**Дипломна робота: What to Watch?**

Автор: [Ваше ім’я]  
Керівник: [ПІБ керівника]  
Заклад: [Назва ЗВО]  
Дата: [ММ.РРРР]

**Анотація**

Короткий виклад мети, завдань, технологій та результатів проєкту What to Watch? (Angular + Node.js + Express + MySQL + Docker).

Ключові слова: Angular, Node.js, REST API, MySQL, Docker, аутентифікація, коментарі, фільми.

**Зміст**

* Вступ
  1. Опис проєкту й контекст
  2. Аналіз предметної області та вимог
  3. Огляд технологій і вибір стеку
  4. Проєктування системи
  5. Реалізація Front-end
  6. Реалізація Back-end
  7. Інфраструктура та розгортання
  8. Безпека
  9. Тестування і результати
  10. Оглядова аналітика (Veille)
* Висновки
* Список джерел
* Додатки

**Вступ**

* Мотивація теми: чому каталог фільмів і рекомендацій.
* Цілі: навчальні та технічні.
* Результат: повноцінний CRUD застосунок з авторизацією та коментарями.

**1. Опис проєкту й контекст**

* Постановка задачі.
* Цільова аудиторія (кіномани, невеликі клуби).
* Межі проєкту: базова функціональність без рекомендаційного алгоритму.

**2. Аналіз предметної області та вимог**

**2.1. Бізнес-вимоги**

* Перегляд списку фільмів, фільтр за жанром/роком.
* Перегляд деталей, додавання коментарів.
* Авторизація: реєстрація/вхід.
* Додавання/редагування фільмів.

**2.2. Нефункціональні вимоги**

* Продуктивність, безпека, простота розгортання (Docker), масштабованість.

**2.3. Юзер-сторі (приклади)**

* Як користувач, хочу переглядати список і фільтрувати.
* Як авторизований, хочу додавати фільми та коментувати.

**3. Огляд технологій і вибір стеку**

* Angular 16+/19 — SPA, двосторонній біндінг, роутинг.
* Node.js + Express — REST API.
* MySQL — реляційна БД.
* Docker + phpMyAdmin — інфраструктура та зручне керування.
* Обґрунтування вибору кожної технології.

**4. Проєктування системи**

**4.1. Архітектура**

* Клієнт-сервер з REST API, розділення відповідальностей.

**4.2. ER-діаграма БД**

Вставте діаграму або використайте текстову нотацію з додатка “ER Diagram”.

**4.3. Модель навігації (Front-end)**

* Маршрути: /, /film/:id, /add, /edit/:id, /login, /register.

**4.4. Діаграма послідовності (приклад)**

* Логін → Отримання токена-заглушки.
* Додавання фільму → POST → запис у БД → відповідь → редірект.

**5. Реалізація Front-end**

**5.1. Структура**

* Компоненти: FilmList, FilmDetail, FilmForm, Login, Register, Navbar.
* Сервіси: FilmService, AuthService.

**5.2. Ключові фрагменти**

* Двосторонній біндінг [(ngModel)].
* HTTP-взаємодія з API (GET/POST/PUT).

Вставте скріншоти інтерфейсу:

* Список фільмів (img)
* Деталі фільму (img)
* Форма фільму (img)
* Логін/реєстрація (img)

**6. Реалізація Back-end**

**6.1. Конфігурація БД**

* Пул з’єднань mysql2/promise.
* .env змінні: хост, порт, користувач, пароль, БД.

**6.2. Маршрути**

* /api/health, /api/genres.
* /api/films (CRUD + коментарі).
* /api/auth (login, register, check).

**6.3. Приклади відповідей**

Вставте JSON-відповіді з curl/Postman.

**7. Інфраструктура та розгортання**

* infra/docker-compose.yml — MySQL + phpMyAdmin.
* Імпорт infra/init.sql через phpMyAdmin.
* Запуск: npm run dev для бекенду, ng serve для фронтенду.

**8. Безпека**

* Поточний стан: CORS, авторизація-спроста.
* Ризики: plain-паролі, відсутність JWT, ролей.
* Покращення: bcrypt, JWT + refresh, RBAC, CSRF/XSS захист.

**9. Тестування і результати**

* План тестів: ручні сценарії + curl.
* Результати: знімки екрана, логи.
* Приклади: фільтрація, створення/оновлення фільму, коментарі.

**10. Оглядова аналітика (Veille)**

* Джерела: Angular/Express/MySQL docs, OWASP.
* Знайдені практики: валідація, безпека, деплой.

**Висновки**

* Досягнення, складнощі, уроки.
* Подальші кроки: повна авторизація, завантаження зображень, тести, CI/CD.

**Список джерел**

[1] Angular Docs  
[2] Express Docs  
[3] MySQL Docs  
[4] OWASP Top 10  
[5] Docker Docs

**Додатки**

* ER-діаграма
* SQL-скрипт infra/init.sql
* Ключові фрагменти коду (маршрути, сервіси)
* Postman колекція / curl-команди