**Pure Connect: A basic social media app with**

**news updates for IOS**

**Submitted By**

**Israt Tasnim Esha**

Roll: 1907090



**Submitted To:**

Md. Shahidul Salim

Lecturer

Dept. of Computer Science and Engineering

Khulna University of Engineering & Technology

Most. Kaniz Fatema Isha

Lecturer

Dept. of Computer Science and Engineering

Khulna University of Engineering & Technology

1. Introduction

* 1. Background

Welcome to the magical world of Pokémon, where childhood dreams come alive and adventures await at every corner. Remember when you were captivated by Ash's journey, battling alongside Pikachu, and outsmarting Team Rocket? Now, your childhood fantasy becomes a reality with our new game, **Pokémon Battle**. Dive into a magical world where you become Ash, starting on a thrilling adventure to gather your Pokémon and engage in epic battles that will test your strategy and courage.

In this game, two players stand as the main contenders: **Ash,** the eternal hero, and **Team Rocket**, the evil challengers. Each player must assemble a team of three Pokémon to qualify for the ultimate battle. Ash's loyal companions are **Pikachu**, the electrifying mouse; **Charmander**, the fiery lizard; and **Squirtle**, the water-splashing turtle. On the other side, Team Rocket brings their cunning **Meowth**, faithful **Wobbuffet**, and smoky **Weezing**.

The journey begins with a dramatic twist: all Pokémon are undergoing health check-ups at three different **Pokémon Centers**. Players start with the location of the first center, but the next destinations are revealed by Nurse Joy only after reaching the first Pokémon. Each player travels in three different vans with limited fuel, racing against the clock to reach their Pokémon before the fuel runs out. Time and fuel are precious commodities—will they make it in time to qualify for the competition? After qualifying, the real battle begins. Each player receives an equal amount of coins to buy potions, crucial for healing their Pokémon during intense battles. This adventure is more than just a game; it's a race against time, a test of strategy, and a journey filled with excitement. Does the Pokémon Championship Adventure await—are Ash and Team Rocket ready for the challenge of a lifetime? The dream of becoming a Pokémon master is within reach!

The adventure isn't just about battles; it's about reliving those cherished moments of your childhood, where every corner of the Pokémon world held a new surprise, a new friend, and a new challenge. So, gear up, be like Ash, and get ready to experience the Pokémon journey of your dreams.

* 1. Objectives and Goals:
* **Recreate the Magic:** Capture the wonder and excitement of the Pokémon series in a Python-based game.
* **Strategic Gameplay:** Implement strategic elements that allow players to choose and manage their Pokémon wisely.
* **Engaging Adventure**: Create a fascinating adventure for Ash to retrieve his Pokémon, adding depth and excitement to the gameplay.
* **Epic Battles**: Simulate Pokémon battles with attack, defense, and elixir moves, bringing the thrill of the series to life.
* **Dynamic Swapping**: Allow players to swap Pokémon during battles, adding a layer of strategy and unpredictability.
* **Victory Conditions**: Define clear win/lose conditions based on the health status of all Pokémon, ensuring a fair and exciting competition.

**2. Methodology:**

**2.1 pokemon\_find\_screen.py**

* **Setup**
  + Randomly assign Ash, Team Rocket, and their respective Pokémon to different Pokémon centers and starting locations.
  + Ash knows the location of Pikachu, while Team Rocket knows the location of Meowth.
  + After Ash catches Pikachu, Nurse Joy provides the location of Charmander, followed by Squirtle. Team Rocket follows the same sequence with Meowth, Weezing, and Wobbuffet.
* **Mechanics**
  + Each player has a limit of 15 liters of fuel to catch each Pokémon and a total of 1 minute to catch all their Pokémon. A player is eliminated if they exceed the fuel limit or fail to catch all Pokémon within 1 minute.
  + Ash's movements are controlled by the player, while Team Rocket's movements are guided by an A\* algorithm.

**2.2 elixir\_screen.py**

* **Setup**
  + Each player is allocated the same amount of coins to purchase elixirs.
  + 5 elixirs have varying costs and power values.
* **Mechanics**
  + Players aim to collect elixirs with the highest power within their budget.
  + Ash selects elixirs with user assistance.
  + Team Rocket uses a genetic algorithm with tournament selection for optimal elixir collection.
  + Tournament selection picks the best-performing elixirs for each generation.
  + Final elixir collections are used in Pokémon battles to recover health points.

**2.3 play\_screen.py**

* **Setup**
  + Each player has 3 Pokémon, each starting with 100 health points.
  + Battles are turn-based with two types of actions: attack and defense.
* **Mechanics**
* Pokémon and field types affect damage output:
* Type matches between Pokémon and field increase damage to opponent.
* Type advantages: Fire > Electric, Electric > Water, Water > Fire.
* Players can swap Pokémon if the health points of the swapped Pokémon are not 0.
* Elixirs can be used to boost health points.
* **Decision Making**
  + Ash takes user input for actions, elixir usage, and Pokémon swaps.
  + Team Rocket uses the minimax algorithm to choose actions.
  + Fuzzy logic helps Team Rocket decide when to use elixirs or swap Pokémon.

|  |  |
| --- | --- |
| **Keys** | **Actions** |
| Left | Translate camera in x-axis (-1) |
| Right | Translate camera in x-axis (+1) |
| Up | Translate camera in y-axis (+1) |
| Down | Translate camera in y-axis (-1) |
| + | Translate camera in z-axis (+1) |
| - | Translate camera in z-axis (-1) |
| R | Rotation enabled |
| R+X | Rotate camera in x-axis (+1) |
| R+U | Rotate camera in x-axis (-1) |
| R+1 | Rotate camera in x-axis (+90) |
| R+Y | Rotate camera in y-axis (+1) |
| R+V | Rotate camera in y-axis (-1) |
| R+2 | Rotate camera in y-axis (+90) |
| R+Z | Rotate camera in z-axis (+1) |
| R+W | Rotate camera in z-axis (-1) |
| R+3 | Rotate camera in z-axis (+90) |
| L | Light enabled |
| L+1 | All lights on |
| L+2 | All lights off |
| L+3 | All point lights on, directional and spot lights off |
| L+4 | All point lights off, directional and spot lights on |
| L+5 | All directional light on, point and spot lights off |
| L+6 | All directional light off, point and spot lights on |
| L+7 | All spot light on, directional and point lights off |
| L+8 | All spot light off, directional and point lights on |
| D | Cafeteria door enabled |
| D+1 | Cafeteria door open |
| D+0 | Cafeteria door close |
| E+1 | Lift goes to 1st floor |
| E+0 | Lift goes to ground floor |
| O+1 | Lift door open |
| O+0 | Lift door close |
| C+1 | Curtain open |
| C+0 | Curtain close |
| T+1 | Television on |
| T+0 | Television off |
| F+1 | Fan on |
| F+0 | Fan off |

3. Result

In this section we will discuss the features of our project:

**3.1 View**s

|  |  |
| --- | --- |
|  |  |
| *Fig 1.1: Front view* | *Fig 1.2: Back view* |
|  |  |
| *Fig 1.3: Left view* | *Fig 1.4: Right view* |
|  |  |
| *Fig 1.5: Top view* | *Fig 1.6: Bottom view* |

**3.2 Spots**

|  |
| --- |
|  |
| *Fig 2.1: Cafeteria and kitchen* |
|  |
| *Fig 2.2: Theater* |

**3.3 Daylights**

|  |  |
| --- | --- |
|  |  |
| *Fig 3.1: Morning Scene* | *Fig 3.2: Noon Scene* |
|  |  |
| *Fig 3.3: Afternoon Scene* | *Fig 3.4: Evening Scene* |
|  | |
| *Fig 3.5: Night Scene* | |

**3.4 Lighting**

|  |  |
| --- | --- |
|  |  |
| *Fig 4.1: All lights on* | *Fig 4.2: All lights off* |
|  |  |
| *Fig 4.3: Only point lights on* | *Fig 4.4: Only point lights off* |
|  |  |
| *Fig 4.5: Only directional light on* | *Fig 4.6: Only directional light off* |
|  |  |
| *Fig 4.7: Only spotlight on* | *Fig 4.8: Only spotlight off* |

**3.5 Objects**

|  |  |
| --- | --- |
|  |  |
| *Fig 5.1: Chair-Table* | *Fig 5.2: Tool* |
|  |  |
| *Fig 5.3: Interior design* | |
|  |  |
| *Fig 5.4: Mirror and sink* | *Fig 5.5: Kitchen sink* |
|  | |
| *Fig 5.6: Sofa and stair* | |
|  | |
| *Fig 5.7: Kitchen* | |
|  | |
|  |  |
| *Fig 5.8: Television* | |

**3.6 Light Objects**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| *Fig 6.1: Cylinder light* | *Fig 6.2: Star light* | *Fig 6.3: Torus light* |

**3.7 Moving Objects: Translation**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| *Fig 7.1: Ballon* | *Fig 7.2: Light ballon* | |
|  |  | |
| *Fig 7.3: Lift on ground floor* | *Fig 7.4: Lift on first floor* | |

**3.8 Moving Objects: Rotation**

|  |  |
| --- | --- |
|  |  |
| *Fig 8.1: Fan off* | *Fig 8.2: Fan on* |

**3.9 Moving Objects: Scaling**

|  |  |
| --- | --- |
|  |  |
| *Fig 9.1: Cafeteria door opened* | *Fig 9.2: Cafeteria door closed* |
|  |  |
| *Fig 9.3: Curtain opened* | *Fig 9.4: Curtain closed* |
| *Fig 9.5: Lift door opened* | *Fig 9.6: Lift door closed* |

**4. Discussion:**

The game is divided into three main phases: catching Pokémon, collecting elixirs, and battling. Each phase uses different AI algorithms to make the computer opponent strong. A\* algorithm to help Team Rocket find the quickest way to catch their Pokémon in time. Team Rocket (Computer) uses a genetic algorithm to choose the best elixirs. In the battle , Team Rocket uses the minimax algorithm to decide the best moves. Fuzzy logic is also used to decide when to use elixirs or swap Pokémon. This algorithms make the computer opponent unbeatable.

**5. Conclusion:**

We have make a battle game that allow human player to compete against a computer controlled opponent. Computer use different AI algorithm like A\*, Genetic algorithm, Fuzzy logic, minimax which makes the computer opponent unbeatable. The number of Pokémon available in the game is fixed. The user interface (UI) is basic and could be improved to provide a better visual and interactive experience. Additionally, during battles, players have only two options attack or defense without unique attacks for different Pokémon, which reduces the depth of strategy. Moreover, the opponent is always the same (Team Rocket), which can make the gameplay feel boring. Improving these areas could make the game more dynamic.

**6. Future Plans:**

Looking ahead, we want to make **Pokémon Battle** even better:

* **Add More Opponents**: Introduce a variety of opponents, such as different gym leaders, to provide more diverse challenges and strategies for players to face.
* **Increase Pokémon Variety and Unique Attacks**: Increase the number of available Pokémon and include unique attacks for each, adding more depth and strategy to the battles.
* **Improve Pokémon Catching and Map Design**: Make the experience of catching Pokémon more exciting with beautifully designed maps and dynamic catching mechanics.

**7. References:**

* [**https://www.pygame.org/news**](https://www.pygame.org/news)
* [**https://en.wikipedia.org/wiki/Pok%C3%A9mon\_(video\_game\_series)**](https://en.wikipedia.org/wiki/Pok%C3%A9mon_(video_game_series))