

# Also Section 1: Background Research

## 1.1: Analysis of the TCP protocol and the TCP/IP model

### The TCP/IP Model

The TCP/IP model is a conceptual framework that organizes network protocols into four functional layers:

1. **Network Access Layer** (Link Layer): Handles the physical connection to the network and hardware addressing
2. **Internet Layer**: Manages logical addressing and routing of packets between networks
3. **Transport Layer**: Provides end-to-end communication services between applications
4. **Application Layer**: Enables user applications to access network services

### TCP Protocols

TCP (Transmission Control Protocol) is a cornerstone of the modern Internet. As a connection-oriented protocol, TCP is responsible for ensuring that data sent between computers is delivered reliably and in order. It plays a critical role in client/server applications by underpinning many of the everyday services we use, from web browsing to email.

### TCP Protocols vs. TCP/IP Model

It's important to distinguish between these two related terms:

- **TCP/IP Model**: The entire four-layer conceptual framework described above
- **TCP Protocols**: Specifically refers to the Transmission Control Protocol, which operates at the Transport Layer of the model

The TCP protocol is just one component of the broader TCP/IP suite, albeit a crucial one that gave the model part of its name.

### TCP/IP vs. OSI Model

The TCP/IP model differs from the seven-layer OSI (Open Systems Interconnection) model. Here's how they correspond:

OSI Model	TCP/IP Model
Application Layer	Application Layer
Presentation Layer	Application Layer
Session Layer	Application Layer
Transport Layer	Transport Layer
Network Layer	Internet Layer
Data Link Layer	Network Access Layer
Physical Layer	Network Access Layer

The TCP/IP model is more pragmatic and implementation-focused, while the OSI model (which predates the TCP/IP model and the Internet itself) is more theoretical and comprehensive.

## The Internet Hourglass Model

The Internet architecture is often visualized as an hourglass:

- At the top: Many applications (HTTP, FTP, SMTP, etc.)
- At the narrow waist: IP (Internet Protocol)
- At the bottom: Many physical networks (Ethernet, Wi-Fi, fiber, etc.)

TCP sits in the layer just above the narrow waist. The hourglass model illustrates how IP serves as a universal interconnection protocol that allows many applications to run over many different physical networks.

The narrow waist design provides modularity: innovations can occur above and below IP without changing the core architecture, enabling the Internet's remarkable growth and adaptability.

## TCP vs. UDP

Both TCP and UDP are Transport Layer protocols, but they serve different purposes:

### TCP (Transmission Control Protocol):

- Connection-oriented
- Reliable delivery with acknowledgments and retransmissions
- Flow control and congestion management
- In-order delivery of data

- Error detection and correction
- Ideal for applications requiring accuracy (web browsing, file transfers, email)

#### **UDP (User Datagram Protocol):**

- Connectionless
- No guarantee of delivery, ordering, or duplicate protection
- No flow control or congestion management
- Minimal header overhead
- Faster with lower latency
- Preferred for applications valuing speed over reliability (video streaming, online gaming, DNS lookups)

The choice between TCP and UDP depends on application requirements: reliability and completeness (TCP) versus speed and efficiency (UDP).

## **1.2: Existing Site Analysis (Letterboxd)**

**Letterboxd** is a web-based platform designed primarily for film enthusiasts. It allows users to keep a record of the films they have watched, rate and review movies, and create personalized lists. The site incorporates social features—such as following other users and engaging with reviews—that enable community interaction without overwhelming the core functionality. Its design makes use of modern web technologies (CSS and JavaScript) to deliver a dynamic interface and an interactive user experience, including search and filtering tools to help users quickly locate films or related content.

### **Layout Responsiveness:**

The idea behind responsive design is to provide a seamless experience across desktops, tablets, and smartphones by dynamically rearranging and scaling content. Responsive design uses fluid grids, flexible images, and CSS media queries to adjust a website's layout in real time based on the browser's dimensions.

Closely related is the concept of adaptive design, which involves creating several distinct layouts for specific device sizes. The website detects the visitor's device—often through the user agent—and then delivers the most appropriate fixed layout. While adaptive design can offer a highly optimized experience for each device type, it is generally less flexible than responsive design.

## Responsiveness of Letterboxd

Surprisingly, the Letterboxd website seems to be lacking with regards to layout responsiveness, because when the browser window (on a desktop) is resized and the website's layout doesn't adjust—causing elements like the edges of a large image to simply go out of view— which seems to indicate that the design is fixed. In this scenario, the site doesn't use responsive or adaptive techniques to reflow or scale its content based on the viewport size. Instead, the dimensions are set in absolute values (such as pixels), so when the viewport shrinks, parts of the content are effectively cropped rather than repositioned or resized to fit the new window dimensions.

A search through the `head` tag of the homepage reveals the following viewport `meta` tag:

```
<meta name="viewport" content="width=1024" />
```

which seems to indicate that the desktop version of the website is designed with a fixed width in mind.

The layout does adapt to the smaller form-factor of mobile phones and tablets. On larger screens (for eg. desktop), the page is laid out in multiple columns (a main content column plus a sidebar), whereas on smaller screens, the sidebar is merged into the main content flow.

In terms of web-terminology, at wider breakpoints (desktop), the layout is multi-column, with the main content area and a separate right sidebar. At narrower breakpoints (mobile), CSS media queries trigger a single-column layout, causing the sidebar's content to flow into (or stack beneath) the main column. Since the meta viewport is locked to `width=1024` on a desktop, effectively the fluid media queries are being bypassed, and the browser continues rendering the layout as if the viewport is 1024px wide—even if the desktop window is physically shrunk below that.

## Navigation

Letterboxd employs a clear, globally consistent navigation system that helps users quickly access the site's core areas. On desktop, the primary navigation bar is positioned in the header and presents key sections (such as Home, Films, Lists, Members, and Journal) as horizontally arranged links. Account-related actions (e.g. Sign in and Create account) and a dedicated search field are also prominently featured, contributing to intuitive site exploration.

In the mobile version, the navigation adapts to smaller screens by reflowing or merging sidebar elements into the main content flow. This results in a single-column layout or a collapsible menu that maintains accessibility while reducing clutter. The design uses visual cues (like hover effects and iconography) to reinforce the interactive nature of navigation elements, enhancing the overall usability of the site.

## Client-side technologies: Javascript

JavaScript plays a crucial role in enhancing the functionality and user experience of the Letterboxd website.

- **Enabling Interactivity:** JavaScript allows users to interact with the site seamlessly. For example, clicking buttons (such as "Log" to add a movie) or opening dropdown menus (like privacy settings) triggers dynamic responses without requiring a full page reload. It also powers features like tooltips, which provide additional information when you hover over certain elements.
- **Dynamic Content Loading:** Instead of loading all content at once, JavaScript ensures that images (such as movie posters) are loaded only when they come into view. This improves page performance and reduces initial load times. It also fetches new data (e.g., reviews, lists) in the background, allowing the page to update dynamically without refreshing.
- **Form Handling and Validation:** JavaScript enhances form interactions by providing autocomplete suggestions as you type in search bars or tag fields. It also validates user input (e.g., ensuring required fields are filled out) before submitting forms, improving data accuracy and user experience.
- **Session and State Management:** JavaScript manages user sessions, such as tracking whether a user is logged in. This determines which features are visible (e.g., the "Log" button for diary entries). It also stores user preferences (e.g., mobile site settings) using cookies or local storage, ensuring a consistent experience across visits.
- **Analytics and Tracking:** JavaScript integrates with tools like Google Analytics to track user behavior, such as page views and interactions. This data helps the site understand usage patterns and improve its offerings.
- **Ad Management:** JavaScript handles the loading and display of advertisements. It also provides options for users to upgrade to an ad-free experience.
- **Interactive Components:** JavaScript powers interactive elements like star rating systems, where users can click stars to rate movies, and custom dropdown menus for selecting options.
- **Cross-Browser Compatibility:** JavaScript ensures the site functions correctly across different browsers and devices by loading appropriate code for each environment.
- **Embedded Content:** JavaScript manages embedded content, such as YouTube videos, ensuring they play correctly and integrate smoothly with the rest of the page.
- **Performance Optimization:** JavaScript improves site performance by loading resources efficiently (e.g., preloading scripts) and only fetching data when needed.

## Dynamic Content Loading using Javascript

In this section, a deeper dive will be taken on **one** particular role of Javascript on the Letterboxd website, that of "lazy image loading".

The following code fragment (with less relevant bits snipped out for clarity) is included in one of the scripts loaded by the page:

```
loadReallyLazyPosters: function(n) {
  n instanceof jQuery || (n = $(n));
  const c = ss(); // Get some configuration or state
  const l = d => $("body.person-yir").data("owner") || ...; // Determine poster owner
  based on context

  n.each(function() {
    var d = $(this);
    d.removeClass("really-lazy-load");
    var h = d.data("type"), g = d.data("item-id"), y = d.data("film-slug"), ...; //
    Extract data attributes
    // Determine poster context (e.g., optimised, custom, etc.)
    if (person && person.optimisedPosters) ...;
    else {
      I || (I = "std");
      var T = ...; // Check for alternative or custom posters
      T === "optimised" ? ... : T && ...;
    }
    // Construct AJAX URL for fetching the poster
    var v = d.data("film-adult"),
        A = baseUrl + "/ajax/poster/" + h + "/" + y + "/" + I + "/" + p + "x" + b + ...,
        E = d.data("cache-busting-key");
    E && (A = A + "?k=" + E);
    // Make AJAX request to load the poster
    $.ajax({
      url: A,
      success: function(m, w, C) {
        const L = d[0].querySelector("img");
        var _ = $(m), V = _.find("img").attr("srcset");

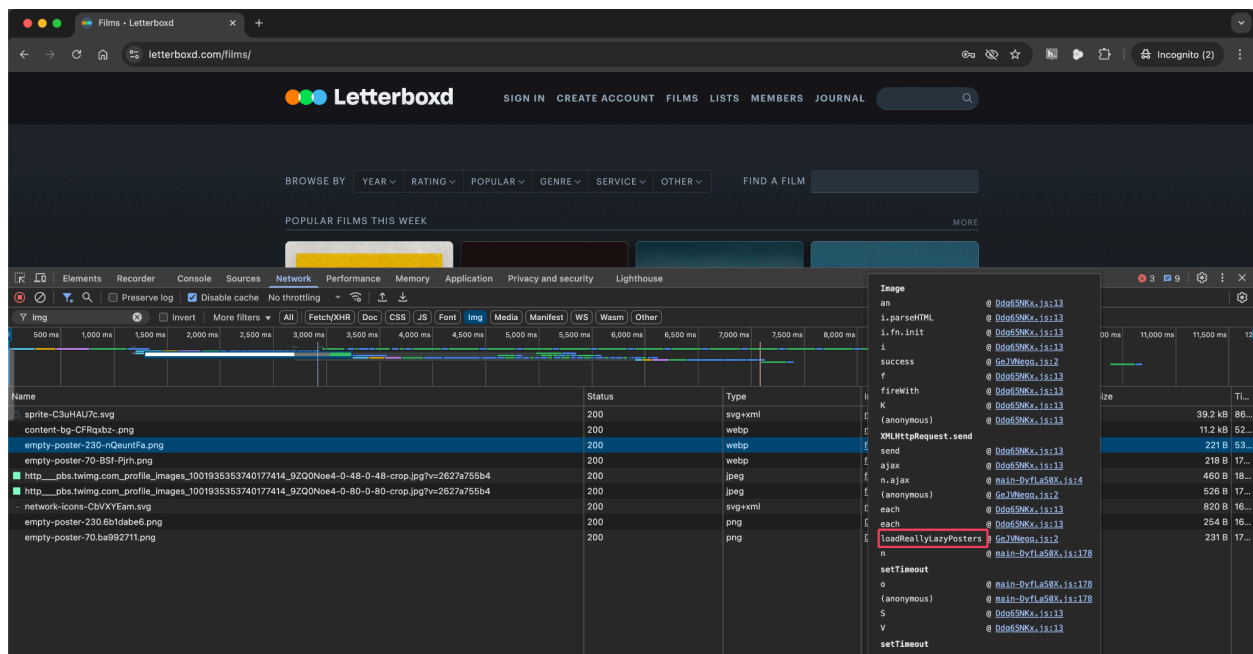
        // Update the DOM with the loaded image
        if (d.hasClass("react-component")) ...; // Handle React components
        else {
          _.addClass(d.attr("class"), ...; // Replace placeholder with new image
          _.insertAfter(d), d.remove(), bxd.layout(_);
        }

        // Trigger custom event for image source change
        const H = d[0].offsetParent ? d[0] : _[0],
            W = H.querySelector("img");
        L && W.src !== L.src && R("postersrcchange", H, {
          img: W,
          src: { newValue: W.src, oldValue: L.src },
          ...W.srcset && { srcset: { newValue: W.srcset, oldValue: L.srcset } }
        });
      }
    });
  });
},
```

Explanation:

The code manipulates elements with the `really-lazy-load` class by dynamically loading poster images via AJAX requests and updating the DOM. It first removes the `really-lazy-load` class from the elements and extracts relevant data attributes (e.g., image type, dimensions, and context) to construct a URL for fetching the poster. Once the image is retrieved, the code replaces the placeholder element with the loaded image, updates its attributes (e.g., `src` and `srcset`), and ensures proper layout adjustments. This approach improves performance by loading images only when needed, enhancing the user experience.

The code was discovered by using Chrome Developer Tools, using the Network tab, forcing a page reload with the cache disabled and then hovering over one of the javascript files in the **Initiator** column. This helped trace the network requests back to the specific function highlighted in the screenshot.



## Client-side technologies: CSS

Cascading Style Sheets (CSS) play a critical role in modern websites like Letterboxd, defining the visual presentation, layout, and responsiveness of the page. CSS is responsible for styling text, arranging elements, creating visual hierarchies, and ensuring a consistent user experience across devices. Without CSS, a webpage loses its polished appearance and functionality, reverting to a raw, unstyled document.

To demonstrate what a stark difference when a webpage is stripped of all styling, I ran the javascript statement `document.querySelector("head").remove()`; which removes the `<head>` element which contains links to all the external stylesheets used by the website.

The screenshot shows the Letterboxd homepage with all styling removed. The browser address bar shows 'letterboxd.com'. The page content is unformatted, with links like 'Sign in', 'Create account', 'Films', 'Lists', 'Members', and 'Journal' appearing as plain text. A search bar is present. The main heading 'Letterboxd — Your life in film' is underlined. Below it, a list of links is shown. A search bar with the text 'Search:' and a 'Search' button is visible. The main content area features a list of films, including 'Mickey 17 (2025)' and 'Paddington in Peru'. Each film entry includes a poster image, the title, and statistics like 'Views: 804.03k' and 'Likes: 287.24k'. At the bottom, the browser's developer console is open, showing the command `document.querySelector("head").remove();` executed.

Letterboxd — Your life in film

- [Sign in](#)
- [Create account](#)
- [Films](#)
- [Lists](#)
- [Members](#)
- [Journal](#)



Search:  Search

[Mickey 17 \(2025\)](#)

**Track films you've watched.**  
**Save those you want to see.**  
**Tell your friends what's good.**

[Get started — it's free!](#)

The social network for film lovers. Also available on [iOS](#), [Apple TV](#) and [Android](#)

-   
◦ Views: 804.03k  
◦ Likes: 287.24k  
[Mickey 17 \(2025\)](#)
-   
◦ Views: 173.82k  
◦ Likes: 52.97k

Elements Recorder Console Sources Network Performance Memory Application Privacy and

top Filter

```
> document.querySelector("head").remove();
```

As evident in the screenshot, when the `<head>` element is removed from the Letterboxd homepage, the page undergoes a dramatic transformation. The navigation bar, which typically



features styled links and a search bar, becomes a simple list of plain text links. The search bar reverts to a basic input box with a default button, devoid of any custom styling. Film titles and descriptions, such as "Mickey 17 (2025)," appear as unstyled text, losing their visual emphasis and hierarchy. Metrics like "Views" and "Likes" are displayed as plain text without any formatting or alignment.

The layout collapses into a vertical stack of elements, as the grid and flexbox structures are no longer applied. Images remain visible but lack borders, margins, or responsive sizing. Interactive elements, such as dropdowns or lazy-loaded content, cease to function due to the removal of associated scripts. The result is a stark contrast to the polished, user-friendly interface of the original site, highlighting the essential role of CSS in creating an engaging and functional web experience.

This simple demonstration with the accompanying screenshot, highlights the role and importance that CSS plays in transforming raw HTML into a visually appealing and interactive interface.

## Search Facility

Letterboxd's search functionality is a powerful tool designed to help users navigate its extensive database of films, reviews, lists, members, and more. The search box, located near the top of the page, allows users to enter queries, though it does not support suggest-as-you-type functionality. Instead, users must press enter or click the search button to view results.

- **Search Results and Filtering:** Search results are displayed in a structured format, with options to filter by type using the "Show Results for" section on the right. Users can narrow results to specific categories, such as films, reviews, lists, members, tags, journal articles, and podcast episodes. For example, searching for "Toy Story" returns a list of films with details like release year, alternative titles, and director information.
- **Advanced Search Features:** Letterboxd's advanced search capabilities, detailed in their FAQ, allow users to refine queries using:
  - Search Operators: - Quotation marks for exact phrases (e.g., "Toy Story").
  - Boolean operators (`AND`, `OR`, `NOT`) to combine or exclude terms.
  - Wildcards (`\*`) to match partial terms (e.g., `anim\*` for "animation" or "animated").
  - Field-Specific Searches such as by Films, Lists, Members, Tags, etc.
  - Date Range Filters: Filter results by year or date ranges (e.g., `year:1995`).
  - Full-Text Search: Search within reviews, journal articles, and podcast episodes for detailed discussions.

Users can combine multiple filters and operators for highly specific queries.

## Review and Rating Facilities

Letterboxd positions itself as the “the social network for film lovers”, i.e. a platform where users can review, rate, and share films they have watched. Users can give movies a rating on a five-star scale (with half-star options) and write reviews that range from short comments to longer, detailed posts. Reviews can be tagged for easier discovery, saved as drafts for later editing, and include spoiler warnings if needed. Additionally, other users can like and comment on reviews, which helps highlight popular or useful feedback.

In addition to reviews, Letterboxd lets users keep a film diary by logging movies along with the dates and their ratings. Users can also create custom lists of films, whether organized by theme or ranking, and share these lists with others who can then like or comment on them. These features make it simple for users to track their movie-watching history, find new films, and engage with a community of other film enthusiasts.

## 2: Server-side Design

### 2.1 Examples of Class Data

#### Actor

Attribute	Datatype	Example 1	Example 2
firstName	String	"Tom"	"Emma"
lastName	String	"Hanks"	"Watson"
dateOfBirth	Date	1956-07-09	1990-04-15
nationality	String	"American"	"British"
isActive	Boolean	true	false

---

#### Movie

Attribute	Datatype	Example 1	Example 2
name	String	"Inception"	"Stranger Things"
director	String	"Christopher Nolan"	"The Duffer Brothers"
genre	String	"Sci-Fi"	"Thriller"
releaseDate	Date	2010-07-16	2016-07-15
pgRating	String	"PG-13"	"TV-MA"
description	String	"A mind-bending thriller about dreams within dreams."	"A supernatural thriller set in the 1980s."
recommendationDescription	String	"Highly recommended for fans of complex narratives."	"Recommended for suspense and strong character development."

Attribute	Datatype	Example 1	Example 2
recommendationScore	double	9.0	8.5
averageRating	double	8.7	8.2
isMovie	Boolean	true	false

---

## User

Attribute	Datatype	Example 1	Example 2
firstName	String	"Alice"	"Bob"
lastName	String	"Smith"	"Johnson"
dateOfBirth	Date	1995-03-22	1988-11-02
country	String	"USA"	"Canada"
email	String	" <a href="mailto:alice@example.com">alice@example.com</a> "	" <a href="mailto:bob@example.com">bob@example.com</a> "
userName	String	"alice95"	"bobby88"
password	String	"hashedPwd1"	"hashedPwd2"
preferredMovie	String	"The Matrix"	"Interstellar"
preferredTVShow	String	"Breaking Bad"	"Game of Thrones"

---

## UserReview

Attribute	Datatype	Example 1	Example 2
authorName	String	"Alice"	"Bob"
title	String	"Amazing movie"	"Not as good as expected"

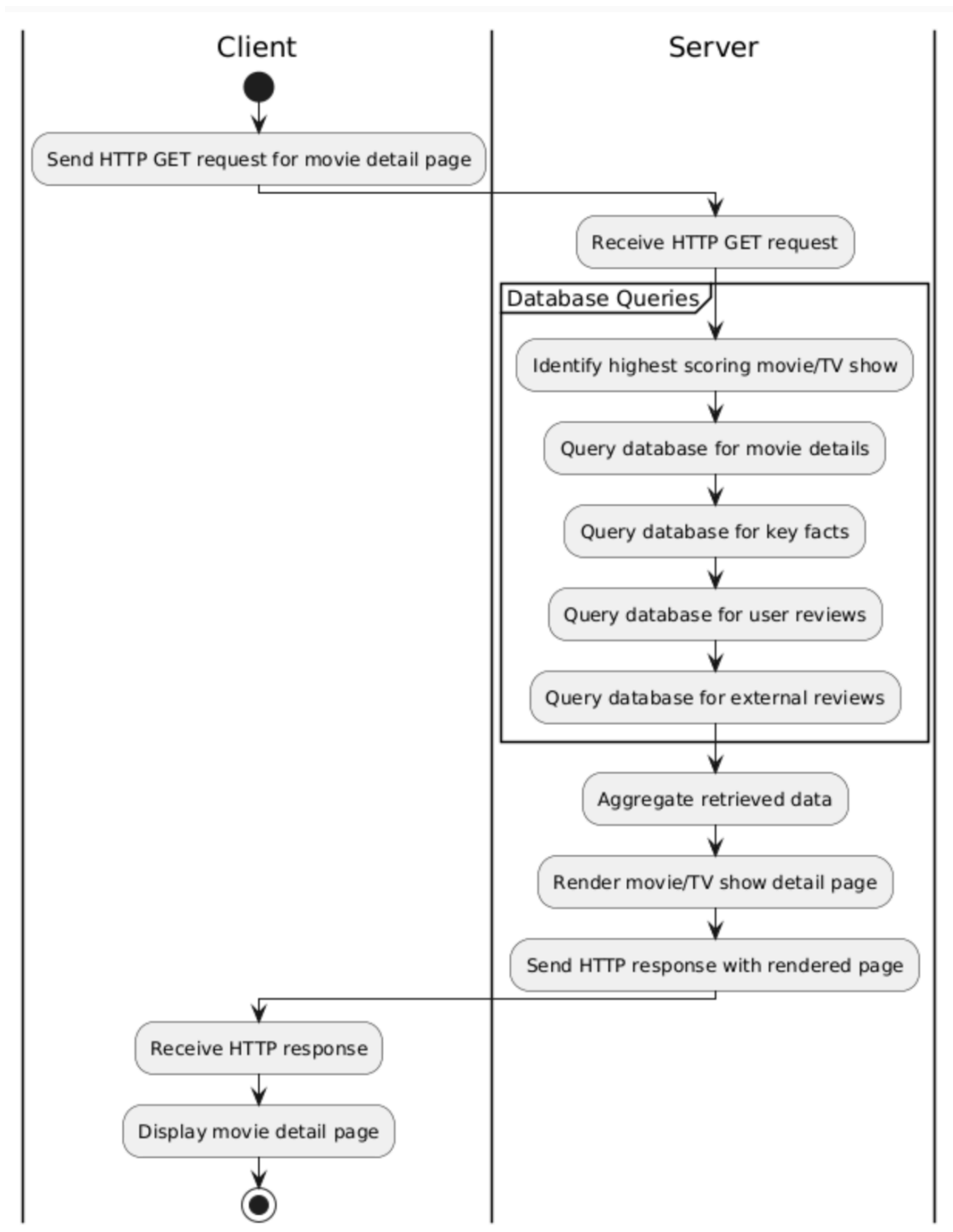
Attribute	Datatype	Example 1	Example 2
content	String	"I loved the visual effects and storyline."	"The plot was confusing and pacing felt off."
score	double	9.0	6.5

---

### ExternalReview

Attribute	Datatype	Example 1	Example 2
sourceName	String	"IMDB"	"Rotten Tomatoes"
sourceLocation	String	" <a href="https://www.imdb.com/title/tt1375666/">https://www.imdb.com/title/tt1375666/</a> "	" <a href="https://www.rottentomatoes.com/m/inception">https://www.rottentomatoes.com/m/inception</a> "
isOnline	Boolean	true	true
rating	double	8.8	8.5
description	String	"Critically acclaimed for its originality and execution."	"Praised for its direction and cinematography."

## 2.2 Server Activity Diagram



The diagram above shows the client-server interaction flow for displaying a movie detail page in the streaming service website:

1. **Client Side** (left column):
  - The process begins with the client sending an HTTP GET request for a movie detail page
  - After server processing, the client receives an HTTP response
  - Finally, the client displays the movie detail page to the user
2. **Server Side** (right column):
  - The server receives the HTTP GET request
  - It identifies the highest scoring movie/TV show (using a database query)
  - Then performs a series of database queries (shown in a boxed section):
    - Query for movie details
    - Query for key facts about the movie
    - Query for user reviews
    - Query for external reviews
  - The server aggregates all the retrieved data
  - It renders the movie/TV show detail page with the compiled information
  - Finally, it sends an HTTP response containing the rendered page back to the client

## 2.3 Site Map

The functionality of the movie/TV recommendation web application is split across a set of PHP pages as described below. A visual representation of the conceptual sitemap (showing the relationship between the PHP pages) as well as a UI/link-structure-centric sitemap are shown later.

## Movie Recommendation System Website Structure

### Home Page (home.php)

- **Description:** Main landing page showcasing featured content
- **Content:**
  - Featured carousel of top 3 movies/TV shows with thumbnail images
  - Brief summary information for each (title, genre, your rating)
  - Navigation menu (global header)
  - Login/profile status indicator
  - Banner link to streaming service overview page
- **Links to:**
  - Individual movie/TV show pages
  - Streaming service overview
  - User login/registration
  - Rankings page

## Individual Movie/TV Show Pages (movie.php?id=[movie\_id])

- **Description:** Detailed view for a specific movie or TV show
- **Content:**
  - Hero image/poster
  - Title and key facts (director, cast, release date, age rating, genre, duration)
  - Plot synopsis
  - Editorial recommendation (your review)
  - Recommendation score (out of 10)
  - User reviews section (with average user rating)
  - External review links (3 official sources)
  - Related content recommendations
- **Links to:**
  - Home page
  - Rankings page
  - User review submission form (if logged in)
  - External review sources
  - Streaming service overview

## Rankings Page (rankings.php)

- **Description:** Sortable table of all movies/TV shows
- **Content:**
  - Filterable/sortable table with columns:
    - Title
    - Genre
    - Release date
    - Your recommendation score
    - Average user score
    - PG rating
  - Sort controls for each column
  - Brief info for each entry
- **Links to:**
  - Individual movie/TV show pages
  - Home page
  - Streaming service overview

## Streaming Service Overview (service.php)

- **Description:** Information about the chosen streaming service
- **Content:**
  - Brand information and description
  - Subscription plans and pricing
  - Platform availability (mobile, web, smart TV)
  - Featured content highlights



- Multimedia content (videos, images)
- Service advantages and special features
- **Links to:**
  - Home page
  - Subscription signup page
  - App download links
  - Individual movie/TV show pages

### **User Login/Registration Pages (login.php, register.php)**

- **Description:** Authentication pages
- **Content:**
  - Login form (username/password)
  - Registration form (username, password, email, date of birth, etc.)
  - Password recovery option
- **Links to:**
  - Home page
  - Registration page (from login)
  - Login page (from registration)

### **User Profile Page (profile.php)**

- **Description:** User account management
- **Content:**
  - Profile information (username, email, etc.)
  - Favorite movie/TV show selection
  - List of user's reviews
  - Account settings
- **Links to:**
  - Home page
  - User's reviews
  - Edit profile page
  - Favorite movie/TV show pages

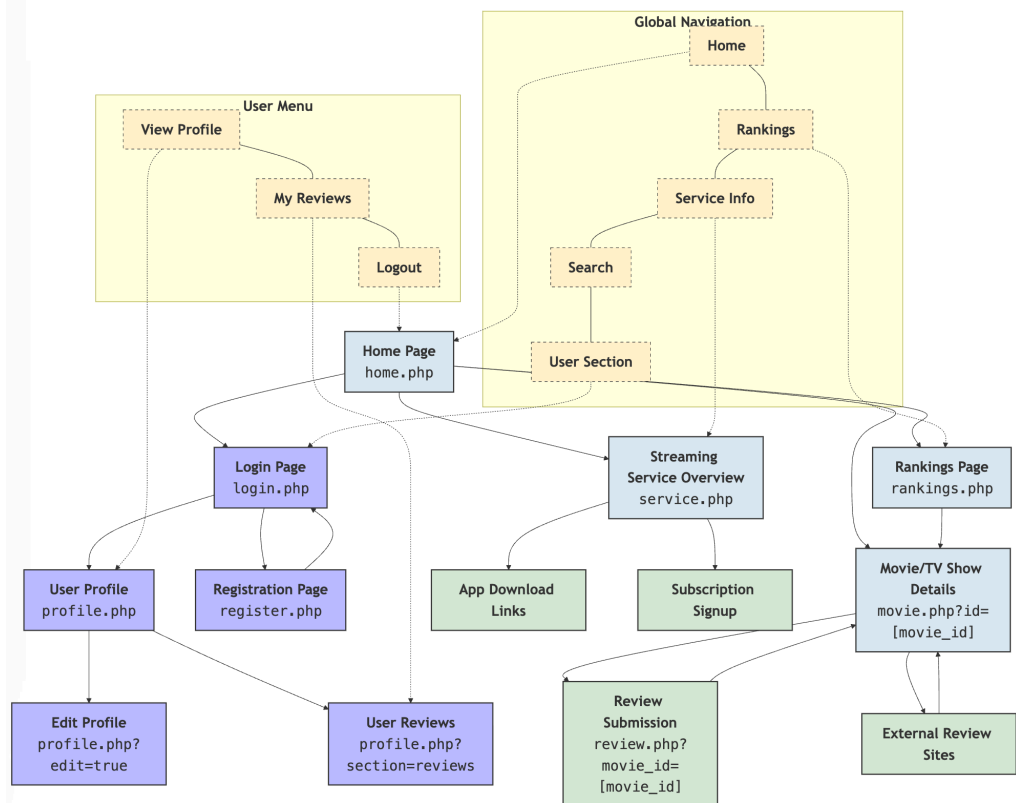
### **Review Submission Form (review.php?movie\_id=[movie\_id] or embedded in movie.php)**

- **Description:** Form to submit movie/TV show reviews
- **Content:**
  - Movie/TV show information
  - Rating input (out of 10)
  - Review title field
  - Review content text area
  - Submit button
- **Links to:**
  - Individual movie/TV show page

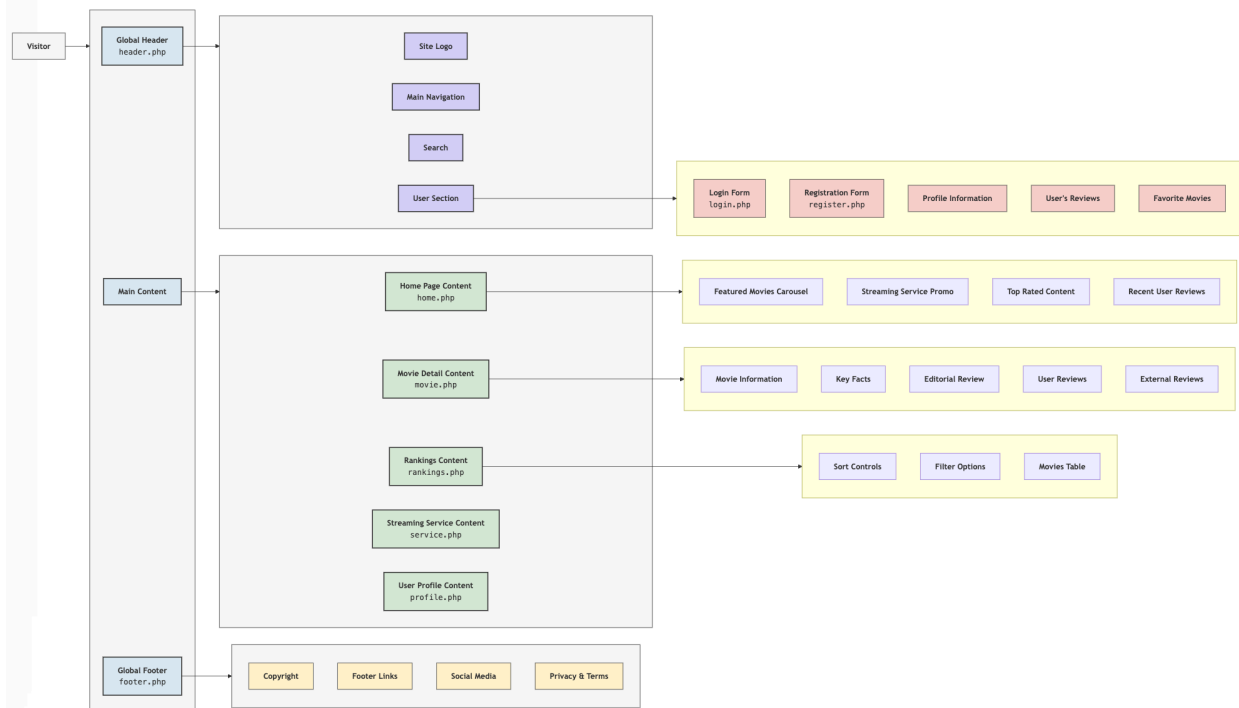
- User profile

## Navigation Structure

- **Global Header** (present on all pages):
  - Logo (links to home)
  - Main navigation menu:
    - Home
    - Rankings
    - Streaming Service Info
  - Search function
  - User section (login/register or username + dropdown)
- **Global Footer** (present on all pages):
  - Copyright information
  - About/Contact links
  - Privacy policy
  - Terms of service
- **User Dropdown Menu** (when logged in):
  - View Profile
  - My Reviews
  - Logout
- **Breadcrumb Navigation** (on content pages):
  - Shows current location in site hierarchy



## Conceptual SiteMap



UI-centric sitemap showing how the various links might be laid out on the page of the desktop version of the app.

# Request-Response Flows for Key Interactions

## 1. Home Page to Movie Details Flow

Request:

- **Type:** HTTP GET
- **URL:** `movie.php?id=123`
- **Parameters:** `id` (movie identifier)
- **Initiated by:** User clicking on a movie thumbnail/link

Response:

- **Status Code:** 200 OK
- **Content Type:** text/html
- **Data Returned:**
  - Fully rendered HTML page containing:
    - Movie title, poster, and basic information
    - Key facts (director, cast, release date, etc.)
    - Editorial recommendation and score
    - External review links
    - User reviews (if any)
    - Login prompt or review form (based on authentication status)

Server-Side Processing:

1. Process GET request and extract movie ID
2. Query database for movie details
3. Query database for key facts
4. Query database for editorial review
5. Query database for user reviews
6. Query database for external reviews
7. Aggregate all retrieved data
8. Render complete page with all components
9. Return HTML response to client

## 2. Rankings Page Interaction Flow

Initial Request:

- **Type:** HTTP GET
- **URL:** `rankings.php`
- **Parameters:** None (default sorting)

- **Initiated by:** User clicking "Rankings" in navigation

Initial Response:

- **Status Code:** 200 OK
- **Content Type:** text/html
- **Data Returned:**
  - HTML page with sortable table of all movies/TV shows
  - Default sorting by recommendation score

Sorting Request:

- **Type:** HTTP GET
- **URL:** `rankings.php?sort=user_rating&order=desc`
- **Parameters:**
  - `sort`: Field to sort by (e.g., user\_rating, year, title)
  - `order`: Sort direction (asc/desc)
- **Initiated by:** User clicking on column header or sort control

Sorting Response:

- **Status Code:** 200 OK
- **Content Type:** text/html
- **Data Returned:**
  - Updated HTML with re-sorted table data
  - Active sort indicators on relevant columns

Server-Side Processing:

1. Process request (with or without sorting parameters)
2. Query database for all movies/shows with appropriate ORDER BY clause
3. Generate sortable table with all entries
4. Add sort controls and indicators
5. Return HTML response to client

### 3. User Authentication Flow

Login Page Request:

- **Type:** HTTP GET
- **URL:** `login.php`
- **Parameters:** None
- **Initiated by:** User clicking "Login" button

Login Page Response:

- **Status Code:** 200 OK

- **Content Type:** text/html
- **Data Returned:** HTML page with login form

Login Submission:

- **Type:** HTTP POST
- **URL:** `login.php`
- **Parameters:** None in URL
- **Body Data:**
  - `username`: User's login name
  - `password`: User's password
- **Initiated by:** User submitting the login form

Successful Login Response:

- **Status Code:** 302 Found
- **Headers:**
  - `Location`: URL to redirect to (home.php or previous page)
  - `Set-Cookie`: Session data
- **Data Returned:** Redirect headers only

Failed Login Response:

- **Status Code:** 401 Unauthorized
- **Content Type:** text/html
- **Data Returned:** Login form with error message

Server-Side Processing:

1. Process GET request to show form or POST request for authentication
2. For POST: validate credentials against database
3. If valid: create session, set cookies, redirect to home
4. If invalid: return login form with appropriate error

## 4. User Review Submission Flow

Review Form Request:

- **Type:** HTTP GET
- **URL:** `review.php?movie_id=123`
- **Parameters:** `movie_id` (movie being reviewed)
- **Initiated by:** User clicking "Write a Review" button
- **Requires:** Authenticated session

Review Form Response:

- **Status Code:** 200 OK (if authenticated) or 302 Found (if not authenticated)

- **Content Type:** text/html
- **Data Returned:**
  - If authenticated: HTML form for submitting review
  - If not authenticated: Redirect to login page

#### Review Submission:

- **Type:** HTTP POST
- **URL:** `review.php`
- **Parameters:** None in URL
- **Body Data:**
  - `movie_id`: ID of movie being reviewed
  - `rating`: Numeric score (1-10)
  - `title`: Review headline
  - `content`: Full review text
- **Initiated by:** User submitting the review form

#### Successful Submission Response:

- **Status Code:** 302 Found
- **Headers:** `Location: movie.php?id=123` (return to movie page)
- **Data Returned:** Redirect headers only

#### Failed Submission Response:

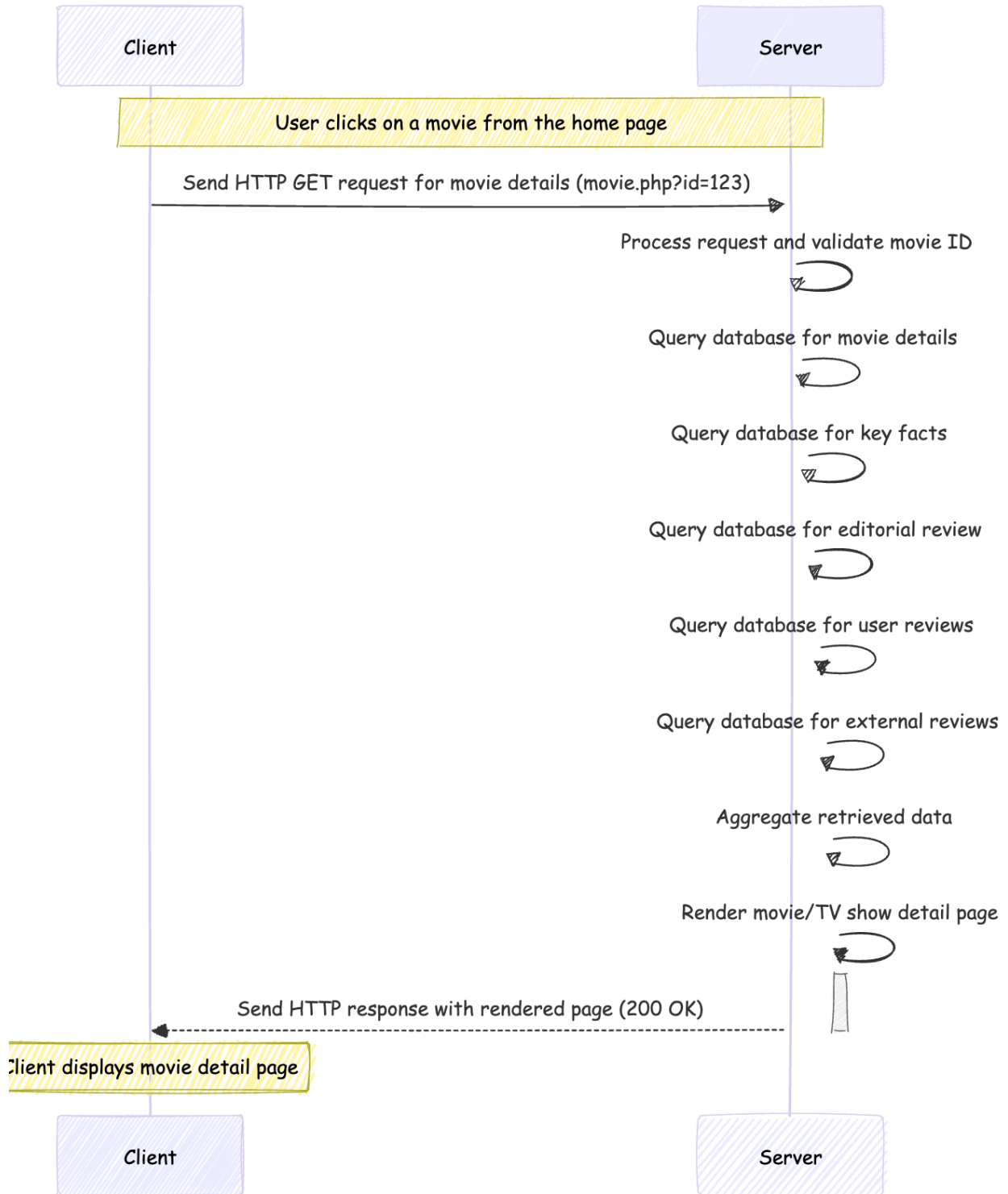
- **Status Code:** 400 Bad Request
- **Content Type:** text/html
- **Data Returned:** Form with validation errors

#### Server-Side Processing:

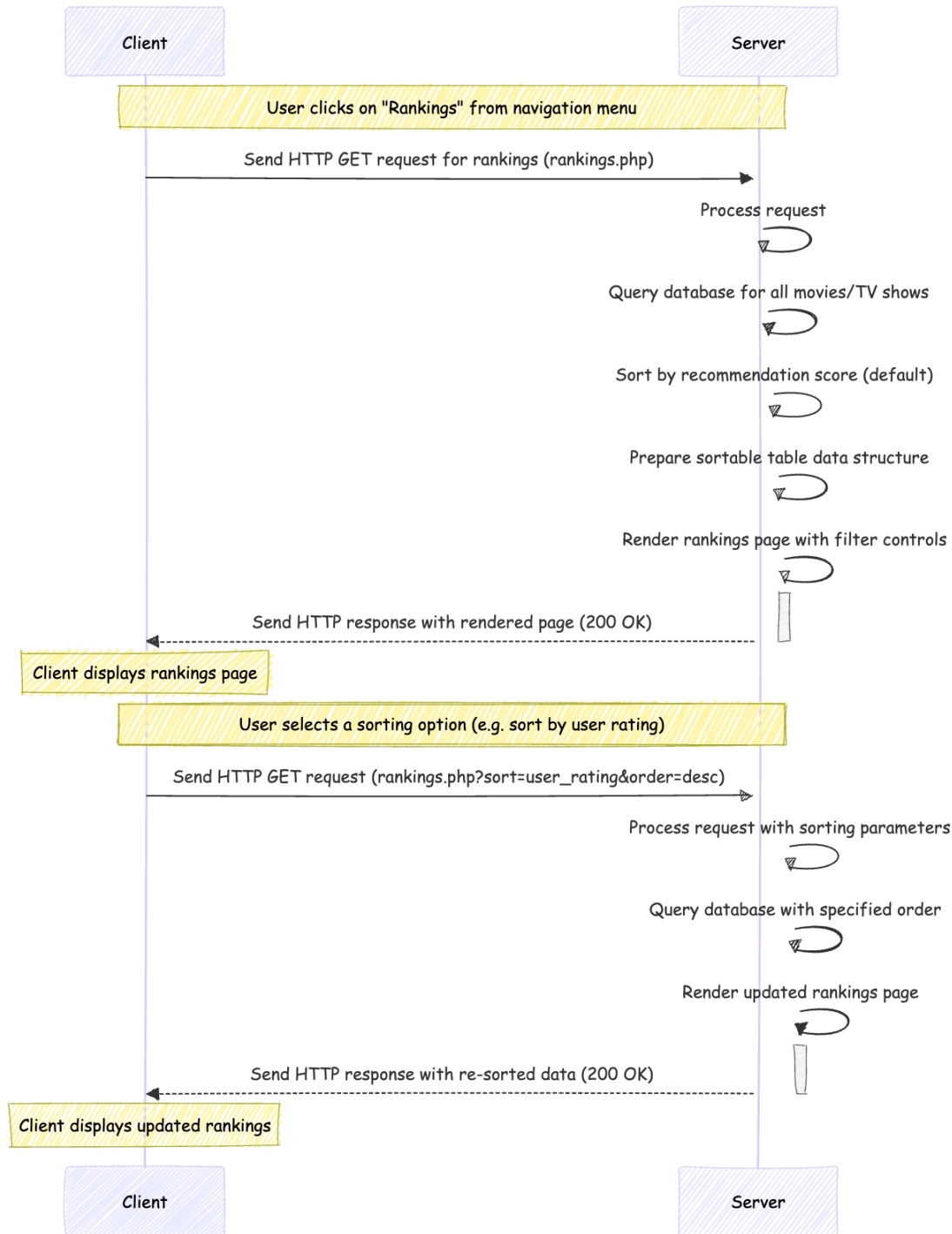
1. Process GET request (showing form) or POST request (submitting review)
2. For GET: verify authentication, check if user already reviewed this movie
3. For POST: validate all form data
4. If valid: store review in database, update average ratings
5. Redirect back to movie details page which will now include the new review

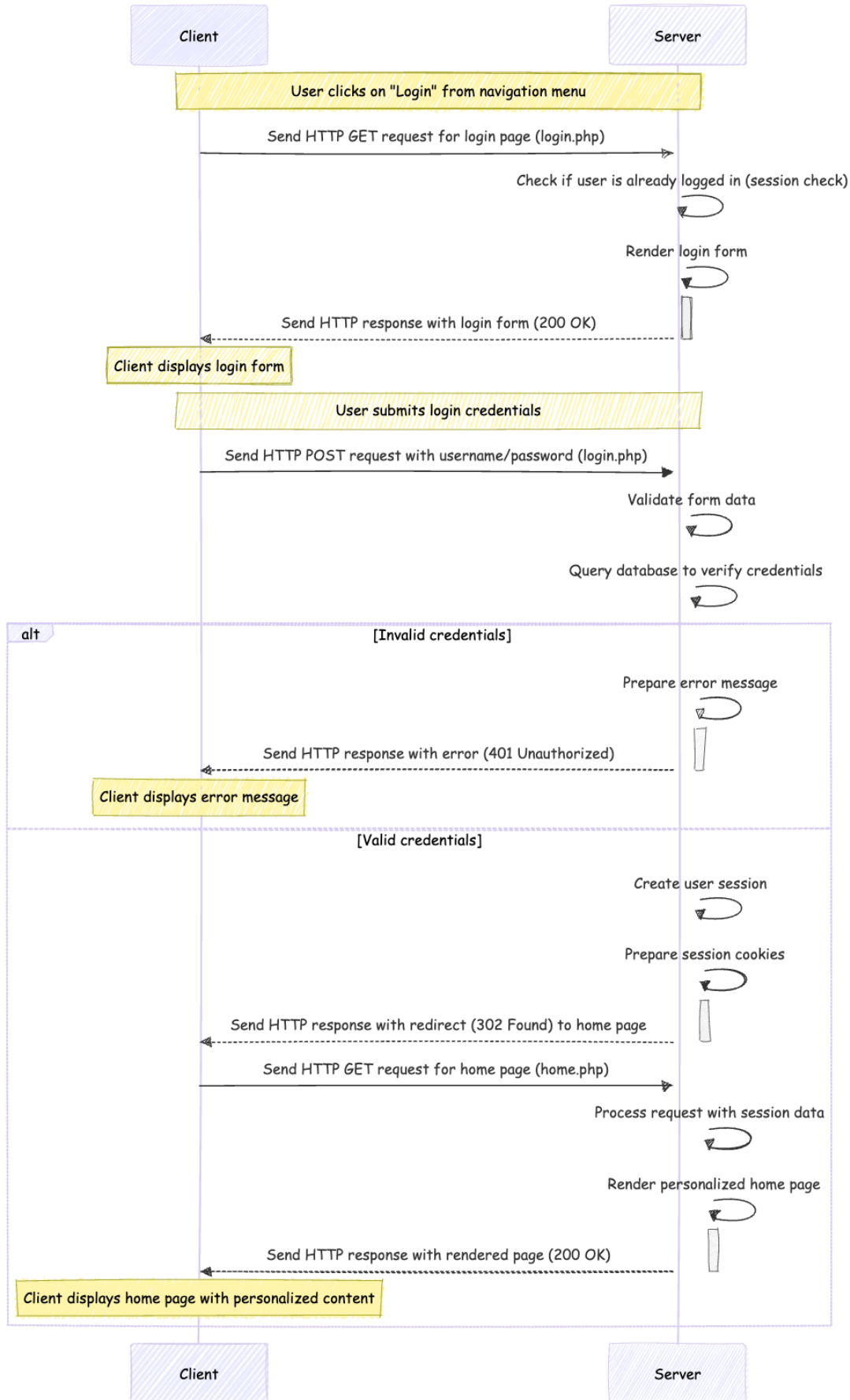
# Sequence Diagrams of Flows for Key Interactions

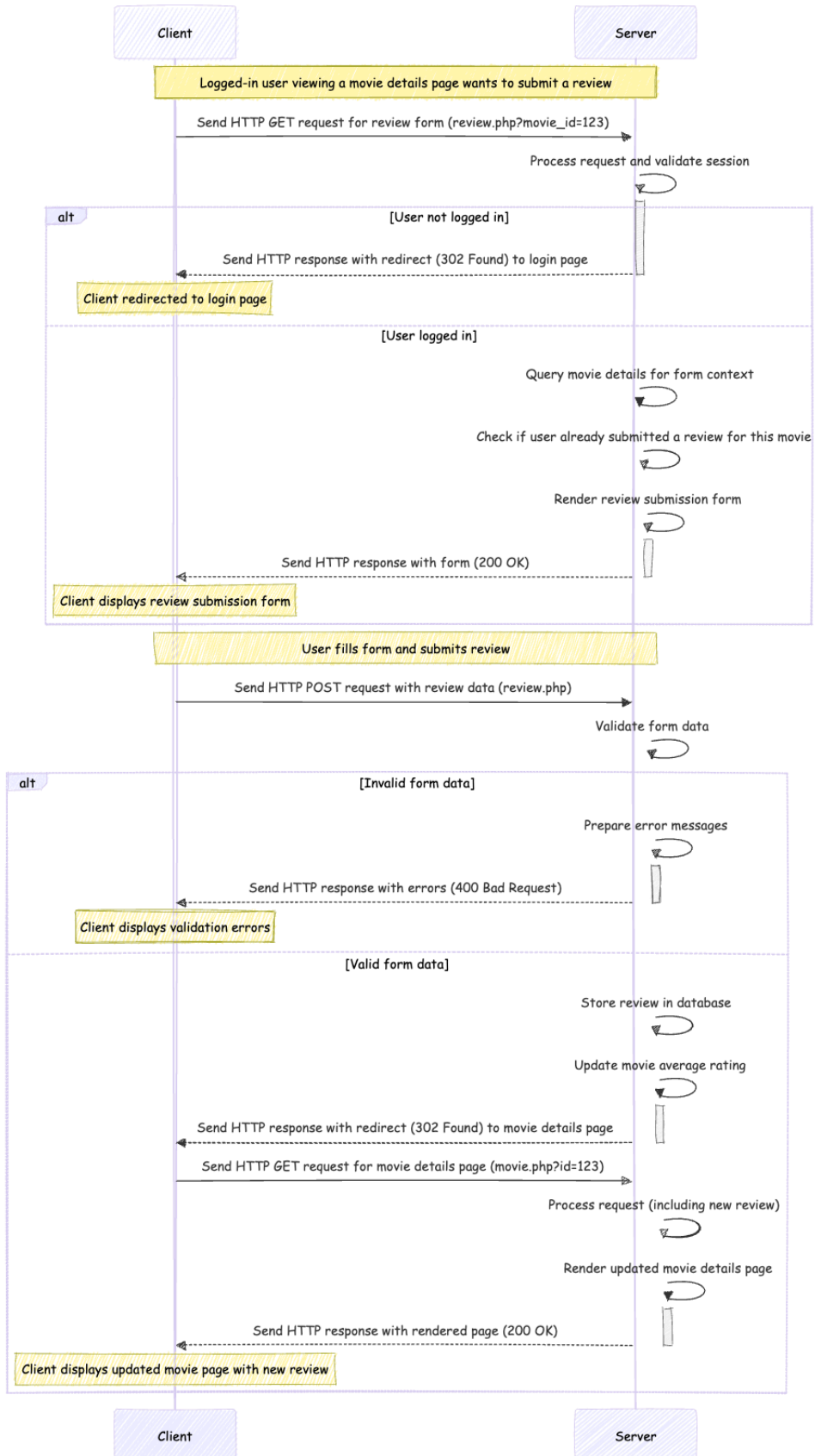
The page flows described previously are represented as sequence diagrams below:











## 3.2 Navigation Systems

The navigation system diagram illustrates how users will move through the different pages of our "PickFlix" movie recommendation website.

At the top, we have the "PERSISTENT NAVIGATION BAR" which includes five main buttons (Home, Movies, TV Shows, My Profile, and Login/Logout). This navigation bar appears on every page of the website, allowing users to quickly jump to major sections from anywhere in the application.

The diagram shows several key pages represented as nodes:

- Home Page
- Movies Listing Page
- TV Shows Listing Page
- User Profile Page
- Login Page
- Movie Details Page
- TV Show Details Page
- Streaming Service Overview

The arrows between these pages represent navigation paths, with labels explaining which UI elements trigger the navigation. For example:

From the Home Page, users can:

- Click on movie thumbnails to reach specific Movie Details Pages
- Click on TV show thumbnails to reach TV Show Details Pages
- Click the "Learn About Our Service" button to go to the Streaming Service Overview

From Movie Details Pages, users can:

- Click on "Similar Movies" links to navigate to other Movie Details Pages
- Use the "Write a Review" button to go to their Profile Page
- Click "Add to Watchlist" to update their Profile Page watchlist

From TV Show Details Pages, similar navigation options exist, allowing users to explore related shows or add them to their profile.

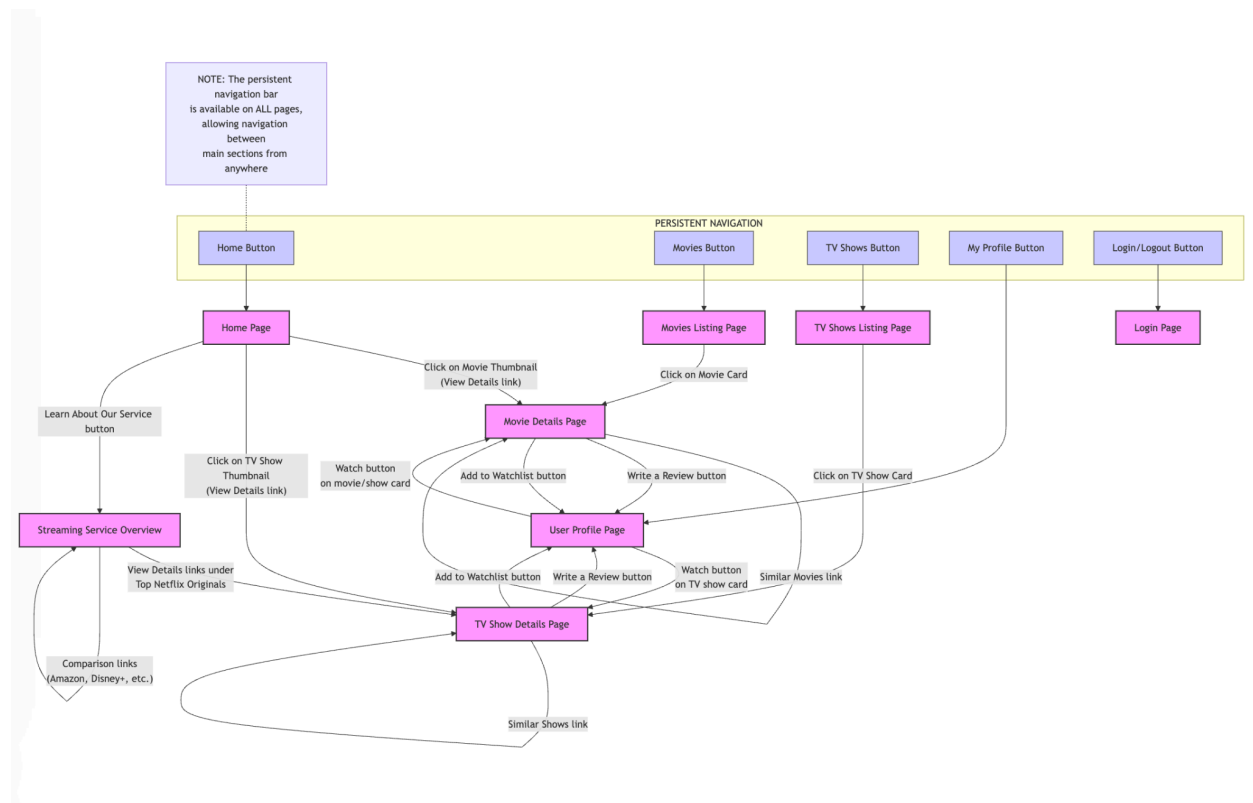
The Streaming Service Overview page provides links to specific TV shows through the "Top Netflix Originals" section, and also contains comparison links to other streaming services.

From the User Profile Page, users can click "Watch" buttons on their saved content to navigate directly to the corresponding Movie or TV Show Details Pages.

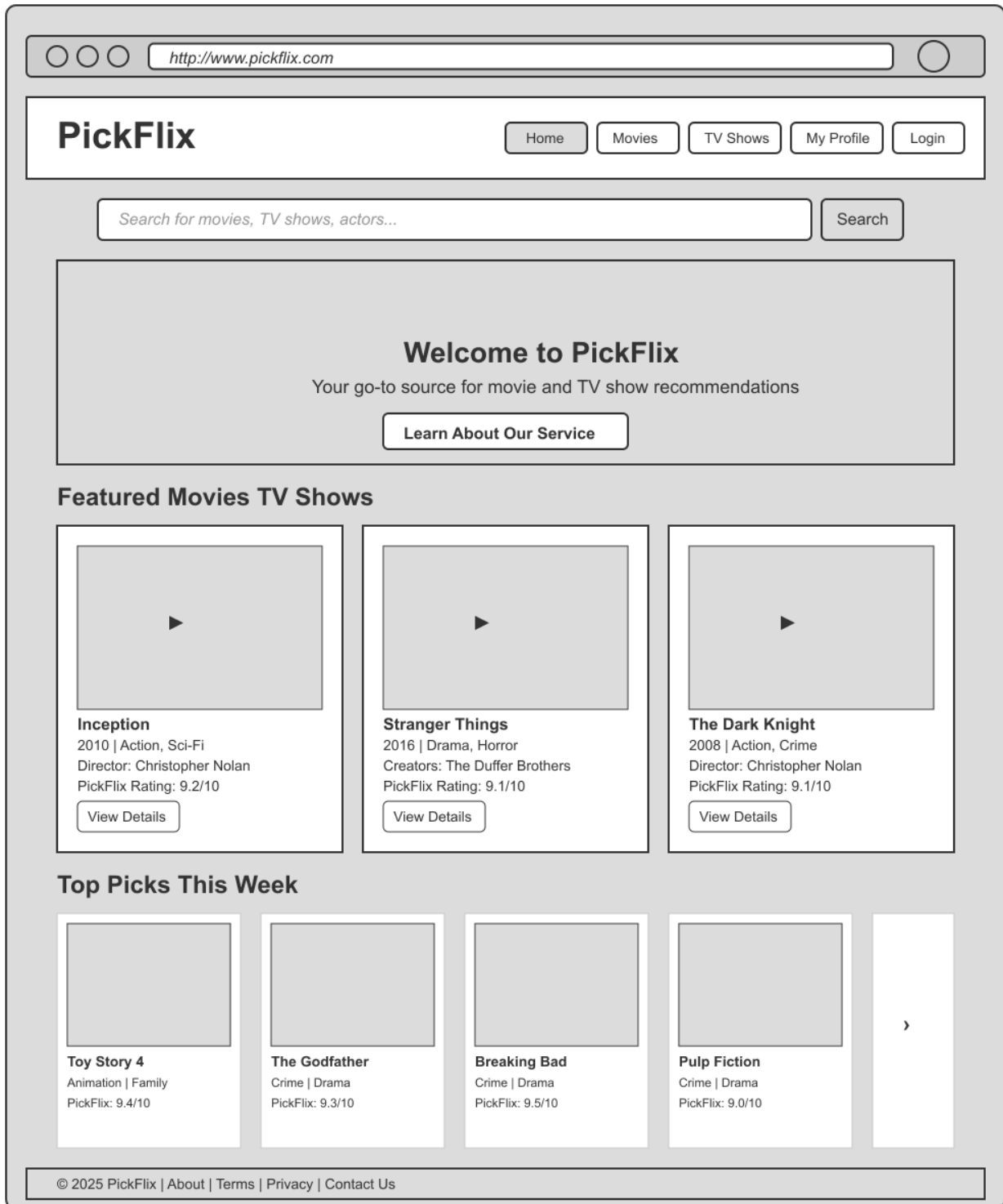
The Movies and TV Shows listing pages allow users to browse collections and click on specific items to view their details.

Throughout the diagram, we see a comprehensive web of connections showing how all parts of the application are linked through intuitive UI controls. The note at the bottom reminds us that the persistent navigation bar provides consistent access to main sections from any page in the application.

The color coding (pink for pages and light blue for navigation buttons) helps visually distinguish the different types of elements in the diagram.



### 3.3 Example Application Page Designs



## Welcome to PickFlix

Your go-to source for movie and TV show recommendations

[Learn About Our Service](#)

### Featured Movies TV Shows



#### Inception

2010 | Action, Sci-Fi

Director: Christopher Nolan

PickFlix Rating: 9.2/10

[View Details](#)

#### Stranger Things

2016 | Drama, Horror

Creators: The Duffer Brothers

PickFlix Rating: 9.1/10

[View Details](#)

#### The Dark Knight

2008 | Action, Crime

Director: Christopher Nolan

PickFlix Rating: 9.1/10

[View Details](#)

### Top Picks This Week



#### Toy Story 4

Animation | Family

PickFlix: 9.4/10



#### The Godfather

Crime | Drama

PickFlix: 9.3/10



#### Breaking Bad

Crime | Drama

PickFlix: 9.5/10



#### Pulp Fiction

Crime | Drama

PickFlix: 9.0/10





http://www.pickflix.com/streaming-services/netflix



# PickFlix

[Home](#)[Movies](#)[TV Shows](#)[My Profile](#)[Login](#)

## Netflix Streaming Service Overview

**NETFLIX**

### Stream thousands of movies and TV shows

Netflix is a subscription-based streaming service that allows members to watch TV shows and movies on an internet-connected device.

Website: [www.netflix.com](http://www.netflix.com)

Founded: 1997

## Subscription Plans

### Basic

Price: \$9.99/month

Video quality: Good

Resolution: 720p

Devices: 1 at a time

Ad-supported

### Standard

Price: \$15.49/month

Video quality: Better

Resolution: 1080p

Devices: 2 at a time

Ad-free

### Premium

Price: \$19.99/month

Video quality: Best

Resolution: 4K+HDR

Devices: 4 at a time

Ad-free

## Available On

[Smart TVs](#)[Game Consoles](#)[iOS](#)[Android](#)[Web](#)[Streaming Devices](#)

## Top Netflix Originals

[Stranger Things](#)[View Details](#)[The Crown](#)[View Details](#)[Squid Game](#)[View Details](#)[Money Heist](#)[View Details](#)[›](#)

Compare with other streaming services:

[Amazon Prime Video](#)[Disney+](#)[HBO Max](#)[Hulu](#)

© 2025 PickFlix | [About](#) | [Terms](#) | [Privacy](#) | [Contact Us](#)







Here's a description of the four wireframes that have been created for the Movie/TV Show recommendation website:

### **Home Page Wireframe**

This page serves as the main entry point to your PickFlix website. It features a clean header with navigation options for Home, Movies, TV Shows, My Profile, and Login. The search bar allows users to find content easily. The main content is organized into three featured movies/TV shows (Inception, Stranger Things, and The Dark Knight), each with preview images, key information, ratings, and "View Details" links. Below that is a scrollable "Top Picks This Week" section showing additional recommendations. The design uses a simple hand-drawn aesthetic that clearly presents content options while providing easy navigation to individual movie/TV show pages and the streaming service overview.

### **Streaming Service Overview Wireframe**

This page provides comprehensive information about Netflix as a streaming platform. The header maintains consistent navigation with other pages. The main content includes the Netflix logo, a brief description of the service, and key information like the website and founding date. The page is organized into clear sections: Subscription Plans (Basic, Standard, Premium) with details on pricing and features; Available Platforms showing where Netflix can be accessed; and Top Netflix Originals featuring content unique to the platform with links to individual show pages. There's also a comparison section at the bottom allowing users to explore other streaming services.

### **TV Show Details Wireframe**

This page focuses on a specific TV show (Stranger Things), providing comprehensive information for users. It maintains the same navigation as other pages for consistency. The main section includes a large preview area with a play button for trailers. Below are key details about the show: title, creators, cast, and a synopsis. A unique feature is the Seasons section, which lists each season with episode counts, release years, and individual ratings. Action buttons allow users to watch, view trailers, or add to their watchlist. The right sidebar shows similar shows and user reviews, helping visitors discover related content and see what others think of the show.

### **User Profile Wireframe**

This page gives users a personalized space to manage their content preferences. The consistent navigation has "My Profile" highlighted to show the current location. The profile section displays the user's information (AbuJassim92) with key stats about their activity. Tab navigation allows switching between Watchlist, Reviews, Preferences, and Activity sections. The main content shows the Watchlist with filtering options and thumbnails of saved content (Inception, Stranger Things, The Dark Knight, The Crown, Interstellar), each with key information and action buttons to watch or remove items. This design makes it easy for users to manage their personal recommendations and track content they want to watch.