W poniższym dokumencie chciałbym opisać przebieg pracy i podjęte decyzje na każdym etapie tworzenia aplikacji kina.

1. Kompromisy czasowe:
2. Nie realizuję obsługi sieci kin – skalowanie sprzedaży biletów dla każdego kina ma niewielką wartość dydaktyczną w porównaniu do kosztu czasowego zaimplementowania dodatkowych tabel baz danych dla kin oraz tabelki transferowej dla
3. Package:
4. Microsoft.EntityFrameworkCore, Microsoft.EntityFrameworkCore.SqlServer, Microsoft.EntityFrameworkCore.Tools

Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

<https://github.com/dotnet/aspnetcore/issues/35834>

<https://github.com/Microsoft/DockerTools/issues/99>

1. Functional requirements
2. Ticket reservations allowing the user to choose the movie, the time and the seats. Powinien być łatwy w obsłudze ~~i umożliwiać użytkownikom płatność online.~~
3. Repertoire schedule with movie info and screening time
4. Movie info: for each movie there will be TMDb API description, casting, director, score, reviews(?) and a trailer.
5. News and events: the Cinema should offer some miscellaneous events to be more attractive to wider audiences.
6. Movie scores system ~~i recenzji~~ (also the ones they didnt buy tickets for – TMDb also offers movie scores and if this decision leads to poor movie scored people might still be interested to check out movies with the score discrepancy from TMDb’ score. Also the scores could be used later on for machine learning recommendations), that will help other users pick the movie.
7. Newsletter system for logged users (not to overwhelm the new to the site) that lets the user keep track of what’s new in the Cinema and in the repertoire
8. Time constraint compromises:
9. Reviews dont bring much educational value to this project so they get skipped
10. Online payment system with Stripe could be very difficult to do on my setup considering the difficulties i had with security certificates dotnet dev-certs on my pc
11. Recommendation system that relies on users’ movie gradings and heavily on their ticket purchases would be fun with machine learning but i will skip it for now
12. Storing images would be cool to do on the cloud but learning AWS feels like a bit much rn.
13. App users:
14. People curious about what the Cinema offers
15. People interested in buying the tickets
16. An admin updating the schedule and the repertoire
17. UML usecase diagram:

UML usecase diagram for the Cinema

Obraz zawierający diagram

Opis wygenerowany automatycznie

1. Database