Wykonawca:

Przemysław Pyrda

**2048** – **jest to gra logiczna stworzona przez włoskiego programistę Gabrielego Cirulli.**

**Celem gry jest takie przesuwanie i łączenie pojawiających się na planszy klocków, aby utworzyć klocek z liczbą 2048. Rozgrywka 2048 odbywa się na planszy podzielonej na kratki o wymiarach 4×4. Na planszy pojawiają się klocki z numerami, które przesuwają się w kierunku, który wskazał gracz. Poruszają się, dopóki nie natrafią na koniec planszy lub inną przeszkodę. Po każdym ruchu na wolnym polu pojawia się jeden klocek o wartości 2 lub 4. Jeżeli dwa klocki o tej samej wartości połączą się w czasie ruchu, stworzą jeden klocek o wartości ich sumy. Zsumowany klocek nie może się już łączyć z innymi w tym samym ruchu.**

**Gracz odnosi zwycięstwo, gdy uda mu się utworzyć klocek o wartości 2048. Po jego zdobyciu gracz może kontynuować grę, aby zdobyć więcej punktów. Gra kończy się, gdy na planszy nie ma wolnych miejsc i nie ma żadnych klocków, które mogłyby się połączyć w jeden w kolejnym ruchu.**



Plan projektu:

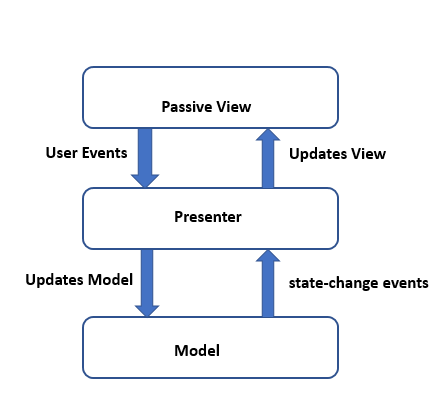
1. Przygotowanie szablonu planszy
2. Przygotowanie klas
3. Stworzenie logiki gry
4. Utworzenie kafelków z liczbami i renderowanie ich na planszy
5. Zaimplementowanie funkcjonalności, zachowania kafelków(łączenie się lub zatrzymywanie koło innych bądź blisko krawędzi planszy )
6. Implementacja sterowania kafelkami (przesuwanie góra, dół, lewo, prawo)
7. Stworzenie przejrzystego układu graficznego
8. Dodanie warunków zakończenia gry
9. Dodanie punktacji
10. Przygotowanie testów

Stos technologiczny:

Język programowania java

Środowisko android studio

Architektura MVP



1. **Model** – jest warstwą zarządzania danymi. Do obowiązków modeli wchodzą: korzystanie z API, buforowanie danych, zarządzanie bazami danych etc.
2. **View** – jest warstwą, która wyświetla dane i reaguje na działania użytkownika. W systemie Android to może być *Activity*, *Fragment*, custom-owe *View* lub *Dialog*.
3. **Presenter** – jest warstwą pośrednikiem pomiędzy *View* a model. Presenter **zawiera całą waszą logikę**, czyli komunikowanie się *View* z model, działania w tle aplikacji, przetwarzanie danych do wyświetlenia we *View*