Università degli Studi di Napoli Parthenope

Dipartimento di Scienze e Tecnologie

Corso di Basi di Dati e Laboratorio

ED Films

Emanuele D’Ambrosio MAT - 0124002587

Domenico Zeno MAT - 0124002479

Outline:

A Web Page is to be created for the development of a platform for users to watch movies.

A user within the Page will be able to perform various operations by clicking on the different settings at the top of the screen. Also, to the right of the screen, there will be a search bar where the user will be able to perform the operation of searching by name for a particular movie.

Web Pages:

Home

The Home is the opening page of the platform. Below the taskbar applications has been made a Slideshow containing images accompanied with concise sentences to guide the user on the various functionality of the page and the different usage scenarios in which it could be used.

Film

Within this page are movies arranged in a grid that the user can view only after performing the Login operation.

Note - By intersecting this project with the Software Engineering project, we will print all the movies in the database on the screen. So also for the search function by name in the search bar.

Access

Access is the third button within the taskbar. Clicking on it will open under the heading Login, a drop-down menu containing two possible operations: Sign and Login.

Sign

The user will be taken to the Registration page after clicking on the Login button and then the Registration button on the drop-down menu. Within this page, the user will be faced with a Formbox where they will have to enter the various information to perform the Registration operation. So Username (which will be unique), Password, e-mail, First Name, Last Name, and Date of Birth.

The Date of Birth is an essential field since there are films that are subject to age limits. So within the "Date of Birth" field to be filled in, a function was applied to run a check and see if the user has an appropriate age range. In case the user has an age that is not in the range, an error message will appear and the content in the Date of Birth field will be deleted.

Login

The user will be taken to the Registration page after clicking on the Login button and then the Login button from the drop-down menu. Within this page, the user will need to enter the username and password provided at Registration to complete the operation.

Note - By intersecting this project with the Software Engineering project, we will compare the credentials entered by the user with those within the database.

HTML:

Home:

```
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>EDFilms</title>
          <link rel="icon" href="Icona_Scheda.png" sizes="32x32">
          <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
rbsA2VBKQhggwzxH7pPCaAqO46MgnOM80zW1RWuH61DGLwZJEdK2Kadq2F9CUG65"
crossorigin="anonymous">
    <link rel="stylesheet" href="style.css">
  </head>

  <body style="background-color: black;">

    <!-- Define the navbar background as Dark Purple -->
    <nav class="navbar navbar-expand-lg" style="background-color: #33006E;">

    <!-- Container for the navbar content with the same Background Color -->
    <div class="container-fluid" style="background-color: #33006E;">

    <!-- Logo  -->
    <a class="navbar-brand" href="#">
      <img src="Logo.png" alt="Bootstrap" width="120" height="90">
    </a>

    <!-- Navbar toggler button for small screens -->
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarScroll" aria-controls="navbarScroll" aria-expanded="false" aria-
label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>

    <!-- Navbar links and dropdown menu -->
    <div class="collapse navbar-collapse" id="navbarScroll">
      <ul class="navbar-nav me-auto my-2 my-lg-0 navbar-nav-scroll" style="--bs-
scroll-height: 100px;">

      <!-- Navbar Home button to get to the Home page -->
      <li class="nav-item">
        <a class="nav-link active" aria-current="page" href="Home.html"
style="color: white;">
          <h4>Home</h4>
        </a>
      </li>

      <!-- Navbar Film button to get to the Film page -->
      <li class="nav-item">
        <a class="nav-link" href="nuovo.html" style="color: white;">
          <h4>Film</h4>
        </a>
      </li>

      <!-- Accesso Dropdown with text -->
      <li class="nav-item dropdown">
          <a class="nav-link dropdown-toggle" href="#" role="button" data-bs-
toggle="dropdown" aria-expanded="false" style="color: white;">
            <h4>Accesso</h4>
          </a>

        <!-- Dropdown menu with Login and Registrazione links -->
          <ul class="dropdown-menu">
            <!-- Login and Registrazione buttons to get to the Login and
Registrazione pages-->
            <li><a class="dropdown-item" href="Login.html">Login</a></li>
            <li><a class="dropdown-item" href="Registrazione.html">Registrazione</
a></li>
          </ul>
      </li>

      <!-- Search form function with a button -->
      <form class="d-flex" role="search" onsubmit="event.preventDefault();
SEARCH()" method="POST">
        <input class="form-control me-2" type="search" placeholder="Cerca..." aria-
label="Search" name="search_query" id="nome">
        <button class="btn btn-outline-success" type="submit"
style="color:#FFFFFF">Ricerca</button> <!-- Search Button-->
      </form>

      </ul>
    </div>

  </div>
  </nav>
```

```html
    <!-- Carousel container with id "carouselExampleCaptions" and data-bs-ride
attribute set to "false" -->
    <div id="carouselExampleCaptions" class="carousel slide" data-bs-ride="false">

      <!-- Carousel indicators for each slide -->
      <div class="carousel-indicators">
        <button type="button" data-bs-target="#carouselExampleCaptions" data-bs-
slide-to="0" class="active" aria-current="true" aria-label="Slide 1"></button>
        <button type="button" data-bs-target="#carouselExampleCaptions" data-bs-
slide-to="1" aria-label="Slide 2"></button>
        <button type="button" data-bs-target="#carouselExampleCaptions" data-bs-
slide-to="2" aria-label="Slide 3"></button>
      </div>

      <!-- Carousel inner container with individual slides -->
      <div class="carousel-inner">

        <!-- Slide 1 with an image and a hidden caption for medium and larger
screens -->
        <div class="carousel-item active">
          <img src="SLD3.png" class="d-block w-100" alt="...">
            <div class="carousel-caption d-none d-md-block"> <!-- Caption content
for the first slide -->
            </div>
        </div>

        <!-- Slide 2 with an image and a hidden caption for medium and larger
screens -->
        <div class="carousel-item">
          <img src="SLD4.png" class="d-block w-100" alt="...">
            <div class="carousel-caption d-none d-md-block"> <!-- Caption content
for the second slide -->
            </div>
        </div>

        <!-- Slide 3 with an image and a hidden caption for medium and larger
screens -->
        <div class="carousel-item">
          <img src="SLD1.png" class="d-block w-100" alt="...">
            <div class="carousel-caption d-none d-md-block"> <!-- Caption content
for the third slide -->
            </div>
        </div>
      </div>

      <!-- Carousel navigation buttons for previous and next slides -->
      <button class="carousel-control-prev" type="button" data-bs-
target="#carouselExampleCaptions" data-bs-slide="prev">
```

```
        <span class="carousel-control-prev-icon" aria-hidden="true"></span>
        <span class="visually-hidden">Previous</span>
      </button>

      <button class="carousel-control-next" type="button" data-bs-
target="#carouselExampleCaptions" data-bs-slide="next">
        <span class="carousel-control-next-icon" aria-hidden="true"></span>
        <span class="visually-hidden">Next</span>
      </button>
    </div>

    <!-- Inclusion of Bootstrap library (version 5.2.3) -->
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-kenU1KFdBIe4zVF0s0G1M5b4hcpxyD9F7jL+jjXkk+Q2h455rYXK/7HAuoJl+0I4"
crossorigin="anonymous"></script>

    <!-- Custom JavaScript script for search function -->
    <script>
    //Definition of the search function
      function SEARCH() {
      // Obtaining the value of the search input
      let nome = document.getElementById('nome').value;
      // Construction of the search URL with the parameter 'name'
      let url = http://127.0.0.1/api/search?nome=${encodeURIComponent(nome)};

      // Executing a fetch request to the server
      fetch(url, {
                  method: 'GET',
                  headers: {
                            'Content-Type': 'application/json'
                           },
                  credentials: 'include'
                 })
                // Check if the answer is positive
                .then(response => {
                    if (!response.ok) {
                        throw new Error('Network response was not ok');
                    }
                    return response.json(); // Return of the response as JSON
                })
                .then(responseJson => {
                  // Check if there is data about movies in the answer
                    if (responseJson.film) {
                        // Serialize the movie object into a JSON string and pass
it into the URL
                        let filmData =
encodeURIComponent(JSON.stringify(responseJson.film));
```

```
                        // Redirects to the search.html page with the data in the
URL
                        window.location.href = search.html?nome=${filmData};
                    } else {
                        console.log(responseJson.error); // Error message if no
movie was found
                    }
                })
                .catch(error => {
                  // Error handling in the console
                    console.error('Error:', error);
                });
            }
        </script>
    </body>
</html>
```

Registrazione:

On the pages following Home, we have avoided reporting the Head and Navbar code.

```
<!-- Registration Form Container -->
<div class="container mt-5">
    <!-- Row for form alignment -->
    <div class="row justify-content-center">
        <!-- Form Column -->
        <div class="col-md-5">
            <!-- Registration Form -->
            <form onsubmit="event.preventDefault(); INSERIMENTO()" method="post"
class="custom-form">
                <!-- Title -->
                <h4 class="card-title text-center">Registration</h4>

                <br>

                <!-- First Row: Username and Password -->
                <div class="row mb-3">
                    <!-- Username Input -->
                    <div class="col-md-6">
                        <label for="name" class="form-label">Username:</label>
                        <input type="text" class="form-control" id="username"
name="name" required>
                    </div>

                    <!-- Script to reset username field -->
                    <script>
                        document.addEventListener("DOMContentLoaded", function () {
                            // Reset the username field on page load
                            document.getElementById("username").value = "";
                        });
                    </script>

                    <!-- Password Input -->
                    <div class="col-md-6">
                        <label for="password" class="form-label">Password:</label>
                        <input type="password" class="form-control" id="password"
name="password" required>
                    </div>
                </div>

                <!-- Second Row: First Name and Last Name -->
                <div class="row mb-3">
                    <!-- First Name Input -->
                    <div class="col-md-6">
                        <label for="name" class="form-label">First Name:</label>
                        <input type="text" class="form-control" id="name"
name="nome" required>
                    </div>

                    <!-- Last Name Input -->
                    <div class="col-md-6">
                        <label for="surname" class="form-label">Last Name:</label>
                        <input type="text" class="form-control" id="surname"
name="surname" required>
                    </div>
                </div>

                <!-- Third Row: Email and Date of Birth -->
                <div class="row mb-3">
                    <!-- Email Input -->
                    <div class="col-md-6">
                        <label for="email" class="form-label">Email:</label>
                        <input type="email" class="form-control" id="email"
name="email" required>
                    </div>

                    <!-- Date of Birth Input -->
                    <div class="col-md-6">
                        <label for="date" class="form-label">Date of Birth:</label>
                        <input type="date" class="form-control" id="date"
name="date" placeholder="DD/MM/YYYY" required>
                    </div>
                </div>

                <br>

                <!-- Submit Button -->
                <button class="btn btn-outline-success mx-auto d-block"
type="submit" style="color: #FFFFFF;">Confirm</button>
```

```
            </form>
        </div>
    </div>
</div>

<!-- Bootstrap JavaScript Library -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-kenU1KFdBIe4zVF0s0G1M5b4hcpxyD9F7jL+jjXkk+Q2h455rYXK/7HAuoJl+0I4"
crossorigin="anonymous"></script>

<script>
    document.addEventListener("DOMContentLoaded", function () {
        // Function to validate user's age
        function checkAge() {
            // Retrieve the date of birth input element
            var dateInput = document.getElementById("date");

            // Delay the age validation by 1500 milliseconds (1.5 seconds)
            setTimeout(function () {
                // Get the user's birth date value
                var birthDate = new Date(dateInput.value);

                // Calculate the user's age
                var age = new Date().getFullYear() - birthDate.getFullYear();

                // Check if the age is within a valid range (10 to 140 years)
                if (age < 10 || age > 140) {
                    // Clear the date of birth input if invalid
                    dateInput.value = "";

                    // Display an error message for an invalid birth date
                    alert("The entered date of birth is not valid. Please verify
and try again.");
                }
            }, 1500); // 1.5 seconds delay
        }

        // Attach an event listener to the date of birth input to trigger the age
validation function
        document.getElementById("date").addEventListener("input", checkAge);
    });
// Function to handle form submission and send data to the server
function INSERIMENTO() {
    // Prepare the data from form inputs
    let requestBody = {
        username: document.getElementById('username').value,
        nome: document.getElementById('name').value,
        cognome: document.getElementById('surname').value,
        email: document.getElementById('email').value,
        password: document.getElementById('password').value,
        date: document.getElementById('date').value,
    };

    // Convert data to JSON format
    let jsonString = JSON.stringify(requestBody);

    // Send a POST request to the server's API endpoint
    fetch('http://127.0.0.1/api/inserimento', {
        method: 'POST',
        withCredentials: true,
        headers: {
            'Content-Type': 'application/json'
        },
        credentials: 'include',
        body: jsonString
    })
    // Parse the response from the server as JSON
    .then(response => response.json())
    // Handle the response data
    .then(response => {
        // Log the response data to the console
        console.log(response);
        // Redirect the user to 'nuovo.html' upon successful submission
        window.location.href = 'nuovo.html';
    });
}
```

Login:

```
<!-- Login Form Container -->
<div class="container mt-5">
    <!-- Row for form alignment -->
    <div class="row justify-content-center">
        <!-- Form Column -->
        <div class="col-md-3">
            <!-- Login Form -->
            <form onsubmit="event.preventDefault(); LOGIN()" method="post"
class="custom-form">
                <!-- Title -->
                <h4 class="card-title text-center">Login</h4>
                <br>
                <!-- First row: Username -->
                <div class="row mb-3">
                    <div class="col-md-12">
                        <label for="name" class="form-label">Username:</label>
                        <input type="text" class="form-control" id="name"
name="name" required>
                    </div>
                </div>
                <!-- Second row: Password -->
                <div class="col-md-12">
                    <label for="password" class="form-label">Password:</label>
                    <input type="password" class="form-control" id="password"
name="password" required>
                </div>
                <!-- Third row (email) -->
                <br>
                <!-- Submit Button -->
                <button class="btn btn-outline-success mx-auto d-block"
type="submit" style="color: #FFFFFF;">Confirm</button>
            </form>
        </div>
    </div>
</div>

<!-- Bootstrap JavaScript Library -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-kenU1KFdBIe4zVF0s0G1M5b4hcpxyD9F7jL+jjXkk+Q2h455rYXK/7HAuoJl+0I4"
crossorigin="anonymous"></script>

<!-- JavaScript function for handling login -->
<script>
    function LOGIN() {
        // Prepare the data from form inputs
        let requestBody = {
            name: document.getElementById('name').value,
            password: document.getElementById('password').value
        };

        // Convert data to JSON format
        let jsonString = JSON.stringify(requestBody);

        // Send a POST request to the server's login API endpoint
        fetch('http://127.0.0.1/api/login', {
            method: 'POST',
            withCredentials: true,
            headers: {
                'Content-Type': 'application/json'
            },
            credentials: 'include',
            body: jsonString
        })
        // Parse the response from the server as JSON
        .then(response => response.json())
        // Handle the response data
        .then(response => {
            // Log the response data to the console
            console.log(response);
            // Redirect the user to 'nuovo.html' upon successful login
            window.location.href = 'nuovo.html';
        });
    }
</script>
```

Film:

```
<html>
<head>
    <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>EDFilms</title>

    <link rel="icon" href="Icona_Scheda.png" sizes="32x32">
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
rbsA2VBKQhggwzxH7pPCaAqO46MgnOM80zW1RWuH61DGLwZJEdK2Kadq2F9CUG65"
crossorigin="anonymous">

    <link rel="stylesheet" href="style.css">

</head>
<body>
    <nav class="navbar navbar-expand-lg" style="background-color: #33006E;">

        <div class="container-fluid" style="background-color: #33006E;">
            <a class="navbar-brand" href="#"><img src="Logo.png" alt="Bootstrap"
width="120" height="90"></a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
data-bs-target="#navbarScroll" aria-controls="navbarScroll" aria-expanded="false"
aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarScroll">
                <ul class="navbar-nav me-auto my-2 my-lg-0 navbar-nav-scroll"
style="--bs-scroll-height: 100px;">

                    <li class="nav-item">
                        <a class="nav-link active" aria-current="page"
href="Home.html" style="color: white;"><h5>Home</h5></a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="nuovo.html" style="color:
white;"><h5>Film</h5></a>
                    </li>
                </ul>

                <form class="d-flex" role="search"
onsubmit="event.preventDefault(); SEARCH()" method="POST">
                  <input class="form-control me-2" type="search"
placeholder="Cerca..." aria-label="Search" name="search_query" id="nome">
                  <button class="btn btn-outline-success" type="submit"
style="color:#FFFFFF">Ricerca</button>
              </form>

            </div>
        </div>
    </nav>
    <div class="modal-overlay">
    <div class="modal-content">
      <div id="innerDiv"> ciao<!-- Verify that this element contains the content
you want to modify -->
        <!-- Content that will be modified -->
      </div>
    </div>
  </div>
    <div id="filmsContainer" class="cont">
        <div class="container">
             <!-- Movie data will be dynamically added here -->
        </div>
    </div>

    <script>
       window.onload = function() {console.log("ccc");
    getFilmLibrary();
    setTimeout(function() {
        animateDivsToCenter(); // Move the divs to the center with animation
    }, 100); // Delay the animation by 100 milliseconds after the page loads

    // Activate a click event handler for each div with the class animatedDiv as
well
    document.querySelectorAll('.animatedDiv').forEach(div => {
        div.addEventListener('click', () => {
            const imgSrc = div.querySelector('img').getAttribute('src');
            const videoSrc = imgSrc.replace(/\.(jpg|jpeg)$/i, '.mp4');
            const h1Element = div.querySelector('h1');
            const h1Text = h1Element.textContent;
            const ulElement = div.querySelector('ul');
            const liElements = ulElement.querySelectorAll('li');

            openModal(videoSrc, h1Text, liElements);
        });
    });
};

function getFilmLibrary() {
    let mouseLeftContainer = true;
    let modalTimeout;
    fetch('http://127.0.0.1/api/films', {
        method: 'GET',
        withCredentials: true,
        headers: {
            'Content-Type': 'application/json'
        },
        credentials: 'include'
    })
    .then(response => {
        if (!response.ok) {
            throw new Error('Network response was not ok');
        }
        return response.json();
    })
    .then(responseJson => {
        if (responseJson.films && responseJson.films.length > 0) {
            let filmLibrary = responseJson.films;
            let filmsContainer = document.getElementById('filmsContainer');

            filmLibrary.forEach(film => {
                let filmDiv = document.createElement('div');
                filmDiv.classList.add('animatedDiv');
                const randomClass = getRandomClass();
                filmDiv.classList.add(randomClass);

                const truncatedName = film.NOME_FILM.length > 11 ? `$
{film.NOME_FILM.slice(0, 11)}..` : film.NOME_FILM;

                filmDiv.innerHTML = `
                    <img src="uploads/${film.NOME_FILM}.jpg" alt="$
{film.NOME_FILM}" style="width: 100%; height: 70%;" />
                    <h1>${truncatedName}</h1>
                    <ul class="a">
                        <li>Nome: ${film.NOME_FILM}</li>
                        <li>Descrizione: ${film.DESCRIZIONE}</li>
                        <li>ANNO:${film.ANNO}</li>
                    </ul>
                `;

                filmsContainer.appendChild(filmDiv);

                filmDiv.addEventListener('click', function() {
                    const imgSrc = this.querySelector('img').getAttribute('src');
                    const videoSrc = imgSrc.replace(/\.(jpg|jpeg)$/i, '.mp4');
                    const h1Element = this.querySelector('h1');
                    const h1Text = h1Element.textContent;
                    const ulElement = this.querySelector('ul');
                    const liElements = ulElement.querySelectorAll('li');

                    openModal(videoSrc, h1Text, liElements);
                });

                filmDiv.addEventListener('mouseleave', function(event) {
                    const isMouseOut = !this.contains(event.relatedTarget);
                    const innerDiv = document.getElementById('innerDiv');
                    const isMouseInsideInnerDiv =
innerDiv.contains(event.relatedTarget);

                    if (isMouseOut && !isMouseInsideInnerDiv) {
                        mouseLeftContainer = true;
                        modalTimeout = setTimeout(() => {
                            if (mouseLeftContainer) {
                                closeModal();
                            }
                        }, 200);
                    }
                });

                filmDiv.addEventListener('mouseenter', function() {
                    mouseLeftContainer = false;
                    clearTimeout(modalTimeout);
                });
            });

            function closeModal() {
                const modalOverlay = document.querySelector('.modal-overlay');
                const innerDiv = document.getElementById('innerDiv');

                innerDiv.innerHTML = ''; // Remove the content
                modalOverlay.style.display = 'none'; // Hide the modal overlay
            }

            // After adding all the divs, let's add the function to center them
            animateDivsToCenter();
        } else {
            console.log('Nessun film trovato.');
        }
    })
    .catch(error => {
        console.error('Error:', error);
    });
}
```

```
        function getRandomClass() {
            const classes = ['blue', 'red', 'green', 'yellow']; // List of
available classes
            const randomIndex = Math.floor(Math.random() * classes.length); //
Generate a random index
            return classes[randomIndex]; // Return a random class from the list
        }

        function animateDivsToCenter() {
            // Select the container and all animated divs
            var container = document.querySelector('.container');
            var divs = document.querySelectorAll('.animatedDiv');
            console.log(divs);
            // Initialize variables for calculations
            var containerWidth = container.offsetWidth;
            var totalWidth = 0;
            var currentLineHeight = 45;
            var currentLineWidth = 0;
            var lineHeight = 0;
            var marginLeft = 10; // Space between divs

            // Loop through each div to position them in the container
            divs.forEach(function(div) {
                var divWidth = div.offsetWidth;
                var divHeight = div.offsetHeight;

                // If the div exceeds the container's width, move to the next line
                if (currentLineWidth + divWidth > containerWidth) {
                currentLineHeight += lineHeight + marginLeft;
                currentLineWidth = 0;
                lineHeight = 0;
                }

                // Set the position of the div
                div.style.top = currentLineHeight + 'px';
                div.style.left = currentLineWidth + 'px';

                // Update values for the next div placement
                currentLineWidth += divWidth + marginLeft;
                totalWidth = Math.max(totalWidth, currentLineWidth);
                lineHeight = Math.max(lineHeight, divHeight);
            });

            // Set the container's width to accommodate all positioned divs
            container.style.width = totalWidth + 'px';
        }
        // This function will handle opening the screen in the center
function openModal(videoSrc, h1Text, liElements) {console.log("openModal function
called");
    const modalOverlay = document.querySelector('.modal-overlay');
    const innerDiv = document.getElementById('innerDiv');

    const videoElement = document.createElement('video');
    videoElement.setAttribute('src', videoSrc);
    videoElement.setAttribute('width', '700');
    videoElement.setAttribute('height', '400');
    videoElement.setAttribute('controls', 'true');

    const textContainer = document.createElement('div');
    const h1TextElement = document.createElement('h1');
    const ulTextElement = document.createElement('ul');
    h1TextElement.textContent = h1Text;
    liElements.forEach(li => {
        const liText = document.createElement('li');
        liText.textContent = li.textContent;
        ulTextElement.appendChild(liText);
    });
    textContainer.appendChild(h1TextElement);
    textContainer.appendChild(ulTextElement);

    // To update innerDiv with the video and text
    innerDiv.innerHTML = '';
    innerDiv.appendChild(videoElement);
    innerDiv.appendChild(textContainer);

    // To set the style to display the screen in the center
    innerDiv.style.position = 'fixed';
    innerDiv.style.top = '50%';
    innerDiv.style.left = '50%';
    innerDiv.style.transform = 'translate(-50%, -50%)';
    modalOverlay.style.display = 'flex';
}
            function SEARCH() {
            let nome = document.getElementById('nome').value;
            let url = `http://127.0.0.1/api/search?nome=$
{encodeURIComponent(nome)}`;

            fetch(url, {
                method: 'GET',
                headers: {
                    'Content-Type': 'application/json'
                },
                credentials: 'include'
            })
            .then(response => {
                if (!response.ok) {
                    throw new Error('Network response was not ok');
                }
                return response.json(); // We expect the response as JSON
            })
            .then(responseJson => {
                if (responseJson.film) {
                    // Serialize the 'film' object into a JSON string and pass it
in the URL
                    let filmData =
encodeURIComponent(JSON.stringify(responseJson.film));

                    // Redirect to the search.html page with the data in the URL
                    window.location.href = `search.html?nome=${filmData}`;
                } else {
                    console.log(responseJson.error); // Send a message in the
console window if there was an error
                }
            })
            .catch(error => {
                console.error('Error:', error);
            });
        }
    </script>
</body>
</html>
```

Search Film:

```
<!--Placeholder for film details -->
<div align="center" class="cont">
  <div class="container">
    <!-- Display film information in an animatedDiv -->
    <div class="animatedDiv blue">
      <!-- Image related to the film -->
      <img src="uploads/" + film.NOME_FILM + ".jpg") alt="Blue Image" style="width:
100%; height: 70%;" />
      <!-- Film details: name, category, description, year -->
      <h1 id="name"></h1>
      <ul class="a">
        <li align="center" id="coso"></li>
        <li id="desc"></li>
        <li id="anno"></li>
      </ul>
    </div>
  </div>
</div>
<!-- JavaScript to populate film details -->
<script>
  document.addEventListener('DOMContentLoaded', function() {
    // Fetch film data from URL params
    const urlParams = new URLSearchParams(window.location.search);
    const filmData = urlParams.get('nome');
    // Check if film data exists
    if (filmData) {
      // Parse the film data into JSON format
      let film = JSON.parse(decodeURIComponent(filmData));

      // Check if film and its name exist
      if (film && film.NOME_FILM) {
        // Construct the image name and path
        let imageName = film.NOME_FILM + ".jpg";
        let imagePath = "uploads/" + imageName;

        // Select the image element and update its source
        let imgElement = document.querySelector('.animatedDiv.blue img');
        imgElement.src = imagePath;

        // Select elements to display film details and update their content
        let elem = document.getElementById("name");
        let elem1 = document.getElementById("coso");
        let elem2 = document.getElementById("desc");
        let elem3 = document.getElementById("anno");

        // Update content based on film details
        if (film.NOME_FILM) {
          elem.innerHTML = film.NOME_FILM; // Film name
          elem1.innerHTML = film.NOME_FILM; // Film name
          elem2.innerHTML = film.DESCRIZIONE; // Film description
          elem3.innerHTML = film.ANNO; // Film year
        } else {
          console.log('Il campo NOME_FILM non è definito nel dato film');
        }
      } else {
        console.log('Nessun dato film salvato');
      }
    }
  });
</script>
```

App.py:

```
from flask import Flask, render_template, redirect, request, jsonify,url_for
from flask_sqlalchemy import SQLAlchemy
import os
import hashlib
import flask_login

app = Flask(_name_)
app.config['SQLALCHEMY_DATABASE_URI'] =
'mysql+mysqlconnector://root:root@mysql/tech'
```

```
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

db = SQLAlchemy(app)

class Utente(db.Model):
    _tablename_ = 'utente'

    USERNAME = db.Column(db.String(80), primary_key=True)
    NOME = db.Column(db.String(80))
    COGNOME = db.Column(db.String(80))
    MAIL = db.Column(db.String(255))
    PASSWORD = db.Column(db.String(255))
    DATA = db.Column(db.Date)
    RUOLO=db.Column(db.String(80))

class Film(db.Model):
    _tablename_ = 'film'

    COD_FILM = db.Column(db.Integer, primary_key=True)
    NOME_FILM = db.Column(db.String(80))
    DESCRIZIONE = db.Column(db.String(300))
    ANNO=db.Column(db.Integer)

class Categoria(db.Model):
    _tablename_ = 'categoria'

    ID_CATEGORIA = db.Column(db.Integer, primary_key=True)
    NOME_CATEGORIA = db.Column(db.String(80))

class CategorieDiFilm(db.Model):
    _tablename_ = 'categorie_di_film'

    ID_CATEGORIA = db.Column(db.Integer, db.ForeignKey('categoria.ID_CATEGORIA'),
primary_key=True)
    COD_FILM = db.Column(db.Integer, db.ForeignKey('film.COD_FILM'),
primary_key=True)

class Recensioni(db.Model):
    _tablename_ = 'recensioni'

    COD_FILM = db.Column(db.Integer, db.ForeignKey('film.COD_FILM'),
primary_key=True)
    USERNAME = db.Column(db.String(80), db.ForeignKey('utente.USERNAME'),
primary_key=True)
    VALUTAZIONE = db.Column(db.Float, db.CheckConstraint('VALUTAZIONE IN (0.5, 1,
1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5)'))

@app.route('/test')
def index():
    utente_trovato = Utente.query.filter_by(USERNAME="Codino").first().USERNAME
    film_trovato = Film.query.filter_by(COD_FILM=1).first().COD_FILM

    if utente_trovato and film_trovato :
        cod_film = film_trovato
        return str(cod_film)  # Convert the integer to a string before returning
it.
    else:
        return "Film non trovato"

@app.route('/inserimento', methods=['POST'])
def inserimento():
    data = request.get_json()

    # Example: save data received from frontend in desired manner.
    username = data.get('username')
    nome = data.get('nome')
    cognome = data.get('cognome')
    email = data.get('email')
    password = data.get('password')
    date = data.get('date')
    #role =data.get('role')
    hashed_password = hashlib.sha256(password.encode('utf-8')).hexdigest()

     # Create a new user.
    nuovo_utente = Utente(
        USERNAME=username,
        NOME=nome,
        COGNOME=cognome,
        MAIL=email,
        PASSWORD=hashed_password,
        DATA=date,
        RUOLO="UTENTE"
    )

    # Add user to database and commit changes.
    try:
        db.session.add(nuovo_utente)
        db.session.commit()
        return jsonify(success=True)
    except Exception as e:  # 'Exception' should be lowercase
        return jsonify(success=False, error=str(e))  # Returning an error message
as a JSON response.

@app.route('/film', methods=['POST'])
def inserimento_film():
    data = request.get_json()
    cod_film=data.get('cod_film')
    nome_film = data.get('nome_film')
    descrizione = data.get('descrizione')
    # Creating a new movie.
    nuovo_film = Film(COD_FILM=cod_film,NOME_FILM=nome_film,
DESCRIZIONE=descrizione)

    try:
        # Adding the movie to the database and committing the changes.
        db.session.add(nuovo_film)
        db.session.commit()
        return jsonify(success=True)
    except Exception as e:
        return jsonify(success=False, error=str(e))  # Return of an error message
as a JSON response.

@app.route('/categoria', methods=['POST'])
def inserimento_categoria():
    data = request.get_json()
    id_cat=data.get('id_cat')
    nome_categoria = data.get('nome_categoria')

    # Creating a new category.
    nuova_categoria = Categoria(ID_CATEGORIA=id_cat,NOME_CATEGORIA=nome_categoria)

    # Adding the category to the database and committing the changes.
    try:
        db.session.add(nuova_categoria)
        db.session.commit()
        return jsonify(success=True)
    except Exception as e:
        return jsonify(success=False, error=str(e))  # Returning an error message
as a JSON response.

@app.route('/recensione', methods=['POST'])
def inserimento_recensione():
    data = request.get_json()

    cod_film = data.get('cod_film')
    username = data.get('username')
    valutazione = data.get('valutazione')
    valutazione = float(valutazione)

    # Creating a new review.
    nuova_recensione = Recensioni(COD_FILM=cod_film, USERNAME=username,
VALUTAZIONE=valutazione)
    if Utente.query.filter_by(USERNAME=username).first().USERNAME and
str(Film.query.filter_by(COD_FILM=cod_film).first().COD_FILM) and 0.5 <=
valutazione <= 5:
    # Adding the review to the database and committing the changes.
        try:
            db.session.add(nuova_recensione)
            db.session.commit()
            return jsonify(success=True)
        except Exception as e:
            return jsonify(success=False, error=str(e))  # Returning an error
message as a JSON response.
    else:
        return jsonify(success=False, error="Condizioni non soddisfatte")

@app.route('/categoria_film', methods=['POST'])
def inserimento_categoria_film():
    data = request.get_json()

    id_categoria = data.get('id_categoria')
    cod_film = data.get('cod_film')

    # Creating a new movie category.
    nuova_categoria_film = CategorieDiFilm(ID_CATEGORIA=id_categoria,
COD_FILM=cod_film)
    if str(Categoria.query.filter_by(ID_CATEGORIA=id_categoria).first) and
str(Film.query.filter_by(COD_FILM=cod_film).first):
    # Adding the movie category to the database and committing the changes.
        try:
            db.session.add(nuova_categoria_film)
            db.session.commit()
            return jsonify(success=True)
        except Exception as e:
            return jsonify(success=False, error=str(e))  # Return an error message
as a JSON response.
    else:
        return jsonify(success=False, error="Condizioni non soddisfatte")

@app.route('/login', methods=['POST'])
def login():
    try:
        data = request.get_json()

        username = data.get('name')
        password = data.get('password')
        hashed_password = hashlib.sha256(password.encode('utf-8')).hexdigest()
            # Check if the user exists in the database.
        user = Utente.query.filter_by(USERNAME=username).first().USERNAME
        p = Utente.query.filter_by(USERNAME=username).first().PASSWORD
        if user==username and p==hashed_password:
            return jsonify({'user': user})
            #return render_template('nuovo.html', user=user)
            #Passing 'user' to search.html template.
        else:
            # If login was not successful, you may show an error message.
            return jsonify(success=False, error="Condizioni non soddisfatte")
    except Exception as e:
        print(str(e))  # Print the error to debug
```

```python
        return jsonify({'error': 'Si è verificato un errore interno'}), 500

@app.route('/search', methods=['GET'])
def search_movie():
    try:
        nome=request.args.get('nome')

        film=Film.query.filter_by(NOME_FILM=nome).first()
        #results = Film.query.filter(Film.NOME_FILM.like(f'%{nome}%')).all()
            # If present in db then also present in uploads folder.
        if film:
            film_serialized = {'COD_FILM': film.COD_FILM, 'NOME_FILM':
film.NOME_FILM,'DESCRIZIONE': film.DESCRIZIONE, 'ANNO': film.ANNO}  #convert object
to a dictionary.
            return jsonify({'film': film_serialized})
            #return render_template('search.html', film=film)  # pass 'film' to
search.html template.
        else:
            return jsonify({'film non trovato'})
    except Exception as e:
        print(str(e))  # print error for debugging
        return jsonify({'error': 'Si è verificato un errore interno'}), 500


if _name_ == '_main_':
@app.route('/films', methods=['GET'])
def get_all_films():
    try:
        films = Film.query.all()  # Ottieni tutti i film dalla tabella

        # Serializza la lista dei film in un formato comprensibile per la risposta
JSON
        serialized_films = [{
            'COD_FILM': film.COD_FILM,
            'NOME_FILM': film.NOME_FILM,
            'DESCRIZIONE': film.DESCRIZIONE,
            'ANNO': film.ANNO
        } for film in films]

        return jsonify({'films': serialized_films})  # Restituisci la lista di film
in formato JSON
    except Exception as e:
        print(str(e))  # Stampa l'errore per il debug
        return jsonify({'error': 'Si è verificato un errore interno'}), 500
app.run(debug=True, host='0.0.0.0', port=1200)
```