



POLSKO-JAPOŃSKA AKADEMIA TECHNIK KOMPUTEROWYCH

Wydział Informatyki

Katedra Baz Danych

Specjalizacja Bazy Danych

Michał Tulej

Nr albumu s25645

System Aukeyjny

Praca Inżynierska

Promotor

Krzysztof Bajszczyk

miejsce, miesiąc, rok obrony

Spis treści

1.	Faza analizy	3
1.1.	Wstęp	3
1.2.	Cele i założenia	3
1.3.	Wymagania funkcjonalne	4
1.3.1.	Opis funkcjonalności systemu	4
1.3.2.	Diagram przypadków użycia	6
1.3.3.	Scenariusze przypadków użycia	7
1.4.	Wymagania niefunkcjonalne	12
2.	Faza projektowania	13
2.1.	Ogólny opis architektury	13
2.2.	Aplikacje Klientkie	18
2.2.1.	Opis bibliotek oraz frameworków aplikacji frontendowej	18
2.2.2.	Opis bibliotek oraz szkieletów aplikacji backendowej	19
2.3.	Warstwa trwałości	22
2.3.1.	Rodzaj bazy danych	22
2.3.2.	Diagram encji	23
2.4.	Testy i wdrażanie	25
2.4.1.	Testy jednostkowe	26
2.4.2.	Opis przepływu	27
3.	Interfejs Użytkownika	29
4.	Wyniki i dyskusja	39
5.	Spis ilustracji	41
6.	Bibliografia	42

1. Faza analizy

1.1. Wstęp

Wraz z postępem cyfryzacji i szybkim rozwojem technologii informacyjnych, aplikacje aukcyjne są teraz niezbędnym elementem nowoczesnego handlu online. Pozwalają one nie tylko na transakcje kupna i sprzedaży różnego rodzaju produktów, lecz również na wprowadzenie nowatorskich strategii biznesowych, które rewolucjonizują sposób, w jaki klienci angażują się na rynku. W wskazanej pracy dyplomowej skupię się na stworzeniu aplikacji aukcyjnej, która wykorzystuje nowoczesne technologie oraz zasady UX/UI w celu stworzenia intuicyjnego i funkcjonalnego narzędzia dla użytkowników.

Mój projekt skupia się na przedstawieniu etapów tworzenia i uruchamiania aplikacji aukcyjnej, gdzie istotne są użytkowe, bezpieczeństwo transakcji i skuteczna komunikacja pomiędzy użytkownikami. W obliczu zwiększającej się konkurencji na rynku e-commerce, kluczowe jest zrozumienie potrzeb i oczekiwań potencjalnych klientów, co pozwoli na stworzenie produktu, który wyróżnia się spośród innych propozycji.

W pracy poruszę także trudności związane z tworzeniem aplikacji aukcyjnej, takie jak kontrola danych i zapewnienie poufności danych użytkowników. Również skoncentruję się na korzystaniu z frameworków i technologii, takich jak ASP.NET Core i Bootstrap, które umożliwiają szybkie tworzenie responsywnych i atrakcyjnych interfejsów użytkownika. Stworzenie aplikacji aukcyjnej przyczyni się do rozwoju zawodowego i praktycznego wykorzystania wiedzy teoretycznej w programowaniu, analizie wymagań i zarządzaniu projektem. Uważam, że moja praca przyczyni się do rozwoju lokalnego rynku e-commerce oraz będzie inspiracją dla kolejnych projektów w tej branży.

1.2. Cele i założenia

Celem aplikacji jest zapewnienie intuicyjnego interfejsu oraz bezpieczeństwa dzięki któremu bez żadnych ograniczeń będzie mogła z niego korzystać osoba pełnoletnia niezależnie w jakim wieku jest. Oprócz głównego celu, aplikacja umożliwia zapewnia:

- Zautomatyzowanie oraz ułatwienie wystawiania, licytowania, sprawdzania aukcjonowanych przedmiotów
- Zapewnienie bezpieczeństwa danych użytkowników przechowywanych w bazie danych
- Uniknięcie błędów ludzkich, które występują w formie zapominania o stacjonarnych aukcjach. Podczas internetowych aukcji, są wysyłane przypomnienia.

1.3.Wymagania funkcjonalne

1.3.1. Opis funkcjonalności systemu

Projektowana aplikacja ma na celu wsparcie firm obsługujących aukcje stacjonarne. Wdrożenie się na rynek internetowy, oraz zapewnienie swoim klientom efektywnego zarządzania aukcjami. Aplikacja posiada dwupoziomowe rozróżnianie użytkowników. W pierwszej kolejności dzielimy ich na dwie kategorie:

- Użytkownik zalogowany
- Użytkownik niezalogowany

Dla użytkownika który nie jest zarejestrowany w serwisie, możliwe jest przeglądanie aktualnych aukcji oraz tych objętych promocją na głównej stronie. Możliwe jest również zarejestrowanie się jako pełnoprawny użytkownik oraz zalogowanie.

Użytkownik zalogowany dzieli jest rozróżniany na podstawie roli którą wybrał podczas rejestrowania :

- Użytkownik Aukcjoner
- Użytkownik

Użytkownik posiada wszystkie funkcjonalności Aukcjонера, z wyjątkiem dodawania Aukcji. System automatycznie przydziela statusy. Są one dostępne do wglądu przez użytkowników zarejestrowanych. Automatycznie jest przydzielany status „active” po utworzeniu aukcji, natomiast po zakończeniu, „inactive”.

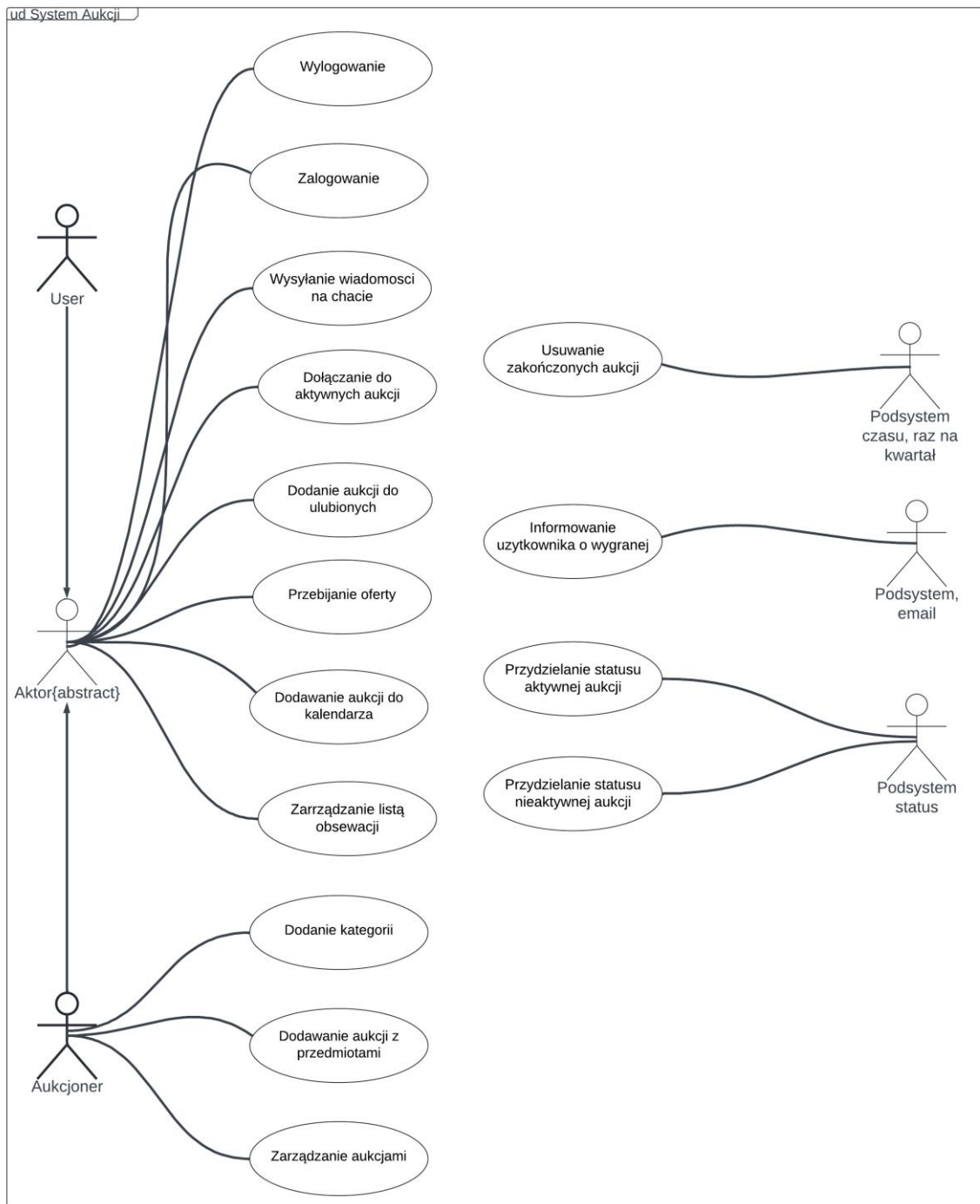
System będzie obsługiwał następujące funkcje :

- Dodawanie aukcji – w której ustalone jest m.in. cena wywoławcza, cena kup teraz, data zakończenia oraz data rozpoczęcia.
- Rejestrowanie użytkowników – walidowanie danych, oraz zapis w bazie danych
- Logowanie użytkowników – system wykorzystując JWT Token sprawdza użytkownika w bazie danych
- Ulubione aukcje – aukcje traktowane są jako obserwowane, oraz wyświetlane w oddzielnym miejscu w aplikacji
- Przeglądanie zakupionych przedmiotów w aukcji
- System powiadomień - który wyświetla niebanalne powiadomienia odnośnie zmienionej, przedmiocie który został kupiony opcją „Kup Teraz” przez innego użytkownika, dla zaistniałej korelacja pomiędzy użytkownikiem a przedmiotem.

Dostępne będą jeszcze powiadomienia przypominające o zakończeniu, rozpoczęciu aukcji, wygranej.

- Konwersacja - odnośnie produktu na specjalnie dedykowanym chacie, podczas licytowania
- Przebijanie oferty – użytkownik zapisany pozytywnie na aukcje, dostaje możliwość licytowania jej przedmiotów. Nie musi licytować wszystkich, ale jedynie te które go interesują. Istnieje możliwość kupienia przedmiotu „Kup Teraz” za ustaloną przez użytkownika cenę. Kupowany przedmiot uzależniony jest od najwyższej kwoty oferty z jaką został przebity. Dodawane jest do tej oferty 10zł , i zostaje ona wyświetlana w polu do przebijania oferty.
- Dodawanie aukcji – tylko jeden rodzaj roli w systemie może dodawać aukcje, jest to Aukcjoner. W żaden inny sposób nie można dodać aukcji w systemie. Funkcjonalności zmieniają się w zależności czy użytkownik jest zalogowany.
- Zarządzanie aukcjami – funkcjonalność pozwala dodać, usunąć, zmodyfikować aktualną aukcje przed jej rozpoczęciem.
- Kalendarz użytkownika – możliwość dodania aukcji do kalendarza, który w przejrzysty i klarowny sposób wyświetla użytkownikowi jego aukcje.
- Zapisywanie na aukcje – każdy zalogowany użytkownik ma możliwość zapisać się na aukcje, oprócz tego który założył aukcje. System nie pozwala licytować aukcji osobie która ją założyła.
- Dodanie kategorii – jeżeli znajduje się odpowiednia kategoria na liście, użytkownik może z niej skorzystać. Aplikacja umożliwia dodanie kategorii która pasuje dla danego przedmiotu, która będzie dostępna dynamicznie podczas dodawania aukcji.
- Przeglądanie aukcji oraz przedmiotów – użytkownik zalogowany oraz niezalogowany ma możliwość sprawdzania aukcji i przedmiotów.
- Ranking licytacji – wyświetla użytkownikowi najwyższe dla licytacji w odpowiednim polu tekstowym.
- Aukcje – dostępne jest sortowanie aukcji ze względu na status

1.3.2. Diagram przypadków użycia



1. Diagram przypadków użycia

1.3.3. Scenariusze przypadków użycia

Dodanie aukcji

Aktor: Użytkownik systemu

Warunek początkowy: Użytkownik ma rolę Aukcjoner.

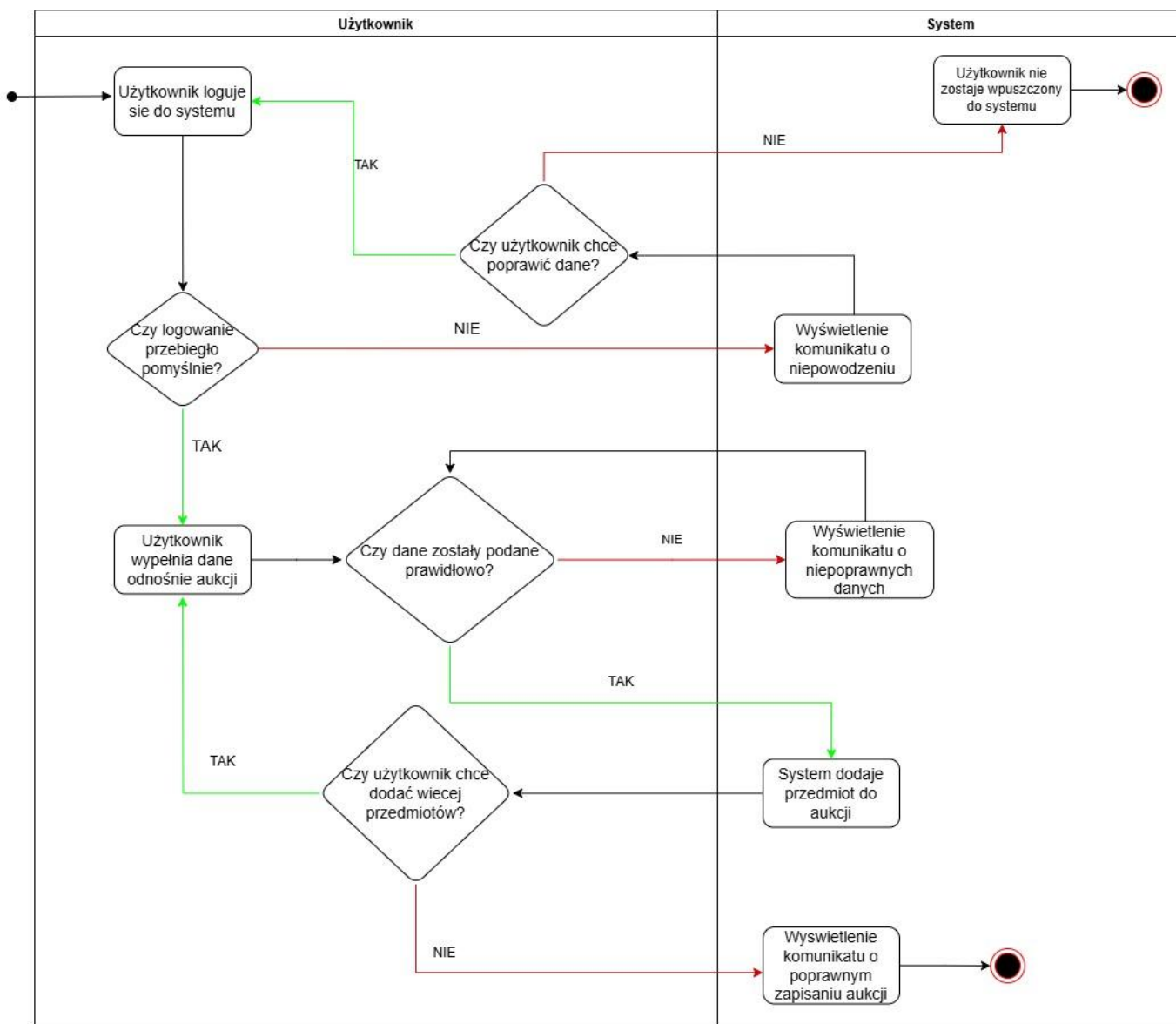
Warunek końcowy:

Dla przebiegu pomyślnego – Pomyślnie dodanie do aukcji systemowych oraz zapis do bazy danych.

Dla przebiegu niepomyślnego – Dane odnośnie aukcji nie zostaną zapisane w bazie danych.

Przebieg główny:

1. Użytkownik loguje się do systemu
 - a. System poprawnie loguje użytkownika
 - b. System kończy przypadek użycia i wyświetla komunikat o braku użytkownika.
2. Użytkownik wypełnia wszystkie wymagane pola tj. Auction Name, Category, Description, Start Date, End Date, Auction Photo oraz po naciśnięciu „Add item” dane przedmiotu który chce wystawić na aukcji
 - a. System poprawnie dołącza przedmiot do aukcji
 - b. System wyświetla komunikat o brakujących danych, i każe wypełnić/poprawić dane.
3. Po poprawnym dodaniu użytkownik winien zobaczyć swój przedmiot w tabeli Item List wraz z danymi które wypełnił oraz zdjęciem
4. Po zakończeniu dodawania przedmiotów do aukcji, użytkownik naciska „Save” i dostaje komunikat o poprawnym zapisaniu aukcji
 - a. Użytkownik dostaje komunikat o błędzie
 - b. Użytkownik zakończy niepowodzeniem dodanie aukcji
5. System weryfikuje datę rozpoczęcia i zakończenia aukcji, tworzy strukturę katalogową w której są zapisane zdjęcia. System przydziela odpowiedni status aukcji.
6. Użytkownik zakończy przypadek użycia



2.Scenariusz przypadku użycia " Dodanie Aukcji"

Licytowanie Aukcji

Aktor: Użytkownik systemu

Warunek początkowy: Użytkownik nie uczestniczy w swojej własnej aukcji.

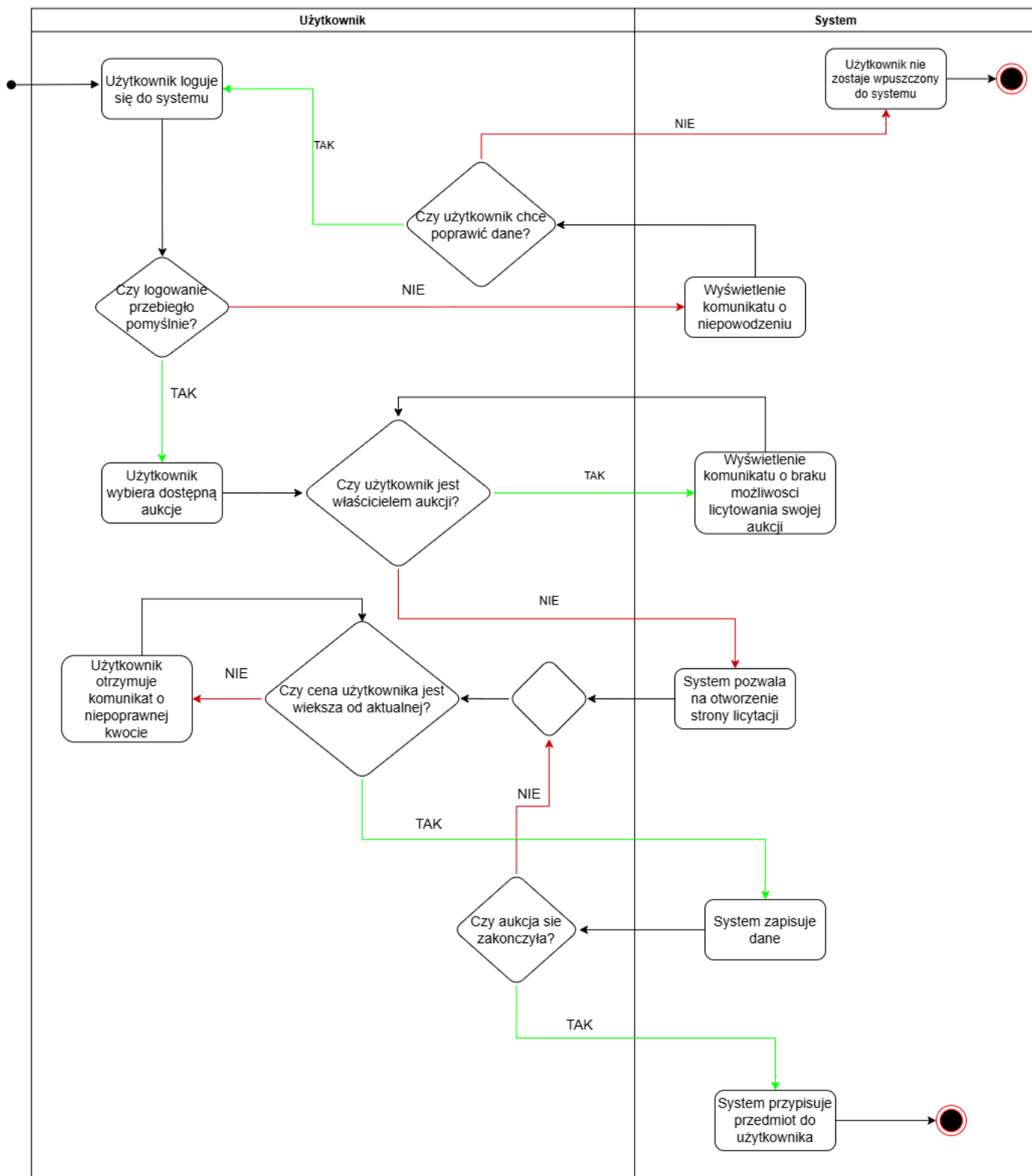
Warunek końcowy:

Dla przebiegu pomyślnego – Cena użytkownika, zostanie najwyższą i zapisze się w rankingu na miejscu pierwszym.

Dla przebiegu niepomyślnego – Wyskoczy komunikat o niepoprawnej cenie przebidania, system nie zapisze oferty przebidania dla danego przedmiotu.

Przebieg główny:

1. Użytkownik loguje się do systemu Aukcji.
2. System weryfikuje dane użytkownika.
3. Użytkownik aplikacji, zapisuje się poprzez naciśnięcie przycisku przy otwartej aukcji i interesującym go przedmiocie „Join”
4. System weryfikuje czy użytkownik nie jest właścicielem aukcji
 - a. Po poprawnej weryfikacji, system wpuszcza na stronę z licytowanie przedmiotu
 - b. System wyświetla komunikat o braku możliwości licytowania swoich aukcji.
5. Użytkownik licytuje przedmiot określoną kwotą
 - a. System zapisuje w bazie danych, informacje o kwocie przebidania oraz rezerwuje ofertę najwyższą wraz z użytkownikiem
 - b. System wyświetla komunikat że kwota którą użytkownik chce przebić licytowany przedmiot jest zbyt mała, i sugeruje ją poprawić
6. System sprawdza czas aukcji, i po jego upływie zamyka aukcję i przypisuje wygraną przedmiotu do użytkownika.
7. System przypisuje wygrany przedmiot do użytkownika w bazie danych
8. Użytkownik może zobaczyć wygrany przedmiot w zakładce „Bought Items”



3.Scenariusz przypadku użycia " Licytowanie Aukcji"

Logowanie użytkownika

Aktor: Użytkownik systemu

Warunek początkowy: Istniejące konto użytkownika

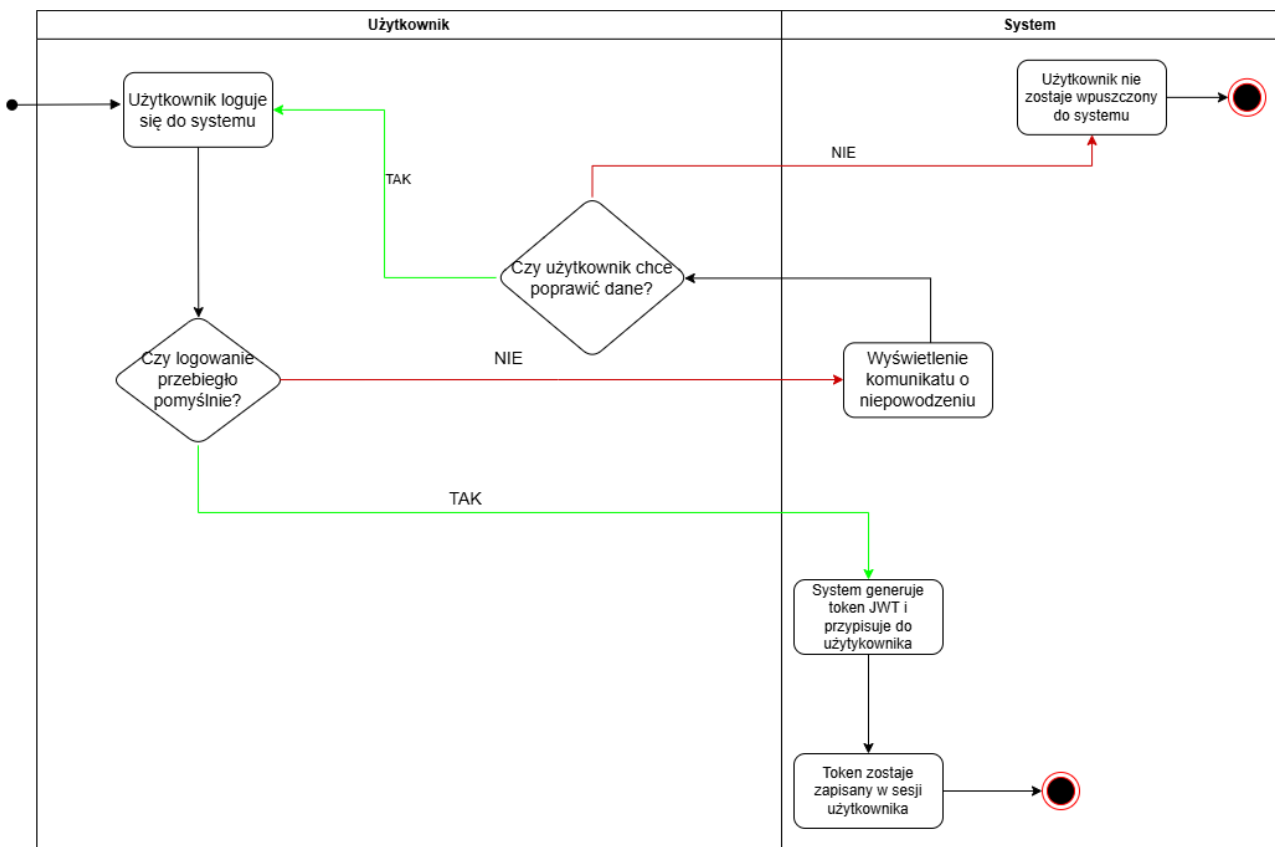
Warunek końcowy:

Dla przebiegu pomyślnego – Użytkownik poprawnie loguje się do serwisu i otrzymuje JWT token

Dla przebiegu niepomyślnego – Użytkownik nie loguje się do systemu

Przebieg główny:

1. Użytkownik loguje się do aplikacji
2. Użytkownik wprowadza swoje dane logowania
3. System weryfikuje dane
 - a. System wyświetla stosowny komunikat o niepoprawnych danych logowania
4. System generuje token JWT, z określoną rolą użytkownika. Odpowiada tokenem oraz zapisuje go w sesji.
5. Użytkownik otrzymuje token.



4.Scenariusz przypadku użycia "Logowanie użytkownika"

1.4. Wymagania niefunkcjonalne

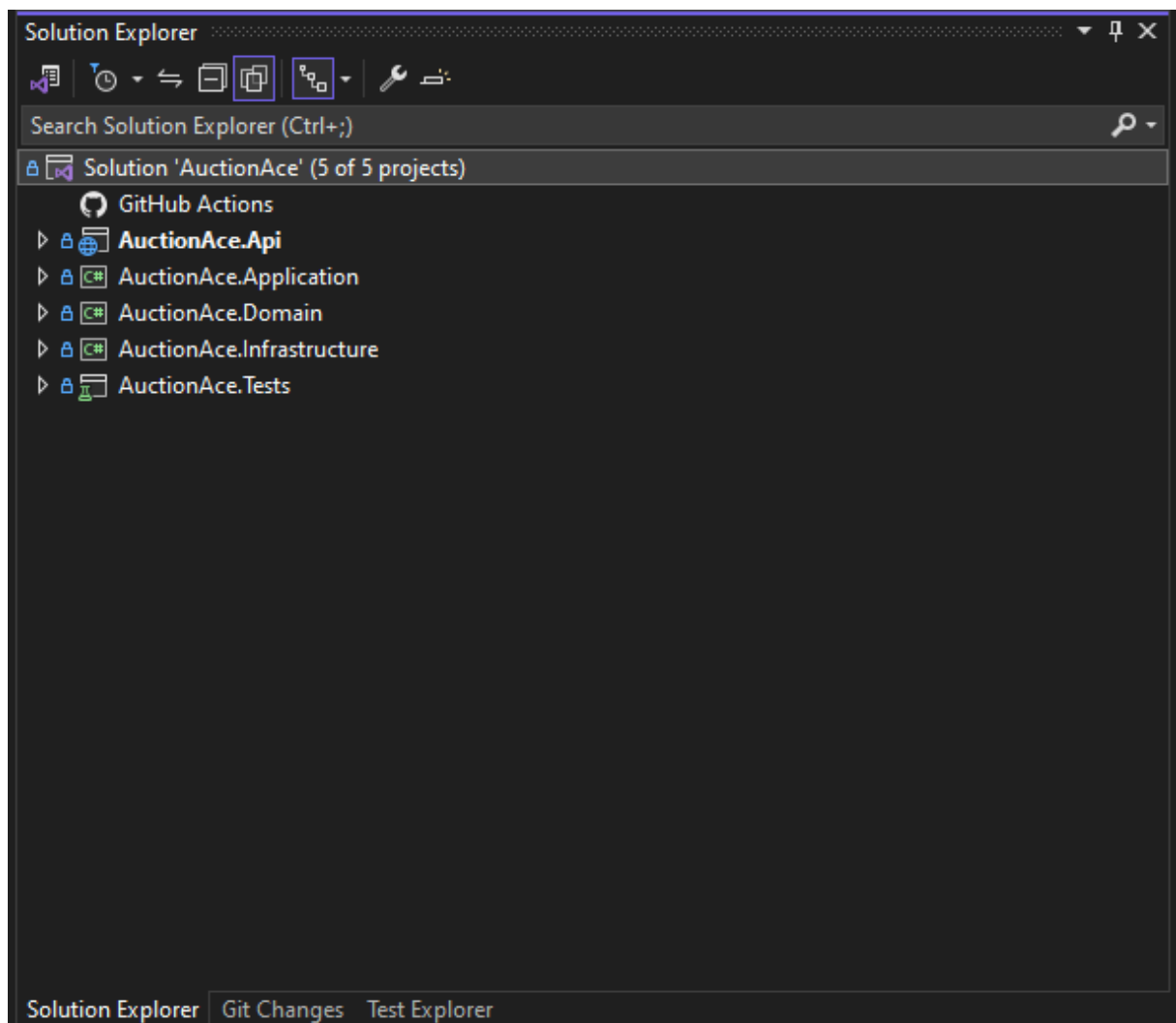
Dla systemu aukcyjnego, wymagania niefunkcjonalne obejmują następujące aspekty:

- **Wydajność** – system powinien obsłużyć jednocześnie do 1000 użytkowników, bez widocznych spadków wydajnościowych. Czas odpowiedzi na zapytania użytkowników nie będzie przekraczać 2 sekund w 95% przypadków.
- **Skalowalność** – dzięki zastosowanej architekturze projektowej, system jest bardzo prosty w rozbudowywaniu, dzięki czemu możemy wprost proporcjonalnie rozszerzać możliwości systemu.
- **Dostępność** – system powinien być dostępny przez 99,9%, co oznacza 52,56 minut maksymalnie niedostępności systemu, w którym przewidywane są aktualizacje.
- **Niezawodność** – odporny na błędy i awarie system. Tworzący automatyczne kopie zapasowe bazy danych, przynajmniej raz w tygodniu.
- **Kompatybilność** – system powinien być kompatybilny z najbardziej popularnymi przeglądarkami typu Chrome, Firefox, Safari, Edge.
- **Użyteczność** – prosty i łatwy interfejs użytkownika który pozwala w łatwy sposób nawigować między funkcjami.
- **Bezpieczeństwo** – logowanie użytkownika zabezpieczone nietrywialną usługą autentykacji do aplikacji opartej na tokenie. Nie ma możliwości nieautoryzowanego dostępu do aplikacji bez wcześniejszego zarejestrowania się.
- **Pojemność** – system powinien przechowywać dane do 1 000 000 użytkowników oraz 10 000 danych na temat aukcji i przedmiotów.

2. Faza projektowania

2.1.Ogólny opis architektury

Do stworzenia aplikacji została wybrana architektura warstwowa(layered architecture) która określa jasno jakie warstwy mają zdefiniowaną odpowiedzialność. Ma to na celu uporządkowanie i podział aplikacji na warstwy w celu zwiększenia modularności, czytelności i utrzymania kodu, co pozwala na lepsze zarządzanie złożonością aplikacji. Każda warstwa ma sztywno określone zależności, co pozwala na każdym programiście odnaleźć się w nowej aplikacji.



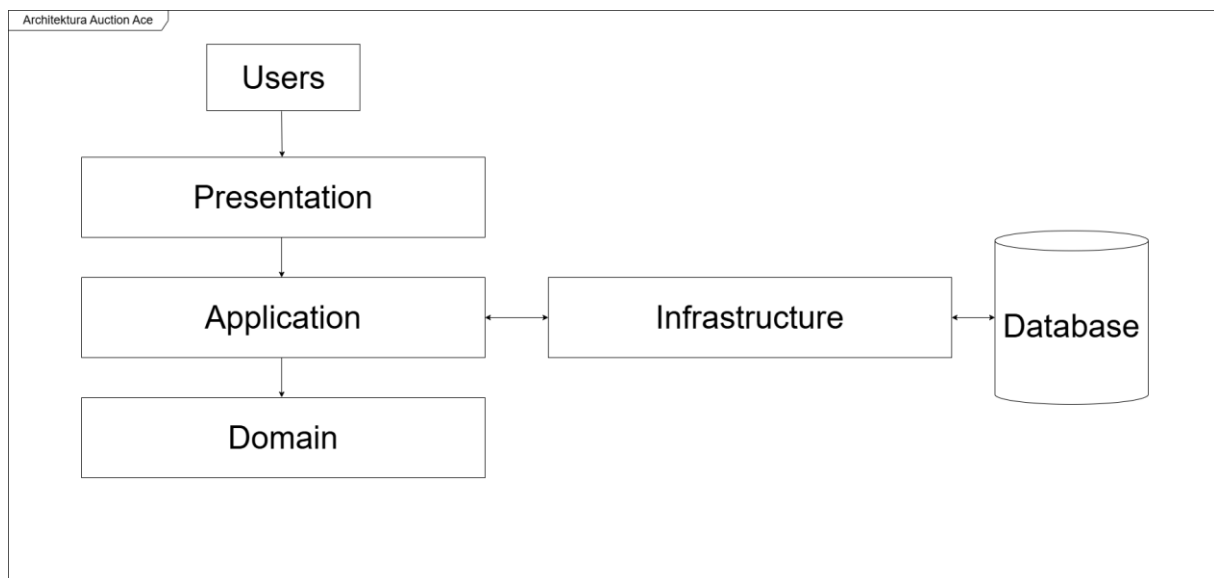
5.Architektura Aplikacji

Oto główne cele architektury warstwowej:

1. Rozdzielenie odpowiedzialności

- Warstwy odpowiadają za określony zestaw funkcji, dzięki czemu łatwiej jest rozwijać i utrzymywać kod.

- Warstwa prezentacji(Api/Presentation) zajmuje się wyłącznie wszelkimi operacjami, które zawierają interakcje użytkownika ze stroną.
- Warstwa logiki biznesowej (Application) zajmuje się przetwarzaniem danych i regułami biznesowymi
- Warstwa dostępu do danych (Infrastructure) obsługuje połączenia do bazy danych oraz wyciąga z niej wszelkie potrzebne informacje.
- Warstwa Domenowa odpowiedzialna jest za przechowywanie modeli danych używanych w aplikacji, co pozwala na uniknięcie przeciekania danych, ponieważ nie są używane bazodanowe modele.



6.Architektura Auction Ace (schemat)

2. Łatwość testowania i debugowania

Dzięki architekturze warstwowej poszczególne warstwy systemu są od siebie niezależne i można je niezależnie testować i debugować. Niezależność w testowaniu warstw jest niezwykle przydatna i ułatwiająca pracę programiście w testowaniu projektu aplikacji. W **warstwie logiki** można testować reguły biznesowe i algorytmy bez konieczności używania bazy danych ani interfejsu użytkownika.

W **warstwie dostępu do danych**, w wygodny sposób poprzez podstawianie danych możemy testować zgodność zapytań SQL, zgodność modeli danych oraz obsługę błędów tj. brak połączenia z bazą danych.

W **warstwie prezentacji** można testować wygląd interfejsu i jego responsywność i poprawność interakcji z użytkownikiem, poprawność przechwytywania danych wprowadzonych, niezależnie od działania logiki biznesowej.

3. Elastyczność i możliwość modyfikacji

Zastosowanie zmian w modelach bazodanowych poprzez dokonanie zmian w bazie danych przechodzi bardzo prosto. Nie jest wymagana zmiana logiki, jedynie co potrzeba to zmienić modele domenowe o potrzebne dane. Elastyczność warstw, pozwala na łatwą manipulację

4. Reużywalność kodu

Metody dostępne w serwisach jak i metody które wyciągają dane w repozytoriach, mogą być w łatwy sposób wykorzystywane w różnych miejscach np. w kontrolerze który umożliwia nam wykorzystywanie różnych metod z różnych serwisów.

```
namespace AuctionAce.Api.Controllers
{
    1 reference
    public class AuctionController : Controller
    {
        private readonly AuctionService _auctionService;
        private readonly LoginService _loginService;
        private readonly CategoryService _categoriesService;
        private readonly WishListService _wishlistService;

        0 references
        public AuctionController(AuctionService auctionService, LoginService loginService, CategoryService categoriesService, WishListService wishlistService)
        {
            _auctionService = auctionService;
            _loginService = loginService;
            _categoriesService = categoriesService;
            _wishlistService = wishlistService;
        }

        [HttpGet]
    }
}
```

7.Przykład reużywalności kodu

5. Zwiększenie skalowalności

Dzięki wyraźnemu rozdzieleniu odpowiedzialności między warstwami można wprowadzać nowe funkcjonalności, zmieniać istniejące moduły lub dodawać nowe bez ryzyka destabilizacji całej aplikacji.

6. Łatwość wprowadzania nowych technologii

System może ewoluować wraz z rozwojem technologii, a zmiany mogą być wdrażane w sposób elastyczny i kontrolowany.

Dzięki izolacji niezależności warstw można zmienić framework frontendowy bez konieczności modyfikacji logiki biznesowej czy warstwy dostępu.

Można również przejść na inny system bazodanowy lub ORM bez konieczności zmiany logiki.

7. Ułatwienie pracy zespołowej oraz bezpieczeństwo

Zastosowanie takiej architektury znacząco wspiera pracę zespołową oraz wprowadza mechanizmy zwiększające bezpieczeństwo aplikacji. Dzięki jasnemu podziałowi na warstwy, każda część aplikacji jest odpowiedzialna za konkretne zadania, co umożliwia jednoczesną pracę kilku zespołów nad różnymi aspektami projektu. Dodatkowo, warstwy można zabezpieczać indywidualnie, minimalizując ryzyko naruszeń i błędów.

Każda warstwa spełnia określone odpowiedzialności, są one nie bez przyczyny rozdzielone na osobne rozwiązania. Zajmują się one następującymi zadaniami:

- **Presentation** – warstwa prezentacji odpowiada za interakcje z użytkownikiem oraz zbieranie danych które zostały przez niego wprowadzone na stronie internetowej. Tutaj są wprowadzane dane za pomocą elementów interfejsu, takich jak formularze, pola tekstowe, przyciski czy listy rozwijane. Dane te są następnie przekazywane do kolejnych warstw aplikacji w celu ich przetwarzania i wykonywania odpowiednich operacji. W tym miejscu przechowywana jest cała struktura katalogowa wszystkich aukcji dostępnych w AuctionAce. Znajdują się tutaj również kontrolery które wywołują odpowiednie operacje po, przechwyceniu danych. W aplikacji aukcyjnej również zostały zastosowane „Huby” które wykorzystują bibliotekę SignalR i pozwalają w czasie rzeczywistym aktualizować dane. Również razem współpracują tutaj modele które reprezentują określony wygląd oraz wnętrze danych, oraz widoki dzięki którym możemy definiować wygląd strony internetowej.



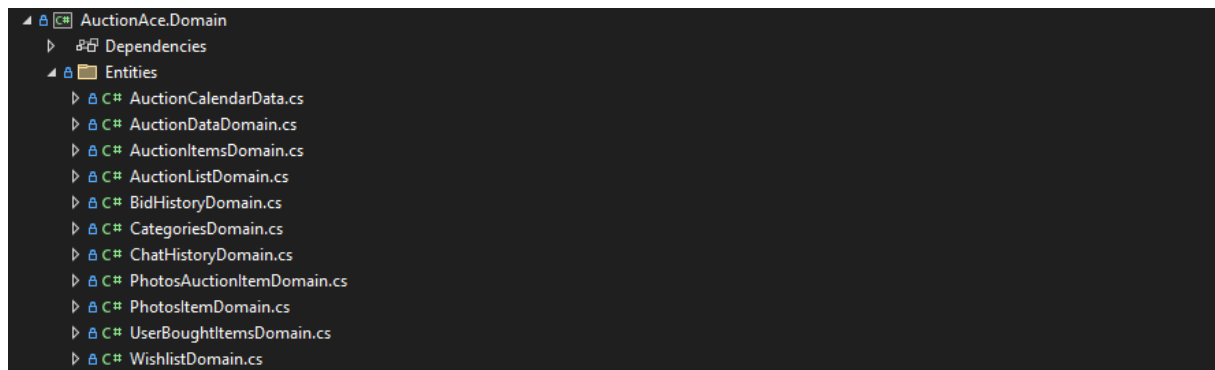
8. Warstwa Prezentacji

- **Application** – warstwa aplikacji jest inaczej mózgiem aplikacji. Znajduje się w niej cała logika biznesowa aplikacji. Wykonują się tutaj przeróżne metody oraz algorytmy, które mają wpływ na całą aplikację. Znajdziemy w niej interfejsy, które posłużyły do wyciągnięcia metod do testowania. Middleware znalazły zastosowanie do określania autoryzowanego dostępu do zasobów aplikacji. Działają one cały czas w tle aplikacji nasłuchując żądań użytkownika. Serwisy które służą do logicznych zadań lub wywołań repozytoriów.



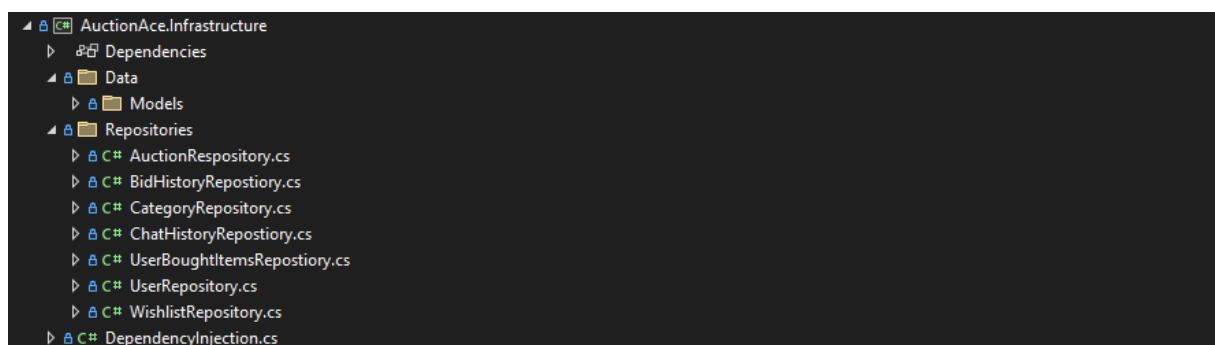
9. Warstwa Aplikacji

- Domain – warstwa aplikacji odpowiedzialna za przechowywanie i udostępnianie domenowych modeli danych. Przepływ w aplikacji, jest ograniczony w dwie strony. Podczas pobierania informacji z bazy danych, nasze dane przychodzą do serwisów w postaci modeli bazodanowych. Zawierają one dane zarówno potrzebne jak i nie potrzebne dla użytkownika końcowego. Z pomocą przychodzą modele domenowe, którym przypisuje się tylko te dane, które będą nam potrzebne. Reszta danych jest nie wykorzystywana i zostaje rozproszona warstwie aplikacji.



10. Warstwa Domenowa

- Infrastructure – warstwa infrastruktury aplikacji zajmuje się zarządzaniem wszystkimi elementami które wspierają funkcjonowanie aplikacji, ale nie są bezpośrednio związane z logiką biznesową ani interakcją z użytkownikiem. Obejmuje to m.in. zarządzaniem bazami danych, połączeniami z zewnętrznymi serwisami, dostępem do zasobów modeli bazy danych. Jest swoistym fundamentem aplikacji na którym opierają się warstwy systemu. Znajdują się w niej encje bazodanowe, dzięki którym możemy serializować obiekty. Zawierają się w również wszelkie połączenia do bazy danych, podzielone na odpowiadające im serwisy. Dzięki takiemu podejściu, warstwa odpowiedzialna za logikę biznesową może w zupełności skupić się na realizacji funkcji aplikacji, a nie na kwestiach technicznych.



11. Warstwa Infrastruktury

2.2.Aplikacje Klientkie

Warstwa kliencka inaczej można ją określić jako warstwa (User Interface), w której znajdują się wszystkie interakcje użytkownika z aplikacją. W niej użytkownicy wprowadzają dane, przeglądają dane, sortują. Klientka strona aplikacji, wysyła zapytania do backendowej strony, oraz pokazuje w przyjazny sposób użytkownikowi otrzymane dane.

Frameworki/Biblioteki są to gotowe rozwiązania, które używane przez programistów, pomagają w pisaniu wysokiej jakości kodu.

Główne składowe aplikacji frontendowych:

- Interface użytkownika
- Komunikacja z backendem
- Obsługa lokalnych zasobów
- Responsywność
- Bezpieczeństwo

Część odpowiadająca za logikę biznesową oraz przetwarzanie żądań użytkowników którzy korzystają z aplikacji. Posiadają serwisy które działają w tle i pozwalają na zarządzanie informacjami. Jedną z ważniejszych funkcji jest łączenie się do bazy danych oraz pozwalanie na przetwarzanie informacji zdobytych od użytkowników obsługujących frontendową część aplikacji.

Główne składowe aplikacji backendowych :

- Logik biznesowych
- Interfejsów
- Autoryzacji i uwierzytelnień
- Zarządzania błędami
- Integracje z frameworkami lub serwisami zewnętrznymi API
- Modeli danych

2.2.1. Opis bibliotek oraz frameworków aplikacji frontendowej

Zostały wykorzystane następujące biblioteki lub frameworki :

- Microsoft.AspNetCore.Components.QuickGrid.EntityFrameworkAdapter – biblioteka wykorzystywana do wyświetlania danych, w uporządkowany sposób danych w tabeli. Integruje funkcje QuickGrid wraz z bazą danych zarządzaną przez Entity Framework, dzięki czemu możemy w prosty sposób możemy wyświetlać dane pobrane z bazy danych.

- **CalendarJs** – biblioteka w aplikacji wykorzystana jest do wyświetlania w sposób przyjazny dla użytkownika, dzięki której można zobaczyć aukcje na które jest zapisany użytkownik. W wygodny sposób możemy przechodzić pomiędzy widokami dziennym, tygodniowymi oraz miesięcznymi. Jest tam dokładnie podana data oraz czas trwania wydarzenia.
- **Bootstrap 5** – framework, dzięki któremu w prosty sposób mogę rozlokować responsywnie elementy na stronie, dzięki zastosowaniu systemowi Grid. Bootstrap są to gotowe stylizacje strony tj. przyciski, przeglądanie zdjęć (Carousele). Daje to również możliwość konteneryzacji niektórych modułów, odpowiednie modyfikowanie ich wyglądu, dzięki opakowaniu ich w klasy kontenerów. W ten sposób są dużo prościej zarządzalne.
- **SweetAlert** – biblioteka która wyświetla użytkownikowi nowoczesne okna dialogowe, które informują, ostrzegają, komunikują o błędach albo mówią nam o osiągnięciu sukcesu, w funkcjonalności aplikacji. Łatwo personalizowane okienka pozwalają dostosować ich wygląd i wyświetlany tekst pod konkretną akcję w aplikacji. Dzięki responsywności łatwo dopasowują się do interakcji których potrzebujemy.
- **Ajax** – umożliwia przesyłanie zapytań http do wybranych kontrolerów. W prezentowanej aplikacji Ajax zbiera dane, które użytkownik uzupełnia na frontendzie aplikacji. Wykorzystując JavaScript można w prosty sposób przekazać dane do backendowych kontrolerów. Dużym plusem tej technologii jest jej asynchroniczne działanie. Użytkownik może korzystać ze strony i nie czekać na załadowanie się odpowiednich danych. Dynamiczność pozwala nam na wyświetlanie/aktualizowanie elementów bez potrzeby przeładowywania strony internetowej.

2.2.2. Opis bibliotek oraz szkieletów aplikacji backendowej

Zostały wykorzystane następujące biblioteki lub frameworki :

- **SignalR** – Microsoftowa Biblioteka która umożliwia dwukierunkową wymianę danych między klientem i serwerem. W aplikacji Aukcyjnej bardzo przydatną rzeczą którą udostępnia SignalR jest tworzenie grup (pokoi) i separacji danych które mają być udostępnione tylko dla określonej grupy osób np. podbijanie cen aukcji tylko i wyłącznie dla konkretnego przedmiotu i powinny widzieć historie zmian tylko osoby które dołączą do danej aukcji. Również chat aukcji został

stworzony za pomocą tej biblioteki, co pozwala na wyrażanie opinii użytkowników na temat bieżącej aukcji. Wykorzystując huby w prosty sposób wywołujemy metody serwisowe i uzyskujemy bądź aktualizujemy dane. Skalowalność pozwala na obsługę dużej liczby jednocześnie wysyłanych zapytań, jednocześnie SignalR wybiera automatycznie najbardziej optymalną ścieżkę komunikacji (Web Sockets, Server-Sent Events, Long Polling).

- `Microsoft.EntityFrameworkCore` – framework (ORM) który pozwala na korzystanie z bazy danych operując na modelach (obiektach). Dedykowanym językiem do „rozmawiania” z bazą jest język LINQ, pozwala ominąć budowanie nie efektywnych zapytań SQL, poprzez dostarczenie metod które pozwalają na operacje związane z bazą danych. ORM automatycznie mapuje encje z bazy danych do gotowych klas w języku C#, lub pozwala na podstawie klas wygenerować bazę danych. W przypadku podejścia Code First, zapewnia migracje, swojego rodzaju logi z których odczytujemy wszystkie zmiany w bazie danych. Framework jest w pełni asynchroniczny który pozwala w pełni wykorzystać jego potencjał w budowaniu aplikacji. Ważną cechą jest relacyjność pomiędzy encjami, która jest automatycznie przebudowywana w bazie danych po wykonaniu odpowiednich kroków.
- `System.IdentityModel.Tokens.Jwt` (Json Web Token) – biblioteka służąca do pomocy programiście w generowaniu tokenów służących do uwierzytelnienia użytkownika w aplikacji. Odpowiednie zaszyfowanie danych odnośnie np. roli użytkownika skutecznie pozwala na manipulację dostępem do określonych zasobów API. Równocześnie prosto można zweryfikować wygenerowany token na podstawie podpisu i sprawdzić czy nie został zmanipulowany poprzez osoby nie posiadające odpowiednich uprawnień.
- `Microsoft.AspNetCore.Authentication.JwtBearer` – pakiet .Net Core dzięki któremu możliwa jest weryfikacja i dostęp do zasobów udostępnionych przez API użytkownika. Po jego zastosowaniu wymagane jest dołączenie do odpowiednich endpointów słowa kluczowego [Authentication]. Na podstawie tego podejścia system zwraca uwagę na dołączane nagłówki do zapytań, i wspólnie z JWT Tokenem podejmowana jest decyzja o przepuszczeniu użytkownika do określonego zasobu.

- Microsoft.EntityFrameworkCore.SqlServer – pośrednik który umożliwia użycie baz danych z serwerem SQL. Zapewnia pełną integrację ORM z bazą danych.
- Microsoft.VisualStudio.Web.CodeGeneration.Design – pakiet Nugget, który umożliwia podejście Database First. Generowanie modeli bazodanowych rozpoczyna się od zaczytania ich z bazy danych oraz wygenerowania gotowych klas zawierających odzwierciedlenie bazy danych. Po takim zabiegu możemy efektywnie używać ORM, i wymieniać informacje.

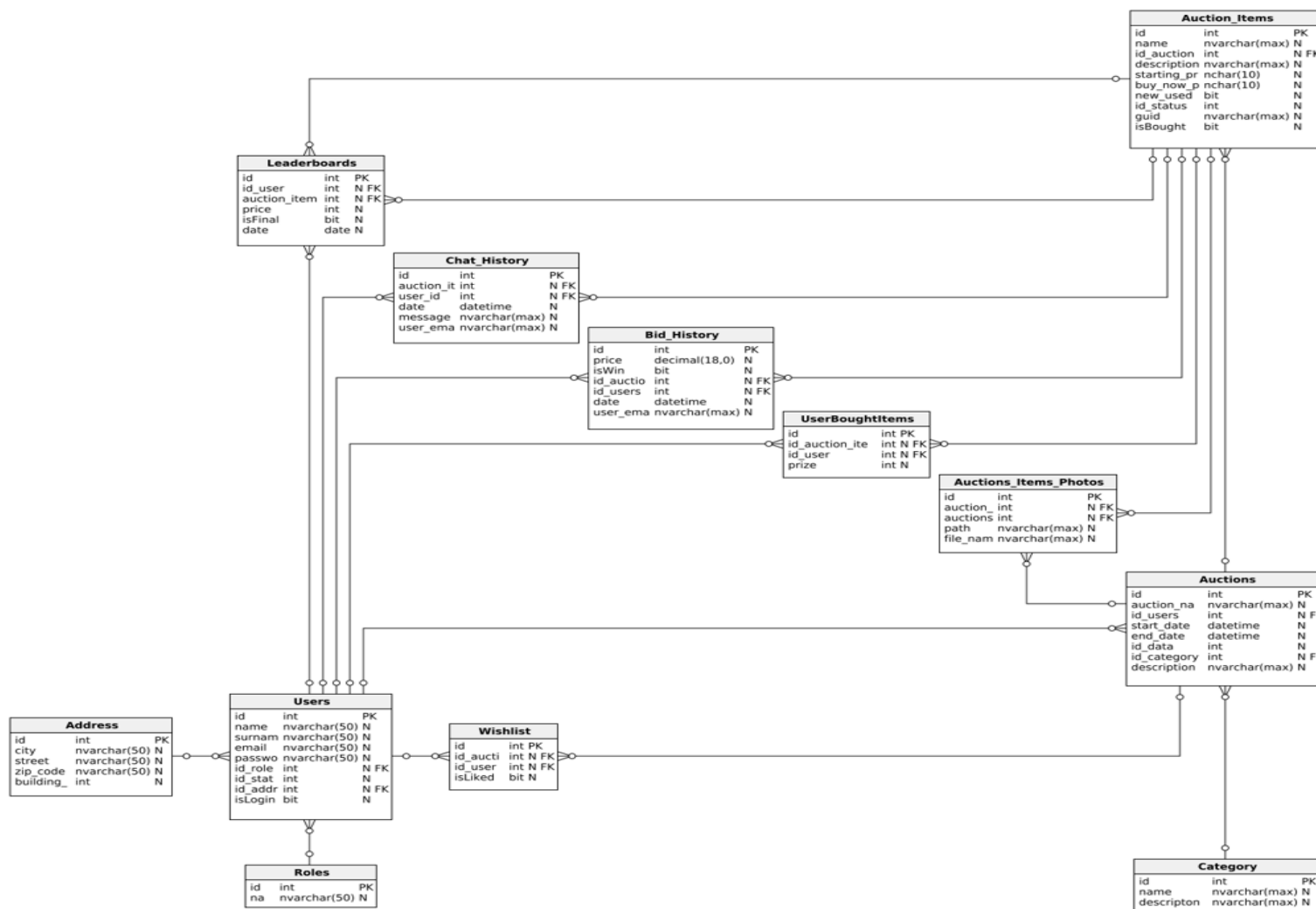
2.3. Warstwa trwałości

Kluczowe jest wybór odpowiedniego podejścia do zapisu danych generowanych przez naszą aplikację. Są różne podejścia odnośnie bezpieczeństwa, skalowalności czy odporności na niebezpieczeństwa z zewnątrz dla naszego sposobu przechowywania danych. Osoby niezwiązane bezpośrednio z informatyką, mogą nie zdawać sobie sprawy jak ważna przy wyborze technologii, jest wybranie poprawnej bazy danych. Przy nie poprawnym dobraniu naszego docelowego źródła danych, możemy paść ofiarą ataków hakerskich, jak i zarówno aplikacja może wydawać się nie dopracowana przez to że będzie działać bardzo powolnie z racji na duże zbiory danych. Regularne aktualizowanie naszego zaplecza technologicznego jest ważniejsze od ciągłego rozwijania aplikacji. Jednoznacznie porównując rodzaje ataków możemy stracić zaufanie publiczne do naszej firmy, po wycieku danych klientów. Warto również poświęcić uwagę na robienie kopii zapasowych. Są od tego specjalne oprogramowania oraz odpowiednio przeszkoleni ludzie, którzy mają wiedzę odnośnie poprawnego zarządzania danymi wrażliwymi.

2.3.1. Rodzaj bazy danych

Do realizacji aplikacji została użyta baza danych SQL, zarządzana przez SQL Server Management Studio 20. Jest to relacyjna baza danych, dzięki której efektywnie można zarządzać zbiorami danych oraz je przechowywać. Podejście pozwala definiować encje i między nimi relacje, co było niezwykle ważne w aplikacji, ponieważ dla niektórych operacji na użytkownikach czy ich interakcjach z aukcjami, potrzebowałem niejednokrotnie wybierać dane z kilku tabel. Zastosowane podejście relacyjne, w prosty sposób pozwalało na dostęp do tych danych. Management studio posiada intuicyjny interface, który pomaga w tworzeniu poprawnie zbudowanej bazy danych. Najbardziej przydatną opcją w tworzeniu bazy danych okazało się tworzenie dynamicznego diagramu bazy danych. Pomaga to w ustanawianiu relacji między tabelami, dokładne ustawienie typu danych jakie ma przyjmować encja czy definiowanie kluczy. Zawsze przy zapisywaniu takich diagramów jest od razu weryfikowana ich spójność, co zapobiega tworzeniu się błędów. W podejściu Database First, okazało się to niezwykle pomocne, ponieważ podczas scaffoldingu bazy danych do projektu, miałem pewność że wszystko będzie działało poprawnie. Nauczony doświadczeniem z zajęć, widziałem również że bazy danych stawiane na środowisku SQL, cechują się wysoką dostępnością, na czym bardzo zależało mi w procesie budowania aplikacji. Rozważane były również nierelacyjne bazy danych, jednakże ze względu na strukturalny charakter danych w projekcie oraz relacje między nimi, zdecydowałem się na wybór SQL.

2.3.2. Diagram encji



12.Diagram encji

Powyższy diagram encji opisuje zależności między relacjami w bazie danych oraz przedstawia możliwości bazy danych. Umożliwia on zarządzanie użytkownikami, aukcjami, oraz wgląd do danych funkcjonalnych wywodzących się z systemu aukcyjnego. Model jest kompleksowy i obejmuje wiele tabel powiązanych relacjami, co pozwala na przechowanie i przetwarzanie różnych typów danych związanych z system aukcji. Poniżej znajduje się opis poszczególnych encji:

- Users – tabela przechowuje dane odnośnie użytkowników, takie jak imię, nazwisko, email, adres, rola i status logowania. Jest ona powiązana tabelą Address i Roles.
- Address – przechowuje szczegółowe dane odnośnie adresu użytkownika
- Roles – podstawowa tabela w której znajdziemy spis ról w systemie. Rozdzielone na użytkownika oraz aukcjonera. Na tej podstawie wyznaczany jest dostęp użytkownika do atrybutów systemu.
- Auction_Items – opisuje wystawione przedmioty na aukcjach, w tym nazwę, opis, cenę wywoławczą, status, stan przedmiotu i podstawowe informacje. Jest ona powiązana z Auctions_Items_Photos, Auctions, Category.
- Auctions – tabela przechowuje informacje odnośnie aukcji, jej nazwę, opis oraz daty rozpoczęcia i zakończenia.
- Category – zawiera kategorie przedmiotów.
- Auctions_Items_Photos – są w niej opisane wszelkie ścieżki dostępowe do plików które użytkownik wrzuca jako zdjęcia aukcji.
- Wishlist – tabela listy życzeń użytkownika, przechowuje polubione aukcje.
- Bid_History – historia ofert, wszystkich użytkowników licytujących aukcje w systemie.
- UserBoughtItems – lista przedmiotów kupionych w systemie przez określonych użytkowników.
- Leaderboards – tabela rankingowa, która śledzi najlepszych użytkowników w kontekście aukcji i wygranych przedmiotów.
- Chat_History – historia wiadomości czatu, przypisana do aukcji i użytkowników.

2.4. Testy i wdrażanie

Współczesne aplikacje internetowe, wymagają niezwykle wysokiej niezawodności i dokładności działania, szczególnie w obszarach kluczowych dla użytkownika. W konsekwencji faza testowania stanowi istotny element cyklu życia rozwoju aplikacji. Proces testowania ułatwia nie tylko wczesną identyfikację błędów kodowania, ale także weryfikuje, czy wszystkie funkcjonalności aplikacji działają zgodnie z ustalonymi specyfikacjami projektowymi. Szczególnie krytycznym aspektem procedury testowej jest walidacja dokładności logiki biznesowej aplikacji, co osiąga się poprzez wdrożenie testów jednostkowych.

W trakcie opracowywania aplikacji aukcyjnej duży nacisk położono na wykonanie i administrowanie testami jednostkowymi z wykorzystaniem szeroko rozpoznawanego frameworka xUnit. Testy jednostkowe służą jako podstawowy instrument umożliwiający ocenę funkcjonalności poszczególnych komponentów systemu w izolacji od szerszej bazy kodu. W ramach aplikacji aukcyjnej metodologia ta ma szczególne znaczenie, ponieważ wiele aspektów działania systemu zależy od skomplikowanych algorytmów i interakcji z bazą danych. Podstawowym celem testowania jednostkowego było sprawdzenie integralności podstawowych funkcjonalności aplikacji.

W fazie testowania zastosowano metodologie obejmujące „mockowanie” co umożliwiała symulowanie działania zależności aplikacji tj. bazy danych, w celu złagodzenia potencjalnych zakłóceń podczas testowania jednostkowego. Należy podkreślić że wybór Xunit jako struktury testowej był celowy. Platforma ta zapewnia kompleksowe funkcje konfiguracji testów i ułatwia skuteczne zarządzanie cyklami życia testów poprzez mechanizmy inicjalizacji i dystrybucji zasobów. Ponadto Xunit wykazuje dużą kompatybilność ze współczesnymi środowiskami programistycznymi, co znacznie upraszcza integrację testów jednostkowych w aplikacji.

Wdrożenie testów jednostkowych doprowadziło do zidentyfikowania i rozwiązania kilku potencjalnych problemów przed wejściem aplikacji w fazę integracji i testowania systemu. Procedura ta nie tylko poprawiła stabilności i jakość kodu, ale także przyczyniła się do skrócenia czasu potrzebnego na naprawienie błędów na kolejnych etapach rozwoju projektu. Poniższe podsekcje określają specyfikacje przeprowadzonych testów jednostkowych i wyjaśniają postęp procesu testowania i wdrażania aplikacji.

2.4.1. Testy jednostkowe

Nazwa metody testującej składa się z:

1. Nazwy testowanej metody – wskazuje która metoda jest testowana
2. Warunki wejściowe – nazwa opisująca dane które zostają wprowadzone w jakich okolicznościach test jest wykonywany.
3. Oczekiwany wynik – informujący, co test powinien zwrócić.

```
[Fact]
0 | 0 references
public async Task UserRegister_WithValidUser_ShouldReturnTrue()
{
```

13.Przykład nazewnictwa testów

W powyższym przykładzie nazwa metody `UserRegister_WithValidUser_ShouldReturnTrue` jasno określa, że testowana jest metoda Rejestracji użytkownika która ma poprawne dane i powinna zwrócić `true`, w przypadku poprawnego zarejestrowania danych użytkownika w systemie.

Ponadto zastosowane zostało nazewnictwo wewnątrz metod, które pozwala odnaleźć każdemu programiście co zostało zrobione w metodzie i jak można ją poprawić lub ulepszyć:

- `// Arrange` – pod komentarzem są linijki odpowiedzialne za przygotowanie danych do podstawienia do metody
- `// Act` – działanie, czyliwołanie metody odpowiedzialnej z serwisu
- `// Assert` – oczekiwane dane zwrócone, w przypadku poprawnego przejścia testu

```
[Fact]
0 | 0 references
public async Task UserLogin_WithValidCredentials_ShouldReturnUser()
{
    // Arrange
    var email = "test@example.com";
    var password = "testpassword";
    var expectedUser = new Infrastructure.Data.Models.User
    {
        Id = 25,
        Email = email,
        Name = "Test",
        Surname = "User"
    };

    _userServiceMock
        .Setup(x => x.UserLogin(email, password))
        .ReturnsAsync(expectedUser);

    // Act
    var result = await _userServiceMock.Object.UserLogin(email, password);

    // Assert
    Assert.NotNull(result);
    Assert.Equal(email, result.Email);
    Assert.Equal("Test", result.Name);
    Assert.Equal("User", result.Surname);
}
```

14.Przykładowa metoda z testów

2.4.2. Opis przepływu

Rozpoczęcie przepływu danych zaczyna się od dziedziczenia. Przez takie podejście, można uzyskać dostęp do metod serwisu.

```
namespace AuctionAce.Application.Services
{
    4 references
    public class UserService : IUserService
    {
        public readonly UserRepository _userRepository;

        0 references
        public UserService(UserRepository userRepository)
        {
            _userRepository = userRepository;
        }
    }
}
```

15.Opis przepływu testów zdjęcie 1

Ekstraktując metody, do IUserInterface możemy później je wywołać na rzecz testu.

```
namespace AuctionAce.Application.Interfaces
{
    3 references
    public interface IUserService
    {
        4 references | 1/1 passing
        Task<User> UserLogin(string email, string password);
        4 references | 1/1 passing
        Task<bool> UserRegister(User user);
    }
}
```

16.Opis przepływu testów zdjęcie 2

Później następuje wywołanie konstruktora który inicjalizuje frameworka odpowiedzialnego za Mockowanie danych.

```
namespace AuctionAce.Tests.Tests.User
{
    1 reference
    public class UserServiceTests
    {
        private readonly Mock<IUserService> _userServiceMock;

        0 references
        public UserServiceTests()
        {
            _userServiceMock = new Mock<IUserService>();
        }
    }
}
```

17.Opis przepływu testów zdjęcie 3

Z takimi dostęпами do odpowiednich metod, możemy napisać test, przygotować dane oraz wywołać inetersująca nas konkretnie metode i podstawić do niej dane i zobaczyć co ona będzie zwracać. Dzięki temu możemy sprawdzić poprawność działania metody w różnych scenariuszach, analizować jej wyniki w zależności od wprowadzonych danych oraz upewnić się że, spełnia oczekiwania i działa zgodnie z założeniami.

```

[Fact]
public async Task UserLogin_WithValidCredentials_ShouldReturnUser()
{
    // Arrange
    var email = "test@example.com";
    var password = "testpassword";
    var expectedUser = new Infrastructure.Data.Models.User
    {
        Id = 25,
        Email = email,
        Name = "Test",
        Surname = "User"
    };

    _userServiceMock
        .Setup(x => x.UserLogin(email, password))
        .ReturnsAsync(expectedUser);

    // Act
    var result = await _userServiceMock.Object.UserLogin(email, password);

    // Assert
    Assert.NotNull(result);
    Assert.Equal(email, result.Email);
    Assert.Equal("Test", result.Name);
    Assert.Equal("User", result.Surname);
}

```

18.Opis przepływu testów zdjęcie 4

Na podstawie tego, możemy upewnić się że kod jest wolny od błędów, wszystkie zaprogramowane testy funkcjonalności są wolne od błędów i na tym etapie działają poprawnie, a ewentualne regresje są wychwytywane na wczesnym etapie budowania aplikacji aukcyjnej.

The screenshot shows the Test Explorer window in Visual Studio. The status bar at the top indicates 'Ready' with '0 Warnings' and '0 Errors'. The test results are organized in a tree view on the left, showing the following structure:

Test	D...	Tr...	E
▲ ✓ AuctionAce.Tests (5)	83...		
▲ ✓ AuctionAce.Tests.Tests.Auc...	42...		
▲ ✓ AuctionServiceTests (3)	42...		
✓ AddAuctionAsync_Sho...	40...		
✓ GetAuctionsAsync_Sh...	1 ms		
✓ GetRemainingTimeFor...	1 ms		
▲ ✓ AuctionAce.Tests.Tests.User	41...		
▲ ✓ UserServiceTests (2)	41...		
✓ UserLogin_WithValidC...	1 ms		
✓ UserRegister_WithVali...	40...		

On the right side, the 'Group Summary' for 'AuctionAce.Tests' is displayed:

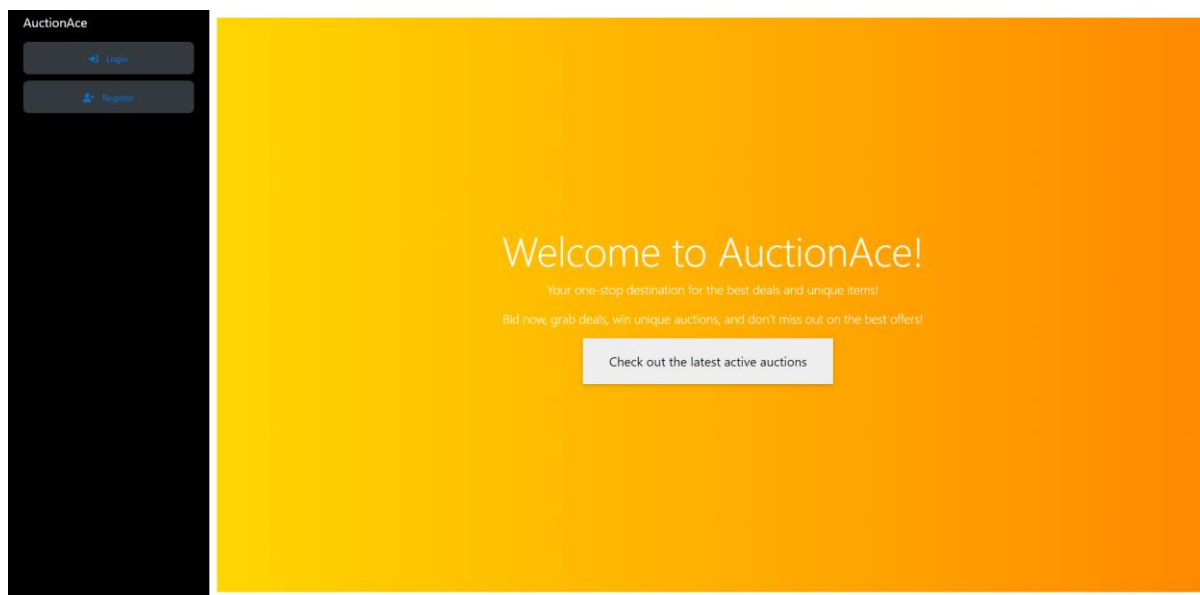
- Tests in group: 5
- Total Duration: 83 ms
- Outcomes: 5 Passed

At the bottom, the 'Test Explorer' tab is active, alongside 'Solution Explorer' and 'Git Changes'.

19.Opis przepływu testów zdjęcie 5

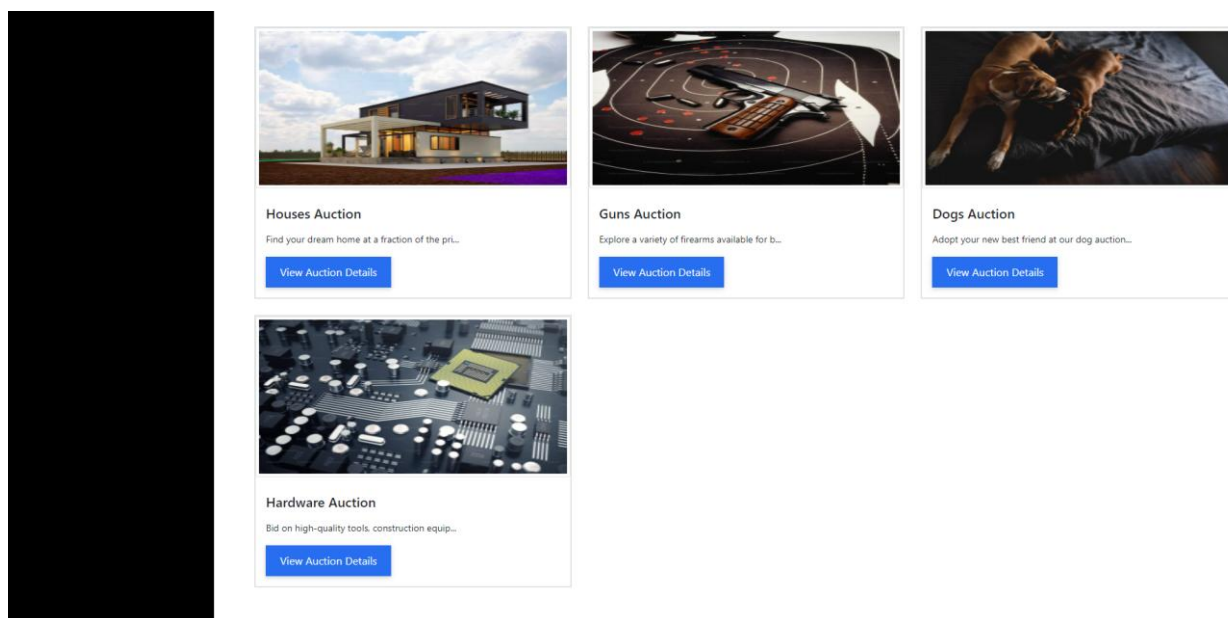
3. Interface Użytkownika

Wchodząc na stronę serwisu aukcyjnego AuctionAce, użytkownik ma do wyboru naciśnięcie 3 przycisków które wywołują określone akcje.



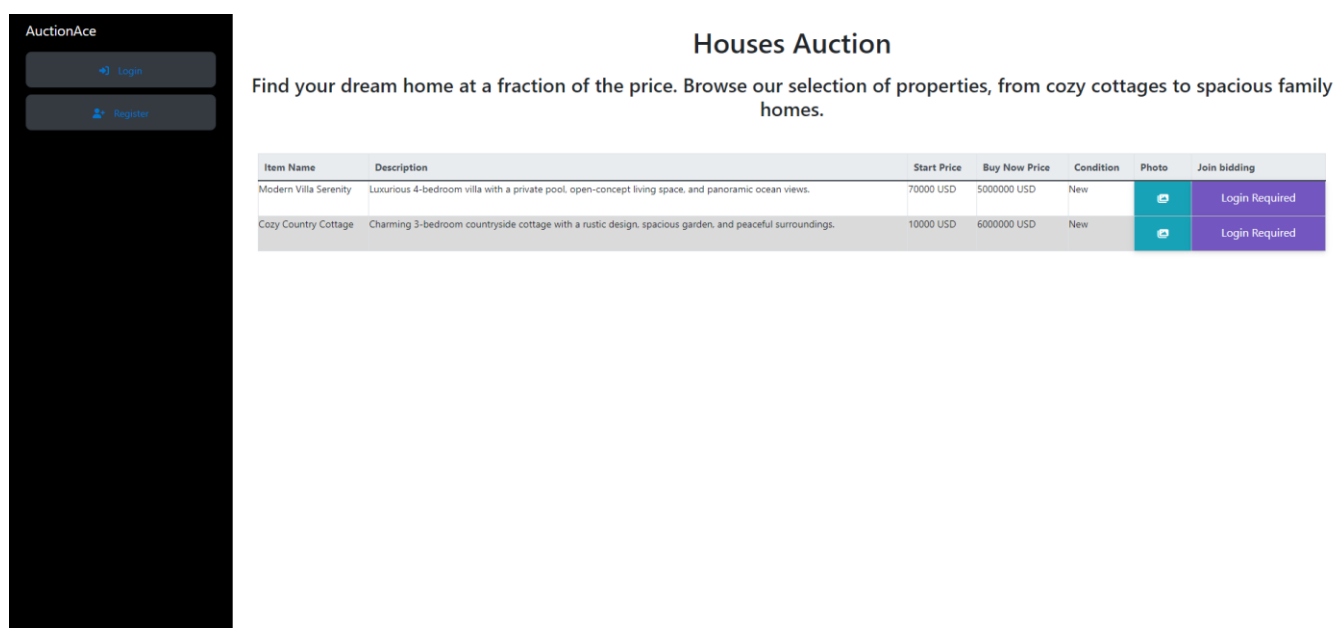
20.Interface użytkownika

Po naciśnięciu przycisku „Check out latest active auctions” użytkownik zostaje zescrollowany niżej na tej samej stronie do przykładowych aktywnych aukcji, dla których może tylko i wyłącznie przeglądać jakie przedmioty są dostępne do licytacji, ale nie może licytować.



21.Interface użytkownika ostatnie aukcje

Natomiast po naciśnięciu „View Auction Details” zostanie pokazany następujący interface:



22. Detale aukcji

Możemy przeglądać oferty, zdjęcia oraz stan produktu. Bardzo klarownie widać że potrzebne jest zalogowanie się, żeby licytować dany produkt.

W celu zarejestrowania oraz zalogowania są dwa oddzielne przyciski które wywołują formularze dla rejestracji oraz logowania:

- Rejestracja użytkownika

Register

×

Name

Surname

Email address

We'll never share your email with anyone else.

Password

Confirm Password

Select User Type

User

Auctioneer

Submit

23. Rejestracja użytkowników

- Wpisanie podstawowych informacji o użytkowniku, oraz wybranie konkretnej roli (radio button)

- Logowanie użytkownika

Login

×

Email address

We'll never share your email with anyone else.

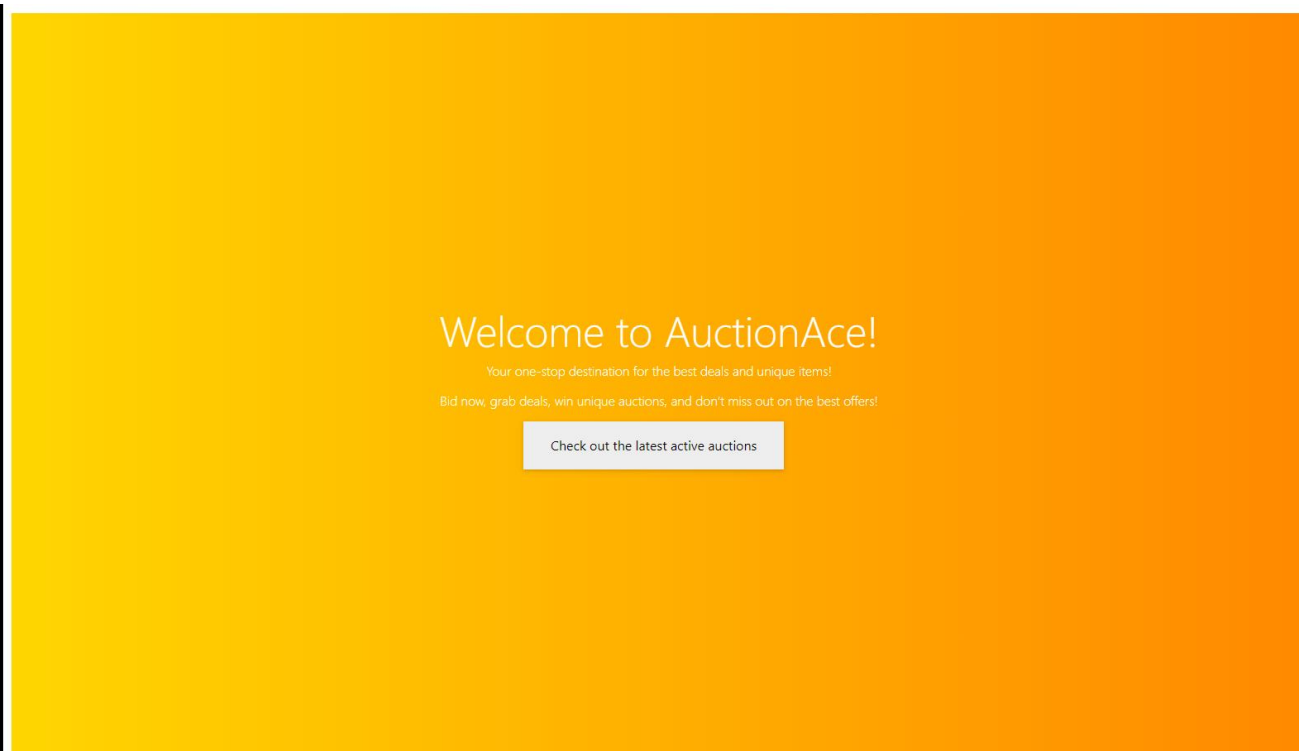
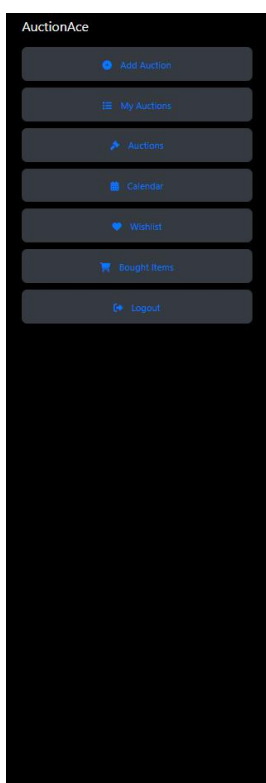
Password

Submit

Don't have an account? [Register here.](#)

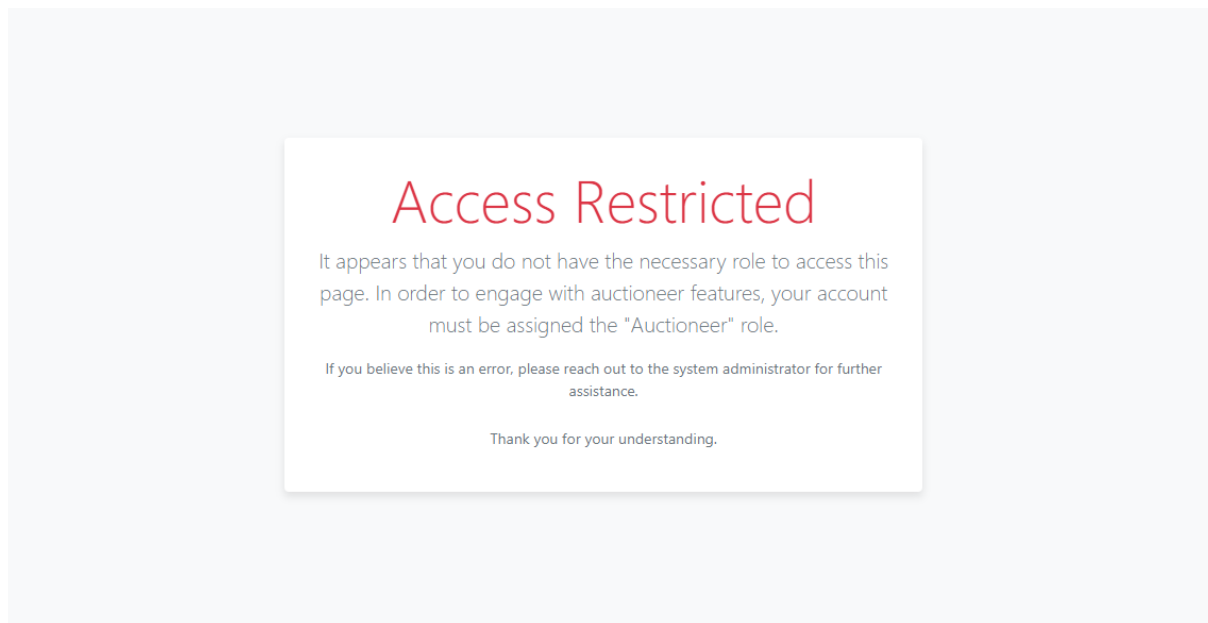
24. Logowanie użytkownika

Po zalogowaniu lub poprawnym zarejestrowaniu konta w serwisie, użytkownik powinien ujrzeć główny interfejs roboczy, który oferuje wszystkie funkcjonalności serwisu aukcyjnego. Wygląda on bardzo podobnie jak interfejs główny dla nie zalogowanego użytkownika, z tą różnicą że można tutaj w pełni korzystać z funkcjonalności znajdujących się w AuctionAce.



25. Widok zalogowanego użytkownika

Użytkownik który nie ma roli Aukcjonera, po wejściu w „Add Auction” zobaczy komunikat o nie autoryzowanym dostępie, dzięki któremu może dojść do wniosku żeby zaktualizować konto do wyższych uprawnień.



26. Nieautoryzowany dostęp

Po naciśnięciu przycisku „Add Auction” wraz z odpowiednią rolą, system aukcji pokaże interface dzięki któremu możemy zakładać aukcje, kategoryzować, ustalać daty otwarcia oraz zakończenia.

AuctionAce

Add Auction

My Auctions

Auctions

Calendar

Wishlist

Bought Items

Logout

Auction Form

Auction Name

Category

Select a category

Description

Start Date

dd.mm.rrrr --:--

End Date

dd.mm.rrrr --:--

Auction Photo

Choose file

Browse

Item List

Name	Description	Start Price	Buy Now Price	Condition	Photo
<div><div>Add item</div><div>Save</div></div>					

27. Dodanie aukcji Interface

Żeby dodać przedmiot do aukcji, użytkownik musi wypełnić wymagane pola w odpowiednim oknie modalnym, do którego dostaje się naciskając przycisk „Add item”. Aukcjoner może dodać wiele zdjęć przedmiotu który wystawia, określić cenę wywoławczą, cenę kup teraz oraz stan przedmiotu jako nowy lub używany. Po poprawnym dodaniu przedmiotu, użytkownik może podejrzeć we wcześniejszym interfacie, wszystkie dodane przedmioty.

Add item

Name

Item Description

Item Photo

Start Price

Buy Now Price

State

Used

New

Save

28.Dodanie przedmiotu Interface

Najważniejszym aspektem systemu, jest możliwość licytowania oraz swobodnego przeglądania licytowanych przedmiotów (zdjęć, cen wywoławczych oraz startowych). Intuicyjne podejście do przedstawiania danych, w przyjazny sposób dla ludzkiego oka, sprawia że użytkownik nie czuje zmęczenia podczas przeglądania aukcji. Widok zapewnia

wszelkie dane, tj. status aukcji, czas otwarcia w przypadku gdy aukcja oczekuje na date otwarcia, czas pozostały do zakończenia, opis oraz tytuł aukcji. Zastosowane zostały sortowania ze względu na kategorie oraz na status aukcji. Można również dodawać licytacje do swoich ulubiony, dla których jest specjalna zakładka „Wishlist”. Czerwone serduszka w lewym górnym rogu, symbolizują ze licytacja została oznaczona jako ulubiona, w przeciwnym wypadku są wyszarzone.

Show Active

Show Inactive

Show Pending

Show All

Categories

Pending

Bike Auction

Bid on a wide range of bicycles, from road bikes to mountain bikes. Whether you're a casual rider or a cycling enthusiast, find your perfect ride here...

Auction starts in: 0d 10h 55m 1s

View Auction Details

Pending

Car Auction

Discover amazing deals on cars of all makes and models. From sedans to SUVs, find your next vehicle at unbeatable prices.

Auction starts in: 1d 10h 58m 1s

View Auction Details

Active

Dogs Auction

Adopt your new best friend at our dog auction! Find various breeds, ages, and temperaments to bring home the perfect companion.

Time left: 13d 11h 2m 1s

View Auction Details

Active

Guns Auction

Explore a variety of firearms available for bidding. From antique collectibles to modern firearms, find what suits your needs.

Time left: 5d 11h 5m 1s

View Auction Details

Active

Hardware Auction

Bid on high-quality tools, construction equipment, and hardware supplies. Perfect for contractors, DIY enthusiasts, and professionals alike.

Time left: 8d 11h 7m 1s

View Auction Details

Inactive

Houses Auction

Find your dream home at a fraction of the price. Browse our selection of properties, from cozy cottages to spacious family homes.

View Auction Details

29.Widok kategorii, statusu aukcji

Użytkownik zalogowany może podpatrzeć każdą aukcję, nawet te które zostały zamknięte. Po wejściu do określonej licytacji, w przejrzysty sposób pokazane są wszelkie dane, zdjęcia potrzebne do podjecia decyzji, czy licytować.

Guns Auction

Explore a variety of firearms available for bidding. From antique collectibles to modern firearms, find what suits your needs.

Item Name	Description	Start Price	Buy Now Price	Condition	Photo	Join bidding
Shadow Sniper Rifle	Precision long-range sniper rifle replica with a sleek design, adjustable scope, and realistic bolt-action mechanism.	5000 USD	25000 USD	Used		Join
AK-47 Classic	Iconic AK-47 replica with a wooden finish, authentic details, and a detachable magazine for a true collector's experience.	5000 USD	50000 USD	New		Join
M249 Saw Replica	Heavy-duty M249 light machine gun replica with a realistic belt-fed design, folding bipod, and durable construction.	5000 USD	10000 USD	Used		Closed

30.Interface przed licytacją

Po wejściu do pokoju aukcji, użytkownik widzi szczegółowy interface związany z licytacją. Na górze ekranu wyświetlane jest zdjęcie przedmiotu wystawionego na aukcji wraz z nawigacją do przeglądania zdjęć. Po prawej stronie znajduje się tabela z listą aktualnych ofert, gdzie wyświetlane są dane użytkownika (Nick) w postaci email oraz cena, którą zaoferował.

Pod zdjęciem widoczna jest aktualna cena przedmiotu, oznaczona dużą czcionką. Tuż obok znajdują się dwa główne przyciski „Current Price” oraz „Bid”, jeden z nich jest zablokowany i wskazuje aktualną najwyższą cenę produktu, drugi odpowiada za przebijanie oferty. Chęć wysłania oferty niższej niż aktualnie najwyższa, skńczy się błędem w postaci alertu, który podpowiada użytkownikowi żeby rozważył zmianę ceny. W środkowej części ekranu znajduje się licznik odmierzający pozostały czas do zakończenia aukcji (w formacie dni, godzin, minut i sekund). W dolnej części ekranu użytkownik może wymieniać wiadomości z innymi uczestnikami aukcji za pomocą wbudowanego czatu. Przesyłanie wiadomości są opatrzone znacznikiem czasu i adresem e-mail użytkownika. Pod polem wpisywania wiadomości znajduje się przycisk „Send” do wysyłania komunikatów. Można również wyjść z aukcji poprzez naciśnięcie „Quit” oraz kupić aukcję natychmiast „Buy Now” za określoną cenę, ta operacja natychmiast zamyka licytację przedmiotu. Właściciel aukcji ma zablokowane pola „Buy Now” oraz „Bid”.



Quit

Buy Now : 1000 USD

7d 10h 18m 2s

#	Nick	Price
1	michaltulej1@gmail.com	255 USD
2	test.user@auctionace.pl	250 USD

255

Current Price

255

Bid

(12.18 22:53): test.user@auctionace.pl: Hello!
(12.18 22:54): test.auctioner@auctionace.pl: Hey what's up bro?
(12.18 22:54): test.auctioner@auctionace.pl: this auction is so awesome
(12.18 22:54): test.user@auctionace.pl: yeah, thats true, and so cheap items
(12.18 22:55): michaltulej1@gmail.com: i will win this auction :)

Type message...

Send

Kalendarz aukcji umożliwia użytkownikowi, pełniącemu rolę aukcjonera, podgląd wszystkich zaplanowanych aukcji w formie przejrzystego terminarza. Użytkownik może przełączać się pomiędzy różnymi widokami: miesięcznym, tygodniowym i dziennym za pomocą opcji widocznych u góry kalendarza. Widok miesięczny prezentuje wszystkie aukcje zaplanowane na konkretny dzień, które wyświetlane są w formie kolorowych pasków. Zielony kolor oznacza aktywne aukcje, a szary może reprezentuje aukcje które dopiero mają się rozpocząć. Czerwony natomiast, te które zostały już zamknięte. Każda z licytacji zawiera podstawowe dane tj. godzina rozpoczęcia np. „9:31a” (a oznacza AM) oraz nazwe aukcji, jak „House Auction”. Dzięki nawigacji umieszczonej w prawym górnym rogu kalendarza, użytkownik może przemieszczać się pomiędzy kolejnymi miesiącami, używając strzałek do przewijania. Kalendarz pozwala na szybki podgląd nadchodzących wydarzeń, organizacje harmonogramu oraz śledzenie aktywnych aukcji w jednym miejscu, co znacznie ułatwia planowanie i zarządzanie czasem aukcjonera. Na przykład , 3 grudnia 2024 o godzinie 9:31 zaplanowano „House Auction” a 14 grudnia 2024 roku o godzinie 9:22 widoczna jest „Dogs Auction”. Dzięki tej funkcjonalności aukcjoner może skutecznie kontrolować wszystkie zaplanowane wydarzenia oraz lepiej organizować swoją pracę.

monthweekday

December 2024

<<<>>>

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
		9:31a Houses Auction				
8	9	10	11	12	13	14
9:31a Houses Auction						9:22a Dogs Auction
15	16	17	18	19	20	21
9:31a Houses Auction						
9:22a Dogs Auction						
	9:26a Guns Auction					
	9:28a Hardware Auction					
		9:16a Bike Auction		9:19a Car Auction		
22	23	24	25	26	27	28
9:31a Houses Auction						
9:22a Dogs Auction						
9:26a Guns Auction						
9:28a Hardware Auction						
9:16a Bike Auction						
9:19a Car Auction						

Zakładka „Wishlist” umożliwia użytkownikowi przeglądanie aukcji, które zostały dodane do listy ulubionych. Dzięki tej funkcji użytkownik może w łatwy sposób śledzić interesujące go przedmioty, którymi jest zainteresowany. W widoku tej zakładki jest lista kart z ulubionymi aukcjami. Każda oferta zawiera tytuł aukcji, krótki opis oferty, daty rozpoczęcia i zakończenia aukcji oraz kategorię do której należy dana aukcja. W dolnej części karty znajduje się przycisk „View Auction”, który po kliknięciu przekierowuje użytkownika do szczegółowych informacji o wybranej aukcji. Zakładka dostępna jest z menu bocznego pod ikoną serca, a karty aukcji są w czytelnym układzie siatki, co ułatwia przeglądanie i zarządzanie ulubionymi aukcjami.

Car Auction

Discover amazing deals on cars of all makes and models. From sedans to SUVs, find your next vehicle at unbeatable prices.

Start Date: 19.12.2024 09:19:00

End Date: 25.12.2024 09:19:00

Category: Cars

View Auction

Dogs Auction

Adopt your new best friend at our dog auction! Find various breeds, ages, and temperaments to bring home the perfect companion.

Start Date: 14.12.2024 09:22:00

End Date: 31.12.2024 09:23:00

Category: Dogs

View Auction

Bike Auction

Bid on a wide range of bicycles, from road bikes to mountain bikes. Whether you're a casual rider or a cycling enthusiast, find your perfect ride here!

Start Date: 18.12.2024 09:16:00


End Date: 25.12.2024 09:16:00

Category: Bike, Motorbike

View Auction

33.Widok ulubionych aukcji użytkownika

Wszystkie kupione przedmioty użytkowników systemu aukcyjnego, znajdują się w odpowiedniej zakładce oznaczonej „Bought Items”. Każda karta wyświetla podstawowe informacje o zakupionym przedmiocie takie jak cene za którą został przedmiot wylosowany, nazwa, stan oraz opis.




AK-47 Classic

Price: 50000 USD

Condition: new

Iconic AK-47 replica with a wooden finish, authentic details, and a detachable magazine for a true collector's experience.




Epson EcoTank L3250

Price: 1000 USD

Condition: used

Wireless all-in-one inkjet printer with high-capacity ink tanks, mobile printing, and cost-effective refillable design.



French Bulldog Bella

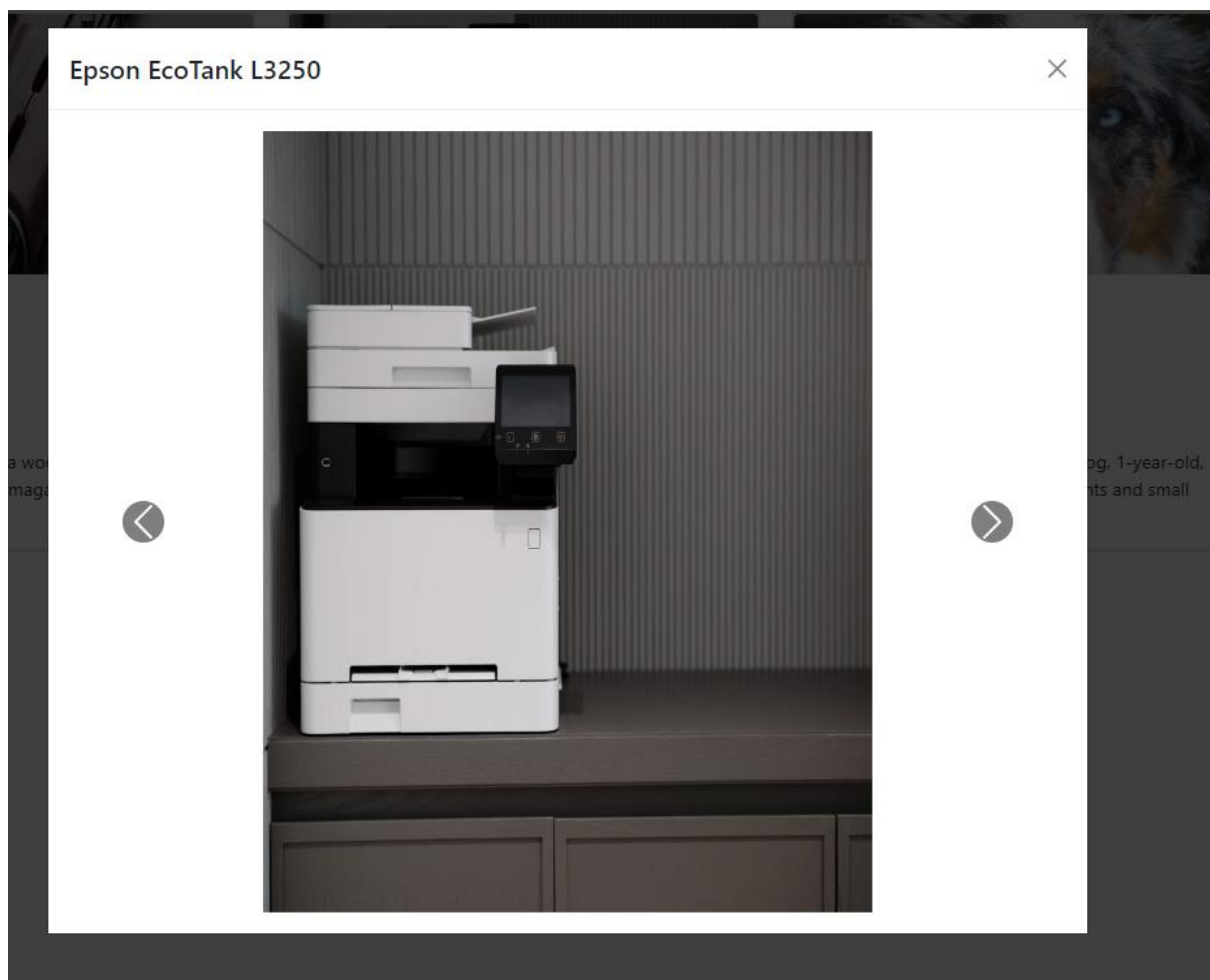
Price: 160000 USD

Condition: new

Adorable and playful French Bulldog, 1-year-old, compact size, perfect for apartments and small spaces.

34.Kupione aukcje użytkownika

Również w tym miejscu są dostępne zdjęcia kupionych przedmiotów. Użytkownicy mogą je przeglądać w formie karuzeli, która pozwala na łatwe przełączanie się między poszczególnymi zdjęciami za pomocą strzałek lub automatycznego przewijania. Dzięki temu, każdy zakupiony przedmiot jest przedstawiany w pełnej okazałości, umożliwiając szybki podgląd wszystkich dostępnych fotografii.



35.Przykład przeglądania zdjęć rzeczy kupionych

Dzięki tej funkcjonalności użytkownicy mogą dokładnie obejrzeć szczegóły przedmiotów, co ułatwia ocenę zakupu. Możliwość przeglądania zdjęć w wysokiej jakości zapewnia lepszą prezentację przedmiotów i poprawia komfort korzystania z aplikacji.

4. Wyniki i dyskusja

Stworzenie pełnoprawnego serwisu obsługującego aukcje poszła zgodnie z założeniami. W ostatecznej wersji, nie występują żadne opóźnienia. Wszelkie interakcje użytkownika ze stroną, przebiegają dynamicznie oraz sprawnie. System umożliwia płynną obsługę kluczowych funkcji, takich jak dodawanie przedmiotów, składanie ofert oraz aktualizowanie statusów w czasie rzeczywistym. Dzięki zastosowaniu technologii asynchronicznych i dynamicznego odświeżania treści, użytkownicy mogą liczyć na intuicyjną i responsywną obsługę. W efekcie serwis spełnia założenia projektu i zapewnia użytkownikom pozytywne doświadczenia podczas licytacji.

Opracowany system aukcyjny „AuctionAce” spełnia podstawowe role jak i te bardziej złożone. Niemniej istnieje wiele dróg w które mógłby zostać rozwinięty, aby stać się bardziej funkcjonalny i konkurencyjny na rynku. Poniżej znajdują się najważniejsze z nich:

- Implementacja modułu z obsługą płatności

Użytkownicy po wygraniu licytacji, mieliby możliwość opłacenia przedmiotu. System pobierał by 5% prowizji od każdej zaistniałej transakcji. Istniały by różne metody płatności tj:

- Płatności online (karty płatnicze)
Transakcje które realizowane są poprzez karty debetowe lub kredytowe. Nadzór nad takimi transakcjami utrzymują Visa, Mastercard, American Express.
- Płatności cyfrowe
Szybkie płatności z wykorzystaniem telefonu Google Pay/Apple Pay.
- Płatności kryptowalutowe
Anonimowe płatności które, swoje bezpieczeństwo oraz nadzór utrzymują poprzez platformę Blockchain.
- Tradycyjne przelewy bankowe
- Portfele cyfrowe
Linki do wymiany transakcji poprzez portfele zakładane na popularnych serwisach tj. Revolut, Skrill, Payoneer.
- Płatności za pobraniem
- Wewnętrzna waluta
Tokeny za które można licytować aukcje, oraz je wypłacać.

- Dodanie kantoru

Aukcje przebiegały by w określonych walutach, w zależności od narodowości sprzedającego lub po prostu do wyboru sprzedającego. Użytkownicy w tej zakładce mieliby możliwość konwersji swoich pieniędzy, po kursach pobranych z API Narodowego Banku Polskiego. Automatycznie waluty by pojawiały się na kontach.

- Dodanie powiadomień push o przebiegu, zakończeniu, rozpoczęciu obserwowanej aukcji

W przypadku jakiegokolwiek interakcji z aukcją, powiadomienia w postaci dymków wyskakiwały by w obszarze roboczym użytkownika.

- Powiadomienia email na adres użytkownika o wygraniu aukcji, dodaniu aukcji, kończącej się aukcji

Wykorzystanie serwera SMTP, dzięki któremu automatyczne emaile z określonymi wiadomościami, trafiałyby na email użytkownika podany podczas rejestracji.

- Opinie odnośnie użytkowników wystawiających aukcje

System opinii i recenzji który odnosiłby się do Aukcjonerów. Opinie wystawiały by tylko osoby które kupiły przedmioty. Każdy z użytkowników mógłby sprawdzić przed licytowaniem wiarygodność sprzedawcy.

- Tematyczny blog dyskusyjny/helpdesk

Budowanie społeczności która mogłaby podzielić się swoimi zakupami oraz problemami z szerszym gronem odbiorców. Zawierały by się tam również pomoce odnośnie strony technicznej na stronie.

- Subskrypcja Premium

Użytkownicy którzy wykupili premium za określoną kwotę, mają możliwość tworzenia aukcji premium, które były by znacznie lepiej traktowane przez algorytm pozycjonowania aukcji. Specjalna otoczka która by znacząco wyróżniała daną aukcję.

5. Spis ilustracji

1.Diagram przypadków użycia.....	6
2.Scenariusz przypadku użycia " Dodanie Aukcji".....	8
3.Scenariusz przypadku użycia " Licytowanie Aukcji"	10
4.Scenariusz przypadku użycia "Logowanie użytkownika"	11
5.Architektura Aplikacji.....	13
6.Architektura Auction Ace (schemat).....	14
7.Przykład reużywalności kodu.....	15
8.Warstwa Prezentacji	16
9.Warstwa Aplikacji.....	16
10.Warstwa Domenowa	17
11.Warstwa Infrastruktury	17
12.Diagram encji	23
13.Przykład nazewnictwa testów	26
14.Przykładowa metoda z testów	26
15.Opis przepływu testów zdjęcie 1	27
16.Opis przepływu testów zdjęcie 2.....	27
17.Opis przepływu testów zdjęcie 3.....	27
18.Opis przepływu testów zdjęcie 4.....	28
19.Opis przepływu testów zdjęcie 5.....	28
20.Interface użytkownika	29
21.Interface użytkownika ostatnie aukcje	29
22.Detale aukcji.....	30
23.Rejestracja użytkowników	30
24.Logowanie użytkownika	31
25.Widok zalogowanego użytkownika	31
26.Nieautoryzowany dostęp	32
27.Dodanie aukcji Interface	32
28.Dodanie przedmiotu Interface	33
29.Widok kategorii, statusu aukcji	34
30.Interface przed licytacją	34
31.Widok licytacji	35
32.Kalendarz Użytkownika.....	36

33. Widok ulubionych aukcji użytkownika.....	37
34. Kupione aukcje użytkownika	37
35. Przykład przeglądania zdjęć rzeczy kupionych.....	38

6. Bibliografia

- .Net Framework - dokumentacja
<https://learn.microsoft.com/en-us/dotnet/framework/>
- C# - dokumentacja
<https://learn.microsoft.com/en-us/dotnet/csharp/>
- SQL Server Management Studio – dokumentacja
<https://learn.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16>
- Książka „Czysta architektura. Struktura i design oprogramowania. Przewodnik dla profesjonalistów” Martin Robert C. rok 2022
- HTML5
https://www.w3schools.com/Html/html_intro.asp
- Bootstrap 5 – dokumentacja
<https://getbootstrap.com/docs/5.0/getting-started/introduction/>
- JavaScript – dokumentacja
<https://developer.mozilla.org/en-US/docs/Web/JavaScript>
- SignalR – dokumentacja
<https://learn.microsoft.com/en-us/aspnet/signalr/>
- JWT Token
<https://learn.microsoft.com/en-us/aspnet/core/security/authentication/jwt-authn?view=aspnetcore-9.0&tabs=windows>
<https://jwt.io/>
- Zintegrowany język LINQ
<https://learn.microsoft.com/en-us/dotnet/csharp/linq/>