**The Art Dealer Game Project**

**Documentation**

**By**

**Swamy Gorla (L30096992)**

**and**

**Suresh Reddy Muvva (L30086343)**

**Software Engineering**

**CPSC-60500-002**

**Table of Contents Page No.**

1. **Introduction………………………………………………………………………4**
   1. Purpose…………………………………………………………………4
   2. Scope…………………………………………………………………...4
   3. Audience……………………………………………………………….4
   4. Overview of the Game………………………………………………....4
2. **Project Overview………………………………………………………………….4**
   1. Game Concept……………………………………………………….…4
   2. Target Audience (K-8 Students) …………………………………….…5
   3. Core Objectives (Educational Goals) ………………………………….5
3. **System Design………………………………………………………………….….6**
   1. Architecture Diagram……………………………………………….….6
   2. Description of the diagram………………………………………….….6
   3. Modules & Components…………………………………………….….7
4. **Implementation…………………………………………………………………....7**
   1. Game Logic & Features………………………………………………...7
   2. Stages of the Game (Stage 1, 2, 3) ……………………………………..8
   3. Technologies Used (Python, Tkinter, etc.) ………………………….….8
5. **User Interface…………………………………………………………………...…9**
   1. Screenshots of Game UI…………………………………………….….9
   2. User Interaction Flow………………………………………………….11
6. **Game Mechanics………………………………………………………………….11**
   1. Scoring System………………………………………………………...11
   2. Difficulty Levels and Adjustments…………………………………….11
   3. Multiplayer and Single-Player Modes…………………………………12
7. **Teaching Aspects………………………………………………………………….12**
   1. How the Game Teaches Computation & Math………………………...12
   2. Integration of Computational Thinking Concepts………………….….12
8. **Development Process………………………………………………………….….12**
   1. Agile Development Approach………………………………………....12
   2. Task Breakdown……………………………………………………….12
   3. Challenges and Solutions……………………………………………...13
9. **Testing & Debugging………………………………………………………….….13**
   1. Unit Testing Approach………………………………………………....13
   2. Example Test Cases……………………………………………………13
   3. Bug Fixing and Troubleshooting………………………………………13
10. **Deployment…………………………………………………………………….…14**
    1. Steps for Running the Game…………………………………….……14
    2. System Requirements…………………………………………………14
11. **Conclusion…………………………………………………………...……………14**
    1. Summary of Achievements……………………………...…………….14
    2. Future Enhancements…………………………………...………….….15
12. **References……………………………………………………………………………15**

**1.0 Introduction**

**1.1 Purpose**

The purpose of The Art Dealer Game is to provide a platform where K-8 students can learn fundamental mathematical and computational thinking concepts through an interactive and engaging medium. The game is designed to teach students pattern recognition, logic, and problem-solving skills using fun and intuitive gameplay.

**1.2 Scope**

This documentation provides a comprehensive overview of the project, covering the design, implementation, and testing phases. It is intended to guide educators and developers through the mechanics and objectives of the game.

**1.3 Audience**

This document is meant for educators, educational technologists, developers, and students interested in understanding or expanding upon the game. It also serves as a resource for future developers who may want to enhance or adapt the game for other educational purposes.

**1.4 Overview of the Game**

The game is divided into three stages, each aimed at a different age group (K-2, 3-5, 6-8). The focus is on helping students recognize patterns, solve mathematical puzzles, and develop computational thinking skills in an enjoyable way. The game can be played in single-player or multiplayer mode, and difficulty levels adjust dynamically based on the player's performance and grade level.

**2.0 Project Overview**

**2.1 Game Concept**

In The Art Dealer Game, players assume the role of an art dealer who must solve various puzzles related to pattern recognition and logic in order to progress through the levels. The game introduces students to computational thinking and math concepts through tasks that simulate real-world problem-solving scenarios in a playful and interactive environment.

**2.2 Target Audience**

The primary audience for the game is K-8 students, segmented into three age groups:

* Stage 1: Grades K-2 (Basic math and pattern recognition)
* Stage 2: Grades 3-5 (Intermediate puzzles with logic and math challenges)
* Stage 3: Grades 6-8 (Advanced computational thinking tasks)

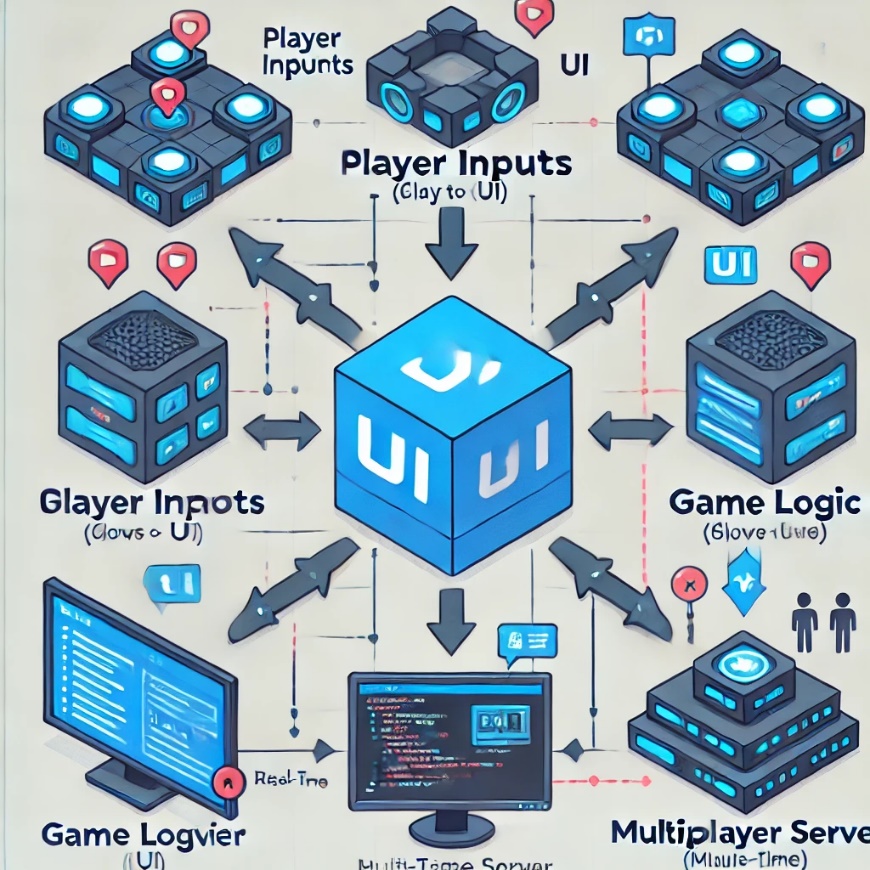
The game is designed to be age-appropriate, with increasing difficulty levels and complexity to cater to the cognitive development of each age group.

**2.3 Core Objectives**

* Educational: Teach fundamental computational thinking and mathematical skills.
* Engagement: Maintain high levels of engagement through interactive, gamified learning.
* Scalability: Adapt to different grade levels by adjusting difficulty in real-time.

**3.0 System Design**

**3.1 Architecture Diagram**

**

**3.2 Description of the diagram:**

1. **Player Inputs**: The player interacts with the game through the UI, providing input such as selecting answers, navigating menus, and selecting multiplayer or single-player modes.
2. **UI (User Interface)**: The UI serves as the primary medium through which the player interacts with the game. It captures player input (e.g., mouse clicks, keyboard entries) and displays the game state. It passes inputs to the Game Logic module and receives updated game states to render on the screen.
   * **Input from Player → UI → Game Logic**
   * **Updated Game State → UI**
3. **Game Logic**: This module contains the core logic of the game, including the generation of puzzles, scoring system, and difficulty adjustments. It processes the player's inputs from the UI and sends feedback to the UI, updating the game state.
   * **Game State → UI**
   * **Player Input → Game Logic**
4. **Multiplayer Server (if in multiplayer mode)**: When two players connect for a multiplayer session, the Multiplayer Server ensures that both players' actions are synchronized in real time. It sends and receives updates between the Game Logic and the UI of both players, making sure that both game states remain consistent.
   * **Player 1 Game Logic → Multiplayer Server → Player 2 Game Logic**
   * **Multiplayer Sync → Game Logic → UI**
5. **Score Tracker**: The score tracker receives input from the Game Logic, tracking how well players perform based on time, accuracy, and other metrics. The score is then displayed in the UI for the player to see.

**3.3 Modules & Components**

* Game Logic: Manages the core gameplay loop, including puzzle generation, scoring, and level progression.
* UI (User Interface): Built using the Tkinter library, the UI handles all player interactions, menu navigation, and game controls.
* Score & Performance Tracker: A module to track players' scores based on their performance, time to solve puzzles, and accuracy.
* Multiplayer: The multiplayer component manages game sessions between two players and synchronizes game states.

**4.0 Implementation**

**4.1 Game Logic & Features**

The game logic is implemented in Python, with each stage tailored to the needs of the respective age group. Key features include:

* Puzzle Generation: Each stage generates puzzles dynamically to keep the content fresh and challenging.
* Single Player Mode: Players can progress through the game individually, focusing on learning at their own pace.
* Multiplayer Mode: In this mode, players compete against each other to solve puzzles faster than their opponent.
* Difficulty Adjustments: The difficulty increases automatically based on the player's performance. Younger players will start with simple patterns and math problems, while older players will face more complex tasks involving computation and logic.

**4.2 Stages of the Game**

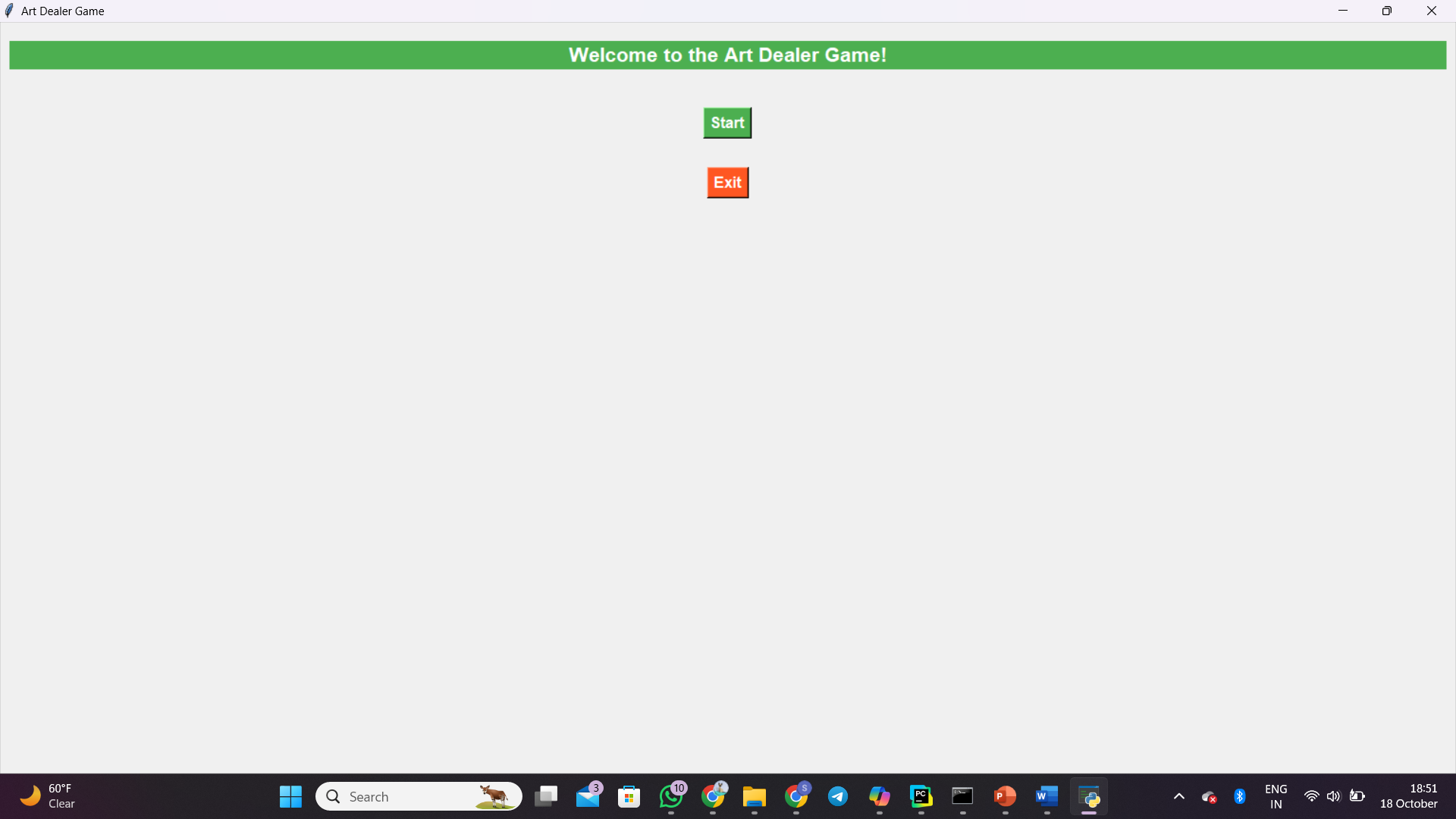
* Stage 1 (Grades K-2): Simple patterns and basic arithmetic problems.
* Stage 2 (Grades 3-5): Intermediate problems, introducing more complex math concepts.
* Stage 3 (Grades 6-8): Advanced puzzles requiring deeper computational thinking, such as decomposition and pattern recognition.

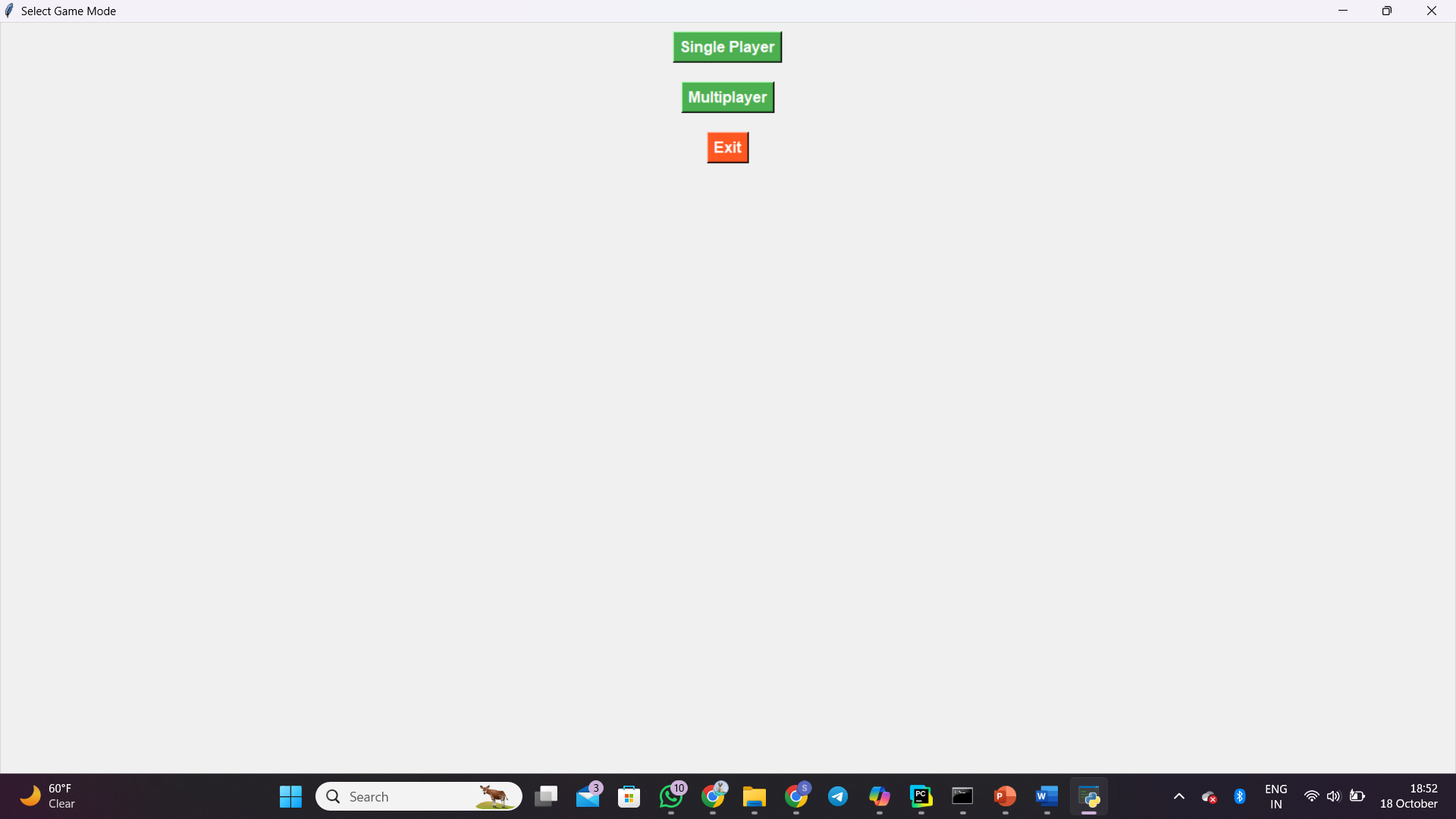
**4.3 Technologies Used**

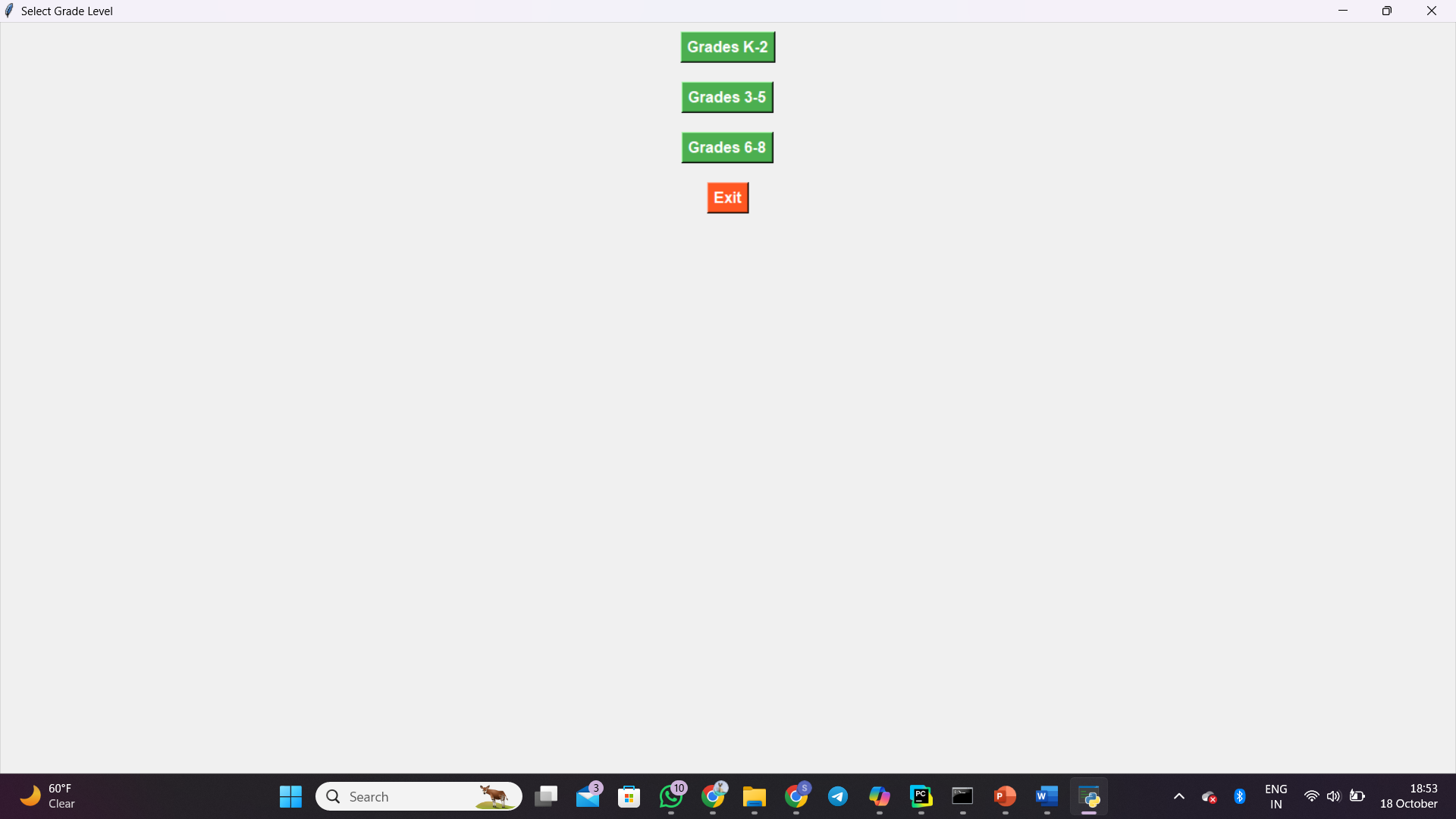
* Python: The primary programming language for the game logic and mechanics.
* Tkinter: Python’s standard GUI library for building the game interface.
* SQLite: For storing player data and scores.

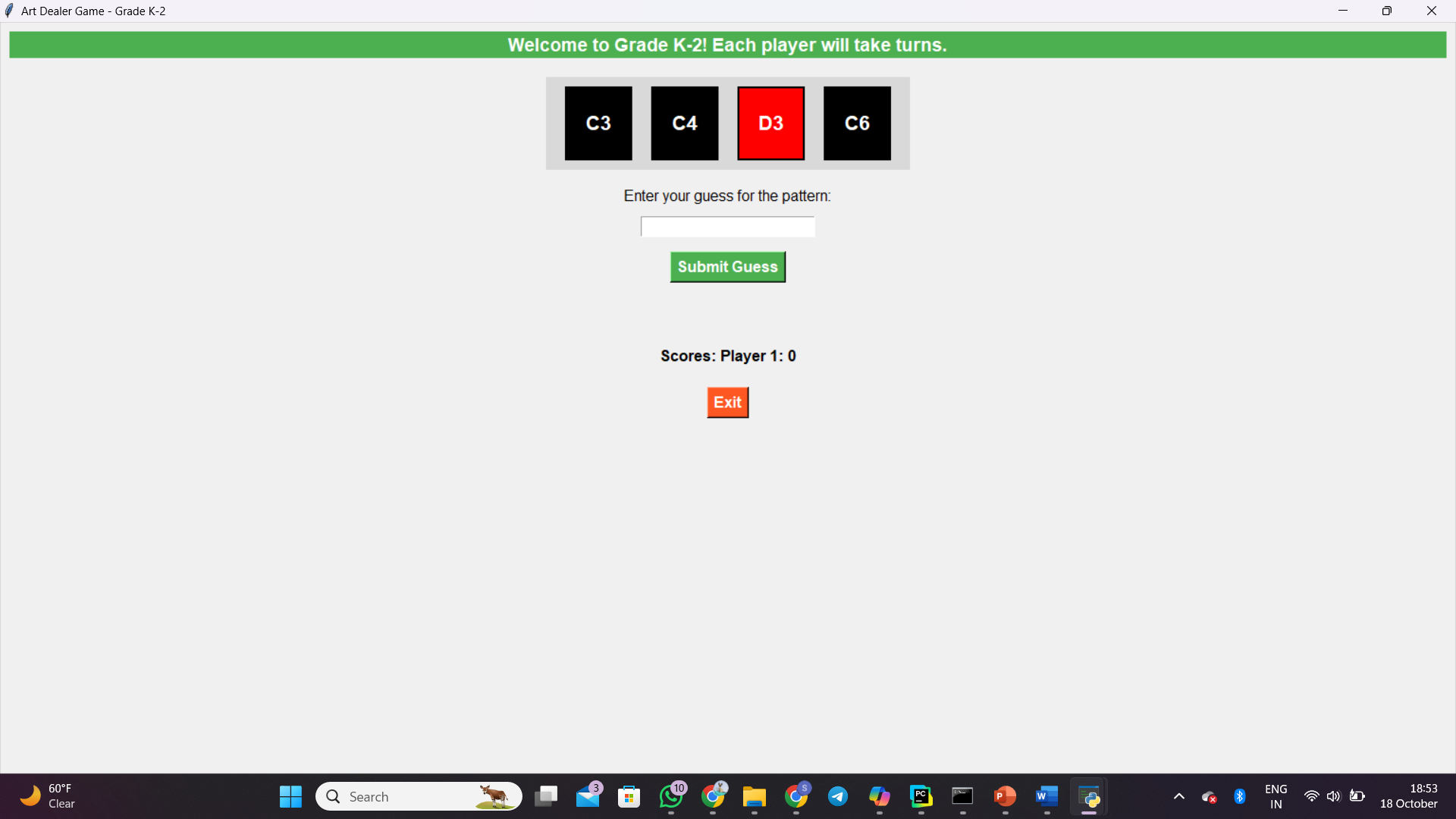
**5.0 User Interface**

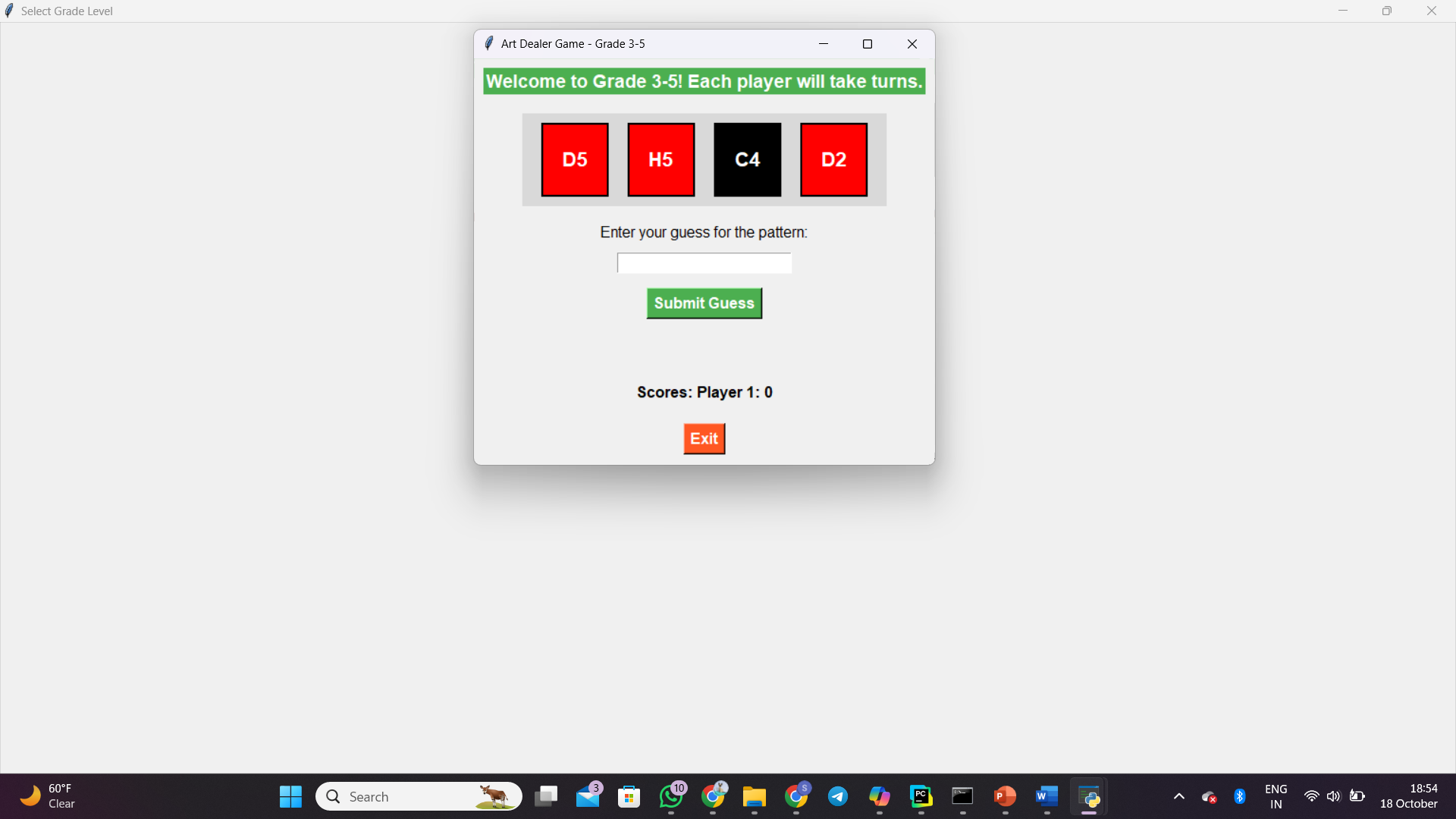
**5.1 Screenshots of Game UI**

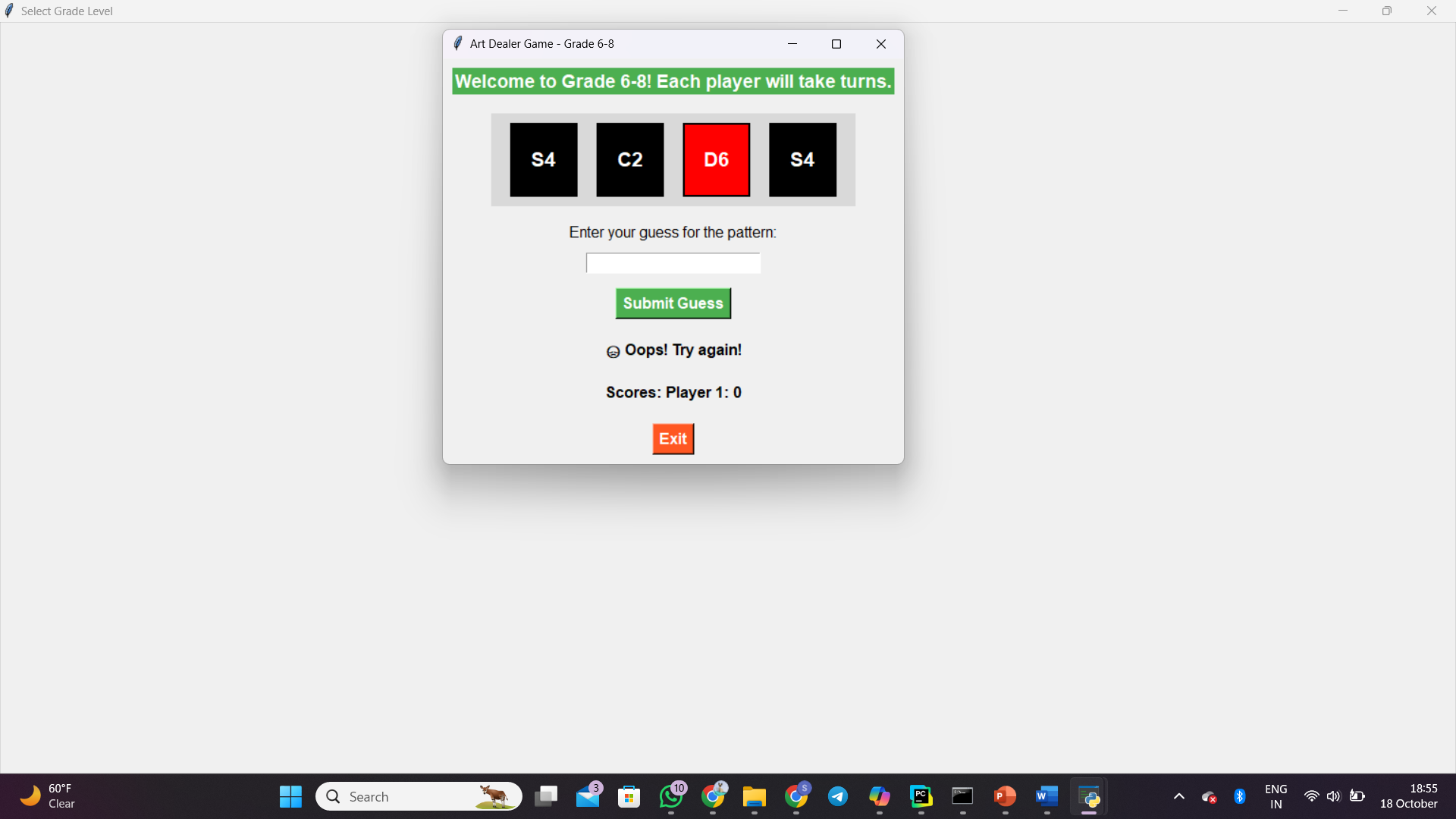












**5.2 User Interaction Flow**

1. Main Menu: Select between Single Player and Multiplayer mode.
2. Gameplay Screen: Players solve puzzles presented based on their selected stage.
3. Score Display: After completing a puzzle, players see their performance score.

**6.0 Game Mechanics**

**6.1 Scoring System**

Points are awarded based on:

* Speed: How quickly a puzzle is solved.
* Accuracy: Correctness of the answers.

**6.2 Difficulty Levels and Adjustments**

The game dynamically adjusts its difficulty based on the player's performance. If a player is struggling, the game will lower the difficulty to keep the experience fun and educational.

**6.3 Multiplayer and Single-Player Modes**

In Single-Player Mode, players work on solving puzzles at their own pace. In Multiplayer Mode, two players compete in real-time to complete puzzles faster than their opponent. Both modes track performance and adjust difficulty as needed.

**7.0 Teaching Aspects**

**7.1 How the Game Teaches Computation & Math**

* Stage 1 focuses on basic math and pattern recognition, teaching children how to identify sequences and solve simple arithmetic problems.
* Stage 2 introduces more complex patterns and teaches players how to apply logic to solve puzzles.
* Stage 3 incorporates computational thinking, requiring players to break down complex problems into manageable parts.

**7.2 Integration of Computational Thinking Concepts**

* Pattern Recognition: Players must identify trends and patterns in the puzzles.
* Decomposition: In later stages, players learn to break down a problem into smaller steps, aiding in the process of computational thinking.
* Abstraction: The game introduces abstraction in its highest difficulty stages by asking players to generalize solutions to more complex problems.

**8.0 Development Process**

**8.1 Agile Development Approach**

We used an Agile development approach, with bi-weekly sprints to ensure continuous progress and iterative feedback from playtesting sessions.

**8.2 Task Breakdown**

* Team Member 1:
  + Designed the game logic and scoring system.
  + Developed the UI with Tkinter.
* Team Member 2:
  + Implemented multiplayer features and testing.
  + Managed documentation and ensured educational alignment with K-8 standards.

**8.3 Challenges and Solutions**

* Challenge: Getting multiplayer synchronization right.
  + Solution: Introduced a robust system of game-state updates and latency checks.
* Challenge: Making the UI intuitive for K-8 students.
  + Solution: Conducted user testing with students and iterated on the design based on their feedback.

**9.0 Testing & Debugging**

**9.1 Unit Testing Approach**

We wrote unit tests for each game component, including the puzzle generation system, the scoring algorithm, and the multiplayer connection system.

Example Test Cases

* Test Case 1: Ensure that puzzles are solvable and follow a logical pattern.
* Test Case 2: Test multiplayer synchronization by having two players solve a puzzle simultaneously.

Bug Fixing and Troubleshooting

* Identified and fixed bugs related to performance tracking in multiplayer mode.
* Addressed issues where the difficulty wasn't adjusting properly for students struggling with the puzzles.

**10.0 Deployment**

**10.1 Steps for Running the Game**

1. Install Python (version 3.9 or higher) or any compatible Python compilers like Jupiter Notebook, PyCharm etc….
2. Install required libraries using the following command: pip install tkinter sqlite3.
3. Clone the project repository.
4. Run the game using the command: python main.py.

**10.2 System Requirements**

* Operating System: Windows, macOS, Linux.

**11.0 Conclusion**  
**11.1 Summary of Achievements**

The following goals have been successfully attained by The Art Dealer Game:

* **Engaging Learning Environment:** Through gaming, the game teaches K–8 kids computational thinking and pattern identification in an engaging and entertaining setting.
* **Age-appropriate Content**: The game skilfully modifies its degree of difficulty according to the player's grade level, making it both demanding and playable by all.
* **Multiple Player and Single Player Modes**: Having both single player and multiplayer options promotes peer rivalry and cooperation, improving social skills and teamwork.
* **User-Friendly Interface:** Students can easily explore the game thanks to its intuitive design and user interface, which frees them up to concentrate on learning rather than being irritated by difficult controls.
* **Real-Time Feedback:** By giving students immediate feedback on their performance, the scoring system helps them identify their areas of strength and growth.
* **Robust Documentation:** The project documentation is extensive and provides educators and developers with a great resource by outlining the game's features, design, implementation, and educational advantages.

**11.2 Future Enhancements**

Although The Art Dealer Game has achieved its initial goals, there are a few improvements that might be made for further iterations:

* **Increased Game Content:** Adding more stages, obstacles, and visual elements can provide players a wider variety of learning opportunities and keep them interested.
* **Customizable Avatars:** Encouraging gamers to design and alter their avatars may boost player engagement and improve personalization.
* **Other Educational Resources:** Teachers may optimize the educational value of the game by including other materials, such activity guides or lesson plans.
* **Improved Analytics Dashboard:** Creating an analytics tool to monitor student performance and growth can give teachers important information for more individualized education.
* **Mobile Version:** The game might be made more accessible by including a mobile version that kids could play on different kinds of devices.
* **Community Engagement Tools:** Including chat or forum tools can help users connect with one another and develop a sense of community that promotes cooperation and strategy sharing.
* **Frequent Updates and Feedback Loops:** By creating a user feedback system, the game may be improved over time in response to player experiences and recommendations.

**12.0 References**

1. **Educational Resources**
   * CSTA (Computer Science Teachers Association). (2020). *K-12 Computer Science Framework*. Retrieved from CSTA Framework.
   * ISTE (International Society for Technology in Education). (2017). *ISTE Standards for Students*. Retrieved from ISTE Standards.
2. **Game Design Principles**
   * Fullerton, T. (2018). *Game Design Workshop: A Playcentric Approach to Creating Innovative Games*. CRC Press.
   * K. H. Squire & S. Barab (2004). “Advancing the Design of Educational Games.” *Games, Learning, and Society: Learning and Meaning in the Digital Age*, pp. 10-20.
3. **Python and Game Development**
   * Pygame Documentation. (n.d.). Retrieved from Pygame Documentation
   * Zelle, J. (2010). *Python Programming: An Introduction to Computer Science*. Franklin, Beedle & Associates Inc.
4. **Educational Psychology**
   * Piaget, J. (1973). *To Understand is to Invent: The Future of Education*. Viking Press.
   * Bruner, J. (1996). *The Culture of Education*. Harvard University Press.
5. **Computational Thinking**
   * Wing, J. M. (2006). “Computational Thinking.” *Communications of the ACM*, 49(3), 33-35. DOI: 10.1145/1118178.1118213.
6. **Game Testing and Quality Assurance**
   * B. M. K. Ma & R. E. K. LeBlanc (2002). “Game Testing: A Practical Approach.” *Game Development Conference*.
7. **Agile Development**
   * Beck, K. et al. (2001). *Manifesto for Agile Software Development*. Retrieved from [Agile Manifesto](https://agilemanifesto.org/).
   * Cohn, M. (2004). *User Stories Applied: For Agile Software Development*. Addison-Wesley.
8. **User Interface Design**
   * Norman, D. A. (2013). *The Design of Everyday Things: Revised and Expanded Edition*. Basic Books.
   * **Shneiderman, B., & Plaisant, C. (2010). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Pearson.**
9. **Pattern Recognition in Education**
   * **K. P. Murphy & A. L. Beck (2018). “Teaching Pattern Recognition: What Teachers Need to Know.” *Journal of Educational Psychology*.**
10. **Programming Resources**
    * **SQLAlchemy Documentation. (n.d.). Retrieved from SQLAlchemy Documentation.**
    * **Python Documentation. (n.d.). Retrieved from** [**Python Docs**](https://docs.python.org/3/)**.**