

MoX (*Movies and TV Shows Explorer*)

Ditto (Iacob Eliza-Maria, 2A2)

1. Introducere

Ditto este o aplicatie de tip WEB care prezinta informatii despre filme si seriale, functionalitatea de filtrare si sortare a acestora, statistici si stiri relevante.

Utilizatorii logati au functionalitatea de a lasa review-uri show-urilor pentru a-si impartasi experienta de vizionare cu ceilalti utilizatori ai aplicatiei.

2. Implementare backend

Pentru aplicatia de API am creat un fisier Docker ce creeaza doua containere de PHP si baze de date. Container-ul PHP foloseste versiune 8.2, contine serverul Nginx, instaleaza bibliotecile si extensiile PHP necesare. Configuratia NGINX este facuta in asa fel incat sa folosim rutarea costumizata.

```
FROM php:8.2-fpm

RUN apt-get update -q && \
    apt-get install -qy \
        libzip-dev \
        git \
        unzip \
        default-mysql-client

# Install Nginx and necessary libraries
RUN apt-get update && apt-get install -y \
    nginx \
    libpq-dev \
    default-mysql-client

# Install necessary PHP extensions
RUN docker-php-ext-install pdo pdo_mysql zip

# Install Composer
COPY --from=composer /usr/bin/composer /usr/bin/composer


# Copy Nginx configuration
COPY ./docker/nginx/nginx.conf /etc/nginx/sites-available/default
```

In clasa Router am definit metodele GET, POST, PUT, DELETE și rutele care au fost organizate în așa fel incat sa duca către o metoda din controller-ul respectiv.

```

class Router
{
    1 usage
    private $routes = [
        'GET' => [
            '/api/shows/{id}' => ['controller' => 'ShowsController', 'method' => 'getById'],
            '/api/search/shows' => ['controller' => 'ShowsController', 'method' => 'search'],
            '/api/shows' => ['controller' => 'ShowsController', 'method' => 'getAll'],
            '/api/filters/shows' => ['controller' => 'ShowsController', 'method' => 'getFilters'],
            '/api/statistics/top-rated' => ['controller' => 'StatisticsController', 'method' => 'getTopRated'],
            '/api/statistics/popular' => ['controller' => 'StatisticsController', 'method' => 'getPopular'],
            '/api/reviews/{id}' => ['controller' => 'ReviewsController', 'method' => 'getByShowId'],
            '/api/reviews' => ['controller' => 'ReviewsController', 'method' => 'getAll'],
            '/api/news' => ['controller' => 'NewsController', 'method' => 'getAll'],
        ],
        'POST' => [
            '/api/reviews' => ['controller' => 'ReviewsController', 'method' => 'add'],
            '/api/users/login' => ['controller' => 'UsersController', 'method' => 'login'],
            '/api/users/logout' => ['controller' => 'UsersController', 'method' => 'logout'],
            '/api/users/register' => ['controller' => 'UsersController', 'method' => 'register'],
            '/api/shows' => ['controller' => 'ShowsController', 'method' => 'add'],
            '/api/shows/import' => ['controller' => 'ShowsController', 'method' => 'import'],
        ],
        'PUT' => [
            '/api/shows/{id}' => ['controller' => 'ShowsController', 'method' => 'update']
        ],
        'DELETE' => [
            '/api/reviews/{id}' => ['controller' => 'ReviewsController', 'method' => 'delete']
        ]
    ];
}

```

- ▼  Controllers
 - © BaseController.php
 - © ErrorController.php
 - © NewsController.php
 - © ReviewsController.php
 - © ShowsController.php
 - © StatisticsController.php
 - © UsersController.php

```

class StatisticsController extends BaseController
{
    no usages  ± Iacob Eliza-Maria
    public function getTopRated($params, $urlParams)
    {
        $data = [
            'movies' => TheMovieDbHelper::callStatisticsEndpoint(),
            'tv_shows' => TheMovieDbHelper::callStatisticsEndpoint( statisticType: 'top_rated', showType: 'tv'),
        ];

        if (isset($urlParams['export']) && isset($urlParams['type'])) {
            $this->exportToFile($urlParams['export'], $urlParams['type'], $data);

            return;
        }

        $this->returnJsonResponse($data, statusCode: 200);
    }

    no usages  ± Iacob Eliza-Maria
    public function getPopular($params, $urlParams)
    {
        $data = [

```

Fisierele composer.json si composer.lock contin bibliotecile adaugate cu toate detaliile acestora.

In TheMovieDbHelper.php facem call-ul la baza de date externa [The Movie Database](#) și ne conecteaza cu ajutorul librăriei Guzzle. Aceasta ne ajuta in controller-ele Statistics and News, dar și în Shows unde avem nevoie de date externe pentru show-uri.

In clasa UserController, functia login() se creaza token-ul de forma “id.user,parola” pe care apoi il codam in formatul base64. Îl vom folosi pentru metode de tip PUT, POST sau DELETE unde vom avea nevoie de autorizatie sa le executăm.

```

if ($body['password'] === DataEncryptor::decryptData($user['password'])) {
    $authData = $user['id'].'.'.$user['password'];
    $this->returnJsonResponse(['data' => $user, 'auth' => 'Bearer ' . base64_encode($authData)], statusCode: 200);
} else {
    $this->returnJsonResponse(['errors' => 'Wrong password!'], statusCode: 400);
}

```

3. Implementare frontend

Datele din formularul de inregistrare ca utilizator al aplicatiei sunt trimise catre un endpoint din API care adauga utilizatorul in baza de date pentru a se putea loga.

```
$(document).ready(function() {
    $('#registrationForm').submit(function(event) {
        event.preventDefault();

        var username = $('#username').val();
        var password = $('#password').val();
        var email = $('#email').val();
        var age = $('#age').val();
        var country = $('#country').val();

        // Make an AJAX request to the API for authentication
        $.ajax({
            url: 'http://localhost/api/users/register',
            method: 'POST',
            data: JSON.stringify({
                username: username,
                password: password,
                email: email,
                age: age,
                country: country
            }),
            success: function(response) {
                console.log(response);

                window.location.href = 'login.html';
            },
            error: function(xhr, status, error) {
                console.log(xhr, error, status);
            }
        });
    });
});
```

La logare, se creeaza o sesiune în care adaug date despre utilizator.

```
localStorage.setItem('session', JSON.stringify({
  Authorization: response.auth,
  username: response.data.username,
  user_type: response.data.type
}));
```

Pagina cu detaliile unui show foloseste fisierul checkSession.js pentru a verifica daca un utilizator este logat sau nu, daca va putea lasa un review show-ului. De asemenea, acesta schimba functionalitatea butonulu Login, care se transforma in Logout si sterge sesiunea cand interactionam cu el.

```
function deleteSession() {
  localStorage.removeItem('session');
  window.location.href = 'login.html';
}

function checkSession() {
  var session = localStorage.getItem('session');

  if (session) {
    session = JSON.parse(session);

    console.log('Logged in as: ' + session.username);

    var log = document.getElementById('log');
    document.getElementById('log').innerHTML = 'Logout';
    log.addEventListener('click', deleteSession);
  }
}

checkSession();
```

```
<div class="rightSection">
  <a class="links" href="movies.html">Shows</a>
  <a class="links" href="statistics.html">Statistics</a>
  <a class="links" href="news.html">News</a>
  <a class="links" href="about.html">About</a>
  <a id="log" class="links" href="login.html">Login</a>
</div>
</nav>
<script src="checkSession.js"></script>
```

```
<form id="postReview" hidden>
  <h7>Add Your Review</h7>
  <br>
  <label for="review"></label>
  <textarea id="review" name="review" rows="4" cols="50" required></textarea>
  <br>
  <label for="ratingform" min="1" max="10">Rating(1-10):</label>
  <input type="number" id="ratingform" name="ratingform" min="1" max="10" required>
  <br>
  <input type="submit" value="Submit">
</form>
```

```
if (localStorage.getItem('session')) {
  document.getElementById('postReview').removeAttribute("hidden");
}
```