

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ УПРАВЛІННЯ
ПОСТАВКАМИ МАТЕРІАЛЬНИХ РЕСУРСІВ ПРИ РЕАЛІЗАЦІЇ
ІНВЕСТИЦІЙНИХ ПРОЄКТІВ
IMPLEMENTATION OF THE MANAGEMENT METHODS FOR INVENTORY
SUPPLY IN INVESTMENT PROJECTS

Виконав: студент 4 курсу, групи КНТ-128сп
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Снігур М.М.

(прізвище та ініціали)

Керівник Дубровін В.І.

(прізвище та ініціали)

Рецензент Неласа Г.В.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 121 Інженерія програмного забезпечення
 (код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.

С.О. Субботін

“ ” 20 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Снігура Максима Михайловича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмна реалізація методів управління поставками матеріальних ресурсів при реалізації інвестиційних проєктів

Implementation of the management methods for inventory supply in investment projects

керівник проєкту (роботи) Дубровін Валерій Іванович, к.т.н., професор,
 (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “30” березня 2021 року № 103

2. Строк подання студентом проєкту (роботи) 29 травня 2021 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1.Опис предметної області. 2. Розробка структури програми. 3. Основні рішення щодо реалізації компонентів системи 4. Керівництво програміста. 5. Керівництво оператора.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконання завдання
1-5 Основна частина	Дубровін В.І., к.т.н., професор		
Нормоконтроль	Липовець М.В., асистент		

7. Дата видачі завдання “28” лютого 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Опис предметної області	1 тиждень	Завдання, ТЗ
2	Розробка структури сайту	2 тижні	Розділ 1
3	Основні рішення щодо реалізації компонентів системи	3-4 тиждень	Розділ 2
4	Методика роботи користувача з системою	5-6 тижні	Розділ 3
5	Тестування роботи вебсайту	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Додатки
7	Захист роботи	9 тиждень	

Студент

(підпис) Снігур М.М.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Дубровін В.І.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 60 с., 18 рис., 1 табл., 2 дод., 11 джерел.

PHP, ФРЕЙМБОРК, JAVASCRIPT, HTML, PHPStorm, REST API, РОЗМІТКИ ГІПЕРТЕКСТУ, CSS.

Об'єкт дослідження – управління ресурсами інвестиційних проєктів.

Предмет дослідження – програмне забезпечення для управління ресурсами інвестиційних проєктів.

Мета роботи – створення клієнт-серверного програмного забезпечення для полегшення управління ресурсами при інвестиціях.

Матеріали, методи та технічні засоби: клієнтська та серверна частина реалізована за допомогою фреймворку Yii 2, з використанням архітектури REST API. Відображення клієнтської частини зроблено за допомогою каскадних таблиць стилів CSS3, мови розмітки HTML5, мови програмування Javascript, бібліотеки JQuery, та фреймворку Bootstrap. Вся робота була виконана в інтегрованому середовищі розробки PHPStorm.

Результати. Створено клієнт-серверне програмне забезпечення, виконане у вигляді вебінтерфейсу з використанням PHP, REST API, HTML, CSS та мови JavaScript.

Висновки. Розроблено надійну серверну структуру, інтерфейс взаємодії з іншими програмними засобами через архітектуру REST API зручний інтерфейс користувача.

Галузь використання – інвестиції у біржовий ринок.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 60 pages, 18 figures, 1 tables, 2 appendixes, 11 sources.

PHP, FRAMEWORK, JAVASCRIPT, HTML, PHPStorm, REST API, HYPERTEXT MARKINGS, CSS.

The object of research - resource management of investment projects.

The subject of research - software for resource management of investment projects.

The purpose of the work is to create client-server software to facilitate resource management in investments.

Materials, methods and technical means: client and server part implemented using the Yii 2 framework, using the REST API architecture. The client part is displayed using cascading CSS3 style sheets, HTML5 markup language, javascript programming language, JQuery library, and Bootstrap framework. All work was performed in an integrated PHP Storm development environment.

Results. Client-server software created in the form of a web interface using PHP, REST API, HTML, CSS and JavaScript.

Conclusions. Developed a reliable server structure, interface for interaction with other software through the REST API architecture, user-friendly interface.

Area of use - investment in the stock market.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Опис предметної області	10
1.1 Основні поняття	10
1.2 Огляд існуючих програмних засобів	10
1.2.1 Сайт «alpari.com»	11
1.2.2 Сайт «amelok.com».....	12
1.2.3 Сайт «delta-capitals.com»	13
1.3 Постановка завдання роботи.....	15
2 Розробка структури програми.....	16
2.1 Архітектурний шаблон	16
2.2 Вебсервер	17
2.3 Мова програмування PHP	18
2.4 PHP фреймворк Yii2	19
2.5 Система керування базою даних mysql	20
2.6 Огляд особливостей середовища розробки phpstorm.....	22
2.7 Аналіз можливості, HTML, CSS та Bootstrap	23
2.8 Мова програмування javascript	24
2.9 Висновки	26
3 Основні рішення щодо реалізації компонентів системи.....	27
3.1 Загальна архітектура програми.....	27
3.2 Структура розробленої бази даних	27
3.3 Структура файлів проєкту.....	30
3.4 Висновки	32
4 Керівництво програміста.....	33
4.1 Призначення та умови застосування.....	33
4.2 Характеристики програми.....	34
4.2.1 Звертання до програми	34

4.2.2 Вхідні і вихідні дані	35
5 Керівництво оператора	36
5.1 Тестування клієнтської частини сайту.....	36
5.2 Звернення до програми.....	36
5.3 Висновки	42
Висновки	43
Перелік джерел посилання	44
Додаток А Текст програми.....	45
Додаток Б Слайди презентації	53

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

- API – Application programming interface;
- CSS – Cascading Style Sheets;
- DB – Data Base;
- JS – JavaScript;
- HTML – Hypertext Markup Language;
- ІК – інтерфейс користувача;
- ҮІІ – Yes It Is!;
- ООП – об'єктно-орієнтоване програмування;
- ПЗ – програмне забезпечення.

ВСТУП

Досвід економіки вказує на те, що інвестування є найважливішим джерелом економічного зростання та стабільності, а також фінансовою основою прогресу. Об'єктивний підхід до інвестицій припускає необхідність використання науково обґрунтованих механізмів управління, що забезпечують максимальне врахування діючих ризиків, аналіз ефективності реалізованих заходів і ухвалення оптимальних рішень при здійсненні інвестиційних проєктів [1].

Одним з характерних і найпоширеніших видів креативної діяльності в менеджменті, маркетингу, комерції, створення нових видів продукції, послуг і процесів є розробка і реалізація різних за призначенням і змісту інвестиційних проєктів.

Інвестиційне проєктування в даний момент є одним з найбільш ефективних способів вкладення капіталу в різні інвестиційні проєкти з метою його примноження. Завдяки комп'ютерним системам стало набагато зручніше ввести інформацію про свої інвестиції в електронному вигляді.

Реалізація інвестиційних процесів передбачає використання методів управління інвестиційними проєктами, тобто вимагає вирішення цілого комплексу спеціалізованих завдань.

Одним з основних і традиційних інструментів управління інвестиційними проєктами служить мережеве модерування ресурсами.

Метою дипломної роботи є створення програмного забезпечення для управління ресурсами інвестиційних проєктів. Програмне забезпечення повинно відповідати наступним вимогам:

- доступність;
- надійність
- швидкість;
- ефективність.

1 ОПИС ПРЕДМЕТОЇ ОБЛАСТІ

1.1 Основні поняття

Інвестиційний менеджмент - це процес управління всіма аспектами інвестиційної діяльності суб'єкта підприємництва [2]. Основна мета інвестиційного менеджменту забезпечити найефективнішу реалізацію інвестиційної стратегії суб'єктів підприємницької діяльності.

Інвестиційний менеджмент має виконувати певні завдання, які полягають у забезпеченні:

- високих темпів економічного розвитку фірми (підприємства) за рахунок ефективної інвестиційної діяльності;
- максимізації доходів (прибутку від інвестиційної діяльності);
- мінімізації інвестиційних ризиків.

Сервер - програмний компонент обчислювальної системи, що виконує сервісні (обслуговуючі) функції по запиту клієнта, надаючи йому доступ до певних ресурсів або послуг [3].

Інтерфейс користувача забезпечує взаємодію користувача з сервером за допомогою елементів інтерфейсу, які є зрозумілими для користувача. Через інтерфейс користувач сприймає всю програму в цілому, тому дуже важливим аспектом є легкий, зрозумілий та швидкий інтерфейс, який буде надавати користувачу максимальну кількість задоволення від взаємодії з інтерфейсом.

1.2 Огляд існуючих програмних засобів

На сьогодні в мережі Інтернет існує безліч ресурсів для менеджменту інвестицій у біржовій сфері. Розглянемо декілька аналогів:

- сайт «alpari.com»;
- сайт «amelok.com»;
- сайт «platform.delta-capitals.com».

1.2.1 Сайт «alpari.com»

Сайт «alpari.com» працює на Django (Python) та призначений для залучення користувачів для інвестицій в біржовий ринок. Даний сайт наведено на рисунку 1.1.

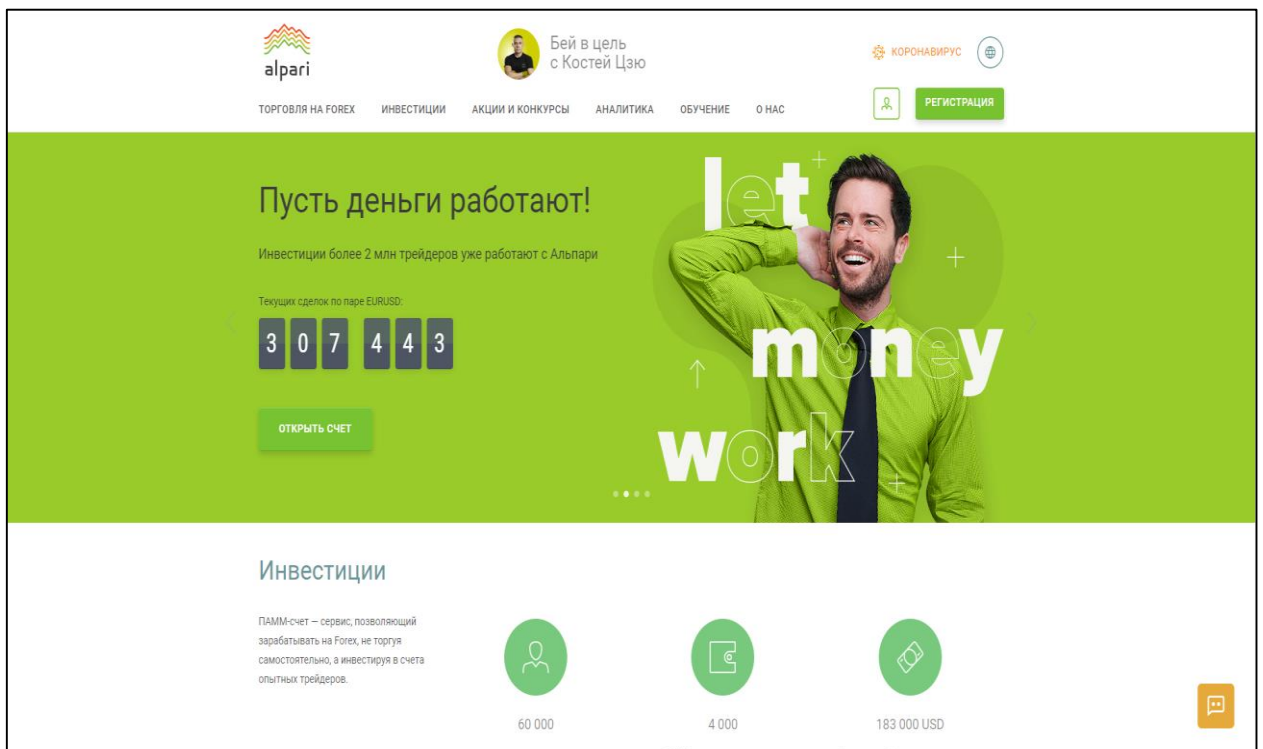


Рисунок 1.1 – Сайт alpari.com

Сайт включає в себе наступні сторінки:

- а) «Торговля на Forex» – сторінка, яка містить базову інформацію про інвестиції, біржу та платформи для трейдингу;
- б) «Инвестиции» – сторінка, яка містить інформацію про різні типи інвестицій, інвестиційні портфелі та мобільний застосунок;
- в) «Акции и конкурсы» – сторінка, яка містить точні акції;
- г) «Аналитика» – сторінка, яка містить інформацію про курси валют та акцій;
- д) «Обучение» – сторінка, яка містить відео та фото матеріали для навчання;

е) «О нас» – сторінка, яка містить юридичну інформацію про компанію;

Переваги:

- простий інтерфейс;
- інформативність сайту;
- вдало підібрані кольори сайту;
- багатий функціонал сайту.

Недоліки:

- немає локалізації;
- сайт реалізований на Django, що не надає достатньої надійності та стабільності при роботі з біржою.

На мій суб'єктивний погляд, саме цей сайт є найбільш невдалою реалізацією серед усіх прикладів, наведених нижче. Це обумовлено невдалим розташуванням елементів сайту, що негативно відображується на загальному користувацькому досвіді.

1.2.2 Сайт «amelok.com»

Сайт «amelok.com» виступає брокером клієнтів, які не мають досвіду у біржовій торгівлі, та працює на мові програмуванні PHP. Даний сайт наведено на рисунку 1.2.

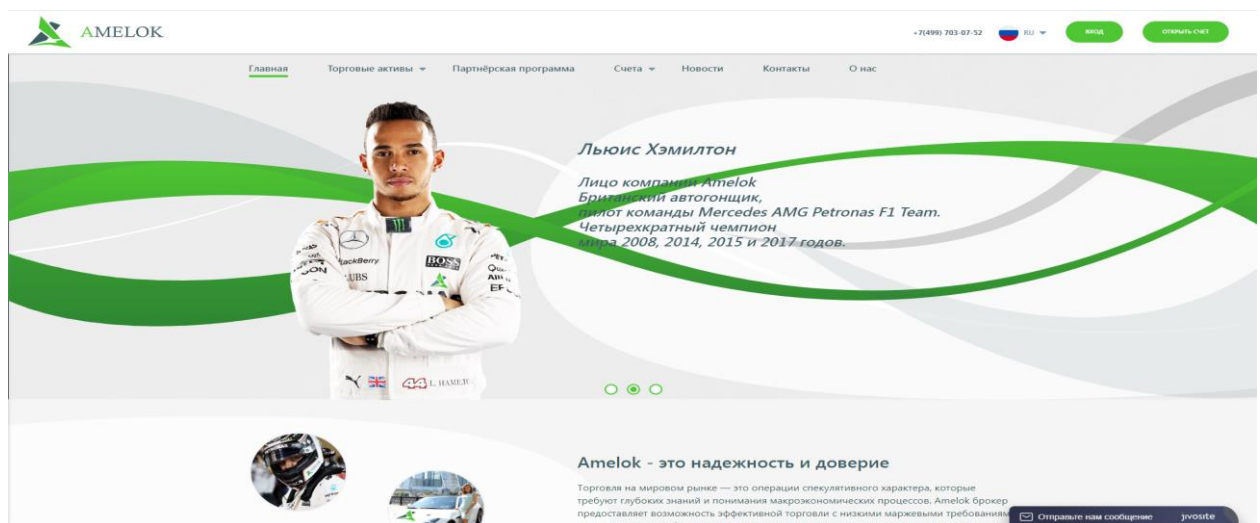


Рисунок 1.2 – Сайт amelok.com

Сайт включає в себе наступні сторінки:

- а) «Главная» – сторінка, яка містить докладну інформацію про компанію;
- б) «Торговые активы» – сторінка, яка містить інформацію про активи, з якими взаємодіє сайт;
- в) «Партнё́рская программа» – сторінка, яка містить дані про систему партнерів;
- г) «Счета» – сторінка, яка містить докладні данні про типи рахунків;
- д) «Новости» – сторінка, яка містить новини сайту;
- е) «Контакты» – сторінка, яка містить контактні данні компанії;
- ж) «О нас» – сторінка, яка містить юридичну інформацію про компанію.

Переваги:

- простота використання;
- багатомовність;
- зручна система навігації.

Недоліки:

- відсутність адміністративної панелі;
- багатомовність реалізована за допомогою Google Translate.

Я вважаю що цей приклад є найбільш вдалою версією реалізацію ідеї, бо сайт має чудовий інтерфейс, з яким можна легко і інтуїтивно взаємодіяти, а також через добре підібрану палітру кольорів та зважене розташування елементів.

1.2.3 Сайт «delta-capitals.com»

Сайт «delta-capitals.com» був розроблений за допомогою фреймворку Yii2 та React, які взаємодіють через REST API та сокети. Даний сайт наведено на рисунку 1.3.

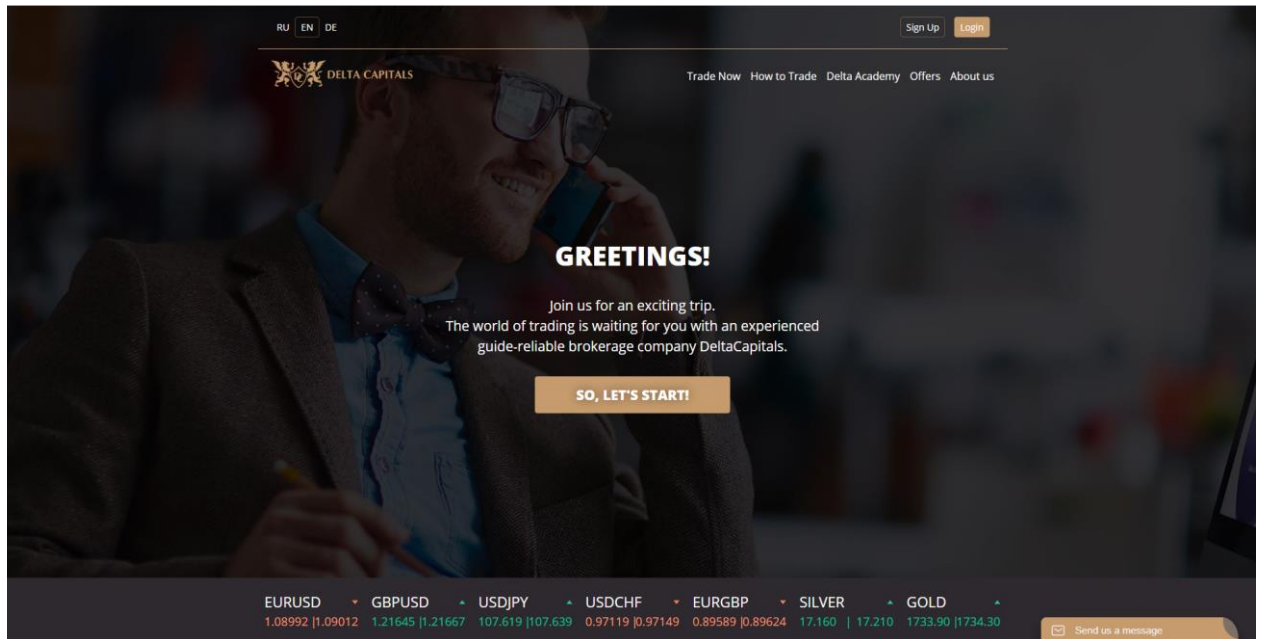


Рисунок 1.3 – Сайт delta-capitals.com

Сайт включає в себе наступні сторінки:

- а) «Главная» – сторінка, яка містить останні новини сервісу та базову інформацію про сайт;
- б) «Как торговать» – сторінка, яка містить докладну інформацію про типи рахунків та специфікації активів;
- в) «Академия Delta» – сторінка, яка містить вебінари по темі біржових інвестицій;
- г) «Спецпредложения» – сторінка, яка містить список усіх акцій;
- д) «О компании» – сторінка, яка містить докладну юридичну інформацію про компанію.

Переваги:

- зручний інтерфейс;
- сайт написаний на новітніх технологіях;
- багатомовність.

Недоліки:

- відсутність адміністративної панелі;
- незрозуміла система навігації.

Об'єктивно цей сайт має найбільш складні та сучасні технології у своїй структурі. Однак через це стабільність роботи сайту постраждала, що дуже важливо для будь-якого програмного продукту.

1.3 Постановка завдання роботи

Метою роботи є створення клієнт-серверного застосунку, що повинен надавати можливість інвестувати у різні біржові активи, відображати інформацію про відкриті рахунки та поточний баланс.

Інтерфейс повинен бути доступним, простим у використанні та надавати користувачу максимум позитивного користувацького досвіду.

Серверна частина повинна бути надійною, швидкою, стійкою до відмов та достатньо гнучкою, щоб витримувати велику кількість користувачів,

Інтерфейс взаємодії (API) повинен бути надійним та зрозумілим для взаємодії.

На основі проведеного аналізу різних сайтів була сформована постановка задачі для розробки даного проєкту.

Основним задачами програмного продукту є:

- реєстрація та аутентифікація користувачів;
- верифікація користувача;
- захист особистих даних користувача;
- створення інвестиційних рахунків для біржової торгівлі;
- поповнення та вивід коштів з інвестиційного рахунка;
- актуальна інформація про стан інвестиційного рахунка;
- внутрішній переказ між інвестиційними рахунками.

2 РОЗРОБКА СТРУКТУРИ ПРОГРАМИ

Даний розділ присвячений опису та обґрунтуванню програмно-технічних рішень, які були прийняті під час створення ПЗ.

ПЗ складається з чотирьох основних частин:

- користувацький та адміністративний інтерфейс;
- інтерфейс прикладного програмування (API);
- база даних (БД).

Користувацький та адміністративний інтерфейс відповідають за відображення даних клієнту та адміністратору відповідно.

Інтерфейс прикладного програмування відповідає за надання даних про інвестиційні рахунки, їх обробку та збереження.

2.1 Архітектурний шаблон

Під час вибору архітектури сайту, було розглянуто декілька шаблонів. Після оглядів усіх недоліків та сильних сторін різних архітектурних шаблонів, було обрано сучасний архітектурний підхід модель-вид-контролер (від англ. Model-View-Controller, MVC). Основний підхід при розробці з використанням цього архітектурного шаблону – відсутність залежностей між користувацьким інтерфейсом, даними та логікою програми. MVC включає в себе модель (дані), вид (інтерфейс) та контролер (процеси, які обробляють введені користувачем дані) [4].

Модель – структурує та готує дані до відповідної форми на основі інструкцій, отриманих від контролера.

Вид – відображає дані.

Контролер – приймає команди від користувача, відправляє команди до моделі для оновлення даних, відправляє інструкції до виду для оновлення інтерфейсу.

Принцип роботи MVC зображений на рис. 2.1.

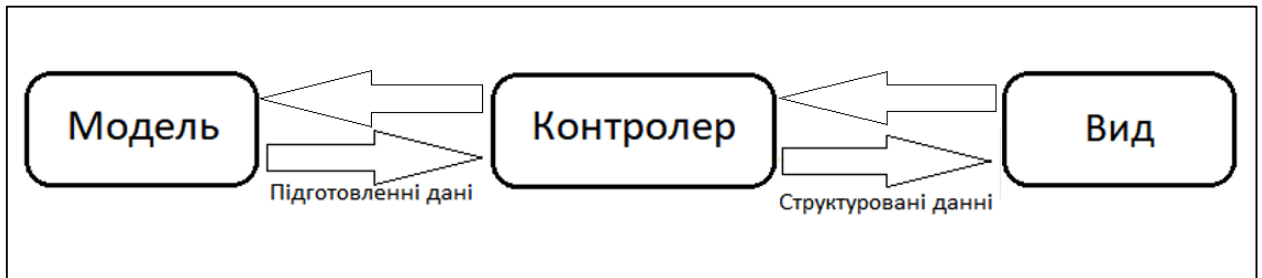


Рисунок 2.1 – Принцип роботи MVC

2.2 Вебсервер

В якості вебсерверу було вибрано Apache. Apache - HTTP сервер, що має високу надійність і гнучкість. Під HTTP сервером слід розуміти програмне забезпечення для обробки HTTP запитів [5].

Основна робота Apache це обробка і відповідь на HTTP запити і генерування динамічного змісті сторінок. Гнучкість досягається шляхом використання файлу .htaccess, завдяки якому можна перевизначити глобальні налаштування сервера Apache.

У якості проксі-серверу було вибрано Nginx. Nginx - HTTP-сервер, зворотний проксі-сервер, поштовий проксі-сервер, а також TCP / UDP проксі-сервер загального призначення. Уже тривалий час він обслуговує сервери багатьох високонавантажених проєктів [6].

Проксі-сервер - проміжний сервер в комп'ютерних мережах, що виконує роль посередника між користувачем і цільовим сервером, що дозволяє клієнтам як виконувати непрямі запити [7] (приймаючи і передаючи їх через проксі-сервер) до інших мережних служб, так і отримувати відповіді.

2.3 Мова програмування PHP

В якості мови програмування для написання серверної частини програмного продукту було обрано мову PHP.

PHP – скриптова мова загального призначення, інтенсивно застосовується для розробки вебдодатків та має відкритий вихідний код. Аббревіатура PHP означає «Hypertext Preprocessor» («Препроцесор Гіпертексту»).

Ті факти, що сама мова програмування має слабку типізацію (не вимагає оголошення типів даних), а також те, що синтаксис мови бере початок з C, Java і Perl, робить PHP досить простою мовою програмування для освоєння.

PHP досить гнучка мова програмування, з її допомогою можна створювати застосунки, використовуючи наступні парадигми програмування:

- імперативне програмування;
- функціональне програмування;
- об'єктно-орієнтоване програмування (ООП);
- процедурне програмування;
- рефлексивне програмування.

На даний момент найбільш популярні версії PHP є 5 та 7.

Остання, сьома версія мови внесла багато виправлень та покращень, зробивши PHP ще більш ООП наведеною мовою. Серед нових функцій PHP 7 також можна виділити Null-коалесцентний оператор та оператор "космічний корабель", які дають змогу покращити зовнішній вигляд коду, та підвищити його зрозумілість для програміста.

Однак основною, та найважливішою частиною оновлення PHP є неймовірний приріст до швидкодії, який був досягнутий завдяки рефактору коду ядра системи.

При виборі мови програмування для створення програмного продукту, було створено список критеріїв, яким повинна відповідати мова

програмування. Список критеріїв та порівняння різних мов програмування наведені у таблиці 2.1.

Таблиця 2.1 – Порівняння мов програмування

Критерії	Мови програмування		
	PHP	C#	Python
Динамічна типізація	+	+	+
Явна типізація	±	-	±
Неявна типізація	+	+	+
Багатоплатформність	+	+	+
Декларація типів та повертаємих значень	+	-	-

Хоча мова PHP, головним чином, призначена для роботи в середовищі вебсерверів, область його застосування не обмежується тільки цим. Ці фактори, у копії з іншими перевагами, такими як швидкість роботи та велика кількість різних фреймворків, забезпечили мові програмування PHP велику популярність серед розробників та дружнє товариство.

Беручи до уваги усі переваги та велику кількість навчальної документації, для розробки погромного продукту було обрано мову програмування PHP.

2.4 PHP фреймворк Yii2

Для розробки вебзастосунку було прийнято рішення використати PHP фреймворк Yii2.

Yii2 - об'єктно-орієнтована компонентний фреймворк, який реалізує парадигму MVC та призначений для швидкої розробки сучасних вебзастосунків. У основі фреймворку закладена архітектура MVC, що надає змогу створювати гнучкі та зрозумілі структуровані програмні продукти [8].

Великою перевагою Yii2 є простий та елегантний код, який облегує вивчення функцій фреймворку та надає змогу писати зрозумілий та ефективний код з мінімальними витратами часу. Також фреймворк є

надзвичайно легким в освоєнні завдяки розширюваності. Користувач може налаштувати або замінити практично будь-яку частину основного коду або модулів системи.

Yii2 є full-stack фреймворком і включає в себе багато можливостей: конструктори запитів та ActiveRecord для реляційних та NoSQL баз даних, просту підтримку REST API, багаторівневе кешування та інші.

Також базові можливості Yii2 можна розширити модулями. З їх допомогою досить легко можна додати будь-який функціонал до системи, що дуже скорочує час на розробку програмного продукту та ще більше полегшує знайомство з фреймворком.

Важливою перевагою Yii2 є добре задокументований функціонал фреймворку. Документація є до усіх функцій, на декількох мовах (українська, англійська, іспанська тощо).

Ключовими перевагами, завдяки яким було обрано саме фреймворк Yii2 можна назвати:

- висока продуктивність щодо інших фреймворків, написаних на PHP;
- використання AJAX і інтеграція з jQuery;
- генерація базового PHP-коду для CRUD-операцій;
- підтримка REST;
- міграції бази даних.

Хоча Yii2 є досить простим для освоєння фреймворком, він потребує розуміння принципів ООП та MVC.

2.5 Система керування базою даних MySQL

Усі дані, які зберігаються в БД, мають бути однозначні, достовірні, несуперечливі. Для створення структури БД, яка відповідає вказаним вимогам, найкраще підходить реляційна БД.

Реляційна база даних - це набір даних з зумовленими зв'язками між ними. Ці дані організовані у вигляді набору таблиць, що складаються із

стовпців і рядків. У таблицях зберігається інформація про об'єкти, представлених в базі даних. У кожному стовпчику таблиці зберігається певний тип даних, в кожному осередку - значення атрибута. Кожна стока таблиці являє собою набір пов'язаних значень, що відносяться до одного об'єкту або сутності. Кожен рядок в таблиці може бути позначена унікальним ідентифікатором, званим первинним ключем, а рядки з декількох таблиць можуть бути пов'язані з допомогою зовнішніх ключів. До цих даних можна отримати доступ багатьма способами, і при цьому реорганізовувати таблиці БД не потрібно.

SQL – декларативна мова програмування, яка застосовується для створення, модифікації та управління даними в реляційній базі даних, керованої відповідною системою управління базами даних.

Система управління базами даних(СКБД) - комплекс програм, що дозволяють створити базу даних (БД) і маніпулювати даними (вставляти, оновлювати, видаляти і вибирати). Система забезпечує безпеку, надійність зберігання і цілісність даних, а також надає кошти для адміністрування БД.

В якості системи керування базою даних було обрано MySQL.

MySQL - вільна реляційна система управління базами даних. Входить до складу серверів WAMP, AppServ, LAMP і в портативні збірки серверів Денвер, XAMPP, VertrigoServ [9]. Зазвичай MySQL використовується як сервер, до якого звертаються локальні або видалені клієнти, проте в дистрибутив входить бібліотека внутрішнього сервера, що дозволяє включати MySQL в автономні програми.

Перевагами система управління базами даних MySQL перед її конкурентами, такими як MariaDB чи MSSQL Server є:

- простота використання;
- безпека;
- великий функціонал;
- масштабованість
- швидкість.

Хоча MySQL має великі переваги перед конкурентами, що дало змогу обрати саме її як основну систему управління базами даних, однак існують певні недоліки:

- низька швидкість розробки;
- мала надійність зв'язків між транзакціями.

2.6 Огляд особливостей середовища розробки PhpStorm

Для розробки програмного продукту було використано IDE PhpStorm.

IDE(Інтегроване середовище розробки) - система програмних засобів, яка використовується програмістами для розробки програмного забезпечення, які включають у себе текстовий редактор, транслятор, засоби автоматизації збирання, відладчик.

PhpStorm - комерційна крос-платформна інтегроване середовище розробки для PHP. Розробляється компанією JetBrains на основі платформи IntelliJ IDEA. PhpStorm являє собою інтелектуальний редактор для PHP, HTML і JavaScript з можливостями аналізу коду на льоту, запобігання помилок в коді і автоматизованими засобами рефакторинга для PHP і JavaScript [10].

IDE PhpStorm інтегрована з системами контролю версій, підтримує віддалене розгортання, бази даних і SQL, інструменти командного рядка, Docker, Composer, REST-клієнт і багато інших інструментів.

У цьому випадку усі сторінки вебсайту містять посилання на інші сторінки, і користувач може легко перейти з будь-якої сторінки практично на будь-яку іншу. Ця схема може перетворитися у лабіринт, якщо вийде з-під контролю, але вона популярна у тих випадках, коли посиланнями на документи користуються не надто часто. На рисунку 2.4 нижче представлений доволі простий приклад такої організації.

Ключовими перевагами PhpStorm є:

- візуальний відладчик для PHP-застосунків;

- підтримка Vagrant, Composer, вбудований REST клієнт, CLI, SSH консоль;
- HTML, CSS, JavaScript редактор;
- інтеграція з системами управління версіями;
- кросплатформність.

2.7 Аналіз можливосте, HTML, CSS та Bootstrap

HTML (англ. HyperText Markup Language – мова розмітки гіпертекстових документів) - це мова розмітки, яка використовується для візуального і логічного структурування контенту на вебсторінці. Документ, створений за допомогою HTML оброблюється браузером та відтворюється на екрані у звичному для людини вигляді.

Безпосереднє сама мова HTML складається з міток. Саме ці мітки або теги (tags), укладені в кутові дужки, формують елементи, з яких і складається код HTML і які потім браузер інтерпретує у зрозумілому та доступному вигляді на вебсторінці.

На даний момент існує 5 версій HTML. Останній стандарт (HTML5) був закінчена у 2014 році, та мав мету поліпшити рівень підтримки мультимедіа-технологій з одночасним збереженням зворотної сумісності, та зберегти простоту аналізу для парсерів.

У HTML5 реалізовано безліч нових синтаксичних особливостей. Наприклад, елементи <video>, <audio> і <canvas>, а також можливість використання SVG і математичних формул.

CSS (англ. Cascading Style Sheets – Каскадні таблиці стилів) - це мова, яка відповідає за візуальне уявлення документів користувачеві. CSS являється формальною мовою опису зовнішнього вигляду документа, написаного з використанням мови розмітки.

CSS використовується для завдання кольорів, шрифтів, стилів, розташування окремих блоків і інших аспектів представлення зовнішнього

вигляду на вебсторін. Крім того, CSS дозволяє представляти один і той же документ в різних стилях або методах виведення, таких як екранне уявлення, друковане подання, читання голосом (спеціальним голосовим браузером або програмою читання з екрану), або при виведенні пристроями, що використовують шрифт Брайля.

Bootstrap - фреймворк на основі HTML і CSS. Він містить стилі для основних елементів, які застосовуються при створенні вебсторінок. Використання такого фреймворку значно прискорює процес створення сторінок. Стандартні стилі легко міняти, що забезпечує гнучкий і простий процес створення макетів сайтів.

Переваги Bootstrap:

- наявність мінливої мови стилів (Less);
- кросбраузерність;
- висока швидкість роботи;
- простота у використанні.

Однак фреймворк має і свої недоліки:

- незначні проблеми при роботі з Javascript;
- конфлікти деяких інструментів;
- тривалість виправлень недоліків фреймворку.

Основною метою розробки CSS було відділення опису логічної структури вебсторінки (яке проводиться за допомогою HTML або інших мов розмітки) від опису зовнішнього вигляду цієї вебсторінки (яке тепер проводиться за допомогою формальної мови CSS).

2.8 Мова програмування JavaScript

JavaScript - це мова керування сценаріями перегляду гіпертекстових сторінок Web на стороні клієнта.

Основна ідея JavaScript полягає в можливості зміни значень атрибутів HTML-контейнерів і властивостей середовища відображення в процесі перегляду HTML-сторінки користувачем. При цьому перезавантаження сторінки не відбувається. Це надає великий простір для створення інтерактивності між клієнтом та вебсторінкою.

Основні архітектурні риси: динамічна типізація, слабка типізація, автоматичне керування пам'яттю, прототипе програмування, функції як об'єкти першого класу.

Хоча мова була створена з оглядом на Java, не можна сказати що вона є дуже простою в освоєнні, через неочевидні помилки та взаємодії на рівні компілятора, результат яких іноді важко передбачити без певного досвіду у програмуванні на цій мові програмування. Це обумовлено частковою асинхронністю мови. Тобто деякі функції програми будуть виконуватися по черзі, у той час коли асинхронні функцію можуть бути виконані раніше, ніж програміст очікує, так як вони не знаходяться у загальній синхронній черзі на виконання [11].

Під час написання програмного продукту, для полегшення задач та прискорення роботи було використано бібліотеку JQuery.

JQuery - набір функцій JavaScript, що фокусується на взаємодії JavaScript і HTML. Бібліотека jQuery допомагає легко отримувати доступ до будь-якого елементу DOM, звертатися до атрибутів і вмісту елементів DOM, маніпулювати ними. Також бібліотека jQuery надає зручний API для роботи з AJAX.

DOM - це об'єктна модель документа, яку браузер створює в пам'яті комп'ютера на підставі HTML-коду, отриманого ним від сервера. Принцип роботи DOM зображено на рисунку 2.2.

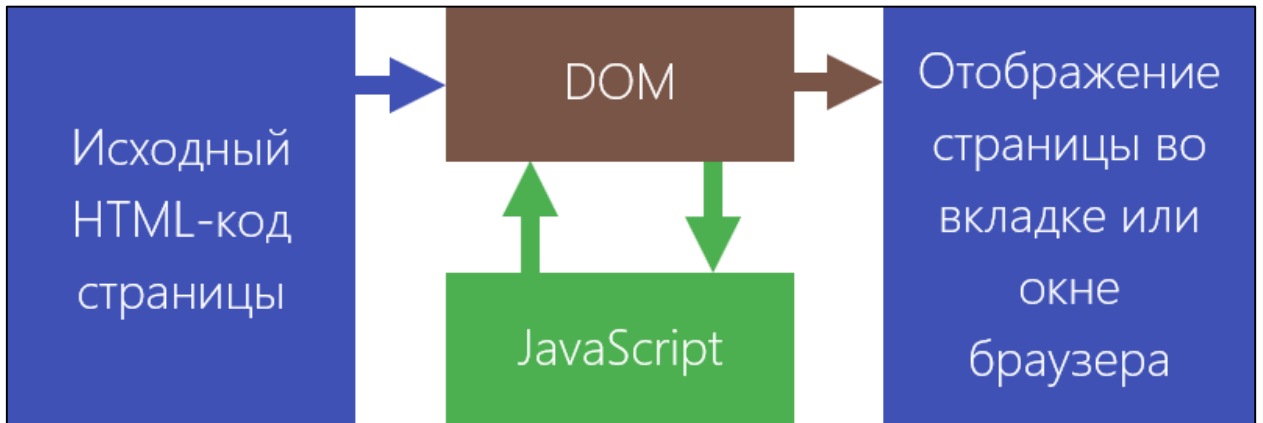


Рисунок 2.2 – Принцип работы DOM

AJAX - підхід до побудови призначених для користувача інтерфейсів вебдодатків, що полягає в «фоновому» обміні даними браузера з вебсервером. В результаті при оновленні даних вебсторінка не перезавантажується повністю, і вебдодатки стають швидше і зручніше.

2.9 Висновки

У другому розділі описано та проаналізовано технології, які будуть використовуватися при розробці програмного засобу. Був проведений аналіз мов програмування та середовищ розробки, за допомогою яких буде проводитись розробка програмного засобу.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Загальна архітектура програми

Архітектура програми побудована на взаємодії компонентів програми через вебсервер apache, за допомогою HTTP-запитів. Сервер отримує запит від користувача, надісланий веббраузером, обробляє його, використовуючи дані отримані з бази даних, формує відповідь, та надсилає її до браузера, який в свою чергу інтерпретує відповідь та відображає її користувачу. Архітектура проєкту відображена на рисунку 3.1.

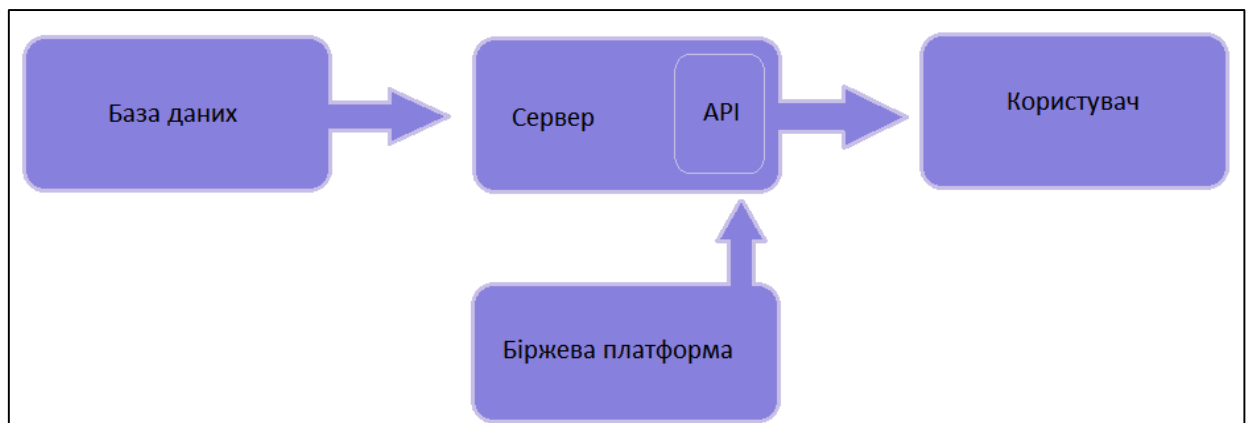


Рисунок 3.1 – Загальна архітектура програмного забезпечення

3.2 Структура розробленої бази даних

В процесі розробки програми було створено БД для зберігання інформації про користувачів та дані про їх інвестиційні рахунки.

БД, використана у програмі, складається з таблиць, пов'язаних між собою відношеннями різних типів. Були створені таблиці, описані нижче.

Таблиця user містить базові дані про користувача:

- id – унікальний ідентифікатор користувача;
- first_name – ім'я користувача;
- last_name – прізвище користувача;

- username – логін користувача;
- auth_key – тимчасовий ключ доступу;
- secret_key – ключ для шифрування;
- password_hash – хеш паролю користувача;
- email – пошта користувача;
- promo_code_id – ідентифікатор промо коду;
- status – статус користувача;
- created_at – дата створення користувацького запису;
- updated_at – дата останнього оновлення користувацького запису;
- language – мова інтерфейсу користувача;
- mt_account – перелік інвестиційних рахунків користувача.

Таблиця profile містить детальні дані про користувача:

- id – унікальний ідентифікатор;
- user_id – унікальний ідентифікатор користувача;
- phone – номер телефону користувача;
- country – країна користувача;
- birthday – день народження користувача;
- address – адреса проживання користувача;
- city – місто проживання користувача;
- zip_code – поштовий індекс міста;
- account_type_id – тип аккаунту;
- verification – статус верифікації аккаунту.

Таблиця refferal_codes містить дані про коди для реферальної системи:

- id – унікальний ідентифікатор;
- user_id – унікальний ідентифікатор користувача;
- code – унікальний реферальний код користувача;
- status – статус реферальної програми;
- account – реферальний аккаунт.

Таблиця refferals містить дані про ієрархію користувачів реферальної системи:

- id – унікальний ідентифікатор;
- user - унікальний ідентифікатор користувача (власника);
- user_id – унікальний ідентифікатор користувача (реферала);
- date – дата створення запису;
- parent_id – користувач, за яким закріплений власник запису;
- status – статус запису.

Таблиця documents описує документи, завантажені користувачем на підтвердження:

- id – унікальний ідентифікатор;
- user_id – унікальний ідентифікатор користувача;
- document – посилання на документ;
- type – тип документу;
- status – статус документу;
- created_at – дата завантаження документу.

Таблиця notifications містить дані про сповіщення, надісланні користувачу:

- id – унікальний ідентифікатор;
- user_id – унікальний ідентифікатор користувача;
- type – тип сповіщення;
- title – шаголовок сповіщення;
- message – текст сповіщення;
- date – дата створення сповіщення;
- status – статус сповіщення.

Структура БД зображена на рис. 3.2.

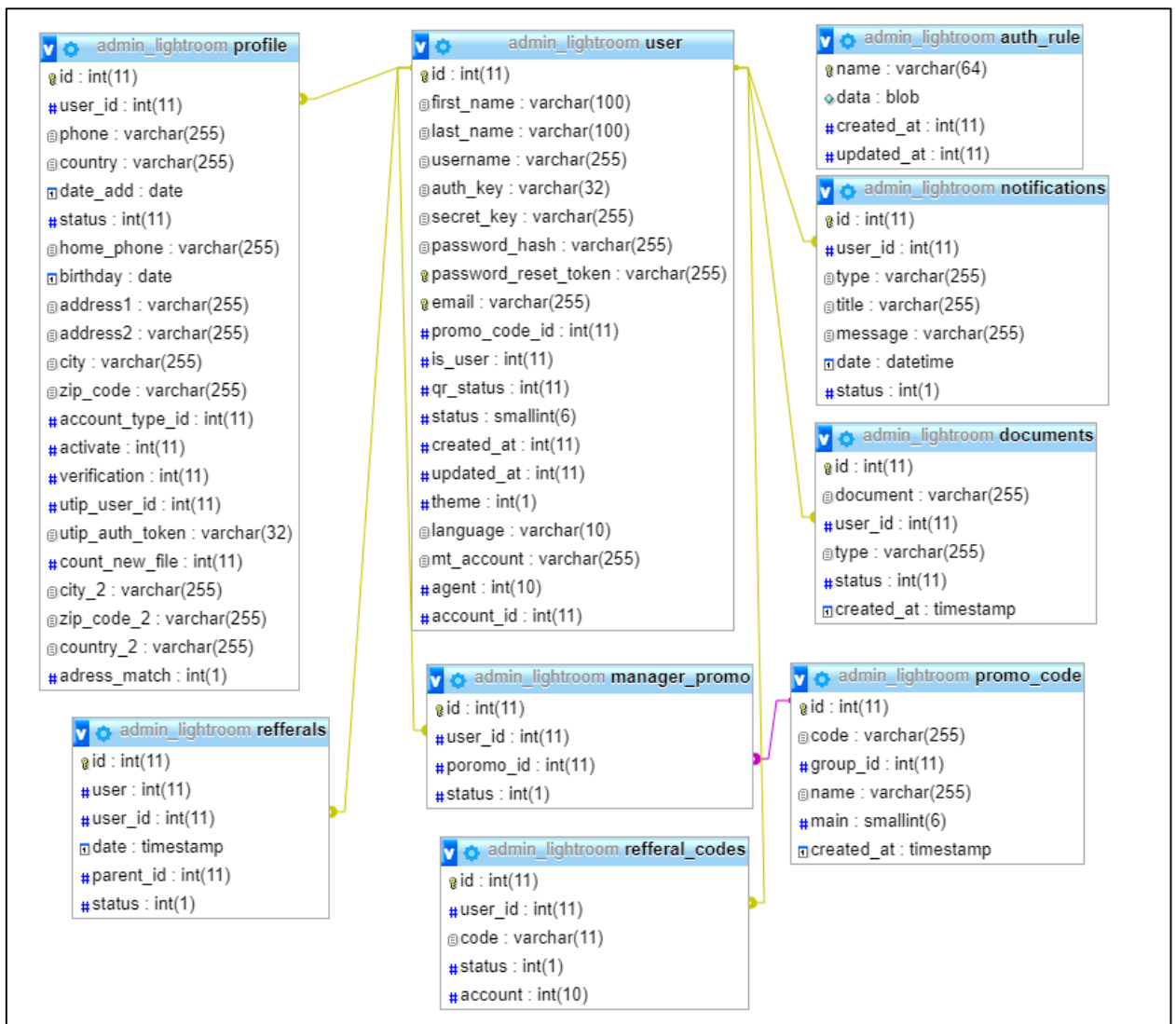


Рисунок 3.2 – Структура БД

3.3 Структура файлів проєкту

Архітектура MVC, яку використовує фреймворк Yii2, змушує дотримуватися певної структури файлів та папок при створенні програмного продукту. В кореновому каталозі повинні бути папки backend, для адміністративної частини, frontend, для користувацької частини, та common для компонентів, які використовуються в обох програмних застосунках.

Структура Yii2 передбачає розділення користувацької та адміністративної частини на два окремих застосунка, в середні яких буде застосовано архітектуру MVC.

Архітектура MVC потребує розміщення файлів моделі у каталозі `model`, файлів контролерів у каталозі `controller`, а файлів представлень у каталозі `view`.

Каталог `common` не має своєї директорії `web`, яка призначена для доступу вебкористувачам. Це обумовлено тим, що цей каталог не являє собою окремий проєкт, а існує для усунення проблеми дублювання коду, яких використовується одночасно у проєктах `backend` и `frontend`.

Окрім цього, файли сторонніх бібліотек, встановлених за допомогою менеджера `Composer`, зберігаються у каталозі `vendor`.

Графічне зображення файлової структури адміністративної частини проєкту наведено на рис. 3.3, а клієнтської – на рис 3.4.

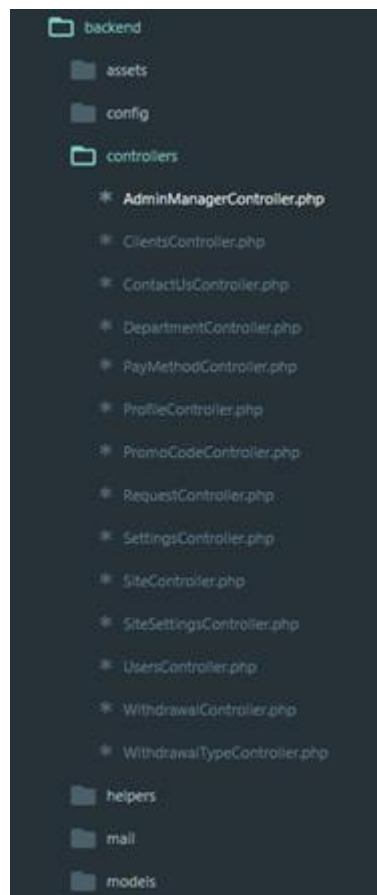


Рисунок 3.3 – Файлова структура адміністративної частини проєкту

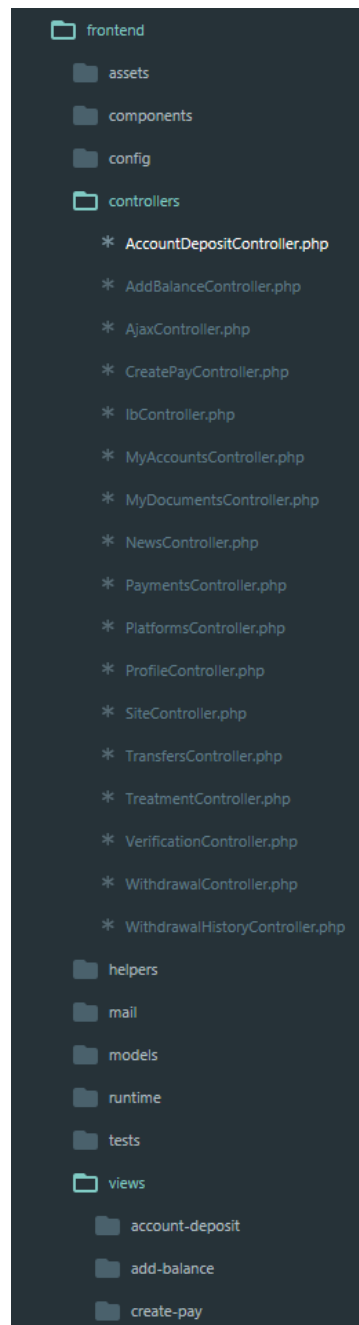


Рисунок 3.4 – Файлова структура клієнтської частини проєкту

3.4 Висновки

У третьому розділі було описано рішення, прийняті під час розробки кожної з частин системи, яка повністю відповідає поставленим вимогам.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення та умови застосування

Розроблене програмне забезпечення призначене для спрощення процесу менеджменту своїх інвестицій на баржевому ринці, шляхом взаємодії з торгівельною платформою MetaTrader 5. Біржові інвестиції являють собою досить складний процес. Саме цю проблему вирішує розроблене програмне забезпечення, надаючи простий та зрозумілий інтерфейс для користувача, та адміністратора.

Використовуючи даний програмний застосунок, користувач має змогу швидко отримати доступ до своїх інвестиційних рахунків з будь-якої точки планети за допомогою мережі Інтернет.

Програма виконує наступні функції:

- реєстрація адміністратора та менеджера;
- зберігання конфіденційних даних користувача;
- створення інвестиційних рахунків та перегляд їх історії;
- поповнення балансу інвестиційного рахунку;
- обмін та конвертація балансу між рахунками;
- проведення операцій над рахунком;
- верифікація користувачів;
- переклад контенту;
- створення користувацьких нотифікацій;
- вивід коштів на банківський рахунок.

Для нормального функціонування програми апаратна частина повинна відповідати наступним мінімальним вимогам:

- сучасний центральним процесор з 4 ядрами (Intel Xeon або AMD Threadripper) та тактовою частотою не менш ніж 2 ГГц;
- наявність 4 Гб або більше оперативної пам'яті;
- наявність не менше 50 Гб вільного простору на серверному накопичувачі.

Також, для програмна частина повинна відповідати наступним умовам:

- будь-яка ОС, сумісна з іншим програмним забезпеченням (CentOS, Ubuntu, Windows Server тощо);
- встановлена СКБД MySQL 5.5 чи вище;
- встановлений вебсервер (Apache, Nginx);
- підтримка PHP 7.2;
- налаштований поштовий сервер для відправки електронної пошти за допомогою стандартної функції PHP mail();
- встановлений пакетний менеджер Composer;
- встановлена система керування версіями Git.

4.2 Характеристики програми

Для реалізації вебсервера, необхідного для правильної роботи програмного забезпечення, було обрано мову програмування PHP. Також, для оформлення клієнтської та адміністративної частини сайту було використано мови HTML та CSS. Інтерактивність вебдокументу надає мова програмування Javascript, за допомогою бібліотеки JQuery, та використанням технології AJAX. В якості інтегрованого середовища розробки було використано IDE PHPStorm. Для створення середовища функціонування програми було використано ОС Ubuntu, основана на базі Linux. Дані зберігаються в реляційній базі даних під керування СКБД MySQL.

4.2.1 Звертання до програми

Незареєстровані користувачі будуть автоматично переадресовані на сторінку аутентифікації. Перегляд сторінок програмного засобу здійснюється за допомогою переходів за посиланнями.

Взаємодія клієнтської частини з користувачем здійснюється за допомогою графічного інтерфейсу.

Взаємодія серверної частини з торгівельною платформою здійснюється за допомогою REST API.

4.2.2 Вхідні і вихідні дані

Вхідних дані: HTTP-запити від користувачів.

Вихідні дані: HTTP-відповіді та файли для друку.

5 КЕРІВНИЦТВО ОПЕРАТОРА

5.1 Тестування клієнтської частини сайту

Розроблене програмне забезпечення призначене для спрощення процесу менеджменту своїх інвестицій на біржовому ринці, шляхом взаємодії з торгівельною платформою MetaTrader 5. Це дозволить користувачам легше зрозуміти інвестиційні моделі на біржовому ринці, та облегшить вхід до світу інвестицій.

Програма виконує наступні функції:

- реєстрація користувача;
- зберігання конфіденційних даних користувача;
- створення інвестиційних рахунків;
- поповнення балансу інвестиційного рахунку;
- обмін та конвертація балансу між рахунками;
- вивід коштів на банківський рахунок.

Для нормального доступу як до клієнтської, так і до адміністративної частин програми достатньо будь-якого пристрою зі встановленим сучасним веббраузером.

5.2 Звернення до програми

Для початку роботи з клієнтською частиною потрібно перейти за вебадресою <https://dev.rannforex.asia/login> зареєструватися, пройти підтвердження, ввести ім'я користувача та пароль для доступу на сайт.

Форма аутентифікації клієнтської частини сайту зображена на рис. 5.1.

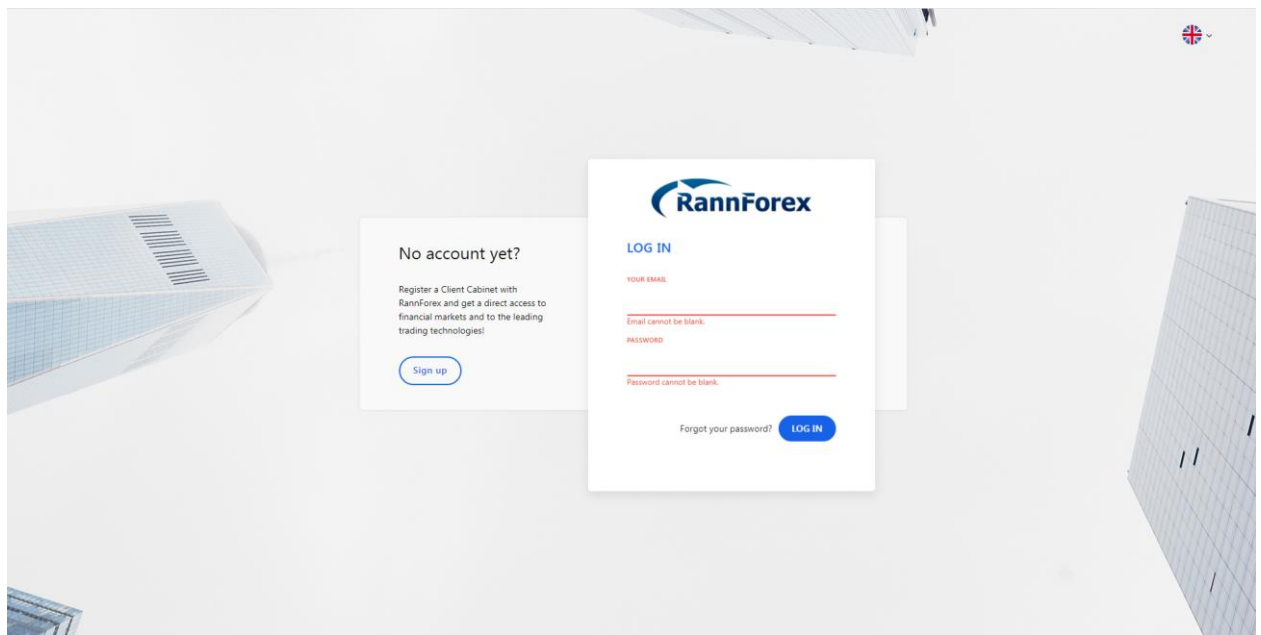


Рисунок 5.1 – Форма аутентифікації клієнтської частини сайту

Після аутентифікації, користувач потрапляє на головну сторінку клієнтської частини вебсайту, де відображається поточна інформація про усі відкриті інвестиційні рахунки. Головна сторінка зображена на рис. 5.2.

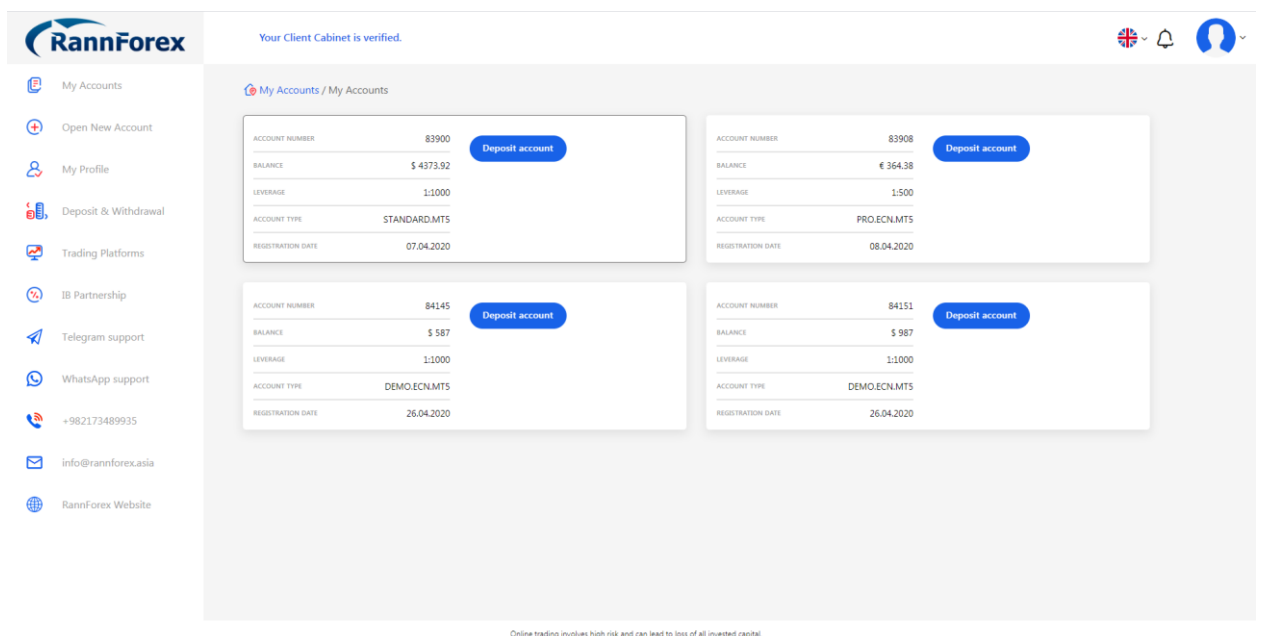


Рисунок 5.2 – Головна сторінка клієнтської частини сайту

Щоб відкрити інвестиційний рахунок, необхідно заповнити особисті дані та пройти верифікацію користувача, завантаживши документи, які підтверджують вашу особистість. Для цього необхідно натиснути у меню кнопку My Profile. На вкладниці account data необхідно заповнити обов'язкові поля, та натиснути save. Форма інформації про клієнта зображена на рисунку 5.3.

The screenshot shows a web form titled "My Accounts / My Profile" with a sub-tab "ACCOUNT DATA". The form is divided into two columns. The left column contains fields for EMAIL (prybylov.v@gmail.com), PHONE, FIRST NAME (Selby), LAST NAME (Khuzwayo), and DATE OF BIRTH (2 February 1993). The right column contains fields for REGISTRATION ADDRESS (Sonnenbergstrasse 48), CITY (Schwerzenbach1), COUNTRY (Russia), and POSTAL CODE (8603). A "SAVE" button is located at the bottom right. A note at the bottom left states: "To change your phone or email address, please contact our client support .". A legend indicates that an asterisk (*) denotes required fields.

Рисунок 5.3 – Форма інформації про клієнта

Щоб завантажити документи для верифікації необхідно перейти до розділу Verification та завантажити документи. Форма завантаження документів зображена на рисунку 5.4. Після завантаження документів, необхідно дочекатися підтвердження модератором.

Рисунок 5.4 – Форма завантаження документів

Після верифікації аккаунту можна створити інвестиційний рахунок. Для цього необхідно перейти на сторінку Open New Account, вибрати бажані налаштування рахунку, такі як тип рахунку, валюту рахунку та кредитне плече. Після цього необхідно натиснути кнопку Open an account, щоб створити рахунок з бажаними налаштуваннями на торгівельній платформі. Форма створення рахунку зображена на рисунку 5.5.

Рисунок 5.5 – Форма завантаження документів

Для початку роботи з адміністративною частиною потрібно перейти за вебадресою <https://dev.rannforex.asia/admin/en/login> та ввести ім'я користувача та пароль для доступу до адміністративної частини.

Форма аутентифікації адміністративної частини сайту зображена на рис. 5.6.

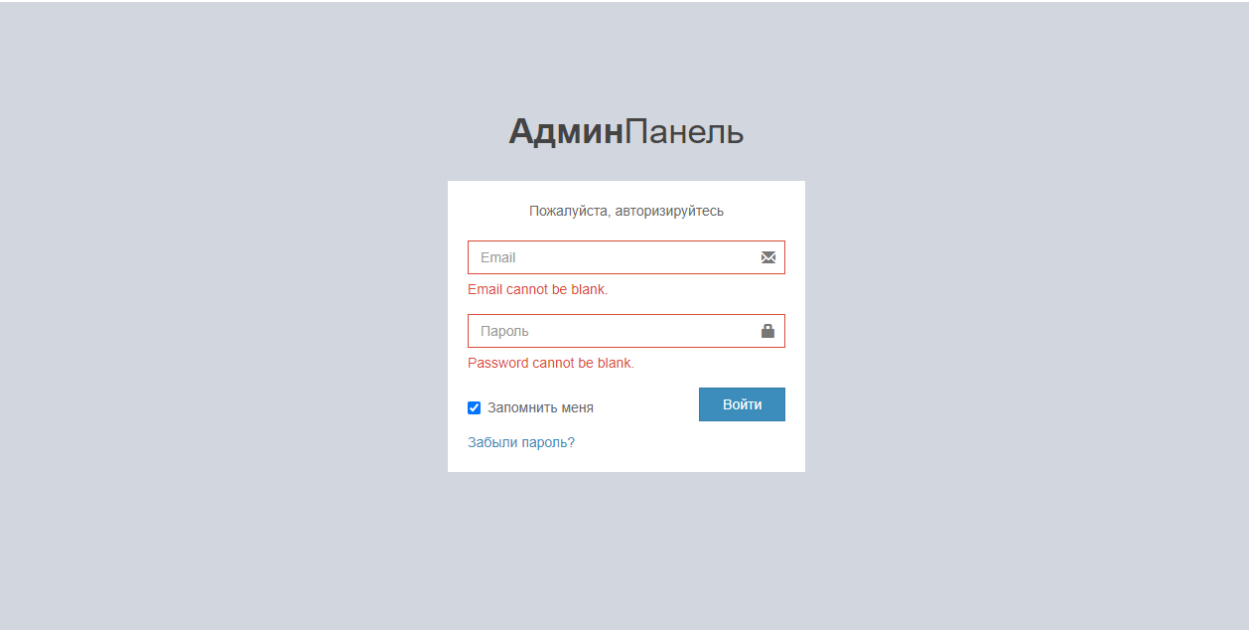


Рисунок 5.6 – Форма аутентифікації адміністративної частини сайту

Після аутентифікації, користувач потрапляє на головну сторінку адміністративної частини вебсайту, яка зображена на рис. 5.7.

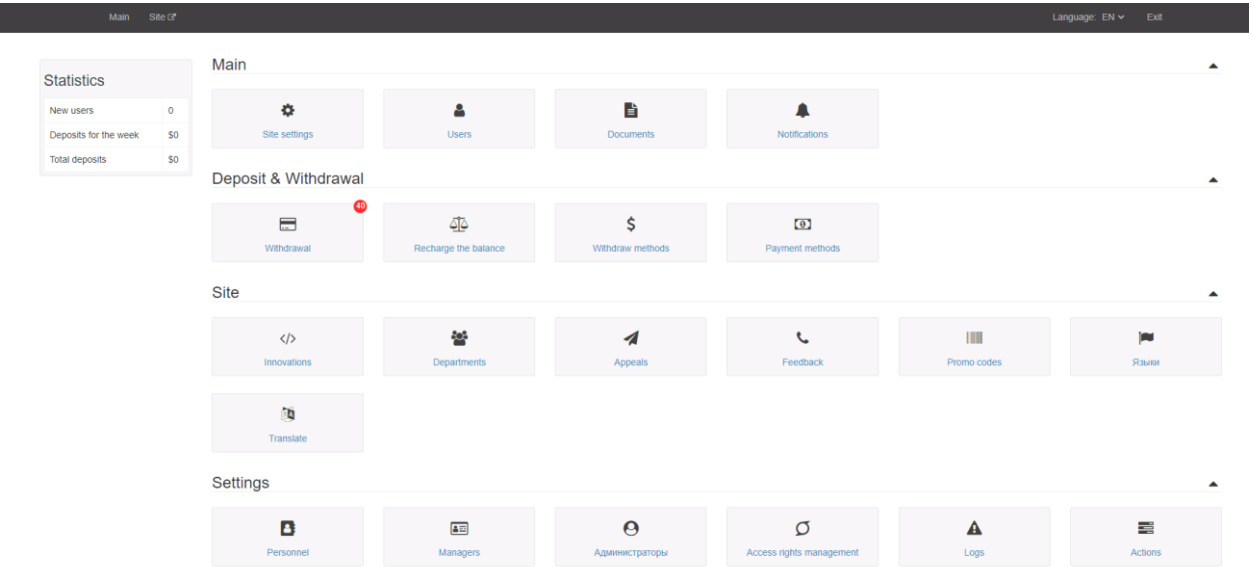


Рисунок 5.7 – Форма аутентифікації адміністративної частини сайту

На головній сторінці адміністративної частини сайту відображені пункти меню, з різноманітним функціоналом. Щоб переглянути список закріплених за адміністратором користувачів, необхідно перейти до пункту меню Users. На рисунку 5.8 зображено відображення користувачів сайту.

Home / Clients

Search

#	User	Phone number	Accounts	Verification documents	Status	Actions
1	test test ID: 6146 IDMT: Awaiting Verification cojhevxltitlqmdavls@ttirv.com Add manager	+914444433333		New: 0 Approved: 0 Not approved: 0	Active	
2	test test ID: 6145 IDMT: Awaiting Verification developorange@gmail.com Add manager	+989123457693		New: 0 Approved: 0 Not approved: 0	Blocked	
3	محمدحسين فلاح زاده ID: 6144 IDMT: Awaiting Verification dkhfc06384@gmail.com Add manager	+989175111788		New: 0 Approved: 0 Not approved: 0	Blocked	
4	Mohammad hosen Mohammadi ID: 6143 IDMT: Awaiting Verification Mohammadhm1375@gmail.com Add manager	+989017228281		New: 0 Approved: 0 Not approved: 0	Blocked	
5	omid qassemi ID: 6137 IDMT: Awaiting Verification omidghaasemi@yahoo.com Add manager	+989351956322		New: 0 Approved: 0 Not approved: 0	Active	
6	سندباد نجفیان ID: 6134 IDMT: Verified S.najafian90@gmail.com Add manager	+989125064925		New: 0 Approved: 0 Not approved: 0	Active	

Рисунок 5.8 – Інтерфейс відображення користувачів сайту

Для підтвердження верифікації користувача необхідно підтвердити документи, завантажені користувачем. Для цього, на головній сторінці сайту, необхідно натиснути на пункт меню Documents. Після цього користувач потрапляє на сторінку верифікації документів. Щоб переглянути документ, завантажений користувачем, адміністратору необхідно натиснути на кнопку view. Щоб змінити статус документу, необхідно у випадяючому меню вибрати бажаний статус. Інтерфейс верифікації документів зображено на рисунку 5.9.

Date	Photo	Verification
ID пользователя: 6124		
2020-04-21 20:42:09	View	Verified ▾
ID пользователя: 6111		
2020-04-20 12:19:19	View	Rejected ▾
2020-04-20 12:19:19	View	Rejected ▾
ID пользователя: 6078		
2020-03-16 16:52:18	View	Verified ▾
ID пользователя: 1		
2018-09-10 13:35:19	View	Verified ▾
2018-09-11 06:25:08	View	Rejected ▾
2018-09-11 08:45:25	View	Verified ▾

Рисунок 5.9 – Інтерфейс верифікації документів

5.3 Висновки

В даному розділі описано призначення програми та надано інструкцію для її використання у ролі адміністратора та користувача.

ВИСНОВКИ

Під час виконання дипломної кваліфікаційної роботи бакалавра було розроблено програмне забезпечення та інтерфейс для керування інвестиційними рахунками для біржової торгівлі.

Було проведено аналіз предметної області та сучасних технологій, враховані можливі труднощі при розробці програмного забезпечення, та на їх основі поставлено та вирішено наступні задачі:

- обрано технології та інструменти, за допомогою яких було створене програмне забезпечення;
- налаштування серверу;
- створення зрозумілого та простого користувацького інтерфейсу;
- налаштування взаємодії сервера та клієнта через AJAX;
- налаштування взаємодії з торгівельною платформою через REST API.

Використовуючи такі технології як PHP, Javascript, HTML, CSS і MySQL, було розроблено програмне забезпечення, яке включає в себе динамічні сторінки та програмний інтерфейс засобу для взаємодії між двома програмами.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Научно электронная библиотека «КИБЕРЛЕНИНКА» [Электрон. ресурс] – Режим доступа: <https://cyberleninka.ru/article/n/analiz-programmnyh-sredstv-upravleniya-investitsionnymi-proektami/viewer>
2. Баффетт, У. Ессе про інвестиції, корпоративних фінансах та управлінні компаніями / Уоррен Баффетт, 2014. – 217 с.
3. Мерецков, О. М. Применение ИКТ в ВУЗе / Олег Мерецков, 2019. – 108 с.
4. MVC [Electronic resource]. – Access mode: <https://techterms.com/definition/mvc>
5. Что такое PHP? [Электрон. ресурс]. – Режим доступа: <http://www.php.su/php/?php>
6. Nginx [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Nginx>
7. Что такое прокси сервер и зачем он нужен [Электрон. ресурс]. – Режим доступа: <https://sonikelf.ru/proxy/>
8. Введение: О Yii | Полное руководство [Электрон. ресурс]. – Режим доступа: <https://www.yiiframework.com/doc/guide/2.0/ru/intro-yii>
9. MySQL [Electronic resource]. – Режим доступа: <https://searchoracle.techtarget.com/definition/MySQL>
10. Функциональность - PHPStorm [Электрон. ресурс]. – Режим доступа: <https://www.jetbrains.com/ru-ru/phpstorm/features/>
11. Обзор JavaScript [Электрон. ресурс]. – Режим доступа: https://developer.mozilla.org/ru/docs/Web/JavaScript/Guide/JavaScript_Overview

ДОДАТОК А
Текст програми

```

<?php
namespace frontend\controllers;

use common\models\Statuses;
use common\models\UserProfile;
use Yii;
use yii\filters\AccessControl;
use yii\web\Controller;
use backend\helpers>MainHelper;
use common\helpers\Utip;
use common\models\User;
use frontend\models\ChangePassword;
use frontend\models\ProfileForm;
use yii\web\Response;
use yii\widgets\ActiveForm;

class ProfileController extends Controller
{

    public $layout = 'account';

    public function behaviors()
    {
        return [
            'access' => [
                'class' => AccessControl::className(),
                'only' => ['*'],
                'rules' => [
                    [
                        'allow' => true,
                        'roles' => ['@'],
                    ],
                ],
            ],
        ];
    }

    public function actionIndex()
    {
        $profile = new ProfileForm();

        $request = Yii::$app->request->isAjax ? 'renderAjax' : 'render';
        $u_login = MainHelper::UtipLogin();
        $account = null;
        $user = User::find()
            ->joinWith(['userProfile'])
            ->where(['user.id' => Yii::$app->user->identity->getId()])
            ->one();

        if ($profile->load(Yii::$app->request->post())) {
            if(Yii::$app->params['trade_system'] == 'utip') {

```

```

        $utip = Utip::UpdateAccountInfo($user->userProfile->utip_user_id, $u_login, [
            'first_name' => $profile->name,
            'second_name' => $profile->surname,
            'phone' => $profile->phone,
            'date_of_birth' => mktime(10,10,10, date('m', strtotime($profile->b_year.'-'. $profile->b_month.'-'. $profile->b_day)), date('d', strtotime($profile->b_year.'-'. $profile->b_month.'-'. $profile->b_day)), date('Y', strtotime($profile->b_year.'-'. $profile->b_month.'-'. $profile->b_day))),
            'address' => $profile->address,
            'city' => $profile->city,
            'postcode' => $profile->zipcode,
            'country' => $profile->country
        ]);

        $u_profile = UserProfile::findOne(['user_id' => $user->id]);

        $user->first_name = $profile->name;
        $user->last_name = $profile->surname;
        $u_profile->phone = $profile->phone;
        $u_profile->birthday = date('Y-m-d', strtotime($profile->b_year.'-'. $profile->b_month.'-'. $profile->b_day));
        $u_profile->address1 = $profile->address;
        $u_profile->city = $profile->city;
        $u_profile->zip_code = $profile->zipcode;

        if ($utip['result'] == true && $user->save(false) && $u_profile->save(false)) {

            \common\modules\logger\model\ActionLog::add('success', Yii::t('app', 'Изменил личные данные'), Yii::$app->user->identity->id);

            Yii::$app->session->setFlash('success', Yii::t('app', 'Профиль успешно изменен'));

            return true;
        } else {
            \common\modules\logger\model\ActionLog::add('error', Yii::t('app', 'Пытался изменить личные данные'), Yii::$app->user->identity->id);
        }
    } else if (Yii::$app->params['trade_system'] == 'mt5') {
        $u_profile = UserProfile::findOne(['user_id' => $user->id]);

        $user->first_name = $profile->name;
        $user->last_name = $profile->surname;
        $u_profile->phone = $user->userProfile->phone;
        $u_profile->country = $profile->country;
        $u_profile->birthday = date('Y-m-d', strtotime($profile->b_year.'-'. $profile->b_month.'-'. $profile->b_day));
        $u_profile->address1 = $profile->address;
        $u_profile->city = $profile->city;
        $u_profile->zip_code = $profile->zipcode;
        if ($user->save(false) && $u_profile->save(false)) {

```

```

        \common\modules\logger\model\ActionLog::add('success', Yii::t('app', 'Изменил
личные данные'), Yii::$app->user->identity->id);

        Yii::$app->session->setFlash('success', Yii::t('app', 'Профиль успешно изменен'));

    } else {
        \common\modules\logger\model\ActionLog::add('error', Yii::t('app', 'Пытался
изменить личные данные'), Yii::$app->user->identity->id);
    }
}

if(Yii::$app->params['trade_system'] == 'utip') {
    if(isset($user->userProfile->utip_user_id)) {
        $account = Utip::GetAccountInfo($user->userProfile->utip_user_id, $u_login);
        $profile->phone = $account['data']['phone'];
        $profile->birthday = date('Y-m-d', $account['data']['date_of_birth']);
    }
} else if(Yii::$app->params['trade_system'] == 'mt5') {
    $user_data = Yii::$app->user->identity;
    $account['data']['first_name'] = $user_data['first_name'];
    $account['data']['second_name'] = $user_data['last_name'];
    $account['data']['email'] = $user_data['email'];
    $account['data']['address'] = $user_data->userProfile['address1'];
    $account['data']['city'] = $user_data->userProfile['city'];
    $account['data']['country'] = $user_data->userProfile['country'];
    $account['data']['postcode'] = $user_data->userProfile['zip_code'];
    $profile->phone = $user_data->userProfile['phone'];
    $profile->birthday = $user_data->userProfile['birthday'];
}
// $user_data = Yii::$app->user->identity;
// \common\components\VD::pretty($user_data->userProfile);

return $this->$request('index', [
    'user' => $user,
    'account' => isset($account['data']) ? $account['data'] : $account,
    'profile' => $profile
]);
}

public function actionChangePassword()
{
    $pass = new ChangePassword();
    $request = Yii::$app->request->isAjax ? 'renderAjax' : 'render';
    $message = null;

    $u_login = MainHelper::UtipLogin();

    $user = User::find()
        ->joinWith(['userProfile'])
        ->where(['user.id' => Yii::$app->user->identity->getId()])
        ->one();

```

```

        if ($pass->load(Yii::$app->request->post()) && $pass->validate()) {
            if(Yii::$app->params['trade_system'] == 'utip') {
                $utip = Utip::ChangePassword($user->userProfile->utip_user_id, $pass-
>newPassword, $u_login);
                $user->password_hash = Yii::$app->getSecurity()->generatePasswordHash($pass-
>newPassword);

                if($utip['result'] == true && $user->save(false)) {
                    Yii::$app->session->setFlash('info', Yii::t('app', 'Пароль успешно изменен.'));

                    return true;
                } else {
                    Yii::$app->session->setFlash('error', Yii::t('app', 'Ошибка при сохранении.'));
                    return false;
                }
            } else {
                $user->password_hash = Yii::$app->getSecurity()->generatePasswordHash($pass-
>newPassword);
                if($user->save(false)) {
                    Yii::$app->session->setFlash('info', Yii::t('app', 'Пароль успешно изменен.'));
                    if (Yii::$app->language == 'fa') {
                        Yii::$app->mailer->compose(
                            ['html' => '@frontend/mail/change-fa',
                            [
                                'user' => $user,
                                'logo' => Yii::getAlias('@frontend/web/img/logo.png')
                            ]
                        ]
                        )
                        ->setTo($user->email)
                        ->setFrom([Yii::$app->params['adminEmail'] => Yii::$app-
>params['noreplyEmail']])
                        ->setSubject(Yii::t('app', 'Ваш пароль был изменен'))
                        ->send();
                    } else {
                        Yii::$app->mailer->compose(
                            ['html' => '@frontend/mail/change',
                            [
                                'user' => $user,
                                'logo' => Yii::getAlias('@frontend/web/img/logo.png')
                            ]
                        ]
                        )
                        ->setTo($user->email)
                        ->setFrom([Yii::$app->params['adminEmail'] => Yii::$app-
>params['noreplyEmail']])
                        ->setSubject(Yii::t('app', 'Ваш пароль был изменен'))
                        ->send();
                    }
                }

                Yii::$app->user->logout();

                return $this->goHome();
            }
        }
    }
}

```

```

    } else {
        Yii::$app->session->setFlash('error', Yii::t('app', 'Ошибка при сохранении.));
        return false;
    }
}

$message = Yii::$app->session->get('qr');
Yii::$app->session->set('qr', null);

return $this->$request('change-password', [
    'pass' => $pass,
    'message' => $message,
]);
}

public function actionValidatePassword()
{
    $model = new ChangePassword();
    $request = \Yii::$app->getRequest();
    if ($request->isPost && $model->load($request->post())) {
//        \Yii::$app->response->format = Response::FORMAT_JSON;
        return ActiveForm::validate($model);
    }
}

public function actionQrCode()
{
    Yii::$app->controller->enableCsrfValidation = false;

    if (Yii::$app->request->isAjax) {
        $user = User::findOne(Yii::$app->user->identity->getId());
        $user->qr_status = Statuses::STATUS_ACTIVE;

        if ($user->save(false)) {
            $g = new \Google\Authenticator\GoogleAuthenticator();
            $explode = explode('@', $user->email);

            return json_encode([
                'success' => true,
                'data' => [
                    'secret' => $user->secret_key,
                    'img' => ''
                ]
            ]);
        }

        return json_encode(['success' => false]);
    }
    return json_encode(['success' => false]);
}

```

```

}

<?php
namespace frontend\models;

use Yii;
use yii\base\Model;
use common\models\User;

/**
 * Password reset request form
 */
class PasswordResetRequestForm extends Model
{
    public $email;

    /**
     * {@inheritdoc}
     */
    public function rules()
    {
        return [
            ['email', 'trim'],
            ['email', 'required'],
            ['email', 'email'],
            ['email', 'exist',
                'targetClass' => '\common\models\User',
                'filter' => ['status' => User::STATUS_ACTIVE],
                'message' => Yii::t('app', 'There is no user with this email address.')
            ],
        ];
    }

    public function attributeLabels()
    {
        return [
            'email' => Yii::t('app', 'Email'),
        ];
    }

    /**
     * Sends an email with a link, for resetting the password.
     *
     * @return bool whether the email was send
     */
    public function sendEmail()
    {
        /* @var $user User */
        $user = User::findOne([
            'status' => User::STATUS_ACTIVE,

```

```

        'email' => $this->email,
    ));

    if (!$user) {
        return false;
    }

    if (!$User::isPasswordResetTokenValid($user->password_reset_token)) {
        $user->generatePasswordResetToken();
        if (!$user->save()) {
            return false;
        }
    }

    if (Yii::$app->language == 'fa') {
        return Yii::$app
            ->mailer
            ->compose(
                ['html' => '@frontend/mail/passwordResetToken-fa-html'],
                ['user' => $user]
            )
            ->setFrom([Yii::$app->params['adminEmail'] => Yii::$app->params['noreplyEmail']])
            ->setTo($this->email)
            ->setSubject(Yii::t('app', 'Password reset'))
            ->send();
    } else {
        return Yii::$app
            ->mailer
            ->compose(
                ['html' => 'passwordResetToken-html', 'text' => 'passwordResetToken-text'],
                ['user' => $user]
            )
            ->setFrom([Yii::$app->params['adminEmail'] => Yii::$app->params['noreplyEmail']])
            ->setTo($this->email)
            ->setSubject(Yii::t('app', 'Password reset'))
            ->send();
    }
}
}
}

```

ДОДАТОК Б
Слайди презентації

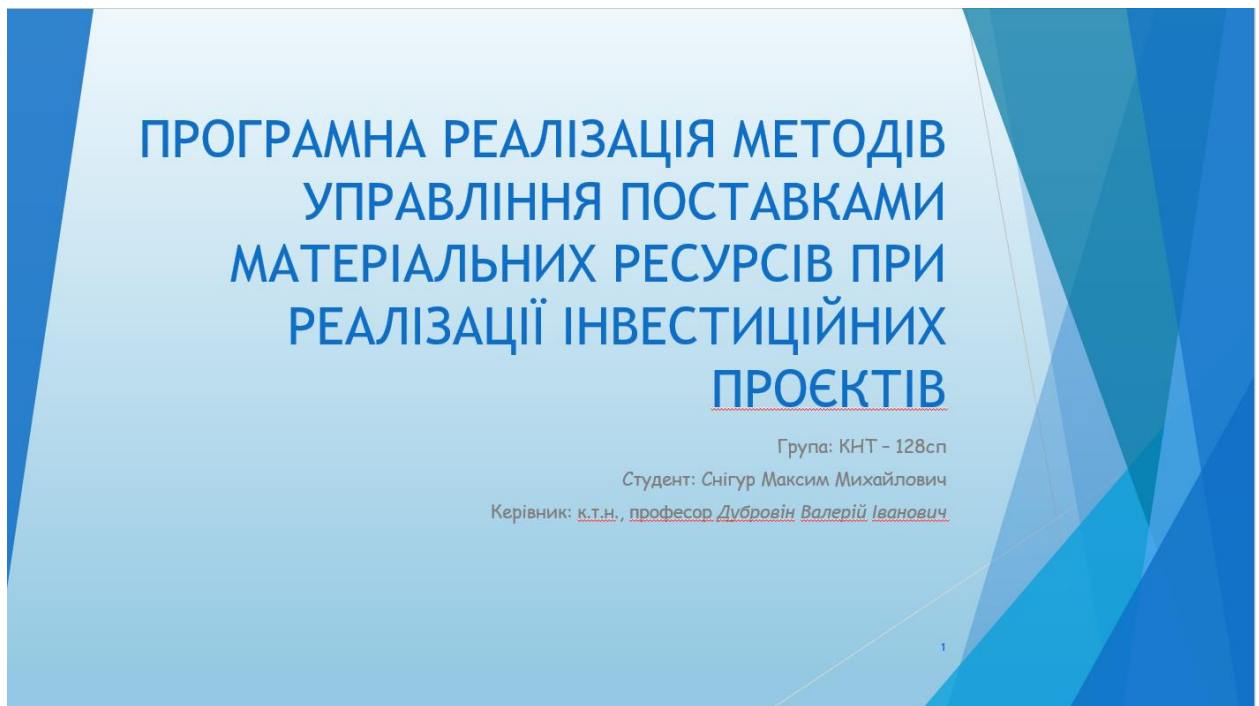


Рисунок Б.1 – Тема роботи

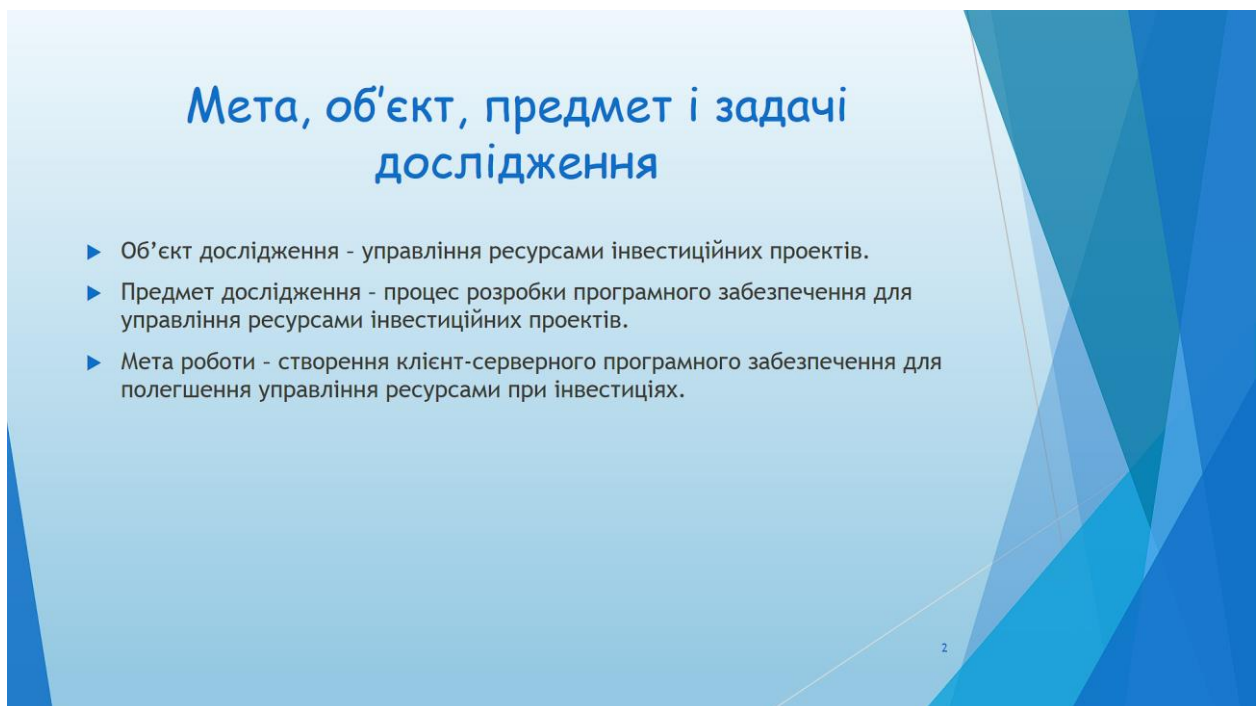


Рисунок Б.2 – Мета роботи та задачі дослідження

Огляд існуючих рішень amelok.com

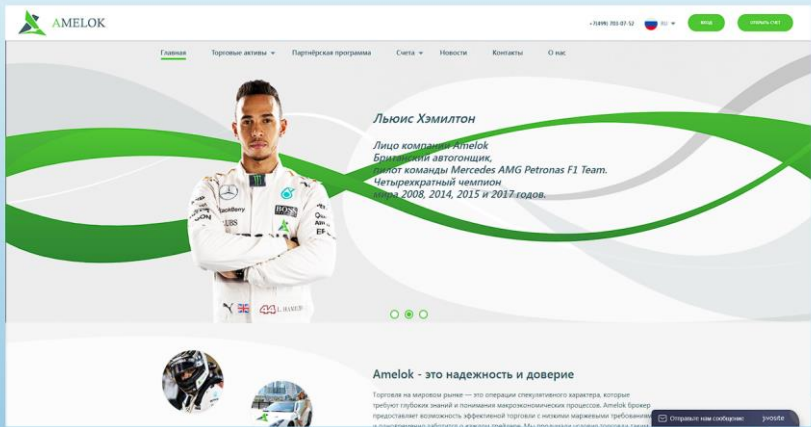


Рисунок Б.3 – Огляд існуючих рішень

Огляд існуючих рішень delta-capitals.com

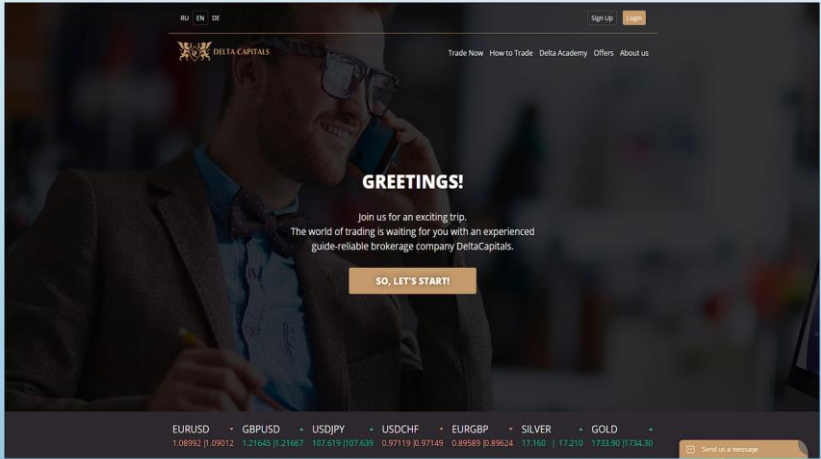


Рисунок Б.4 – Огляд існуючих рішень

Порівняння мов програмування

Критерії	Мови програмування		
	PHP	C#	Python
Динамічна типізація	+	+	+
Явна типізація	±	-	±
Неявна типізація	+	+	+
Багатоплатформність	+	+	+
Декларація типів та повертаємих значень	+	-	-

5

Рисунок Б.5 – Порівняння мов програмування

Вибір мови програмування

Мовою програмування було обрано PHP через його переваги:

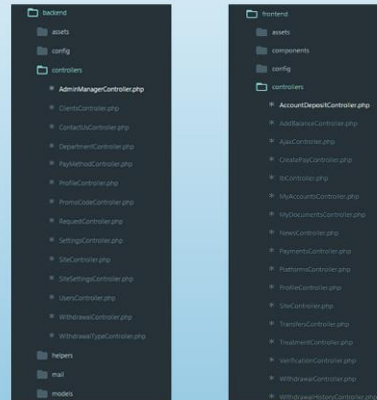
- ▶ ядро PHP дозволяє виконувати сценарії з дуже високою швидкістю;
- ▶ можливість прямого вбудовування у HTML код сторінок;
- ▶ багатоплатформність;
- ▶ хороша задокументованість;
- ▶ наявність простого та швидкого фреймворку Yii2;
- ▶ просте інтегрування REST API.

6

Рисунок Б.6 – Вибір мови програмування

Файлова структура адміністративної та користувацької частини

- /controllers - контролери;
- /views - сторінки (вигляди);
- /web - картинки, шрифти, каскадні таблиці стилів;
- common/config - загальні налаштування сайту;
- common/models - моделі.

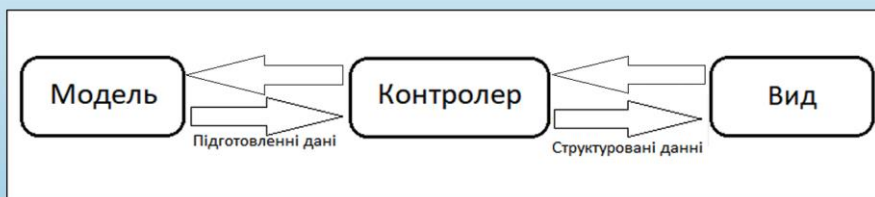


7

Рисунок Б.7 – Файлова структура проекту

Структура програми

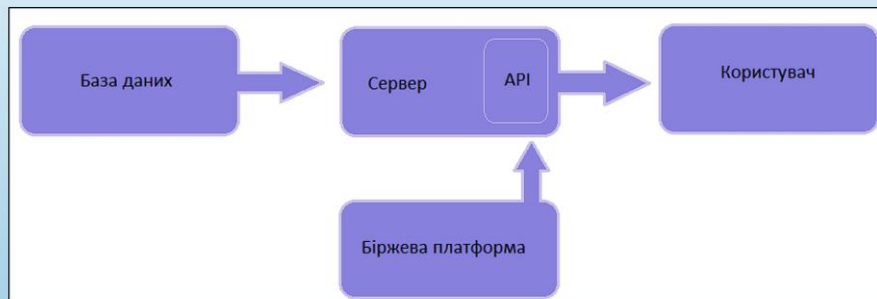
Після аналізу предметної області було обрано сучасний архітектурний підхід модель-вид-контролер (MVC)



8

Рисунок Б.8 – Структура програми

Загальна архітектура програми



9

Рисунок Б.9 – Загальна архітектура програми

Використані технології

- ▶ PHP
- ▶ Yii2
- ▶ MySQL
- ▶ JavaScript
- ▶ JQuery
- ▶ HTML
- ▶ CSS
- ▶ Bootstrap
- ▶ AJAX/PJAX

10

Рисунок Б.10 – Використані технології

Розроблені моделі

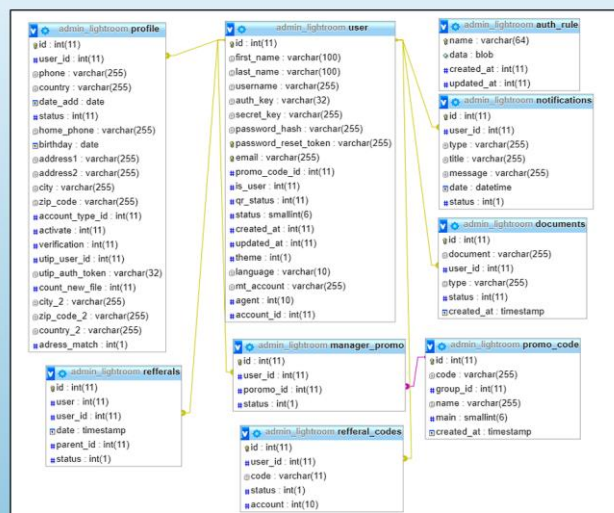
Під час розробки системи були розроблені наступні моделі:

- ▶ User
- ▶ UserProfile
- ▶ Statistic
- ▶ Request
- ▶ Notifications
- ▶ IbStatistic
- ▶ Patterns
- ▶ PayMethods
- ▶ PayFunctions
- ▶ Refferal
- ▶ RefferalCodes
- ▶ TmpCodes
- ▶ Statuses

11

Рисунок Б.13 – Розроблені моделі

Реляційна модель бази даних



12

Рисунок Б.12 – Реляційна модель бази даних

Сторінка «Open new account»

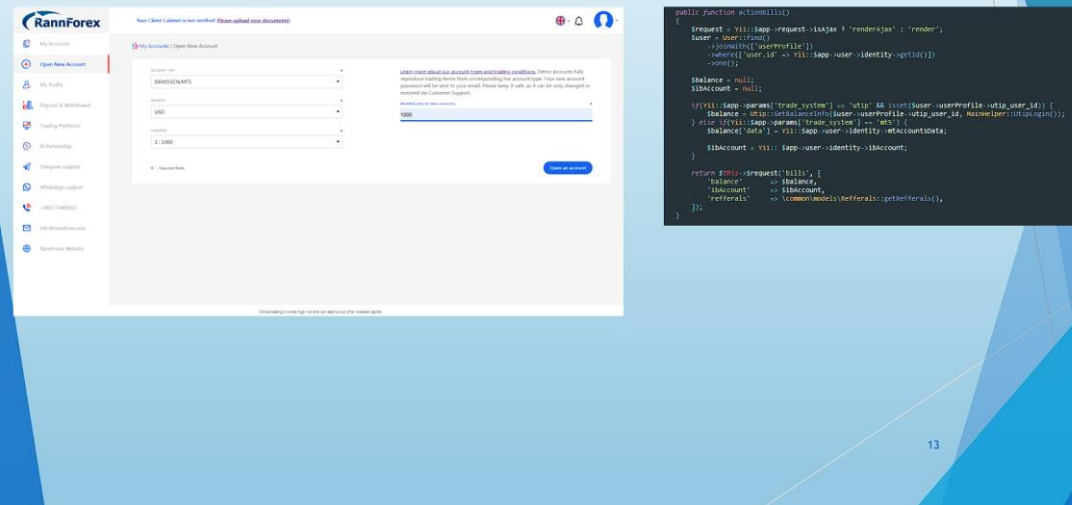


Рисунок Б.13 – Сторінка «Open new account»

Висновки:

Під час виконання дипломної кваліфікаційної роботи бакалавра було розроблено програмне забезпечення та інтерфейс для керування інвестиційними рахунками для біржової торгівлі.

Було проведено аналіз предметної області та сучасних технологій, враховані можливі труднощі при розробці програмного забезпечення, та на їх основі поставлено та вирішено наступні задачі:

- ▶ налаштування серверу;
- ▶ створення зрозумілого та простого користувацького інтерфейсу;
- ▶ налаштування взаємодії сервера та клієнта через AJAX;
- ▶ налаштування взаємодії з торгівельною платформою через REST API.

Рисунок Б.14 – Висновки