Maciej Bogusławski, Hubert Kaczyński, Amadeusz Lewandowski, Bartosz Żelazko

BD1 2024Z ZESPÓŁ 03

Gra strategiczna „Habemus Rex!”

Projekt stanowi implementację desktopowej gry ekonomiczno-strategicznej "Habemus Rex!". Gra polega na zarządzaniu państwem w roli XVIII-wiecznego króla elekcyjnego poprzez budowę infrastruktury, rekrutowanie jednostek wojskowych, podbijanie nowych ziem i sprawne gospodarowanie zasobami.

Gra oparta jest o technologie Java oraz JavaFX. Baza danych została stworzona przy użyciu technologii MySQL.

Film prezentujący prototyp

Poniższy link prowadzi do filmu pokazujący wczesne etapy pracy nad naszym projektem (w rapach etapu II projektu z przedmiotu PAP), klarownie przekazujący zamysł i założenia gry.

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

Opis bazy danych

Przygotowana przez nas baza danych pozwala na efektywne zarządzanie systemem gry „Habemus Rex!”. Baza danych odpowiada za przechowywanie kluczowych informacji, zmienianych dynamicznie bądź będących stałymi parametrami kształtującymi charakter rozgrywki.

Tabela users przechowuje dane użytkowników. Zawiera unikalny identyfikator (user_id) oraz nazwę (name).

Tabela sessions reprezentuje sesje gry powiązane z użytkownikami. Każda sesja jest przypisana do jednego użytkownika.

Tabela fractions opisuje frakcje dostępne w grze. Zawiera nazwę, opis oraz unikalny identyfikator (fraction_id).

Tabela players łączy graczy (zarówno ludzkich, jak i wirtualnych) z sesjami. Określa typ gracza (USER/AI) oraz przynależność do frakcji.

Tabela territories przechowuje informacje o terytoriach (np. miasta), w tym ich typ, geografię, współrzędne (x, y) oraz obraz panelu.

Tabela fraction_starting_territories definiuje początkowe terytoria przypisane do każdej frakcji. Łączy frakcje z terytoriami.

Tabela territory_ownership wskazuje aktualnych właścicieli terytoriów w ramach sesji.

Tabela buildings_info opisuje typy budynków, ich nazwy, opisy oraz obrazy używane w grze.

Tabela constructions rejestruje budynki wzniesione na konkretnych terytoriach w ramach sesji.

Tabela resources zawiera listę zasobów (np. złoto, drewno) używanych w grze, wraz z opisami i obrazami.

Tabela territory_base_resources określa podstawowe zasoby dostępne na terytoriach (np. ilość złota w Warszawie).

Tabele building_cost, building_production i building_upkeep stanowią tabele pomocnicze dla budynków, pokazując koszty budowy, produkcję zasobów oraz koszty utrzymania budynków.

Tabela player_resources przechowuje ilość zasobów posiadanych przez każdego gracza.

Tabela unit_types opisuje typy jednostek wojskowych (np. piechota, husaria), ich statystyki bojowe oraz obrazy.

Tabele unit_cost i unit_upkeep stanowią tabele pomocnicze dla jednostek, pokazując koszty rekrutacji oraz ich koszty utrzymania.

Tabela armies pokazuje jednostki wojskowe należące do graczy w ramach sesji (typ jednostki i ich liczba).

Tabela battles rejestruje informacje o bitwach – lokalizację, uczestników bitwy (atakujący/obrońcy) oraz wynik.

Tabela logs przechowuje logi zmian tabel dynamicznych w bazie danych (np. dodanie/usunięcie rekordu) wraz ze znacznikiem czasu.

Opis skryptów

Główne skrypty implementujące strukturę bazy danych, wypełnianie bazy danymi oraz wyzwalacze, procedury, funkcje, kursory oraz skrypty testujące znajdują się w folderze BD1 w branchu main.

Plik database_creation.sql definiuje strukturę bazy danych (tabele, relacje, klucze obce).

Plik database_fill.sql wypełnia bazę danymi, z których korzysta gra.

Plik show_rich_players.sql stanowi implementację kwerendy z kursorem, która pokazuje graczy, którzy mają więcej danego zasobu niż wynosi średnia w sesji.

Plik adjust_building_costs.sql stanowi implementację procedury AdjustBuildingCosts, która automatycznie zwiększa koszty budynków, jeśli ponad 33% graczy w sesji posiada dany budynek.

Plik session_cursor.sql zawiera procedurę wraz z kursorem umożliwiającą usunięcie danej ilości nadmiarowych sesji gry dla wszystkich użytkowników oraz testy do tejże procedury.

Plik log_triggers.sql zawiera triggery, które przy usuwaniu, dodawaniu lub zmienianiu danych dynamicznych dodaje informacje o tych działaniach do tabeli log i krótki test. Na większą skalę można je przetestować używając pliku do ładowania danych do pliku po wgraniu tych triggerów do bazy.

W pliku resource_costs_management.sql znajdują się dwie procedury i funkcja oraz skrypty do ich testowania. Funkcja calculate_player_net_income przyjmuje jako argumenty id gracza i surowca, a następnie zwraca wypadkową wszystkich kosztów i przychodów dla tego gracza i surowca. Procedura remove_unmaintainable_buildings przyjmuje jako argumenty id gracza i surowca, a następnie jeśli dochód danego surowca jest ujemny, to usuwa danemu graczu budynki aż się skończą lub wyjdzie na plus. Procedura remove_unmaintainable_units działa podobnie jak poprzednia procedura, tylko zamiast budynków usuwa jednostki. Instrukcje SELECT obecne w pliku odpowiadają za testowanie kodu.

W pliku army_triggers_and_functions.sql znajdują się trigger i funkcje związane z zarządzaniem jednostkami i bitwami. W pliku tym znajduje się trigger, który zmienia przynależność terytorium po wygranej bitwie. Dodatkowo zostały zaimplementowane zostały funkcje które obliczają siły ofensywne i defensywne dla graczy a także wyniki bitew. W pliku tym znajdują się także testy wyżej wymienionych elementów.

W pliku army_strength_size_selects.sql znajdują się zapytania select testujące elementy bazodanowe odpowiedzialne za jednostki.

Plik army_cost_selects.sql zawiera testy do bazy danych zawierających różne selecty odnoszące się do graczy, ich zasobów oraz jednostek i ich kosztów.

Historia commitów tworzonych skryptów znajduje się w branchu game w folderze HabemusRex/sql.

Szczegóły techniczne aplikacji w Javie

Dogłębne szczegóły dotyczące założeń oraz implementacji projektu od strony programowania w Javie znajdują się w sprawozdaniach w folderze docs. Sprawozdania te zostały utworzone w ramach realizacji projektu z przedmiotu Programowanie Aplikacyjne.

W folderze src/test/java na branchu game znajdują się liczne testy jednostkowe odnośnie współpracy gry z bazą danych na poziomie Javy przy użyciu technologii Hibernate.

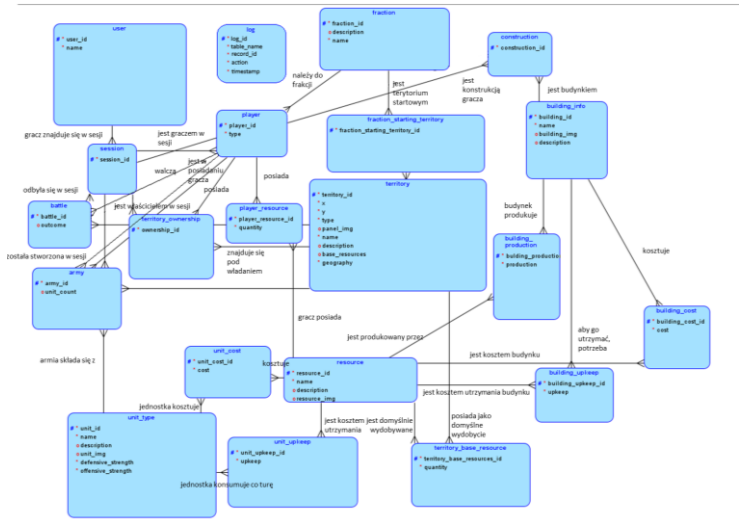
Instalacja i uruchomienie

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”.

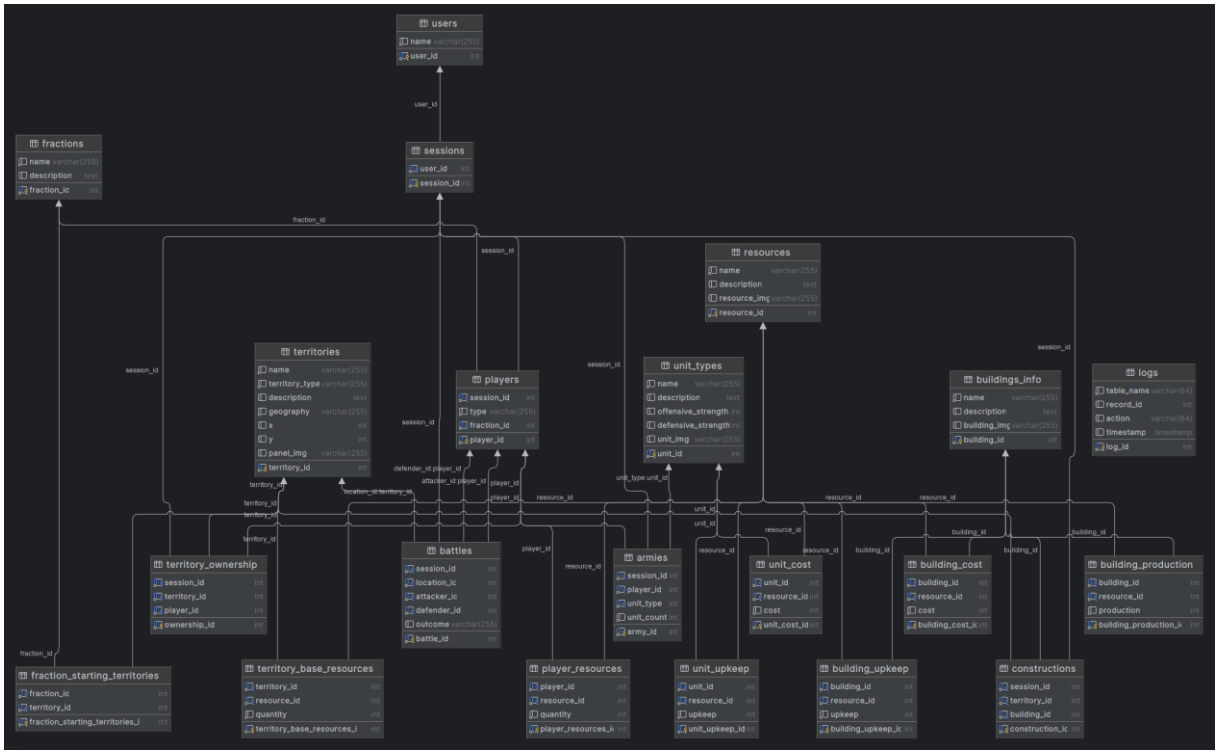
Plik .jar w branchu main skompilowany został tak, aby do jego uruchomienia nie było potrzebne manualne zainstalowanie biblioteki JavaFX. Ten sam plik gry uruchomić można zarówno w systemie Windows, jak i systemie Linux. Preferowanym sposobem instalacji i uruchomienia jest uruchomienie skryptu z katalogu 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 21 tabel. Poniżej znajduje się diagram ER.



Poniżej znajduje się diagram fizyczny.



Oba diagramy znajdują się w pełnej rozdzielczości w katalogu BD1 w branchu main.