ЗМІСТ

[1 Вимоги до програмного продукту 6](#_Toc58443159)

[1.1 Визначення потреб споживача 6](#_Toc58443160)

[1.1.1 Ієрархія потреб споживача 6](#_Toc58443161)

[1.1.2 Деталізація матеріальної потреби 7](#_Toc58443162)

[1.2 Бізнес-вимоги до програмного продукту 7](#_Toc58443163)

[1.2.1 Опис проблеми споживача 7](#_Toc58443164)

[1.2.1.1 Концептуальний опис проблеми споживача 7](#_Toc58443165)

[1.2.1.2 Метричний опис проблеми споживача 8](#_Toc58443166)

[1.2.2 Мета створення програмного продукту 8](#_Toc58443167)

[1.2.2.1 Проблемний аналіз існуючих програмних продуктів 8](#_Toc58443168)

[1.2.2.2 Мета створення програмного продукту 8](#_Toc58443169)

[1.2.3 Назва програмного продукту 9](#_Toc58443170)

[1.2.3.1 Гасло програмного продукту 9](#_Toc58443171)

[1.2.3.2 Логотип програмного продукту 10](#_Toc58443172)

[1.3 Вимоги користувача до програмного продукту 10](#_Toc58443173)

[1.3.1 Історія користувача програмного продукту 10](#_Toc58443174)

[1.3.2 Діаграма прецедентів програмного продукту 11](#_Toc58443175)

[1.3.3 Сценаріїв використання прецедентів програмного продукту 11](#_Toc58443176)

[1.4 Функціональні вимоги до програмного продукту 16](#_Toc58443177)

[1.4.1. Багаторівнева класифікація функціональних вимог 16](#_Toc58443178)

[1.4.2 Функціональний аналіз існуючих програмних продуктів 19](#_Toc58443179)

[1.5 Нефункціональні вимоги до програмного продукту 19](#_Toc58443180)

[1.5.1 Опис зовнішніх інтерфейсів 19](#_Toc58443181)

[1.5.1.1 Опис інтерфейса користувача 19](#_Toc58443182)

[1.5.1.1.1 Опис INPUT-інтерфейса користувача 19](#_Toc58443183)

[1.5.1.1.2 Опис OUTPUT-інтерфейса користувача 21](#_Toc58443184)

[1.5.1.2 Опис інтерфейсу із зовнішніми пристроями 23](#_Toc58443185)

[1.5.1.3 Опис програмних інтерфейсів 24](#_Toc58443186)

[1.5.1.4 Опис інтерфейсів передачі інформації 24](#_Toc58443187)

[1.5.1.5 Опис атрибутів продуктивності 24](#_Toc58443188)

[2 Планування процесу розробки програмного продукту 26](#_Toc58443189)

[2.1 Планування ітерацій розробки програмного продукту 26](#_Toc58443190)

[2.2 Концептуальний опис архітектури програмного продукту 27](#_Toc58443191)

[2.3 План розробки програмного продукту 27](#_Toc58443192)

[2.3.1 Оцінка трудомісткості розробки програмного продукту 27](#_Toc58443193)

[2.3.2 Визначення дерева робіт з розробки програмного продукту 30](#_Toc58443194)

[2.3.3 Графік робіт з розробки програмного продукту 33](#_Toc58443195)

[3 Проектування програмного продукту 34](#_Toc58443196)

[3.1 Концептуальне та логічне проектування структур даних програмного продукту. 34](#_Toc58443197)

[3.1.1 Концептуальне проектування на основі UML-діаграми концептуальних класів. 34](#_Toc58443198)

[3.1.2 Логічне проектування структур даних. 34](#_Toc58443199)

[3.2 Проектування програмних класів 35](#_Toc58443200)

[3.3 Проектування алгоритмів роботи методів програмних класів 35](#_Toc58443201)

[3.4 Проектування тестових наборів методів програмних класів 40](#_Toc58443202)

[4 Конструювання програмного продукту 42](#_Toc58443203)

[4.1 Особливості конструювання структур даних 42](#_Toc58443204)

[4.1.1 Особливості інсталяції та роботи з СУБД 42](#_Toc58443205)

[4.1.2 Особливості створення структур даних 43](#_Toc58443206)

[4.2 Особливості конструювання програмних модулів 44](#_Toc58443207)

[4.2.1 Особливості роботи з інтегрованим середовищем розробки 44](#_Toc58443208)

[4.2.2 Особливості створення програмної структури з урахуванням спеціалізованого Фреймворку 44](#_Toc58443209)

[4.2.3 Особливості створення програмних модулів 47](#_Toc58443210)

[4.2.4 Особливості розробки алгоритмів методів програмних класів або процедур/функцій 48](#_Toc58443211)

[5 Розгортання та валідація програмного продукту 62](#_Toc58443212)

[5.1 Інструкція з встановлення програмного продукту 62](#_Toc58443213)

[5.2 Інструкція з використання програмного продукту 63](#_Toc58443214)

[5.3 Результати валідації програмного продукту 66](#_Toc58443215)

[Висновок 68](#_Toc58443216)

# 1 Вимоги до програмного продукту

1.1 Визначення потреб споживача

### 1.1.1 Ієрархія потреб споживача

Відомо, що в теорії маркетингу потреби людини можуть бути представлені у вигляді ієрархії потреб ідей американського психолога Абрахама Маслоу включають рівні:

* фізіологія (вода, їжа, житло, сон);
* безпека (особиста, здоров&#39;я, стабільність),
* приналежність (спілкування, дружба, любов),
* визнання (повага оточуючих, самооцінка),
* самовираження (вдосконалення, персональний розвиток).

На рисунку 1.1 представлено одну ієрархію потреби споживача, яку

хотілося б задовольнити, використовуючи майбутній програмний продукт.



Рис. 1.1 – Ієрархія потреб споживача нашого ПП.

### 1.1.2 Деталізація матеріальної потреби

Матеріальні потреби – це сукупність матеріальних благ, які мають матеріальну форму і виступають як головний спонукальний мотив трудової діяльності людини.

Споживач – це людина, яка споживає продукти виробництва і сфери послуг для відтворення робочої сили.

Тому була винайдена деталізація матеріальної потреби для нашого ПП.

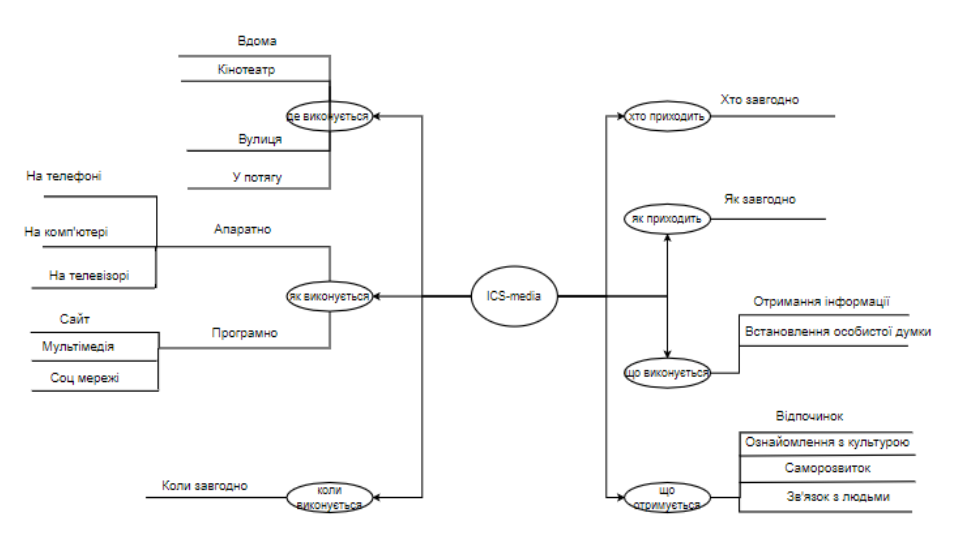


Рис. 1.2 – Деталізація матеріальних потреб.

## 1.2 Бізнес-вимоги до програмного продукту

### 1.2.1 Опис проблеми споживача

#### 1.2.1.1 Концептуальний опис проблеми споживача

Умова задоволення потреб переведення даних в інформацію: Доступність, Представленість мовою споживача, Інтерес, Актуальність.

Загальний опис проблеми: неможливо загружати фільми особистого виробництва з можливістю виплати виробнику.

#### 1.2.1.2 Метричний опис проблеми споживача

Метричні показники незадоволеності споживача: виробники не можуть загружати свої творіння на окремий сервіс та отримувати гроші з можливості користувачів платити за фільм творця.

Рівень користності для виробників СL можна визначити як

CL = NA / N,

NA – кількість просмотрів;

N – кількість донатів.

### 1.2.2 Мета створення програмного продукту

#### 1.2.2.1 Проблемний аналіз існуючих програмних продуктів

На таблиці 1.1 вказано аналіз існуючих програмних продуктів.

#### 1.2.2.2 Мета створення програмного продукту

Надати можливість загружати свої фільми людям які бажають отримувати винагороду.

Таблиця 1.1 – Аналіз продуктів.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Назва пробукту | Вартість | Ступінь готовності | Примітка |
| 1 | ivi.ru | Безкоштовно/  платно | 4 | Немає знижок для студентів. Більшість фільмів платні. |
| 2 | film.ru | Безкоштовно | 2 | Працює не стабільно. Використовує сторонні ресурси. |
| 3 | gidonline.io | Безкоштовно | 3 | Багато реклами. Часто фільми у поганому якості. |
| 4 | оkko.tv | Безкоштовно/  платно | 4 | Не доступно в Україні. |
| 5 | baskino.me | Безкоштовно | 2 | Багато реклами. |

### 1.2.3 Назва програмного продукту

#### 1.2.3.1 Гасло програмного продукту

«Загружайте, дивіться!» - ICS - media.

#### 1.2.3.2 Логотип програмного продукту



Рис. 1.3 – Логотип ICS – media.

## 1.3 Вимоги користувача до програмного продукту

### 1.3.1 Історія користувача програмного продукту

* Як гість, я можу зареєструватися.
* Як гість, я можу включити будь-який фільм з каталогу.
* Як гість, я можу сортувати каталог.
* Як гість, я можу шукати за назвою.
* Як користувач, я можу коментувати.
* Як користувач, я можу віддячити творця.
* Як користувач, я можу завантажити фільм.
* Як адміністратор, я можу видаляти фільми.

### 1.3.2 Діаграма прецедентів програмного продукту

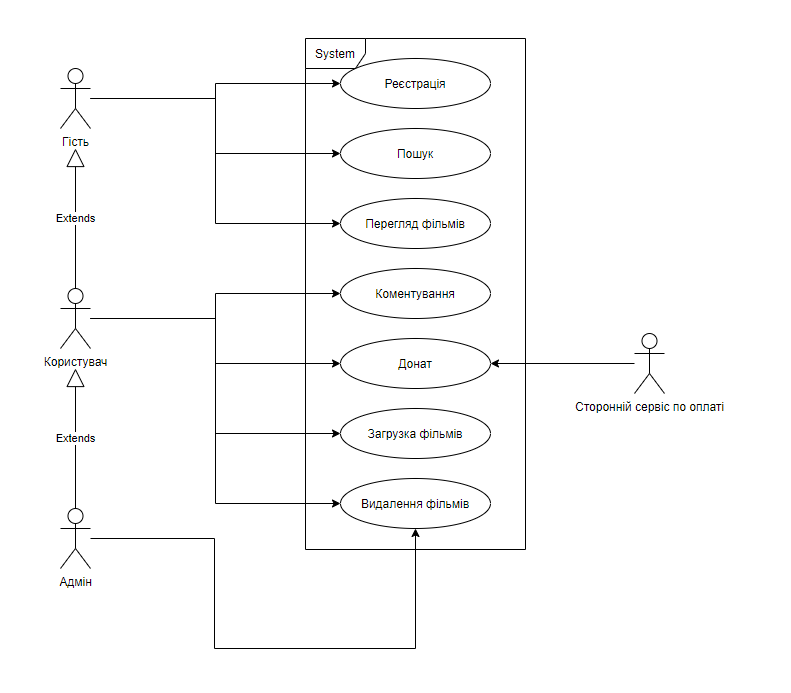


Рис. 1.3 – Діаграма прецедентів.

### 1.3.3 Сценаріїв використання прецедентів програмного продукту

* Реєстрація користувача.

Передумова: Користувач має знаходитись на сайті.

Актори які можуть виконувати: Гість.

Актор-основний ініціатор: Гість.

Успішний результат: Гість успішно зареєструвався.

Успішний сценарій:

1. ПП дає можливість реєстрації;

2. Гість переходить до сторінки реєстрації;

3. ПП дає можливість ввести такі дані:

- Логін;

- Пароль.

4. Гість вводить свої дані до полей Логін, Пароль;

5. ПП видає повідомлення «Успішне створення користувача»;

Альтернативний сценарій:

4.1.1 Гісь залишає невірні поля (пусті);

4.1.1 Повідомлення, про те що поля пусті.

4.2.1. Гість вводить логін який вже існує в базі;

4.2.2. Повідомлення про те що користувач вже існує за даним логіном.

* Перегляд фільму.

Передумова: Користувач повинен перейти на сторінку фільму.

Актори які можуть виконувати: Гість, Адміністратор, Користувач.

Актор-основний ініціатор: Користувач, Гість, Адміністратор.

Успішний результат: Фільм який вибраний Користувачем успішно запустився, та працює без багів.

Успішний сценарій:

1. ПП видає список фільмів на головній сторінці;

2. Користувач переходить на сторінку будь-якого фільму;

3. Користувач натискає на кнопку старта відео;

4. Запускається фільм.

Альтернативний сценарій:

4.1. Повідомлення, що до фільму немає доступу в даний момент.

* Додавання фільмів.

Передумова: Користувач має бути на головній сторінці після «Реєстрації».

Актори які можуть виконувати: Адміністратор, Користувач.

Актор-основний ініціатор: Користувач, Адміністратор.

Успішний результат: Фільм успішно завантажено до сайту, його можна знайти у пошуку.

Успішний сценарій:

1. ПП дає можливість перейти до особистого кабінету;

2. Користувач переходить до особистого кабінету;

3. ПП дає можливість перейти до сторінки загрузки фільму;

4. Користувач переходить до сторінки загрузки;

5. ПП дає можливість загрузити свій фільм з такими полями:

- Назва;

- Жанр;

- Країна;

- Посилання на постер;

- Рік;

- Опис;

- URL;

6. Користувач вводить дані, та клікає «Завантажити»;

7. ПП видає повідомлення що фільм загружено.

Альтернативний сценарій:

4.1 Користувач залишає поля пустими;

4.2 ПП видає повідомлення що поля не можуть бути пустими.

* Пожертвування творцям фільмів.

Передумова: Користувач має бути на головній сторінці після «Реєстрації».

Актори які можуть виконувати: Адміністратор, Користувач.

Актор-основний ініціатор: Сторонній сервіс по переводу коштів.

Успішний результат: Успішно відправлено гроші до творця.

1. ПП видає список фільмів на головній сторінці;

2. Користувач переходить до сторінки будь-якого фільму;

3. ПП дає можливість пожертвування;

4. Користувач клікає на кнопку «Пожертвувати»;

5. Відкривається сторонній сервіс для переводу коштів, та користувач проводить дії на зовнішньому сервісі;

6. Сервіс перенаправляє до ПП;

7. ПП видає повідомлення про успішний перевід коштів.

Альтернативний сценарій:

3.1 ПП не видає можливості пожертвування, бо користувач який загрузив фільм не ввів свої банківські дані.

7.1 ПП видає повідомлення про не успішний перевід коштів.

* Коментування фільмів.

Передумова: Користувач має бути на сторінці фільму як у «Перегляд фільму».

Актори які можуть виконувати: Адміністратор, Користувач.

Актор-основний ініціатор: Користувач.

Успішний результат: Успішно прикріплено коментар до фільму.

1. ПП дає можливість написати коментар;

2. Користувач пише свій коментар та нажимає кнопку відправити;

3. ПП видає повідомлення що успішно прикріплено коментар до фільму.

Альтернативний сценарій:

2.1 Користувач залишає поле коментаря пустим та нажимає кнопку відправити;

2.2. Повідомлення, що коментар не має бути пустим.

* Видалення фільмів.

Передумова: Користувач має бути на головній сторінці після реєстрації «Реєстрації» та має бути хоча б один фільм загружено.

Актори які можуть виконувати: Адміністратор, Користувач.

Актор-основний ініціатор: Користувач, Адміністратор.

Успішний результат: Успішно видалено фільм.

1. ПП дає можливість перейти до особистого кабінету;

2. Користувач переходить до особистого кабінету;

3. ПП дає можливість переглянути свої завантажені фільмів;

4. Користувач переходить до сторінки свого фільму;

5. ПП дає можливість видалити фільм;

6. Користувач клікає на кнопку видалення фільму;

7. Повідомлення що фільм видалено.

Альтернативний сценарій:

1.1 Адміністратор переходить до головної сторінки сайту.

1.2 Адміністратор переходить до сторінки будь-якого фільму з головної сторінки

1.3 ПП дає можливість видалити фільм;

1.4 Адміністратор клікає на кнопку видалення фільму;

1.5 Повідомлення що фільм видалено.

* Використання пошуку

Передумова: Користувач має знаходитись на головній сторінці ПП.

Актори які можуть виконувати: Гість, Адміністратор, Користувач.

Актор-основний ініціатор: Користувач, Гість, Адміністратор.

Успішний результат: Пошук видає коректний результат введений Користувачем.

Успішний сценарій:

1. ПП видає строку пошуку.

2. Користувач вводить запит у строку пошуку;

3. Пошук видає потрібний запит який ввів користувач.

Альтернативний сценарій:

2.1. Користувач вводить невірний (пустий) запит, або запит даних яких не існує;

2.2. Повідомлення, про те що немає даних за введеним запитом.

## 1.4 Функціональні вимоги до програмного продукту

### 1.4.1. Багаторівнева класифікація функціональних вимог

Таблиця 1.2 – Класифікація функцій.

|  |  |
| --- | --- |
| Ідентифікатор  функції | Назва функції |
| FR1 | Реєстрація користувача. |
| FR1.1 | ПП дає можливість ввести данні . |
| FR1.2 | Користувач передає свої данні. |
| FR1.3 | ПП перевіряє введені дані. |
| FR1.4 | ПП надає доступ користувача до сайту. |
| FR2 | Здійснити пошук фільму. |
| FR2.1 | ПП надає можливість ввести дані за якими буде знаходитися фільм. |
| FR2.2 | Користувач вводить данні фільму. |
| FR2.3 | ПП видає користувачеві бажаний фільм. |
| FR3 | Пожертвування на фільм. |
| FR3.1 | ПП надає можливість пожертвувати творцям фільмів. |
| FR3.2 | ПП перенаправляє користувача на сторонній сервіс по оплаті. |
| FR3.3 | Користувач здійснює оплату через сторонній сервіс. |
| FR3.4 | ПП висилає користувачу мели з підтвердження пожертвування. |
| FR4 | Коментування фільмів. |
| FR4.1 | ПП надає можливість залишити коментар. |
| FR4.2 | Користувач залишає коментар. |
| FR4.3 | ПП залишає коментар на сторінці фільму. |
| FR5 | Перегляд фільмів. |
| FR5.1 | ПП надає можливість дивитися фільм. |
| FR5.2 | Користувач переходить до сторінки фільму. |
| FR5.3 | ПП коректо запускає фільм. |
| FR6 | Завантаження фільмів. |
| FR6.1 | ПП надає можливість завантажити фільм. |
| FR6.2 | Користувач завантажує фільм. |
| FR6.3 | ПП видає фільм в каталозі. |
| FR7 | Видалення фільмів. |
| FR7.1 | ПП надає можливість видалити фільм. |
| FR7.2 | Користувач видаляє фільм. |
| FR7.3 | ПП більше не має фільм в каталозі. |

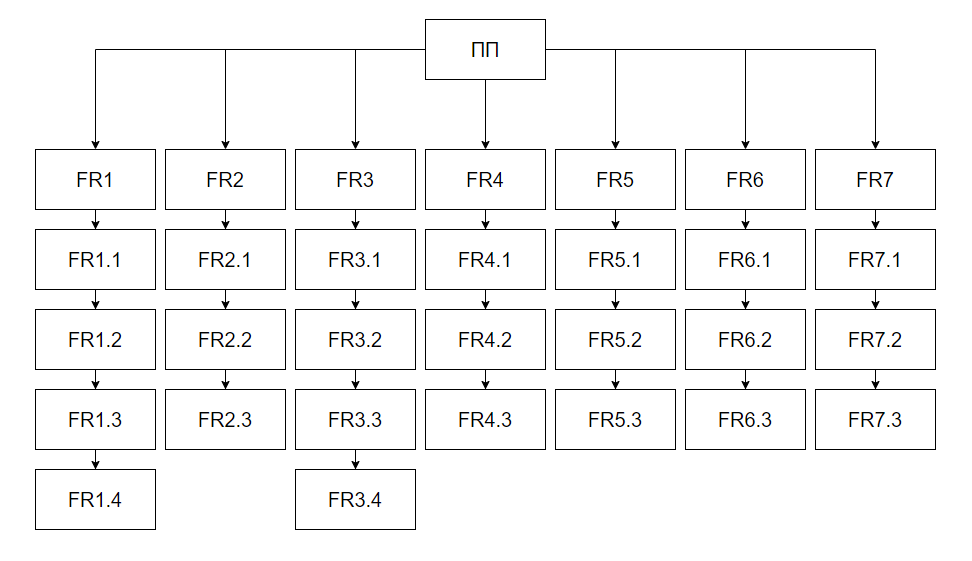


Рис 1.4 – Дерево ієрархій функцій.

### 1.4.2 Функціональний аналіз існуючих програмних продуктів

Таблиця 1.3 – Таблиця аналізу існуючих програмних продуктів.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ідентифікатор функції | Гид онлайн | Ivi | Baskino | Okko | Megogo |
| FR1 | + | + | + | + | + |
| FR2 | + | + | + | + | + |
| FR3 | - | - | - | - | - |
| FR4 | + | + | + | + | + |
| FR5 | + | + | + | + | + |
| FR6 | - | - | - | - | - |
| FR7 | - | - | - | - | - |

## 1.5 Нефункціональні вимоги до програмного продукту

### 1.5.1 Опис зовнішніх інтерфейсів

#### 1.5.1.1 Опис інтерфейса користувача

##### 1.5.1.1.1 Опис INPUT-інтерфейса користувача

Таблиця 1.4 – Опис INPUT.

|  |  |  |
| --- | --- | --- |
| Ідентифікатор функції | Засіб INPUT-потоку | Особливості використання |
| FR1.1 | Комп’ютерна  клавіатура;  маніпулятор типу «миша»;  сенсорний екран; | Використання кнопок миші для навігації по сторінці та клікання по кнопках та віртуальної чи фізичної клавіатури для введення даних про користувача. |
| FR1.2 |
| FR1.3 |
| FR1.4 |
| FR2.1 | Використання кнопок миші для навігації по сторінці та клікання по кнопках та віртуальної чи фізичної клавіатури для введення даних фільм (назва). |
| FR2.2 |
| FR2.3 |
| FR3.1 | Використання кнопок миші для навігації по сторінці та клікання по кнопках та віртуальної чи фізичної клавіатури для введення даних для донату. |
| FR3.2 |
| FR3.3 |
| FR3.4 |
| FR4.1 | Використання кнопок миші для навігації по сторінці та клікання по кнопках та віртуальної чи фізичної клавіатури для введення тексту коментаря |
| FR4.2 |
| FR4.3 |
| FR5.1 | Використання кнопок миші для запуску фільма. |
| FR5.2 |
| FR5.3 |
| FR6.1 | Використання кнопок миші для навігації по сторінці та клікання по кнопках та віртуальної чи фізичної клавіатури для введення даних про фільм. |
| FR6.2 |
| FR6.3 |
| FR7.1 | Використання кнопок миші для навігації по сторінці для видалення фільму. |
| FR7.2 |
| FR7.3 |

##### 1.5.1.1.2 Опис OUTPUT-інтерфейса користувача

OUTPUT – потік у нашому ПП це вікно браузера.

Таблиця 1.5 – Макети.

|  |  |
| --- | --- |
| Ідентифікатор функції | OUTPUT |
| FR1 |  |
| FR2 |  |
| FR3 |  |
| FR4 |  |
| FR5 |  |
| FR6 |  |
| FR7 |  |

#### 1.5.1.2 Опис інтерфейсу із зовнішніми пристроями

Таблиця 1.6 – Опис зовнішніх інтерфейсів.

|  |  |
| --- | --- |
| Ідентифікатор функції | Зовнішній пристрій |
| FR1 | Смартфон, планшет, Desktop-персональний комп`ютер, laptop; |
| FR2 |
| FR3 |
| FR4 |
| FR5 |
| FR6 |
| FR7 |

#### 1.5.1.3 Опис програмних інтерфейсів

Таблиця 1.7 – Версії програмних інтерфейсів.

|  |  |
| --- | --- |
| Версії операційних систем та програмних бібліотек, які знадобляться при реалізації більшості функцій ПП. | Для користування:   * Windows 95 – 10, Linux. * Будь-який браузер   Для розробки:   * PHP 7 * Сервіс по оплаті – Intercassa. * Бібліотека RedBean для роботи з БД. |

#### 1.5.1.4 Опис інтерфейсів передачі інформації

Таблиця 1.8 – Версії програмних інтерфейсів.

|  |  |
| --- | --- |
| Опис інтерфейсів передачі інформації, які знадобляться при реалізації більшості функцій ПП. | * Провідні інтерфейси: Ethernet; * Безпровідні інтерфейси: Wi-Fi; |

#### 1.5.1.5 Опис атрибутів продуктивності

Таблиця 1.11 – Опис атрибутів продуктивності.

|  |  |
| --- | --- |
| Ідентифікатор функції | Максимальний час реакції ПП на дії користувачів (секунди) |
| FR1 | 15 |
| FR2 | 15 |
| FR3 | 30 |
| FR4 | 10 |
| FR5 | 50 |
| FR6 | 20 |
| FR7 | 10 |

# 2 Планування процесу розробки програмного продукту

## 2.1 Планування ітерацій розробки програмного продукту

Таблиця 2.1 – Ітерації розробки.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ідентифікатор функції | Назва функції | Функціональні  залежності | Вплив на досягнення мети, % | Пріоритет  функції |
| FR1 | Реєстрація користувача. | - | 10% | C |
| FR2 | Здійснити пошук фільму. | - | 5% | M |
| FR3 | Пожертвування на фільм. | FR1 | 25% | M |
| FR4 | Коментування фільмів. | FR1 | 5% | S |
| FR5 | Перегляд фільмів. | FR6 | 25% | M |
| FR6 | Завантаження фільмів. | FR1 | 25% | S |
| FR7 | Видалення фільмів. | FR1 | 5% | S |

## 2.2 Концептуальний опис архітектури програмного продукту

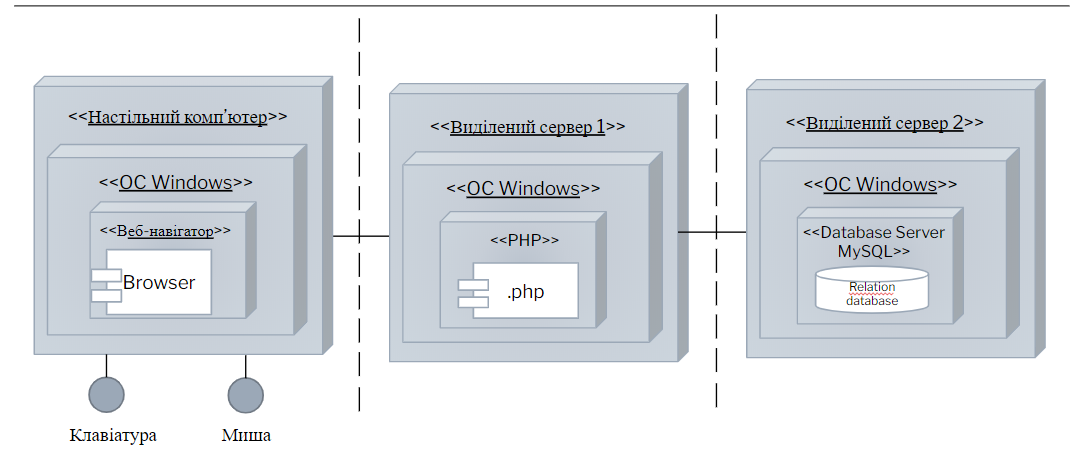


Рис. 2.1 – Діаграма архітектури.

## 2.3 План розробки програмного продукту

### 2.3.1 Оцінка трудомісткості розробки програмного продукту

Таблиця 2.1 – Вагові коефіцієнти акторів.

|  |  |
| --- | --- |
| Актор | Ваговий коефіцієнт |
| Гість | 1 |
| Користувач | 2 |
| Адміністратор | 3 |

Таблиця 2.2 – Вагові коефіцієнти прецедента.

|  |  |  |  |
| --- | --- | --- | --- |
| Ідентифікатор функції | Тип  прецедента | Кількість кроків  сценарію | Ваговий  коефіцієнт |
| FR1 | Середній | 4-7 | 15 |
| FR2 | Простий | <=3 | 10 |
| FR3 | Середній | 4-7 | 5 |
| FR4 | Простий | <=3 | 5 |
| FR5 | Простий | <=3 | 15 |
| FR6 | Простий | <=3 | 10 |
| FR7 | Простий | <=3 | 5 |

Визначення UUCP :

А = 3 + 2 + 1 = 6

UC = 15 + 10 + 5 + 5 + 15 + 10 + 5 = 65

UUCP = 6 + UC = 71

Таблиця 2.3 – Визначення технічної складності.

|  |  |  |
| --- | --- | --- |
| Показник | Опис показника | Вага |
| Т1 | Розподілена система | 1 |
| Т2 | Висока продуктивність (пропускна здатність) | 2 |
| Т3 | Робота кінцевих користувачів в режимі онлайн | 1 |
| Т4 | Складна обробка даних | 0,5 |
| Т5 | Повторне використання коду | 0,5 |
| Т6 | Простота установки | 0,5 |
| Т7 | Простота використання | 1 |
| Т8 | Переносимість | 1 |
| Т9 | Простота внесення змін | 2 |
| Т10 | Паралелізм | 0 |
| Т11 | Спеціальні вимоги до безпеки | 1 |
| Т12 | Безпосередній доступ до системи з боку зовнішніх користувачів | 3 |
| Т13 | Спеціальні вимоги до навчання користувачівь | 1 |

Визначення TCF :

TCF = 0,6 + (0,01 \* (STi \* Вага\_i))

TCF = 0,6 + (0,01 \* (14,5)) = 0,745

Таблиця 2.4 – Рівень класифікації.

|  |  |  |
| --- | --- | --- |
| Показник | Опис показника | Вага |
| F1 | Знайомство з технологією | 2 |
| F2 | Досвід розробки додатків | 1 |
| F3 | Досвід використання об'єктно-орієнтованого підходу | 1 |
| F4 | Наявність провідного аналітика | 0 |
| F5 | Мотивація | 3 |
| F6 | Стабільність вимог | 4 |
| F7 | Часткова зайнятість | 1 |
| F8 | Складні мови програмування | 1 |

Визначення EF :

ЕF = 1,4 + (-0,03 \* (SFi \* Вага\_i)) = 1

UUCP = 71

TCF = 0,745

ЕF = 1

UCP = UUCP \* TCF \* EF = 52,895

Трудомісткість = 52,895 \* 20 = 1057,9 люд./год.

### 2.3.2 Визначення дерева робіт з розробки програмного продукту

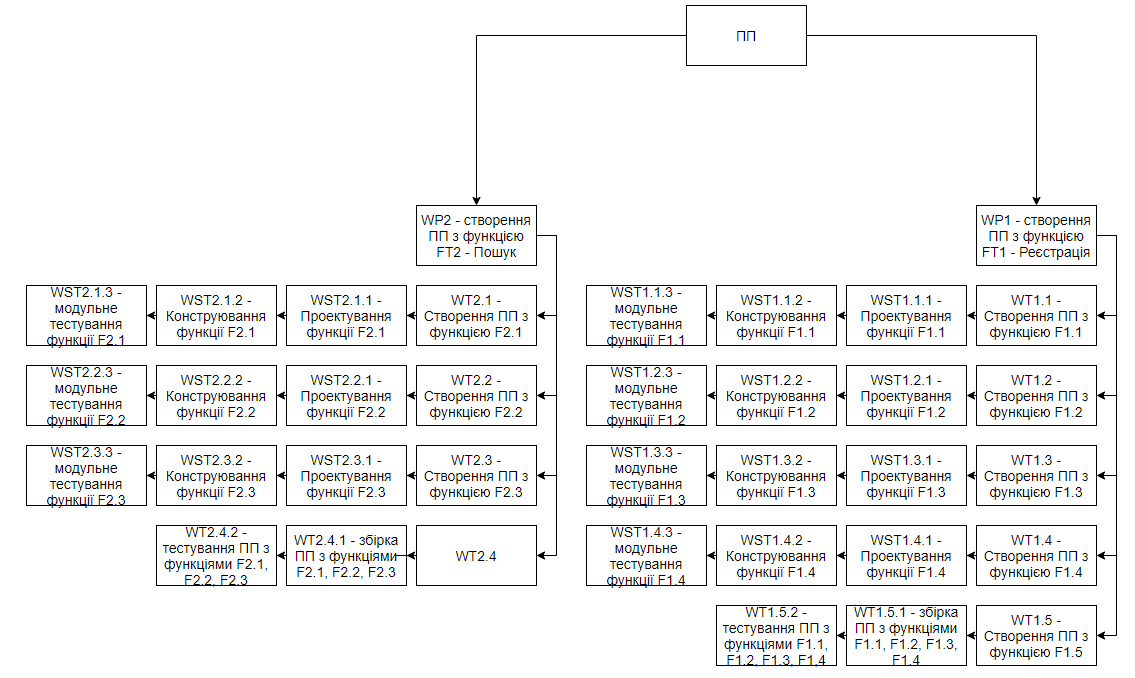


Рис. 2.2.1 – Дерево робіт.

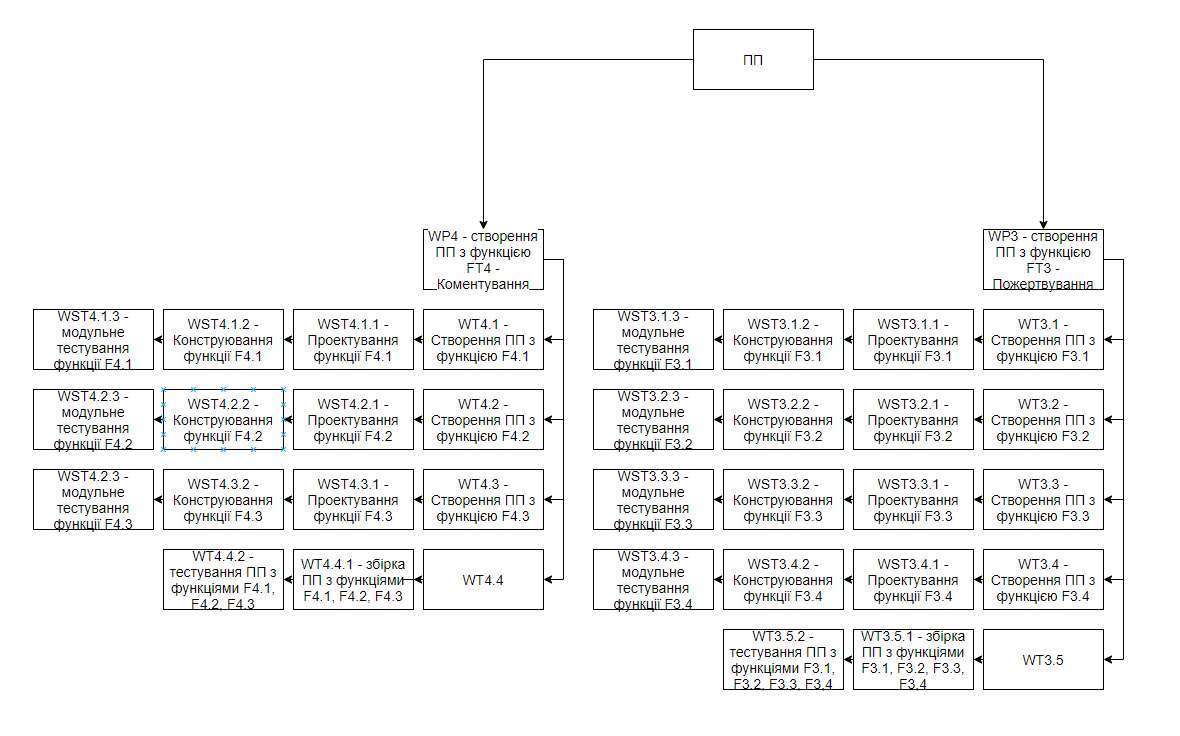


Рис. 2.2.2 – Дерево робіт.

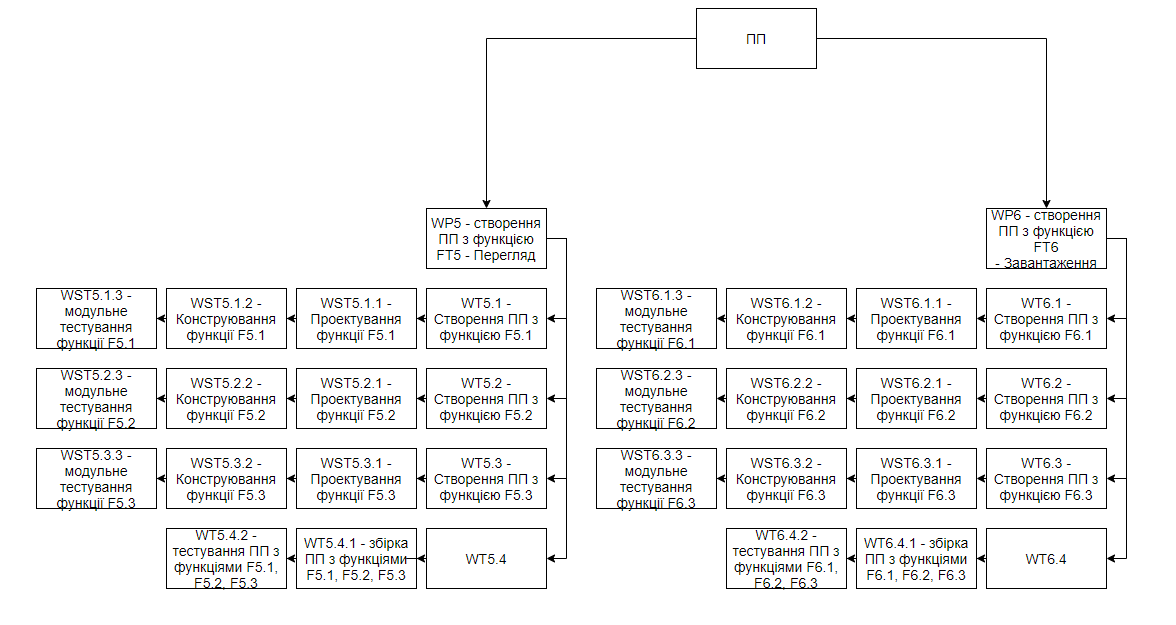


Рис. 2.2.3 – Дерево робіт.

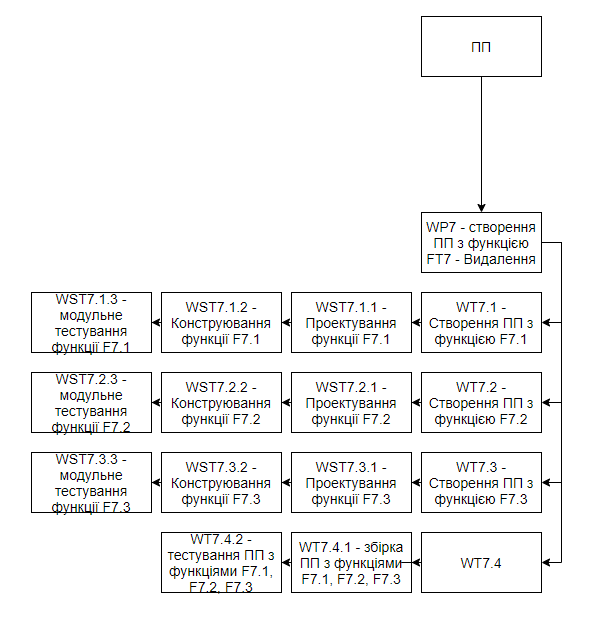


Рис. 2.2.4 – Дерево робіт.



Рис. 2.3 – Таблиця з описом підзадач із закріпленням виконавців.

### 2.3.3 Графік робіт з розробки програмного продукту

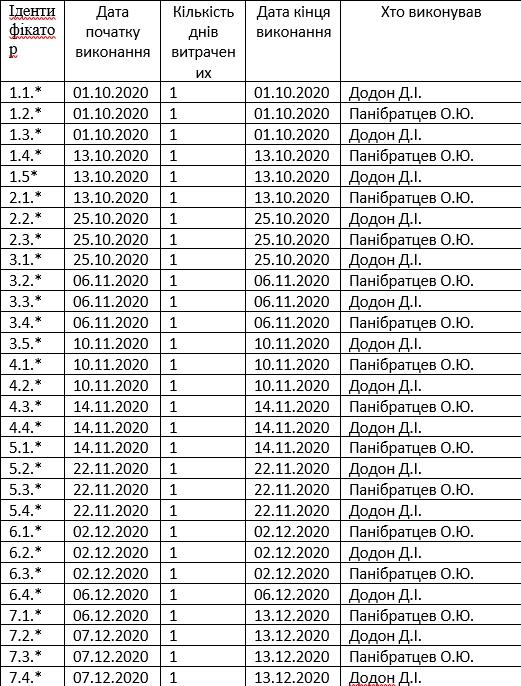


Рис. 2.4 – Графік робіт.

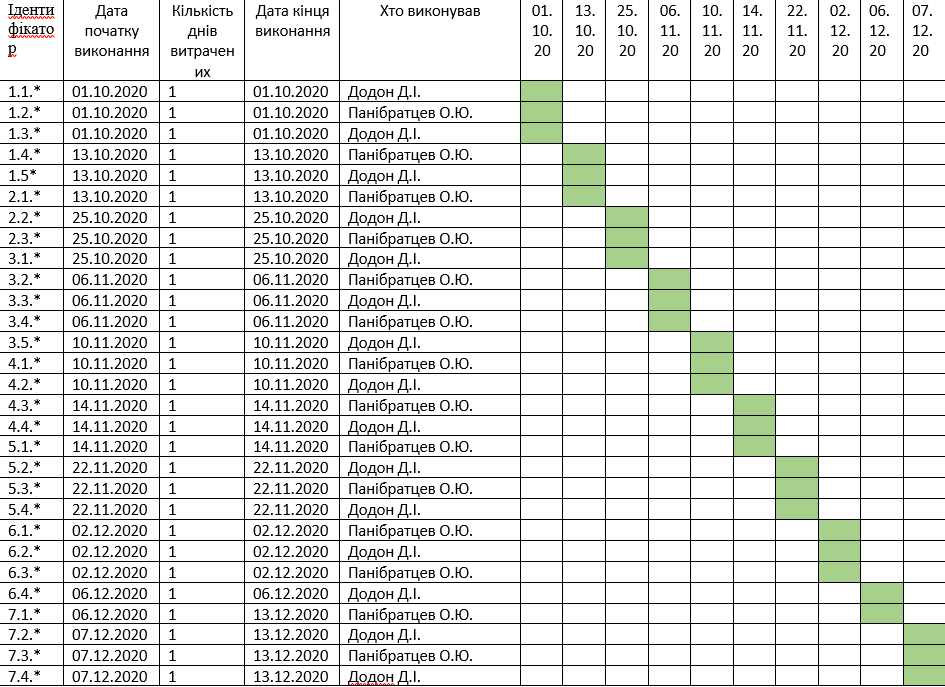


Рис. 2.5 – Діаграма Ганта.

# 3 Проектування програмного продукту

## 3.1 Концептуальне та логічне проектування структур даних програмного продукту.

### 3.1.1 Концептуальне проектування на основі UML-діаграми концептуальних класів.

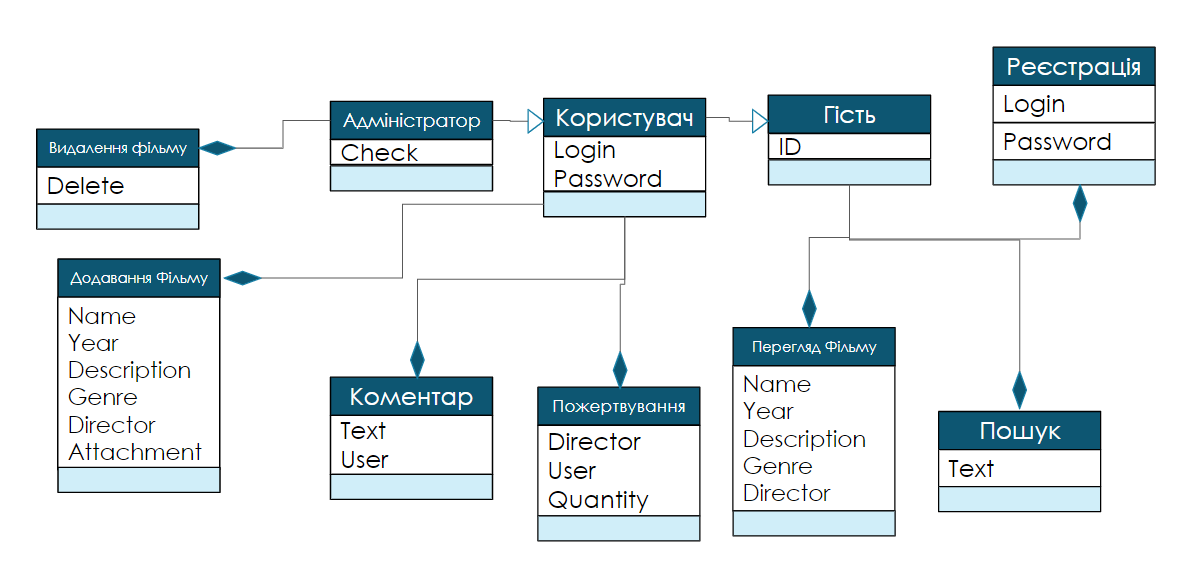


Рис. 3.1 – Діаграма концептуальних класів.

### 3.1.2 Логічне проектування структур даних.

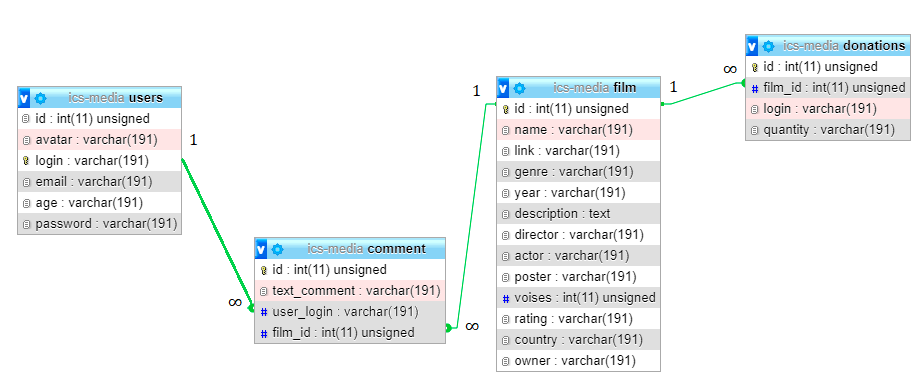


Рис. 3.2 – Проектування бази даних.

## 3.2 Проектування програмних класів

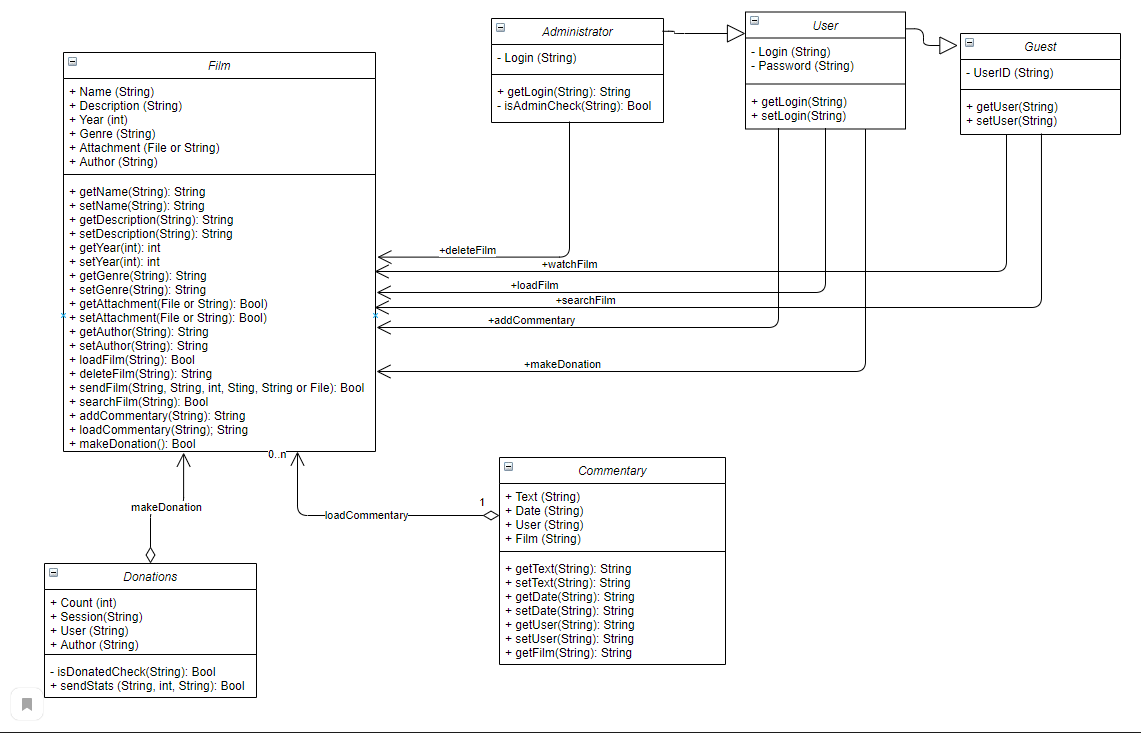


Рис. 3.2 – Діаграма програмних класів.

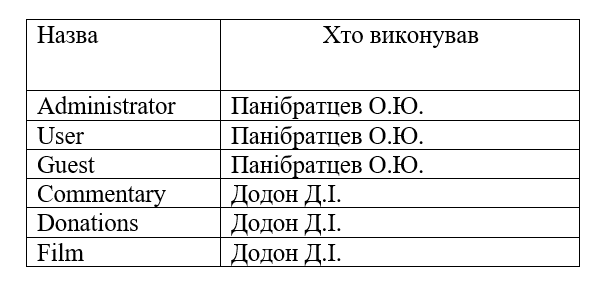


Рис. 3.3 – Задачі для учасників.

## 3.3 Проектування алгоритмів роботи методів програмних класів

Код UML – діаграми для реєстрації:

@startuml

title Registration()

start

:Виведення на екран полей для вводу для реєстрації;

note right

Mockup полей можна знайти у пункті 1.5.1.1.2 (FR1)

end note

if (Введені дані в усі обов'язкові поля?) then (yes)

:Введення в базу даних нового облікового запису користувача;

note left

INSERT INTO users (id, login, password) VALUES (1, 'user', 'password');

end note

else (no)

:Виведення помилки реєстрації;

endif

stop

@enduml



Рис. 3.4 – UML – алгоритм реєстрації.

Код UML – діаграми для пошуку:

@startuml

title Search()

start

:Виведення на екран поля для пошуку;

note right

Mockup полей можна знайти у пункті 1.5.1.1.2 (FR2)

end note

:Вводиться запрос до строки пошуку;

:Відправляеться зарос до бази;

note left

SELECT \* FROM film WHERE film.name = "filmName"

end note

if (База має такий фільм) then (yes)

:Введення на екран користувача сторінку фільму

else (no)

:Виведення про відсутність фільма;

endif

stop

@enduml

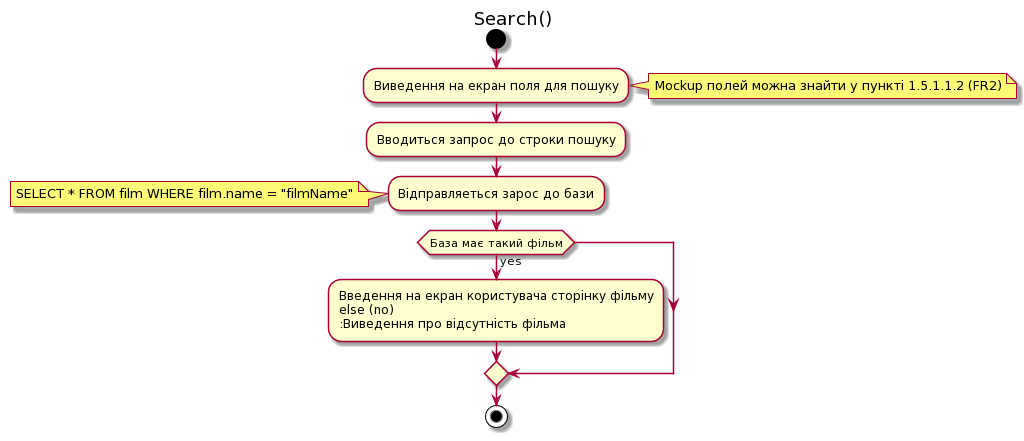


Рис. 3.5 - UML – діаграма для пошуку.

Код UML – діаграми для донату:

@startuml

title Donate()

start

:Виведення на екран поля для донату;

note right

Mockup полей можна знайти у пункті 1.5.1.1.2 (FR3)

end note

:Вводиться кількість грошей до строки донату;

:Перенаправлення до зовнішнього сервісу;

if (Оплата успішна) then (yes)

:Введення на екран успішного повідомлення та відправка даних до бази;

note left

INSERT INTO donations (id, film\_id, login, quantity) VALUES (id, f\_id, login, 999)

end note

else (no)

:Виведення про не успішний перевід коштів;

endif

stop

@enduml

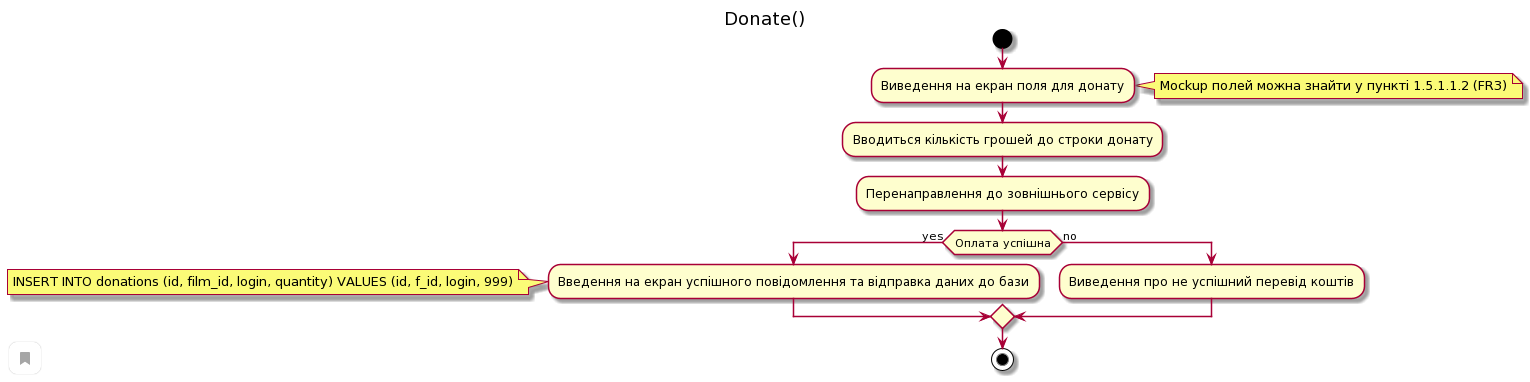


Рис. 3.6 – UML – діаграма для донату.

Код UML – діаграми для коментування:

@startuml

title Commentaries()

start

:Виведення на екран поля для вводу коментарів;

note right

Mockup полей можна знайти у пункті 1.5.1.1.2 (FR4)

end note

:Вводиться текст коментарю та відправляється до бази;

note left

INSERT INTO comment (id, text\_comment, user\_login, film\_id) VALUES (id, 'Классный фильм.', 'admin', f\_id)

end note

@enduml

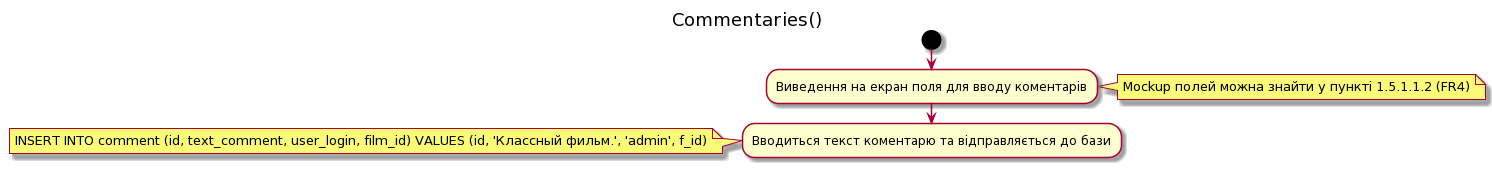


Рис. 3.7 – UML – діаграма для коментування.

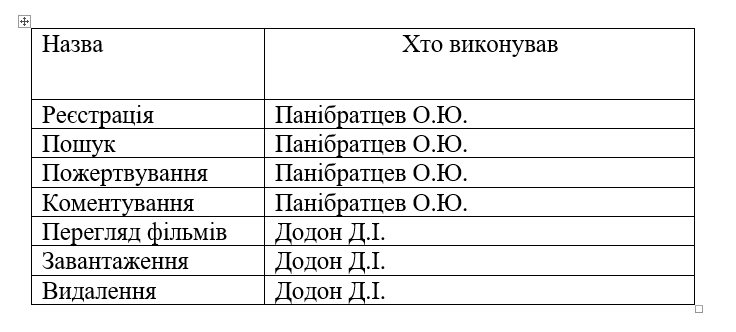


Рис. 3.8 – Таблиця учасників.

## 3.4 Проектування тестових наборів методів програмних класів

Таблиця 3.1 – Тестовий набор для реєстрації.

|  |  |  |
| --- | --- | --- |
| № тесту | Опис значень вхідних даних | Опис очікуваних значень результату |
| 1 | 1. login = ' ' 2. password = ‘’ 3. натиснути send | Повідомлення про невведені дані. |
| 2 | 1. login = ‘login’ 2. password = ' ' 3. натиснути send | Повідомлення про невведені дані. |
| 3 | 1. login = ' ' 2. password = ‘password’ 3. натиснути send | Повідомлення про невведені дані. |
| 4 | 1. login = ‘login’ 2. password = 'password' 3. Натиснути send | Залоговано і перенесло до головної сторінки |

Таблиця 3.2 – Тестовий набор для перегляду фільмів.

|  |  |  |
| --- | --- | --- |
| № тесту | Опис значень вхідних даних | Опис очікуваних значень результату |
| 1 | film\_id = “1” | Загрузка фрейму з фільмом. |
| 2 | film\_id = “231242” | ПП не видає інформації. |

Таблиця 3.3 – Тестовий набор для додавання фільмів.

|  |  |  |
| --- | --- | --- |
| № тесту | Скріпти | Опис очікуваних значень результату |
| 1 | 1. $id = id  $film = film  $poster = poster  $name = name  $link = link  $description = description  2. Натиснути відправити. | Повідомлення що фільм додано, можна знайти в каталозі. |
| 2 | 1. Будь-яке поле з першого кейсу пусте.  2. Натиснути відправити. | «Вы пропустили это поле» |



Рис. 3.9 – Таблиця учасників.

# 4 Конструювання програмного продукту

## 4.1 Особливості конструювання структур даних

### 4.1.1 Особливості інсталяції та роботи з СУБД

В даному ПП використовувалася СКБД MySQL 5.0.4 з програмою XAMPP.

Встановлення на Windows:

1. Переходимо на сайт розробника.

2. Встановлюємо собі на ПК XAMPP.

3. Запускаємо XAMPP і можемо перейти до бази MySQL клікнувши на кнопку Admin.

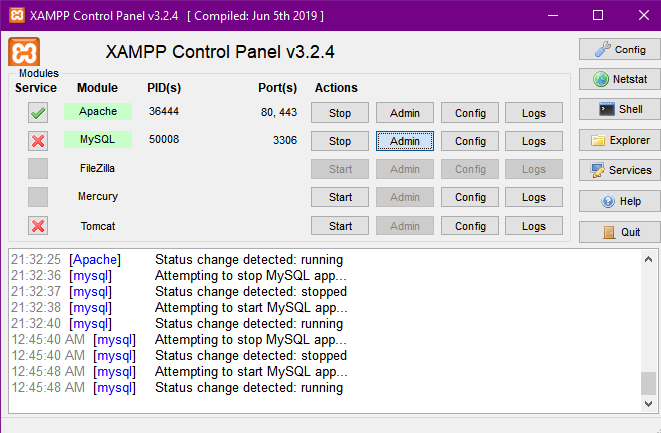


Рис. 4.1 – Вікно XAMPP.

### 4.1.2 Особливості створення структур даних

Створення таблиці фільмів:

Таблиця з фільмами містить інформацію про фільми, а саме – назва, автор, год, жанр, опис, коментарі за айді та рейтинг.

CREATE TABLE `film` (

  `id` int(11) NOT NULL PRIMARY KEY,

  `name` varchar(191),

  `link` varchar(191),

  `genre` varchar(191),

  `year` int(11),

  `description` text,

  `director` varchar(191),

  `actor` varchar(191),

)

Таблиця користувачів містить унікальний логін та пароль.

CREATE TABLE `users` (

  `login` varchar(191) NOT NULL PRIMARY KEY,

  `password` varchar(191))

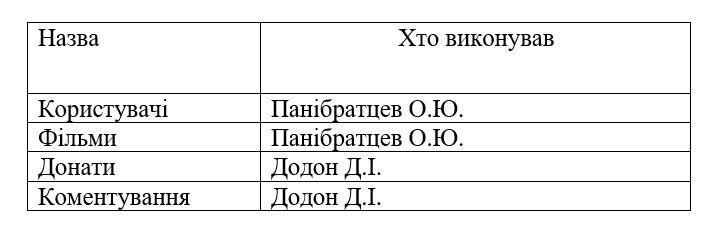


Рис. 4.2 – Таблиця учасників.

## 4.2 Особливості конструювання програмних модулів

### 4.2.1 Особливості роботи з інтегрованим середовищем розробки

Так як розробка проходила в середовищі Sublime, ми мали можливість застосовувати багато плагінів. Sublime дуже швидкий, безкоштовний, інтуїтивно зрозумілий.

Також, Sublime має швидкий доступ до github. З ним, ми набагато швидше робили push.

### 4.2.2 Особливості створення програмної структури з урахуванням спеціалізованого Фреймворку

Особливість створення нашого коду в тому, що ми використовували бібліотеку яка набагато зручніше працювала з БД. Називається вона RedBean.

Особливості написання коду коментування:

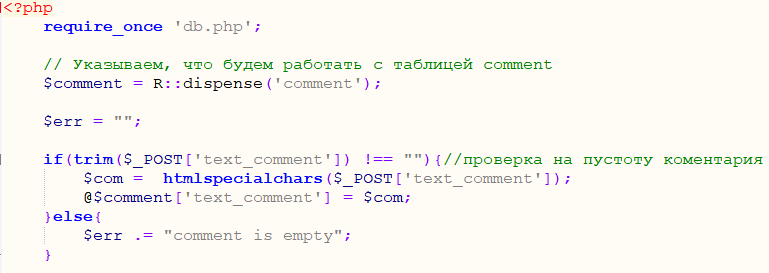


Рис. 4.3 – Приклад конекту до БД та перевірки на пустоту.

У цьому моменті ми бачимо як відбувається конект до БД та вибір таблиці.

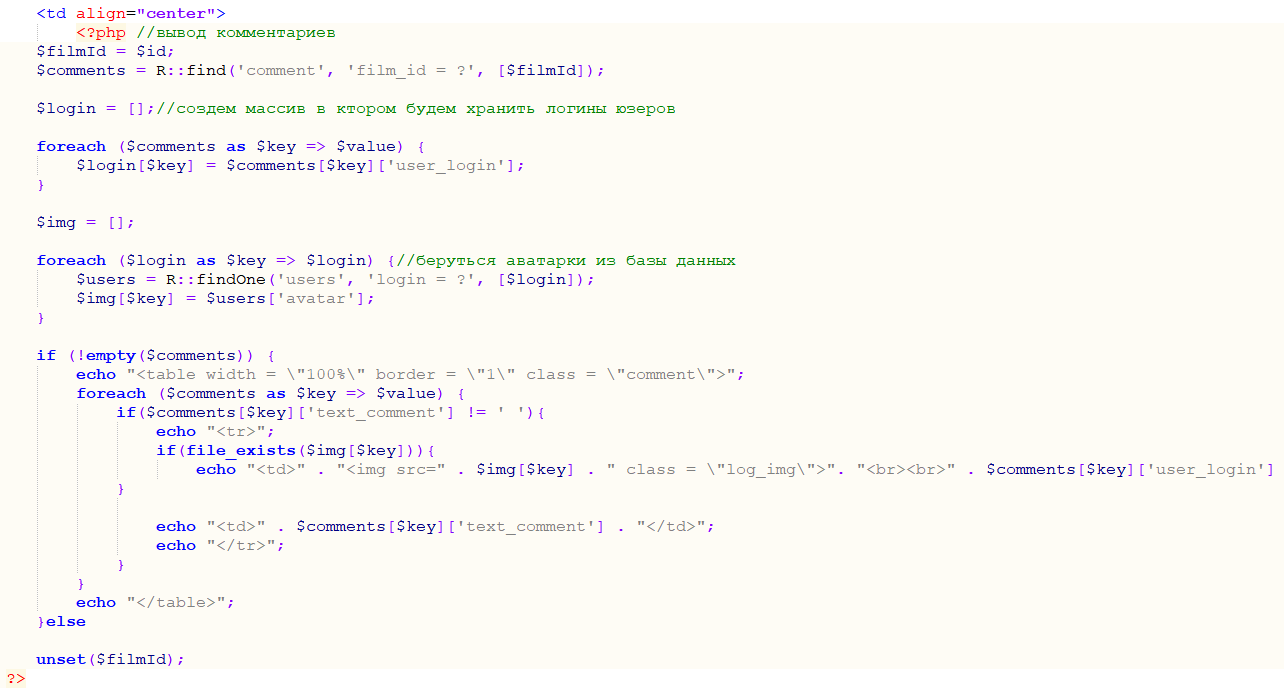


Рис. 4.4 – Вивід коментарів до сторінки фільму.

Особливості написання коду перегляду фільмів:

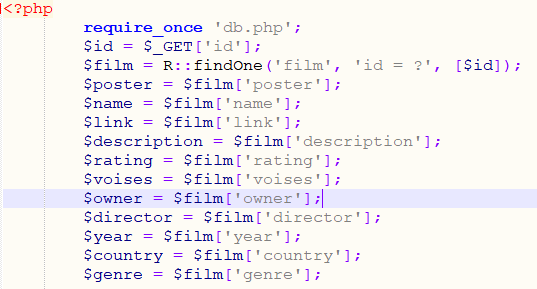


Рис. 4.5 – Приклад додавання даних с таблиці в масив.

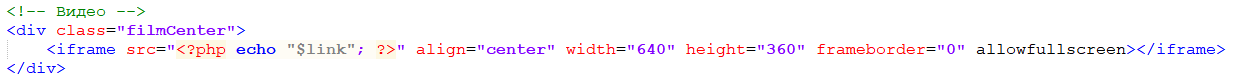


Рис. 4.6 – Додавання фільму до сторінки.

У цьому прикладі можна побачити як відбувається заповнення масиву даними про фільм, а потім, фільм відображається на сторінці в <iframe>.

Особливості написання коду про додавання фільму:



Рис. 4.7 – Додавання фільму.

Можемо побачити перевірку на пустоту полей.

Знов заповняється масив даних $film, а потім загружається до БД командою R::store.

Особливості написання коду про видалення фільму:

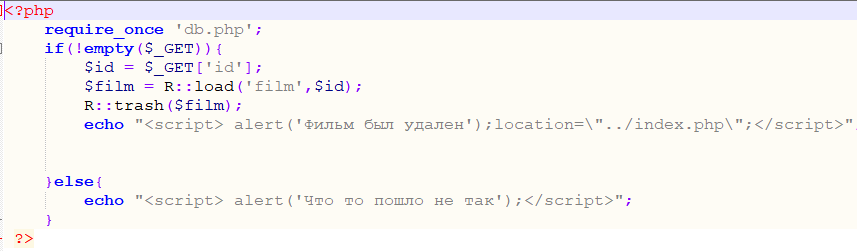


Рис. 4.8 – Видалення фільмів.

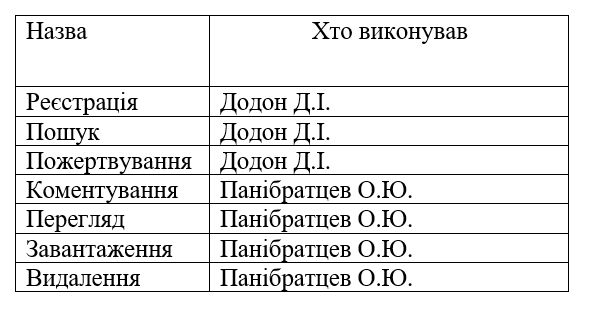


Рис. 4.9 – Таблиця учасників.

Бачимо найпростішу функцію яка просто видаляє поле. R::trash.

### 4.2.3 Особливості створення програмних модулів

Всі наші програмні модулі зберігаються на власному ПК.

Атож, нам потрібно їх тримати на сервері. Як ми вже знаємо – сервер XAMPP. Працює на Apache.   
  
 Тому всі програмні модулі тримаються у нас у файлах на локальному диску, та, у випадку необхідності, запускаються з відкритим портом до глобальної мережи інтернет.

Все програмні модулі зберігаються у папці cgi (мається на увазі «Сценарії користування», тобто – ядро).

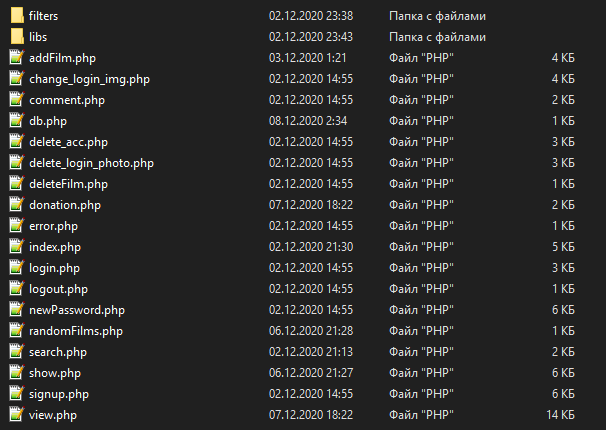


Рис. 4.10 – Структура програмних модулів.

### 4.2.4 Особливості розробки алгоритмів методів програмних класів або процедур/функцій

Особливості реєстрації:

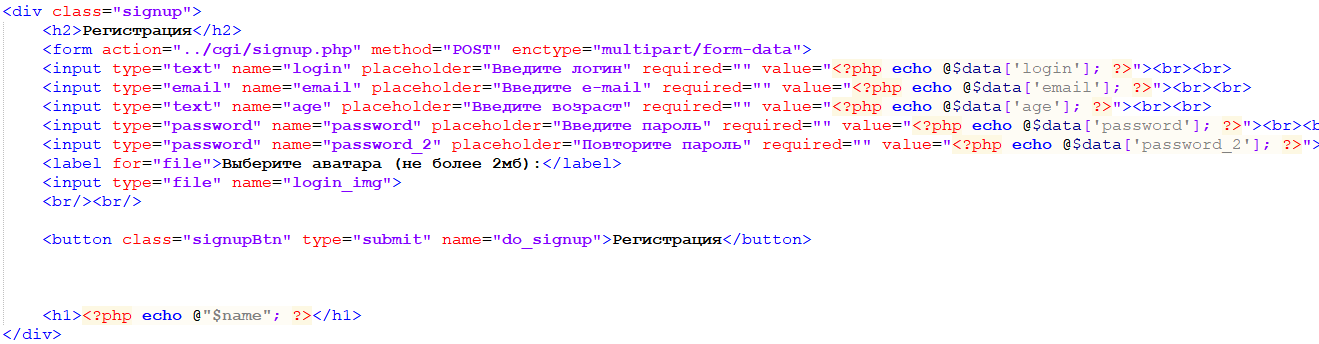


Рис. 4.11 – Реєстрація.

Реєстрація – це проста html форма, в яку впроваджено php змінні.

Особливості пошуку:

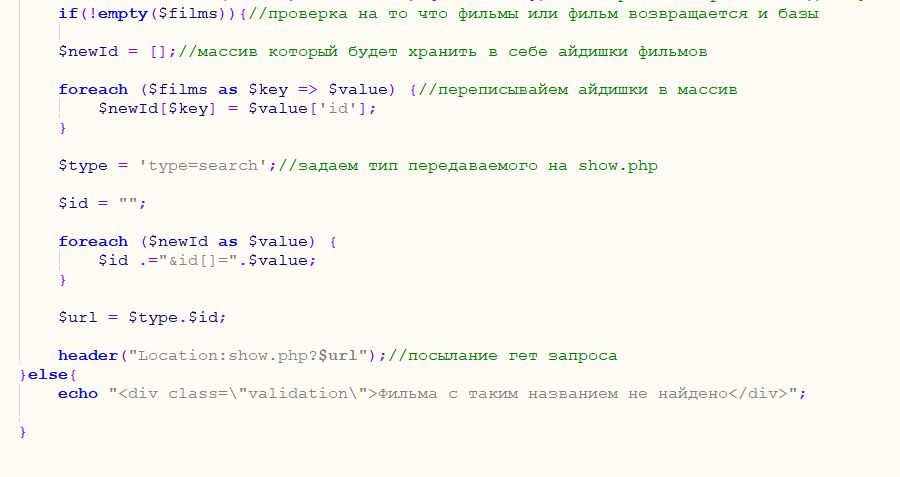


Рис. 4.12 – Пошук.

Пошук працює з GET запитом із БД. Забирається масив фільмів, та зранюється з запитом у пошуку.

Особливості донатування:

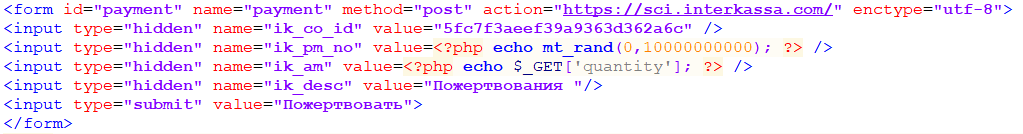


Рис. 4.13 – Донат.

Донат використовує зовнішній сервіс intercassa. Модуль використовує рандомайзер для того щоб зробити рандомний номер заказу.

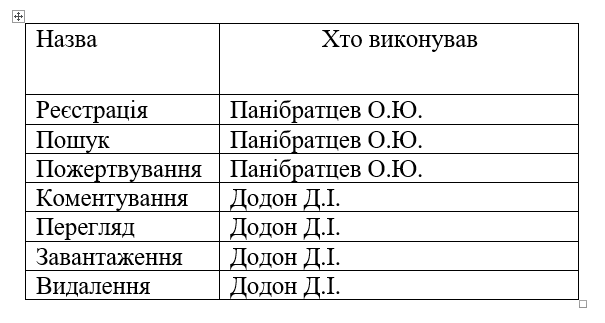


Рис. 4.14 – Таблиця учасників.

4.3 Тестування програмних модулів

Особливості тестування:

BDD тести - це простий текст, на людській мові, написаний у формі історії (сценарію), яка описує деяку поведінку.

У Cucumber для написання тестів використовується Gherkin-нотація, яка визначає структуру тесту і набір ключових слів. Тест записується в файл з розширенням \*.feature і може містити як один, так і більш сценаріїв.

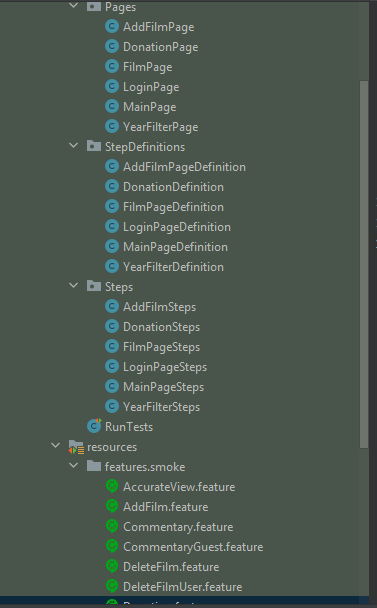


Рис. 4.15 – Файлова структура автоматичних тестів.

Serenity - это фреймворк с открытым исходным кодом для написания функциональных тестов, в который из коробки встроена система генерации отчетов о выполненных тестах.

Тести робились з чотирьома видами написання сценаріїв.

У цьому виді прописано сценарії використання програмою. Тестування робиться з зовнішнім інтерфейсом (метод BlackBox).

Сценарій тестування функції коментування:

Feature: User can add commentary  
 Scenario: User must be logged in  
 Given User can access main page "http://lopik.zapto.org/"  
 Then User click at cabinet button  
 And user click to authorization  
 Then user enter login2 "user"  
 And user enter passwordd  
 Then click Log In button  
 Then user click at any page of film  
 And user enter his commentary "check"  
 Then user click send commentary button  
 Given user can see his commentary  
 Then user click send commentary button  
 And page reloaded

Сценарій тестування функції додавання фільму:

Feature: User can add film to account  
 Scenario:  
 Given User can access main page "http://lopik.zapto.org/"  
 Then User click at cabinet button  
 And user click to authorization  
 Then user enter login "admin"  
 And user enter passwordr  
 Then click Log In button  
 Then User click at cabinet button  
 Given user can add film by clicking button  
 Then user enter name of film "name"  
 And user enter url of film "check"  
 And user enter genre of film "genre"  
 And user enter year of film "2020"  
 And user enter director of film "director"  
 And user enter country of film "country"  
 And user delete url of film  
 Then click send button  
 And Validation message appeared  
 And user enter url of film "check"  
 And user enter link to poster "poster"  
 And user enter description "description"  
 Then click send button

Сценарій тестування функції видалення фільму:

Feature: Admin can delete film  
 Scenario:  
 Given User can access main page "http://lopik.zapto.org/"  
 Then User click at cabinet button  
 And user click to authorization  
 Then user enter login "admin"  
 And user enter passwordr  
 Then click Log In button  
 Then user click at any page of film  
 And click delete button

Сценарій тестування функції перегляду фільму.

Feature: user can watch films  
 Scenario:  
 Given User can access main page "http://lopik.zapto.org/"  
 Then user click at any page of film  
 Then user can see frame

Отримання елементів зі сторінки коментарів.

@FindBy(xpath = "/html/body/div[2]/form/input[3]")  
public WebElement sendCommentary;  
  
@FindBy(xpath = "//\*[contains(text(), 'Check')]")  
public WebElement textCommentary;

Отримання елементів зі сторінки додавання фільмів.

@FindBy(name = "filmName")  
public WebElement name;  
  
@FindBy(name = "filmURL")  
public WebElement url;  
  
@FindBy(name = "genre")  
public WebElement genre;  
  
@FindBy(name = "year")  
public WebElement year;  
  
@FindBy(name = "director")  
public WebElement director;  
  
@FindBy(name = "country")  
public WebElement country;  
  
@FindBy(name = "actor")  
public WebElement actor;  
  
@FindBy(name = "poster")  
public WebElement poster;  
  
@FindBy(className = "addBtn")  
public WebElement addButton;  
  
@FindBy(name = "description")  
public WebElement description;

Отримання елементів зі сторінки видалення фільмів.

@FindBy(xpath = "/html/body/div[2]/table/tbody/tr/td[5]/a")  
public WebElement deleteButton;

Отримання елементів зі сторінки перегляду фільмів.

@FindBy(className = "filmCenter")  
public WebElement cinema;

Опис шагів тестування коментування:

@Then("^user enter his commentary \"([^\"]\*)\"$")  
public void user\_enter\_his\_commentary(String arg1) {  
 filmPageSteps.enterCommentary();  
}  
  
@Then("^user click send commentary button$")  
public void user\_click\_send\_commentary\_button() {  
 filmPageSteps.loadCommentary();  
}  
  
@Given("^user can see his commentary$")  
public void user\_can\_see\_his\_commentary() {  
 filmPageSteps.isCommentary();  
}

Опис шагів тестування додавання фільмів:

@Then("^user enter name of film \"([^\"]\*)\"$")  
public void user\_enter\_name\_of\_film(String arg1) {  
 addFilmSteps.enterName();  
}  
  
@Then("^user enter url of film \"([^\"]\*)\"$")  
public void user\_enter\_url\_of\_film(String arg1) {  
 addFilmSteps.enterUrl();  
}  
  
@Then("^user enter genre of film \"([^\"]\*)\"$")  
public void user\_enter\_genre\_of\_film(String arg1) {  
 addFilmSteps.enterGenre();  
}  
  
@Then("^user enter year of film \"([^\"]\*)\"$")  
public void user\_enter\_year\_of\_film(String arg1) {  
 addFilmSteps.enterYear();  
}  
  
@Then("^user enter director of film \"([^\"]\*)\"$")  
public void user\_enter\_director\_of\_film(String arg1) {  
 addFilmSteps.enterDirector();  
}  
  
@Then("^user enter country of film \"([^\"]\*)\"$")  
public void user\_enter\_country\_of\_film(String arg1) {  
 addFilmSteps.enterCountry();  
}  
  
@Then("^user enter link to poster \"([^\"]\*)\"$")  
public void user\_enter\_link\_to\_poster(String arg1) {  
 addFilmSteps.enterPoster();  
}  
  
@Then("^user enter description \"([^\"]\*)\"$")  
public void user\_enter\_description(String arg1) {  
 addFilmSteps.enterDescription();  
}  
  
@Then("^click send button$")  
public void click\_send\_button() {  
 addFilmSteps.clickSubmit();  
}

Опис шагів тестування видалення фільмів:

@Then("^click delete button$")  
public void click\_delete\_film() {  
 filmPageSteps.clickDelete();  
}

Опис шагів тестування перегляду фільмів:

@Then("^user can see frame$")  
public void user\_watch\_film(){  
 filmPageSteps.isWatch();  
}

Результати тестів:

Результати тестування коментування:

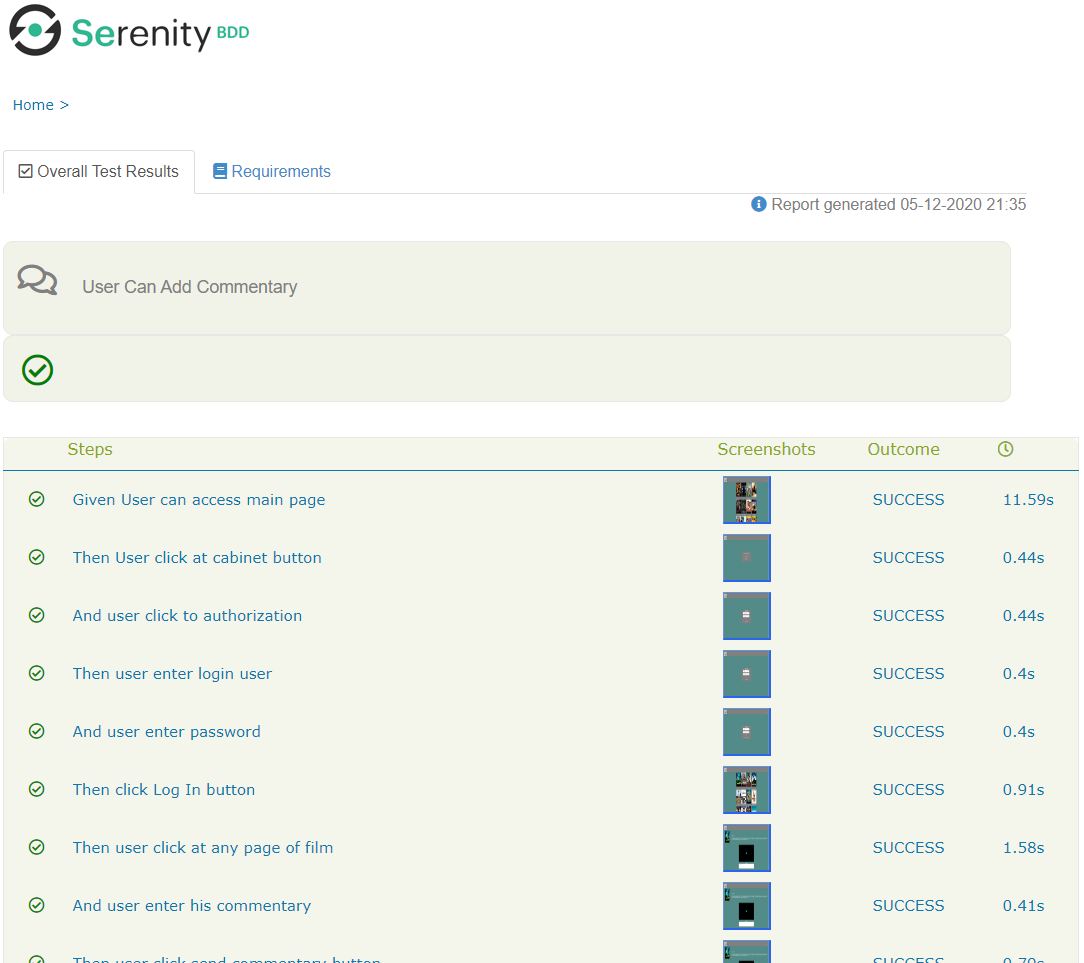


Рис. 4.16 – Результат тесту коментування.

Результати тестування додавання фільмів:

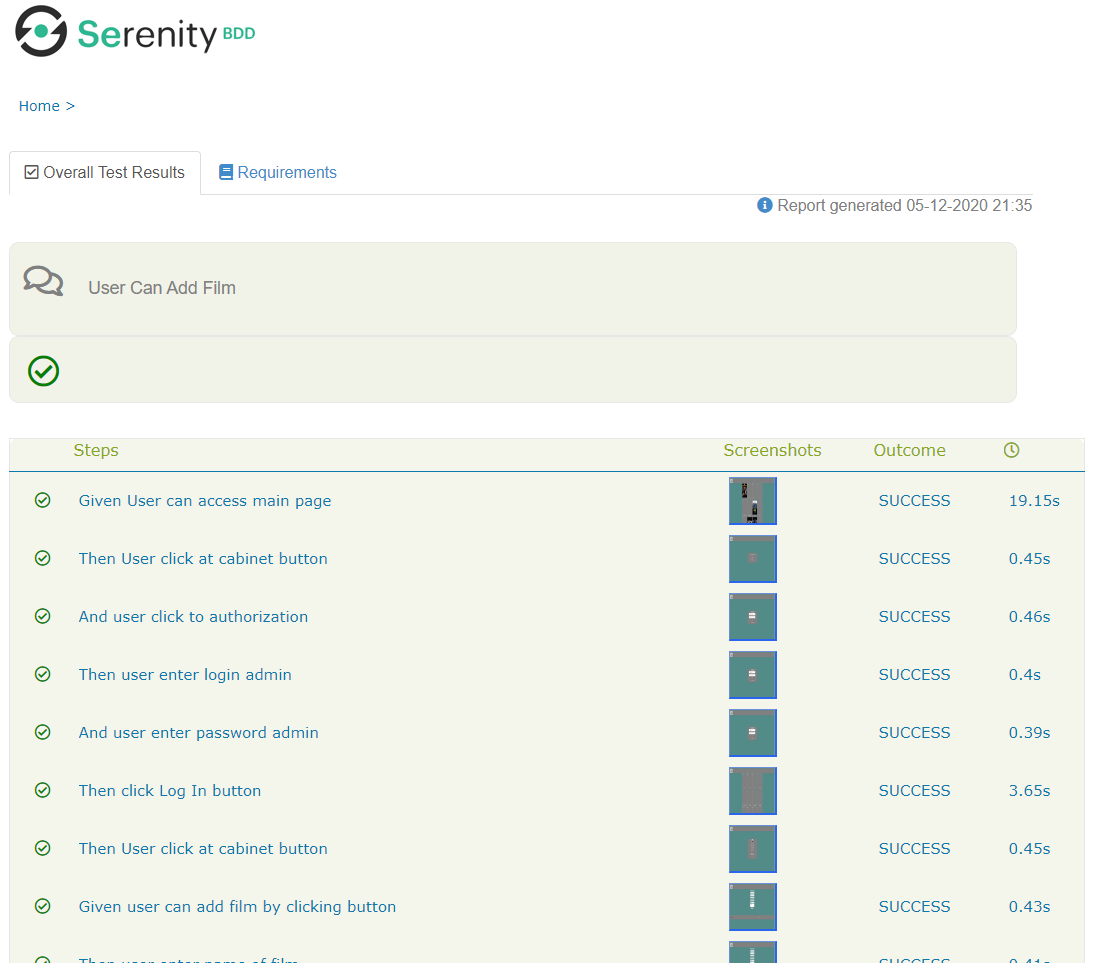


Рис. 4.17 – Результат тесту додавання фільму.

Результати тестування виделання фільмів:

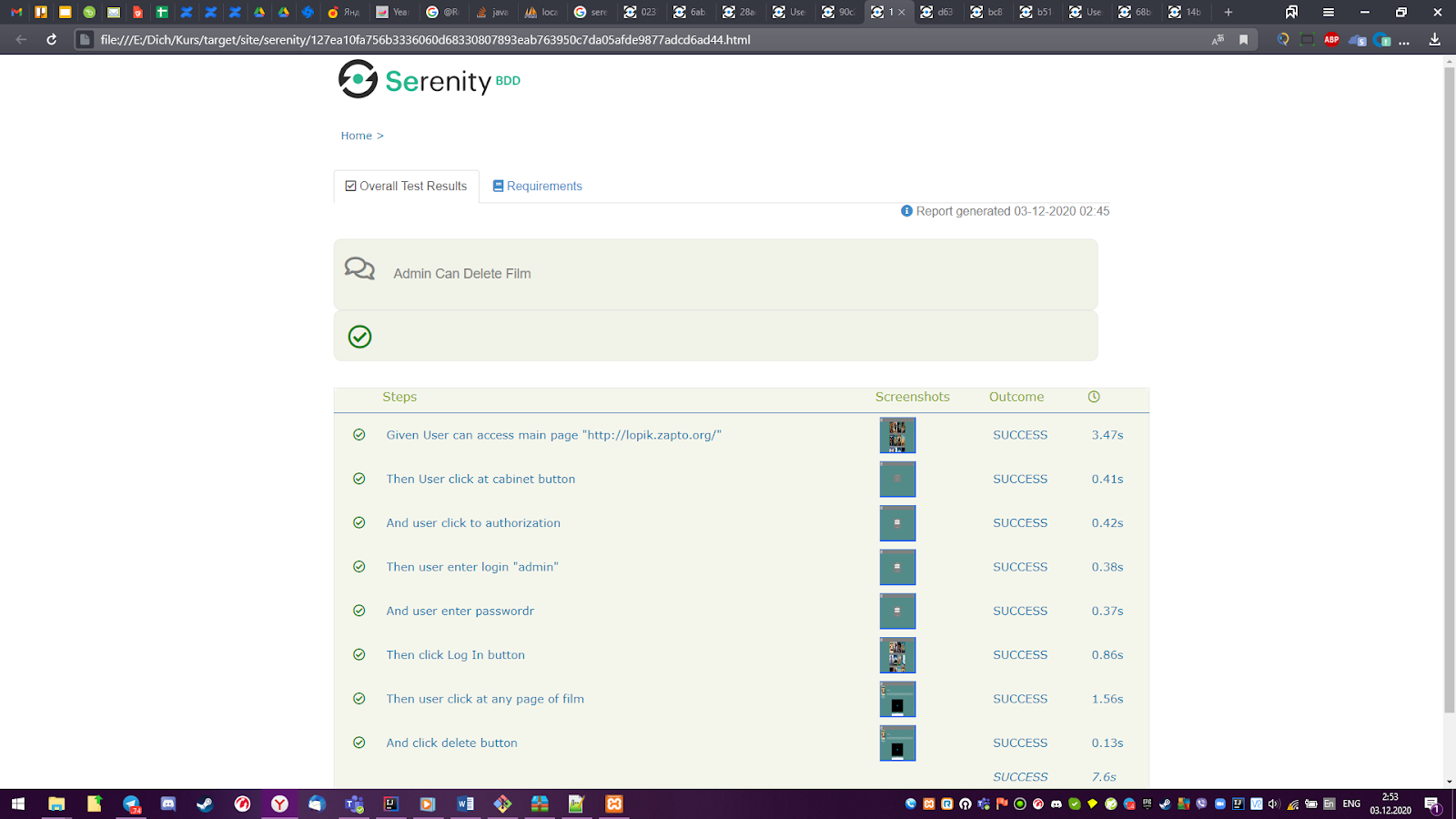


Рис. 4.18 -

Результати тестування перегляду фільмів:

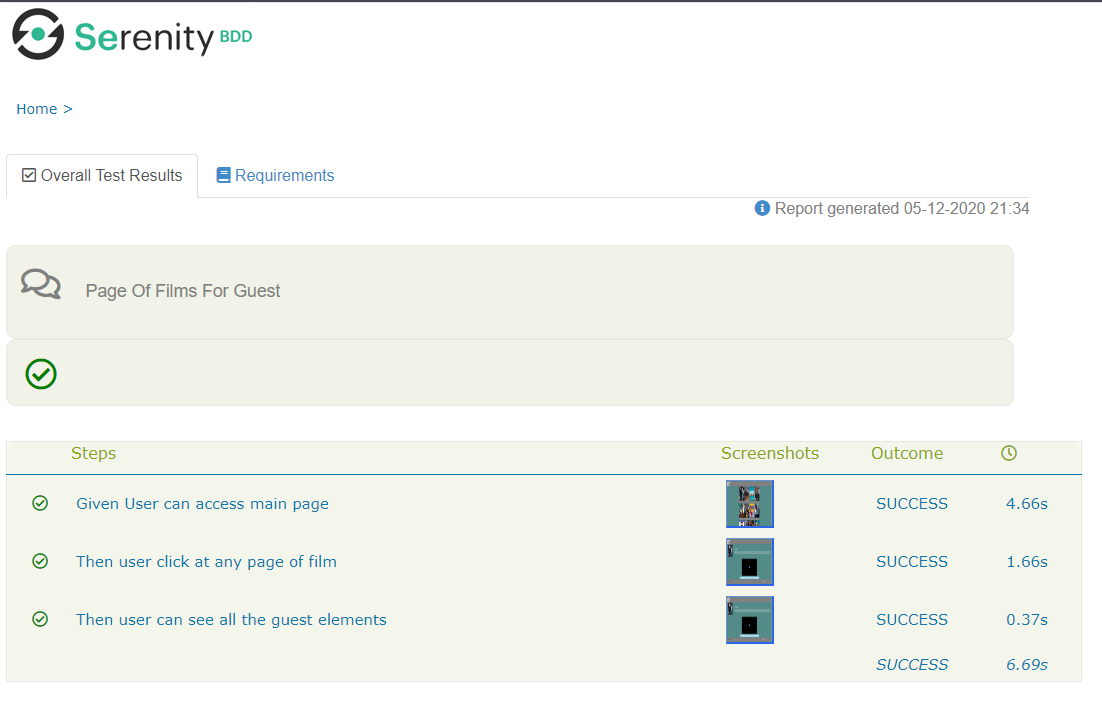


Рис. 4.19 – Результат тестування перегляду фільмів.

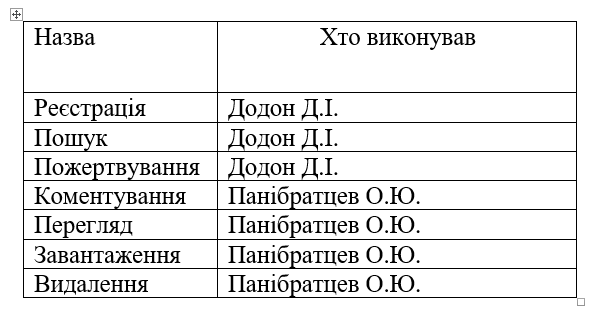


Рис. 4.20 – Таблиця учасників.

# 5 Розгортання та валідація програмного продукту

## 5.1 Інструкція з встановлення програмного продукту

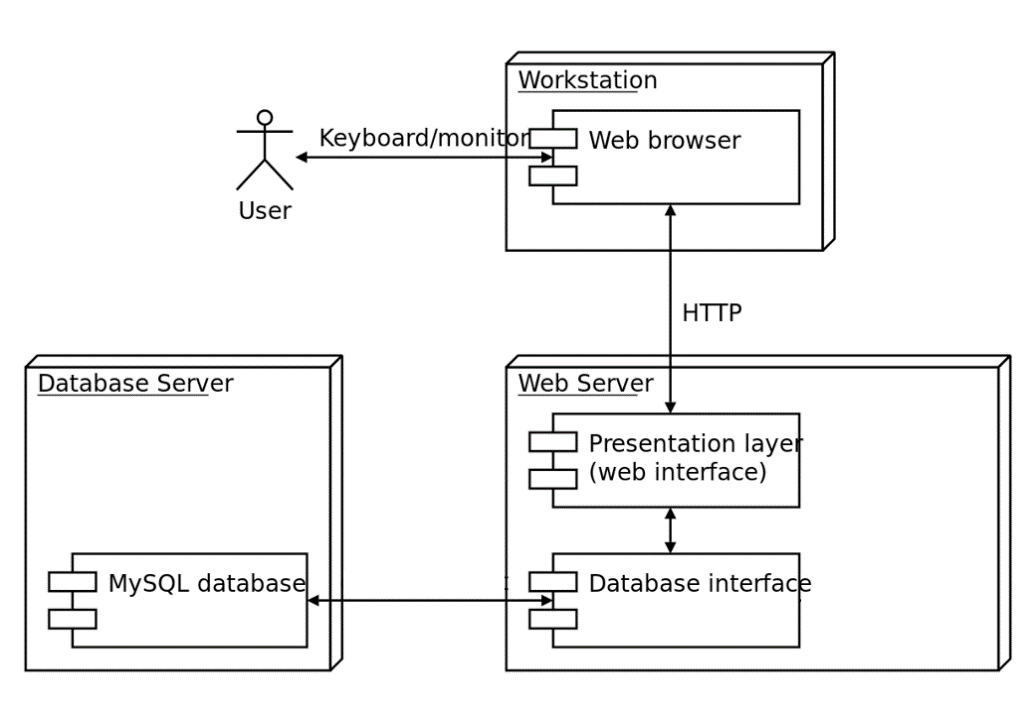


Рис. 5.1 – Діаграма розгортання.

Встановлення на Windows серверу:

1. Переходимо на сайт розробника XAMPP та загружаємо файл.

2. Встановлюємо собі на ПК XAMPP.

3. Запускаємо XAMPP і можемо перейти до бази MySQL та запустити сайт клікнувши на кнопку Admin.

Встановлення сайту для розробників:

1. Переходимо на GitHub репозиторій цього проекту.

2. Робимо клон собі на комп’ютер.

3. Переміщуємо папку template яка була зклонована разом с проектом у папку htdocs програми XAMPP.

## 5.2 Інструкція з використання програмного продукту

Інструкція з реєстрації:

1. Ви повинні зайти до нашого сайту, та натиснути кнопку кабінету яка знаходиться біля лого нашого сайту.

2. Далі треба перейти до реєстрації, та заповнити поля які потребує ПП.

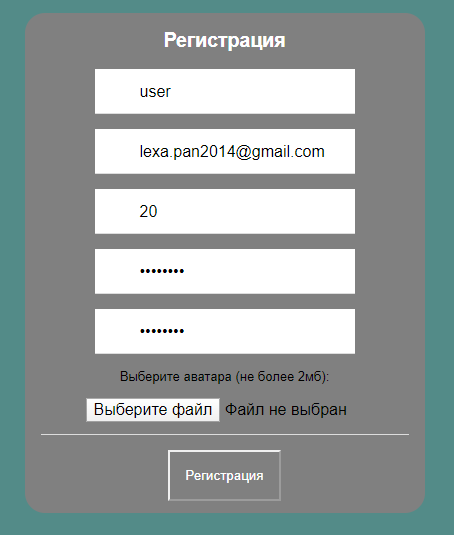


Рис. 5.2 – Реєстрація.

Інструкція з перегляду фільмів:

1. Ви повинні зайти до нашого сайту та перейти до сторінки будь якого фільму з головної сторінки, або використавши пошук.

2. Натискаєте кнопку «Грати» в центрі вікна з фільмом.

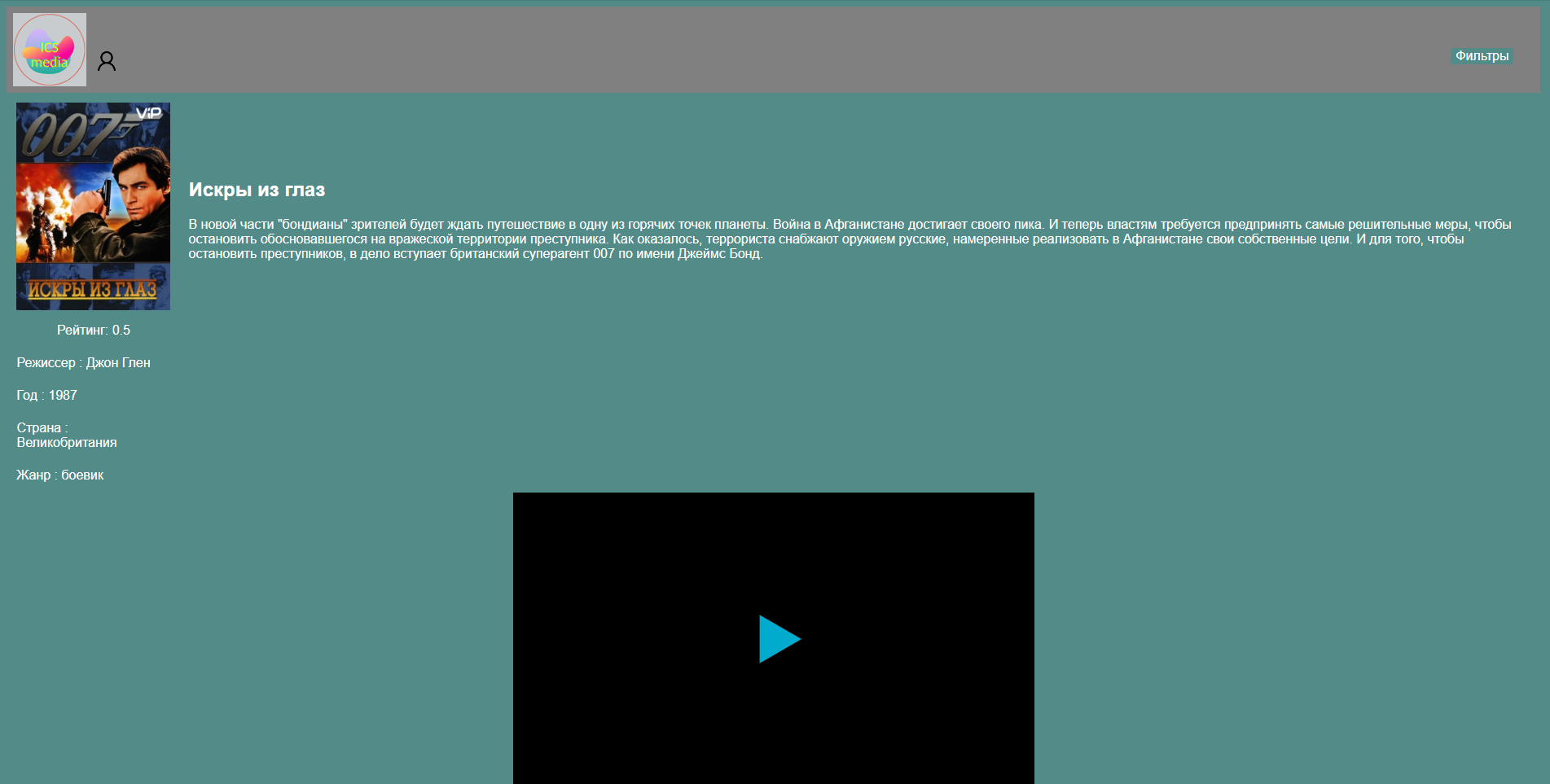


Рис. 5.3 – Перегляд фільму.

Інструкція з пошуку:

1. Ви повинні зайти до нашого сайту та натиснути на кнопку Фільтр яка знаходиться в хедері праворуч.

2. Вводите запит який має назву фільма, та натискаєте кнопку «Пошук».

3. ПП видає плити з фільмами які співпадають з запитом, ви можете вибрати фільм.

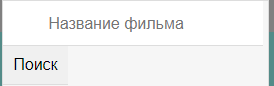


Рис. 5.4 – Пошук.

Інструкція з коментування:

1. Ви повинні знаходитись на головній сторінці після реєстрації.

2. Переходимо на будь-яку сторінку фільму та шукаємо поле для коментарів.

3. Вводимо коментар, та клікаємо кнопку відправити.

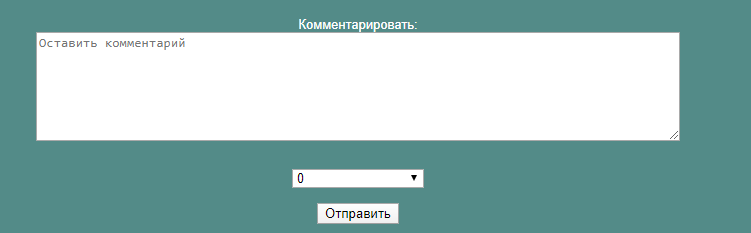


Рис. 5.5 – Коментування.

Інструкція з додавання фільму:

1. Ви повинні знаходитись на сайті після реєстрації.

2. Переходимо до сторінки кабінету, та клікаємо на кнопку «Добавити фільм»

3. Вводимо всі дані які необхідні ПП, та клікаємо кнопку «Відправити».

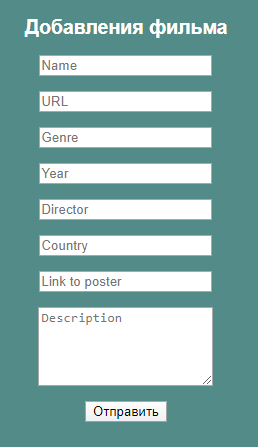


Рис. 5.6 – Додавання фільму.

Інструкція з пожертвування:

1. Ви повинні знаходитись на нашому сайті після реєстрації.

2. Переходимо до будь-якого фільму та шукаємо поле для донату. Вводимо туди значення скільки бажаємо задонатити.

3. Проходимо шаги для відправлення коштів.



Рис. 5.7 – Пожертвування.

Інструкція для видалення фільмів для адмінів:

1. Ви повинні бути на нашому сайті після реєстрації.

2. Переходимо до будь якої сторінки фільму, та клікаємо кнопку видалити.



Рис. 5.8 – Видалення фільму.

## 5.3 Результати валідації програмного продукту

Шаги досягнення мети:

1. Користувач переходить до сторінки нашого сайту.

2. Користувач переходить до кабінету.

3. Користувач переходить до сторінки реєстрації.

4. Користувач проходить реєстрацію.

5. Користувач переходить до сторінки будь-якого фільму.

6. Користувач вводить кількість донату та клікає кнопку «Відправити».

6 шагів досягнення мети.

# Висновок

Наш ПП надає можливість отримувати кошти за свої режисерські творіння.

Заради підтвердження цього можна спробувати зробити це у тестовому режимі.

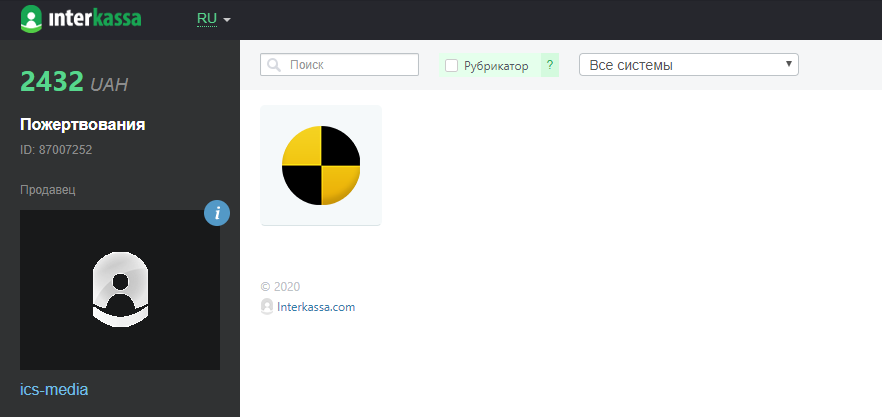


Рис. 6.1 – Тестовий режим донатів.

При проектуванні нашого ПП ми стикнулися з проблемами:

1) обмеженість в часі.

2) складність розробки ПП.

3) недостатні знання де-яких інструментів.

Все перелічені прецеденти були зроблені вчасно.