

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

кафедра програмних засобів

Звіт
з розрахунково-графічного завдання
з дисципліни «Архітектура та проектування програмного забезпечення»
на тему:
«Інтернет-аукціон (OnAuction)»

Виконав

ст. гр. КНТ-137

В.В. Козлов

Прийняв

доцент, к.т.н.

А.О. Олійник

асистент

С.Д. Леощенко

м. Запоріжжя

2020 рік

РЕФЕРАТ

ПЗ: 102 с., 42 рис., 6 таблиць, 3 додатки.

Об'єкт проектування – ASP.NET веб-застосунок.

Область дослідження – технологія створення веб-застосунків ASP.NET, та функціональні можливості мови програмування C#.

Тема проекту – розробка веб-застосунку для інтернет-аукціону (OnAuction).

Мета проекту – розробка веб-застосунку для інтернет-аукціону з мінімалістичним й інтуїтивно зрозумілим інтерфейсом, що дозволяє користувачу брати участь в онлайн аукціонах.

Веб-застосунок використовує шаблон проектування MVC та реалізує мережеву архітектуру клієнт-сервер.

Розроблений веб-застосунок відповідає таким критеріям:

- мінімалістичний й інтуїтивно зрозумілий інтерфейс;
- розмежування функцій на чотири групи користувачів: неавторизований користувач, покупець, продавець та адміністратор;
- наявність бази даних для зберігання інформації;

Веб-застосунок було розроблено за допомогою інтегрованого середовища розробки Microsoft Visual Studio, з використанням таких технологій, як .NET Framework, ASP.NET, C#, HTML, CSS, JavaScript, Razor, JQuery, Bootstrap, Entity Framework та MVC Framework.

Веб-застосунок призначений для роботи на пристроях, за допомогою одного з браузерів: Google Chrome, Microsoft Edge, Opera, Mozilla Firefox.

В даній роботі також розглянуто можливості використання сторонніх бібліотек, взаємодія з Entity Framework та MVC Framework.

ASP.NET, .NET, C#, HTML, CSS, JAVASCRIPT, BOOTSTRAP, JQUERY, RAZOR, SQL, ENTITY FRAMEWORK, MVC FRAMEWORK, WEB, WWW, MVC

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IDE – Integrated development environment

IIS – Internet Information Services

MVC – шаблон проектування Model-View-Controller (Модель-Представлення-Контролер)

RAM – Random Access Memory

SQL – Structured Query Language

БД – База даних

ОЗП – Оперативно запам'ятовуючий пристрій

ТЗ – Технічне завдання

ЗМІСТ

Реферат	2
Перелік умовних позначень	3
Зміст	4
Вступ.....	6
1 Вимоги до системи.....	7
1.1 Короткий опис проекту.....	7
1.1.1 Ціль проекту	7
1.1.2 Учасники проекту	7
1.2 Огляд існуючих аналогів програмного засобу.....	8
1.2.1 NewAuction (newauction.com.ua)	8
1.2.2 Violity (auction.violity.com).....	9
1.2.3 eBay (ebay.com).....	10
1.2.4 Підсумки огляду аналогів	11
1.3 Системні сервіси.....	12
1.3.1 Межі системи.....	12
1.3.2 Функціональні вимоги	12
1.3.3 Вимоги до даних.....	15
1.4 Системні обмеження	15
1.4.1 Вимоги до продуктивності.....	15
1.4.2 Вимоги до безпеки	15
1.4.3 Вимоги до інтерфейсу.....	15
1.4.4 Експлуатаційні вимоги	16
1.4.5 Проектні питання	16

2 Розробка моделей системи	18
2.1 Загальна схема роботи системи	18
2.2 Діаграма класів	19
2.3 Діаграма компонентів	21
2.4 Діаграма прецедентів	22
2.5 Діаграма станів	25
2.6 Діаграма діяльності	26
2.7 Діаграма кооперацій	26
2.8 Діаграма послідовностей	27
3 Архітектура та структура програми	28
3.1 Архітектура веб-застосунку на основі ASP.NET MVC Framework	29
4 Звернення та використання системи	39
4.1 Призначення й умови застосування програми	39
4.2 Звертання до програми	40
4.3 Початкові і вихідні дані	41
4.4 Використання програми	41
5 Тестування та аналіз результатів тестування	46
Висновки	47
Додаток А Технічне завдання	48
Додаток В Тексти програми	53
Додаток С Опис програми	100

ВСТУП

З кожним днем інформаційні технології, що з успіхом впроваджуються в повсякденне життя сучасної людини, стають її невід'ємною частиною. А комп'ютери, які раніше використовували тільки на промислових виробництвах, тепер застосовують в різних сферах діяльності: від стартапів та бізнес проектів до перегляду фільмів та геймінгу.

Однією з популярних галузей використання комп'ютера та мережі інтернет стала онлайн торгівля. Можливість маленьких компаній конкурувати за ринок з великими брендами сприяє стрімкому розвитку цього напрямку. Не стали винятком й інтернет-аукціони.

Онлайн застосунки, що реалізують функції звичних аукціонів в мережі інтернет користуються великою популярністю. На інтернет-аукціонах зараз продають і купують практично все що завгодно, починаючи від особистих речей та завершуючи ставками за рекламні пости й різноманітні сервіси та послуги.

Створення таких сервісів – це неймовірно цікавий та корисний досвід не тільки для новачка в програмуванні, а і для фахового спеціаліста. Адже при написанні таких веб-застосунків необхідно не тільки досконало знати використовувані технології розробки та програмування, а й добре розбиратися в предметній області.

Саме тому було прийнято рішення створити вільне програмне забезпечення, яке дозволяло би виконувати вищезазначені функції, та могло би використовуватися в якості інтернет аналога звичайному аукціону.

1 ВИМОГИ ДО СИСТЕМИ

1.1 Короткий опис проекту

1.1.1 Ціль проекту

Ціль проекту – розробка вільного програмного забезпечення, а саме веб-застосунку для інтернет-аукціону (OnAuction), який міг би використовуватися в якості інтернет аналога звичайному аукціону.

1.1.2 Учасники проекту

Учасники проекту: замовник, старший розробник, дизайнер, front-end розробник, back-end розробник, тестувальник.

Замовник – висуває вимоги до проекту та побажання щодо його реалізації, забезпечує фінансування на розробку системи, приймає розроблений продукт.

Старший розробник – співпрацює з замовником, розподіляє завдання серед підлеглих, супроводжує проект на всіх стадіях розробки, веде контроль за термінами виконання проекту, оформлює супроводжувальну документацію.

Дизайнер – розроблює макет інтерфейсу системи згідно з правилами ергономічності та вимогами до функціональності.

Back-end розробник – розроблює програмно-апаратну частину системи та реалізує програмну логіку відповідно до функціональних вимог.

Front-end розробник – розроблює клієнтську частину системи, а саме: реалізує користувацький інтерфейс відповідно до макету дизайнера, з'єднує його з програмно-апаратною частиною системи.

Тестувальник – перевіряє розроблену програмну систему на відсутність проблем при експлуатації, її відповідність до висунутих замовником вимог.

1.2 Огляд існуючих аналогів програмного засобу

Перед початком розробки необхідно дослідити доступні веб-застосунки на ринку, що виконують аналогічні функції. Проаналізувати їх переваги та недоліки й виділити основні технічні та програмні властивості, якими має володіти створюване програмне забезпечення.

1.2.1 NewAuction (newauction.com.ua)

NewAuction – це український інтернет-аукціон та торговий майданчик. Умови інтернет-аукціону відповідають типу відкритого або англійського аукціону.

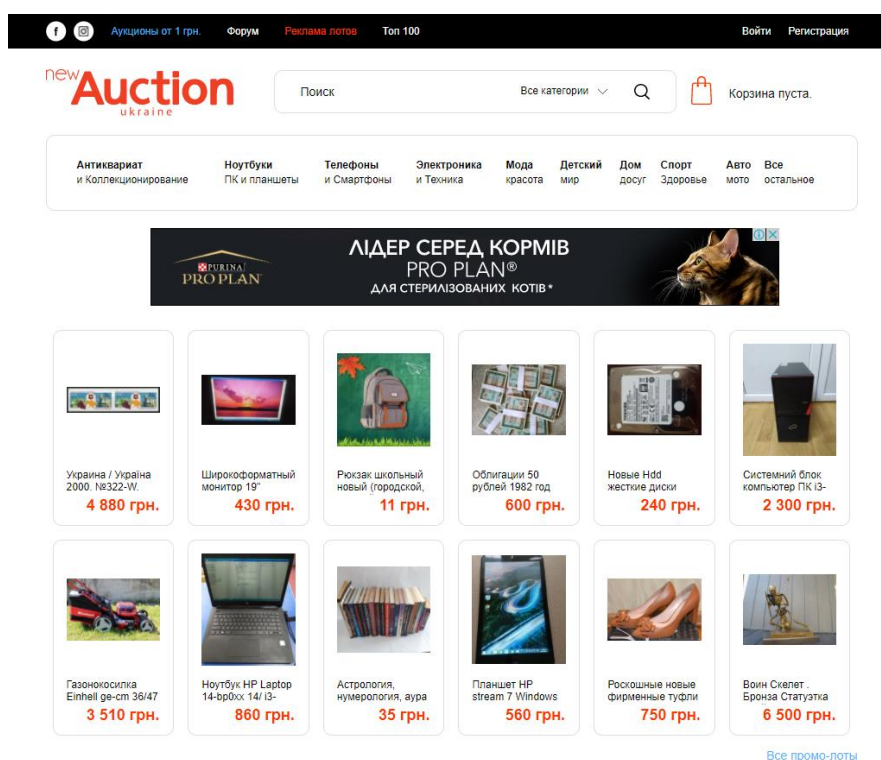


Рисунок 1.1 – Інтерфейс веб-застосунку NewAuction

Серед переваг веб-застосунку можна виділити такі:

- простий та інтуїтивно зрозумілий інтерфейс;
- адаптивність;

- наявність різних категорій товарів;
- грошова одиниця – гривня.

Серед недоліків веб-застосунку можна виділити такі:

- наявність нав'язливої реклами;
- відсутність підтримки української мови;
- закритий список всіх ставок на лот.

1.2.2 Violity (auction.violity.com)

Violity – це майданчик для продажу й покупки антикваріату та інших ексклюзивних предметів колекціонування в основі якого лежить використання аукціону. На даний момент цей сервіс є одним з найпопулярніших інтернет-аукціонів в Україні.

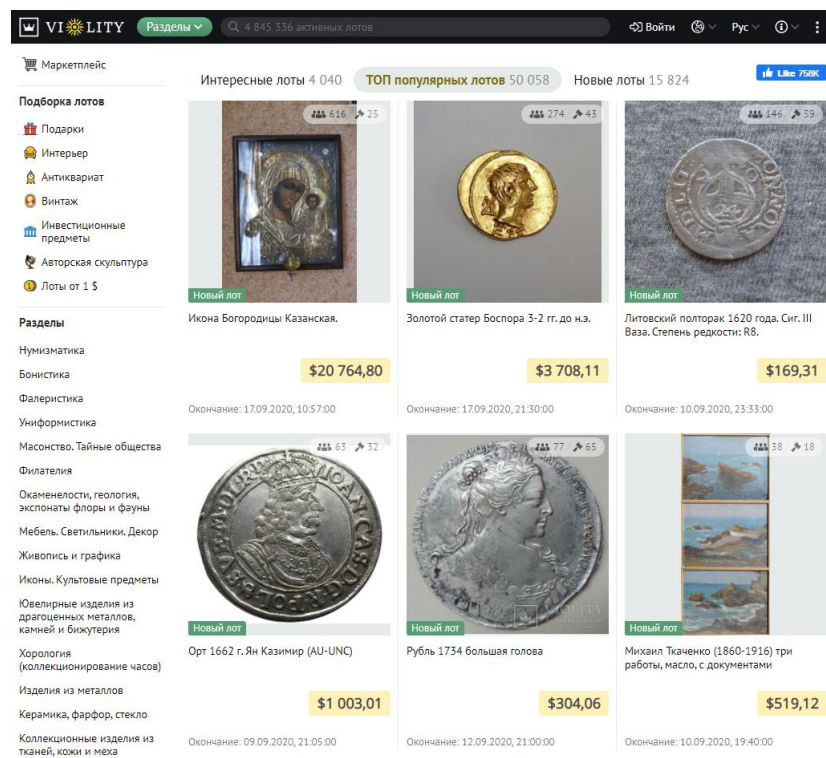


Рисунок 1.2 – Интерфейс веб-застосунку Violity

Серед переваг веб-застосунку можна виділити такі:

- простий та інтуїтивно зрозумілий інтерфейс;

- адаптивність;
- наявність різних категорій товарів;
- простота використання;
- підтримка української мови;
- відкритий список ставок на лот.

Серед недоліків веб-застосунку можна виділити такі:

- грошова одиниця – долар США;
- використовується тільки для антикваріату та колекціонування.

1.2.3 eBay (ebay.com)

eBay – веб-застосунок однойменної американської компанії, що надає послуги в областях інтернет-аукціонів і інтернет-магазинів. eBay є одним з найпопулярніших інтернет-аукціонів не тільки в Америці, а й у всьому світі. Серед представлених лотів можна знайти не тільки предмети колекціонування а й звичайні повсякденні речі та пристрої.

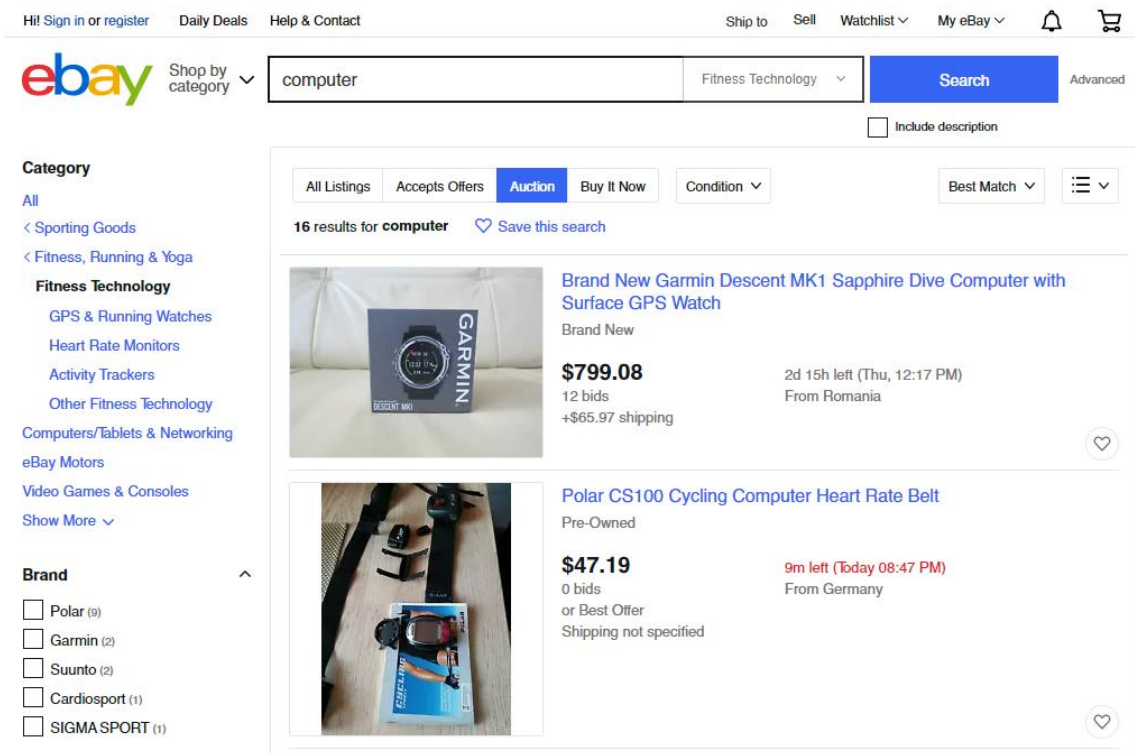


Рисунок 1.3 – Інтерфейс веб-застосунку eBay

Серед переваг веб-застосунку можна виділити такі:

- адаптивність;
- наявність різних категорій товарів;
- простота використання;
- відкритий список ставок на лот;

Серед недоліків веб-застосунку можна виділити такі:

- застарілий інтерфейс;
- відсутність підтримки української мови;
- грошова одиниця – долар США.

1.2.4 Підсумки огляду аналогів

В результаті проведення аналізу існуючих аналогів розроблюваного програмного забезпечення було визначено, що жоден з існуючих веб-застосунків не реалізує в повній мірі усіх функцій, що вимагаються, та повністю не орієнтований на українського користувача.

Розглянувши існуючі аналоги прийнято до уваги позитивне враження від простого, інтуїтивно зрозумілого та неперенавантаженого інтерфейсу користувача в системі в порівнянні з багатофункціональним але важким до сприйняття інтерфейсом.

В таблиці 1.1 наведено порівняльну характеристику веб-застосунку, що розроблюється, з існуючими аналогами.

Таблиця 1.1 – Порівняльна характеристика

	NewAuction	Violity	eBay	OnAuction
Простий та інтуїтивно зрозумілий інтерфейс	+	+	-	+
Адаптивність	+	+	+	+
Наявність різних категорій товарів	+	+	+	+
Відсутність нав'язливої реклами	-	+	+	+
Підтримка української мови	-	+	-	+
Основна грошова одиниця - гривня	+	-	-	+
Відкритий список ставок на лот	-	+	+	+
Орієнтованість на різні типи лотів	+	-	+	+

1.3 Системні сервіси

1.3.1 Межі системи

Заплановано розробку повного функціоналу програмного засобу та ведення документації, яка необхідна для супроводу та підтримки системи.

1.3.2 Функціональні вимоги

Система має клієнт-серверну архітектуру. На серверній стороні відбувається робота й обробка даних, а на стороні клієнта представлено інтерфейс для взаємодії з програмною системою.

Система повинна передбачати багаторівневий режим доступу до функціоналу: неавторизований користувач, покупець, продавець та адміністратор.

На рівні доступу «Неавторизований користувач» має бути реалізовано такі функції:

- можливість перегляду списку виставлених лотів;
- перегляд списку виставлених лотів за вибраною категорією;
- перегляд лоту;
- перегляд поточних ставок на лот;

На рівні доступу «Покупець» має бути реалізовано такі функції:

- можливість перегляду списку виставлених лотів;
- перегляд списку виставлених лотів за вибраною категорією;
- перегляд лоту;
- можливість зробити ставку на лот;
- можливість моментального викупу лоту;
- перегляд поточних ставок на лот;
- перегляд своїх активних ставок;
- перегляд своєї історії ставок.

На рівні доступу «Продавець» має бути реалізовано такі функції:

- можливість перегляду списку виставлених лотів;
- перегляд списку виставлених лотів за вибраною категорією;
- додавання нового лоту;
- перегляд лоту;
- перегляд поточних ставок на лот;
- перегляд своїх лотів.

На рівні доступу «Адміністратор» має бути реалізовано такі функції:

- можливість перегляду списку виставлених лотів;
- перегляд списку виставлених лотів за вибраною категорією;
- перегляд лоту;
- перегляд поточних ставок на лот;
- моніторинг ходу торгів;
- моніторинг вдалих угод.

Аналіз функціональних вимог до веб-застосунку «OnAuction» представлено у вигляді карти користувацьких історій (таблиця 1.2).

Таблиця 1.2 – Карта користувацьких історій сервісу «OnAuction»

Історія	Роль	Активність	Мета	Критерії прийнятності	Коментарі
Веб-застосунок					
Робота з акаунтом	Неавторизований користувач	Реєстрація акаунту	Створення нового акаунту користувача	Унікальне значення логіна користувача	Створення запису в базі даних
		Авторизація	Ідентифікація користувача	Відповідність логіна та пароля до існуючого запису в БД	
Робота з системою	Всі групи користувачів	Перегляд списку виставлених лотів	Переглянути список виставлених лотів	Тільки активні лоти	Відображення тільки активних лотів
		Перегляд списку лотів за вибраною категорією	Переглянути список лотів відповідної категорії	Категорія заздалегідь визначена. Тільки активні лоти	Відображення тільки активних лотів

Продовження таблиці 1.2

Історія	Роль	Активність	Мета	Критерії прийнятності	Коментарі
Робота з системою	Всі групи користувачів	Перегляд лоту	Переглянути інформацію про лот	Поле ставки на лот тільки для покупця	Переглянути можна не тільки активні лоти, а й завершені.
		Перегляд поточних ставок на лот	Переглянути список зроблених ставок на лот		Список ставок доступний навіть для завершеного лоту
	Покупець	Зробити ставку на лот	Зробити ставку на лот	Тільки для активного лоту	Ставка має бути вище ніж попередня з урахуванням кроку ставки або дорівнювати початковій вартості. Створення запису в базі даних
		Моментальний викуп лоту	Моментально викупити лот	Тільки для активного лоту.	Якщо ставка на лот вище вартості моментального викупу, то відбудеться моментальний викуп лоту
		Перегляд своїх активних ставок	Переглянути список своїх активних ставок	Свої активні ставки	В списку відображено не всі ставки на лот, а найбільші
		Перегляд своєї історії ставок	Переглянути список своєї історії ставок	Свої не активні ставки	В списку відображено найбільшу ставку й статус (виграш/програш)
		Додавання нового лоту	Додати новий лот		Створення запису в базі даних
	Продавець	Перегляд своїх лотів	Переглянути список своїх лотів	Тільки свої лоти	В списку відображено логін, найбільшу ставку й статус процесу
Керування системою	Адміністратор	Моніторинг ходу торгів	Перегляд списку зроблених ставок	Всі ставки	Інформація про лот, ставку та користувача
		Моніторинг вдалих угод	Перегляд списку завершених лотів	Всі завершені лоти	Детальний опис угоди

1.3.3 Вимоги до даних

Персональна інформація, лоти, категорії, ставки та вдалі угоди мають зберігатися в базі даних на сервері. Всі графічні файли з зображенням лотів, які завантажують користувачі, мають зберігатися у відповідній папці «Images» на сервері.

1.4 Системні обмеження

1.4.1 Вимоги до продуктивності

Програма повинна надавати реакцію щодо дії користувача на елемент керування швидше ніж за 0.5с.

1.4.2 Вимоги до безпеки

Враховуючи розмежування функціональних можливостей різних груп користувачів програмного засобу необхідно забезпечити авторизацію на основі реєстраційних даних.

1.4.3 Вимоги до інтерфейсу

Візуально зрозумілий та привабливий інтерфейс сприяє більш приємній та ефективній роботі з програмою. Він має бути реалізований інтуїтивно зрозумілим для користувача – непрофесіонала в комп'ютерній галузі.

При розробці графічного інтерфейсу за мету має бути визначено створення функціонального, проте простого, неперенавантаженого зайвою інформацією, інтерфейсу. Інтерфейс, створений за такими принципами

сприятиме зручному використанню веб-застосунку, та підвищить конкурентоздатність системи.

В процесі розробки графічного користувацького інтерфейсу перевага має бути віддана спокійним, однотонним відтінкам, які привертають увагу користувача.

За допомогою графічного інтерфейсу користувач має отримувати підказки стосовно правильності вводу даних та коректності своїх дій. Інтерфейс має коректно відображатися на різних пристроях, що відповідають мінімальним технічним вимогам, та в різних браузерах.

1.4.4 Експлуатаційні вимоги

Основною вимогою до серверної сторони є наявність машини Windows Server з підтримкою IIS (Internet Information Services) та системи керування реляційною базою даних MS SQL Server.

Як апаратно-технічні засоби для експлуатації програмного засобу на клієнтській стороні повинні використовуватися пристрої з доступом до мережі Internet та інтернет-браузером з оновленнями 2019 року. До складу засобів також повинні входити пристрій виводу інформації (монітор, дисплей) та маніпулятор вводу інформації (клавіатура, миша, сенсорний дисплей).

1.4.5 Проектні питання

Враховуючи поставлені цілі та завдання до веб-застосунку «OnAuction», розроблено графік роботи учасників над системою (табл. 1.3).

Таблиця 1.3 – Графік роботи учасників над системою «OnAuction»

Учасник	02.09 – 08.09	09.09 – 15.09	16.09 – 22.09	23.09 – 06.10	07.10 – 28.10	04.11 – 10.11	11.11 – 24.11	25.11 – 08.12
Замовник								

Продовження таблиці 1.3

Учасник	02.09 — 08.09	09.09 — 15.09	16.09 — 22.09	23.09 — 06.10	07.10 — 28.10	04.11 — 10.11	11.11 — 24.11	25.11 — 08.12
Ст. розробник								
Дизайнер								
Back-end розробник								
Front-end розробник								
Тестувальник								

Враховуючи графік роботи учасників розробки системи, а, також, ринкові ціни на відповідні послуги сплановано витрати на розробку системи (табл. 1.4).

Таблиця 1.4 – Витрати на розробку веб-застосунку «OnAuction»

Стаття	Вартість, грн.
Заробітна плата старшого розробника	7500
Заробітна плата дизайнера	2000
Заробітна плата Back-end розробника	4000
Заробітна плата Front-end розробника	3000
Заробітна плата тестувальник	1000
Амортизація обладнання	1500

Бюджет розробки системи є сумою всіх витрат на розробку системи та, відповідно, складає 17500 грн.

2 РОЗРОБКА МОДЕЛЕЙ СИСТЕМИ

2.1 Загальна схема роботи системи

На рисунку 2.1 зображено загальну схему роботи системи «OnAuction», що реалізує мережеву архітектуру клієнт-сервер.



Рисунок 2.1 – Загальна схема роботи системи «OnAuction»

Протягом роботи системи користувач надсилає запити до програми, взаємодіючи з елементами керування в графічному інтерфейсі користувача (веб-інтерфейсі). Перед відправкою даних на стороні клієнта відбувається їх попередня перевірка на валідність та відповідність до моделі. Перевірені дані надсилаються на сервер. Сервер отримує дані, виконує на своїй стороні перевірку на валідність та відповідність до моделі. Після перевірки отриманих даних, сервер, в разі необхідності, звертається до бази даних, яка повертає необхідні серверу дані. Отримані дані сервер надсилає до веб-інтерфейсу, де вони інтерпретуються у зрозумілій людині вигляд.

2.2 Діаграма класів

На рисунку 2.2 зображено діаграму класів веб-застосунку «OnAuction».

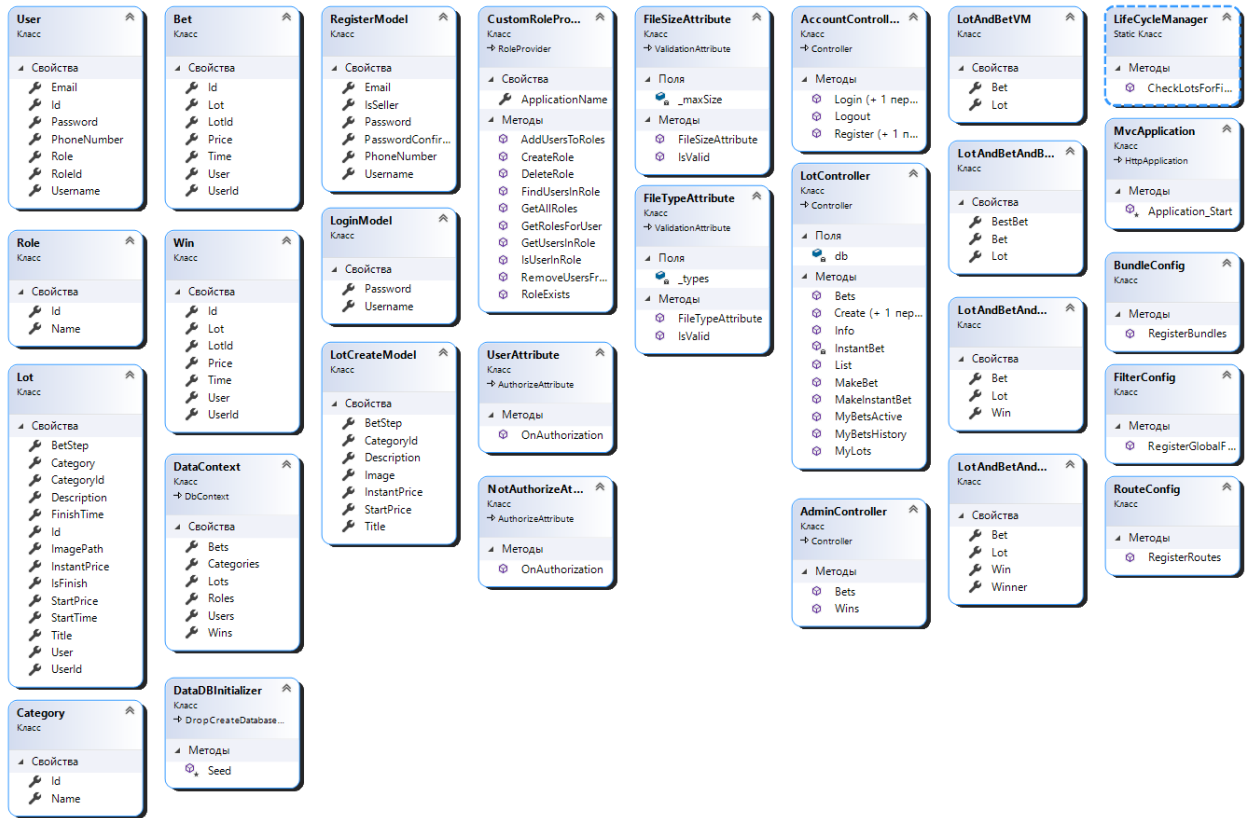


Рисунок 2.2 – Діаграма класів веб-застосунку «OnAuction»

Розроблена діаграма класів відображає всі наявні класи веб-застосунку:

- Клас User – клас, що описує модель користувач та використовується як елемент бази даних;
- Клас Role – клас, що описує модель ролі користувача та використовується як елемент бази даних;
- Клас Lot – клас, що описує модель лоту та використовується як елемент бази даних;
- Клас Category – клас, що описує модель категорії лоту та використовується як елемент бази даних;
- Клас Bet – клас, що описує модель ставки та використовується як елемент бази даних;

- Клас Win – клас, що описує модель вдалої угоди та використовується як елемент бази даних;
- Клас DataContext – клас, що реалізує контекст даних та є посередником між веб-застосунком та базою даних;
- Клас DataDBInitializer – клас, що відповідає за ініціалізацію бази даних;
- Клас RegisterModel – клас, що відповідає за модель реєстрації нового користувача;
- Клас LoginModel – клас, що відповідає за модель авторизації користувача;
- Клас LotCreateModel – клас, що відповідає за модель створення нового лоту;
- Клас CustomRoleProvider – клас, що реалізує користувацький провайдер ролей (менеджер ролей);
- Клас UserAttribute – клас, що реалізує атрибут авторизації User;
- Клас NotAuthorizeAttribute – клас, що реалізує атрибут авторизації NotAuthorize;
- Клас FileSizeAttribute – клас, що реалізує атрибут валідації для розміру файлу;
- Клас FileTypeAttribute – клас, що реалізує атрибут валідації для типу файлу (графічні зображення);
- Клас AccountController – клас, що відповідає за обробку запиту користувача, що направлений на контроллер Account;
- Клас LotController – клас, що відповідає за обробку запиту користувача, що направлений на контроллер Lot;
- Клас AdminController – клас, що відповідає за обробку запиту користувача, що направлений на контроллер Admin;
- Клас LotAndBetVM – додатковий клас, що використовується для передачі моделі в представлення (поєднує декілька моделей, а отже й має абривіатуру VM – ViewModel);

- Клас LotAndBetAndBestBetVM – додатковий клас, що використовується для передачі моделі в представлення;
- Клас LotAndBetAndWinVM – додатковий клас, що використовується для передачі моделі в представлення;
- Клас LotAndBetAndWinAndUserVM – додатковий клас, що використовується для передачі моделі в представлення;
- Клас LifeCycleManager – статичний клас, що відповідає за життєвий цикл веб-застосунку, а саме за обробку ходів та завершення аукціону;
- Клас MVCApplication – глобальний клас веб-застосунку, що відповідає за ініціалізацію системи та керування життєвим циклом;
- Клас BundleConfig – клас, що відповідає за реєстрацію додаткових модулів системи (розширень системи);
- Клас FilterConfig – клас, що відповідає за реєстрацію глобальних фільтрів (атрибути авторизації, валідації, тощо);
- Клас RouteConfig – клас, що відповідає за визначення системи маршрутизації в веб-застосунку.

2.3 Діаграма компонентів

На рисунку 2.3 зображено діаграму компонентів веб-застосунку «OnAuction».

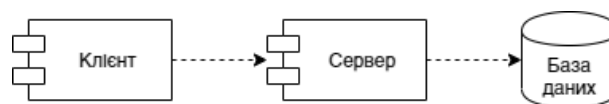


Рисунок 2.3 – Діаграма компонентів веб-застосунку «OnAuction»

Програмна система складається з таких частин:

- клієнтська частина (веб-інтерфейс): відповідає за отримання запитів від користувача, попередню валідацію введених даних, надсилання даних та отримання даних з сервера, інтерпретацію даних у зрозумілий для людини вигляд;

- серверна частина (сервер): відповідає за обробку запитів, валідацію отриманих даних та надання відповіді на запит;
- база даних: відповідає за зберігання даних та надання необхідної інформації на SQL запити.

2.4 Діаграма прецедентів

Розроблену та побудовану діаграму прецедентів веб-застосунку, на якій зображено відношення між акторами та прецедентами в системі, зображено на рисунку 2.4.

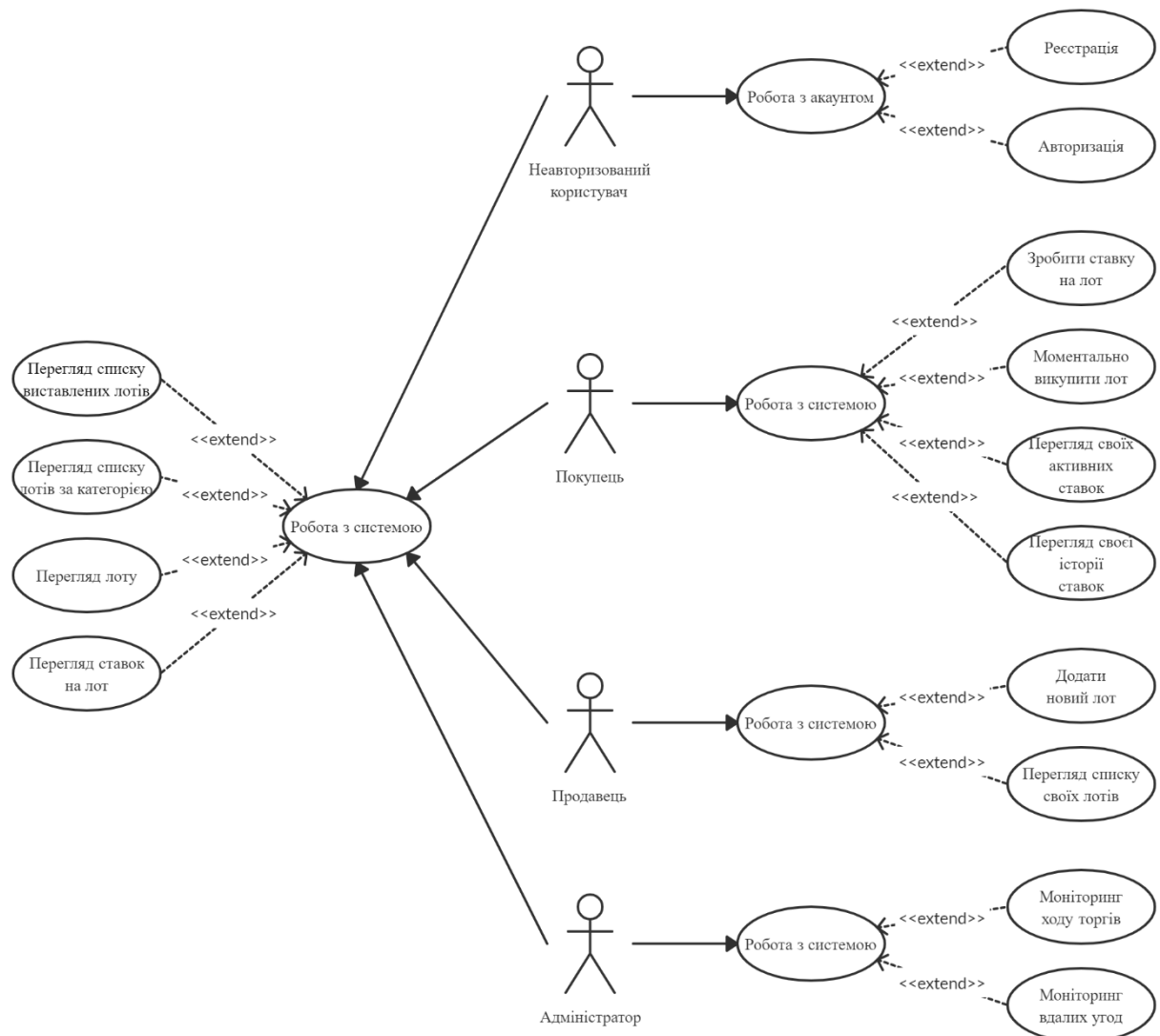


Рисунок 2.4 – Діаграма прецедентів веб-застосунку «OnAuction»

Для роз'яснення діаграми прецедентів наведено етапи потоків системи (табл. 2.1).

Таблиця 2.1 – Етапи потоків системи

Основний потік	Альтернативні потоки	Потоки помилок
1. Авторизація користувача	A.1 Реєстрація користувача	E.1 Відображення помилки про некоректність даних в реєстраційній формі E.2 Помилка авторизації
2 Користувач переглядає список лотів		
3 Користувач переглядає лот та супутню інформацію	A.2 Лот не існує	
4 Користувач додає новий лот		E.3 Помилка про невідповідність прав доступу користувача E.4 Відображення помилки про некоректність даних в лоті
5 Користувач переглядає список своїх лотів		E.3 Помилка про невідповідність прав доступу користувача
6 Користувач робить ставку на лот	A.3 Лот завершений	E.4 Відображення помилки про некоректність даних в ставці
7 Користувач переглядає свої ставки		E.3 Помилка про невідповідність прав доступу користувача
8 Користувач переходить до панелі адміністратора		E.3 Помилка про невідповідність прав доступу користувача

Етапи основного потоку подій:

1 Авторизація користувача.

A.1 Реєстрація користувача.

E.1 Відображення помилки про некоректність даних в реєстраційній формі.

E.2 Помилка авторизації.

2 Користувач переглядає список лотів та ставок.

3 Користувач переглядає лот та супутню інформацію.

A.2 Лот не існує.

4 Користувач додає новий лот.

Е.3 Помилка про невідповідність прав доступу користувача.

Е.4 Відображення помилки про некоректність даних в лоті.

5 Користувач переглядає список своїх лотів

Е.3 Помилка про невідповідність прав доступу користувача.

6 Користувач робить ставку на лот

А.3 Лот завершений

Е.3 Помилка про невідповідність прав доступу користувача.

7 Користувач переглядає свої ставки

Е.3 Помилка про невідповідність прав доступу користувача.

8 Користувач переходить до панелі адміністратора

Е.3 Помилка про невідповідність прав доступу користувача.

Етапи альтернативних потоків:

А.1 Реєстрація користувача

1 Користувач переходить на сторінку реєстрації.

2 Користувач заповнює необхідні поля реєстраційної форми.

3 Користувач підтверджує реєстрацію.

4 Якщо користувач ввів неправильно дані, то перехід до Е.1.

Е.1 Відображення помилки про некоректність даних в реєстраційній формі

А.2 Лот не існує

1 Якщо лот не існує то повернення коду 404.

А.3 Лот завершений

1 Користувач втрачає можливість зробити ставку на такий лот.

Етапи потоків помилок:

Е.1 Відображення помилки про некоректність даних в реєстраційній формі

1 Відображення помилки про некоректність даних поруч з полями в реєстраційній формі.

Е.2 Помилка авторизації

1 Відображення інформації про неможливість авторизації користувача (неправильно вказано логін або пароль, акаунту з таким логіном не існує).

Е.3 Помилка про невідповідність прав доступу користувача

1 Якщо користувач не є частиною групи користувачів, яким дозволено виконати відповідні дії, то повернення коду 404.

Е.4 Відображення помилки про некоректність даних в лоті

1 Відображення помилки про некоретність даних поруч з полями в формі створення лоту.

2.5 Діаграма станів

Розроблено і побудовано діаграму станів для головних процесів програми: процес існування лоту та процес існування ставки (рис. 2.5).

Лот при додаванні в систему знаходиться в стані «Новий лот». Після валідації даних та підтвердження створення він переходить до стану «Активний лот», а після завершення аукціону – до стану «Завершений лот».

Ставка при додаванні в систему знаходиться в стані «Нова ставка». Після валідації даних та додавання ставки вона переходить до стану «Активна ставка», а після завершення аукціону – до стану «Зіграна ставка».



Рисунок 2.5 – Діаграма станів веб-застосунку «OnAuction»

2.6 Діаграма діяльності

Розроблено і побудовано діаграму діяльності програми, яка показує переходи між різними видами діяльності (рис. 2.6).

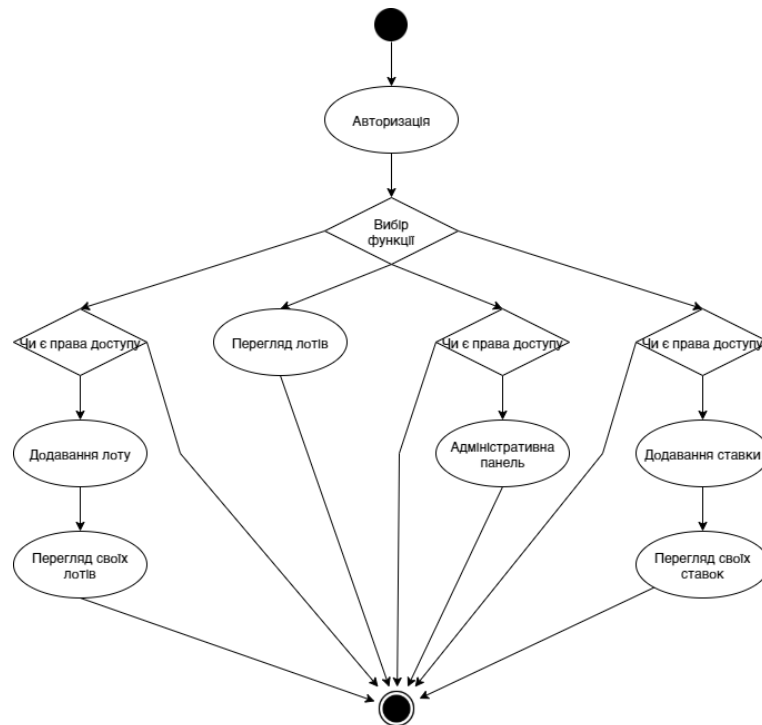


Рисунок 2.6 – Діаграма діяльності веб-застосунку «OnAuction»

2.7 Діаграма кооперацій

Розроблено та побудовано діаграму кооперацій системи (рис. 2.7).

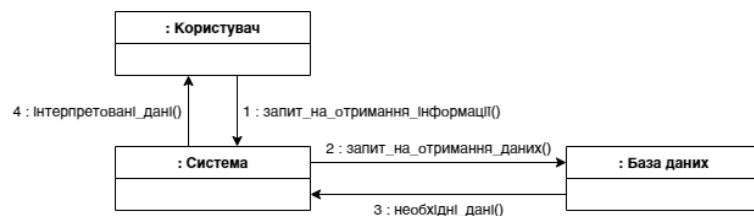


Рисунок 2.7 – Діаграма кооперації веб-застосунку «OnAuction»

Користувач надсилає запит до системи. Система оброблює запит і, в разі потреби, звертається до бази даних. База даних, при звертанні системи до неї,

надає відповідні дані. Система оброблює дані до прийнятного для користувача вигляду та відображає їх.

2.8 Діаграма послідовностей

Розроблено і побудовано діаграму послідовностей веб-застосунку, яка показує учасників взаємодій та послідовність повідомлень, якими вони обмінюються (рис. 2.8). Розглянуто взаємодію авторизованого користувача, системи та бази даних.

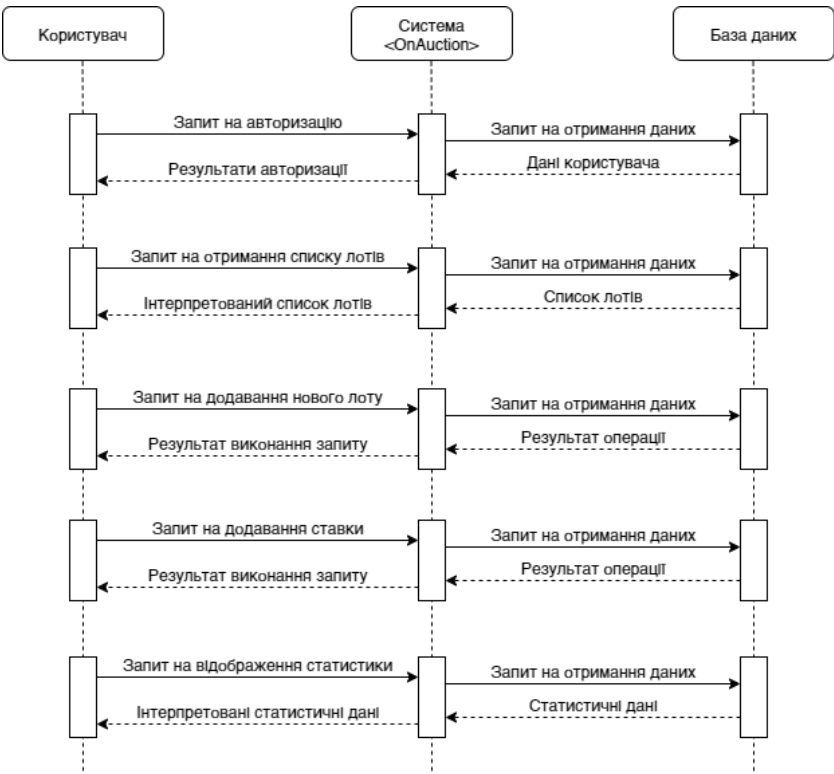


Рисунок 2.8 – Діаграма послідовностей веб-застосунку «OnAuction»

3 АРХІТЕКТУРА ТА СТРУКТУРА ПРОГРАМИ

Золотим стандартом архітектури, що використовується для побудови будь-якого веб-застосунку вважається шаблон проектування MVC. Архітектура сервісу «OnAuction» на основі предметної області була побудована за допомогою саме цього шаблону проектування.

Шаблон MVC (Model-View-Controller) – це фундаментальний шаблон проектування, який знайшов застосування в багатьох технологіях, однією з яких є розробка веб-систем. Цей шаблон передбачає поділ системи на три взаємопов'язані частини: модель даних (модель), інтерфейс користувача (представлення) та модуль керування (контролер). Застосовується для відокремлення даних (моделі) від інтерфейсу користувача (представлення) так, щоб зміни інтерфейсу користувача мінімально впливали на роботу з даними, а зміни в моделі даних могли здійснюватися без змін інтерфейсу користувача.

В архітектурі системи «OnAuction» модель представляє собою набір класів що використовується для зберігання, обробки та обміну даними з базою даних. Модель представлена файлами у директорії «Models». До представлення відносяться файли у директорії «Views», а до контролерів файли у директорії «Controllers». При цьому розташування і найменування файлів повинно строго відповідати цим вимогам. Схема архітектури системи на основі предметної області зображена на рисунку 3.1.

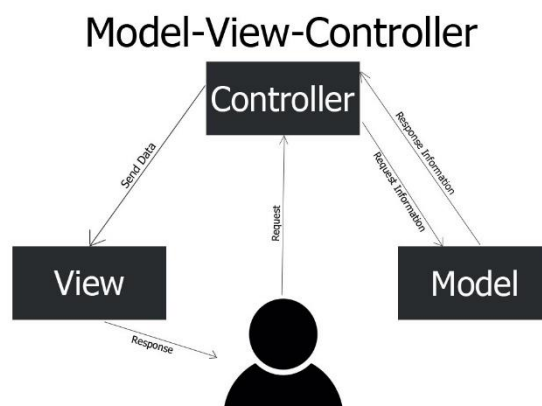


Рисунок 3.1 – Архітектура системи на основі шаблону MVC

3.1 Архітектура веб-застосунку на основі ASP.NET MVC Framework

В якості середовища розробки було обрано IDE Microsoft Visual Studio. Версії Visual Studio є абсолютно безкоштовними для учнів, студентів та розробників програм з відкритим програмним кодом.

Для розробки веб-застосунку було обрано відкритий C# .NET фреймворк ASP.NET MVC Framework, який реалізує шаблон MVC і має всі необхідні засоби для якісної та швидкої розробки проекту.

Використання сторонніх бібліотек допомагає розширювати функціонал програмного забезпечення при цьому зменшуючи час розробки та створюючи програмний код, що можна використовувати як вже налагоджений компонент. Зважаючи на це, було прийнято рішення використовувати систему NuGet. NuGet – система управління пакетами для платформ розробки Microsoft, в першу чергу бібліотек .NET Framework.

До компонентів архітектури ASP.NET MVC веб-застосунку (рис. 3.2) входять:

- список підключених сервісів (Connected Services);
- налаштування властивостей системи (Properties);
- перелік посилань на сторонні бібліотеки (Посилання);
- набір файлів для зберігання даних програми (App_Data);
- файли, що відповідають за запуск та ініціалізацію веб-застосунку (App_Start, Global.asax);
- набір статичних файлів, що використовуються в системі (Content);
- набір файлів, що відповідають за обробку запитів користувачів (Controllers);
- набір файлів зі шрифтами, що використовуються (fonts);
- набір файлів, що відповідають за бізнес логіку та обмін даними з базою даних (Model);
- набір Java Script файлів (Scripts);

- набір файлів, що відповідають за логіку відображення даних – представлення (View);
- логотип веб-застосунку (favicon.ico);
- файли конфігурацій системи (Web.config, packages.config).

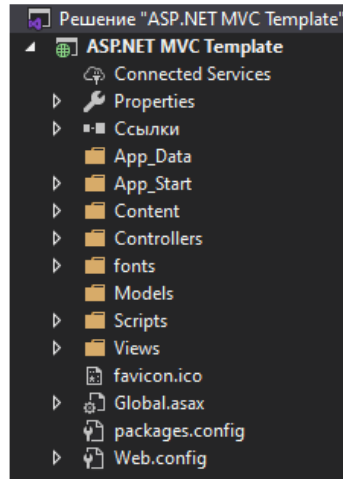


Рисунок 3.2 – Компоненти архітектури ASP.NET MVC веб-застосунку

Структура архітектури ASP.NET MVC веб-застосунку зображена на рисунку 3.3.

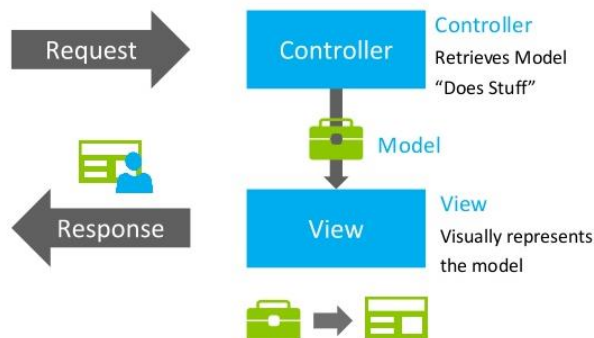


Рисунок 3.3 – Структура архітектури ASP.NET MVC веб-застосунку

В результаті аналізу архітектури веб-застосунку ASP.NET MVC було прийнято рішення додати декілька власних компонентів:

- Attributes – набір класів, що відповідають за атрибути валідації даних на стороні серверу;
- Filters – набір класів, що відповідають за атрибути авторизації користувача;

- Images – директорія для завантаження користувацьких зображень. Більш економний підхід для роботи з зображеннями на відміну від їх зберігання в базі даних. Зображення зберігається на сервері, а в базу даних записується шлях до файлу;

- LifeCycle – набір класів, що відповідають за життєвий цикл системи;

- Providers – набір класів, що реалізують менеджер ролей. Дозволяє створити різні групи користувачів та керувати їх доступом до методів контролера.

На основі вдосконаленої архітектури веб-застосунку було побудовано структуру файлів проекту (рис. 3.4).

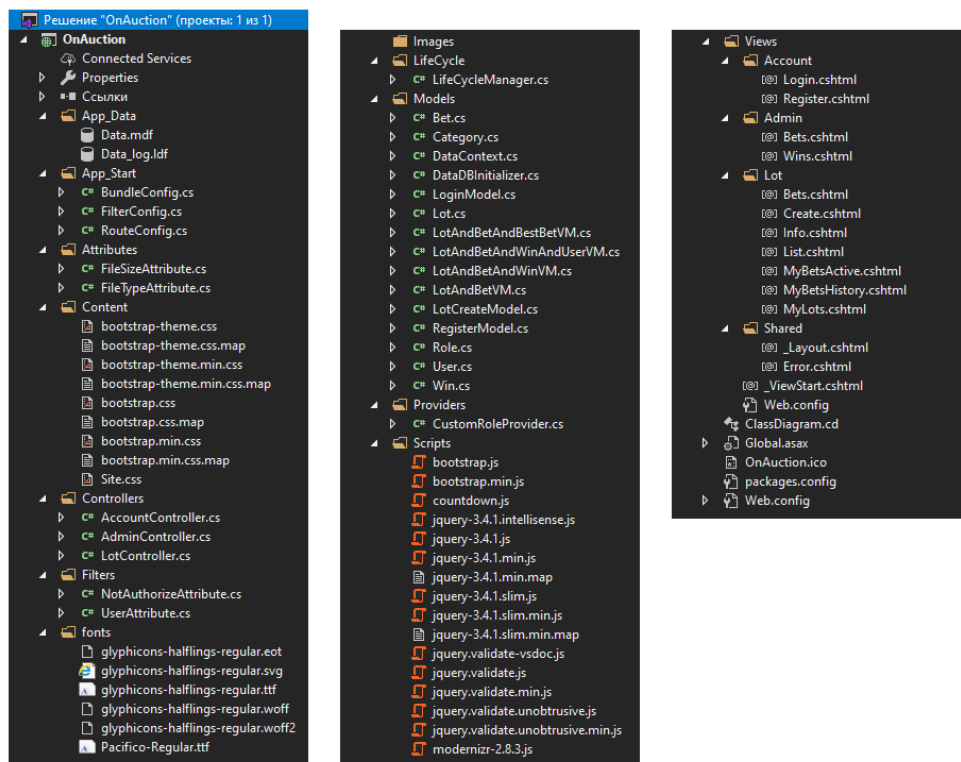


Рисунок 3.4 – Структура файлів веб-застосунку «OnAuction»

Усі файли в директорії «Controllers» мають суфікс Controller та відповідають за обробку запитів користувачів.

Усі файли в директорії «Models» описують моделі та бізнес-логіку.

Усі піддиректорії в директорії «Views» повинні відповідати назві свого контролера, а файли у них відповідним методам «Action».

Усі файли в директорії «Attributes» та «Filters» мають суфікс Attribute та реалізують атрибути валідації та авторизації відповідно.

В якості системи керування базою даних було обрано Microsoft SQL Server, а саме Microsoft SQL LocalDB. Для реалізації взаємодії веб-застосунку та бази даних було використано NuGet пакет Entity Framework та підхід Code First для створення бази даних.

Entity Framework – це спеціальна об’єктно-орієнтована технологія на базі фреймворка .NET для роботи з даними. Якщо традиційні засоби ADO.NET дозволяють створювати підключення, команди та інші об’єкти для взаємодії з базами даних, то Entity Framework являє собою більш високий рівень абстракції, який дозволяє абстрагуватися від самої бази даних і працювати з даними незалежно від типу сховища. Якщо на фізичному рівні ми оперуємо таблицями, індексами, первинними і зовнішніми ключами, то на концептуальному рівні, який нам пропонує Entity Framework, ми вже працюємо з об’єктами (вони ж і є класи, що визначені в моделі).

На рисунку 3.5 зображено спроектовану схему бази даних веб-застосунку «OnAuction».

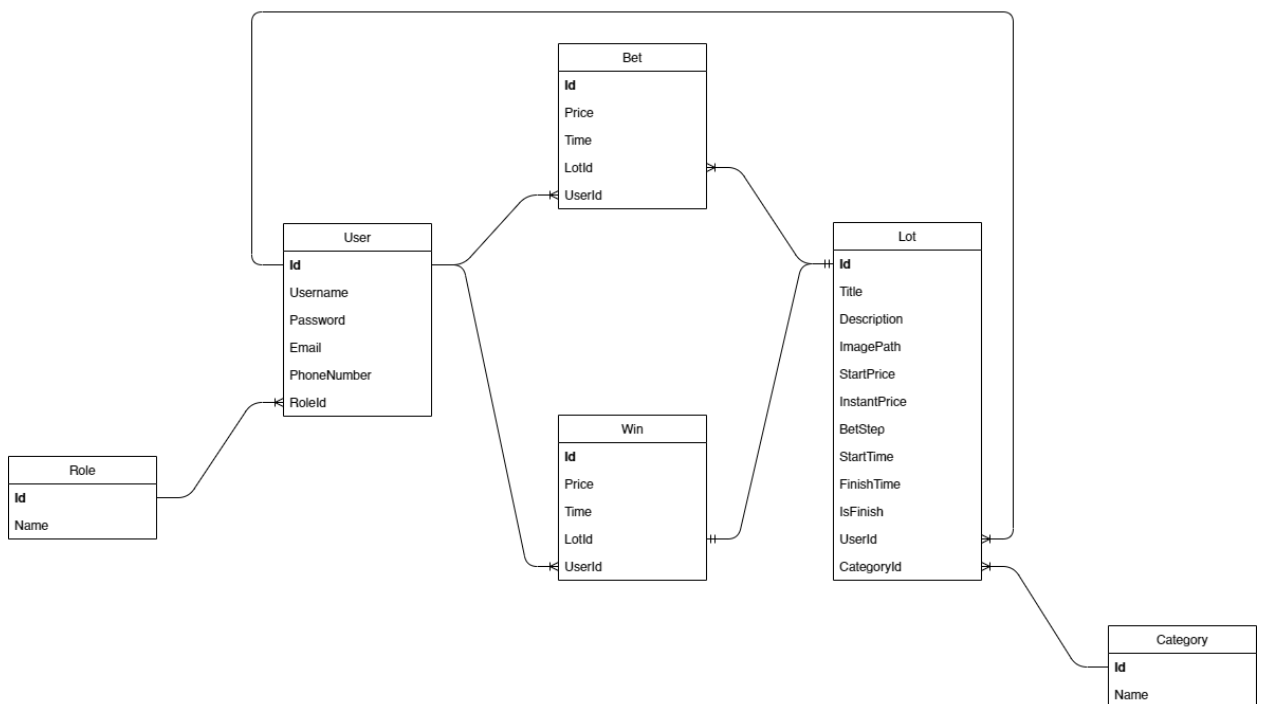


Рисунок 3.5 – Схема бази даних веб-застосунку «OnAuction»

Під час розробки графічного інтерфейсу користувача було використано вільний набір інструментів для створення сайтів і веб-додатків Bootstrap.

Bootstrap – це безкоштовний набір інструментів з відкритим кодом, призначений для створення веб-сайтів та веб-додатків, який містить шаблони CSS та HTML для типографіки, форм, кнопок, навігації та інших компонентів інтерфейсу, а також додаткові розширення JavaScript. Він спрощує розробку динамічних веб-сайтів і веб-додатків.

Для додавання динамічності веб-сторінкам та валідації даних на стороні клієнта було використано популярну JavaScript-бібліотеку з відкритим кодом jQuery.

jQuery – це 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.

Інструменти Bootstrap, jQuery та jQuery.Validation було завантажено за допомогою системи керування пакетами NuGet.

Відповідно до технічного завдання в системі повинно бути забезпечене розмежування користувачів на чотири групи: «неавторизовані користувачі», «покупці», «продавці» та «адміністратори». Отже необхідно забезпечити перевірку прав доступу, можливість реєстрації та авторизації користувачів в системі.

Для реалізації даної задачі доцільно використовувати існуючі методи. В якості способу аутентифікації користувача було обрано підхід під назвою «Аутентифікація форм». Аутентифікація форм ґрунтується на основі використання куки наборів.

Основний етап підходу під назвою «Аутентифікація форм» – це створення аутентифікаційного тікета – тобто деякого квитка безпеки, за яким веб-додаток буде впізнавати користувача. Веб-застосунок встановлює цей тікет для браузера у вигляді куки-набору а потім за необхідністю перевіряє його та верифікує користувача.

На рисунку 3.6 зображено інтерфейс головної сторінки веб-застосунку «OnAuction».

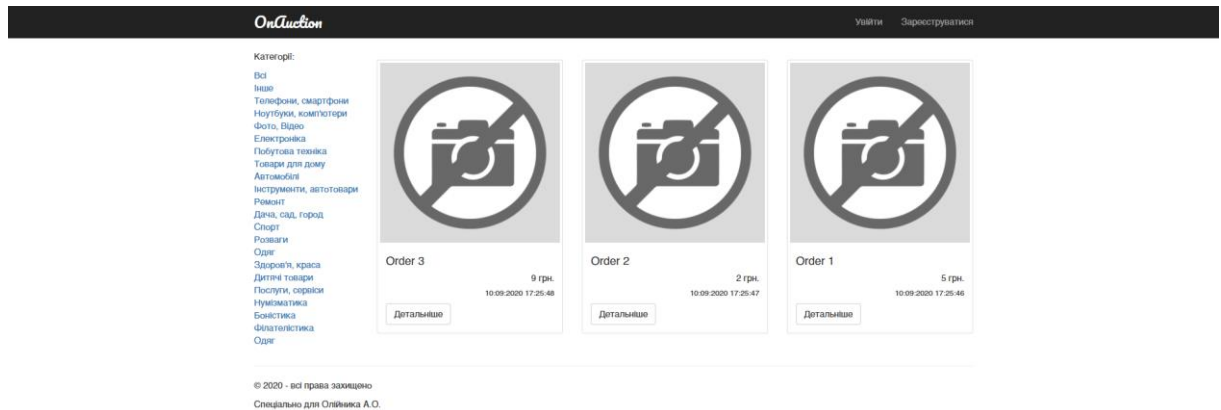


Рисунок 3.6 – Інтерфейс головної сторінки веб-застосунку «OnAuction»

На рисунку 3.7 зображено інтерфейс сторінки авторизації.

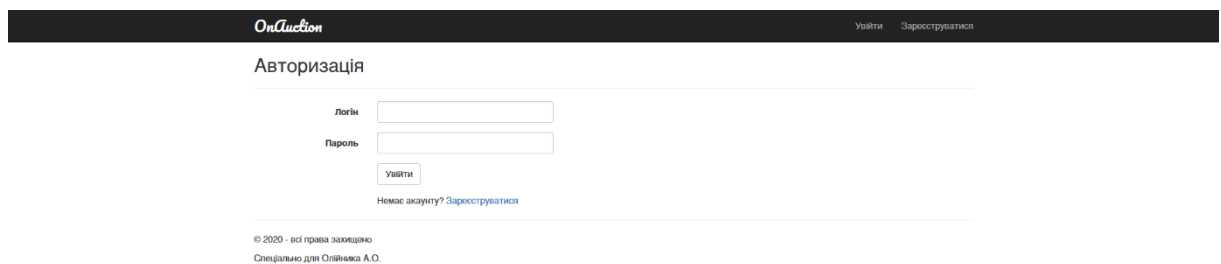


Рисунок 3.7 – Інтерфейс сторінки авторизації

На рисунку 3.8 зображено інтерфейс сторінки реєстрації.

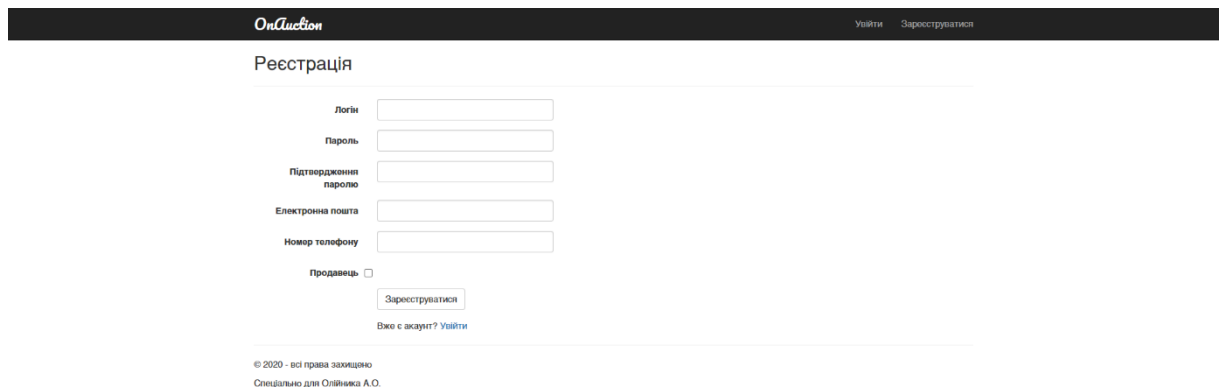


Рисунок 3.8 – Інтерфейс сторінки реєстрації

На рисунку 3.9 зображено інтерфейс сторінки перегляду активного лоту користувачем в ролі «покупець».

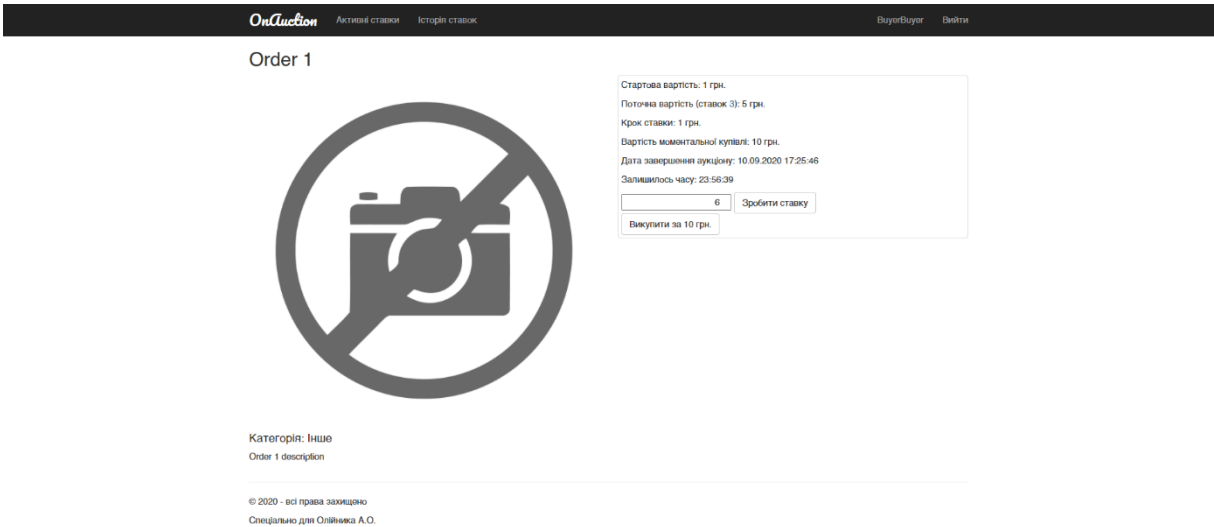


Рисунок 3.9 – Інтерфейс сторінки перегляду активного лоту

На рисунку 3.10 зображено інтерфейс сторінки перегляду завершеного лоту.

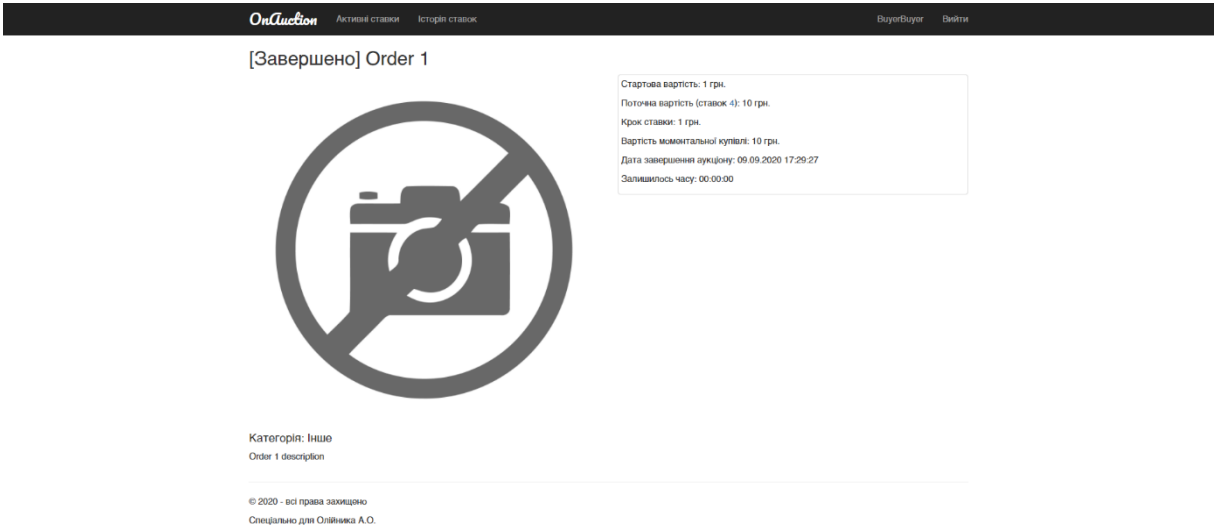


Рисунок 3.10 – Інтерфейс сторінки перегляду завершеного лоту

На рисунку 3.11 зображено інтерфейс сторінки перегляду ставок на лот.

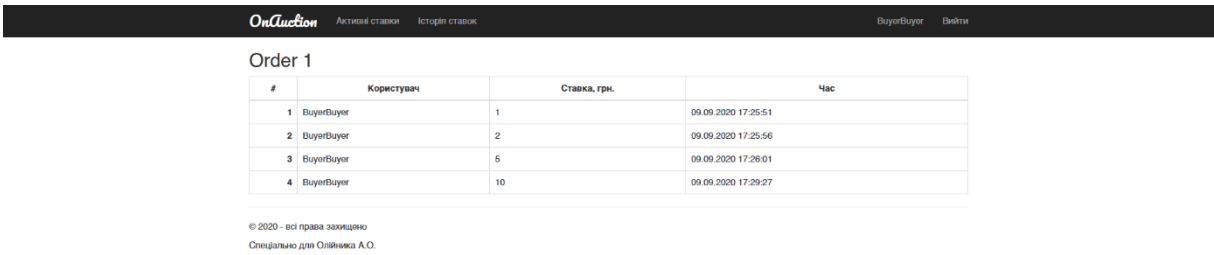


Рисунок 3.11 – Інтерфейс сторінки перегляду ставок на лот

На рисунку 3.12 зображено інтерфейс сторінки перегляду своїх активних ставок.

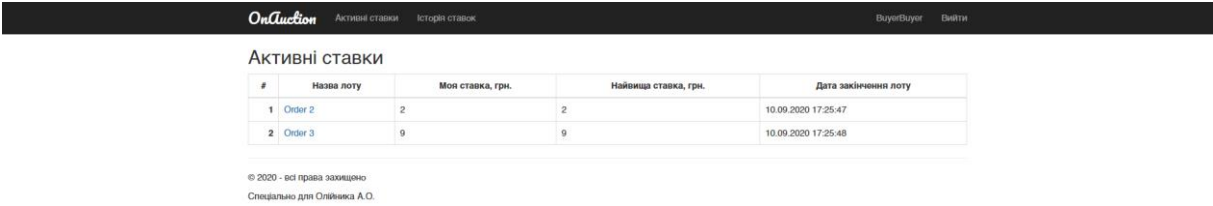


Рисунок 3.12 – Інтерфейс сторінки перегляду своїх активних ставок

На рисунку 3.13 зображено інтерфейс сторінки перегляду своїх ставок.

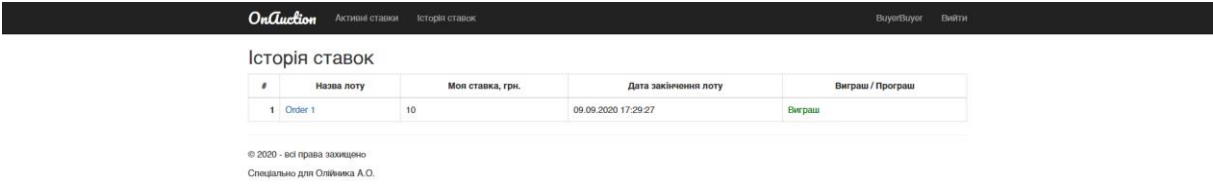


Рисунок 3.13 – Інтерфейс сторінки перегляду історії своїх ставок

На рисунку 3.14 зображено інтерфейс сторінки додавання нового лоту.

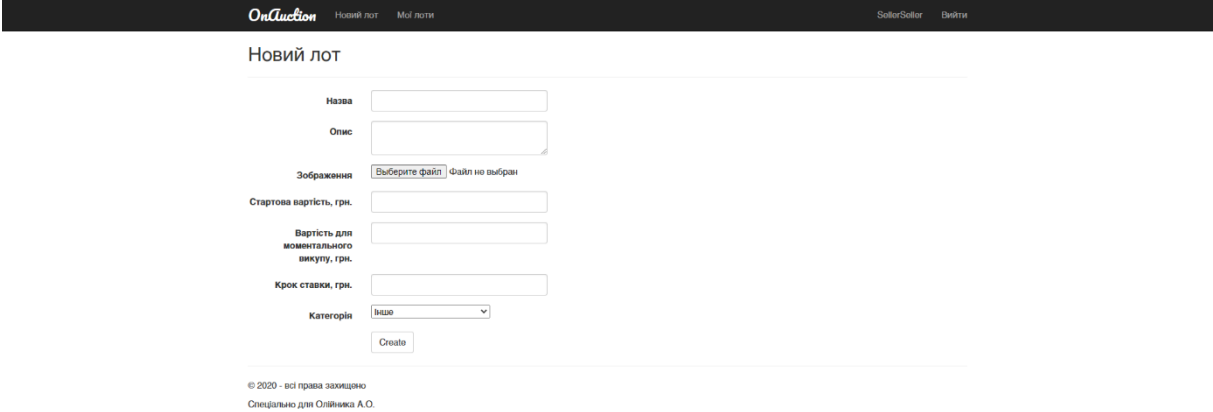


Рисунок 3.14 – Інтерфейс сторінки додавання нового лоту

На рисунку 3.15 зображено інтерфейс сторінки перегляду своїх лотів.

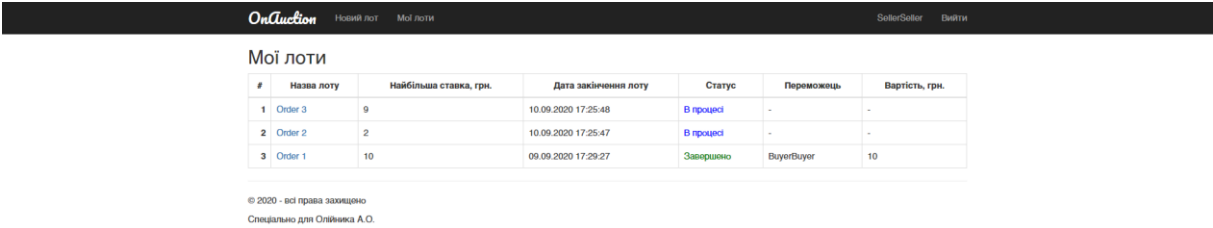


Рисунок 3.15 – Інтерфейс сторінки перегляду своїх лотів

На рисунку 3.16 зображено інтерфейс сторінки перегляду всіх ставок в системі.

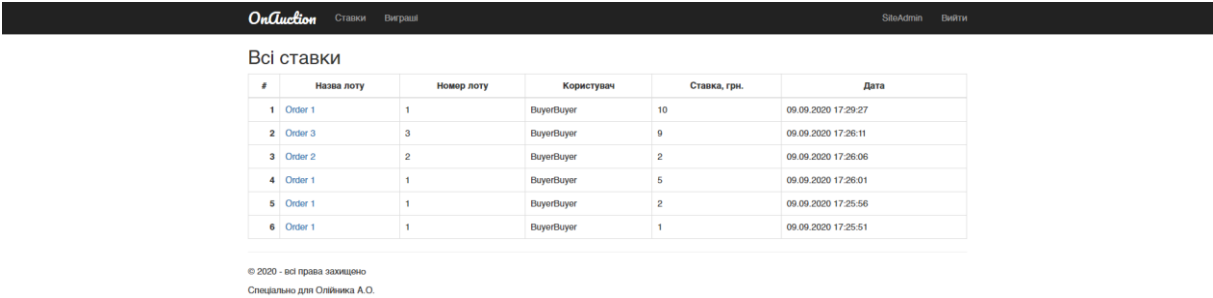


Рисунок 3.16 – Інтерфейс сторінки перегляду всіх ставок в системі

На рисунку 3.17 зображено інтерфейс сторінки перегляду всіх виграшів в системі.

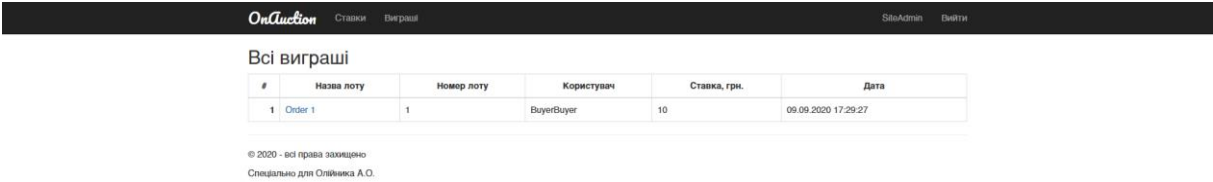


Рисунок 3.17 – Інтерфейс сторінки перегляду всіх виграшів в системі

На рисунку 3.18 зображено приклад інтерфейсу сторінки з помилками валідації даних.

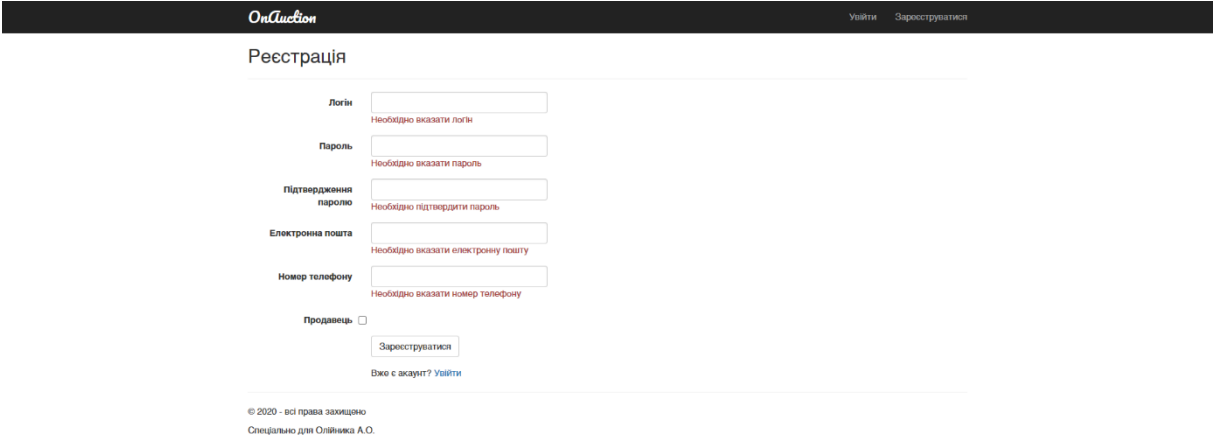


Рисунок 3.18 – Інтерфейс сторінки з помилками валідації даних

Особливу увагу було приділено тому, щоб зробити сторінки веб-застосунку адаптивними.

На рисунку 3.19 зображено результат адаптації сторінки до розмірів вікна браузера.

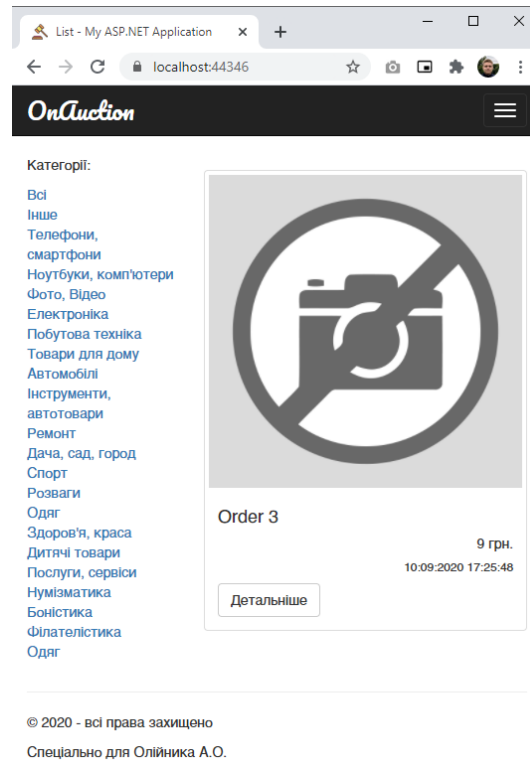


Рисунок 3.19 – Інтерфейс головної сторінки (тест на адаптивність)

4 ЗВЕРНЕННЯ ТА ВИКОРИСТАННЯ СИСТЕМИ

4.1 Призначення й умови застосування програми

Основне призначення програми – розробка системи, яка могла би використовуватися в якості інтернет аналога звичайному аукціону.

Функції програми можна поділити на чотири частини: для неавторизованого користувача, для покупця, для продавця та для адміністратора.

Функції для неавторизованого користувача включають:

- перегляд списку виставлених лотів;
- перегляд списку виставлених лотів за вибраною категорією;
- перегляд лоту;
- перегляд поточних ставок на лот;

Функції для покупця включають:

- перегляд списку виставлених лотів;
- перегляд списку виставлених лотів за вибраною категорією;
- перегляд лоту;
- можливість зробити ставку на лот;
- можливість моментального викупу лоту;
- перегляд поточних ставок на лот;
- перегляд своїх активних ставок;
- перегляд своєї історії ставок.

Функції для продавця включають:

- перегляд списку виставлених лотів;
- перегляд списку виставлених лотів за вибраною категорією;
- додавання нового лоту;
- перегляд лоту;
- перегляд поточних ставок на лот;

- перегляд своїх лотів.

Функції для адміністратора включають:

- перегляд списку виставлених лотів;
- перегляд списку виставлених лотів за вибраною категорією;
- перегляд лоту;
- перегляд поточних ставок на лот;
- моніторинг ходу торгів;
- моніторинг вдалих угод.

Основною вимогою до комп'ютера на серверній стороні є наявність доступу до мережі Internet, машини Windows Server з підтримкою IIS (Internet Information Services) та системи керування реляційною базою даних MS SQL Server. Додаткові вимоги: RAM 4Gb чи більше.

Як апаратно-технічні засоби для експлуатації програмного засобу на клієнтській стороні повинні використовуватися пристрої з доступом до мережі Internet та інтернет-браузером з оновленнями 2019 року. До складу засобів також повинні входити пристрій виводу інформації (монітор, дисплей) та маніпулятор вводу інформації (клавіатура, миша, сенсорний дисплей). Додаткові вимоги: RAM 4Gb чи більше.

4.2 Звертання до програми

Користувач взаємодіє з програмною системою за допомогою веб-сторінки в браузері, шляхом натискання на елементи керування чи вводом відповідних даних. Введені користувачем дані проходять валідацію на стороні клієнта (заповнення всіх обов'язкових полів, відповідність введеної інформації вимогам, що були висунуті, тощо). Потім дані мережею Інтернет передаються до серверу. Сервер виконує валідацію даних на своїй стороні, оброблює дані і, відповідно до запиту, звертається до бази даних. Після обробки даних сервер надсилає їх на веб-сторінку, де дані, видані на запит користувача, інтерпретуються у зрозумілий для людини вигляд.

4.3 Початкові і вихідні дані

В залежності від задачі, що виконується, початковими даними є: дані для авторизації та реєстрації, дані про ставку та лот. Також початковими даними можна вважати сам запит до сервера (будь-яка URL адресу).

Вихідними даними є відповідь на запит користувача, в залежності від запиту користувачу має відобразитися відповідне представлення (веб-сторінка).

4.4 Використання програми

На рисунку 4.1 зображено сторінку реєстрації

Рисунок 4.1 – Сторінка реєстрації

На рисунку 4.2 зображено сторінку авторизації

Рисунок 4.2 – Сторінка авторизації

На риснку 4.3 зображено сторінку з лотами

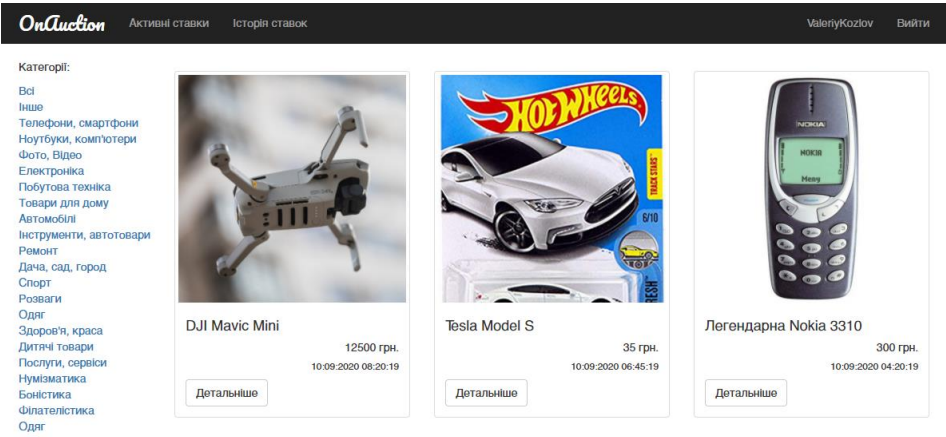


Рисунок 4.3 – Сторінка з лотами

На рисунку 4.4 зображено сторінку перегляду активного лоту з трекером часу, що залишився та полями для здійснення ставки користувачем в ролі «покупець».

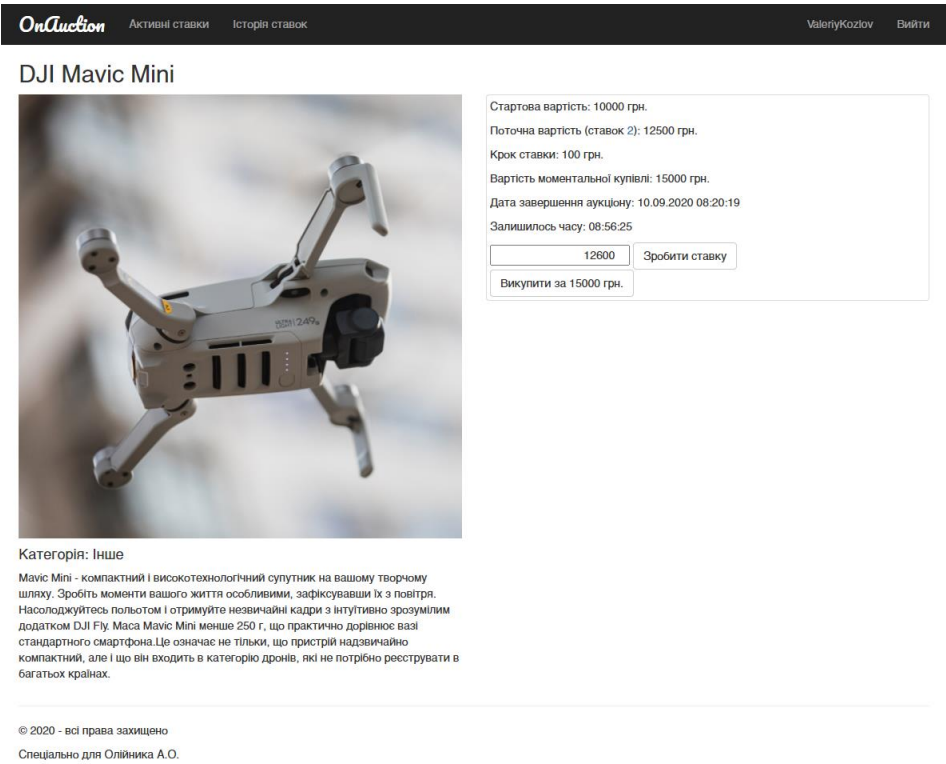



Рисунок 4.4 – Сторінка активного лоту

На рисунку 4.5 зображено сторінку перегляду лоту, аукціон на який вже завершено.

OnAuction
Активні ставки
Історія ставок
ValeriyKozlov
Вийти

[Завершено] DJI Mavic Mini



Стартова вартість: 10000 грн.
Поточна вартість (ставок 3): 15000 грн.
Крок ставки: 100 грн.
Вартість моментальної купівлі: 15000 грн.
Дата завершення аукціону: 09.09.2020 23:24:10
Залишилось часу: 00:00:00

Категорія: Інше

Mavic Mini - компактний і високотехнологічний супутник на вашому творчому шляху. Зробіть моменти вашого життя особливими, зафіксувавши їх з повітря. Насолоджуйтесь польотом і отримуйте незвичайні кадри з інтуїтивно зрозумілим додатком DJI Fly. Маса Mavic Mini менше 250 г, що практично дорівнює вазі стандартного смартфона. Це означає не тільки, що пристрій надзвичайно компактний, але і що він входить в категорію дронів, які не потрібно реєструвати в багатьох країнах.

© 2020 - всі права захищено
Спеціально для Олійника А.О.

Рисунок 4.5 – Сторінка завершеного лоту

На рисунку 4.6 зображено сторінку перегляду ставок на лот

OnAuction
Активні ставки
Історія ставок
ValeriyKozlov
Вийти

DJI Mavic Mini

#	Користувач	Ставка, грн.	Час
1	IvanIvanov	12000	09.09.2020 09:30:19
2	SergiySergiev	12500	09.09.2020 14:05:19
3	ValeriyKozlov	15000	09.09.2020 23:24:10

© 2020 - всі права захищено
Спеціально для Олійника А.О.

Рисунок 4.6 – Сторінка перегляду ставок на лот

На рисунку 4.7 зображено сторінку перегляду своїх ставок

OnAuction
Активні ставки
Історія ставок
IvanIvanov
Вийти

Активні ставки

#	Назва лоту	Моя ставка, грн.	Найвища ставка, грн.	Дата закінчення лоту
1	Lenovo Ideapad 520-15IKB	19000	19000	10.09.2020 03:20:19
2	Tesla Model S	35	50	10.09.2020 06:45:19

© 2020 - всі права захищено
Спеціально для Олійника А.О.

Рисунок 4.7 – Сторінка перегляду своїх ставок

На рисунку 4.8 зображено сторінку перегляду історії своїх ставок

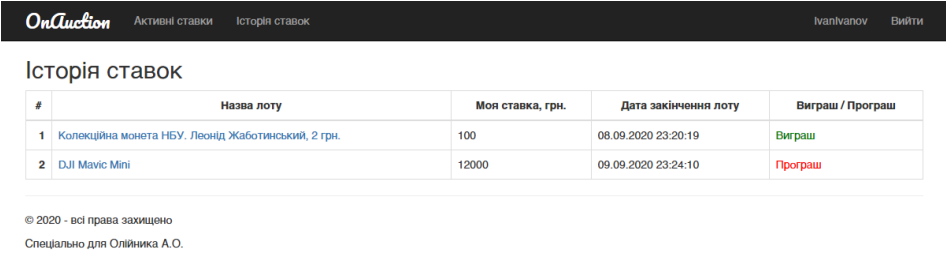


Рисунок 4.8 – Сторінка перегляду історії своїх ставок

На рисунку 4.9 зображено сторінку додавання нового лоту

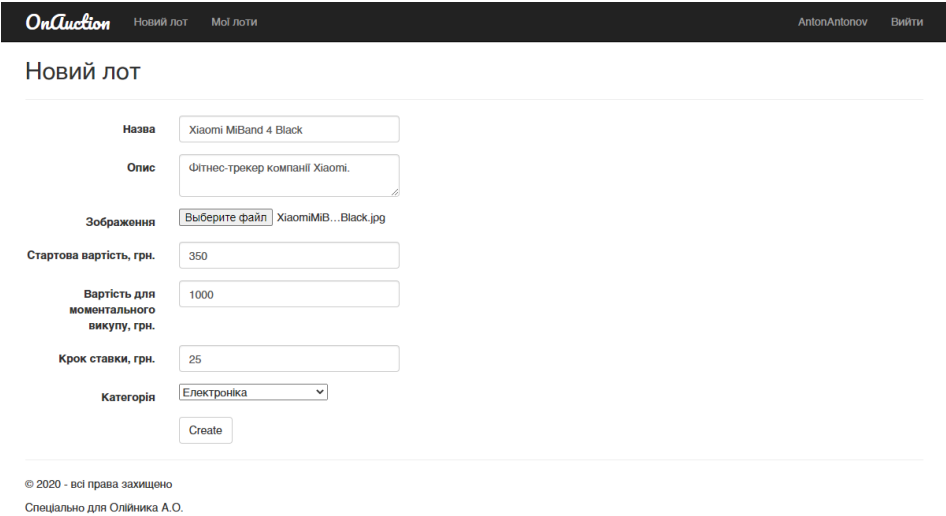


Рисунок 4.9 – Сторінка додавання нового лоту

На рисунку 4.10 зображено сторінку перегляду своїх лотів

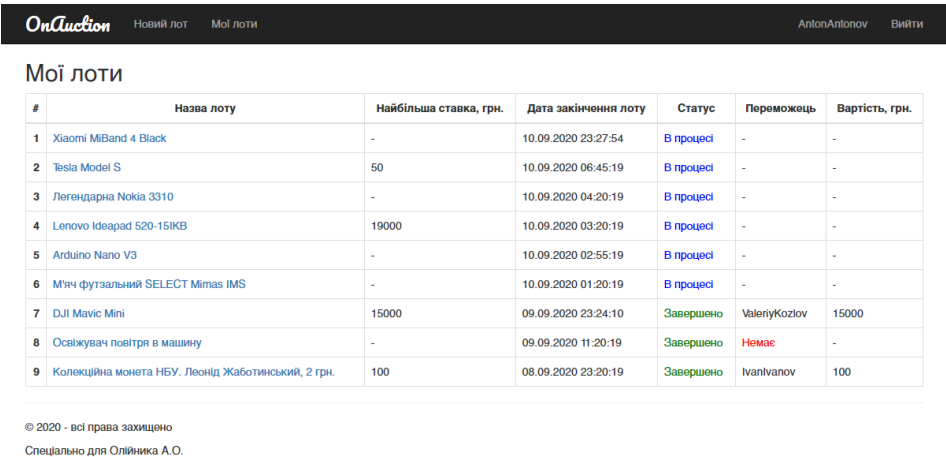


Рисунок 4.10 – Сторінка перегляду своїх лотів

На рисунку 4.11 зображено сторінку перегляду всіх ставок в системі «OnAuction».

OnAuction

СтавкиВиграші

AdminAdminВийти

Всі ставки

#	Назва лоту	Номер лоту	Користувач	Ставка, грн.	Дата
1	Tesla Model S	7	ValeriyKozlov	50	09.09.2020 23:25:05
2	DJI Mavic Mini	8	ValeriyKozlov	15000	09.09.2020 23:24:10
3	Lenovo Ideapad 520-15IKB	5	IvanIvanov	19000	09.09.2020 16:30:19
4	DJI Mavic Mini	8	SergiySergiev	12500	09.09.2020 14:05:19
5	Tesla Model S	7	IvanIvanov	35	09.09.2020 09:55:19
6	DJI Mavic Mini	8	IvanIvanov	12000	09.09.2020 09:30:19
7	Tesla Model S	7	SergiySergiev	25	09.09.2020 08:30:19
8	Tesla Model S	7	IvanIvanov	10	09.09.2020 07:45:19
9	Lenovo Ideapad 520-15IKB	5	SergiySergiev	17500	09.09.2020 04:20:19
10	Колекційна монета НБУ. Леонід Жаботинський, 2 грн.	1	IvanIvanov	100	08.09.2020 17:20:19
11	Колекційна монета НБУ. Леонід Жаботинський, 2 грн.	1	SergiySergiev	25	08.09.2020 04:55:19
12	Колекційна монета НБУ. Леонід Жаботинський, 2 грн.	1	IvanIvanov	10	08.09.2020 02:20:19

© 2020 - всі права захищено

Спеціально для Олійника А.О.

Рисунок 4.11 – Сторінка перегляду всіх ставок

На рисунку 4.12 зображено сторінку перегляду всіх вдалих угод.

OnAuction

СтавкиВиграші

AdminAdminВийти

Всі виграші

#	Назва лоту	Номер лоту	Користувач	Ставка, грн.	Дата
1	DJI Mavic Mini	8	ValeriyKozlov	15000	09.09.2020 23:24:10
2	Колекційна монета НБУ. Леонід Жаботинський, 2 грн.	1	IvanIvanov	100	08.09.2020 23:20:19

© 2020 - всі права захищено

Спеціально для Олійника А.О.

Рисунок 4.12 – Сторінка перегляду всіх вдалих угод

5 ТЕСТУВАННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ ТЕСТУВАННЯ

Порядок проведення тестування:

- перевірка на завантаження веб-застосунку;
- перевірка на коректність відображення елементів інтерфейсу та адаптивність інтерфейсу в браузері;
- перевірка авторизації користувача (коректність входу до системи, наявність доступу до функцій застосунку відповідно до своєї групи);
- перевірка авторизації адміністратора (коректність входу до системи, наявність доступу до функцій застосунку відповідно до своєї групи);
- перевірка валідації даних на стороні клієнта;
- перевірка на стресостійкість (успішне відновлення роботи системи після перезапуску сервера).

В таблиці 5.1 наведено чек-лист тестування веб-застосунку «OnAuction».

Таблиця 5.1 – Чек-лист тестування веб-застосунку «OnAuction»

Параметр	Windows 10 Google Chrome 85.0	Windows 10 Microsoft Edge 85.0	Windows 10 Opera 64.5	Windows 10 Mozilla Firefox 71.0
Завантаження веб-застосунку	Пройдено	Пройдено	Пройдено	Пройдено
Коректність відображення та адаптивність	Пройдено	Пройдено	Пройдено	Пройдено
Авторизація користувача	Пройдено	Пройдено	Пройдено	Пройдено
Авторизація адміністратора	Пройдено	Пройдено	Пройдено	Пройдено
Валідація даних	Пройдено	Пройдено	Пройдено	Пройдено
Стресостійкість	Пройдено	Пройдено	Пройдено	Пройдено

ВИСНОВКИ

Під час розробки програмної системи було проаналізовано сучасні технології створення та розробки клієнт-серверних застосунків.

Для проектування веб-застосунку було використано передові технології: .NET Framework, ASP.NET, C#, HTML, CSS, JavaScript, Razor, JQuery, Bootstrap, Entity Framework та MVC Framework.

В якості середовища розробки було обрано IDE Microsoft Visual Studio, адже воно адаптоване під обрану мову програмування, має велику кількість плагінів та надає достатній функціонал для розробки. Також перевагою цієї IDE є те, що версії Visual Studio є абсолютно безкоштовними для учнів, студентів та розробників програм з відкритим програмним кодом.

Для розробки веб-застосунку було обрано відкритий C# .NET фреймворк ASP.NET MVC Framework, який реалізує шаблон MVC і має всі необхідні засоби для якісної та швидкої розробки проекту.

При розробці програмної системи було проаналізовано і застосовано на практиці методи валідації даних, як на стороні клієнта, так і на стороні сервера, методи авторизації та взаємодії з базою даних.

Створений продукт успішно пройшов тестування, відповідає меті розробки та всім вимогам, що були пред'явлені до програми в технічному завданні. Це свідчить про злагоджену роботу усіх учасників розробки, правильну організацію процесу розробки, успішно виконаний аналіз вимог.

Архітектура розробленого проекту дозволяє модифікувати програму за потреби в майбутньому, наприклад – для підтримки інших типів аукціонів, різних грошових одиниць, тощо.

ДОДАТОК А
ТЕХНІЧНЕ ЗАВДАННЯ

A.1 Найменування

Найменування веб-застосунку – інтернет-аукціон (OnAuction).

A.2 Область застосування веб-застосунку

Веб-застосунок призначений для кінцевого користувача в якості програми для проведення онлайн аукціонів.

A.3 Підстава для розробки

Підставою для розробки є завдання на лабораторні роботи з дисципліни «Архітектура та проектування програмного забезпечення» на тему «Варіант 9. Інтернет-аукціон», затверджене науковим керівником.

A.4 Мета розробки та її призначення

Метою створення програмної системи є розробка веб-застосунку для інтернет-аукціону з мінімалістичним й інтуїтивно зрозумілим інтерфейсом, що дозволяє користувачу брати участь в онлайн аукціонах.

A.5 Основні вимоги до програмного продукту

Система повинна мати клієнт-серверну архітектуру.

На серверній стороні має відбуватися робота й обробка даних, а на стороні клієнта відображатися інтерфейс користувача для взаємодії з програмною системою.

Система повинна передбачати багаторівневий режим доступу до функціоналу:

- неавторизований користувач;
- покупець;
- продавець;
- адміністратор.

На рівні доступу «Неавторизований користувач» має бути реалізовано такі функції:

- можливість перегляду списку виставлених лотів;
- перегляд списку виставлених лотів за вибраною категорією;
- перегляд лоту;
- перегляд поточних ставок на лот;

На рівні доступу «Покупець» має бути реалізовано такі функції:

- можливість перегляду списку виставлених лотів;
- перегляд списку виставлених лотів за вибраною категорією;
- перегляд лоту;
- можливість зробити ставку на лот;
- можливість моментального викупу лоту;
- перегляд поточних ставок на лот;
- перегляд своїх активних ставок;
- перегляд своєї історії ставок.

На рівні доступу «Продавець» має бути реалізовано такі функції:

- можливість перегляду списку виставлених лотів;
- перегляд списку виставлених лотів за вибраною категорією;
- додавання нового лоту;
- перегляд лоту;
- перегляд поточних ставок на лот;
- перегляд своїх лотів.

На рівні доступу «Адміністратор» має бути реалізовано такі функції:

- можливість перегляду списку виставлених лотів;

- перегляд списку виставлених лотів за вибраною категорією;
- перегляд лоту;
- перегляд поточних ставок на лот;
- моніторинг ходу торгів;
- моніторинг вдалих угод.

При розробці графічного інтерфейсу за мету має бути визначено створення функціонального, проте простого, неперенавантаженого зайвою інформацією, інтерфейсу.

В процесі розробки графічного користувацького інтерфейсу перевага має бути віддана спокійним, однотонним відтінкам, які привертають увагу користувача.

За допомогою графічного інтерфейсу користувач має отримувати підказки стосовно правильності вводу даних та коректності своїх дій. Інтерфейс має коректно відображатися на різних пристроях, що відповідають мінімальним технічним вимогам, та в різних браузерах.

Експлуатаційними вимогами є вимоги до безвідмовної роботи серверної частини та відповідного оновлення користувацького інтерфейсу.

А.6 Вимоги до видів забезпечення і компонентів

Основною вимогою до комп'ютера на серверній стороні є наявність доступу до мережі Internet, машини Windows Server з підтримкою IIS (Internet Information Services) та системи керування реляційною базою даних MS SQL Server. Додаткові вимоги: RAM 4Gb чи більше.

Як апаратно-технічні засоби для експлуатації програмного засобу на клієнтській стороні повинні використовуватися пристрої з доступом до мережі Internet та інтернет-браузером з оновленнями 2019 року.

До складу засобів також повинні входити пристрій виводу інформації (монітор, дисплей) та маніпулятор вводу інформації (клавіатура, миша, сенсорний дисплей). Додаткові вимоги: RAM 4Gb чи більше.

A.7 Обмеження на установку і виконання програми

Програмну систему, що розробляються, можна встановити на будь-якому пристрої, що відповідає вимогам до видів забезпечення (пункт A.5).

A.8 Вимоги до програмної документації

Програмне забезпечення має складатися з пояснювальної записки, технічного завдання, опису програми та текстів програми.

A.9 Умови експлуатації

Клієнтська частина веб-застосунку призначена для роботи на пристроях, за допомогою одного з браузерів: Google Chrome 85.0 (чи новіше), Microsoft Edge 85.0 (чи новіше), Opera 65.0 (чи новіше), Mozilla Firefox 71.0 (чи новіше).

A.10 Вимоги до маркування й пакування

Програмна система може поставлятися на диску чи на флеш накопичувачі. На упакуванні має бути зазначена назва програмного продукту – «OnAuction» стилізована шрифтом «Pacifico Regular».

A.11 Вимоги до транспортування та збереження

Вимоги до транспортування та збереження аналогічні тим, що висуваються до накопичувачів на яких зберігається програмний продукт.

ДОДАТОК В
ТЕКСТИ ПРОГРАМИ

BundleConfig.cs

```
using System.Web;
using System.Web.Optimization;

namespace OnAuction
{
    public class BundleConfig
    {
        // Дополнительные сведения об объединении см. на странице
        https://go.microsoft.com/fwlink/?LinkId=301862
        public static void RegisterBundles(BundleCollection bundles)
        {
            bundles.Add(new ScriptBundle("~/bundles/jquery").Include(
                "~/Scripts/jquery-{version}.js"));

            bundles.Add(new ScriptBundle("~/bundles/jqueryval").Include(
                "~/Scripts/jquery.validate*"));

            // Используйте версию Modernizr для разработчиков, чтобы учиться работать.
            // Когда вы будете готовы перейти к работе,
            // готово к выпуску, используйте средство сборки по адресу
            https://modernizr.com, чтобы выбрать только необходимые тесты.
            bundles.Add(new ScriptBundle("~/bundles/modernizr").Include(
                "~/Scripts/modernizr-*"));

            bundles.Add(new ScriptBundle("~/bundles/bootstrap").Include(
                "~/Scripts/bootstrap.js"));

            bundles.Add(new StyleBundle("~/Content/css").Include(
                "~/Content/bootstrap.css",
                "~/Content/site.css"));
        }
    }
}
```

FilterConfig.cs

```
using System.Web;
using System.Web.Mvc;

namespace OnAuction
{
    public class FilterConfig
    {
        public static void RegisterGlobalFilters(GlobalFilterCollection filters)
        {
            filters.Add(new HandleErrorAttribute());
        }
    }
}
```

RouteConfig.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using System.Web.Routing;

namespace OnAuction
{
    public class RouteConfig
    {
        public static void RegisterRoutes(RouteCollection routes)
        {
            routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

            routes.MapRoute(
                name: "Default",
                url: "{controller}/{action}/{id}",
                defaults: new { controller = "Lot", action = "List", id =
UrlParameter.Optional }
            );
        }
    }
}
```

FileSizeAttribute.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Runtime.InteropServices;
using System.Web;

namespace OnAuction.Attributes
{
    public class FileSizeAttribute : ValidationAttribute
    {
        private readonly int _maxSize;

        public FileSizeAttribute(int maxSize)
        {
            _maxSize = maxSize;
        }

        public override bool IsValid(object value)
        {
            var file = value as HttpPostedFileBase;

            if (file == null)
            {
                return true;
            }
        }
    }
}
```

```

        if (file.ContentLength > _maxSize * 1024 * 1024)
        {
            return false;
        }
        else
        {
            return true;
        }
    }
}
}

```

FileTypeAttribute.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;

namespace OnAuction.Attributes
{
    public class FileTypeAttribute : ValidationAttribute
    {
        private readonly List<string> _types;

        public FileTypeAttribute(string types)
        {
            _types = types.Split(',').ToList();
        }

        public override bool IsValid(object value)
        {
            var file = value as HttpPostedFileBase;

            if (file == null)
            {
                return true;
            }

            foreach (var type in _types)
            {
                if (file.FileName.EndsWith(type))
                {
                    return true;
                }
            }

            return false;
        }
    }
}

```


Site.css

```

body {
    padding-top: 50px;
    padding-bottom: 20px;
}

.body-content {
    padding-left: 15px;
    padding-right: 15px;
}

.dl-horizontal dt {
    white-space: normal;
}

input,
select,
textarea {
    max-width: 280px;
}

@font-face {
    font-family: "Pacifico";
    src: url("../fonts/Pacifico-Regular.ttf");
}

.navbar-brand {
    font-family: Pacifico;
    font-size: 24px;
    color: white !important;
}

.row.equal {
    display: flex;
    display: -webkit-flex;
    flex-wrap: wrap;
}

.thumbnail {
    height: 100%;
    margin-bottom: 0px;
}

.p-15 {
    padding: 15px;
}

.wrap-text {
    white-space: nowrap;
    overflow: hidden;
    text-overflow: ellipsis;
}

.w-100-right {
    width: 100%;
    text-align: right;
}

```

```

.crop {
    width: 100%;
}

.crop-wrapper {
    overflow: hidden;
    position: relative;
    width: 100%;
    padding-bottom: 100%;
}

.crop-content {
    position: absolute;
    width: 100%;
    height: 100%;
    background-color: #dbdbdb;
    display: flex;
    justify-content: center;
    align-items: center;
}

```

AccountController.cs

```

using OnAuction.Filters;
using OnAuction.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using System.Web.Security;

namespace OnAuction.Controllers
{
    public class AccountController : Controller
    {
        [NotAuthorize]
        public ActionResult Register()
        {
            return View();
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Register(RegisterModel model)
        {
            if (ModelState.IsValid)
            {
                User user = null;
                using (DataContext db = new DataContext())
                {
                    user = db.Users.FirstOrDefault(x => x.Username == model.Username);
                }
                if (user == null)
                {
                    using (DataContext db = new DataContext())
                    {

```

```

        db.Users.Add(new User {Username = model.Username, Password =
model.Password, Email = model.Email, PhoneNumber = model.PhoneNumber, RoleId =
model.IsSeller ? 2 : 1});
        db.SaveChanges();

        user = db.Users.FirstOrDefault(x => x.Username == model.Username &&
x.Password == model.Password);
    }

    if (user != null)
    {
        return RedirectToAction("Login");
    }
    else
    {
        ModelState.AddModelError("", "Користувач з таким логіном вже існує");
    }
}
return View(model);
}

public ActionResult Logout()
{
    FormsAuthentication.SignOut();
    return RedirectToAction("List", "Lot");
}

[NotAuthorize]
public ActionResult Login()
{
    return View();
}

[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Login(LoginModel model)
{
    if (ModelState.IsValid)
    {
        User user = null;
        using (DataContext db = new DataContext())
        {
            user = db.Users.FirstOrDefault(x => x.Username == model.Username &&
x.Password == model.Password);
        }
        if (user != null)
        {
            FormsAuthentication.SetAuthCookie(model.Username, true);
            return RedirectToAction("List", "Lot");
        }
        else
        {
            ModelState.AddModelError("", "Не правильний логін чи пароль");
        }
    }
    return View(model);
}
}
}

```

AdminController.cs

```

using OnAuction.Filters;
using OnAuction.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using System.Data.Entity;
using OnAuction.LifeCycle;

namespace OnAuction.Controllers
{
    public class AdminController : Controller
    {
        [User(Roles = "admin")]
        public ActionResult Bets()
        {
            LifecycleManager.CheckLotsForFinish();

            List<Bet> Bets;
            using(DataContext db = new DataContext())
            {
                Bets = db.Bets.Include(x => x.Lot).Include(x =>
x.User).OrderByDescending(x => x.Time).ToList();
            }

            return View(Bets);
        }

        [User(Roles = "admin")]
        public ActionResult Wins()
        {
            LifecycleManager.CheckLotsForFinish();

            List<Win> Wins;
            using (DataContext db = new DataContext())
            {
                Wins = db.Wins.Include(x => x.Lot).Include(x =>
x.User).OrderByDescending(x => x.Time).ToList();
            }

            return View(Wins);
        }
    }
}

```

LotController.cs

```

using OnAuction.Filters;
using OnAuction.Models;
using System;
using System.Collections.Generic;
using System.IO;

```

```

using System.Linq;
using System.Web;
using System.Web.Mvc;
using System.Data.Entity;
using OnAuction.LifeCycle;

namespace OnAuction.Controllers
{
    public class LotController : Controller
    {
        DataContext db = new DataContext();

        [User(Roles = "seller")]
        public ActionResult Create()
        {
            SelectList categories = new SelectList(db.Categories, "Id", "Name");
            ViewBag.Categories = categories;
            return View();
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Create(LotCreateModel model)
        {
            if(ModelState.IsValid)
            {
                DateTime Date = DateTime.Now;
                string imagePath = null;

                if(model.Image != null)
                {
                    imagePath = "~/Images/" + Date.ToString("yyyyMMdd_HHmssfff") +
Path.GetExtension(model.Image.FileName);
                    model.Image.SaveAs(Server.MapPath(imagePath));
                }

                if(model.InstantPrice < model.StartPrice)
                {
                    model.InstantPrice = model.StartPrice;
                }

                using (DataContext db = new DataContext())
                {
                    int userId = db.Users.FirstOrDefault(x => x.Username ==
User.Identity.Name).Id;

                    db.Lots.Add(new Lot {
                        Title = model.Title,
                        Description = model.Description,
                        ImagePath = imagePath,
                        StartPrice = model.StartPrice,
                        InstantPrice = model.InstantPrice,
                        BetStep = model.BetStep,
                        StartTime = Date,
                        FinishTime = Date.AddDays(1),
                        IsFinish = false,
                        UserId = userId,
                        CategoryId = model.CategoryId });

                    db.SaveChanges();
                }
            }
        }
    }
}

```

```

        return RedirectToAction("List", "Lot");
    }
    SelectList categories = new SelectList(db.Categories, "Id", "Name");
    ViewBag.Categories = categories;
    return View(model);
}

public ActionResult List(int? category)
{
    LifecycleManager.CheckLotsForFinish();

    var Lots = db.Lots.AsQueryable();

    if (category != null)
    {
        Lots = Lots.Where(x => x.CategoryId == category);
    }

    var BestBets = db.Bets
        .GroupBy(bet => bet.LotId)
        .Select(grp => grp.OrderByDescending(bet => bet.Price).FirstOrDefault());

    var LotAndBestBet = Lots
        .Where(lot => !lot.IsFinish)
        .GroupJoin(BestBets, lot => lot.Id, bet => bet.LotId, (Lot, Bets) => new {
Lot, Bets })
        .SelectMany(grp => grp.Bets.DefaultIfEmpty(), (grp, BestBet) => new
LotAndBetVM { Lot = grp.Lot, Bet = BestBet })
        .OrderByDescending(l => l.Lot.FinishTime);

    ViewBag.LotAndBestBet = LotAndBestBet;
    ViewBag.Categories = db.Categories;

    return View();
}

public ActionResult Info(int LotId)
{
    LifecycleManager.CheckLotsForFinish();

    Lot Lot = db.Lots
        .Include(lot => lot.Category)
        .FirstOrDefault(lot => lot.Id == LotId);

    if (Lot == null)
    {
        return HttpNotFound();
    }

    Bet BestBet = db.Bets
        .Where(bet => bet.LotId == LotId)
        .OrderByDescending(bet => bet.Price)
        .FirstOrDefault();

    ViewBag.BetsCount = db.Bets.Where(bet => bet.LotId == LotId).Count();

    return View(new LotAndBetVM { Lot = Lot, Bet = BestBet});
}

[HttpPost]
[User(Roles = "buyer")]
[ValidateAntiForgeryToken]

```

```

public void MakeBet(int LotId, int Price)
{
    LifecycleManager.CheckLotsForFinish();

    DateTime CurrentDate = DateTime.Now;

    Lot Lot = db.Lots.FirstOrDefault(lot => lot.Id == LotId && !lot.IsFinish &&
lot.FinishTime > CurrentDate);

    if(Lot == null)
    {
        Response.Redirect("Info?LotId=" + LotId.ToString(), false);
        return;
    }

    if(!((Price - Lot.StartPrice) >= 0 && (Price - Lot.StartPrice) % Lot.BetStep
== 0))
    {
        string errorMessage = "Вартість ставки не відповідає вимогам";
        Response.Write("<script language='javascript'>window.alert('" +
errorMessage + "');window.location.href='Info?LotId=" + LotId.ToString() +
"'</script>");
        return;
    }

    Bet BestBet = db.Bets
        .Where(bet => bet.LotId == LotId)
        .OrderByDescending(bet => bet.Price)
        .FirstOrDefault();

    if(BestBet != null)
    {
        if(Price <= BestBet.Price)
        {
            string errorMessage = "Вартість поточної ставки має бути більше ніж
попередня";
            Response.Write("<script language='javascript'>window.alert('" +
errorMessage + "');window.location.href='Info?LotId=" + LotId.ToString() +
"'</script>");
            return;
        }
    }

    if(Price >= Lot.InstantPrice)
    {
        InstantBet(LotId, CurrentDate);
        return;
    }

    using (DataContext db = new DataContext())
    {
        int userId = db.Users.FirstOrDefault(x => x.Username ==
User.Identity.Name).Id;

        db.Bets.Add(new Bet {
            Price = Price,
            Time = CurrentDate,
            LotId = LotId,
            UserId = userId });

        db.SaveChanges();
    }
}

```

```

        Response.Redirect("Info?LotId=" + LotId.ToString(), false);
        return;
    }

    [HttpPost]
    [User(Roles = "buyer")]
    [ValidateAntiForgeryToken]
    public void MakeInstantBet(int LotId)
    {
        LifecycleManager.CheckLotsForFinish();

        InstantBet(LotId, DateTime.Now);
    }

    private void InstantBet(int LotId, DateTime CurrentDate)
    {
        Lot Lot = db.Lots.FirstOrDefault(lot => lot.Id == LotId && !lot.IsFinish &&
lot.FinishTime > CurrentDate);

        if (Lot == null)
        {
            Response.Redirect("Info?LotId=" + LotId.ToString(), false);
            return;
        }

        Lot.IsFinish = true;
        Lot.FinishTime = CurrentDate;

        using (DataContext db = new DataContext())
        {
            int userId = db.Users.FirstOrDefault(x => x.Username ==
User.Identity.Name).Id;

            db.Bets.Add(new Bet
            {
                Price = Lot.InstantPrice,
                Time = CurrentDate,
                LotId = LotId,
                UserId = userId
            });

            db.Wins.Add(new Win{
                Price = Lot.InstantPrice,
                Time = CurrentDate,
                LotId = LotId,
                UserId = userId});

            db.Lots.Attach(Lot);
            db.Entry(Lot).Property(lot => lot.IsFinish).IsModified = true;
            db.Entry(Lot).Property(lot => lot.FinishTime).IsModified = true;

            db.SaveChanges();
        }

        Response.Redirect("Info?LotId=" + LotId.ToString(), false);
        return;
    }

    public ActionResult Bets(int LotId)
    {
        LifecycleManager.CheckLotsForFinish();
    }

```



```

        Lot Lot = db.Lots.FirstOrDefault(lot => lot.Id == LotId);

        List<Bet> Bets = db.Bets.Where(bet => bet.LotId == LotId).Include(bet =>
bet.User).OrderBy(bet => bet.Price).ToList();

        if(Lot == null)
        {
            return HttpNotFound();
        }

        ViewBag.Lot = Lot;

        return View(Bets);
    }

    [User(Roles="buyer")]
    public ActionResult MyBetsActive()
    {
        LifecycleManager.CheckLotsForFinish();

        int userId = db.Users.FirstOrDefault(x => x.Username ==
User.Identity.Name).Id;

        var MyBets = db.Bets
            .Include(bet => bet.Lot)
            .Where(bet => bet.UserId == userId && !bet.Lot.IsFinish)
            .GroupBy(bet => bet.LotId)
            .Select(grp => grp.OrderByDescending(bet =>
bet.Price).FirstOrDefault());

        var BestBets = db.Bets
            .Where(bet => !bet.Lot.IsFinish)
            .GroupBy(bet => bet.LotId)
            .Select(grp => grp.OrderByDescending(bet =>
bet.Price).FirstOrDefault());

        List<LotAndBetAndBestBetVM> MyBetAndBestBet = MyBets
            .Join(BestBets, bet => bet.LotId, bet => bet.LotId, (bet,
bestbet) => new { Bet = bet, BestBet = bestbet })
            .OrderByDescending(vm => vm.Bet.Lot.FinishTime)
            .Join(db.Lots, bets => bets.Bet.LotId, lot => lot.Id, (bets,
lot) => new LotAndBetAndBestBetVM { Lot = lot, Bet = bets.Bet, BestBet =
bets.BestBet})
            .ToList();

        return View(MyBetAndBestBet);
    }

    [User(Roles = "buyer")]
    public ActionResult MyBetsHistory()
    {
        LifecycleManager.CheckLotsForFinish();

        int userId = db.Users.FirstOrDefault(x => x.Username ==
User.Identity.Name).Id;

        var MyBets = db.Bets
            .Include(bet => bet.Lot)
            .Where(bet => bet.UserId == userId && bet.Lot.IsFinish)
            .GroupBy(bet => bet.LotId)

```

```

        .Select(grp => grp.OrderByDescending(bet =>
bet.Price).FirstOrDefault());

        var WinBets = db.Wins.Where(win => win.Lot.IsFinish);

        List<LotAndBetAndWinVM> MyBetAndWin = MyBets
        .Join(WinBets, bet => bet.LotId, win => win.LotId, (bet, win) =>
new { Bet = bet, Win = win })
        .OrderByDescending(vm => vm.Bet.Lot.FinishTime)
        .Join(db.Lots, bets => bets.Bet.LotId, lot => lot.Id, (bets,
lot) => new LotAndBetAndWinVM { Lot = lot, Bet = bets.Bet, Win = bets.Win })
        .ToList();

        return View(MyBetAndWin);
    }

    [User(Roles = "seller")]
    public ActionResult MyLots()
    {
        LifecycleManager.CheckLotsForFinish();

        LifecycleManager.CheckLotsForFinish();

        int userId = db.Users.FirstOrDefault(x => x.Username ==
User.Identity.Name).Id;

        var MyLots = db.Lots.Where(lot => lot.UserId == userId);

        var BestBets = db.Bets
        .Include(bet => bet.Lot)
        .Where(bet => bet.Lot.UserId == userId)
        .GroupBy(bet => bet.LotId)
        .Select(grp => grp.OrderByDescending(bet =>
bet.Price).FirstOrDefault());

        var Wins = db.Wins
        .Include(win => win.User)
        .Include(win => win.Lot)
        .Where(win => win.Lot.UserId == userId);

        List<LotAndBetAndWinAndUserVM> MyLotsAndWins = MyLots
        .GroupJoin(Wins, lot => lot.Id, win => win.LotId, (lot, win) => new { lot,
win })
        .SelectMany(grp => grp.win.DefaultIfEmpty(), (grp, win) => new { grp.lot,
win, win.User })
        .GroupJoin(BestBets, grp => grp.lot.Id, bet => bet.LotId, (grp, bet) =>
new { grp.lot, bet, grp.win })
        .SelectMany(grp => grp.bet.DefaultIfEmpty(), (grp, bet) => new
LotAndBetAndWinAndUserVM { Lot = grp.lot, Bet = bet, Win = grp.win, Winner =
grp.win.User })
        .OrderByDescending(grp => grp.Lot.FinishTime)
        .ToList();

        return View(MyLotsAndWins);
    }
}
}

```

NotAuthorizeAttribute.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace OnAuction.Filters
{
    public class NotAuthorizeAttribute : AuthorizeAttribute
    {
        public override void OnAuthorization(AuthorizationContext filterContext)
        {
            bool auth = filterContext.HttpContext.User.Identity.IsAuthenticated;

            if (auth)
            {
                filterContext.Result = new HttpNotFoundResult();
            }
        }
    }
}
```

UserAttribute.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace OnAuction.Filters
{
    public class UserAttribute : AuthorizeAttribute
    {
        public override void OnAuthorization(AuthorizationContext filterContext)
        {
            bool auth = false;

            foreach (var role in Roles.Split(',').ToList())
            {
                if (filterContext.HttpContext.User.IsInRole(role))
                {
                    auth = true;
                }
            }

            if (!auth)
            {
                filterContext.Result = new HttpNotFoundResult();
            }
        }
    }
}
```

LifeCycleManager.cs

```

using OnAuction.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace OnAuction.LifeCycle
{
    public static class LifeCycleManager
    {
        public static void CheckLotsForFinish()
        {
            using (DataContext db = new DataContext())
            {
                List<Lot> Lots = db.Lots.Where(lot => !lot.IsFinish && lot.FinishTime <=
DateTime.Now).ToList();

                foreach (Lot Lot in Lots)
                {
                    Bet BestBet = db.Bets
                        .Where(bet => bet.LotId == Lot.Id)
                        .OrderByDescending(bet => bet.Price)
                        .FirstOrDefault();

                    if (BestBet != null)
                    {
                        db.Wins.Add(new Win
                        {
                            Price = BestBet.Price,
                            Time = Lot.FinishTime,
                            LotId = Lot.Id,
                            UserId = BestBet.UserId
                        });
                    }

                    Lot.IsFinish = true;

                    db.Lots.Attach(Lot);
                    db.Entry(Lot).Property(lot => lot.IsFinish).IsModified = true;

                    db.SaveChanges();
                }
            }
        }
    }
}

```

Bet.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;

```

```

using System.Web;

namespace OnAuction.Models
{
    public class Bet
    {
        public int Id { get; set; }
        [Required]
        public int Price { get; set; }
        [Required]
        public DateTime Time { get; set; }

        [Required]
        public int LotId { get; set; }
        public Lot Lot { get; set; }
        [Required]
        public int UserId { get; set; }
        public User User { get; set; }
    }
}

```

Category.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class Category
    {
        public int Id { get; set; }
        [Required]
        public string Name { get; set; }
    }
}

```

DataContext.cs

```

using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class DataContext : DbContext
    {
        public DbSet<User> Users { get; set; }
        public DbSet<Role> Roles { get; set; }
    }
}

```

```

        public DbSet<Lot> Lots { get; set; }
        public DbSet<Category> Categories { get; set; }
        public DbSet<Bet> Bets { get; set; }
        public DbSet<Win> Wins { get; set; }
    }
}

```

DataDBInitializer.cs

```

using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class DataDBInitializer : DropCreateDatabaseAlways<DataContext>
    {
        protected override void Seed(DataContext db)
        {
            DateTime CurrentTime = DateTime.Now;

            db.Roles.Add(new Role { Id = 1, Name = "buyer" });
            db.Roles.Add(new Role { Id = 2, Name = "seller" });
            db.Roles.Add(new Role { Id = 3, Name = "admin" });

            db.Users.Add(new User
            {
                Id = 1,
                Username = "AdminAdmin",
                Password = "AdminAdmin",
                Email = "AdminAdmin@OnAuction.com",
                PhoneNumber = "380000000000",
                RoleId = 3
            });

            db.Users.Add(new User
            {
                Id = 2,
                Username = "AntonAntonov",
                Password = "AntonAntonov",
                Email = "SergievSergiy@OnAuction.com",
                PhoneNumber = "380000000001",
                RoleId = 2
            });

            db.Users.Add(new User
            {
                Id = 3,
                Username = "IvanIvanov",
                Password = "IvanIvanov",
                Email = "IvanIvanov@OnAuction.com",
                PhoneNumber = "380000000002",
                RoleId = 1
            });

            db.Users.Add(new User

```

```

{
    Id = 4,
    Username = "SergiySergiev",
    Password = "SergiySergiev",
    Email = "SergiySergiev@OnAuction.com",
    PhoneNumber = "380000000003",
    RoleId = 1
});

db.Categories.Add(new Category { Name = "Інше" });
db.Categories.Add(new Category { Name = "Телефони, смартфони" });
db.Categories.Add(new Category { Name = "Ноутбуки, комп'ютери" });
db.Categories.Add(new Category { Name = "Фото, Відео" });
db.Categories.Add(new Category { Name = "Електроніка" });
db.Categories.Add(new Category { Name = "Побутова техніка" });
db.Categories.Add(new Category { Name = "Товари для дому" });
db.Categories.Add(new Category { Name = "Автомобілі" });
db.Categories.Add(new Category { Name = "Інструменти, автотовари" });
db.Categories.Add(new Category { Name = "Ремонт" });
db.Categories.Add(new Category { Name = "Дача, сад, город" });
db.Categories.Add(new Category { Name = "Спорт" });
db.Categories.Add(new Category { Name = "Розваги" });
db.Categories.Add(new Category { Name = "Одяг" });
db.Categories.Add(new Category { Name = "Здоров'я, краса" });
db.Categories.Add(new Category { Name = "Дитячі товари" });
db.Categories.Add(new Category { Name = "Послуги, сервіси" });
db.Categories.Add(new Category { Name = "Нумізматика" });
db.Categories.Add(new Category { Name = "Боністика" });
db.Categories.Add(new Category { Name = "Філателістика" });
db.Categories.Add(new Category { Name = "Одяг" });

db.Lots.Add(new Lot
{
    Id = 1,
    Title = "Колекційна монета НБУ. Леонід Жаботинський, 2 грн.",
    Description = "Посвящена легендарному українському спортсмену важкоатлету
Леоніду жаботинському.\n" +
        "Номінал 2 грн.\n" +
        "Випущена 23.01.2018\n",
    ImagePath = "~/Images/order1.jpg",
    StartPrice = 10,
    InstantPrice = 250,
    BetStep = 5,
    StartTime = CurrentTime.AddDays(-2),
    FinishTime = CurrentTime.AddDays(-1),
    IsFinish = false,
    UserId = 2,
    CategoryId = 18,
});

db.Lots.Add(new Lot
{
    Id = 2,
    Title = "Освіжувач повітря в машину",
    Description = "Освіжувач повітря в машину з ароматом лимону та лайму.",
    ImagePath = null,
    StartPrice = 25,
    InstantPrice = 100,
    BetStep = 1,
    StartTime = CurrentTime.AddDays(-2).AddHours(12),
    FinishTime = CurrentTime.AddDays(-1).AddHours(12),
    IsFinish = false,

```

```

        UserId = 2,
        CategoryId = 9,
    });

    db.Lots.Add(new Lot
    {
        Id = 3,
        Title = "М'яч футзальний SELECT Mimas IMS",
        Description = "Футзальний м'яч клубних матчів і інтенсивних тренувань.\n"
+
        "Має сертифікат IMS(International Match Standart).\n" +
        "Використовується для проведення клубних футзальних матчів і інтенсивних
тренувань.\n" +
        "Призначений для гри на паркеті, і штучному покритті.",
        ImagePath = "~/Images/order3.jpg",
        StartPrice = 500,
        InstantPrice = 1500,
        BetStep = 50,
        StartTime = CurrentTime.AddDays(-1).AddHours(2),
        FinishTime = CurrentTime.AddHours(2),
        IsFinish = false,
        UserId = 2,
        CategoryId = 12,
    });

    db.Lots.Add(new Lot
    {
        Id = 4,
        Title = "Arduino Nano V3",
        Description = "Arduino Nano V3.0 - це маленьке, готове до використання і
добре працює з макетної платі пристрій, розроблене на мікроконтролері ATmega328.",
        ImagePath = "~/Images/order4.jpg",
        StartPrice = 10,
        InstantPrice = 250,
        BetStep = 5,
        StartTime = CurrentTime.AddDays(-1).AddHours(3).AddMinutes(35),
        FinishTime = CurrentTime.AddHours(3).AddMinutes(35),
        IsFinish = false,
        UserId = 2,
        CategoryId = 5,
    });

    db.Lots.Add(new Lot
    {
        Id = 5,
        Title = "Lenovo Ideapad 520-15IKB",
        Description = "Color: Black\n" +
        "SSD + HDD: 128 + 1024 Gb\n" +
        "CPU: Intel Core I5-7200U\n" +
        "RAM: 8 Gb\n" +
        "GPU: Nvidia GeForce 940MX, 2 Gb",
        ImagePath = "~/Images/order5.jpg",
        StartPrice = 15000,
        InstantPrice = 25000,
        BetStep = 100,
        StartTime = CurrentTime.AddDays(-1).AddHours(4),
        FinishTime = CurrentTime.AddHours(4),
        IsFinish = false,
        UserId = 2,
        CategoryId = 3,
    });

```



```

db.Lots.Add(new Lot
{
    Id = 6,
    Title = "Легендарна Nokia 3310",
    Description = "Та сама легендарна, оригінальна Nokia 3310",
    ImagePath = "~/Images/order6.jpg",
    StartPrice = 300,
    InstantPrice = 1000,
    BetStep = 10,
    StartTime = CurrentTime.AddDays(-1).AddHours(5),
    FinishTime = CurrentTime.AddHours(5),
    IsFinish = false,
    UserId = 2,
    CategoryId = 2,
});

db.Lots.Add(new Lot
{
    Id = 7,
    Title = "Tesla Model S",
    Description = "Базова машинка HotWheels\n"+
        "Модель: Tesla Model S",
    ImagePath = "~/Images/order7.jpg",
    StartPrice = 10,
    InstantPrice = 100,
    BetStep = 5,
    StartTime = CurrentTime.AddDays(-1).AddHours(7).AddMinutes(25),
    FinishTime = CurrentTime.AddHours(7).AddMinutes(25),
    IsFinish = false,
    UserId = 2,
    CategoryId = 1,
});

db.Lots.Add(new Lot
{
    Id = 8,
    Title = "DJI Mavic Mini",
    Description = "Mavic Mini - компактний і високотехнологічний супутник на
вашому творчому шляху. Зробіть моменти вашого життя особливими, зафіксувавши їх з
повітря. Насолоджуйтесь польотом і отримуйте незвичайні кадри з інтуїтивно зрозумілим
додатком DJI Fly.\n" +
        "Маса Mavic Mini менше 250 г, що практично дорівнює вазі стандартного
смартфона. Це означає не тільки, що пристрій надзвичайно компактний, але і що він
входить в категорію дронів, які не потрібно реєструвати в багатьох країнах.",
    ImagePath = "~/Images/order8.jpg",
    StartPrice = 10000,
    InstantPrice = 15000,
    BetStep = 100,
    StartTime = CurrentTime.AddDays(-1).AddHours(9),
    FinishTime = CurrentTime.AddHours(9),
    IsFinish = false,
    UserId = 2,
    CategoryId = 1,
});

db.Bets.Add(new Bet
{
    Id = 1,
    Price = 10,
    Time = CurrentTime.AddDays(-2).AddHours(3),
    LotId = 1,
    UserId = 3

```

```

});

db.Bets.Add(new Bet
{
    Id = 2,
    Price = 25,
    Time = CurrentTime.AddDays(-2).AddHours(5).AddMinutes(35),
    LotId = 1,
    UserId = 4
});

db.Bets.Add(new Bet
{
    Id = 3,
    Price = 100,
    Time = CurrentTime.AddDays(-2).AddHours(18),
    LotId = 1,
    UserId = 3
});

db.Bets.Add(new Bet
{
    Id = 4,
    Price = 17500,
    Time = CurrentTime.AddDays(-1).AddHours(5),
    LotId = 5,
    UserId = 4
});

db.Bets.Add(new Bet
{
    Id = 5,
    Price = 19000,
    Time = CurrentTime.AddDays(-1).AddHours(17).AddMinutes(10),
    LotId = 5,
    UserId = 3
});

db.Bets.Add(new Bet
{
    Id = 6,
    Price = 10,
    Time = CurrentTime.AddDays(-1).AddHours(8).AddMinutes(25),
    LotId = 7,
    UserId = 3
});

db.Bets.Add(new Bet
{
    Id = 7,
    Price = 25,
    Time = CurrentTime.AddDays(-1).AddHours(9).AddMinutes(10),
    LotId = 7,
    UserId = 4
});

db.Bets.Add(new Bet
{
    Id = 8,
    Price = 35,
    Time = CurrentTime.AddDays(-1).AddHours(10).AddMinutes(35),
    LotId = 7,

```

```

        UserId = 3
    });

    db.Bets.Add(new Bet
    {
        Id = 9,
        Price = 12000,
        Time = CurrentTime.AddDays(-1).AddHours(10).AddMinutes(10),
        LotId = 8,
        UserId = 3
    });

    db.Bets.Add(new Bet
    {
        Id = 10,
        Price = 12500,
        Time = CurrentTime.AddDays(-1).AddHours(14).AddMinutes(45),
        LotId = 8,
        UserId = 4
    });

    base.Seed(db);
}
}
}

```

LoginModel.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class LoginModel
    {
        [Required(ErrorMessage = "Необхідно вказати логін")]
        [RegularExpression(@"[A-Za-z0-9_]{3,16}", ErrorMessage = "Некоректний логін")]
        [DisplayName("Логін")]
        public string Username { get; set; }

        [Required(ErrorMessage = "Необхідно вказати пароль")]
        [StringLength(32, MinimumLength = 8, ErrorMessage = "Довжина паролю має бути від 8 до 32 символів")]
        [DataType(DataType.Password)]
        [DisplayName("Пароль")]
        public string Password { get; set; }
    }
}

```

Lot.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class Lot
    {
        public int Id { get; set; }
        [Required]
        public string Title { get; set; }
        [Required]
        public string Description { get; set; }
        public string ImagePath { get; set; }
        [Required]
        public int StartPrice { get; set; }
        [Required]
        public int InstantPrice { get; set; }
        [Required]
        public int BetStep { get; set; }
        [Required]
        public DateTime StartTime { get; set; }
        [Required]
        public DateTime FinishTime { get; set; }
        [Required]
        public bool IsFinish { get; set; }

        //[Required]
        public int? UserId { get; set; }
        public User User { get; set; }
        [Required]
        public int CategoryId { get; set; }
        public Category Category { get; set; }
    }
}
```

LotAndBetAndBestBetVM.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class LotAndBetAndBestBetVM
    {
        public Lot Lot { get; set; }
        public Bet Bet { get; set; }
        public Bet BestBet { get; set; }
    }
}
```

```
}
```

LotAndBetAndWinAndUserVM.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class LotAndBetAndWinAndUserVM
    {
        public Lot Lot { get; set; }
        public Bet Bet { get; set; }
        public Win Win { get; set; }
        public User Winner { get; set; }
    }
}
```

LotAndBetAndWinVM.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class LotAndBetAndWinVM
    {
        public Lot Lot { get; set; }
        public Bet Bet { get; set; }
        public Win Win { get; set; }
    }
}
```

LotAndBetVM.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class LotAndBetVM
    {
        public Lot Lot { get; set; }
    }
}
```

```

    public Bet Bet { get; set; }
}
}

```

LotCreateModel.cs

```

using OnAuction.Attributes;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class LotCreateModel
    {
        [Required(ErrorMessage = "Необхідно вказати назву")]
        [DisplayName("Назва")]
        public string Title { get; set; }
        [Required(ErrorMessage = "Необхідно вказати опис")]
        [DataType(DataType.MultilineText)]
        [DisplayName("Опис")]
        public string Description { get; set; }
        [DisplayName("Зображення")]
        [FileSize(16, ErrorMessage = "Розмір зображення має бути менше 16 Мб")]
        [FileType(".png, .jpg, .jpeg, .bmp, .gif", ErrorMessage = "Зображення не відповідає формату (*.png, *.jpg, *.jpeg, *.bmp, *.gif)")]
        public HttpPostedFileBase Image { get; set; }
        [Required(ErrorMessage = "Необхідно вказати стартову вартість")]
        [Range(1, int.MaxValue, ErrorMessage = "Стартова вартість має бути не менше ніж 1 грн.")]
        [DisplayName("Стартова вартість, грн.")]
        public int StartPrice { get; set; }
        [Required(ErrorMessage = "Необхідно вказати вартість для моментального викупу")]
        [Range(1, int.MaxValue, ErrorMessage = "Вартість для моментального викупу має бути не менше ніж 1 грн.")]
        [DisplayName("Вартість для моментального викупу, грн.")]
        public int InstantPrice { get; set; }
        [Required(ErrorMessage = "Необхідно вказати крок ставки")]
        [Range(1, int.MaxValue, ErrorMessage = "Крок ставки має бути не менше ніж 1 грн.")]
        [DisplayName("Крок ставки, грн.")]
        public int BetStep { get; set; }
        [Required(ErrorMessage = "Необхідно вказати категорію лоту")]
        [DisplayName("Категорія")]
        public int CategoryId { get; set; }
    }
}

```

RegisterModel.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class RegisterModel
    {
        [Required(ErrorMessage = "Необхідно вказати логін")]
        [RegularExpression(@"[A-Za-z0-9_]{3,16}", ErrorMessage = "Некоректний логін")]
        [DisplayName("Логін")]
        public string Username { get; set; }

        [Required(ErrorMessage = "Необхідно вказати пароль")]
        [StringLength(32, MinimumLength = 8, ErrorMessage = "Довжина паролю має бути від 8 до 32 символів")]
        [DataType(DataType.Password)]
        [DisplayName("Пароль")]
        public string Password { get; set; }

        [Required(ErrorMessage = "Необхідно підтвердити пароль")]
        [DataType(DataType.Password)]
        [Compare("Password", ErrorMessage = "Введені паролі не співпадають")]
        [DisplayName("Підтвердження паролю")]
        public string PasswordConfirm { get; set; }

        [Required(ErrorMessage = "Необхідно вказати електронну пошту")]
        [EmailAddress(ErrorMessage = "Некоректна електронна пошта")]
        [DisplayName("Електронна пошта")]
        public string Email { get; set; }

        [Required(ErrorMessage = "Необхідно вказати номер телефону")]
        [RegularExpression(@"380[0-9]{9}", ErrorMessage = "Некоректний номер телефону. Формат: 380xxxxxxxxx")]
        [DisplayName("Номер телефону")]
        public string PhoneNumber { get; set; }

        [Required]
        [DisplayName("Продавець")]
        public bool IsSeller { get; set; }
    }
}

```

Role.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;

```

```

namespace OnAuction.Models
{
    public class Role
    {
        public int Id { get; set; }
        [Required]
        public string Name { get; set; }
    }
}

```

User.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;
using System.Web.Providers.Entities;

namespace OnAuction.Models
{
    public class User
    {
        public int Id { get; set; }
        [Required]
        public string Username { get; set; }
        [Required]
        public string Password { get; set; }
        [Required]
        public string Email { get; set; }
        [Required]
        public string PhoneNumber { get; set; }

        [Required]
        public int RoleId { get; set; }
        public Role Role { get; set; }
    }
}

```

Win.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;

namespace OnAuction.Models
{
    public class Win
    {
        public int Id { get; set; }
    }
}

```



```

[Required]
public int Price { get; set; }
[Required]
public DateTime Time { get; set; }

[Required]
public int LotId { get; set; }
public Lot Lot { get; set; }
[Required]
public int UserId { get; set; }
public User User { get; set; }
    }
}

```

CustomRoleProvider.cs

```

using OnAuction.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Data.Entity;

namespace OnAuction.Providers
{
    public class CustomRoleProvider : RoleProvider
    {
        public override string ApplicationName { get => throw new
NotImplementedException(); set => throw new NotImplementedException(); }

        public override void AddUsersToRoles(string[] usernames, string[] roleNames)
        {
            throw new NotImplementedException();
        }

        public override void CreateRole(string roleName)
        {
            throw new NotImplementedException();
        }

        public override bool DeleteRole(string roleName, bool throwOnPopulatedRole)
        {
            throw new NotImplementedException();
        }

        public override string[] FindUsersInRole(string roleName, string
usernameToMatch)
        {
            throw new NotImplementedException();
        }

        public override string[] GetAllRoles()
        {
            throw new NotImplementedException();
        }

        public override string[] GetRolesForUser(string username)

```

```

    {
        string[] roles = new string[] { };

        using(DataContext db = new DataContext())
        {
            User user = db.Users.Include(x => x.Role).FirstOrDefault(x => x.Username
== username);
            if(user != null && user.Role != null)
            {
                roles = new string[] { user.Role.Name };
            }
        }

        return roles;
    }

    public override string[] GetUsersInRole(string roleName)
    {
        throw new NotImplementedException();
    }

    public override bool IsUserInRole(string username, string roleName)
    {
        bool outputResult = false;

        using(DataContext db = new DataContext())
        {
            User user = db.Users.Include(x => x.Role).FirstOrDefault(x => x.Username
== username);
            if (user != null)
            {
                if (user.Role != null && user.Role.Name == roleName)
                    outputResult = true;
            }
        }

        return outputResult;
    }

    public override void RemoveUsersFromRoles(string[] usernames, string[]
roleNames)
    {
        throw new NotImplementedException();
    }

    public override bool RoleExists(string roleName)
    {
        throw new NotImplementedException();
    }
}
}

```

countdown.js

```

var remaining = $("span.jb_timer").text(),
    regex = /\d{1,2}/g,
    matches = remaining.match(regex),
    hours = matches[0],

```

```

    minutes = matches[1],
    seconds = matches[2],
    remainingDate = new Date();

    remainingDate.setHours(hours);
    remainingDate.setMinutes(minutes);
    remainingDate.setSeconds(seconds);

    var intvl = setInterval(function () {
        var totalMs = remainingDate.getTime(),
            hours, minutes, seconds;

        remainingDate.setTime(totalMs - 1000);

        hours = remainingDate.getHours();
        minutes = remainingDate.getMinutes();
        seconds = remainingDate.getSeconds();

        if (hours === 0 && minutes === 0 && seconds === 0) {
            clearInterval(intvl);
        }

        $("span.jb_timer").text(
            (hours >= 10 ? hours : "0" + hours) + ":" +
            (minutes >= 10 ? minutes : "0" + minutes) + ":" +
            (seconds >= 10 ? seconds : "0" + seconds));
    }, 1000);

```

Login.cshtml

```

@model OnAuction.Models.LoginModel

@{
    ViewBag.Title = "Login";
}

<h2>Авторизація</h2>

@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        <hr />
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        <div class="form-group">
            @Html.LabelFor(model => model.Username, htmlAttributes: new { @class =
"control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Username, new { htmlAttributes = new { @class
= "form-control" } })
                @Html.ValidationMessageFor(model => model.Username, "", new { @class = "text-
danger" })
            </div>
        </div>

        <div class="form-group">

```

```

        @Html.LabelFor(model => model.Password, htmlAttributes: new { @class =
"control-label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.Password, new { htmlAttributes = new { @class
= "form-control" } })
            @Html.ValidationMessageFor(model => model.Password, "", new { @class = "text-
danger" })
        </div>
    </div>

    <div class="form-group">
        <div class="col-md-offset-2 col-md-10">
            <input type="submit" value="Увійти" class="btn btn-default" />
        </div>
    </div>

    <div class="form-group">
        <div class="col-md-offset-2 col-md-10">
            Немає акаунту? @Html.ActionLink("Зареєструватися", "Register")
        </div>
    </div>
</div>
}

@section Scripts {
    @Scripts.Render("~/bundles/jqueryval")
}

```

Register.cshtml

```

@model OnAuction.Models.RegisterModel

@{
    ViewBag.Title = "Register";
}

<h2>Рєєстрація</h2>

@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        <hr />
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        <div class="form-group">
            @Html.LabelFor(model => model.Username, htmlAttributes: new { @class =
"control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Username, new { htmlAttributes = new { @class
= "form-control" } })
                @Html.ValidationMessageFor(model => model.Username, "", new { @class = "text-
danger" })
            </div>
        </div>

        <div class="form-group">

```

```

        @Html.LabelFor(model => model.Password, htmlAttributes: new { @class =
"control-label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.Password, new { htmlAttributes = new { @class
= "form-control" } })
            @Html.ValidationMessageFor(model => model.Password, "", new { @class = "text-
danger" })
        </div>
    </div>

    <div class="form-group">
        @Html.LabelFor(model => model.PasswordConfirm, htmlAttributes: new { @class =
"control-label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.PasswordConfirm, new { htmlAttributes = new {
@class = "form-control" } })
            @Html.ValidationMessageFor(model => model.PasswordConfirm, "", new { @class =
"text-danger" })
        </div>
    </div>

    <div class="form-group">
        @Html.LabelFor(model => model.Email, htmlAttributes: new { @class = "control-
label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.Email, new { htmlAttributes = new { @class =
"form-control" } })
            @Html.ValidationMessageFor(model => model.Email, "", new { @class = "text-
danger" })
        </div>
    </div>

    <div class="form-group">
        @Html.LabelFor(model => model.PhoneNumber, htmlAttributes: new { @class =
"control-label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.PhoneNumber, new { htmlAttributes = new {
@class = "form-control" } })
            @Html.ValidationMessageFor(model => model.PhoneNumber, "", new { @class =
"text-danger" })
        </div>
    </div>

    <div class="form-group">
        @Html.LabelFor(model => model.IsSeller, htmlAttributes: new { @class =
"control-label col-md-2" })
        <div class="col-md-10">
            <div class="checkbox">
                @Html.EditorFor(model => model.IsSeller)
                @Html.ValidationMessageFor(model => model.IsSeller, "", new { @class =
"text-danger" })
            </div>
        </div>
    </div>

    <div class="form-group">
        <div class="col-md-offset-2 col-md-10">
            <input type="submit" value="Зареєструватися" class="btn btn-default" />
        </div>
    </div>

    <div class="form-group">

```

```

        <div class="col-md-offset-2 col-md-10">
            Вже є акаунт? @Html.ActionLink("Увійти", "Login")
        </div>
    </div>
</div>
}

@section Scripts {
    @Scripts.Render("~/bundles/jqueryval")
}

```

Admin => Bets.cshtml

```

@model List<OnAuction.Models.Bet>

@{
    ViewBag.Title = "Bets";
}

<h2>Bci ставки</h2>

<table class="table table-bordered">
    <thead>
        <tr>
            <th class="text-center">#</th>
            <th class="text-center">Назва лоту</th>
            <th class="text-center">Номер лоту</th>
            <th class="text-center">Користувач</th>
            <th class="text-center">Ставка, грн.</th>
            <th class="text-center">Дата</th>
        </tr>
    </thead>
    <tbody>
        @for (int i = 0; i < Model.Count(); ++i)
        {
            <tr>
                <th class="text-right">@(i + 1)</th>
                <td class="wrap-text"
title="@Model[i].Lot.Title">@Html.ActionLink(Model[i].Lot.Title, "Info", "Lot", new {
LotId = Model[i].LotId}, "")</td>
                <td>@Model[i].LotId</td>
                <td>@Model[i].User.Username</td>
                <td>@Model[i].Price</td>
                <td>@Model[i].Time.ToString("dd.MM.yyyy HH:mm:ss")</td>
            </tr>
        }
    </tbody>
</table>

```

Wins.cshtml

```

@model List<OnAuction.Models.Win>

```

```

@{
    ViewBag.Title = "Wins";
}

<h2>Всі виграші</h2>

<table class="table table-bordered">
    <thead>
        <tr>
            <th class="text-center">#</th>
            <th class="text-center">Назва лоту</th>
            <th class="text-center">Номер лоту</th>
            <th class="text-center">Користувач</th>
            <th class="text-center">Ставка, грн.</th>
            <th class="text-center">Дата</th>
        </tr>
    </thead>
    <tbody>
        @for (int i = 0; i < Model.Count(); ++i)
        {
            <tr>
                <th class="text-right">@(i + 1)</th>
                <td class="wrap-text"
title="@Model[i].Lot.Title">@Html.ActionLink(Model[i].Lot.Title, "Info", "Lot", new {
LotId = Model[i].LotId }, "")</td>
                <td>@Model[i].LotId</td>
                <td>@Model[i].User.Username</td>
                <td>@Model[i].Price</td>
                <td>@Model[i].Time.ToString("dd.MM.yyyy HH:mm:ss")</td>
            </tr>
        }
    </tbody>
</table>

```

Lot => Bets.cshtml

```

@model List<OnAuction.Models.Bet>

@{
    ViewBag.Title = "Bets";
}

<h2 class="wrap-text">@(((OnAuction.Models.Lot)ViewBag.Lot).Title)</h2>

<table class="table table-bordered">
    <thead>
        <tr>
            <th class="text-center">#</th>
            <th class="text-center">Користувач</th>
            <th class="text-center">Ставка, грн.</th>
            <th class="text-center">Час</th>
        </tr>
    </thead>
    <tbody>
        @for (int i = 0; i < Model.Count(); ++i)
        {

```

```

        <tr>
            <th class="text-right">@(i + 1)</th>
            <td>@Model[i].User.Username</td>
            <td>@Model[i].Price</td>
            <td>@Model[i].Time.ToString("dd.MM.yyyy HH:mm:ss")</td>
        </tr>
    }
</tbody>
</table>

```

Create.cshtml

```

@model OnAuction.Models.LotCreateModel

@{
    ViewBag.Title = "Create";
}

<h2>Новий лот</h2>

@using (Html.BeginForm("Create", "Lot", FormMethod.Post, new { enctype =
"multipart/form-data" }))
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        <hr />
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        <div class="form-group">
            @Html.LabelFor(model => model.Title, htmlAttributes: new { @class = "control-
label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Title, new { htmlAttributes = new { @class
= "form-control" } })
                @Html.ValidationMessageFor(model => model.Title, "", new { @class = "text-
danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.Description, htmlAttributes: new { @class =
"control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Description, new { htmlAttributes = new {
@class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Description, "", new { @class =
"text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.Image, htmlAttributes: new { @class = "control-
label col-md-2" })
            <div class="col-md-10">
                @Html.TextBoxFor(model => model.Image, new { type = "file", access =
"Image/*" })
                @Html.ValidationMessageFor(model => model.Image, "", new { @class = "text-
danger" })
            </div>
        </div>
    </div>

```



```

    </div>
  </div>

  <div class="form-group">
    @Html.LabelFor(model => model.StartPrice, htmlAttributes: new { @class =
"control-label col-md-2" })
    <div class="col-md-10">
      @Html.EditorFor(model => model.StartPrice, new { htmlAttributes = new {
@class = "form-control" } })
      @Html.ValidationMessageFor(model => model.StartPrice, "", new { @class =
"text-danger" })
    </div>
  </div>

  <div class="form-group">
    @Html.LabelFor(model => model.InstantPrice, htmlAttributes: new { @class =
"control-label col-md-2" })
    <div class="col-md-10">
      @Html.EditorFor(model => model.InstantPrice, new { htmlAttributes = new {
@class = "form-control" } })
      @Html.ValidationMessageFor(model => model.InstantPrice, "", new { @class =
"text-danger" })
    </div>
  </div>

  <div class="form-group">
    @Html.LabelFor(model => model.BetStep, htmlAttributes: new { @class =
"control-label col-md-2" })
    <div class="col-md-10">
      @Html.EditorFor(model => model.BetStep, new { htmlAttributes = new {
@class = "form-control" } })
      @Html.ValidationMessageFor(model => model.BetStep, "", new { @class =
"text-danger" })
    </div>
  </div>

  <div class="form-group">
    @Html.LabelFor(model => model.CategoryId, htmlAttributes: new { @class =
"control-label col-md-2" })
    <div class="col-md-10">
      @Html.DropDownListFor(model => model.CategoryId, ViewBag.Categories as
SelectList, new { htmlAttributes = new { @class = "form-control" } })
      @Html.ValidationMessageFor(model => model.CategoryId, "", new { @class =
"text-danger" })
    </div>
  </div>

  <div class="form-group">
    <div class="col-md-offset-2 col-md-10">
      <input type="submit" value="Create" class="btn btn-default" />
    </div>
  </div>
</div>
}

@section Scripts {
    @Scripts.Render("~/bundles/jqueryval")
}

```

Info.cshtml

```

@model OnAuction.Models.LotAndBetVM

@{
    ViewBag.Title = "Info";
}

@if (Model.Lot.IsFinish)
{
    <h2>[Завершено] @Model.Lot.Title</h2>
}
else
{
    <h2>@Model.Lot.Title</h2>
}

<div clas="container">
    <div class="row">
        <div class="col-xs-12 col-sm-12 col-md-6 col-lg-6 col-xl-6">
            @if (Model.Lot.ImagePath == null)
            {
                
            }
            else
            {
                
            }
            <h4>
                Категорія: @Model.Lot.Category.Name
            </h4>
            <p>
                @Model.Lot.Description
            </p>
        </div>
        <div class="col-xs-12 col-sm-12 col-md-6 col-lg-6 col-xl-6">
            <div class="thumbnail">
                <p>Стартова вартість: @Model.Lot.StartPrice грн.</p>
                <p>
                    Поточна вартість (ставок
                    @Html.ActionLink(((int)ViewBag.BetsCount).ToString(), "Bets", new { LotId =
                    Model.Lot.Id })):
                    @if (Model.Bet == null)
                    {
                        @Model.Lot.StartPrice
                    }
                    else
                    {
                        @Model.Bet.Price
                    }
                    грн.
                </p>
                <p>Крок ставки: @Model.Lot.BetStep грн.</p>
                <p>Вартість моментальної купівлі: @Model.Lot.InstantPrice грн.</p>
                <p>Дата завершення аукціону: @Model.Lot.FinishTime.ToString("dd.MM.yyyy
                HH:mm:ss")</p>
                <p>
                    Залишилось часу:
                    @if (Model.Lot.IsFinish || Model.Lot.FinishTime <= DateTime.Now)

```

```

        {
            <span>00:00:00</span>
        }
        else
        {
            <span class="jb_timer">@((Model.Lot.FinishTime -
DateTime.Now).Hours.ToString("D2")):@((Model.Lot.FinishTime -
DateTime.Now).Minutes.ToString("D2")):@((Model.Lot.FinishTime -
DateTime.Now).Seconds.ToString("D2"))</span>
        }
    </p>

    @if (!Model.Lot.IsFinish && User.IsInRole("buyer"))
    {
        using (Html.BeginForm("MakeBet", "Lot", FormMethod.Post))
        {
            @Html.AntiForgeryToken()

            <input type="hidden" name="LotId" value="@Model.Lot.Id" />

            if (Model.Bet == null)
            {
                <input type="number" name="Price" inputmode="numeric" style="text-
align: right;" min="@Model.Lot.StartPrice" value="@Model.Lot.StartPrice"
step="@Model.Lot.BetStep" />
            }
            else
            {
                <input type="number" name="Price" inputmode="numeric" style="text-
align: right;" min="@((Model.Bet.Price + Model.Lot.BetStep)" value="@((Model.Bet.Price
+ Model.Lot.BetStep)" step="@Model.Lot.BetStep" />
            }
            <input type="submit" value="Зробити ставки" class="btn btn-default"
/>
        }
        using (Html.BeginForm("MakeInstantBet", "Lot", FormMethod.Post))
        {
            @Html.AntiForgeryToken()

            <input type="hidden" name="LotId" value="@Model.Lot.Id" />

            <input type="submit" value="Викупити за @Model.Lot.InstantPrice грн."
class="btn btn-default" />
        }
    }
</div>
</div>
</div>
</div>

@section Scripts {
    @Scripts.Render("~/bundles/jquery")
    @Scripts.Render("~/Scripts/countdown.js")
}

```

List.cshtml

```

@using OnAuction.Models

@{
    ViewBag.Title = "List";
}

<h1></h1>

<div class="container">
    <div class="row">
        <div class="col-xs-4 col-sm-4 col-md-2 col-lg-2 col-xl-2">
            <p>Kateropii:</p>
            <ul class="list-unstyled">
                <li>
                    @Html.ActionLink("Bci", "List")
                </li>
                @foreach (Category category in ViewBag.Categories)
                {
                    <li>
                        @Html.ActionLink(category.Name, "List", new { category = category.Id
            })
                    </li>
                }
            </ul>
        </div>
        <div class="col-xs-8 col-sm-8 col-md-10 col-lg-10 col-xl-10">
            <div class="row equal">
                @foreach (LotAndBetVM lot in ViewBag.LotAndBestBet)
                {
                    <div class="p-15 col-xs-12 col-sm-12 col-md-6 col-lg-4 col-xl-3">
                        <div class="thumbnail">
                            <div class="crop">
                                <div class="crop-wrapper">
                                    <div class="crop-content">
                                        @if (lot.Lot.ImagePath == null)
                                        {
                                            
                                        }
                                        else
                                        {
                                            
                                        }
                                    </div>
                                </div>
                            </div>
                        </div>
                        <div class="caption">
                            <h4 class="wrap-text"
title="@lot.Lot.Title">@lot.Lot.Title</h4>
                            <h5 class="w-100-right">
                                @if (lot.Bet == null)
                                {
                                    @lot.Lot.StartPrice.ToString()
                                }
                                else
                                {

```

```

        @lot.Bet.Price.ToString()
    }
    грн.
</h5>
<h6 class="w-100-right">
    @lot.Lot.FinishTime.ToString("dd:MM:yyyy HH:mm:ss")
</h6>
    <a href="@Url.Action("Info", "Lot", new { LotId =
lot.Lot.Id })" class="btn btn-default">Детальніше</a>
</div>
</div>
</div>
</div>
</div>
</div>

```

MyBetsActive.cshtml

```

@model List<OnAuction.Models.LotAndBetAndBestBetVM>
@{
    ViewBag.Title = "MyBetsActive";
}

<h2>Активні ставки</h2>

<table class="table table-bordered">
    <thead>
        <tr>
            <th class="text-center">#</th>
            <th class="text-center">Назва лоту</th>
            <th class="text-center">Моя ставка, грн.</th>
            <th class="text-center">Найвища ставка, грн.</th>
            <th class="text-center">Дата закінчення лоту</th>
        </tr>
    </thead>
    <tbody>
        @for (int i = 0; i < Model.Count; ++i)
        {
            <tr>
                <th class="text-right">@(i + 1)</th>
                <td class="wrap-text"
title="@Model[i].Lot.Title">@Html.ActionLink(Model[i].Lot.Title, "Info", "Lot", new {
LotId = Model[i].Lot.Id }, "")</td>
                @if (Model[i].Bet.Price == Model[i].BestBet.Price)
                {
                    <td>@Model[i].Bet.Price</td>
                }
                else
                {
                    <td style="color: red;">@Model[i].Bet.Price</td>
                }
                <td>@Model[i].BestBet.Price</td>
                <td>@Model[i].Lot.FinishTime.ToString("dd.MM.yyyy HH:mm:ss")</td>
            </tr>
        }
    </tbody>
</table>

```

```

    </tbody>
</table>

```

MyBetsHistory.cshtml

```

@model List<OnAuction.Models.LotAndBetAndWinVM>
@{
    ViewBag.Title = "MyBetsHistory";
}

<h2>Історія ставок</h2>

<table class="table table-bordered">
    <thead>
        <tr>
            <th class="text-center">#</th>
            <th class="text-center">Назва лоту</th>
            <th class="text-center">Моя ставка, грн.</th>
            <th class="text-center">Дата закінчення лоту</th>
            <th class="text-center">Виграш / Програш</th>
        </tr>
    </thead>
    <tbody>
        @for (int i = 0; i < Model.Count; ++i)
        {
            <tr>
                <th class="text-right">@(i + 1)</th>
                <td class="wrap-text"
title="@Model[i].Lot.Title">@Html.ActionLink(Model[i].Lot.Title, "Info", "Lot", new {
LotId = Model[i].Lot.Id }, "")</td>
                <td>@Model[i].Bet.Price</td>
                <td>@Model[i].Lot.FinishTime.ToString("dd.MM.yyyy HH:mm:ss")</td>
                @if (Model[i].Bet.UserId == Model[i].Win.UserId)
                {
                    <td style="color: green;">Виграш</td>
                }
                else
                {
                    <td style="color: red;">Програш</td>
                }
            </tr>
        }
    </tbody>
</table>

```

MyLots.cshtml

```

@model List<OnAuction.Models.LotAndBetAndWinAndUserVM>
@{
    ViewBag.Title = "MyLots";
}

```

```

<h2>Мої лоти</h2>

<table class="table table-bordered">
  <thead>
    <tr>
      <th class="text-center">#</th>
      <th class="text-center">Назва лоту</th>
      <th class="text-center">Найбільша ставка, грн.</th>
      <th class="text-center">Дата закінчення лоту</th>
      <th class="text-center">Статус</th>
      <th class="text-center">Переможець</th>
      <th class="text-center">Вартість, грн.</th>
    </tr>
  </thead>
  <tbody>
    @for (int i = 0; i < Model.Count; ++i)
    {
      <tr>
        <th class="text-right">@(i + 1)</th>
        <td class="wrap-text"
title="@Model[i].Lot.Title">@Html.ActionLink(Model[i].Lot.Title, "Info", "Lot", new {
LotId = Model[i].Lot.Id }, "")</td>
        @if (Model[i].Bet != null)
        {
          <td>@Model[i].Bet.Price</td>
        }
        else
        {
          <td>-</td>
        }
        <td>@Model[i].Lot.FinishTime.ToString("dd.MM.yyyy HH:mm:ss")</td>
        @if (Model[i].Lot.IsFinish && Model[i].Win != null)
        {
          <td style="color: green;">Завершено</td>
          <td>@Model[i].Winner.Username</td>
          <td>@Model[i].Win.Price</td>
        }
        else if (Model[i].Lot.IsFinish)
        {
          <td style="color: green;">Завершено</td>
          <td style="color: red;">Немає</td>
          <td>-</td>
        }
        else
        {
          <td style="color: blue;">В процесі</td>
          <td>-</td>
          <td>-</td>
        }
      </tr>
    }
  </tbody>
</table>

```

_Layout.cshtml

```
<!DOCTYPE html>
```

```

<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="icon" href="~/OnAuction.ico" type="image/vnd.microsoft.icon">
  <title>@ViewBag.Title - OnAuction</title>
  @Styles.Render("~/Content/css")
  @Scripts.Render("~/bundles/modernizr")
</head>
<body>
  <div class="navbar navbar-inverse navbar-fixed-top">
    <div class="container">
      <div class="navbar-header">
        <button type="button" class="navbar-toggle" data-toggle="collapse" data-
target=".navbar-collapse">
          <span class="icon-bar"></span>
          <span class="icon-bar"></span>
          <span class="icon-bar"></span>
        </button>
        @Html.ActionLink("OnAuction", "List", "Lot", new { area = "" }, new {
@class = "navbar-brand" })
      </div>
      <div class="navbar-collapse collapse">
        <ul class="nav navbar-nav">
          @if (User.Identity.IsAuthenticated)
          {
            if (User.IsInRole("admin"))
            {
              <li>@Html.ActionLink("Ставки", "Bets", "Admin")</li>
              <li>@Html.ActionLink("Виграші", "Wins", "Admin")</li>
            }
            else if (User.IsInRole("buyer"))
            {
              <li>@Html.ActionLink("Активні ставки", "MyBetsActive", "Lot")</li>
              <li>@Html.ActionLink("Історія ставок", "MyBetsHistory",
"Lot")</li>
            }
            else if (User.IsInRole("seller"))
            {
              <li>@Html.ActionLink("Новий лот", "Create", "Lot")</li>
              <li>@Html.ActionLink("Мої лоти", "MyLots", "Lot")</li>
            }
          }
        </ul>
        <ul class="nav navbar-nav navbar-right">
          @if (User.Identity.IsAuthenticated)
          {
            <li class="navbar-text">@User.Identity.Name</li>
            <li>@Html.ActionLink("Вийти", "Logout", "Account")</li>
          }
          else
          {
            <li>@Html.ActionLink("Увійти", "Login", "Account")</li>
            <li>@Html.ActionLink("Зареєструватися", "Register", "Account")</li>
          }
        </ul>
      </div>
    </div>
  </div>
</div>

```



```

<div class="container body-content">
    @RenderBody()
    <hr />
    <footer>
        <p>&copy; @DateTime.Now.Year - всі права захищено</p>
        <p>Спеціально для Олійника А.О.</p>
    </footer>
</div>

@Scripts.Render("~/bundles/jquery")
@Scripts.Render("~/bundles/bootstrap")
@RenderSection("scripts", required: false)
</body>
</html>

```

Error.cshtml

```

<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <meta name="viewport" content="width=device-width" />
    <link rel="icon" href="~/OnAuction.ico" type="image/vnd.microsoft.icon">
    <title>Помилка</title>
</head>
<body>
    <hgroup>
        <h1>Помилка.</h1>
        <h2>При обробці запиту на сервері "OnAuction" виникла помилка.</h2>
    </hgroup>
</body>
</html>

```

_ViewStart.cshtml

```

@{
    Layout = "~/Views/Shared/_Layout.cshtml";
}

```

Global.asax

```

using OnAuction.Models;
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using System.Web.Optimization;

```

```

using System.Web.Routing;

namespace OnAuction
{
    public class MvcApplication : System.Web.HttpApplication
    {
        protected void Application_Start()
        {
            Database.SetInitializer(new DataDBInitializer());

            AreaRegistration.RegisterAllAreas();
            FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);
            RouteConfig.RegisterRoutes(RouteTable.Routes);
            BundleConfig.RegisterBundles(BundleTable.Bundles);
        }
    }
}

```

Web.config

```

<?xml version="1.0" encoding="utf-8"?>
<!--
    Дополнительные сведения о настройке приложения ASP.NET см. на странице
    https://go.microsoft.com/fwlink/?LinkId=301880
-->
<configuration>
    <configSections>
        <!-- For more information on Entity Framework configuration, visit
        http://go.microsoft.com/fwlink/?LinkId=237468 -->
        <section name="entityFramework"
            type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework,
            Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"
            requirePermission="false" />
    </configSections>
    <connectionStrings>
        <add name="DataContext" connectionString="Data
Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename='|DataDirectory|\Data.mdf';Integrated
Security=True" providerName="System.Data.SqlClient" />
    </connectionStrings>
    <appSettings>
        <add key="webpages:Version" value="3.0.0.0" />
        <add key="webpages:Enabled" value="false" />
        <add key="ClientValidationEnabled" value="true" />
        <add key="UnobtrusiveJavaScriptEnabled" value="true" />
    </appSettings>
    <system.web>
        <authentication mode="Forms">
            <forms name="auth" timeout="21600" loginUrl="~/Account/Login" />
        </authentication>
        <compilation debug="true" targetFramework="4.7.2" />
        <customErrors mode="On"/>
        <httpRuntime targetFramework="4.7.2" />
        <roleManager enabled="true" defaultProvider="DefaultRoleProvider">
            <providers>
                <add name="DefaultRoleProvider"
                    type="OnAuction.Providers.CustomRoleProvider" />
            </providers>
        </roleManager>
    </system.web>
</configuration>

```

```

</system.web>
<runtime>
  <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
    <dependentAssembly>
      <assemblyIdentity name="Antlr3.Runtime" publicKeyToken="eb42632606e9261f"
/>
      <bindingRedirect oldVersion="0.0.0.0-3.5.0.2" newVersion="3.5.0.2" />
    </dependentAssembly>
    <dependentAssembly>
      <assemblyIdentity name="Newtonsoft.Json" publicKeyToken="30ad4fe6b2a6aeed"
/>
      <bindingRedirect oldVersion="0.0.0.0-12.0.0.0" newVersion="12.0.0.0" />
    </dependentAssembly>
    <dependentAssembly>
      <assemblyIdentity name="System.Web.Optimization"
publicKeyToken="31bf3856ad364e35" />
      <bindingRedirect oldVersion="1.0.0.0-1.1.0.0" newVersion="1.1.0.0" />
    </dependentAssembly>
    <dependentAssembly>
      <assemblyIdentity name="WebGrease" publicKeyToken="31bf3856ad364e35" />
      <bindingRedirect oldVersion="0.0.0.0-1.6.5135.21930"
newVersion="1.6.5135.21930" />
    </dependentAssembly>
    <dependentAssembly>
      <assemblyIdentity name="System.Web.Helpers"
publicKeyToken="31bf3856ad364e35" />
      <bindingRedirect oldVersion="1.0.0.0-3.0.0.0" newVersion="3.0.0.0" />
    </dependentAssembly>
    <dependentAssembly>
      <assemblyIdentity name="System.Web.WebPages"
publicKeyToken="31bf3856ad364e35" />
      <bindingRedirect oldVersion="1.0.0.0-3.0.0.0" newVersion="3.0.0.0" />
    </dependentAssembly>
    <dependentAssembly>
      <assemblyIdentity name="System.Web.Mvc" publicKeyToken="31bf3856ad364e35"
/>
      <bindingRedirect oldVersion="1.0.0.0-5.2.7.0" newVersion="5.2.7.0" />
    </dependentAssembly>
  </assemblyBinding>
</runtime>
<entityFramework>
  <providers>
    <provider invariantName="System.Data.SqlClient"
type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer" />
  </providers>
</entityFramework>
<system.codedom>
  <compilers>
    <compiler language="c#;cs;csharp" extension=".cs"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35" warningLevel="4"
compilerOptions="/langversion:default /nowarn:1659;1699;1701" />
    <compiler language="vb;vbs;visualbasic;vbscript" extension=".vb"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35" warningLevel="4"
compilerOptions="/langversion:default /nowarn:41008 /define:_MYTYPE=\"Web\"
/optionInfer+" />
  </compilers>
</system.codedom>
</configuration>

```

ДОДАТОК С
ОПИС ПРОГРАМИ

С.1 Загальні відомості

Програмна система створена в якості веб-застосунку для інтернет-аукціону з мінімалістичним й інтуїтивно зрозумілим інтерфейсом, що дозволяє користувачу брати участь в онлайн аукціонах.

Веб-застосунок використовує шаблон проектування MVC та реалізує мережеву архітектуру клієнт-сервер. На серверній стороні відбувається робота й обробка даних, а на стороні клієнта представлено інтерфейс для взаємодії з програмною системою.

Сторона клієнта розроблена у вигляді веб-сторінок. Сторона серверу знаходиться на окремому віддаленому комп'ютері

С.2 Функціональне призначення

Функціональне призначення програми – інтернет аналог звичайному аукціону, що використовується для проведення аукціонів за допомогою мережі Internet.

С.3 Опис логічної структури

Програмна система складається з таких частин:

- клієнтська частина: відповідає за отримання запитів від користувача, попередню валідацію введених даних, надсилання даних та отримання даних з сервера, інтерпретацію даних у зрозумілий для людини вигляд;
- серверна частина (сервер): відповідає за обробку запитів, валідацію на сервері отриманих даних та надання відповіді на запит;
- база даних: відповідає за зберігання даних та надання необхідної інформації на SQL запити.

С.4 Використані технічні засоби

Для проектування веб-застосунку було використано такі технології: .NET Framework, ASP.NET, C#, HTML, CSS, JavaScript, Razor, JQuery, Bootstrap, Entity Framework та MVC Framework.

С.5 Виклик та завантаження

Система використовує IIS (Internet Information Services) для доступності користувачам. Запуск починається з файлу Global.asax та файлів директорії App_Start. Завантаження клієнтської частини проходить за допомогою звертання до url адреси наданої сервером. Сервер потребує лише одного виклику для початку роботи, далі він працює самостійно.

С.6 Вхідні та вихідні дані

В залежності від задачі, що виконується, вхідними даними є: дані для авторизації та реєстрації, дані про ставку та лот. Також вхідними даними можна вважати сам запит до сервера (будь-яка URL адресу).

Вихідними даними є відповідь на запит користувача, в залежності від запиту користувачу має відобразитися відповідне представлення (веб-сторінка).