Maciej Bogusławski, Hubert Kaczyński, Amadeusz Lewandowski, Bartosz Żelazko

PAP 2024Z ZESPÓŁ 16 ETAP 2

Gra strategiczna „Habemus Rex!”

Link do filmu prezentującego prototyp

Film przedstawia instalację oraz uruchomienie gry w środowisku Linuxowym oraz prezentuje zaimplementowane funkcjonalności.

https://www.youtube.com/watch?v=NOt--hrCztk

Realizacja wymagań funkcjonalnych

Spośród 20 pierwotnych wymagań funkcjonalnych rozpoczętych zostało 16, a 7 wymagań zostało w pełni wdrożonych.

Wymaganie	Stan	Co zostało zrobione?
Implementacja funkcjonalności interaktywnej mapy	100%	Mapa regionu, możliwość przesuwania i przybliżania/oddalania mapy, zjawiska pogodowe, blokowanie wyjścia poza mapę, reprezentacja pól na mapie, wybór pól na mapie
Implementacja ekranu startowego gry	100%	Ekran startowy otwierający grę, możliwość startu gry lub natychmiastowego wyjścia
Implementacja mechaniki pól	100%	Wczytywanie nazwy, opisu i obrazu wybranego miasta, wczytywanie budynków wybudowanych w danej sesji gry dla danego miasta oraz ich ikon
Implementacja funkcjonalności wizualnego menu budowy budynków	100%	Lista budynków dostępnych w grze wraz z ich ikonami, nazwami, opisami oraz statystykami, interaktywny przycisk informujący o możliwości wybudowania danego budynku lub o istnieniu danego budynku w danym mieście, natychmiastowe odświeżanie listy budynków po wybudowaniu budynku
Implementacja wizualnych elementów odpowiedzialnych za prezentację statystyk	100%	Pasek zasobów z ikonami reprezentującymi dane surowce oraz informacjami o obecnym stanie surowców i turowym balansie ich przychodów i wydatków, elementy prezentujące statystyki budynków oraz jednostek, przyciski budowania dopasowujące się do obecnej możliwości zakupienia budynku przez gracza, możliwość podglądu opisu budynków po najechaniu na nie myszą
Implementacja funkcjonalności wizualnego menu rekrutowania jednostek	100%	Menu armii wywoływane z paska zasobów, podgląd ikon jednostek, ich nazw, opisów, kosztu, utrzymania, ataku, obrony, przycisk umożliwiający rekrutację jednostek, informacja o posiadaniu jednostek danego typu w armii
Kreacja oprawy graficznej rozgrywki	100%	Ikony, miniatury i obrazy, tło ekranu startowego, animacja pogody na mapie, animacja przycisku następnej tury, animacje przycisków po najechaniu myszą, graficzne wykończenia pasków, menu i paneli, dobór czcionek, efekty cienia i poświaty wokół tekstu oraz obrazów
Implementacja funkcjonalności zapisu i odczytu stanu gry	80%	Pełna implementacja klasy wczytującej i zapisującej dane dynamiczne do bazy oraz odpowiednio filtrującej i zwracającej potrzebne dane w zależności od wartości parametrów - brak stuprocentowego wykonania wynika z niedokończenia implementacji pozostałych wymagań (np. walki)
Implementacja logiki budynków	70%	Dodanie budynków dostępnych w grze wraz z ich statystykami, nazwami, opisami i miniaturami do bazy danych, filtrowanie wybudowanych na danym terytorium budynków w panelu miasta oraz panelu konstrukcji budynków, mechanika poboru zasobów gracza po wybudowaniu budynku, mechanika analizowania możliwości wybudowania budynku na danym terytorium
Implementacja logiki jednostek bojowych	50%	Dodanie jednostek bojowych dostępnych w grze wraz z ich statystykami, nazwami, opisami, analizowanie posiadania przez gracza armii danej jednostek oraz możliwości rekrutacji nowych jednostek bojowych
Implementacja logiki zasobów	40%	Dodanie zasobów dostępnych w grze wraz z ich nazwami, opisami i ikonami do bazy danych oraz tabel wyznaczających koszt, produkcję oraz utrzymanie budynków i jednostek, modyfikowanie stanu zasobów
Implementacja mechaniki upływu czasu	30%	Licznik tur, przycisk następnej tury, mechanizm odświeżającego się zegara systemowego
Implementacja funkcjonalności związanych z turowością gry	20%	Dodanie tabel wyznaczających koszt, produkcję i utrzymanie budynków i jednostkę co turę do bazy danych, przechowywanie informacji o obecnej turze, stworzenie klasy filtrującej koszt, produkcję i utrzymanie danego budynku
Implementacja logiki działania przeciwników	10%	Dodanie pojęcia frakcji oraz wirtualnych graczy w formie tabel do bazy danych
Implementacja funkcjonalności odpowiedzialnej za logikę walki	10%	Dodanie jednostek bojowych dostępnych w grze wraz ze statystykami ich siły i zdrowia do bazy danych, implementacja mechanizmów kontrolujących liczność jednostek danego typu w armii
Implementacja reagowania na zdarzenia awaryjne	10%	Kontrolowanie możliwości zakupienia danego budynku przez gracza poprzez porównywanie kosztów budynku i aktualnych zasobów gracza
Implementacja funkcjonalności wizualnego przedst. walki	0%	-
Implementacja mechaniki powiadomień	0%	-
Stworzenie mechaniki punktacji oraz zwycięstwa	0%	-
Stworzenie ekranu pomocy	0%	-

Lista wymagań funkcjonalnych wraz z ich pełnymi opisami znajduje się w katalogu docs w ramach sprawozdania z Etapu I projektu.

W ramach implementacji funkcjonalności dodatkowych wykraczających poza powyższe pierwotne wymagania funkcjonalne, w ramach Etapu II stworzono również udźwiękowienie gry (w postaci soundtracku i efektów dźwiękowych) oraz utworzono mechanizm API, pozwalający na zdalny dostęp do bazy danych gry, umożliwiając w ten sposób obsługę wielu użytkowników jednocześnie. Kod mechanizmu API znajduje się w branchu „server”. Połączenie API z kodem źródłowym gry nie zostało wdrożone w ramach etapu II i stanowić będzie przedmiot dalszych prac.

Opis zrealizowanego prototypu

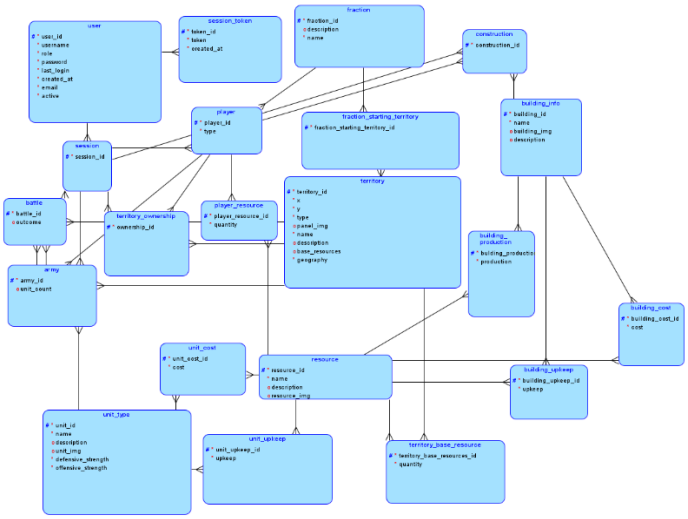
Gra „Habemus Rex!” stworzona została na bazie technologii Java 21 oraz JavaFX w wersji 21.0.5. W celu uruchomienia bazy danych, ustanowienia z nią połączenia oraz interakcji z bazą gra korzysta z narzędzia Docker oraz biblioteki Hibernate. W celu automatyzacji budowania projektu gra korzysta również z narzędzia Maven.

Kod gry znajduje się w branchu „game”. W pakiecie aplikacji w folderze cachedData znajdują klasy odpowiedzialne za wczytywanie i zapisywanie danych statycznych i dynamicznych gry. W folderze entities znajdują się klasy ORM, odpowiedzialne za mapowanie elementów gry, umożliwiając interakcje z bazą danych. W folderze media znajduje się klasa odpowiedzialna za elementy audio. W folderze ui znajdują się klasy reprezentujące elementy interfejsu graficznego gry. Klasa Main odpowiada za uruchomienie gry.

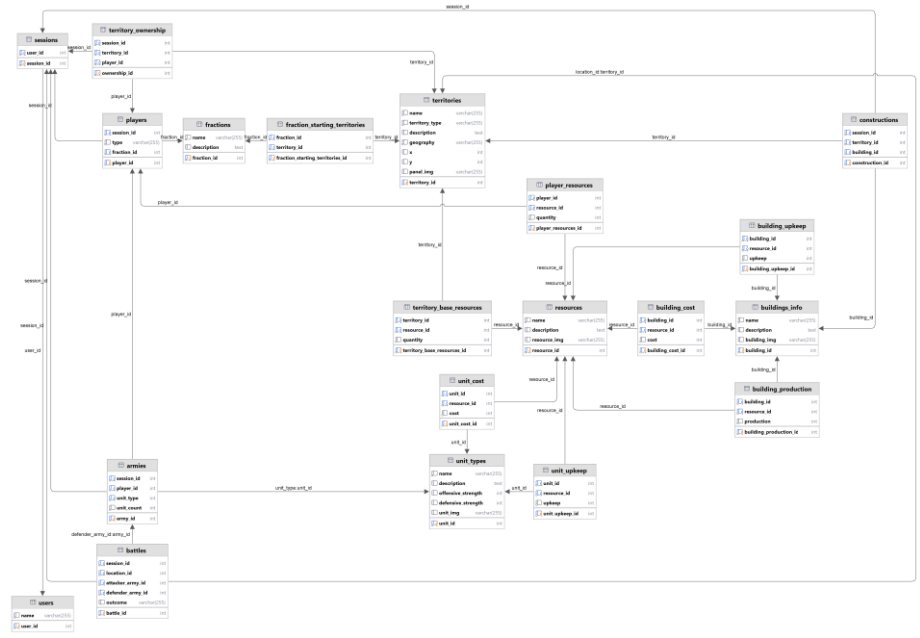
Plik .jar w branchu main skompilowany został tak, aby do jego uruchomienia nie było potrzebne manualne zainstalowanie biblioteki JavaFX. Ten sam plik gry uruchomić można zarówno w systemie Windows, jak i systemie Linux. Preferowanym sposobem instalacji i uruchomienia jest uruchomienie skryptu z katalogu głównego – sposób ten został przedstawiony w filmie prezentującym prototyp oraz w README w katalogu głównym.

Schemat bazy danych

Baza danych gry „Habemus Rex!” składa się z 20 tabel. Poniżej znajduje się diagram ER.



Poniżej znajduje się diagram fizyczny.



Oba diagramy znajdują się w pełnej rozdzielczości w katalogu docs/etap II w branchu main.