**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ**

**Кафедра математичного моделювання**

**КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА**

на тему: **«РОЗРОБКА СЕРВЕРУ ПЛАТФОРМИ INCUBE»**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Виконав: студент | | 2 | курсу, групи | | 8.51325 |
| спеціальності | | | |  | |
| 8.05010302 програмна інженерія | | | | | |
| (шифр і назва спеціальності) | | | | | |
| С.О Мова | | | | | |
| (ініціали та прізвище) | | | | | |
| Керівник | доцент, к.т.н. Чопоров С.В. | | | | |
|  | (посада, вчене звання, науковий ступінь, прізвище та ініціали) | | | | |
|  | | | | | |
| Рецензент |  | | | | |
|  | (посада, вчене звання, науковий ступінь, прізвище та ініціали) | | | | |
|  | | | | | |

Запоріжжя – 2016

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ** | | | | | |
| **ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ** | | | | | |
| Факультет | | математичний | | | |
| Кафедра | математичного моделювання | | | | |
| Рівень вищої освіти | | | | | магістр |
| Спеціальність | | | 8.5010302 інженерія програмного забезпечення | | |
|  | | | | (шифр і назва) | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ЗАТВЕРДЖУЮ**  Завідувач кафедри математичного моделювання, к.ф.-м.н., доцент | | | | | | | |
|  | | | | | Лісняк А.О. | |
| (підпис) | | | |  | | | |
|  | | | | | | | |
| « |  | » |  | | | 2016 р. | |

**ЗАВДАННЯ**

**НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ (СТУДЕНТЦІ)**

|  |
| --- |
| Мові Сергію Олексійовичу |

(прізвище, ім’я та по-батькові)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. Тема роботи (проекту) | Розробка серверу платформи InCube | | | | | | | | | | | | |
|  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | |
| керівник роботи (проекту) | | | Сергій Вікторович Чопоров, к.т.н. | | | | | | | | | | |
|  | | | (прізвище, ім’я та по-батькові, науковий ступінь, вчене звання) | | | | | | | | | | |
|  | | | | | | | | | | | | | |
| затверджені наказом ЗНУ від | | | | « |  | | » |  | 2016 року № |  | |
|  | | | | | | | | | | | | | |
| 2. Строк подання студентом роботи | | | | | |  | | | | | | | |
|  | | | | | | | | | | | | | |
| 3. Вихідні дані до роботи | | 1. Постановка задачі. | | | | | | | | | | | |
| 2. Перелік літератури. | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | |
| 4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) | | | | | | | | | | | | |  |
| 1. Постановка задачі. | | | | | | | | | | | | | |
| 2. Основні теоретичні відомості. | | | | | | | | | | | | | |
| 3. | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | |
| 5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень) | | | | | | | | | | |  | | |
| Презентація | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | |

6. Консультанти розділів роботи

|  |  |  |  |
| --- | --- | --- | --- |
| **Розділ** | **Прізвище, ініціали та посада консультанта** | **Підпис, дата** | |
| **завдання видав** | **завдання прийняв** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |
| --- | --- |
| 7. Дата видачі завдання |  |

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Назва етапів кваліфікаційної роботи** | **Строк виконання етапів роботи** | **Примітка** |
| 1. | Розробка плану роботи. |  |  |
|  |  |  |  |
| 2. | Збір вихідних даних. |  |  |
|  |  |  |  |
| 3. | Обробка методичних та теоретичних |  |  |
|  | джерел. |  |  |
|  |  |  |  |
| 4. | Розробка першого та другого розділу. |  |  |
|  |  |  |  |
| 5. | Розробка третього розділу. |  |  |
|  |  |  |  |
| 6. | Оформлення та нормоконтроль |  |  |
|  | кваліфікаційної роботи. |  |  |
|  |  |  |  |
| 7. | Захист кваліфікаційної роботи. |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  |  | С.О. Мова |
|  | (підпис) |  | (ініціали та прізвище) |
|  | | | |
| Керівник роботи |  |  | С.В. Чопоров |
|  | (підпис) |  | (ініціали та прізвище) |

**Нормоконтроль пройдено**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Нормоконтролер | |  |  | О.С. Козлова |
|  | (підпис) |  | (ініціали та прізвище) |

# РЕФЕРАТ

Кваліфікаційна робота бакалавра: 62 с., 26 рис., 21 джерело.

Об’єкт дослідження: технології для розробки веб-додатків, наукова та проектна діяльності.

Предмет дослідження: методи та засоби реалізації та супроводу веб-додатків, існуючі популярні інвестиційні платформи.

Мета роботи: розробка внутрішньої структури платформи InCube.

Методи дослідження: серверне програмне забезпечення Apache, мова програмування PHP, програмне забезпечення для підтримки баз даних, операційна система CentOS, існуючі фреймворки для розробки веб-додатків.

Результатом роботи є сайт відкритої регіональної платформи науково-виробничого партнерства InCube, внутрішня структура якого розроблена за допомогою фреймворку Larevel 5 з підтримкою PHP 7 та бази даних MySQL.

Продуктивність та ефективність розробленого програмного продукту залежить від апаратного та програмного оснащення серверу та від якості Інтернет трафіку провайдера.

Результатом роботи користується відділ проектної діяльності Запорізького національного університету.

APACHE, CENTOS, PHP, MYSQL, LARAVEL FRAMEWORK, YII FRAMEWORK, SYMFONY FRAMEWORK, WEB DEVELOPMENT.

# ABSTRACT

Кваліфікаційна робота бакалавра: 62 с., 26 рис., 21 джерело.

Об’єкт дослідження: веб-технології для розробки веб-додатків.

Предмет дослідження: методи та засоби реалізації та супровіду веб-додатків.

Мета роботи: розробка внутрішньої структури платформи InCube.

Методи дослідження: серверне програмне забезпечення Apache, мова програмування PHP, програмне забезпечення для підтримки баз даних, операційна система CentOS, існуючі фреймворки для розробки веб-додатків.

Результатом роботи є сайт відкритої регіональної платформи науково-виробничого партнерства «InCube», внутрішня структура якого розроблена за допомогою фреймворку Laravel 5 з підтримкою PHP 7 та бази даних MySQL.

Продуктивність та ефективність розробленого програмного продукту залежить від апаратного та програмного оснащення серверу та від якості Інтернет трафіку від провадера.

Результатом роботи користується відділ проектної діяльності Запорізького національного університету.

APACHE, CENTOS, PHP, MYSQL, LARAVEL FRAMEWORK, YII FRAMEWORK, SYMFONY FRAMEWORK, WEB DEVELOPMENT.

# ЗМІСТ

Завдання на кваліфікаційну роботу  [2](#_Toc468640488)

[Реферат 4](#_Toc468640492)

[Abstract 5](#_Toc468640493)

[Вступ 8](#_Toc468640496)

[1 Аналіз стану проблеми 10](#_Toc468640497)

[1.1 Основні поняття стартапу та його види 10](#_Toc468640498)

[1.2 Інтернет-платоформи 16](#_Toc468640504)

[1.3 Постановка задачі 22](#_Toc468640507)

[1.4 Технічне завдання 23](#_Toc468640508)

[2 Проектування платформи incube 25](#_Toc468640512)

[2.1 Загальна структура сайту 25](#_Toc468640513)

[2.2 Існуючі фреймворки для розробки сайтів 27](#_Toc468640514)

[2.3 Веб-сервери 37](#_Toc468640519)

[3 Розробка серверної частини платформи incube 47](#_Toc468640525)

[3.1 Підключення до БД 47](#_Toc468640526)

[3.2 Командний рядок Artisan 48](#_Toc468640527)

[3.3 Реалізація архітектури сайту 48](#_Toc468640528)

[3.4 Розробка ролей користувачів 50](#_Toc468640529)

[3.5 Розробка персонального кабінету користувачів сайту 56](#_Toc468640530)

[3.6 Розробка механізму демонстрації проектів 57](#_Toc468640531)

[Висновки 60](#_Toc468640532)

[Перелік посилань 61](#_Toc468640533)

**ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ**

|  |  |
| --- | --- |
| ОС | операційна система |
| СП | середовище програмування |
| БД | база даних |
| ІС | інформаційна система |
| ТЗ | технічне завдання |
| СУБД | система управління базами даних |

# ВСТУП

У наш час кожна велика організація, фірма, підприємства, державні установи, учбові заклади мають свої офіційні сайти, які носять характер послуг та інформаційний характер. Також зараз стало популярно стартувати свій бізнес чи проект (такі як стартап), кошти на який часто збирається на іноваційних платформах. У сучасній науці про бізнес існує безліч визначень стартапу.

Стартап – це компанія зі скороченим циклом виходу на ринок, створена максимально швидко за мінімальні кошти і, як правило, є інноваційною. Ще 30 років тому мало хто уявляв собі можливість створення швидкого, але дуже прибуткового бізнесу, цикл життя якого не перевищує й року. Створення власного стартапу – це відмінний спосіб заробити ім’я та гроші в світі.

Існують декілька найбільш ефективних інноваційних платформ за допомогою яких можна розмістити свій стартап:

* відкрита інноваційна платформа innoLab;
* перша регіональна платформа професійного успіху Смарт-Up;
* інвестиційна платформа Startup network;
* інвестиційна платформа Kickstarter;
* інвестиційна платформа для компаній Gust;
* тощо.

Актуальність даної роботи обумовлена тим, що при науково-виробничому партнерстві, не має системи, яка покликана об’єднати освіту, науку, владу і бізнес. Тобто зараз більшість інноваційних платформ є приватними і більшість з них орієнтована на збір коштів та інколи ще на збір команди для реалізації проекту. В більшості розвинених країн світу саме вищі навчальні заклади є ініціаторами та авторами інноваційних проектів, які потрібні місцевим установам, підприємствам, тощо. В Україні проектна діяльність в вищих навчальних закладах лише починається, де студент або викладач може працювати над своїм проектом одночасно або навіть об’єднувати своє навчання або наукову діяльність та отримувати за це кошти. І студент при закінчені навчання матиме змогу почати свій бізнес, чи матиме гарне портфоліо для отримання бажаної роботи.

Розробка регіональної платформи науково-виробничого партнерства InCube є колективною та розділена на три основних частини:

* розробка серверу платформи InCube;
* розробка тонкого клієнту платформи InCube;
* розробка адміністративних модулів платформи InCube.

Ця кваліфікаційна робота описує першу частину розробки.

Об’єкт дослідження *–* веб-технології для розробки веб-додатків,процес та методи наукової та проектної діяльностей.

Предмет дослідження – методи та засоби реалізації та супровіду веб-додатків, існуючі платформи для проектної діяльності.

Мета кваліфікаційної роботи: розробка внутрішньої частини регіональної платформи науково-виробничого партнерства InCube.

Для досягнення поставленої мети треба вирішити наступні завдання:

* розглянути поняття стартапів і бізнес моделей;
* дослідити вже існуючі платформи для проектної діяльності;
* підготовка проекту InCube. Розглянути основні положення та завдання;
* дослідити методи та засоби розробки та запуску веб-додатків;
* розробка внутрішньої структури сайту регіональної платформи науково-виробничого партнерства InCube;
* налаштування серверного програмного забезпечення для запуску платформи InCube.

# 1 АНАЛІЗ СТАНУ ПРОБЛЕМИ

## 1.1 Основні поняття стартапу та його види

## 1.1.1 Визначення стартапу

В останні роки таке явище в бізнесі, як стартап, стало не просто популярним, а, мабуть, і модним. Про стартап говорять на телебаченні, про нього видаються книги. Виникають нові тематичні ресурси в Інтернеті та не всім зрозуміло, чи є відмінності між стартапом і «новоспеченою» компанією. Чи має стартап якусь свою специфіку? Або цей термін служить для позначення будь-якого перспективного починання?

Слово «стартап» походить від англійського поняття start up – «запускати» і означає щойно створену або що ще знаходиться в процесі створення компанію [1].

Мається на увазі, що у цієї компанії є якась бізнес-ідея, яка потребує розвитку та просування, але її творці поки зайняті дослідженнями ринку і пошуком коштів для її реалізації.

Іноді стартапами називають компанії, які збираються запропонувати споживачам інноваційні товари і послуги, але в даний час знаходяться в процесі пошуку підходящих бізнес-технологій та фінансової підтримки [1]. Майбутнє таких компаній виглядає невизначеним.

Стартапом може вважатися будь-яка компанія, незалежно від її сфери діяльності. Втім, в деяких колах стартапами називають тільки починання у сфері високих технологій, Інтернет-бізнесі і «суміжних дисциплінах».

Поняття «стартап» виникло в 30-ті роки минулого століття в Америці. Саме тоді два студенти – Хьюлетт і Паккард – заснували крихітне підприємство і назвали його «start up». Виявилося, що у компанії велике майбутнє – в наші дні вона відома під ім’ям Hewlett-Packard, або HP [2].

Коли заходить розмова про стартапи, наші співвітчизники зазвичай уявляють собі інтернет-ресурси, що виділяються серед собі подібних. Найвідомішими та успішними стартапами називають хостинг відеофайлів Youtube, «народну енциклопедію» – Вікіпедію (без посилань на яку не обходиться жодна дискусія), а також найбільші соціальні мережі – в першу чергу Facebook і ВКонтакте.

В інших країнах поняття «стартап» тлумачиться трохи інакше. Найвиразніші приклади стартапів – це IT-корпорації «зі світовим ім’ям»: Microsoft, Apple, Google. Історія цих гігантів і справді починалася з малого – а саме, з невеликої групи однодумців, що горять бажанням втілити в життя свої ідеї.

## 1.1.2 Основні відмінності стартапів

Як ви, напевно, помітили, стартап – це, досить, розмите поняття. Тому слід виділити наступні його характерні риси (визнані багатьма підприємцями):

* курс на новий продукт. Найчастіше суть стартапу полягає в просуванні новаторських ідей або удосконалення старих. Завдяки цьому стартапи можуть конкурувати навіть з найпотужнішими корпораціями, які в більшості випадків не обтяжують себе введенням інновацій і продовжують пропонувати ті товари та послуги, які давно зарекомендували себе. Ось чому будь-який стартап – це стрибок у невідомість;
* в основі стартапу лежить цікава бізнес-ідея. Не секрет, що без ідеї не було б взагалі ніякого бізнесу. Однак, в стартапі ідея має особливу цінність. Якщо вона обіцяє перспективи, то може бути дуже і дуже дорогою. Для порівняння: в Інтернеті повнісінько стандартних ідей для відкриття власної справи, і вони абсолютно безкоштовні. Іноді до них навіть додаються бізнес-плани;
* творці стартапів молоді. Багато відомих успішні стартапів – дітища вчорашніх студентів. Згідно зі статистичними даними, середній вік стартапера – 25 років. Ось чому нові починання іноді з ноткою зневаги іменують «гаражним бізнесом»: молодим людям, ще не розбагатівшим, ніде зібратися для обговорення своїх ідей, окрім як в гаражах і на знімних квартирах;
* творці стартапів ставлять на карту все. Як правило, ініціатори стартапу вкладають всю душу в свою ідею. Щоб реалізувати її, людина готова ризикувати по-крупному. Автора стартапу не хвилює той факт, що ціна перемоги може виявитися занадто високою: якщо він зацікавлений в успіху, то зробить все можливе, щоб домогтися його;
* ініціативна група працює за ідею, а не заради прибутку. Складно розвивати якесь починання поодинці. Тому у створенні стартапу бере участь ціла команда на чолі з ідейним лідером. Кожен з членів цієї команди виконує певні функції: один може розробляти перспективну ідею, інший – планувати бізнес на базі цієї ідеї, третій – шукати постачальників, клієнтів, джерела фінансування. Вони чудово розуміють, що їм не вдасться досягти всього й одразу. Зате вони зацікавлені у своїй роботі і вірять у можливість отримання прибутку;
* проблеми з фінансуванням. Мабуть, нестача коштів для втілення ідеї в життя є головною характерною ознакою стартапу. У цьому немає нічого дивного, адже стартапи створюються захопленими молодими людьми, у яких є величезна кількість енергії, але немає грошей. Ось чому стартапи потребують інвесторів – людей, які зацікавляться свіжими ідеями і допоможуть реалізувати їх, вклавши свої кошти в ці проекти. Не в останню чергу інвестора приваблює і прибуток.

## 1.1.3 Види стартапів

Стартапи прийнято класифікувати за кількома ознаками – наприклад, за особливостями продукції і ринком збуту. Розрізняють такі види стартапів:

* «Успішні копії». У цю групу входять численні вітчизняні проекти, які в деякій мірі є клонами зарубіжних. У першу чергу це соціальні мережі. Як виявилося, такі копії дуже перспективні [1];
* «Агресивні прибульці». Ця група об’єднує стартапи, орієнтовані на захоплення якогось сегмента ринку і вигнання з нього конкуруючих компаній. Впровадження продукту на ринок може бути реалізовано за рахунок його виняткової цінової переваги [1];
* «Темні конячки». Так можна назвати стартапи, перспективи яких не ясні зважаючи на їх новаторства. З одного боку, творці цих стартапів йдуть на великий ризик, роблячи спробу просувати їх, а з іншого боку – у разі успіху їм гарантований колосальний прибуток [1].

## 1.1.4 Етапи розвитку стартапів

Можна виділити основні етапи розвитку більшості стартапів.

Пасивний етап, або Pre-Seed stage. На даному етапі відбувається пошук ідеї і розробка технічних способів її реалізації. Ініціативна група робить аналіз ринку, пише бізнес-план, формулює техзавдання. Далі йдуть: створення прототипу продукту, тестування його версій, вивчення попиту та пошук джерел фінансування. Якщо не вдається знайти інвестора, проект «затухає». На жаль, з більшістю стартапів саме це і трапляється [3].

Запуск, або Startup Stage. Отже, інвестор знайдений: продукт можна випускати на ринок. Опинившись в ринкових умовах, продукт повинен довести свою перевагу перед аналогами. Але на цьому етапі обійти конкурентів не так-то просто. Творцям стартапу слід проявити завзятість, продемонструвати творче мислення і ділову хватку. Саме зараз проект піддається найбільшому ризику. Якщо цільова аудиторія залишиться байдужою, його історія на тому і завершиться [3].

Зростання, або Growth Stage. Припустимо, проект вижив в конкурентній боротьбі. Продукт користується попитом і потихеньку захоплює ринкову нішу, на яку орієнтувалися його розробники. Настає час виходити на точку беззбитковості і приносити інвесторам деякий прибуток [3].

Розширення, або Expansion Stage. Цілі, відображені в бізнес-плані, нарешті досягнуті. Але компанія-розробник на цьому не зупиняється – вона продовжує просувати своє дітище на нових ринках. Її позиціям вже нічого не загрожує: вона пізнавана, її продукція користується стабільним попитом, доходи поступово зростають [3].

Вихід, або Exit Stage. Коли компанія досягає піку свого розвитку, інвестори, які здійснили фінансування проекту, відмовляються від своєї частки в цьому бізнесі і продають її великим гравцям. Цей крок приносить їм гарний прибуток. Можна сказати, заради цього моменту вони і вкладають гроші в перспективну справу. Втім, окремі інвестори зберігають свою частку і використовують її в якості джерела постійного доходу [3].

**1.1.5 Інтереси автора стартапу та інвестора**

І автор ідеї, і інвестор, що допомагає її втілити, рівною мірою зацікавлені у фінансовому успіху стартапу. У разі перемоги кожен з них отримає свою вигоду. Правда, при цьому інвестор ризикує більше. Але зате і прибуток його буде істотніше, ніж прибуток автора стартапу [4].

Інтерес розробника. Вкладаючи сили в стартап, розробник прагне втілити в життя свої ідеї, набути успіх і визнання, а також він просто займається улюбленою справою і сподівається, що його старання будуть винагороджені. У деяких випадках так і відбувається [4].

Інтерес інвестора. Інвестор, природно, сприймає стартап трохи інакше. Для нього головним є примноження капіталу. Перед ним два шляхи: вкласти кошти в бізнес, що розвивається, а потім продати свою частку, яка до цього моменту буде дорого коштувати, або залишити її собі і отримувати з неї пасивний дохід [4].

Автор стартапу і інвестор заздалегідь домовляються про те, яким чином вони розділять дохід від реалізації проекту. Як правило, інвестор отримує більшу частину, і це цілком закономірно, адже саме він забезпечує фінансову підтримку і піддається ризику у випадку невдачі. Розробник отримує в середньому від 5 до 10% прибутку, але якщо гра дійсно варта свічок, то ці гроші гарантують йому безбідне існування [4].

В даний час зацікавленість бізнесменів та інвесторів у створенні стартапів продовжує зростати – незважаючи на те, що перспективи компаній-новаторів складно назвати райдужними. Адже не більше 10% всіх проектів отримують гідний розвиток.

Протягом найближчих кількох років активно розвиватися наступні напрямки: фінансові послуги, екологічні технології, медичне обладнання, біофармацевтичні, медійні проекти, соціальні мережі, телекомунікації, напівпровідники, здоров’я, споживчі товари та послуги.

Як бачите, стартап – це не просто молода компанія. Стартап – це фірма, створена амбітними молодими людьми, які готові просувати свій проект за всяку ціну. Оскільки стартапери завжди відчувають нестачу фінансування, на допомогу їм приходять інвестори, залучені можливістю заробити. Без них проект не має шансів.

**1.2 Інтернет-платоформи**

**1.2.1 Поняття інтернет-платформи та їх види**

Інтернет-платформа – онлайн сервіс, платформа чи сайт, використаний для побудови та відображення інформації, ідей тощо. Подібні ресурси дозволяються користувачам в Інтернеті обмінюватися різноманітними   
речами [5].

Моделі взаємодії між суб’єктами електронної торгівлі реалізуються на певних інтернет-платформах. Для кожної моделі існує своя, найбільш зручне й оптимальне середовище. В даний час в мережі Інтернет утворилося кілька видів сайтів, які займаються інтернет-торгівлею:

Торговий майданчик (інвестиційні та інноваційні платформи) – це великий сайт або сукупність кількох сайтів, які об’єднує безліч незалежних один від одного продавців. Такий майданчик ідеально підходить для електронної торгівлі, яка реалізується відповідно моделі   
Business-to-business (B2B) [5].

Інтернет-магазин – це спеціалізований сайт, на якому продавець (власник даного магазину) пропонує свій товар споживачам. Як правило, кожен такий магазин спеціалізується на певному конкретному виді товарів (побутова техніка, сувенірна продукція, меблі та інше). Реалізується відповідно до моделі Business-to-consumer (В2С) [5].

Інтернет-аукціон (онлайновий аукціон) – це спеціалізований інтернет ресурс, створений для того, щоб звести між собою покупців та продавців по моделі взаємодії Consumer-to-Consumer (C2C). На такому сайті кожна фізична особа може купувати і продавати товари за коштами аукціону [5].

**1.2.2 Сучасні інвестиційні та інноваційні інтернет-платформи**

Можна виділити такі сучасні інвестиційні та інноваційні інтернет-платформи:

ІnnoLab – відкрита інноваційна навчальна платформа, що об’єднує зусилля студентів та науково-педагогічних працівників вищих навчальних закладів у підтримці інноваційної діяльності в Україні та Білорусії [6]. Інтерфейс платформи зображений на рисунку 1.1:



Рисунок 1.1 – Відкрита інноваційна навчальна платформа ІnnoLab

Смарт-Up – історія першої регіональної платформи професійного успіху Смарт-Up почалась з утворення команди однодумців, які керуються у житті принципами розвитку, позитивного мислення, відповідальності та необхідності власними силами вирішувати проблеми, не чекаючи змін «згори». У формі такої ініціативної групи ми ведемо діяльність з 2012 року, і за цей час встигли індивідуально та колективно реалізувати декілька десятків соціальних ініціатив.

Постійне бажання сприяти позитивним змінам у громаді та розуміння принципів дієвого підходу для досягнення мети стали основними поштовхами для оформлення нашої команди у якості громадської організації. Маючи статус офіційно зареєстрованої з 22 січня 2016 року, ми не тільки змогли розширити коло нашої діяльності, а й якісно покращили її за рахунок співпраці з іншими зацікавленими сторонами [7].

Підтвердженням професіоналізму стало отримання міні-гранту громадською організацією протягом першого місяця існування! Нині, наша діяльність не обмежується лише грантовою підтримкою. Ми радо долучаємось до цікавих і корисних ініціатив, виступаємо партнерами у регіональних і всеукраїнських програмах, та реалізуємо власні соціальні проекти, у дієвості та результативності яких впевнені на всі 100% [7]. Інтерфейс платформи зображений на рисунку 1.2:

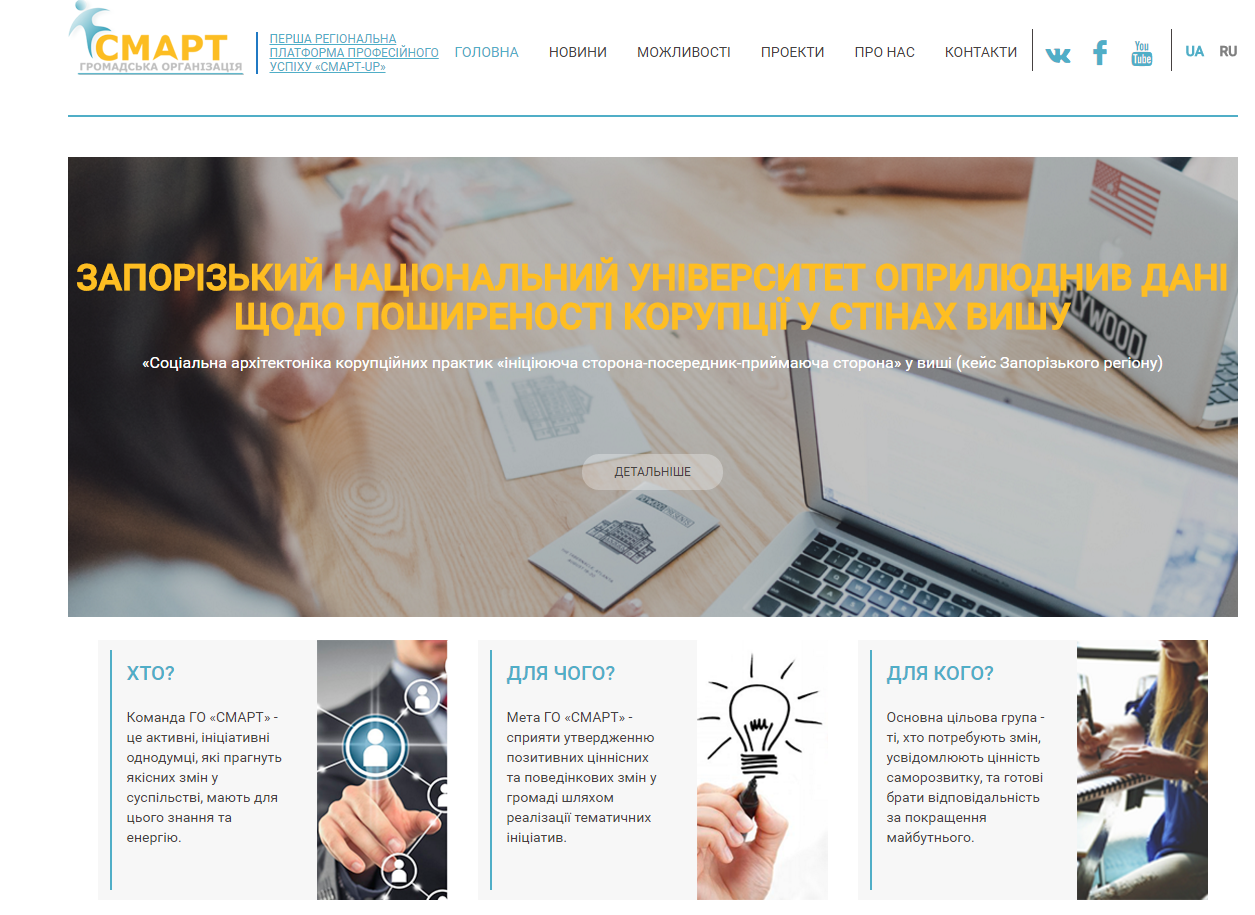


Рисунок 1.2 – Перша регіональна платформа професійного успіху Смарт-Up

Startup.Network – інвестиційна платформа Startup.Network допомагає інвесторам вибрати найкращі можливості для вкладення коштів; починаючим підприємцям, які хочуть реалізувати свій стартап, – знайти приватних інвесторів (бізнес-ангелів), сид-фонди (seed funds), венчурні фонди (venture funds) та інших інвесторів, які пропонують інвестиції в приватні компанії, професійним консультантам, які хотіли б турбуватись про інвестиційні консалтинги й продавати підприємцям свої знання, – знайти потужний потік проектів для відбору і подальшої співпраці [8]. Інтерфейс платформи зображений на рисунку 1.3:

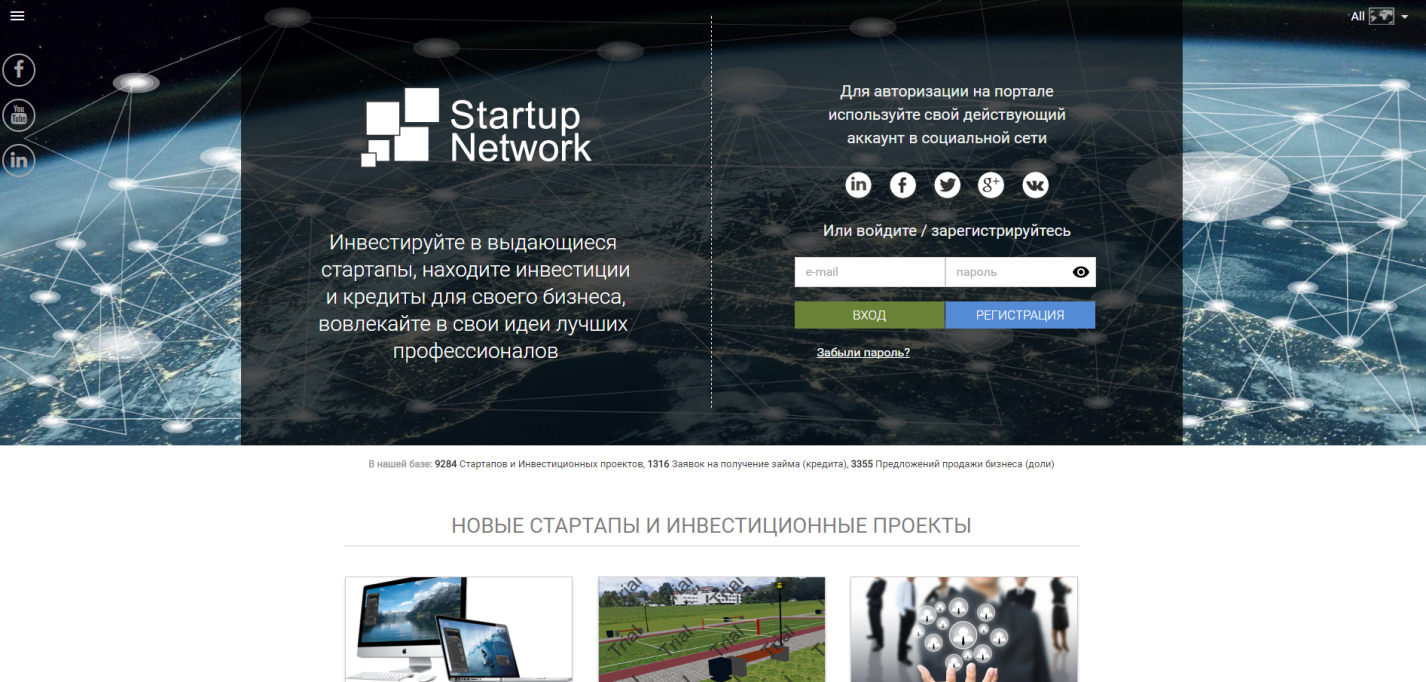


Рисунок 1.3 – Інвестиційна платформа StartupNetwork

Kickstarter – сайт фінансування творчих проектів за схемою краудфандингу. Краудфандинг (англ. Crowdfunding, crowd – «громада, гурт, юрба», funding – «фінансування»), тобто «Громадське фінансування», «Фінансування громадою» – це співпраця людей, які добровільно об’єднують свої гроші або інші ресурси разом, як правило через Інтернет, щоб підтримати зусилля інших людей або організацій. Фінансування за схемою краудфандингу може виконувати різні функції – допомога постраждалим від стихійних лих, підтримка з боку вболівальників чи фанатів, підтримка політичних кампаній, фінансування стартап-компаній та малого бізнесу, створення вільного програмного забезпечення і багато ін.. Kickstarter фінансує різноманітні проекти, у 13-ти категоріях: мистецтво, комікс, танець, дизайн, мода, фільми і відео, їжа, відеоігри, музика, фотографія, видавництво, технологія, театр [9].

Інтерфейс платформи зображений на рисунку 1.4:

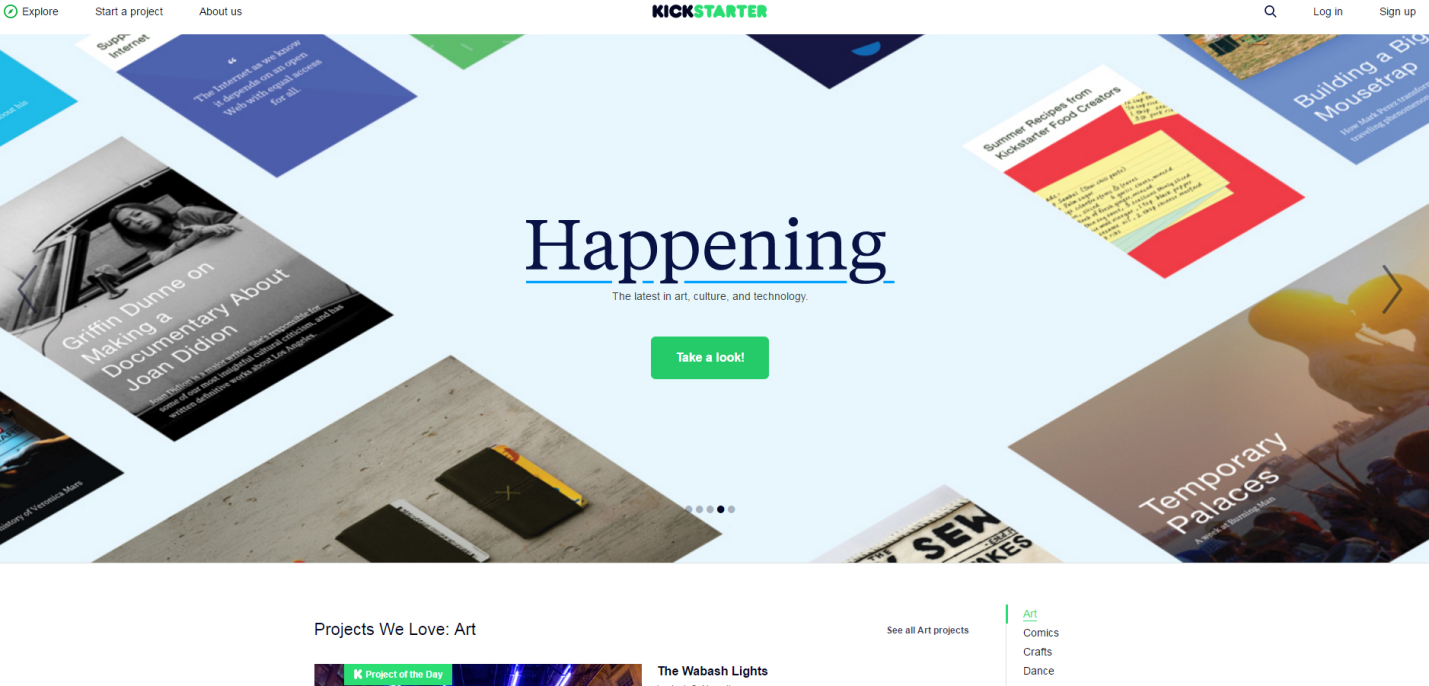


Рисунок 1.4 – Інвестиційна платформа Kickstarter

Gust – глобальна платформа для залучення інвестицій ранної стадії і управління ними [11]. Завдяки Gust професійні підприємці можуть співпрацювати з найуспішнішими інвесторами і отримувати підтримку практично з будь-яких питань взаємодії з інвесторами: від підготовки першої презентації до виходу інвестора з проекту. Gust рекомендують провідні світові асоціації бізнес-ангелів та венчурних фондів. На платформі Gust працюють понад 750 інвестиційних організацій в 65 країнах. Вже понад 125 000 початківців компаній зареєструвалися в системі, щоб встановити контакти і почати співпрацювати з більш ніж 45 000 акредитованих інвесторів. Gust – приватна компанія. Вона була заснована в 2004 р під назвою Angelsoft. Центральний офіс Gust розташований в м.Нью-Йорку, штат Нью-Йорк, центр розробок – у Ванкувері, Британська Колумбія, а європейський офіс – в Парижі, Франція [10]. Інтерфейс платформи зображений на рисунку 1.5:



Рисунок 1.5 – Інвестиційна платформа для компаній Gust

Вивчивши існуючі популярні інвестиційні платформи можна прийти до висновку, що усі вони мають дружній інтерфейс користувача та гарний дизайн. Але дослідивши зовнішній код сторінок цих платформ також можна прийти до висновку, що для розробки більшості платформ використовували програмне забезпечення для управління вмістом (з англ. Content Management Software – коротко CMS), наприклад платформа Startup.Network була розроблена та супроводжеється за допомогою CMS від 1С – Бітрікс, яка є не дешевим програмним забезпеченням.

## 1.3 Постановка задачі

Використовуючи існуючі веб-технології розробити відкриту регіональну платформу науково-виробничого процесу InCube.

Ініціатором (фактичним замовником) створення сайту платформи InCube став проректор з науково-педагогічної та навчальної роботи Запорізького національного університету, голова науково-методичної ради – Гура Олександр Іванович, який створив творчу групу для формування наступних основних характеристик та функціональних можливостей сайту:

* сайт передбачає 4 ролі користувача: замовник, інвестор, проектант та виконавець;
* користувач необмежений у виборі ролі та може бути одночасно будь-яким з вищезазначених ролей, але не в одному проекті;
* сайт повинен мати для кожної з ролей відповідну форму заявки для заповнення;
* мова сайту – українська, без можливості зміни, але передбачити майбутнє розширення локалізації сайту;
* у структурі сайту передбачити окрему сторінку для виводу усіх видів заявок;
* у структурі сайту передбачити сторінку з опублікованими новинами та іншою корисною інформацією;
* сайт повинен мати адаптивну верстку;
* створення особистого кабінету користувача;
* створена заявка будь-якої ролі користувача публікується на сайті тільки адміністраторами платформи;
* контактні дані у опублікованих заявках не публікуються;
* сайт передбачає функціональну можливість зв’язку між заявками різних типів. Тобто до опублікованої заявки замовника можливо прив’язати заявку інвестора чи проектанта. До заявки проектанта можливо прив’язати заявку інвестора чи подати резюме.

## 1.4 Технічне завдання

Найменування програмного засобу: Відкрита регіональна платформа науково-виробничого партнерства InCube.

Призначення і область застосування: інноваційна проектна діяльність у регіоні.

Загальний опис: сайт надає змогу публікувати проекти для знаходження інвесторів та інших учасників для реалізації чи впровадження цього проекту. Також сайт надає інформацію про існуючі гранти, конкурси, тощо для проектної діяльності.

## 1.4.1 Вимоги до сайту

Усі елементи сайту повинні коректно відображатись на будь-якому популярному браузері. Основними функціями сайту є:

* публікація проектів, завдань, інвестиційних заявок, резюме;
* створення груп для реалізації проектів;
* публікація інформації про існуючі гранти, конкурси, події, тощо, які зв’язані з проектною діяльністю.

Вимоги до функціональних характеристик. сайт повинен забезпечувати наступні властивості:

* гнучкість – користувач може діяти згідно будь-якої з чотирьох ролей;
* безпека – обмежити доступ до адміністративної частини сайту звичайним користувачам, обмежити перегляд окремих полів опублікованих заявок інших користувачів;
* безвідмовність – дії користувача будь-якого рівня знань не повинні призвести до критичних помилок роботи сайту;
* дружній інтерфейс – загальна структура сайту, його дизайн, та вигляд окремих елементів сайту повинні бути інтуїтивно зрозумілими для користувача.

## 1.4.2 Вимоги до надійності

Для надійного функціонування сайту потрібно забезпечити виконання наступних вимог:

* будь-які дії користувача не призведуть до критичних помилок на самому сайті та у роботі сервера;
* обмежений доступ до серверу(хосту) на якому функціонує сайт;
* систематичне резервне копіювання бази даних сайту;
* усі наявні паролі користувачів зберігати у зашифрованому вигляді.

## 1.4.3 Вимоги до експлуатації

Для повноцінного функціонування сайту платформи InCube потрібен сервер з серверною операційною системою, на якому повинен бути налаштовані SSH та FTP доступи, 20 гігабайт виділеної пам’яті, 1 гігабайт оперативної пам’яті, передвстановлені Apache, PHP, MySQL для сумісної роботи між собою. Також сервер повинен мати якісне інтернет-підключення.

# 2 ПРОЕКТУВАННЯ ПЛАТФОРМИ INCUBE

## 2.1 Загальна структура сайту

Як вказано за технічним завданням сайт повинен мати чотири ролі користувача:

* замовник – це користувач, який подає проблему(завдання) для вирішення;
* інвестор – це користувач, що хоче вкласти свої кошти у проект чи завдання;
* проектант – це користувач, що пропонує свій проект для вирішення певної проблеми та для пошуку інвесторів або учасників у проекті;
* виконавець – це користувач, що пропонує свої послуги у реалізації проекту. Це може бути хоч підприємець, що пропонує своє обладнання, або фахівець, який може вирішити частину завдань у проекті.

Вищезазначене означає, що кожна роль на сайті бути мати свою сторінку на сайті і кожна така сторінка містить запропоновані пропозиції для відповідної ролі і можливість подати пропозицію, не прив’язану до жодної з інших.

Початком проектування є побудова загальної структури сайту платформи InCube. На рисунку 2.1 зображено схему загальної структури сайту платформи:

За рисунком 2.1 як і на всіх сайтах є головна сторінка, на якій буде представлена загальна статична інформація та динамічна інформація з десятьма новішими опублікованими пропозиціями, та новини сайту.

Також сайт має форму реєстрації та входу. Окремо на сайті за структурою є сторінка з усіма опублікованими пропозиціями, розділених засобами верстки на окремі категорії. Кожна пропозиція окрім резюме має змогу надати свої пропозиції до інших пропозицій.

Головна сторінка

Статичні сторінки

Створити/Редагувати/Видалити

пости

Вхід / Реєстрація

Усі пропозиції

Замовник

Інвестор

Скинути пароль

Проектант

Виконавець

Блог

Особистий кабінет

Зміна ос. даних

Адмін. частина

Усі подані пр.

Продивитись пропозиції

Подати заявку

Подати заявку

Продивитись завдання

Подати заявку

Продивитись з. / пр.

Подати заявку

Продивитись пр. / інв. З.

Подати заявку

Продивитись з./пр.

Оповіщення

Продивитись пости

Створити/Редагувати/Видалити

Створити/Редагувати/Видалити

Рисунок 2.1 Загальна структура сайту платформи InCube

Кожний користувач має свій особистий кабінет, де відображаються його подані пропозиції, особиста інформація (яку можна змінити) та оповіщення.

Під статичними сторінками потрібно розуміти сторінки, на яких рідко змінюється інформація, наприклад сторінка з контактними даними, загальною інформацією про діяльність платформи, тощо.

## 2.2 Існуючі фреймворки для розробки сайтів

Фреймворк (Framework англ. Каркас) - це фундамент, який визначає архітектуру майбутнього програми і містить у собі налагоджений код, для вирішення часто використовуваних завдань веб-розробника, таких як робота з формами, базою даних, шаблонами, тощо [11]. На базі фреймворка можна з нуля написати свою CMS, розробити якийсь сервіс або веб-додаток.

До плюсів фреймворків можна віднести наступне:

* якісний матеріал – фреймворки написані розробниками для розробників, що дозволяє мати на виході прекрасно написаний код і своєчасне виправлення помилок;
* гнучкість і масштабування – завжди має гнучке рішення нестандартних завдань і можливість майбутнього розширення функціоналу шляхом підключення сторонніх бібліотек або окремих класів;
* продуктивність – швидше тільки чистий PHP, який як правило далеко не безпечний;
* безпека – забираються всі проломи в безпеці, практично немає вузьких місць для SQL-ін’єкцій;
* затребуваність – класних програмістів зі знанням якогось php-фреймворка одиниці і вони завжди затребувані.

До мінусів фреймворків можна віднести наступне:

* похибка – фреймворки використовують не тільки професійні програмісти, але і ті, що тільки навчаються, тому створений проект не завжди буде написаний грамотно;
* важко обслуговувати – якщо проект створював один розробник, а потім з якихось причин він пропадає або просто відмовляється супроводжувати створений ним проект, то його подальший розвиток або обслуговування стає більш складним завданням і часто не вигідним заняттям;
* не для кожного – якщо розробник зовсім не знайомий або слабо розбирається в PHP, тоді фреймворк не підійде для розробки;
* ціна розробки – вартість стандартного сайту зробленого на фреймворку з нуля буде дорожче, ніж на готової CMS, тому що часу на розробку витрачатися в кілька разів більше.

Популярними фреймворками вважаються:

* CodeIgniter;
* Yii;
* Laravel;
* Symphony;

## 2.2.1 CodeIgniter

CodeIgniter – PHP-фреймворк з десятирічним стажем і дуже простим процесом установки, що вимагає мінімальної конфігурації, так що почати буде просто [12]. Хороший вибір, якщо є ризик конфлікту різних PHP-версій: він ідеально працює практично на всіх платформах віртуального і виділеного хостингу.

Серед ключових можливостей цього продукту можна виділити:

* повна сумісність з PHP останньої версії 7;
* простий у вивченні – порівнюючи CodeIgniter з альтернативними рішеннями можна сміливо сказати, що він найбільш простий. Крім добре продуманої архітектури проекту, розробники забезпечили свій продукт відмінною документацією [12]. Документація написана докладно, добре структурована і оперативно оновлюється;
* безпека – проблеми в системі безпеки притаманні всім веб-проектам, проте з CodeIgniter багато з них вирішуються автоматично [12]. Наприклад, для точної фільтрації отриманих даних на предмет XSS є вбудована функція, яка дозволяє не тільки видалити, але і зберегти в лог небезпечні дані;
* можливість розширення – CodeIgniter дуже гнучкий і добре піддається розширенню [12]. Навчити фреймворк новим можливостям не займає багато часу – досить підключити потрібний пагін, хелпер, бібліотеку, тощо і почати працювати з новим функціоналом;
* аctive Record – В CodeIgniter використовується модифікована версія патерну Active Record Database [12]. За допомогою Active Record, робота з базою даних пришвидшується. Крім спрощеного доступу до БД, цей клас дозволяє не звертати уваги про використовувану СУБД – є можливість побудувати додаток, використовуючи MySQL, а потім також легко запустити його під Oracle. Також при складанні запитів все значення екрануються, що убезпечує роботу з СУБД.
* висока продуктивність – великі можливості не вплинули на продуктивність фреймворка, що зумовлено малими розміром ядра [12];
* архітектура MVC. CodeIgniter заточений під архітектуру MVC (Model-View-Controller), що дозволяє відокремити логіку від відображення [12]. При роботі в команді (програмісти та дизайнери), такий підхід є дуже затребуваний;
* підтримка шаблонізатора [12]. Незважаючи, на те, що CodeIgniter не вимагає обов’язкового використання шаблонізатора, при бажанні ним можна скористатися;
* дружність до пошукових систем [12]. Завдяки своїм URL, Code Igniter відмінно працює з пошуковими система і піддається індексації.

## 2.2.2 Yii

Yii – це високоефективний, заснований на компонентної структурі PHP-фреймворк для швидкої розробки великих веб-додатків. Він дозволяє максимально застосувати концепцію повторного використання коду і може істотно прискорити процес веб-розробки [13].

На Yii вплинули наступні CMF і CMS:

* Prado – узятий за основу в сенсі ідей. Це компонентна орієнтованість і події, модульна архітектура, загальний інтерфейс взаємодії з базою даних, інтернаціоналізація, локалізація і інші фічі і патерни;
* Ruby on Rails – спосіб побудови конфігураціі до додатка, принцип реалізації ORM;
* jQuery – це javascript бібліотека інтегрована в Yii. В тому сенсі, що все ж з Yii це серверне програмування, але віджети, форми та інші html частини згенеровані разом з прикріпленням jQuery. Зокрема це робиться для клієнтської валідації, для ajax валідації форм;
* Symfony – узятий принцип фільтрів і архітектура розширень;
* Joomla – взята модульна архітектура і інтернаціоналіція.

Про версії. Версія 1.0 вже не підтримується. Вимоги версії 1.1. \* – це PHP 5.1.0. і вище. Вимоги другої версії – це PHP 5.4 і вище.

Фреймворк має наступні якості:

* Yii один з найшвидших;
* простий у вивченні;
* структура проекту будується по паттерну MVC;
* прошарок з набором інтерфейсів для роботою з БД. Для роботи з базою даних є набори класів DAO і ActiveRecord. DAO – це набір класів для безпосереднього завдання запитів до бази даних. ActiveRecord – класи, що реалізують підхід ORM, де таблиця представляється класом, а записи таблиці є екзеплярами класу. Все (DAO і ActiveRecord) працює через PDO, звідси випливає великий список підтримуваних баз даних: SQLite, MySQL чи MariaDB, PostgreSQL, SQL Server, Oracle, Firebird чи Interbase 6 та інші. Варто зауважити, що через DAO задаються безпосередньо SQL запити, синтаксис яких може все таки відрізнятися для різних СУБД [13];
* є вбудована підтримка інтернаціоналізації. До речі, до місця чи ні буде сказано, але там так само є можливість побудови словоформ, як наприклад 1 гривня, 10 гривень [13];
* генерація базового коду PHP генератором Gii, який є модулем Yii, спочатку вбудованим. Можна сгенерувати модель, контролер або відразу модель, контролер і views для операцій CRUD на сутністю, таблицею. У Gii є шаблони генерації коду, які можна змінити, щоб файли генерувалися в тому вигляді, в якому ви хочете. Хочете щоб записи виводилися таблично або «плиткою» - робимо свій шаблон генеруємо. Якщо проект довгий, то можна замислитися складанням різних шаблонів конкретно для цього проекту, що дуже спростить життя і прискорить процес розробки [13];
* кешування сторінок і окремих фрагментів;
* підтримка тем оформлення та шаблонізаторів;
* можливість простого підключення сторонніх класів, бібліотек. При чому це робиться дуже просто. Можна підключити сторонню CRM, якщо вбудована не влаштовує.
* можна дуже легко підключити Yii до іншого проекту. Підключається файл, створюється екземпляр додатку і його можна використовувати. Можна користуватися Yii API через цей екземпляр додатку. Додатком в Yii є головний клас, який містить в собі всю інформацію, всі підключені компоненти;
* міграції бази даних – зміна структури та інформації в таблицях бази даних при роботі ве-додатку;
* підтримка REST;
* аутентифікація і авторизація. Розмежування доступу будується на основі ролей – RBAC. Будується ієрархія Роль – Завдання – Операція. Причому деревоподібна структура може бути великою. Роль присвоюється користувачу. Користувач може мати кілька ролей. І по цій ієрархії можна чи є зв’язок між операцією, завданням і користувачем. Тим самим перевірити доступність даного дії для користувача. Ця ієрархія може зберігатись як в базі, так і в файлі чи іншим чином. Авторизація виконана у вигляді фільтра до контролера. Фільтр в Yii – це код, який виконується перед виконанням дії контролера і вирішує, чи може виконатися дію контролера при поточних умовах. У контексті авторизації може поточний користувач виконати поточну операцію. До речі, є ще одна фішка - в цій ієрархії можна описати додаткові умови - бізнес-логіку на допущення / недопущення при інших рівних умовах, але при поточних вхідних даних, умови виконання. Наприклад - розрахований на багато користувачів блог і редагувати пости можна, але тільки господареві поста;
* перехоплення і обробка помилок;
* валідація вхідних даних, записуваних даних, що є додатковою безпекою;
* автоматичне тестування;
* є великі можливості по розширенню функціоналу - компоненти і поведінки, модулі, тощо;
* так само є хороші доповнення для безпеки. Є можливості для запобігання міжсайтового скриптинга, запобігання підробки міжсайтових запитів, запобігання атак через cookie;
* віджети, форми, таблиці та інші виведені на сайті об’єкти часто виводяться з прикріпленням jQuery для додаткового функціоналу клієнтської валідації, ajax валідації та іншої функціональності.

## 2.2.3 Symfony

Компоненти фреймворку Symfony використовуються в багатьох чудових проектах, наприклад CMS Drupal 8, phpBB і Laravel. Компоненти Symfony – це PHP-бібліотеки, які можна використовувати повторно і виконувати з їх допомогою безліч завдань: створення форм, конфігурацію об’єктів, маршрутизацію, аутентифікацію, створення шаблонів і багато чого ще [14]. Будь-який компонент можна встановити за допомогою Composer-менеджера пакетів для PHP.

Symfony працює на повторно використовуваних компонентах і забезпечує кращу модульність. Symfony також використовує модель і контролер для розробки веб-додатків, який може виглядати погано для багатьох нових розробників, але це працює [14]. Крім того, Symfony є хорошим прикладом модульного фреймворка.

Symfony працює з наступними базами даних:

* Microsoft BI;
* MongoDB;
* MySQL;
* NoSQL;
* PostgreSQL;
* CouchDB;
* DynamoDB;
* GemFire;
* GraphDB;
* MemBase;
* MemCacheDB;
* Oracle;
* Apache Jackrabbit.

## 2.2.4 Laravel

Laravel – безкоштовний, з відкритим кодом PHP-фреймворк, створений Тейлором Отвеллом і призначений для розробки веб-додатків відповідно до шаблону model – view – controller (MVC). Деякими з особливостей Laravel є модульна система упаковки з виділеним менеджером залежностей, різні способи для доступу до реляційних баз даних, утиліти, які допомагають в розгортанні додатків і технічного обслуговування, а також його орієнтація на синтаксичний цукор (англ. syntactic sugar – термін, що позначає доповнення синтаксису мови програмування, які не додають нових можливостей, а роблять використання мови більш зручним для людини) [15].

Станом на березень 2015 року, Laravel вважається одним з найпопулярніших PHP фреймворків, разом з Symfony2, CodeIgniter, Yii2 та іншими.

Ключові особливості, що лежать в основі архітектури Laravel:

* пакети (англ. Packages) – дозволяють створювати і підключати модулі в форматі Composer до додатка на Laravel. Багато додаткових можливостей вже доступні у вигляді модулів [15];
* eloquent ORM – реалізація шаблону проектування ActiveRecord на PHP. Дозволяє строго визначити відносини між об’єктами бази даних. Стандартний для Laravel будівник запитів Fluent підтримується ядром Eloquent [15];
* логіка додатка – частина додатка, що розроблюється оголошена або за допомогою контролерів, або маршрутів (функцій-замикань). Синтаксис оголошень схожий на синтаксис, який використовується в каркасі Sinatra [15];
* зворотня маршрутизація пов’язує між собою посилання і маршрути, що генеруються додатком, дозволяючи змінювати останні з автоматичним оновленням пов’язаних з ними посилань. При створенні посилань за допомогою іменованих маршрутів Laravel автоматично генерує   
  кінцеві URL [15];
* REST-контролери – додатковий шар для поділу логіки обробки GET і POST запитів протоколу HTTP [15];
* автозавантаження класів – механізм автоматичного завантаження класів PHP без необхідності підключення файлів та їх визначень в include. Завантаження на вимогу запобігає завантаження непотрібних компонентів – завантажуються тільки ті з них, які дійсно використовуються [15];
* укладачі уявлень (англ. View composers) – блоки коду, які виконуються при генерації уявлення (шаблону) [15];
* інверсія управління (англ. Inversion of Control) – дозволяє отримувати екземпляри об’єктів за принципом зворотного управління. Також може використовуватися для створення і отримання об’єктів-одинаків (англ. Singleton) 15];
* міграції – система управління версіями для баз даних. Дозволяє пов’язувати зміни в коді програми зі змінами, які потрібно внести в структуру БД, що спрощує розгортання і оновлення програми 15];
* модульне тестування (юніт-тести) – грає дуже велику роль в Laravel, який сам по собі містить велику кількість тестів для запобігання регресій (помилок внаслідок поновлення коду або виправлення інших помилок) [15];
* сторінковий висновок (англ. Pagination) – спрощує генерацію сторінок, замінюючи різні способи вирішення цієї задачі єдиним механізмом, вбудованим в Laravel [15];
* шаблонізатор Blade, що дозволяє швидке об’єднання роботи дизайнера (фронтенд) та розробника (бекенд) [15].

Laravel – перший фреймворк, що підтримує PSR-4. Прямо з коробки Composer автоматично завантажить всі класи з каталогу app, використовуючи стандарт автозавантаження PSR-4. Це означає, що у вас може бути один простір імен для вашого застосування, і ви можете структурувати його, як захочете і як вважаєте за логічним. Laravel не вимагає від вас розміщення певних файлів в певних папках.

У Laravel є чудова можливість, яка називається запитами форм. Це перевірка запитів для ваших контролерів. Це не просто перевірка даних, а повна обробка запиту.

Laravel Homestead – офіційна упакована Vagrant-"коробка", що надає середовище розробки, не вимагаючи установки PHP, HHVM, веб-сервера і будь-якого іншого ПЗ на локальну машину.

Це віртуальна машина з встановленим ПЗ, в яке входять:

* Ubuntu 14.04;
* PHP 5.6;
* HHVM;
* Nginx;
* MySQL;
* Postgres;
* Node (з Bower, Grunt і Gulp);
* Redis;
* Memcached;
* Beanstalkd;
* Laravel Envoy;
* Fabric + розширення HipChat.

Група розробників платформи InCube, вивчивши наявні фреймворки, вирішили використовувати Laravel 5. Однією з причин вибору цього фреймворку було отримання додаткового досвіду роботи з ним, вбудований шаблонізатор Blade та легка організація маршрутизації.

## 2.3 Веб-сервери

Під веб-сервером розуміємо ПЗ, що забезпечить розробку та роботу сайту платформи InCube та які працюють з протоколом HTTP.

## 2.3.1 Apache HTTP-сервер

Apache HTTP-сервер – відкритий веб-сервер Інтернет для UNIX-подібних, Microsoft Windows, Novell NetWare та інших   
операційних систем [16].

Apache розроблюється та підтримується спільнотою розробників відкритого програмного забезпечення під керівництвом Apache Software Foundation.

В 1996 році Apache обійшов NCSA HTTPd із того часу є найбільш популярним веб-сервером у світі [16].

Веб-сервер Apache є самостійним, некомерційним, вільно розповсюджуваним продуктом. Продукт підтримує безліч можливостей, багато з яких реалізовані як скомпільовані модулі, які розширюють основні функціональні можливості. Вони різняться від серверної підтримки мов програмування до схем аутентифікації. Існують інтерфейси для підтримки мов програмування Perl, Python, Tcl і PHP [16].

Популярні методи стискування на Apache включають зовнішній модуль mod\_gzip, створений для зменшення розміру веб-сторінок, переданих по HTTP.

Функції віртуального хостингу дозволяють одній інсталяції Apache обслуговувати різні веб-сайти. Наприклад, одна машина, з однією інсталяцією Apache може одночасно містити www.example.com, www.test.com, test47.test-server.test.com, тощо.

Apache передусім використовується для передачі через HTTP статичних та динамічних веб-сторінок у всесвітній павутині. Багато веб-застосунків спроектовано, зважаючи на середовище і можливості, які надає цей веб-сервер.

Продукт може працювати як кешувальний проксі-сервер, що дозволяє істотно підвищити продуктивність роботи користувачів локальної мережі при роботі з документами, розташованими в Інтернет. Можна задавати такі параметри і налаштування проксі-сервера:

* типи файлів, які необхідно кешувати або навпаки, не включати в кеш;
* максимальний обсяг дискового простору, відведений під кеш;
* періодичний перегляд і індексування бази даних кеша з метою вивільнення дискового простору шляхом видалення застарілих об’єктів.

Apache зіграв ключову роль у початковому зростанні всесвітньої павутини, і продовжує бути найпопулярнішим у світі веб-сервером, де-факто платформою, на яку орієнтуються інші веб-сервери.

Ядро Apache включає в себе основні функціональні можливості, такі як обробка конфігураційних файлів, протокол HTTP і система завантаження модулів. Ядро (на відміну від модулів) повністю розробляється Apache Software Foundation, без участі сторонніх програмістів.

Теоретично, ядро apache може функціонувати в чистому вигляді, без використання модулів. Однак, функціональність такого рішення вкрай обмежена [16].

Ядро Apache повністю написано на мові програмування C 16].

Система конфігурації Apache заснована на текстових конфігураційних файлах. Має три умовних рівня конфігурації:

* конфігурація сервера (httpd.conf).
* конфігурація віртуального хоста (httpd.conf з версії 2.2 extra/httpd-vhosts.conf).
* конфігурація рівня директорії (.htaccess).

Має власну мову конфігураційних файлів, засновану на блоках директив. Практично всі параметри ядра можуть бути змінені через конфігураційні файли. Більша частина модулів має власні параметри [16].

Частина модулів використовує в своїй роботі конфігураційні файли операційної системи (наприклад /etc/passwd і /etc/hosts).

Крім цього, параметри можуть бути задані через ключі командного рядка.

Для веб-сервера Apache існує безліч моделей симетричного мультипроцесування. Основні з них:

* worker (Apache Software Foundation) – гібридна мультипроцесорна-багатонитева модель. Зберігаючи стабільність мультипроцесорних рішень, вона дозволяє обслуговувати велику кількість клієнтів з мінімальним використанням ресурсів. Для середньо-навантажені веб-серверів. Для ОС: Linux, FreeBSD;
* pre-fork (Apache Software Foundation) – MPM, заснована на попередньому створенні окремих процесів, не використовує механізм threads. Для організації безпеки і стабільністі за рахунок ізоляції процесів один від одного, збереження сумісності зі старими бібліотеками, що не підтримують threads. Для ОС: Linux, FreeBSD;
* perchild (Apache Software Foundation) – гібридна модель, з фіксованою кількістю процесів. Для ОС Linux;
* netware (Apache Software Foundation) – багатонитева модель, оптимізована для роботи в середовищі NetWare. Для ОС: Novell NetWare;
* winnt (Apache Software Foundation) – багатонитева модель, створена для операційної системи Microsoft Windows.

## 2.3.2 Cherokee HTTP Server

Cherokee HTTP Server – вільний кросплатформовий веб-сервер, написаний на мові програмування С. Підтримує всі сучасні технології, включаючи FastCGI, SCGI, PHP, CGI, SSI, HTTPS (TLS і SSL), віртуальні хости, балансування навантаження і інші [17]. Розширюємо, завдяки підтримці плагінів. Основний упор при розробці робиться на високу продуктивність і швидкість роботи. Наявність веб-інтерфейсу дозволяє спростити настройку. У ряді тестів показує більш високу швидкість роботи, в порівнянні з такими серверами, як Lighttpd і Nginx [17].

Основний засіб конфігурації Cherokee – програма cherokee-admin, яка відкриває на час запуску адміністративний веб-інтерфейс (за замовчуванням за адресою 127.0.0.1:9090). Для входу в нього використовуються ім’я користувача та тимчасовий пароль, які виводить cherokee-admin.

Завдяки цьому інтерфейсу конфігурації Cherokee відмінно підходить для швидкого розгортання веб-сервера, наприклад для тестування веб-додатків (Cherokee йде з заготовками для Django, Ruby on Rails).

Конфігурація з командного рядка також підтримується, для цього використовується програма cherokee-tweak. Або можна вручну редагувати конфігураційний файл.

## 2.3.3 Internet Information Services

IIS (Internet Information Services, до версії 5.1 – Internet Information Server) – це набір серверів для декількох служб Інтернету від компанії Майкрософт. IIS поширюється з операційними системами родини Windows NT [18].

Основний компонент IIS – веб-сервер, який дозволяє розміщувати в Інтернеті сайти. IIS підтримує протоколи HTTP, HTTPS, FTP, POP3, SMTP, NNTP. IIS другий за популярністю веб-сервер за кількістю сайтів, після Apache HTTP Server.

Веб-сервер IIS працює з багатьма технологій створення веб-додатків:

* ASP.NET – розроблена Microsoft технологія; для IIS це - основне на сьогоднішній день засіб створення веб-додатків і веб-служб. IIS 6.0 поставляється разом з операційними системами, в які також спочатку входить .NET Framework, так що підтримка ASP.NET начебто вже вбудована в IIS 6.0; для більш ранніх версій необхідно окремо завантажити і встановити .NET Framework [18];
* ASP – передувала ASP.NET технологія створення динамічних веб-сторінок на основі сценаріїв. Входить в поставку IIS починаючи з версії 3.0 [18];
* CGI – стандартна межплатформенная низкоуровневая технологія створення динамічних веб-сторінок [18];
* FastCGI – клієнт-серверний протокол взаємодії веб-сервера і додатки [18];
* ISAPI – низькорінева технологія, аналогічна інтерфейсу модулів Apache, що надає повний доступ до всіх можливостей IIS, можливість розробки веб-додатків в машинному коді і можливість перевизначення частини функцій IIS і додавання до нього функцій, як пов’язаних з генерацією контента, так і не пов’язаних з цим . Підсистема виконання скриптів ASP і підсистема ASP.NET виконані як модулі ISAPI [18];
* SSI – включення в одні сторінки тексту з інших сторінок. Строго кажучи, веб-додатком не є, оскільки IIS підтримує лише обмежений набір можливостей і без того малофункціонального SSI. Зокрема, IIS5 підтримує тільки статичне включення і ігнорує команди умовного розгалуження [18].

Сам сервер підтримує тільки CGI, FastCGI, ISAPI і SSI. Всі інші технології є надбудовами, які працюють через CGI, FastCGI або ISAPI.

За допомогою CGI програми для IIS можуть розроблятися на основі практично будь-яких, в тому числі сторонніх, інструментів, що допускають запис в стандартний потік виведення і читання змінних середовища - Perl, C / С ++ і навіть засобами інтерпретатора командного рядка Cmd.exe.

Технологія ISAPI дозволяє, з одного боку, створювати спеціальні програми для IIS, що вимагають особливо тісної взаємодії з механізмом сервера, а з іншого боку, є зручною платформою для організації ефективної взаємодії IIS з іншими технологіями розробки веб-додатків – наприклад, PHP і Perl [18].

Починаючи з версії 6.0 служба WWW підтримує такі методи аутентифікації, тобто визначення особистості користувача по імені і паролю:

* анонімна аутентифікація (anonymous authentication) - визначення особистості користувача не виконується [18];
* базова аутентифікація (basic authentication) – ім’я та пароль передаються по мережі відкритим текстом [18];
* дайджест-аутентифікація (digest authentication) - пароль обробляється хеш-функцією перед відправкою по мережі, що унеможливлює його прочитання в разі перехоплення зловмисником [18];
* вбудована аутентифікація Windows (integrated Windows authentication) – виконується спроба входу на сервер з тими ж обліковими даними, під якими працює браузер користувача [18];
* аутентифікація для доступу до UNC-ресурсів (UNC authentication) – ім’я та пароль передаються віддаленим сервером на якому знаходиться опублікований в IIS UNC-ресурс, і віддалений сервер виконує аутентифікацію [18];
* аутентифікація з використанням .NET Passport (.NET Passport Authentication) (видалена в Windows Server 2008 і IIS 7.0) – для аутентифікації використовується служба .NET Passport [18];
* аутентифікація з використанням клієнтського сертифіката (certificate authentication) – для аутентифікації користувач повинен надати SSL-сертифікат [18].

## 2.3.4 NGINX

NGINX (engine x) – вільний веб-сервер і проксі-сервер. Є версії для сімейства Unix-подібних операційних систем (FreeBSD, GNU/Linux, Solaris, Mac OS X) та Microsoft Windows [19].

* + обслуговування статичних запитів, індексних файлів, автоматичне створення списку файлів, кеш дескрипторів відкритих файлів;
  + акселероване проксіювання з підтримкою кешування;
  + акселерована підтримка FastCGI і memcached серверів, простий розподіл навантаження і відмово стійкість;
  + модульність, фільтри, gzip, byte-ranges (докачка), chunked відповіді, HTTP-аутентифікація, SSI-фільтр;
  + вкладені запити на одній сторінці виконуються паралельно;
  + підтримка SSL;
  + експериментальна підтримка вбудованого Perl;
  + експериментальна підтримка HTTP/2;
  + перенаправлення користувача на IMAP/POP3-бекенд з використання зовнішнього HTTP-сервера аутентифікації;
  + проста аутентифікація (LOGIN, USER/PASS);
  + підтримка SSL і StartTLS.

В NGINX робочі процеси обслуговують одночасно безліч з’єднань, мультиплексуючи їх викликами операційної системи select, epoll (Linux) і kqueue (FreeBSD). Робочі процеси виконують цикл обробки подій від дескрипторів (див. подієво-орієнтоване програмування). Отримані від клієнта дані розбираються з допомогою кінцевого автомата. Розібраний запит послідовно обробляється ланцюжком модулів, що задається конфігурацією.

Відповідь клієнту формується в буферах, які зберігають дані або в пам’яті, або вказують на відрізок файлу. Буфери об’єднуються в ланцюжки, що визначають послідовність, в якій дані будуть передані клієнтові. Якщо операційна система підтримує ефективні операції вводу-виводу, такі як writev і sendfile, то NGINX застосовує їх при нагоді [19].

Віртуальні сервери. Конфігурація HTTP-сервера NGINX розділяється на віртуальні сервери (директива server). Віртуальні сервери поділяються на локації (location). Для віртуального сервера можливо задати адреси і порти, на яких будуть прийматися з’єднання, а також імена, які можуть включати \* для позначення довільній послідовності в перший і останній частині, або задаватися регулярним виразом. Локації можуть задаватися точним URI, частиною URI, або регулярним виразом. location’и можуть бути налаштовані для обслуговування запитів зі статичного файлу, проксування на http, fastcgi чи memcached сервер [19].

Управління пам’яттю. Для ефективного управління пам’яттю nginx використовує пули. Пул – це послідовність попередньо виділених блоків динамічної пам’яті. Довжина блоку змінюється в дапазоні від 1 до 16 кілобайт. Спочатку під пул виділяється лише один блок. Блок розділяється на зайняту область і незайняту. Виділення дрібних об’єктів виконується шляхом просування покажчика на незайняту область з урахуванням вирівнювання. Якщо незайнятої області в усіх блоках бракує для виділення нового об’єкта, то виділяється новий блок. Якщо розмір виділеного об’єкта перевищує значення константи NGX\_MAX\_ALLOC\_FROM\_POOL, або довжину блоку, то він повністю виділяється з купи [19].

Таким чином, дрібні об’єкти виділяються дуже швидко і мають накладні витрати тільки на вирівнювання.

Географічна класифікація клієнтів. NGINX містить модуль географічної класифікації клієнтів за IP-адресою. У його основу входить база даних відповідності IP-адрес географічному регіону, представлена у вигляді Radix tree (стиснуте префіксне дерево або стиснений бор) в оперативній пам’яті. nginx попередньо розподіляє перші кілька рівнів дерева, таким чином, щоб вони займали рівно 1 сторінку пам’яті. Це гарантує, що при пошуку IP-адреси для перших декількох вузлів при трансляції адреси завжди знайдеться запис в TLB.

## 2.3.4 Lighttpd

Lighttpd (також «lighty», «лайті») – веб-сервер, розроблений з розрахунком на швидкість, захищеність і відповідність стандартам. Це вільне програмне забезпечення, розповсюджується за ліцензією BSD. lighttpd працює в GNU/Linux та інших Unix-подібних операційних системах, а також у Microsoft Windows [20].

Основними можливостями є:

* підтримка стиснення віддаваного контенту «на льоту»;
* HTTP-аутентифікація;
* перезапис URL;
* SSL;
* автоматичний розподіл навантаження;
* візуалізація статистики за допомогою RRDtool.

Lighttpd підтримує інтерфейси CGI, SCGI, FastCGI і AJP. Це дозволяє використовувати веб-застосунки, написані практично будь-якою мовою програмування [20].

Проект lighttpd почався з бажання автора реалізувати веб-сервер, який зміг би витримати одночасно 10 тисяч з’єднань. lighttpd використовує так звану асинхронну обробку мережевих з’єднань. Завдяки цьому завантаженість сервера (на відміну від Apache) при доступі до файлів на диску не залежить від кількості запитів від користувачів [20].

У lighttpd можна використовувати особливі системні виклики для підвищення швидкості обробки файлів. При цьому задіюються специфічні для конкретної ОС виклики функцій ядра.

Найчастіше lighttpd (як і має схожу архітектуру nginx) використовується   
  
для віддачі статичного вмісту [20], в той час як його генерацією займається більш складний веб-сервер:

* Яндекс – використовує lighttpd в деяких своїх проектах: Карти, Маркет;
* Google – в своїх проектах, зокрема s.ytimg.com, який віддає усі зображення для youtube.com, в тому числі всі прев’ю для відео (в трьох дозволах) – використовує lighttpd-yt (модифікація від Google);
* Вікіпедія – використовує lighttpd як кешуючий проксі-сервер (для віддачі статичного вмісту);
* WOT – використовує lighttpd на своїх серверах;
* Ubuntu – використовує lighttpd на деяких серверів оновлення.

Так як фізично сайт платформи InCube повинен розташовуватись на віддаленому сервері для того, щоб його супроводжувати зі сторони серверу потрібно надати SHH та FTP доступи адміністратору. Тому саме так на рисунку 2.2 зображена загальна схема розгортання сайту.

Комп’ютер

Сервер

Потрібні засоби зв’язку та доступи: Інтренет підключення, SHH, FTP.

Рисунок 2.2 Схема розгортання сайту платформи InCube

# 3 РОЗРОБКА СЕРВЕРНОЇ ЧАСТИНИ ПЛАТФОРМИ INCUBE

Виконуючи дипломну роботу, мною було розроблено:

* бізнес-логіку Front-end частини сайту;
* серверну частину сайту Incube;

Означені вище роботи, проводились із використанням PHP Framework – Laravel 5.

## 3.1 Підключення до БД

Завдяки логічно побудованій архітектурі Laravel Framework, налаштувати проект дуже просто, адже все основні налаштування знаходяться у одному файлі - .env [21], зміст якого зображено на рисунку 3.1.

Так, для того, щоб встановити з’єднання із базою даних потрібно заповнити поля з приставкою DB\_ (див. рис. 3.1). БД – під’єднано.

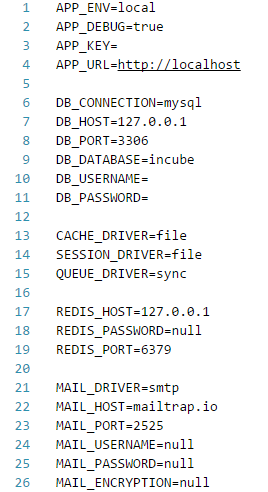


Рисунок 3.1 – Файл конфігурації .env

## 3.2 Командний рядок Artisan

Artisan – назва інтерфейсу командного рядка, що входить до складу Laravel. Він надає корисні команди для використання під час розробки програми. Працює на основі потужного компонента – Symfony Console [21].

Завдяки даному інтерфейсу процес створення нових функціональних частин проекту стає легше. Так, наприклад, для створення нового контролеру потрібно виконати команду, зображену на рисунку 3.2.



Рисунок 3.2 – Команда створення нового контролеру

## 3.3 Реалізація архітектури сайту

Під час проектування сайту, було прийнято рішення розділити сайт на дві основні частини:

* адміністративну (частина для управління та модерування інформацією, яку вносять користувачі сайту);
* клієнтську (частина, що дозволяє користувачам сайту вносити та переглядати інформацію про проекти та ін.).

Для реалізації даної архітектури потрібно розробити різні простори імен у додатку, а саме:

* admin;
* frontend.

Тобто, адресування сайту буде збудовано за схемою, показаною на рисунку 3.3. А саме, усі запити з простору імен із назвою “Frontend” будуть виконуватися за допомогою контролерів з даного простору імен, а запити “Admin”, відповідно своїми контролерами.



Рисунок 3.3 – Адресування сайту

Крім того, адміністративна частина захищена через механізм фільтрації запитів – Middleware.

HTTP Middleware (посередники) – це фільтри обробки   
HTTP-запиту [21].

Тобто механізм Middleware не пропустить до кабінету адміністратора користувача системи, що не має відповідної ролі.

Також Middleware забезпечує додавання особливих заголовків (наприклад, CORS http-відповідь програми) або логування всіх http-запитів.

У Laravel є кілька стандартних middleware. Це middlewares для реалізації режиму обслуговування сайту ("сайт тимчасово не працює, зайдіть пізніше"), здійснюють функції авторизації, CSRF-захисту тощо.

Враховуючи вищевказану архітектуру сайту, директива контролерів має вигляд зображений на рис. 3.4.

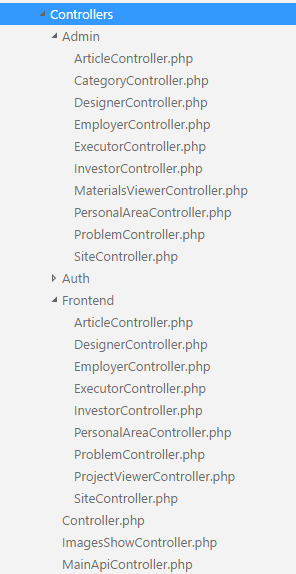


Рисунок 3.4 – Схема контроллерів сайту

## 3.4 Розробка ролей користувачів

Основними ролями користувачів сайту є:

– замовник;

– інвестор;

– проектант;

– виконавець.

Виходячи із вищевказаних даних потрібно розробити компактну та водночас універсальну структуру програмного коду.

Кожен із зареєстрованих користувачів сайту може створити певний тип документу, а саме:

* замовник – створює питання (проблему) для вирішення іншими учасниками проекту;
* інвестор – створює заявку на інвестування, тобто бажає вкласти свої кошти у проект чи завдання;
* проектант – пропонує свій проект для вирішення певної проблеми або для пошуку інвесторів та учасників у проекті;
* виконавець – пропонує свої послуги у реалізації проекту.

Для роботи з таблицями у БД Laravel використовує Eloquent.

ORM Eloquent – красива і проста реалізація патерну ActiveRecord для роботи з базою даних [21]. Кожна таблиця має відповідний клас-модель, який використовується для роботи з цією таблицею. Моделі дозволяють читати дані з таблиць і записувати дані в таблицю.

Створювати нові моделі можна за допомогою Artisan команди, що зображено на рисунку 3.5



Рисунок 3.5 – Команда створення нової моделі

Заявки, створені користувачами, зберігаються у одній таблиці в БД, її структуру показано на рисунку 3.6. Дана таблиця дозволяє зберігати усі заявки, створені користувачами, відповідно до ролі кожного з них, тобто саме ту інформацію, що безпосередньо відноситься до кожного типу, поданого документу.

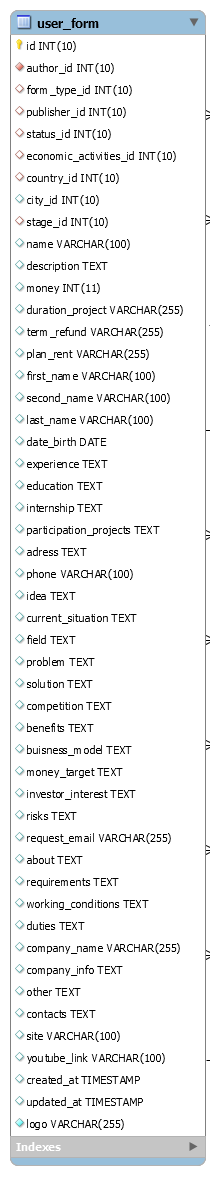


Рисунок 3.6 – Таблиця для зберігання заявок користувачів

А для розділення інформації, відповідно до ролі користувача, що її створює, було розроблено ще одну таблицю, що зображено на рисунку 3.7, а механізм зв’язку з таблицею заявок та іншими, показано на рис. 3.8.

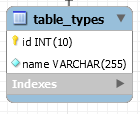


Рисунок 3.7 – Таблиця типів заявок користувачів

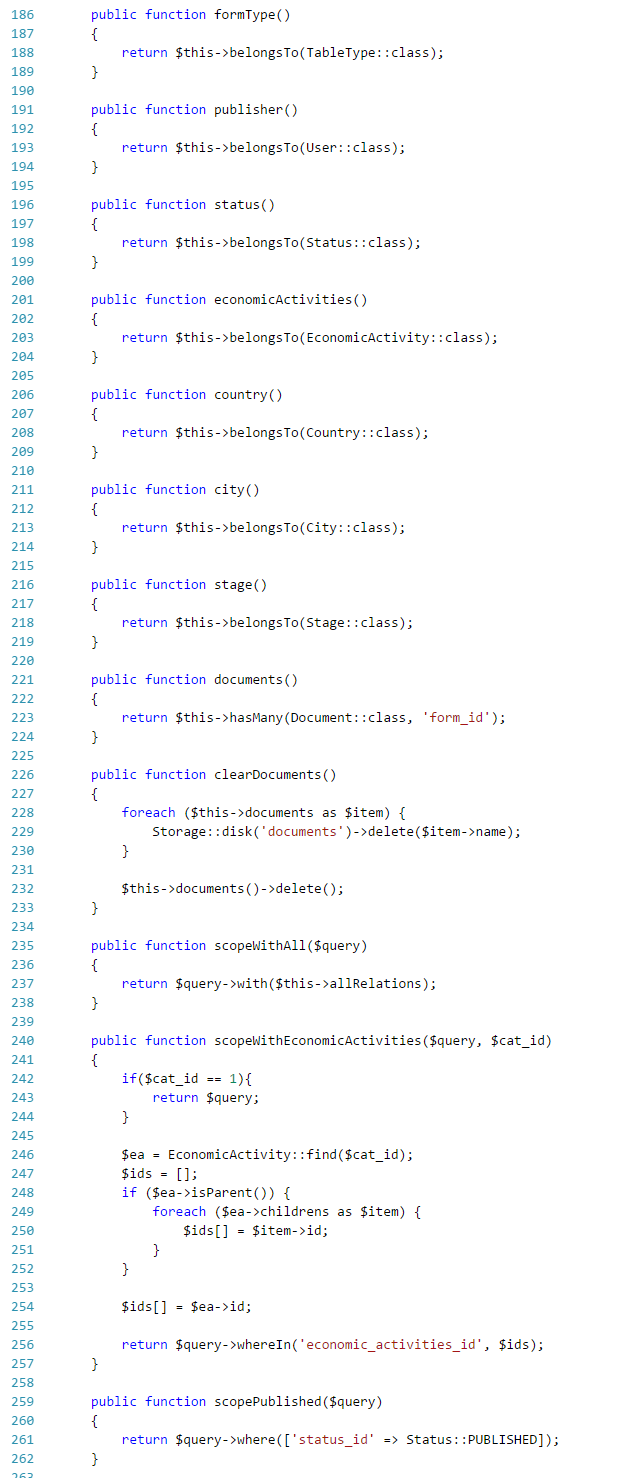


Рисунок 3.8 – Механізм зв’язку з таблицею заявок

На відміну від однієї, єдиної для всіх типів заявок таблиці, кожна роль користувача буде мати свій, окремий контролер, доступ до функцій якого забезпечується завдяки механізму маршрутизації (роутингу). Маршрути для даного контролеру показано на рисунку 3.9.

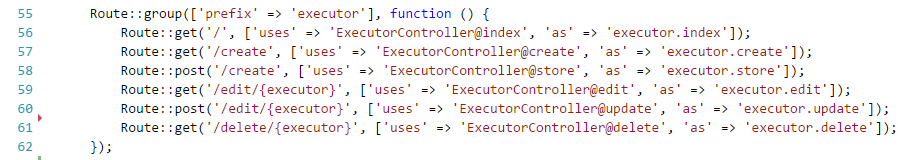


Рисунок 3.9 – Маршрутизація для контролеру виконавця

Виходячи з рисунку 3.9 ми можемо побачити основні складові контролеру – екшени, а саме:

* index;
* create;
* store;
* edit;
* update;
* delete.

Розглянемо структуру контролеру Виконавця та кожного його екшену быльш докладно.

На початку контролеру викликається конструктор даного класу, у якому задіяна вже відомий нам механізм middleware для перевірки ролі користувача, що використав цей контролер (див. рис. 3.10).

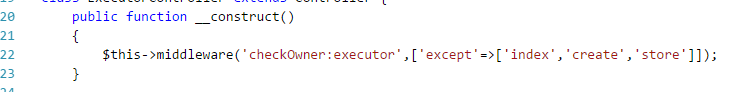


Рисунок 3.10 – Перевірка ролі користувача у конструкторі класу

Функціональне призначення index екшену – підготовка всіх потрібних даних та передача їх до виду, що забезпечує зображення головної сторінки даного розділу (див. рис. 3.11).

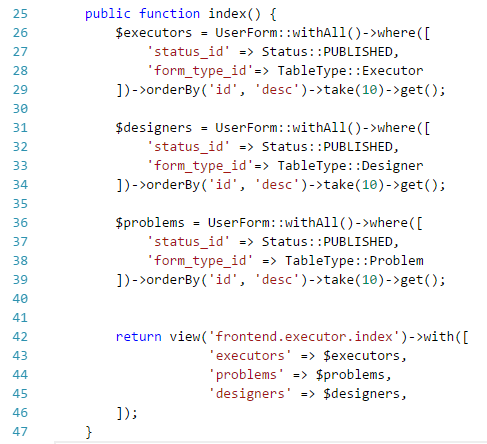


Рисунок 3.11 – Програмний код index екшену

Екшени create та edit, програмний код яких зображено на рис. 3.12, призначені для виводу форм створення та редагування нових заявок на виконання існуючих проблем.

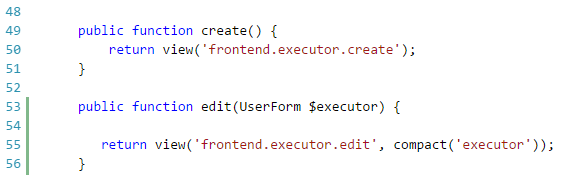


Рисунок 3.12 – Програмний код екшенів create та edit

Так само, як і вище згадані, екшени store та update мають однакове функціональне призначення, а саме – опрацювати та зберегти дані, отримані після створення або редагування заявки виконавця. Розглянемо згадані вище екшени на прикладі одного з них, а саме – store (див. рис. 3.13).



Рисунок 3.13 – Програмний код store екшену

Призначення останнього екшену у контролері – видаляти обрану заявку. Цей екшен називається delete та має наступний програмний код   
(див. рис. 3.14).

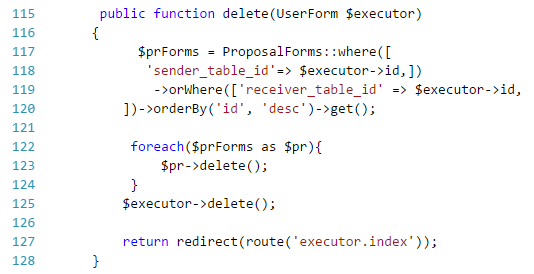


Рисунок 3.14 – Програмний код delete екшен

Отже, ми розглянули схему роботу контролеру. Подібну структуру має більшість контролерів даного сайту, як у клієнтській, так і у адміністративній частині.

## 3.5 Розробка персонального кабінету користувачів сайту

Персональний кабінет – це місце, де кожен зареєстрований користувач може керувати усіма матеріалами, що розмістив на сайті. Роутинг персонального кабінету користувача зображено на рис. 3.15.

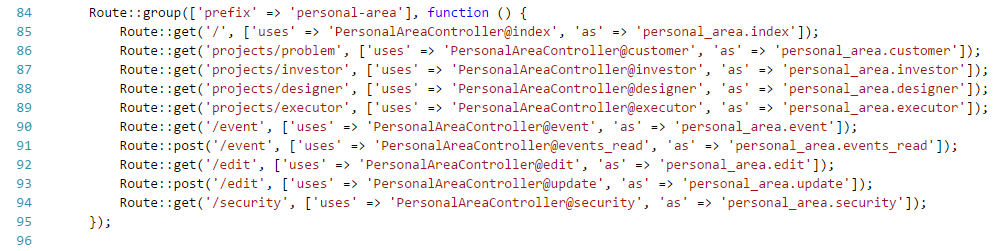


Рисунок 3.15 – Роутинг персонального кабінету користувача

Виходячи з цього можна виділити наступні екшени, програмний код яких показано на рис. 3.16, персонального кабінету:

* customer – виведення та редагування всіх питань (проблем) створених користувачем;
* investor – виведення та редагування всіх заявок на інвестування створених користувачем;
* designer – виведення та редагування всіх проектів створених користувачем;
* executor – виведення та редагування всіх резюме створених користувачем;
* event – виведення та менеджмент сповіщень, що надійшли користувачу;
* edit – редагування інформації про користувача;
* security – налаштування, зв’язані з безпекою профілю користувача (зміна паролю).

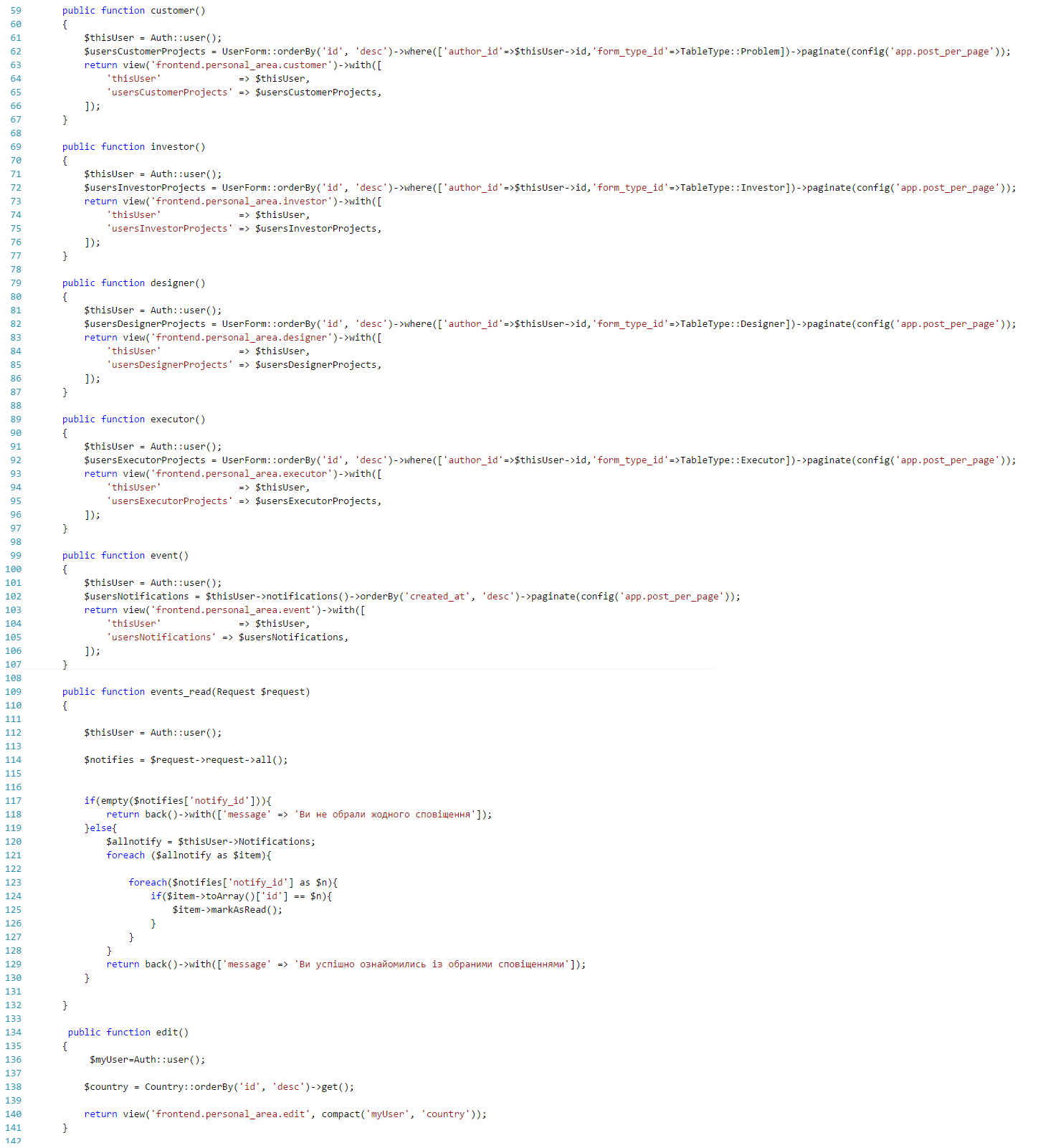


Рисунок 3.16 – Програмний код контролеру персонального кабінету

## 3.6 Розробка механізму демонстрації проектів

Роботою із усіма розміщеними на сайті проектами займається ProjectViewerController, програмний код якого показано на рисунку 3.17. Даний контролер виводить усі проекти обраної категорії, або повну інформацію про обраний проект.

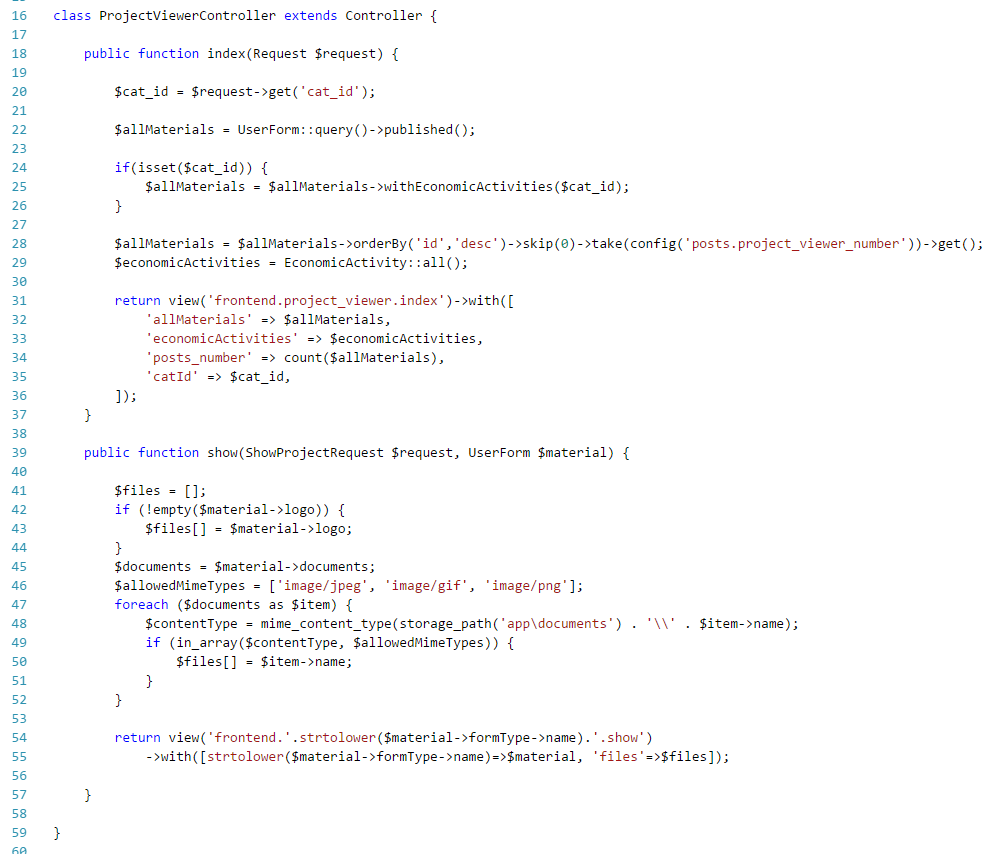


Рисунок 3.17 – Програмний код контролеру перегляду проектів

Крім того, існує механізм виводу матеріалів із вибраної категорії без перезавантаження сторінки. Програмний код даного механізму зображено на рисунку 3.18. Він працює наступним чином:

* відслідковуються натискання на кнопку зміни категорії матеріалів;
* запам’ятовуються всі потрібні параметри (категорія, номер сторінки);
* виконується Ajax-запит, що надає контролеру потрібні   
  дані (див. рис. 3.19);
* отримані дані обробляються і розміщуються на сторінці.

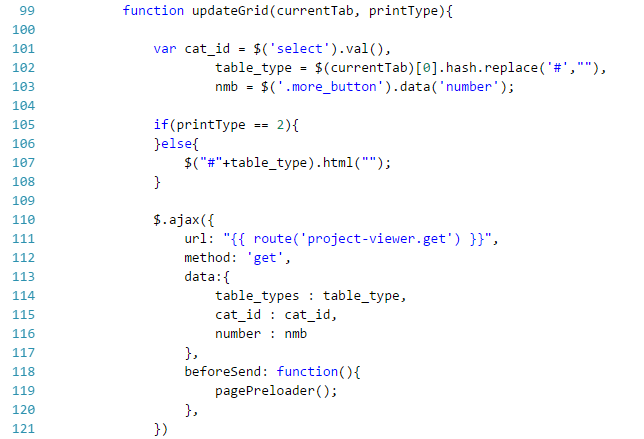


Рисунок 3.18 – Механізм виводу матеріалів сайту без перезавантаженя сторінки

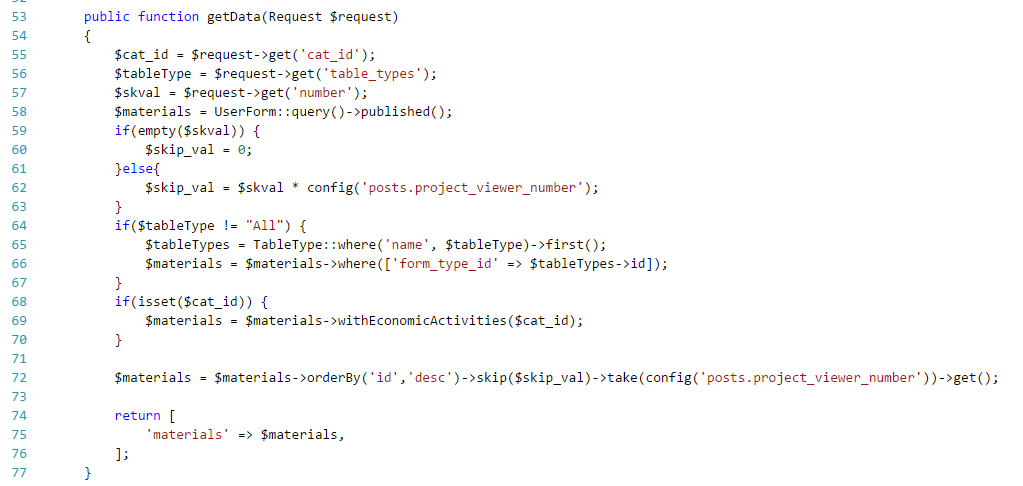


Рисунок 3.19 – Контролер, що обробляє Ajax-запити

# ВИСНОВКИ

У ході виконання дипломної роботи було проаналізовано поняття стартапу, дано визначення даного поняття, виділено основні його види. Також досліджено сучасні інвестиційні та інноваційні інтернет-платформи, їх сильні та слабкі сторони.

Після докладного аналізу предметної області було складено технічне завдання. Виходячи з якого, визначено структуру майбутнього сайту, розписано бізнес логіку frontend частини проекту, в результаті чого, стало можливим означити архітектурні особливості backend частини сайту.

Після розробки технічного завдання та архітектури сайту, мною були проаналізовані сучасні PHP фреймворки, та обраний один із них – Laravel 5. Виходячи із технічних вимог обраного фреймворку, також були проаналізовані сучасні популярні ВЕБ-сервери.

Завдяки використанню сучасного і потужного фреймворку, веб-сайт вдалося розробити швидким, інформативним, та відповідаючим усім вимогам, поставленим до нього у технічному завданні.

Підсумовуючи вищесказане, мною була проаналізована предметна область даного питання, знайдено та досліджено найпопулярніші існуючі інвестиційні платформи, розроблено технічне завдання та знайдено найкращі інструменти для розробки власного сайту.

Отже, у результаті виконання даної дипломної роботи було розроблено серверну частину платформи “Incube”.

# ПЕРЕЛІК ПОСИЛАНЬ

1. Стартап – Википедия [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Стартап – Название с экрана.
2. In and Around Language: What's Up with "Startup"? – [Электронный ресурс]. – Access mode: <http://www.thecrimson.com/article/2011/11/17/startup-language-idea/> – Title screen.
3. Startup = Growth – [Электронный ресурс]. – Access mode: http://www.paulgraham.com/growth.html – Title screen.
4. Патрик В. Стартап вокруг клиента. Как построить бизнес правильно с самого начала [Текст] / Патрик Власковиц, Брент Купер – Манн, 2013. –   
   21 с.
5. Майкл Х. Платформа. Как стать заметным в Интернете. Пошаговое руководство для всех, кому есть что сказать или что продать [Текст] / Майкл Х. – Манн, 2013. – 56 с.
6. Innolab Project | Innovation Laboratories – Іннолаб [Электронный ресурс]. – Режим доступа: http://www.innolabs.org – Название с экрана.
7. SAIUP - Проект сприяння академічній доброчесності в Україні – SAIUP [Электронный ресурс]. – Режим доступа: http://www.saiup.org.ua – Название с экрана.
8. Инвестиции. Инвестирование и инвестор. Венчурные инвестиции в стартапы – Startup.Network [Электронный ресурс]. – Режим доступа: https://startup.network – Название с экрана.
9. Kickstarter – [Электронный ресурс]. – Access mode: https://www.kickstarter.com – Title screen.
10. Gust | Startup Funding Investing – Gust [Электронный ресурс]. – Access mode: https://gust.com – Title screen.
11. Фреймворк – Википедия [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Фреймворк – Название с экрана.
12. Rob F. Codeigniter 2 Cookbook [Текст] / Rob Foster – Packt publishing, 2014. – 115 с.
13. Уайнсет Дж. Разработка веб-приложений в Yii 2 [Текст] / Уайнсет Дж., Сафронов М. – ДМК Пресс, 2015. – 46 с.
14. Sebastien A. Extending Symfony 2 Web Application Framework [Текст] / Sebastien Armand. – Packt publishing, 2013. – 128 с.
15. Terry M. Laravel Application Development Cookbook [Текст] / Terry Matulla. – Packt publishing, 2013. – 62 с.
16. Jessy R. Apache HTTP Server [Текст] / Jessy Russell. – Bookvika publishing, 2013. – 36 с.
17. Lambert M. Cherokee HTTP Server [Текст] / Lambert M. – BetaScript publishing, 2014. – 68 с.
18. Крис А. Администрирование сервера IIS 7 [Текст] / Крис Адамс. –Syngress, 2013. – 108 с.
19. Clement N. Nginx HTTP Server Second Edition [Текст] / Clement Nedelcu – Packt publishing, 2015. – 75 с.
20. Lighttpd – fly light [Электронный ресурс]. – Access mode: https://www.lighttpd.net – Title screen.
21. Laravel - The PHP Framework For Web Artisans [Электронный ресурс]. – Access mode: https://laravel.com/docs/5.3– Title screen.