**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ  
імені ІГОРЯ СІКОРСЬКОГО»**

**Факультет інформатики та обчислювальної техніки**

**Кафедра інформаційних систем та технологій**

**Курсова робота**

**з дисципліни «Програмування»**

**на тему: «Система проведення тендерів»**

Виконав:

студент 1 курсу, групи ІА-33

Мартинюк Юрій Романович\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_

(підпис)

Керівник:

асистент кафедри ІСТ

Мягкий Михайло Юрійович \_\_\_\_\_\_\_\_\_\_

(підпис)

Засвідчую, що у цій курсовій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент \_\_\_\_\_\_\_\_\_\_

(підпис)

Київ – 2024 року

ЗМІСТ

[ВСТУП 3](#_Toc168381833)

[1 ВИМОГИ ДО СИСТЕМИ 5](#_Toc168381834)

[1.1 Функціональні вимоги до системи 5](#_Toc168381835)

[1.2 Нефункціональні вимоги до системи 6](#_Toc168381836)

[2 СЦЕНАРІЇ ВИКОРИСТАННЯ СИСТЕМИ 7](#_Toc168381837)

[2.2 Опис сценаріїв використання системи 8](#_Toc168381838)

[3 АРХІТЕКТУРА СИСТЕМИ 19](#_Toc168381839)

[4 РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ 22](#_Toc168381840)

[4.1 Загальна структура проекту 22](#_Toc168381841)

[4.2 Компоненти рівня доступу до даних 25](#_Toc168381842)

[4.3 Компоненти рівня бізнес-логіки 30](#_Toc168381843)

[4.4 Компоненти рівня інтерфейсу користувача 60](#_Toc168381844)

[ВИСНОВКИ 109](#_Toc168381845)

[ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 111](#_Toc168381846)

[ДОДАТОК Б Вигляд зі сторони користувача 113](#_Toc168381847)

# ВСТУП

У сучасному світі, де ефективність і прозорість бізнес-процесів мають першорядне значення, система проведення тендерів забезпечує справедливу конкуренцію та ефективне використання ресурсів. Система, така як конкурентні торги, дозволяє компаніям або організаціям визначати найбільш відповідних постачальників через об'єктивну та методичну процедуру відбору. Це завдання полягає в тому, щоб зрозуміти, як працюють різні сутності тендерної системи, розглядаючи приклади з участю різних акторів, а саме: Користувача, Тендерної пропозиції, Тендеру тощо.

Тендер є формальним запрошенням, яке запитує пропозиції на постачання певних товарів або послуг. Це зазвичай здійснюється будь-якою організацією, яка потребує таких предметів, щоб потенційні постачальники могли представити свої пропозиції. Отже, це створює основу для конкурентного відбору, надаючи рівні шанси всім зацікавленим сторонам. Крім того, цей процес спрямований на створення конкурентних умов, що дозволяють організації отримати найкращі ціни, а також якісні послуги або товари. Тендерна пропозиція становить відповідь постачальника, що містить повну інформацію про умови та ціни. Ці два елементи дуже важливі, оскільки вони сприяють прозорості та ефективності під час закупівель.

Користувач і Хазяїн тендеру, які є гравцями системи, виконують різні ролі з відповідними правами та обов'язками. Користувачі можуть переглядати активні тендери, шукати їх за ключовими словами, створювати тендерні пропозиції або ставати хазяїнами тендерів. Такі особи здебільшого представляють компанії або організації, які шукають можливості для ведення бізнесу, отримуючи нові контракти після перемоги в торгах. З іншого боку, Хазяїн тендеру може видаляти, переглядати, генерувати URL, запускати, зупиняти, переглядати пропозиції та інше. Тому він/вона не лише керує, а й забезпечує відповідність усіх учасників встановленим вимогам та критеріям відбору.

Ця робота розгляне основні аспекти та труднощі, з якими стикаються учасники та власники тендерів, а також запропонує потенційні шляхи підвищення ефективності та автоматизації процесу проведення тендерів. Зокрема, буде розглянуто питання забезпечення рівних можливостей у бізнес-транзакціях; мінімізації ймовірності корупції при укладанні контрактів; підвищення ефективності у виборі постачальників або підрядників та заощадження витрат. Метою цього дослідження є не лише аналітичний підхід, але й рекомендації заходів, які б підвищили рівень продуктивності, водночас забезпечуючи відкритість під час тендерних процесів.

Основна мета цього проекту – глибоко зрозуміти, як працюють тендерні системи, їх структури та операції, що дозволить оцінити, чи є вони корисними для загальної ефективності організацій. Дослідження має намір запропонувати нові ідеї для вирішення проблем, які виникають у сучасних тендерних системах, краще їх зрозумівши. Досліджуючи різні сценарії, де взаємодіють різні користувачі, я сподіваюся встановити місця, де потрібно внести зміни, щоб наші тендери стали ефективнішими та прозорішими.

Крім того, дослідження також заглибиться в правові та нормативні рамки, що стосуються тендерів, зосереджуючись на дотриманні законів та інших нормативних вимог. Це передбачає аналіз місцевих законів, що регулюють закупівлі, разом із міжнародними стандартами, які керують такими процесами. Додатково буде оцінено досвід різних країн щодо успішного впровадження ефективних тендерних систем, щоб запозичити їхній досвід, зробивши його найкращим для нашого контексту, а також навчитися на їхніх помилках для майбутнього успіху.

Таким чином, курсова робота ставить амбітну мету: не лише дослідити існуючу систему проведення тендерів, але й запропонувати конкретні заходи для її вдосконалення. Це включає рекомендації щодо використання сучасних технологій, таких як блокчейн та штучний інтелект, для підвищення прозорості та ефективності тендерних процесів. Відтак, результати цього дослідження можуть бути корисними не лише для академічного середовища, але й для практичного застосування в бізнесі та державному управлінні.

# 1 ВИМОГИ ДО СИСТЕМИ

## 1.1 Функціональні вимоги до системи

Функціонал системи проведення тендерів повинен бути задовільним для такого переліку дій:

– користувач, який не є зареєстрованим (гість, або відвідувач) повинен мати можливість для перегляду активних тендерів, інформацію про ці тендери та тендерні пропозиції для приблизного оцінювання потреб хазяїна тендеру;

– незареєстрований користувач повинен мати можливість пошуку тендерів за ключовими словами, які вводить цей користувач;

– незареєстрований користувач повинен мати можливість створити власний акаунт для розширення функціоналу, який надає цей сервіс;

– зареєстрований користувач (гість, який створив власний акаунт) повинен мати повний перелік функціоналу, який надається незареєстрованому користувачу, та розширювати його функціонал;

– зареєстрований користувач повинен мати можливість створити власний тендер та стати його хазяїном;

– хазяїн тендеру повинен мати можливість редагування, активації/деактивації, видалення власного тендеру без перешкод, які теоретично можуть виникнути;

– хазяїн тендеру повинен мати можливість власноруч обрати тендерну пропозицію після закінчення дії тендеру, яка підкріплена під цей тендер та в подальшому стане переможцем (виконавцем) цього тендеру;

– зареєстрований користувач повинен мати можливість написати відгук (перелік власних компетенцій) у відповідь на будь-який тендер з подальшим зворотнім зв’язком;

– зареєстрований користувач повинен мати можливість створення тендерної пропозиції до будь-якого тендеру та стати хазяїном пропозиції;

– хазяїн тендерної пропозиції повинен мати можливість редагування та видалення цієї тендерної пропозиції без перешкод;

– між хазяїном тендеру та переможцем цього тендеру повинен бути зворотній зв’язок у вигляді чату, де обидві сторони матимуть можливість обговорити питання стосовно виконання.

## 1.2 Нефункціональні вимоги до системи

Система проведення тендерів повинна відповідати таким нефункціональним вимогам:

– система повинна мати відкриту архітектуру;

– система повинна мати графічний інтерфейс для зручного використання користувачем;

– інтерфейс користувача повинен бути легким у використанні, візуально приємним та мати інтуїтивно зрозумілий функціонал;

– система повинна задовільняти умовам безпеки для користувача, наприклад, хешування паролів, перевірку на те, чи є цей користувач хазяїном тендеру/тендерної пропозиції при редагування чи видаленні, генерації унікальних ідентифікаторів для кожного створеного тендеру, пропозиції та відгуку;

– система повинна бути продуктивною.

# 2 СЦЕНАРІЇ ВИКОРИСТАННЯ СИСТЕМИ

2.1 Діаграма прецедентів

Діаграма прецедентів системи представлена на рис. 2.1.

Акторами даної системи проведення тендерів є користувачі системи: незареєстрований (гість) та зареєстрований користувачі.

Зареєстрованому користувачу доступна уся функціональність, що і незареєстрованому, а також у зареєстрованого користувача повинна бути розширена функціональність: створення/редагування/активація/видалення тендерів (редагування та інше повинно бути доступним за умови, якщо користувач є хазяїном тендеру), створення/видалення тендерних пропозиції, написання відгуків (переліку компетентності) та вибір хазяїном тендеру переможця власного тендеру.

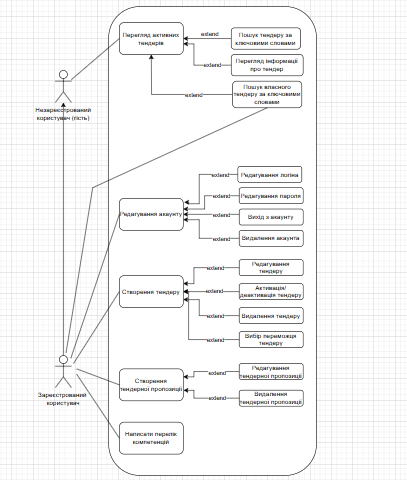


Рисунок 2.1 – Діаграма прецедентів для системи проведення тендерів

## 2.2 Опис сценаріїв використання системи

Детальні описи сценаріїв використання наведено у таблицях 2.1 – 2.10.

Таблиця 2.1 – Сценарій використання «Пошук активних тендерів за ключовими словами»

|  |  |
| --- | --- |
| Назва | Пошук активних тендерів за ключовими словами |
| ID | 1 |
| Опис | Будь-який користувач має можливість власноруч знайти певний тендер, використовуючи ключові слова |
| Актори | Зареєстрований користувач, незареєстрований користувач |
| Вигоди компанії | Пошук тендерів за власними словами надає користувачу можливість знайти тендер з потрібною тематикою, а отже, підвищуються шанси на створення тендерної пропозиції та використання даного сервісу, а це – прибуток для компанії |
| Частота користування | Постійно |
| Тригери | Користувач вводить ключові слова у полі для пошуку |
| Передумови | Пошукове поле доступне на головній сторінці системи |
| Постумови | Користувач потрапляє на сторінку з переліком тендерів, які відповідають даним ключовим словам |
| Основний розвиток | Користувач вводить ключові слова, натискає кнопку «Пошук» біля поля для введення |
| Альтернативні розвитки | – |
| Виняткові ситуації | У разі відсутності тендерів з такими ключовими словами, користувач отримує повідомлення про відсутність тендерів з такими ключовими словами |

В таблиці 2.2 представлений сценарій використання «Перегляд детальної інформації по тендеру»

Таблиця 2.2 – Сценарій використання «Перегляд детальної інформації по тендеру»

|  |  |
| --- | --- |
| Назва | Перегляд детальної інформації по тендеру |
| ID | 2 |
| Опис | Користувач обирає тендер та переглядає детальну інформацію |
| Актори | Зареєстрований користувач, незареєстрований користувач |
| Вигоди компанії | Знання деталей тендеру дає користувачу ширші розуміння суті цього тендеру, а це – утримання клієнтів на цьому сервісі. |
| Частота користування | Постійно |
| Тригери | Користувач натискає на кнопку «Деталі тендеру», яка розташована під кожним тендером |
| Передумови | На головній сторінці або на сторінці знайдених тендерів є список активних тендерів, які можна переглянути |
| Постумови | Користувач бачить детальну інформацію про обраний тендер |
| Основний розвиток | Користувач натискає на кнопку, перед ним з’являється детальна інформація про тендер |
| Альтернативні розвитки | – |
| Виняткові ситуації | – |

В таблиці 2.3 представлений сценарій використання «Перегляд головної сторінки із активними тендерами»

Таблиця 2.3 – Сценарій використання «Перегляд головної сторінки із активними тендерами»

|  |  |
| --- | --- |
| Назва | Перегляд головної сторінки із активними тендерами |
| ID | 3 |
| Опис | Користувач переглядає головну сторінку із активними тендерами |
| Актори | Зареєстрований користувач, незареєстрований користувач |
| Вигоди компанії | Головна сторінка важлива для зацікавлення нових користувачів, та актуальна інформація буде утримувати старих |
| Частота користування | Постійно |
| Тригери | Користувач переходить на сайт/запускає додаток |
| Передумови | Немає |
| Постумови | Користувач потрапляє на головну сторінку |
| Основний розвиток | Користувач запускає додаток/переходить на сайт |
| Альтернативні розвитки | – |
| Виняткові ситуації | – |

В таблиці 2.4 представлений сценарій використання «Створення/редагування/видалення тендеру»

Таблиця 2.4 – Сценарій використання «Створення/редагування/видалення тендеру»

|  |  |
| --- | --- |
| Назва | Створення/редагування/видалення тендеру |
| ID | 4 |
| Опис | Користувач має можливість створити власний тендер та стати його хазяїном, після створення користувач може рдагувати або видалити тендер |
| Актори | Зареєстрований користувач, хазяїн тендеру |
| Вигоди компанії | Створення тендерів – це суть цієї системи, без неї вона втратить актуальність, а отже, її відсутність не вигідна компанії |
| Частота користування | Часто (коли користувач захоче створити/редагувати/видалити тендер) |
| Тригери | Користувач натискає на відповідні кнопки на головній сторінці та на сторінці користувача |
| Передумови | Користувач знаходиться на головній сторінці або сторінці користувача |
| Постумови | Користувач потрапляє на сторінку для створення/редагування тендеру |
| Основний розвиток | Користувач вводить інформації про власний тендер, після створення потрапляє на сторінку з деталями тендеру. При редагуванні/видаленні все те саме, тільки користувач потрапляє на головну сторінку |
| Альтернативні розвитки | Користувач вирішив не створювати/редагувати тендер, тому переходить на головну сторінку або сторінку користувача |
| Виняткові ситуації | При введенні не того типу даних, користувач отримує повідомлення «Введіть правильний тип даних». Якщо ім’я тендеру уже наявне в базі даних, то користувач отримує відповідне повідомлення |

В таблиці 2.5 представлений сценарій використання «Запуск/зупинка тендеру»

Таблиця 2.5 – Сценарій використання «Запуск/зупинка тендеру»

|  |  |
| --- | --- |
| Назва | Запуск/зупинка тендеру |
| ID | 5 |
| Опис | Хазяїн тендеру має можливість запустити або зупинити тендер |
| Актори | Хазяїн тендеру |
| Вигоди компанії | Можливість запуску та зупинки тендеру надає користувачам більше можливостей для управління, отже, користувачам краще використовувати цей сервіс, а це – прибуток компанії |
| Частота користування | Помірно часто |
| Тригери | Користувач на сторінці редагування натискає відповідну кнопку |
| Передумови | Користувач перебуває на сторінці редагування |
| Постумови | Користувачу виводиться інформація про тендер, а також те, чи є цей тендер активним |
| Основний розвиток | Користувач натискає відповідну кнопку і після цього потрапляє на сторінку з інформацією про тендер |
| Альтернативні розвитки | Вихід з цієї сторінки |
| Виняткові ситуації | – |

В таблиці 2.6 представлений сценарій використання «Створення/редагування/видалення тендерної пропозиції»

Таблиця 2.6 – Сценарій використання «Створення/редагування/видалення тендерної пропозиції»

|  |  |
| --- | --- |
| Назва | Створення/редагування/видалення тендерної пропозиції |
| ID | 6 |
| Опис | Користувач має можливість створити пропозиції до певного тендеру, після – редагувати/видалити її |
| Актори | Зареєстрований користувач, автор пропозиції |
| Вигоди компанії | Це також суть цієї системи, без неї – система не є актуальною, тому за її відсутності у компанії не буде прибутків |
| Частота користування | Користувачі зазвичай помірно створюють пропозиції |
| Тригери | Користувач натискає кнопку «Створити тендерну пропозицію» або «Редагувати пропозицію» |
| Передумови | Користувач знаходиться на головній сторінці, де під кожним тендером є кнопка для створення пропозиції. Для редагування – на сторінці користувача |
| Постумови | Користувач потрапляє на сторінку для створення пропозиції |
| Основний розвиток | Користувач створює/редагує/видаляє власну пропозицію |
| Альтернативні розвитки | Користувач передумав створювати/редагувати/видаляти пропозицію, тому може перейти на головну сторінку |
| Виняткові ситуації | При введення не того типу даних, користувач отримує повідомлення «Введіть правильний тип даних» |

В таблиці 2.7 представлений сценарій використання «Написання відгуку (переліку компетенцій) хазяїну тендеру»

Таблиця 2.7 – Сценарій використання «Написання відгуку (переліку компетенцій) хазяїну тендеру»

|  |  |
| --- | --- |
| Назва | Написання відгуку (переліку компетенцій) хазяїну тендеру |
| ID | 7 |
| Опис | Користувач має можливість (за потреби) написати перелік компетенцій хазяїну тендеру |
| Актори | Зареєстрований користувач |
| Вигоди компанії | Надавання зворотного зв’язку збільшує шанси на успішну угоду, а отже – прибуток для компанії |
| Частота користування | Не часто |
| Тригери | Користувач натискає на відповідну кнопку під описом тендеру |
| Передумови | Користувач знаходиться на головній сторінці або на сторінці з детальною інформацією про тендер |
| Постумови | Користувач потрапляє на сторінку для написання переліку компетенцій |
| Основний розвиток | Користувач натискає відповідну кнопку та створює відгук (перелік компетенцій) |
| Альтернативні розвитки | Користувач передумав писати відгук, тому він може перейти на головну сторінку |
| Виняткові ситуації | При введення не того типу даних, користувач отримує повідомлення «Введіть правильний тип даних» |

В таблиці 2.8 представлений сценарій використання «Вибір хазяїном тендеру переможця (виконавця) свого тендеру»

Таблиця 2.8 – Сценарій використання «Вибір хазяїном тендеру переможця (виконавця) свого тендеру»

|  |  |
| --- | --- |
| Назва | Вибір хазяїном тендеру переможця (виконавця) свого тендеру |
| ID | 8 |
| Опис | Після завершення часу тендеру, хазяїн цього тендеру може обрати переможця (виконавця) цього тендеру |
| Актори | Зареєстрований користувач (хазяїн тендеру) |
| Вигоди компанії | Вибір переможця тендерної пропозиції – важливий функціонал даної системи, тому без неї вона втрачає сенс, а отже – компанія не отримає прибутки |
| Частота користування | Один раз на один тендер |
| Тригери | Користувач натискає відповідну кнопку під своїм тендером на сторінці користувача |
| Передумови | Користувач (хазяїн тендеру) знаходиться на сторінці користувача |
| Постумови | Користувач (хазяїн тендеру) обирає переможця зі списку тендерних пропозиції, які відсортовані за прайсом |
| Основний розвиток | Користувач обирає переможця свого тендеру |
| Альтернативні розвитки | Користувач не хоче вибирати переможця з певних причин, пому він може вернутися назад |
| Виняткові ситуації | – |

В таблиці 2.9 представлений сценарій використання «Пошук власних тендерів за ключовими словами»

Таблиця 2.9 – Сценарій використання «Пошук власних тендерів за ключовими словами»

|  |  |
| --- | --- |
| Назва | Пошук власних тендерів за ключовими словами |
| ID | 9 |
| Опис | Користувач має можливість знайти власний тендер за ключовими словами |
| Актори | Зареєстрований користувач |
| Вигоди компанії | Пошук власних тендерів надає зручні умови для користувачів, а отже, більше шансів утримати користувача на цьому сервісі |
| Частота користування | Нечасто |
| Тригери | Користувач вводить ключові слова у поле для пошуку та натискає відповідну кнопку |
| Передумови | Користувач знаходиться на сторінці користувача |
| Постумови | Користувач отримує список власних тендерів з такими ключовими словами |
| Основний розвиток | Користувач вводить ключові слова у відповідному полі та натискає кнопку «Пошук» |
| Альтернативні розвитки | – |
| Виняткові ситуації | При відсутності тендерів з такими ключовими словами, користувач отримує відповідне повідомлення |

В таблиці 2.10 представлений сценарій використання «Зворотній зв’язок між хазяїном тендеру та виконавцем»

Таблиця 2.10 – Сценарій використання «Зворотній зв’язок між хазяїном тендеру та виконавцем»

|  |  |
| --- | --- |
| Назва | Зворотній зв’язок між хазяїном тендеру та виконавцем |
| ID | 10 |
| Опис | Хазяїн тендеру та переможець цього тендеру мають можливість для зворотного зв’язку один з одним |
| Актори | Хазяїн тендеру, переможець цього тендерцу |
| Вигоди компанії | Більше шансів на успішне виконання тендеру, а отже – утримання цих користувачів, у результаті – прибуток для компанії |
| Частота користування | Часто, враховуючи те, що це дія між вдома користувачами |
| Тригери | Користувач натискає відповідну кнопку під власним тендером, в якого уже визначено переможця, або тендерною пропозицією, яка є переможцем тендеру |
| Передумови | Користувач знаходиться на сторінці користувача |
| Постумови | Зв’язок між хазяїном тендеру та виконавцем |
| Основний розвиток | Користувач потрапляє в чат |
| Альтернативні розвитки | – |
| Виняткові ситуації | – |

# 3 АРХІТЕКТУРА СИСТЕМИ

У цій частині описана загальна структура системи проведення тендерів: шаблон, глобальна структура кожної частини роботи (вигляд, контролери, моделі).

Загальна архітектура системи наведена на рис. 3.1.

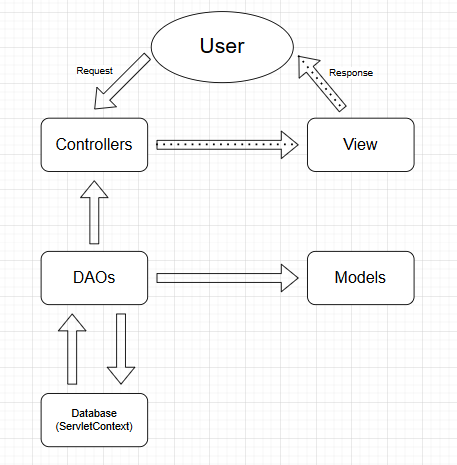


Рисунок 3.1 – Загальна архітектура системи

Система проведення тендерів складається з наступних елементів:

– графічний інтерфейс (view);

– серверна частина (controllers, DAOs);

– база даних (ServletContext).

Під час написання застосунку використовувався шаблон MVC – Model, View, Controller – це простий шаблон для оформлення логіки роботи програми.

У даній системі проведення тендерів графічний інтерфейс відіграє важливу роль: приємне відображення інформації – один з ключових моментів цієї роботи. За допомогою графічного інтерфейсу користувач взаємодіє з системою: надсилає запити та отримує відповідь. Надіслані запити обробляються за допомогою контролерів. На них формується та надсилається відповідь користувачу у вигляді JSP-сторінок. До контролерів за допомогою об’єктів для доступу до даних відбувається передача інформації про тендери, тендерні пропозиції, чати та інформація про користувача.

До серверної частини цієї системи належать наступні складові::

– вигляд

– контролери;

– моделі;

– об’єкти доступу до даних;

– репозиторій.

Під час роботи сервісу користувач використовує графічний інтерфейс, сформований за допомогою JSP-сторінок. В залежності від сторінки за запиту, який формує користувач, інформація з цих запитів передається до відповідних контролерів. Залежно від контролера, серверна частина виконує бізнес-логіку, яка встановлена в цьому контролері, наприклад, зареєструвати новий акаунт, створити тендер, написати переможцю тендеру і тд. Інформація з бази даних (у цьому випадку базою даних слугує ServletContext) передається та обробляється за допомогою DAO – Data Access Object. Ці об’єкти мають в собі набір методів для обробки, доступу та створенню нових даних. Дані створюються за допомогою відповідних класів: кожен клас відповідає певному об’єкту в базі даних. У цих класах присутні методи для отримання, редагування чи виведення певної інформації, яка присутня в цьому об’єкті. Після обробки інформації на контролері, відповідний контролер формує відповідь та перенаправляє користувача на певну сторінку, в залежності від того, який запит сформував користувач, яка потрібна, приміром, для виведення відредагованої інформації тендеру чи тендерної пропозиції. Це загальний опис цього веб-застосунку, який написаний для глобального розуміння цієї системи.

# 4 РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

Цей розділ містить інформацію про реалізацію компонентів системи, їх взаємозв’язки та функціонування.

4.1 Загальна структура проекту

Ця діаграма зв’язків діаграма 4.1 описує загальну структуру системи та взаємозв’язки між компонентами.

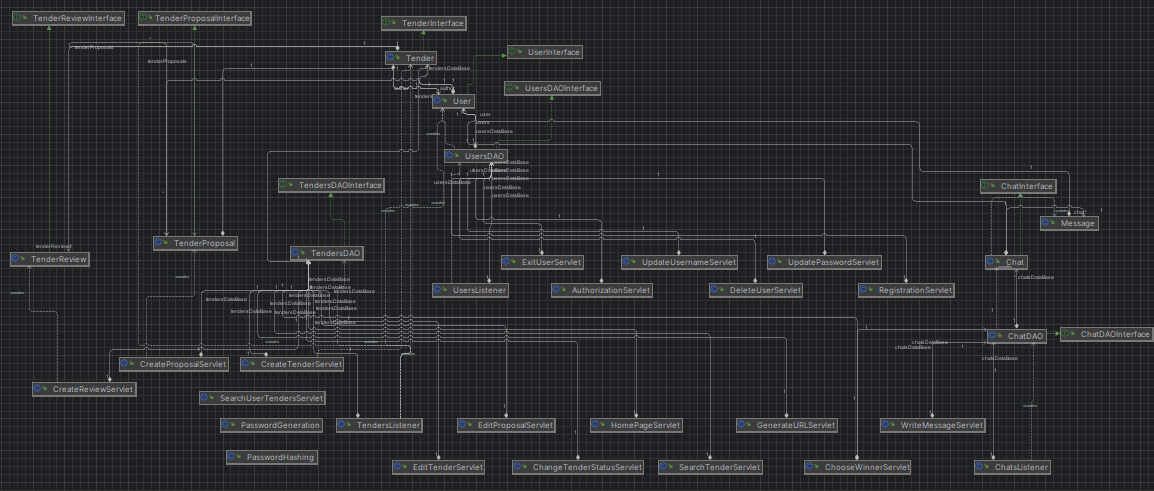
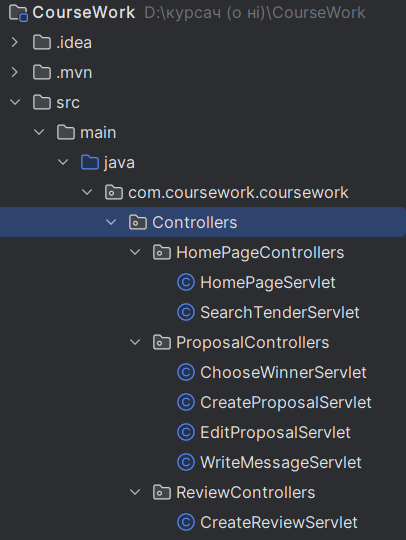
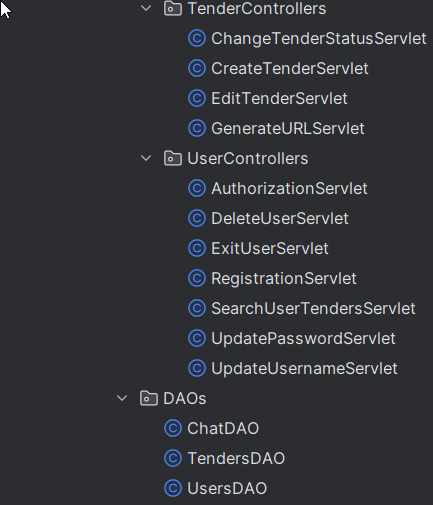
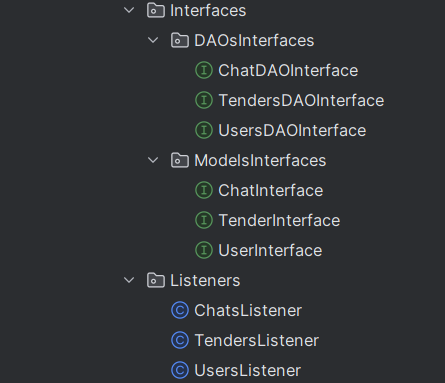


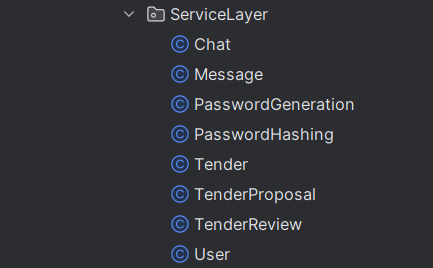
Рисунок 4.1 – Діаграма взаємозв’язків між компонентами системи

Загальна структура проекту представлена на рис.4.2









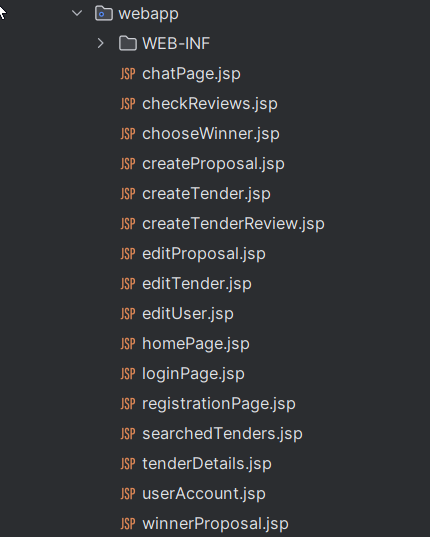


Рисунок 4.2 – Загальна структура проекту

Проект складається з веб-ресурсів, бібліотек, та вихідного коду, який в свою чергу можна поділити на компоненти рівня доступу до даних, компоненти бізнес-логіки та веб-компоненти.

## 4.2 Компоненти рівня доступу до даних

Основні сутності та інтерфейси рівня доступу до даних наведені на рис. 4.3

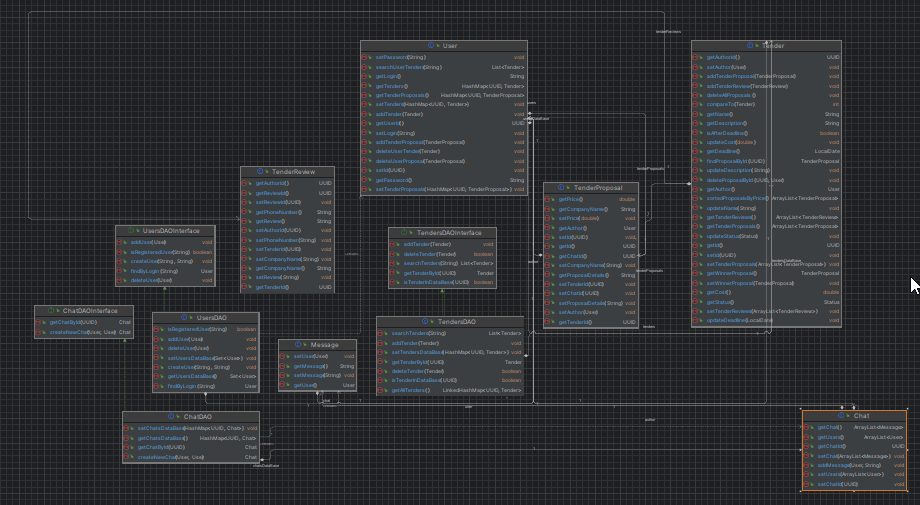


Рисунок 4.3 – Основні сутності та інтерфейси рівня доступу до даних

Компонентами рівня доступу до бази даних у цьому проекті слугують об’єкти доступу до даних (DAOs). Їх є три: UsersDAO, TendersDAO та ChatDAO. Кожен з цих класів імплементує відповідний інтерфейс, в якому прописані базові методи для додавання, видалення, редагування, пошук за певним значенням для доступу до об’єкту. UsersDAO дозволяє працювати з об’єктами, які представляють користувача у цій системі. TendersDAO надає методи для роботи з тендерами, тендерними пропозиціями, відгуками. ChatDAO працює з чатами та повідомленнями для створення зворотного зв’язку між хазяїном тендеру та переможцем.

Реалізація UsersDAO:

package com.coursework.coursework.DAOs;  
  
import com.coursework.coursework.Interfaces.DAOsInterfaces.UsersDAOInterface;  
import com.coursework.coursework.ServiceLayer.User;  
  
import java.util.\*;  
  
public class UsersDAO implements UsersDAOInterface {  
 private Set<User> usersDataBase = new TreeSet<>(Comparator.*comparing*(User::getLogin));  
  
 public void setUsersDataBase(Set<User> usersDataBase) {  
 this.usersDataBase = usersDataBase;  
 }  
  
 public Set<User> getUsersDataBase() {  
 return usersDataBase;  
 }  
  
 @Override  
 public boolean isRegisteredUser(String username) {  
 for (User user: usersDataBase) {  
 if (username.equals(user.getLogin())) {  
 return true;  
 }  
 }  
 return false;  
 }  
  
 @Override  
 public User findByLogin(String usernaem) {  
 for (User user : usersDataBase) {  
 if (user.getLogin().equals(usernaem)) {  
 return user;  
 }  
 }  
 return null;  
 }  
  
 @Override  
 public synchronized void createUser(String username, String password) {  
 if (!isRegisteredUser(username)) {  
 usersDataBase.add(new User(username, password));  
 }  
 }  
  
 @Override  
 public synchronized void addUser(User user) {  
 if (!isRegisteredUser(user.getLogin())) {  
 usersDataBase.add(user);  
 }  
 }  
  
 @Override  
 public synchronized void deleteUser(User user) {  
 if (isRegisteredUser(user.getLogin())) {  
 usersDataBase.remove(user);  
 }  
 }  
}

Реалізація TendersDAO:

package com.coursework.coursework.DAOs;  
  
import com.coursework.coursework.Interfaces.DAOsInterfaces.TendersDAOInterface;  
import com.coursework.coursework.ServiceLayer.Tender;  
import com.coursework.coursework.ServiceLayer.User;  
  
import java.util.\*;  
import java.util.stream.Collectors;  
  
public class TendersDAO implements TendersDAOInterface {  
 private Map<UUID, Tender> tendersDataBase = new LinkedHashMap<>(15, 0.65f);  
  
 public LinkedHashMap<UUID, Tender> getAllTenders() {  
 return tendersDataBase.entrySet()  
 .stream()  
 .sorted(Map.Entry.*comparingByValue*(Comparator.*comparing*(Tender::getName)))  
 .collect(Collectors.*toMap*(  
 Map.Entry::getKey,  
 Map.Entry::getValue,  
 (e1, e2) -> e1,  
 LinkedHashMap::new));  
 }  
  
 public void setTendersDataBase(HashMap<UUID, Tender> tendersDataBase) {  
 this.tendersDataBase = tendersDataBase;  
 }  
  
 @Override  
 public boolean isTenderNameInDataBase(String name) {  
 return tendersDataBase.values().stream()  
 .anyMatch(tender -> tender.getName().equalsIgnoreCase(name));  
 }  
  
 @Override  
 public void addTender(Tender tender) {  
 tendersDataBase.put(tender.getId(), tender);  
 }  
  
 @Override  
 public Tender getTenderById(UUID id) {  
 return tendersDataBase.get(id);  
 }  
  
 @Override  
 public boolean isTenderInDataBase(UUID id) {  
 return tendersDataBase.containsKey(id);  
 }  
  
 @Override  
 public boolean deleteTender(Tender tender) {  
 User user = tender.getAuthor();  
 if (isTenderInDataBase(tender.getId())) {  
 tendersDataBase.remove(tender.getId());  
 user.deleteUserTender(tender);  
 tender.deleteAllProposals();  
 return true;  
 } else return false;  
 }  
  
 @Override  
 public synchronized List<Tender> searchTenders(String keyword) {  
 return tendersDataBase.values().stream()  
 .filter(tender -> tender.getStatus() == Tender.Status.*ACTIVE*)  
 .filter(tender -> tender.getName().toLowerCase().contains(keyword.toLowerCase()))  
 .collect(Collectors.*toList*());  
 }  
}

Реалізація ChatDAO:

package com.coursework.coursework.DAOs;  
  
import com.coursework.coursework.Interfaces.DAOsInterfaces.ChatDAOInterface;  
import com.coursework.coursework.ServiceLayer.Chat;  
import com.coursework.coursework.ServiceLayer.User;  
  
import java.util.HashMap;  
import java.util.UUID;  
  
public class ChatDAO implements ChatDAOInterface {  
 private HashMap<UUID, Chat> chatsDataBase = new HashMap<>();  
  
 public HashMap<UUID, Chat> getChatsDataBase() {  
 return chatsDataBase;  
 }  
  
 public void setChatsDataBase(HashMap<UUID, Chat> chatsDataBase) {  
 this.chatsDataBase = chatsDataBase;  
 }  
  
 public Chat getChatById(UUID chatId) {  
 return chatsDataBase.get(chatId);  
 }  
  
 public Chat createNewChat(User user1, User user2) {  
 Chat chat = new Chat(user1, user2);  
 chatsDataBase.put(chat.getChatId(), chat);  
 return chat;  
 }  
}

## 4.3 Компоненти рівня бізнес-логіки

Компонентами бізнес-логіки слугують контролери (у цьому випадку - сервлети), які приймають HTTP запити, обробляють інформацію та об’єкти, та генерують відповідь користувачу. Сервлет HomePageServlet ініціалізує та передає на домашню сторінку усі активні тендери у відсортованому порядку. Інші сервлети виконують дії, описані в їх назві, тобто заплутатися буде важко. Кожен з цих контролерів отримує параметри з JSP-сторінок, перевіряє отримані параметри , перевіряє чи присутній користувач у даній сесії, звіряє належність певного об’єкта цьому користувачу.

Сервлет HomePageServlet:

package com.coursework.coursework.Controllers.HomePageControllers;  
  
import com.coursework.coursework.DAOs.TendersDAO;  
import com.coursework.coursework.ServiceLayer.Tender;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import java.io.IOException;  
import java.util.Map;  
import java.util.UUID;  
  
@WebServlet("/homePage")  
public class HomePageServlet extends HttpServlet {  
  
 private TendersDAO tendersDataBase;  
  
 @Override  
 public void init() {  
 tendersDataBase = (TendersDAO) getServletContext().getAttribute("tendersDataBase");  
 }  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 Map<UUID, Tender> tenders = tendersDataBase.getAllTenders();  
 request.setAttribute("tenders", tenders);  
 request.getRequestDispatcher("homePage.jsp").forward(request, response);  
  
  
 }  
}

Сервлет SearchTendersServlet:

package com.coursework.coursework.Controllers.HomePageControllers;  
  
import com.coursework.coursework.DAOs.TendersDAO;  
import com.coursework.coursework.ServiceLayer.Tender;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
  
import java.io.IOException;  
import java.util.List;  
  
@WebServlet("/searchTender")  
public class SearchTenderServlet extends HttpServlet {  
  
 private TendersDAO tendersDataBase;  
  
 @Override  
 public void init() {  
 tendersDataBase = (TendersDAO) getServletContext().getAttribute("tendersDataBase");  
 }  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 String keyword = request.getParameter("keyword");  
 if (keyword == null || keyword.trim().isEmpty()) {  
 response.sendRedirect("tendersPage.jsp");  
 return;  
 }  
  
 List<Tender> searchResults = tendersDataBase.searchTenders(keyword);  
  
 request.setAttribute("searchedTenders", searchResults);  
  
 request.getRequestDispatcher("searchedTenders.jsp").forward(request, response);  
 }  
}

Сервлет ChooseWinnerServlet:

package com.coursework.coursework.Controllers.ProposalControllers;  
  
import com.coursework.coursework.DAOs.ChatDAO;  
import com.coursework.coursework.DAOs.TendersDAO;  
import com.coursework.coursework.ServiceLayer.Chat;  
import com.coursework.coursework.ServiceLayer.Tender;  
import com.coursework.coursework.ServiceLayer.TenderProposal;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
  
import java.io.IOException;  
import java.util.UUID;  
  
@WebServlet("/chooseWinner")  
public class ChooseWinnerServlet extends HttpServlet {  
  
 TendersDAO tendersDataBase;  
 ChatDAO chatsDataBase;  
  
 @Override  
 public void init() {  
 tendersDataBase = (TendersDAO) getServletContext().getAttribute("tendersDataBase");  
 chatsDataBase = (ChatDAO) getServletContext().getAttribute("chatsDataBase");  
 }  
  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 String proposalId = request.getParameter("winnerId");  
 String id = request.getParameter("tenderId");  
  
 if (proposalId == null) {  
 response.sendError(400, "Немає Id пропозиції");  
 return;  
 }  
  
 if (id == null) {  
 response.sendError(400, "Немає Id тендеру");  
 return;  
 }  
  
 UUID winnerId = UUID.*fromString*(proposalId);  
 UUID tenderId = UUID.*fromString*(id);  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 response.sendError(400, "Користувача не знайдено");  
 return;  
 }  
  
 Tender tender = tendersDataBase.getTenderById(tenderId);  
  
 if (user != tender.getAuthor()) {  
 response.sendError(400, "Ви не є власником тендеру");  
 return;  
 }  
  
 TenderProposal winnerProposal = tender.findProposalById(winnerId);  
 Chat chat = chatsDataBase.createNewChat(user, winnerProposal.getAuthor());  
 winnerProposal.setChatId(chat.getChatId());  
 tender.setWinnerProposal(winnerProposal);  
  
 request.setAttribute("chooseWinnerSuccess", "Переможця тендеру \"" + tender.getName() + "\" успішно обрано");  
 request.getRequestDispatcher("userAccount.jsp").forward(request, response);  
  
 }  
}

Сервлет CreateProposalServlet:

package com.coursework.coursework.Controllers.ProposalControllers;  
  
import com.coursework.coursework.DAOs.TendersDAO;  
import com.coursework.coursework.ServiceLayer.Tender;  
import com.coursework.coursework.ServiceLayer.TenderProposal;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import java.io.IOException;  
import java.util.UUID;  
  
@WebServlet("/createProposal")  
public class CreateProposalServlet extends HttpServlet {  
  
 private TendersDAO tendersDataBase;  
  
 @Override  
 public void init() {  
 tendersDataBase = (TendersDAO) getServletContext().getAttribute("tendersDataBase");  
 }  
  
 protected void doPost(HttpServletRequest request, HttpServletResponse response)  
 throws ServletException, IOException {  
  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 request.setAttribute("errorMessage", "Для відгуку на тендер потрібно авторизуватися");  
 request.getRequestDispatcher("loginPage.jsp").forward(request, response);  
 return;  
 }  
  
 UUID tenderId;  
 try {  
 tenderId = UUID.*fromString*(request.getParameter("id"));  
 } catch (NumberFormatException e) {  
 response.sendError(500, "Недійсний id тендеру");  
 return;  
 }  
  
 String proposalDetails = request.getParameter("proposalDetails");  
 String companyName = request.getParameter("companyName");  
 double price;  
  
 try {  
 price = Double.*parseDouble*(request.getParameter("price"));  
 } catch (NumberFormatException e) {  
 request.setAttribute("errorProposal", "Недійсна ціна");  
 request.getRequestDispatcher("createProposal.jsp?id" + tenderId).forward(request, response);  
 return;  
 }  
  
 if (companyName.isEmpty()) {  
 request.setAttribute("errorProposal", "Назва компанії повинна бути заповнена");  
 request.getRequestDispatcher("createProposal.jsp?id" + tenderId).forward(request, response);  
 return;  
 }  
  
 Tender tender = tendersDataBase.getTenderById(tenderId);  
 if (tender == null) {  
 response.sendError(404, "Тендер не знайдено");  
 return;  
 }  
 if (price > tender.getCost()) {  
 request.setAttribute("errorProposal", "Ціна пропозиції повинна бути меншою, ніж бюджет на тендер");  
 request.getRequestDispatcher("createProposal.jsp?id" + tenderId).forward(request, response);  
 return;  
 }  
  
 TenderProposal proposal = new TenderProposal(tender.getId(), companyName, proposalDetails, price, user);  
 tender.addTenderProposal(proposal);  
 user.addTenderProposal(proposal);  
  
 request.setAttribute("proposalSuccess", "Тендерну пропозицію успішно створено ");  
 request.getRequestDispatcher("tenderDetails.jsp?tenderId=" + tenderId).forward(request, response);  
 }  
}

Сервлет EditProposalServlet:

package com.coursework.coursework.Controllers.ProposalControllers;  
  
import com.coursework.coursework.DAOs.TendersDAO;  
import com.coursework.coursework.ServiceLayer.Tender;  
import com.coursework.coursework.ServiceLayer.TenderProposal;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import java.io.IOException;  
import java.util.UUID;  
  
@WebServlet("/editProposal")  
public class EditProposalServlet extends HttpServlet {  
  
 private TendersDAO tendersDataBase;  
  
 @Override  
 public void init() {  
 tendersDataBase = (TendersDAO) getServletContext().getAttribute("tendersDataBase");  
 }  
  
 protected void doPost(HttpServletRequest request, HttpServletResponse response)  
 throws ServletException, IOException {  
  
 String tenderIdStr = request.getParameter("tenderId");  
 String proposalIdStr = request.getParameter("proposalId");  
  
 if (tenderIdStr == null || proposalIdStr == null) {  
 response.sendError(400, "ID тендеру або пропозиції не вказано");  
 return;  
 }  
  
 UUID tenderId = UUID.*fromString*(tenderIdStr);  
 UUID proposalId = UUID.*fromString*(proposalIdStr);  
  
 Tender tender = tendersDataBase.getTenderById(tenderId);  
  
 if (tender == null) {  
 response.sendError(400, "Тендер не знайдено");  
 return;  
 }  
  
 TenderProposal proposal = tender.findProposalById(proposalId);  
  
 if (proposal == null) {  
 response.sendError(400, "Пропозицію не знайдено");  
 return;  
 }  
  
 String companyName = request.getParameter("companyName");  
 String proposalDetails = request.getParameter("proposalDetails");  
 double price;  
  
 try {  
 price = Double.*parseDouble*(request.getParameter("price"));  
 } catch (NumberFormatException e) {  
 request.setAttribute("errorProposal", "Недійсна ціна");  
 request.getRequestDispatcher("editProposal.jsp").forward(request, response);  
 return;  
 }  
  
 if (companyName.isEmpty()) {  
 request.setAttribute("errorProposal", "Назва компанії повинна бути заповнена");  
 request.getRequestDispatcher("editProposal.jsp").forward(request, response);  
 return;  
 }  
  
 if (price > tender.getCost()) {  
 request.setAttribute("errorProposal", "Ціна пропозиції повинна бути меншою, ніж бюджет на тендер");  
 request.getRequestDispatcher("editProposal.jsp").forward(request, response);  
 return;  
 }  
  
 proposal.setCompanyName(companyName);  
 proposal.setProposalDetails(proposalDetails);  
 proposal.setPrice(price);  
  
 request.setAttribute("editProposalSuccess", "Тендерну пропозицію успішно оновлено");  
 request.getRequestDispatcher("userAccount.jsp").forward(request, response);  
 }  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 String tenderIdStr = request.getParameter("tenderId");  
 String proposalIdStr = request.getParameter("proposalId");  
  
 if (tenderIdStr == null || proposalIdStr == null) {  
 response.sendError(400, "ID тендеру або пропозиції не вказано");  
 return;  
 }  
  
 UUID tenderId = UUID.*fromString*(tenderIdStr);  
 UUID proposalId = UUID.*fromString*(proposalIdStr);  
  
 Tender tender = tendersDataBase.getTenderById(tenderId);  
  
 if (tender == null) {  
 response.sendError(400, "Тендер не знайдено");  
 return;  
 }  
  
 TenderProposal proposal = tender.findProposalById(proposalId);  
  
 if (proposal == null) {  
 response.sendError(400, "Пропозицію не знайдено");  
 return;  
 }  
  
 User user = (User) request.getSession().getAttribute("user");  
 tender.deleteProposalById(proposal.getId(), user);  
 user.deleteUserProposal(proposal);  
  
 request.setAttribute("editProposalSuccess", "Пропозицію успішно видалено");  
 request.getRequestDispatcher("userAccount.jsp").forward(request, response);  
 }  
}

Сервлет WriteMessageServlet:

package com.coursework.coursework.Controllers.ProposalControllers;  
  
import com.coursework.coursework.DAOs.ChatDAO;  
import com.coursework.coursework.ServiceLayer.Chat;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
  
import java.io.IOException;  
import java.util.UUID;  
  
@WebServlet("/writeMessage")  
public class WriteMessageServlet extends HttpServlet {  
 private ChatDAO chatsDataBase;  
  
 @Override  
 public void init() {  
 chatsDataBase = (ChatDAO) getServletContext().getAttribute("chatsDataBase");  
 }  
  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 User user = (User) request.getSession().getAttribute("user");  
 String id = request.getParameter("chatId");  
  
 if (id == null || id.isEmpty()) {  
 response.sendError(500, "Id чату відсутній");  
 return;  
 }  
  
 UUID chatId = UUID.*fromString*(id);  
  
 String message = request.getParameter("message");  
  
 if (message == null || message.isEmpty()) {  
 request.setAttribute("errorMessage", "Ви не написали повідомлення");  
 request.getRequestDispatcher("chatPage.jsp?chatId=" + chatId).forward(request, response);  
 }  
  
 Chat chat = chatsDataBase.getChatById(chatId);  
 chat.addMessage(user, message);  
  
 request.getRequestDispatcher("chatPage.jsp?chatId=" + chatId).forward(request, response);  
  
 }  
}

Сервлет CreateReviewServlet:

package com.coursework.coursework.Controllers.ReviewControllers;  
  
import com.coursework.coursework.DAOs.TendersDAO;  
import com.coursework.coursework.ServiceLayer.Tender;  
import com.coursework.coursework.ServiceLayer.TenderReview;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
  
import java.io.IOException;  
import java.util.UUID;  
import java.util.regex.Pattern;  
  
@WebServlet("/createReview")  
public class CreateReviewServlet extends HttpServlet {  
  
 private TendersDAO tendersDataBase;  
 private static final String *PHONE\_NUMBER\_PATTERN* = "\\+?\\d{1,3}?\\s?-?\\d{1,4}?\\s?-?\\d{1,4}?\\s?-?\\d{1,9}";  
  
 @Override  
 public void init() {  
 tendersDataBase = (TendersDAO) getServletContext().getAttribute("tendersDataBase");  
 }  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 User user = (User) request.getSession().getAttribute("user");  
  
 String id = request.getParameter("tenderId");  
  
 if (id == null) {  
 response.sendError(500, "Недійсний id тендеру");  
 return;  
 }  
  
 UUID tenderId = UUID.*fromString*(id);  
 String companyName= request.getParameter("companyName");  
  
 if (companyName == null || companyName.isEmpty()) {  
 request.setAttribute("errorReview", "Назва компанії повинна бути заповнена");  
 request.getRequestDispatcher("createTenderReview.jsp").forward(request, response);  
 return;  
 }  
  
 String review = request.getParameter("review");  
  
 if (review == null || review.isEmpty()) {  
 request.setAttribute("errorReview", "Поле відгуку пусте");  
 request.getRequestDispatcher("createTenderReview.jsp").forward(request, response);  
 return;  
 }  
  
 String phoneNumber = request.getParameter("phoneNumber");  
  
 if (!isValidPhoneNumber(phoneNumber)) {  
 request.setAttribute("errorReview", "Введіть правильний номер (без введення +380)");  
 request.getRequestDispatcher("createTenderReview.jsp").forward(request, response);  
 return;  
 }  
  
 Tender tender = tendersDataBase.getTenderById(tenderId);  
  
 if (tender == null) {  
 response.sendError(404, "Тендер не знайдено");  
 return;  
 }  
  
 TenderReview tenderReview = new TenderReview(tenderId, user.getUserId(), review, phoneNumber, companyName);  
 tender.addTenderReview(tenderReview);  
  
 request.setAttribute("successReviewMessage", "Відгук успішно створено");  
 request.getRequestDispatcher("homePage").forward(request, response);  
 }  
  
 private boolean isValidPhoneNumber(String phoneNumber) {  
 return Pattern.*matches*(*PHONE\_NUMBER\_PATTERN*, phoneNumber);  
 }  
}

Сервлет ChangeTenderStatusServlet:

package com.coursework.coursework.Controllers.TenderControllers;  
  
  
import com.coursework.coursework.DAOs.TendersDAO;  
import com.coursework.coursework.ServiceLayer.Tender;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
  
import java.io.IOException;  
import java.util.UUID;  
  
@WebServlet("/changeTenderStatus")  
public class ChangeTenderStatusServlet extends HttpServlet {  
  
 private TendersDAO tendersDataBase;  
  
 @Override  
 public void init() {  
 tendersDataBase = (TendersDAO) getServletContext().getAttribute("tendersDataBase");  
 }  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException {  
  
 User user = (User) request.getSession().getAttribute("user");  
 String id = request.getParameter("id");  
 String newStatus = request.getParameter("newStatus");  
  
 if (id == null) {  
 response.sendError(400, "Немає Id тендеру");  
 return;  
 }  
  
 UUID tenderId = UUID.*fromString*(id);  
  
 if (user == null) {  
 response.sendError(400, "Користувача не знайдено");  
 return;  
 }  
  
 if (!tendersDataBase.isTenderInDataBase(tenderId)) {  
 response.sendError(400, "Тендер з таким id відсутній у базі даних"); // тут проблема  
 }  
  
 Tender tender = tendersDataBase.getTenderById(tenderId);  
  
 if (user != tender.getAuthor()) {  
 response.sendError(400, "Ви не є власником тендеру");  
 return;  
 }  
  
 if (newStatus.equals("activate")) {  
 tender.updateStatus(Tender.Status.*ACTIVE*);  
 } else tender.updateStatus(Tender.Status.*INACTIVE*);  
  
 response.sendRedirect("tenderDetails.jsp?id=" + tenderId);  
 }  
  
}

Сервлет CreateTenderServlet:

package com.coursework.coursework.Controllers.TenderControllers;  
  
import com.coursework.coursework.DAOs.TendersDAO;  
import com.coursework.coursework.ServiceLayer.Tender;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import java.io.IOException;  
import java.time.LocalDate;  
import java.time.format.DateTimeParseException;  
  
@WebServlet("/createTender")  
public class CreateTenderServlet extends HttpServlet {  
  
 private TendersDAO tendersDataBase;  
  
 @Override  
 public void init() {  
 tendersDataBase = (TendersDAO) getServletContext().getAttribute("tendersDataBase");  
 }  
  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException {  
  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 request.setAttribute("errorMessage", "Для відгуку на тендер потрібно авторизуватися");  
 request.setAttribute("addressing", "createTender");  
 request.getRequestDispatcher("loginPage.jsp").forward(request, response);  
 return;  
 }  
  
 String name = request.getParameter("tenderName");  
 String description = request.getParameter("tenderDescription");  
 String deadlineStr = request.getParameter("tenderDeadline");  
 String costStr = request.getParameter("tenderCost");  
  
 if (name == null || description == null || deadlineStr == null || costStr == null) {  
 request.setAttribute("errorTender", "Введіть дані");  
 request.getRequestDispatcher("createTender.jsp").forward(request, response);  
 return;  
 }  
  
 if (tendersDataBase.isTenderNameInDataBase(name)) {  
 request.setAttribute("errorTender", "Таке ім'я тендеру уже існує");  
 request.getRequestDispatcher("createTender.jsp").forward(request, response);  
 return;  
 }  
  
 LocalDate deadline;  
 double cost;  
  
 try {  
 deadline = LocalDate.*parse*(deadlineStr);  
 } catch (DateTimeParseException e) {  
 request.setAttribute("errorTender", "Ведіть правильний формат дати");  
 request.getRequestDispatcher("createTender.jsp").forward(request, response);  
 return;  
 }  
  
 try {  
 cost = Double.*parseDouble*(costStr);  
 } catch (NumberFormatException e) {  
 request.setAttribute("errorTender", "Неправильно введена ціна");  
 request.getRequestDispatcher("createTender.jsp").forward(request, response);  
 return;  
 }  
  
 Tender tender = new Tender(name, description, deadline, cost, user);  
 tendersDataBase.addTender(tender);  
 user.addTender(tender);  
 response.sendRedirect("tenderDetails.jsp?id=" + tender.getId());  
 }  
}

Сервлет EditTenderServlet:

package com.coursework.coursework.Controllers.TenderControllers;  
  
import com.coursework.coursework.DAOs.TendersDAO;  
import com.coursework.coursework.ServiceLayer.Tender;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import java.io.IOException;  
import java.time.LocalDate;  
import java.time.format.DateTimeParseException;  
import java.util.UUID;  
  
@WebServlet("/editTender")  
public class EditTenderServlet extends HttpServlet {  
  
 private TendersDAO tendersDataBase;  
  
 @Override  
 public void init() {  
 tendersDataBase = (TendersDAO) getServletContext().getAttribute("tendersDataBase");  
 }  
  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 String id = request.getParameter("id");  
  
 if (id == null) {  
 response.sendError(400, "Немає Id тендеру");  
 return;  
 }  
  
 UUID tenderId = UUID.*fromString*(id);  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 response.sendError(400, "Користувача не знайдено");  
 return;  
 }  
  
 Tender tender = tendersDataBase.getTenderById(tenderId);  
 if (user != tender.getAuthor()) {  
 response.sendError(400, "Ви не є власником тендеру");  
 return;  
 }  
  
 String name = request.getParameter("name");  
 String description = request.getParameter("description");  
 String deadlineStr = request.getParameter("deadline");  
 String costStr = request.getParameter("cost");  
  
 if (name == null || description == null || deadlineStr == null || costStr == null) {  
 request.setAttribute("errorEditTender", "Введіть дані");  
 request.getRequestDispatcher("editTender.jsp").forward(request, response);  
 return;  
 }  
  
 if (tendersDataBase.isTenderNameInDataBase(name)) {  
 request.setAttribute("errorEditTender", "Таке ім'я тендеру уже існує");  
 request.getRequestDispatcher("editTender.jsp").forward(request, response);  
 return;  
 }  
  
 LocalDate deadline;  
 double cost;  
  
 try {  
 deadline = LocalDate.*parse*(deadlineStr);  
 } catch (DateTimeParseException e) {  
 request.setAttribute("errorEditTender", "Ведіть правильний формат дати");  
 request.getRequestDispatcher("editTender.jsp").forward(request, response);  
 return;  
 }  
  
 try {  
 cost = Double.*parseDouble*(costStr);  
 } catch (NumberFormatException e) {  
 request.setAttribute("errorEditTender", "Неправильно введена ціна");  
 request.getRequestDispatcher("editTender.jsp").forward(request, response);  
 return;  
 }  
  
 tender.updateName(name);  
 tender.updateDescription(description);  
 tender.updateDeadline(deadline);  
 tender.updateCost(cost);  
  
 request.setAttribute("successEditMessage", "Тендер успішно оновлено");  
 response.sendRedirect("tenderDetails.jsp?id=" + tender.getId());  
 }  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 User user = (User) request.getSession().getAttribute("user");  
 String id = request.getParameter("id");  
  
 if (id == null) {  
 response.sendError(400, "Немає Id тендеру");  
 return;  
 }  
  
 UUID tenderId = UUID.*fromString*(id);  
 boolean isDeletedTender;  
  
 if (user != tendersDataBase.getTenderById(tenderId).getAuthor()) {  
 response.sendError(500, "Ви не є власником");  
 return;  
 }  
  
 isDeletedTender = tendersDataBase.deleteTender(tendersDataBase.getTenderById(tenderId));  
  
  
 if (!isDeletedTender) {  
 request.setAttribute("deleteError", "Тендер не видалено");  
 request.getRequestDispatcher("editTender.jsp").forward(request, response);  
 } else request.getRequestDispatcher("homePage").forward(request, response);  
 }  
}

Сервлет GenerateURLServlet:

package com.coursework.coursework.Controllers.TenderControllers;  
  
import com.coursework.coursework.DAOs.TendersDAO;  
import com.coursework.coursework.ServiceLayer.Tender;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
  
import java.io.IOException;  
import java.util.UUID;  
  
@WebServlet("/generateURL")  
public class GenerateURLServlet extends HttpServlet {  
  
 TendersDAO tendersDataBase;  
  
 @Override  
 public void init() {  
 tendersDataBase = (TendersDAO) getServletContext().getAttribute("tendersDataBase");  
 }  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 String id = request.getParameter("id");  
  
 if (id == null) {  
 response.sendError(400, "Немає Id тендеру");  
 return;  
 }  
  
 UUID tenderId = UUID.*fromString*(id);  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 response.sendError(400, "Користувача не знайдено");  
 return;  
 }  
  
 Tender tender = tendersDataBase.getTenderById(tenderId);  
 if (user != tender.getAuthor()) {  
 response.sendError(400, "Ви не є власником тендеру");  
 return;  
 }  
  
 request.setAttribute("URL", generateURL(tender));  
 request.getRequestDispatcher("tenderDetails.jsp?id=" + tenderId).forward(request, response);  
 }  
  
 private String generateURL(Tender tender) {  
 if (tendersDataBase.isTenderInDataBase(tender.getId())) {  
 return "http://localhost:8080/CourseWork\_war\_exploded/tenderDetails.jsp?id=" + tender.getId();  
 } else return "Ви не є власником цього тендеру";  
 }  
}

Сервлет AuthorizationServlet:

package com.coursework.coursework.Controllers.UserControllers;  
  
import com.coursework.coursework.DAOs.UsersDAO;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import java.io.IOException;  
  
import static com.coursework.coursework.ServiceLayer.PasswordHashing.*hashPassword*;  
  
@WebServlet("/authorization")  
public class AuthorizationServlet extends HttpServlet {  
  
 private UsersDAO usersDataBase;  
  
 @Override  
 public void init() {  
 usersDataBase = (UsersDAO) getServletContext().getAttribute("usersDataBase");  
 }  
  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 String username = request.getParameter("username");  
 String password = request.getParameter("password");  
 String addressing = request.getParameter("addressing");  
  
 if (usersDataBase.isRegisteredUser(username)) {  
 User user = usersDataBase.findByLogin(username);  
  
 if (user != null && user.getPassword().equals(*hashPassword*(password))) {  
 request.getSession().setAttribute("user", user);  
  
 if (addressing.equals("createTender")) {  
 request.getRequestDispatcher("createTender.jsp").forward(request, response);  
 return;  
 }  
 if (addressing.equals("review")) {  
 String tenderId = request.getParameter("tenderId");  
 request.getRequestDispatcher("createTenderReview.jsp?id=" + tenderId).forward(request, response);  
 return;  
 }  
 if (addressing.equals("createProposal")) {  
 String tenderId = request.getParameter("tenderId");  
 request.getRequestDispatcher("createProposal.jsp?id=" + tenderId).forward(request, response);  
 return;  
 }  
  
 response.sendRedirect("homePage");  
  
 } else {  
 request.setAttribute("errorMessage", "Неправильний логін або пароль");  
 request.getRequestDispatcher("loginPage.jsp").forward(request, response);  
 }  
 } else {  
 request.setAttribute("errorMessage", "Користувача з таким логіном не знайдено");  
 request.getRequestDispatcher("loginPage.jsp").forward(request, response);  
 }  
 }  
}

Сервлет DeleteUserServlet:

package com.coursework.coursework.Controllers.UserControllers;  
  
import com.coursework.coursework.DAOs.UsersDAO;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import java.io.IOException;  
  
@WebServlet("/deleteUser")  
public class DeleteUserServlet extends HttpServlet {  
  
 private UsersDAO usersDataBase;  
  
 @Override  
 public void init() {  
 usersDataBase = (UsersDAO) getServletContext().getAttribute("usersDataBase");  
 }  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response)  
 throws IOException {  
  
 User user = (User) request.getSession().getAttribute("user");  
  
  
 if (user != null && usersDataBase.isRegisteredUser(user.getLogin())) {  
  
 usersDataBase.deleteUser(user);  
 request.getSession().invalidate();  
 response.sendRedirect("homePage.jsp");  
  
 } else {  
 response.sendError(500, "Помилка при видаленні облікового запису");  
 }  
 }  
}

Сервлет ExitUserServlet:

package com.coursework.coursework.Controllers.UserControllers;  
  
import com.coursework.coursework.DAOs.UsersDAO;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import java.io.IOException;  
  
@WebServlet("/exitUser")  
public class ExitUserServlet extends HttpServlet {  
  
 private UsersDAO usersDataBase;  
  
 @Override  
 public void init() {  
 usersDataBase = (UsersDAO) getServletContext().getAttribute("usersDataBase");  
 }  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response)  
 throws IOException, ServletException {  
  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user != null && usersDataBase.isRegisteredUser(user.getLogin())) {  
 request.getSession().invalidate();  
 response.sendRedirect("homePage");  
 } else {  
 response.sendError(500, "Помилка при виході з облікового запису");  
 }  
 }  
}

Сервлет RegistrationServlet:

package com.coursework.coursework.Controllers.UserControllers;  
  
import com.coursework.coursework.DAOs.UsersDAO;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import java.io.IOException;  
  
@WebServlet("/registration")  
public class RegistrationServlet extends HttpServlet {  
  
 private UsersDAO usersDataBase;  
  
 @Override  
 public void init() {  
 usersDataBase = (UsersDAO) getServletContext().getAttribute("usersDataBase");  
 }  
  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 String username = request.getParameter("username");  
 String password = request.getParameter("password");  
  
  
 if (usersDataBase.isRegisteredUser(username)) {  
 request.setAttribute("errorMessage", "Користувач з таким логіном зареєстрований");  
 request.getRequestDispatcher("registrationPage.jsp").forward(request, response);  
 } else {  
 usersDataBase.createUser(username, password);  
 request.setAttribute("successMessage", "Ви успішно зареєструвалися");  
 request.getRequestDispatcher("registrationPage.jsp").forward(request, response);  
 }  
 }  
}

Сервлет SearchUserTendersServlet:

package com.coursework.coursework.Controllers.UserControllers;  
  
import com.coursework.coursework.ServiceLayer.Tender;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
  
import java.io.IOException;  
import java.util.List;  
  
@WebServlet("/searchUserTenders")  
public class SearchUserTendersServlet extends HttpServlet {  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 User user = (User) request.getSession().getAttribute("user");  
  
 String keyword = request.getParameter("keyword");  
  
 if (keyword == null || keyword.trim().isEmpty()) {  
 response.sendRedirect("homePage");  
 return;  
 }  
  
 List<Tender> searchResults = user.searchUserTenders(keyword);  
  
 request.setAttribute("searchedTenders", searchResults);  
  
 request.getRequestDispatcher("searchedTenders.jsp").forward(request, response);  
 }  
}

Сервлет UpdatePasswordServlet:

package com.coursework.coursework.Controllers.UserControllers;  
  
import com.coursework.coursework.DAOs.UsersDAO;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import java.io.IOException;  
  
@WebServlet("/updatePassword")  
public class UpdatePasswordServlet extends HttpServlet {  
  
 private UsersDAO usersDataBase;  
  
 @Override  
 public void init() {  
 usersDataBase = (UsersDAO) getServletContext().getAttribute("usersDataBase");  
 }  
  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response)  
 throws ServletException, IOException {  
  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 System.*out*.println("user is null");  
 }  
  
 if (user != null) {  
  
 String newPassword = request.getParameter("newPassword");  
  
 if (newPassword != null && !newPassword.isEmpty() && !usersDataBase.isRegisteredUser(user.getLogin())) {  
 user.setPassword(newPassword);  
 request.getSession().setAttribute("user", user);  
 request.setAttribute("passwordMessage", "Пароль успішно змінений");  
  
 } else {  
 if (newPassword == null) {  
 request.setAttribute("passwordError", "Ви не ввели новий пароль");  
 }  
 }  
 request.getRequestDispatcher("editUser.jsp").forward(request, response);  
  
 } else {  
 response.sendError(500, "Об'єкт користувача нульовий");  
 }  
 }  
}

Сервлет UpdateUsernameServlet:

package com.coursework.coursework.Controllers.UserControllers;  
  
import com.coursework.coursework.DAOs.UsersDAO;  
import com.coursework.coursework.ServiceLayer.User;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import java.io.IOException;  
  
@WebServlet("/updateUsername")  
public class UpdateUsernameServlet extends HttpServlet {  
  
 private UsersDAO usersDataBase;  
  
 @Override  
 public void init() {  
 usersDataBase = (UsersDAO) getServletContext().getAttribute("usersDataBase");  
 }  
  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response)  
 throws ServletException, IOException {  
  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 System.*out*.println("user is null");  
 }  
  
 if (user != null) {  
  
 String newUsername = request.getParameter("newUsername");  
  
 if (newUsername != null && !newUsername.isEmpty() && !usersDataBase.isRegisteredUser(newUsername)) {  
 user.setLogin(newUsername);  
 request.getSession().setAttribute("user", user);  
 request.setAttribute("usernameMessage", "Логін успішно змінений");  
  
 } else {  
 if (newUsername == null || newUsername.isEmpty()) {  
 request.setAttribute("usernameError", "Ви не ввели новий логін");  
 } else {  
 request.setAttribute("usernameError", "Цей логін вже зайнятий");  
 }  
 }  
 request.getRequestDispatcher("editUser.jsp").forward(request, response);  
  
 } else {  
 response.sendError(500, "Об'єкт користувача нульовий");  
 }  
 }  
}

## 4.4 Компоненти рівня інтерфейсу користувача

До даних компонентів належать JSP-сторінки – сторінки, які (у цьому випадку) використовують HTML та CSS для створення вигляду, передачі параметрів, написання форм та, загалом, взаємодії користувача з сервісом. Також у моєму випадку використовуються скриплети (рядки Java коду) та технологія JSTL для зручного написання перевірок для відображення кнопок та циклів для виведення інформації про тендери, пропозиції, відгуки та повідомлення. Назва кожної сторінки відповідає тому, для чого вона призначена, отже, тут також важко заплутатися.

chatPage.jsp (сторінка для переписування між двома користувачами):

<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ **page** import="java.util.UUID" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.User" %>  
<%@ **page** import="com.coursework.coursework.DAOs.ChatDAO" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.Chat" %>  
<%@ **page** contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  
<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <title>Повідомлення</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 display: flex;  
 flex-direction: column;  
 min-height: 100vh;  
 }  
 .container {  
 max-width: 800px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 flex: 1;  
 display: flex;  
 flex-direction: column;  
 }  
 .messages {  
 flex: 1;  
 padding: 20px;  
 overflow-y: auto;  
 max-width: 100%;  
 }  
 .message-item {  
 margin-bottom: 10px;  
 padding: 10px;  
 background-color: #444;  
 border: 1px solid #555;  
 border-radius: 4px;  
 word-wrap: break-word;  
 }  
 .form-container {  
 width: 100%;  
 background-color: #333;  
 border-top: 1px solid #555;  
 padding: 10px 20px;  
 display: flex;  
 justify-content: center;  
 position: fixed;  
 bottom: 0;  
 }  
 .form-container form {  
 display: flex;  
 max-width: 800px;  
 width: 100%;  
 }  
 .form-container input[type="text"] {  
 flex: 1;  
 padding: 10px;  
 font-size: 16px;  
 border: 1px solid #555;  
 border-radius: 4px;  
 margin-right: 10px;  
 background-color: #444;  
 color: #ddd;  
 }  
 .form-container input[type="submit"] {  
 padding: 10px 20px;  
 font-size: 16px;  
 border: none;  
 border-radius: 4px;  
 background-color: #28a745;  
 color: white;  
 cursor: pointer;  
 }  
 .form-container input[type="submit"]:hover {  
 background-color: #218838;  
 }  
 .error {  
 color: red;  
 margin-top: 5px;  
 }  
 </style>  
</head>  
<body>  
<%  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 response.sendError(500, "Ви не ввійшли");  
 return;  
 }  
  
 String id = request.getParameter("chatId");  
  
 if (id == null || id.isEmpty()) {  
 response.sendError(500, "Id чату відсутнє");  
 return;  
 }  
  
 UUID chatId = UUID.*fromString*(id);  
 ChatDAO chatsDataBase = (ChatDAO) request.getServletContext().getAttribute("chatsDataBase");  
 Chat chat = chatsDataBase.getChatById(chatId);  
  
 if (!chat.getUsers().contains(user)) {  
 response.sendError(500, "Ви не є учасником цього чату");  
 return;  
 }  
  
 request.setAttribute("chat", chat);  
%>  
<div class="container">  
 <div class="messages">  
 <**c:choose**>  
 <**c:when** test="${empty chat.chat}">  
 <h2>У цьому чаті ще немає повідомлень</h2>  
 </**c:when**>  
 <**c:otherwise**>  
 <**c:forEach** var="message" items="${chat.chat}">  
 <div class="message-item">  
 <p><strong>${message.user.login}:</strong> ${message.message}</p>  
 </div>  
 </**c:forEach**>  
 </**c:otherwise**>  
 </**c:choose**>  
 </div>  
</div>  
<div class="form-container">  
 <form action="writeMessage" method="post">  
 <label>  
 <span class="error">${errorMessage}</span><br>  
 <input type="text" name="message" placeholder="Введіть ваше повідомлення" required>  
 </label>  
 <input type="hidden" name="chatId" value="${chat.chatId}">  
 <input type="submit" value="Відправити">  
 </form>  
</div>  
</body>  
</html>

checkReviews.jsp (сторінка для перегляду відгуків):

<%@ **page** import="java.util.UUID" %>  
<%@ **page** import="com.coursework.coursework.DAOs.TendersDAO" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.Tender" %>  
<%@ **page** contentType="text/html;charset=UTF-8" %>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Відгуки на тендер</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 max-width: 800px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 }  
 .header a {  
 color: #fff;  
 text-decoration: none;  
 padding: 10px 20px;  
 background-color: #444;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 display: inline-block;  
 margin-bottom: 10px;  
 }  
 .header a:hover {  
 background-color: #555;  
 }  
 .tender {  
 padding: 10px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 margin-bottom: 20px;  
 background-color: #444;  
 }  
 .review {  
 margin-left: 20px;  
 padding: 5px;  
 border-left: 2px solid #555;  
 background-color: #555;  
 }  
 .button {  
 display: inline-block;  
 padding: 10px 20px;  
 margin-top: 10px;  
 background-color: #008CBA;  
 color: #fff;  
 text-decoration: none;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 }  
 .button:hover {  
 background-color: #005f7f;  
 }  
 </style>  
</head>  
<body>  
<%  
 String id = request.getParameter("id");  
  
 if (id != null && !id.isEmpty()) {  
 UUID tenderId = UUID.*fromString*(request.getParameter("id"));  
 TendersDAO tendersDAO = (TendersDAO) request.getServletContext().getAttribute("tendersDataBase");  
 Tender tender = tendersDAO.getTenderById(tenderId);  
  
 if (tender == null) {  
 response.sendError(500, "Тендер за таким id не знайдено");  
 } else {  
 request.setAttribute("tender", tender);  
 }  
 } else {  
 response.sendError(500, "Id тендеру відсутнє");  
 return;  
 }  
%>  
  
<div class="container">  
 <h1>Відгуки на тендер "${tender.name}"</h1>  
  
 <**c:if** test="${empty tender.tenderReviews}">  
 <h2>У вас поки немає відгуків</h2>  
 </**c:if**>  
 <div class="tender">  
 <**c:if** test="${not empty tender.tenderReviews}">  
 <**c:forEach** var="review" items="${tender.tenderReviews}">  
 <div class="review">  
 <p>Назва компанії: ${review.companyName}</p>  
 <p>${review.review}</p>  
 <p>Номер телефону: ${review.phoneNumber}</p>  
 </div>  
 </**c:forEach**>  
 </**c:if**>  
 </div>  
 <a href="userAccount.jsp" class="button">Повернутися назад</a>  
</div>  
</body>  
</html>

chooseWinner.jsp (сторінка для вибору переможця тендеру):

<%@ **page** import="java.util.UUID" %>  
<%@ **page** import="com.coursework.coursework.DAOs.TendersDAO" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.Tender" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.User" %>  
<%@ **page** contentType="text/html;charset=UTF-8" %>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ **taglib** prefix="fmt" uri="jakarta.tags.fmt" %>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Вибір переможця</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 max-width: 800px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 }  
 .header {  
 text-align: right;  
 margin-bottom: 20px;  
 }  
 .header a {  
 color: #fff;  
 text-decoration: none;  
 padding: 10px 20px;  
 background-color: #444;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 display: inline-block;  
 margin-bottom: 10px;  
 }  
 .header a:hover {  
 background-color: #555;  
 }  
 h1 {  
 text-align: center;  
 }  
 .proposal {  
 padding: 10px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 margin-bottom: 20px;  
 background-color: #444;  
 }  
 .proposal h3 {  
 margin-top: 0;  
 }  
 .proposal p {  
 margin: 5px 0;  
 }  
 input[type="submit"] {  
 padding: 10px 20px;  
 background-color: #008CBA;  
 color: #fff;  
 border: none;  
 border-radius: 5px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
 }  
 input[type="submit"]:hover {  
 background-color: #005f7f;  
 }  
 </style>  
</head>  
<body>  
<%  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 response.sendError(500, "ви не ввійшли");  
 }  
  
 String id = request.getParameter("id");  
  
 if (id != null && !id.isEmpty()) {  
 UUID tenderId = UUID.*fromString*(request.getParameter("id"));  
 TendersDAO tendersDAO = (TendersDAO) request.getServletContext().getAttribute("tendersDataBase");  
 Tender tender = tendersDAO.getTenderById(tenderId);  
  
 if (tender == null) {  
 response.sendError(500, "Тендер за таким id не знайдено");  
 } else {  
 request.setAttribute("tender", tender);  
 }  
  
 assert tender != null;  
  
 if (user != tender.getAuthor()) {  
 response.sendError(500, "ви не є власником тендеру");  
 return;  
 }  
 } else {  
 response.sendError(500, "Id тендеру відсутнє");  
 return;  
 }  
%>  
<div class="container">  
 <div class="header">  
 <**c:if** test="${not empty user}">  
 <p>Ви увійшли як: ${user.getLogin()}</p>  
 <a href="userAccount.jsp">Кабінет користувача</a>  
 </**c:if**>  
 <**c:if** test="${empty user}">  
 <a href="loginPage.jsp">Увійти</a>  
 </**c:if**>  
 </div>  
 <h1>Кандидати в переможці</h1>  
 <**c:if** test="${not empty tender.tenderProposals}">  
 <**c:forEach** var="proposal" items="${tender.tenderProposals}">  
 <div class="proposal">  
 <h3>${proposal.companyName}</h3>  
 <p>${proposal.proposalDetails}</p>  
 <p>${proposal.author.login}</p>  
 <p><**fmt:formatNumber** value="${proposal.price}" type="currency" currencySymbol="₴"/></p>  
 <form action="chooseWinner" method="post">  
 <input type="hidden" name="winnerId" value="${proposal.id}">  
 <input type="hidden" name="tenderId" value="${tender.id}">  
 <input type="submit" value="Вибрати переможцем">  
 </form>  
 </div>  
 </**c:forEach**>  
 </**c:if**>  
 <**c:if** test="${empty tender.tenderProposals}">  
 <h3>Немає тендерних пропозицій</h3>  
 </**c:if**>  
</div>  
</body>  
</html>

createProposal.jsp (сторінка для створення пропозиції):

<%@ **page** import="java.util.UUID" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.User" %>  
<%@ **page** contentType="text/html;charset=UTF-8" %>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <title>Створення тендерної пропозиції</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 max-width: 600px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);  
 }  
 h1 {  
 text-align: center;  
 color: #fff;  
 }  
 form {  
 margin-top: 20px;  
 }  
 label {  
 display: block;  
 margin-bottom: 5px;  
 color: #bbb;  
 }  
 input[type="text"],  
 textarea,  
 input[type="number"] {  
 width: calc(100% - 22px);  
 padding: 10px;  
 margin-bottom: 10px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 background-color: #444;  
 color: #fff;  
 box-sizing: border-box;  
 }  
 textarea {  
 resize: vertical;  
 }  
 button[type="submit"] {  
 background-color: #008CBA;  
 color: #fff;  
 border: none;  
 padding: 10px 20px;  
 border-radius: 5px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
 }  
 button[type="submit"]:hover {  
 background-color: #005f7f;  
 }  
 .error {  
 color: red;  
 margin-top: 5px;  
 }  
 </style>  
</head>  
<body>  
<%  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 request.setAttribute("addressing", "createProposal");  
 request.setAttribute("tenderId", UUID.*fromString*(request.getParameter("id")));  
 request.getRequestDispatcher("loginPage.jsp").forward(request, response);  
 return;  
 }  
%>  
<div class="container">  
 <h1>Створення тендерної пропозиції</h1>  
 <form action="createProposal" method="post">  
 <input type="hidden" name="id" value="${param.id}">  
 <label for="companyName">Назва компанії:</label>  
 <input type="text" id="companyName" name="companyName" required>  
 <label for="proposalDetails">Деталі пропозиції:</label>  
 <textarea id="proposalDetails" name="proposalDetails" rows="4" required></textarea>  
 <label for="price">Ціна:</label>  
 <input type="number" id="price" name="price" step="1000.0" required>  
 <span class="error">${errorProposal}</span><br>  
 <button type="submit">Створити пропозицію</button>  
 </form>  
</div>  
</body>  
</html>

createTender.jsp (сторінка для створення тендеру):

<%@ **page** import="com.coursework.coursework.ServiceLayer.User" %>  
<%@ **page** contentType="text/html;charset=UTF-8" %>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <title>Створення тендеру</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 max-width: 600px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);  
 }  
 h1 {  
 text-align: center;  
 color: #fff;  
 }  
 form {  
 margin-top: 20px;  
 }  
 label {  
 display: block;  
 margin-bottom: 5px;  
 color: #bbb;  
 }  
 input[type="text"],  
 textarea,  
 input[type="number"],  
 input[type="date"] {  
 width: calc(100% - 22px);  
 padding: 10px;  
 margin-bottom: 10px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 background-color: #444;  
 color: #fff;  
 box-sizing: border-box;  
 }  
 textarea {  
 resize: vertical;  
 }  
 button[type="submit"] {  
 background-color: #008CBA;  
 color: #fff;  
 border: none;  
 padding: 10px 20px;  
 border-radius: 5px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
 }  
 button[type="submit"]:hover {  
 background-color: #005f7f;  
 }  
 </style>  
</head>  
<body>  
<%  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 request.setAttribute("addressing", "createTender");  
 request.getRequestDispatcher("loginPage.jsp").forward(request, response);  
 return;  
 }  
%>  
<div class="container">  
 <h1>Створення тендеру</h1>  
 <form action="createTender" method="post">  
 <label for="tenderName">Назва тендеру:</label>  
 <input type="text" id="tenderName" name="tenderName" required>  
 <label for="tenderDescription">Опис тендеру:</label>  
 <textarea id="tenderDescription" name="tenderDescription" rows="4" required></textarea>  
 <label for="tenderDeadline">Кінцевий термін:</label>  
 <input type="date" id="tenderDeadline" name="tenderDeadline" required>  
 <label for="tenderCost">Вартість:</label>  
 <input type="number" id="tenderCost" name="tenderCost" step="1000.0" required>  
 <button type="submit">Створити тендер</button>  
 </form>  
</div>  
</body>  
</html>

createTenderReview.jsp (сторінка для створення відгуку):

<%@ **page** import="com.coursework.coursework.ServiceLayer.User" %>  
<%@ **page** import="java.util.UUID" %>  
<%@ **page** contentType="text/html;charset=UTF-8" %>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <title>Відгук на тендер</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 max-width: 600px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);  
 }  
 h1 {  
 text-align: center;  
 color: #fff;  
 }  
 form {  
 margin-top: 20px;  
 }  
 label {  
 display: block;  
 margin-bottom: 5px;  
 color: #bbb;  
 }  
 input[type="text"],  
 textarea,  
 input[type="number"],  
 input[type="date"] {  
 width: calc(100% - 22px);  
 padding: 10px;  
 margin-bottom: 10px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 background-color: #444;  
 color: #fff;  
 box-sizing: border-box;  
 }  
 textarea {  
 resize: vertical;  
 }  
 button[type="submit"] {  
 background-color: #008CBA;  
 color: #fff;  
 border: none;  
 padding: 10px 20px;  
 border-radius: 5px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
 }  
 button[type="submit"]:hover {  
 background-color: #005f7f;  
 }  
 </style>  
</head>  
<body>  
<%  
 String id = request.getParameter("id");  
  
 if (id != null && !id.isEmpty()) {  
 UUID tenderId = UUID.*fromString*(id);  
 request.setAttribute("tenderId", tenderId);  
 } else {  
 response.sendError(500, "Id тендеру відсутнє");  
 return;  
 }  
  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 request.setAttribute("addressing", "review");  
 request.setAttribute("tenderId", UUID.*fromString*(request.getParameter("id")));  
 request.getRequestDispatcher("loginPage.jsp").forward(request, response);  
 return;  
 }  
%>  
<div class="container">  
 <h1>Відгук на тендер</h1>  
 <form action="createReview" method="get">  
 <input type="hidden" id="tenderId" name="tenderId" value="${tenderId}">  
 <label for="companyName">Назва компанії:</label>  
 <input type="text" id="companyName" name="companyName" required>  
 <label for="review">Перелік власних компетенцій:</label>  
 <textarea id="review" name="review" rows="4" required></textarea>  
 <label for="phoneNumber">Номер телефону:</label>  
 <input type="text" id="phoneNumber" name="phoneNumber" required>  
 <button type="submit">Надіслати відгук</button>  
 </form>  
</div>  
</body>  
</html>

editProposal.jsp (сторінка для редагування пропозиції):

<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ **page** import="java.util.UUID" %>  
<%@ **page** import="com.coursework.coursework.DAOs.TendersDAO" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.Tender" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.User" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.TenderProposal" %>  
<%@ **page** contentType="text/html;charset=UTF-8" %>  
<html>  
<head>  
 <title>Редагування пропозиції</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 max-width: 600px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);  
 }  
 h2 {  
 text-align: center;  
 color: #fff;  
 }  
 form {  
 margin-top: 20px;  
 }  
 label {  
 display: block;  
 margin-bottom: 5px;  
 color: #bbb;  
 }  
 input[type="text"],  
 textarea,  
 input[type="number"],  
 input[type="date"] {  
 width: calc(100% - 22px);  
 padding: 10px;  
 margin-bottom: 10px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 background-color: #444;  
 color: #fff;  
 box-sizing: border-box;  
 }  
 textarea {  
 resize: vertical;  
 }  
 button[type="submit"], button[type="reset"] {  
 background-color: #008CBA;  
 color: #fff;  
 border: none;  
 padding: 10px 20px;  
 border-radius: 5px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
 margin-right: 10px;  
 }  
 button[type="submit"]:hover, button[type="reset"]:hover {  
 background-color: #005f7f;  
 }  
 .delete-button {  
 background-color: #ff4d4d;  
 }  
 .delete-button:hover {  
 background-color: #cc0000;  
 }  
 </style>  
</head>  
<body>  
<%  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 response.sendError(500, "Ви не ввійшли");  
 return;  
 }  
  
 String proposalId = request.getParameter("proposalId");  
 String id = request.getParameter("id");  
  
 if (id != null && !id.isEmpty() && proposalId != null && !proposalId.isEmpty()) {  
 UUID tenderId = UUID.*fromString*(id);  
 UUID propId = UUID.*fromString*(proposalId);  
 TendersDAO tendersDAO = (TendersDAO) request.getServletContext().getAttribute("tendersDataBase");  
 Tender tender = tendersDAO.getTenderById(tenderId);  
  
 if (tender == null) {  
 response.sendError(500, "Тендер за даним id не знайдено");  
 return;  
 }  
  
 request.setAttribute("tender", tender);  
 TenderProposal proposal = tender.findProposalById(propId);  
  
 if (proposal == null) {  
 response.sendError(500, "Пропозиція за даним id не знайдена");  
 return;  
 }  
 request.setAttribute("proposal", proposal);  
  
 } else {  
 response.sendError(500, "Id тендеру відсутнє");  
 return;  
 }  
%>  
  
<div class="container">  
 <h2>Редагування пропозиції</h2>  
 <form action="editProposal" method="post">  
 <input type="hidden" name="tenderId" value="${tender.id}">  
 <input type="hidden" name="proposalId" value="${proposal.id}">  
 <label for="companyName">Назва компанії:</label>  
 <input type="text" id="companyName" name="companyName" value="${proposal.companyName}" required>  
 <label for="proposalDetails">Деталі пропозиції:</label>  
 <textarea id="proposalDetails" name="proposalDetails" rows="4" required>${proposal.proposalDetails}</textarea>  
 <label for="price">Ціна:</label>  
 <input type="number" id="price" name="price" step="1000.0" value="${proposal.price}" required>  
 <div style="text-align: center;">  
 <button type="submit">Зберегти зміни</button>  
 <button type="reset">Скинути</button>  
 </div>  
 </form>  
  
 <form action="editProposal" method="get" onsubmit="return confirm('Ви впевнені, що хочете видалити цю пропозицію?');">  
 <input type="hidden" name="tenderId" value="${tender.id}">  
 <input type="hidden" name="proposalId" value="${proposal.id}">  
 <div style="text-align: center; margin-top: 10px;">  
 <button type="submit" class="delete-button">Видалити</button>  
 </div>  
 </form>  
  
</div>  
  
</body>  
</html>

editTender.jsp (сторінка для редагування тендеру):

<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ **page** import="java.util.UUID" %>  
<%@ **page** import="com.coursework.coursework.DAOs.TendersDAO" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.Tender" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.User" %>  
<%@ **page** contentType="text/html;charset=UTF-8" %>  
<html>  
<head>  
 <title>Редагування тендеру</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 max-width: 800px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 }  
 .header a {  
 color: #fff;  
 text-decoration: none;  
 padding: 10px 20px;  
 background-color: #444;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 display: inline-block;  
 margin-bottom: 10px;  
 }  
 .header a:hover {  
 background-color: #555;  
 }  
 form {  
 margin-top: 20px;  
 text-align: center;  
 }  
 label {  
 display: block;  
 margin-bottom: 5px;  
 color: #ccc;  
 text-align: left;  
 }  
 input[type="text"],  
 textarea,  
 input[type="number"],  
 input[type="date"] {  
 width: calc(100% - 24px);  
 padding: 10px;  
 margin-bottom: 20px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 background-color: #444;  
 color: #fff;  
 box-sizing: border-box;  
 }  
 textarea {  
 resize: vertical;  
 }  
 button {  
 background-color: #008CBA;  
 color: #fff;  
 border: none;  
 padding: 10px 20px;  
 border-radius: 5px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
 margin: 5px 0;  
 }  
 button:hover {  
 background-color: #005f7f;  
 }  
 </style>  
</head>  
<body>  
<%  
 User user = (User) request.getSession().getAttribute("user");  
  
 if (user == null) {  
 response.sendError(500, "ви не ввійшли");  
 }  
  
 String id = request.getParameter("id");  
  
 if (id != null && !id.isEmpty()) {  
 UUID tenderId = UUID.*fromString*(request.getParameter("id"));  
 TendersDAO tendersDAO = (TendersDAO) request.getServletContext().getAttribute("tendersDataBase");  
 Tender tender = tendersDAO.getTenderById(tenderId);  
  
 if (tender == null) {  
 response.sendError(500, "Тендер за таким id не знайдено");  
 } else {  
 request.setAttribute("tender", tender);  
 }  
  
 assert tender != null;  
  
 if (user != tender.getAuthor()) {  
 response.sendError(500, "ви не є власником тендеру");  
 }  
 } else {  
 response.sendError(500, "Id тендеру відсутнє");  
 return;  
 }  
%>  
  
<div class="container">  
 <h2>Редагування тендеру</h2>  
 <form action="editTender" method="post">  
 <input type="hidden" name="id" value="${tender.id}" />  
 <label for="name">Назва тендеру:</label>  
 <input type="text" id="name" name="name" value="${tender.name}" required/><br/>  
 <label for="description">Опис тендеру:</label>  
 <textarea id="description" name="description" required>${tender.description}</textarea><br/>  
 <label for="deadline">Кінцевий термін:</label><br>  
 <input type="date" id="deadline" name="deadline" value="${tender.deadline}" required><br>  
 <label for="cost">Сума:</label>  
 <input type="number" id="cost" name="cost" value="${tender.cost}" required/><br/>  
 <button type="submit">Зберегти</button>  
 </form>  
  
 <form action="editTender" method="get" onsubmit="return confirm('Ви впевнені, що хочете видалити цей тендер?');">  
 <input type="hidden" name="id" value="${tender.id}" />  
 <button type="submit">Видалити</button>  
 </form>  
  
 <form action="changeTenderStatus" method="get">  
 <input type="hidden" name="id" value="${tender.id}" />  
 <button type="submit">  
 <**c:if** test="${tender.status eq 'INACTIVE'}">  
 Активувати  
 <input type="hidden" name="newStatus" value="activate" />  
 </**c:if**>  
 <**c:if** test="${tender.status eq 'ACTIVE'}">  
 Деактивувати  
 <input type="hidden" name="newStatus" value="deactivate" />  
 </**c:if**>  
 </button>  
 </form>  
</div>  
  
</body>  
</html>

editUser.jsp (сторінка для редагування користувача):

<%@ **page** contentType="text/html;charset=UTF-8" %>  
<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <title>Редагування облікового запису</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 max-width: 600px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);  
 }  
 h1, h3 {  
 text-align: center;  
 color: #fff;  
 }  
 form {  
 margin-top: 20px;  
 }  
 label {  
 display: block;  
 margin-bottom: 5px;  
 color: #bbb;  
 }  
 input[type="text"],  
 input[type="password"] {  
 width: calc(100% - 22px);  
 padding: 10px;  
 margin-bottom: 10px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 background-color: #444;  
 color: #fff;  
 box-sizing: border-box;  
 }  
 .message, .error {  
 display: block;  
 margin-bottom: 10px;  
 color: #0f0;  
 }  
 .error {  
 color: #f00;  
 }  
 button[type="submit"] {  
 background-color: #008CBA;  
 color: #fff;  
 border: none;  
 padding: 10px 20px;  
 border-radius: 5px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
 }  
 button:hover {  
 background-color: #005f7f;  
 }  
 .return-button {  
 background-color: #444;  
 color: #fff;  
 text-decoration: none;  
 border-radius: 5px;  
 padding: 10px 20px;  
 display: inline-block;  
 transition: background-color 0.3s;  
 margin-top: 20px;  
 }  
  
 .return-button:hover {  
 background-color: #555;  
 }  
 </style>  
</head>  
<body>  
<div class="container">  
 <h1>Редагування облікового запису</h1>  
  
 <form action="updateUsername" method="post">  
 <h3>Редагування логіна</h3>  
 <label for="newUsername">Новий логін:</label>  
 <input type="text" id="newUsername" name="newUsername" required>  
 <span class="message">${usernameMessage}</span>  
 <span class="error">${usernameError}</span>  
 <button type="submit">Оновити логін</button>  
 </form>  
  
 <form action="updatePassword" method="post">  
 <h3>Редагування пароля</h3>  
 <label for="newPassword">Новий пароль:</label>  
 <input type="text" id="newPassword" name="newPassword" required>  
 <span class="message">${passwordMessage}</span>  
 <span class="error">${passwordError}</span>  
 <button type="submit">Оновити пароль</button>  
 </form>  
  
 <h3>Дії з обліковим записом</h3>  
 <form action="exitUser" method="get">  
 <button type="submit">Вийти з облікового запису</button>  
 </form>  
 <form action="deleteUser" method="get">  
 <button type="submit">Видалити обліковий запис</button>  
 </form>  
  
 <a href="userAccount.jsp" class="button return-button">Повернутися на сторінку користувача</a>  
  
</div>  
</body>  
</html>

homepage.jsp (головна сторінка):

<%@ **page** contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ **taglib** prefix="fmt" uri="jakarta.tags.fmt" %>  
<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Домашня сторінка</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 max-width: 800px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 }  
 .header {  
 text-align: right;  
 margin-bottom: 20px;  
 }  
 .header a {  
 color: #fff;  
 text-decoration: none;  
 padding: 10px 20px;  
 background-color: #444;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 display: inline-block;  
 margin-bottom: 10px;  
 }  
 .header a:hover {  
 background-color: #555;  
 }  
 .create-tender-button {  
 text-align: right;  
 margin-bottom: 20px;  
 }  
 .create-tender-button a {  
 color: #fff;  
 text-decoration: none;  
 padding: 10px 20px;  
 background-color: #008CBA;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 }  
 .create-tender-button a:hover {  
 background-color: #005f7f;  
 }  
 form {  
 text-align: center;  
 margin-bottom: 40px;  
 }  
 input[type="text"] {  
 width: calc(100% - 24px);  
 padding: 10px;  
 margin-bottom: 20px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 background-color: #444;  
 color: #fff;  
 }  
 input[type="submit"] {  
 padding: 10px 20px;  
 background-color: #008CBA;  
 color: #fff;  
 border: none;  
 border-radius: 5px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
 }  
 input[type="submit"]:hover {  
 background-color: #005f7f;  
 }  
 .tenders {  
 margin-bottom: 40px;  
 }  
 .tender {  
 padding: 10px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 margin-bottom: 20px;  
 background-color: #444;  
 }  
 .tender h2 {  
 margin-top: 0;  
 }  
 .tender p {  
 margin: 5px 0;  
 }  
 .tender a {  
 color: #fff;  
 text-decoration: none;  
 padding: 5px 10px;  
 background-color: #008CBA;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 }  
 .tender a:hover {  
 background-color: #005f7f;  
 }  
 </style>  
</head>  
<body>  
<div class="container">  
 <div class="header">  
 <**c:if** test="${not empty user}">  
 <p>Ви увійшли як: ${user.getLogin()}</p>  
 <a href="userAccount.jsp">Кабінет користувача</a>  
 </**c:if**>  
 <**c:if** test="${empty user}">  
 <a href="loginPage.jsp">Увійти</a>  
 </**c:if**>  
 <div class="create-tender-button">  
 <a href="createTender.jsp">Створити тендер</a>  
 </div>  
 </div>  
  
 <form action="searchTender" method="GET">  
 <label for="keyword"></label>  
 <input type="text" id="keyword" name="keyword" placeholder="Введіть ключові слова">  
 <input type="submit" value="Пошук">  
 </form>  
  
 <h1>Список тендерів</h1>  
 <div class="tenders">  
 <**c:forEach** var="tenderEntry" items="${tenders}">  
 <div class="tender">  
 <**c:if** test="${tenderEntry.value.status eq 'ACTIVE'}">  
 <**c:set** var="tenderId" value="${tenderEntry.key}" />  
 <h2>${tenderEntry.value.name}</h2>  
 <p>${tenderEntry.value.author.login}</p>  
 <p>${tenderEntry.value.description}</p>  
 <p><**fmt:formatNumber** value="${tenderEntry.value.cost}" type="currency" currencySymbol="₴"/></p>  
 <a href="tenderDetails.jsp?id=${tenderId}" class="button">Деталі тендеру</a>  
 <**c:if** test="${tenderEntry.value.author ne user and not tenderEntry.value.isAfterDeadline()}">  
 <a href="createProposal.jsp?id=${tenderId}" class="button">Створити пропозицію</a>  
 <a href="createTenderReview.jsp?id=${tenderId}" class="button">Відгукнутися</a>  
 </**c:if**>  
 <**c:if** test="${tenderEntry.value.author == user}">  
 <a href="editTender.jsp?id=${tenderId}" class="button">Редагувати ваш тендер</a>  
 </**c:if**>  
 </**c:if**>  
 </div>  
 </**c:forEach**>  
 </div>  
</div>  
</body>  
</html>

loginPage.jsp (сторінка для авторизації):

<%@ **page** contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <title>Вхід до акаунту</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 width: 300px;  
 margin: 50px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);  
 }  
 h2 {  
 text-align: center;  
 color: #ddd;  
 }  
 label {  
 color: #ddd;  
 }  
 input[type="text"], input[type="password"] {  
 width: calc(100% - 24px);  
 padding: 10px;  
 margin: 5px 0 20px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 background-color: #444;  
 color: #fff;  
 box-sizing: border-box;  
 }  
 input[type="submit"] {  
 width: 100%;  
 padding: 10px;  
 margin-top: 10px;  
 background-color: #008CBA;  
 color: white;  
 border: none;  
 border-radius: 5px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
 }  
 input[type="submit"]:hover {  
 background-color: #005f7f;  
 }  
 .button {  
 height: 35px;  
 width: 100%;  
 margin-top: 20px;  
 background-color: #008CBA;  
 color: #fff;  
 text-align: center;  
 text-decoration: none;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 display: flex;  
 align-items: center;  
 justify-content: center;  
 }  
 .button:hover {  
 background-color: #005f7f;  
 }  
 .error {  
 color: red;  
 margin-top: 5px;  
 }  
 .back-button {  
 margin-top: 20px;  
 background-color: #444;  
 }  
 .back-button:hover {  
 background-color: #555;  
 }  
 </style>  
</head>  
<body>  
<div class="container">  
 <h2>Вхід до акаунту</h2>  
 <form action="authorization" method="post">  
 <label for="username">Логін:</label><br>  
 <input type="text" id="username" name="username"><br>  
 <label for="password">Пароль:</label><br>  
 <input type="password" id="password" name="password"><br>  
 <span class="error">${errorMessage}</span><br>  
 <input type="hidden" id="tenderId" name="tenderId" value="${tenderId}">  
 <input type="hidden" id="addressing" name="addressing" value="${addressing}">  
 <input type="submit" id="login" value="Увійти">  
 </form>  
  
 <a href="registrationPage.jsp" class="button">Зареєструйтеся тут</a>  
 <form action="homePage" method="get">  
 <input type="submit" class="back-button" value="Повернутися на домашню сторінку">  
 </form>  
</div>  
</body>  
</html>

registrationPage.jsp (сторінка для реєстрації):

<%@ **page** contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  
<%@ **page** import="static com.coursework.coursework.ServiceLayer.PasswordGeneration.generatePassword" %>  
<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <title>Реєстрація</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 width: 300px;  
 margin: 50px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);  
 }  
 h2 {  
 text-align: center;  
 color: #ddd;  
 }  
 label {  
 color: #ddd;  
 }  
 input[type="text"], input[type="password"] {  
 width: calc(100% - 24px);  
 padding: 10px;  
 margin: 5px 0 20px;  
 border: 1px solid #555;  
 border-radius: 5px;  
 background-color: #444;  
 color: #fff;  
 box-sizing: border-box;  
 }  
 input[type="submit"] {  
 width: 100%;  
 padding: 10px;  
 margin-top: 10px;  
 background-color: #008CBA;  
 color: white;  
 border: none;  
 border-radius: 5px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
 }  
 input[type="submit"]:hover {  
 background-color: #005f7f;  
 }  
 .button {  
 height: 35px;  
 width: 100%;  
 margin-top: 20px;  
 background-color: #008CBA;  
 color: #fff;  
 text-align: center;  
 text-decoration: none;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 display: flex;  
 align-items: center;  
 justify-content: center;  
 }  
 .button:hover {  
 background-color: #005f7f;  
 }  
 .error {  
 color: red;  
 margin-top: 5px;  
 }  
 .message {  
 color: green;  
 margin-top: 5px;  
 }  
 </style>  
</head>  
<body>  
<div class="container">  
 <h2>Зареєструйте свій акаунт тут</h2>  
 <form action="registration" method="post">  
 <label for="username">Логін:</label><br>  
 <input type="text" id="username" name="username"><br>  
 <label for="password">Пароль:</label><br>  
 <input type="text" id="password" name="password" value="<%=*generatePassword*()%>"><br>  
 <span class="error">${errorMessage}</span><br>  
 <span class="message">${successMessage}</span><br>  
 <input type="submit" id="login" value="Зареєструватися">  
 </form>  
 <a href="loginPage.jsp" class="button">Ввійти</a>  
 <a href="homePage" class="button">Повернутися на домашню сторінку</a>  
</div>  
</body>  
</html>

searchedTenders.jsp (сторінка для виведення знайдених тендерів):

<%@ **page** contentType="text/html;charset=UTF-8" %>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ **taglib** prefix="fmt" uri="jakarta.tags.fmt" %>  
<html>  
<head>  
 <title>Пошук тендерів</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .header {  
 text-align: right;  
 margin: 20px;  
 }  
 .header a {  
 color: #fff;  
 text-decoration: none;  
 padding: 10px 20px;  
 background-color: #444;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 display: inline-block;  
 margin-bottom: 10px;  
 }  
 .header a:hover {  
 background-color: #555;  
 }  
 .button {  
 display: inline-block;  
 padding: 10px 20px;  
 margin-top: 10px;  
 background-color: #008CBA;  
 color: #fff;  
 text-decoration: none;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 }  
 .button:hover {  
 background-color: #005f7f;  
 }  
 .tenders {  
 margin: 20px auto;  
 max-width: 800px;  
 }  
 .tender {  
 padding: 20px;  
 margin-bottom: 20px;  
 background-color: #333;  
 border: 1px solid #555;  
 border-radius: 8px;  
 }  
 .tender h2 {  
 margin-top: 0;  
 color: #ddd;  
 }  
 .tender p {  
 margin: 5px 0;  
 color: #ddd;  
 }  
 h1 {  
 text-align: center;  
 color: #fff;  
 }  
 </style>  
</head>  
<body>  
<div class="header">  
 <**c:if** test="${not empty user}">  
 <p>Ви увійшли як: ${user.getLogin()}</p>  
 <a href="userAccount.jsp" class="button">Кабінет користувача</a>  
 </**c:if**>  
 <**c:if** test="${empty user}">  
 <a href="loginPage.jsp">Увійти</a>  
 </**c:if**>  
 <a href="homePage" class="button">Повернутися до списку тендерів</a>  
</div>  
  
<h1>Список знайдених тендерів</h1>  
<div class="tenders">  
 <**c:forEach** var="tender" items="${searchedTenders}">  
 <div class="tender">  
 <**c:set** var="tenderId" value="${tender.id}" />  
 <h2>${tender.name}</h2>  
 <p>${tender.author.login}</p>  
 <p>${tender.description}</p>  
 <p>${tender.deadline}</p>  
 <p><**fmt:formatNumber** value="${tender.cost}" type="currency" currencySymbol="₴"/></p>  
 <a href="tenderDetails.jsp?id=${tenderId}" class="button">Деталі тендеру</a>  
 <**c:if** test="${tender.author ne user and not tender.isAfterDeadline()}">  
 <a href="createProposal.jsp?id=${tenderId}" class="button">Створити пропозицію</a>  
 <a href="createTenderReview.jsp?id=${tenderId}" class="button">Відгукнутися</a>  
 </**c:if**>  
 <**c:if** test="${tender.author == user}">  
 <a href="editTender.jsp?id=${tenderId}" class="button">Редагувати ваш тендер</a>  
 </**c:if**>  
 </div>  
 </**c:forEach**>  
</div>  
</body>  
</html>

tenderDetails.jsp (сторінка для виведення інформації про тендер):

<%@ **page** contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ **taglib** prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ **page** import="com.coursework.coursework.DAOs.TendersDAO" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.Tender" %>  
<%@ **page** import="java.util.UUID" %>  
<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Деталі тендеру</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 max-width: 800px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 }  
 .header {  
 text-align: right;  
 margin-bottom: 20px;  
 }  
 .header a {  
 color: #fff;  
 text-decoration: none;  
 padding: 10px 20px;  
 background-color: #444;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 display: inline-block;  
 margin-bottom: 10px;  
 }  
 .header a:hover {  
 background-color: #555;  
 }  
 .tender, .proposal {  
 margin-bottom: 20px;  
 padding: 20px;  
 background-color: #444;  
 border: 1px solid #555;  
 border-radius: 8px;  
 }  
 .button {  
 display: inline-block;  
 padding: 10px 20px;  
 margin-top: 10px;  
 background-color: #008CBA;  
 color: #fff;  
 text-decoration: none;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 }  
 .button:hover {  
 background-color: #005f7f;  
 }  
 .message {  
 color: green;  
 margin-top: 5px;  
 }  
 </style>  
</head>  
<body>  
  
<%  
 String id = request.getParameter("id");  
  
 if (id != null && !id.isEmpty()) {  
 UUID tenderId = UUID.*fromString*(request.getParameter("id"));  
 TendersDAO tendersDAO = (TendersDAO) request.getServletContext().getAttribute("tendersDataBase");  
 Tender tender = tendersDAO.getTenderById(tenderId);  
  
 if (tender == null) {  
 response.sendError(500, "Тендер за таким id не знайдено");  
 return;  
 } else {  
 request.setAttribute("tender", tender);  
 }  
 } else {  
 response.sendError(500, "Id тендеру відсутнє");  
 return;  
 }  
%>  
  
<div class="container">  
 <div class="header">  
 <**c:if** test="${not empty user}">  
 <p>Ви увійшли як: ${user.login}</p>  
 <a href="userAccount.jsp" class="button">Кабінет користувача</a>  
 </**c:if**>  
 <**c:if** test="${empty user}">  
 <a href="loginPage.jsp">Увійти</a>  
 </**c:if**>  
 </div>  
  
 <h1>Деталі тендеру</h1>  
 <div class="tender">  
 <h2>${tender.name}</h2>  
 <p>${tender.description}</p>  
 <p>${tender.deadline}</p>  
 <p><**fmt:formatNumber** value="${tender.cost}" type="currency" currencySymbol="₴"/></p>  
 <p>${tender.status}</p>  
 <**c:if** test="${not empty user and tender.authorId == user.userId}">  
 <a href="checkReviews.jsp?id=${tender.id}" class="button">Переглянути відгуки</a>  
 <a href="editTender.jsp?id=${tender.id}" class="button">Редагувати ваш тендер</a>  
 <a href="generateURL?id=${tender.id}" class="button">Згенерувати посилання на тендер</a>  
 <p>${URL}</p>  
 </**c:if**>  
 </div>  
  
 <span class="message">${proposalSuccess}</span><br>  
 <span class="message">${successEditMessage}</span><br>  
 <span class="message">${successReviewMessage}</span><br>  
 <**c:if** test="${tender.author ne user and not tender.isAfterDeadline()}">  
 <a href="createProposal.jsp?id=${tender.id}" class="button">Створити пропозицію</a>  
 <a href="createTenderReview.jsp?id=${tender.id}" class="button">Відгукнутися</a>  
 </**c:if**>  
 <h2>Тендерні пропозиції</h2>  
 <div class="proposals">  
 <**c:if** test="${not empty tender.tenderProposals}">  
 <**c:forEach** var="proposal" items="${tender.tenderProposals}">  
 <div class="proposal">  
 <h3>${proposal.companyName}</h3>  
 <p>${proposal.proposalDetails}</p>  
 <p>${proposal.author.login}</p>  
 <p><**fmt:formatNumber** value="${proposal.price}" type="currency" currencySymbol="₴"/></p>  
 <**c:if** test="${user.userId eq proposal.author.userId}">  
 <a href="editProposal.jsp?proposalId=${proposal.id}&id=${proposal.tenderId}" class="button">Редагувати пропозицію</a>  
 </**c:if**>  
 </div>  
 </**c:forEach**>  
 </**c:if**>  
 <**c:if** test="${empty tender.tenderProposals}">  
 <h3>Немає тендерних пропозицій</h3>  
 </**c:if**>  
 </div>  
  
 <a href="homePage" class="button">Повернутися до списку тендерів</a>  
</div>  
</body>  
</html>

userAccount.jsp (сторінка користувача):

<%@ **page** contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ **taglib** prefix="fmt" uri="jakarta.tags.fmt" %>  
<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Сторінка користувача</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 }  
  
 .container {  
 max-width: 800px;  
 margin: 0 auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);  
 }  
  
 h2, h3 {  
 color: #008CBA;  
 }  
  
 label {  
 color: #008CBA;  
 }  
  
 input[type="text"], input[type="password"] {  
 width: 100%;  
 padding: 10px;  
 margin: 5px 0;  
 border: 1px solid #ccc;  
 border-radius: 3px;  
 box-sizing: border-box;  
 }  
  
 input[type="submit"] {  
 width: 100%;  
 padding: 10px;  
 margin-top: 10px;  
 background-color: #4CAF50;  
 color: white;  
 border: none;  
 border-radius: 3px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
 }  
  
 input[type="submit"]:hover {  
 background-color: #45a049;  
 }  
  
 .message {  
 color: green;  
 margin-top: 5px;  
 }  
  
 .button {  
 height: 30px;  
 width: 100%;  
 margin-top: 10px;  
 background-color: #008CBA;  
 color: #fff;  
 text-align: center;  
 text-decoration: none;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 display: flex;  
 align-items: center;  
 justify-content: center;  
 }  
  
 .button:hover {  
 background-color: #005f7f;  
 }  
  
 .tender, .proposal {  
 margin-bottom: 20px;  
 padding: 15px;  
 background-color: #444;  
 border: 1px solid #555;  
 border-radius: 8px;  
 }  
 </style>  
</head>  
<body>  
<div class="container">  
 <h2>Інформація про користувача</h2>  
 <p>Логін: ${user.login}</p>  
 <a href="editUser.jsp" class="button">Редагувати обліковий запис</a>  
 <h2>Тендери</h2>  
 <span class="message">${chooseWinnerSuccess}</span><br>  
  
 <form action="searchUserTenders" method="GET">  
 <label for="keyword"></label>  
 <input type="text" id="keyword" name="keyword" placeholder="Введіть ключові слова">  
 <input type="submit" value="Пошук">  
 </form>  
  
 <**c:if** test="${empty user.tenders}">  
 <h3>У вас поки немає тендерів</h3>  
 </**c:if**>  
 <**c:forEach** var="tenderEntry" items="${user.tenders}">  
 <div class="tender">  
 <**c:set** var="tenderId" value="${tenderEntry.key}" />  
 <h2>${tenderEntry.value.name}</h2>  
 <p>${tenderEntry.value.author.login}</p>  
 <p>${tenderEntry.value.description}</p>  
 <p><**fmt:formatNumber** value="${tenderEntry.value.cost}" type="currency" currencySymbol="₴"/></p>  
 <**c:if** test="${not empty tenderEntry.value.winnerProposal}">  
 <a href="winnerProposal.jsp?id=${tenderId}" class="button">Переможець тендеру</a>  
 </**c:if**>  
 <a href="tenderDetails.jsp?id=${tenderId}" class="button">Деталі тендеру</a>  
 <a href="checkReviews.jsp?id=${tenderId}" class="button">Переглянути відгуки</a>  
 <a href="editTender.jsp?id=${tenderId}" class="button">Редагувати ваш тендер</a>  
 <**c:if** test="${tenderEntry.value.isAfterDeadline() and empty tenderEntry.value.winnerProposal}">  
 <a href="chooseWinner.jsp?id=${tenderId}" class="button">Вибір переможця</a>  
 </**c:if**>  
 </div>  
 </**c:forEach**>  
  
 <h2>Тендерні пропозиції</h2>  
 <span class="message">${editProposalSuccess}</span><br>  
 <**c:if** test="${empty user.tenderProposals}">  
 <h3>У вас поки немає пропозицій</h3>  
 </**c:if**>  
 <**c:if** test="${not empty user.tenderProposals}">  
 <**c:forEach** var="proposalEntry" items="${user.tenderProposals}">  
 <div class="proposal">  
 <h2>${proposalEntry.value.companyName}</h2>  
 <p>${proposalEntry.value.proposalDetails}</p>  
 <p><**fmt:formatNumber** value="${proposalEntry.value.price}" type="currency" currencySymbol="₴"/></p>  
 <a href="editProposal.jsp?proposalId=${proposalEntry.value.id}&id=${proposalEntry.value.tenderId}" class="button">Редагувати пропозицію</a>  
 <**c:if** test="${not empty proposalEntry.value.chatId}">  
 <a href="chatPage.jsp?chatId=${proposalEntry.value.chatId}" class="button">Написати автору пропозиції</a>  
 </**c:if**>  
 </div>  
 </**c:forEach**>  
 </**c:if**>  
  
 <a href="homePage" class="button">Повернутися на домашню сторінку</a>  
</div>  
</body>  
</html>

winnerProposal.jsp (сторінка для перегляду переможця та створення чату):

<%@ **taglib** prefix="fmt" uri="jakarta.tags.fmt" %>  
<%@ **page** import="com.coursework.coursework.ServiceLayer.Tender" %>  
<%@ **page** import="com.coursework.coursework.DAOs.TendersDAO" %>  
<%@ **page** import="java.util.UUID" %>  
<%@ **page** contentType="text/html;charset=UTF-8" %>  
<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Переможець тендеру</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background-color: #222;  
 color: #ddd;  
 margin: 0;  
 padding: 0;  
 }  
 .container {  
 max-width: 800px;  
 margin: 20px auto;  
 padding: 20px;  
 background-color: #333;  
 border-radius: 8px;  
 }  
 .header a {  
 color: #fff;  
 text-decoration: none;  
 padding: 10px 20px;  
 background-color: #444;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 display: inline-block;  
 margin-bottom: 10px;  
 }  
 .header a:hover {  
 background-color: #555;  
 }  
 .button {  
 display: inline-block;  
 padding: 10px 20px;  
 margin-top: 10px;  
 background-color: #008CBA;  
 color: #fff;  
 text-decoration: none;  
 border-radius: 5px;  
 transition: background-color 0.3s;  
 }  
 .button:hover {  
 background-color: #005f7f;  
 }  
 </style>  
</head>  
<body>  
<%  
 String id = request.getParameter("id");  
  
 if (id != null && !id.isEmpty()) {  
 UUID tenderId = UUID.*fromString*(request.getParameter("id"));  
 TendersDAO tendersDAO = (TendersDAO) request.getServletContext().getAttribute("tendersDataBase");  
 Tender tender = tendersDAO.getTenderById(tenderId);  
  
 if (tender == null) {  
 response.sendError(500, "Тендер за таким id не знайдено");  
 return;  
 } else {  
 request.setAttribute("winnerProposal", tender.getWinnerProposal());  
 request.setAttribute("tenderName", tender.getName());  
 }  
 } else {  
 response.sendError(500, "Id тендеру відсутнє");  
 return;  
 }  
%>  
  
<div class="container">  
 <h1>Переможець тендеру "${tenderName}"</h1>  
 <div class="proposal">  
 <p>Компанія: ${winnerProposal.companyName}</p>  
 <p>Автор: ${winnerProposal.author.login}</p>  
 <p>Деталі пропозиції: ${winnerProposal.proposalDetails}</p>  
 <p>Ціна: <**fmt:formatNumber** value="${winnerProposal.price}" type="currency" currencySymbol="₴"/></p>  
 </div>  
 <a href="chatPage.jsp?chatId=${winnerProposal.chatId}" class="button">Написати автору пропозиції</a>  
 <a href="homePage" class="button">Повернутися до списку тендерів</a>  
</div>  
  
</body>  
</html>

# ВИСНОВКИ

Виконуючи дану курсову роботу на цю тематику було проаналізовано схожі за призначенням та функціональністю системи для проведення тендерів. Було виділено функціональні можливості, сутності, реалізація даної системи, що згодом використовувалося для написання схожого веб-застосунку з таким функціоналом.

Спочатку задачею стояло формування функціональних та нефункціональних вимог до системи, які в подальшому визначили вектор написання програмної реалізації цієї системи. Наступним кроком було визначення технологій, які б використовувалися у цій роботі. Мною було обрано мову Java як основу цього творіння, а IntelliJ IDEA – як зручну та наповнену функціоналом середу розробки програмного забезпечення. Для організації зберігання даних про користувачів, наявні тендери, тендерні пропозиції, відгуки та чати мною була обрана проста реалізація – використання об’єктів доступу до даних (DAO) та компонент Servlet API – WebListener. Це дозволило зробити фундамент для системи. Для захисту даних користувачів було написано класи для генерації та хешування паролів. При реєстрації в системі сервер надає можливість згенерувати надійний пароль, а далі – його хешувати за допомогою спеціального методу та зберігати цей хеш у якості пароля. Для генерації відповідей у формі графічного інтерфейсу було обрано HTML, технологія JSP та бібліотека JSTL для зручної роботи з динамічними сторінками.

Після написання веб-застосунку були описані можливі сценарії використання функціоналу цієї системи, а саме: реєстрація, перегляд активних тендерів, створення тендеру, його редагування, запуск та зупинка, створення тендерної пропозиції, її редагування та видалення, написання переліку компетенцій як відгук хазяїну тендеру, пошук тендерів за ключовими словами, пошук власних тендерів, генерація URL-посилання для переходу на деталі тендеру, вибір переможця тендеру та зворотній зв’язок між хазяїном тендеру та переможцем. Весь цей функціонал дозволяє повною мірою використовувати даний сервіс для проведення тендерів.

Далі, було розроблено загальну архітектуру системи. Завдяки шаблону проектування MVC, систему було розділено на логічні рівні. Структурований та правильно написаний код дозволяє легко додавати новий функціонал, або змінювати старий за потреби.

Потім було написано реалізацію кожного компоненту системи: контролерів, моделей, об’єктів доступу до даних та вигляд. Кожний компонент відповідає за свою функцію: контролери обробляють HTTP запити та генерують відповіді користувачу; моделі дають можливість створити потрібний нам об’єкт, наприклад, тендер; об’єкти доступу до даних дають можливість додавання, редагування чи видалення об’єктів, а також здійснювати пошук за ключовими словами; вигляд надав можливість красиво та охайно оформити графічний інтерфейс системи так, щоб ним зручно можна було користуватися.

Роблячи підсумки написаного вище, можна виокремити те, що ця система при правильному використанні та розширені функціоналу є доволі таки конкурентоспроможною. Додаючи новий функціонал, можна розширити можливості використання цієї системи та повністю ввійти в ринок проведення тендерів. Як для першої курсової, на мою особисту думку, це непоганий старт для загального розвитку в сфері написання веб-застосунків, проте мені є куди рости, адже це тільки початок дивовижної подорожі у цей світ.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Українська система проведення тендерів [Електронний ресурс] – Режим доступу до ресурсу: <https://prozorro.gov.ua/uk>
2. Опис загальної інформації про Servlet API [Електронний ресурс] – Режим доступу до ресурсу: <https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%BB%D0%B5%D1%82>
3. Java на стороні сервера [Електронний ресурс] – Режим доступу до ресурсу: <https://www.youtube.com/watch?v=gPBMJNq0_hQ>
4. Servlet API Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://tomcat.apache.org/tomcat-5.5-doc/servletapi/overview-summary.html>
5. Servlet API Documentation (Oracle) [Електронний ресурс] – Режим доступу до ресурсу: <https://docs.oracle.com/javaee%2F7%2Fapi%2F%2F/javax/servlet/Servlet.html>
6. Об’єктно-орієнтоване програмування [Електронний ресурс] – Режим доступу до ресурсу: <https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F>
7. Java Server Page (JSP) [Електронний ресурс] – Режим доступу до ресурсу: <https://uk.wikipedia.org/wiki/JSP>
8. Introduction to JSP [Електронний ресурс] – Режим доступу до ресурсу: <https://www.geeksforgeeks.org/introduction-to-jsp/>
9. HTML підручник [Електронний ресурс]. Режим доступу: [https://w3schoolsua.github.io/html/#gsc.tab=0](https://w3schoolsua.github.io/html/%23gsc.tab=0)
10. CSS підручник [Електронний ресурс]. Режим доступу: [https://w3schoolsua.github.io/css/#gsc.tab=0](https://w3schoolsua.github.io/css/%23gsc.tab=0)
11. [Електронний ресурс]. Режим доступу:

ДОДАТОК А  
Лістинг програми

[masonabor/CourseWork\_Martyniuk\_Yurii\_IA-33: Курсова робота (github.com)](https://github.com/masonabor/CourseWork_Martyniuk_Yurii_IA-33)

# ДОДАТОК Б Вигляд зі сторони користувача

Вигляд головної сторінки наведено на рис. Б.1

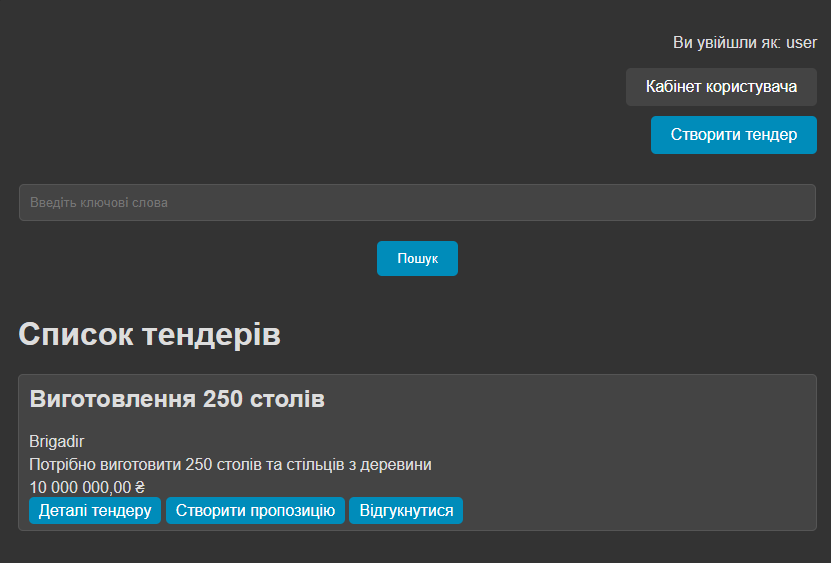


Рис Б.1 – Головна сторінка

Вигляд сторінки авторизації наведено на рис. Б.2

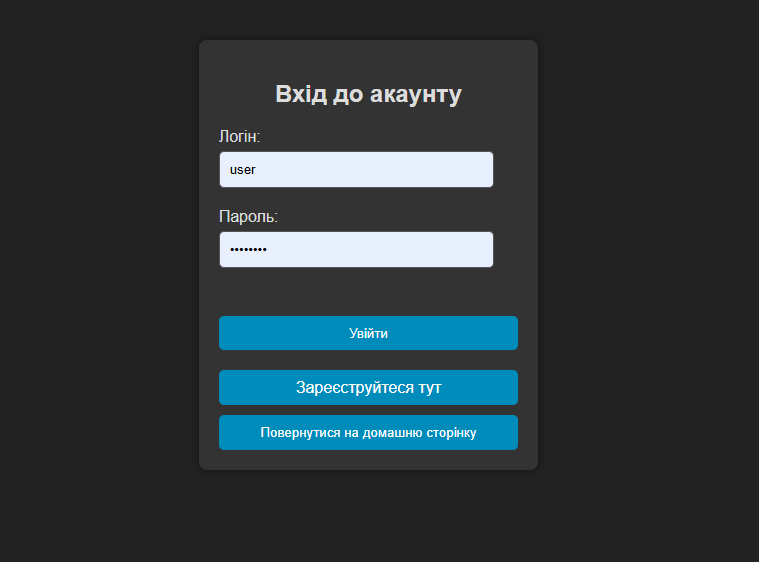


Рис Б.2 – Сторінка авторизації

Вигляд сторінки реєстрації наведено на рис. Б.3

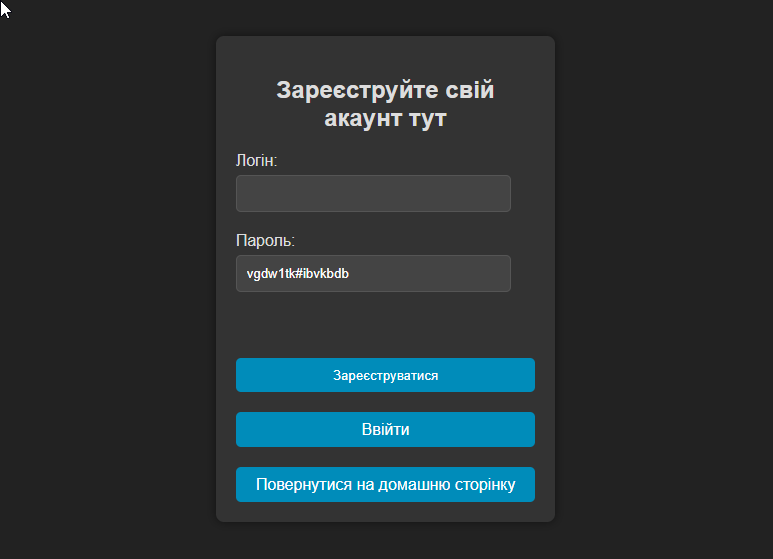


Рис Б.3 – Сторінка реєстрації

Вигляд сторінки користувача наведено на рис. Б.4

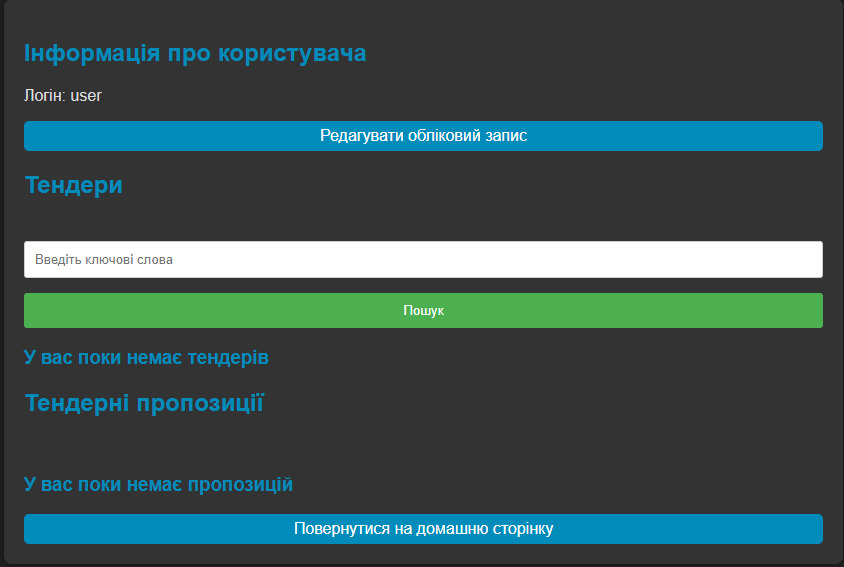


Рис Б.4 – Сторінка користувача

Вигляд сторінки з результатами пошуку наведено на рис. Б.5

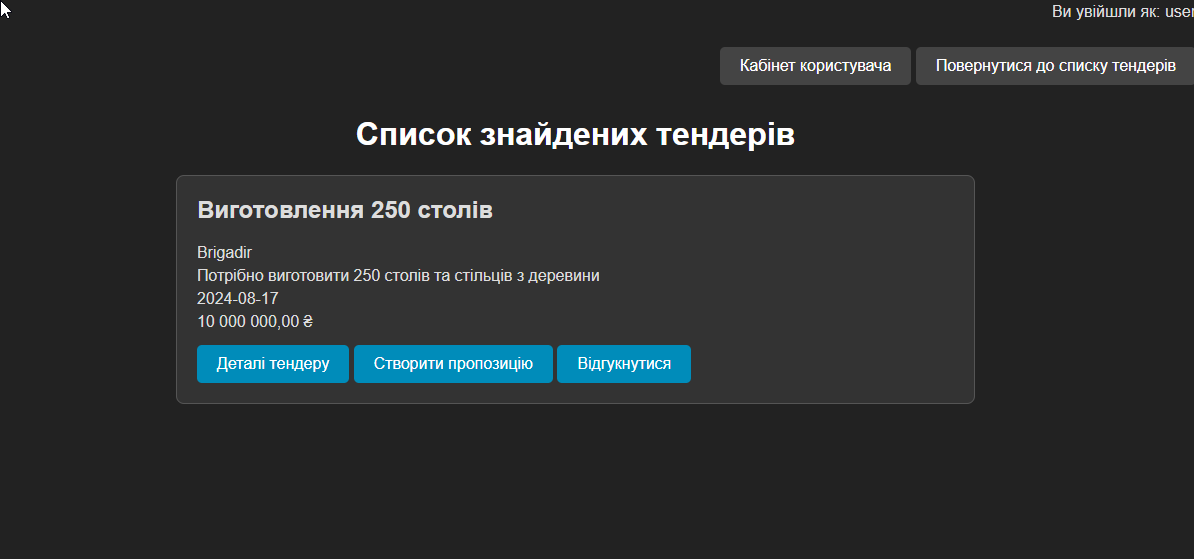


Рис Б.5 – Сторінка з результатами пошуку

Вигляд сторінки з деталями тендеру наведено на рис. Б.6

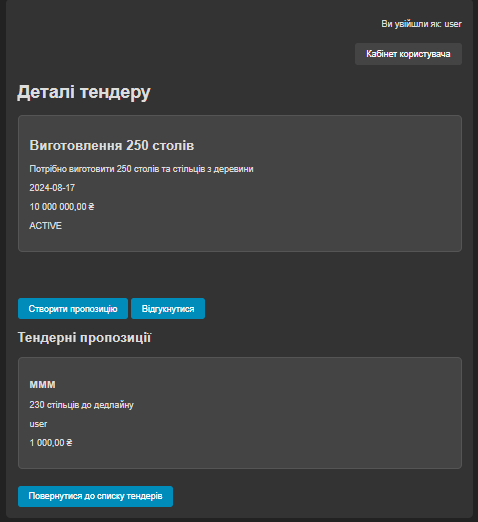


Рис Б.6 – Деталі тендеру

Вигляд сторінки створення тендерної пропозиції наведено на рис. Б.7

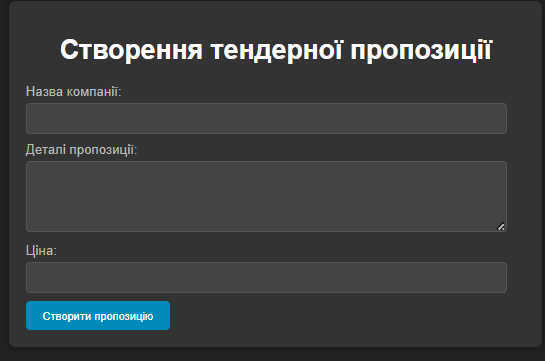


Рис Б.7 – Сторінка створення пропозиції

Вигляд сторінки створення відгуку наведено на рис. Б.8

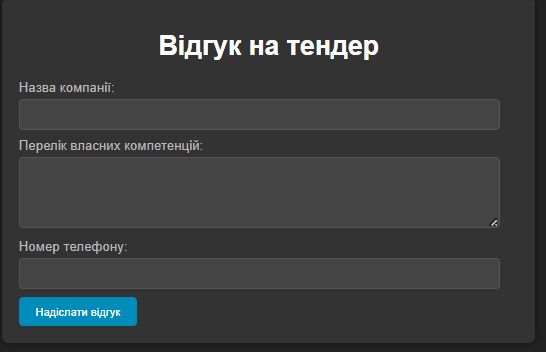


Рис Б.8 – Сторінка написання відгуку

Вигляд сторінки редагування пропозиції наведено на рис. Б.9

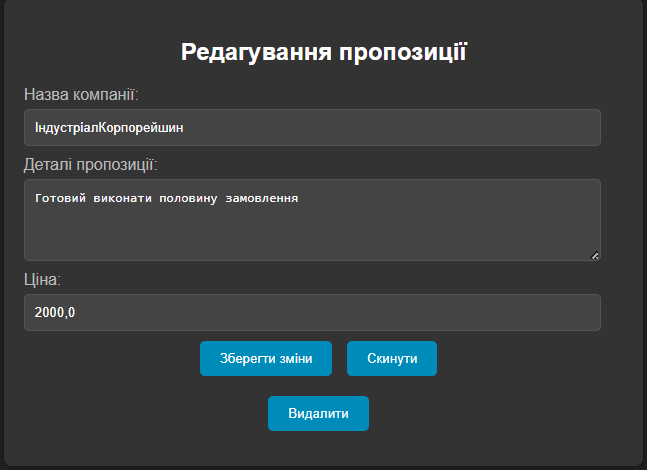


Рис Б.9 – Сторінка редагування пропозиції

Вигляд сторінки редагування користувача наведено на рис. Б.10

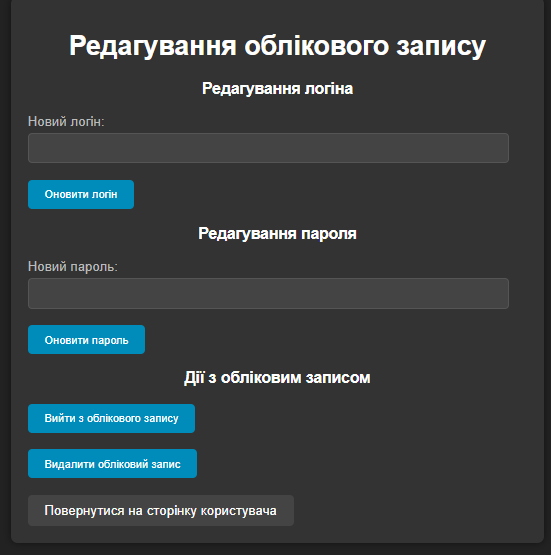


Рис Б.10 – сторінка редагування/виходу/видалення користувача

Вигляд сторінки виведення відгуків наведено на рис. Б.11

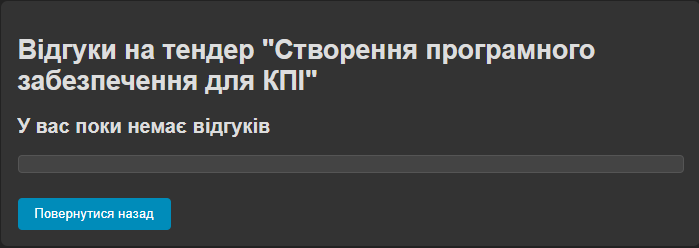


Рис Б.11 – Перегляд відгуків

Вигляд сторінки редагування тендеру наведено на рис. Б.12

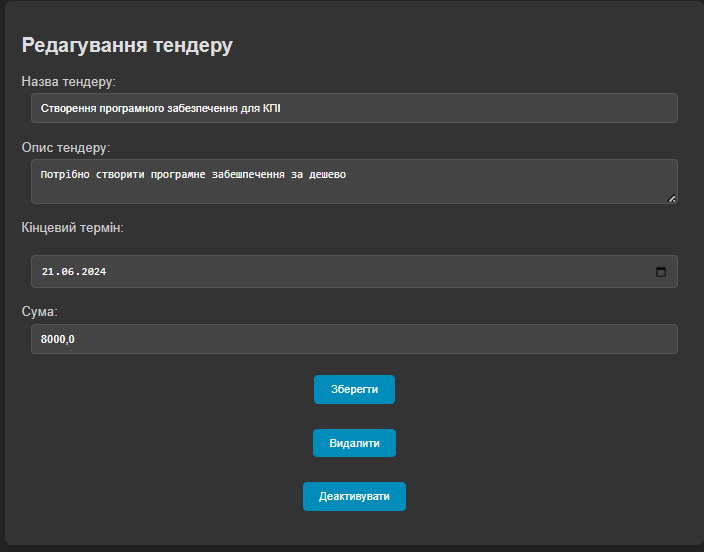


Рис Б.12 – Редагування тендеру

Вигляд сторінки створення тендеру наведено на рис. Б.13

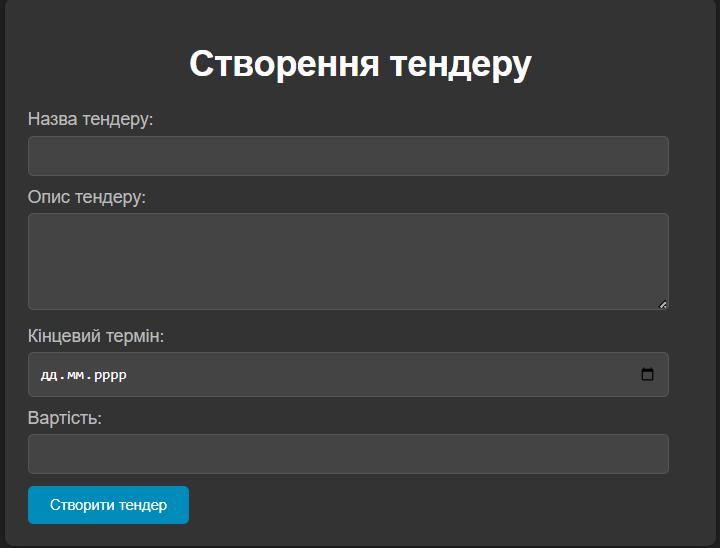


Рис Б.13 – Створення тендеру