МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КІБЕРБЕЗПЕКИ, КОМП'ЮТЕРНОЇ І ПРОГРАМНОЇ ІНЖЕНЕРІЇ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
Нестеренко К.С.
"____"_____2023 р.

ДИПЛОМНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСНИКА ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

Тема: "Веб-сервіс для створення колекцій сображень NFT"

Виконавець: Кірпач Крістіна Сергіївна

Керівник: Горський Олексій Миколайович

Нормоконтролер: ст. вкл. Трофимчук Вікторія Миколаївна

КИЇВ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії
Кафедра інженерії програмного забезпечення
Освітній ступінь магістр
Спеціальність 121 Інженерія програмного забезпечення
Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Нестеренко К.С.
"___" _____ 2023 р

ЗАВДАННЯ
на виконання дипломної роботи студентки
Кірпач Крістіни Сергіївни

1. Тема дипломної роботи: «Веб-сервіс для створення колекцій зображень NFT», затверджена наказом ректора від 04.10.2023 р. № 2034/ст.

2. Термін виконання проекту: з 30.10.2023 р. до 08.12.2023 р.

3. Вихідні данні до проекту: програмний продукт розробити мовою JavaScript в середовищі розробки WebStorm.

4. Зміст пояснювальної записки:
   1. Загальна постановка задачі і аналіз її предметної області.
   2. Вимоги до програмного засобу.
   3. Проектування програмного засобу та огляд його архітектури.
   4. Рекомендації для користувача.

5. Перелік обов'язкових слайдів презентації:
   1. Мета дипломного проекту.
   2. Об'єкт розробки.
   3. Характеристика підходів до генерації колекцій зображень NFT.
   4. Архітектура програмного забезпечення.
   5. Сценарій використання.
   6. Прототип програмного засобу.

6. Календарний план-графік

№ пор	Завдання	Термін виконання	Відмітка про виконання
1.	Розробка та затвердження графіка роботи. Ознайомлення з постановкою задачі та вивчення літератури Написання розділів ПЗ, представлення керівнику.	01.09.23 – 20.09.23	
2.	Надання на перший нормо-контроль ПЗ - (обов’язково - титульної, завдання, графіка, реферат, список скорочень, зміст, вступ, список джерел, розділи ПЗ).	20.09.23 – 21.10.23	
3.	Редагування пояснювальної записки, графічного матеріалу.	21.10.23 – 12.11.23	
4.	Проходження першого нормо-контролю.	12.11.23 – 14.11.23	
5.	Проходження контролю ПЗ на плагіат.	15.12.23 – 18.12.23	
6.	Отримання відгуку керівника.Підготовка презентації та тексту доповіді. Попередній захист.	18.12.23 – 20.12.23	
7.	Проходження нормо-контролю, перепліт пояснювальної записки. Отримання рецензії.	20.12.23 – 21.12.23	
8.	Здати секретарю ДЕК: ПЗ, ГМ, CD-R з електронними версіями ПЗ, ГМ, презентацію, відгук керівника, рецензію, довідку про успішність, 2 папки, 2 конверта.	напередодні захисту	
9.	Захист дипломної роботи перед ЕК	25.12.23 – 26.12.23	

Керівник дипломної роботи: Горський Олексій Миколайович

Виконала: _____ _____ Кірпач Крістіна Сергіївна

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Веб-сервіс для створення колекцій зображень NFT»: 84 с., 20 рис., 2 табл., 21 використаних джерел.

Об’єкт дослідження: веб-сервіс для створення колекцій зображень NFT.

Мета дипломної роботи: підвищення зручності та швидкості створення колекцій зображень NFT шляхом генераціїї малюнків програмним кодом.

Методи дослідження: розробка зручного веб-сервісу для швидкого створення колекцій зображень для користувачів без знань мов програмування та подальшого використання у сфері NFT.

В процесі роботи, був зроблений глубокий аналіз скільки часу витрачають звичайні цифрові ходжники на створення колекцій зображень. В результаті чого виявилося, що за допомогою програмного коду та певних алгоритмів можна набагато швидше генерувати та створювати колекції зображень великого розміру.

Результати магістерської роботи можуть бути використані під час проведення наукових досліджень, при розробці програмних засобів призначених для створення колекцій малюнків для сфери NFT.

Розробка програми проводилася у середовищі WebStorm 2022.2, на мові програмування JavaScript.

NFT КОЛЕКЦІЇ, ЗРУЧНИЙ ДОДАТОК, СПРИЯТЛИВІСТЬ ІНФОРМАЦІЇ, ШВИДКЕ СТВОРЕННЯ, ГЕНЕРАЦІЯ МАЛЮНКІВ, ЗАОЩЯДЖЕННЯ ЧАСУ.

ЗМІСТ

СПИСОК СКОРОЧЕНЬ

NFT - Non-Fungible Token

HTML - HyperText Markup Language

UI – User Interface

UX - User Experience

CSS - Cascading Style Sheets

AI - Artificial Intelligence

DB – Database

JS – Java Script

DOM – Document Object Model

I/O – Input/Output

JSON – JavaScript Object Notation

API – Application Programming Interface

SRS – Software Requirements Specification

HTTP - HyperText Transfer Protocol

HTTPS - HyperText Transfer Protocol Secure

UML – Unified Modeling Language

URL – Uniform Resource Locator

SDK – Software Development Kit

LAN – Local Area Network

RAM - Random Access Memory

CPU – central processing unit

GPU - graphics processing unit

SQL - Structured Query Language

CSR - Certificate Signing Request

QA - Quality Assurance

IEEE - Institute of Electrical and Electronics Engineers

GMP - Good Manufacturing Practice

PUR – Process User Requirements

JWT - JSON Web Tokens

SVG - Scalable Vector Graphics

AJAX - Asynchronous Javascript and XML

BaaS - Backend-as-a-Service

IAM - Identity and Access Management

ВСТУП

Сьогоднішній сектор NFT вже достатньо розвинений, щоб продемонструвати різноманітність користувальницьких програм, і він стає все більш зручним для користувачів. Володіючи справжньою цифровою експертизою, художники тепер можуть легко створювати та продавати свої твори мистецтва. Внутрішні права на цифрову власність є дуже важливою та новаторською розробкою, яка дає творцям можливість створювати нові канали розповсюдження, можливості для співпраці та, зрештою, більший контроль над своїми творами. Творчі здібності вихідців із цифрових технологій захопили ринок NFT, і тепер вони продають на аукціоні NFT за сотні тисяч або навіть мільйони доларів у криптовалютах. Любителі колекціонування тепер мають такі платформи, як NBA TopShot, де, як старі добрі бейсбольні картки, вони можуть придбати свої улюблені моменти з найпопулярнішої баскетбольної ліги у світі. Це усуває необхідність турбуватися про зберігання тонких паперових шматків стандартних карток, як і будь-що інше, які також записуються в блокчейні. Іншим сектором, де NFT можуть викликати інтерес, є ринок ігор з підтвердженням власності. CryptoKitties є одними з перших прикладів цифрових предметів колекціонування, які отримали унікальне програмування даних. За допомогою NFT можна продавати майже будь-які віртуальні активи, включаючи зображення, музику, твори та 3D-моделі. Але здебільшого ми маємо на увазі твори цифрового мистецтва.

Часто колекція NFT-зображень — це безліч різних версій одного зображення, яке відрізняється кольором, розміром, фоном тощо. І для того, щоб художник не намалював вручну 1000 однакових зображень, були створені арт-генератори NFT.

Тому було вирішено створити новий додаток для створення колекцій зображень NFT. Додаток надасть можливість завантажувати, зберігати, видаляти, завантажувати та генерувати колекції NFT. Користувач зможе зареєструватися та мати власний обліковий запис. Програмне забезпечення

відповідатиме наведеним вище функціональним і нефункціональним вимогам.

РОЗДІЛ 1.

ЗАГАЛЬНА ПОСТАНОВКА ПРОБЛЕМИ ТА АНАЛІЗ ЇЇ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Загальний оглад NFT.

Один із трендів у сфері криптовалют, NFT, поєднує високі технології з прекрасним мистецтвом. Здатність відрізнити оригінальну роботу художника від звичайних репродукцій — це те, що робить явище таким, яким воно є. Крім того, ви можете володіти оригіналом, який виявився дуже цінним у випадках, коли копії по суті ідентичні оригіналу.

Цифровий актив, відомий як NFT, включає такі речі, як музика, мистецтво, фільми, ігрові предмети тощо. В даний час невзаємозамінні токени (NFT) широко поширені [1].

Зазвичай вони кодуються за допомогою того самого базового програмного забезпечення, що й багато інших криптовалют, і їх зазвичай купують і продають онлайн за допомогою криптовалют.

NFT існують з 2014 року, але нещодавно вони набули популярності, оскільки стають все більш популярним засобом купівлі та продажу цифрового мистецтва. Лише у 2021 році ринок NFT спостерігав приголомшливі витрати в 41 мільярд доларів, що майже стільки ж, скільки весь світовий ринок образотворчого мистецтва.

NFT часто є єдиними в своєму роді або, принаймні, одними з дуже обмеженої партії та мають відмінні ідентифікаційні коди. За словами Аррі Ю, керуючого директора Yellow Umbrella Ventures і голови блокчейн-ради Cascadia Washington Industry Association, NFT по суті створюють цифровий дефіцит.

Будь-хто може безкоштовно переглянути зображення, а також весь колаж із фотографій онлайн. Отже, навіщо комусь витрачати мільйони на щось, що можна легко завантажити або зробити знімок екрана?

Оскільки NFT дозволяє покупцеві зберегти право власності на оригінальний предмет. Крім того, він має вбудовану автентифікацію, яка є доказом права власності.

Для колекціонерів «цифрові хвастоща» є чи не важливішою за сам предмет.

Це різко контрастує з більшістю цифрового контенту, який майже завжди має нескінченну кількість. Теоретично, зменшення пропозиції активу повинно збільшити його вартість, якщо на нього є попит.

Блокчейн, розподілена база даних, яка зберігає транзакції, є місцем, де знаходяться NFT. Швидше за все, ви найбільше знайомі з блокчейном як технологією, що лежить в основі криптовалют.

Хоча їх можна використовувати в інших блокчейнах, NFT спеціально зберігаються в блокчейні Ethereum.

NFT створюється або «карбується» з цифрових об’єктів, які представляють як матеріальні, так і нематеріальні елементи, включно з:

- Графічне мистецтво
- GIF-файли
- Відео та спортивні моменти
- Предмети колекціонування
- Віртуальні аватари та скіни відеоігор
- Дизайнерські кросівки
- Музика

Навіть твіти враховуються. Співзасновник Twitter Джек Дорсі продав свій перший твіт як NFT за понад 2,9 мільйона доларів.

NFT – це, по суті, цифрові версії справжніх предметів колекціонування. Таким чином, покупець отримує цифровий файл, а не справжню картину маслом для демонстрації на стіні.

Крім того, вони отримують одноосібні права власності. Оскільки NFT використовують технологію блокчейн, перевірити право власності та передавати токени між власниками легко.

NFT можуть мати лише одного власника одночасно. Метадані NFT також можуть містити певні дані, які зберігав винахідник [2]. Художники, наприклад, можуть підписувати свої твори мистецтва, ввівши свій підпис у файл.

Програмування, яке часто використовує NFT, можна порівняти з програмуванням криптовалют, таких як Bitcoin або Ethereum, але на цьому схожість закінчується.

Маючи можливість продавати або обмінювати один на одного, фізичні гроші та криптовалюти є «замінними». Долар завжди коштує іншого долара, а вартість одного біткойна завжди еквівалентна вартості іншого біткойна. Завдяки своїй взаємозамінності криптовалюта є надійним методом транзакцій у блокчейні.

NFT унікальні. Оскільки всі вони мають цифровий підпис, NFT не можна обмінювати або прирівнювати один до одного (отже, не взаємозамінні). Оскільки обидва вони є NFT, одне відео NBA Top Shot, наприклад, не еквівалентно щоденним відео. Якщо на те пішло, один кліп NBA Top Shot навіть не завжди еквівалентний іншому.

NFT і технологія блокчейн дають художникам і виробникам контенту особливий шанс монетизувати свої роботи. Наприклад, художники більше не зобов'язані продавати свої роботи через галереї чи аукціонні будинки. Натомість виконавець може продавати його як NFT безпосередньо споживачеві, дозволяючи йому зберігати більшу частину доходу від продажів. Крім того, коли митець продає свою роботу новому власнику, він може запрограмувати роялті, щоб отримати відсоток від угоди. Оскільки художники зазвичай не отримують більше прибутку після початкового продажу, це бажана функція.

Заробляння грошей за допомогою NFT не обмежується мистецтвом. Щоб зібрати гроші на благодійність, такі компанії, як Taco Bell і Charmin, виставили на аукціон тематичні твори мистецтва NFT. З найвищими ставками 1,5 обгорнутого ефіру (WETH), або 3723,83 доларів США на момент написання статті, NFT-арт Taco Bell був розпроданий за кілька хвилин після того, як Charmin назвав його «NFTP».

Nyan Cat, GIF-зображення 2011 року, що зображує кота з тілом у формі поп-тарту, було продано в лютому приблизно за 600 000 доларів. А станом на

кінець березня продажі NBA Top Shot перевищили 500 мільйонів доларів. Один кліп Леброна Джеймса NFT був проданий за понад 200 000 доларів [3].

Навіть такі відомі особи, як Снуп Догг і Ліндсі Лохан, підхоплюють сек'юритизовані NFT і випускають оригінальні спогади, твори мистецтва та моменти.

Якщо ви хочете створити власну колекцію NFT, вам потрібно придбати деякі ключові предмети.

По-перше, вам знадобиться отримати цифровий гаманець, який дозволить зберігати NFT і криптовалюти.

Ймовірно, вам знадобиться придбати певну криптовалюту, наприклад ефір, залежно від того, які валюти приймає ваш постачальник NFT.

Ви можете купити криптовалюту за допомогою кредитної картки на таких платформах, як Coinbase, Kraken, eToro і навіть PayPal і Robinhood. Потім ви зможете перемістити його з біржі на обраний вами гаманець.

Розвитком відносно простої ідеї криптовалют є незамінні токени. Для багатьох категорій активів, таких як нерухомість, кредитні контракти та твори мистецтва, сучасні фінансові системи включають складні системи торгівлі та фінансування. NFT сприяють переосмисленню цієї інфраструктури, забезпечуючи цифрове представлення фізичних активів.

Щоб було зрозуміло, ні концепція використання унікальної ідентифікації, ні ідея цифрового представлення реальних товарів не є новими. Однак ці ідеї стають потужною силою для змін у поєднанні з перевагами захищеного від втручання блокчейну смарт-контрактів.

Ринкова ефективність є, мабуть, найбільш очевидною перевагою NFT. Трансформація фізичного активу в цифровий спрощує процедури та позбавляє посередників.

NFT, які представляють фізичні або цифрові твори мистецтва в блокчейні, усувають необхідність в агентах, дозволяючи художникам безпосередньо взаємодіяти зі своєю аудиторією. Крім того, вони можуть покращити корпоративні процедури.

Наприклад, NFT для пляшки вина спростить спілкування з нею для різних учасників ланцюга поставок і допоможе відстежувати її створення, походження та продаж протягом усього процесу.

Для одного зі своїх клієнтів консалтингова компанія Ernst & Young вже створила таке рішення. Незамінні токени також чудово підходять для керування ідентифікацією. Розглянемо ситуацію, коли фізичний паспорт потрібен у кожному пункті в'їзду та виїзду. Можна спростити процедури в'їзду та виїзду для юрисдикцій шляхом перетворення індивідуальних паспортів на NFT, кожен з яких має свої особливі відмінні якості. NFT також можна використовувати для керування ідентифікацією в цифровій сфері, розширюючи цей варіант використання.

Завдяки дробленню таких матеріальних активів, як нерухомість, NFT можуть також демократизувати інвестування. Цифрову нерухомість можна розділити між багатьма власниками набагато легше, ніж фізичну.

Ця етика токенізації не повинна обмежуватися нерухомістю; воно також може застосовуватися до інших активів, таких як твори мистецтва. Тож картина не обов'язково має мати лише одного власника.

Можливі численні власники цифрової версії, кожен з яких відповідає за частину картини. Такі угоди можуть підвищити його вартість і дохід.

Розвиток нових ринків та шляхів інвестування є найбільш інтригуючою перспективою для NFT. Уявіть собі земельну ділянку, поділену на кілька ділянок, кожна зі своїми особливостями та формами власності.

Одна з ділянок може бути поряд з пляжем, інша може бути в комплексі з розважальними закладами, а третя може бути мікрорайоном. Кожна ділянка землі є окремою, має різну ціну та представлена NFT залежно від її особливостей.

Додавши відповідні метадані в кожен окремий NFT, можна спростити складний і бюрократичний процес торгівлі нерухомістю.

Таке поняття вже використовується Decentraland, платформою віртуальної реальності, що працює на блокчейні Ethereum.

Ідея токенізованих земельних ділянок (різних за вартістю та розташуванням) може бути реалізована в реальному світі, коли NFT просуваються та стають більш інтегрованими у фінансову систему.

1.2. Аналіз основних аспектів використання NFT у цифровому мистецтві.

NFT не є взаємозамінними, оскільки кожен з них є унікальним цифровим активом і не може бути взаємозамінним або відтвореним. Наприклад, біткойн взаємозамінний, ви можете обміняти один біткойн на інший і отримати те саме.

У той час як така унікальна картина, як «Мона Ліза», є незамінною. Звичайно, ви можете купити відбиток Мони Лізи, але існує лише одна оригінальна картина Мони Лізи [4]. Права власника NFT прописані в смарт-контракті і, як правило, обмежуються можливістю купувати, володіти та продавати.

Отже, купуючи NFT, інвестор не стає власником виключних прав на оцифрований таким чином об’єкт мистецтва. Причому всі відносини такого характеру поширюються і регулюються тільки в рамках конкретної блокчейн-мережі.

Хоча елемент незамінності NFT є одним із факторів, який визначає дефіцитність і цінність твору мистецтва NFT, цінність NFT також може залежати від кількох факторів: команди, яка стоїть за ним, варіантів його використання, спільноти чи пропозиції на ринку та попит.

Звичайні переваги NFT, такі як підтвердження права власності, свобода видачі та незмінність записів, зробили революцію в тому, як ми створюємо, збираємо та володіємо мистецтвом. Нижче ми висвітлили ще сім способів, як NFT сколихнули світ мистецтва.

У певному сенсі галереї можна розглядати як охоронців мистецьких робіт, оскільки вони вибирають, які мистецькі роботи вони демонструватимуть.

Порівняно з традиційними мистецькими закладами, ринки NFT набагато більше включають нових митців і концепції нішевого мистецтва. Це дозволило

художникам процвітати за межами структури галерей та аукціонних будинків.

Зазвичай, коли митець продає свій твір мистецтва, він нічого не заробляє на наступних перепродажах. NFT зробили революцію на ринку вторинного мистецтва, і митці можуть визначати суму роялті, яку вони отримують за допомогою смарт-контракту – блокчейн-програми, яка самостійно виконує завдання, коли виконуються певні умови. Художники можуть встановити правило в рамках смарт-контракту, яке ініціює виплату роялті щоразу, коли їх мистецтво NFT переходить з рук на вторинний ринок.

Традиційно галереї та деякі онлайн-ринки беруть відсоток від продажів від доходу художника. За допомогою NFT художники можуть безпосередньо взаємодіяти з покупцями та продавати свої твори мистецтва на ринках NFT без потреби в агентах чи посередниках [5].

Автентичність і походження твору мистецтва є одним із факторів, які впливають на рішення покупця. Перевіркою цих показників зазвичай є аукціонні будинки чи галереї, але хоча помилки походження та добре виготовлені підробки є рідкістю, вони трапляються.

Оскільки NFT захищено в мережі блокчейн, будь-хто може відстежувати його транзакції та історію попереднього володіння, щоб перевірити його автентичність. Більше того, незмінний характер облікових книг блокчейну значно мінімізує ризик підробки.

Цифрове мистецтво — не єдине середовище, яке може виграти від NFT. Що стосується фізичних творів мистецтва, NFT можуть функціонувати як цифровий сертифікат, який підтверджує автентичність творів мистецтва, оскільки деякі художники карбують цифрові та фізичні твори разом як один NFT. Порівняно з аркушем паперу, NFT є більш безпечним способом фіксації права власності, перевірки його походження та зниження ризиків підробки.

Зростання популярності NFT і доступність ринків NFT підняли завісу таємниці, що вкриває художні галереї та аукціонні будинки. З такими престижними музеями, як Державний Ермітаж, які визнають NFT як форму мистецтва, а художні галереї NFT з’являються по всьому світу. Мистецтво

стало більш доступним, ніж будь-коли раніше, представляючи митців ширше охоплення, зв’язуючи їх із потенційними клієнтами з усіх верств суспільства.

Як нещодавно визнаний вид мистецтва, NFT розсунули творчі та інноваційні межі з моменту свого зародження, а деякі видатні митці грали з їхнім потенціалом.

У 2021 році всесвітньо відомий художник і сумнозвісний арт-провокатор Демієн Герст запустив концептуальний погляд на потенціал NFT. «The Currency» — це колекція крапкового мистецтва, намальованого вручну, що складається з 10 000 NFT з 10 000 відповідних фізичних версій формату A4. Кожна фізична частина має водяний знак і голограму, що ускладнює її підробку.

Але ось нюанс: покупці твору мистецтва мають вибрати, чи хочуть вони зберегти цифрову чи фізичну версію твору мистецтва. Їм дається рік з моменту надсилання NFT для прийняття рішення. Якщо вони вирішать зберегти NFT, фізичні ілюстрації будуть знищені, і навпаки. Остаточні рішення будуть прийняті в липні 2022 року.

1.3. Призначення та технології формування NFT

Зараз багато людей створюють мистецтво NFT і продають його в Інтернеті. Колекціонування NFT стало дуже популярним.

Оскільки кожен витвір мистецтва унікальний, колекціонери виплачують мільйони доларів, щоб купити всю колекцію NFT.

Якщо ви плануєте створювати колекції NFT, ви можете скористатися генератором зображень NFT.

Це програмне забезпечення допомагає вам швидко створювати унікальні зображення NFTS за допомогою базового зображення.

Генератор NFT — це нова захоплююча революція у світі блокчейнів. Ця нова технологія створила абсолютно нову категорію колекціонерів і новий ринок унікальних колекцій NFT.

Це програмне забезпечення на основі штучного інтелекту, яке може

створювати колекції NFT. Вам не потрібен технічний досвід для створення NFT. Це готове до використання програмне рішення, за допомогою якого ви можете створювати мільйони унікальних ілюстрацій NFT [6].

Ви можете використовувати це програмне забезпечення для створення дуже прибуткового бізнесу, якщо у вас є точні маркетингові стратегії. Більшість людей і художників використовують це програмне забезпечення для створення та продажу колекцій NFT на блокчейні.

Програмне забезпечення для створення графічних зображень NFT дуже гнучке та просте у використанні. Він працює на передовій технології штучного інтелекту, щоб створити кілька унікальних NFT з однієї базової ілюстрації.

Якщо ви художник, який знає цифрове мистецтво, ви можете використовувати це програмне забезпечення. Ви можете створити просту базову ілюстрацію самостійно. Коли ваш цифровий ресурс буде готовий, ви можете завантажити файл у програмне забезпечення.

Переконайтеся, що ви завантажуєте базовий файл у правильному форматі. Коли це буде зроблено, ви повинні встановити рідкість для різних шарів ілюстрації. Окрім цього, вам також потрібно додати назву для колекції NFT разом із описом. Це допоможе вам створити метадані для колекції NFT. Метадані пропонують інформацію про твір мистецтва та допомагають дізнатися, кому вони належать. Тепер програмне забезпечення запрацює зачарованим і почне створювати колекцію NFT з єдиного основного твору мистецтва.

Він використовуватиме штучний інтелект для створення унікальних шарів різних ілюстрацій, вносячи зміни в базовий файл.

Ви можете вибрати кількість NFT, яку потрібно створити. Це допоможе вам створити тисячі NFT. Після завершення колекції ви можете завантажити результати.

Ви отримаєте повну колекцію NFT, готову до карбування. Тепер ви можете відкрито продавати його на платформі блокчейн.

Для створення NFT не потрібно наймати розробника. За допомогою

цього програмного забезпечення ви можете створити його самостійно.

Особливості генераторів мистецтва NFT:

Прості у використанні.

Кодування не потрібне, створюйте свої шари, імпортуйте ресурси, натисніть «Створити», і ви готові до використання.

Експортуйте зображення, gif або відео.

Ви можете імпортувати зображення, gif-файли та відео, і ми створимо вашу колекцію у вибраному вами форматі.

Рідкість шарів

У великій колекції вам може не знадобитися накладати шар на кожне NFT. Ви можете створити рідкість шару, щоб він застосовувався лише певний відсоток разів.

Рідкість атрибутів

Подібно до рідкісності шару, ви можете створювати різні об’єкти, які будуть рідкіснішими за інші. Ви зможете легко встановити, яку зміну для атрибута слід застосувати.

Метадані

Застосунок автоматично генерує метадані, сумісні з Ethereum, Solana та Cardano Blockchain.

Сторінка засобу перегляду метаданих також дозволяє оновлювати метадані після створення колекції.

Обліковий запис не потрібен

Щоб розпочати створення колекції, не потрібно створювати обліковий запис.

Автоматичне збереження

Застосунок автоматично зберігає ваші проекти у веб-переглядачі, тому, коли ви повернетеся, ваші шари, конфігурація та файли все ще існуватимуть.

Експортуйте та продавайте свої NFT

Коли ваш NFT буде готовий, експортуйте його у вибраний формат і продайте в Інтернеті обраному вами покупцеві.

Переваги створення застосунку для генераціїї колекцій зображень NFT :

Глобальна мережа

За допомогою генератора NFT ви можете просто створювати та показувати свої NFT на провідному ринку та залучати більше користувачів для своєї колекції NFT.

Легко масштабується

Незалежно від того, чи є у вас тисячі NFT, генератор NFT може легко керувати вашою колекцією NFT, не турбуючись про її безпеку.

Кілька блокчейнів

Наша платформа підтримує різні типи блокчейнів, дозволяючи користувачам легко налаштувати свою колекцію NFT без будь-яких труднощів.

NFT Airdrops

Виберіть правильний Airdrop для своєї кампанії та покращте залучення користувачів, створюючи цінні моменти для своїх користувачів.

Соціальні токени

Створіть соціальні токени для своєї колекції NFT, щоб підвищити їхню фінансову цінність і розпочати захоплююче змагання серед ваших шанувальників.

Початкові пропозиції NFT (INO)

За допомогою Initial NFT Offerings ви можете викликати ажіотаж і інтерес до своєї нової колекції NFT, пропонуючи обмежений набір початкових NFT.

1.4. Аналіз схожиш застосунків.

Сьогодні існує кілька різних онлайн-сервісів для генерації мистецтва NFT. Всі вони мають досить схожі характеристики.

Appypie NFT Generator – це програмне забезпечення для створення NFT, яке дозволяє створювати цифрове мистецтво та перетворювати його на NFT без кодування. Ви можете створювати власні цифрові NFT за допомогою генератора NFT, не знаючи коду [7].

Конструктор NFT від Appypie надає вам ідеальні ресурси, необхідні для

створення унікального, захоплюючого та вишуканого мистецтва NFT.

За допомогою спеціального творця зображень NFT на основі шаблонів від Appy Pie Design ви можете створювати приголомшливі зображення та колекції NFT.

Переваги:

- Без коду (тільки в PRO версії)
- Шаблони
- Легко масштабується до 10000 виходів
- Інструмент NFT Airdrop

Недоліки:

- Обмежений обсяг пам’яті до 50 МБ
- Відсутність засобу для видалення фону
- Відсутність експорту SVG
- Відсутність спільного доступу для команди

На рис. 1.1 ми можемо побачити, загальний вигляд цього застосунку.

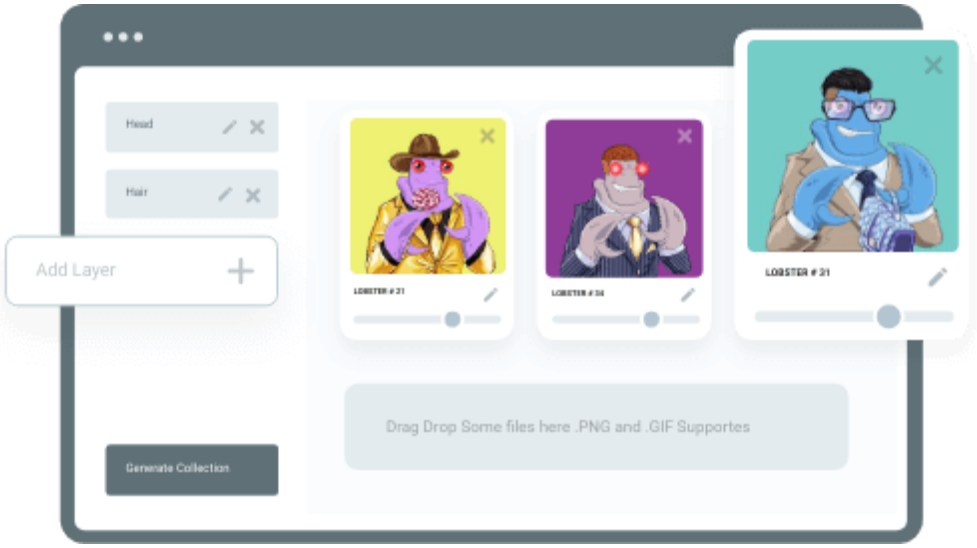

Рис. 1.1. Інтерфейс застосунку FreeFormatter.

Fotor NFT Creator – за допомогою NFT-мейкера Fotor ви можете швидко й без особливих зусиль створювати мистецтво NFT і розкривати свій творчий

потенціал. Візьміть унікальний модельний метод і техніку машинного навчання та поєднайте їх із улюбленими техніками олійного живопису. Завантажте своє зображення, виберіть свій улюблений стиль художнього ефекту, і AI NFT Maker автоматично створить для вас ілюстрацію NFT. Ви будете вражені результатами.

Переваги:

- Безкоштовно, PRO і PRO+
- Без коду
- Конструктор рідкості
- Кілька блокчейнів

Недоліки:

- Обмежені можливості безкоштовної версії
- Метадані підтримуються не всіма блокчейнами

На рис. 1.2 ми можемо побачити, які функції виконує цей застосунок.

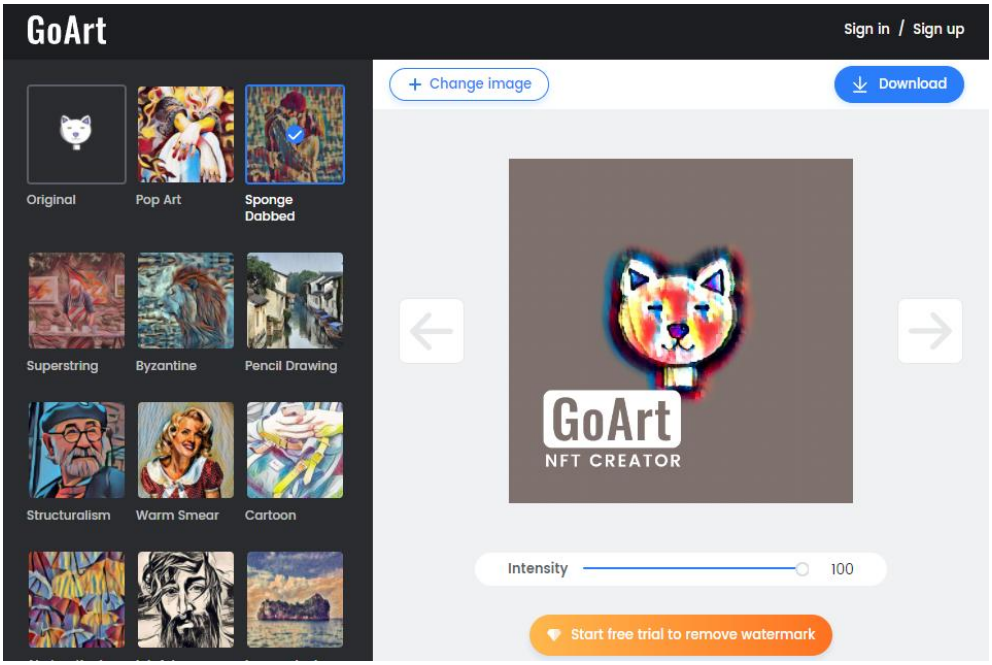

Рис. 1.2. Функції тестового інструменту Fotor NFT Creator s.

OneMint NFT Art Generator – якщо вам потрібна рідкість і створення кількох символів, OneMint також має те, що вам потрібно. Що вони дійсно оцінили в зручності платформи, що є набагато важливішим, ніж думають користувачі [8].

Ви не лише ступаєте у відносно новий простір, але й створюєте власне мистецтво, тож хочете, щоб усе було якомога простіше, а OneMint – це користувальницький інтерфейс/UX.

Переваги:

- Багато інструментів для створення NFT
- Має калькулятор цін
- Рідкість атрибутів
- Для Ethereum

Недоліки:

- Обмежені можливості безкоштовної версії
- Має витрати на карбування

На рис. 1.3 ми можемо побачити основін функції цього інструменту.

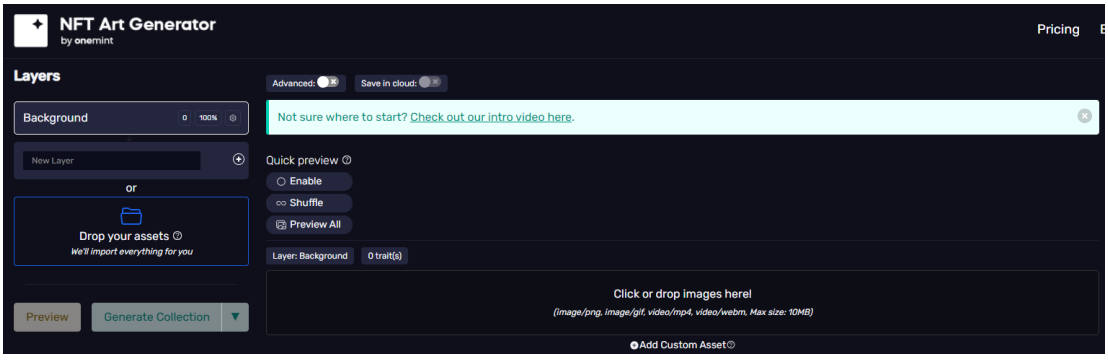

Рис. 1.3. Функції застосунку OneMint NFT Art Generator.

NightCafe Studio – це програма AI Art Generator із кількома методами створення мистецтва AI.

Використовуючи передачу нейронних стилів, ви можете перетворити свою фотографію на шедевр. Використовуючи штучний інтелект, що перетворює текст у зображення, ви можете створити ілюстрацію лише з текстової підказки. Ви можете створювати зображення NFT двома способами.

Ваші твори належать вам, і ви можете робити з ними все, що завгодно (за умови, що ви — або маєте дозвіл — власника авторських прав на будь-які вхідні зображення та відповідно до законів про авторське право у вашій юрисдикції).

Ось перший спосіб:

- Введіть текстову підказку в інструмент із вашої глибокої уяви (наприклад “Пост-апокаліптична країна чудес”).
- Дозвольте інструменту створити для вас зображення.
- Виготовте зображення як NFT на ринку NFT.

Другий спосіб:

- Завантажте зображення в програмне забезпечення.
- Дайте йому текстову підказку.
- Дозвольте інструменту створити вихідне зображення на основі вашого вхідного зображення та тексту.
- Виготовте зображення як NFT на ринку NFT.

Переваги:

- Інструмент простий у використанні. Крім того, є чудова безкоштовна пробна версія, якою можна скористатися без реєстрації та введення даних кредитної картки.
- Ви можете використовувати новітні сучасні алгоритми штучного інтелекту, щоб створити мистецтво з тексту.
- Зображення генеруються так, щоб виглядати як те, що міг створити справжній художник.

Недоліки:

- Ви повинні придбати кредити, щоб створити більше зображень.
- Інструмент не підходить для створення колекцій NFT із десятків зображень.

NightCafe Creator спеціалізується на створенні мистецтва NFT. За допомогою цього інструменту легко зробити купу зображень і перетворити їх на NFT-арт миттєво [9].

На Рис. 1.4 ми можемо розглнути головні функціїї цього застосунку.

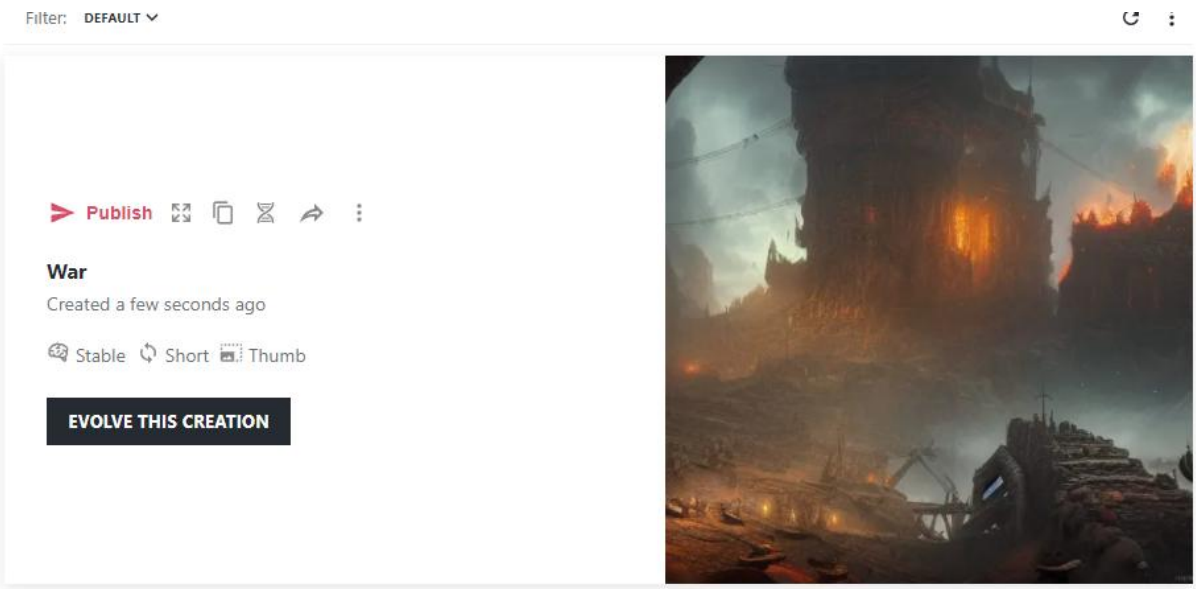

Рис. 1.4. Головні функціїї NightCafe Studio.

Висковки

Колекція NFT дозволяє шанувальникам криптовалют легко мати відлік своїх криптоактивів. Це все, про що ви можете подбати з колекцією NFT. Основною функцією колекції NFT є отримання та зберігання NFT у безпечному, безпечному та зручному режимі.

Другою функцією NFT Collection є приєднання до блокчейну та перевірка ваших NFT. Якщо NFT є неприйнятним або посилання не працює, вам буде

надіслано сповіщення, яке дасть можливість вжити заходів.

Генератори мистецтва NFT дозволяють кожному створювати колекції мистецтва NFT, не потребуючи розуміння кодування чи складнощів створення цього виду мистецтва. Насправді найкращі генератори зображень NFT використовуються для створення колекцій зображень NFT, створених процедурою.

Ви можете легко створити NFT, ввівши ключові слова або фрази в один із найкращих генераторів NFT, доступних сьогодні. Генератор мистецтва штучного інтелекту використовує алгоритми для оцінки незліченних творів мистецтва та створення оригінальних візуальних зображень, які служать відмінною візуальною інтерпретацією вихідних текстів.

РОЗДІЛ 2.
ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1. Специфікація вимог до програмного забезпечення.

Специфікація вимог до програмного забезпечення – це офіційний звіт, який виступає як представлення програмного забезпечення, яке дає змогу клієнтам перевірити, чи воно відповідає їхнім вимогам.

Крім того, він містить вимоги користувача до системи, а також детальні специфікації системних вимог тощо. Це специфікація для конкретного програмного продукту, програми або набору програм, які виконують певні функції в певному середовищі.

Він переслідує кілька цілей залежно від того, хто його пише. По-перше, специфікація вимог може бути написана клієнтом системи. По-друге, специфікація вимог до програмного забезпечення може бути написана розробником системи.

Ці два методи створюють абсолютно різні ситуації та встановлюють абсолютно різні цілі для документа. Перший випадок, використовується для визначення потреб і очікувань користувачів.

Другий випадок, пишеться для різних цілей і служить договірним

документом між замовником і виковавцем.

Специфікація вимог до програмного забезпечення мінімізує час і зусилля, необхідні розробникам для досягнення бажаних цілей, а також мінімізує вартість розробки. Хороша специфікація вимог визначає, як програма взаємодіятиме з апаратним забезпеченням системи, іншими програмами та користувачами в різноманітних ситуаціях.

Оцінюються такі параметри, як швидкість роботи, час відгуку, доступність, портативність, ремонтопридатність, площа, безпека та швидкість відновлення після несприятливих подій. Методи визначення SRS описані в специфікації IEEE (Інститут інженерів з електротехніки та електроніки) 830-1998.

Ключові компоненти специфікації вимог до програмного забезпечення. Основні розділи специфікації вимог до програмного забезпечення:

Бізнес-чинники – у цьому розділі описуються причини, за якими клієнт хоче побудувати систему, включаючи проблеми з поточною системою та можливості, які надасть нова система.

Бізнес-модель – у цьому розділі описується бізнес-модель замовника, яку має підтримувати система, включаючи організаційний, бізнес-контекст, основні бізнес-функції та діаграми процесу.

Функціональні та системні вимоги – цей розділ зазвичай складається з вимог, які організовані в ієрархічній структурі. Бізнес/функціональні вимоги знаходяться на верхньому рівні, а детальні системні вимоги вказані як дочірні елементи.

Випадки використання для бізнесу та системи – цей розділ складається з діаграми варіантів використання уніфікованої мови моделювання (UML), що відображає ключові зовнішні об’єкти, які взаємодіятимуть із системою, і різні варіанти використання, які вони повинні виконувати.

Технічні вимоги – у цьому розділі перераховано нефункціональні вимоги, які складають технічне середовище, де програмне забезпечення має працювати, і технічні обмеження, за яких воно має працювати.

Якості системи - цей розділ використовується для опису нефункціональних вимог, які визначають атрибути якості системи, такі як надійність, придатність до обслуговування, безпека, масштабованість, доступність і зручність обслуговування.

Обмеження та припущення - цей розділ містить будь-які обмеження, накладені замовником на проект системи.

Він також містить припущення групи розробників вимог щодо того, що очікується під час проекту.

Критерії прийняття – у цьому розділі детально описано умови, які мають бути виконані, щоб клієнт міг прийняти остаточну систему.

Призначення специфікації вимог до програмного забезпечення. Специфікація вимог до програмного забезпечення є основою всього проекту організації.

Він встановлює рамки, яких дотримуватимуться всі команди розробників.

Він надає важливу інформацію всім командам, включаючи розробку, операції, забезпечення якості (QA) і технічне обслуговування, забезпечуючи згоду команд.

Використання специфікації вимог допомагає підприємству підтвердити, що вимоги виконано, і допомагає керівникам підприємств приймати рішення щодо життєвого циклу свого продукту, наприклад, коли вилучити функцію.

Крім того, написання специфікації вимог може допомогти розробникам скоротити час і зусилля, необхідні для досягнення своїх цілей, а також заощадити гроші на вартості розробки.

Специфікація вимог програмного забезпечення повинна мати такі характеристики:

Правильність – має точно відображати функціональність і специфікацію продукту в будь-який момент часу.

Однозначніссть – не повинні виникати плутанини щодо тлумачення вимог.

Повність – має містити всі функції, які запитує клієнт.

Послідовність – у всьому документі необхідно дотримуватися однакових абревіатур і умовних позначень.

Рейтинг важливості та/або стабільності – кожна вимога важлива. Але деякі з них є терміновими та мають бути виконані перед іншими вимогами, а деякі можуть бути відкладені. Кожну вимогу краще класифікувати за її важливістю та стабільністю.

Піддається перевірці – специфікація вимог піддається перевірці, лише якщо кожну заявлену вимогу можна перевірити. Вимогу можна перевірити, якщо існує якийсь метод кількісної оцінки, чи відповідає кінцеве програмне забезпечення цій вимозі.

Модифікованість – специфікація має чітко визначати кожну вимогу систематично. Якщо є будь-які зміни, конкретні вимоги та залежні від них можуть бути змінені відповідно без впливу на інші.

Простежуваність – специфікація вимог є простежуванною, якщо походження кожної з вимог є зрозумілим і якщо це дозволяє легко посилатися на кожну вимогу в майбутньому розвитку.

Деякі з цілей, яких повинна досягти специфікація вимог, полягають у:

- Надавати клієнту зворотний зв’язок, гарантуючи, що IT-компанія розуміє проблеми, які має вирішувати система програмного забезпечення, і способи вирішення цих проблем.
- Допомозі розділити проблему на більш дрібні компоненти, просто записавши вимоги.
- Прискорити процеси тестування та перевірки.
- Сприяти оглядам.

2.2. Системні вимоги проекту.

Для певних елементів наведено як мінімальні, так і рекомендовані системні вимоги. Наприклад, відеогра може працювати з мінімальними

специфікаціями ЦП і ГП, але працюватиме краще із запропонованим апаратним забезпеченням.

Потужніший процесор і графічна карта можуть призвести до кращої графіки та вищої частоти кадрів. У процесі проектування системи системні вимоги також використовуються як відправна точка.

Різні системні моделі, такі як об’єктна модель або модель потоку даних, можуть служити основою для специфікації вимог.

Операційні системи та обсяг дискового простору, необхідний для встановлення програми, є двома прикладами системних вимог, які не можна змінити.

Інші вимоги, такі як CPU, GPU та RAM, можуть значно відрізнятися від мінімальних і рекомендованих специфікацій [11].

Переконайтеся, що ваша система відповідає необхідним вимогам під час придбання або оновлення програмного забезпечення, щоб гарантувати позитивний досвід користувача.

Мінімальний:

- Операційна система: Windows 7, 8, 10, MacOS або Linux.
- Процесор: процесор Intel або AMD з частотою 1 Гц.
- Оперативна пам'ять: 2 Гб
- Відеоадаптер: nVidia GeForce GT 660 або nVidia GeForce GT 630
- Дискове сховище: 1 ГБ вільного дискового простору.
- Обладнання введення: клавіатура та миша.
- Роздільна здатність монітора: 1280x800
- Швидкість Інтернету: для активації програмного забезпечення необхідне підключення до Інтернету, 25 Мб/с.

Рекомендовано:

- Операційна система: Windows 7,8,10 або MacOS.
- Процесор: процесор Intel або AMD з частотою 3,2 Гц.
- Оперативна пам'ять: 4 Гб
- Відеоадаптер і монітор: nVidia GeForce GTX 1050 або nVidia GeForce

GTX 1050 TI або Quadro T1000

- Дискове сховище: 4 ГБ вільного дискового простору.
- Дайте мені введення: клавіатура та миша.
- Роздільна здатність монітора: 1920x1080
- Швидкість Інтернету: для активації програмного забезпечення необхідне підключення до Інтернету, 100 Мб/с.

2.3. Функціональні вимоги проекту.

Функціональні вимоги визначають завдання, яке має виконати система або системний компонент. Є кілька способів продемонструвати це. Найбільш характерними з них є приклади використання та текстові описи в документах.

Функціональна вимога часто має чітку назву, номер, коротке обґрунтування та опис.

Мета цієї інформації — допомогти читачеві зрозуміти, чому така потреба потрібна, і відслідковувати її протягом розробки системи.

Важливою частиною вимоги є стисле та доступне пояснення необхідної поведінки.

Описана поведінка може бути наслідком корпоративних чи комерційних обмежень або може бути виявлена під час сеансів виявлення з користувачами, зацікавленими сторонами та іншими експертами в галузі всередині організації.

Численні вимоги можуть бути виявлені в процесі розробки варіантів використання.

Коли це трапляється, аналітик вимог може створити потребу-заповнювач із назвою та підсумком, а потім досліджувати деталі пізніше, коли вони будуть більш повно зрозумілі.

Таблиця 2.1 описує основні функціональні вимоги проекту, а також їх описи.

Таблиця 2.1.
Функціональні вимоги

Функціональність	Опис
Логін	Користувач повинен мати можливість увійти за допомогою дійсного облікового запису Google
Вихід	Користувач повинен мати можливість вийти з системи.
Вибір назви колекції NFT	Користувач повинен мати можливість вибрати назву для своєї колекції NFT.
Зміна назви колекції NFT	Користувач повинен мати можливість змінити назву своєї колекції NFT.
Вибір кількості видань колекції NFT	Користувач повинен мати можливість вибрати кількість видань для своєї колекції NFT.
Додавання шарів зображення.	Користувач повинен мати можливість додавати шари свого зображення NFT.
Видалення шарів зображення.	Користувач повинен мати можливість видаляти шари свого зображення NFT.
Завантаження зображень для шарів.	Користувач повинен мати можливість завантажувати зображення для шарів.
Створення образу NFT	Користувач повинен мати можливість генерувати зображення NFT за допомогою шарів.
Збереження результату	Користувач повинен мати можливість зберегти результат генерації.

Продовження таблиці 2.1.

Перегляд збережених колекцій	Користувач повинен мати можливість переглядати збережені колекції NFT.
Видалення збереженої колекції	Користувач повинен мати можливість видалити збережену колекцію NFT.
Завантаження збережених колекцій.	Користувач повинен мати можливість завантажити збережені колекції.

Перехід на сайт обміну NFT.	Користувач повинен мати можливість переходу на сайт NFT exhanhe.

2.4. Вимоги користувача.

Вимоги до користувача зазвичай пишуться під час обговорення варіантів використання для проекту. Визначення вимог виконується замовником або менеджерами продукту, які знають, як вбудована система використовуватиметься користувачем.

Багато вимог до користувача стосуються того, як користувач буде взаємодіяти з системою і чого він очікує. Якщо в системі є аспект екрану або людино-машинного інтерфейсу, вимога користувача може базуватися на тому, що відбувається, коли користувач вибирає дію на екрані.

Можливо, натисканням кнопки не тільки запускається процес, але й відбувається перехід на інший екран із звуковим сповіщенням. Коли такі вимоги користувача записуються, вони часто можуть розбиватися на кілька системних вимог пізніше через перемикання екранів, максимальні затримки під час запуску процесу та, нарешті, як має виглядати наступний екран.

Однією з підводних каменів є спроба написати системні вимоги під час зустрічі з вимогами користувача. Це часто заважає отримати розуміння вимог користувача, і ключові функціональні елементи можуть бути упущені.

Насправді, як згадувалося раніше, часто краще тримати вимоги користувачів і системні вимоги окремо під час їх відстеження та звітування.

Вимоги користувача часто більш читабельні, зрозумілі та забезпечують краще уявлення про те, як працюватиме система.

Незважаючи на те, що вимогам користувача може бракувати конкретної інформації про те, що насправді має відбуватися в системі, вони все одно цінні тим, що можуть забезпечити загальні очікування функціональності системи.

Нарешті, під час написання вимог користувача доцільно мати можливість відстеження того, звідки походить вимога користувача.

Незалежно від того, чи це від одного клієнта, чи від менеджера продукту, розуміння того, звідки воно надійшло, є важливим. Під час запису вимог користувача бувають випадки, коли окремі сеанси користувача можуть сформувати суперечливі вимоги.

Можливість повернутися до ініціатора та краще зрозуміти варіант використання може допомогти полегшити розшифровку суперечливих вимог користувача.

Це може виникнути з різних точок зору, як-от оператор чи спеціаліст із обслуговування, тому можливість повернутися та вирішити ці розбіжності стає важливою.

Вимоги користувача – це те, що випливає з назви. Це специфікації, встановлені кінцевим користувачем. З точки зору продукту, який необхідно створити, необхідної пропускної здатності та умов, за яких продукт має бути виготовлений, ці вимоги описують, як має працювати об’єкт, частина обладнання або процес.

Вимоги користувача пропонують дані, які можна використовувати для подальшої специфікації, створення та перевірки виробничої системи (тобто проектне рішення від постачальника для задоволення вимог користувача, яке оцінюється під час процесу перегляду проекту/кваліфікації).

Діяльність щодо введення в експлуатацію та кваліфікації має бути організована таким чином, щоб на завершення процесу було письмове підтвердження того, що критерії користувача були задоволені.

ISPE поділяє вимоги користувача на дві категорії: процес і загальні вимоги. Вимоги користувача процесу (PUR) пов’язані з якістю продукції та/або відповідністю нормативним вимогам GMP (належної виробничої практики). Ці вимоги мають бути кваліфіковані або підтверджені як наявні та діючі відповідно до проекту та задокументовані в документації, затвердженій відділом якості.

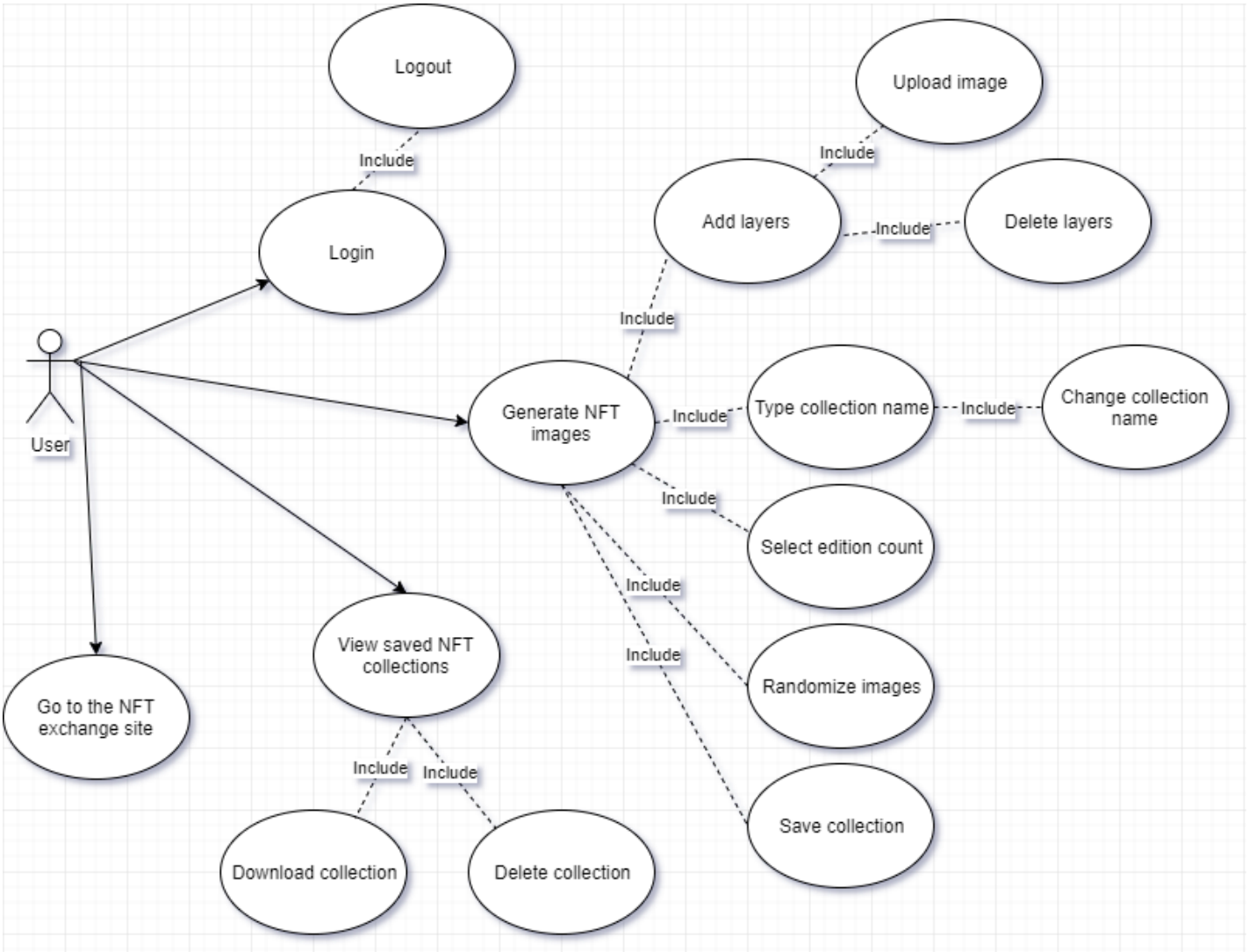

Рис 2.1. Діаграма сценаріїв використання

2.5. Нефункціональні вимоги проекту.

Нефункціональні вимоги – описують характеристики системи та її середовища, а не поведінку системи.

Він також може надавати список обмежень, накладених на дії та функції, які виконує система.

До них відносяться часові обмеження, обмеження процесу розробки системи, стандарти тощо.

Нефункціональні вимоги не мають прямого відношення до функцій системи. Вони пов’язані з функціями інтеграції системи, такими як надійність, час відгуку та розмір системи. Нефункціональні вимоги, такі як пропускна здатність вводу-виводу або формати даних, що використовуються в інтерфейсі системи, також можуть накладати обмеження на систему.

Нефункціональні вимоги засновані на бюджетних обмеженнях, враховують організаційні можливості компанії-розробника, здатність розробленої системи взаємодіяти з іншим програмним забезпеченням і обчислювальними системами, а також зовнішні фактори, такі як правила безпеки, законодавство про захист інтелектуальної власності тощо.

Нефункціональні вимоги визначають цілі та характеристики якості. Атрибути якості — це додаткові описи властивостей продукту, виражені через опис його характеристик, важливих для користувачів або розробників.

Ці характеристики включають:

- легкість і зручність використання;
- легкість пересування;
- цілісність;
- ефективність і стійкість до невдач;
- зовнішні взаємодії між системою та зовнішнім світом;
- обмеження проектування та впровадження. Обмеження стосуються вибору можливості розробки зовнішнього вигляду та структури виробу.

Найпоширенішими з них є продуктивність, масштабованість, портативність, сумісність, надійність, доступність, безпека, локалізація та зручність використання.

Дві основні нефункціональні потреби, без яких не може обійтися жодна система, це продуктивність і масштабованість. Ми об’єднали їх в один розділ, оскільки вони йдуть рука об руку [12].

Продуктивність означає, наскільки швидко програмна система або її окремий компонент реагує на певні дії користувача під час обробки певного робочого навантаження. Враховуючи поточну базу користувачів у цілому, ця статистика часто вказує, як довго користувач повинен чекати, перш ніж відбудеться цільова операція (сторінка візуалізується, транзакція виконується тощо).

Але це не завжди так. У специфікаціях продуктивності можуть перераховуватися непомітні для користувачів фонові завдання, наприклад резервне копіювання. Натомість зосередимося на продуктивності, орієнтованій на користувача.

Масштабованість вимірює найвищі робочі навантаження, які система може впоратися, забезпечуючи при цьому необхідні рівні продуктивності. Коли робочі навантаження збільшуються, ваша система може рости вертикально або горизонтально за допомогою одного з двох методів. До пулу серверів додається більше машин, щоб увімкнути горизонтальне масштабування. Вертикальне масштабування стає можливим завдяки оновленню центрального процесора та оперативної пам’яті поточних машин.

У сфері нефункціональних потреб сумісність і портативність є ще двома важливими факторами.

Те, як система або елемент може бути запущений у тому чи іншому середовищі, визначається переносимістю. Як правило, він містить відомості про апаратне забезпечення, програмне забезпечення чи іншу платформу використання.

Простіше кажучи, він визначає, наскільки безперебійно дії, що

виконуються на одній платформі, виконуються на іншій. Крім того, він визначає, наскільки легко два окремі середовища можуть отримати доступ до різних компонентів системи та спілкуватися з ними.

Сумісність є ще одним компонентом портативності, який описує, як дві системи можуть вижити в одному середовищі. Програмне забезпечення, встановлене в операційній системі, наприклад, має бути сумісним із брандмауером і антивірусним програмним забезпеченням.

Встановлено операційні системи, апаратне забезпечення, браузери, програмне забезпечення та їх версії, переносимість і сумісність.

На даний момент веб-додатки часто розробляються як кросплатформні, кросбраузерні та мобільні.

Портативність - нефункціональні вимоги часто ґрунтуються на попередніх дослідженнях ринку, дослідженнях на місці або аналітичних дослідженнях програмного та апаратного забезпечення, яке використовує цільова аудиторія. Якщо ви працюєте в корпоративному середовищі, і доступ до програмного забезпечення буде здійснюватися через певний список пристроїв і операційних систем, оголосити про сумісність і переносимість просто.

Надійність системи або компонента вказує на ймовірність того, що він буде добре функціонувати протягом заздалегідь визначеного періоду часу за конкретних обставин.

Ця ймовірність зазвичай вказується у відсотках. Наприклад, якщо система має 85-відсотковий рівень надійності протягом місяця, це означає, що протягом цього місяця існує 85-відсоткова ймовірність того, що вона не зазнає критичних збоїв за нормальних умов використання.

Важко вказати катастрофічний збій, час і умови регулярного використання, як ви могли собі уявити. Обчислення середнього часу до відмови або підрахунок кількості критичних помилок, виявлених у виробництві за певний період часу, є двома іншими, дещо простішими підходами до цієї статистики.

Ремонтопридатність — це тривалість часу, необхідного для вирішення проблеми, внесення змін до рішення чи його компонента для покращення продуктивності чи інших характеристик або адаптації до мінливого середовища.

Це можна назвати ймовірністю ремонту з часом, так само як і надійністю. Компонент має 75% шансів бути відремонтованим протягом 24 годин, наприклад, якщо його ремонтопридатність становить 75% протягом цього періоду. Середній час відновлення системи (MTTRS), метрика, часто використовується для оцінки ремонтопридатності.

Доступність означає ймовірність того, що користувач зможе використовувати систему в певний час.

Ви можете визначити його як відсоток часу, протягом якого система доступна для роботи протягом заданого періоду часу, а також його можна представити як очікуваний відсоток успішних запитів.

Наприклад, протягом місяця система може бути доступна 98% часу.

Найважливішим для бізнесу критерієм є, ймовірно, доступність, але для того, щоб описати його, вам також потрібно мати оцінки надійності та ремонтопридатності.

Безпека — це нефункціональний критерій, який гарантує, що кожна частина даних усередині системи буде захищена від атак зловмисного програмного забезпечення та несанкціонованого доступу. Але тут є заковика.

Більшість критеріїв безпеки, які не є функціональними, можна перетворити на фактичні функціональні аналоги.

Щоб запобігти несанкціонованому доступу до панелі адміністратора, ви повинні вказати процес входу та різні ролі користувача як поведінку системи або дії користувача.

Таким чином, у розділі про нефункціональні потреби будуть описані певні небезпеки, які будуть більш детально розглянуті у функціональних вимогах.

Однак це не завжди так. Якщо безпека вашої системи залежить від

певних стандартів і методів шифрування, ці стандарти насправді не описують, як поводиться система; скоріше вони надають інженерам рекомендації щодо реалізації.

Атрибут локалізації визначає, наскільки ефективно система або окремий компонент вписується в загальну структуру цільового місцевого ринку.

Місцеві мови, закони, валюти, культури, правопис та інші фактори є частиною фону. Продукт має бути успішнішим на конкретному цільовому ринку, чим довше він залишається на ньому.

Зручність використання — це ще один традиційний нефункціональний критерій, який відповідає на простий запит: наскільки складним є те, що потрібно використовувати? Визначити ці потреби не так просто, як може здатися.

Існує багато типів критеріїв зручності використання. Однією з найпопулярніших є Nielsen Norman Group, яка пропонує оцінювати зручність використання за п’ятьма параметрами:

- Як швидко користувачі виконують основні дії, коли бачать інтерфейс?
- Як швидко користувачі можуть досягти своїх цілей?
- Як часто користувачі роблять помилки?
- Дизайн приємний у використанні?
- Чи можуть користувачі повернутися до інтерфейсу через деякий час і відразу почати ефективно з ним працювати?

Обов’язково визначте кількісно свої вимоги, щоб визначити, чи відповідає ваша система обмеженням якості.

Необхідно вказати одиниці вимірювання, процедури, які ви будете використовувати, а також ступінь успіху та невдачі.

Подумайте, які ключові інтерфейси та системи потребують таких специфікацій. Встановлення обмежень продуктивності для цих компонентів може бути безглуздим або шкідливим, якщо ваші користувачі ніколи не взаємодіють з ними (наприклад, якщо вони ніколи не використовують панель адміністратора), оскільки ваша команда докладатиме значно більше роботи без

видимої користі.

Таблиця 2.2. перелічує основні нефункціональні вимоги проекту, а також їх описи. Це нефункції, які має виконувати повна програма.

Таблиця 2.2.
Нефункціональні вимоги

Доступність	Програма повинна працювати у веб-браузері. Застосунок повинен підтримувати всі основні сучасні браузери: Google Chrome, Opera, Safari, Mozilla Firefox
Зручність користування	Додаток повинен мати хороший інтерфейс для комфортної роботи з усіма частинами системи. Усі символи, піктограми та зображення мають бути логічно застосовані до відповідної частини системи.
Безпека	У базі даних паролі повинні зберігатися в розшифрованому форматі. Для авторизації слід використовувати Java Web Token, який зберігається в локальному сховищі браузера та надсилається в заголовку запиту. Java Web Token має закінчитися через 24 години.
Ефективність	Програма має правильно завантажувати файли, а всі спливаючі вікна та модальні меню мають відображатися.
Продуктивність	Час відповіді на запит має бути менше 3 с. Програма повинна швидко обробляти всі команди, але за винятком завантаження файлу може зайняти багато часу, це залежить від розміру файлу
Розширюваність	Усі компоненти та класи мають відповідати принципу Open Closed, який дозволяє додавати нові функції, не порушуючи існуючу архітектуру.

Модульність	Програма повинна бути розділена на логічні компоненти - кожен з компонентів повинен виконувати свою функцію
Локалізованість	Система повинна підтримувати англійську мову

2.6. Вимоги до інтерфейсу користувача.

Специфікація інтерфейсу користувача визначає правила взаємодії користувача з певною сторінкою на веб-сайті або екраном у програмі.

Гарні специфікації інтерфейсу користувача враховують дані та контекст користувача в програмі.

Цей тип визначення вимог не замінює дизайн інтерфейсу користувача, але він може допомогти вам керувати вашою командою через процес обдумування дизайну інтерфейсу користувача та того, як користувачі з ним взаємодіятимуть.

Основним джерелом інформації про те, як має працювати програмне забезпечення, є стандарт інтерфейсу користувача. Специфікація інтерфейсу користувача має стосуватися зручності використання, інтернаціоналізації та демонстраційних обмежень на додаток до реалізації.

Ті, хто відповідає за маркетинг, візуальний дизайн і тестування програмного забезпечення, можуть скористатися специфікацією інтерфейсу користувача. Для того, щоб допомогти групі впровадження, визначення інтерфейсу користувача має включати вимоги щодо сумісності, оскільки майбутні дизайнери можуть продовжувати або створювати на основі попередньої роботи.

Специфікацію інтерфейсу користувача можна розглядати як документ, який поєднує функції управління продуктом і впровадження. Однією з найважливіших функцій визначення інтерфейсу користувача є перетворення вимог до продукту в більш чіткий спосіб.

Рівень глибини та формат документа змінюються залежно від потреб організації та практики проектування. Дрібномасштабні прототипи можуть вимагати лише скромної документації з деталями високого рівня [13].

Загалом специфікації вимог пояснюють, що може робити продукт, тоді як специфікації інтерфейсу користувача визначають, як ці потреби досягаються на практиці.

Розробка зовнішнього вигляду - важливий етап роботи над проектом.

Інтерфейс – це все, за допомогою чого учень спілкується з CSR. Організованість проявляється – дуже важлива споживча властивість освітнього продукту.

Оскільки з курсом працює студент (некваліфікований користувач), а кількість робочих сесій, як правило, відносно невелика, тому швидкість і легкість опанування КСВ стає важливою.

Тому необхідно мати можливість випадково опинитися на предметі, що вивчається, і якомога менше думати про способи спілкування з комп'ютером. Крім того, інтерфейс має відображати специфіку дисципліни, що вивчається, зокрема адекватно представляти та сприймати практику в мовній дисципліні – графіку, символіку, взаємодію.

Хоча наразі не існує законодавчих стандартів для графічного інтерфейсу користувача, такі стандарти існують де-факто.

Йдеться, насамперед, про правила використання різних елементів інтерфейсу (елементів керування), таких як смуги прокрутки, кнопки, перемикачі тощо. У зв’язку з цим дизайнер CSR повинен «поважати» умовності щодо варіантів навігації, розміщення елементів навігації, кольорів. і т.д.

Ці ідеї не обмежують дизайн, вони просто поміщають матеріал, що вивчається, у форму, яку можна впізнати з інших додатків, що є найбільш ефективним у цьому конкретному випадку.

Наприклад, якщо ви попросите учня вибрати кілька різних елементів із тих, що представлені на екрані, і використовувати перемикачі для вибору, це може викликати у нього плутанину.

Розробляючи дизайн, необхідно постійно ставити себе на місце користувача і завжди намагатися задовольнити його потреби.

Метою дизайну є не стільки самовираження, скільки презентація об'єкта, що проектується. Для дизайнера на початку має бути важливо не те, що він хоче сказати, а те, як це сприйме споживач.

Тому робота дизайнера повинна будуватися на системі сприйняття типу людей, на яких вона розрахована. Саме в дизайні психологія глядача стає

критичною: якщо не сподобалося, не зрозуміло, значить, не вивчив матеріал, не зацікавився, не запам’ятав, забув.

При виборі візуальних атрибутів інформації, що розміщується на екрані, її верстки, а також при включенні в курс мультимедійних елементів необхідно враховувати психофізіологічні особливості людини.

Перше обмеження, про яке слід знати, це здатність людини до короткочасної пам'яті. Так, «середня» людина не в змозі зберегти в пам'яті інформацію більше ніж про 5 ... 9 об'єктах.

Це означає, що наступний інформаційний кадр не повинен містити більше 9 різних елементів (картинки, фрагменти тексту тощо). Після того як людина припиняє спостереження за об'єктом, його параметри зберігаються в пам'яті протягом обмеженого часу.

Тому, якщо для сприйняття наступного кадру потрібно співвіднести його з одним із представлених раніше, краще відтворити його основні елементи заново (або розмістити гіперпосилання на необхідний кадр).

Інший набір рекомендацій визначається факторами, пов'язаними з право-лівою асиметрією мозку людини. Відомо, що ліва і права півкулі по-різному беруть участь у сприйнятті та обробці інформації.

Зокрема, при запам'ятовуванні слів провідну роль відіграє ліва півкуля, а при запам'ятовуванні образів більш активна права.

Інформація з правого боку екрану надходить безпосередньо в ліву півкулю, а з лівого - в праве (природно, при нормальному зору користувача). Тому текстові повідомлення зазвичай розміщуються в правій частині екрана, а зображення - в лівій.

Дуже серйозним питанням, від якого багато в чому залежить якість сприйняття інформації, є раціональне розміщення даних на екрані та врахування принципів композиції. Не можна забувати і про проблеми психології сприйняття.

Будь-яке зображення викликає у людей асоціації, але є предмети, які викликають однакові асоціації, а є зовсім інші. Це необхідно враховувати при

побудові асоціативних рядів.

Завдання дизайнера – побудувати міцний асоціативний ряд, який допоможе розкрити тему. Перше враження від сторінки завжди важливе.

Користувач, який бажає ознайомитися з матеріалом, повинен спочатку «схопити» текст, коротко прогорнувши сторінку, і вибрати найважливіші слова та вирази.

Висновок

Вимоги до програмного забезпечення — це набір тверджень щодо атрибутів властивостей або якісних систем програмного забезпечення, які використовуються для виклику реалізацій. У результаті аналізу вимог цей об’єкт був створений у процесі визначення вимог до програмного забезпечення.

Для опису вимог можна використовувати як текстові заяви, так і графічні моделі. На етапі проектування програмного забезпечення традиційним технічним методом надається набір вимог. Тести базуються на конкретних вимогах; тому вимоги також включені в процес перевірки програмного забезпечення.

Бізнес-аналітик із відповідними знаннями предметної області повинен зібрати вимоги до проекту лабораторної інформатики вашої організації, щоб розробити успішну стратегію (наприклад, науковий досвід, знання галузі, розуміння лабораторного середовища, знання ІТ, а також досвід у інформатиці система, що впроваджується).

Фазі розробки вимог може передувати техніко-економічне обґрунтування або аналіз концептуального проекту. Ідентифікація вимог (групування, ідентифікація, врахування та уточнення потреб зацікавлених сторін), аналіз (перевірка цілісності та повноти), специфікація (документування вимог) і перевірка допомагають охопити етап розробки вимог.

Перш ніж розпочинати мій проект, вимоги потрібно визначити в повному обсязі.

В результаті основні вимоги до мого проекту були викладені в цій главі.

Найфундаментальнішими є функціональні та нефункціональні вимоги, які були описані в діаграмах і таблицях UML.

РОЗДІЛ 3.

ОГЛЯД ПРОЕКТУВАННЯ ТА АРХІТЕКТУРИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1. Архітектура програмного забезпечення.

Архітектура програмного забезпечення системи відображає організацію або структуру системи та надає пояснення того, як вона поводиться. Система — це набір компонентів, які виконують певну функцію або набір функцій.

Іншими словами, архітектура програмного забезпечення забезпечує міцну основу, на якій можна створювати програмне забезпечення.

Кілька архітектурних рішень і компромісів впливають на якість, продуктивність, ремонтопридатність і загальний успіх системи. Невирішення поширених проблем і довгострокових наслідків може поставити вашу систему під загрозу.

Існує багато високорівневих архітектур і принципів, які зазвичай використовуються в сучасних системах. Їх часто називають архітектурними стилями.

Архітектура програмної системи рідко обмежується одним архітектурним стилем. Натомість комбінація стилів часто становить повну систему.

Архітектура програмного забезпечення розкриває структуру системи, приховуючи деталі реалізації. Архітектура також фокусується на тому, як елементи та компоненти системи взаємодіють один з одним.

Розробка програмного забезпечення глибоко входить до деталей впровадження системи. Завдання проектування включають вибір структур даних і алгоритмів або деталей реалізації окремих компонентів.

Питання архітектури та дизайну часто збігаються. Замість того, щоб використовувати жорсткі та швидкі правила для диференціації архітектури та дизайну, має сенс поєднати їх. У деяких випадках рішення мають явно більш архітектурний характер.

В інших випадках рішення зосереджені на дизайні та тому, як він допомагає реалізувати цю архітектуру.

Важливо зазначити, що хоча архітектура – це дизайн, не весь дизайн є архітектурним.

На практиці саме архітектор проводить межу між архітектурою програмного забезпечення (архітектурним проектуванням) і детальним проектуванням (неархітектурним проектуванням). Немає правил чи вказівок, які підходять для всіх випадків - загалом, робилися спроби формалізувати цю відмінність.

Сучасні тенденції в архітектурі програмного забезпечення припускають, що проект розвивається з часом і що архітектор програмного забезпечення не може знати все наперед, щоб повністю спроектувати систему.

Дизайн зазвичай розвивається на етапах впровадження системи. Архітектор програмного забезпечення постійно перевіряє та перевіряє проект на відповідність вимогам реального світу.

Які проблеми вирішує архітектурний аналіз?

Недоліки програмного забезпечення, які призводять до проблем із безпекою, бувають двох основних видів:

- помилки в реалізації
- недоліки в конструкції

Помилки реалізації становлять щонайменше половину загальної проблеми безпеки програмного забезпечення. Інша половина включає різного роду дефекти програмного забезпечення, які виникають на рівні проектування.

Поділ між дефектами конструкції та помилками приблизно 50/50. Обидва мають бути захищені, щоб забезпечити належне функціонування вашого програмного забезпечення. Ви можете створити найкращий переглядач коду на планеті з найпотужнішими інструментами, відомими людству, але навряд чи ви зможете знайти та виправити дефекти таким чином.

Важлива думка, яка, як мені здається, є джерелом поняття архітектури: «Архітектурні рішення важко змінити», або, іншими словами, «Рішення, які

важко змінити, є архітектурними». І це справді найважливіше.

Наскільки важко змінити рішення, залежить його архітектура. Також важливо, що будь-яке рішення легко змінити наступного дня після його прийняття, або легко змінити рішення, яке мало впливає.

Більш того, в даному випадку «рішення» — це і програмне рішення, і просто угода (яка рано чи пізно все одно закріплюється програмним рішенням).

Загальна структура та поведінка системи представлені програмною архітектурою системи.

Зацікавлені сторони можуть краще зрозуміти та вивчити архітектуру системи, дивлячись на те, як будуть досягнуті такі ключові характеристики, як можливість модифікації, доступність і безпека.

Коли команди приймають рішення щодо системи, а не після впровадження, інтеграції чи розгортання, архітектура програмного забезпечення полегшує аналіз атрибутів системи.

Цей своєчасний аналіз дає змогу командам вирішити, чи забезпечать обрані ними методи прийнятне рішення, створюють вони нову систему, розвивають поточну систему чи оновлюють застарілу систему.

Кожен крок проекту концептуально об’єднаний для всіх зацікавлених сторін завдяки ефективній архітектурі, яка також сприяє гнучкості, економії часу та грошей і ранньому виявленню ризиків проектування.

Може бути важко створити ефективну архітектуру, яка підтримує як довгострокові цілі, так і швидке постачання продукту для сучасних потреб.

Затримки проекту, дорога переробка або ще гірше можуть бути результатом нездатності належним чином визначити, визначити пріоритети та керувати компромісами між архітектурно значущими атрибутами.

Ефективна безперервна еволюція системи стає можливою завдяки ефективній архітектурі програмного забезпечення, що підтримується гнучкими методами архітектури.

Подібні практики включають аналіз розгорнутої системи на відповідність архітектурі та документування архітектурних елементів і зв’язків, призначених

для досягнення ключових якостей.

Ці оцінки повторюються, щоб визначити, чи відповідає архітектура бізнес-цілям і місії організації.

При правильному виконанні ці процедури забезпечують передбачувану якість продукту, менше труднощів у подальшому, економію часу та грошей під час інтеграції та тестування, а також економічно ефективну еволюцію системи.

3.2. Клієнт-серверна архітектура

Архітектура клієнт-сервер базується на двох компонентах: клієнті та сервері.

Клієнт — це комп’ютер на стороні користувача, який надсилає запит на сервер для надання інформації або виконання певних дій.

Сервер - більш потужний комп'ютер або обладнання, призначене для вирішення певних завдань виконання програмних кодів, виконання сервісних функцій за запитом клієнтів, забезпечення доступу користувачів до певних ресурсів, зберігання інформації та баз даних.

Модель такої системи полягає в тому, що клієнт посилає запит на сервер, де він обробляється, а готовий результат надсилається клієнту. Сервер може обслуговувати декілька клієнтів одночасно. Якщо одночасно надходить більше одного запиту, вони ставляться в чергу і виконуються сервером послідовно. Іноді запити можуть мати пріоритети. Запити з більш високим пріоритетом повинні виконуватися раніше [14].

Функції, які реалізовані на стороні сервера:

- зберігання даних, доступ, захист і резервне копіювання.
- обробка запитів клієнтів.
- відправка результату (відповіді) клієнту.

Функції, які реалізовані на стороні клієнта:

- забезпечення інтерфейсу користувача.

- формування запиту до сервера та його відправка.
- отримання результатів запиту та надсилання додаткових команд (запитів на додавання, оновлення або видалення даних).

Архітектура клієнт-сервер визначає принципи зв'язку між комп'ютерами, а правила та взаємодія визначаються в протоколі. Мережевий протокол — це набір правил, за якими комп'ютери взаємодіють у мережі.

Існують поняття побудови клієнт-серверної системи: слабкий клієнт і сильний клієнт.

Слабкий клієнт – потужний сервер. У цій моделі вся обробка інформації передається на сервер, а права доступу клієнта суворо обмежені.

Сервер надсилає відповідь, яка не вимагає додаткової обробки. Клієнт взаємодіє з користувачем, складає і відправляє запит, отримує результат і виводить інформацію на екран.

Сильний клієнт - концепція, в якій частина обробки інформації надається клієнту. У цьому випадку сервер виконує роль сховища даних, а вся робота з обробки та представлення інформації переноситься на комп'ютер клієнта.

Система (додаток), яка базується на взаємодії клієнт-сервер, включає три основні компоненти: представлення даних, прикладний компонент, компонент управління ресурсами та їх зберігання.

Існують дворівнева та трирівнева клієнт-серверна архітектури.

На малюнку 3.1 ви можете побачити загальну структуру дворівневої архітектури.

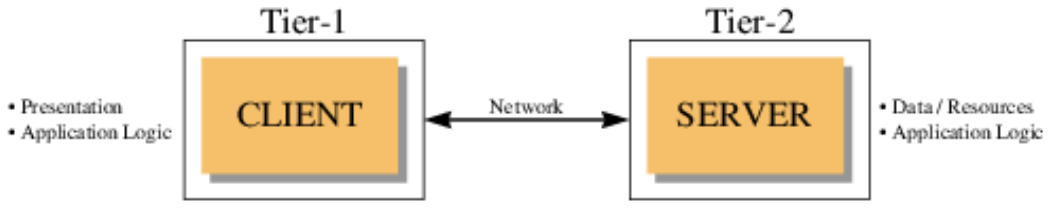

Рис. 3.1. Дворівнева клієнт-серверна архітектура.

Дворівнева архітектура складається з двох вузлів:

- сервер, який відповідає за отримання запитів і відправку відповідей клієнту, використовуючи тільки власні ресурси.
- клієнт, який представляє інтерфейс користувача.

Принцип роботи полягає в тому, що сервер отримує запит, обробляє його і відповідає безпосередньо, без використання сторонніх ресурсів.

На малюнку 3.2 ви можете побачити, які компоненти має трирівнева архітектура клієнт-сервер.

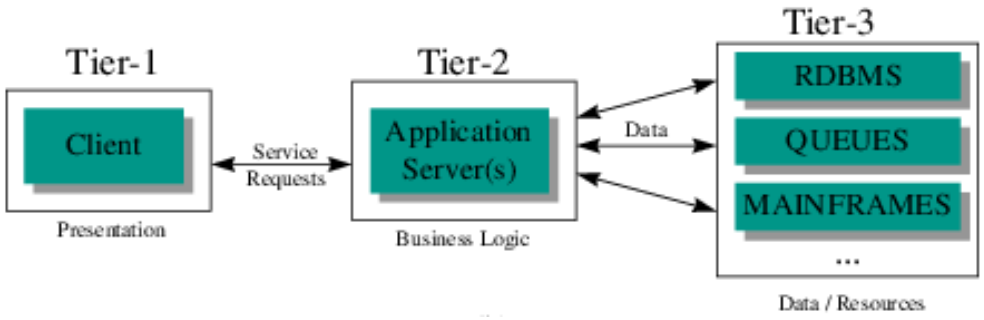

Рис. 3.2. Трьохрівнева клієнт-серверна архітектура.

Трирівнева архітектура складається з наступних компонентів:

- представлення даних – інтерфейс користувача.
- прикладний компонент - сервер додатків.
- управління ресурсами - сервер бази даних, що надає інформацію.

Принцип роботи полягає в тому, що запит клієнта обробляють кілька серверів. Розділення операцій зменшує навантаження на сервер.

Трирівневу архітектуру можна розширити до багаторівневої (N-tier, Multi-tier) шляхом встановлення додаткових серверів. Багаторівнева архітектура дозволяє підвищити ефективність інформаційної системи, а також оптимізувати розподіл її програмно-технічних ресурсів.

Взаємодія клієнт-сервер дозволяє розподіляти функціональні можливості та обчислювальне навантаження між клієнтськими програмами (замовниками послуг) і серверними програмами (постачальниками послуг).

Численні клієнти (віддалені процесори) використовують централізований сервер для запиту та отримання послуг, який відомий як клієнт-серверна архітектура (хост-комп'ютер). Клієнтські комп'ютери пропонують інтерфейс, за допомогою якого користувач комп'ютера може запитувати у сервера послуги та переглядати результати, які надає сервер.

Запити клієнтів отримують сервери, які згодом відповідають на них. Клієнтам не потрібно знати характеристики системи (тобто апаратного та програмного забезпечення), яка надає послуги, оскільки сервер має надавати їм стандартизований прозорий інтерфейс.

Клієнти часто розташовуються на робочих станціях або на персональних комп'ютерах, тоді як сервери розміщуються в іншому місці мережі, як правило, на більш потужному обладнанні.

Ця обчислювальна архітектура працює найкраще, коли і клієнт, і сервер мають певні обов'язки, які вони регулярно виконують.

Наприклад, під час обробки медичних записів клієнтський комп'ютер може запускати прикладну програму для введення даних пацієнта, тоді як серверний комп'ютер може запускати іншу програму для керування базою даних, де дані зберігаються постійно.

Клієнтський комп'ютер може одночасно виконувати інші обов'язки, наприклад надсилати електронні листи, тоді як кілька клієнтів можуть одночасно отримувати доступ до даних сервера.

Колишній підхід до «мейнфрейму», за якого централізований мейнфрейм обробляв усі дії пов'язаних із ним «тупих» терміналів, повністю відрізняється від клієнтсько-серверної моделі, оскільки і клієнтські, і серверні комп'ютери розглядаються як інтелектуальні об'єкти.

Використання інформаційних технологій у бізнесі сьогодні є більш актуальним. Організаціям було б добре витратити значні кошти в цій галузі, щоб збільшити свої шанси конкурувати на світовому ринку.

Згідно з Client/Server Architecture (2011), «конкурентоспроможна глобальна економіка забезпечить старіння та невідомість для тих, хто не може або не хоче конкурувати».

У цій заяві стверджується, що для того, щоб організації могли досягти своїх бізнес-цілей, вони повинні реінжинірингувати поточні організаційні структури та бізнес-практики.

У двох словах, це фундаментальна вимога змінюватися разом із технологічним прогресом. Тому, щоб покращити бізнес-процеси та досягти успіху або вижити на світовому ринку, фірми повинні запровадити метод для отримання та обробки своїх корпоративних даних.

У підході клієнт/сервер усі клієнтські запити обробляються та обробляються сервером, що відображає логічну перспективу розподіленої корпоративної обробки.

Це також можна розглядати як новаторське досягнення для сектору обробки даних. Найефективнішим джерелом технологій, які надають співробітникам повноваження та відповідальність, є обчислення клієнт/сервер.

Основними факторами, що обумовлюють попит на обчислення клієнт/сервер, є «потужність робочих станцій, розширення можливостей робочих груп, збереження існуючих інвестицій, дистанційне керування мережею та бізнес, орієнтований на ринок». Клієнт/серверна архітектура 2011 року. Клієнт/серверні обчислення мають величезний прогрес у комп’ютерній індустрії, залишаючи недоторканими будь-яку область чи куточок.

Для створення клієнт/серверних програм часто потрібні гібридні таланти. До них належать проектування бази даних, обробка транзакцій, комунікаційні можливості, проектування та розробка графічного інтерфейсу користувача тощо.

Просунуті програми вимагають знання компонентної інфраструктури та розподілених об’єктів. Техніка клієнт/сервер, яка найчастіше використовується сьогодні, — це налаштування локальної мережі ПК, оптимізоване групове/пакетне використання.

Завдяки ліквідації обчислень, орієнтованих на хост, це, по суті, стало відправною точкою для багатьох нових розосереджених організацій.

Основні характеристики клієнт-серверної архітектури:

- Для клієнтських і серверних пристроїв потрібні різні апаратні та програмні ресурси.
- Клієнтське та серверне обладнання може надходити від кількох постачальників.
- Збільшення кількості клієнтських комп'ютерів і вертикальна масштабованість (міграція на більш потужний сервер або на багатосерверне рішення)
- Щоб встановити зв’язок і доставити або отримати дані, клієнтська або серверна програма безпосередньо взаємодіє з протоколом транспортного рівня.
- Один комп’ютер серверного класу може надавати численні служби одночасно; однак для кожної служби потрібна окрема серверна програма.
- Щоб надсилати або отримувати певні повідомлення, транспортний протокол використовує протоколи нижчого рівня. Отже, для того, щоб виконати клієнт або сервер, комп’ютеру потрібен повний стек протоколів.

Системи «клієнт-сервер» і «однорангові» системи відрізняються головним чином тим, що однорангові системи дозволяють одноранговим користувачам виконувати функції як постачальників послуг, так і споживачів послуг. В архітектурі клієнт-сервер існують визначені клієнти, які запитують послуги, і сервери, які пропонують послуги.

Крім того, клієнт-серверні системи є дорожчими для створення, ніж однорангові системи, і вимагають центрального файлового сервера.

У той час як однорангові системи покладаються на кінцевих користувачів для керування безпекою, клієнт-серверна модель надає клієнтам доступ до виділеного файлового сервера, який забезпечує надійніший захист. Однорангові мережі також працюють гірше, оскільки кількість вузлів зростає, але клієнт-серверні системи більш надійні та масштабовані до будь-якого розміру. Тому середовище, у якому вам потрібно реалізувати, визначатиме, який вам слід вибрати.

За допомогою технологій організації часто шукають шанси зберегти послуги та конкуренцію за якість, щоб утримати свої позиції на ринку, і тут модель клієнт/сервер має значний вплив.

Коли буде реалізовано обчислення клієнт/сервер, продуктивність підвищиться в результаті використання недорогих користувальницьких інтерфейсів, покращеного зберігання даних, широкого підключення та надійних служб додатків.

За допомогою обізнаного працівника, який може маніпулювати даними та належним чином реагувати на помилки, якщо це правильно впроваджено, це здатне змінити поведінку організації.

Покращений обмін даними: дані, які зберігаються під час звичайних бізнес-операцій і змінюються на сервері, стають доступними конкретним користувачам (клієнтам) через безпечне з’єднання. Використання мови структурованих запитів (SQL) заохочує відкритий доступ з усіх точок зору клієнта, а також демонструє прозорість мережевих послуг, яка показує, що користувачі обмінюються ідентичними даними.

Інтеграція послуг: кожен клієнт має можливість використовувати інтерфейс настільного комп’ютера для доступу до корпоративних даних, позбавляючись від необхідності входу в термінальний режим або інший процесор.

За допомогою баз даних і серверів додатків, розташованих у мережі, інструменти робочого столу, такі як електронні таблиці та презентації Power Point, можна використовувати для роботи з корпоративними даними для отримання корисної інформації.

Спільні ресурси на різноманітних платформах: програми клієнт/сервер створюються незалежно від апаратної платформи чи технічної експертизи відповідного програмного забезпечення (програмне забезпечення операційної системи), сприяючи створенню атмосфери для відкритих обчислень і зобов'язуючи користувачів використовувати клієнти та сервери (бази даних, програми). , сервери зв'язку).

Взаємодія даних: SQL, галузевий стандарт мови визначення даних і доступу, використовується всіма інструментами розробки клієнт/серверних програм для підключення до внутрішнього сервера бази даних.

Це забезпечує послідовне адміністрування корпоративних даних. Користувачі та програми можуть отримати консолідовану картину бізнес-даних, розкиданих на кількох платформах, завдяки передовим продуктам баз даних.

Завдяки можливості оновлювати кілька місць це зберігає цілісність бази даних, одночасно вимагаючи високоякісного опису та відновлення, а не використання однієї цільової платформи.

Можливість обробки даних незалежно від місця розташування: ми живемо в час, коли системи, орієнтовані на користувача, поступово витісняють системи, орієнтовані на машину.

Системи, орієнтовані на машину, такі як мейнфрейми та міні-мікро-додатки, мали відмінні платформи доступу, а продуктивність, безпека, параметри навігації та функціональні клавіші були легкодоступними. Користувачі можуть безпосередньо входити в систему за допомогою клієнта/сервера незалежно від розташування процесорів або їхньої технології.

Просте обслуговування: архітектура клієнт/сервер має перевагу розподіленого підходу, який розподіляє обов'язки між окремими машинами,

під’єднаними через мережу. Хоча це не впливає на клієнтів, сервер легко замінити, відремонтувати, оновити та перенести. Інкапсуляція - це термін для цього неусвідомлення змін.

Безпека: сервери мають потужніші системи контролю доступу та керування ресурсами, щоб гарантувати, що лише авторизовані клієнти можуть отримувати доступ або змінювати дані, а також успішно керувати оновленнями серверів.

3.3. Програмна реалізація

Додаток є веб-сервісом, тому працюватиме в браузері. Додаток створено з використанням мови JavaScript і кількох різних бібліотек для цієї мови, таких як Bootstrap, RequireJs, Canvas API, JSZIP.

Основним елементом, який дозволяє нам працювати з картинками, є Canvas API.

Canvas API надає засоби для малювання графіки через JavaScript і HTML-елемент <canvas>.

Серед іншого, його можна використовувати для анімації, ігрової графіки, візуалізації даних, обробки фотографій і обробки відео в реальному часі. API Canvas здебільшого зосереджений на 2D-графіці. WebGL API, який також використовує елемент <canvas>, малює 2D- і 3D-графіку з апаратним прискоренням.

Canvas — це API двовимірного малювання, нещодавно доданий до HTML і підтримується більшістю браузерів (навіть бета-версією Internet Explorer 9).

Canvas дозволяє малювати все, що завгодно, безпосередньо у веб-переглядачі без використання плагінів, таких як Flash або Java. Завдяки оманливо простому API Canvas може революціонізувати те, як ми створюємо веб-додатки для всіх пристроїв, а не лише для настільних комп’ютерів [15].

Canvas — це API для двовимірного малювання.

По суті, браузер надає вам прямокутну область на екрані, яку ви можете малювати. Ви можете малювати лінії, фігури, зображення, текст; майже все, що

ви хочете.

Canvas спочатку був створений Apple для своїх віджетів Dashboard, але з тих пір він був прийнятий усіма великими постачальниками браузерів і тепер є частиною специфікації HTML 5.

Важливо розуміти, що Canvas призначений для малювання пікселів. Він не має форм і векторів. Немає об'єктів, до яких можна приєднати обробники подій. Він просто малює пікселі на екрані. Як ми побачимо, це і сильна, і слабка сторона.

Є чотири способи малювати речі в Інтернеті: полотно, SVG, CSS і пряма анімація DOM. Канва відрізняється від трьох інших:

SVG: SVG — це векторний API, який малює фігури. Кожна фігура має об'єкт, до якого можна приєднати обробники подій. Якщо ви збільшите масштаб, форма залишиться гладкою, тоді як полотно стане піксельним.

CSS: CSS насправді стосується стилів елементів DOM. Оскільки для речей, які ви малюєте в Canvas, немає об'єктів DOM, ви не можете використовувати CSS для стилізації. CSS впливатиме лише на прямокутну область самого полотна, тож ви можете встановити рамку та колір фону, але це все.

Анімація DOM: DOM, або об'єктна модель документа, визначає об'єкт для всього на екрані. Анімація DOM за допомогою CSS або JavaScript для переміщення об'єктів у деяких випадках може бути плавнішою, ніж у Canvas, але це залежить від реалізації вашого браузера.

Отже, коли слід використовувати Canvas замість елементів SVG, CSS або DOM? Що ж, Canvas є нижчим рівнем, ніж інші, тому ви можете мати більше контролю над малюнком і використовувати менше пам'яті, але за рахунок необхідності писати більше коду. Використовуйте SVG, якщо у вас є фігури, які потрібно відобразити на екрані, як-от карта, створена в Adobe Illustrator. Використовуйте анімацію CSS або DOM, якщо у вас є великі статичні області, які ви хочете анімувати, або якщо ви хочете використовувати 3D-перетворення. Для діаграм, графіків, динамічних діаграм і, звичайно, відеоігор, Canvas є

чудовим вибором. Пізніше ми обговоримо кілька бібліотек, які дозволять вам робити векторні/об’єктно-орієнтовані речі за допомогою Canvas.

На щастя, тепер Canvas є стабільним API, і більшість сучасних браузерів певною мірою його підтримують. Навіть Internet Explorer підтримує його, починаючи з IE 9, і його реалізація дуже хороша.

Підтримувані браузери:

- Safari 3.0+
- Chrome 10+
- Опера 9+
- FireFox 4.0+
- Internet Explorer 9.0+

Що стосується мобільних пристроїв, більшість платформ смартфонів підтримують це, оскільки більшість із них засновані на WebKit, який уже давно має хорошу підтримку.

Я точно знаю, що webOS, iOS і Android це підтримують. Я вважаю, що BlackBerry так, принаймні на PlayBook. У Windows Phone 7 немає, але він може з’явитися в майбутньому оновленні.

Другою дуже важливою частиною програми є Bootstrap.

Безкоштовний фреймворк веб-розробки з відкритим вихідним кодом — Bootstrap. Він пропонує збірник словників для дизайну шаблонів, щоб спростити процес веб-створення адаптивних веб-сайтів, орієнтованих на мобільні пристрої.

Іншими словами, Bootstrap дозволяє веб-дизайнерам швидше створювати веб-сайти, звільняючи їх від тягаря вивчення основних команд і функцій. Він складається зі сценаріїв, створених на платформах HTML, CSS і JS, для різноманітних функцій і операцій, пов’язаних із веб-дизайном.

Основна мета Bootstrap — створювати адаптивні веб-сайти, орієнтовані на мобільні пристрої. Це гарантує належне функціонування елементів інтерфейсу веб-сайту на будь-якому розмірі екрана.

Попередньо скомпільовані та засновані на вихідному коді версії Bootstrap

є обома варіантами. Останньому віддають перевагу досвідчені розробники, оскільки це дозволяє їм адаптувати стилі до своїх проектів.

Наприклад, ви можете отримати доступ до порту Sass, використовуючи версію Bootstrap із «вихідним кодом». Таким чином, ви можете налаштувати та розширити інструмент за потреби, створивши спеціальну таблицю стилів, яка імпортує Bootstrap.

Програма, яка організовує та оновлює фреймворки, бібліотеки та ресурси, є менеджером пакетів, і її також можна використовувати для встановлення Bootstrap.

Найвідоміші менеджери пакетів — Bower, Composer і npm. Поки Composer зосереджується на інтерфейсі, Npm контролює залежності від сервера. Використовуйте Bower замість нього, якщо ви працюєте над проектами на основі PHP.

Через популярність Bootstrap зростає кількість спільнот Bootstrap. Це чудові місця для веб-дизайнерів і розробників, щоб обмінятися інформацією та поговорити про останні виправлення Bootstrap [16].

Перш за все, вивчити Bootstrap просто. Існує багато навчальних посібників і онлайн-форумів, які допоможуть вам розпочати через його популярність.

Простий формат файлу Bootstrap є одним із факторів, що сприяє його величезній популярності серед веб-дизайнерів і розробників. Його файли впорядковано для простого доступу, а для їх редагування потрібне лише практичне знання HTML, CSS і JS.

Теми для відомих систем керування контентом також можна використовувати як навчальні ресурси. Наприклад, Bootstrap, який доступний навіть початківцям веб-розробникам, використовувався для створення більшості тем WordPress.

Bootstrap мінімізує файли CSS і JavaScript, щоб подовжити час, необхідний для завантаження веб-сайту. Крім того, Bootstrap забезпечує одноманітність синтаксису між веб-сайтами та розробниками, що робить його

ідеальним для проектів із залученням команди.

Bootstrap містить вбудовану структуру сітки, що економить ваш час і зусилля, починаючи з нуля. Замість того, щоб додавати медіа-запити до файлу CSS, система сітки, яка складається з рядків і стовпців, дозволяє створити сітку всередині тієї, яка вже існує.

Крім того, архітектура сітки в Bootstrap спрощує процедуру введення даних. Він має масу медіа-запитів, що дозволяє вказати унікальні контрольні точки для кожного стовпця відповідно до вимог вашого веб-проекту.

Зазвичай параметрів за замовчуванням більш ніж достатньо. Вам просто потрібно заповнити контейнери вмістом після створення сітки.

Для кращої підтримки настільних і мобільних проектів сіткова система Bootstrap містить два класи контейнерів: фіксований контейнер (.container) і рідинний контейнер (.container-fluid).

Перший надає контейнер повної ширини, який може адаптувати ваш проект до будь-якої ширини екрана, тоді як перший пропонує контейнер фіксованої ширини.

Доступність вашого веб-сайту в кількох браузерах знижує показники відмов і покращує позиції в пошуковій системі. Bootstrap задовольняє ці критерії, працюючи з останніми ітераціями широко використовуваних браузерів.

Веб-сайти, які використовують Bootstrap, повинні належним чином працювати в менш відомих браузерах, таких як WebKit і Gecko, незважаючи на те, що вони не підтримуються. Однак на невеликих екранах режими та спадні меню можуть бути обмежені.

За допомогою встановлених правил HTML і CSS Bootstrap керує представленням зображень і швидкістю реагування.

Розмір зображень автоматично змінюватиметься залежно від розміру екрана користувача, коли додано клас .img-responsive.

Зменшення розмірів зображень є кроком у процесі оптимізації сайту, тому це покращить продуктивність вашого сайту.

Крім того, Bootstrap пропонує такі класи, як.img-circle і.img-rounded, щоб допомогти змінити форму зображень.

Приклади коду для основних методів також включені в документацію. Щоб уникнути необхідності починати з нуля під час написання коду для ваших програм, ви можете навіть скопіювати та змінити зразки коду.

Для розробників, які зацікавлені в тому, щоб дізнатися, як використовувати цю структуру вперше, Bootstrap пропонує документацію.

Сторінка документації Bootstrap містить такі теми:

- Вміст – охоплює попередньо скомпільований вихідний код Bootstrap.
- Браузери та пристрої – список усіх підтримуваних веб-переглядачів і мобільних браузерів і мобільних компонентів.
- JavaScript – руйнує різні плагіни JS, побудовані на jQuery.
- Тема – пояснює вбудовані змінні Sass для легшого налаштування.
- Інструменти – навчає вас, як використовувати сценарії npm Bootstrap для різних дій.
- Доступність – охоплює структурну розмітку, компоненти, колірний контраст, видимість вмісту та ефекти переходу, функції та обмеження Bootstrap.

Bootstrap має деякі обмеження, які роблять його непридатним для деяких проектів, незважаючи на його переваги.

Оскільки Bootstrap має схожу візуальну естетику, його використання в будь-якому проекті потребує значного налаштування та зміни стилю. Якщо ні, усі веб-сайти, створені за допомогою цього фреймворку, матимуть однакову навігацію, структуру та елементи дизайну, що зробить їх аматорськими.

Велика кількість функціональних можливостей вимагає використання великих файлів. Якщо ви не будете обережні, використання Bootstrap у вашому проекті може спричинити повільніше завантаження веб-сайту та створити навантаження на ваш сервер. Щоб уникнути цієї проблеми, використовуйте мінімізовану версію файлів і додавайте лише ті класи, які вам потрібні.

Найновіші версії популярних браузерів сумісні з Bootstrap, але їхні попередні версії ні. Це означає, що зовнішній вигляд вашого сайту повністю залежатиме від того, наскільки старанно користувачі оновлюють свої браузери.

Ще одним недоліком є той факт, що стилі Bootstrap, як правило, великі. Це може призвести до зайвого виводу HTML і втрати ресурсів центрального процесора.

Хоча Bootstrap дуже простий у використанні, спочатку потрібно трохи навчитися. Необхідно вивчити доступні класи та компоненти, що може бути важко для людини без технічного досвіду.

Набір синтаксису під назвою Bootstrap скомпільований у три основні файли: Bootstrap.css, Bootstrap.js і Glyphicons. Пам’ятайте, що для запуску плагінів і компонентів JS потрібна бібліотека jQuery JS, яка потрібна Bootstrap.

Структура CSS під назвою Bootstrap.css організовує та контролює макет веб-сайту. Тоді як CSS займається самим макетом, HTML займається вмістом і структурою веб-сторінки. З цієї причини обидві структури повинні співіснувати разом для виконання певної дії.

Утиліти в Bootstrap.css дозволяють розробникам створювати узгоджений вигляд на стільки сторінок, скільки необхідно. Завдяки цьому розробнику сайту не доведеться витрачати години на внесення змін вручну.

Вам не потрібно починати кодування з нуля; просто наведіть веб-сторінку на файл CSS. Ви можете внести будь-які необхідні зміни лише в одному файлі.

Функції CSS можна використовувати для форматування інших елементів веб-сайту, таких як таблиці та макети зображень, тому вони не обмежуються лише стилями тексту.

Основним компонентом Bootstrap є Bootstrap.js. Він складається з файлів JavaScript, які відповідають за взаємодію веб-сайту.

Щоб уникнути повторного створення синтаксису JavaScript, розробники часто використовують jQuery, популярну міжплатформенну бібліотеку JavaScript із відкритим кодом.

Ось кілька прикладів можливостей jQuery:

- Виконуйте запити AJAX, як-от динамічне віднімання даних з іншого місця.
- Створюйте віджети за допомогою колекції плагінів JavaScript.
- Створюйте власні анімації за допомогою властивостей CSS.
- Додайте динаміки вмісту сайту.

Хоча Bootstrap може добре працювати з елементами HTML і властивостями CSS, йому потрібен jQuery для створення адаптивного дизайну. В іншому випадку ви обмежені використанням простих статичних компонентів мови CSS [17].

Оскільки піктограми часто представляють дії та дані в інтерфейсі користувача, вони є важливим компонентом інтерфейсу веб-сайту.

Набір Glyphicons Halflings є одним із Glyphicons, який використовує Bootstrap. Незважаючи на просту конструкцію, вони справляються зі своїм основним завданням і відкриті для використання.

Glyphicons пропонує низку наборів преміум-класу для спеціалізованих веб-сайтів, якщо ви шукаєте більш модні значки. Безкоштовне завантаження індивідуальних і тематичних значків також доступне на низці веб-сайтів, зокрема Flaticon, GlyphSearch та Icons8. Ви повинні використовувати властивість CSS font-size, щоб змінити стиль за замовчуванням, якщо ви хочете налаштувати розмір Glyphicons.

Наступні важливі частини - RequireJS і JSZIP, які відповідають за роботу з файлами.

RequireJS — це бібліотека JavaScript і завантажувач файлів, який контролює залежності в модульному програмуванні та між файлами JavaScript. Крім того, це допомагає підвищити швидкість і якість коду.

Девід Марк створив RequireJS, і в 2009 році була доступна перша версія, v1.0.0. Останнім стабільним випуском є версія 2.3.3, і вона є відкритим кодом.

Переваги використання RequireJS:

- Це бібліотека JavaScript з відкритим кодом, ліцензована MIT.
- Пропонується асинхронне завантаження модулів.

- Він може завантажувати вкладені залежності.
- Вам не потрібно турбуватися про запам'ятовування порядку залежностей, якщо у вас багато маленьких файлів.
- Він завантажує кілька файлів JavaScript і підтримує плагіни.

Основні особливості RequireJS:

- Він керує залежностями між файлами JavaScript і покращує швидкість і якість коду.
- Щоб максимізувати продуктивність, він об’єднує та мінімізує модулі в один сценарій.
- У результаті складність коду великих програм зменшується.
- Під час компіляції він збирає кілька файлів JavaScript з різних модулів.

RequireJS можна ініціалізувати, передавши основну конфігурацію в шаблоні HTML через атрибут data-main. Він використовується RequireJS, щоб знати, який модуль завантажити у вашу програму.

Модуль у RequireJS є об’єктом із областю видимості й недоступний у глобальному просторі імен. Тому глобальний простір імен не буде забруднено. Синтаксис RequireJS дозволяє швидше завантажувати модулі, не турбуючись про порядок залежностей. Ви можете завантажити кілька версій одного модуля на одній сторінці.

JSZip — це бібліотека JavaScript для створення, читання та редагування файлів .zip із чудовим і простим API.

Екземпляр JSZip представляє набір файлів. Ви можете додавати їх, видаляти, змінювати. Ви також можете імпортувати наявний файл zip або створити його.

База даних, яка використовується в додатку, це Firebase.

База даних у хмарі Firebase Realtime Database зберігає дані у форматі JSON. Кожен підключений клієнт отримує синхронізацію даних у реальному часі.

Коли ми створюємо міжплатформні програми за допомогою наших пакетів SDK для iOS і JavaScript, усі наші клієнти спільно використовують один екземпляр бази даних у реальному часі та миттєво отримують оновлення з найновішими даними.

Ми можемо зберігати та синхронізувати дані між нашими користувачами в режимі реального часу за допомогою Firebase Realtime Database, бази даних NoSQL [18]. Розробники можуть керувати цим значним об’єктом JSON у режимі реального часу. База даних Firebase надає програмі найновіші значення даних, а також модифікації цих даних за допомогою єдиного API.

Наші споживачі можуть легко отримати доступ до своїх даних з будь-якої платформи, включаючи Інтернет і мобільні пристрої, завдяки синхронізації в реальному часі.

База даних Realtime полегшує спілкування між нашими користувачами. Ми можемо створювати нашу програму без використання серверів, оскільки вона постачається з SDK для мобільних пристроїв і браузерів.

SDK бази даних у реальному часі використовують локальне кешування на пристрої, щоб обслуговувати та зберігати зміни, коли наші користувачі виходять із мережі. Коли пристрій підключається до Інтернету, локальні дані автоматично синхронізуються.

Усі офлайн- та онлайн-послуги можна пропонувати через базу даних у режимі реального часу. Серед цих можливостей доступність із клієнтського пристрою, масштабування численних баз даних і багато інших функцій.

Синхронізація даних використовується базою даних Firebase Real-time замість HTTP-запитів. Протягом мілісекунд будь-який підключений пристрій отримує оновлення. Він пропонує захоплюючий досвід співпраці без урахування мережевого коду.

Оскільки наші дані зберігаються на диску Firebase Database SDK, програми Firebase функціонують нормально, навіть коли вони офлайн. Після відновлення підключення клієнтський пристрій отримує пропущені зміни.

Доступ до бази даних Firebase Real-time можна отримати без сервера додатків. Він доступний безпосередньо через веб-браузер або мобільний пристрій.

Правила безпеки бази даних у реальному часі Firebase, правила на основі виразів, які виконуються під час читання чи запису даних, забезпечують перевірку даних і безпеку.

Ми можемо задовольнити вимоги до даних нашого додатка за допомогою бази даних Firebase Real-time Database на тарифному плані Blaze, розподіляючи наші дані між численними екземплярами бази даних в одному проекті Firebase.

У нашому проекті ми спростимо автентифікацію за допомогою автентифікації Firebase і автентифікуватимемо користувачів у наших екземплярах бази даних.

Правила бази даних реального часу Firebase, які є специфічними для кожного екземпляра бази даних, можна використовувати для обмеження доступу до даних у кожній базі даних.

Отже, Backend-as-a-Service (BaaS) або Firebase — це платформа Google, яка пропонує функції та допомогу в розробці серверної частини ваших Android, iOS або веб-додатків, а також деяких продуктів, які підтримують Unity3D.

Основні функції:

- База даних Firebase: використовується для зберігання даних користувачів, таких як повідомлення чату, відомості про користувачів та інші метадані.
- Хмарне сховище Firebase: використовується для зберігання створеного користувачами вмісту, як-от зображення профілю, мультимедійні повідомлення тощо.
- Firebase Cloud Messaging: використовується для надсилання сповіщень.
- Firebase Remote Config: використовується для виконання A/B-тестування на ходу.

Бази даних реального часу, безсумнівно, були першим продуктом, який зберіг свою привабливість з часом. Після інтеграції хмарного сховища NoSQL база даних Firebase Realtime дозволяє програмам отримувати доступ до міжплатформних даних у реальному часі.

Крім того, ви можете працювати без підключення до Інтернету завдяки нашій базі даних Realtime. Навіть коли ви перебуваєте в режимі офлайн і починаєте синхронізацію після підключення до Інтернету, дані все ще кешуються в пам’яті вашого смартфона.

Крім того, включення функції автентифікації Firebase усуває занепокоєння користувачів щодо безпеки їхніх даних. Так, ви можете вказати дозволи на дані під час використання бази даних Realtime, що є ще однією фантастичною перевагою використання Firebase [19].

Функція Cloud Firestore у Firebase є додатковою перевагою. Завдяки цій базі даних NoSQL програмістам легко транспортувати та зберігати дані для зовнішньої та внутрішньої розробки.

Крім того, його хмарна база даних відома своїми оновленнями в реальному часі, моделями даних, що адаптуються, сумісністю в автономному режимі та швидкими запитами даних.

Подібним чином Cloud Firestore пропонує просту інтеграцію Google Cloud та інших продуктів Firebase. Ви можете отримати доступ до величезної кількості даних, використовуючи ці елементи для розробки програми.

Крім того, Cloud Firestore використовує керування ідентифікацією та доступом (IAM) і суворі рекомендації щодо захисту даних, щоб розвіяти хвилювання програмістів щодо безпеки.

Послуги безпечного та швидкого хостингу, які пропонує Firebase, є ще однією інтригуючою особливістю.

Хостинг Firebase підтримує веб-програми, динамічний і статичний вміст. Крім того, можливість хостингу Firebase доступна незалежно від того, чи хочете ви розмістити API, HTML, CSS або мікросервіси Express.js. Це означає, що Firebase розміщує широкий спектр вмісту.

Підтримка ідентичності користувача є важливою для забезпечення безпеки програм. Однією з найпривабливіших функцій цієї платформи є автентифікація Firebase за підтримки Google, яка є дуже привабливою в сучасному кліматі безпеки в Інтернеті.

Він надає прості в інтеграції бібліотеки інтерфейсу користувача, API на стороні сервера та SDK для перевірки користувачів перед тим, як вони отримають доступ до певної програми Firebase.

Окрім використання електронних адрес, паролів і номерів телефонів для цього процесу, Firebase Authentication також підтримує федеративних постачальників ідентифікаційних даних. Насправді користувачі можуть входити у свої програми за допомогою Google, Twitter, GitHub, Facebook тощо.

Ще одна перевага Firebase, яка спонукає розробників обирати цю платформу розробки програмного забезпечення, — це її послуги тестування. Firebase надає своїм користувачам різні послуги тестування, а не одну.

Важко зрозуміти продуктивність вашої програми, не знаючи поведінки та точки зору користувача.

Для вирішення цієї проблеми Firebase представила свій інструмент моніторингу продуктивності. Він пропонує статистичні відомості про веб-програми, програми для Android та iOS і інформує вас про можливі сфери вдосконалення. Це не сповільнює вашу програму, як це роблять деякі інші системи моніторингу продуктивності.

За допомогою цього інструменту ви можете відстежувати запити HTTP або HTTPS, розмір корисного навантаження, мережеву активність, час відповіді та рівень успішності. Ще однією перевагою інструменту моніторингу продуктивності Firebase є надання індивідуальних звітів на основі географічної інформації, рівня ОС, пристрою та версії програми.

Крім того, Firebase Dynamic Links має вирішальне значення для ваших цифрових маркетингових кампаній.

За допомогою соціальних медіа, маркетингу електронною поштою та реклами через SMS, безсумнівно, легко збільшити кількість установок додатків.

Для соціальних медіа та афілійованих маркетингових кампаній він також пропонує короткі URL-адреси [20].

Після встановлення технологій можна побудувати діаграму класів. Діаграма класів є статичною діаграмою. Він представляє статичний вигляд програми. діаграма класів використовується не тільки для візуалізації, опису та документування багатьох частин системи, але також для створення виконуваного коду для програмного забезпечення.

Діаграми класів показують атрибути та операції класу, а також обмеження системи. Оскільки діаграми класів є єдиними діаграмами UML, які можуть бути безпосередньо зіставлені з об'єктно-орієнтованими мовами, вони часто використовуються в моделюванні об'єктно-орієнтованих систем [21].

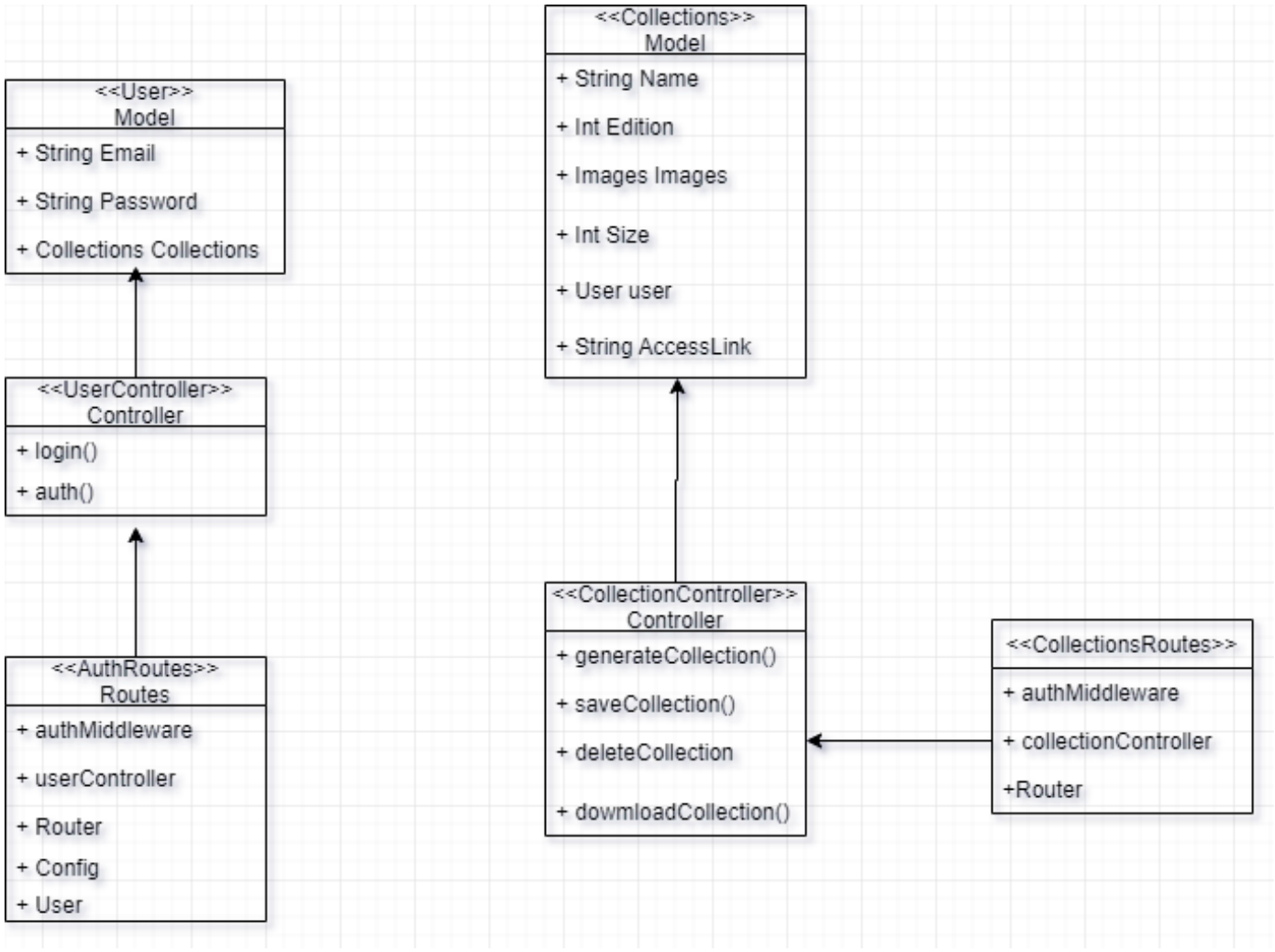

Рис. 3.3. Діаграма класів

Рис.3.3 коротко описує структуру серверної частини моєї програми.

Основні класи серверної частини:

- User – клас використовується для визначення користувача та описує його модель даних.
- UserController – клас, який забезпечує функції входу та автентифікації для користувачів.
    - Login() – надає можливість входу користувача в систему за допомогою дійсної електронної пошти та пароля.
    - Auth() – перевіряє наявність такого користувача в базі даних і надає йому доступ по токену JWT
- Collection – клас використовується для визначення колекцій зображень і описує її модель даних.
- Collection controller – клас, який надає методи для створення колекцій, створення зображень, збереження колекцій, завантаження колекцій, видалення колекцій і отримання колекцій із бази даних.
    - generateCollection() – створення колекції з використанням різних зображень.
    - downloadCollection() – робить запит і надає можливість завантажувати колекції з сервера.
    - deleteCollection() – робить запит і надає можливість видаляти існуючі колекції.
    - saveCollection() – зберігає згенеровану колекцію в базу даних.
- Auth Middleware — проміжне програмне забезпечення (служба інфраструктури) між IIS і контролерами, яке відповідає за автентифікацію користувача та можливі помилки, які можуть виникнути.
- Auth Router – клас, що відповідає за запити до сервера, вхід і аутентифікацію.
- Collections Router – клас, що відповідає за запити до сервера для роботи з колекціями.

Висновок

Документування архітектури спрощує процес взаємодії між учасниками проекту, дозволяє фіксувати рішення, прийняті на ранніх етапах проектування, щодо високорівневого дизайну системи та повторно використовувати елементи цього дизайну та шаблони в інших проектах.

Крім того, клієнт-серверна архітектура відноситься до системи, яка розміщує, надає та керує більшістю ресурсів і послуг, які запитує клієнт. У цій моделі всі запити та послуги доставляються через мережу, і її також називають моделлю мережевих обчислень або мережею клієнт-сервер.

Загалом, переваги використання клієнт-серверної архітектури такі:

- Це централізована система, яка зберігає всі дані та елементи керування в одному місці.
- Це забезпечує високий рівень масштабованості, організації та ефективності.
- Це економічно вигідно, особливо з точки зору обслуговування.
- Це дозволяє відновлювати дані.
- Це дозволяє балансувати навантаження, що оптимізує продуктивність.
- Це дозволяє різним платформам обмінюватися ресурсами.

РОЗДІЛ 4.

ПОСІБНИК КОРИСТУВАЧА

4.1. Огляд.

«NFT Art Generator» — це веб-програма, яка дозволяє генерувати, створювати, зберігати, завантажувати та працювати з колекціями зображень NFT.

Він складається з наступних сторінок:

• головна сторінка – для створення нових колекцій зображень NFT.

• Сторінка MyNFT – для роботи зі збереженими колекціями.

Головна сторінка містить панель навігації та функції для створення колекцій зображень NFT.

На цій сторінці ви можете встановити кількість шарів, написати назву колекції, встановити випуск колекції, створити та зберегти нову колекцію.

Сторінка MyNFT містить список збережених колекцій. На цій сторінці ви можете завантажити та видалити свої колекції зображень NFT.

Крім того, використовуючи посилання на панелі навігації на головній сторінці, ви можете бути перенаправлені на OpenSea. OpeanSea — це децентралізований ринок незамінних токенів для купівлі, продажу та торгівлі NFT.

Окрім цифрового мистецтва, є предмети колекціонування, ігрові предмети, доменні імена, навіть цифрові представлення фізичних активів.

По суті, OpenSea схожий на eBay для NFT з мільйонами активів, організованих у сотні категорій.

Торгівля на OpenSea є мінімальною довірчою операцією. Вам не потрібно вірити в те, що ваш контрагент поводитиметься чесно, і вам навіть не потрібно довіряти OpenSea.

Ваші транзакції залежатимуть від технологій, а не від репутації, і від розумних контрактів, а не від третіх сторін.

4.2. Керівництво користувача.

Авторизація

Щоб скористатися всіма функціями програми, користувач повинен спочатку увійти, використовуючи дійсний обліковий запис Google. Просто натисніть кнопку «Увійти» та виберіть обліковий запис, щоб продовжити.

На малюнку 4.1 ви можете побачити процес входу в програму.

Рис. 4.1. Процес входу

Головна сторінка

Після успішного входу користувач перейде на головну сторінку програми, де він зможе створити нові колекції зображень NFT. Крім того, буде панель навігації з посиланнями на різні сторінки програми.

На малюнку 4.2 ви можете побачити, як виглядає головна сторінка та які функції вона надає.

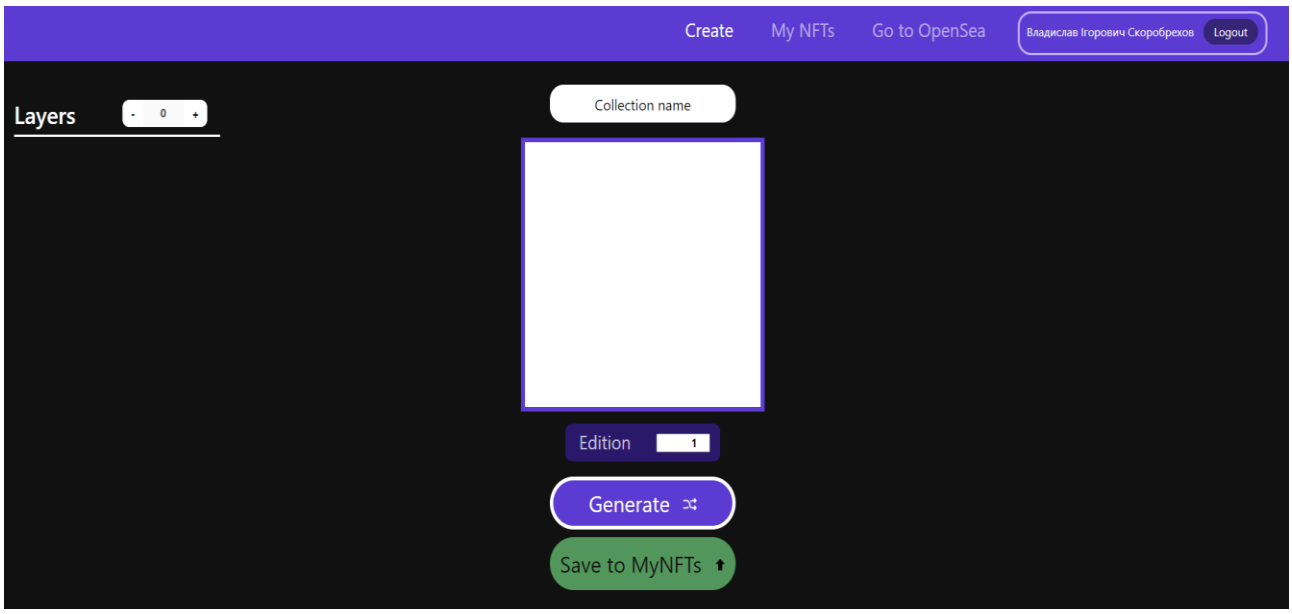

Рис. 4.2. Головна сторінка

Створити нову колекцію

Щоб розпочати створення нової колекції, спочатку потрібно ввести назву та вибрати потрібну кількість шарів. Після цього вам потрібно завантажити різні зображення на шари, встановити видання, а потім створити нову колекцію зображень NFT.

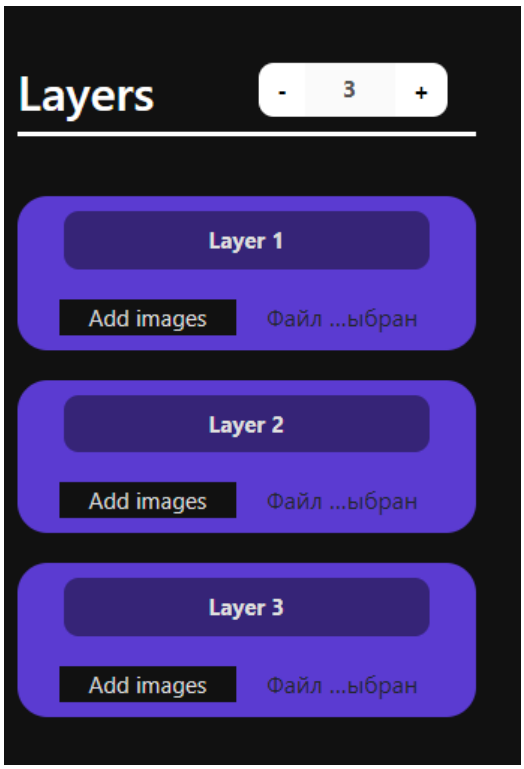

Рис. 4.3. Додавання шарів

Тут ви можете додавати або видаляти шари. Якщо кількість шарів необмежена, просто натисніть кнопки «-» або «+», щоб збільшити або зменшити їх кількість.

Після того, як ви закінчите з підрахунком шарів, вам потрібно завантажити в них зображення.

Натиснувши «Додати зображення», відкриється меню, у якому можна вибрати зображення для завантаження.

На малюнку 4.4 ви можете побачити процес завантаження зображень у шари.

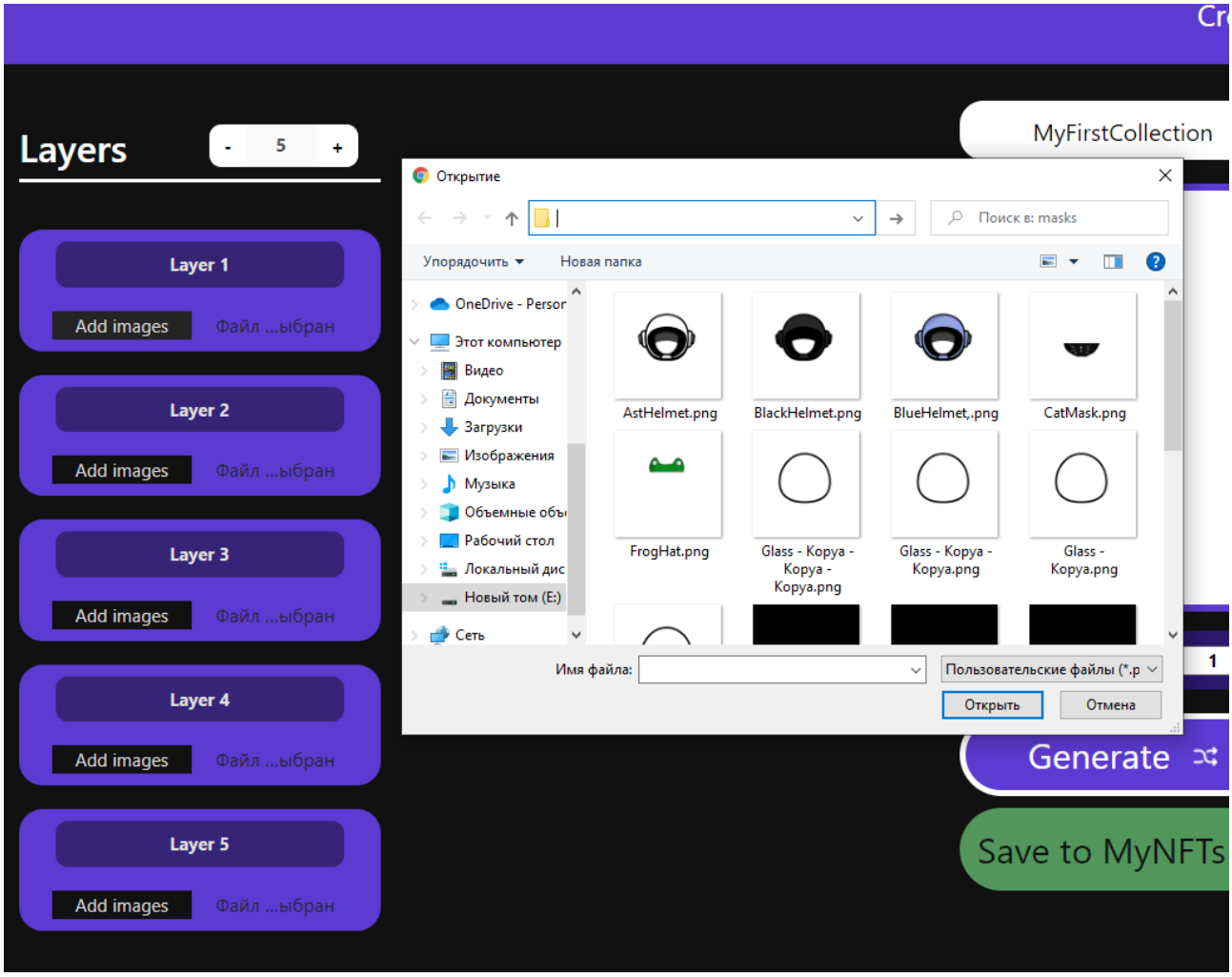

Рис. 4.4. Завантажте зображення

Після вибору та натискання «Відкрити» ваші зображення будуть успішно завантажені. Так потрібно зробити для кожного з шарів.

На малюнку 4.5 можна побачити результат завантаження зображень. Крім того, ви зможете побачити, скільки файлів було завантажено для кожного шару.

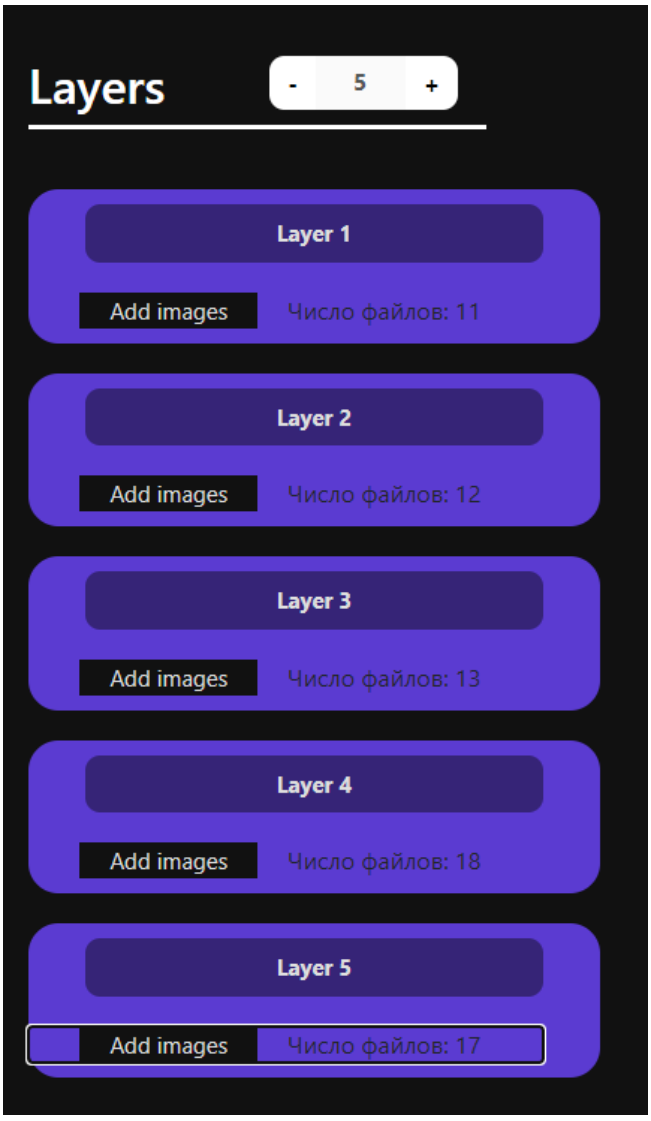

Рис. 4.5. Результат завантаження зображень

Після завершення цієї частини ми можемо переходити до створення нової колекції. Залишається лише ввести назву та вибрати номер видання колекції.

Рис. 4.6. Введення назви колекції

Після цього вам потрібно натиснути кнопку «Створити», і буде створено нове зображення. На малюнку 4.6 ви можете побачити результат цього процесу.

Рис. 4.6. Створено нове зображення

Щоб зберегти нове згенероване зображення, просто натисніть кнопку «Зберегти в MyNFT». На малюнку 4.7 ви можете побачити результат збереження вашого зображення.

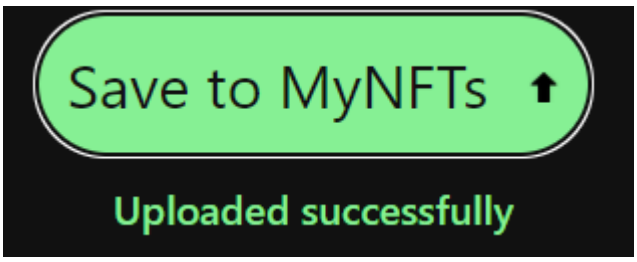

Рис. 4.7. Збережено створене зображення

Щоб переглянути збережену колекцію зображень на панелі навігації, вам потрібно натиснути кнопку «Мої NFT», і ви будете перенаправлені на наступну сторінку.

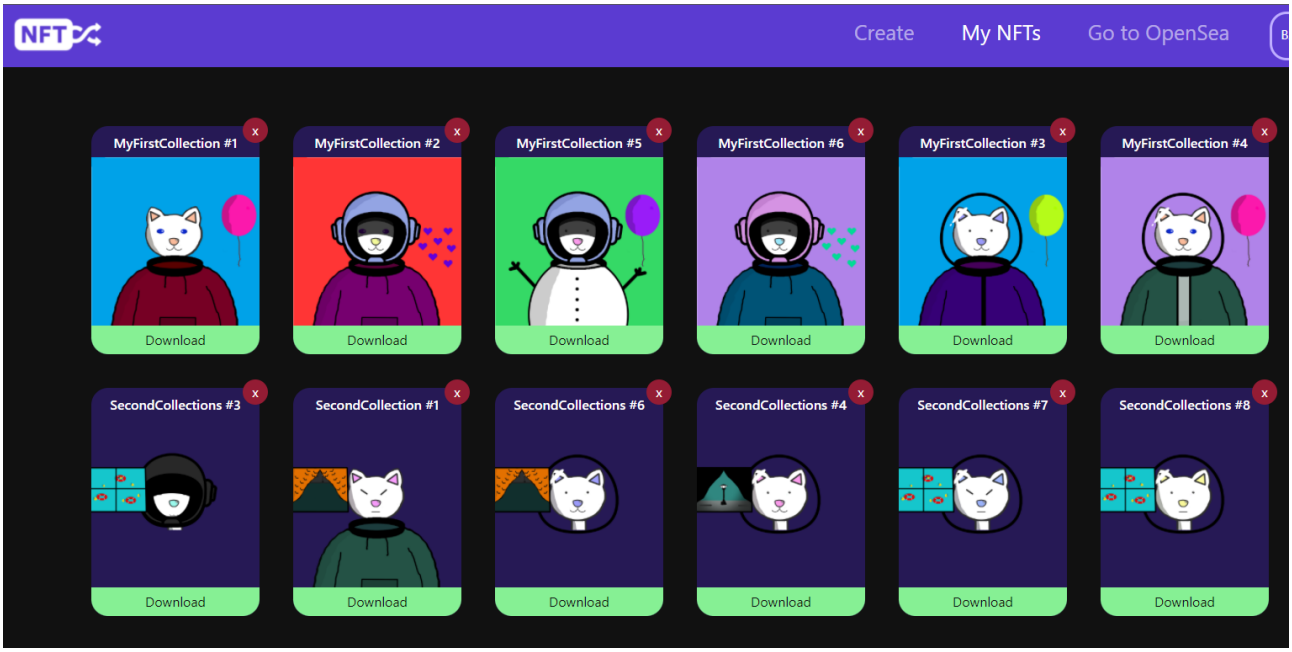

Рис. 4.7. Моя сторінка NFT

На цій сторінці ви можете завантажити або видалити свої NFT. Якщо ви хочете завантажити, просто натисніть кнопку «Завантажити».

Рис. 4.8. Збережене зображення

На малюнку 4.9 можна побачити результат завантаження збережених колекцій.

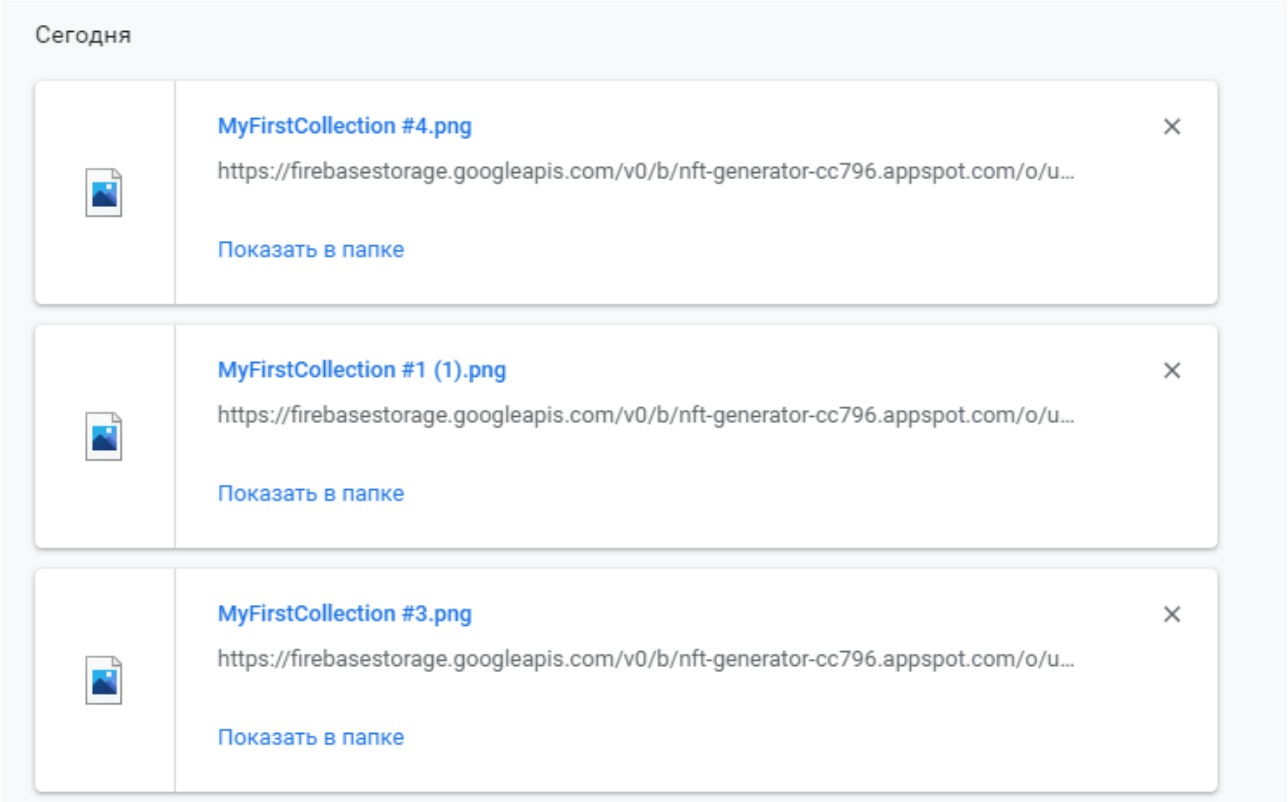

Рис. 4.9. Результат завантаження зображення

А якщо ви хочете видалити один, просто натисніть кнопку «X». На малюнку 4.10 ви можете побачити результат видалення збережених колекцій.

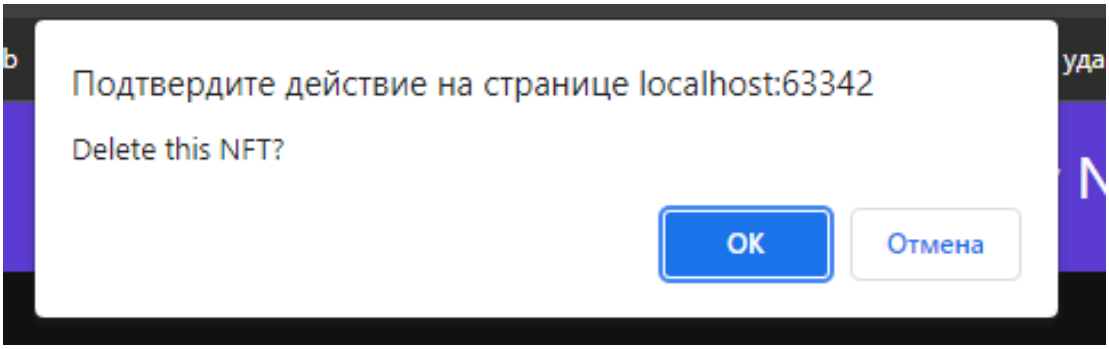

Рис. 4.10. NFT видалено

Висновок

У цьому розділі було описано керівництво користувача та основні можливості системи для комфортного зберігання та генерації колекцій зображень NFT. Тут ви можете створювати нові колекції, зберігати колекції, видаляти колекції, а також завантажувати.

NFT Art Generator — це дуже проста програма, яка допомагає вам працювати з вашими колекціями NFT.

ВИСНОВОК

Під час роботи над дипломним проектом було проведено дослідження з метою вивчення поточної ситуації генерації мистецтва NFT. Проаналізувавши основні особливості та прийоми генерації, я зрозуміла, як цей процес можна вдосконалити та доповнити. З поглибленням у сфері NFT виявилося, наскільки цей токен популярний і важливий, а тому написане програмне забезпечення є хорошою та якісною перспективою. Таким чином, за допомогою «NFT Art Generator» люди зможуть не тільки дуже швидко створювати свої колекції зображень NFT, але й зберігати їх і передавати на ринок обміну NFT.

Я надала детальні вимоги до програмного забезпечення, як функціонального, так і нефункціонального, яке було реалізовано в готовій системі. Окремо були описані системні вимоги, які не дуже високі, оскільки програма повинна відповідати кожному комп'ютеру.

Проект було реалізовано з використанням дуже популярної архітектури клієнт-серверного програмного забезпечення, а також популярного фреймворку Bootstrap з відкритим кодом для кращих функцій інтерфейсу користувача.

В ході роботи весь описаний функціонал було реалізовано, програмою вже можуть користуватися реальні користувачі, щоб допомогти створити свої колекції зображень NFT.

Результати роботи можуть бути використані під час наукових досліджень, при розробці програмного забезпечення, призначеного для створення колекцій креслень для сфери NFT.