1. Technical Documentation

- **Mechanic Name:** Pacman 3D - Aktywowanie przedmiotów
- **Mechanic Summary:** Gracz może zebrać jeden z przedmiotów dostępnych na mapie. Przedmioty zapewniają specjalne, tymczasowe umiejętności – np. zwiększenie prędkości poruszania się lub teleportację. Gracz może nosić tylko jeden przedmiot naraz i aktywuje go ręcznie klawiszem E.
- **Prototype Overview (Design Pillars):** Pacman od zawsze opierał się na Zręczności, Taktyce i znajomości zasad rządzących grą. System przedmiotów umożliwi rozwój wszystkich tych trzech kategorii, a sama mechanika aktywacji jest jedną z głównych części tego systemu.

---

2. Implementation Notes

- **Code/Blueprint Overview:** Gracz ma 1 slot na przedmiot, który jest aktywowany klawiszem “E”. Jeśli postać gracza stanie na ikonce przedmiotu na mapie, przedmiot zostanie podniesiony, a aktualnie trzymany przedmiot pojawi się obok gracza na mapie. Aktualnie trzymany przedmiot ma swoją reprezentację graficzną w GUI (prawy dolny róg), ikona zmienia jasność w zależności czy przedmiot można aktywować czy jest na cooldownie.
- **Key Functions and Events:** Aktywacja przyciskiem. Podniesienie przedmiotu poprzez wejście z nim w kolizję na mapie. Warunki aktywowania przedmiotu (np. teleportacja - miejsce w którym postać ma się pojawić musi sprawdzać czy nie zachodzi kolizja z obiektami w grze i w odpowiedni sposób umożliwić lub nie zajście akcji)
- **Modular Components:** Dodawanie przedmiotów polega na stworzeniu 3 obiektów.
  1. Skryptu który dziedziczy po abstrakcyjnej klasie Item (z racji prototypowego bałaganu trzeba też pamiętać o ręcznym zaimplementowaniu w metodzie Use() aktywacji Cooldownu podczas użycia)
  2. GameObject’u zawierającego skrypt przedmiotu (zwany dalej Item), (improwizowane rozwiązanie, które umożliwiło nie stosowanie bazy danych wszystkich przedmiotów - należy pamiętać, że skrypt musi mieć ręcznie podpięty materiał, ustawiony czas cooldownu, prefab odpowiadającemu mu item puppet (game object reprezentujący go na mapie) i indywidualne dla każdego przedmiotu parametry, których wymaga jego logika np. odległość teleportacji)
  3. Game Object’u Item Puppet (temu z kolei ręcznie przypisujemy gameobject Itemu z podpunktu 2 - jak już pisałem prototypowe rozwiązanie omijające bazy danych) jeśli zapomnimy ręcznie podpiąć materiał w item puppet to i tak zostanie załadowany z (gameobject) Item.
  4. Co do wyświetlania przedmiotów w GUI, wystarczy użyć prefabu Canvas’u, on już będzie wykrywał sam przedmioty
  5. Skrypt Player, jest Singletonem i ma zbyt dużo funkcjonalności jak na swoją nazwę - w skrócie system sterowania, aktualizacji UI jak i podnoszenia/upuszczania przedmiotów jest w nim zaimplementowany lub z niego korzysta. Nie jest niszczony przy zmianie scen (więc możliwe, że ostatni przedmiot który podniesiemy przeniesie

się z nami na następny poziom, lub poleci nam null przy przechodzeniu na inną scenę [*])

---

3. List of Files

- **Files:** Myślę, że to pole pokrywa się z tym co napisałem powyżej.
  Projekt wykorzystuje: 6 skryptów w tym
  - 2 przedmioty
  - Abstrakcyjną klasę Item
  - bazową klasę do tworzenia ItemPuppet (skrypt wymagany w prefabach)
  - ItemUI - odpowiadający za aktualizacją GUI
  - Player - “Unholy amalgamation of many functions” obsługa sterowania, singleton, obsługa komunikacji pomiędzy obiektami w grze odnoszącymi sięd o systemu przedmiotów.

  Prefaby do systemu przedmiotów:
  - Item Puppet - przygotowany wzorzec, w którym mamy do podpięcia tylko materiał i (gameobject) przedmiotu, który mu odpowiada
  - (gameobject) Item - pusty game object do którego podpinamy Skrypt z logiką przedmiotu - w podpiętym skrypcie uzupełniamy zmienne z poziomu inspektora - cały gameobject wrzucamy jako prefab z powrotem do assetów
  - 2 przykładowe przedmioty ([1 skrypt, 1 (gameobject) Item, 1 Item Puppet] - na przedmiot) łącznie 6 plików
  - prefab Canvasu z UI i podpiętym skryptem do aktualizacji obrazka przedmoitu
  - przykładowa scena z implementacją systemu