МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,

управління та адміністрування

Кафедра інформаційних технологій

Кваліфікаційна робота бакалавра

на тему: Розробка веб-ресурсу для підтримки спілкування між студентами за допомогою фреймворку Laravel

Виконала студентка групи К-18
спеціальності 122 «Комп'ютерні науки»
Роменська Анастасия Рсланівна

Керівник доцент, доктор філософії
Бучинська Ірина Вікторівна

Консультант _____

_____

Рецензент професор д.ф.-м.н.
Ковальчук Володимир Володимирович

Одеса 2022

Оглавление

ВСТУП

Сценарій веб-ресурсу представляє собою набір модулів у вигляді файлів з розширенням php, що реалізують можливості обміну інформацією користувачів в мережі. Областю застосування даної дипломної роботи можуть бути завдання забезпечення спілкування, користувачів між комп'ютерами по мережі Інтернет.

Актуальність вивчення даного питання пов'язана з затребуваністю подібних ресурсів в мережі інтернет. Завдяки форумам люди можуть спілкуватися, обговорювати різні питання і проблеми, що незручно робити посредствам коментарів до записів в блогах, соціальних мереж, чатів та гостьових книг. На форумах можна писати більш розгорнуто, вставляти картинки, давати посилання і робити цитати. Цитування поста опонента - улюблений спосіб спілкування на форумах, а цитування частинами, «вирваними» з контексту пропозиціями - один з головних прийомів діалогу між людьми і багато хто знаходить це дуже зручним.

Пости на форумах, що пройшли перевірку модератором за певними критеріями, залишаються там назавжди і стають доступними для пошукових систем, які знаходять їх за запитом користувачів мережі інтернет що робить ці пости корисними не тільки для тих людей які обговорювали подібну тему раніше, але і в подальшому для тих, хто стикається зі схожими питаннями. У соціальних проектах і в програмах обміном миттєвими повідомленнями (skype, icq, viberi ін) подібні функції не реалізовані або присутні частково.

У той час як форум - це якесь «місце», в якому постійно кипить якесь життя: йдуть обговорення тієї чи іншої новини, проекту, it-пристрої і др.і кожна людина може знайти собі однодумців на улюблені йому теми. Ще один важливий аспект - це завоювання аудиторії молодим і талановитим письменником, художником, музикантом, і ін. Для цього більш зручний і підходить блог, де ти головний, а інші читачі. На практиці - на активному форумі тебе буде читати хоч хтось і шансів бути поміченим значно більше, а у не розкрученого блогає велика ймовірність так і залишитися з нульовим коментарієм.

Метою даної роботи є програмна реалізація веб-ресурсу для спілкування студентів за допомогою фреймворку Laravel .

Для досягнення мети необхідно вирішити наступні задачі:

– розглянути існуючі аналоги веб-ресурсів для спілкування;

– провести проектування веб-ресурсу для спілкування;

– реалізувати систему спілкування між користувачами за допомогою фреймвоку Laravel;

– провести тестування веб-ресурсу;

– розробити заходи з охорони праці.

1 ОГЛЯД СИСТЕМ АНАЛОГІВ ЗАСОБІВ ВЕБ СПІЛКУВАННЯ ТА ТЕХНОЛОГІЙ ЇХ РОЗРОБКИ

1.1 Огляд існуючих веб-форумів

Перед початком розробки інтернет сценарію веб-форум був проведений аналіз подібного роду веб-додатків в мережі інтернет і був зроблений висновок що в даний час існує досить багато рішень такого завдання ,зважаючи на актуальність об'єкта досліджень. Але часто такі веб-форуми вимагають оплати послуг за надання доступу до розділів позначаючи їх закритим або працюють із застарілими даними.

У цій частині було протестував 3 веб-форуму.

Перший веб-форум, який я хотів би розглянути це - Форум комп’ютерноїдопомоги[1] (рис. 1.1).

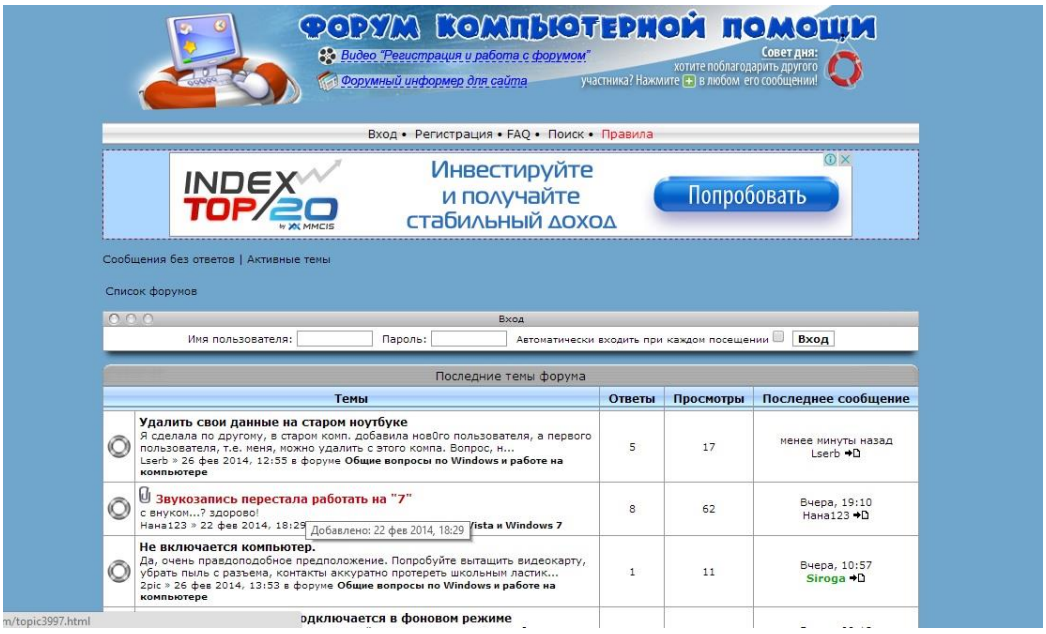

Рисунок 1.1 -Головна сторінка веб-форуму

Перевагами цього форуму є зручна класична навігація по темам розташована в середині екрану, також є посилання «Теми без відповідей» і «Активні теми» натиснувши на які в першому випадку можна побачити список тем, які залишилися без відповіді адміністрації або учасників форуму і в другому випадком будуть відображені теми які користуються найбільшою популярністю у користувачів, так звані «гарячі теми»(рис. 1.2).

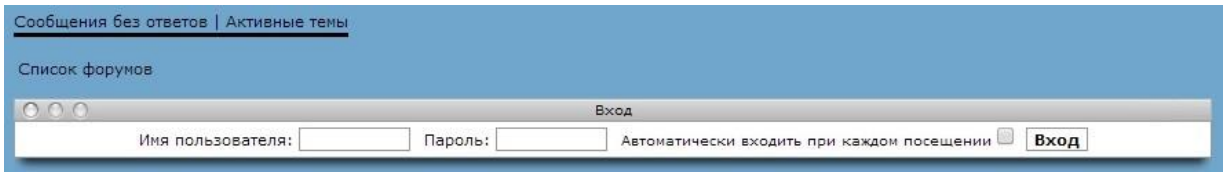

Рисунок 1.2 - Структурна схема програмного продукту

Головним недоліком даного форуму є спливаюча реклама при відвідуванні топіка теми, а також реклама, розташована по обидва боки сторінки, яка відволікає користувача, а також численні кнопки призову перейти на сторінку рекламованого продукту(рис. 1.3).

Рисунок 1.3 - Приклад спливаючої реклами

При введенні свого e-mail адреси в поле на пошту починає приходити велика кількість спам повідомлень що доставляє незручність;

Наступним форумом який я хотів би розглянути є Форум Дизайнр [2] (рис. 1.4).

Рисунок 1.4 - Головна сторінка веб-форуму

При аналізі даного продукту було виявлено що для пошуку по форуму ще потребує детального вивчення інструкції і правильного введення ключових слів використовуючи спеціальні приставки AND щоб визначити слова, які повинні бути в результатах, OR для слів, які можуть бути в результатах.Дані правила є не для всіх зрозумілими що утруднить комфортний пошук по форуму(рис. 1.5).

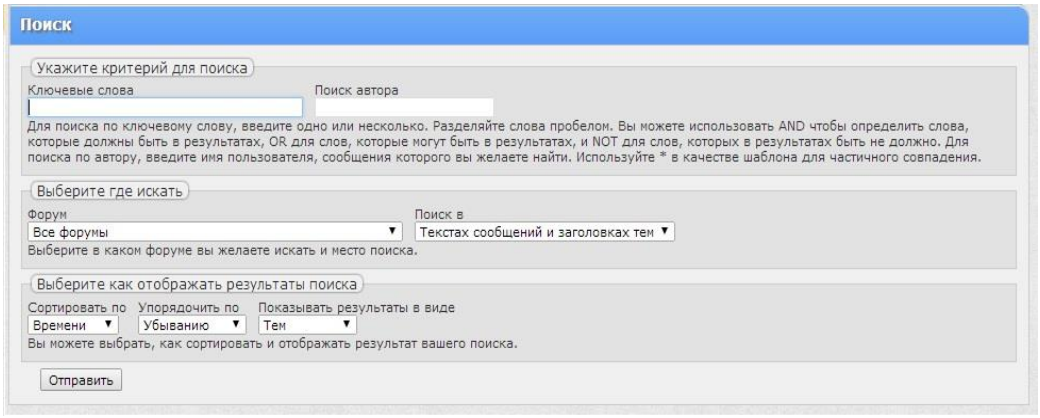

Рисунок 1.5 - Форма пошуку за темами форуму

На даному форумі, як і на попередньому є велика кількість реклами що згубно впливає на комфортне використання сервісу. З плюсів можна також відзначити зручну навігацію, розташовану в центральній частині екрана що дозволяє легко орієнтуватися по форуму.(рис. 1.6).

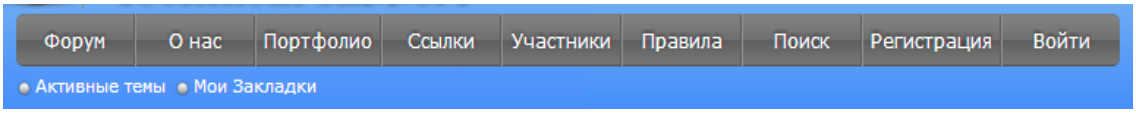

Рисунок 1.6 - Навігаційне меню форуму

Ще однією перевагою є кнопка «Інформація | Налаштування », яка розташовується у верхній частині екрану(рис. 1.7).

Рисунок 1.7 - Приклад кнопки навігації

Її активація розгортає навігаційну панель в якій користувач може авторизуватися або зареєструватися, а учаснику який побажав залишитися під обліковим записом «Гість» відкрито розділ «Діалог з адміністрацією».(рис. 1.8).

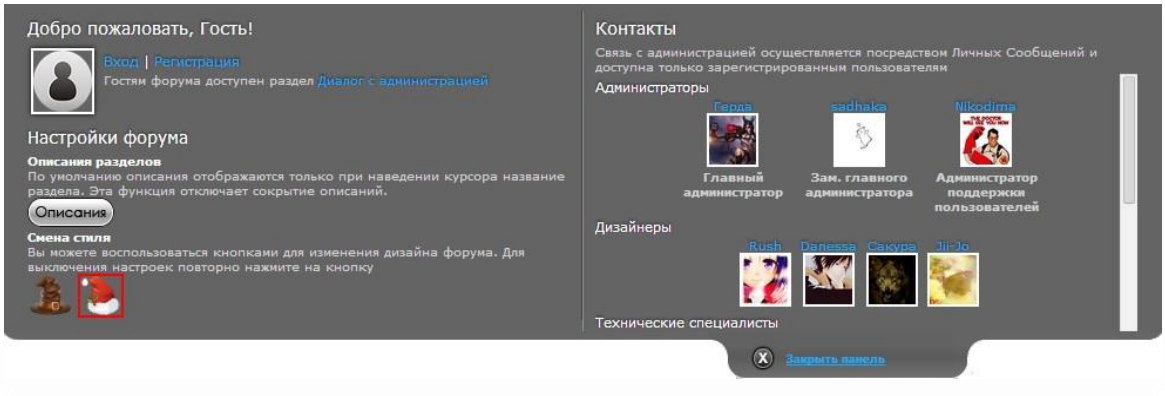

Рисунок 1.8 - Розгорнутий приклад кнопки навігації

Таким чином основними недоліками форумів є: спливаюча реклама та реклама відволікаюча від перегляду, незручна сортування, не завжди інтуїтивно зрозумілий інтерфейс;

При аналізі наступного Одеського форуму було виявлено також велику кількість реклами,яка постійно з’являється на сторінках(рис. 1.9).

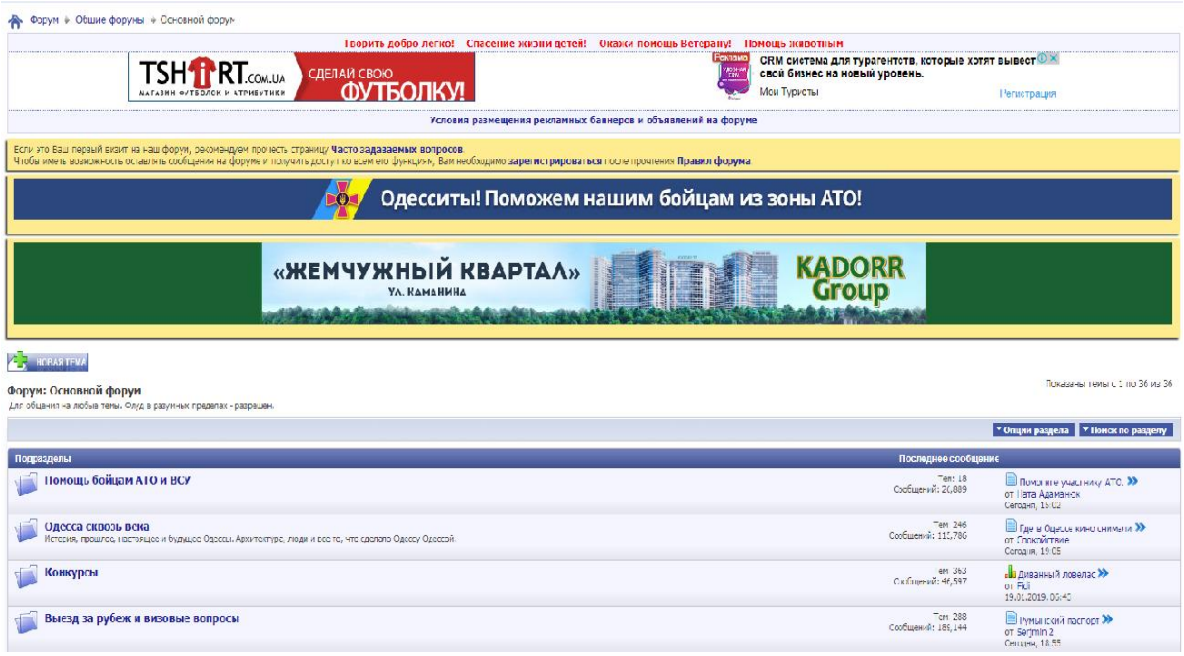

Рисунок 1.9 - Приклад реклами

З плюсів можна відзначити великий вибір підрозділів на форумі, що допомагає швидко орієнтуватися і знаходити потрібну вам тему. Так само є можливість сортувати теми по найпопулярнішим і темам без відповіді(рис. 1.10).

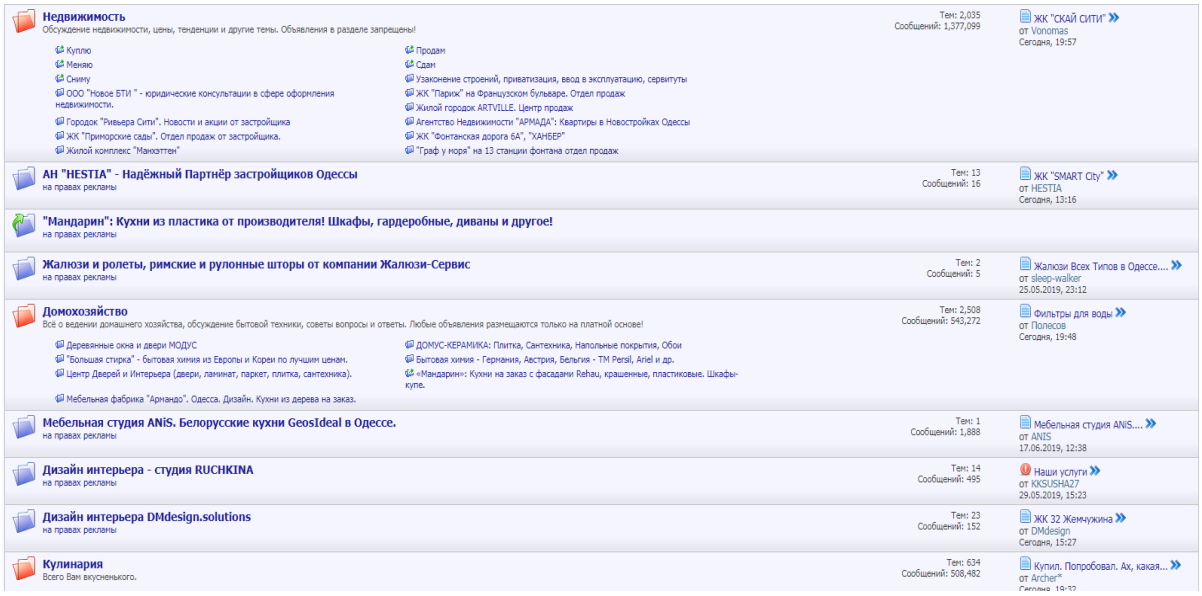

Рисунок 1.10 - Приклад розділів форуму

Що щодо пошуку на форумі, використовується досить таки складна система, яка не зовсім інтуїтивно зрозуміло для звичайного користувача(рис. 1.11).

Рисунок 1.11 - Форма пошуку за темами форуму

Для більш наглядного порівняння створимо таблицю 1.1 в якій відзначимо основні схожості та розбіжності форумів.

Таблиця 1.1 - Результати порівнянь аналогів

Ознаки	Назва форуму			
	Форум Дізайн	Форум комп'ютерної допомоги	Одеський форум	Кваліфікаційна робота
Зручна навігация за темами	ні	так	так	так
Інтуітивно зрузумілий інтерфейс	ні	так	так	так
Наявність реклами	так	так	ні	ні
Зручний пошук тем на форумі	ні	ні	так	так
Можливість відсортувати теми за категоріями	ні	ні	ні	так
Можливість побачити популярні теми	ні	ні	так	так
Можливість побачити теми без відповіді	ні	ні	так	так
Адаптивний дизайн	ні	ні	ні	так

1.2 Засоби розробки

1.2.1 Вибір фреймворку

Фреймворк - це програмне забезпечення, що полегшує розробку і об'єднання різних компонентів великого програмного проекту. На відміну від бібліотеки функцій фреймворк накладає обмеження на структуру і логіку програмного продукту [4].

Веб-фреймворк призначений для створення динамічних веб-сайтів, мережевих додатків, сервісів і ресурсів. Веб-фреймворки містять логіку обробки HTTP-запитів, спрощують доступ до баз даних і інше[5].

Фреймворк є надбудовою над мовою програмування и дозволяє конструюваті програми з сторонніх модулів, легко їх розшірюваті и модіфікуваті. С одного боку, фреймворк вводить обмеження на структуру файлів, стиль оформлення коду, правила з розділення логіки і має функції, які можуть зовсім не використовуватися в готовому рішенні. З іншого боку, фреймворки скорочують час проектування і розробки додатків, виключають дублювання коду, а також спрощують супровід проектів. Крім того, фреймворки супроводжуються документацією, навколо фреймворків утворюються співтовариства, що містять приклади і базу знань[6].

При виборі фреймворка для розробки необхідно враховувати патерни, які використовує фреймворк, наявність документації та розмір спільноти навколо фреймворка, а також можливість розширення функціоналу проектованої системи за рахунок власних і сторонніх розширень[7].

Для порівняння були обрані PHP-фреймворки Yii 2 [8] - як найпопулярніший фреймворк в Україні і Laravel [9] - як найпопулярніші фреймворки в світі. [10] У таблиці 1.2 представлено порівняння PHP-фреймворків Yii 2 і Laravel.

Порівняння PHP

Критерії -фреймворків	Фреймворк	
	Yii 2	Laravel
Вимоги	PHP 5.4.0 і вище.	PHP 5.5.9 і вище. Розширення PHP: - OpenSSL; - PDO; - Mbstring; - Tokenizer.
Паттерн	MVC	MVC
Розширення	Підтримуються (composer)	Підтримуються (composer)
Міграції	Підтримуються	Підтримуються ; Інструмент наповнення даними.
ORM (об'єктно реляційне відображення)	Active Record	Eloquent ORM (Active Record)
Валідація форм	Підтримуються	Підтримуються
Локалізация	Підтримуються	Підтримуються
Англомовне співтовариство	Документація, форуми, блоги	Документація, відео-уроки (Laracast), форуми, блоги
Україномовне співтовариство	Документація, форуми, блоги	Документація, форуми, блоги

Завдяки зручності, розвиненому спільноті, а також високу популярність для розробки інформаційної системи був обраний PHP - фреймворк Laravel.

1.2.2 PHP-фреймворк Laravel

Веб-додаток інформаційної системи розробляється з використанням PHP-фреймворку Laravel версії 5.4 з відкритим вихідним кодом.

За результатами опитування розробників в 2018 році Laravel посів перше місце за популярністю в номінаціях «Фреймворк корпоративного рівня» і«Фреймворк для особистих проектів».[10] Також репозиторій Laravel є найпопулярнішим серед PHP репозиторіїв GitHub, що ще раз підтверджує

інтерес розробників до даного фреймворку.[11]

Laravel використовує архітектурний шаблон Model-View-Controller (MVC),

який є дуже популярним в веб-розробці і дозволяє відокремити бізнес-логіку від її подання.

Концепція MVC дозволяє розділити дані, подання та обробку дій користувачів на три компоненти[12]:

а) модель - надає знання: дані та методи роботи з цими даними, реагує на запити, змінюючи свій стан. Модель забезпечує дотримання всіх бізнес-правил, які застосовуються до даних;

б) подання - відповідає за візуальне представлення моделі, як правило являє собою HTML-розмітку в браузері;

в) контролер - забезпечує зв'язок між поданням і моделлю: контролює введення даних користувачем і використовує модель і уявлення для реалізації необхідної реакції.

На (рис. 1.12) представлена архітектура додатки, розробленого з використанням фреймворку Laravel[13].

При взаємодії з веб-додатком браузер відправляє запит, який приймається веб-сервером і передається в компонент Laravel, що відповідає за маршрутизацію.

Маршрутизатор Laravel приймає запит і на основі шаблону URL-маршрутизації перенаправляє його для обробки в відповідний метод класу контролера.

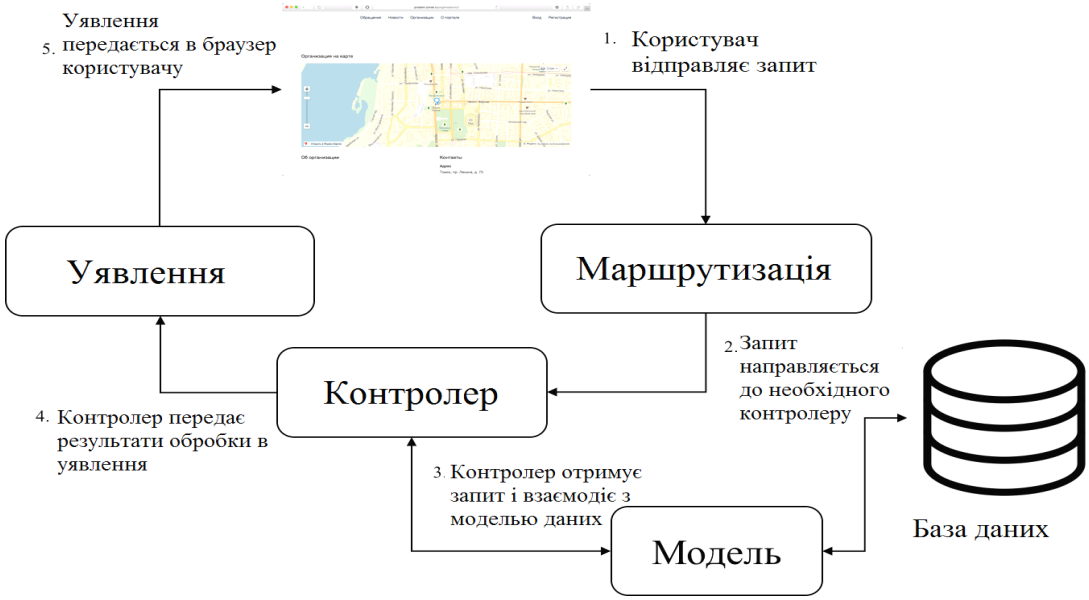

Рисунок 1.12 – Архітектура програми Laravel

Далі обробкою запиту займається контролер, який, як правило, взаємодіє з моделлю, генерує відповідь у вигляді подання (HTML, CSS і зображень) і повертає веб-сторінку в браузер користувача.

1.2.3Веб-серверApache

Apache - це багатоплатформність веб-сервера з відкритим вихідним кодом, що підтримує протокол HTTP[14].

Основними достоїнствами Apache вважаються надійність і гнучкість конфігурації. Веб-сервер має власну мову конфігураційних файлів, заснований на блоках директив. Практично всі параметри ядра сервера можуть бути змінені через конфігураційні файли[15]. Apache має модульну архітектуру, що дозволяє розширювати його функціональність шляхом підключення додаткових модулів. Сервер Apache підтримує одночасну роботу, що дозволяє обслуговувати велику кількість клієнтів. Кількість клієнтів, що може обслуговуватися одночасно, обмежується лише апаратними ресурсами і операційною системою [16].

За даними організації Netcraft найбільш популярним веб-сервером (серед активних сайтів) в світі є Apache з часткою 49,2% за станом на березень 2018 року.

1.2.4 Система управління базами даних MySQL

База даних (БД) - представлена в об'єктної формі сукупність самостійних матеріалів (статей, розрахунків, нормативних актів, судових рішень та інших подібних матеріалів), систематизованих таким чином, щоб ці матеріали могли бути знайдені і оброблені за допомогою електронної обчислювальної машини (ЕОМ) [17].

База даних є інформаційною моделлю предметної області. Звернення до баз даних здійснюється за допомогою системи управління базами даних (СКБД). СУБД забезпечує підтримку створення баз даних, централізованого управління та організації доступу до них різних користувачів[18]. В рамках роботи використовується СУБД MySQL.

MySQL - вільна система керування базами даних. В даний час розробку і підтримку MySQL здійснює корпорація Oracle. Дана СУБД є рішенням для малих і середніх додатків. До основних плюсів MySQL можна віднести високу швидкість роботи, швидкість обробки даних і оптимальну надійність. Важливим фактором є те, що дана СУБД поширюється безкоштовно і представляє собою програмне забезпечення з відкритим кодом.

MySQL займає другий рядок у світовому рейтингу популярності систем управління базами даних за даними на травень 2018року.[19]

1.2.5 Локальний веб-сервер OpenServer

В процесі розробки інформаційної системи необхідно регулярно проводити функціональне тестування - перевіряти працездатність функцій системи. Функціональне тестування зручно проводити на локальному веб-сервері, розташованому на комп'ютері, де ведеться розробка інформаційної системи.

Використання локального веб-серверу не накладає обмежень ні на розмір серверної і веб-частин інформаційної системи, ні на використання процесорного часу або оперативної пам'яті сервера, так як сервером є комп'ютер, на якому здійснюється розробка. Ще одна перевага використання локального веб-сервера - то, що результат роботи відразу ж видно в браузері, так як немає необхідності завантажувати файли інформаційної системи на віддалений сервер.

Щоб уникнути ручного регулювання і настройки веб- сервера Apache і MySQL існує додаток OpenServer, що включає в себе веб-сервер Apache, СУБД MySQL, інтерпретатор PHP і інші компоненти, необхідні для локальної веб-розробки. OpenServer поставляється в двох версіях: безоплатній та платній (професійної). Для функціонального тестування інформаційної системи досить безкоштовної версії OpenServer, так як вона містить всі необхідні для роботи компоненти (веб-сервер Apache, СУБД MySQL і PHP).

1.2.6 Інтегроване середовище розробки PhpStorm

Інтегровані середовища розробки (ICP) - комплекс програмних засобів, який використовується програмістами для розробки програмного забезпечення.[20]

Як правило, середовище розробки включає:

– текстовий редактор;

– компілятор і / або інтерпретатор;

– засоби автоматизації збірки;

– відладчик.

Інтегровані середовища розробки спрямовані на максимізацію продуктивності програміста завдяки тісно пов'язаним компонентів з простими для користувача інтерфейсами. Це дозволяє розробнику здійснювати менше дій для доступу до різних режимів. Зазвичай ICP орієнтовані на певний мову програмування, надаючи набір функцій, який найближче відповідає парадигм цієї мови програмування. Як правило, ICP є єдину програму, в якій проводиться вся розробка.[21]

PhpStrom - інтегроване середовище розробки для програмістів PHP.

Можливості PhpStorm:

– редактор PHP, HTML, CSS і JavaScript коду з підсвічуванням синтаксису;

– навігація по коду і пошук використань;

– автодоповнення PHP, HTML, CSS і JavaScript коду;

– підтримка утиліт командного рядка і SSH консолі.

– розширення функціональності середовища розробки за рахунок установки плагінів;

– крос-платформенность (Mac OS X, Windows, Linux);

– інструменти роботи з базами даних, SQL-редактор;

– інтеграція з системами управління версіями;

–віддалене розгортання додатків і автоматична синхронізація з використанням FTP, SFTP і так далі.

PhpStrom розробляється і підтримується компанією JetBrains. Учні та викладачі навчальних закладів можуть вільно користуватися продуктами JetBrains (в тому числі PhpStorm) протягом року.

У PhpStorm є плагін TheLaravelIDEHelper, який полегшує розробку додатків Laravel, додаючи підказки при написанні коду.

2 ПРОЕКТУВАННЯ ВЕБ-РЕСУРСУ ДЛЯ СПІЛКУВАННЯ СТУДЕНТІВ

2.1 Мета та задачі веб-ресурсу

Розробляємий веб-ресурс - це клієнт-серверне застосовання, основною функцією якого є надання користувачу інформації за необхідними темами та можливість створювати власні теми для подальшої отримки інформації.

Мета ІС: веб-ресурс повинен зберігати інформацію та також надавати зручний інтерфейс для подальшої роботи з нею, надавати можливість користувачу отримувати або обговорювати цікаву для нього інформацію з іншими користувачами в мережі.

Цільова аудиторія: інформаційна система призначена для підтримки спілкування студентів з потрібних їм тем в мережі інтернету. Засобом створення і відправки тем є інтернет-портал.

Головними функціями застосовання є:

– перехід на сторінку сайту;

– отримання необхідної інформації;

– створення потрібної теми.

Опис вхідних та вихідних інформаційних потоків продукту (рис. 2.1).

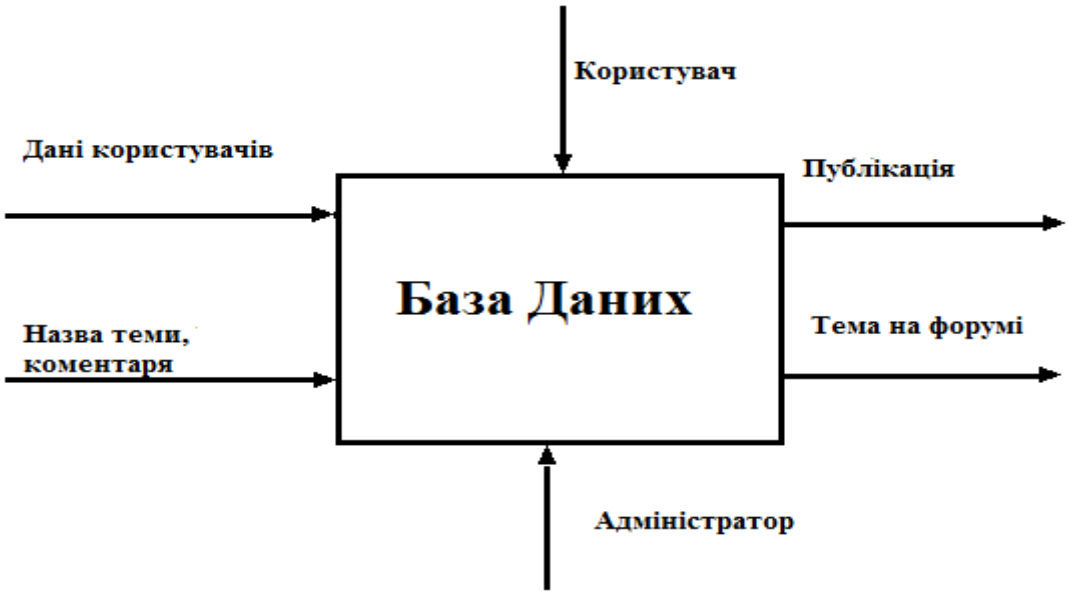

Рисунок 2.1 - Опис вхідних та вихідних інформаційних потоків форуму

Користувач може створювати пости, призначається час і дата їх публікації в певних темах. Результатом роботи цього додатка є опубліковані теми, пости.

Адміністратор має можливість видаляти всі теми на форумі.

2.2 Управління вимогами до дипломної програми

2.2.1 Типи користувачів

В таблице 2.1 описаны все виды пользователей возможные в данном веб-ресурсе.

Таблиця 2.1- Опис ролей

Актор	Короткий опис
Користувач	Користувач, що володіє базовими правами доступу
Адміністратор	Користувач, що володіє повними правами доступу
Гість	Користувач, що володіє обмеженими правами доступу

Користувач: здатний створювати свої запити, видаляти і зберігати.

Гість: здатний переглянути головну сторінку і зареєструватися.

Адміністратор: здатний управляти користувачами і їх запитами.

2.2.2Функціональні вимоги

1. Реєстрація та авторизація користувача

F1.1 Реєстрація

F1.1.1 Перехід на головну сторінку неавторизованих користувачів

F1.1.2 Введення даних (email і пароль)

F1.1.3 Натискання на кнопку "Зареєструватися"

F1.1.4 Перевірка на наявність користувачів з там самим emailадресою

Функція доступна для користувача Гість.

F1.2 Авторизація

F1.2.1 Перехід на головну сторінку неавторизованим

F1.2.2 Натискання на кнопку "Увійти"

F1.2.3 Введення даних (email і пароль)

F1.2.4 Перевірка введенних даних(все поля повинні бути заповненні)

Функція доступна для Користувача.

2. Управління темами

F2.1 Додавання теми

F2.1.1 Перехід на сторінку створення теми

F2.1.2 Введення данних

F1.2.3 Перевірка введенних даних(все поля повинні бути заповненні)

F2.1.4 Натискання кнопки зберегти тему

F2.2 Видалення теми

F2.2.1 Перехід на сторінку з темою

F2.2.2 Выбор необходимой темы

F2.2.3 Натискання кнопки видалення теми

F2.2.4 Натискання кнопки підтвердження видалення теми

F2.3 Сортування тем

F2.3.1 Перехід на сторінку з популярними темами

F2.3.2 Перехід на сторінку з темами без відповіді

F2.3.3 Пошук тем за категоріями

F2.4Пошук тем за назвою

F2.4.1 Введення назви теми

F2.4.2 Натискання кнопки пошук

F2.4.3 Отримання необхідної теми

F3. Управління коментарями

F3.1 Додавання коментаря

F3.1.1 Перехід на сторінку теми

F3.1.2 Введення коментаря

F3.1.3 Збереження коментаря

F3.2 Редагування коментаря

F3.2.1 Перехід на сторінку з коментарем

F3.2.2 Натискання кнопки редагування

F3.2.3 Збереження коментаря

F3.3 Видалення коментаря

F3.3.1 Перехід на сторінку з коментарем

F3.3.2 Натискання кнопки видалення коментаря

F3.4Оцінка коментаря

F3.4.1 Користувач має можливість оцінити коментар

F4 Профайл пользователя

F4.1 Перехід на сторінку з про файлом

F4.2 Можливість завантажити фото про файлу

F4.3 Перевірка завантажуємого файлу(має бути формату png,jpeg, і т.д.)

2.2.3Вимоги до безпеки форуму

Першим кроком на шляху до безпеки форуму було введення реєстрації. Також необхідно домогтися неможливості отримання користувачами доступу до внутрішніх сторінок форуму з інших доменів і захист від "DoS" атак. Надання користувачам обмежених прав так само сприяє високій захищеності форуму. Збереження паролів користувача повинна забезпечуватися одностороннім шифруванням.

2.3 Моделювання прецедентів

Для детальнішого розгляду процесу є можливість побудови додактові, більш детальніші диаграми використання. Наприклад, побудуємо діаграму прецедентів для блоку отримання інформації з сайту (рис. 2.2).

Ця діаграма надасть нам більше розуміння про можливі варіанті поведінки користувача у обраному сценарії. Аналогічні діаграми можут бути запрошено для будь-якого процесу.

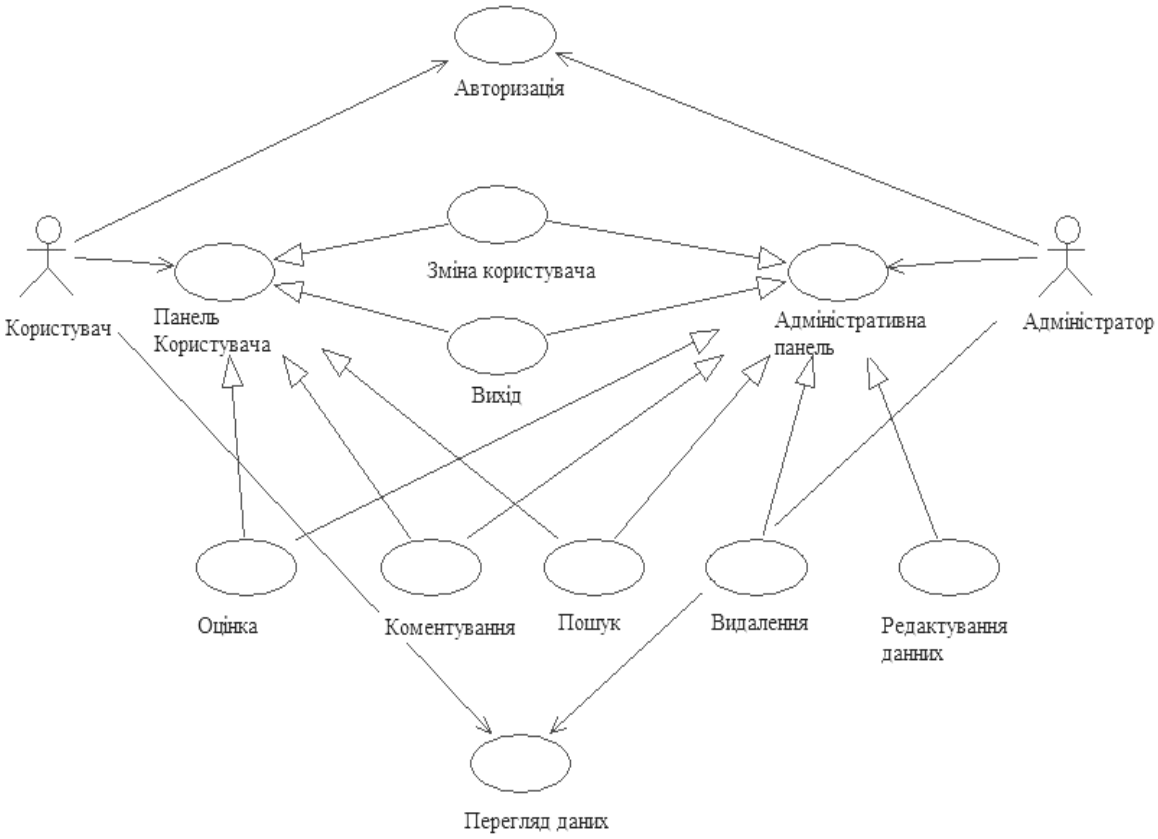

Рисунок 2.2-Діаграма прецедентів

У таблицях 2.2 описані прецеденти, які використовуються в діаграмі варіантів використання.

Перегляд - користувач або адміністратор може переглядати будь-які дані на форумі. Це варіант використання виводить інформацію на екран.

Оцінка- користувач або адміністратор може оцінювати статті в інформаційній системі. Цей варіант використання оцінює статтю.

Таблиця 2.2 - Опис прецедентів

Прецедент	Короткий опис
1	2
Авторизація	Надання прав для роботи в системі
Панель Користувача	Вхід під обліковим записом користувача, вихід
Панель Адміністратора	Вхід під обліковим записом адміністратора, вихід
Зміна користувача	Перехід у вікно авторизації з логіном
Вихід	Перехід на початкову сторінку
Оцінка	Оцінка статті
Коментування	Коментарі або поправки вмісту
Пошук	Пошук вміступо форуму
Редагування даних	Редагування тем та їх змісту
Видалення	Видалення даних
Перегляд даних	Користувачі і Гості можуть переглядати будь-які дані

Пошук - користувач або адміністратор може шукати інформацію на форумі по імені або фрагменту тексту статті. Цей варіант використання шукає інформацію за ключовими словами або фразами в ІС.

Редагування даних - адміністратор форуму може вносити зміни в дані, а саме в статті і пости. Даний варіант використання вносить зміни в записи ІС через запити із зазначенням id статті і поста.

Видалення даних - адміністратор форуму може видаляти дані з інформаційної системи. Даний варіант використання видаляє дані за допомогою запиту на видалення із зазначенням id статті і поста.

Авторизація - адміністратор і користувач для роботи в інформаційній системі виробляють авторизацію для отримання прав на внесення або зміну інформації. Даний варіант використання виробляє авторизацію користувачів за логіном і паролем. Після верифікації даних варіант використання дозволяє або не дозволяє працювати в системі.

Коментування - користувач і адміністратор можуть коментувати ті чи інші дані на форумі. Даний варіант використання записує в інформаційну базу текст для статті в таблицю з коментарями.

2.4 Нефункціональні вимоги

1.1 Дизайн і розмітка (UI)

NF1.1 Веб-додаток повинен являти собою багатосторінковий сайт

NF1.2 Веб-сервіс повинен використовувати сучасні технології HTML5 та CSS3, при цьому зберігаючи сумісність з браузерами як:

– IE 11;

– Microsoft Edge (Версії 16 та вище);

– Mozilla Firefox (Версії 59 та вище);

– Google Chrome (Версії 64 та вище);

– Opera (Версії 51 та вище).

2. Взаємодія з користувачем

NF2.1 Час відгуку сторінки повинно бути не більше 4 секунд

NF2.2 Інтерфейс повинен бути зручним, інтуїтивно зрозумілим.

3. Требования к хостингу приложения

NF3.1 API сервер додатки Algolia повинен мати прийнятну продуктивність на наступній конфігурації заліза: 64-бітна архітектура, 2 або більше ядер, 4 або більше гігабайт оперативної пам'яті, 6 або більше гігабайт місця на накопичувачі.

NF3.2 Frontend сервер додатка повинен мати прийнятну продуктивність на наступній конфігурації заліза: 64-бітна архітектура, 2 або більше ядер, 2 або більше гігабайт оперативної пам'яті, 1 або більше гігабайт місця на накопичувачі.

NF3. 3 Сервера повинні бути запущені наWindowsOC 7 - 10, Linux.

2.5 Ідентифікація архетипу веб-ресурсу

Веб-додаток - клієнт-серверний додаток, створене за допомогою фремворка Laravel в якому клієнт взаємодіє з сервером за допомогою браузера, а за сервер відповідає веб-сервер.

Логіка веб-додатки розподілена між сервером і клієнтом, зберігання даних здійснюється, переважно, на сервері, обмін інформацією відбувається по мережі. Одним з переваг такого підходу є той факт, що клієнти не залежать від конкретної операційної системи користувача, тому веб-додатки є межплатформенному службами.

2.6 Обмеження архітектури програми

Замовник джерело даних в БД.

Замовник использовать фремворк Laravel.

Веб-ресурс є джерелом користувацького інтерфейсу.

Веб-ресурс зберігає серверні дані для обмена информации.

2.7 Подання користувацького інтерфейсу

Інтерфейс користувача – система правил і засобів, що регламентує і забезпечує взаємодію програми з користувачем.

Засобами є пристрої введення та виведення, команди, меню, кнопки і так далі. Правила визначають семантику використання перерахованих коштів. Як правило, сучасні програми підтримують роботу зі стандартними пристроями введення і виведення (клавіатура, миша, дисплей), тому в рамках призначеного для користувача інтерфейсу їх не згадують явно, а розглядають дизайн і розташування вікон, меню і елементів управління програми або системи.

Для інтерфейсу, особливо візуального, існує ряд загальноприйнятих вимог: дружність, передбачуваність, простота, привабливість, цілісність.

Дружність означає те, що інтерфейс повинен враховувати психологічні і фізіологічні особливості людини для забезпечення максимально можливого комфорту і ефективності вирішення завдань, а не підлаштовуватися під особливості програми або пристрою.

Передбачуваність дозволяє користувачеві ефективно застосовувати накопичений досвід (досвід реальному житті і досвід роботи з іншими системами або програмами). Інтуїтивно зрозумілі інтерфейси дозволяють користувачеві з меншими зусиллями «здогадуватися» про те, як вирішити ту чи іншу задачу, без необхідності спеціального навчання або читання документації.

Простота призначеного для користувача інтерфейсу означає те, що інтерфейс повинен бути максимально простим, тобто управління має вимагати мінімальної кількості маніпуляцій.

Привабливість (естетичність) означаєте, що розташування елементів інтерфейсу має бути гармонійним (впорядкованим, беззайвої порожнечі і тісноти), колірнагамма - привабливою, аленедратує,зображення - красивими.

Цілісність означає те, що призначений для користувача інтерфейс повинен бути виконаний в єдиному стилі із застосуванням одних і тих же принципів у всіх частинах системи. У проектованої системі повинен дотримуватися єдиний стиль іменування елементів і їх взаємного розташування. Всі значки, іконки і написи повинні бути виконані в єдиному стилі і в єдиній кольоровій гамі.

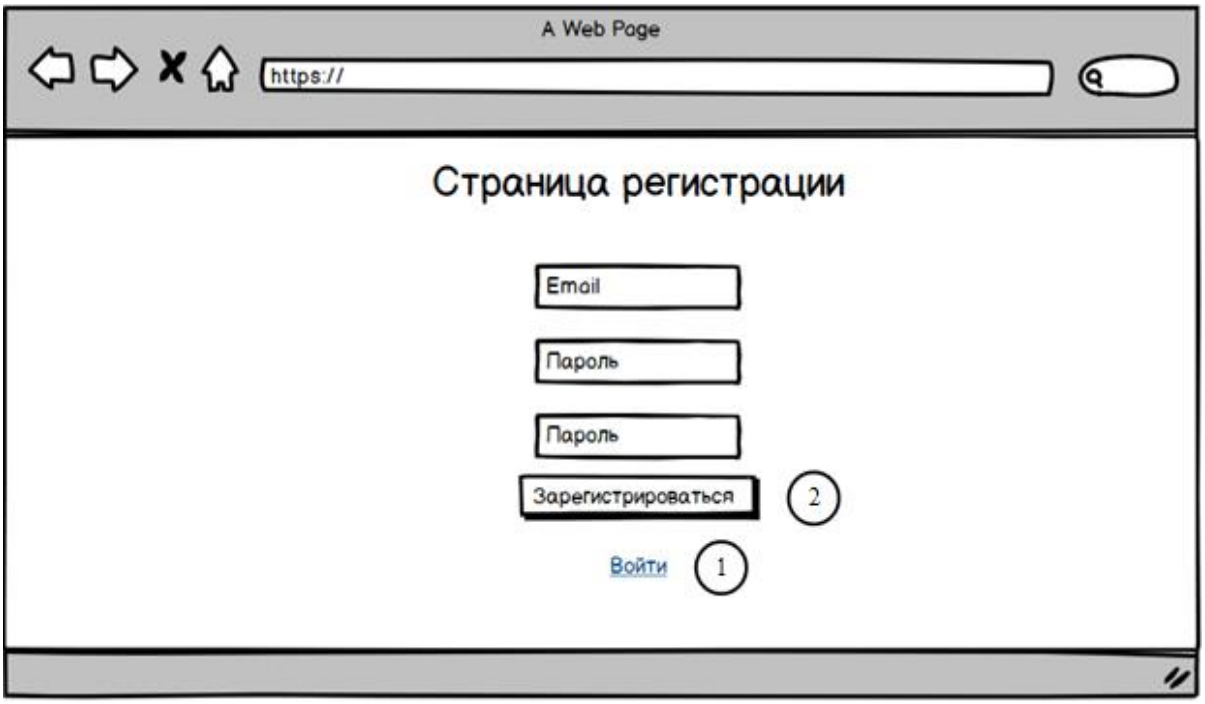

Рисунок 2.3 - Мокап реєстрації

Вікно авторизації (Рисунок 2.4) - користувач може авторизуватися на форумі.

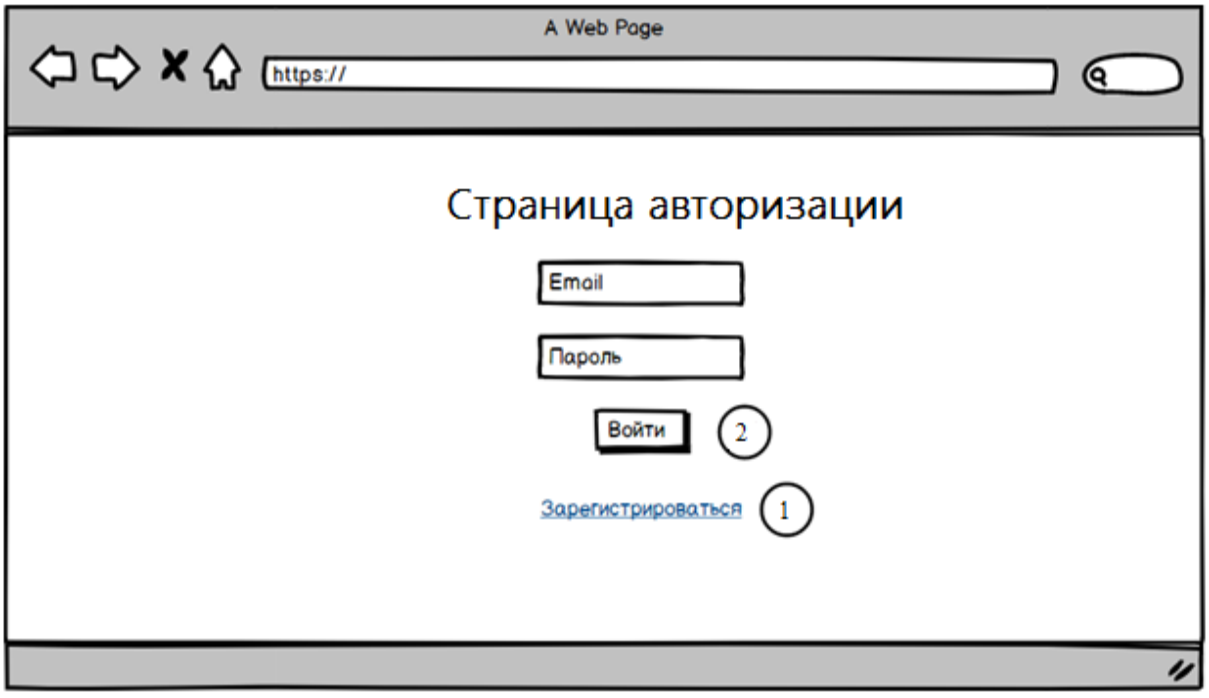

Рисунок 2.4 – Мокапавторизації

Веб сторінка являє собою форму для заповнення даних. Гість вводить свої дані і входить на сайт вже в статусі Користувач.

Вікно перегляду тем (рис. 2.5) - користувач може переглянути всі теми на форумі.

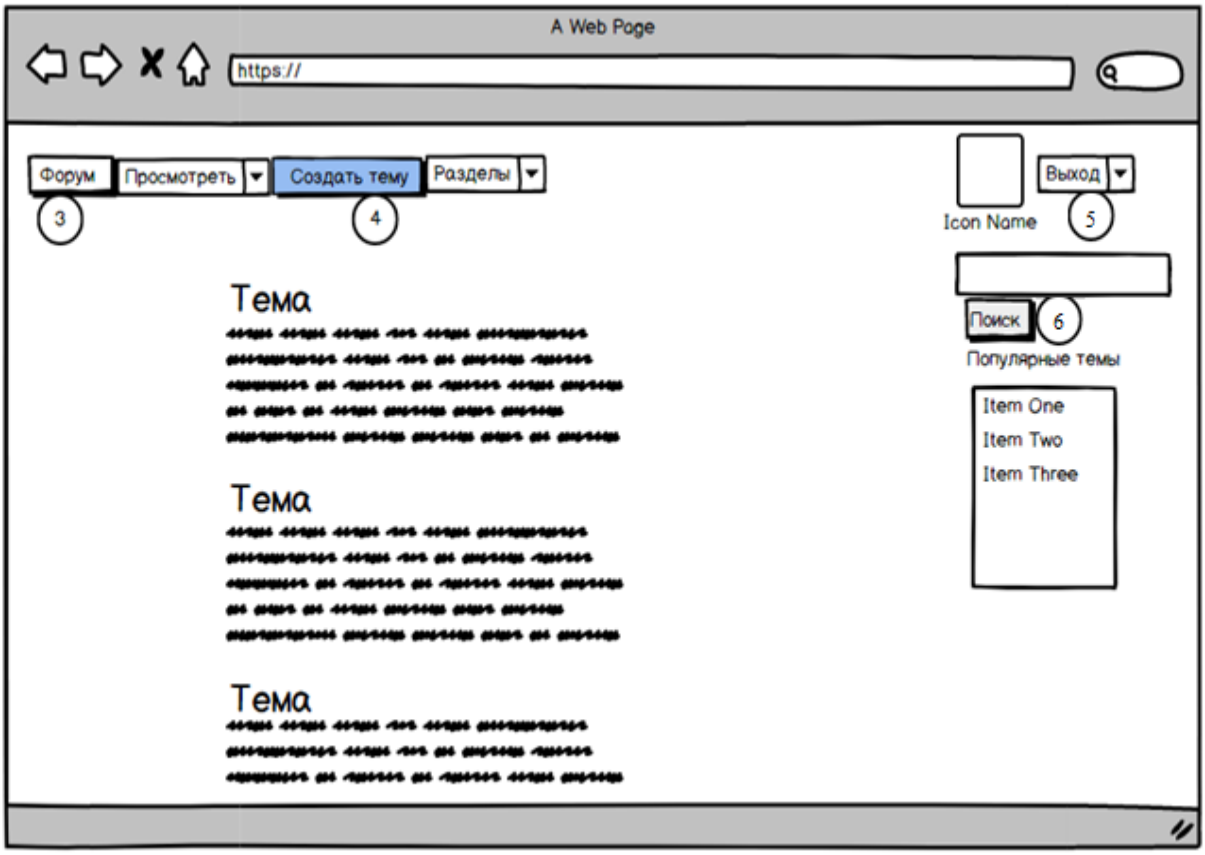

Рисунок 2.5– Мокапголовної веб-сторінки

У верхній частині web-сторінки знаходиться меню web-сайту, тобто гіперпосілання натіснувші на які ми можемо перейти на інші сторінки сайту, а також іконка користувача, при натісканні на неї користувач перейшовши на свій профайл.

Під меню знаходиться основний контент, відображаються всі тими форумом з можлівістю переходу на конкретно цікавить користувача тему. У правій части веб Сторінки находится поиск по сайту и відображення популярних тем на форумі.

Вікно створення теми (рис. 2.6) - користувач може створювати теминафорумі.

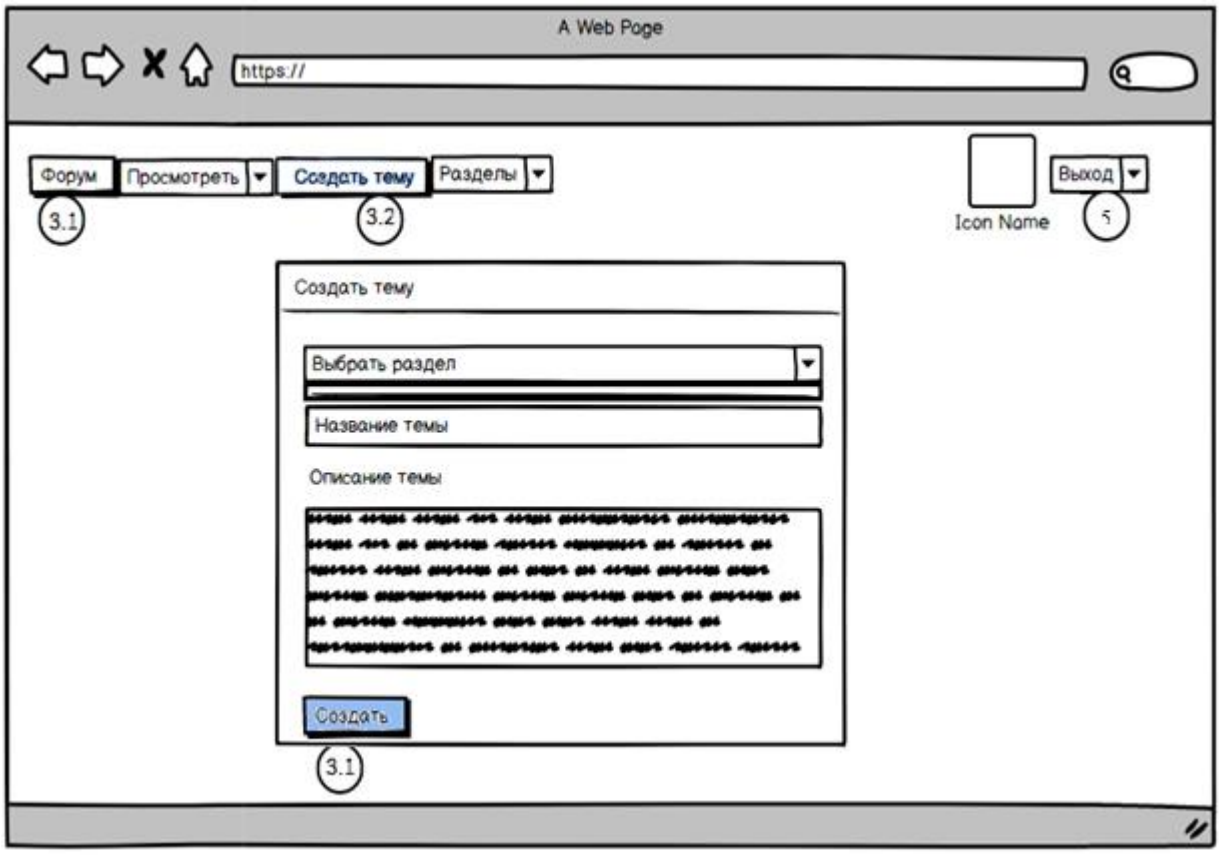

Рисунок 2.6 – Веб-сторінка створення теми

Веб сторінка престовляет собою форму для Заповнення Даних. Авторизований користувач вібірає розділ в якому хоче розмістіті свою тему, Назву и ее описание, после чего натіскає на кнопу создать и переходити на сторінку своєї створеної тими.

2.8 Логічне уявлення IC (Logical View)

Для створення абстрактної уяви про майбутню систему, логічне уявлення системи, що описує взаємодію системи з користувачами і підмодулями.

Інформаційна система має трирівневу архітектуру.

Трирівнева архітектура - архітектурна модель програмного комплексу, що передбачає наявності в ньому трьох компонентів: клієнта, сервера додатків і сервера баз даних. На (рис2.8) представлена схема трирівневої архітектури[22].

Рівень представлення - верхній рівень, відповідає за відображення інформації, наданою кінцевому користувачу. Рівень містить найпростішу бізнес-логіку: перевірку вводятьсязначень на допустимість і відповідність формату, а також нескладні операції з уже завантаженими даними.

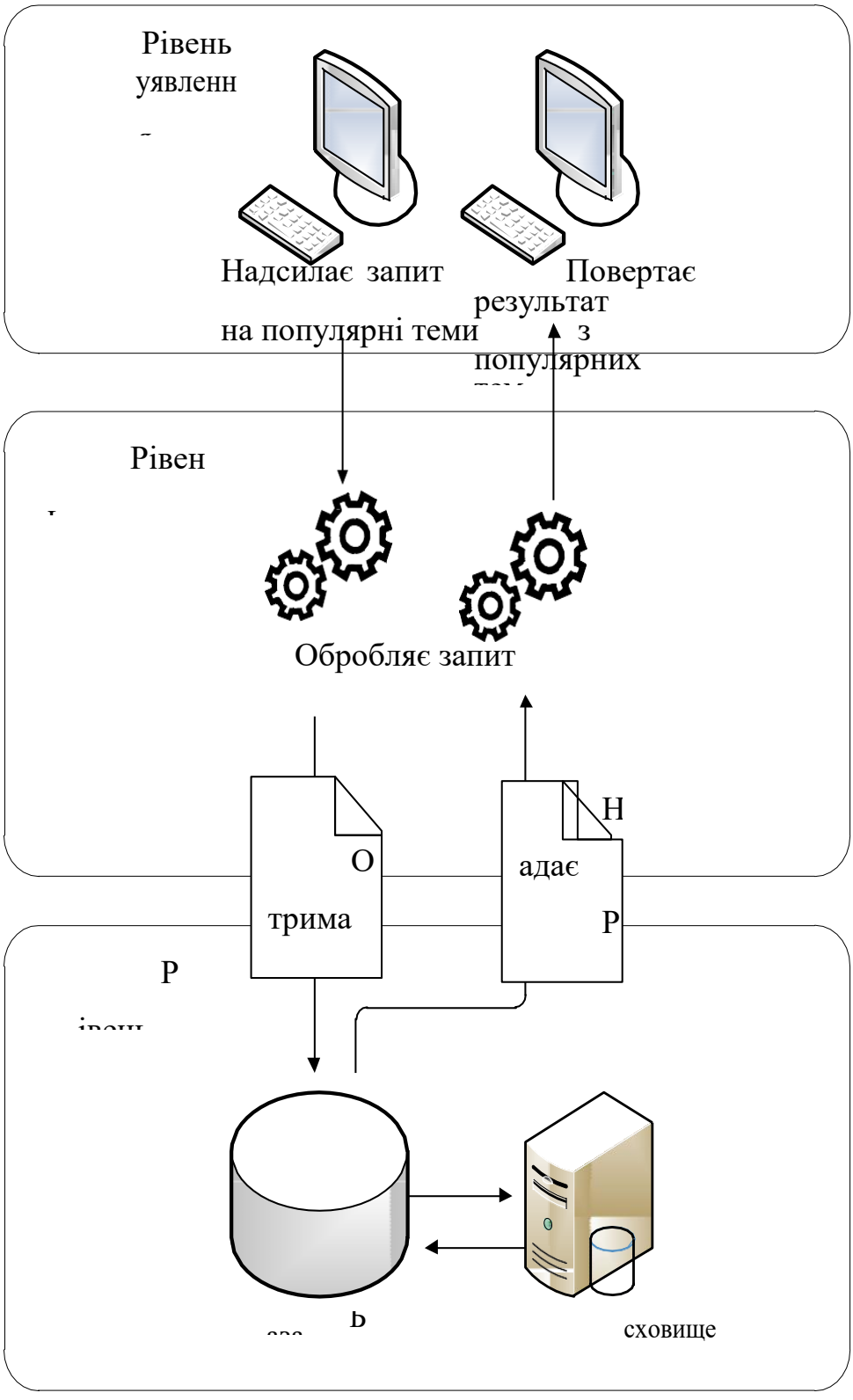

Рисунок 2.7 – Схема логіки роботи системи та зв’язку компонентів

Рівень додатків - середній рівень, містить основну частину бізнес-логіки,містить моделі класів.

Рівень даних - забезпечує зберігання даних, реалізується, як правило, засобами систем управління базами даних .

Трирівнева архітектура володіє високою гнучкістю і відкритістю, підвищує надійність і захищеність системи, її переносимість, масштабованість і відмовостійкість. Недоліками трирівневої архітектури є підвищена складність створення додатків, а також складність розгортання і адміністрування.

На цій діаграмі можливо побачити взаємодію користувача із веб-ресурсом. Користувач робить запит на отримання популярних тем на форумі. Програма обробляє запит, отримує список тем на форумі з бази даних і сортує їх за популярністю, після чого виводить результат запиту користувачеві на екрані,де він користується.

В рамках даної роботи рівень даних представлений системою управління базами даних MySQL, рівень додатків - PHP-фреймворком Laravel, рівень представлення - будь-яким сучасним браузером.

2.9 Уявлення розгортання IC (DeploymentView)

Веб-ресурс працює завдяки веб-браузеру, мережі інтернет та серверу(рис. 2.8).

Клієнт потрапляє до веб-браузеру, посилає запит, веб-ресурс створює об'єкт Application і визначає середовище виконання, перевіряються внутрішні файли, конфіги і реєструється сервіс-провайдери. Після цього йде перевірка запиту через маршрути в (Routes) і передається обробка запиту відповідному контролеру, об'єкт Request вирушає в Application, контролер здійснює деякі дії в залежності від запиту і передає об'єкт Response в відображення (Views). Відображення відображають отримані дані заданим чином, забезпечуючи HTTP-відповідь.Фреймворк Laravel використовує Eloquent ORM, забезпечує гарну, просту реалізацію ActiveRecord для роботи з нашою базою даних. Кожна таблиця бази даних має відповідну «Модель», яка використовується для

взаємодії з таблицею.

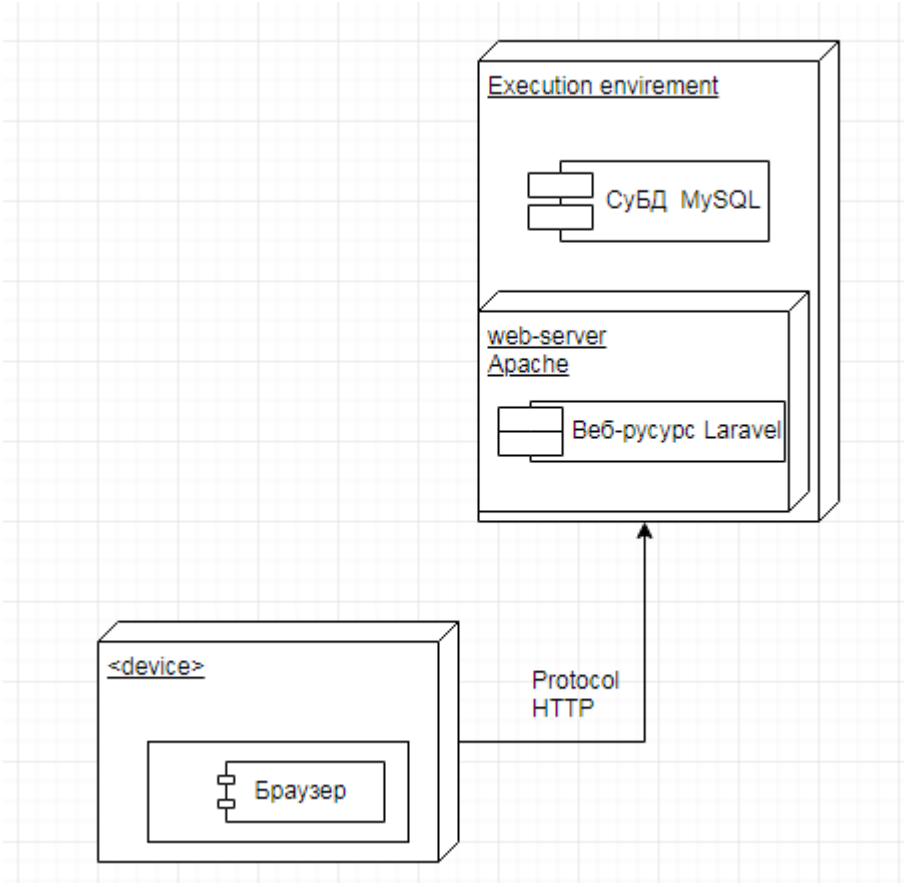

Рисунок 2.8 – Діаграма розгортання (DeploymentView)

2.10Уявлення слоїв IC (Design View)

На рис. (2.9) представлена діаграма слоїв веб-ресурсу.

–PresentationLayer (складається з основних предстовленій і контролерів)

– ApplicationLayer (містить основні сервіси та функції серверу; саме ці компоненти створюють основну бізнес-логіку застосування;).

– DomainLayer (сутності та сховище;).

– DB (містить основні дані проекту)

Вся система базується на шаблоні MVC (Model View Controller), що надається системою фреймворку Laravel. Також даний фреймворк реалізує трьохярусну архітектуру (three-tier architecture), тому система має три слої: слой уявлення, слой логіки та слой даних. На рівні даних описуються моделі даних, що описують сутності бази даних. За допомогою засобів фреймворку моделі автоматично використовують вбудовані Eloquent ORM, конструктор запитів та засоби взаємодії з базою даних.

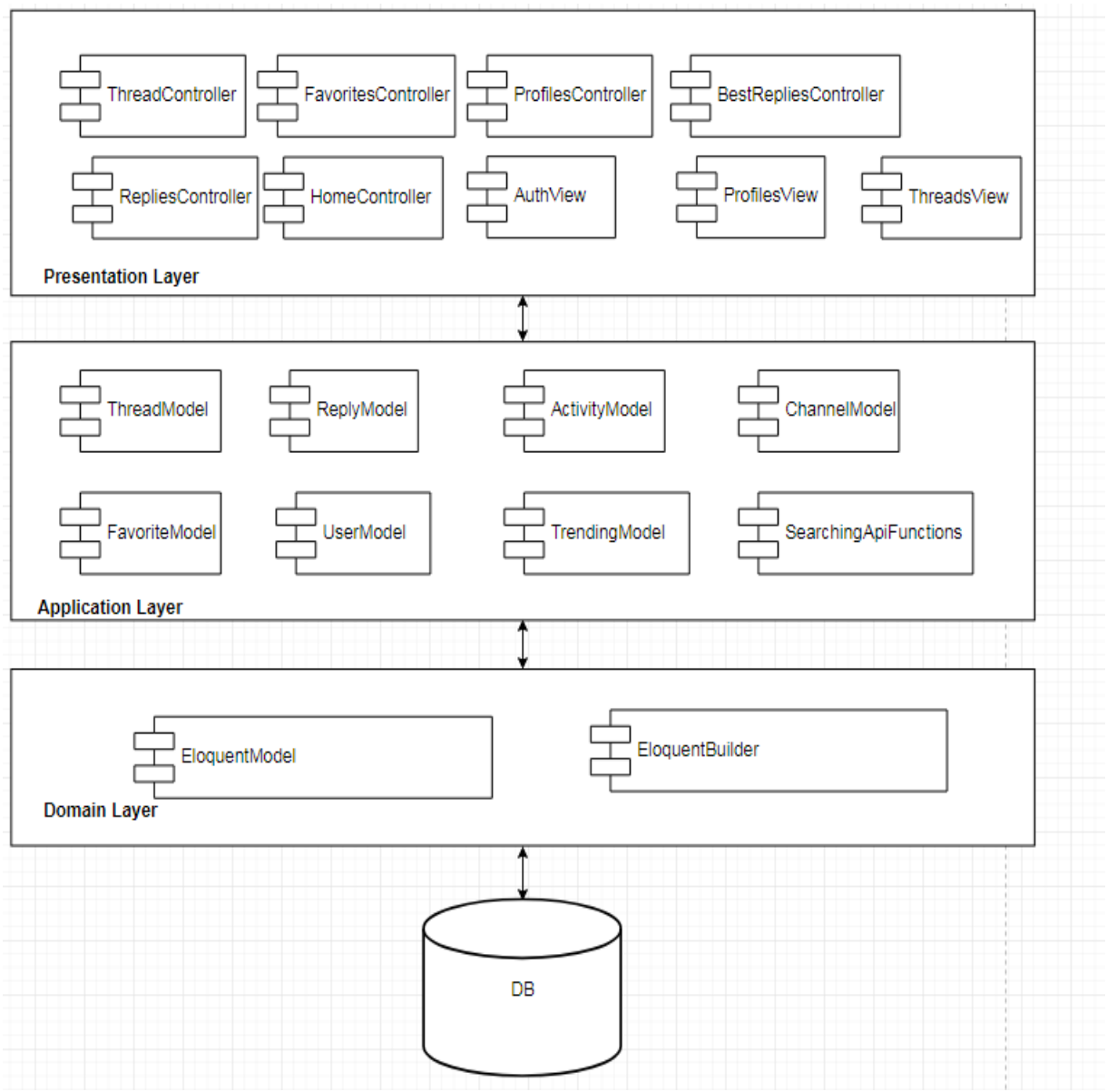

Рисунок 2.9 – Діаграма слоїв ІС (DesignView)

На рівні логіки присутні класи, що , які складають основну бізнес логіку застосування, також виконують додактові функції.Уровень отображения Views описує логіку надавання сторінок клієнту за його запитами, містить компоненти, які складаються з html-, css- і js-файлів та відповідають за відображення та розташування елементів системи на екрані, їхні анімацій.

2.11 Уявлення процесів ІС

Завдяки використанню фреймворку Laravel більша частина алгоритмів роботи виконується його інструментами. Проте, можна описати декідька ключових алгоритмів роботи програми.

Сценарій 1. Основним завданням веб-ресурсу є висновок тим на форумі і їх сортування. У цьому сценарії ми опишемо отримання популярних тем на форумі. Для детальної візуалізації процесса побудуємо діаграмі активності (рис 2.10) та послідовності веб-ресурсу для сценарію Отримання популярних тем на форумі.

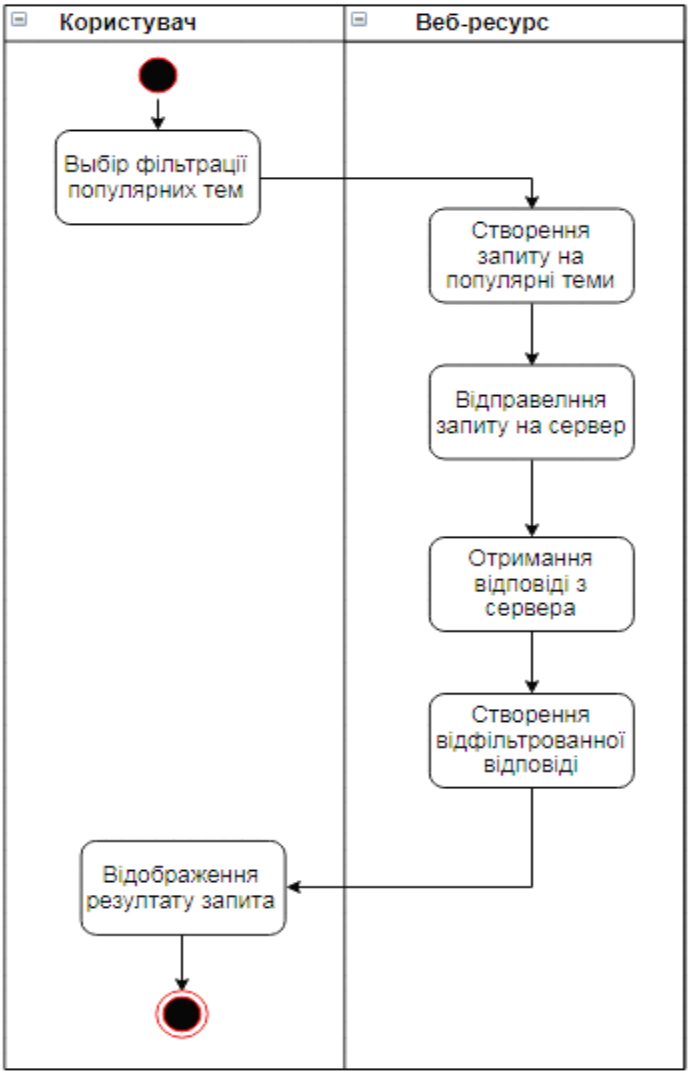

Рисунок 2.10 – Діаграма активності веб-ресурсу для популярних тем

Оператор вибирає фільтрацію тем за популярністю, створюється get запит .Перевіряється Route,до якого контролеру він належить та який метод відповідає за нього,після цього обробляється запит в ThreadController, за відображення даних користувачеві в ThreadController відповідає медоти show ().

Bshow() методі веб-ресурс звертається до моделіThreadModel, яка відповідає за дані в нашій базі даних, вона обробляє наш запит, отримує дані та відправляє їх назад в ThreadController, де відбувається їх фільтрація і передача оброблених даних назад в ThreadsView для відображення користувачеві в браузері.

Також представимо діаграму послідовності для цього сценарію на (рис 2.11).

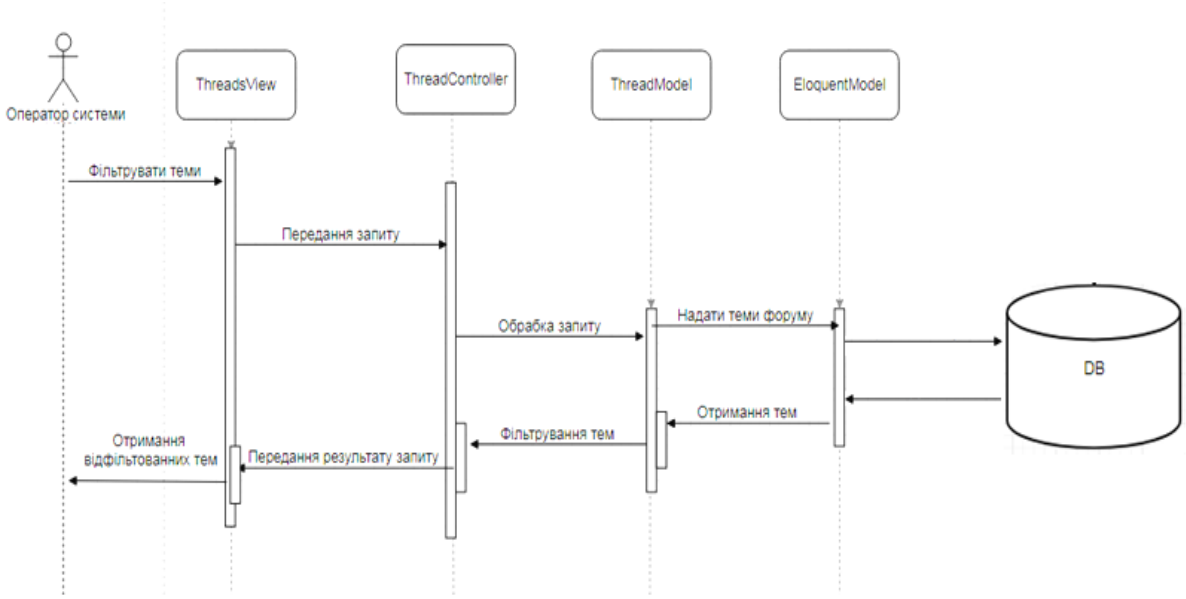

Рисунок 2.11– Діаграма послідовності веб-ресурсу для популярних тем

2.12 Уявлення даних IC (DataView)

На рисунку 2.12 зображено діаграму відношень (ER diagram) класів:

– User (класс,который содержит все данные о пользователе,такие как его логин,пароль,имя и т.д);

–Thread(класс,который содержит о темах на форуме, такие как пользователь,создавший тему, рубрика к которой относится данная тема, количество посещений,комментарии,подписку и тд);

– Reply (класс , который содержит все данные о комментариях и пользователях,которые их оставили);

– Activity(класс предстовляет собой полиморфную связь между класами Thread и Reply);

– Channel (класс, который содержит все данные о рубриках);

–Reply (класс , который содержит все данные о комментариях и пользователях,которые их оставили);

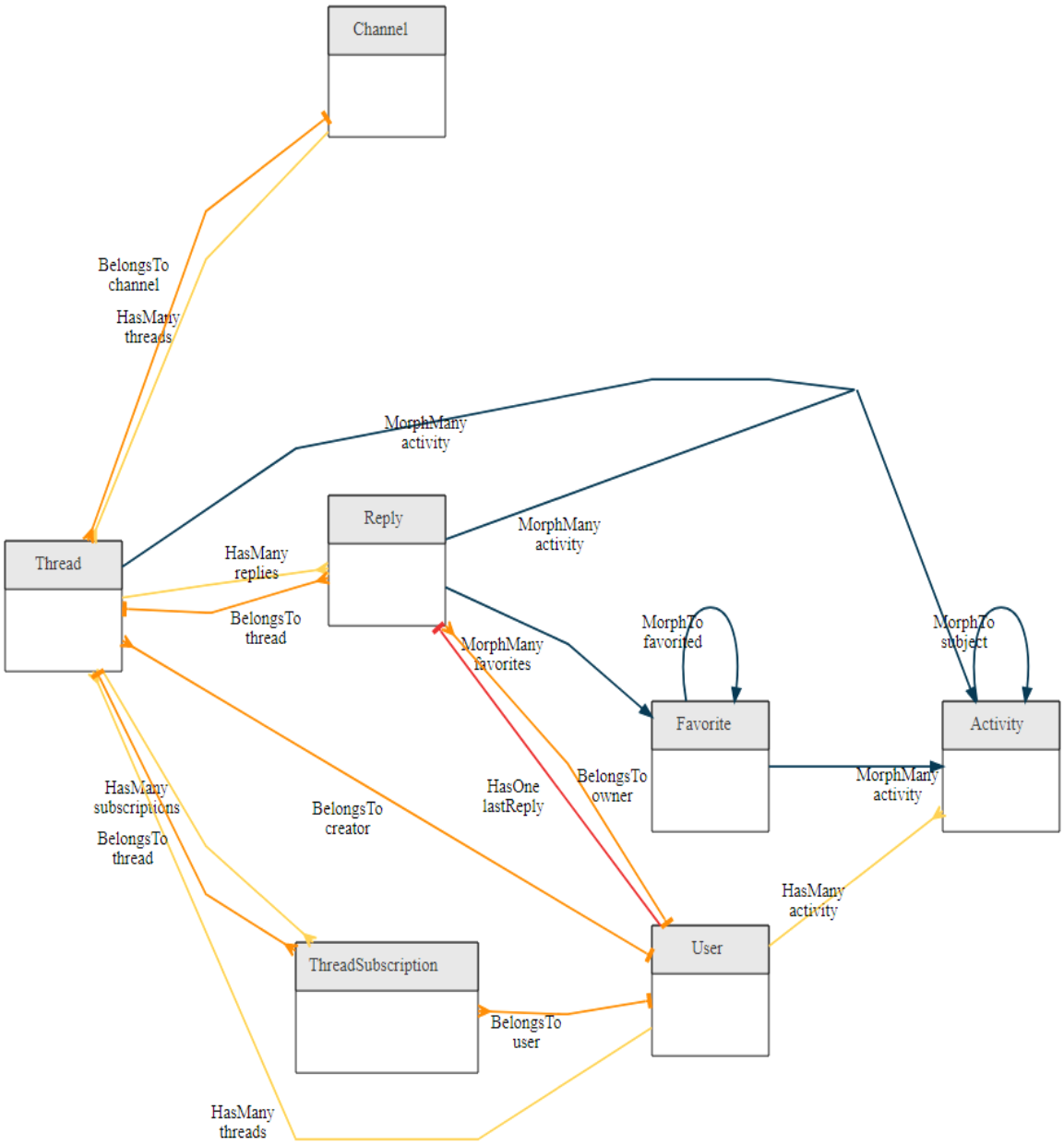

Рисунок 2.12 – Доменна модель системи

– Activity(класс предстовляет собой полиморфную связь между классами Thread и Reply);

–Channel (класс, который содержит все данные о рубриках);

–Favorite (класс , который содержит данные о понравившихся комментариях).

Усі сутності повинні зберігатися у релятивній базі даних для забезпечення можливості зберегти результаті, швидко отримувати дані, проводити фільтрацію.

Сховище даних буде одним та має тип MySQL. У якості ORM фреймворку буде використовуват Eloquent ORM. Реляційна схема бази даних зображена на (рис. 2.13) .

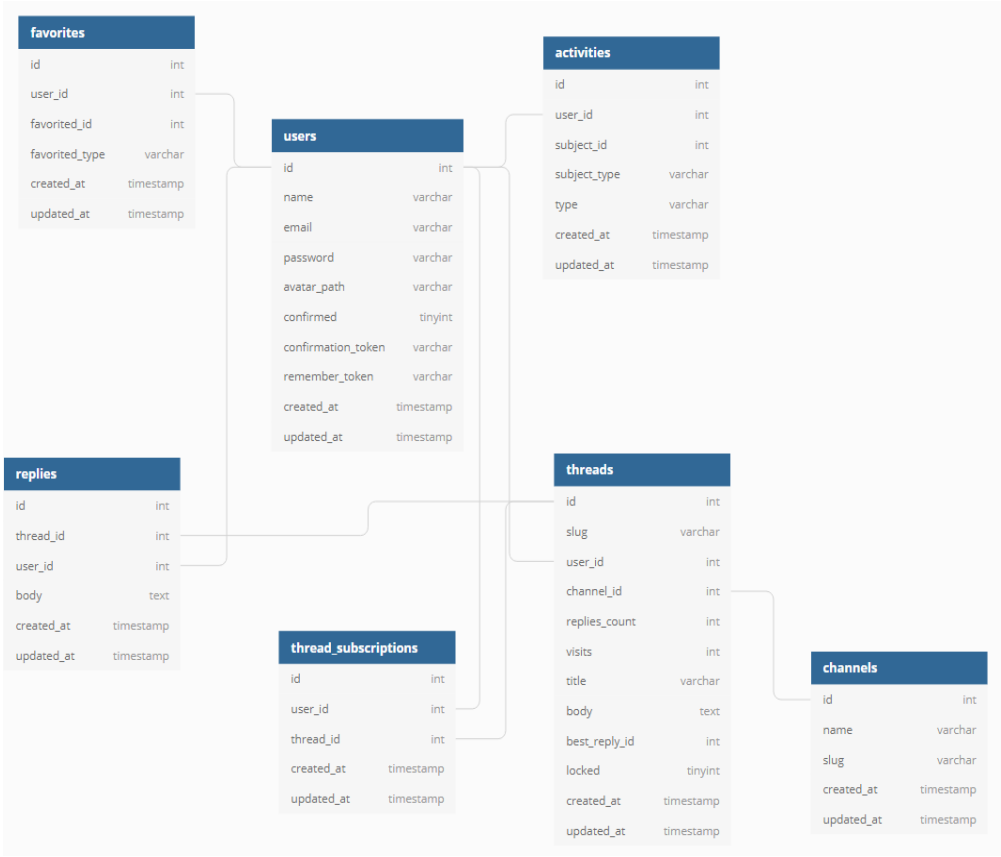

Рисунок 2.13 – Схема реляційної БД

Eloquent - красива і проста реалізація шаблону ActiveRecord в Laravel для роботи з базами даних. Кожна таблиця має відповідний клас-модель, який використовується для роботи з цією таблицею. Моделі дозволяють запитувати дані з таблиць, а також вставляти в них нові записи.

Також наведена діаграма комунікації (рис 2.14)

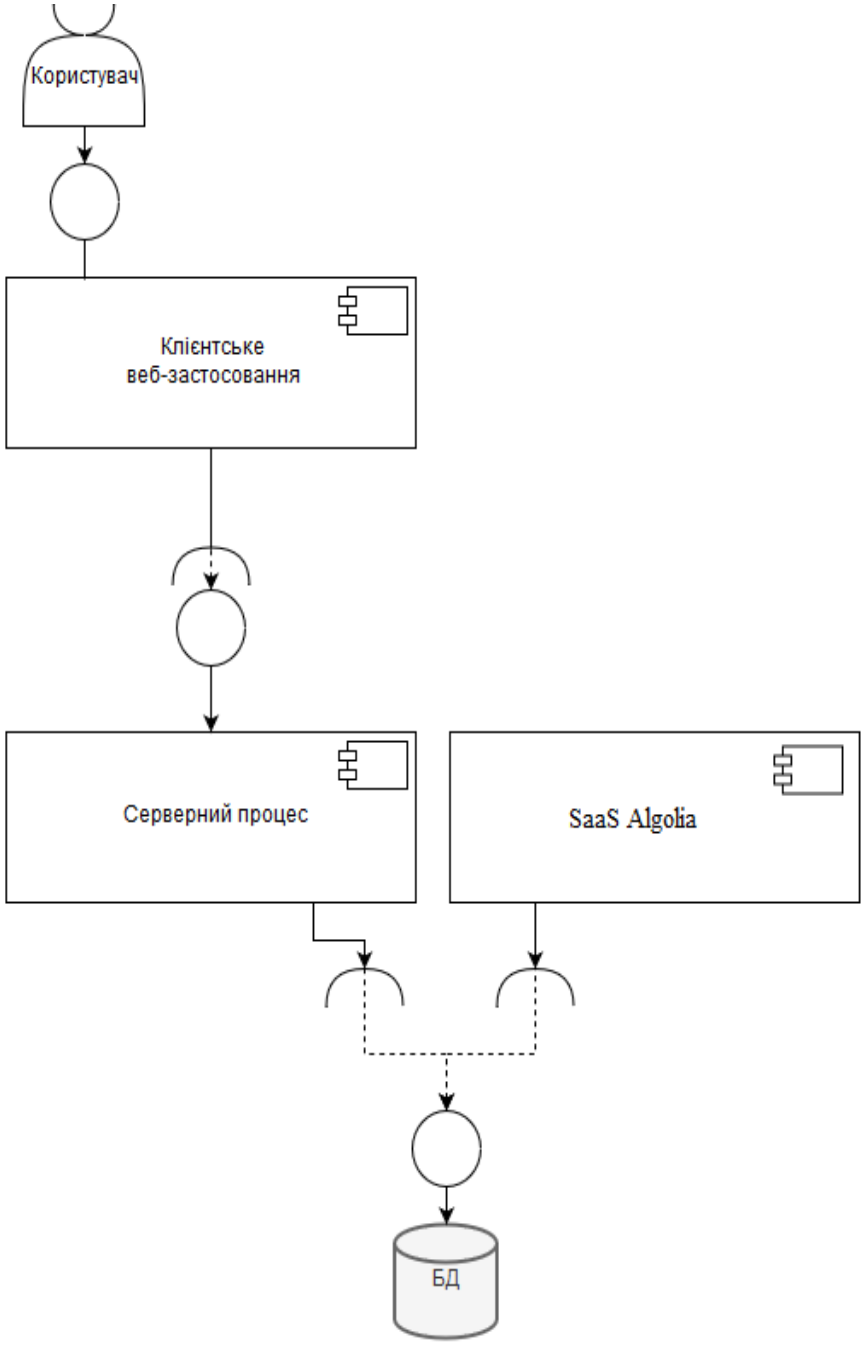

Рисунок 2.14 – Діаграма комунікаціїІС (CommunicationView)

Взаємодія між клієнтом та сервером виконується завдяки RESTAPI. Кожний маршрут також повинен мати асоційований з ним метод HTTP, тобто GET, POST, PUT або DELETE. В свою чергу, сервер повинен відповідати вимогам і стандартам, тобто повертати коректний статус запиту, коректно сповіщати про помилки, валідувати всі, хто входить запити тощо.

Algolia - пропріетарний SaaS, який індексує наш веб-ресурс і дає API для пошуку по його сторінках. Так само є API для відправки документів клієнта, підтримка контекстно-залежних запитів, і працює їх система дуже швидко.

2.13 Уявлення безпеки IC

Безпека є важливим аспектом при розробці веб-додатків. У Laravel є механізми для захисту веб-сайту. Нижче перераховані деякі з них. Зберігання паролів - Laravel надає клас з ім'ям «Hash», який забезпечує безпечне кешування Bcrypt.

Захист від CSRF (межсайтовой підробки запиту) / XSS крос-сайтовий скриптинга - атаки з використанням крос-сайтовий скриптинга здійснюються, коли зловмисник має можливість розмістити код JavaScript на стороні клієнта на сторінці, що переглядається іншими користувачами. Щоб захиститися від подібного типу атак, ми перевіряємо призначені для користувача дані і не допускаємо додавання небезпечних сигнатур. У шаблонах Blade ми використовуємо синтаксис подвійний прив'язки ({{$ value}}).

Laravel дозволяє легко захистити наш веб-ресурс від атак з підробкою міжсайтових запитів (CSRF). Підробка міжсайтових запитів - тип атаки на сайти, при якому несанкціоновані команди виконуються від імені аутентифицированного користувача. Laravel автоматично генерує CSRF- "токен" для кожної активної користувальницької сесії в додатку. Цей токен використовується для перевірки того, що саме авторизований користувач робить запит в додаток. При визначенні кожної HTML-форми ми включаємо в неї приховане поле CSRF-токена, щоб посередник CSRF-захисту міг перевірити запит.

Протидія SQL-ін'єкцій - вразливість до SQL-ін'єкцій пов'язана з вставкою нефільтрованих довільних даних користувача в SQL-запит. За замовчуванням Laravel захищає нас від такого типу атак, оскільки і конструктор запитів, і Eloquent використовують клас об'єктів даних PHP (PDO). PDO використовує підготовлені оператори, які дозволяють безпечно передавати будь-які параметри без необхідності їх видалення і санації.

Cookies - безпечні за замовчуванням - Laravel дозволяє просто створювати, читати і відключати файли cookie за допомогою класу Cookie. У Laravel всі файли cookie автоматично підписуються і шифруються. Це означає, що якщо вони будуть підроблені, Laravel автоматично відкине їх. Це також означає, що ви не зможете вважати їх на стороні клієнта за допомогою JavaScript.

Примусове використання HTTPS при обміні конфіденційними даними - HTTPS блокує перехоплення зловмисникам в одній мережі конфіденційної інформації, такої як змінні сеансу, і реєстрацію з її допомогою від імені жертви.

2.14 Опис стеку технологій

IDEPhpStorm - це інтегроване середовище розробки на PHP з інтелектуальним редактором, яка глибоко розуміє код, підтримує PHP 5.3-7.3 для сучасних і класичних проектів, забезпечує краще в індустрії автодоповнення коду, рефакторинг, запобігання помилок нальоту і підтримує змішування мов.

Composer - це менеджер залежностей для інтерпретатора мови PHP.

Для розробки усієї системи вікорістовується фреймворк Laravel v5.5..

Для написання програмного коду вікорістовується мова програмування Php. Для розробки модулю пошуку по сайту використовувалися API Algolia.

Для реалізації шаблонів веб-сторінок вікорістовується сучасна технологія HTML5 та Специфікація CSS3, так само використовувався Laravel шаблонизатор Blade зі своїм набором директив.

Для Підвищення якості та швідкості розробки розміткі веб-сторінок вікорістовується бібліотека елементів Twitter Bootstrap.

Для зберігання даних використовувалася база даних MySQL.

Для модульного тестування проекту вікорістовується система PHPUnit.

3 РОЗРОБКА ВЕБ-РЕСУРСУ ДЛЯ СПІЛКУВАННЯ МІЖ СТУДЕНТАМИ ЗА ДОПОМОГОЮ ФРЕЙМВОРКУ LARAVEL

3.1 Уявлення структури програмного коду

Розбиття файлів вихідного коду програмного забезпечення дозволяє створити таку структуру проекту, що полегшує процес розробки та робить можливим паралельну розробку з мінімальними конфліктами у вихідному коді. Наявність модульності у проекті також дозволяє повторно використовувати ті чи інші набори файлів вихідного коду.

У проектах на мові програмування Php класи можуть зберігатися у окремих файлах з розширенням .php, а також у модулях. Так як використовуються фреймворк Laravel, виникає необхідність дотримуватися норм щодо організації проекту в узгодженні зі специфікаціями до структури проекту при використанні даного фреймворку.

Файлова структура проекту представлена на (рис. 3.1).

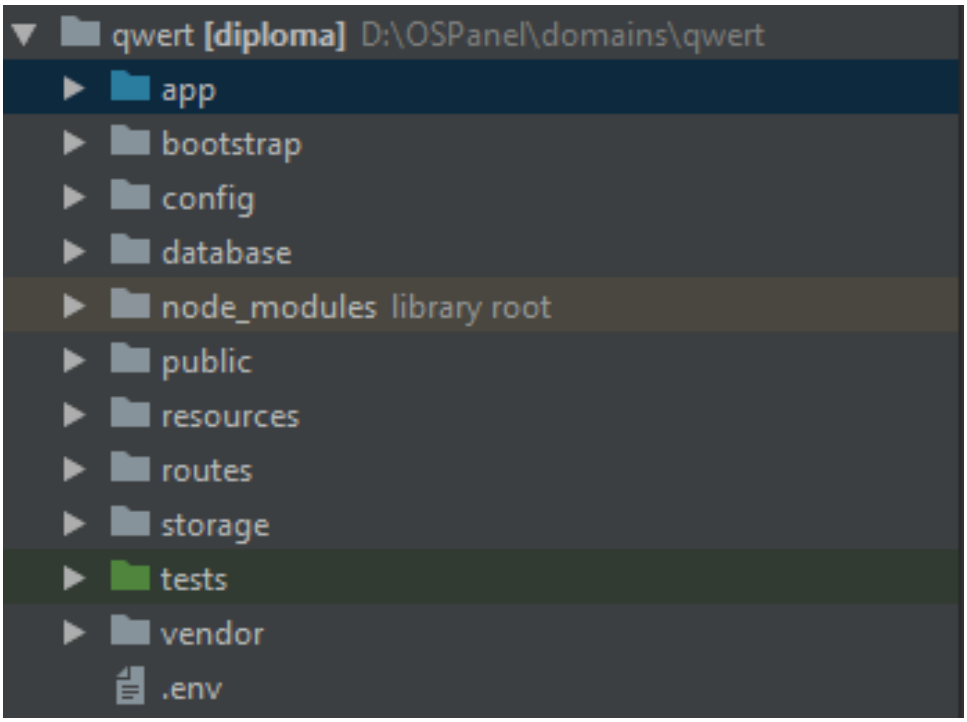

Рисунок 3.1 – Файлова структура проект

Як видно, стандартна файлова структура складається з великої кількості каталогів.

В таблиці 3.1 наведено короткий виклад функцій кожного з цих файлів і папок:

Таблиця 3.1– Опис структури проекту

Файл	Описання файлу
1	2
/app/	Складається з контролерів MVC моделі, моделей ORM і утиліт консолі.
/bootstrap/	Містить файли, які здійснюють початкову завантаження (bootstraping) фреймворка і налаштовують автозагрузку класів.
/config/	У цій папці зберігаються файли для настройки всіляких модулів Laravel: починаючи з бази даних і кешування, закінчуючи модулем відправки листів і файлової системою.
/database/	У цій папці зберігаються файли міграцій, сідування і генерації баз даних з моделей.
/node_modules/	Папка, в якій зберігаються підключаються JS модулі, встановлені за допомогою утиліти nvm.
/public/	Тільки ця папка на сервері доступна зовнішнього світу. На цей каталог повинен посилатися веб-сервер. Він містить завантажувальний файл index.php, який запускає ядро Laravel. Цей каталог також може бути використаний для зберігання будь-яких загальнодоступних статичних файлів, таких як CSS, JavaScript, зображення та інші.
/resources/	У цій папці знаходяться ресурси для роботи з фронтенда: вихідні JavaScript і css файли для збірок, файли локалізації та файли представлень.

Продовження таблиці 3.1

/routes/	Тут зберігаються файли для конфігурації модуля маршрутизації
1	2
/storage/	Ця папка призначена для зберігання генерованих роботою Laravel файлів: сесій, кеша і балок.
/tests/	Як випливає з назви, в цій папці зберігаються файли модульних тестів.
/vendor/	Це місце для всіх сторонніх частин програми. У типовому додатку Laravel в нього включається вихідний код Laravel, його залежності, а також плагіни, які містять додаткові попередньо упаковані функції.
.env	З цього файлу Laravel бере environment-змінні, наприклад, тут, за допомогою зміни значення змінної DEBUG, можна задати стан Laravel додатки - production або debug.

3.2 Уявлення про класи ІС

Кожен із перелічених файлів містить набір класів,та функцій реалізованих для створення веб-ресурсу.Нижче будуть представлені основні класси цього проекту.

Директорія /app/ містить у собі такіконтроллери як: BestRepliesController,Controller, FavoritesController, HomeController, LockedThreadsController, ProfilesController ,Replies Controller, SearchController, ThreadsController, ThreadSubscriptionController.

Такожцядиректоріяміститьусобіпереліквсіхмоделей: Activity, Channel, Favorite, Reply , Thread , ThreadSubscription, Trending, User.

Директорія /routes/ містить в собі файл web.php, в якому зберігаються усі наші маршрути.

Директорія /database/ містить в собі все наші міграції та фабрики, ось деякі з них: create_user_table.php, create_replies_table.php, create_threads_table.php, create_channels_table.com, create_activites_table.php, ModelFactory.php.

Директорія /resources/views містить у собі наші уявлення для веб-ресурсу. Існує два головних layouts: app.blade.php, nav.blade.php. Також э директорія уявлень тем на форумі /resources/views/threads яка містить в собі такі уявлення: create.blade.php, index.blade.php, reply.blade.php, search.blade.php, show.blade.php.

3.3 Інфраструктурне уявлення ІС (InfrastructureView)

Так як веб-ресурс розгортається у браузері, нижчих привидів список підтрімуваніх версій браузерів:

– IE 11;

– Microsoft Edge (Версії 16 та вищє);

– Mozilla Firefox (Версії 59 та вищє);

– Google Chrome (Версії 64 та вищє);

– Opera (Версії 51 та вищє).

Для розгорнутого веб-сервера на віддаленому хостингу виділяється апаратна частина з процесором Intel Xeon x64 CPU E5-2670 v2 2.50GHz та ОЗУ DDR3 1333MHz 4Gb, з віхідом в ятір Інтернет та статичним IP-адресою.

На стороні клієнта, предусмотрена будь-який сучасний web-браузер Із підтрімкою HTML5 та JavaScript НЕ нижчих Версії 1.5.

3.4 Стратегія доставки продукту (DeliveryStrategyView)

Так як інформаційна система представляет собою веб-ресурс, програмний продукт планується доставляті за посиланням адреси веб-сайту http.studforum.com.

3.5 Керування програмним кодом ІС

Управління програмним кодом віконується с помощью системи контролю версій Git та его веб-оболонки GitHub, что дозволяє зберігаті та вести Історію написання програмного продукту з можлівістю відкату чи перегляду попередніх змін.

Вихідний код системирозміщен у публічному репозиторії.

Типліцензії: MITLicense, що робить це застосування відкритим до копіювання та модифікацій.

Після завершення розробки були розраховані метрики, що демонструють кількісну складову реалізації системи(табл. 3.1).

Таблиця 3.1 – Метрики системи контролю версій Git

Метрика	Значення
Кількість комітів у masterbranch	28
Загальна кількість бранчів	5
Кількість закритих PR	2

3.6 Розрахунок метрик програмного коду ІС

Для розрахунку метрик коду представимо їх у відповідній таблиці (табл. 3.2).

Таблиця 3.2 – Метрика програмного коду проекту

Метрика	Одиниця вимірювання
1	2
Загальна кількість рядків коду в проекті ІС	2754
Середня кількість рядків коду у класі	87
Максимальная кількість рядків коду у класі	289
Середня кількість рядків коду у методі	26
Максимальна кількість рядків у одному методі	58
Максимальна глибина дерева спадкування	1
Средня цикломатична складність	1.33
Максимальна цикломатична складність	1.5
Коментування коду, %	40
Покриття коду тестами, %	45

3.7 Контрольний список з якості реализации ІС

Було створено контрольний список та відображено у відповідній таблиці (табл. 3.3).

Таблиця 3.3 – Контрольний список з якості реалізації ІС

Ствердження	Відповідь	Обгрунтування у разі негативної відповіді
1	2	3
Використання Dependency Injection	Так	
Використання логування	Так	
Використання модульного тесту	Так	
Захист від SQL-ін’єкцій	Так	
Захист від JS-ін'єкцій	Так	
Використаня валідації в усіх полях вводу форм для інтерфейсу	Так	
Використання інсталяторів або інтернет-магазинів	Ні	Система є веб-сайтом і доступна в мережі Інтернет
Використання засобів синхронізації багатопотоковісті	Ні	Багатопотоковість не запроваджена.
Підтримка глобалізації ІС	Так	
Відсутність зашитих до програмного коду конфігураціоних параметрів застосування	Так	
Використання UI-паттернів під час розробки UI	Так	
Враховані codestyleguidelines для обраної мови програмування	Так	
Враховані guidelines для розробки UI обраної ОС	Так	

3.8 Документація ІС

Для веб-ресурсу розробимо два документи: керівництво користувача, керівництво адміністратора ІС.

3.8.1 Керівництво користувача.

Для використання данного веб-ресурсу, необхідно використовувати один із сучасних браузерів (Google Chrome 56, Microsoft Edge 15, Mozilla Firefox 51, Opera 43) та відповідний ПК для задовільнення апаратних вимог браузеру.

Реєстрація, авторизація та зміна даних.

Для реєстрації необхідно перейти на сторінку реєстраціїї, натиснувши кнопку «Реєстрація», заповнити всі необхідні поля. Після коректного воду даних треба натиснути кнопку «Зареєструватися»

Якщо треба змінити дані введені при першому запуску, треба перейти до свого профілю, натиснувши на кнопку «Профіль», натиснути на кнопку «Змінити дані», відредагувати дані та натиснути кнопку «Зберегти».

Для авторизації необхідно натиснути кнопку «Авторизація», ввести логін і пароль, та натиснути кнопку «Увійти» (рис 3.2).

3.8.2 Перегляд тем

Для перегляду тем, необхідно зайти на головну старніцах сайту, де будуть відображенням всі опубліковані теми на форумі (рис. 3.2).

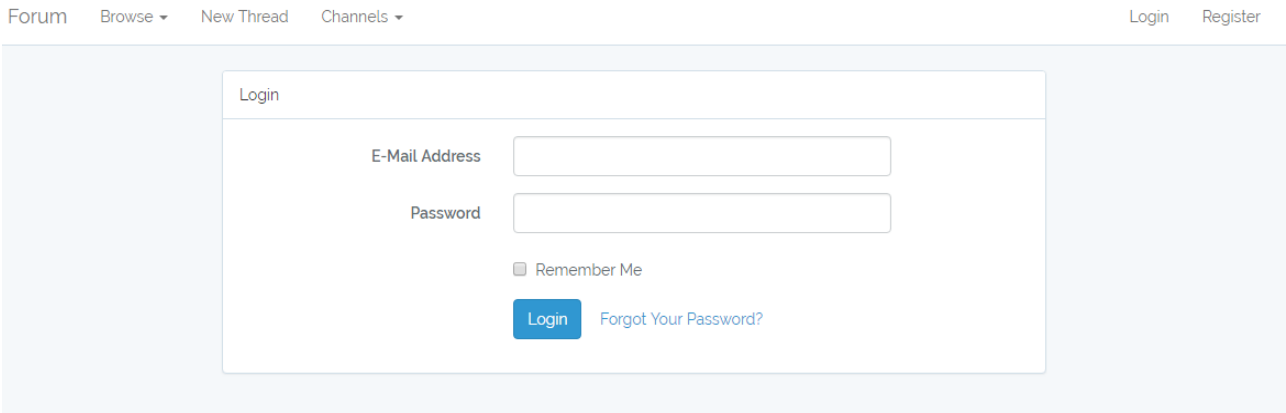

Рисунок 3.2 – Скріншот авторизації

Кожна тема складається з розділу заголовка і розділу опису, заголовок теми є посилання на цю тему, перейшовши по якій можна буде більш детально вивчити її, а так само взяти участь в діскуссіі.

Так же в цій темі можна буде оцінити вподобаний вам коментарнатиснувши на іконку like, і підписатися на дану тему, натиснувши в правому куті «Підписатися» (рис. 3.3).

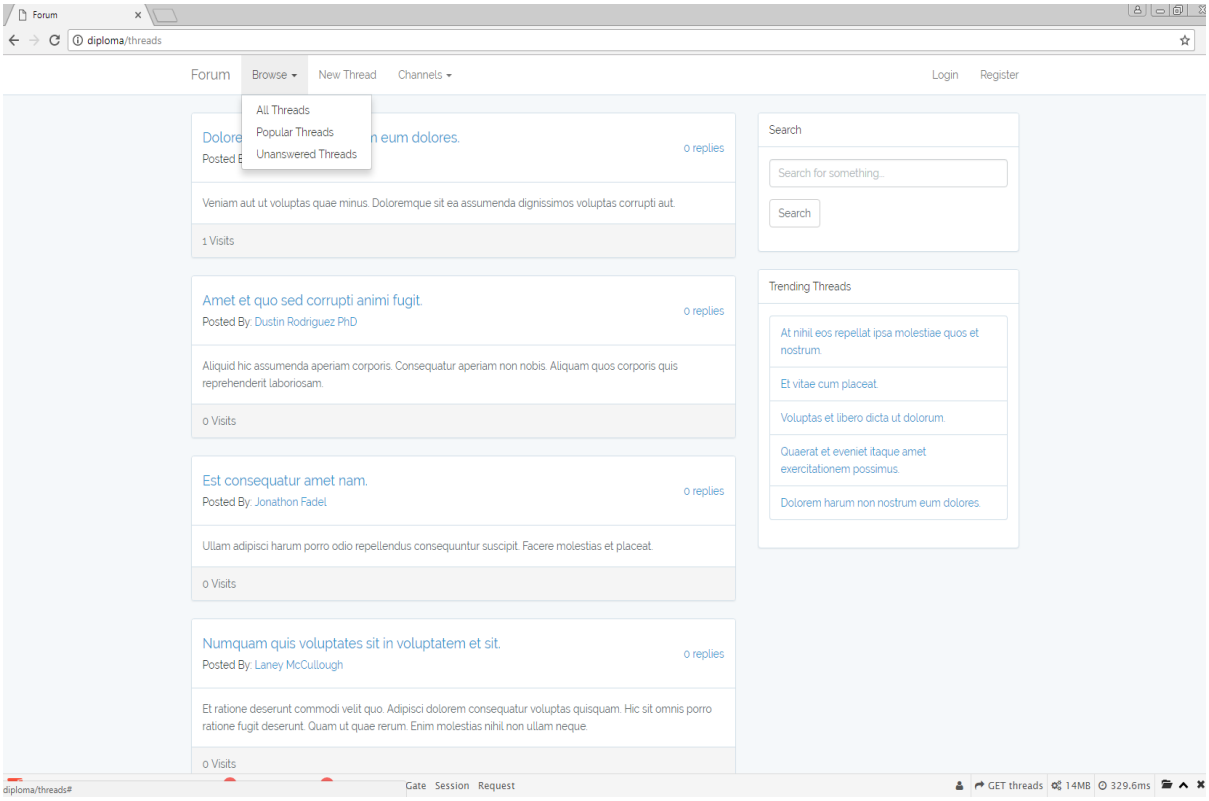

Рисунок 3.3– Скріншот перегляду тем

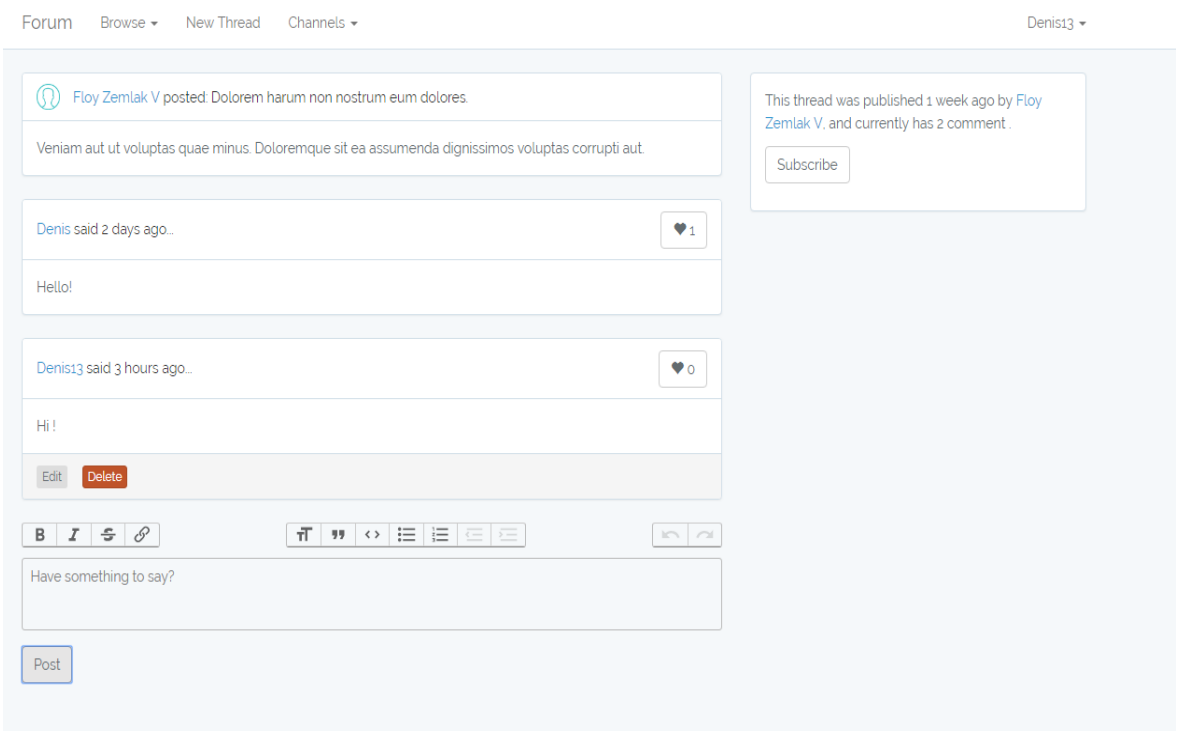

Рисунок 3.4– Скріншот спілкування в темі

У навігаційному меню зліва розташовані пункти: «Теми форуму» і «Теми». У пункті справа - зазначено ім'я зареєстрованого користувача. Всі три пункти є заголовками випадають меню.

При натисканні на пункт користувача можна перейти:

– до профілю користувача (рис. 3.5) зі списком створених ним тем і повідомлень, користувач може перейти за своїми темам і повідомленнями та відредагувати їх або видалити;

– вийти з облікового запису користувача.

При натисканні на пункт «Теми», користувач буде надано всі рубрики тем, перейшовши за якими, можна буде побачити усі теми створені за даній рубриці (рис. 3.6).

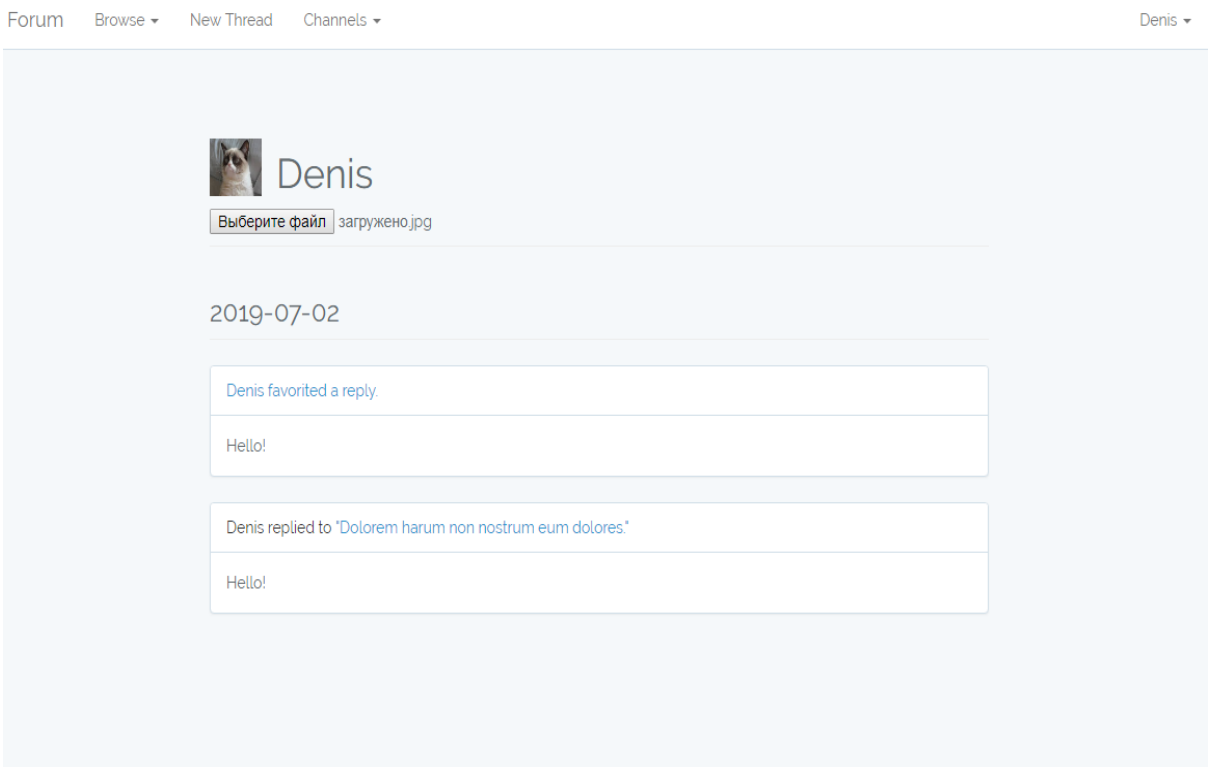

Рисунок 3.5 – Сторінка про файлу користувача

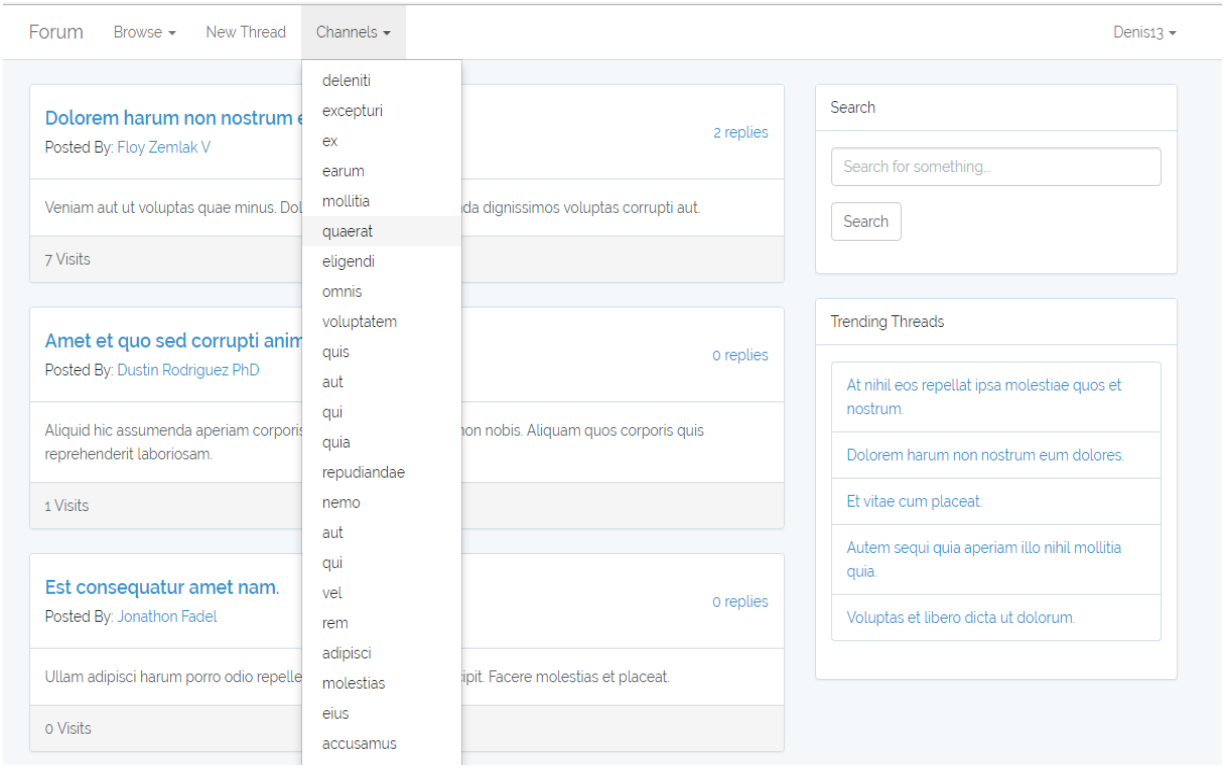

Рисунок 3.6 – Скріншот рубриць тем

При натисканні на пункт Теми можна перейти:

– до створення нової теми;

– до відображення власних тем;

– до популярних тем на форумі;

– до тем без відповіді;

В правому куті знаходиться панель пошуку по форуму. Користувач вводить потрібну йому інформацію і натискає на кнопку «Пошук», після чого потрапляє на сторінку з темами содержащию цікавить користувача інформацію, перейшовши за якими він може більш детально вивчити тему.

3.8.3 Адміністрування Користувачів.

Адміністратор має ті ж можливості що і звичайні користувачі, додатково до цього він може видаляти всі теми і повідомлення, а так само блокувати користувачів.

3.9 Розробка тест плану

Тестування інформаційної системи це етап, що проводиться для покращення якості програмного продукту. Для даної системи необхідно провести 4 типи тестування:

– функціональне тестування;

– модульне тестування;

– тестування обсягом;

– навантажувальне тестування;

Функціональне тестування – це проходження всіх функціональних вимог, що ставились перед системою та пошук розходжень між наявним функціоналом та бажаним.

Модульне тестування – це тестування методів системи. Для модульного тестування буде використовуватись система PHPUnit – это система для юнит-тестирования приложений, написанных на языке PHP. Под "юнит" понимаются небольшие блоки кода, например отдельные методы класса.

Для тестування об’ємів буде використовуватись програмний інтерфейс до бази даних MySQL.

Навантажувальне тестування буде проводитись за допомогою застосунку Apache JMeter.

Тестування буде проводитись на комп’ютері під управлінням ОС Windows 10 64x з процессором Intel Pentium 2117U із частотою 1,8 ГГц та оперативною пам’яттю у 8 Гбайт.

На якість навантажувального тестування буде впливати обчислювальна потужність машини, на якій розгорнуто веб-сервер. На якість розроблених модульних тестів впливає складність на розміри методів, до яких розробляються тести. Чим більше логіки реалізує метод, чим більше сторонніх сутностей він використовує, тим складніше розробити повні модульні тести.

3.10 Розробка тест-кейсів для функціонального тестування

Для проведення функціонального тестування треба розробити спеціальні тест-кейси. Для цього необхідно створити список з роз’ясненням кожного тест-кейсу.Список має таку структуру: ідентифікатор – номер функціональної вимоги – опис тесту – передумови – що треба зробити – що повинно відбутись.

1. FR-1.1.1 – валідація даних – аккаунт для реєстрації незайнято – треба ввести неправильні дані в поля логіну чи паролю і натиснути кнопку реєстрації – в результаті повинне висвітитись попередження про невірний формат даних.

2. FR-1.1.2 – реєстрація – аккаунт для реєстрації незайнятий – треба ввести вірний формат даних в поля логіну та паролю – в результаті виведеться сповіщення про успішну реєстрацію.

3. FR-1.1.3 – вибір фото – аккаунт для реєстрації незайнято – треба натиснути на кнопку вибору фото та завантажити його з файлової системи – фото буде встановлене в полі Рисунок профілю.

4. FR-1.2.1 – валідація даних – аккаунт для авторизації існує – ввести неправильні дані в поля логіну чи паролю і натиснути кнопку авторизації – перенапрям на сторінку зі сповіщенням про помилку в даних профілю.

5. FR-1.2.2 – авторизація – аккаунт для авторизації існує – треба ввести вірний формат даних в поля логіну та паролю – в результаті буде проведений вхід у аккаунт та з’явиться кнопка переходу до персональної сторінки – буде показане попередження про невірний формат даних.

6.FR-2.1.1 – додавання теми – статус користувача не є Гість, треба ввести вірний формат данних в поля тема, опис та вибрати рубрику до якої відноситься, та нажатии кнопку Створити – в результаті повинна пройти валідація данних та відбудиться перехід на головну сторінку, якщо не пройде,буде вказано які данні не пройшли валідацію, та повторне їх заповнювання.

7. FR-2.2 – видалення теми – статус користувача має бути Адміністратор чи авторизованний користувач, який створив цю тему – треба перейди до свого профайлу , обрати тему, та натиснути кнопку Вилучити – - статус відповіді з серверу 200 ,тема та всі її комментарії повинні видалитись з бази данних.

8.FR-2.3 – сортування тем – статус користувача має бути будь який, користувач натискає на кнопку сортування тем, обирає яке буде зроблене сортування та натискає на нього - статус відповіді з серверу 200, відповідь містить в собі всі теми відсортованні за популярністю.

9.FR-2.4 – пошук тем за назвою - статус користувача має бути будь який, користувач вводить інтересуючи його тему в поля пошуку,натискає на кнопку пошуку тем, вводить інтересуючи його тему - статус відповіді з серверу 200, відповідь містить в собі всі теми,які містять в собі шуканої інформації.

3.11 Протокол проведення функціонального тестування

Виконаємо тестування розробленної системи та проведемо відповідність між очікуваним результатом та отриманим. Дані, що отримаємо при функціональному тестуванні занесемо до таблиці 3.4.

Таблиця 3.4 – Функціональне тестування

Номер тест-кейсу	Очікуваний результат	Отриманий результат	Коментарі
Тест кейс № 1	В результаті повинне висвітитись попередження про невірний формат даних	Висвітилось попередження про невірний формат даних.	Виконано
Тест кейс № 2	Виведеться сповіщення про успішну реєстрацію	З’явилось сповіщення про успішну реєстрацію	Виконано
Тест кейс № 3	Фото буде встановлене в полі зображення профілю	Після вибору фото воно було встановлено в поле зображення профілю	Виконано
Тест кейс № 4	Перенапрям на сторінку зі сповіщенням про помилку в даних профілю	Після натискання кнопки «Увійти» було виконано перенаправлення на сторіку з текстом помилки	Виконано
Тест кейс № 5	Буде показане попередження про невірний формат даних у вигляді сповіщення під полем вводу даних про авторизацію	Показане попередження при вводі невірних даних	Виконано
Тест кейс № 6	Буде створенна нова тема ,та відображена на головній сторінці	Створена тема, данні збереженні.	Виконано
Тест кейс № 7	Буде видалена тема,та всі її комментарі	Видалена тема,всі дані видалені з БД.	Виконано
Тест кейс № 8	Користувач отримує відсортовані теми	Користувач отримав відсортовані теми	Виконано
Тест кейс № 9	Користувач повинен отримати інтересуючи теми	Користувачу відобразились інтересуючи його теми	Виконано

За результатами функціонального тестування можна зазначити, що функціональні тести були виконані та відповідають заданим вимогам.

3.12 Проведення модульного тестування

Для проведення модульних тестів скористаємось бібліотекою phpunit(рис. 3.6).

Загальне покриття коду складає 45% покритих строчок. Дані про покриття вихідного коду модульними тестами здобуто за допомогою утиліти coverage. Дана утиліта надає значення покриття програмного коду модульними тестами за файлами з вихідним кодом. Настільки малий відсоток покриття тестами пояснюється тим фактом, що в фреймворку Laravel значна кількість коду є необхідним для праці системи. Ці рядки не несуть бізнес-логіки та не повинні бути покритими тестами.

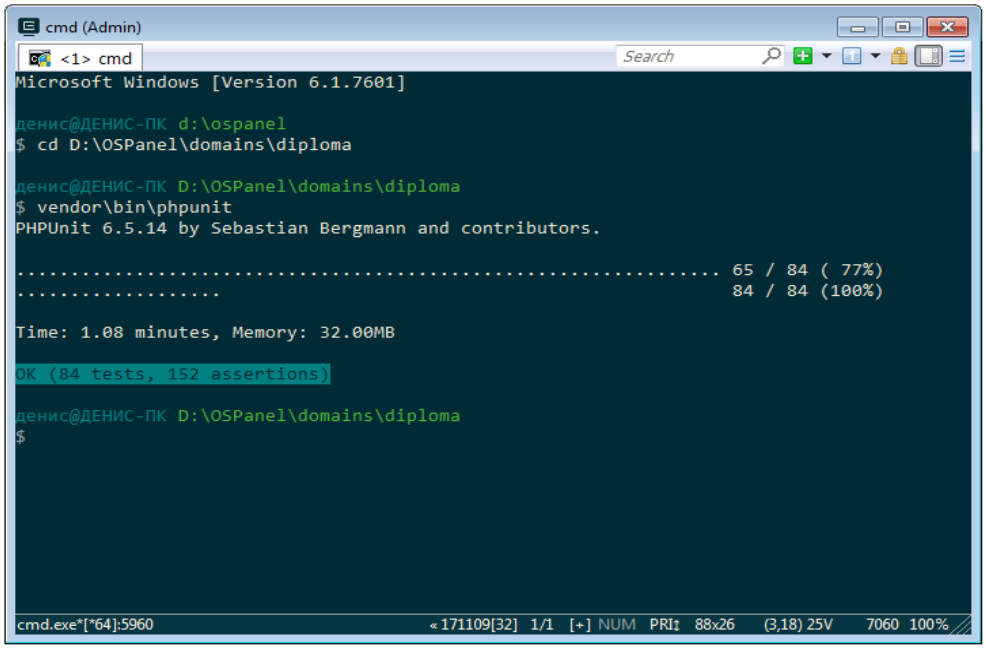

Рисунок 3.6 – Покриття коду модульними тестами

Для демонстрації тестів візьмемо один з тест-кейсів та опишемо принцип його роботи. Прикладом буде тест-кейс для класів моделей, які відображають сутності бази даних (рис 3.7).

```
class UpdateThreadsTest extends TestCase
{
use RefreshDatabase;
public function setUp()
    {
parent::setUp();
$this->withExceptionHandling();
$this->signIn();
    }
/** @test */
function unauthorized_users_may_not_update_threads()
    {
$thread = create('App\Thread', ['user_id' => create('App\User')->id]);
$this->patch($thread->path(), [])->assertStatus(403);
    }
/** @test */
function a_thread_requires_a_title_and_body_to_be_updated()
    {
$thread = create('App\Thread', ['user_id' => auth()->id()]);
$this->patch($thread->path(), [
'title' =>'Changed'
])->assertSessionHasErrors('body');
$this->patch($thread->path(), [
'body' =>'Changed'
])->assertSessionHasErrors('title');
    }
/** @test */
function a_thread_can_be_updated_by_its_creator()
    {
$thread = create('App\Thread', ['user_id' => auth()->id()]);
$this->patch($thread->path(), [
'title' =>'Changed',
'body' =>'Changed body.'
]);
tap($thread->fresh(), function ($thread) {
$this->assertEquals('Changed', $thread->title);
$this->assertEquals('Changed body.', $thread->body);
        });
    }
}
```

Рисунок 3.7 – Приклад тест-кейсу для оновлення тем

У першому тесті перевіряється можливість оновлювати теми не авторизованному користувачу.

У другому тесті перевіряється щоб в оновленній темі були заповненні данні по темі, та її описанню.

У третьому тесті перевіряється щоб тема була оновлена лише її творцем.

Таким чином було проведено тестування більшої частини класів проекту, що піддається написанню модульних тестів. Класи, що надаються фреймворком, не були протестовані, так як це було б надмірно.

Важливим моментом при проведенні модульного тестування є контролювання надмірності складання тестів, особливо при використанні фреймворку: більшість інструментів вже створені на протестовані, тому доцільно складати модульні тести лише для створених під час розробки нових полів та поведінок.

3.13 Проведення тестування навантаженням

Для тестування роботи з базою даних треба провести тестування навантаженням. Для цього база даних заповнюється великим числом записів і перевіряються типові операції з базою даних.

Щоб зробити таке тестування, в базу даних, у кожну з таблиць було дано по 100 000 записів і було пророблено типові дії для веб-сайту. Результати тестування об’ємів (час реакції застосунку на ці дії) представлені у таблиці 3.5

Проаналізувавши результати тестування можна зробити висновок, що робота з базою даних відбувається у звичайному порядку, хоча при наявності великої кількості записів час відгуку програми збільшується, але не на дуже великі величини.

Таблиця 3.5 – Результат тестування навантаженням

Номер тест-кейсу	Опис тест-кейсу	Час відгуку, мс
Тест кейс № 1	Авторизація користувача на веб-сайті	356
Тест кейс № 2	Реєстрація користувача на веб-сайті	467
Тест кейс № 3	Пошук тем за сортуванням	894
Тест кейс № 4	Створення теми	473
Тест кейс № 5	Редагування користувача	398
Тест кейс № 6	Видалення теми з БД	502

ВИСНОВКИ

В процесі виконання кваліфікаційної роботи був розроблений веб-ресурсдля спілкування студентів, за допомогою фреймворку Laravel. Мета, що була поставлена в дипломній роботі була цілком виконана.

Проаналізувавши існуючі аналоги, було вирішено спроектувати та реалізувативеб-ресурс у вигляді з RESTfulAPI-сервером.Розробка системи проходила у середовищі PhpShtorm.

Було проведено проектування системи, де були сформовані функціональні та нефункціональні вимоги, побудована діаграма прецедентів, яка ілюструє ці вимоги, розглянуті питання обмеження архітектури системи та спроектований інтерфейс користувача. Також були сформовані представлення про рівні, дані, інтерфейси та процеси програмної системи. Спроектована база даних, розглянуті питання доставки продукту, описаний стек технологій, що задіяний у розробці даної системи.

На стадії реалізації була розроблена структура проекту, діаграма класів. Також були розглянуті питання управління програмним кодом з використанням системи контролю версій Git та її веб-оболонки GitHub. Розраховані метрики програмного коду та оформлений контрольний список по якості реалізації системи. Розроблена документація, яка включає в себе інструкцію користувача з детальним оглядом усього функціоналу та опис ресурсів RESTfulAPI. Для створеної програмно-апаратної системи був розроблений тест-план, який включає в себе декілька видів тестування. Для проведення модульного тестування були створені модульні тести, які покривають 35% програмного коду, розроблені 84 тест-кейсів для функціонального тестування, які покривають всі функціональні вимоги до системи.

Також були розглянуті питання охорони праці програміста на стадії розробки ним програмно-апаратного комплексу. Були вирішені питання, що пов’язані з розробкою заходів та обиранням засобів, які забезпечують повну

безпеку при роботі з технікою, обладнанням, які також забезпечують здорову атмосферу робочого приміщення, зокрема питання оптимізації параметрів мікроклімату у приміщенні з ПК та вибір кондиціонера для забезпечення необхідних кліматичних умов.

Таким чином, мета та усі поставлені до даної роботи завдання були успішно виконані.

ПЕРЕЛІК ПОСИЛАНЬ

1. Форум комп'ютерної допомоги, [Електронний ресурс] – Режим доступу: http://konstantinfirst.com/forum;

2.Форум дизайн, [Електронний ресурс] – Режим доступу: http://forumdts.org/;

3.Одеський форум, [Електронний ресурс] – Режим доступу: https://forumodua.com;

4.Фреймворк [Электронний ресурс] – Режим доступу:https://ru.wikipedia.org/wiki/Фреймворк

5. Каркас веб-приложений [Электронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Каркас_веб-приложений

6. Какие задачи возникают в результате отказа от использования веб-фреймворков?[Электронныйресурс]/П.Волынцев;Блоговеб-разработке и веб-технологиях– Режим доступу: http://copist.ru/blog/2014/12/10/framework-less-todo-list/.

7. Зачем использовать фреймворк? [Электронний ресурс] / СЕОлетик. URL: http://paperplane.su/why-use-framework/

8.YiiPHPFramework: BestforWeb 2.0 Development [Электронний ресурс] / YiiSoftwareLLC. – Режим доступу: http://www.yiiframework.com/

9.Laravel – ThePHPFrameworkForWebArtisans [Электронний ресурс] / TaylorOtwell. – Режим доступу: https://laravel.com/,

10. The Best PHP Framework for 2015: SitePoint Survey Results [Электроннийресурс] / SitePoint Pty. Ltd– Режимдоступу: http://www.sitepoint.com/best-php- framework-2015-sitepoint-survey-results/

11. Search PHP repositories [Электронний ресурс] Режимдоступу:https://github.com/search?q=stars%3A%3E0&type=Repositories&ref=advsear ch&l=PHP

12.Model-View-Controller [Электронний ресурс] / Wikipedia. – Режим доступу:https://ru.wikipedia.org/wiki/Model-View-Controller

13.ArchitectureofLaravelApplications [Электронний ресурс] / Laravelbook. –

Режим доступу: http://laravelbook.com/laravel-architecture/

14. Знакомство с веб-сервером Apache [Электронний ресурс] / .hostinfo справочная информация и практические советы. Режим доступу: http://hostinfo.ru/articles/220

15.ApacheHTTPServer [Электронний ресурс] / Wikipedia. – Режим доступу: https://ru.wikipedia.org/wiki/Apache_HTTP_Server

16.HTTP сервераpache[Электронний ресурс] / ApacheDev.ru. – Режим доступу:http://apache2dev.ru/2006/03/12/the-apache-modeling-project-glava-3-chast- 1/#a3

17. База данных [Электронний ресурс] / Wikipedia. – Режим доступу: https://ru.wikipedia.org/wiki/База_данных

18. База данных: основные понятия [Электронний ресурс] / Wiki-учебник по веб-технологиям. – Режим доступу: http://www.webmasterwiki.ru/mysql

19. MySQL [Электронний ресурс] / Wikipedia. – Режим доступу: https://ru.wikipedia.org/wiki/MySQL.

20. Интегрированная среда разработки [Электронний ресурс] / Wikipedia. – Режимдоступу:https://ru.wikipedia.org/wiki/Интегрированная_среда_разработки.

21. Интегрированная среда разработки [Электронний ресурс] / Академик. – Режим доступу: http://dic.academic.ru/dic.nsf/ruwiki/940808

ДОДАТОК А ЛІСТИНГ ПРОГРАМНОГО КОДУ

Модель Thread.php

```
<?php

namespace Tests\Unit;

use App\Notifications\ThreadWasUpdated;
use Illuminate\Foundation\Testing\DatabaseMigrations;
use Illuminate\Support\Facades\Notification;
use Tests\TestCase;

class ThreadTest extends TestCase
{
use DatabaseMigrations;

protected $thread;

public function setUp()
    {
parent::setUp();

$this->thread = create('App\Thread');
    }

/** @test */
function a_thread_has_a_path()
    {
$thread = create('App\Thread');

$this->assertEquals(
"/threads/{$thread->channel->slug}/{$thread->slug}", $thread->path()
        );
    }

/** @test */
function a_thread_has_a_creator()
    {
$this->assertInstanceOf('App\User', $this->thread->creator);
    }

/** @test */
function a_thread_has_replies()
    {
$this->assertInstanceOf(
'Illuminate\Database\Eloquent\Collection', $this->thread->replies
);
    }

/** @test */
public function a_thread_can_add_a_reply()
    {
$this->thread->addReply([
'body' =>'Foobar',
'user_id' =>1
]);

$this->assertCount(1, $this->thread->replies);
    }

/** @test */
function a_thread_notifies_all_registered_subscribers_when_a_reply_is_added()
    {
```

```
        Notification::fake();

$this->signIn()
            ->thread
->subscribe()
            ->addReply([
'body' =>'Foobar',
'user_id' =>999
]);

        Notification::assertSentTo(auth()->user(), ThreadWasUpdated::class);
    }

/** @test */
function a_thread_belongs_to_a_channel()
    {
$thread = create('App\Thread');

$this->assertInstanceOf('App\Channel', $thread->channel);
    }

/** @test */
function a_thread_can_be_subscribed_to()
    {
$thread = create('App\Thread');

$thread->subscribe($userId = 1);

$this->assertEquals(
1,
$thread->subscriptions()->where('user_id', $userId)->count()
        );
    }

/** @test */
function a_thread_can_be_unsubscribed_from()
    {
$thread = create('App\Thread');

$thread->subscribe($userId = 1);

$thread->unsubscribe($userId);

$this->assertCount(0, $thread->subscriptions);
    }

/** @test */
function it_knows_if_the_authenticated_user_is_subscribed_to_it()
    {
$thread = create('App\Thread');

$this->signIn();

$this->assertFalse($thread->isSubscribedTo);

$thread->subscribe();

$this->assertTrue($thread->isSubscribedTo);
    }

/** @test */
function a_thread_can_check_if_the_authenticated_user_has_read_all_replies()
    {
$this->signIn();
```

```
$thread = create('App\Thread');

        tap(auth()->user(), function ($user) use ($thread) {
$this->assertTrue($thread->hasUpdatesFor($user));

$user->read($thread);

$this->assertFalse($thread->hasUpdatesFor($user));
        });
    }

/** @test */
function a_threads_body_is_sanitized_automatically()
    {
$thread = make('App\Thread', ['body' =>'<script>alert("bad")</script><p>This is
okay.</p>']);

$this->assertEquals("<p>This is okay.</p>", $thread->body);
    }
}
```

ДОДАТОК Б МАРШРУТИRESTAPI

```
<?php


Route::get('/', function () {
return view('welcome');
});

Auth::routes();

Route::get('/home', 'HomeController@index');

Route::get('threads', 'ThreadsController@index')->name('threads');
Route::get('threads/create', 'ThreadsController@create');
Route::get('threads/search', 'SearchController@show');
Route::get('threads/{channel}/{thread}', 'ThreadsController@show');
Route::patch('threads/{channel}/{thread}', 'ThreadsController@update');
Route::delete('threads/{channel}/{thread}', 'ThreadsController@destroy');
Route::post('threads', 'ThreadsController@store')->middleware('must-be-
confirmed');
Route::get('threads/{channel}', 'ThreadsController@index');

Route::post('locked-threads/{thread}', 'LockedThreadsController@store')-
>name('locked-threads.store')->middleware('admin');
Route::delete('locked-threads/{thread}', 'LockedThreadsController@destroy')-
>name('locked-threads.destroy')->middleware('admin');

Route::get('/threads/{channel}/{thread}/replies', 'RepliesController@index');
Route::post('/threads/{channel}/{thread}/replies', 'RepliesController@store');
Route::patch('/replies/{reply}', 'RepliesController@update');
Route::delete('/replies/{reply}', 'RepliesController@destroy')-
>name('replies.destroy');

Route::post('/replies/{reply}/best', 'BestRepliesController@store')->name('best-
replies.store');

Route::post('/threads/{channel}/{thread}/subscriptions',
'ThreadSubscriptionsController@store')->middleware('auth');
Route::delete('/threads/{channel}/{thread}/subscriptions',
'ThreadSubscriptionsController@destroy')->middleware('auth');

Route::post('/replies/{reply}/favorites', 'FavoritesController@store');
Route::delete('/replies/{reply}/favorites', 'FavoritesController@destroy');

Route::get('/profiles/{user}', 'ProfilesController@show')->name('profile');
Route::get('/profiles/{user}/notifications',
'UserNotificationsController@index');
Route::delete('/profiles/{user}/notifications/{notification}',
'UserNotificationsController@destroy');

Route::get('/register/confirm', 'Auth\RegisterConfirmationController@index')-
>name('register.confirm');

Route::get('api/users', 'Api\UsersController@index');
Route::post('api/users/{user}/avatar', 'Api\UserAvatarController@store')-
>middleware('auth')->name('avatar');
```