate adults after traumatic brain injuries, strokes, and other neurological diseases.

**BrainBeat Features**

1. BrainBeat Cloud – syncs training progress & scores to the parent’s dashboard
2. Parent Dashboard – keep track of your child’s progress at a glance!
3. Improved Progress Tracker – track key metrics that show real-time progress! Unlock Badges & Achievement Awards – keeps motivation high!
4. New Leaderboards – see how you rank amongst your peers!
5. Email Coaching – ensures you get the most out of BrainBeat
6. User Management – now add and manage up to 5 household users!

**2.2.4 MentalUP**

MentalUP Brain is one of the leading brands in the EdTech industry, MentalUP is a Pedagogical Product certified, The Educate Awards winner education platform shaped by the collaborative study of academicians, child development experts, and game designers.

MentalUP Brain Stimulators Games for Kids help kids improve their skills including visual attention, divided attention, selective attention, focusing, problem-solving, and imagination. Improve your child’s cognitive skills with online mind games, have fun and learn at the same time.

**MentalUP Features**

1. Fun & improving free brain games
2. Personalized games for the brain development
3. Scientifically approved entertaining online brain games

**2.2.5 Khan Academy**

Khan Academy offers practice exercises, instructional videos, and a personalized learning dashboard that empower learners to study at their own pace in and outside of the classroom.

They tackle math, science, computing, history, art history, economics, and more, including K-14 and test preparation (SAT, Praxis, LSAT) content. They focus on skill mastery to help learners establish strong foundations, so there's no limit to what they can learn next.

**Khan Academy Features**

1. Activity Dashboard
2. Activity Tracking
3. Blended Learning
4. Content Library
5. Gamification
6. Instructor-Led Courses
7. Interactive Content
8. Mobile Access
9. Multi-Language
10. Progress Tracking
11. Quizzes
12. Self-Learning
13. Self-Paced Courses
14. Skills Tracking

**2.2.6 Roblox**

Roblox’s mission is to bring the world together through play. We enable anyone to imagine, create, and have fun with friends as they explore millions of immersive 3D experiences, all built by a global community of developers. It is described as a site created for gamers by gamers.

The other main function of the platform is socializing, users are encouraged to make friends with other online players. The site allows users to “Imagine, socialize, chat, play, create, interact and relate with others in many ways.” The Roblox Suite allows gamers to create their own game or create another world with friends or ‘virtual explorers’.

**Roblox Features**

1. Chat, Privacy & Filtering
2. Blocking Users & Reporting Abuse

**2.3 System Architecture**

1. ADHD content and MCQ test
2. Learning phase
3. Game playing phase
4. Report and analysis

**2.3.1 ADHD content and MCQ test**

We set some medical information and some questions based on this information for parents to see how they are aware of ADHD and to increase their knowledge about how to deal perfectly with their ADHD child.

**2.3.2 Learning phase**

We thought of the easiest and most appropriate way to display educational content for children in a way that suits their interests, so we chose to display it in the form of animated cartoon videos in order to make children enjoy the presentation and learning as well.

1. Alphabet:We teach the child the shape and arrangement of the alphabet by showing him/her an animated educational video so that he/she does not find it difficult to understand or memorize the shape and arrangement of the letters.
2. The child learns mathematics and its basis, such as the form of numbers, the method of counting, how to add two numbers together or subtract two numbers from each other, as well as the method of multiplying two numbers, then we provide him/her with an educational game after each lesson so that he applies what he has learned and thus learns here the ability to complete the homework required of him/her.

**2.3.3 Game playing phase**

We designed games to teach the child how to apply what he/she has learned and to teach him/her how to do the required tasks depending on himself/herself and his/her thinking and teach him/her how to plan to achieve goals.

1. **Alphabet Game**

The game is a word that appears on the screen, and there are boxes on it with the letters of the word, and the child must collect them in order.

1. **Find the missing Alphabet Game**

The game is a word that appears on the screen, but there are missing letters from the word, and the child is required to choose the letter that is missing from the existing boxes.

1. **Memory Game**

The game is a box with numbers, and what is required is that the child collides in two with the same number and behind each other in order to disappear.

1. **Order Game**

The game consists of boxes with numbers on it, but it is not visible. It appears when the child collides in the box. It is required that the child arrange these numbers in ascending or descending order according to the question.

1. **Math Game**

The game consists of levels, each of which is a mathematical problem that appears and is required to solve correctly so that a portal appears from which the next level enters.

The child is required in each level to collect 3 stars (one star for the required issue and 2 for collecting the coins in this level) - this is the method of calculating the score in each game.

**2.3.4 Report and analysis**

We set a final progress report analysis in order to enable the parent to monitor the child's progress and performance in learning, and it includes a summary of the game activity, achievements goals that the child recorded, and its performance in conducting the game tasks

**2.4 Edutropolis Features**

**2.4.1 Designed to help ADHD children to learn while having fun.**

It helps children to learn while playing useful games that enable them to gain new experiences, increase focus and enthusiasm, and improve their mental abilities to solve problems.

**2.4.2 It’s an educational personalized experience and allows parents to track their child's performance.**

We thought of a game that mixes learning with luxury to give the child serious scientific skills and outputs a final report for parents to track their child's performance.

**2.4.3 Short educational animated videos**

We thought of the easiest and most appropriate way to display educational content for children in a way that suits their interests, so we chose to display it in the form of animated cartoon videos in order to make children enjoy the presentation and learning as well.

**2.4.4 The child learns Alphabet**

We teach the child the shape and arrangement of the alphabet by showing him/her an animated educational video so that he/she does not find it difficult to understand or memorize the shape and arrangement of the letters

1. **Alphabet Game**

The game is a word that appears on the screen, and there are boxes on it with the letters of the word, and the child must collect them in order.

1. **Find the missing Alphabet Game**

The game is a word that appears on the screen, but there are missing letters from the word, and the child is required to choose the letter that is missing from the existing boxes.

**2.4.5 It addresses basic math, auditory processing, homework skills, and tasks specific to the individual**

The child learns mathematics and its basis, such as the form of numbers, the method of counting, how to add two numbers together or subtract two numbers from each other, as well as the method of multiplying two numbers, then we provide him/her with an educational game after each lesson so that he applies what he has learned and thus learns here the ability to complete the homework required of him/her.

1. **Memory Game**

The game is a box with numbers, and what is required is that the child collides in two with the same number and behind each other in order to disappear.

1. **Order Game**

The game consists of boxes with numbers on it, but it is not visible. It appears when the child collides in the box. It is required that the child arrange these numbers in ascending or descending order according to the question.

1. **Math Game**

The game consists of levels, each of which is a mathematical problem that appears and is required to solve correctly so that a portal appears from which the next level enters.

The child is required in each level to collect 3 stars (one star for the required issue and 2 for collecting the coins in this level) - this is the method of calculating the score in each game.

**2.4.6 Game Fixed/Dynamic** **Obstacles**

**There are two types of Obstacles in this game**

**1. Fixed Obstacle**

There are two types of Fixed Obstacles:

1. Type that if the child is mistaken, it will decrease the number of the answer
2. Type in which if the child is mistaken, the answer number will be increased (because the answer remains wrong and the gate the child crosses through will disappear).

**2. Dynamic Obstacle**

The Dynamic Obstacle: from its name, the child walks behind the player, and if the child messes with it, it increases or decreases the number of the answer to the question. It's similar to the Fixed Obstacle, but it is animated.

**2.4.7 Learning, Game time spent & Playing Scoring**

One of the most important things we built on our idea was to teach the child within the framework of teaching the child the importance of time and how to use it in the right ways. So we set a specific time for each educational game so that we teach the child to focus on solving tasks to get scores and complete tasks on time.

1. **Timer tool**

We used it to measure the amount of time a child spends to do a task required of the game and to increase the child’s focus and attention, so when the child completes the task before the time is up he/she will get a high score.

1. **Game Shop**

During the game, the child can collect coins if he manages to exceed the required amount during each level and with these coins, he can buy the items he needs from the store for each level such as changing the character.

**2.4.8 Final Progress Report Analysis**

We set a final progress report analysis in order to enable the parent to monitor the child's progress and performance in learning, and it includes a summary of the game activity, achievements goals that the child recorded, and its performance in conducting the game tasks

**Progress tracking**

We set progress tracking to help parents track what their child achieved in the past or current levels to track his/her performance.

**2.4.9 Instructions for parents include how to treat the child**

We set some medical information and some questions based on this information for parents to see how they are aware of ADHD and to increase their knowledge about how to deal perfectly with their ADHD child.

**2.4.10 The child will Play funny games based on planning and finishing tasks**

We designed games to teach the child how to apply what he/she has learned and to teach him/her how to do the required tasks depending on himself/herself and his/her thinking and teach him/her how to plan to achieve goals.

**2.5 Edutropolis Benefits**

**2.5.1 Benefits for the child:**

1. Easy to use and play for children and parents
2. Improves child’s focus energy on the right things
3. Increases his/her efficiency and performance
4. The child will learn:
   * Alphabets
   * The basics of arithmetic and counting numbers
   * How to add numbers
   * How to subtract numbers
   * How to multiply numbers
   * How to plan, finish tasks, and general focus and attention
5. The child realizes the importance of time by paying attention to time by completing the tasks required of him on time in order to get gifts
6. The child learns how to get points for buying things from the store
7. The child learns the ability to follow directions.
8. Improved mindfulness.
9. Improve impulse control
10. Increases attention.
11. Improves memory.
12. Develops thinking & problem-solving skills.

**2.5.2 Benefits for parents:**

1. Increase awareness of ADHD
2. The parent learns new things about the condition of ADHD and also learns how to deal with the child in a better way
3. He sees how the child responds, changes in his behavior for the better, and how quickly he learns and receives information
4. Realizes the importance of providing a suitable environment for the child to learn properly
5. Activity-tracking capabilities.

**2.6 Project Risk Management**

**What is risk management?**

Risk management is the process of identifying, assessing, and controlling threats to an organization's capital and earnings. These risks stem from a variety of sources including financial uncertainties, legal liabilities, technology issues, strategic management errors, accidents, and natural disasters.

A successful risk management program helps an organization consider the full range of risks it faces. Risk management also examines the relationship between risks and the cascading impact they could have on an organization's strategic goals.

**Why is risk management important?**

Risk management has perhaps never been more important than it is now. The risks modern organizations face have grown more complex, fueled by the rapid pace of globalization. New risks are constantly emerging, often related to and generated by the now-pervasive use of digital technology.

**2.6.1 Parent awareness about ADHD responsibilities**

ADHD is one of the most common neurodevelopmental disorders in childhood. It is usually first diagnosed in childhood and often lasts into adulthood. Children with ADHD may have trouble paying attention, controlling impulsive behaviors (may act without thinking about what the result will be), or be overly active.

In most cases, ADHD is best treated with a combination of behavior therapy and medication. For preschool-aged children (4-5 years of age) with ADHD, behavior therapy, particularly training for parents, is recommended as the first line of treatment before medication is tried. What works best can depend on the child and family. Good treatment plans will include close monitoring, follow-ups, and making changes, if needed, along the way.

And for those reasons, we need the parents to have a clear awareness of ADHD responsibilities.

So, we set some medical information and some questions based on this information for parents to see how they are aware of ADHD and to increase their knowledge about how to deal perfectly with their ADHD child.

The Parents need to pass the MCQ test with 80% results to access the Game Application for their children.

**2.6.2 Quantity of games**

A child with ADHD might get bored with just a small number of educational games if he mastered them all, so we made a variety of infinity educational games, and he can play them an unlimited number of times with different problems to solve every time to pass the game level.

**Chapter 3: Project Development Methodology**

**3.1 Software Development Life cycle**

Software development life cycle is the most important element in software development. It depicts the necessary phases in software development.

Software Development Life Cycle (SDLC) is a process of designing, building and maintaining software systems. Typically, it includes various phases from preliminary development analysis to post-development software testing and evaluation. It also consists of the models and methodologies that development teams use to develop the software systems, which the methodologies form the framework for planning and controlling the entire development process.

A software application or an information system is designed to perform a particular set of tasks. Often, this set of tasks that the system will perform provides well-defined results, which involve complex computation and processing. It is therefore a harsh and tedious job to govern the entire development process to ensure that the end-product comprises a high degree of integrity and robustness, as well as user acceptance. Thus, a systematic development process which is able to emphasize on the understanding of the scope and complexity of the total development process is essential to achieve the said characteristics of a successful system.

Software Engineering (SE) is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software, and the study of these approaches; that is, the application of engineering to software because it integrates significant mathematics, computer science and practices whose origins are in Engineering. Various processes and methodologies have been developed over the last few decades to improve software quality, with varying degrees of success. However it is widely agreed that no single approach will prevent project overruns and failures in all cases. Software projects that are large, complicated, poorly-specified, and involve unfamiliar aspects, are still particularly vulnerable to large, unanticipated problems. A software development process is a structure imposed on the development of a software product.

There are several models for such processes, each describing an approach test to a variety of tasks or activities that take place during the process. It aims to be the standard that defines all the tasks required for developing and maintaining software. These classic software life cycle models usually include some version or subset of the following activities:

* Planning and Visualization
* Requirement Analysis
* Software Modeling and Design
* Coding
* Testing and integration
* Deployment and Maintenance

**Fig. 3.1 SDLC**

And so far, there exist many SDLC models, one of which is the Agile model which will be explained in the next session.

**3.1.1 Agile Software Development**

Agile development is based on the idea of incremental and iterative development, in which the phases within a development life cycle are revisited over and over again. It iteratively improves software by using customer feedback to converge on solutions.

In agile development, rather than a single large process model that is implemented in conventional SDLC, the development life cycle is divided into smaller parts, called “increments” or “iterations”, in which each of these increments touch on each of the conventional phases of development. According to Agile Manifesto, the major factors of agile factors include the following four:

1. Early customer involvement

2. Iterative development

3. Self-organizing teams

4. Adaptation to change

There are currently six methods that are identified as agile development methods, which are Crystal methodologies, dynamic software development method, feature-driven development, lean software

development, scrum, and extreme programming.

Agile, also called an iterative life cycle model as it does not attempt to start with a full specification of requirements. Instead, development begins by specifying and implementing just part of the software, which can then be reviewed in order to identify further requirements. This process is then repeated, producing a new version of the software for each cycle of the model.

## **Main agile software development lifecycle phases**

The agile lifecycle is a structured series of stages that a product goes through. It consists of six phases:

### **Meet & Plan for requirements determination**

Stakeholders conduct an overall project assessment to determine the time and resources required for the development process. At the same stage, the owner assesses the risks and prioritizes the various functions depending on their business value.

1. **Design**

The software owner meets with the software development team and introduces them to the requirements outlined in the first step. The group then discusses the sequence for introducing functions and identifies the essential tools – the programming language, syntax libraries, and basic frameworks. At the same stage, software development teams can prototype the expected user interface.

1. **Development and coding**

After agreeing on the plan with the customer, the team develops the product itself. The product is delivered in stages, in separate sprints, each designed to improve the current version of the product. The initial release is likely to undergo many changes to provide improved functionality and new features.

Each cycle includes testing, and the final product must also undergo final testing. For this phase, you can use Scrum and the Kanban methodology, the development process based on individual tasks.

1. **Integration and testing**

At this point, the product becomes available to consumers, so the team must conduct a series of tests to ensure that the software is fully functional. If potential bugs or flaws are found, the developers will fix them immediately. At this stage, they also collected consumer feedback.

1. **Implementation and releasing**

The software is now fully deployed and available to customers. This action puts him in the maintenance phase. During this phase, the software development team provides ongoing support to keep the system running smoothly and fix any new bugs. Over time, further iterations are possible to update an existing product or add other functionality.

1. **Feedback**

That is the last stage of the Agile development cycle. After completing all the previous stages of development, the development team presents to the owner the result achieved in meeting the requirements. After that, the Agile software development phases start over – either with a new iteration or moving to the next stage and scaling Agile.

**Fig. 3.2 Agile model phases**

**Advantages:** The most important advantages of this model are the ability to respond to the changing requirements of the project. This ensures that the efforts of the development team are not wasted, which is often the case with the other methodologies. The changes are integrated immediately, which saves trouble later. There is no guesswork between the development team and the customer, as there is face to face communication and continuous inputs from the client. The documents are to the point, which does not leave any space for ambiguity. The culmination of this is that high quality software is delivered to the client in the shortest period of time and leaves the customer satisfied.

**Disadvantages:** If the projects are smaller projects, then using the agile model is certainly profitable, but if it is a large project, then it becomes difficult to judge the efforts and the time required for the project in the software development life cycle. Since the requirements are ever-changing, there is hardly any emphasis, which is laid on designing and documentation. Therefore, chances of the project going off the track easily are much more. The added problem is if the customer representative is not sure, then the project going off track increases manifold. Only senior developers are in a better position to take the decisions necessary for the agile type of development, which leaves hardly any room for newbie programmers, until it is combined with the senior’s resources.

**3.2 Chosen Methodology**

The Scrum approach is our chosen methodology as the main idea of Scrum is that systems development involves several environmental and technical variables (e.g. requirements, time frame, resources, and technology) that are likely to change during the process. This makes the development process unpredictable and complex, requiring flexibility of the systems development process for it to be able to respond to the changes. Scrum helps to improve the existing engineering practices (e.g. testing practices) in an organization, for it involves frequent management activities aiming at consistently identifying any deficiencies or impediments in the development process as well as the practices that are used.

In scrum approach responsibilities are divided based on roles and there are six identifiable roles in Scrum that have different tasks and purposes during the process and its practices: Scrum Master, Product Owner, Scrum Team, Customer, User and Management.#

**3.2.1 Roles and responsibilities**

**Scrum Master**

Scrum Master is a new management role introduced by Scrum. Scrum Master is responsible for ensuring that the project is carried through according to the practices, values and rules of Scrum and that it progresses as planned. Scrum Master interacts with the project team as well as with the customer and the management during the project. He is also responsible for ensuring that any impediments are removed and changed in the process to keep the team working as productively as possible.

**Product Owner**

Product Owner is officially responsible for the project, managing, controlling and making visible the Product Backlog list. He is selected by the Scrum Master, the customer and the management. He makes the final decisions of the tasks related to product Backlog, participates in estimating the development effort for Backlog items and turns the issues in the Backlog into features to be developed.

**Scrum Team**

Scrum Team is the project team that has the authority to decide on the necessary actions and to organize itself in order to achieve the goals of each Sprint. The scrum team y is involved, for example, in effort estimation, creating the Sprint Backlog, reviewing the product Backlog list and suggesting impediments that need to be removed from the project.

**User**

Users participate in the tasks related to product Backlog items for the system being developed or enhanced.

**Management**

Management is in charge of final decision making, along with the charters, standards and conventions to be followed in the project. Management also participates in the setting of goals and requirements. For example, the management is involved in selecting the Product Owner, gauging the progress and reducing the Backlog with Scrum Master.

Scrum does not require or provide any specific software development methods/practices to be used. Instead, it requires certain management practices and tools in the various phases of Scrum to avoid the chaos caused by unpredictability and complexity.

**3.2.2 Scrum Practices**

Scrum does not require or provide any specific software development methods/practices to be used. Instead, it requires certain management practices and tools in the various phases of Scrum to avoid the chaos caused by unpredictability and complexity.

In the following, the description of Scrum practices is given:



**Fig. 3.3 Practices of Scrum**

**Product Backlog :** Product Backlog defines everything that is needed in the final product based on current knowledge. Thus, Product Backlog defines the work to be done in the project. It comprises a prioritized and constantly updated list of business and technical requirements for the system being built or enhanced. Backlog items can include, for example, features, functions, bug fixes, defects, requested enhancements and technology upgrades. Also issues requiring solution before other Backlog items can be done are included in the list. Multiple actors can participate in generating Product Backlog items, such as customer, project team, marketing and sales, management and customer support.

This practice includes the tasks for creating the Product Backlog list, and controlling it consistently during the process by adding, removing, specifying, updating, and prioritizing Product Backlog items. The Product Owner is responsible for maintaining the Product Backlog.

**Sprint:** Sprint is the procedure of adapting to the changing environmental variables (requirements, time, resources, knowledge, technology etc.). The Scrum Team organizes itself to produce a new executable product increment in a Sprint that lasts approximately thirty calendar days. The working tools of the team are Sprint Planning Meetings, Sprint Backlog and Daily Scrum meetings. The Sprint with its practices and inputs is illustrated in Figure 3.3.

**Sprint Planning:** meeting A Sprint Planning Meeting is a two-phase meeting organized by the Scrum Master. The customers, users, management, Product Owner and Scrum Team participate in the first phase of the meeting to decide upon the goals and the functionality of the next Sprint. The second phase of the meeting is held by the Scrum Master and the Scrum Team focusing on how the product increment is implemented during the Sprint.

**Sprint Backlog:** Sprint Backlog is the starting point for each Sprint. It is a list of Product Backlog items selected to be implemented in the next Sprint. The items are selected by the Scrum Team together with the Scrum Master and the Product Owner in the Sprint Planning meeting, on the basis of the prioritized items (see 3.3) and goals set for the Sprint. Unlike the Product Backlog, the Sprint Backlog is stable until the Sprint (i.e. 30 days) is completed. When all the items in the Sprint Backlog are completed, a new iteration of the system is delivered.

**Daily Scrum meetings:** Daily Scrum meetings are organized to keep track of the progress of the Scrum Team continuously and they also serve as planning meetings: what has been done since the last meeting and what is to be done before the next one. Also problems and other variable matters are discussed and controlled in this short (approximately 15 minutes) meeting held daily. Any deficiencies or impediments in the systems development process or engineering practices are looked for, identified and removed to improve the process. The Scrum Master conducts the Scrum meetings. Besides the Scrum team also the management, for example, can participate in the meeting.

**Sprint Review:** meeting On the last day of the Sprint, the Scrum Team and the Scrum Master present the results (i.e. working product increment) of the Sprint to the management, customers, users and the Product Owner in an informal meeting. The participants assess the product increment and make the decision about the following activities. The review meeting may bring out new Backlog items and even change the direction of the system being built.

Our scrum process includes three phases: pre-game, development and post-game

**The pre-game phase** includes two sub-phases: Planning and Architecture/High level design. Planning includes the definition of the system being developed. A Product Backlog list (see 3.3) is created containing all the requirements that were currently known in our system. The requirements can originate from our targeted Users which are the kids who are diagnosed with ADHD and their parents. The requirements are prioritized and the effort needed for their implementation is estimated. The product Backlog list is constantly updated with new and more detailed items, as well as with more accurate estimations and new priority orders. Planning also includes the definition of the project tools and other resources, risk assessment and controlling issues and training needs. At every iteration, the updated product Backlog is reviewed by our team so as to gain our commitment for the next iteration.

The following table shows a part of our backlog when our team was still working on the project :

|  | **Product backlog items** | **Estimation of difficulty** | **Priority** | **Sprint** | **Status** | **Estimation of effort** |
| --- | --- | --- | --- | --- | --- | --- |
| **1** | Data Flow Diagram | low | 5 | 1 | Done | 4 |
| **2** | Use Case Diagram | low | 5 | 1 | In Progress | 3 |
| **3** | Sequence Diagram | medium | 4 | 1 | To Do | 10 |
| **4** | Entity Relationship Diagram | low | 5 | 1 | In Progress | 6 |
| **5** | welcome page | low | 7 | 1 | Done | 3 |
| **6** | Learning Phases | medium | 3 | 5 | To Do | 1 |
| **7** | MCQ Test Phases | low | 3 | 5 | To Do | 6 |
| **8** | Parents Registration | low | 9 | 1 | Done | 5 |
| **9** | Login | low | 10 | 1 | Done | 3 |
| **10** | Game Design | high | 1 | 2 | To Do | 72 |
| **11** | Splash Screen | medium | 6 | 4 | To Do | 12 |
| **12** | Menu Screen | medium | 6 | 4 | To Do | 8 |
| **13** | Game Categories | high | 1 | 2 | To Do | 10 |
| **14** | Memory game design | high | 1 | 2 | To Do | 6 |
| **15** | Order game design | high | 1 | 2 | To Do | 4 |
| **16** | Math game design | high | 1 | 3 | To Do | 7 |
| **17** | Alphabet collecting game design | high | 1 | 3 | To Do | 13 |
| **18** | Finding missing alphabet game design | high | 1 | 3 | To Do | 13 |
| **19** | Scoring System | high | 2 | 4 | To Do | 11 |
| **20** | Educational Content | low | 3 | 3 | To Do | 5 |
| **21** | Game Shop | low | 8 | 6 | To Do | 2 |
| **22** | Reporting System | medium | 5 | 4 | To Do | 8 |
| **23** | Database | high | 3 | 6 | To Do | 48 |
| **25** | Game Theme | medium | 9 | 6 | To Do | 10 |

**In the architecture phase**: the high level design of the system including the architecture is planned based on the current items in the Product Backlog. In case of an enhancement to an existing system, the changes needed for implementing the Backlog items are identified along with the problems they may cause. A design review meeting is held to go over the proposals for the implementation and decisions are made on the basis of this review. In addition, preliminary plans for the contents of releases are prepared.

**The development phase:** (also called the game phase) is the agile part of the Scrum approach. This phase is treated as a "black box" where the unpredictable is expected. The different environmental and technical variables (such as time frame, quality, requirements, resources, implementation technologies and tools, and even development methods) identified in Scrum, which may change during the process, are observed and controlled through various Scrum practices during the Sprints of the development phase. Rather than taking these matters into consideration only at the beginning of the software development project, Scrum aims at controlling them constantly in order to be able to flexibly adapt to the changes. In the development phase the system is developed in Sprints. Sprints are iterative cycles where the functionality is developed or enhanced to produce new increments. Each Sprint includes the traditional phases of software development: requirements, analysis, design, evolution and delivery (Figure 3.3) phases. The architecture and the design of the system evolve during the Sprint development. One Sprint is planned to last from one week to one month. There may be, for example, three to eight Sprints in one systems development process before the system is ready for distribution. Also there may be more than one team building the increment.

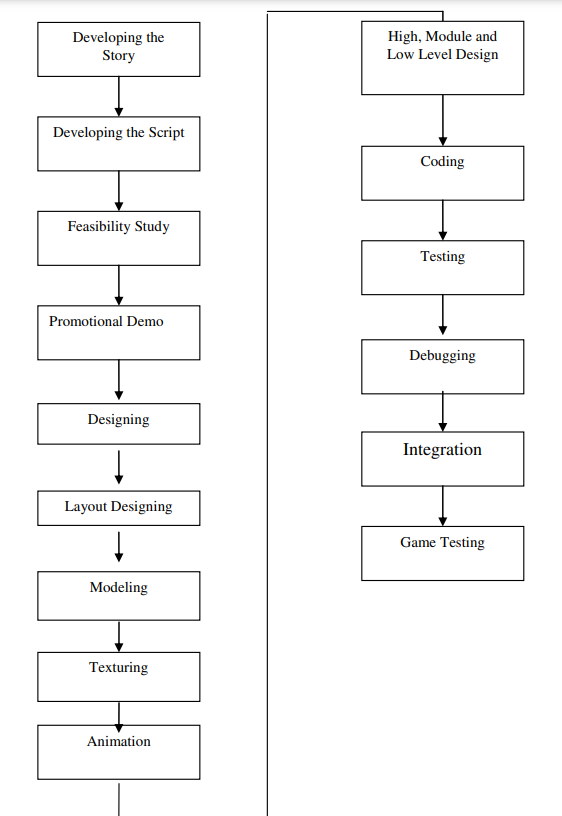
The development phase will be illustrated with all details in the next chapters.

**The post-game phase:** contains the closure of the release. This phase is entered when an agreement has been made that the environmental variables such as the requirements are completed. In this case, no more items and issues can be found nor can any new ones be invented. The system is now ready for the release and the preparation for this is done during the post-game phase, including the tasks such as documentation, integration and system testing to release a complete usable software

**3.2.3 Game Development Phases**

The birth of video games has slowly shifted the meaning of traditional games into a digitalized multimedia game. The term ‘games’ refers to the meaning of video games. Nowadays, games can be played on almost any device, and that is why developing games can be a profitable industry. To support the growth of the gaming industry, several original equipment manufacturers (OEM) publicly distribute their software development kit (SDK) and application programming interface (API) to attract people to become “indie developers” . According to Pressman, games are a kind of software which provides entertainment . However, game development using only software development life cycle (SDLC) faces several challenges. While SDLC is a systematic process of engineering to develop software, games are not purely a product of pure engineering. Game also is not just pure art, a creation of creativity and imaginative thinking, but game is more like a craft, created from the combination of interleaving, multidisciplinary aspects, from art, music, programming, acting, and the management and integration of those aspects. Therefore, a game development requires specific guidelines which govern its development process, the game development lifecycle (GDLC).

The phases in the game development life cycle are discussed in the following sections and the flow is shown in figure 3.4.





**Fig. 3.4 Phases in Games Development**

**Developing the story**

To start with, a core theme is developed with a well defined storyline. The story is generally targeted at a specific age group. Every story and game starts with an idea or a concept. This idea is then developed further. Characters are decided, situations, events and places are created, relationships determined and so on. Following that, various conflict and resolution scenarios are added. All the stories follow a similar structure. Apparently the translation of a story into a game is mere conflict resolution that can be easily achieved based on the designer’s creativity.

**Developing the script**

The story is then fine tuned to obtain a tight script that ensures a seamless flow of the game from one level to the other level. The better the script the better the game will be. The script writing needs a great deal of imagination and creativity. The key to writing a good script is to capture the imagination of the childrens. The characters and the backdrops are created and decided during this phase.

**Feasibility study**

This study is needed to assess the feasibility of a game. In addition it helps to freeze the requirements, the scope, profitability and other inferences of the proposed game. The areas to be reviewed within the feasibility study are overall analyses of the requirements, the pricing, technical, organizational, cultural and legal issues and the schedule of the project. This is a phase where project leaders, project design and development personnel and research group will be working together to create a game as a complete product. The programmers analyze and explain the programming limitations to the project managers. The outcome of the feasibility study is the feasibility study report, which comprises two parts - management summary and technical specification. The findings and recommendations are tabulated in the report.

**Promotional Demo**

A plan for the promotional demo for the game is prepared and is executed. Promotional demo is needed to attract potential customers. At this stage, since the game is not implemented the live demonstration is not possible. Hence a basic trial version is created for promotions.

**Designing**

The three main designing elements are the character design, background design and sets & props design. Computer games utilize characters to a large extent. In Character Design, while designing a convincing character for games, emphasis is given to characters with strong personalities and who can be of visual interest. This designing phase requires various techniques like use of visual design themes, silhouettes, poses, facial expressions and behavior of the characters.

**Layout design**

In this stage the layout for the entire game and for all the levels is designed. The layout represents the various sets, the passage ways with hurdles and the escape routes.. Every type of game design map will have certain constraints. Covering all the constraints in the map is quite challenging.

**Modeling**

Modeling is like character modelings, rules modeling. Models have to provide enough data so that the finer details can be rendered in an effective manner. There are many advanced techniques which are not very expensive, and they offer lots of features for creating models. One such advanced technique is the use of normal mapping.

**Texturing**

Texturing is a technique for adding detail, providing surface texture or color so that a realistic look can be given to the characters, sets and props. It is very common in almost all 3D applications.

**Animation**

After designing, modeling and texturing stages, the characters have to be animated to represent the sequence of actions in the game. Because the flow of the game is not linear, mesh methodology is used to animate a character.

The animation phase includes motion capture (mocap) animation and keyframe animation. In mocap animation each and every movement of the characters or objects enacted by humans or trained animals with either optical or magnetic sensors placed on them , is captured and recorded to create a series of scenes. These movements are later applied to the models. The only disadvantage is that it requires use of expensive camera/magnetic systems and very specific hardware and software. Key frames are snapshots of an image at a single point in time. Keyframe animation is the cycling of key frames to give the illusion of movement.

**High and Low level of Design**

The high level design gives the overall design of the game by identifying all the elements. It shows the abstract view of the system and it hides some of the details. The design also gives the relationships between the various modules involved in the development process in the form of data flow, flow charts, data structures etc., In module design high level design is segregated into various modules. The low level design details the high level design (i.e) the logic behind each of the modules is defined .

**Coding**

Coding is an important part of the game development process. This phase includes not only coding for making the gaming interactive but also for implementing the interface design. Various game engines are available in the market to code a game and execute the code and edit the game. There are separate sets of tools for making 2D games and 3D games. The 2D games tools include, GameMaker, Multimedia Fusion 2, Construct, Flixel, FlashPunk and Stencyl. 3D games tools are Unity , UDK , XNA , BlitzMax , jMonkeyEngine and Torque to name a few. Most games today on the PC platform are written using C++.

**3.2.4 How to limit the amount of iteration during video gaming development?**

Iteration was mentioned and tackled in a subtle manner and that behavior can be considered expected, since iteration is normalized in the video game industry. steadily the mentality of overworking in the video game industry is becoming a staple. Alas, there were many video game developers who nevertheless, without condemning iteration or identifying it as a problem, proposed effective solutions, like Kitase for FF7 Remake. These solutions are not mere assumptions or hopeful thinking, they are based on actual facts and have been put to practice and proved to be successful in limiting or eliminating iteration in rather unique ways.

These methods, depending on the situation, demand the reliance of developers either to already established franchises or to successful mechanics and aesthetics found in previous, unrelated video games. For instance, sequels of the games can just build upon the previous titles mechanics and expand on them. Another solution is that a good enough game can be published and then the sequels can use the exact same mechanics and design choices until the last one reaches the fullest potential that was initially envisioned for the first title. These solutions greatly accelerate the development and publishing pace and allow the companies to put in the market more products of a good quality at a faster pace.

Furthermore, another alternative solution is to increase the interaction of developers and customers to such an extent that developers will constantly seek how customers react and respond to choices about the game during the development. The concept of alliance with the customers may seem paradoxical at first, since it is not a practice that the majority of companies do, but this new practice offers many advantages. More importantly, constant interactivity with the customers can provide feedback that will prevent the need of retesting or trying to guess the tastes of the audience, thus limiting the need of iteration.

There are other types of interactions with customers or in other words allying with them. For instance, customers don’t have to interfere during the development of the game, a method that will alleviate the frustration of some developers who might feel that their work is compromised. Instead, customers can have the opportunity to receive the game in such a state that they can construct it to their own image, with the features they want to see. Thus, without the need of polishing the game, the developing company doesn’t have to conduct the time consuming qualitative iteration in the last stages of development. Such is the case of Bethesda, which has pushed the boundaries of interacting with the customers even further, as the company publishes games with clear problems and deficiencies (and at the same time rich in context and gameplay value) and leaves their corrections to the customers themselves by enabling the customers to do modifications to the games. Alternatively, a game can be published in fractures or episodes until gradually all the episodes will constitute a full game. In that way, the production pace of products is accelerated and the developers spend less resources for a small size product. In that regard, a small size product demands less iteration to be materialized. The developers can also evaluate the commercial reception of that small size product before they proceed to develop the other episodes or in other words before they dedicate the resources to formulate a fully fledged game.

**3.2.5 How Agile development’s iteration influences/affects the development of a game?**

This data stems from both the conducted interviews and also the collected secondary data. Through the intermediate theory, these different types of data act supplementary to each other.

It is important to note the level of understanding of Agile methodology by the video game industry’s associates. One important finding is that the Agile methodology is widely used by many developers in the video gaming industry. The notion and the secondary data also provides confirmation towards that conclusion. The theoretical framework provides data as well confirming that Agile methodology is the favorite methodology of managers in the video game industry. Agile is viewed by managers as ideal to develop video games and it turns into a staple in the industry. What also becomes staple in the industry are the characteristic features of Agile like the examined iteration in this thesis. In that sense, it becomes clear that the extensive adoption of Agile methodology results in the extensive adoption of iteration as well. This observation supports the significance and validity of that research question, by proving that the Agile methodology is becoming mainstream. The effects of Agile can be seen frequently and will be seen more often, the more developers decide to adopt it. products, the revenue in return is going to be smaller. It should be noted that project failure is not necessary a result of managerial incompetence but it’s due to the nature of iteration. Agile’s iteration commands and proclaims essentially that employees should never settle to staples and commonalities and demands constant experimentation to achieve uniqueness. This mentality goes against the best interest of any company to make profits.

According to all interviewees, iteration is not affecting the quality of creativity. The decision making in that area is mostly a collective process that takes into account both creative sensibilities and also practical issues found often in the video game industry, like whether some choices have an appeal to customers or that the creative vision is not static and constantly evolving and changing in accordance to the development of the video game. This is an important finding in this thesis, as artistic integrity constitutes the only layer of development that is not only unaffected by iteration but in return is enhanced by it. It seems to be widely accepted by developers that constant retesting and the benefit of more time adds value to the artistic vision and more chances to realize itself. This finding contradicts the findings of other researches which claim that artists resist the principles of Agile due to concerns over restricting their creativity.

**3.3 Applications of Chosen Methodology**

In this section, we briefly discuss some of the existing systems and games that are developed using agile methodology and scrum approach.

**FunPhy**

FunPhy which stands for Fun Physics and it is a mobile game application that supports fun learning for students especially for physics subjects. It was designed by using Agile methodology.

This educational game is intended for junior high schools so that it can help students to master physics. An important characteristic in the system development with Agile method is oriented to the completion of the information system and the software application as a product . Compared to processes and tools, the Agile method focuses on the interaction between individuals . In addition, a change arose during the development process gives better response than following the set plan from the first step of the project.

****

**Fig. 3.5 FunPhy Game**

**PRO FEEL GOLF**

Definitely, Agile is a proven way to achieve success. It helps in easy tracking of the progress and turns the whole process into a scalable one, that’s why pro feel golf game is also developed based on Agile methodology using Unity3D technology and it is a Sports-simulation Genre game for iOS & Android.



**Fig. 3.6 PRO FEEL GOLF**

**Game**

**Kids Games: For Toddlers 3-5**

It is an amazing learning and educational game for children of all ages. It has been developed using agile methodology and it is designed to help teach colors, shapes, coordination, motor skills, memory, and more

**Fig. 3.7 Kids Games**

**Antidote COVID-19**

This game is another example of the games that use agile methodology in its development process.The Antidote COVID-19 is a game to turn complex, scientific information into a fun learning experience. During the course of the game, players will learn about their immune system, pathogens, vaccines and how to protect themselves from COVID-19.

**Fig. 3.8 Antidote COVID-19 Game**

**Chapter 5: Software Analysis and Design**

**5.1 Introduction**

This document is a software requirement specification for the Edutropolis Game Project which is a desktop application. After giving information about the definition of the project at the beginning part of the document, we will give a complete description for an overview and list the requirements which meet the needs of the users.

**5.2 Purpose of document**

This document aims to give a brief description of the Edutropolis Game Project. With the help of this document the needs of the Game Project and the solution that will be provided to that needs will be clearly presented. In other words, the purpose is to outline the functional requirements of the Edutropolis Game. This document is intended for:

* Instructors
* Developers
* Testers

**5.3 Project Requirements**

**5.3.1 Requirements:**

* Define the function of the system from the client’s viewpoint.
* The requirements establish the system’s functionality, constraints, and goals by consultation with the customers and users.
* The requirements may be developed in a self-contained study or may emerge incrementally.
* The requirements form the basis for acceptance testing.
* The development team and client need to work together closely during the requirements phases of the software project.
* The requirements must be developed in a manner that is understandable by both the client and the development staff.

**5.3.2 Requirement Goals:**

* Understand the requirements in appropriate detail.
* Define requirements in a manner that is clear to the client. This may be a written specified prototype system or other forms of communication.
* Define requirements in a manner that is clear to people who will design, implement and maintain the system.
* Ensure that the client and developers understand requirements and their implications.

**5.3.3 Functional Requirements:**

Describe the functions that the system must perform. They are identified by analyzing the use made of the system and include topics such as

* Functionality.
* Data.
* User interfaces.

**5.3.4 Non-Functional Requirements:**

Requirements that are not directly related to the functions that the system must perform.

* Product requirements
* Performance, reliability, portability, etc.…..
* Organizational requirements
* Delivery, training, standards, etc.……..
* External requirements
* Legal, interoperability, etc.…….
* Marketing and public relations.

| ID | Requirement | Related Features | Functional/ Non-Functional |
| --- | --- | --- | --- |
| 1 | Educational Game | * Educational Material for Children from 5 to 15 years old. * For Children diagnosed with ADHD (Attention deficit hyperactivity disorder) * Based on play-to-learn methods. | Functional |
| 2 | 3D | * 3D geometry for the environment and characters * Low Poly environment and characters * Use simple texture and materials * render the Scene using perspective * include entire movement through three-dimensional planes * able to move around in a "real world" setting where they can turn 360 degrees * objects have length, height, and depth * Objects can fall into space out the level edges. * elements like light and sound can behave as they do in real life | Functional |
| 3 | Educational Material | * Introduced through simple animated video * Have sounds and music to make learning easy and fun * Pre-recorded educational animated videos * Maths Educational Material * Alphabets Educational Material * Introduced through Educational Games | Functional |
| 4 | Maths Educational Material | * Pre-recorded educational animated videos: * Adding numbers educational animated video * Subtracting numbers educational animated video * Multiplication numbers educational animated video * Numbers Comparing educational animated video * Counting numbers educational animated video * 3D Maths games: * Ordering numbers game * Numbers memory game * Maths problem game * Maths wheel game | Functional |
| 5 | Alphabets Educational Material | * Pre-recorded educational animated videos: * educational animated video for 26 English character * Every character have a unique learning song * Every song have many examples for the character * 3D Alphabet games: * Collecting letters game * Find the missing letter | Functional |
| 6 | Ordering numbers game | * Having a number of boxes containing some random numbers * The numbers on the boxes need to be ordered (ascending order) * If the player gets the wrong order the numbers disappears | Non-Functional |
| 7 | Numbers memory game | * Having a number of boxes containing some random pair of numbers * The player should remember the number on each box and hit the matching pairs | Non-Functional |
| 8 | Maths problem game | * The player should write the solution for the maths problem equation in each level. * The player avoids the obstacles because they can mess with his solution | Non-Functional |
| 9 | Maths wheel game | * An infinity game where the player solves math problem equations appears in the center of the wheel * The wheel contains 4 numbers and one of these numbers is the solution * The player should pass the ball through the part containing the right solution for the equation. | Non-Functional |
| 10 | Collecting letters game | * Having a number of boxes containing some Alphabets * These alphabets together make a word * The player should collect the alphabets in the correct order of the word inside the level | Non-Functional |
| 11 | Find the missing letter | * Having a number of boxes containing some Alphabets * The player should hit the box containing the right letter for the word | Non-Functional |
| 12 | Ball (Playable character) | * Rolling ball * Soccer ball skin * Can be controlled by keyboard * Can move in all 4 directions (up, down, left, right) * Can access the game modes * Can access the shop * Can hit the obstacles * Destroy the obstacles with an ability | Non-Functional |
| 13 | ADHD Education Building | * Contain ADHD basic knowledge * Contain ADHD basic knowledge MCQ test * Contain 3 screens for ADHD basic knowledge * Contain 3 screens for ADHD basic knowledge MCQ test * The adult must pass 80% of the test to unlock the game | Functional |
| 14 | Obstacles | * Have Chasing Cubes * Have Score Manipulation Cubes * Have Level Edges | Functional |
| 15 | Chasing Cubes  (Game Obstacle) | * Have a specific range of area * If the player in the this range, the cube will chase the player’s ball * The Chasing Cube trys to push the ball out of the level edges | Non-Functional |
| 16 | Score Manipulation Cubes  (Game Obstacle) | * Have a specific range of area * If the player is in this range, the cube will chase the player’s ball * If the Score Manipulation Cube hit the ball on Maths problem game mode it adds or subtracts the result of the math problem * It has 2 types: adding cube or subtracting cube | Non-Functional |
| 17 | Level Edges  (Game Obstacle) | * Restart the level if the player falls out of the level edges * The level has 4 edges | Non-Functional |
| 18 | Score System | * The score consists of stars and coins * Each level need 3 stars to be completed * Each game level has a number of coins * The player gets 1 star for solving the level problem * The player gets 2 stars for collecting the level conis | Functional |
| 19 | Coins (Score System) | * Each game level has number of coins * The player gets 2 stars for collecting the level conis | Non-Functional |
| 20 | Stars (Score System) | * Each level needs 3 stars to be completed * The player gets 1 star for solving the level problem * The player gets 2 stars for collecting the level coins. | Non-Functional |
| 21 | Game Environment | * City (Buildings: Game Modes) * Shop (Skins - Special Ability) * Level (Timer - Problem - Obstacles - Score - Portal - playground) * Education Building (Classes) | Functional |
| 22 | City  (Game Environment) | * The main environment is a city * Each building of the city represents a game mode * When the ball hits any of the game mode buildings it enters the level selection menu of the game mode * When the player hit the shop it enters the shop and choose from different options * When the ball hit the Educational Building it enters Educational Building and access the educational content there. | Functional |
| 23 | Shop  (Game Environment) | * When the ball hits the shop it enters it * The player can choose the skins and buy the locked skins. * The player can choose and buy the locked ability. | Functional |
| 24 | Level  (Game Environment) | * The level playground has a square shape. * The ball moves in this square shape. * All the coins and obstacles spawned inside the playground randomly. * The portal shaws on the edge of the playground when the player ends the level. | Functional |
| 25 | Education Building (Game Environment) | * The ball can hit the educational building. * The player accesses the educational content through the educational building. |  |
| 26 | Maths Wheel Building (Game Environment) | * The ball can hit the educational building * The player can access the Maths Wheel Building through the educational building. | Functional |
| 27 | Score Report | * Total Score * Total Stars * Percent of Complete * Report for every Game Mode * Total Report for all Games | Functional |
| 28 | Report for every Game Mode  (Score Report) | * Total Score for each game mode separately * Total Stars for each game mode * Percent of Complete | Non-Functional |
| 29 | Total Report for all Games  (Score Report) | * Total Score for all the game modes together * Total Stars for all the game modes together * Percent of Complete | Non-Functional |
| 30 | The program shall have a GUI | All user interaction will occur through the GUI | Functional |
| 31 | Game modes | The program has 6 game modes. | Functional |
| 32 | Save game | The program shall have a method of saving the game. | Functional |
| 33 | Score Tracker | The program shall have a method for keeping track of high scores for different modes | Functional |
| 34 | main menu | The program shall have a main menu | Functional |
| 35 | High Scores screen | The program shall have a High Scores screen | Functional |
| 36 | multiple level GUIs | The program shall have multiple levels GUIs | Functional |
| 37 | Run on PC | The system will run on any personal computer with specs meeting or beyond the following: 1.5ghz CPU, 512MB RAM, 500MB Free Hard Drive Space, and a 64MB GPU. | Functional |
| 38 | keyboard and mouse | The PC running the system must have a keyboard and mouse | Functional |
| 39 | description of each mode | A short description of each mode will be available in the active system | Functional |
| 40 | background music | The game will load the background music for specific levels these files will be in .mp3 format for usability on most systems. | Functional |
| 41 | Sound | * In-game sound * Menu sound | Functional |
| 42 | Game icons | * 2D simple icons | Functional |
| 43 | UI  (Game User Interface) | Simple game design with buttons and  screens that the player can interact with  to play in an easy experience. | Functional |
| 44 | Input Controllers | Give players the ability to control the game  by using the keyboard inputs. | Non-Functional |

**5.4 Software Architecture**

**What is Software Architecture?**

Software architecture refers to the fundamental structures of a software system and the discipline of creating such structures and systems. Each structure comprises software elements, relations among them, and properties of both elements and relations. The architecture of a software system is a metaphor, analogous to the architecture of a building. It functions as a blueprint for the system and the developing project, laying out the tasks necessary to be executed by the design teams.

Software architecture is about making fundamental structural choices that are costly to change once implemented. Software architecture choices include specific structural options from possibilities in the design of the software. For example, the systems that controlled the Space Shuttle launch vehicle had the requirement of being very fast and very reliable. Therefore, an appropriate real-time computing language would need to be chosen. Additionally, to satisfy the need for reliability the choice could be made to have multiple redundant and independently produced copies of the program, and to run these copies on independent hardware while cross-checking results.

Documenting software architecture facilitates communication between stakeholders, captures early decisions about the high-level design, and allows the reuse of design components between projects.

**5.4.1 Game Architecture Definition**

Game architecture means the various software components of a video game, such as the game’s client, protocol, external servers, and any other software or copyrighted content required for a video game to function.

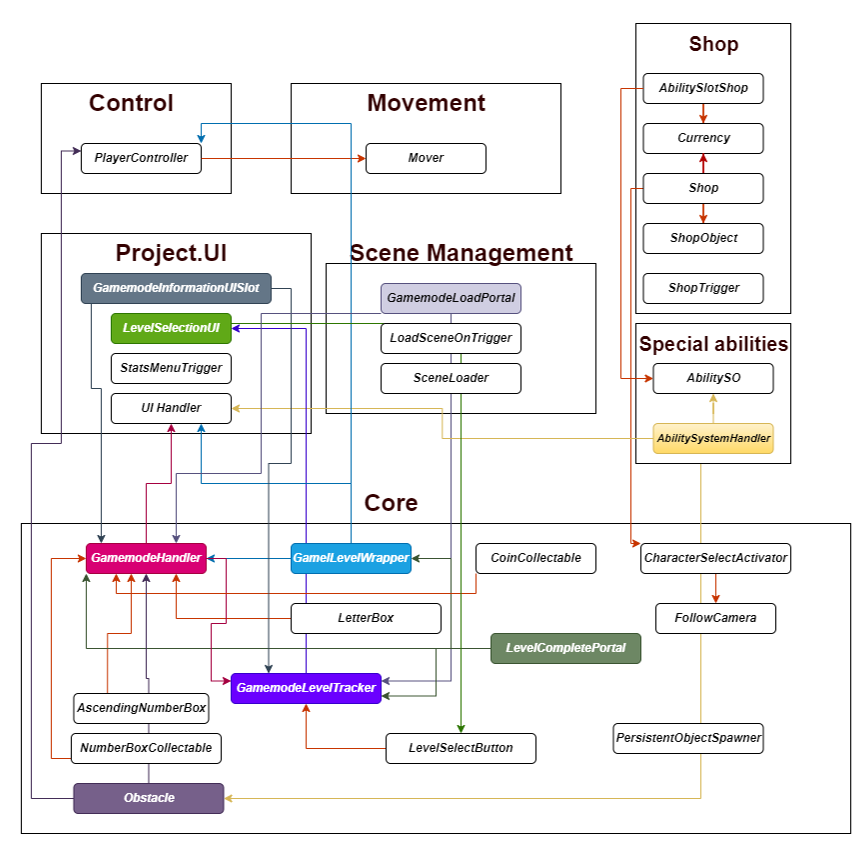
**5.4.2 Entity Relationship Diagram (ERD)**

**What is an ER diagram?**

An Entity-Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects, or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education, and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals, and connecting lines to depict the interconnectedness of entities, relationships, and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs.

ER diagrams are related to data structure diagrams (DSDs), which focus on the relationships of elements within entities instead of relationships between entities themselves. ER diagrams also are often used in conjunction with data flow diagrams (DFDs), which map out the flow of information for processes or systems.

**5.4.3 Game Architecture Components & ER Diagram**

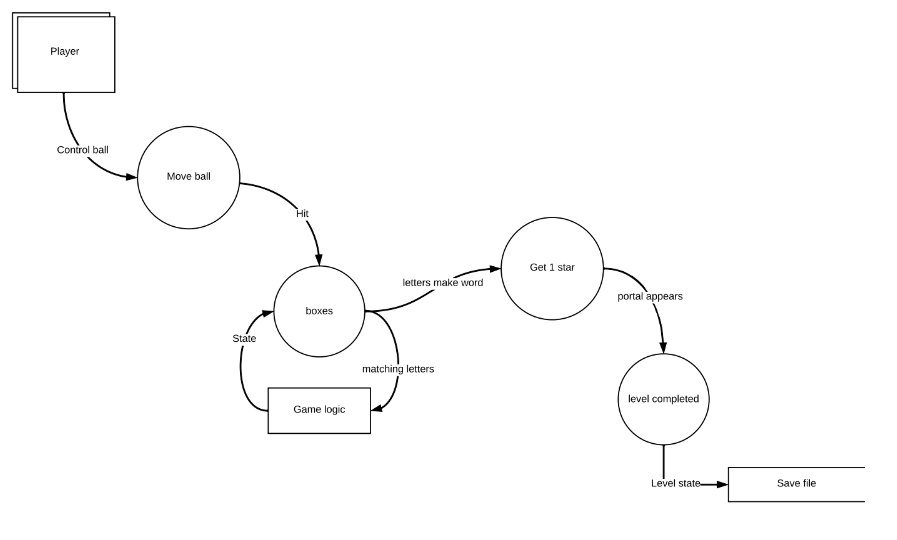


**5.5 Data Flow Diagram**

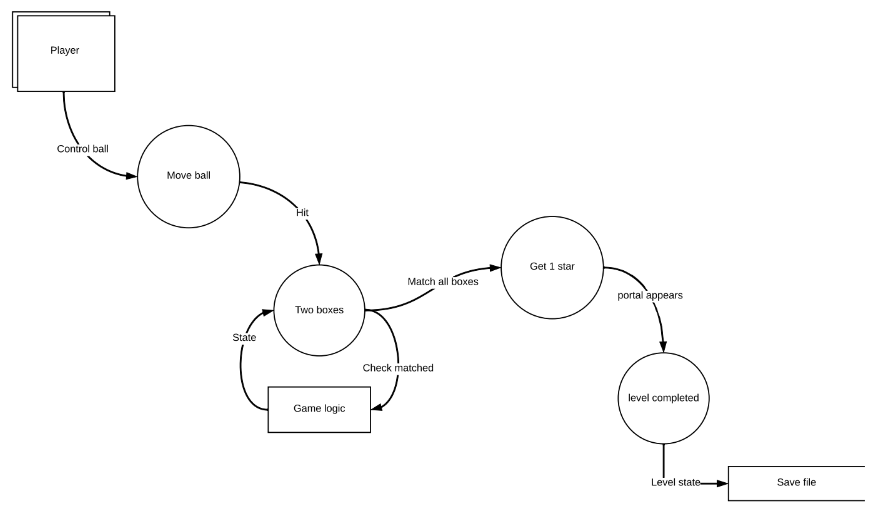
**What is the Data Flow Diagram?**

* It aids in describing the boundaries of the system.
* It is beneficial for communicating existing system knowledge to the users.
* A straightforward graphical technique that is easy to recognize.
* DFDs can provide a detailed representation of system components.

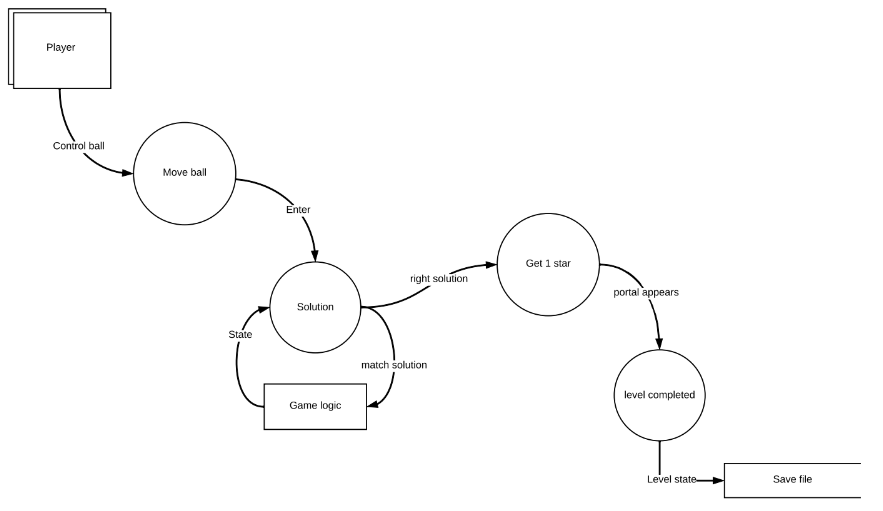
**5.5.1 Alphabet Collecting Game Data Flow Diagram**



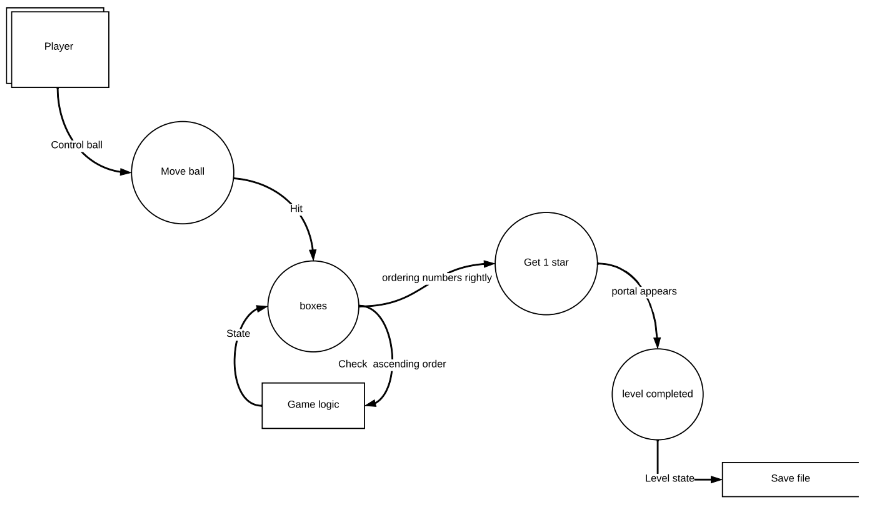
**5.5.2 Memory Game Data Flow Diagram**



**5.5.3 Math Problem Game Data Flow Diagram**



**5.5.4 Number Order Game Data Flow Diagram**



**5.5.5 Missing Alphabet Game Data Flow Diagram**



**5.6 System Design**

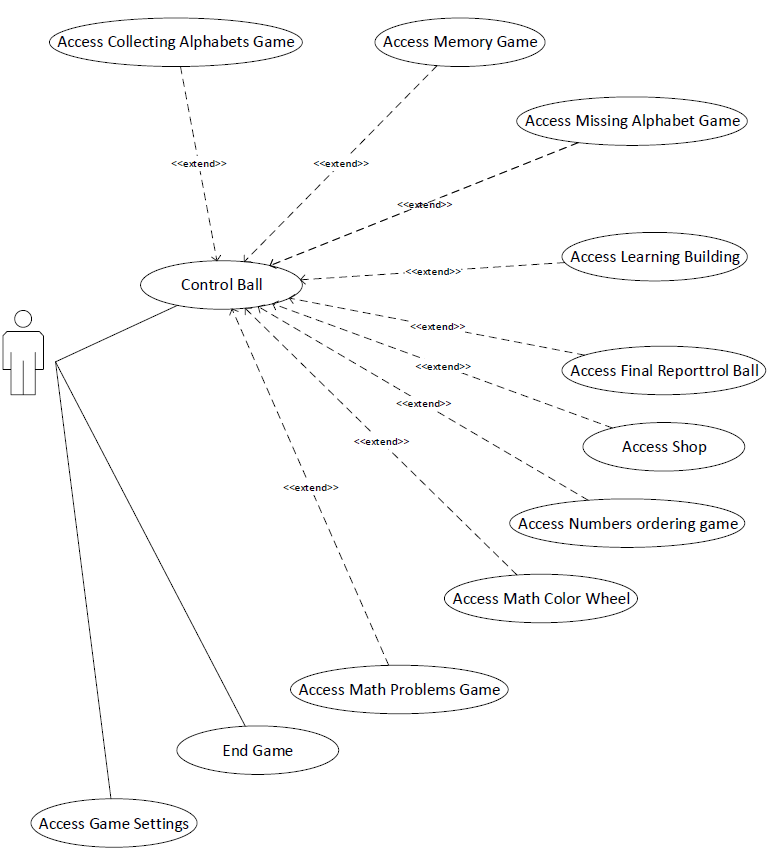
**5.6.1 Use Case Diagram**

**What is a use case diagram?**

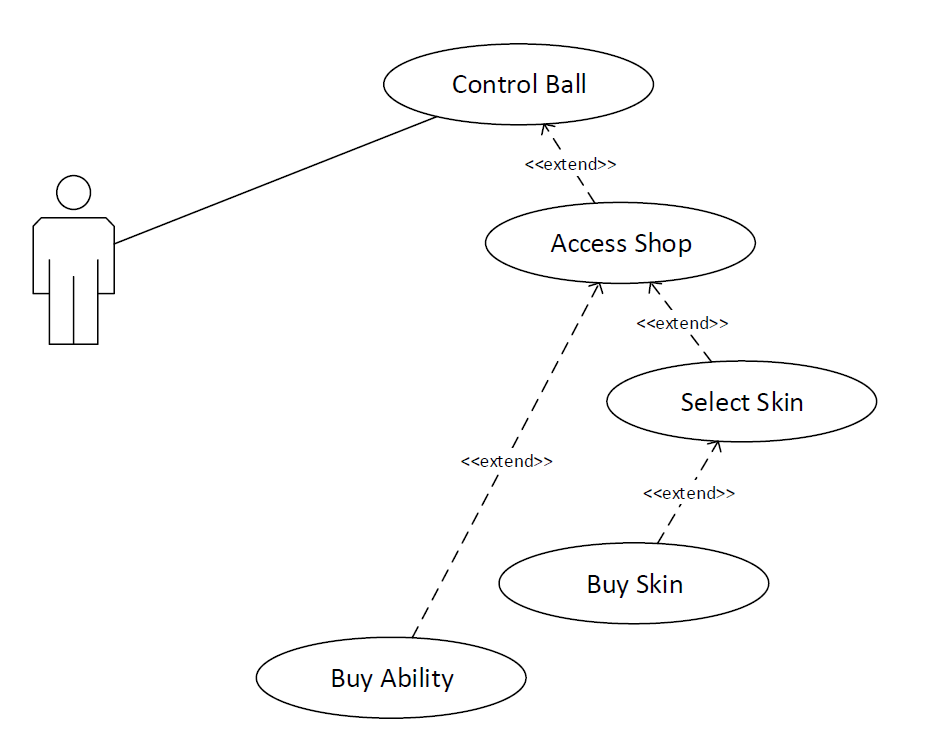
In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:

* Scenarios in which your system or application interacts with people, organizations, or external systems
* Goals that your system or application helps those entities (known as actors) achieve
* The scope of your system.

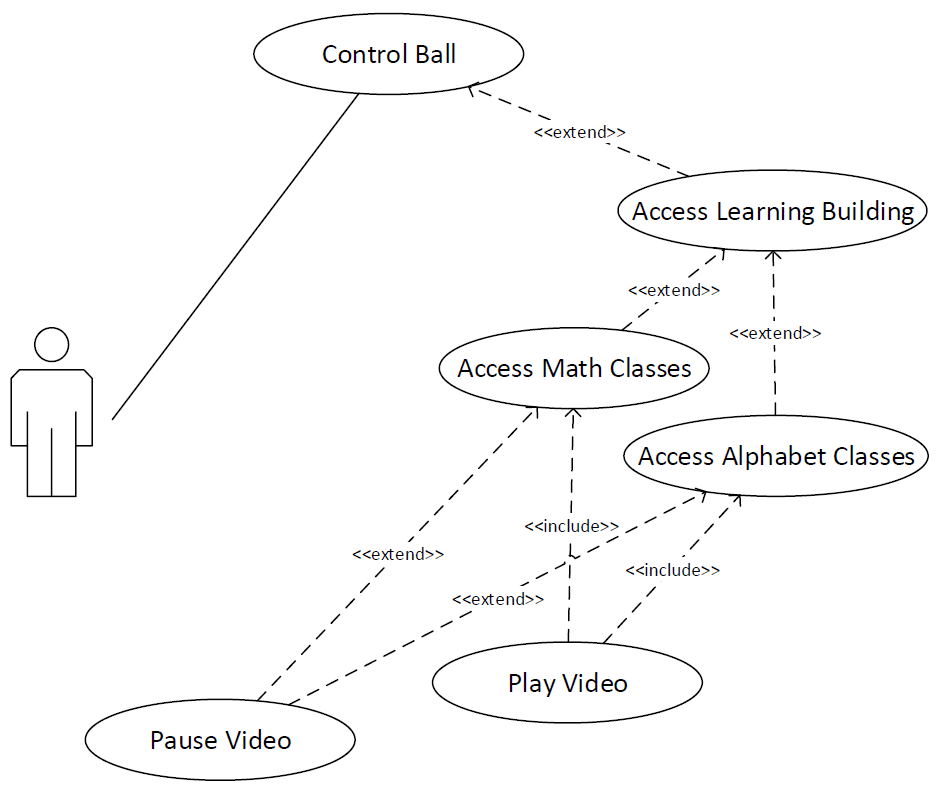
**5.6.2 Main Screen Use Case Diagram**



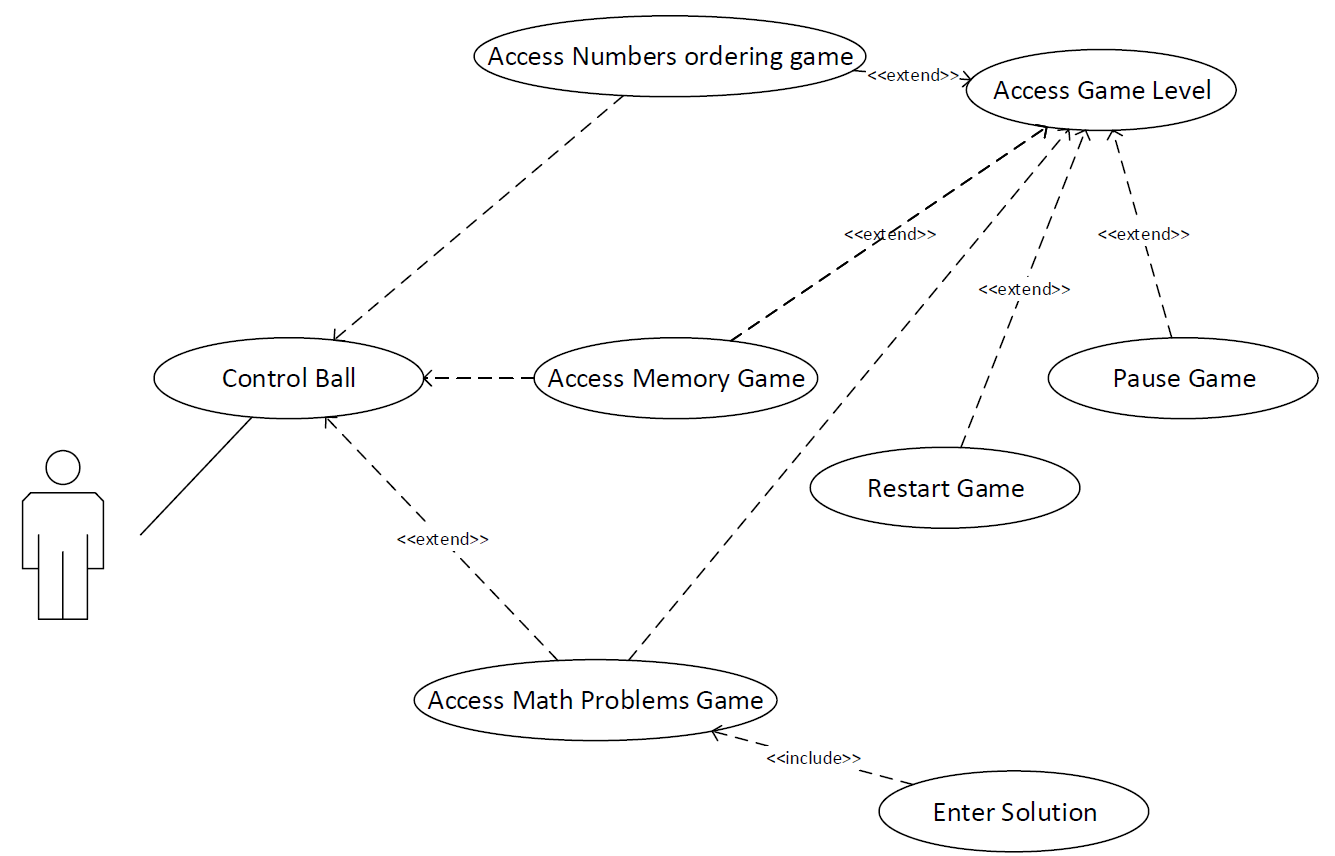
**5.6.3 Shop Use Case Diagram**



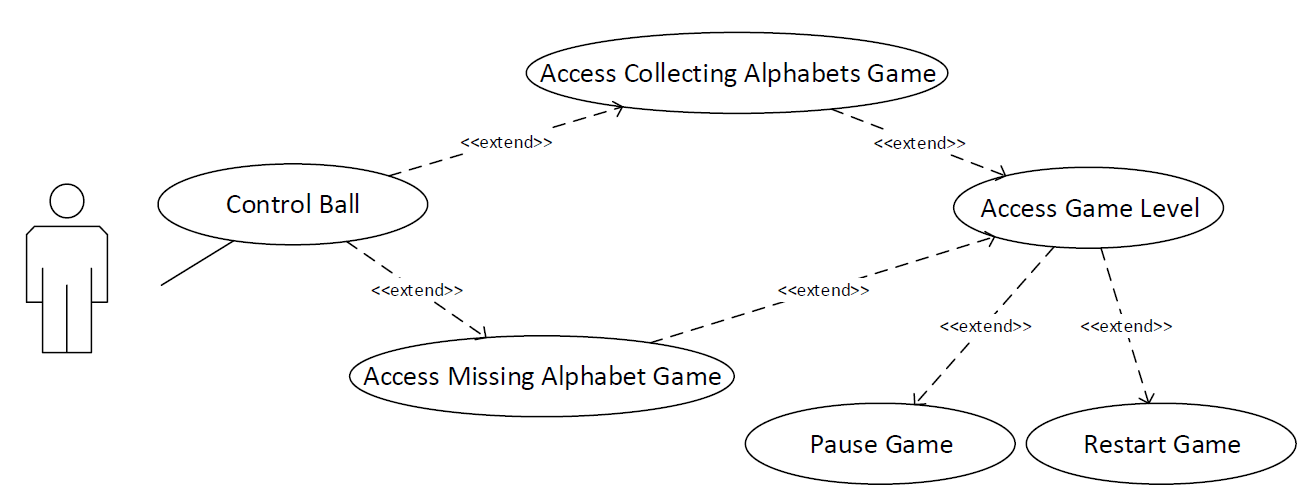
**5.6.4 Learning Building Use Case Diagram**



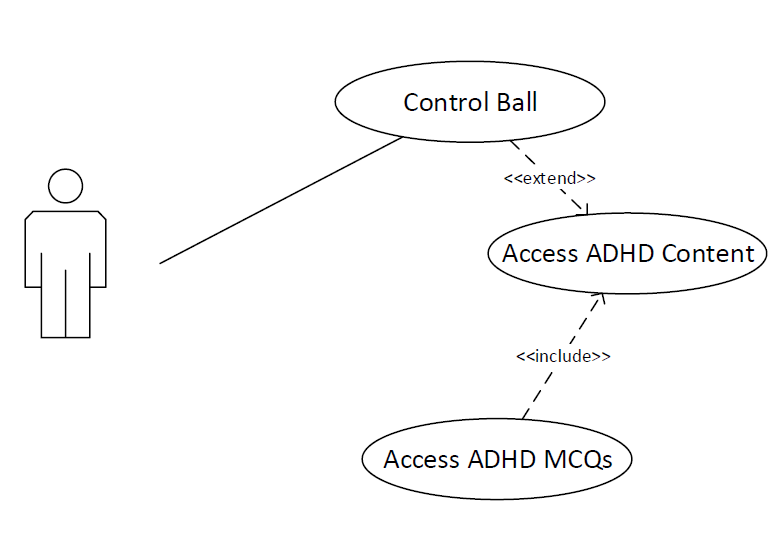
**5.6.5 Math Games Use Case Diagram**



**5.6.6 Alphabet Games Case Diagram**



**5.6.7 Adult User Use Case Diagram**



**5.7 Conclusion**

In this document, the project we develop that implements a self-education game is explained in detail by using use-case diagrams, entity-relationship diagrams, data flow diagrams, flow chart diagrams, and game architecture components diagrams. Firstly, the Edutropolis project that we try to develop is explained briefly. Then the gameplay is explained with use cases. All the possible actions in the game are explained in detail. Secondly, non-functional requirements are explained in separate sections.

**Chapter 6: Project Software Tools and Technologies**

**6.1 Unity Game Engine**

**What is Unity Game Engine?**

Unity is a cross-platform game engine developed by Unity Technologies, first announced and released in June 2005 at Apple Worldwide Developers Conference as a Mac OS X game engine. The engine has since been gradually extended to support a variety of desktop, mobile, console and virtual reality platforms. It is particularly popular for iOS and Android mobile game development and is considered easy to use for beginner developers and is popular for indie game development.

The engine can be used to create three-dimensional (3D) and two-dimensional (2D) games, as well as interactive simulations and other experiences. The engine has been adopted by industries outside video gaming, such as film, automotive, architecture, engineering, construction, and the United States Armed Forces.

**6.1.1Game Engine Architecture**

**What Is a Game?**

We probably all have a pretty good intuitive notion of what a game is. The general term “game” encompasses board games like chess and Monopoly, card games like poker and blackjack, casino games like roulette and slot machines, military war games, computer games, various kinds of play among children, and the list goes on. In academia we sometimes speak of “game theory,” in which multiple agents select strategies and tactics in order to maximize their gains within the framework of a well-defined set of game rules. When used in the context of console or computer-based entertainment, the word “game” usually conjures images of a three-dimensional virtual world featuring a humanoid, animal, or vehicle as the main character under player control. (Or for the old geezers among us, perhaps it brings to mind images of two-dimensional classics like Pong, Pac-Man, or Donkey Kong.) In his excellent book, A Theory of Fun for Game Design, Raph Koster defi nes a “game” to be an interactive experience that provides the player with an increasingly challenging sequence of patterns which he or she learns and eventually masters. Koster’s assertion is that the activities of learning and mastering are at the heart of what we call “fun,” just as a joke becomes funny the moment we “get it” by recognizing the pattern.

For the purposes of this book, we’ll focus on the subset of games that comprise two- and three-dimensional virtual worlds with a small number of players (between one and 16 or thereabouts). Much of what we’ll learn can also be applied to Flash games on the Internet, pure puzzle games like Tetris, or massively multiplayer online games (MMOG). But our primary focus will be on game engines capable of producing first-person shooters, third-person action/platform games, racing games, fighting games, and the like.

**What Is a Game Engine?**

The term “ game engine” arose in the mid-1990s in reference to first-person shooter (FPS) games like the insanely popular Doom by id Software. Doom was designed with a reasonably well-defined separation between its core soft - ware components (such as the three-dimensional graphics rendering system, the collision detection system, or the audio system) and the art assets, game worlds, and rules of play that comprised the player’s gaming experience. The value of this separation became evident as developers began licensing games and re-tooling them into new products by creating new art, world layouts, weapons, characters, vehicles, and game rules with only minimal changes to the “engine” software. This marked the birth of the “mod community ”—a group of individual gamers and small independent studios that built new games by modifying existing games, using free toolkits provided by the original developers. Towards the end of the 1990s, some games like Quake III Arena

and Unreal were designed with reuse and “ modding” in mind. Engines were made highly customizable via scripting languages like id’s Quake C, and engine licensing began to be a viable secondary revenue stream for the developers who created them. Today, game developers can license a game engine and reuse significant portions of its key software components in order to build games. While this practice still involves considerable investment in custom software engineering, it can be much more economical than developing all of

the core engine components in-house. The line between a game and its engine is often blurry. Some engines make a reasonably clear distinction, while others make almost no attempt to separate the two. In one game, the rendering code might “know” specifi - cally how to draw an orc. In another game, the rendering engine might provide

general-purpose material and shading facilities, and “orc-ness” might

be defined entirely in data. No studio makes a perfectly clear separation between the game and the engine, which is understandable considering that the definitions of these two components often shift as the game’s design solidifies.

Arguably a data-driven architecture is what differentiates a game engine from a piece of software that is a game but not an engine. When a game contains hard-coded logic or game rules, or employs special-case code to render specific types of game objects, it becomes difficult or impossible to reuse that software to make a different game. We should probably reserve the term “game engine” for software that is extensible and can be used as the foundation for many different games without major modification.

**6.1.2 Why use unity game engine?**

Unity 1.0 was released in June 2005, when developers Nicholas Francis, Joachim Ante and David Helgason created the first version of what would become one of the most widespread game engines. The three set out to create software that cared for developers’ needs, and with every new version that they launched, the game engine got increasingly better.

Today, Unity is used worldwide and is one of the most popular cross-platform game engines out there. Its community embodies over 2.5 million registered developers, and is a place where both hobbyists and experienced users can share ideas, ask for advice or browse through documentation.

To better understand why Unity is so popular in the game development world, let’s elaborate on some of the main ingredients that contribute to its notoriety.

**1. Multiple platforms**

Although it was first released only for Mac OS X, Unity is now supporting over 25 different platforms, which enable developers to make their game more accessible for the public. Creators have the possibility to deploy on all popular platforms and devices, from Android and iOS to consoles and web plugins, everyone can download their creation, no matter the gadget that they own.

**2. 2D, 3D, VR and AR**

Unity gives developers the opportunity to create not only 3D games, but, since November 2013, specialists can design 2D games as well. And as far as mobile gaming goes, Unity dominates more than half of the market, as creators continue to rely on this game engine to build the most successful games. Unity holds the supremacy for AR and VR as well, with over 60% of the developed content.

**3. Cost**

Another great thing about Unity is that it allows anyone that is interested in game creation, to download a free version. Small, independent developers are encouraged to pursue their vision and are provided with various tools and techniques to help make their ideas come to life. For those who are looking for extra features, there is also a Pro Business plan available, that goes for 150 dollars a month, and includes priority customer service.

**4. Great Graphics**

When it comes to graphics, Unity is known for its high quality, next-level visual effects. The highly customizable rendering technology that the game engine offers, along with a variety of intuitive tools, facilitate the creation of fantastic looking games. In 3D games, developers can generate natural, smooth movements of objects and there are also lots of tutorials to choose from that explain how to do it all.

**5. Unity Asset Store**

Launched in November 2010, Unity’s Asset Store makes game development even easier and more cost-effective. And creators have a variety of assets to choose from; whether they’re looking for specific sounds, 3D structures, patterns, textures or animations, developers can use them to make their vision come alive. These assets are available for both free and paid versions and, furthermore, designers can sell their assets and collect 70 percent of the revenue.

**6. Less coding**

The fact that you don’t need to have a great deal of coding knowledge or experience, is really convenient for those who are at the beginning of their game development career or are just looking to start a hobby. Unity uses C#, JavaScript and Boo, but you can also develop a game from start to finish without writing a single line of code. At the same time, for those who do have coding skills, learning programmes and documentation are available, and can come in handy.

**7. Strong community**

Having such a large community, Unity developers can ask questions and find solutions to their issues by asking other members, all this in a friendly and supportive environment. Moreover, Unity engineers meet with community members annually, in an organised event named after the game engine. During this conference, developers share experiences, exchange ideas, and connect with each other, consolidating the community.

**6.2 Microsoft Visual Studio**

**6.2.1 What is Microsoft Visual Studio?**

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that expand the functionality at almost every level—including adding support for source control systems (like Subversion and Git) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Azure DevOps client: Team Explorer).

Visual Studio supports 36 different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, C++/CLI, Visual Basic .NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML, and CSS. Support for other languages such as Python, Ruby, Node.js, and M among others is available via plug-ins. Java (and J#) were supported in the past.

The most basic edition of Visual Studio, the Community edition, is available free of charge. The slogan for Visual Studio Community edition is "Free, fully-featured IDE for students, open-source and individual developers".

As of 8 November 2021 the current production-ready Visual Studio version is 2022, with older versions such as 2013 and 2015 on Extended Support, and 2017 and 2019 on Mainstream Support.

**6.2.2 Architecture**

Visual Studio does not support any programming language, solution or tool intrinsically; instead, it allows the plugging of functionality coded as a VSPackage. When installed, the functionality is available as a Service. The IDE provides three services: SVsSolution, which provides the ability to enumerate projects and solutions; SVsUIShell, which provides windowing and UI functionality (including tabs, toolbars, and tool windows); and SVsShell, which deals with registration of VSPackages. In addition, the IDE is also responsible for coordinating and enabling communication between services. All editors, designers, project types and other tools are implemented as VSPackages. Visual Studio uses COM to access the VSPackages. The Visual Studio SDK also includes the Managed Package Framework (MPF), which is a set of managed wrappers around the COM-interfaces that allow the Packages to be written in any CLI compliant language. However, MPF does not provide all the functionality exposed by the Visual Studio COM interfaces. The services can then be consumed for creation of other packages, which add functionality to the Visual Studio IDE.

Support for programming languages is added by using a specific VSPackage called a Language Service. A language service defines various interfaces which the VSPackage implementation can implement to add support for various functionalities. Functionalities that can be added this way include syntax coloring, statement completion, brace matching, parameter information tooltips, member lists, and error markers for background compilation.[15] If the interface is implemented, the functionality will be available for the language. Language services are implemented on a per-language basis. The implementations can reuse code from the parser or the compiler for the language. Language services can be implemented either in native code or managed code. For native code, either the native COM interfaces or the Babel Framework (part of Visual Studio SDK) can be used. For managed code, the MPF includes wrappers for writing managed language services.

Visual Studio does not include any source control support built in but it defines two alternative ways for source control systems to integrate with the IDE. A Source Control VSPackage can provide its own customised user interface. In contrast, a source control plugin using the MSSCCI (Microsoft Source Code Control Interface) provides a set of functions that are used to implement various source control functionality, with a standard Visual Studio user interface. MSSCCI was first used to integrate Visual SourceSafe with Visual Studio 6.0 but was later opened up via the Visual Studio SDK. Visual Studio .NET 2002 used MSSCCI 1.1, and Visual Studio .NET 2003 used MSSCCI 1.2. Visual Studio 2005, 2008, and 2010 use MSSCCI Version 1.3, which adds support for rename and delete propagation, as well as asynchronous opening.

Visual Studio supports running multiple instances of the environment (each with its own set of VSPackages). The instances use different registry hives (see MSDN's definition of the term "registry hive" in the sense used here) to store their configuration state and are differentiated by their AppId (Application ID). The instances are launched by an AppId-specific .exe that selects the AppId, sets the root hive, and launches the IDE. VSPackages registered for one AppId are integrated with other VSPackages for that AppId. The various product editions of Visual Studio are created using the different AppIds. The Visual Studio Express edition products are installed with their own AppIds, but the Standard, Professional, and Team Suite products share the same AppId. Consequently, one can install the Express editions side-by-side with other editions, unlike the other editions which update the same installation. The professional edition includes a superset of the VSPackages in the standard edition, and the team suite includes a superset of the VSPackages in both other editions. The AppId system is leveraged by the Visual Studio Shell in Visual Studio 2008.

**6.2.3 Features**

**1. Code editor**

Visual Studio (like any other IDE) includes a code editor that supports syntax highlighting and code completion using IntelliSense for variables, functions, methods, loops, and LINQ queries. IntelliSense is supported for the included languages, as well as for XML, Cascading Style Sheets, and JavaScript when developing web sites and web applications. Autocomplete suggestions appear in a modeless list box over the code editor window, in proximity of the editing cursor. In Visual Studio 2008 onwards, it can be made temporarily semi-transparent to see the code obstructed by it. The code editor is used for all supported languages.

The Visual Studio Code Editor also supports setting bookmarks in code for quick navigation. Other navigational aids include collapsing code blocks and incremental search, in addition to normal text search and regex search. The code editor also includes a multi-item clipboard and a task list. The code editor supports code snippets, which are saved templates for repetitive code and can be inserted into code and customized for the project being worked on. A management tool for code snippets is built in as well. These tools are surfaced as floating windows which can be set to automatically hide when unused or docked to the side of the screen. The Visual Studio code editor also supports code refactoring including parameter reordering, variable and method renaming, interface extraction, and encapsulation of class members inside properties, among others.

**2. Debugger**

Visual Studio includes a debugger that works both as a source-level debugger and as a machine-level debugger. It works with both managed code as well as native code and can be used for debugging applications written in any language supported by Visual Studio. In addition, it can also attach to running processes, monitor, and debug those processes. If source code for the running process is available, it displays the code as it is being run. If source code is not available, it can show the disassembly. The Visual Studio debugger can also create memory dumps as well as load them later for debugging. Multi-threaded programs are also supported. The debugger can be configured to be launched when an application running outside the Visual Studio environment crashes.

The Visual Studio Debugger allows setting breakpoints (which allow execution to be stopped temporarily at a certain position) and watches (which monitor the values of variables as the execution progresses). Breakpoints can be conditional, meaning they get triggered when the condition is met. Code can be stepped over, i.e., run one line (of source code) at a time. It can either step into functions to debug inside it, or step over it, i.e., the execution of the function body isn't available for manual inspection. The debugger supports Edit and Continue, i.e., it allows code to be edited as it is being debugged. When debugging, if the mouse pointer hovers over any variable, its current value is displayed in a tooltip ("data tooltips"), where it can also be modified if desired. During coding, the Visual Studio debugger lets certain functions be invoked manually from the Immediate tool window. The parameters to the method are supplied at the Immediate window.

**3. Designer**

Visual Studio includes a host of visual designers to aid in the development of applications. These tools include:

**1. Windows Forms Designer**

The Windows Forms designer is used to build GUI applications using Windows Forms. Layout can be controlled by housing the controls inside other containers or locking them to the side of the form. Controls that display data (like textbox, list box and grid view) can be bound to data sources like databases or queries. Data-bound controls can be created by dragging items from the Data Sources window onto a design surface. The UI is linked with code using an event-driven programming model. The designer generates either C# or VB.NET code for the application.

**2. WPF Designer**

The WPF designer, codenamed Cider, was introduced with Visual Studio 2008. Like the Windows Forms designer it supports the drag and drop metaphor. It is used to author user interfaces targeting Windows Presentation Foundation. It supports all WPF functionality including data binding and automatic layout management. It generates XAML code for the UI. The generated XAML file is compatible with Microsoft Expression Design, the designer-oriented product. The XAML code is linked with code using a code-behind model.

**3. Web designer/development**

Visual Studio also includes a web-site editor and designer that allows web pages to be authored by dragging and dropping widgets. It is used for developing ASP.NET applications and supports HTML, CSS and JavaScript. It uses a code-behind model to link with ASP.NET code. From Visual Studio 2008 onwards, the layout engine used by the web designer is shared with the discontinued Expression Web. There is also ASP.NET MVC support for MVC technology as a separate download and ASP.NET Dynamic Data project available from Microsoft.

**4. Class designer**

The Class Designer is used to author and edit the classes (including its members and their access) using UML modeling. The Class Designer can generate C# and VB.NET code outlines for the classes and methods. It can also generate class diagrams from hand-written classes.

**5. Data designer**

The data designer can be used to graphically edit database schemas, including typed tables, primary and foreign keys and constraints. It can also be used to design queries from the graphical view.

**6. Mapping designer**

From Visual Studio 2008 onwards, the mapping designer is used by LINQ to SQL to design the mapping between database schemas and the classes that encapsulate the data. The new solution from ORM approach, ADO.NET Entity Framework, replaces and improves the old technology.

**4. Extensibility**

Visual Studio allows developers to write extensions for Visual Studio to extend its capabilities. These extensions "plug into" Visual Studio and extend its functionality. Extensions come in the form of macros, add-ins, and packages. Macros represent repeatable tasks and actions that developers can record programmatically for saving, replaying, and distributing. Macros, however, cannot implement new commands or create tool windows. They are written using Visual Basic and are not compiled. Add-Ins provide access to the Visual Studio object model and can interact with the IDE tools. Add-Ins can be used to implement new functionality and can add new tool windows. Add-Ins are plugged into the IDE via COM and can be created in any COM-compliant languages. Packages are created using the Visual Studio SDK and provide the highest level of extensibility. They can create designers and other tools, as well as integrate other programming languages. The Visual Studio SDK provides unmanaged APIs as well as a managed API to accomplish these tasks. However, the managed API isn't as comprehensive as the unmanaged one. Extensions are supported in the Standard (and higher) versions of Visual Studio 2005. Express Editions do not support hosting extensions.

Visual Studio 2008 introduced the Visual Studio Shell that allows for development of a customized version of the IDE. The Visual Studio Shell defines a set of VSPackages that provide the functionality required in any IDE. On top of that, other packages can be added to customize the installation. The Isolated mode of the shell creates a new AppId where the packages are installed. These are to be started with a different executable. It is aimed for development of custom development environments, either for a specific language or a specific scenario. The Integrated mode installs the packages into the AppId of the Professional/Standard/Team System editions, so that the tools integrate into these editions. The Visual Studio Shell is available as a free download.

After the release of Visual Studio 2008, Microsoft created the Visual Studio Gallery. It serves as the central location for posting information about extensions to Visual Studio. Community developers as well as commercial developers can upload information about their extensions to Visual Studio .NET 2002 through Visual Studio 2010. Users of the site can rate and review the extensions to help assess the quality of extensions being posted. An extension is stored in a VSIX file. Internally a VSIX file is a ZIP file that contains some XML files, and possibly one or more DLL's. One of the main advantages of these extensions is that they do not require Administrator rights to be installed. RSS feeds to notify users on updates to the site and tagging features are also planned.

**6.2.4 Game development with Visual Studio**

The Unity engine integrates into one unparalleled platform to create 2D and 3D games and interactive content. Create once and publish to 21 platforms, including all mobile platforms, WebGL, Mac, PC, and Linux desktop, web, or consoles. Use powerful cross-platform tools to make your interactive content run beautifully on any device.

**1. Debug in Visual Studio**

World-class debugging for Unity games in Visual Studio

Visual Studio brings a premium debugging experience to the Unity game engine. Identify issues quickly by debugging your Unity games in Visual Studio—set breakpoints and evaluate variables and complex expressions. You can debug your Unity game that is running either in the Unity Editor or in the Unity Player, or even debug an external managed DLL in your Unity Project.

**2. Enhance productivity**

Full featured IDE for Unity........

Write code more productively by leveraging all the productivity features that Visual Studio has to offer, such as IntelliSense, refactoring, and code browsing capabilities. Customize the coding environment exactly the way you want–choose your favorite theme, color, fonts, and all the other settings. In addition, use the Unity Project Explorer to navigate and create your Unity scripts–no more switching back and forth between multiple IDEs. Create Unity script methods inside Visual Studio quickly by using the Implement MonoBehaviours and the Quick MonoBehaviours Wizards.

**6.3 Blender Software**

**What is Blender Software?**

Blender is a free and open-source 3D computer graphics software tool set used for creating animated films, visual effects, art, 3D-printed models, motion graphics, interactive 3D applications, virtual reality, and, formerly, video games. Blender's features include 3D modelling, UV mapping, texturing, digital drawing, raster graphics editing, rigging and skinning, fluid and smoke simulation, particle simulation, soft body simulation, sculpting, animation, match moving, rendering, motion graphics, video editing, and compositing.

**6.3.1 Features**

**1. Modeling**

Forensic facial reconstruction of a mummy by Cícero Moraes

Blender has support for a variety of geometric primitives, including polygon meshes, Bézier curves, NURBS surfaces, metaballs, icospheres, text, and an n-gon modeling system called B-mesh. There is also an advanced polygonal modelling system which can be accessed through an edit mode. It supports features such as extrusion, bevelling, and subdividing.

**1.1 Modifiers**

Modifiers apply non-destructive effects which can be applied upon rendering or exporting, such as subdivision surfaces.

**1.2 Sculpting**

Blender has multi-resolution digital sculpting, which includes dynamic topology, "baking", remeshing, re-symmetrization, and decimation. The latter is used to simplify models for exporting purposes (an example being game assets).

**2. Animation**

Blender's keyframed animation capabilities include inverse kinematics, armatures, hooks, curve- and lattice-based deformations, shape keys, non-linear animation, constraints, and vertex weighting. In addition, its Grease Pencil tools allow for 2D animation within a full 3D pipeline.

**3. Rendering**

Rendering of different materials using the Cycles render engine.

Internal render engine with scanline rendering, indirect lighting, and ambient occlusion that can export in a wide variety of formats; A path tracer render engine called Cycles, which can take advantage of the GPU for rendering. Cycles supports the Open Shading Language since Blender 2.65.

Cycles Hybrid Rendering is possible in Version 2.92 with Optix. Tiles are calculated with GPU in combination with cpu.

EEVEE is a new physically based real-time renderer. It works both as a renderer for final frames, and as the engine driving Blender's real-time viewport for creating assets.

**3.1 Texturing and shading**

Blender allows procedural and node-based textures, as well as texture painting, projective painting, vertex painting, weight painting and dynamic painting.

**Why use Blender Software?**

* Its open-source software
* Blender is always free
* No registration or license key is required to use
* You can change the source code however you want
* There is always more than 1 way to perform a task in Blender
* You can import and export many different file formats
* Not only can you animate in 3D but 2D as well
* Almost every creative task can be performed using one software application

**1. Do Almost Everything You Need To On A Single Application**

Many applications are built to perform a specific type of task or a small selection of tasks that are similar to each other. With Blender 3D you have access to almost all the tools that you could require to create what you want in a single application.

For example, you can follow the entire workflow of creating a character model in Blender. With the grease pencil, you can create 2D concept art of your soon-to-be 3D model. Then you can begin the actual modeling stage of the project in the 3D viewport, able to create your model in a variety of ways.

You can use modeling methods in Blender to create the base shape, and then use the sculpting toolkit to create the finer details of your model. Then you can move on to the next stage of the project and add materials and textures to your character and the character clothes with blenders, UV mapping tools and a superb node system for material application.

Then you can move onto the rigging stage of the process where you can prep your character to move in ways natural to humans and prep it for animation, which becomes the next step in the process where you can create animated motions for the character model.

And all of this you are able to do inside of Blender without the need for any other software. If you are following a pure animation workflow your entire project can easily be done in Blender without the need for any other software.

In fact, you can even edit the rendered animations that you created by using the applications video editing suite. Yes, it even has one of those! and you can add effects, filters, and overlays to your animations using Blender node compositor.

This does not mean that Blender cannot do everything, however, and there are some things that are currently beyond its reach.

You cannot 3D print models for example directly from Blender, as it does not have built-in splicing software. Although it can be used to create the models used for 3D printing. Likewise, it cannot be used to create video games but can be used to create all the assets for those games.

That’s why we say Blender can do almost everything, as it can perform many tasks normally performed by various software and reduce the number of applications that you may need to learn.

**2. Shape Blender How You Want With Access To The Source Code**

The reason why Blender is referred to as open-source is that you have full access to the application, including the source code that is used to run the software. If you go to the download page of the blender website, you will be able to access a menu just below the blue download button.

In this menu, you can choose how to install your Blender software, including the operating system that you are downloading the software to as well as if you want to download from a third-party marketplace like steam. And on this list is the option to download the source code along with the Blender libraries.

If you are a programmer, and are able to use Python script, then downloading the source code will allow you to change your downloaded version of Blender however you see fit.

This goes far beyond what you are able to do in the text editor in Blender itself, which is used to improve existing functionality through scripts that may also become add-ons in the future. With access to the source code, you can change anything that you want, from the UI to the way the node editor works, anything is possible.

**Chapter 7: Testing**

**7.1 Testing User Interface**

GUI is what the user sees, and a normal User first observes the design and looks of the Application/Software and how easy it is for him to understand the UI. If a user is not comfortable with the Interface or finds an Application complex to understand he would never use that Application Again. That’s why, GUI is a matter for concern, and proper testing should be carried out in order to eliminate the presence of defects in Graphical user interface (GUI).

In GUI Testing we basically test the visual elements to verify that they are functioning according to requirements in terms of functionality and performance. UI testing ensures that UI functions are bug-free.

**7.1.1 GUI Testing - Approaches**

**1. Manual Based Testing**

In this case, a tester manually uses all the features of the website or app and the requirements stated in the business requirements document to check for any discrepancies. This makes sense when the software has a limited number of UI elements, which is usually the case in initial versions of a website or app. However, given the tech-savvy user base of our times, most expect software with rich, layered user interfaces with hundreds, perhaps thousands of UI elements that require verification.

This makes manual testing inefficient, time-consuming, and prone to human error.

**Capture and Replay**

GUI testing can be done using automation tools. This is done in 2 parts. During Record, test steps are captured by the automation tool. During playback, the recorded test steps are executed on the Application Under Test. Example of such tools – QTP.

**Model-Based Testing**

A model is a graphical description of a system’s behavior. It helps us to understand and predict the system behavior. Models help in a generation of efficient test cases using the system requirements. The following needs to be considered for this model based testing:

1. Build the model
2. Determine Inputs for the model
3. Calculate the expected output for the model
4. Run the tests
5. Compare the actual output with the expected output
6. A decision on further action on the model

Some of the modeling techniques from which test cases can be derived:

Charts – Depicts the state of a system and checks the state after some input.

Decision Tables – Tables used to determine results for each input applied

Model based testing is an evolving technique for generating test cases from the requirements. Its main advantage, compared to above two methods, is that it can determine undesirable states that your GUI can attain.

**7.1.2 The Scope of GUI Testing**

Our GUI Testing covers the gamut of visual indicators and graphic-based icons, fonts, menus, text boxes in home and registration pages and buttons, characters movements, Themes, scoring system and more in the game.

Some of the features included in GUI test suites include:

**1. Functionality**

The purpose of UI is to allow a child to complete a task and it was our priority to create a UI with usability and functionality in mind to make each task easier for the ADHD child. The kid perception of a UI is directly linked to how functional it is, it should be easy for kids to find information and the UI should not have a steep learning curve. It should allow these childrens to pick up on what is being made available to them through the use of the game and use as much common knowledge as a basis for how they will interpret features. A developer can make a UI design more functional by adding elements of personalisation and using proper layouts, mixed with color and shaped designs to make the UI feel better for ADHD kids to interact with. A player’s perception of the information available to them is mostly influenced by the usefulness and design characteristics of the user interface. UIs tools and functions are directly linked to a user’s perception of what information is available to them and how they are to access or assess said information. The UI’s usability and functionality should allow for ease of use for all children of all the ADHD types as possible and we attempt to keep the learning curve as low as possible for them.

**2. Visual Design**

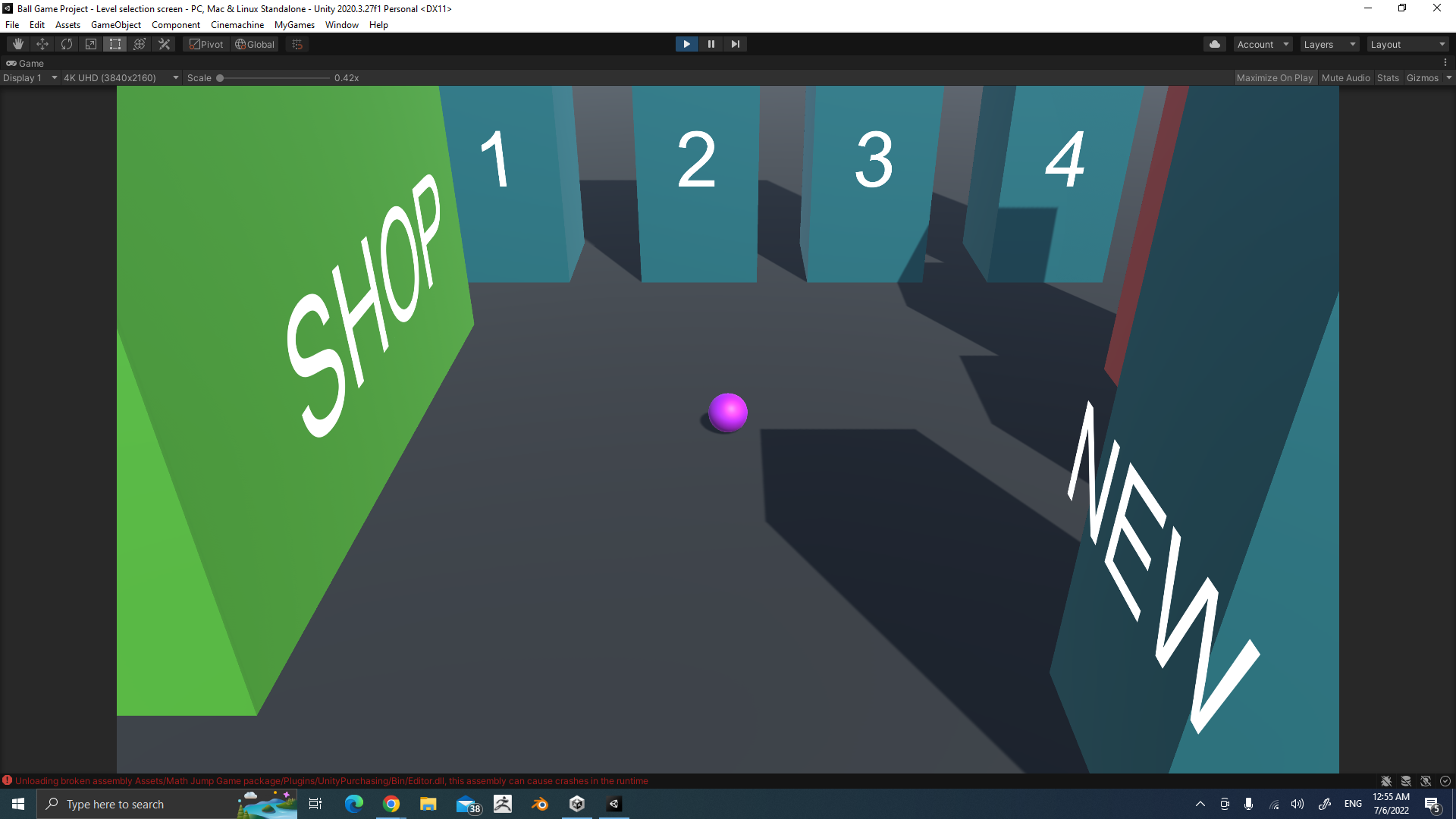
A modern UI should be created with accessibility functions in mind and make its use feel natural to a user. Modern UI can be created with sensors, camera tracking and virtual reality devices in mind and the purpose of a UI is to allow kids to perform tasks in a virtual space. A successful UI should make the action feel as natural to a child as possible through movement and speech control to make him/her feel reality and respond more for learning and focusing. The UI must take advantage of ADHD kids situational awareness and allow them to use real-world knowledge to interact with the virtual world. Our goal is to make a UI that can be understood very easily and allow as many kids as possible to have good situational awareness in the context of the game.

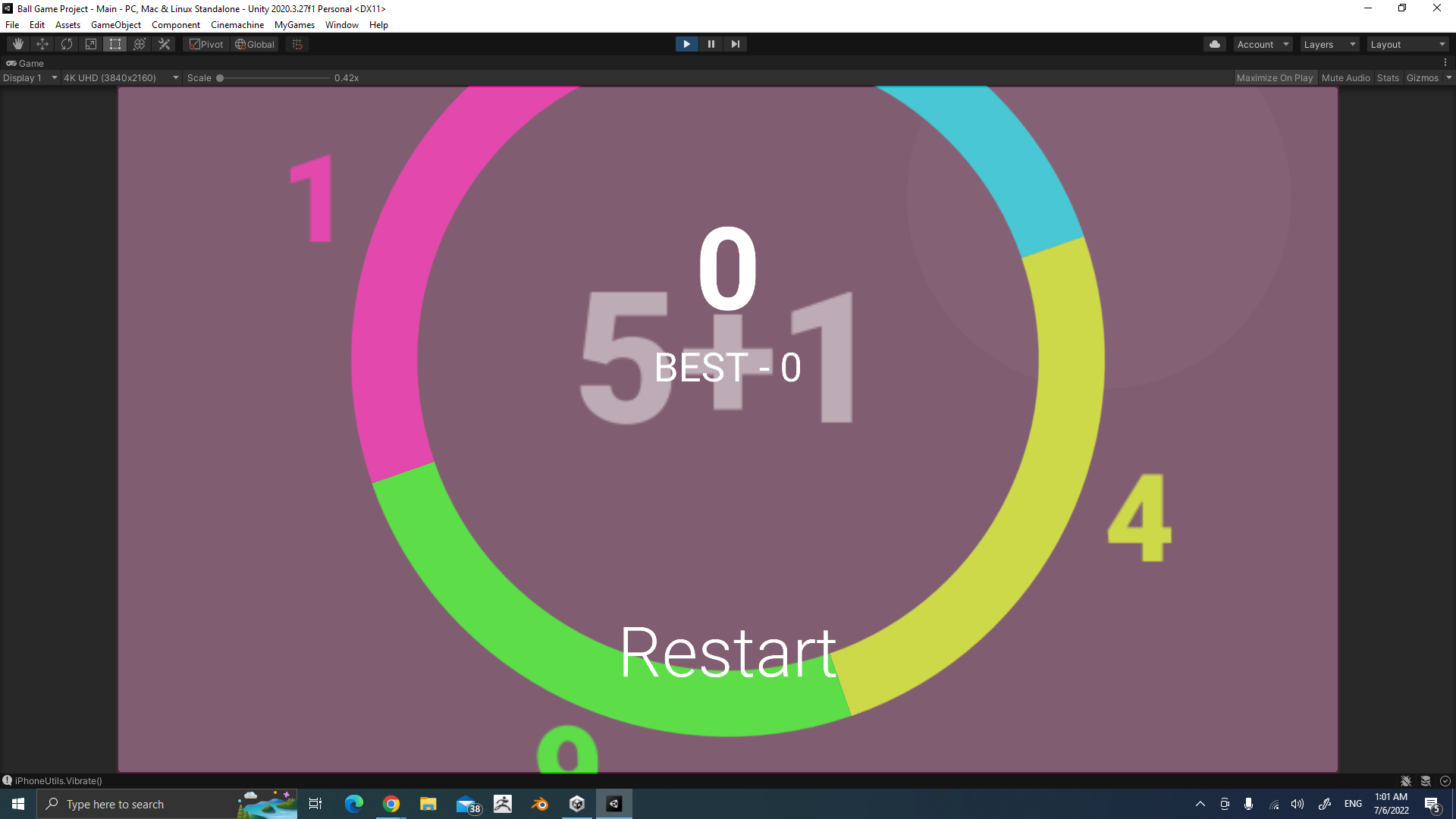
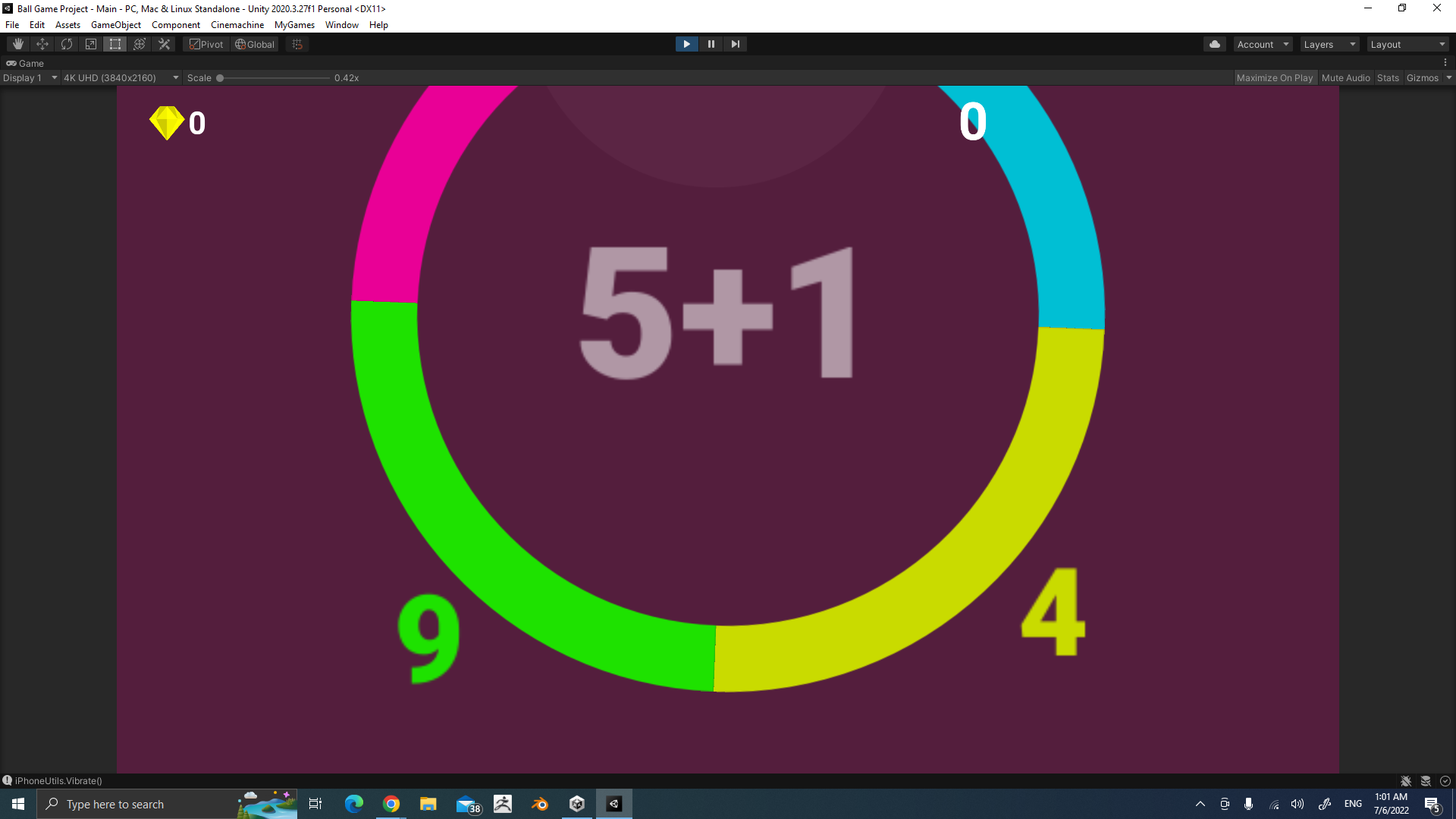
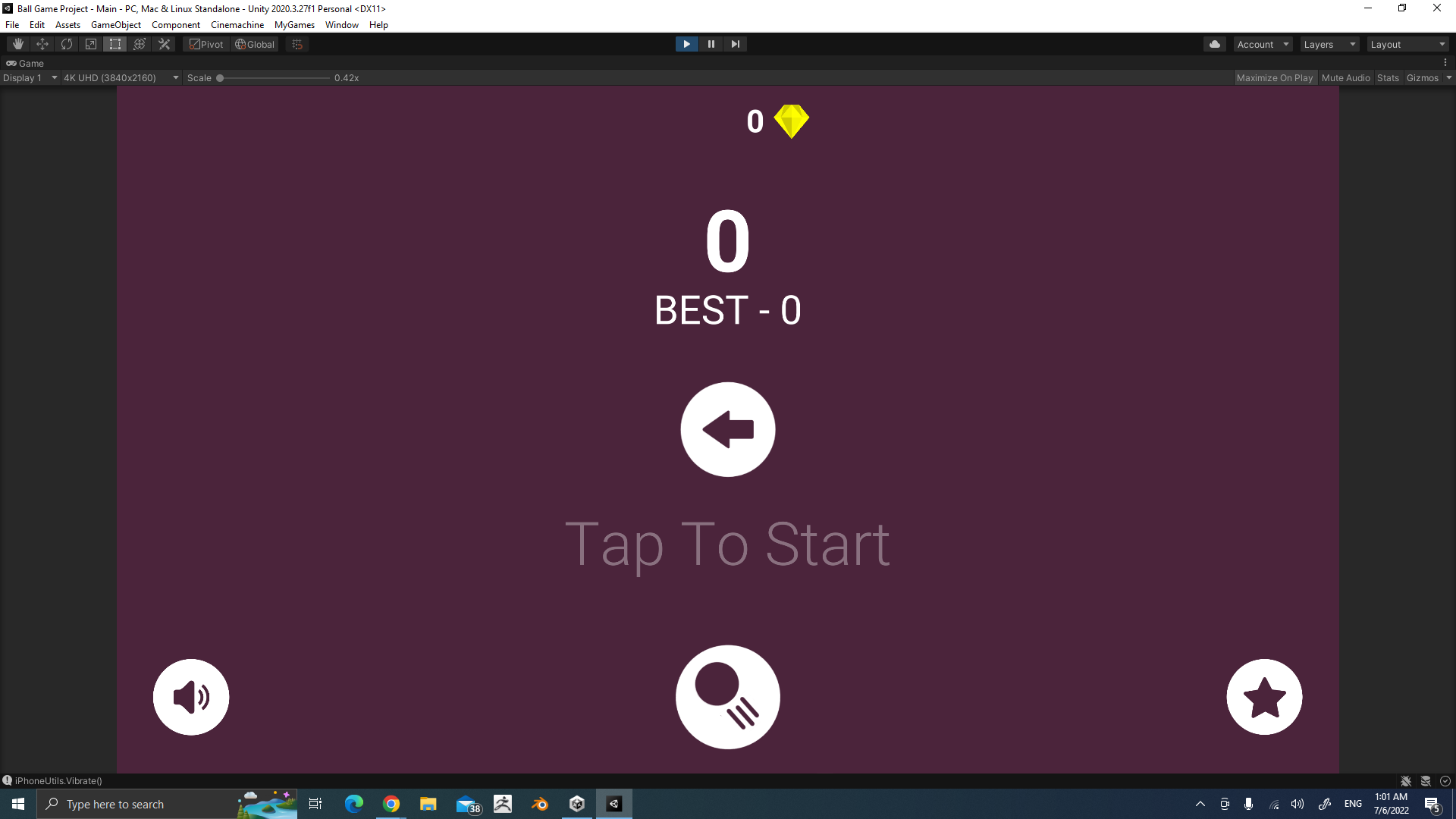
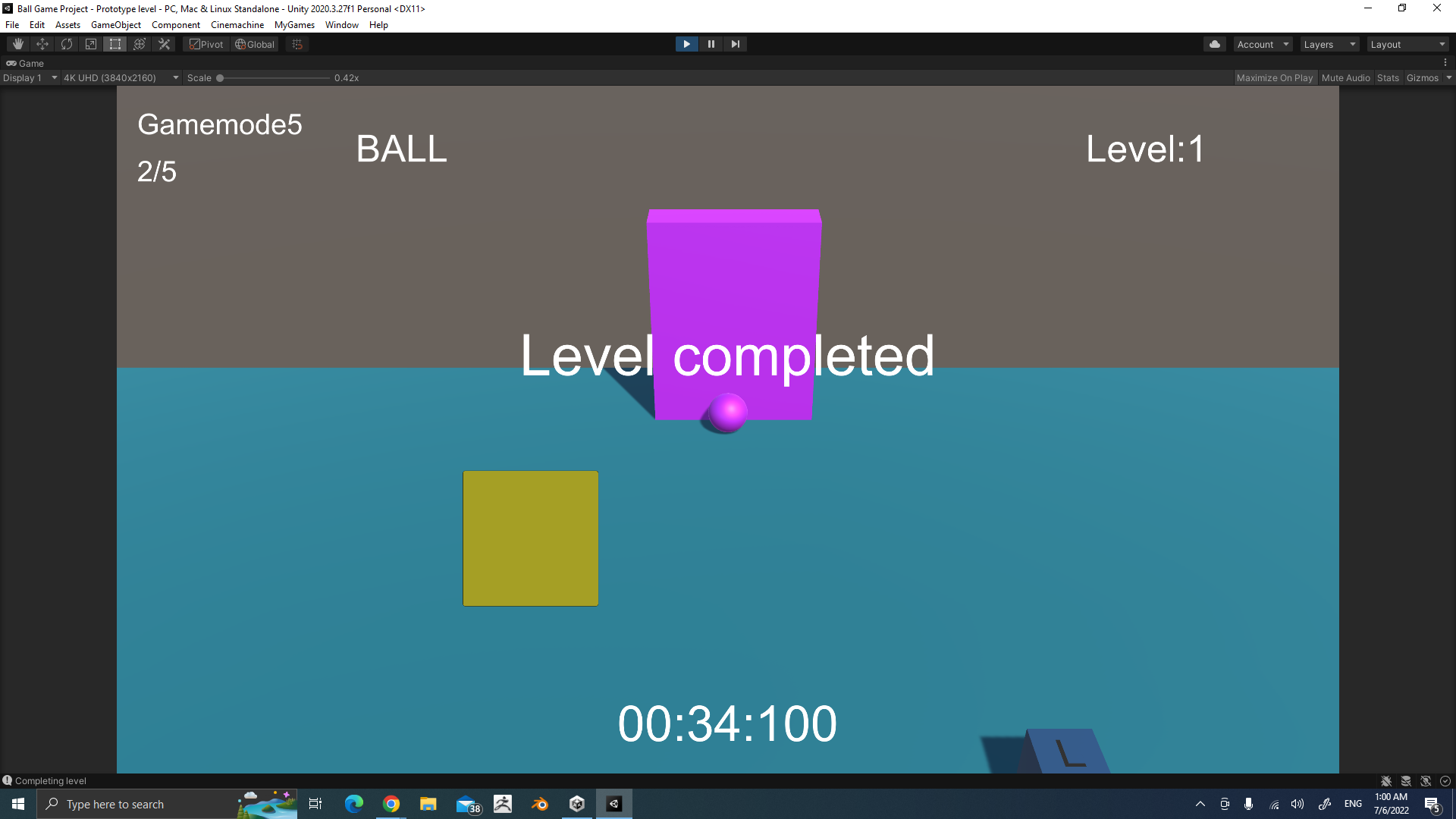
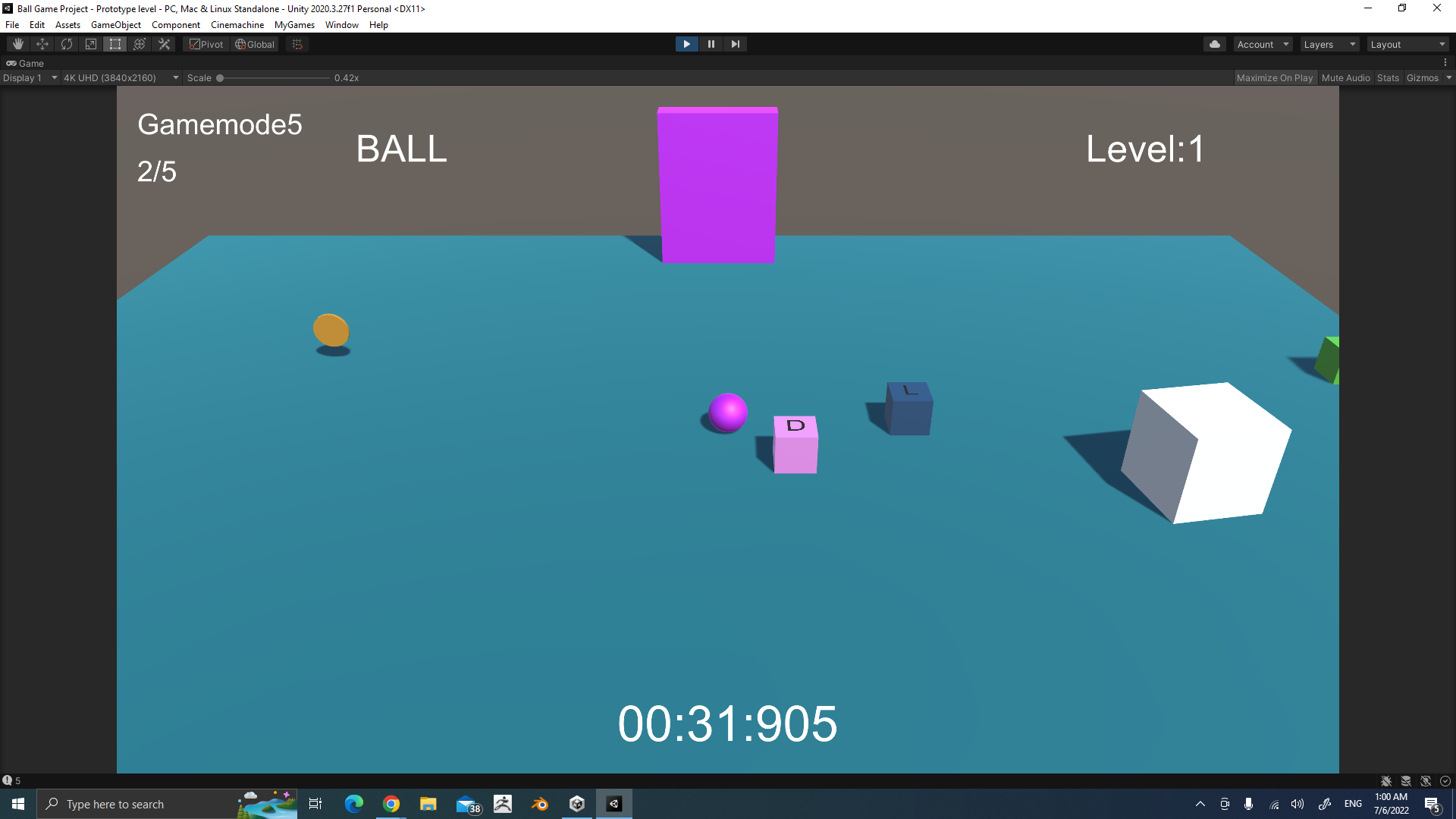
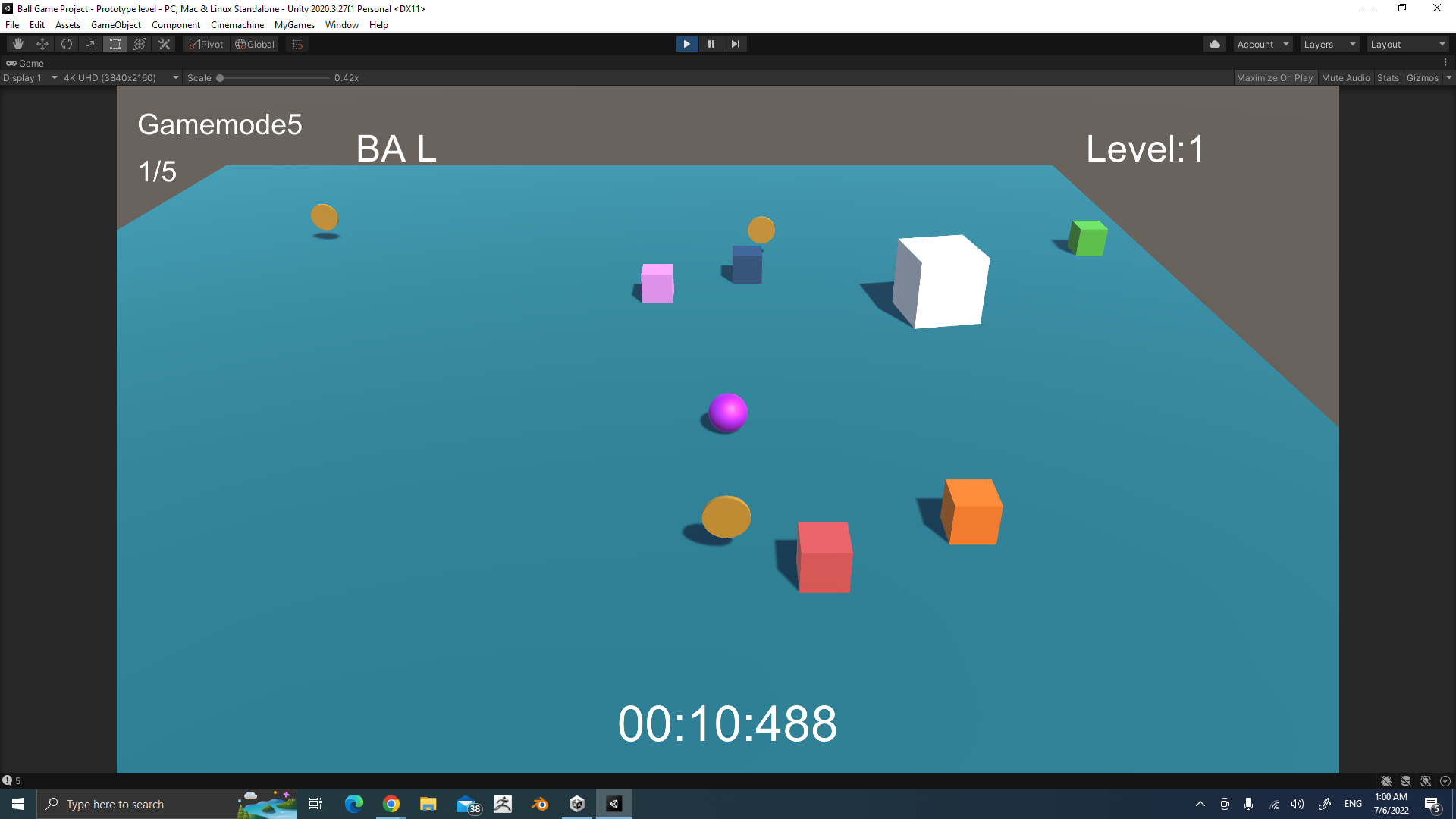
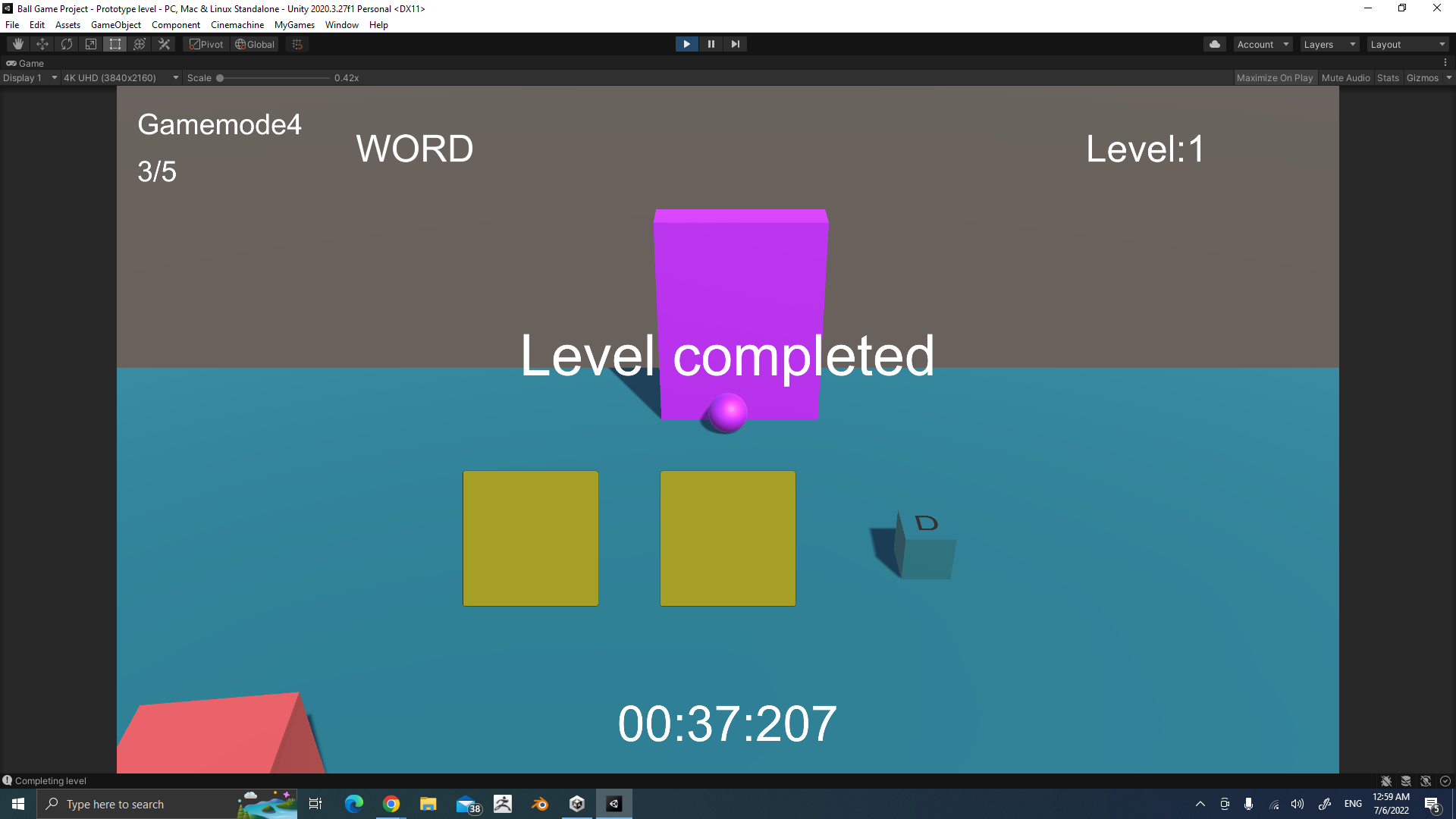
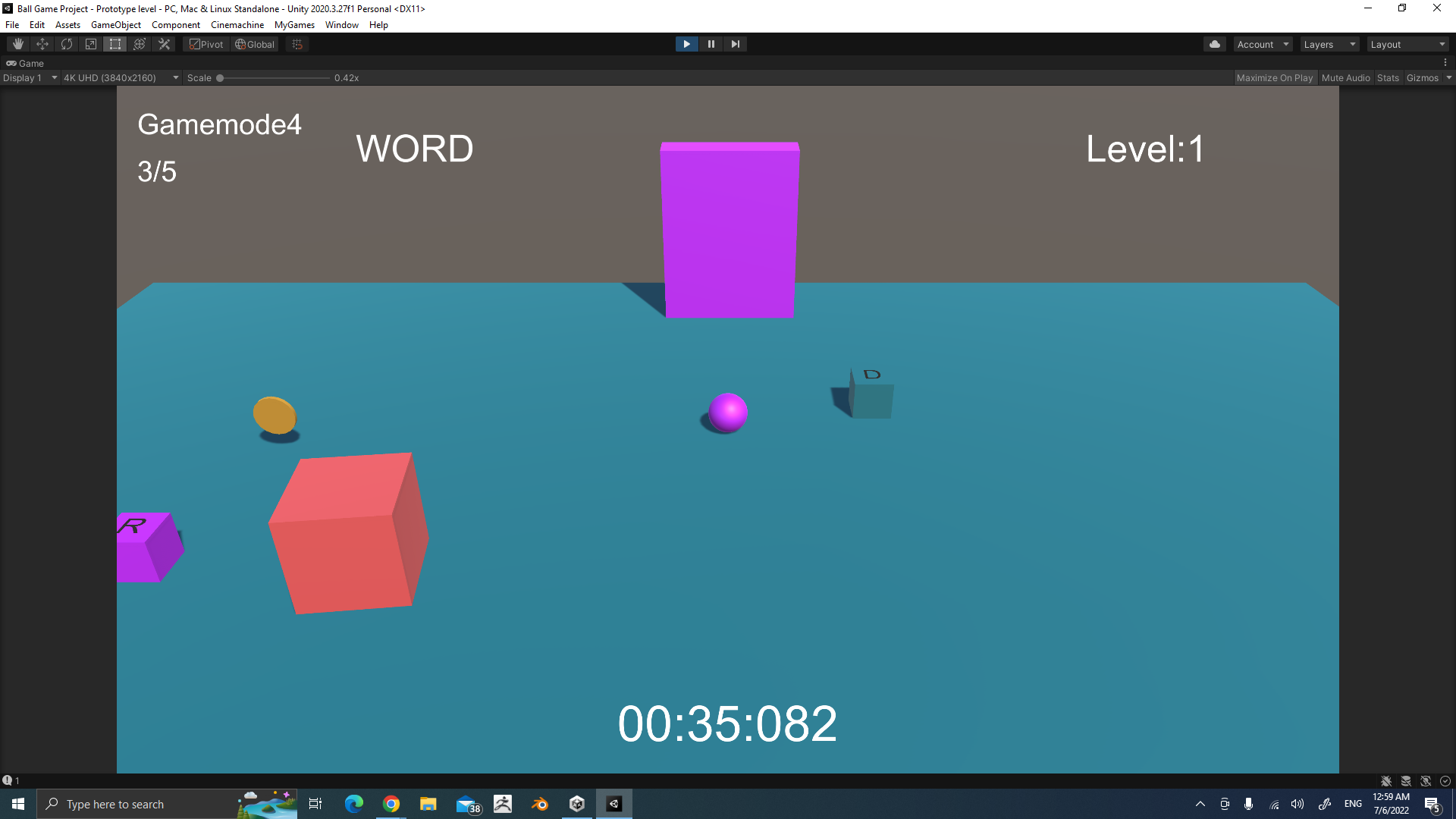
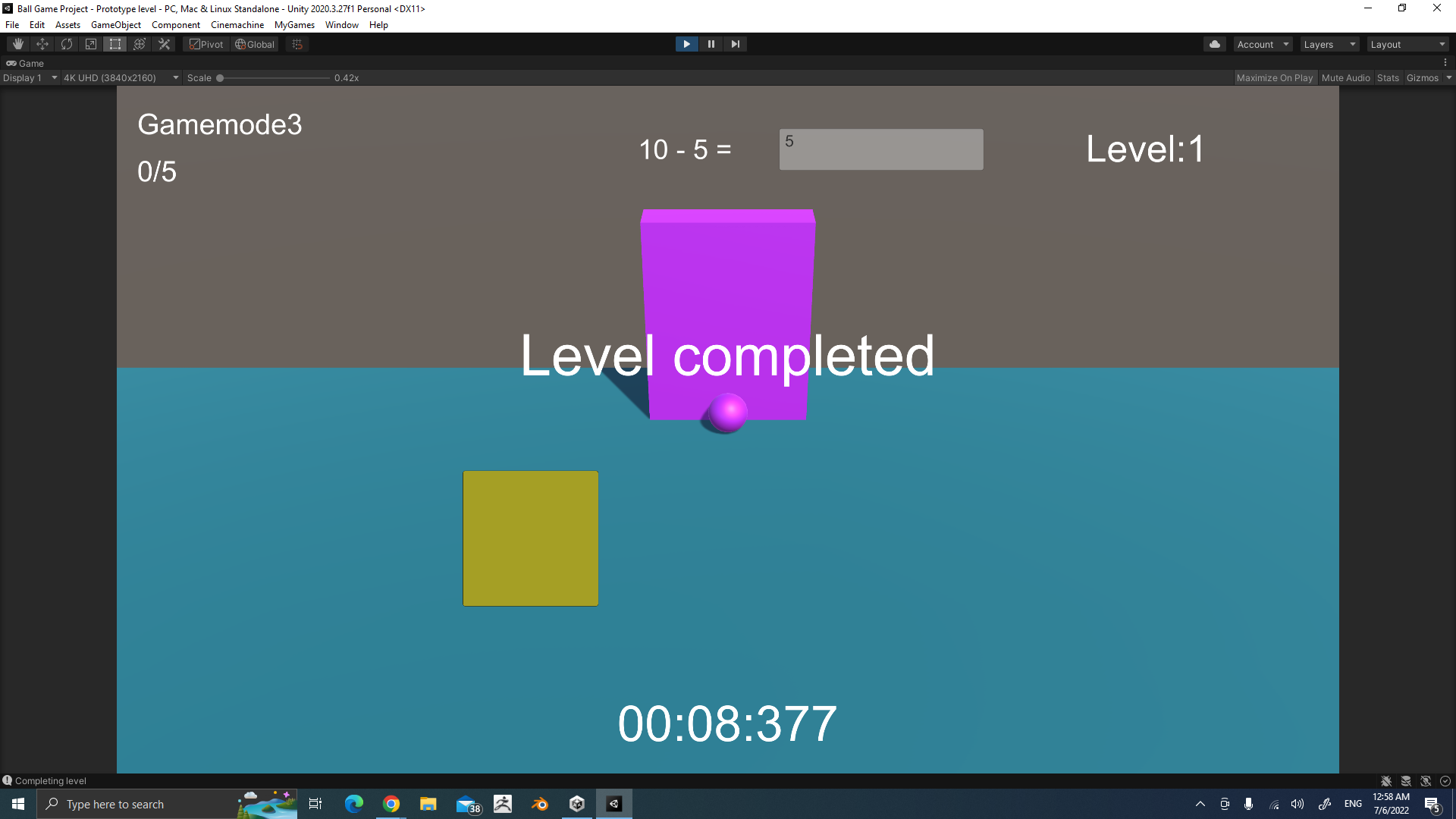
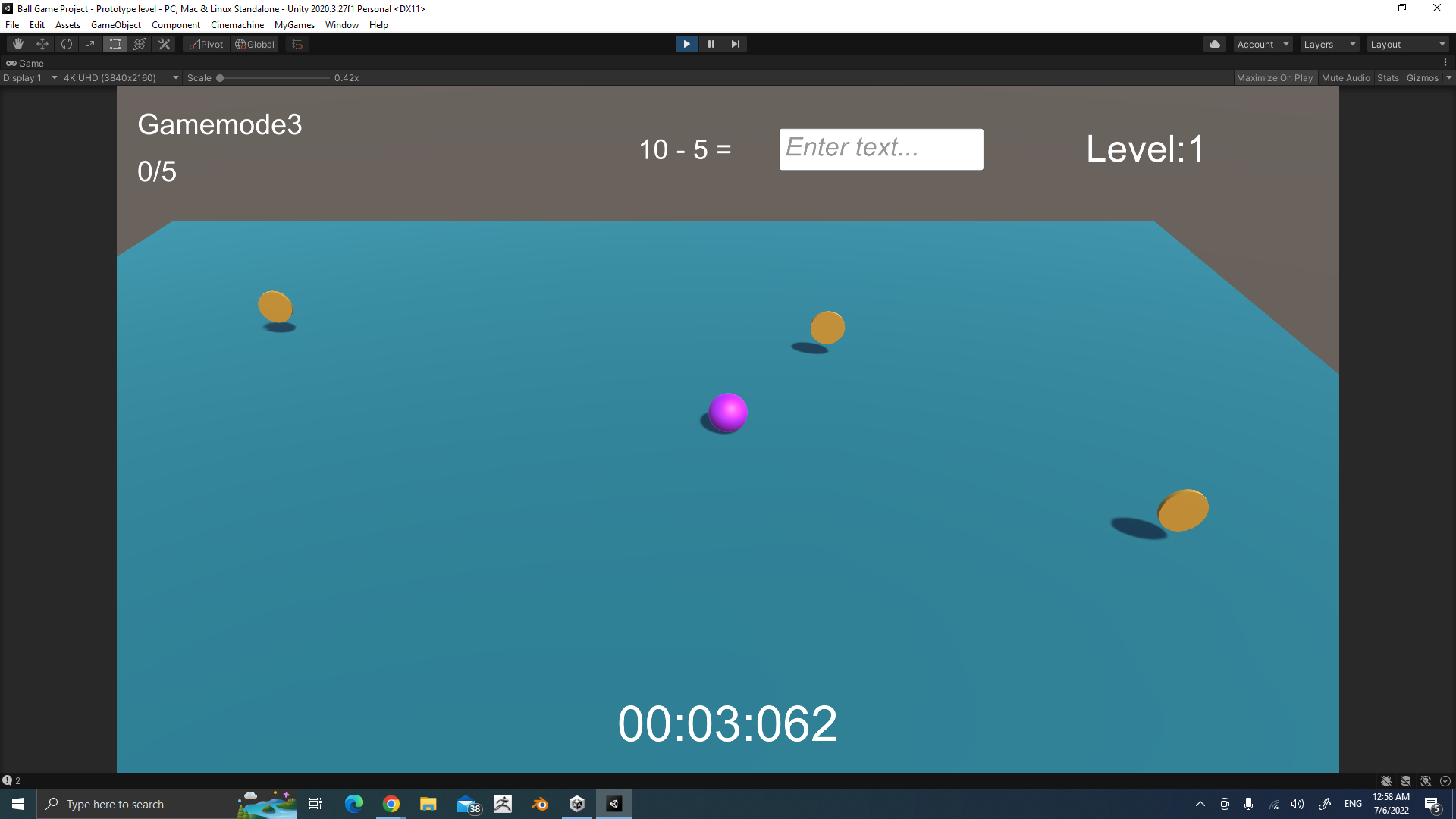
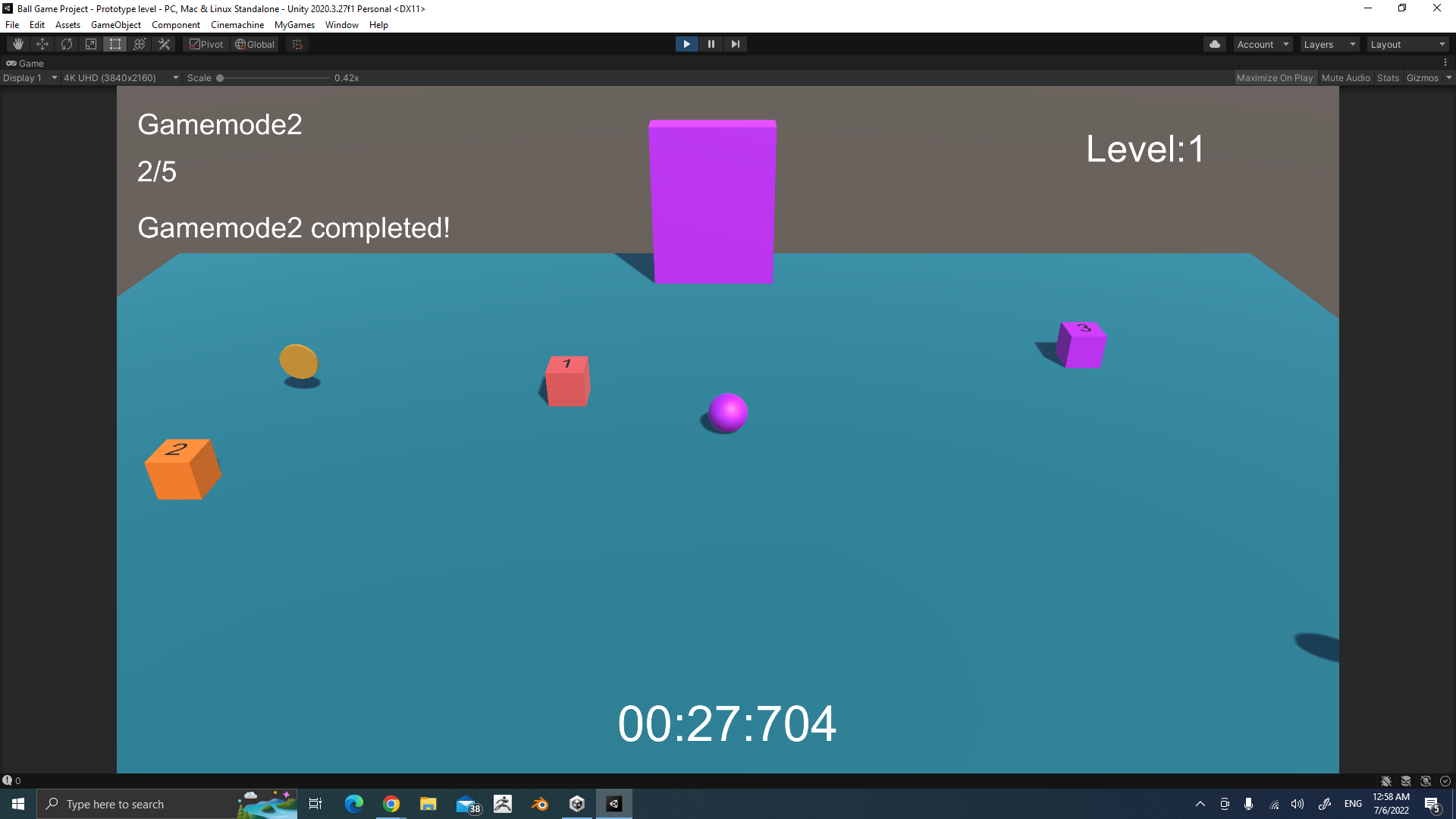
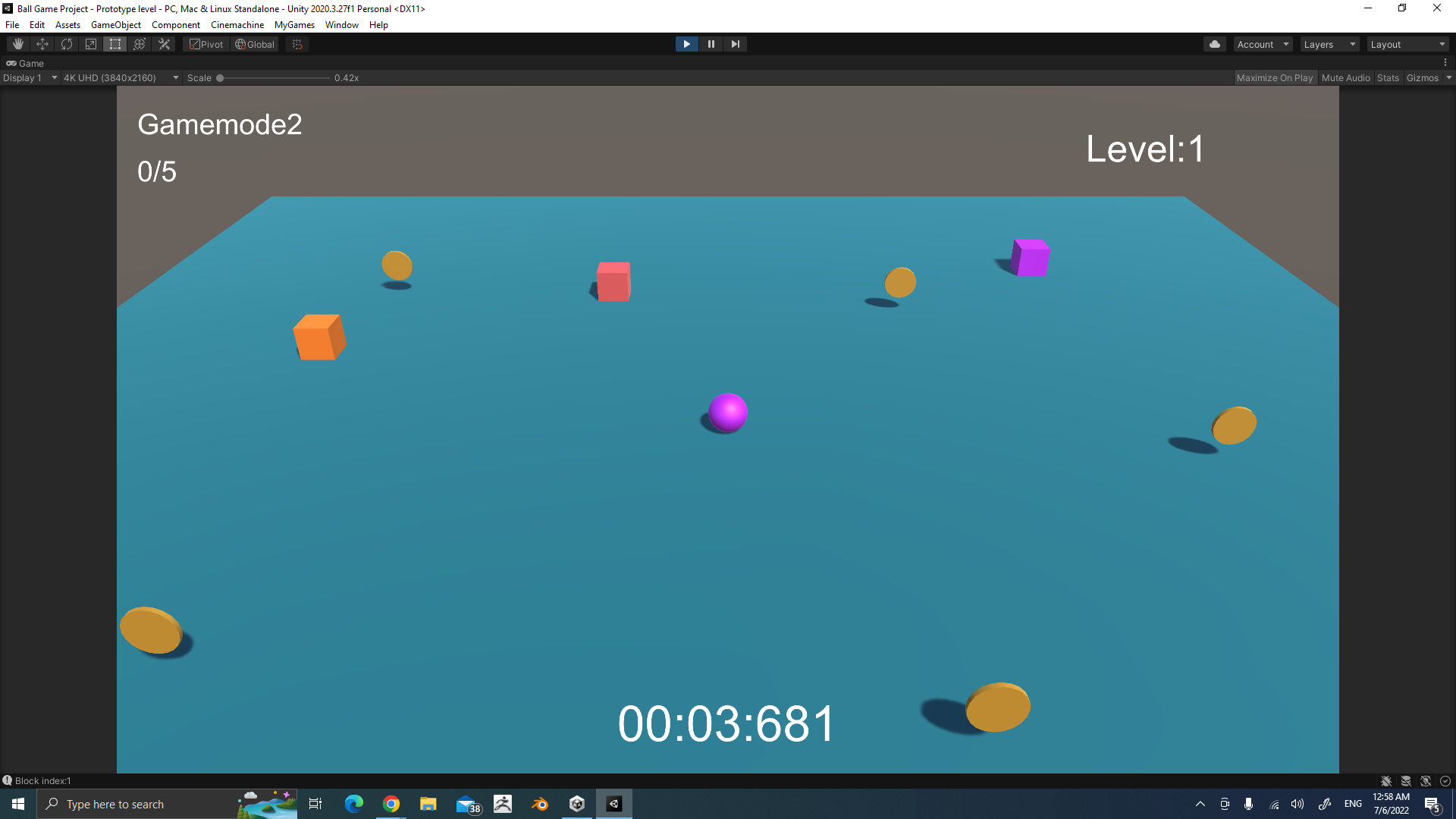
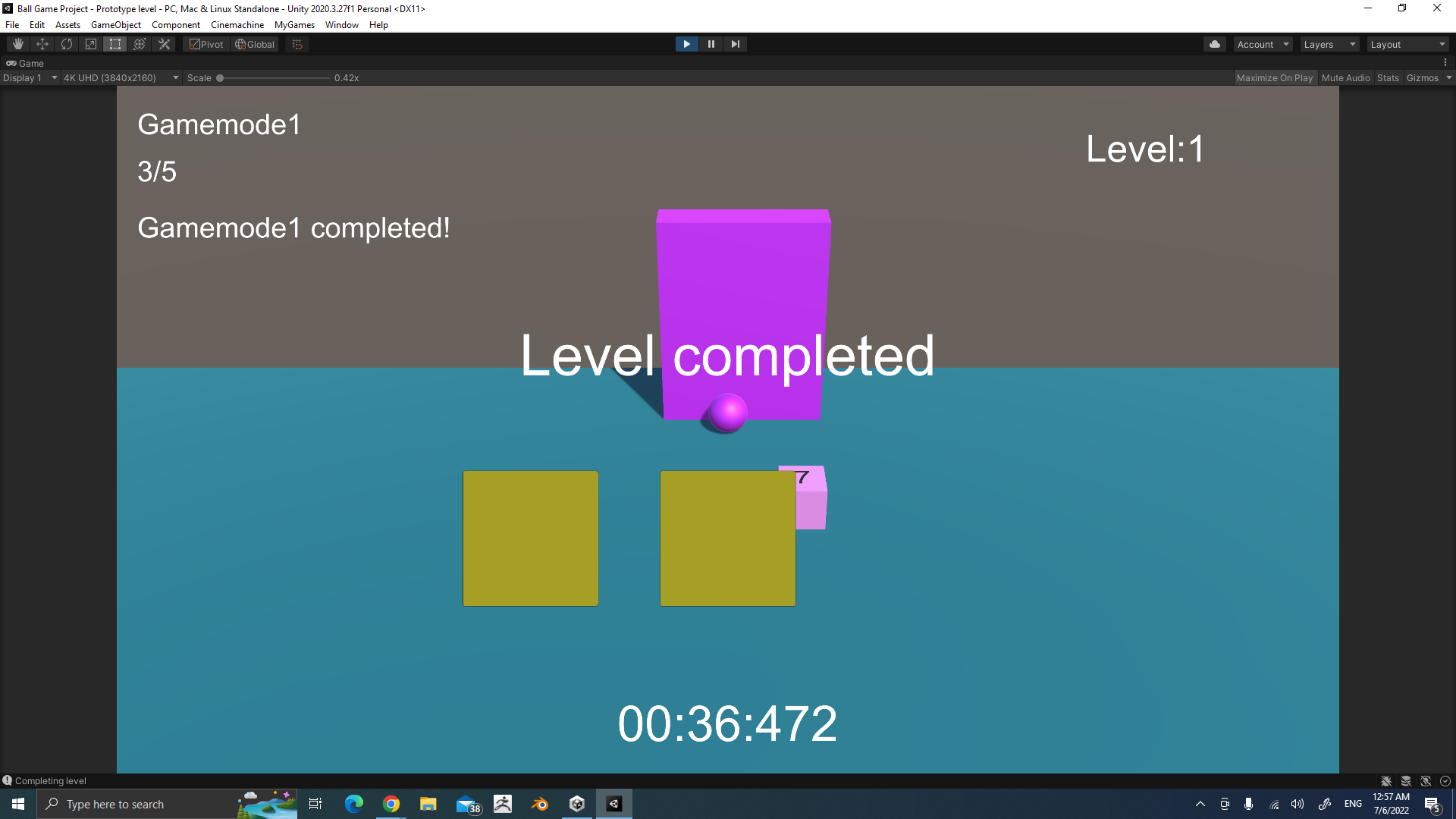
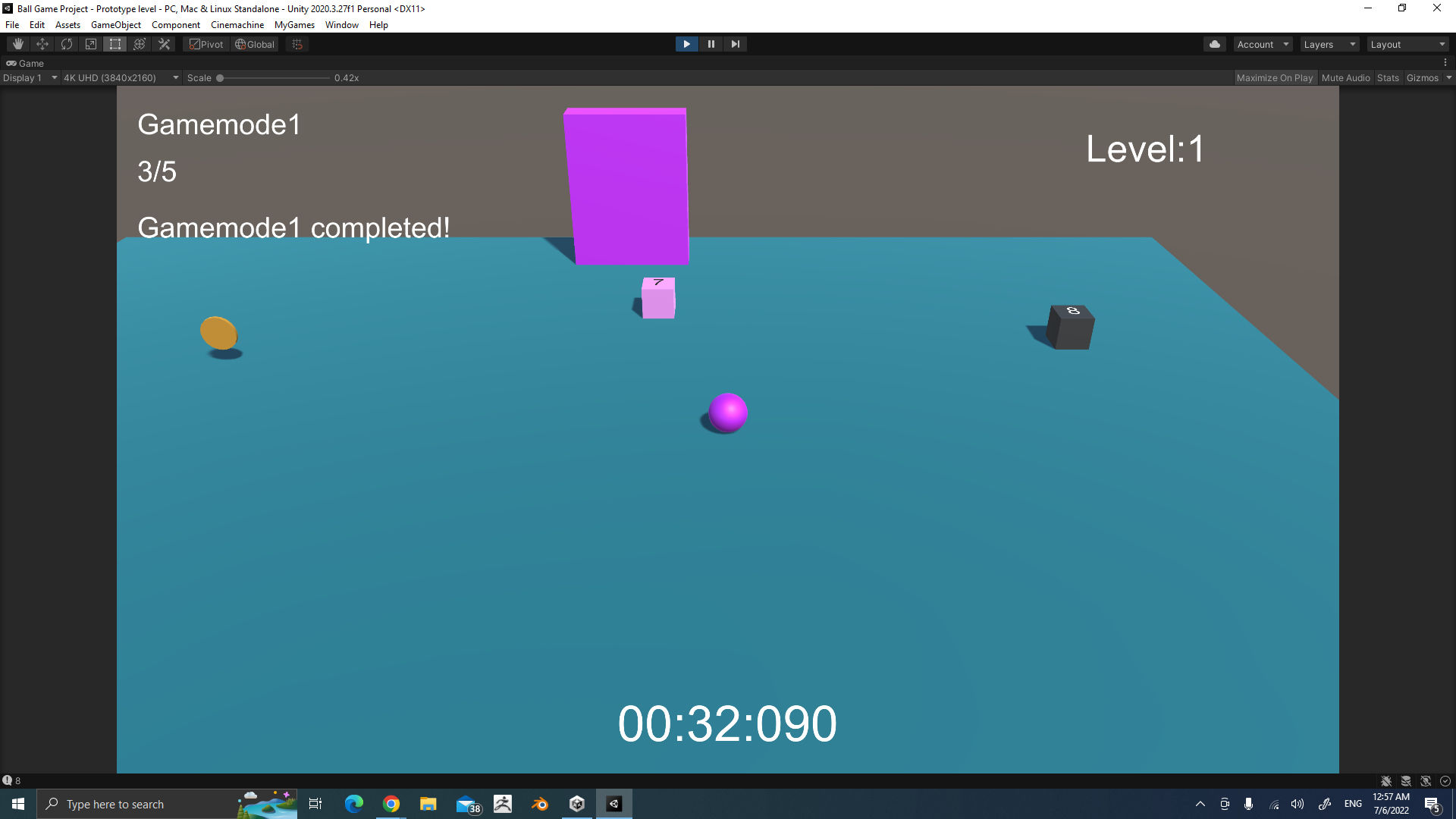
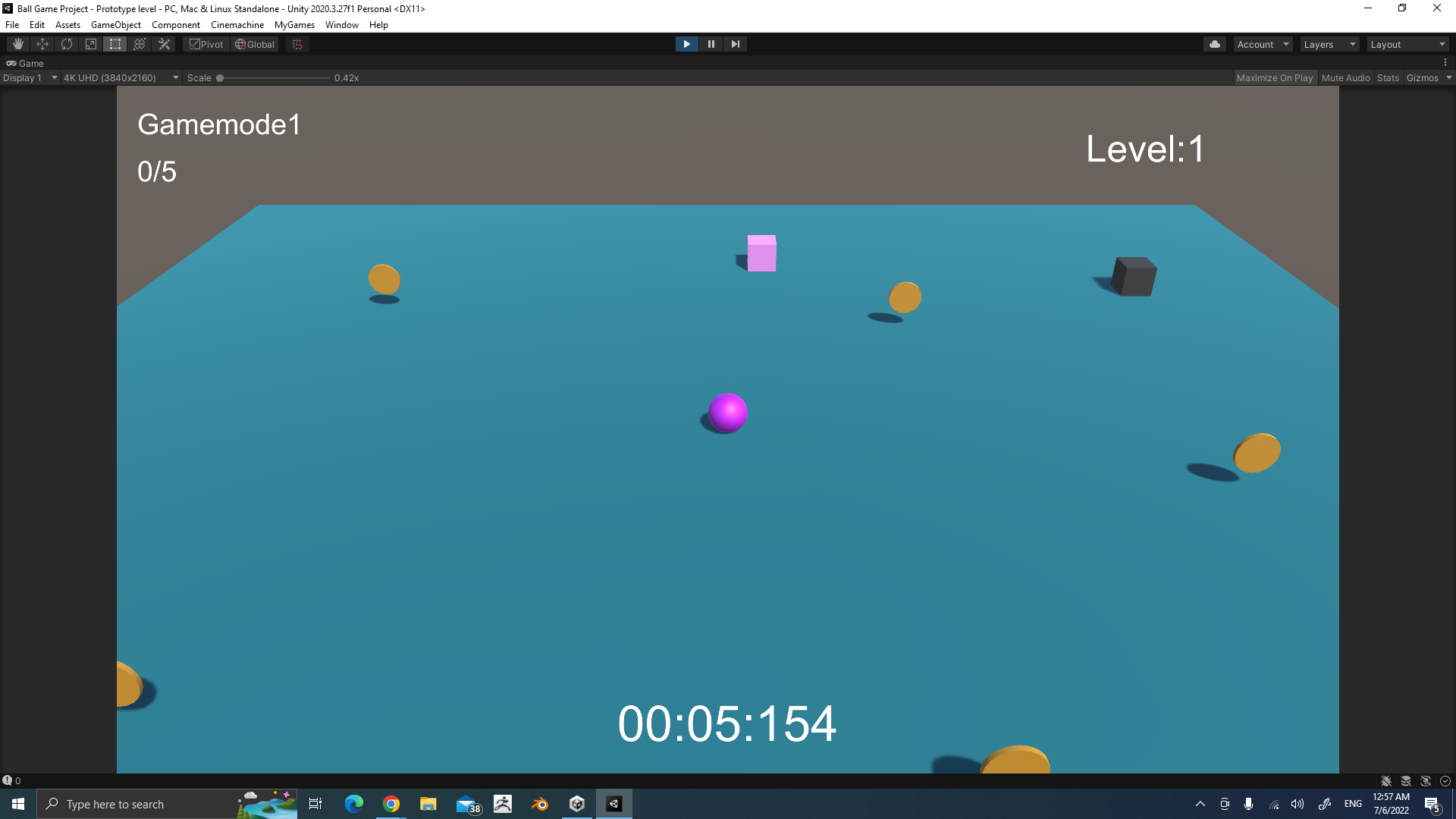
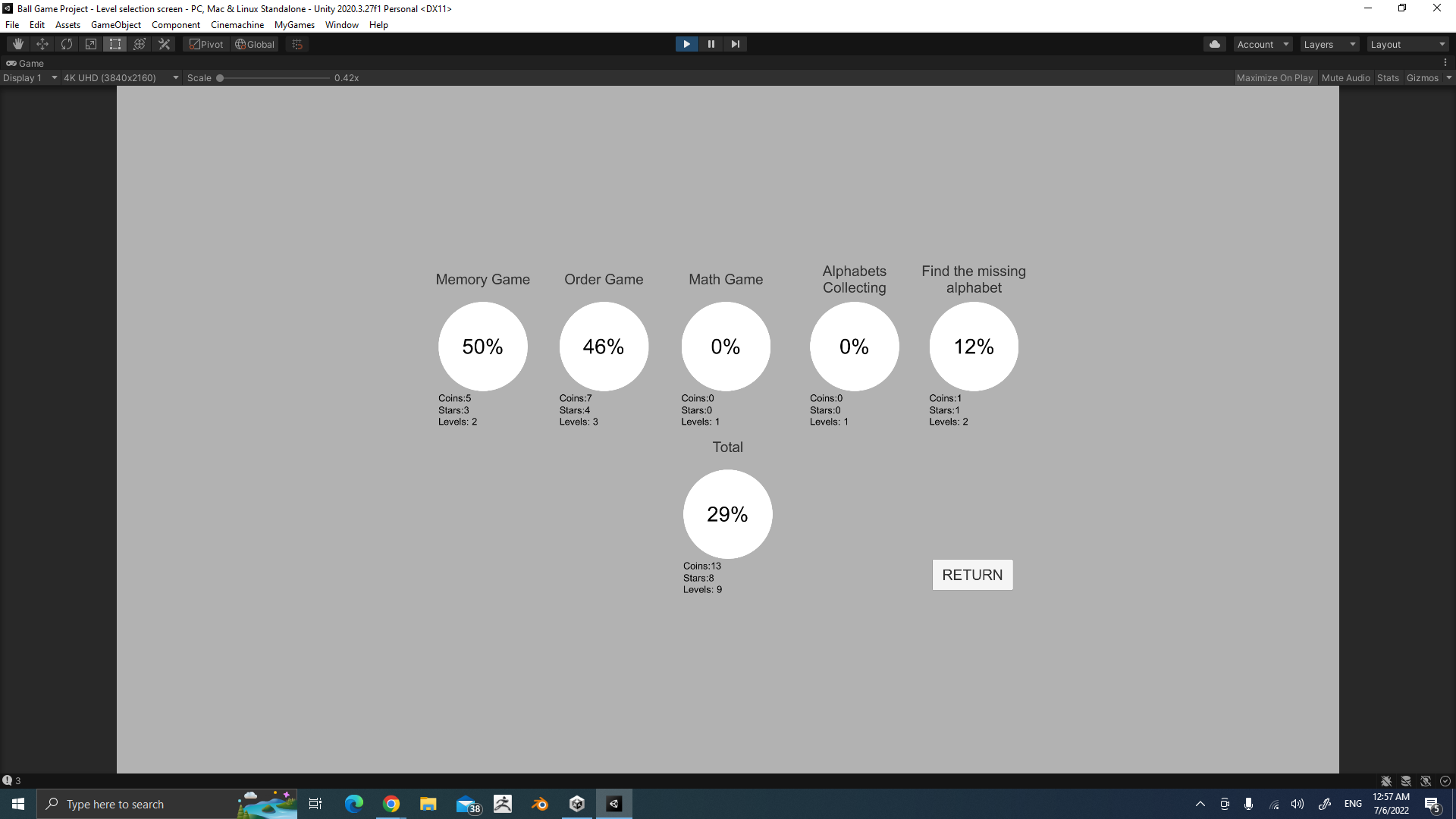
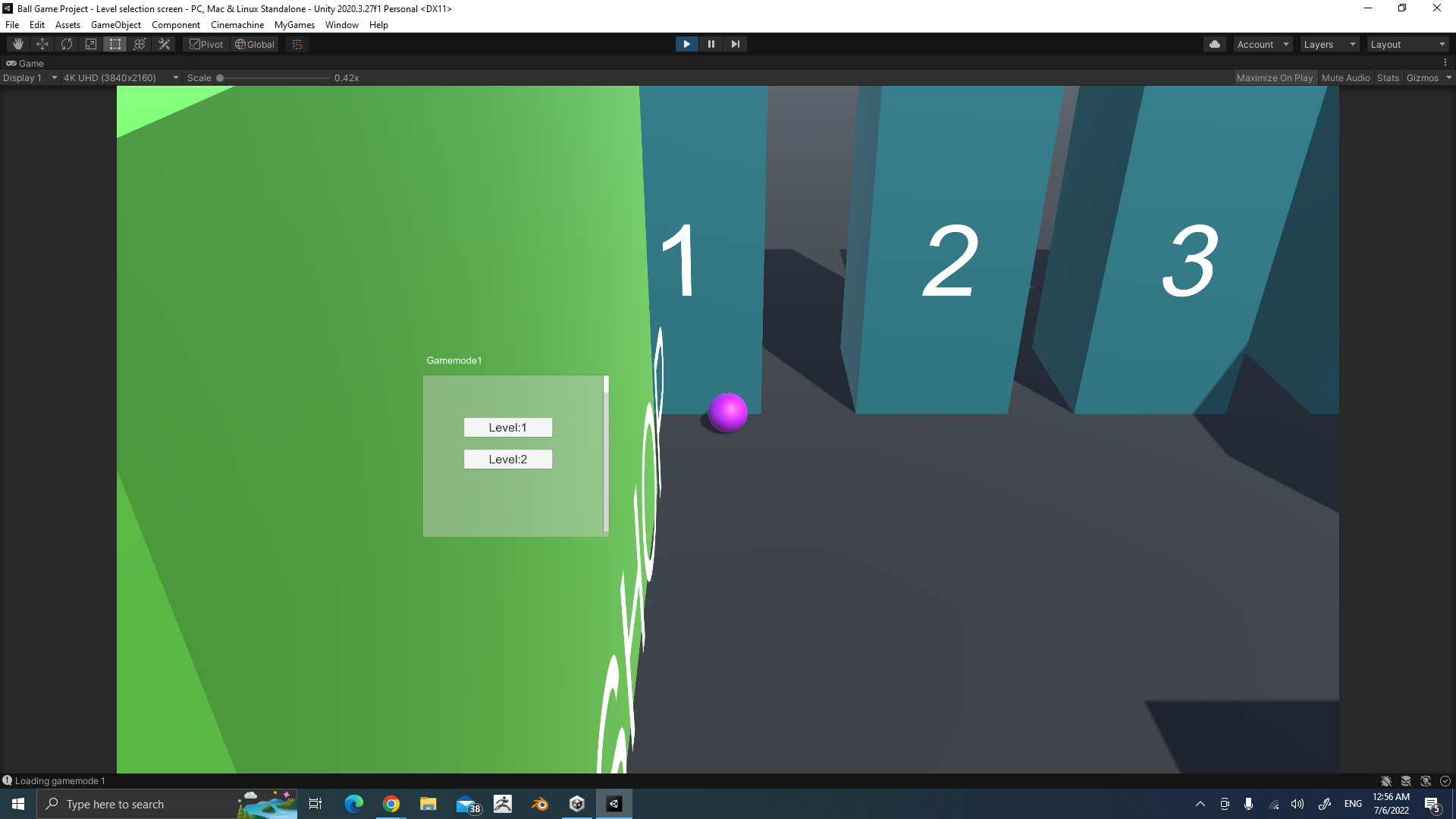
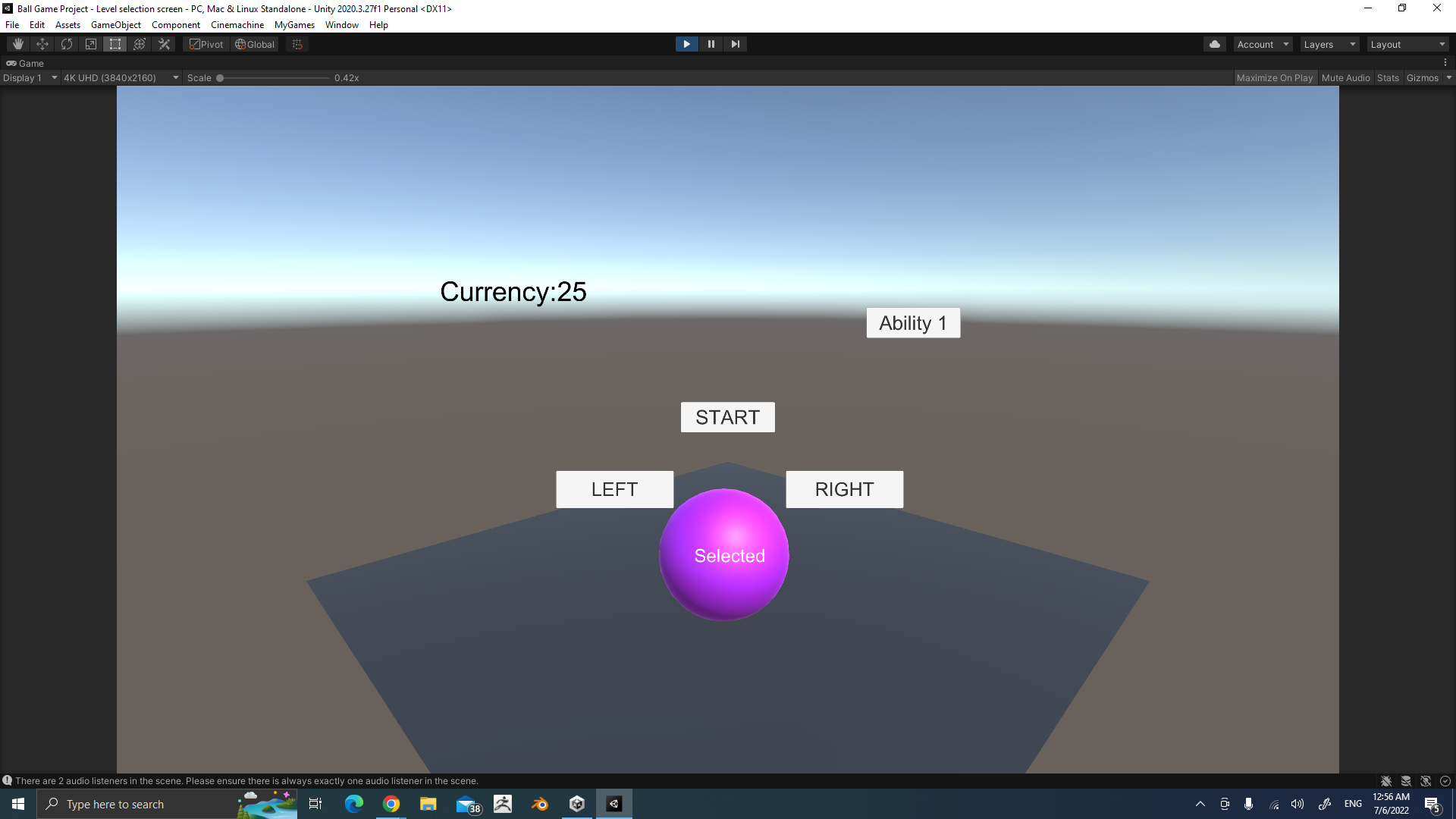
**3. Performance**

Gameplay can be affected by the performance of UI and so we placed the GUI carefully around a gameworld to give the player as much immersion as possible to enhance the gameplay. If the UI of a game contains too much information The children with ADHD will be distracted are forced to slow down gameplay and it stalls the purpose of our game which is aims to educate them and increase their focus in an entertaining way, whereas if there is too little information given on the UI children become frustrated having to again, slow down gameplay to find something they may need, UI should make tasks in-game as quick and seamless as possible to keep them immersed in the game world. The UI can also manipulate kid behavior as UI objects can be placed into the game world to make him/her either want to run towards them or away from them.

**4. Usability and Compliance**

Coming to this part of the game we tried to maximize effectiveness, efficiency and satisfaction, because usability is very important in a game. For example If the player has to struggle with problems that make playing less fun than doing something else, then there is nothing to stop the player from switching off the console. This is a serious risk as the user experience is very sensitive to usability problems. Even the smallest glitch or hiccup in the user interface may render an otherwise good game into a rather annoying experience.





**Chapter 8: Future Works**

**8.1 Future work features**

**8.1.1. Add a Good Background Music**

Adding music that the child likes will help him focus on the game

**8.1.2. Add a feedback page**

So that we can understand the child every day by writing what he likes and dislikes in the content, or adding what is difficult for him to understand so that we can add other content to help him.

**8.1.3. Add a Countdown Timer for games**

As it will help the child to increase his focus and speed as well.

**8.1.4. Add awareness videos for children**

To educate them on how to maintain their tools, how to organize them, and how to focus as well.

**8.1.5. Add voiceover for every level of the game**

to make a game easier for children by adding voices of structures.

**8.1.6. Add a combined test on more than one level determined by the child himself**

In order for the child to be able to test himself on more than one level together, this will increase his focus and ensure the stability of the information.

**8.1.7. Add monitor screen**

To calculate how many times a child uses a game in a week

And the time that the child uses each time he opens the game.

**8.1.8. Increase math levels**

Setting levels suitable for the age group above 15 years.

**8.1.9. Add a specialized section on psychiatry in the game.**

So The child talked with his psychiatrist about the game after he finished each day.

**8.1.10. Determine a weekly and monthly test for the child.**

So that the child's educational level is monitored by the parents to know if he is in progress or not.

**8.1.11. Add a calendar page**

that shows the child the beginning and end of each level

**8.1.12. block ads for children**

As this prevents the child from being distracted and focusing on anything else.

**8.1.13. Adding a button from which the child can stop and save the game**

So that he can stop the game at any time and complete it from where he left off.

**8.1.14. Add a button to control the Game quality like the Graphics of game levels**

**8.1.15. Add a button to continue a game**

**8.1.16. Add a button to change the game themes**

To help change the mood of the child.

**8.1.17. Adding videos for children**

to make it easier for them to learn how to know the system of any level and its tests if they do not know how to use it.

**8.2 Future Plans**

**8.2.1. Increase the number of games.**

1. science levels will be added
2. Entertainment levels will be added after each level

**8.2.2. Add a special level for adults**

1. They will be tested so that they can know if they have ADHD or not
2. Will be Added intelligence levels to them to increase focus
3. Will be Added awareness videos for them

**8.2.3. Add a level for UC-MATH**

This level performs fast and accurate calculations which relies on a tool called the abacus, to enhance dexterity and math solving in the child.

**8.2.4. Distributing the game to all doctors who specialize in treating ADHD.**

**8.2.5. Add challenging levels**

In order to increase the enthusiasm of the child

**8.2.6. Add a full version of 3D game**

To make children more affected with the game.

**8.2.7. Add a specialized section of psychiatry in the game.**

By adding correct information about ADHD. In video or in education content.

**8.3 What is the impact of the future work of our project?**

1. Increasing the number of people with ADHD who will play at different and more levels in the game.

2. Facilitating study at different levels for children with ADHD.

3. ADHD children will learn to deal more with situations that occur throughout the day.

4. Increasing the number of trained centers and intensifying treatment using our project.

5. Increasing the correct definition range for the term ADHD.

6. The presence of an educational reference for the personal situations of children and how to deal with them.

7. Improving the academic education of mathematics for different levels.

8. Knowing the diagnosis and treatment of adults and enhancing their focus.

9. Increasing the level of entertainment with education for children of ADHD.

10. Parents will be able to interact more with their children at different levels of study.

11. Children's education levels will be closely monitored through the results of the week and month tests.

12. Knowing all details of how children start and finish the game by seeing all the reports.

13. Pay attention and learn more about adults with ADHD.

14. Strengthening attention and the link between more than one action in the game.

15. Helping children with ADHD with correct psychological information by adding a specialized section of psychiatry in the game.