ЗМІСТ

[1. ОПИС БАЗИ ПРАКТИКИ 3](#_Toc482225170)

[2. ЗАВДАННЯ ОТРИМАНЕ НА БАЗІ ПРАКТИКИ 4](#_Toc482225171)

[3. РЕЗУЛЬТАТ ВИКОНАННЯ ЗАВДАННЯ НА БАЗІ ПРАКТИКИ 5](#_Toc482225172)

[3.1. Ознайомлення з фреймворком для платформи Java – Spring Framework та його проектом Spring Security 5](#_Toc482225173)

[3.2. Java – застосунок з використанням вище описаних технологій 6](#_Toc482225174)

[4. ЗАВДАННЯ СТОСОВНО ДИПЛОМНОГО ПРОЕКТУ 8](#_Toc482225175)

[5. ОГЛЯД ВЕБ – ЗАСОБІВ ДЛЯ ОБРОБКИ РАСТРОВИХ ЗОБРАЖЕНЬ 9](#_Toc482225176)

[5.1. Теоритичні відомості 9](#_Toc482225177)

[5.2. Огляд продуктів аналогів 11](#_Toc482225178)

[6. ПОСТАНОВКА ЗАВДАННЯ РОЗРОБКИ ВЕБ-СЕРВІСУ ДЛЯ ОПРАЦЮВАННЯ, ПОКРАЩЕННЯ ТА ГЕОМЕТРИЧНОГО ПЕРЕТВОРЕННЯ ОКРЕМИХ ОБ’ЄКТІВ РАСТРОВИХ ЗОБРАЖЕНЬ 16](#_Toc482225179)

[6.1. Постановка завдання 16](#_Toc482225180)

[6.2. Вибір архітектури 16](#_Toc482225181)

[6.3. Вибір програмних засобів 17](#_Toc482225182)

[6.4. Специфікація вимог до програмного продукту 18](#_Toc482225183)

[7. РЕЗУЛЬТАТИ ПРОЕКТУВАННЯ ВЕБ СЕРВІСУ 26](#_Toc482225184)

[ВИСНОВОК 29](#_Toc482225185)

[ЛІТЕРАТУРА 30](#_Toc482225186)

[ДОДАТОК А 31](#_Toc482225187)

# ОПИС БАЗИ ПРАКТИКИ

Компанія «Едвантіс» була створена у 2005 році у Львові. Займається аутсорсинг-послугами для клієнтів з США та Західної Европи. «Едвантіс» було організовано в другій половині 2005 року на базі маленької фірми «BrainPower Software» (один невеличкий замовник, один-два програмісти), яка існувала з 2000 року. Місія компанії — досягнути та утримати найвищий професійний рівень ІТ-послуг задля максимального успіху корпоративних клієнтів у всьому світі; через інновації та цінності стартап-культури сприяти суспільному та індивідуальному розвитку. Головним чином компанія займається розробкою веб-сайтів та додатків для мобільних платформ. Для розробки серверної частини використовуються наступні технології: Java, .Net Framework, PHP, Python. Сервіси інтегруються з платформою Amazon Web Services. Команди розробників в компанії працюють за Scrum методологією. В компанії всі команди розробників підпорядковуються певному аккаунт менеджеру, який в свою чергу підпорядковується виконавчому директору з розробки.

# ЗАВДАННЯ ОТРИМАНЕ НА БАЗІ ПРАКТИКИ

Згідно переддипломної практики на базі практики ТзОВ «Едвантіс», було отримано наступні завдання:

1. Ознайомитися з фреймворком для платформи Java – Spring Framework та його проектом Spring Security.
2. Представити Java – застосунок з використанням вище описаних технологій.

# РЕЗУЛЬТАТ ВИКОНАННЯ ЗАВДАННЯ НА БАЗІ ПРАКТИКИ

## Ознайомлення з фреймворком для платформи Java – Spring Framework та його проектом Spring Security

Spring Framework — це програмний каркас (фреймворк) з відкритим кодом та контейнера з підтримкою інверсії управління для платформи Java. Основні особливості Spring Framework можуть бути використані будь-яким додатком Java, але є розширення для створення веб-додатків на платформі Java EE. Незважаючи на це, Spring Framework не нав'язує якоїсь конкретної моделі програмування, Spring Framework став популярним в спільноті Java як альтернатива, або навіть доповнення моделі Enterprise JavaBean (EJB).

Spring Framework складається з декількох модулів, які надають широкий спектр послуг:

* Контейнер Інверсії управління: Конфігурація компонентів додатків і управління життєвим циклом об'єктів Java, здійснюється головним чином через Інверсію управління.
* Аспектно-орієнтоване програмування: дозволяє реалізувати наскрізні процедури.
* Доступ до даних: робота з реляційною системою управління базами даних на платформі Java з використанням JDBC і об'єктно-реляційні відображення та інструментів з NoSQL баз даних.
* Управління транзакціями: об'єднує кілька API, управління транзакціями та координує операції для Java-об'єктів.
* Модель-Вигляд-Управління (Model-View-Controller): програмний каркас на основі HTTP сервлета, що забезпечує створення веб-додатків і веб-служб RESTful.
* Аутентифікація і авторизація: настроювані процеси безпеки, які підтримують цілий ряд стандартів, протоколів, інструментів і практик за допомогою підпроекту Spring Security (колишня система безпеки АсеґІ для Spring).
* Віддалене управління: конфігураційній вплив і управління Java-об'єктами для місцевої (локальної) або віддаленої конфігурації через JMX.
* Тестування: підтримка класів для написання юніт-тестів та інтеграційних тестів.

Spring Security – це фреймворк, який призначений для надання аутентифікації і авторизації корисувачів в Java - додатках. Як і всі Spring проекти, головною перевагою Spring Security є те, що вона може бути легко розширена для задоволення користувацьких вимог.

Головними особливостями Spring Security є:

* Всебічна і розширювана підтримка аутентифікації і авторизації користувача.
* Захист від атак типу session fixation, clickjacking, cross site request forgery.
* Інтеграція з Servlet API.
* Інтеграції з Spring Web MVC.

## Java – застосунок з використанням вище описаних технологій

Реалізацію даного Java – застосунку можна продемонструвати, використовуючи програму Postman. Ця програма призначена для здійснення http – запитів до контролера Java – застосунку.

На рис. 3.1 продемонстрований запит за шляхом <http://localhost:8080/login> за допомогою методу POST. Цей запит потрібний для здійснення логіну користувача. Отримуючи такий запит мій Java – застосунок генерує токен авторизації користувача. Частина коду застосунку можна знайти в додатку А.

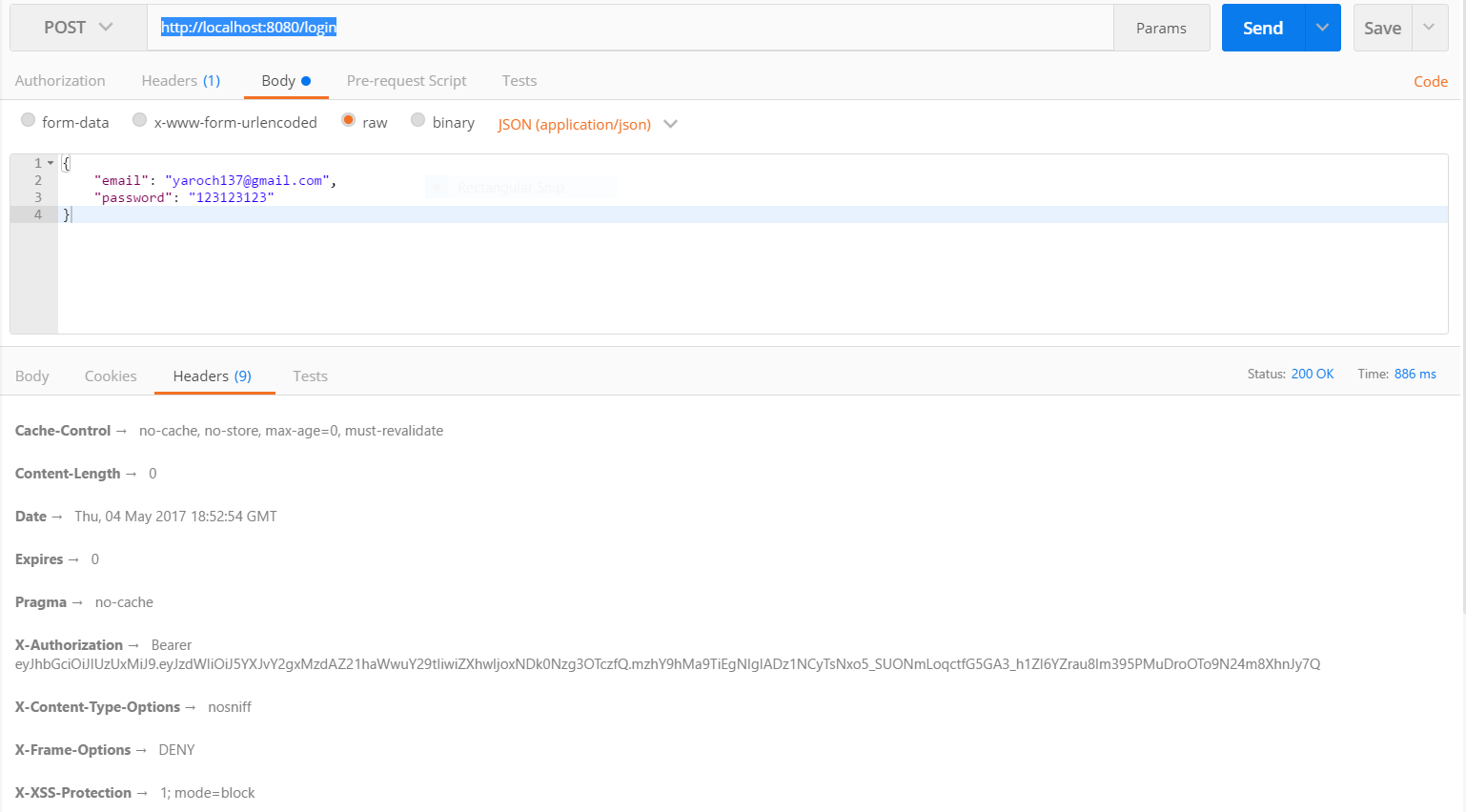


Рис. 3.1 Здійснення логін – запиту на контролер Java – застосунку

# ЗАВДАННЯ СТОСОВНО ДИПЛОМНОГО ПРОЕКТУ

Стосовно дипломного проекту було винесено наступні завдання:

1. Провести аналіз засобів опрацювання цифрових растрових зображень
2. Скласти специфікацію вимог до ПЗ
3. Обґрунтувати вибір технологій розробки
4. Прототип веб сервісу.

# ОГЛЯД ВЕБ – ЗАСОБІВ ДЛЯ ОБРОБКИ РАСТРОВИХ ЗОБРАЖЕНЬ

## Теоритичні відомості

Обробка зображень являє собою спосіб виконання деяких операцій над зображенням, для того, щоб отримати покращене зображення або корисну інформацію з нього. Це тип обробки, в якій вхідними даними є зображення, а вихідними даними можуть бути зображення або характеристики, пов'язані з цим зображенням. На сьогоднішній день, обробка зображень є однією із перспективних технологій. Ці технології утворюють основну область досліджень в інженерних і комп'ютерних дисциплінах.

В основному, обробка зображень включає в себе наступні три етапи:

* імпорт зображення за допомогою інструментів захоплення зображень;
* аналіз і опрацювання вхідних даних;
* отримані дані (оброблене зображення).

Існують два методи, які використовують для обробки зображень, а саме:

* аналоговий;
* цифровий.

Аналогова обробка зображень може використовуватися для паперових копій і фотографії. Цифрові засоби обробки зображень допомагають в опрацюванні вхідних даних за допомогою комп'ютерів.

Методи обробки зображень мають надзвичайно важливе значення в сучасній науці, вони є одними з таких які безперервно розвиваються та вдосконалюються. При цьому під обробкою зображень розуміють не лише поліпшення зорового сприйняття зображень, але й класифікацію об'єктів, що виконується при аналізі зображень [2].

Області застосування методів цифрової обробки в наш час значно розширюються, витісняючи аналогові методи обробки зображень. Методи цифрової обробки широко застосовуються в промисловості, мистецтві, медицині, космосі. Вони застосовуються при керуванні процесами, автоматизації виявлення об'єктів, розпізнаванні образів і в багатьох інших. Цифрова передача зображень із космічних апаратів, цифрові канали передачі сигналів зображень вимагають забезпечення передачі все більших потоків інформації. Формування зображень, поліпшення якості та автоматизація обробки медичних зображень, включаючи зображення, що створюються електронними мікроскопами, рентгенівськими апаратами, томографами тощо, є предметом сучасних досліджень та розробок [2].

Сьогодні важко представити область діяльності, у якій можна обійтися без комп'ютерної обробки зображень. При комп’ютерній обробці зображень вирішується наступні типові завдання:

* геометричні перетворення: обертання і масштабування;
* порівняння зображень;
* колірна корекція: зміна яскравості і контрасту, квантування кольору, перетворення в інший колірний простір;
* інтерполяція і згладжування;
* редагування та ретушування;
* поділ зображення на області (сегментація зображень);
* комбінування зображень різними способами;

На сьогоднішній день існує велика кількість розроблених алгоритмів для вирішення типових задач обробки зображень. Найпоширенішим алгоритмом покращення зображення є алгоритм вирівнювання гістограми цифрового зображення. Гістограма цифрового зображення - це графічне представлення тональних значень пікселів у межах зображення. Таким же чином, як ми можемо графічно представити зріст людей, які перебувають в аудиторії, можна побудувати графік яскравості пікселів у межах зображення. На гістограмі у вигляді графіка показано кількість пікселів зображення, які мають певну яскравість. Тобто, можна сказати, що зображення якісне, якщо його гістограма (гістограма яскравості пікселів) більш-менш рівномірна у всьому діапазоні горизонтальної осі x, яка охоплює значення від 0 до 255; діапазон шкали відповідає 8-бітній глибині кольору.

Основними алгоритмами геометричних перетворень зображення є застосування афінних перетворень на площині. Через свою простоту ці алгоритми широко використовуються в комп’ютерній графіці для геометричних перетворень, адже будь-яке афінне перетворення може бути представлене як послідовність операцій з числа найпростіших: зсув, масштабування, поворот. Кожна операція здійснюється за допомогою перетворення координат методом множення матриці координат зображення на матрицю, яка представляє певну операцію перетворення.

Для розпізнавання об’єктів на зображеннях широко застосовується алгоритм Кенні, який був розроблений австралійським вченим Джоном Кенні. Метою Кенні було розробити оптимальний алгоритм виявлення границь, що задовільняють трьом критеріям:

* гарне виявлення (Кенні трактував цю властивість як підвищення відношення сигнал/шум);
* гарна локалізація (вірне виявлення положення границі);
* єдиний відгук на одну границю.

Основними етапами алгоритму Кенні є:

* розмиття зображення для видалення шуму;
* пошук градієнтів. Границі відмічають там, де градієнт набуває найбільшого значення. Вони можуть мати різні напрямки, тому алгоритм Кенні використовує чотири фільтри для визначення горизонтальних, вертикальних і діагональних ребер в розмитому зображенні;
* пошук локальних максимумів;
* подвійна порогова фільтрація [5].

## Огляд продуктів аналогів

Першим продуктом аналогом було обрано веб-сервіс PIXLR Editor (рис. 5.1). Цей сервіс володіє величезною функціональністю для обробки зображень. Зразу після запуску веб-ресурсу користувач має змогу створити власне зображення для редагування або завантажити вже існуюче. Після вибору зображення користувач може здійснювати опрацювання зображення. Для опрацювання зображення веб-сервіс надає панель інструментів з набором доступних дій над зображенням, включаючи:

* обрізання зображення;
* перенесення окремих фрагментів зображення;
* виділення окремих фрагментів;
* малювання власних об’єктів та їх стирання;
* вставлення тексту на зображення;
* обертання, масштабування;
* зменшення фізичного розміру;
* тощо.

Також веб-сервіс надає можливість управління шарами зображення та перегляду історії маніпуляції над зображенням. Отже, цей веб-сервіс надає дуже широкий спектр можливостей для обробки зображень, але серед недоліків цього сервісу хотілося б відмітити відсутність можливості реєстрації та логіну користувача [1].

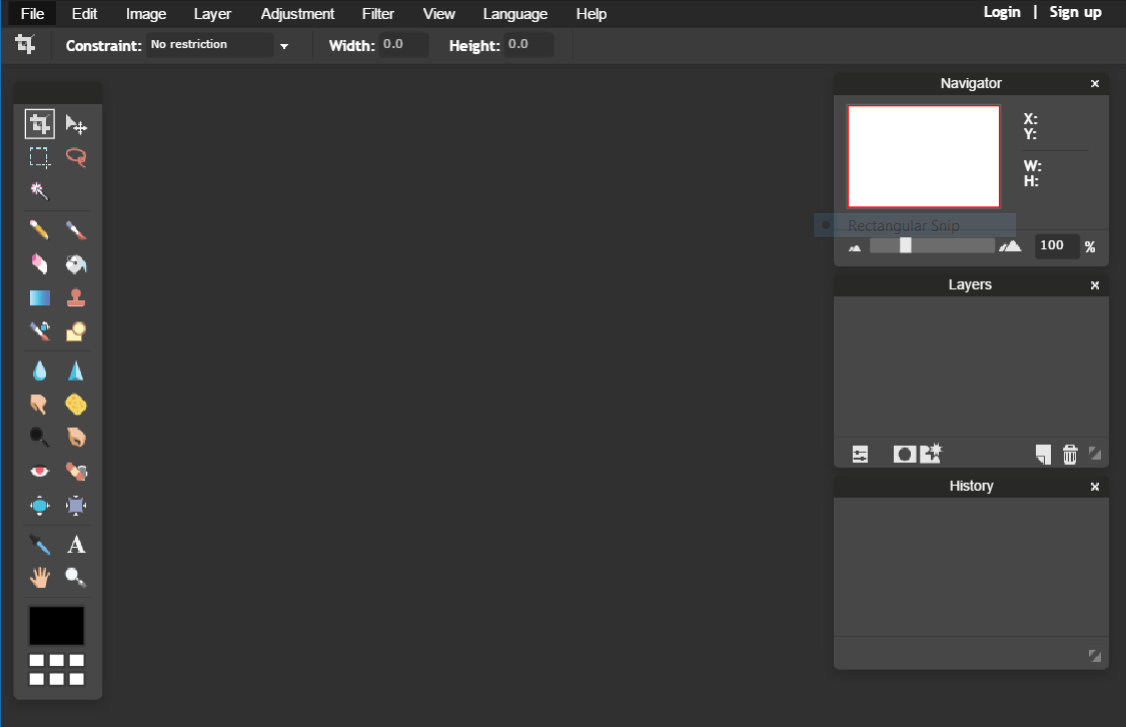


Рис. 5.1 Інтерфейс веб-ресурсу «pixlr.com»

Наступним продуктом – аналогом для огляду було обрано веб – сервіс Free Online Photo Editor (рис. 5.2). Спершу можна зазначити, що цей ресурс є безплатним і не дозволяє реєстрацій та логіну користувача. Користувач має змогу створити власне зображення або завантажити вже існуюче для редагування. На панелі інструментів також є широкий набір інструментів для використання.

Основними інструментами веб-сервісу є:

* геометричні перетворення: обертання, масштабування;
* задання яскравості на контрасту;
* зміна колірного простору зображення;
* зміна фізичного та геометричного розміру;
* застосування різних фільтрів над зображенням;
* стилізація країв зображення;
* можливість збереження в різних форматах;
* тощо.

Даний сервіс є дуже простим та зрозумілим у використанні, тут відсутня можливість виявлення окремих образів на зображеннях. Користувач після опрацювання зображення може поділитися результатами в соціальних мережах: Facebook, Twitter, Google+ та інших. Підтримка різних мов інтерфейсу користувача [2].

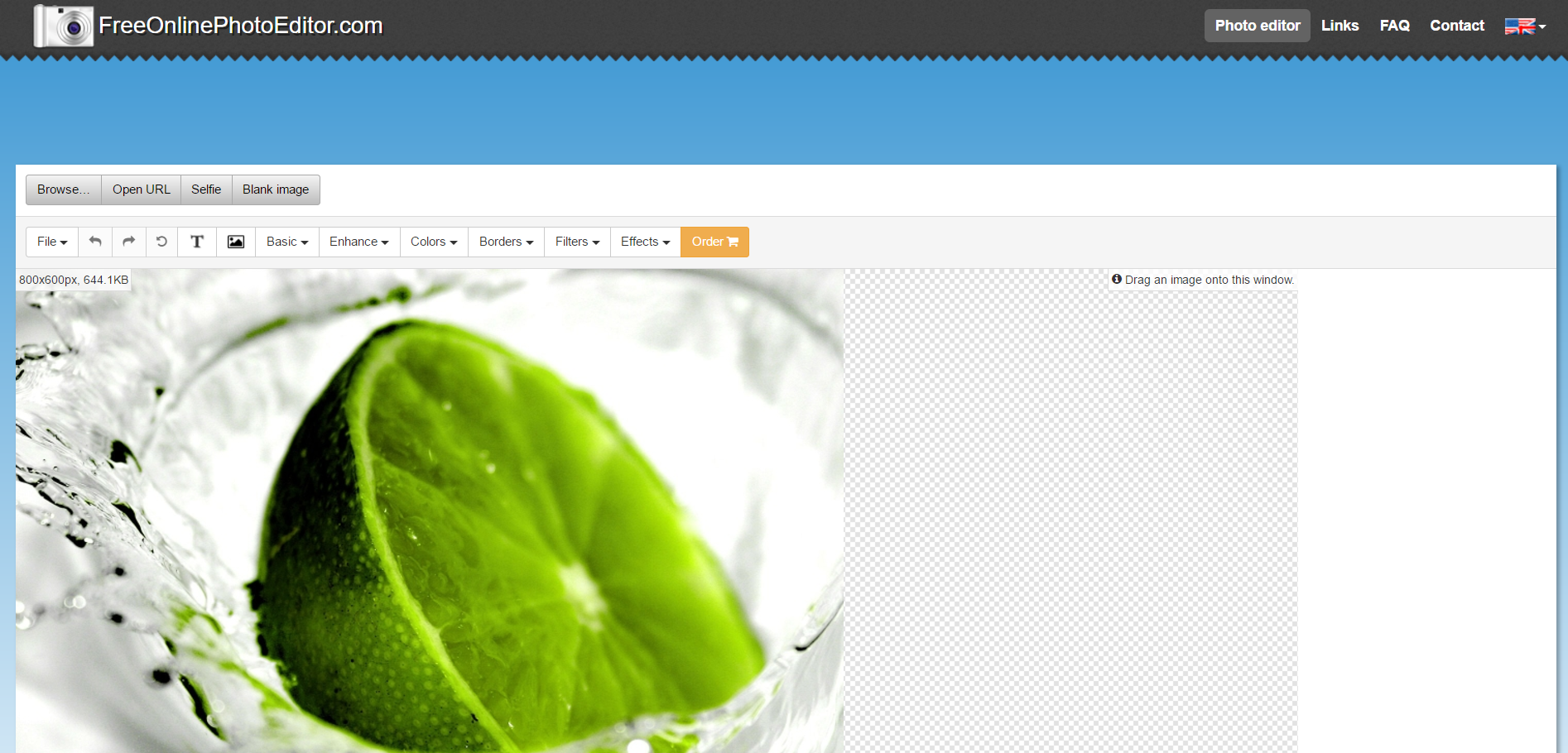


Рис. 5.2 Інтерфейс веб-ресурсу «[www.freeonlinephotoeditor.com](http://www.freeonlinephotoeditor.com)»

Останнім продуктом аналогом для порівняння було обрано веб – сервіс Lunapic Editor (рис. 5.3). На цьому сервісі відсутня реєстрація та логін користувача, що означає зосередження уваги на обробці зображень. Як і у всіх попередніх аналогах, цей продукт дає можливість створення власного, або завантаження вже існуючого зображення. Ще одним із основних недоліків цього веб сервісу варто зазначити відсутність адаптивного дизайну сайту.

Основними функціональними можливостями даного веб-ресурсу є наступні:

* геометричні перетворення: обертання, масштабування;
* скасування останніх змін;
* зміна яскравості та контрасту;
* вставлення користувацького тексту на зображення;
* стилізація країв зображення;
* зміна колірного простору зображення;
* додавання та редагування анімації (.gif);
* побудова гістограми цифрового зображення;
* тощо.

На цьому веб-ресурсі користувач має можливість поділитися своїм зображенням в соціальні мережі: Facebook, Twitter, Google+, Pinterest та інші. Інтерфейс користувача є досить простим і зрозумілим [3].

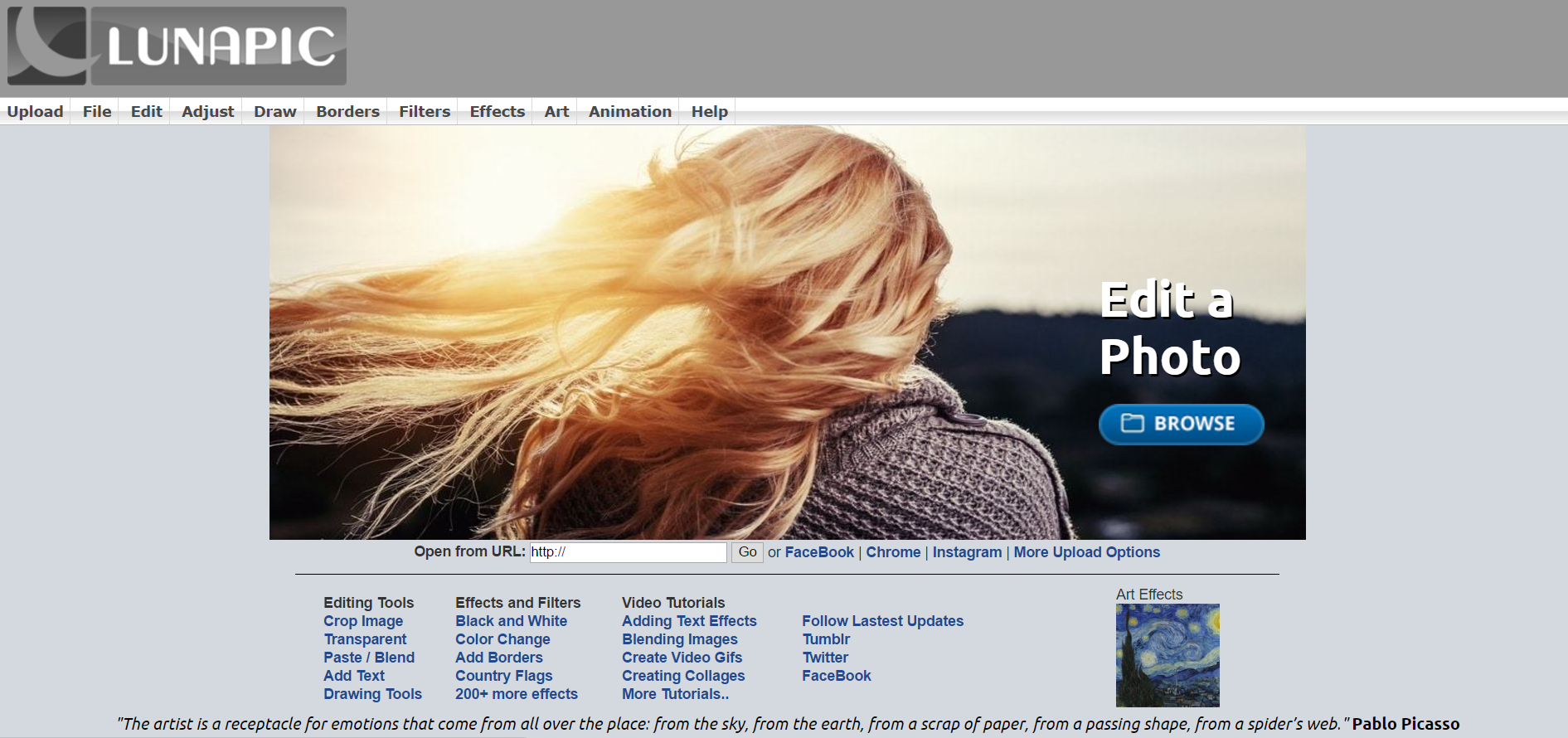


Рис. 5.3 Інтерфейс веб-ресурсу «www141.lunapic.com/editor»

1. ПОСТАНОВКА ЗАВДАННЯ РОЗРОБКИ ВЕБ-СЕРВІСУ ДЛЯ ОПРАЦЮВАННЯ, ПОКРАЩЕННЯ ТА ГЕОМЕТРИЧНОГО ПЕРЕТВОРЕННЯ ОКРЕМИХ ОБ’ЄКТІВ РАСТРОВИХ ЗОБРАЖЕНЬ

## Постановка завдання

Базуючись на інформації з першого розділу, сформулюємо чітке завдання. Завдання полягає в тому, щоб створити веб-сервіс, який надасть змогу користувачам здійснювати обробку зображень та керувати базою даних власних зображень. Для того, щоб здійснювати управління базою даних власних зображень, користувач повинен мати змогу зареєструватися в системі. Після чого, система надасть користувачу відповідні права. Оскільки, опрацювання зображень – це головна задача мого програмного продукту, то система повинна підтримувати всі основні засоби для досягнення цієї мети. Тобто веб-сервіс повинен мати змогу підтримувати наступні можливості:

* вибір зображення для можливості опрацювання;
* зберігання зображення після обробки;
* підтримувати базові геометричні перетворення (обертання, масштабування);
* зміна яскравості та контрасту;
* обрізання зображення;
* автоматичне покращення якості зображення;
* розпізнавання конкретних об’єктів на зображеннях.

Також, важливими елементами веб-сервісу є безпека даних користувача, простий, зрозумілий та адаптивний до всіх розмірів екрану інтерфейс.

## Вибір архітектури

Для побудови мого програмного продукту я обрав клієнт-серверну архітектуру. Оскільки, даний веб-сервіс передбачає побудову клієнтської частини, тому для полегшення роботи та доброго структурування проекту ми повинні використовувати компонентну архітектуру побудови проекту. Клієнтська частина буде представлена будь-яким браузером, тобто, це буде легкий клієнт на стороні користувача. Клієнт буде побудований з використанням окремих модулів, компонентів, сервісів, HTML-шаблонів, використовуючи data binding та dependency injection. Всі ці механізми передбачені JavaScript – фреймворком AngularJS 2. Серверна частина системи буде побудована за методологією REST. В цій методології дані повинні передаватися у вигляді невеликої кількості стандартних форматів (наприклад, HTML, XML, JSON). Сервер буде здійснювати взаємодію з реляційною базою даних для управління користувачами та іншими компонентами системи. Також серверна частина буде приймати зображення від клієнта та здійснюватиме безпосереднє опрацювання цих зображень. Оскільки, опрацювання зображень це трудомісткий процес, то очікування на відповідь сервера буде займати певний час.

## Вибір програмних засобів

Для написання веб-сервісу була вибрана платформа фірми Oracle Java EE. Платформа надає API та виконавче середовище для розробки і виконання корпоративного програмного забезпечення, включаючи мережеві та веб-сервіси.

В якості основного середовища програмування засобами Java EE було вибрано IntelliJ IDEA Ultimate Edition – потужний редактор вихідного коду. Надає можливості автозаповнення з врахуванням вибраного контексту, автоматичне імпортування потрібних бібліотек, оптимізація з врахуванням видалення непотрібних бібліотек. Швидкість початкового налаштування, усі інструменти доступні відразу після встановлення середовища.

Серверна частина буде написана із використанням Spring Framework – програмний каркас на основі HTTP сервлета, що забезпечує створення веб-додатків і веб-служб RESTful [6]. Для відображення між об'єктами та реляційними структурами для платформи Java буде використано Hibernate, яка надає легкий для використання каркас (фреймворк) для відображення між об'єктно-орієнтованою моделлю даних і традиційною реляційною базою даних.

Для розробки клієнтської частини буде використано JavaScript – фреймворк AngularJS 2, який дозволяє розробляти веб – сторінку за технологією SPA (Single Page Application). За допомогою створення власних компонентів, HTML – шаблонів та CSS буде здійснюватися взаємодія з користувачем. Бізнес – логіка на клієнтський частині буде реалізована засобами TypeScript.

Для роботи з базою даних було обрано MySQL Workbench – інструмент для візуального проектування баз даних, що інтегрує проектування, моделювання, створення й експлуатацію БД в єдине безкоштовне оточення для систем баз даних MySQL.

Для побудови проекту буде використано інструмент побудови Java проектів – Maven. Для розгортування веб-сервісу був вибраний сервер Apache Tomcat, так як він простий і швидкий у використанні.

## Специфікація вимог до програмного продукту

1. **Вступ**
   1. **Призначення та мета**

Метою даного проекту є створення веб-сервісу для надання користувачам можливості опрацювання зображень, покращення їх якості та управління збереженими користувачем зображеннями. Веб – сторінка повинна бути інтуїтивно зрозумілою та адаптивною до всіх розмірів екрану.

* 1. **Продукти аналоги**

Порівняльну характеристику продуктів - аналогів подано у табл. 2.1.

Таблиця 2.1

|  |  |  |  |
| --- | --- | --- | --- |
| Можливості | pixlr.com | www.freeonlinephotoeditor.com | www141.lunapic.com/editor |
| Вибір зображення для можливості опрацювання | + | + | + |
| Зберігання зображення після обробки | + | + | + |
| Реєстрація та логін користувача | - | - | - |
| Наявність адаптивного дизайну | + | + | - |
| Роспізнавання конкретних об’єктів на зображеннях | + | - | - |

1. **Загальний опис**
   1. **Характеристики продукту**

На високому рівні абстракції можна виділити наступні основні характеристики:

* реєстрація та логін користувача;
* вибір зображення для можливості опрацювання;
* збереження зображення після обробки;
* підтримка базових геометричних перетворень (обертання, масштабування);
* зміна яскравості та контрасту;
* автоматичне покращення якості зображення;
* роспізнавання конкретних об’єктів на зображеннях;
* додавання зображень в базу даних користувача.
  1. **Класи користувачів**

У моєї системи буде один клас користувачів - споживач системи. Споживачі системи – це основна аудиторія, ті для кого, власне, і розробляється програмний продукт, вони будуть мати змогу опрацьовувати зображення та здійснювати управління своїми збереженими зображеннями.

* 1. **Середовище функціонування**

Серверна частина системи буде підтримуватися на будь-якій операційній системі з будь – якими конфігураціями ПК. Головною вимогою для функціонування є наявність JVM. Клієнтська ж частина вимагатиме веб-браузер з повною підтримкою таких стандартів як HTML 5, CSS 3.

1. **Характеристика системи**
   1. **Реєстрація у системі**
      1. Опис і пріоритет

Пріоритет – високий. Можливість реєстрації користувача у системі.

* + 1. Послідовності дія-відгук

1. Користувач відкриває сторінку реєстрації.
2. Користувач заповнює необхідні поля.
3. Система валідує введену інформацію.
4. Якщо інформація невалідна система повідомляє про це.
5. Користувач підтверджує реєстрацію.
6. Система повідомляє про результат реєстрації.
7. При успішній реєстрації система відправляє повідомлення для перевірки електронної пошти.
   * 1. Функціональні вимоги

REQ 1. Всі необхідні поля мають бути провалідовані.

REQ 2. Довжина паролю: від 6 до 25 символів.

REQ 3. Наявність поля «Повторити пароль».

REQ 4. Унеможливити реєстрацію з вже використаною адресою електронної пошти.

* 1. **Вхід у систему**
     1. Опис і пріоритет

Пріоритет – високий. Можливість входу користувача у систему.

* + 1. Послідовності дія-відгук

1. Користувач відкриває сторінку входу.
2. Користувач заповнює поля «Електронна пошта» та «Пароль».
3. Користувач підтверджує вхід.
4. Якщо результат – успішний, система переадресовує користувача на головну сторінку.
   * 1. Функціональні вимоги.
   1. **Вибір зображення для опрацювання**
      1. Опис і пріоритет

Пріоритет – високий. Можливість вибору зображення для редагування.

* + 1. Послідовності дія-відгук

1. Користувач відкриває сторінку редагування зображення.
2. Користувач вибирає кнопку завантаження зображення.
3. Користувач вибирає певне зображення для опрацювання.
4. Користувач може здійснювати обробку зображення.
   * 1. Функціональні вимоги

REQ 1. Можливість вибору зображень різних форматів.

REQ 2. Можливість вибору тільки одного зображення.

* 1. **Збереження зображення після обробки**
     1. Опис і пріоритет

Пріоритет – високий. Можливість зберегти зображення після обробки.

* + 1. Послідовності дія-відгук

1. Користувач відкриває сторінку редагування зображення.
2. Користувач вибирає кнопку збереження зображення на певний носій.
3. Користувач вибирає певне місце на фізичному носії для збереження.
4. Збережене зображення.
   * 1. Функціональні вимоги

REQ 1. Можливість вибору конкретного шляху на фізичному носії для збереження.

* 1. **Обертання зображення**
     1. Опис і пріоритет

Пріоритет – високий. Можливість обертання зображення на певний кут.

* + 1. Послідовності дія-відгук

1. Користувач відкриває сторінку редагування зображення.
2. Користувач вибирає кнопку обертання зображення.
3. Користувач вибирає певний кут повороту в діалоговому вікні.
4. Користувач нажимає кнопку підтвердження.
5. Обернене зображення.
   * 1. Функціональні вимоги

REQ 1. Можливість вибору кута повороту зображення.

REQ 2. Система повинна валідувати значення кута повороту.

REQ 3. Можливість підтвердити здійснені зміни.

* 1. **Масштабування зображення**
     1. Опис і пріоритет

Пріоритет – високий. Можливість масштабування зображення.

* + 1. Послідовності дія-відгук

1. Користувач відкриває сторінку редагування зображення.
2. Користувач вибирає кнопку масштабування зображення.
3. Користувач вказує значення масштабування.
4. Користувач нажимає кнопку підтвердження.
5. Масштабоване зображення.
   * 1. Функціональні вимоги

REQ 1. Можливість вибору значення масштабування зображення.

REQ 2. Система повинна валідувати значення масштабування.

REQ 3. Можливість підтвердити здійснені зміни.

* 1. **Зміна контрасту зображення**
     1. Опис і пріоритет

Пріоритет – високий. Можливість змінити контраст зображення.

* + 1. Послідовності дія-відгук

1. Користувач відкриває сторінку редагування зображення.
2. Користувач вибирає кнопку зміну контрасту зображення.
3. Користувач вказує значення контрасту.
4. Користувач нажимає кнопку підтвердження.
5. Оновлене зображення.
   * 1. Функціональні вимоги

REQ 1. Можливість вибору значення контрасту зображення.

REQ 2. Система повинна валідувати значення контрасту.

REQ 3. Можливість підтвердити здійснені зміни.

* 1. **Автоматичне покращення якості зображення**
     1. Опис і пріоритет

Пріоритет – високий. Можливість покращити якість зображення.

* + 1. Послідовності дія-відгук

1. Користувач відкриває сторінку редагування зображення.
2. Користувач вибирає кнопку покращення зображення методом вирівнювання гістограми.
3. Користувач нажимає кнопку підтвердження.
4. Оновлене зображення.
   * 1. Функціональні вимоги
   1. **Розпізнавання конкретних об’єктів на зображенні**
      1. Опис і пріоритет

Пріоритет – високий. Можливість автоматичного розпізнавання образів на зображенні.

* + 1. Послідовності дія-відгук

1. Користувач відкриває сторінку редагування зображення.
2. Користувач вибирає кнопку розпізнавання образів.
3. Користувач нажимає кнопку підтвердження.
4. Виділені об’єкти на зображенні та можливість здійснення перетворень над ними.
   * 1. Функціональні вимоги
5. **Вимоги до зовнішніх інтерфейсів**
   1. **Користувацькі інтерфейси**

Користувач спілкуватиметься з системою за допомогою миші та клавіатури або сенсорного екрану.

* 1. **Апаратні інтерфейси**

Апаратні інтерфейси не будуть використовуватися цією програмною системою.

* 1. **Програмні інтерфейси**

Взаємодія з базою даних буде здійснюватися за допомогою MySQL Workbench та засобами бібліотеки Hibernate, яка надає доступ до низькорівневих функцій роботи з базою даних. Hibernate — засіб автоматичного відображення між об'єктами та реляційними структурами для платформи Java.

* 1. **Комунікаційні інтерфейси**

Клієнтська частина системи буде спілкуватися з серверною за допомогою протоколу HTTP/1.1.

1. **Інші нефункціональні вимоги**
   1. **Вимоги продуктивності**

Необхідні оптимальні рішення для забезпечення максимальної актуальності даних, швидкість оновлення і додавання нових подій в систему.

* 1. **Вимоги безпеки**

Необхідно забезпечити безпеку персональних даних користувачів, захист від зловмисників, а також організувати безпечне з’єднання у випадку електронної купівлі квитка.

* 1. **Атрибути якості програмного продукту**

Веб-сервіс сервіс повинен відповідати критеріям якості, таким як надійність, зокрема всі персональні дані повинні бути захищенні, стресостійкість, сервіс повинен витримувати навантаження великої кількості користувачів. Веб – інтерфейс повинен бути зручний у використанні.

* 1. **Інші вимоги**

Юридичні вимоги повинні відповідати закону України.

# 7. РЕЗУЛЬТАТИ ПРОЕКТУВАННЯ ВЕБ СЕРВІСУ

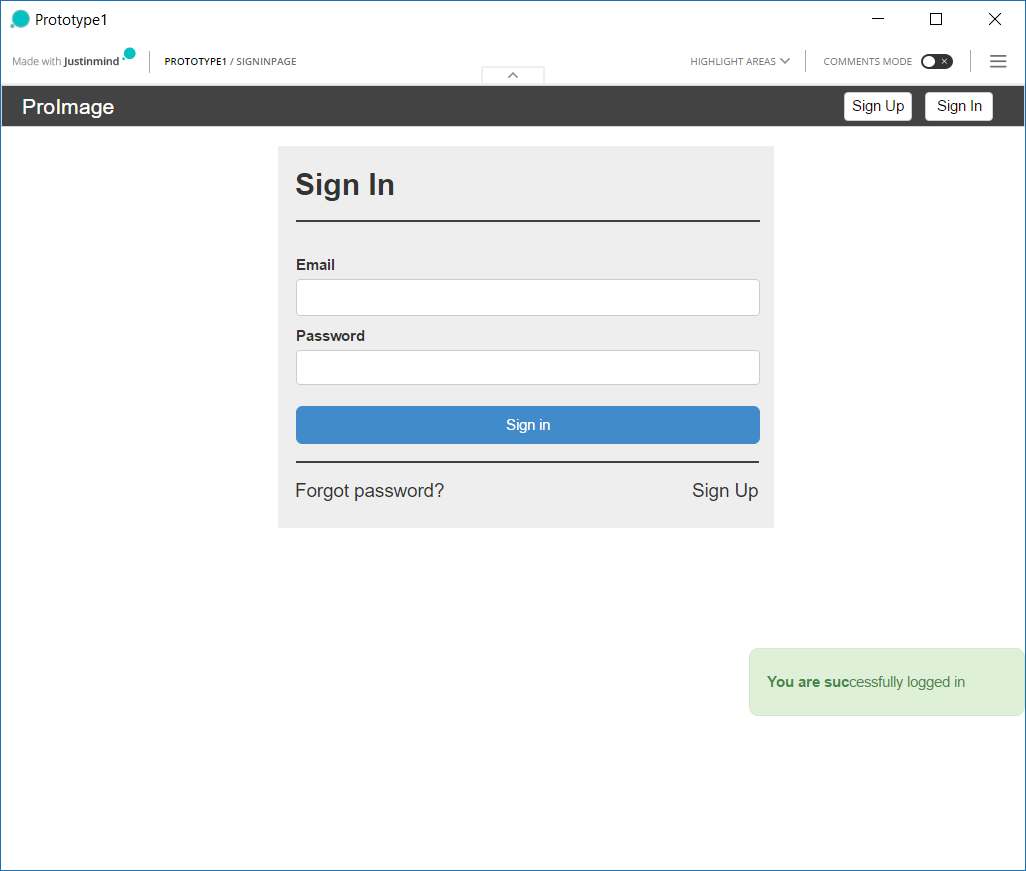


Рис. 7.1 Вікно авторизації

На **вікні авторизації** (Рис. 7.1), для входу, необхідно ввести електронний адрес та пароль і натиснути кнопку «Sign In».

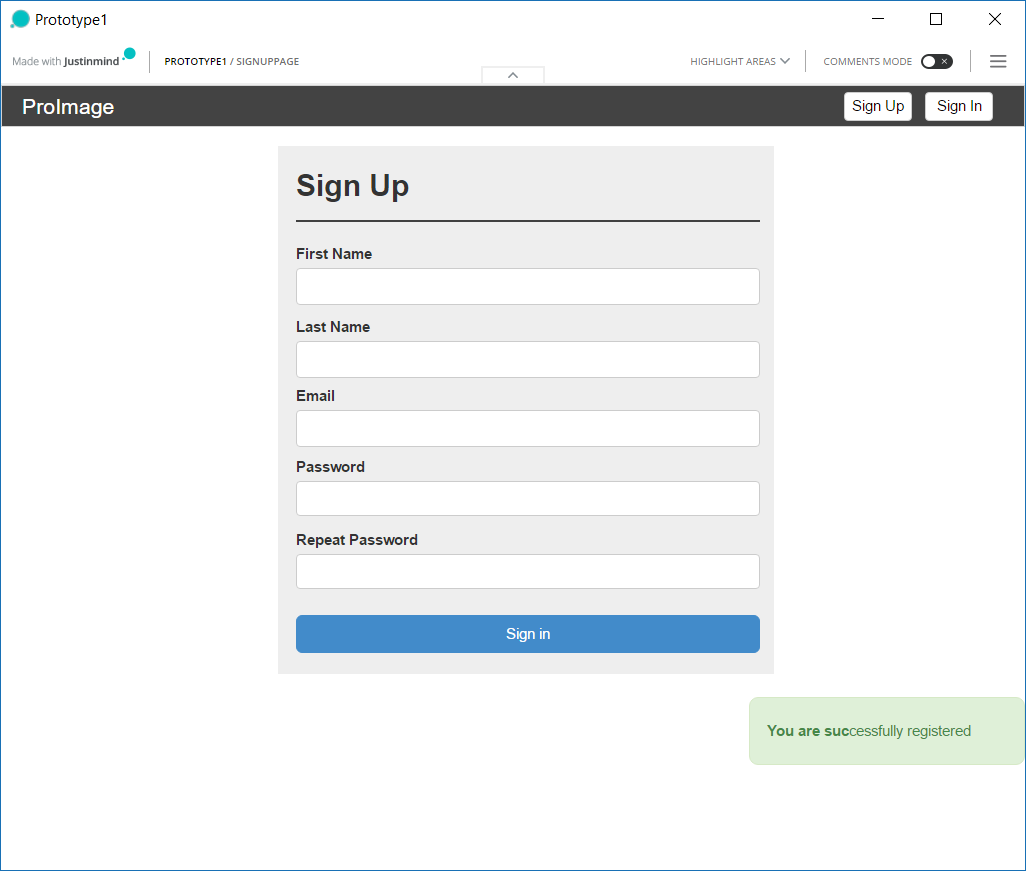


Рис. 7.2 Вікно реєстрації

На **вікні реєстрації** (Рис. 7.2) можна зареєструватися заповнивши всі поля і натиснути кнопку «Sign Up».

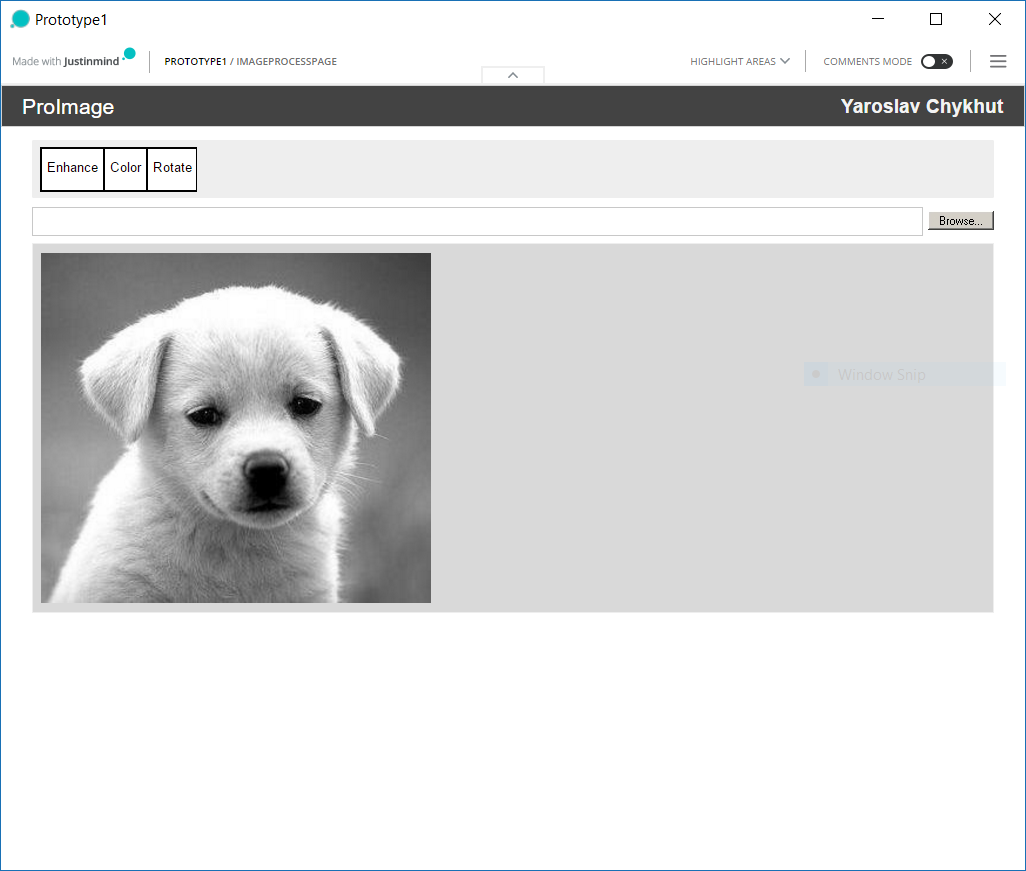


Рис. 7.3 Вікно редагування зображення

На **вікні редагування зображення** (Рис. 7.3) користувач має змогу вибрати будь – яке зображення з фізичного носія та здійснювати різні маніпуляції з цим зображенням за допомогою функцій, які присутні на панелі інструментів.

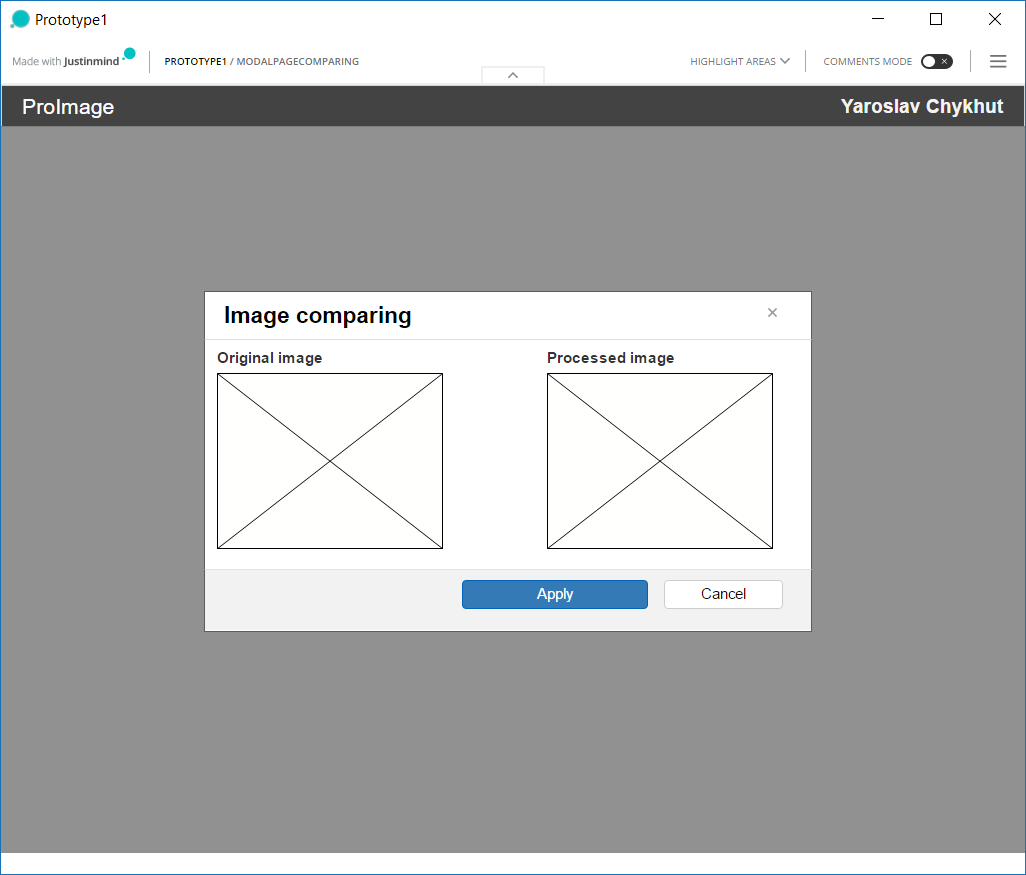


Рис. 7.4 Вікно для порівняння оригінального та обробленого зображення

На **вікні для порівняння оригінального та обробленого зображення** (Рис. 7.4) користувач може порівняти та подивитися результати опрацювання зображення.

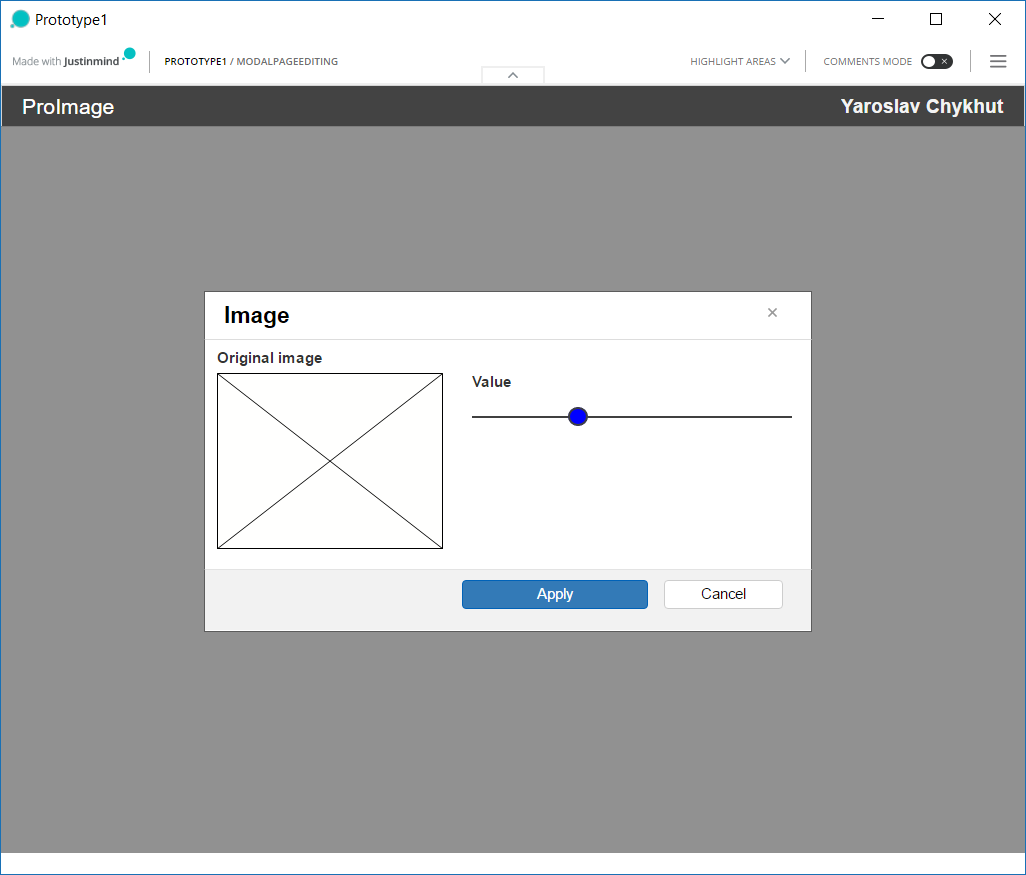


Рис. 7.5 Вікно для перегляду опрацьованого зображення

На **вікні для перегляду опрацьованого зображення** (Рис. 7.5) користувач здійснює встановлення певних параметрів зображення та переглядати його змін в реальному часі.

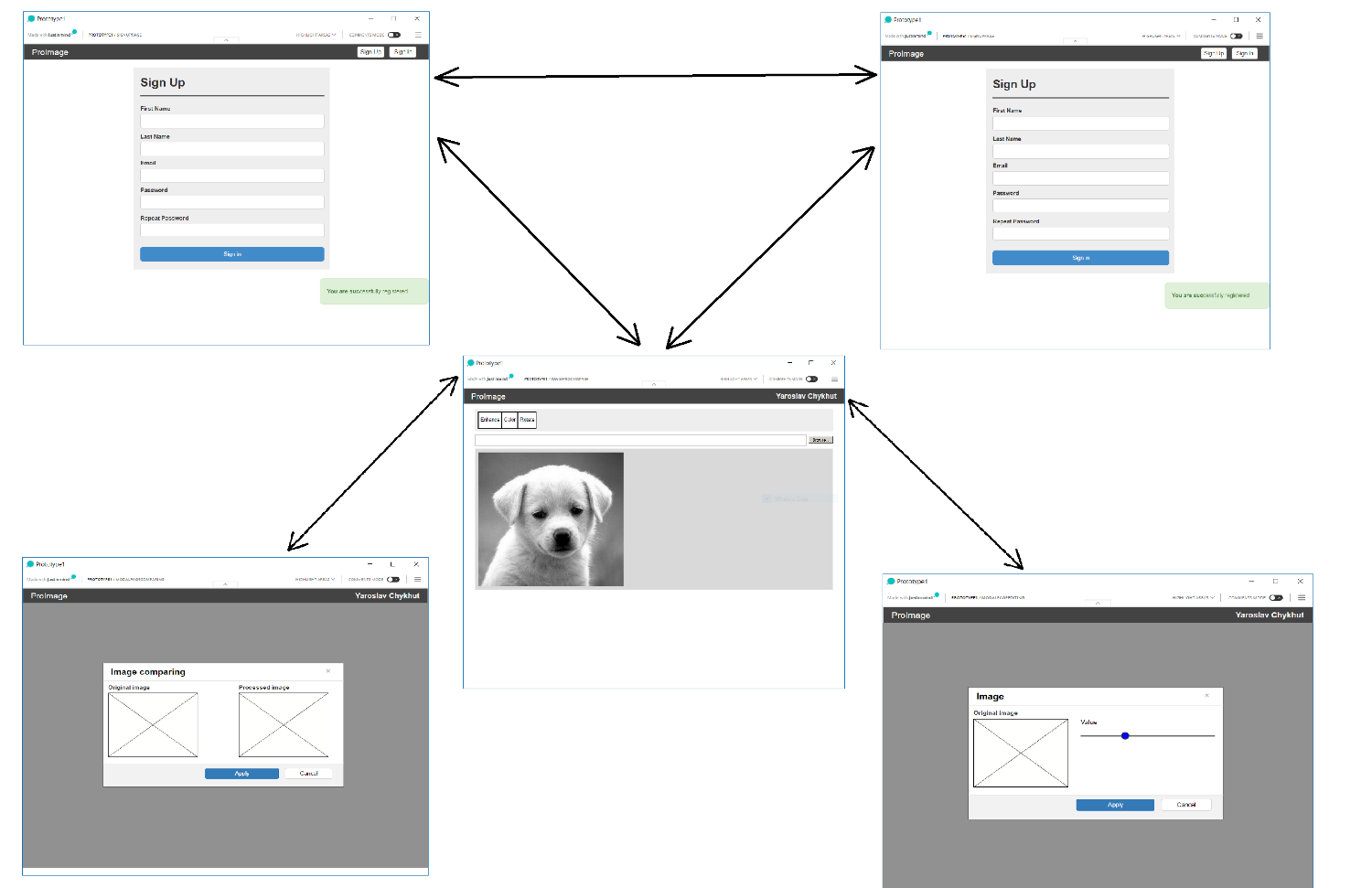


Рис. 7.6 Схема переходів сайту

Схема переходів усіх вікон сайту (Рис. 7.6).

# ВИСНОВОК

На базі практики я отримав професійні навички роботи в команді та освоїв основні засоби та навики побудови веб – сервісів за методологією REST. Використовуючи засоби Spring Framework та Spring Security, розробив програмне забезпечення для реєстрації та логіну користувача. Дане програмне забезпечення повністю відповідає вимогам безпеки користувацьких даних з допомогою використання токена авторизації для кожного користувача. Також вивчив фреймворк Spring Data для опрацювання користувацьких даних в базі даних – MySQL.

Для дипломного проекту була розглянута його предметна область, а саме, засоби опрацювання растрових зображень, були проаналізовані численні інформаційні джерела, щоб зібрати якомога більше інформації для розробки необхідного веб-сервісу. На основі цих даних був написаний перший розділ до дипломного проекту. Також були обрані архітектура та технології, необхідні для розробки, складена специфікація вимог до розробки ПЗ.

# ЛІТЕРАТУРА

1. PIXLR Editor [Електронний ресурс] – Режим доступу: https://pixlr.com/editor.
2. Free Online Photo Editor [Електронний ресурс] – Режим доступу: https://www.freeonlinephotoeditor.com.
3. Lunapic Editor [Електронний ресурс] – Режим доступу: http://www141.lunapic.com/editor.
4. Комп’ютерне моделювання систем та процесів. Методи обчислень. Частина 2: навчальний посібник/ Квєтний Р. Н., Богач І. В., Бойко О. Р., Софина О. Ю., Шушура О.М.; за заг. ред. Р.Н. Квєтного. – Вінниця: ВНТУ, 2012. – 230 с.
5. Алгоритм Кенні [Електронний ресурс] – 2016. – Режим доступу: https://uk.wikipedia.org/wiki/Алгоритм\_Кенні.
6. Документація фреймворку Spring [Електронний ресурс] – Режим доступу: https://spring.io/docs.

# ДОДАТОК А

Вихідний код для здійснення реєстрації та логіну користувача засобами Spring Security.

@Configuration

@EnableWebSecurity

@EnableGlobalMethodSecurity(prePostEnabled = true)

public class ImageApiSecurityConfig extends WebSecurityConfigurerAdapter {

@Value("${auth.logout.redirect.url}")

private String logoutRedirectUrl;

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth

.userDetailsService(userDetailsService())

.passwordEncoder(passwordEncoder());

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http

.csrf().disable()

.sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS)

.and()

.addFilterBefore(jwtLoginFilter(), UsernamePasswordAuthenticationFilter.class)

.addFilterBefore(jwtAuthenticationFilter(), UsernamePasswordAuthenticationFilter.class)

.authorizeRequests()

.antMatchers(HttpMethod.POST, "/login").permitAll()

.antMatchers(HttpMethod.POST, "/users/register").permitAll()

.antMatchers(HttpMethod.GET, "/users/verify/\*").permitAll()

.anyRequest().authenticated()

.and().logout()

.logoutSuccessUrl(logoutRedirectUrl).permitAll();

}

@Bean

@Override

public AuthenticationManager authenticationManagerBean() throws Exception {

return super.authenticationManagerBean();

}

@Bean

public Filter jwtLoginFilter() throws Exception {

return new JWTLoginFilter("/login", authenticationManagerBean());

}

@Bean

public Filter jwtAuthenticationFilter() throws Exception {

return new JWTAuthenticationFilter();

}

@Bean

public PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

@Bean

public UserDetailsService userDetailsService() {

return new SecurityUserDetailsService();

}

}

public class JWTLoginFilter extends AbstractAuthenticationProcessingFilter {

@Autowired

private TokenAuthenticationService tokenAuthenticationService;

public JWTLoginFilter(String url, AuthenticationManager authManager) {

super(new AntPathRequestMatcher(url));

setAuthenticationManager(authManager);

}

@Override

public Authentication attemptAuthentication(HttpServletRequest req, HttpServletResponse res) throws AuthenticationException, IOException, ServletException {

UserDTO userCredentials = GeneralUtils.fromJSONStreamToObject(req.getInputStream(), UserDTO.class);

if(StringUtils.isBlank(userCredentials.email) || StringUtils.isBlank(userCredentials.password)){

throw ExceptionBuilder.BAD\_REQUEST\_MISSING\_PARAMETERS.build("Email and password are required");

}

return getAuthenticationManager().authenticate(

new UsernamePasswordAuthenticationToken(

userCredentials.email,

userCredentials.password,

Collections.emptyList()

)

);

}

@Override

protected void successfulAuthentication(HttpServletRequest request, HttpServletResponse response, FilterChain chain, Authentication authResult) throws IOException, ServletException {

tokenAuthenticationService.addAuthentication(response, authResult.getName());

}

}

public class JWTAuthenticationFilter extends GenericFilterBean {

@Autowired

private TokenAuthenticationService tokenAuthenticationService;

@Autowired

private SecurityContextService securityContextService;

@Override

public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) throws IOException, ServletException {

Authentication authentication = tokenAuthenticationService.getAuthentication((HttpServletRequest)servletRequest);

securityContextService.saveAuthenticationData(authentication);

filterChain.doFilter(servletRequest, servletResponse);

}

}