**Dokument wymagań projektowych**

**Nazwa projektu: MoviePocket**

**Autorzy: Danila Prymak, Aliaksandr Trafimchyk, Anton Pazniak**

**Data: 27.04.23**

**0. Wersje dokumentu**

ver. 0 - 27.04.23

ver 0.1 - 18.05.23

- Volha Andrawa została wrócona do zespołu

- dostosowanie dokumentu do uwag komisji

**1. Elementy składowe projektu (produkty projektu)**

Naszą główną ideą jest stworzenie miejsca, gdzie ludzie będą mogli wyrazić swoją miłość do filmów oraz dzielić się tą miłością z innymi użytkownikami. Nasz serwis pozwoli ludziom śledzić ich historię oglądanych filmów, tworzyć własne listy ulubionych filmów według ich preferencji i udostępniać je innym, aby mogli odkrywać różne gusta ludzi w filmach. Ważne, aby powiedzieć, że użytkownicy nie będą mogli oglądać filmów na naszej platformie, co naszym zdaniem pomaga utrzymać naszą platformę tak Towarzyską, jak to tylko możliwe, co oznacza zmniejszenie rozpraszania uwagi od innych funkcji, które nie są z tym połączone. Planujemy również dodać moduł review. Ludzie będą mogli ocenić film i zostawić krótką recenzję na jego temat. Grupą docelową naszej strony internetowej będą osoby, które chcą połączyć się ze społecznością wokół swoich ulubionych filmów z różnych gatunków i mieć narzędzie, które pomoże zorganizować własną kolekcję filmów w łatwy sposób.

Elementy programistyczne:

* Back-end: Java/Spring
* Front-end: JS/React
* Tests: JEST for React, JUnit 4 for Spring
* ORM: Hibernate
* Baza danych
  + Mysql

Elementy nieprogramistyczne:

* Dokumentacja wizji projektu
* Dokumentacja wymagań projektowych
* Dokumentacja dla użytkownika
* Dokumentacja deweloperska
* Licencja i podział praw własności
* Wymagania funkcjonalne
* UI design (Figma)

**2. Granice projektu**

* **Brak możliwości oglądania filmów na stronie**

Z powodu naruszenia praw własności i trudności z osadzeniem funkcji oglądania nie jest możliwe obejrzenie samego filmu / pokazu na stronie internetowej. Z drugiej strony to pomoże skoncentrować się na głównym celu usługi naszego serwisu, którym jest udostępnianie kolekcji filmów innym osobom, a także koncentrowanie na aspektach społecznych naszej strony.

* **Połączenie z API TMDB**

Niektóre kluczowe funkcje, takie jak baza filmów i programów telewizyjnych, będą pobierane za pomocą odwołania do API TMDB, co sprawia, że nasza usługa jest w jakiś sposób niezawodna od usługi TMDB. Jeśli wystąpią problemy z TMDB, to spowoduje to problemy z MoviePocket.

* **Język interfejsu**

Głównym językiem interfejsu naszego serwisu jest angielski ze względu na fakt, że dane filmowe będą pobierane z angielskiego TMDb.

**3. Lista wymagań funkcjonalnych**

**Użytkownik**

1. Uwierzytelnienie:
   1. Rejestracja: Odwiedzający będzie mógł stworzyć swoje konto (potrzebna poczta).
   2. Logowanie: Odwiedzający będzie mógł zalogować się do swojego konta.
   3. Wylogowanie: Użytkownik będzie mógł wylogować się.
   4. Zmiana hasła: Użytkownik będzie mógł zmienić hasło.
   5. Zapomnienie hasła: Użytkownik będzie mógł odzyskać konto, jeśli zapomniał hasła potrzebna poczta).
   6. Możliwość otrzymania maila z potwierdzeniem na email podany użytkownikiem
2. Zmiana awatara: Użytkownik będzie mógł zmienić awatar swojego profilu.
3. Zmiana informacji o siebie na swojej stronie: Użytkownik będzie mógł zmienić informację o siebie na swojej stronie (z pewnymi ograniczeniami).
4. Oglądanie listę użytkowników, którzy cię obserwują i odwrotnie.
5. Obserwacja (Following) użytkownika: Użytkownik może obserwować wybranego użytkownika, żeby śledzić za jego aktywnościami.
6. Usunięcie konta: Użytkownik będzie mógł usunąć swoje konto.

**Lista Filmów**

1. Utworzenie listy: każdy użytkownik może utworzyć więcej niż jedną listę z krótkim opisem i dodać tam filmy (wizualna część listy jest opcjonalna).
2. Dodawanie filmu do listy: użytkownik może dodawać filmy do listy.
3. Edycja listy: użytkownik może dodawać lub usuwać filmy do już istniejącej listy.
4. Lista prywatna lub publiczna: użytkownik może wybrać, czy lista będzie widoczna na stronie użytkownika, czy nie (prywatna).
5. Sortowanie listy: lista może być posortowana według wybranych kryteriów, takich jak data wydania, tytuł lub ocena.
6. Tworzenie listy ulubionych filmów: każdy użytkownik ma domyślną listę ulubionych, na której może dodawać filmy bezpośrednio ze strony filmu.

**Film**

1. Przejście do strony filmu: użytkownik może przejść do strony każdego filmu/ programu telewizyjnego.
2. Dodawanie do ulubionych: użytkownik może dodać filmy do ulubionych.
3. Ocenianie: użytkownik może ocenić film w skali 1-10.
4. Zostawienie recenzji: użytkownik może zostawić recenzję na stronie filmu, która jest widoczna dla innych.
5. Ocenianie recenzji: użytkownik może ocenić recenzję (like albo dislike).
6. Zwiastun filmu: użytkownik mogą oglądać zwiastun filmu.
7. Oznacz jako obserwowany: użytkownik może odznaczyć film jako obserwowany, który zostanie zapisany na liście obserwowanych użytkownika.
8. Przejście do strony użytkownika twórcy recenzji: użytkownik może przejść na stronę drugiego użytkownika, który zostawił swoją recenzję.
9. Przejście do strony aktora lub reżysera (jeśli istnieje): użytkownik może przejść na stronę aktora lub reżysera ze strony filmu.

**Strona aktora, reżysera**

1. Obejrzenie listy filmów, w których aktor (reżyser) uczestniczył: użytkownik może wejść do tej listy i przejść na stronę filmu.

**Post**

1. Tworzenie posta: użytkownik może tworzyć posty, w których mogą udostępniać jedną ze swoich list i krótki opis.
2. Przeczytanie posta: odwiedzający może wyświetlić post, żeby go przeczytać.
3. Edytowanie posta: użytkownik może edytować swoje posty, które mają status wersji roboczej.
4. Usuwanie posta: administrator może usuwać dowolne posty.
5. Like posta: użytkownik może dodać like do dowolnego z postów i zobaczyć liczbę like w poście.

**Strona główna**

1. Wyszukanie filmu: użytkownik może wyszukiwać filmy według tytułu na pasku wyszukiwania.
2. Obejrzenie trendów: użytkownik może oglądać listę popularnch filmów.
3. Obejrzenie popularnych postów: użytkownik może oglądać listę popularnych postów
4. Przełączenie motywu: możliwość przełączenia motywu na ciemny.

**4. Lista wymagań niefunkcjonalnych**

1. Bezpieczeństwo. Użytkownicy oraz odwiedzające nie mogę korzystać z niczego oprócz powyżej wymienionych funkcjonalności systemu.
2. Bezpieczeństwo zewnętrzne. System powinien być zabezpieczony przed różnymi rodzajami atak oraz zabezpieczona ochrona danych użytkownika.
3. Możliwość otrzymywania informacji od stron trzecich. Na przykład odnośnie reklamy albo dotyczące praw własności.
4. Dostosowanie strony do różnych urządzeń mobilnych. coraz więcej użytkowników korzysta z internetu za pomocą urządzeń mobilnych. Konieczne jest zapewnienie responsywnego projektu strony, który będzie działał sprawnie na różnych urządzeniach, niezależnie od ich rozmiaru i orientacji ekranu.

**5. Kryteria akceptacji projektu dla I semestru prac**

* **Wymagane**
  + Back-end API prototyp
  + UI prototyp dla zakończonych modułów
  + Uwierzytelnienie użytkownika z możliwością potwierdzenia email
  + Moduł strony głównej
  + Moduł Oceniania filmów
  + Moduł Filmu (strona filmów z informacją)
  + integracja bazy danych, odczyt, zapis i odpytywanie danych dla każdego ukończonego komponentu
  + projektowanie i implementacja modułu bazy danych Filmów
  + Funkcje edycji i usuwania konta użytkownika
* **Oczekiwane (Może zostać przeniesiony na drugi semestr lub pominięty)**
  + Funkcjonalność dodawania do ulubionych filmów
  + Moduł Recenzji filmów
  + Tworzenie konta użytkownika u zewnętrznych dostawców OAuth
  + Unit-testy
* **wymagane** – kryteria wymagane muszą być spełnione, aby produkt mógł być odebrany przez klienta (np. dostarczenie prototypu, wdrożenie odpowiednich funkcjonalności, pozytywnie zakończone testy obciążeniowe, zrozumienie procesów zachodzących u klienta), bez ich spełnienia projekt nie może zostać dopuszczony do obrony.
* **oczekiwane –** kryteria oczekiwane przez klienta, pozwalające na dalszy rozwój systemu i na jego potencjalne wdrożenie (np. integracja z wewnętrznym systemem autoryzacji klienta, wdrożenie odpowiednich funkcjonalności).

**6. Mierzalne wskaźniki wdrożeniowe**

* **Aplikacja będzie publicznie dostępna na domenie**
* Liczba użytkowników przekroczy 15 osób
* Produkt będzie zawierał system zbierania filmów w listy i system udostępniania listów pomiędzy użytkownikami. Produkt będzie używany przez nasi wewnętrzni testerzy.
* System będzie skończony ze względu na UI-design
* Każdy użytkownik będzie miał min. 2 listy filmów oraz 3 posta
* Min. Na 10 filmach będą widoczne recenzji (min. 2)

**7. Kryteria akceptacji projektu dla II semestru prac**

* **Wymagane**
  + Funkcjonalność postu(dodawanie, usuwanie, edycja, liking)
  + Zakończona funkcjonalność listów filmów(system sortowania)
  + Funkcjonalność modułu Review
  + Administracja treścią i użytkownikami
  + Gotowy interfejs UI dla każdego modułu
  + Udostępnienie serwisu na publicznym domenie
  + Pełna wersja robocza strony internetowej
  + Dark theme
  + Unit-testy
  + Funkcjonalność dodawania do ulubionych filmów
* **Oczekiwane (Te funkcje mogą być dodawane lub nie))**
  + Funkcjonalność związana ze stronami aktora, reżysera
  + Funkcjonalność wyszukiwania użytkowników, postów
  + Review voting system oraz możliwość sortowania review
  + Avatary użytkownika
  + Implementacja automatyzacji procesów (DevOps)

**8. Organizacja pracy zespołu**

* Danila Prymak
  + Backend, tester
* Aliaksandr Trafimchyk
  + UI design, frontend, tester
* Anton Pazniak
  + Backend, tester

Każda rola będzie zawierać następujące obowiązki:

* Backend developer
  + tworzy, utrzymuje i nadzoruje
* baza danych
* Punkty końcowe REST API
* Logika biznesowa
* Autoryzacja
* odpowiada za Continuous Integration and Delivery
* Frontend developer
  + implementuje projekt stworzony przez projektantów UI
* Angażuje się w dyskurs z projektantami UI, aby osiągnąć równowagę między łatwością wdrożenia, atrakcyjności wizualnej i dostępności
* UI designer
* tworzy projekt dla każdej strony / widoku aplikacji
* angażuje się w dyskurs z programistami frontend, jak zdefiniowano powyżej
* Tester
  + Tworzy unit testy
    - Frontend tester pisze testy, które weryfikują niezmienniki danego przepływu UX
    - Backend tester pisze testy, które sprawdzają integralność danych i logikę biznesową
* tworzy testy End-to-End
  + Razem tworzą testy, które integrują dwie domeny (backend i frontend) poprzez emulację rzeczywistego użytkownika klikając przyciski

**Metoda pracy**

Zastosujemy metodę hybrydową, aby spełnić swoje specyficzne wymagania, wybierając elementy z różnych metod.

Użyjemy tak zwanej mieszanki **metodologii Scrum i Kanban**, aby uzyskać potrzebne zalety od obu z nich i zachować elastyczny przepływ pracy w ramach projektu.

Będziemy mieli 2 główne terminy dostarczenia Oprogramowania (Semestr 1 i 2), każdy z których będzie podzielona na min. **2 przyrosty funkcjalności (sprint).**

Podobnie jak w Scrum będziemy mieli predefiniowaną listę funkcjonalności (wymagane kryteria akceptacji projektu na semestr), które planujemy wdrożyć dla każdego sprintu z priorytetem odpowiednio do znaczenia (wymaganego i oczekiwanego) i podzielonego na historie użytkowników z podzadaniami do zrobienia, ale użyjemy Kanban Just-In-Time aproach, aby zwiększyć elastyczność w zakresie zmiany priorytetów lub dodać inne zadania zgodnie z aktualną sytuacją w sprincie.

**Nie będziemy mieć Scrum mastera**, ponieważ każdy członek zespołu będzie mógł wybierać zadania i zmieniać przepływ pracy, który jest omawiany i zatwierdzany przez wszystkich na **cotygodniowych spotkaniach** (stały kontakt między członkami będzie utrzymywany przy użyciu platform komunikacyjnych jako Teams lub Telegram).

Jako narzędzie do śledzenia zadań użyjemy **tablicy Kanban z zadaniami To-do, In progress i Done** dla każdego sprintu z omawianym WIP (Work in progress) dozwolonym dla członka zespołu z **nie więcej niż 5** zadania z User story / Bug fix / design/ innych zadań).

Wstępnie zdefiniowaliśmy w Jira typy zadań, takie jak **Epic**, **User story** (moduły z wartością business dla projektu i dostosowany do jednego z Epic), **Task** (Brak wartości business np. "Dodaj komentarze do kodu"), **Bugs** (połączone lub wymienne z Github issues)i **Subtasks** zarówno dla stories, jak i zadań.

**Cykl życia zadania:**

* Najpierw pomysł jest omawiany wśród członków zespołu, a następnie dodawany do Jiry do kolumny TODO jako historia użytkownika lub typ zadania z opisem i przypisaną do niego osobą (może to być więcej niż jedna osoba).
* Zadanie jest wykonywane w oddzielnym brunchu (czasami kilka zadań w jednym brunchu, jeśli jest to drobna rzecz) i scalane z przejrzanym przez wszystkich członków pull requestem do dev brunchu.
  + w przypadku znalezienia błędów podczas przeglądu tworzone jest osobne zgłoszenie w GitHub lub zadanie błędu w Jira.
* Po pozytywnej recenzji i poprawkach błędów w dev brunchu jest on łączony z pull requestem do main jako przyrost i koniec sprintu.

**Narzędzia wsparcia projektu**

* Github
  + Kod źródłowy i hosting kontroli wersji
* Microsoft Teams
  + Platforma dla komunikacji, spotkań i współpracy
* Microsoft Word
  + Współpracujący edytor online do pisania dokumentacji
* Jira
  + Narzędzie dla śledzenia workflow projektu oraz przydzielania zadań członkom zespołu
* Figma
  + Narzędzie dla tworzenia prototypu UI-design

**9. Ryzyka projektowe**

* Ten projekt ma duże ryzyko, że nie będzie miał duże zainteresowanie ze strony szerokiej publiczności, ponieważ istnieje wiele analogicznych serwisów.
* Problem z bazą danych - ze względu na ograniczenia naszych zasobów nie mamy możliwości samodzielnie tworzyć bazy danych z filmami. Dlatego będziemy musieli używać gotowych baz danych firm zewnętrznych.
* Problemy z prawami autorskimi i przepisami prawnymi mogą prowadzić do poważnych problemów prawnych, jeśli na stronie zostaną wykorzystane nielegalnie chronione treści.
* Koszty tworzenia i utrzymywania strony mogą być bardzo wysokie, jest to duża strona z dużą ilością informacji. Koszty te obejmują hosting, utrzymanie strony i aktualizację treści.
* Zapewnienie bezpieczeństwa użytkowników i ich danych osobowych jest jednym z najważniejszych aspektów. Należy zastosować odpowiednich narzędzi i praktyki, takich jak szyfrowanie danych i stosowanie bezpiecznych protokołów komunikacyjnych, aby chronić użytkowników przed kradzieżą danych i innymi zagrożeniami.
* Zapewnienie wystarczającej przepustowości i szybkości ładowania strony. Konieczne jest zapewnienie wystarczającej mocy obliczeniowej i przepustowości serwera oraz optymalizacja kodu strony, aby zminimalizować czas ładowania.

**10. Kamienie milowe**

**I semestr:**

* Funkcjonujący system rejestracji/logowania
* System komunikacji pomiędzy back-end i front-end
* Pierwszy interfejs stworzony w React
* Doprecyzowanie dokumentu wymagań
* Funkcjonująca baza danych wraz z możliwością dodawania i edytowania użytkowników, filmów
* Zakończona dokumentacja wizji projektu oraz wymagań
* Ukończony prototyp UI-design/figma oraz bazy danych
* Moduł uwierzytelnienia użytkownika
* Implementacja strony głównej serwisu oraz strony filmu
* Moduł Recenzyj filmów
* Moduł Oceniania filmów
* Ukończony prototyp backend REST API połonczony React frontendem odpowiadający kryterium akceptacji dla 1go semestru

**II semestr:**

* Wdrażanie aplikacji na publicznym domenie
* Skończony moduł listy filmów
* Skończony moduł postów
* Skończony moduł dodawania do ulubionych
* Dodanie dark theme
* Ukończony interfejs użytkownika dla każdego modułu z implementacją ciemnego motywu
* Wykonane wszystkie testy
* Zakończona aplikacja