MOVIE - UNIVERSE

DOKUMENTACJA PROJEKTOWA

*Projekt został przygotowany przez:*

|  |  |  |
| --- | --- | --- |
| **Imię i nazwisko** | **Numer albumu** | **Numer grupy dziekańskiej** |
| Monika Sójkowska | 178399 | KrZZIs3011Io |
| Wojciech Michalik | 176494 | KrZZIs3011Io |
| Bartłomiej Mike | 176509 | KrZZIs3011Io |

Spis treści

[Wstęp 1](#_Toc413523566)

[Założenia aplikacji 1](#_Toc413523567)

[Architektura aplikacji 1](#_Toc413523568)

[Opis technologii 2](#_Toc413523569)

[Wymagania techniczne i wdrożenie aplikacji 3](#_Toc413523570)

# Wstęp

Celem niniejszego dokumentu jest dostarczenie specyfikacji technicznej oraz funkcjonalnej dla aplikacji Movie-Universe. Dokument opisuje założenia projektu, architekturę aplikacji, wymagania techniczne, opis technologii, instrukcję instalacji i wdrożenia aplikacji oraz opis funkcjonalny.

# Założenia aplikacji

Aplikacja Movie-Universe jest aplikacją internetową działającą jako wypożyczalnia filmów on-line. Dzięki Movie-Universe użytkownik ma możliwość przeglądania bazy filmów, czy dodawania recenzji. Jej główną funkcjonalnością jest natomiast możliwość wypożyczania filmów. Aplikacja jest zintegrowana z płatnościami online. Wszystkie te cechy umożliwiają użytkownikowi szybki dostęp do ulubionych filmów bez wychodzenia z domu.

W przyszłości planuje się dodanie modułu statystyk, wyliczania trendów oraz moduł obsługi klienta z systemem powiadomień o nowościach filmowych.

# Architektura aplikacji

Aplikacja Movie-Universe jest napisana w języku skryptowym PHP przy pomocy frameworku Symfony2. Architektura aplikacji opiera się na modelu Model-View-Controller.

Model opiera się na relacyjnej bazie danych PostgreSQL. Dostęp do bazy z poziomu aplikacji został zrealizowany za pomocą technik ORM w implementacji Doctrine2. Zaprojektowane encje, będące obiektami domenowymi aplikacji, są zmapowane 1:1 w stosunku do tabel w bazie. Oprócz encji została zastosowana warstwa dostępu do danych za pomocą klas-repozytoriów.

Controller w aplikacji to moduł realizowany za pomocą „serwisów”. Jest to warstwa aplikacji odpowiadająca za logikę biznesową. Łączy ona moduł interfejsu z modułem dostępu do modelu. Warstwa serwisów korzysta z technik Dependency Injection, wykorzystując kontener wstrzykiwania zależności dostarczony z frameworkiem Symfony2.

Moduł View to moduł interfejsu. Składa się z kontrolerów widoku, schematów widoku i formularzy. Kontrolery widoku zawierają funkcje odpowiadające za obsługę zdefiniowanych ścieżek. Kontroler przechwytuje żądanie, przetwarza je korzystając z warstwy logiki biznesowej oraz zwraca odpowiedź.

# Opis technologii

* Symfony2 - framework udostępniający zestaw narzędzi i technik do budowy aplikacji webowej,
* FOSUserBundle – komponent obsługi użytkownika (rejestracja, logowanie, przypominanie haseł, itp.), autentykacji i autoryzacji,
* Doctrine2 – komponent ORM udostępniający dostęp do danych
* Bootstrap – biblioteka wykorzystana do tworzenia interfejsu graficznego stron internetowych
* GIT – system kontroli wersji, służący do rozproszonej pracy nad projektem
* Composer – menedżer zależności

# Wymagania techniczne i wdrożenie aplikacji

Wymagania:

* Serwer HTTP Apache2 + PHP 5.3.3
* PostgreSQL 9.3
* Composer
* GIT

Przygotowanie bazy danych:

1. Utworzenie nowego użytkownika bazodanowego oraz bazy danych o dowolnej nazwie, której właścicielem jest nowy użytkownik
2. Wykonanie skryptu {APP}/Documentation/DB/movie\_universe\_ddl.sql
3. Wykonanie skryptu {APP}/Documentation/DB/movie\_universe\_dml.sql

Opis instalacji aplikacji:

1. Pobranie źródeł aplikacji:

git clone <https://github.com/bmik/movie-universe.git>

1. Ustawienie w konfiguracji serwera HTTP dostępu do folderu {APP}/Sources/, gdzie {APP} to folder z pobranymi źródłami.
2. Sprawdzenie wymagań silnika Symfony2:

php app/check.php

1. Ustawienie parametrów program w pliku {APP}/Sources/app/config/parameters.yml:

* Parametry „database\_\*” ustawiamy zgodnie z danymi do nowej bazy danych
* dotpay\_id: 72890
* dotpay\_pin: 1234567887654321

1. Instalacja niezbędnych zależności:

composer install --no-dev --optimize-autoloader

Po wykonanych krokach aplikacja jest dostępna pod adresem:

http://{nazwa\_hosta}/app.php/home