# **Movie Randomizer Special Coding Considerations**

## 🧠 Special Coding Considerations (Design choices, constraints, or techniques used to meet project requirements or improve behavior)

### 1. Client-Side Validation

* All form validation (login, register) is done using **JavaScript only**.
* No HTML5 required, type="email", etc. used.
* Example: register.js uses regex for email and custom password checks.

### 2. AJAX (fetch API) Instead of Page Reloads

* All add/remove/search actions happen via fetch() (AJAX), not full form submissions.
* This provides a smoother UX and avoids reloading pages unnecessarily.

### 3. Dynamic Content Rendering

* collection.js and spin.js wait for image loading and animate or reveal results dynamically.
* Movie data is inserted directly into the DOM using innerHTML or element creation.

### 4. Separation of Concerns

* HTML pages are in /htmls or main folder.
* CSS in /Styles, JS in /Js, PHP backend logic in /Server, database scripts in /Database.

### 5. Security Considerations

* Passwords stored as hashed strings using password\_hash() (PHP).
* Inputs checked for duplicates before insert.

### 6. Session-Based Access Control

* Pages like main\_menu.php, watch.php, and manage\_collection.php check for $\_SESSION["user\_id"].
* Redirect to login if not set.

### 7. Live Search State Sync

* After adding/removing a movie, the search input is re-triggered to re-apply filters: ***searchInput.dispatchEvent(new Event("input"));***