**ЗМІСТ**

[ВСТУП 4](#_Toc42700772)

[РОЗДІЛ 1 ВИКОРИСТАНЯ ГРАФІЧНИХ РЕДАКТОРІВ ЗОБРАЖЕНЬ ЛЮДИНОЮ В ЦИФОРВУ ЕПОХУ 6](#_Toc42700773)

[1.1 Опис предметної області 6](#_Toc42700774)

[1.2 Аналітичний огляд існуючих рішень 7](#_Toc42700775)

[РОЗДІЛ 2 ФОРМУВАННЯ ТЕХНІЧНОГО ЗАВДАННЯ ДЛЯ РЕДАКТОРА ЗОБРАЖЕНЬ 13](#_Toc42700776)

[2.1. Найменування, галузь застосування та призначення розробки 13](#_Toc42700777)

[2.2 Вимоги до програмного продукту 13](#_Toc42700778)

[2.3 Опис вхідної інформації 14](#_Toc42700779)

[2.4 Опис результуючої інформації 15](#_Toc42700780)

[2.5 Етапи розробки 16](#_Toc42700781)

[РОЗДІЛ 3 ПРОЄКТУВАННЯ СХЕМИ РОБОТИ 19](#_Toc42700782)

[3.1 Проєктування архітектури програмного коду 19](#_Toc42700783)

[3.2 Розробка структурної схеми проєкту на основі UML-діаграм 21](#_Toc42700784)

[3.3 Проєктування інтерфейсу та структур даних 25](#_Toc42700786)

[РОЗДІЛ 4 ПРОГРАМНА РЕАЛІЗАЦІЯ ПРОЄКТНИХ РІШЕНЬ, ТЕСТУВАННЯ ТА ЕКСПЛУАТАЦІЯ 27](#_Toc42700787)

[4.1 Програмна реалізація проєкту 27](#_Toc42700788)

[4.2 Тестування та розгортання проєкту 29](#_Toc42700794)

[4.3 Інструкція з інсталяції та експлуатації 30](#_Toc42700797)

[РОЗДІЛ 5 РОЗРАХУНОК ЕКОНОМІЧНИХ ПОКАЗНИКІВ 35](#_Toc42700798)

[5.1 Маркетингові дослідження ринку збуту розробленого програмного продукту 35](#_Toc42700799)

[5.2 Розрахунок вартості програмного забезпечення 36](#_Toc42700800)

[5.3 Оцінка економічної ефективності впровадження ПЗ 39](#_Toc42700801)

[ВИСНОВКИ 42](#_Toc42700803)

[СПИСОК ІНФОРМАЦІЙНИХ ДЖЕРЕЛ 43](#_Toc42700804)

[ДОДАТКИ 45](#_Toc42700805)

ВСТУП

Бізнес на даний момент має потребу в швидкій інтеграції графічних редакторів, які надавали б мінімальний функціонал і були відносно прості та доступні.

В сучасному світі цифрова фотографія є невід’ємною частиною життя майже кожної розумної людини, а з кожним роком популярність цифрової фотографії тільки зростає, оскільки з’являються більш потужні камери, на відносно дешевих пристроях.

Однак, майже в одну ногу з фотографією розвивається і сфера програмування, яка пов’язана з обробкою візуальних зображень. На даний момент є багато редакторів фотографій, які дозволяють обробляти фотографії з величезною швидкістю, дозволяють застосовувати велику кількість фільтрів та інструментів, для редагування та потокової обробки медіа, проте більшість із них створені для щоденного використання, однак дуже несправедливо напрям редакторів обійшов бізнес сферу, яка потребує редагування фотографій не менше ніж буденність.

В якості основної мети для написання проєкту було взято ідею зручного, простого та доступного редактора фотографій, що дозволить обробляти зображення, а також буде містити в собі деякий додатковий функціонал, який буде відрізняти його серед інших.

РОЗДІЛ 1 ВИКОРИСТАНЯ ГРАФІЧНИХ РЕДАКТОРІВ ЗОБРАЖЕНЬ ЛЮДИНОЮ В ЦИФОРВУ ЕПОХУ

* 1. Опис предметної області

Темою дипломного проекту є: «Багатофункціональний редактор-конвертор зображень».

Бізнес на даний момент має потребу в швидкій інтеграції графічних редакторів, які надавали б мінімальний функціонал і були відносно прості та доступні.

У даному проекті предметною областю виступає бізнес сфера, фірми або конкретні особи, які потребують швидкого та зручного редагування зображень

* 1. Аналітичний огляд існуючих рішень

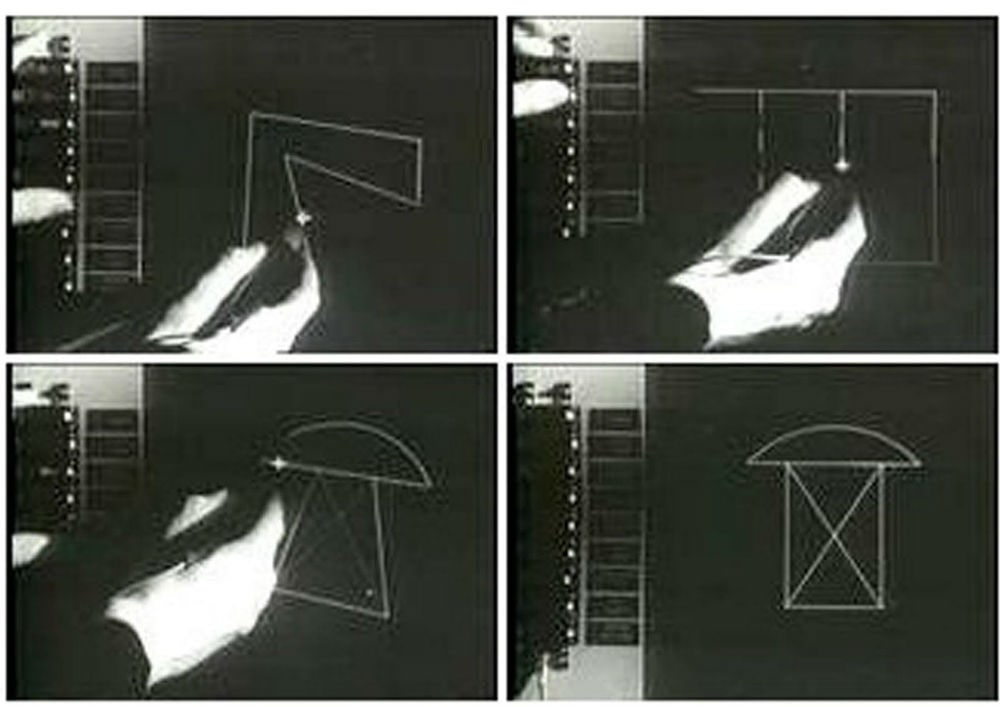
Сфера редагування фотографій далеко не нова, а тому боротьба в ній ведеться доволі довго, ще в 1963 році Айван Сазерленд представив перший в світі графічний редактор і систему автоматизованого проєктування Sketchpad зображено на рисунку 1.1.

Рисунок 1.1 – Зовнішній вигляд першого в світі редактора Sketchpad

З тих часів пройшло вже багато часу і зараз сфера редагування зображень набагато потужніша, ніж в той час і кожний рік ринок представляє нам новіші та потужніші редактори, одним із найкращих редакторів на даний момент є Adobe Photoshop, інтерфейс якого зображено на рисунку 1.2.

Рисунок 1.2 – Графічний вигляд Adobe Photoshop

Переваги:

* потужність, а також великий вибір функціоналу;
* надійність, в ринку тримається доволі довго, тому є первіреним та стійким до помилок;
* робота з майже усіма існуючими на даний момент типами фотографій та іншими медіа форматами.

Недоліки:

* застарілість, складна архітектура та погана якість підтримки коду;
* вартість, покупка ліцензії занадто дорога для середнього користувача;
* складність функціоналу, потрібні деякі навики для роботи з цією програмою.

Також існує усім відома програма Paint, яка встановлена майже на кожному комп’ютері з операційною системою Microsoft Windows, вона є дуже зручною для повсякденного використання, однак все ж таки для бізнесу його функціоналу не достатньо, а технології, які використані для написання цього редактора доволі застарілі.

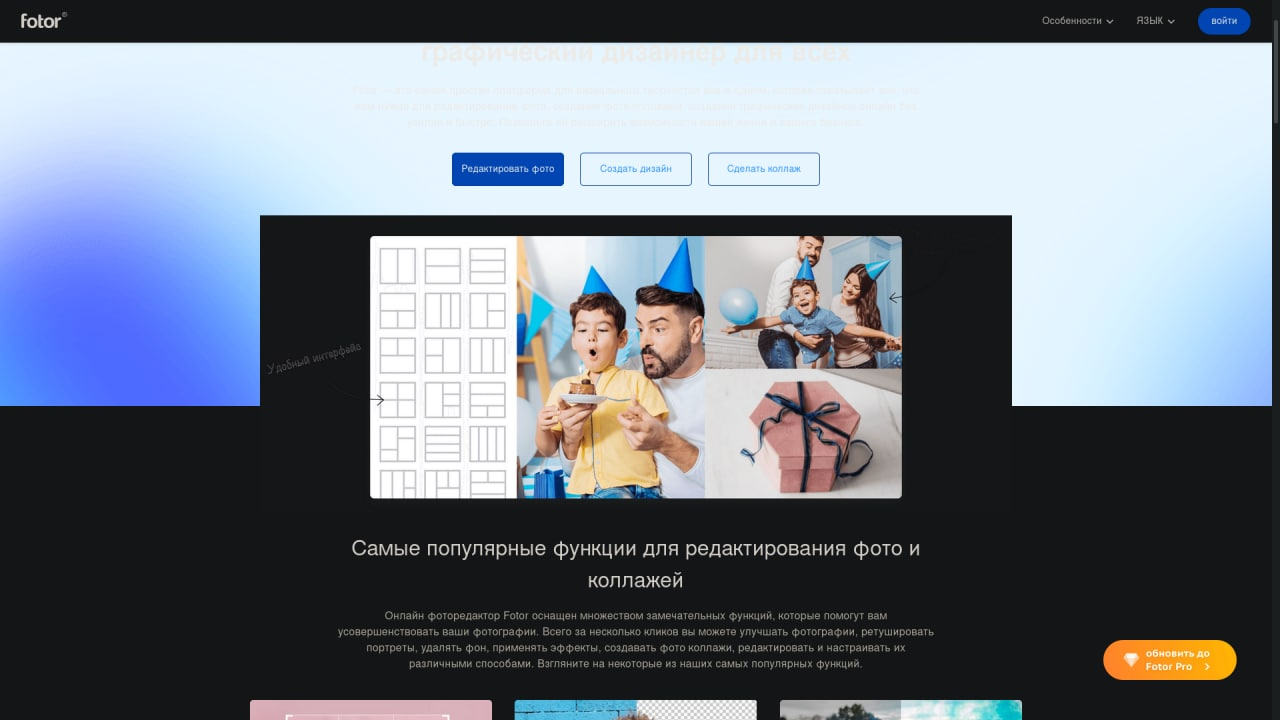
 Окремо вартує відмітити область онлайн редагування зображень, яка зараз є дуже розвиненою, візьмемо до прикладу онлайн сервіс Fotor, зовнішній вигляд якого зображено на рисунку 1.3.

Рисунок 1.3 – Графічний вигляд редактора Fotor

Переваги:

* відносна простота в освоєнні;
* відсутність потреби встановлювати графічний редактор локально;
* інтеграція з почтою та месенджером Instagram.

Недоліки:

* відсутність можливості інтеграції з бізнесом;
* проект написаний з використанням стандартних бібліотек і є строго Web додатком і відповідно, може використовуватись лише у веб середовищі

Отже, ринок редагування фотографій дуже великий, тому зробити однозначно правильний вибір доволі складно. Проаналізувавши наявну ситуацію на ринку, було взято від багатьох редакторів їхні певні особливості та переваги для отримання кращого результату, для япроєктування програмної реалізації графічного редактору.

1. ФОРМУВАННЯ ТЕХНІЧНОГО ЗАВДАННЯ ДЛЯ РЕДАКТОРА ЗОБРАЖЕНЬ
2. Найменування, галузь застосування та призначення розробки

Темою дипломного проєкту є програмний продукт «Багатофункціональний редактор-конвертор зображень».

Метою створення додатка є конструювання відносно простого, однак не менш корисного редактора зображень з максимальним напрямом на зручність архітектури, інтергрованість та маштабованість, який можна буде легко використовувати, а також наданням додаткового функціоналу, такого як відправка повідомлень у месенджер.

Галузь застосування - бізенес сфери так не конретні особі, які зацікавлені в зручному і простому редагуванні зображень

* 1. Вимоги до програмного продукту

Фінальна версія програми повина містити в собі наступний функціонал:

* інтуітивно зрозумілий інтерфейс;
* завантаження та вигрузку зображень;
* надання можливості користувачу застосовувати фільтри до зображення;
* освітлення і затінення зображення, а також можливість використовувати повну градацію сірого кольору;
* надання можливості користувачу обрізати зображення до необхідних йому розмірів;
* функціонал для дзеркального відображення зображення;
* функціонал для відсіювання фону від зображення;
* надання можливості користувачу експортувати свої відредаговані зображення до Телеграму.

Загалом програма складається з двох сервісів: один відповідає за обробку зображень, їх редагування, а інший імпортує зображення до Телеграму, він же і буде являти собою базу для подальшого масштабування, оскільки до цього сервісу зображення попадає в конвертованому форматі Base64, тому ми можемо робити з зображення абсолютно все те саме, що і з Base64 стрічками.

* 1. Опис вхідної інформації

В даному проєкті є декілька видів вхідної інформації, перші дані, які потребує програма від користувача — це інформація, що необхідна для авторизації Телеграм бота, для цього потрібно створити бота в телерамі, і отримати токен типу XXXX:YYYY, який буде потрібно використати у візуальному середовищі програми, тобто в текстовому полі «Token». Щоб розпочати роботу з телеграм ботом потрібно ввести команду, зазвичай вони повинні починатись із спеціальних символів, однак в нашому випадку телеграм може отримувати любу інформацію, головне щоб відбулась відправка повідомлення та початкова ініціалізація

Наступний тип вхідної інформації — зображення у любому форматі, який підтримується браузерами, для завантаження зображення, для завантаження зображення в проекті реалізована кнопка «Download», яка відкриє діалогове вікно.

Після редагування фотографії користувачем необхідно буде використати отриманий токен в полі token, а також добавити (за бажанням) опис в полі «Description».

* 1. Опис результуючої інформації

Вихідною інформацією для редактора зображень буде теж в декількох варіантах, оскільки зображення редагуються в мові програмування java script, де запускається вигрузка редагованого зображення, яке буде містити в собі попереднє ім’я файлу з текстовою припискою «edited», оскільки вигрузка відбувається за допомогою браузерного завантаження, то і відповідно результуюче зображення буде знаходитись в папці яку використовує браузер для збереження файлів.

Інший тип вихідної інформації — це JSON модель, яка буде містити в собі конвертоване в base64 зображення, а також поле з описом до зображення та введеним токеном для Телеграм бота. З точки зору користувача вихідна інформація буде отримана у вигляді отриманого повідомлення в телеграм боті, а також, якщо користувач ввів який-небудь текст до фотографії, то до повідомлення теж буде доданий опис.

* 1. Етапи розробки

Загальна схема розробки програмного забезпечення має такі етапи:

– аналіз та збір інформаціїї;

– формування вимог;

– проектування;

– конструювання або розробка архітектури;

– реалізація;

– тестування;

– запуск та написання документації;

– супровід.

На етапі аналізу та збору інформації розробник повинен проаналізувати актуальну обстановку на ринку, розібратись, до якої предметної області входить проєкт, а також його потенціал, вивчити наявних конкурентів на ринку та провести порівняльні дослідження, деталі цього етапу описані в розділі 1.

При формуванні вимог визначаються основні функції, які розробник хоче закріпити в якості основного функціоналу, а також як будуть взаємодіяти користувачі з цим функціоналом. В якості взаємодії з користувачем було використано концепцію веб додатку, а також сервісу Телеграм, користувач може зайти на веб сторінку, загрузити зображення та виконати необхідні дії, використати фільтри та обрізку, після чого отримати зображення за допомогою скачування через браузер або за допомогою експорту до месенджеру Телеграм за допомогою токена бота.

На етапі проектування формуються чіткі вимоги до проєкту, а також до його архітектури, на цьому етапі відбувається синхронізація між поставленими вимогами та кодом, який буде написаний. Для виконання етапу конструювання потрібно використовувати сторонні інструменти, так як UML діаграми, які дозволяють виписати умови у вигляді візуальних зображень,які дають чітке бачення проєкту, його напрямів та функціоналу. Також на етапі конструювання проектується користувацький інтерфейс та загальна логіка проекту. Детальніше цей процес описано у розділі 3.2

Після проєктування настає етап написання коду, у випадку коли робота відбувається за TDD (test-driven development), то паралельно з написанням коду відбувається його покриття тестами, вірніше, код пишеться під конкретні, наперед задані тестові сценарії. Однак, зазвичай, покриття тестами відбувається після написання коду, коли відбувається етап тестування з використанням необхідних технологій, детальніше про технології тестування описано в розділі 4.2.

Коли проект вже написаний і протестований – починається етап супроводу, його суть заключається в введенні проекту в робоче середовище та поступової оптимізації шляхом виявлення дефектів, які не було помічено на розробці тестування.

Отже, якість проектування напряму впливає на майбутнє проєкту, чим більше часу буде витрачено на проектування, тим менше часу буде затрачено на виправлення критичних помилок та післярелізні покращення.

1. ПРОЄКТУВАННЯ СХЕМИ РОБОТИ
   1. Проєктування архітектури програмного коду

Для розробки редактора фотографій було використано мікросервісну архітектуру.

Мікросервіс – це найпростіша одиниця, сервіс, який приймає вхідні запити для здійснення дії. Це може бути бекенд сервіс, який доступний цілодобово та без вихідних, або функція, яка викликається, коли відбувається подія. Простими словами, функція або набір функцій, доступних через певний API через мережу. Отже, це бекенд служба, розгорнута на сервері. У якомусь сенсі це монолітний додаток. Однак він не несе в собі всю функціональність системи, а лише меншу частинку логіки. На відміну від моноліту, отриманий додаток побудований як набір відносно невеликих незалежних служб, що називаються «мікросервісами», які комунікують через комп’ютерну мережу. Можна сказати, мікросервіси – це ті самі логічні модулі монолітного додатка, які розподілені через комп’ютерну мережу, замість того, щоб працювати в рамках одного процесу (пристрою)

Якщо коротко, то архітектурний стиль мікросервісів — це підхід, коли єдиний додаток будується як сукупність невеликих, самодостатніх, незалежних, не тісно зв’язаних сервісів, що спілкуються між собою за допомогою легких механізмів як то HTTP, gRPC, AMQP. Ці сервіси побудовані навколо бізнес-потреб (кожен відповідальний за конкретний процес) та розгортаються незалежно з використанням повністю автоматизованого середовища. Існує абсолютний мінімум централізованого управління цими сервісами. Самі по собі сервіси можуть бути написані на різних мовах і використовувати різні технології зберігання даних.

Одна з причин використання мікросервісів полягає в тому, що компанії хочуть мати можливість швидко щось змінювати, щоб швидше реагувати на зміни бізнес-вимог, випереджати конкурентів. Мікросервіси допомагають розробникам доставляти зміни швидше, безпечніше і з більш високою якістю, тобто зберігати швидкість розвитку продукту, навіть коли той стає неосяжних розмірів

* 1. Розробка структурної схеми проєкту на основі UML-діаграм

Unified Modeling Language - уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, називаної UML-моделлю. UML був створений для визначення, візуалізації, проєктування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація.

В основному, UML - це мова моделювання загального призначення в області розробки програмного забезпечення. Однак тепер він знайшов своє відображення в документації декількох бізнес-процесів або робочих процесів, наприклад, діаграми активності. Тип UML-діаграм можуть використовуватися в якості заміни для блок-схем. Вони забезпечують як більш стандартизований метод моделювання робочих процесів, так і широкий спектр функцій для підвищення зручності читання та ефективності.

На діаграмі ДП.ПО16.ПІ171.09.000.Д1 показано діаграму варіантів вкиористання, її метою є визначення загальних меж так контексту предметної області, а також формулювання загальних вимог до поведінки

На діаграмі ДП.ПО16.ПІ171.09.000.Д2 наведена діаграма класів, яка відображає класи, їх типи даних, їх зміст та відноешення і служить для представлення статичної структури моделі системи в ООП.

Діаграма станів (вона ж Statechart diagram) яка зображена в додатку ДП.ПО16.ПІ171.09.000.Д3 призначена для демонстрації станів та дій, які до них призводять, під час роботи клієнтської частини програми

Перше, що можна побачити – це вхідна та вихідна точки, программма запускається, після чого переходить в стан очікування зображення, потім очікує редагування зображення від користувача, пілся чого чекає моменту, коли користувач експортує редаговане зображення.

Наступною є діаграма активності, яка є зображена в додатку ДП.ПО16.ПІ171.09.000.Д4 – в ній демонструються основні дії, що можуть відбутись під час роботи програми.

На діаграмі послідовності, яка зображенна в наступному додадатку ДП.ПО16.ПІ171.09.000.Д5 зображено послідовність виконання дій, також взаємоді об’єктів впорядкованих за часом. З точки зору послідовності прграма спочатку очікує завантаження зображення від користувача, після чого відображає загружене зображення, після чого зображення стає доступним для редагування. Після редагування зображення може бути експортоване до телеграму, піля чого телеграм відправить статус відправки повідомлення. Після редагування та експорту зображення може бути завантажено на пристрій.

На діаграмі компонентів ( вона ж Component diagram) яка наведена в додатку додатку ДП.ПО16.ПІ171.09.000.Д6 зображено основін компоненти, з якими працює моя програма. Основнив компонентом є клас SpringApplication, адже він піднімає увесь контекст да будує зв’язки між класами, сервісами, репозиторіями та іншими компонентами програми. Другим по важливості компонентом є головний контролер, який містить в собі сервіс експорту,який в свою чергу має декодер-конвертор зображень, який конверує зображення з формату Base64 у байти, після чого зображення відправляється до телеграму.

Діаграма комунікацій (Communication diagram) відображає взаємодії об'єктів без вказування явних зв’язків, а також дозволяє інтуітивно зрозуміти порядок виклику методів , яка зображена в додатку ДП.ПО16.ПІ171.09.000.Д7.

На діаграмі розгортання в додатку ДП.ПО16.ПІ171.09.000.Д8 зображено основні фізичні компоненти, а також деякі додаткові інструменти, що використовуються під час роботи мого курсового проект.

Ці компоненти складаються з сервера ( яким може виступати локальна машина, віддалений сервер або корпоративну мережу).

У разі невеликої корпоративної мережі, або локальної демонстрації сервер вимагає від користувача мінімальних апаратних та програмних вимог, в часності, програмні вимоги можуть зведені до мінімум, оскільки нам достатньо мати лише Docker, всі інші аспекти вже налаштовані і будуть автоматично підтянуті в контейнер Docker. В апаратних вимогах процесора з частотою більше 2Ггц та наявністю 1 ядра буде цілком достатньо, відеократа має містити не менш як 122Мб пам’яті, а також потрібен дисковий простір у розмірі 200Мб.

* 1. Проєктування інтерфейсу

Користувацький інтерфейс (UI)— це простір, де відбувається взаємодія між людьми та машинами. Мета цієї взаємодії полягає у забезпеченні ефективної роботи та управління машиною з людської сторони, тоді як машина одночасно подає назад інформацію, яка допомагає операторам прийняття рішень запустити певний процес.

В основу розробки інтерфейсу програми було покладено стандартну мову гіпертекстової розмітки, а також таблиці стилів, що робить проєкт максимально легким і при тому мати привабливий вигляд.

Тобто візуальне середовище розроблялось в якості веб-інтерфейсів, які можуть приймати вхідні та вихдні дані, забезпечувати виведення, створенням веб сторінок, які передаються інтернетом і проглядаються користувачем за допомогою програми веб-браузера, інтерфейс програми можна побачити на рисунку 3.1.

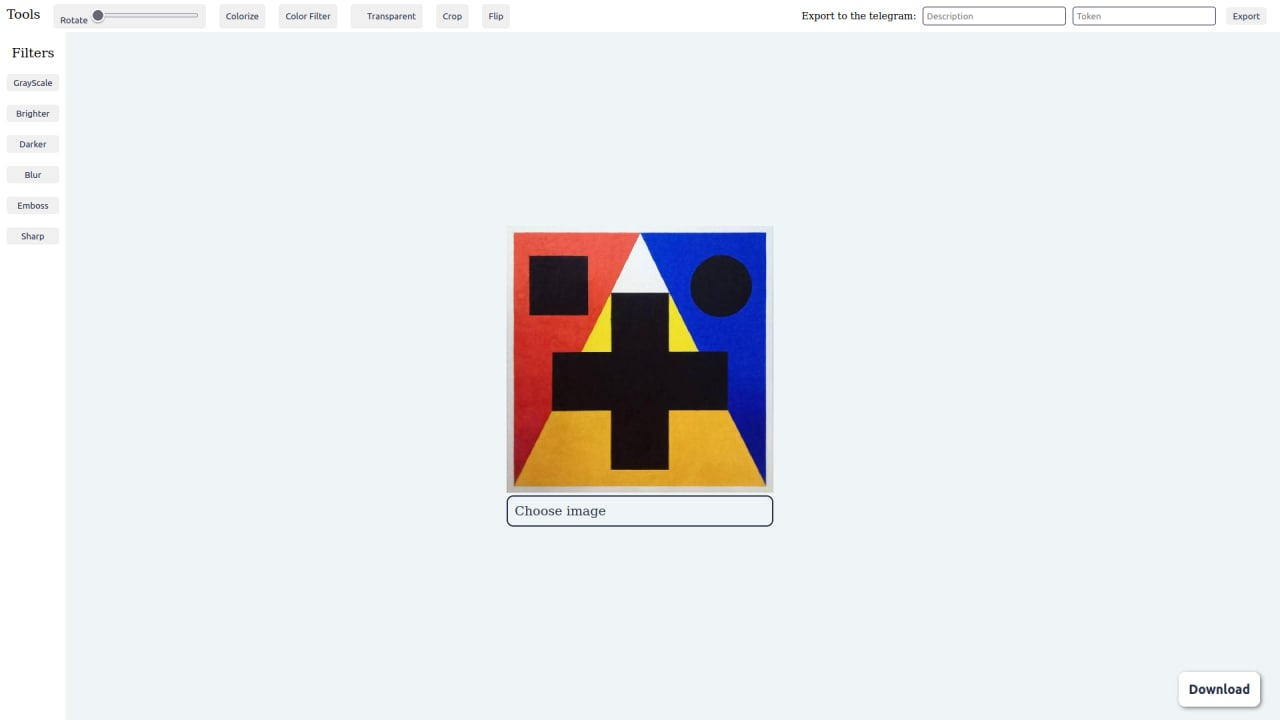


Рисунок 3.1 – Початок роботи

Вебінтерфейси зручні тим, що дають можливість вести спільну роботу співробітникам, які не перебувають в одному офісі (наприклад, вебінтерфейси часто використовуються для заповнення різних баз даних або публікації матеріалів в інтернет-ЗМІ).

Добрим прикладом використання і віддачі вебінтерфейсу є Вікіпедія: практично весь вміст вільної всесвітньої енциклопедії створений і доданий на сторінки сайту за допомогою вебінтерфейсу.

Вебінтерфейс дає можливість універсального віддаленого доступу до служб та пристроїв, у цьому технології практично нема альтернатив. Але водночас, оскільки такий інтерфейс доступний усім, постають серйозні питання забезпечення безпеки, зокрема автентифікація та авторизація користувачів, шифрування переданих даних від сторонніх очей, модерація вмісту тощо.

Проте, проєкт не обмежується лише веб інтерфейсом, тому що має вбудовану інтеграцію з месенджером Телеграм та дозволяє імпортувати до нього редаговані зображення. Для телеграму нам не потрабно будувати інтерфейс, оскільки мессенджер має чудово сконструйований власний інтерфейс.

Отже, цей дипломний проєкт розроблено з використанням HTML та CSS, а також готового інтерфейсу месенджера Телеграм.

1. ПРОГРАМНА РЕАЛІЗАЦІЯ ПРОЄКТНИХ РІШЕНЬ, ТЕСТУВАННЯ ТА ЕКСПЛУАТАЦІЯ
   1. Програмна реалізація проєкту

В прoєкті було використано декілька мов прогармування, а також фреймворк для одної з них: Java, Spring, Java Scrtipt.

Основним JavaScript (JS) — динамічна, об'єктно-орієнтована прототипна мова програмування. Найчастіше використовується для створення сценаріїв вебсторінок, що надає можливість на боці клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд вебсторінки.

JavaScript класифікують як прототипну (підмножина об'єктно-орієнтованої), скриптову мову програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми програмування (імперативну та частково функціональну) і деякі відповідні архітектурні властивості, зокрема: динамічна та слабка типізація, автоматичне керування пам'яттю, прототипне наслідування, функції як об'єкти першого класу.

Мова JavaScript використовується для:

* написання сценаріїв вебсторінок для надання їм інтерактивності;
* програмування на боці сервера (Node.js(Express.js ));
* створення односторінкових та прогресивних вебзастосунків (React, AngularJS, Vue.js;
* стаціонарних застосунків (Electron, NW.js );
* мобільних застосунків (React Native, Cordova);
* сценаріїв в прикладних програмах (наприклад, в програмах зі складу Adobe Creative Suite чи Apache JMeter );
* всередині PDF-документів тощо.

Java — цеоб'єктно-орієнтована мова програмування, випущена 1995 року компанією «Sun Microsystems». В офіційній реалізації Java-програми компілюються у байт-код, який при виконанні інтерпретується віртуальною машиною для конкретної платформи.

«Oracle» надає компілятор Java та віртуальну машину Java, які задовольняють специфікації Java Community Process, під ліцензією GNU General Public License.

Мова значно запозичила синтаксис із C і C++. Зокрема, взято за основу об'єктну модель С++, проте її модифіковано. Усунуто можливість появи деяких конфліктних ситуацій, що могли виникнути через помилки програміста та полегшено сам процес розробки об'єктно-орієнтованих програм. Передусім Java розроблялась як платформо-незалежна мова, тому вона має менше низькорівневих можливостей для роботи з апаратним забезпеченням, що в порівнянні, наприклад, з C++ зменшує швидкість роботи програм. За необхідності таких дій Java дозволяє викликати підпрограми, написані іншими мовами програмування.

Головними концепціями Java є платформа, об’єктивність та безпека.

Платформа заключається в незалежності архітектури, під «незалежністю від архітектури» мається на увазі те, що програма, написана на мові Java, працюватиме на будь-якій підтримуваній апаратній чи системній платформі без змін у початковому коді та перекомпіляції.

Об'єктність в Java є більш об'єктно-орієнтованою ніж C++. Всі дані і дії групуються в класи об'єктів. Виключенням з повної об'єктності є примітивні типи (int, float тощо). Це було свідомим рішенням проєктувальників мови задля збільшення швидкості. Через це Java не вважається повністю об'єктно-орієнтовною мовою.

Безбека є однією з особливостей концепції віртуальної машини полягає в тому, що помилки (виключення) не призводять до повного краху системи. Крім того, існують інструменти, які «приєднуються» до середовища періоду виконання і кожен раз, коли сталося певне виключення, записують інформацію з пам'яті для зневадження програми.

Spring Framework — це програмний каркас (фреймворк) з відкритим кодом та контейнери 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).

В якості аналізу можна сказати, що для кращою мовою для редагування зображень було обрано Java Script, а кращою мовою для написання серверної частини є Java.

* 1. Тестування проєкту

Тестування програмного забезпечення — процес перевірки відповідності заявлених до продукту вимог і реально реалізованої функціональності, здійснюваний шляхом спостереження за його роботою в штучно створених ситуаціях і на обмеженому наборі тестів, обраних певним чином.

Техніка тестування також включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових із метою оцінки

Модульні тести, або unit-тести, розробляють в процесі розробки програмісти та, іноді, тестувальники білої скриньки (white-box testers)Зазвичай unit-тести застосовують для того, щоб упевнитися, що код відповідає вимогам архітектури та має очікувану поведінку.

Модульні тести, або unit-тести, розробляють в процесі розробки програмісти та, іноді, тестувальники білої скриньки (white-box testers).

Зазвичай unit-тести застосовують для того, щоб упевнитися, що код відповідає вимогам архітектури та має очікувану поведінку.

Системне тестування є одним з рівнів [тестування програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F). Системне тестування тестує інтегровану систему для перевірки відповідності всім вимогам. Перевірка повноти та правильності документації користувача є важливою частиною системного тестування. Всі тестові комбінації повинні розроблятися тільки з використанням документації користувача.

Отже, етап тестування є дуже важливим під час розробки програмного забезпечення, оскільки він дає змогу найти недоліки і чітко зрозуміти критичні точки проєкту.

* 1. Інструкція з інсталяції, публікація

Оскільки багатофункціональний редактор-конвертор зображень є веб-додатком, то встановлювати його не потрібно, оскільки він знаходиться на локальному хостингу і доступний за відкритим посиланням, однак проєкт має інтеграцію з телеграмом, а тому потребує встановлення стороннього ПЗ, тобто месенджеру Телеграм.

Tелеграм доступний для різних операційних систем Android/iOS, PC, MAC, LINUX. Завантажити можна з офіційного сайту <https://telegram.org/apps>.

Системні вимоги системні вимоги Telegram Desktop для Ubuntu:

* операційна система Ubuntu;
* оперативна пам’ять 1 ГБ;
* внутрішня пам’ять 100 МБ;

Отож ми бачимо, що в проєкт дійсно легко встановити, а його встановлення не займе багато місця та часу на вашому пристрої.

* 1. Інструкція з експлуатації

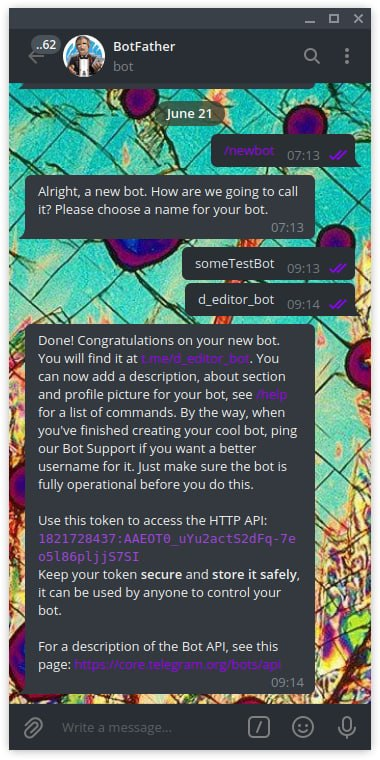
Для початку роботи з багатофункціональним редактором-конвертором зображень нам спочатку потрібно налаштувати створити та налаштувати бота в месенджері телеграм, для цьго розробники телеграму створили спеціального бота, який дозволяє реєструвати інших ботів, він називається BotFather, користувач повинен внести команду /newbot, після чого повинен дати йому ім’я, та унікальну назву, після чого користувач отримує спеціальний, закодованний набір символів, який і є ключем доступу до бота. Процес реєстрації нового бота зображений на рисунку 4.1.

Рисунок 4.1 – Створення бота

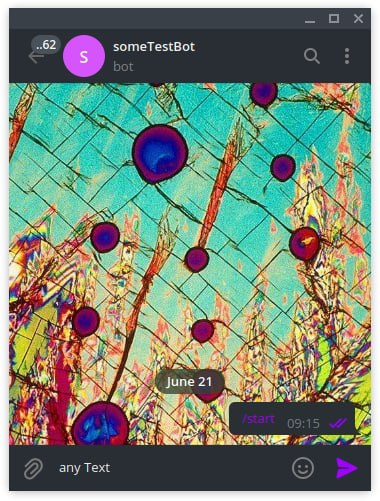
Коли бот вже створений, то потрібна початкова ініціалізація, для цього необхідно зайти на сторінку бота та відправити певний текст, зазвичай телеграм потребує команду «/start», однак в нашому виипадку немає різниці, яку саме команду буде прописано, оскільки ініціалізація всеодно відбудеться, процес ініціалізації ми можемо спостерігати на рисунку 4.2.

Рисунок 425 – Початкова ініціалізація

Пілся успішного створення бота нам достатньо лише скопіювати його токен і ми одразу можемо працювати над редагуванням зображень.

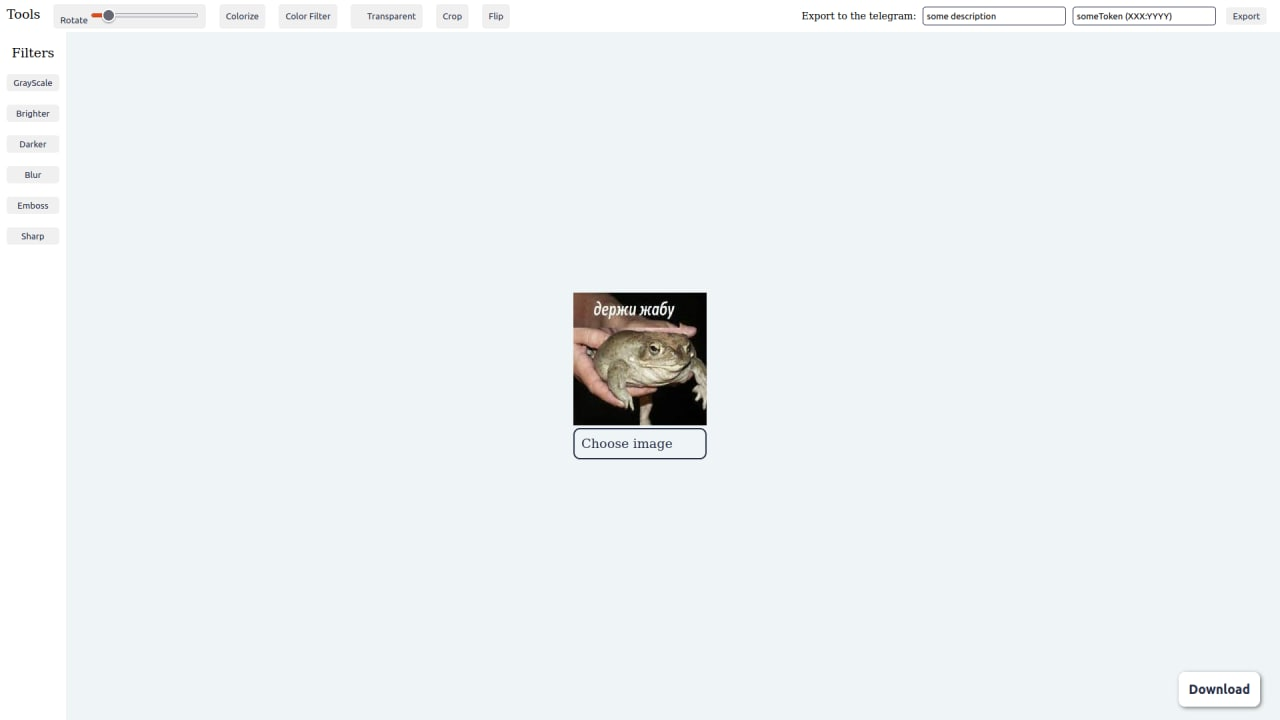
Щоб розпочати редагування користувач повинен перейти у веб-додаток та натиснути клавішу «Choose image», після чого відкриється вікно де можна буде обрати зображення для загрузки. Після заргузки зображення ми можемо починати процес редагування фотографії, а саме використовувати фільтри, обрізати зображення, використовувати зеркальне відображеня, відсіювання фону, а також затемнення і висвітлення загальних кольорів, також можна використовувати градацію сірого і єфект рельєфості. Після редагування фотографії нам потрібно вставити токен у відповідне поле, а також, за бажанням, додати опис до фотографії. Після чого нам потрібно натиснути кнопку «Еxport», що довзолить нам експортувати зображення до телеграму, приклад можна побачити на рисунку 4.3.

Рисунок 4.3 – Експорт зображення

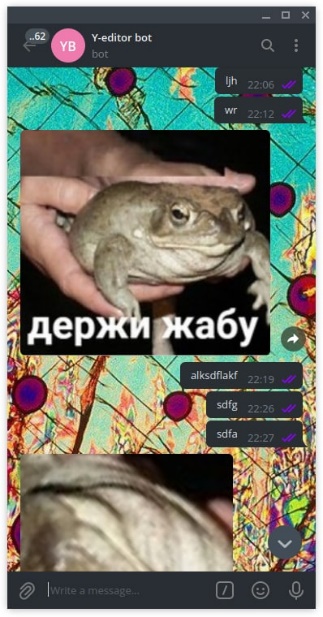
Після експорту фотографії до телеграму в чаті з’явиться повідомлення з фотографією, яку ви редагували, результат експорту можна побачити на рисунку 4.4.

Рисунок 4.4 – Результат експорту зображення

Отже, редактор доволі простий, функціонал легко використовувати, а зображення відображаються після експорту

1. РОЗРАХУНОК ЕКОНОМІЧНИХ ПОКАЗНИКІВ РОЗРОБКИ ІНВЕСТИЦІЙНОЇ ПЛАТФОРМИ
   1. Маркетингові дослідження ринку збуту розробленого програмного продукту

Розробка інвестиційної платформи несе економічний характер, тому будь-яка особа може оперувати фінансовими ресурсами за допомогою цього сервісу. Загалом, проєкт не призначений для отримання прибутку, але зі стрімким зростанням популярності інвестиційно-пошукових платформ, зростає також і попит на них, і з цієї причини доцільно розглянути які ж бувають платформи і який вони несуть прибуток.

Очевидним трендом стала розробка спеціалізованих платформ з можливістю оплати за допомогою крипто валют. Швидкий аналіз показав, що серед комерційних інвестиційних платформ є два ключових напрямки, які розглянемо нижче.

Першим видом є платформи для побудови фінансових взаємовідносин між різними сутностями економічної системи, які в тому чи іншому вигляді призначені для фінансових переводів і, до прикладу, навігації по каталогу інвестиційних сутностей. Гарними прикладом можна платформи для пошуку інвестицій, де користувач може створити одну з декількох сутностей в межах платформи. Такі як фінансовий фонд та інвестиційну фірму. Перша сутність – це елемент платформи, який являє собою фінансову одиницю, що може налагоджувати фінансування другої сутності. В свою чергу Друга сутність, інвестиційна фірма, являється центром накопиченням коштів, для виконання певної роботи. Її основним джерелом ресурсів є фінансовий фонд, виконання роботи чи певних виробничих планів в тій чи іншій мірі має задовольняти запити та очікування даного стейкхолдера.

Другим видом є платформи для дрібного інвестування. Основним джерелом надходження фінансів є сам користувач. В таких платформах зазвичай наявні каталоги сутностей, які потребують мінімального фінансування, задля виконання певного поставленого перед собою завдання: розробка нового технічного засобу, побудова новітньої моделі для роботи з великими даними (Big Data) чи створення вузькоспеціалізованих інструментів для використання в медичній сфері.

На комерційних платформах можна заробляти, але важливо знайти правильний спосіб монетизації наданих послуг. До прикладу можна використати наступні способи монетизації: обмеження доступу до використання функціоналу латентними заборонами з цілю змушення користувача придбати ліцензію в межі платформи, або відраховувати відсоток з коштів, якими оперує користувач у вигляді комісії, яка буде визначатись відносно об’єму коштів які існують в циклі роботи з платформою

* 1. Розрахунок вартості програмного забезпечення

Щоб визначити вартість розробки програмного продукту потрібно розбити процес на невеликі проміжки виконання, атакож провести оцінку тривалості робіт на кожному цих етапів. В таблиці 5.1 наведено основні етапи робіт при створенні ПЗ

Таблиця 5.1 – Етапи розробки ПЗ

|  |  |
| --- | --- |
| Етап | Вид виконуваних робіт |
| 1 | Обґрунтування необхідності розробки системи |
| 2 | Пошук та аналіз існуючих методів та рішень |
| 3 | Вибір та обгрунтування технологій |
| 4 | Проєктування системи |
| 5 | Кодування |
| 6 | Тестування |
| 7 | Впровадження ПЗ |
| 8 | Написання документації |

Розрахунок тривалості кожного етапу проводиться в таблиці 5.2.

Таблиця 5.2 – Розрахунок тривалості робіт

|  |  |  |  |
| --- | --- | --- | --- |
| Етап | Мінімальна кількість, днів | Максимальна кількість, днів | Фактична кількість, днів |
| 1 | 1 | 3 | 1 |
| 2 | 3 | 7 | 5 |
| 3 | *2* | 6 | 4 |
| 4 | *10* | 20 | 15 |
| 5 | 11 | 21 | 18 |
| 6 | 2 | 4 | 3 |
| 7 | 2 | 5 | 4 |
| 8 | 2 | 5 | 4 |
| Всього | 36 | 71 | 54 |

Отже , розробка платформи для пошуку інвестицій зайняла 54 робочих днів, що вкладається в межі тривалості розробки проєкту.

До складу кошторису витрат на розробку програмного забезпечення включають вартість усіх витрат необхідних для реалізації комплексу робіт, що складають зміст даної розробки.

Витрати містять наступні елементи:

основна заробітна плата;

додаткова заробітна плата;

єдиний соціальtий внесок;

амортизація;

інші операційні витрати.

Над проєктом працював один розробник протягом 54 днів з місячним посадовим окладом 10000 грн. Основна заробітна плата за весь період розробки розраховується за наступною формулою:

(5.1)

Тобто в результаті загальна заробітня плата становитиме:

Додаткова заробітна плата становить 11% від основної заробітної плати і дорівнює

Згідно чинного законодавства єдиний соціальний внесок становить 22% від фонду заробітної плати (суму основної та додаткової заробітної плати) та розраховується за формулою:

(5.2)

Соціальний внесок при тому становить:

Нарахування амортизації проводиться для електронно-обчислювальної техніки та програмного забезпечення згідно п. 145.1 Податкового кодексу [8] встановлений мінімально допустимий термін експлуатації ЕОТ та ПЗ – 2 роки.

Амортизація – процес поступового перенесення вартості основних засобів на продукт, що виготовляється з їх допомогою.

Встановимо термін експлуатації ЕОТ 7 років. Тоді річна норма амортизації становитиме:

Балансова вартість основних засобів, до яких входить ноутбук ASUS X552E та мережевий маршрутизатор TP-Link наведена в табл. 5.3.

Таблиця 5.3 – Вартість основних засобів розробки

|  |  |  |
| --- | --- | --- |
| Засіб | Вид засобу | Вартість засобу або ліцензії на його використання, гривень |
| ASUS X552E | Пристрій (ноутбук) | 14500 |
| TP-Link | Мережевий маршрутизатор | 1000 |
| Всього | | 15500 |

Річні амортизаційні відрахування розраховуються за формулою

(5.3)

Оскільки розробка проєкту тривала 55 днів, то амортизаційні відрахування на період розробки становлять 2 216,5 грн.

Інші операційні витрати належать до непрямих витрат та дорівнюють 15% від основної заробітної плати.

Згідно із проведеними розрахунками, загальний кошторис витрат на створення ПЗ наведено у таблиці 5.4.

Таблиця 5.4 – Кошторис витрат

|  |  |
| --- | --- |
| Елементи витрат | Сума, гривень |
| Основна заробітна плата |  |
| Додаткова заробітна плата | 2700 |
| Єдиний соціальний внесок |  |
| Амортизація | 329 |
| Інші витрати |  |
| Всього | 37 249,26 |

Таким чином, вартість розробки платформи для пошуку інвестицій складатиме 37 249,26грн.

* 1. Оцінка економічної ефективності впровадження ПЗ

Визначимо вартість грошового потоку та сумарну вартість варіанту інвестування за ставкою дисконтування 10%, дані наведено у таблиці 5.5.

Обрахуємо грошові потоки від кількості користувачів платформи. Кошти нараховуються при проведенні кожної транзакції користувачів у розмірі 4%.

Грошовий потік для кожного року розрахуємо за формулою:

(5.4)

де Kк – кількість користувачів платформою,

Ас – кількість транзакцій в середньому.

Визначимо середню кількість транзакцій – 230 для кожного користувача.

Для першого року грошовий потік становитиме:

Для другого року грошовий потік становитиме:

Для третього року грошовий потік становитиме:

Таблиця 5.5 – Вартість грошового потоку

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Роки інвестування | Кількість користувачів | Грошовий потік, грн | Коефіцієнт дисконтування | Теперішня вартість грошового потоку, грн |
| 1 |  |  | 0,909 | 33690 |
| 2 |  |  | 0,824 | 59230 |
| 3 |  |  | 0,751 | 82620 |
| Всього | 6500 | 215760 | x | 175540 |

Теперішня вартість грошового потоку визначається за формулою:

 (5.5)

де ГПі – грошовий потік і-того року, тис.грн,

t – роки інвестування,

Кді – коефіцієнт дисконтування, що визначається за формулою:

 (5.6)

де  - ставка дисконтування.

Далі розрахуємо чистий приведений дохід від реалізації проєкту:

(5.7)

де ІК – інвестовані кошти.

Отже, приведений дохід при реалізацій продукту протягом 3 років становить грн.

Період окупності розрахуємо за формулою:

(5.8)

Згідно розрахунків період окупності проєкту складає близько 5,12 років.

В даному розділі було проведено маркетингове дослідження ринку збуту розробленого програмного продукту, визначено трудомісткість розробки, кошторис витрат, та проведено оцінку економічної ефективності впровадження програмного забезпечення.

Розрахунки свідчать, що даний проєкт, в цілому, є прибутковим та економічно вигідним. Протягом 3 років реалізації планується отримати грошові надходження в сумі 175540 грн, з них чистий дохід складатиме грн. Інвестовані кошти повертаються в період 0,84 років.

ВИСНОВКИ

В даному дипломному проєкті було реалізовано мінімальний проект на тему: «Багатофункціональний редактор-конвертор зображень», який дозволяє легко і зручно редагувати зображення різних типів, застосовувати до них різноманітні фільтри, використовувати замінення кольорів, градації сірого, затемнювати та висвітлювати фотографії, а також дозволить обрізати фотографію, та змінювати положення..

Оскільки конкуренція на даний час у проєкта величезна, то в проєкті можна виділити свої переваги і недоліки. До переваг проєкту можна віднести:

відносна простота, можна швидко і легко відредагувати зображення;

відсутність потреби встановлювати додаток, оскільки можна просто зайти на веб сторінку;

зручний інтерфейс;

інтеграція з месенджером Телеграм, можливість відправляти відредаговані зображення.

Недоліки даного програмного забезпечення:

* відносно не великий функціонал;
* відсутність можливості додавати текст, а також смайлики до фотографій;
* потрібне підключення до мережі.

Проєкт задовольняє поставлені вимоги, відповідє ТЗ. Розроблено та протестовано проєкт, який дозволяє редагувати зображення та відправляти їх у якості повідомлень.

СПИСОК ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

* 1. Бенджамин Еванс, Девид Фланаган – Java in a nutshell. 6th edition O`Reilly, 2015 – 415 с.
  2. Бенджамин Мучко. – Gradle in action. 1st edition – Manning, 2014 – 480 с.
  3. Герберт Шилдт - JavaScript. Полное руководство. 11th edition – М.: И.Д. Вильямс, 2018 – 1899 с
  4. Джошуа Блох. Effective Java programming language guide. 3rd edition – М.: И.Д. Вильямс, 2016 – 440 с.
  5. Єгор Бугаєнко – Elegant Objects, Volume 1 – yegor256, 2017 – 229 с
  6. Відео-конференція «Ануар Нурмаканов — Event Sourcing и CQRS на конкретном примере» - https://www.youtube.com/watch?v=AKGT7wkVd34
  7. Відео-лекція «Евгений Борисов – Spring-потрошитель, часть 1» - <https://www.youtube.com/watch?v=BmBr5diz8WA>
  8. Відео-лекція «Евгений Борисов – Spring-потрошитель, часть 2» -https://www.youtube.com/watch?v=cou\_qomYLNU
  9. Відео-лекція «Евгений Борисов – Spring Patterns» - <https://www.youtube.com/watch?v=61duchvKI6o>
  10. Відео-лекція «Евгений Борисов – Spring – Глубоко и не очень» - <https://www.youtube.com/watch?v=nGfeSo52_8A>
  11. Відео-лекція «Евгений Борисов, Кирилл Толкачев – Проклятие Spring Test» - <https://www.youtube.com/watch?v=7mZqJShu_3c>
  12. Відео-лекція «Егор Бугаенко – Объектно-ориентированое вранье» - <https://www.youtube.com/watch?v=lfdAwl3-X_c>.
  13. Інтернет-ресурс для вивчення програмування – <https://metanit.com/>
  14. Інтернет-форум для пошуку вирішення запитань прикладного програмування - <https://stackoverflow.com/>

ДОДАТКИ

Додаток А – Лістинг програм

<!DOCTYPE html>

<html lang="en" dir="ltr">

<head>

<meta charset="utf-8">

<title>Image editor</title>

<script src="./src/foto.min.js"></script>

<script src="./src/index.js"></script>

<link rel="stylesheet" type="text/css" href="styles.css">

</head>

<body>

<div class="functionality">

<div class="operations-wrapper">

<span>Tools</span>

<button class="operation-btn">

Rotate

<input type="range" id="rotate-range" value="0" min="0" max="360" onchange="rotate(this)"/>

</button>

<input type="color" id="color-picker" value="#000000" oninput="makeColorize(this)"/>

<button class="operation-btn" onclick="openColorpicker()">Colorize</button>

<input type="color" id="colorize-color-picker" value="#000000" oninput="applyColorFilter(this)"/>

<button class="operation-btn" onclick="openColorFilterPicker()">Color Filter</button>

<button class="operation-btn" onclick="makeTransparent()">

<div id="color-preview"></div>

Transparent

</button>

<button class="operation-btn" onclick="crop()">Сrop</button>

<button class="operation-btn" onclick="flipVertically()">Flip</button>

</div>

<div class="telegram-export">

Export to the telegram:

<input placeholder="Description" type="text" id="desc">

<input placeholder="Token" type ="text" id="token">

<button class="operation-btn" onclick="telegramExport()">Export</button>

</div>

</div>

<div class="sidebar">

<span>Filters</span>

<button class="operation-btn" onclick="makeGrayScale()">GrayScale</button>

<button class="operation-btn" onclick="makeBright()">Brighter</button>

<button class="operation-btn" onclick="makeDark()">Darker</button>

<button class="operation-btn" onclick="makeBlur()">Blur</button>

<button class="operation-btn" onclick="makeEmboss()">Emboss</button>

<button class="operation-btn" onclick="makeSharp()">Sharp</button>

</div>

<div class="image-wrapper">

<div class="image-editor">

<div class="img-container">

<img src="ERcdr7iX0AYGTI0.jpeg" id="foto-image" class="image" />

</div>

<div class="file-upload-section" onclick="selectImage()">

<input id="foto-file" type="file">

Choose image

</div>

</div>

<button class="download-btn operation-btn" onclick="download()"> Download </button>

</div>

</body>

</html>

body{

background-color: rgb(239, 244, 246);

margin: 0;

font-family: 'Roboto';

}

button{

cursor: pointer;

}

.functionality{

height: 3rem;

background-color: #ffffff;

display: flex;

justify-content: space-between ;

align-items: center;

padding: 0 10px;

}

.operations-wrapper{

display: flex;

margin-left: -10px;

}

.operations-wrapper span{

margin: 0 10px;

padding-top: 4px;

font-size: 1.2rem;

}

.operation-btn{

color: rgb(38, 45, 69);

border: none;

margin: 0 10px;

padding: 5px 10px;

border-radius: 5px;

}

.image-wrapper{

height: calc(100vh - 3rem);

display: flex;

justify-content: center;

align-items: center;

position: relative;

}

.download-btn{

position: absolute;

right: 20px;

bottom: 20px;

border-radius: 10px;

padding: 15px;

background-color: #ffffff;

font-weight: 600;

font-size: 1.2rem;

box-shadow: 2px 2px 5px rgba(0,0,0,0.5);;

}

.file-upload-section{

color: rgb(38, 45, 69);

cursor: pointer;

font-family: 'Roboto';

font-size: 1.2rem;

padding: 10px;

border-radius: 10px;

border: 2px solid rgb(38, 45, 69);

}

.file-upload-section:hover{

background-color: #ffffff;

}

#foto-file {

display: none;

}

.telegram-export{

font-family: 'Roboto';

font-size: .9rem;

display: flex;

align-items: center;

}

.telegram-export input{

border-radius: 5px;

padding: 5px;

border: 1px solid rgb(38, 45, 69);

margin: 0 5px;

}

.telegram-export input:first-child{

margin-left: 10px;

}

.sidebar{

height: calc(100vh - 3rem);

position: fixed;

display: flex;

padding: 10px;

flex-direction: column;

background-color: #ffffff;

text-align: center;

z-index: 1000;

}

.sidebar span{

margin: 10px 0 ;

font-size: 1.2rem;

}

.sidebar .operation-btn{

margin: 10px 0;

}

.image{

max-height: calc(100vh - 100px);

width: auto;

}

#color-picker, #colorize-color-picker {

display: none;

}

#color-preview{

float: left;

width: 15px;

height: 15px;

}

class Foto {

constructor() {

var t = this;

this.operationOrgCanvas = document.createElement("canvas"), this.operationOrgCtx = this.operationOrgCanvas.getContext("2d"), this.operationEditedCanvas = document.createElement("canvas"), this.operationEditedCtx = this.operationEditedCanvas.getContext("2d"), this.fileInput = document.getElementById("foto-file"), this.fileInput.addEventListener("change", function(e) {

t.loadImage()

}), this.image = null, this.imageData = null, this.imageWidth = 0, this.imageHeight = 0, this.convertedToGrayScale = !1, this.previewImageElement = null, this.redPixelMatrix = [], this.greenPixelMatrix = [], this.bluePixelMatrix = [], this.alphaPixelMatrix = [], this.pickedR = "", this.pickedG = "", this.pickedB = "", this.selectedFileName = "", this.selectStart = !1, this.startX = "", this.startY = "", this.endX = "", this.endY = "", this.excludeArea = !1, this.relativeStartX = "", this.relativeStartY = "", this.relativeEndX = "", this.relativeEndY = "", this.pickedR = null, this.pickedG = null, this.pickedB = null, this.selectRect = document.createElement("div"), document.body.appendChild(this.selectRect), this.oldSelectedColorForColorize = null, this.ctrlPressed = !1;

t = this;

document.addEventListener("keydown", function(e) {

17 == e.keyCode && (t.ctrlPressed = !0)

}), document.addEventListener("keyup", function(e) {

t.ctrlPressed = !0

})

}

loadImage() {

var t = document.getElementById("foto-file");

this.selectedFileName = t.files.item(0).name;

var e = new FileReader,

a = this;

e.onload = function(t) {

a.image = new Image, a.image.onload = function() {

a.imageWidth = a.image.width, a.imageHeight = a.image.height, a.operationOrgCanvas.width = a.imageWidth, a.operationOrgCanvas.height = a.imageHeight, a.operationEditedCanvas.width = a.imageWidth, a.operationEditedCanvas.height = a.imageHeight, a.imageData = [], a.operationOrgCtx.clearRect(0, 0, a.operationOrgCanvas.width, a.operationOrgCanvas.height), a.operationEditedCtx.clearRect(0, 0, a.operationEditedCanvas.width, a.operationEditedCanvas.height), a.operationOrgCtx.drawImage(a.image, 0, 0), a.operationEditedCtx.drawImage(a.image, 0, 0), a.previewImage(a.operationOrgCanvas, 0), a.imageData = a.operationOrgCtx.getImageData(0, 0, a.operationOrgCanvas.width, a.operationOrgCanvas.height), a.generatePixelMatrix(), console.log("Pixel Data Loaded")

}, a.image.src = t.target.result

}, e.readAsDataURL(t.files[0])

}

generatePixelMatrix() {

var t = [],

e = [],

a = [],

i = [];

this.redPixelMatrix = [], this.greenPixelMatrix = [], this.bluePixelMatrix = [], this.alphaPixelMatrix = [];

for (var r = 0; r < this.imageData.data.length; r += 4) r / 4 % this.imageWidth == 0 && (0 != r && (this.redPixelMatrix.push(t), this.greenPixelMatrix.push(e), this.bluePixelMatrix.push(a), this.alphaPixelMatrix.push(i)), t = [], e = [], a = [], i = []), t.push(this.imageData.data[r]), e.push(this.imageData.data[r + 1]), a.push(this.imageData.data[r + 2]), i.push(this.imageData.data[r + 3])

}

grayscale() {

for (var t = this.imageData, e = 0; e < t.data.length; e += 4) {

var a = t.data[e],

i = t.data[e + 1],

r = t.data[e + 2];

t.data[e + 3];

t.data[e] = (a + i + r) / 3, t.data[e + 1] = (a + i + r) / 3, t.data[e + 2] = (a + i + r) / 3

}

this.operationEditedCtx.putImageData(t, 0, 0), this.operationOrgCtx.putImageData(t, 0, 0), this.previewImage(), this.convertedToGrayScale = !this.convertedToGrayScale

}

makeBright() {

for (var t = this.imageData, e = 0; e < t.data.length; e += 4) {

var a = t.data[e],

i = t.data[e + 1],

r = t.data[e + 2],

s = t.data[e + 3];

t.data[e] = a + 10, t.data[e + 1] = i + 10, t.data[e + 2] = r + 10, t.data[e + 3] = s

}

this.operationEditedCtx.putImageData(t, 0, 0), this.previewImage()

}

makeDark() {

for (var t = this.imageData, e = 0; e < t.data.length; e += 4) t.data[e] -= 10, t.data[e + 1] -= 10, t.data[e + 2] -= 10, t.data[e + 3] -= 10;

this.operationEditedCtx.putImageData(t, 0, 0), this.previewImage()

}

makeTransparent() {

for (var t = this.imageData, e = 0; e < t.data.length; e += 4) Math.abs(t.data[e] - this.pickedR) < 30 && Math.abs(t.data[e + 1] - this.pickedG) < 30 && Math.abs(t.data[e + 2] - this.pickedB) < 30 && (t.data[e + 3] = 0);

this.operationEditedCtx.putImageData(t, 0, 0), this.previewImage()

}

applyFilter(t) {

for (var e = 0; e < this.imageData.data.length; e += 4) {

var a = parseInt(e / 4 / this.imageWidth),

i = e / 4 % this.imageWidth;

if (0 != a && 0 != i && a != this.imageHeight - 1 && i != this.imageWidth - 1) {

for (var r = 0, s = 0, h = 0, o = 0, d = 0; d < 3; d++)

for (var n = 0; n < 3; n++) null != this.redPixelMatrix[a + (d - 1)] && null != this.redPixelMatrix[a + (d - 1)][i + (n - 1)] && (r += t[d][n] \* this.redPixelMatrix[a + (d - 1)][i + (n - 1)], s += t[d][n] \* this.greenPixelMatrix[a + (d - 1)][i + (n - 1)], h += t[d][n] \* this.bluePixelMatrix[a + (d - 1)][i + (n - 1)], o += t[d][n] \* this.alphaPixelMatrix[a + (d - 1)][i + (n - 1)]);

this.convertedToGrayScale ? (this.imageData.data[e] = (r + s + h) / 3, this.imageData.data[e + 1] = (r + s + h) / 3, this.imageData.data[e + 2] = (r + s + h) / 3, this.imageData.data[e + 3] = o) : (this.imageData.data[e] = r, this.imageData.data[e + 1] = s, this.imageData.data[e + 2] = h, this.imageData.data[e + 3] = o)

}

}

this.operationEditedCtx.putImageData(this.imageData, 0, 0), this.previewImage()

}

applyBlurFilter() {

this.applyFilter([

[1 / 9, 1 / 9, 1 / 9],

[1 / 9, 1 / 9, 1 / 9],

[1 / 9, 1 / 9, 1 / 9]

])

}

applyEmbossFilter() {

this.applyFilter([

[-2, -1, 0],

[-1, 1, 1],

[0, 1, 2]

])

}

applySharpFilter() {

this.applyFilter([

[0, -1, 0],

[-1, 5, -1],

[0, -1, 0]

])

}

applyVintageFilter() {

this.colorFilter("#0000ff"), this.colorFilter("#0000ff"), this.colorFilter("#ec8900")

}

applyCustom() {

this.applyFilter([

[-1, -1, -1],

[2, 2, 2],

[-1, -1, -1]

])

}

flipVertically() {

this.operationEditedCtx.translate(this.imageWidth, 0), this.operationEditedCtx.scale(-1, 1), this.operationEditedCtx.drawImage(this.image, 0, 0), this.operationOrgCtx.translate(this.imageWidth, 0), this.operationOrgCtx.scale(-1, 1), this.operationOrgCtx.drawImage(this.image, 0, 0), this.imageData = this.operationOrgCtx.getImageData(0, 0, this.operationOrgCanvas.width, this.operationOrgCanvas.height), this.generatePixelMatrix(), this.previewImage()

}

flipHorizontally() {

this.operationEditedCtx.translate(0, this.imageHeight), this.operationEditedCtx.scale(1, -1), this.operationEditedCtx.drawImage(this.image, 0, 0), this.operationOrgCtx.translate(0, this.imageHeight), this.operationOrgCtx.scale(1, -1), this.operationOrgCtx.drawImage(this.image, 0, 0), this.imageData = this.operationOrgCtx.getImageData(0, 0, this.operationOrgCanvas.width, this.operationOrgCanvas.height), this.generatePixelMatrix(), this.previewImage()

}

rotate(t) {

this.operationEditedCtx.clearRect(0, 0, this.operationEditedCanvas.width, this.operationEditedCanvas.height),

this.operationEditedCtx.save(),

this.operationEditedCtx.translate(this.imageWidth / 2, this.imageHeight / 2),

this.operationEditedCtx.rotate(t \* Math.PI / 180),

this.operationEditedCtx.drawImage(this.image, -this.image.width / 2, -this.image.width / 2),

this.operationEditedCtx.restore(),

this.operationOrgCtx.clearRect(0, 0, this.operationOrgCanvas.width, this.operationOrgCanvas.height),

this.operationOrgCtx.save(), this.operationOrgCtx.translate(this.imageWidth / 2, this.imageHeight / 2),

this.operationOrgCtx.rotate(t \* Math.PI / 180),

this.operationOrgCtx.drawImage(this.image, -this.image.width / 2, -this.image.width / 2),

this.operationOrgCtx.restore(), this.imageData = this.operationOrgCtx.getImageData(0, 0,

this.operationOrgCanvas.width, this.operationOrgCanvas.height), this.generatePixelMatrix(), this.previewImage()

}

export () {

var t = document.createElement("a");

t.download = this.selectedFileName + "-edited.png", t.href = this.operationEditedCanvas.toDataURL(), t.click()

}

genereateImageForExport () {

return this.operationEditedCanvas.toDataURL();

}

previewImage(t, e, a) {

this.previewImageElement = document.getElementById("foto-image"), this.previewImageElement.setAttribute("draggable", !1);

var i = this;

null != e && 0 == e && (this.previewImageElement.addEventListener("mouseover", function(t) {

this.style.cursor = "crosshair"

}), this.previewImageElement.addEventListener("click", function(t) {

i.relativeStartX = t.offsetX, i.relativeStartY = t.offsetY, i.ctrlPressed && i.pickColorPixel(i.relativeStartX, i.relativeStartY), i.selectStart = !1

}), this.previewImageElement.addEventListener("mousedown", function(t) {

i.selectStart = !0, i.startX = t.clientX, i.startY = t.clientY, i.relativeStartX = t.offsetX, i.relativeStartY = t.offsetY

}), this.previewImageElement.addEventListener("mousemove", function(t) {

i.endX = t.clientX, i.endY = t.clientY, i.selectStart && (i.selectRect.style.position = "fixed", i.selectRect.style.display = "initial", i.selectRect.style.border = "2px dashed black", i.selectRect.style.top = i.startY + "px", i.selectRect.style.left = i.startX + "px", i.selectRect.style.height = i.endY - i.startY + "px", i.selectRect.style.width = i.endX - i.startX + "px")

}), this.previewImageElement.addEventListener("mouseup", function(t) {

i.relativeEndX = t.layerX, i.relativeEndY = t.layerY, i.selectStart = !1, i.selectRect.style.height = "0px", i.selectRect.style.width = "0px", i.selectRect.style.display = "none"

}), this.selectRect.addEventListener("mouseup", function(t) {

i.selectStart = !1

})), this.previewImageElement.src = null == t ? i.operationEditedCanvas.toDataURL() : t.toDataURL()

}

recreateImageObject() {

this.image = new Image, this.image.src = this.operationOrgCanvas.toDataURL()

}

pickColorPixel(t, e) {

var a = this.previewImageElement.width,

i = this.previewImageElement.height,

r = this.imageWidth / a,

s = this.imageHeight / i,

h = parseInt(t \* r),

o = parseInt(e \* s),

d = this.operationOrgCtx.getImageData(h, o, 1, 1).data;

this.pickedR = d[0], this.pickedG = d[1], this.pickedB = d[2], this.pickedA = d[3], document.getElementById("color-preview").style.background = "rgb(" + this.pickedR + ", " + this.pickedG + ", " + this.pickedB + ")"

}

applyColorFilter(t) {

for (var e = .5 \* parseInt(t.substr(1, 2), 16), a = .5 \* parseInt(t.substr(3, 2), 16), i = .5 \* parseInt(t.substr(5, 2), 16), r = this.imageData, s = 0; s < r.data.length; s += 4) r.data[s] <= e && (r.data[s] = e), r.data[s + 1] <= a && (r.data[s + 1] = a), r.data[s + 2] <= i && (r.data[s + 2] = i);

this.operationEditedCtx.putImageData(r, 0, 0), this.operationOrgCtx.putImageData(r, 0, 0), this.previewImage()

}

colorize(t) {

var e = .5 \* parseInt(t.substr(1, 2), 16),

a = .5 \* parseInt(t.substr(3, 2), 16),

i = .5 \* parseInt(t.substr(5, 2), 16);

null != this.oldSelectedColorForColorize && (e = -parseInt(this.oldSelectedColorForColorize.substr(1, 2), 16) + e, a = -parseInt(this.oldSelectedColorForColorize.substr(3, 2), 16) + a, i = -parseInt(this.oldSelectedColorForColorize.substr(3, 2), 16) + i), this.oldSelectedColorForColorize = t;

for (var r = this.imageData, s = 0; s < r.data.length; s += 4)

r.data[s] += e, r.data[s + 1] += a, r.data[s + 2] += i;

this.operationEditedCtx.putImageData(r, 0, 0), this.operationOrgCtx.putImageData(r, 0, 0), this.previewImage()

}

cropSelected() {

var t = this.previewImageElement.width,

e = this.previewImageElement.height,

a = this.imageWidth / t,

i = this.imageHeight / e,

r = this.relativeStartX \* a,

s = this.relativeStartY \* i,

h = parseInt(parseInt(this.selectRect.style.width.replace(/\D/g, "")) \* a),

o = parseInt(parseInt(this.selectRect.style.height.replace(/\D/g, "")) \* i),

d = this.operationEditedCtx.getImageData(r, s, h, o),

n = this.operationOrgCtx.getImageData(r, s, h, o);

this.operationEditedCtx.clearRect(0, 0, this.operationEditedCanvas.width, this.operationEditedCanvas.height), this.operationOrgCtx.clearRect(0, 0, this.operationOrgCtx.width, this.operationOrgCtx.height), this.operationEditedCanvas.width = h, this.operationEditedCanvas.height = o, this.operationOrgCanvas.width = h, this.operationOrgCanvas.height = o, this.operationEditedCtx.putImageData(d, 0, 0), this.operationOrgCtx.putImageData(n, 0, 0), this.imageWidth = h, this.imageHeight = o, this.imageData = this.operationOrgCtx.getImageData(0, 0, this.operationOrgCanvas.width, this.operationOrgCanvas.height), this.generatePixelMatrix(), this.selectRect.style.display = "none", this.previewImage()

}

}

var foto;

window.onload = function(){

foto = new Foto();

}

function selectImage() {

document.getElementById('foto-file').click();

}

function makeGrayScale() {

foto.grayscale();

}

function makeBright() {

foto.makeBright();

}

function makeDark() {

foto.makeDark();

}

function makeBlur() {

foto.applyBlurFilter();

}

function makeEmboss() {

foto.applyEmbossFilter();

}

function makeSharp() {

foto.applySharpFilter();

}

function download() {

foto.export();

}

function openColorpicker() {

document.getElementById("color-picker").click();

}

function makeColorize(elem) {

var color = elem.value;

foto.colorize(color);

}

function openColorFilterPicker(){

document.getElementById('colorize-color-picker').click();

}

function applyColorFilter(elem) {

var color = elem.value;

foto.applyColorFilter(color);

}

function makeTransparent() {

foto.makeTransparent();

}

function crop() {

foto.cropSelected();

}

function flipVertically() {

foto.flipVertically();

}

function rotate(elem) {

var someValue = elem.value;

foto.rotate(someValue);

}

function telegramExport() {

contents = foto.genereateImageForExport();

var xhr = new XMLHttpRequest();

xhr.open("POST", "http://localhost:8899/api/image/receive", true);

xhr.setRequestHeader('Content-Type', 'application/json');

console.log("Image that will send to server: " + contents);

xhr.send(JSON.stringify({

imageBase64: contents,

token: document.getElementById('token').value,

description: document.getElementById('desc').value

}));

}

package io.ibot.image.store.service;

import java.io.IOException;

import java.util.Base64;

import org.springframework.stereotype.Service;

@Service

public class ImageService {

public byte[] convertBase64StringToByte(String base64Image) throws IOException {

String split = base64Image.split(",")[1];

String imageDataString = split.substring(0, split.length()-2);

byte[] decodedBytes = Base64.getDecoder().decode(imageDataString);

return decodedBytes;

}

}

package io.ibot.image.store.service;

import java.awt.image.BufferedImage;

import java.io.ByteArrayInputStream;

import java.io.File;

import java.io.IOException;

import javax.imageio.ImageIO;

import org.springframework.stereotype.Service;

import org.springframework.web.client.RestTemplate;

import com.pengrad.telegrambot.TelegramBot;

import com.pengrad.telegrambot.UpdatesListener;

import com.pengrad.telegrambot.model.request.ParseMode;

import com.pengrad.telegrambot.request.SendPhoto;

import com.pengrad.telegrambot.response.SendResponse;

import io.ibot.image.store.entity.model.Update;

import lombok.RequiredArgsConstructor;

import lombok.extern.slf4j.Slf4j;

@Slf4j

@Service

@RequiredArgsConstructor

public class TelegramBotService {

private final RestTemplate restTemplate;

private Integer chatId;

public void sendImageToTelegram(byte[] imageInByte, String token, String description) {

if (chatId == null) {

getChatId(token);

}

TelegramBot bot = new TelegramBot(token);

bot.setUpdatesListener(updates -> {

return UpdatesListener.CONFIRMED\_UPDATES\_ALL;

});

log.info("chatId in sendPhoto: {}", chatId);

SendPhoto request = new SendPhoto(chatId, generateImageFile(imageInByte))

.parseMode(ParseMode.HTML)

.disableNotification(false)

.caption(description);

SendResponse imgResponse = bot.execute(request);

log.info("Image responce status: {}", imgResponse.isOk());

}

public long getChatId(String token) {

Update update = restTemplate.getForObject(String.format("https://api.telegram.org/bot%s/getUpdates", token),

Update.class);

if (chatId == null) {

chatId = update.getResult()

.get(0)

.getMessage()

.getChat()

.getId();

}

log.info("chat id: {}", chatId);

return (long) chatId;

}

private File generateImageFile(byte[] fileInBytes) {

try {

ByteArrayInputStream bis = new ByteArrayInputStream(fileInBytes);

BufferedImage image = ImageIO.read(bis);

File file = File.createTempFile("image", "png");

ImageIO.write(image, "png", file);

return file;

} catch (IOException e) {

e.printStackTrace();

return null;

}

}

}

package io.ibot.image.store.service;

import java.io.IOException;

import org.springframework.stereotype.Service;

import lombok.RequiredArgsConstructor;

import lombok.extern.slf4j.Slf4j;

@Service

@Slf4j

@RequiredArgsConstructor

public class TelegramImageSenderService {

private final ImageService imageService;

private final TelegramBotService telegramBotService;

public void getImageAndSendToTelegram(String imageInBase64, String token, String description) {

try {

byte[] image = imageService.convertBase64StringToByte(imageInBase64);

telegramBotService.sendImageToTelegram(image, token, description);

log.info("Image size: {}", image.length);

} catch (IOException e) {

log.info("Error while trying to send bot message: {}", e.getMessage());

}

}

}

package io.ibot.image.store.entity.model.message;

import java.util.List;

import lombok.AllArgsConstructor;

import lombok.Builder;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@Builder

@AllArgsConstructor

@NoArgsConstructor

public class Message {

private Integer message\_id;

private From from;

private Chat chat;

private Integer date;

private String text;

private List<Object> entities;

}

package io.ibot.image.store.entity.model;

import io.ibot.image.store.entity.model.message.Message;

import lombok.AllArgsConstructor;

import lombok.Builder;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@Builder

@AllArgsConstructor

@NoArgsConstructor

public class Result {

private Integer update\_id;

private Message message;