

The background is a gradient from dark purple at the top to dark blue at the bottom, speckled with small white dots. On the left side, there are several concentric circles and a large arc with a scale from 140 to 260. The scale is marked with numbers every 10 units (140, 150, 160, 170, 180, 190, 200, 210, 220, 230, 240, 250, 260). There are also smaller circles and arrows scattered across the background.

2SEEORNOT

BILHETEIRA DE CINEMAS

PEDRO IGLÉSIAS 89318

WEI YE 93442

MARIANA LADEIRO 92964

DAVID MORAIS 93147

ROLES

Papel	Responsabilidades	Membro
Team Manager	Assegurar que existe uma distribuição justa das tarefas e que os membros trabalham de acordo com o planeado. Promover ativamente a colaboração entre os membros da equipa bem como tomar a iniciativa para endereçar os eventuais problemas que possam surgir. Assegurar que os resultados esperados do projeto são entregues dentro dos prazos.	Pedro Iglésias
Product Owner	Representa os interesses dos stakeholders. Tem um vasto conhecimento do produto e o seu domínio de aplicação. Pessoa a quem a equipa deverá colocar as dúvidas que tem sobre as features esperadas do produto.	Mariana Ladeiro
Arquiteto	Conhecimento vasto da arquitetura proposta bem como o suporte das tecnologias. Todas as dúvidas que possam surgir relativas ao comportamento de cada componente bem como a interação entre módulos deve ser esclarecida com este.	Wei Ye
DevOps Master	Responsável pelo desenvolvimento e produção de infraestruturas para as configurações requeridas. Assegurar que a ferramenta de desenvolvimento funciona de forma apropriada. Lidera a preparação do repositório git, operações da base de dados, entre outras.	David Moraes
Developer	Contribuir para o desenvolvimento das tarefas.	Todos

IDEIA/OBJECTIVO

Conceito do produto

- Sistema de suporte a cinemas
- Venda e compra de bilhetes
- Informações sobre filmes
- Fazer reviews
- Adicionar, como dono de um cinema, o seu cinema

Personas e Cenários

- **Marta – procura plataforma para adicionar o seu cinema**
 - Envia um request ao admin e, caso aceite, pode gerir os filmes.
- **Lídia – procura um sistema intuitivo e fácil de usar**
 - Efetua rapidamente a compra de bilhetes e, após o filme, faz uma review.
- **Samuel – necessita de uma API com informações sobre filmes para desenvolver uma app**
 - Utiliza a API do site para receber os dados que quer, em formato JSON, através de GETS.

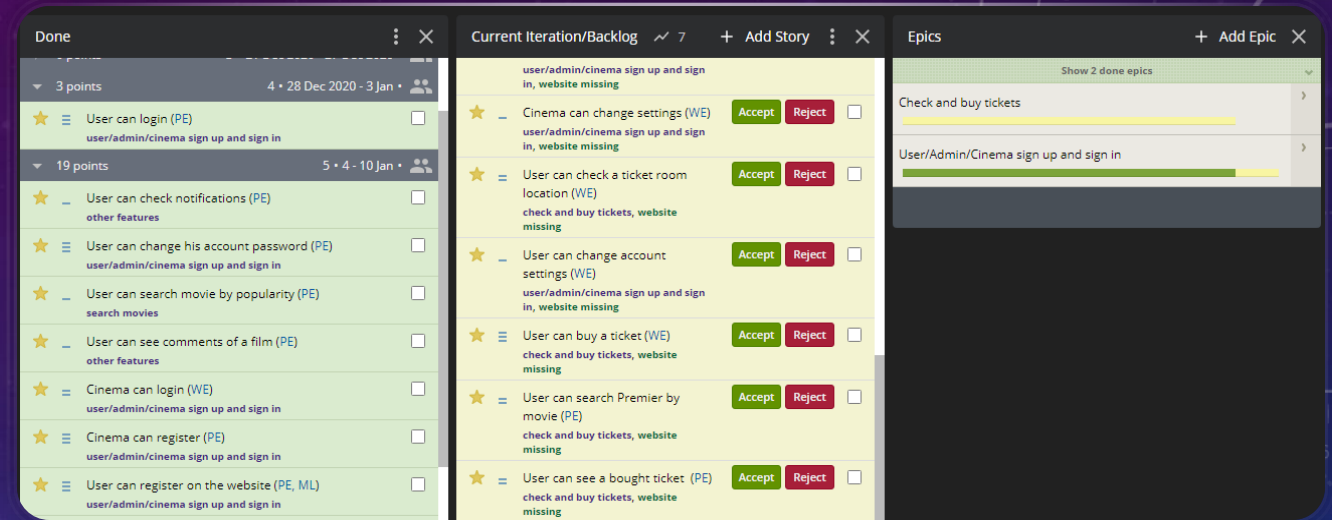
IDEIA/OBJECTIVO

Limitações

- **Segurança na parte do cinema é bastante reduzida**
- **Criar uma release fácil de instalar**
- **O pagamento de um bilhete é meramente uma simulação**
- **Aplicação móvel não foi desenvolvida, como planeado inicialmente**

WORKFLOW / DEVOPS

- Como ferramenta de gestão do backlog foi decidido usar Pivotal Tracker. Esta ferramenta permitirá acompanhar a evolução e o desenvolvimento do projeto
- Facilitou a visualização dos objetivos, permitiu gerir as entregas e as metas ao longo do trabalho e ajudou a dividir o projeto em partes mais simples.
- Para além disso a divisão do trabalho foi em módulos (epics) que por sua vez agrupam várias funcionalidades que estão descritas por user stories.
- As entregas devem ser revistas e analisadas antes de submetidas para que se tenha a certeza de que o objetivo foi concluído e que satisfaz os requisitos para isso no github para aceitar um pull request é necessário revisão e aprovação de pelo menos 50% dos membros neste projeto.
- Também foi criada uma dev branch no github para controlar e testar para verificar se tudo está a funcionar como é suposto.
- Finalmente usamos github issues para discutir qualquer problema que ocorresse.



Database some tables question #13



Iglesias-Leafwind opened this issue on 11 Dec 2020 · 5 comments



Iglesias-Leafwind commented on 11 Dec 2020 · edited

Collaborator

2 of our tables on our db have only 1 primary key like moviebytype and starredin so basically we can only have 1 genre associated with 1 movie but not more than 1 movie and the same for actors and movies



Wei93442 commented on 11 Dec 2020

Collaborator

ah yes, you are right, i will fix the problem, one min.

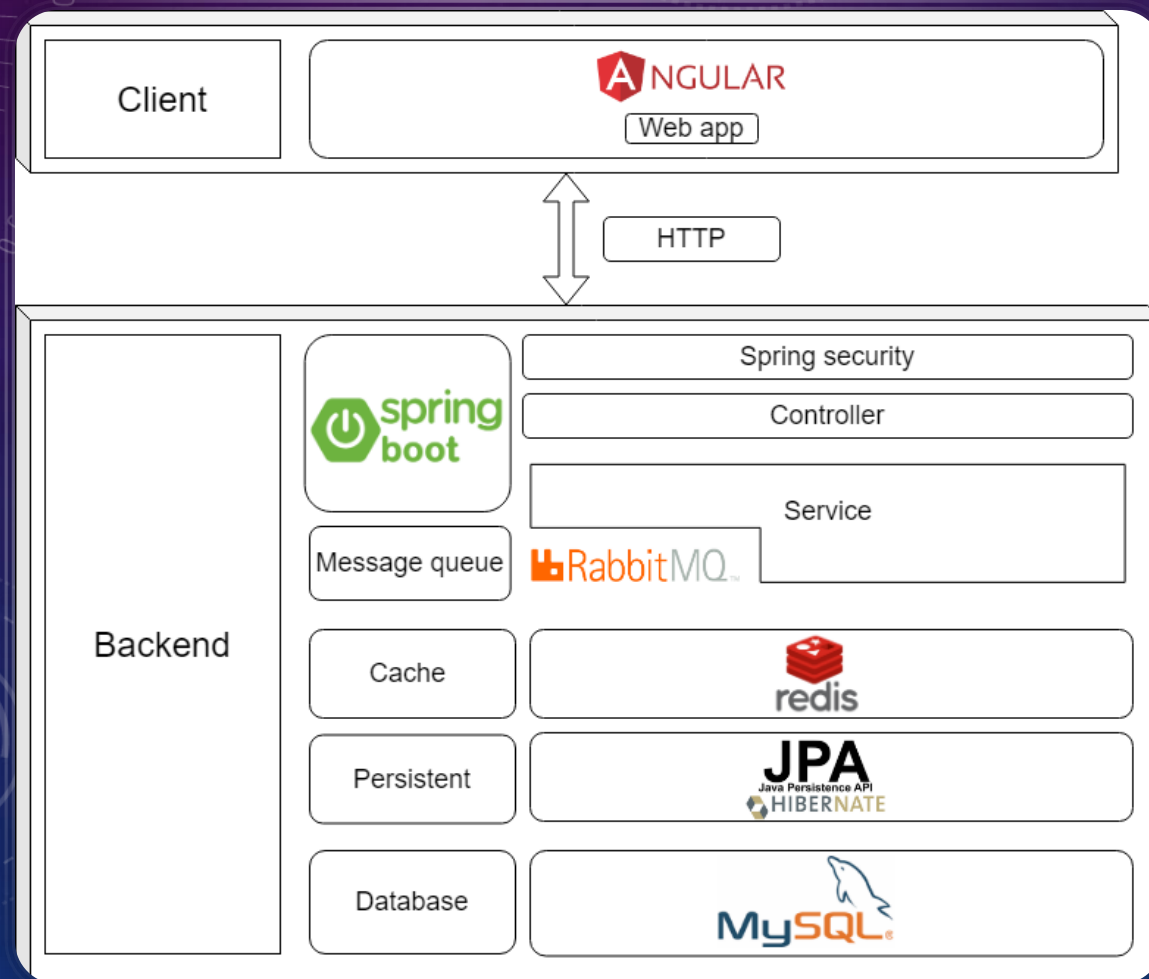
GERAÇÃO DE DADOS

Para a geração de dados foi usado um python script que faz download diretamente ao site (<https://datasets.imdbws.com/>) da imdb um dataset que após ser filtrado dá-nos todos filmes mais recentes (2020/2021) .

```
print("Downloading most recent imdb ids.")
url = 'https://datasets.imdbws.com/title.basics.tsv.gz'
r = requests.get(url, allow_redirects=True)
print("Loading dataset")
f_in = gzip.decompress(r.content)
with open('data.tsv', 'wb') as f_out:
    f_out.write(f_in)
with open('data.tsv', 'r', encoding="utf8") as file:
    print(file.readline())
    while(file.newlines):
        line = file.readline()
        try:
            arr = line.split("\t")
            imdbID, tipo, title, orgTitle, isadult, year, end, runtime, genres = arr
            if(isadult != "0" or tipo != "movie" or int(year) <= 2019 or "Documentary" in genres or runtime == "\\N"):
                continue
            print(imdbID, tipo, year)
            addMovie(imdbID)
        except:
            continue
```

Como esta geração de dados é lenta foi criado outro python script para mudar os dados constantemente para verificarmos as mudanças e utilização da nossa aplicação.

ARQUITETURA



SERVIÇOS/FUNCIONALIDADES

USER CONTROLLER

POST /common/confirm/cinema/{code} confirmRegisterCinema

POST /common/login login

POST /common/register register

POST /common/register/cinema registerCinema

GET /admin/requests getRequest

GET /admin/requests/{requestId} getRequest

PUT /admin/requests/accepted/{requestId} acceptedRequest

PUT /admin/requests/refuse/{requestId} refuseRequest

POST /user/add/favourite/cinema/{cinemaId} addFavouriteCinema

POST /user/add/favourite/film/{filmId} addFavouriteFilm

POST /user/buy/ticket buyTicket

PUT /user/change/avatar change

PUT /user/change/password changePassword

GET /user/favourite/cinema getUserFavouriteCinema

GET /user/favourite/films getUserFavouriteFilms

GET /user/notifications getNotifications

GET /user/payments getPayments

DELETE /user/remove/favourite/cinema/{cinemaId} removeFavouriteCinema

DELETE /user/remove/favourite/film/{filmId} removeFavouriteFilm

SERVIÇOS/FUNCIONALIDADES

FILM CONTROLLER

POST `/admin/add/film` `addFilm`

GET	<code>/common/film/{filmId}</code>	<code>getFilmById</code>
GET	<code>/common/film/{filmId}/premiers{page}</code>	<code>getPremiersByFilm</code>
GET	<code>/common/film/actor/{actor}</code>	<code>getFilmsByActor</code>
GET	<code>/common/film/director/{director}</code>	<code>getFilmsByDirector</code>
GET	<code>/common/film/genre/{genre}</code>	<code>getFilmsByGenre</code>
GET	<code>/common/film/popular</code>	<code>getPopularFilms</code>
GET	<code>/common/film/recent</code>	<code>getRecentFilms</code>
GET	<code>/common/film/title/{title}</code>	<code>getMoviesByTitle</code>
GET	<code>/common/film/year/{year}</code>	<code>getFilmsByYear</code>
GET	<code>/common/genres</code>	<code>getGenres</code>

GET /common/cinema/{cinemaId}/commentPage{page} getCommentsByCinema

GET /common/comment/{parentId}/second/level getSecondLevelComments

GET /common/film/{filmId}/commentPage{page} getCommentsByFilm

GET /common/premier/{premierId}/commentPage{page} getCommentsByPremier

PUT /user/comment/{commentId}/like like

DELETE /user/comment/{commentId}/remove/ removeComment

POST /user/comment/create createComment

SERVIÇOS/FUNCIONALIDADES COMMENT CONTROLLER

SERVIÇOS/FUNCIONALIDADES CINEMA CONTROLLER

PUT /cinema/change/description/{description} changeDescription

POST /cinema/create/premier createPremier

POST /cinema/create/room createRoom

POST /cinema/create/schedule createSchedule

DELETE /cinema/delete/premier/{premierId} deletePremier

DELETE /cinema/delete/schedule/{scheduleId} deleteSchedule

GET /cinema/rooms getRooms

GET /common/cinema/{cinemaId} getCinema

GET /common/cinemas getCinemas

GET /common/premier/{premierId} getPremier

GET /common/schedule/{scheduleId} getSchedule

DEMO DA SOLUÇÃO

