При разработке сайта по поиску фильмов со стороны backend-разработчика должен быть реализованы основные аспекты:

**Масштабируемость**: Использование микросервисной архитектуры позволит разделить функциональность на независимые компоненты, что упростит масштабирование и поддержку системы. Это также включает использование облачных решений для автоматического масштабирования ресурсов в зависимости от нагрузки. Использование облачных решений (например, AWS S3) для хранения медиафайлов, что обеспечит надежность и доступность контента.

**Базы данных**: необходимо спроектировать структуру базы данных для хранения информации о фильмах, пользователях, отзывах и т.д. Это может включать таблицы для фильмов, пользователей и их предпочтений. Выбор подходящей базы данных зависимости от структуры данных и объема информации. Например, реляционные базы данных (PostgreSQL, MySQL) подходят для структурированных данных, тогда как NoSQL (MongoDB) могут быть более эффективными для хранения неструктурированных данных.

**Авторизация:** для реализации безопасных методов авторизации пользователей можно использовать OAuth 2.0 или JWT (JSON Web Tokens). Это позволит обеспечить защиту пользовательских данных и предотвратить несанкционированный доступ. Также можно организовать возможность авторизации через сторонние сервисы (например, Google или Яндекс) для упрощения процесса входа и повышения удобства для пользователей.

**Загрузка контента и его хранение:** для загрузки медиафайлов можно использовать API, который позволит пользователям загружать видео и изображения с валидацией форматов и размеров файлов. Это важно для обеспечения качества загружаемого контента.

Помимо этого, можно использовать сторонние API (например, TMDb или OMDb) для получения актуальной информации о фильмах, постерах и трейлерах.

**Модерация контента**: для реализации модерации можно использовать автоматические фильтры для проверки загружаемого контента на соответствие правилам платформы, а также ручную проверку контента через административную панель для более тщательной оценки сложных случаев.

**Оптимизация производительности:** для этого нужно реализовать подгрузку данных по мере необходимости для улучшения производительности, оптимизировать код на стороне сервера для повышения скорости обработки запросов, а также кэширование для уменьшения нагрузки на сервер и ускорения времени ответа.