Plan projektu – System Zarządzania Wypożyczalnią Filmów

# 1. Wstęp

Projekt zakłada stworzenie systemu do obsługi wypożyczeń filmów w modelu klient-serwer. Aplikacja kliencka (WPF) umożliwia operatorom dodawanie i zarządzanie wypożyczeniami, a WebAPI w ASP.NET Core odpowiada za komunikację z bazą danych MS SQL Server.

# 2. Cel projektu

- Zapewnienie prostego w obsłudze narzędzia do rejestrowania i monitorowania wypożyczeń filmowych.  
- Zwiększenie efektywności operacyjnej wypożyczalni.  
- Umożliwienie generowania raportów i statystyk działalności.

# 3. Organizacja projektu

Projekt realizowany jest w ramach pracy indywidualnej (lub zespołowej – zależnie od kontekstu uczelni), podzielony na etapy: projektowanie, implementacja, testowanie, dokumentacja.

# 4. Struktura organizacyjna

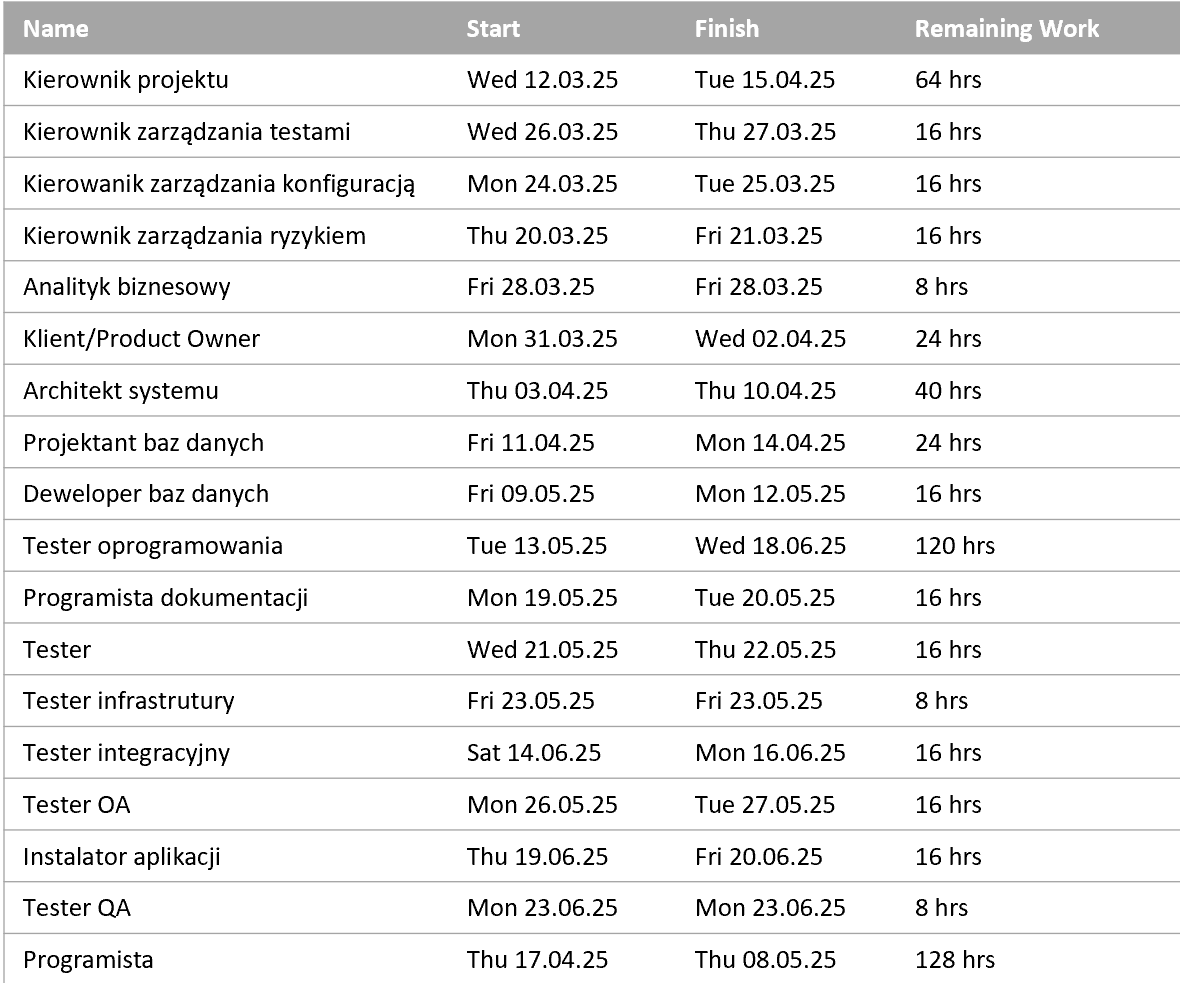
- Kierownik projektu / wykonawca: Szymon Koprowski, Kamil Semczuk  
- Konsultant merytoryczny: Opiekun dydaktyczny  
- Interesariusze: użytkownicy końcowi (pracownicy wypożyczalni)

# 5. Role i odpowiedzialność

| Rola | Zadania |
| --- | --- |
| Kierownik projektu | - Przygotowanie harmonogramu projektu  - Przygotowanie kosztorysu projektu  - Przygotowanie planu zarządzania projektem  - Akceptacja specjalizacji wymagań  - Akceptacja projektu bazy danych |
| Kierownik zarządzania ryzykiem | - Przygotowanie planu zarządzania ryzykiem |
| Kierownik zarządzania konfiguracją | - Przygotowanie planu zarządzania konfiguracją |
| Kierownik zarządzania testami | -Przygotowanie planu zarządzania testami |
| Analityk biznesowy | - Przygotowanie specyfikacji wymagań |
| Klient / Product Owner | - Akceptacja specyfikacji wymagań |
| Architekt systemu | - Przygotowanie analizy specyfikacji wymagań  - Przygotowanie projektu aplikacji |
| Projektant baz danych | - Akceptacja projektu aplikacji  - Przygotowanie projektu bazy danych |
| Programista | - Implementacja aplikacji  - Przygotowanie opisu wersji aplikacji |
| Deweloper baz danych | Implementacja bazy danych |
| Tester oprogramowania | - Wykonanie Unit testów  - Testy smoke  - Testy regresyjne  - Testy funkcjonalności uzupełniających  - Testy wydajnościowe lub bezpieczeństwa |
| Tester infrastruktury | - Przygotowanie środowiska do testowania |
| Tester OA | - Opracowanie przypadków testowych |
| Tester integracyjny | - Testy integracyjne |
| Tester QA | - Raport końcowy z testów |
| Tester | - Projektowanie testów |
| Instalator aplikacji | - Raport instalacji |
| Programista dokumentacji | - Przygotowanie opisu wersji aplikacji |

# 6. Harmonogram projektu

# 7. Kosztorys projektu



# 8. Zasoby projektu

- Sprzęt: komputer z Windows 10/11  
- Oprogramowanie:  
 - Visual Studio 2022  
 - .NET 6/7  
 - SQL Server Express  
 - OxyPlot, iTextSharp  
 - MS Word / Project (dokumentacja)

# 9. Standardy i narzędzia w projekcie

- .NET Core 6.0 (ASP.NET WebAPI)  
- Entity Framework Core 6.0  
- WPF + MVVM  
- OxyPlot 2.1  
- iTextSharp 5.5  
- Git do wersjonowania kodu  
- XML (do eksportu/importu lokalnego)

# 10. Procesy zarządzania

# 10.1 Plan zarządzania konfiguracją

- GitHub do kontroli wersji kodu źródłowego  
- Podział na repozytorium aplikacji WPF i WebAPI

# 10.2 Plan zarządzania ryzykiem

| Id | Ryzyko | Skutek | Działanie minimalizujące |
| --- | --- | --- | --- |
| R1 | Brak połączenia z API | Aplikacja nie działa poprawnie | Testy lokalne + fallback XML |
| R2 | Błędy integracji WPF ↔ API | Błędy w transmisji danych | Wspólne DTO, testy jednostkowe |
| R3 | Utrata danych przy awarii | Brak historii wypożyczeń | Backup SQL + eksporty CSV |

# 10.3 Plan zarządzania testami

- Testy ręczne UI aplikacji WPF  
- Testy endpointów WebAPI (Postman / Swagger)  
- Testy integracyjne (działanie całości po połączeniu)