КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра інтелектуальних програмних систем

Кваліфікаційна робота
на здобуття освітнього рівня бакалавра
за спеціальністю 121 Інженерія програмного забезпечення
на тему:

СТВОРЕННЯ РОЗВАЖАЛЬНОЇ ПЛАТФОРМИ ДЛЯ ВЕБ ЛОТЕРЕЇ З ВИКОРИСТАННЯМ СЕРВІСУ ХМАРНИХ ОБЧИСЛЕНЬ - AWS

Виконав студент 4-го курсу
Богдан ВОЛОХОНЕНКО _____
(підпис)

Науковий керівник:
доцент, кандидат фіз.-мат. наук
Максим ВЕРЕС _____
(підпис)

Засвідчую, що в цій роботі немає
запозичень з праць інших авторів
без відповідних посилань.
Студент _____
(підпис)

Роботу розглянуто й допущено до захисту
на засіданні кафедри
інтелектуальних програмних систем
«25» травня 2022 р.,
протокол № 10
Завідувач кафедри

Олександр ПРОВОТАР _____
(підпис)

Київ – 2022

РЕФЕРАТ

Обсяг роботи 30 сторінок, 9 використаних джерел, 11 рисунків, 1 GIF зображення.

Ключові слова: AWS, Java, Spring Boot, AWS Lambda, AWS S3, мікросервіси, REST, Web-lottery, Telegram Bot, Thymeleaf.

Об’єктом роботи є створення мікросервісної архітектури для розважальної веб-системи мовою Java з використанням популярного фреймворку – Spring та за допомогою сервісу хмарних обчислень - AWS.

Метою дипломної роботи є ознайомлення з концепцією мікросервісів та хмарних обчислень. Особливостями розвитку і застосування мікросервісів, порівняння підходів та концепцій щодо використання такої архітектури, робота з фреймворком Spring, та найпотужнішою хмарною платформою - AWS а також створення на базі отриманих знань власного веб додатку.

Інструментом розробки обрано IntelijIDEA – сучасний редактор програмного коду, розроблений компанією JetBrains, зручний для багатоплатформної розробки веб-застосунків. Редактор містить відлагоджувач, статичний аналізатор коду, інструменти для роботи з Git, підсвічування синтаксису і засоби для покращення коду. Мова програмування в середовищі розробки – Java.

Результати роботи: було досліджено фреймворк Spring та отримані практичні навички з хмарними обчисленнями. У ході виконання створена мікросервісна архітектура з використанням Spring та AWS

Програмний продукт, можна використовувати у майбутньому, розширюючи та доукомплектовуючи функціонал. Веб додаток буде корисний користувачам, оскільки він має достатній функціонал та унікальність.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ.

AWS – Amazon Web Services

IDE – Integrated Design Environment

API – Application Programming Interface

S3 – Simple Storage Service

SQS – Simple Message Service

Java EE – Java Enterprise Edition

JAR – Java Archive

Amazon EC2 – Amazon Elastic Compute Cloud

IOC – Inversion of Control

АОП – Аспектно-орієнтоване програмування

ВСТУП

Оцінка сучасного стану об’єкта розробки. Із розвитком високих технологій мікросервіси набули величезної популярності. Саме вони дали змогу створювати незалежні один від одного сервіси, відходячи від монолітів. Для підтримки належного рівня якості в сучасному технологічному та конкурентному середовищі, відбувається постійне удосконалення даної сфери. Окрема увага приділяється розробці веб додатків, адже саме вони є одним із основних рушіїв стрімкого прогресу даної архітектури. Багато застосунків дозволяють розробникам розробляти незалежну архітектуру.

Актуальність роботи та підстави для її виконання. Важливою сферою застосування мікросервісів є веб додатки. Вони набули широкого розповсюдження серед користувачів мережі Інтернет, оскільки використовують браузерний інтерфейс і, як правило, не передбачають встановлення додаткового програмного забезпечення. Такі додатки потребують лише наявності браузера та підключення до інтернету. Мови програмування, на яких пишуть такі додатки (Java, PHP, Python та інші), добреукомплектовані і пропонують цілий спектр можливостей для їхнього застосування. Вони займають перші місця у рейтингах мов програмування завдяки тому, що мають пряме відношення до розробки веб додатків.

Мета й завдання роботи. Беручи до уваги сучасні тенденції та розвиток засобів для розробки мікросервісів, метою дипломної роботи є структуризація знань про мікросервісну архітектуру, створення веб додатків, використання сервісу хмарних обчислень, публічних API з інших мікросервісів,огляд наявних технологій та вивчення можливостей для їх практичного застосування. Для досягнення мети поставлено такі завдання:

- Ознайомитися із базовою концепцією бекенд фреймворку Spring;
- Ознайомитися із базовою концепцією хмарних обчислень AWS;
- Ознайомитися із базовою концепцією використання Telegram API;
- Використати набуті знання для створення розважальної веб-системи.

Об’єкт, методи й засоби дослідження та розроблення. Об’єктом роботи є процес ознайомлення, використання технологій розробки для створення мікросервісного веб-додатку.

Під час розробки програмного продукту враховувалися сучасні тенденції та рекомендації, які можна знайти на веб-сайтах для програмістів. Зокрема, опрацьовано статті, в яких зроблена оцінка щодо популярності того чи іншого застосунку і його функціоналу, а також розглянуто переваги та недоліки цього ресурсу під час виконання практичної частини.

Інструментом для створення додатку було обрано IntellijIDEA від компанії JetBrains, що підтримує мову програмування Java. Він пропонує широкий вибір бібліотек, фреймворків, плагінів для комфортної розробки серверної та клієнтської частин, а так само має підтримку AWS SDK.

Можливі сфери застосування. На базі реалізованої практичної частини дипломної роботи можна створювати мікросервіси з використанням сервісу хмарних обчислень, використовувати Telegram API. Програмний продукт, розважальну систему, можна використовувати у майбутньому, розширюючи та доукомплектовуючи функціонал. Система буде корисною користувачам, оскільки вона має свою унікальність, зв’язана з комп’ютерною грою та може принести прибуток. Люди вкладають гроші у своє хоббі.

РОЗДІЛ 1. ОЗНАЙОМЛЕННЯ З ПРОГРАМНОЮ АРХІТЕКТУРОЮ

1.1 Мікросервіси

Мікросервісна архітектура (мікросервіси) – це метод розробки програмних систем, який намагається зосередитись на створенні однофункціональних модулів із чітко визначеними інтерфейсами та операціями. Можна описати мікросервіси, як підхід, коли єдиний додаток будується як сукупність невеликих, незалежних, не тісно зв’язаних між собою сервісів, які спілкуються між собою, за допомогою легких механізмів, наприклад: HTTP. Самі по собі сервіси можуть бути написані на різних мовах і використовувати різні технології зберігання даних.

Однією з причин, чому використання мікросервісів набирає оберту серед компаній у тому, що вони хочуть мати можливість швидко щось змінювати, швидко реагувати на зміни бізнес-вимог, щоб випереджати конкурентів. Мікросервіси допомагають розробникам доставляти зміну безпечно і швидко і з високою якістю, зберігати швидкість розвитку продукту, навіть коли той набирає великих розмірів. В порівнянні з тісно зв’язаними сервісами, ми отримуємо можливість проводити зміни з більшою частотою ітерацій, мінімізуючи вплив змін на систему.

Переваги мікросервісів:

- ***Швидке запускання серверів, що забезпечує розробникам більшу продуктивність та прискорене розгортання***.
- ***Незалежність одного сервісу від інших***.
- ***Масштабування кожного сервісу окремо***.
- ***Помилки в одному з мікросервісів не підірвуть роботу системи. Вони не повинні зламати весь додаток.***

1.2 Amazon Web Services

Мікросервісна архітектура (мікросервіси) – це метод розробки програмних систем, який намагається зосередитись на створенні однофункціональних модулів із чітко визначеними інтерфейсами та операціями. Можна описати мікросервіси, як підхід, коли єдиний додаток будується як сукупність невеликих, незалежних, не тісно зв’язаних між собою сервісів, які спілкуються між собою, за допомогою легких механізмів, наприклад: HTTP. Самі по собі сервіси можуть бути написані на різних мовах і використовувати різні технології зберігання даних.

Однією з причин, чому використання мікросервісів набирає оберту серед компаній у тому, що вони хочуть мати можливість швидко щось змінювати, швидко реагувати на зміни бізнес-вимог, щоб випереджати конкурентів. Мікросервіси допомагають розробникам доставляти зміну безпечно і швидко і з високою якістю, зберігати швидкість розвитку продукту, навіть коли той набирає великих розмірів. В порівнянні з тісно зв’язаними сервісами, ми отримуємо можливість проводити зміни з більшою частотою ітерацій, мінімізуючи вплив змін на систему.

Переваги мікросервісів:

- ***Швидке запускання серверів, що забезпечує розробникам більшу продуктивність та прискорене розгортання***.
- ***Незалежність одного сервісу від інших***.
- ***Масштабування кожного сервісу окремо***.
- ***Помилки в одному з мікросервісів не підірвуть роботу системи. Вони не повинні зламати весь додаток.***

Amazon Web Services – це комплексна служба хмарних, віддалених обчислень, яка забезпечує зберігання, пропускну здатність та індивідуальну підтримку інтерфейсів прикладного програмування для інфраструктури хмарних обчислень. Однією з його ключових переваг є можливість замінити початкові капітальні витрати на інфраструктуру нижчими змінними витратами, що масштабуються зі зростанням бізнесу.

Сервіси, що надаються Amazon Web Services, включають: Amazon Elastic Compute Cloud , Amazon Simple Storage Service, Amazon SimpleDB, Amazon Simple Queue Service та Amazon CloudFront та багато інших додаткових сервісів.

Переваги використання AWS:

- ***Заміна початкових разових інвестицій в інфраструктуру - на щомісячні.***

Налаштування локальної інфраструктури вимагає багато часу, грошей і включає замовлення, оплату, встановлення і налаштування дорогого обладнання, і все це необхідно зробити заздалегідь для майбутнього використання обладнання. Завдяки хмарним обчисленням, сервісам - не потрібно витрачати час на ці дії, ви платите лише за те, що використовуєте.

- ***Швидка розробка програми завдяки гнучкості сервісів***

При використанні традиційного підходу: пошук, постачання та запуск серверів можуть займати тижні. За допомогою хмарних сервісів ви можете виділити кількість ресурсів відповідно до ваших потреб. Ви можете розгорнути сотні або навіть тисячі серверів за лічені хвилини, ні з ким не розмовляючи. Це середовище самообслуговування змінюється так само швидко, як ви розробляєте та розгортаєте програми, дозволяючи вашим командам

експериментувати швидше та частіше.

- ***Глобальне покриття***

Важко забезпечити оптимальну продуктивність для широко розподіленої бази користувача з традиційною інфраструктурою, і більшість компаній можуть зосередитися тільки на економії коштів і часу в одному географічному регіоні за раз. За допомогою AWS справа стає інакшою, і ви можете легко розгорнути свої програми в одному або кількох з 9 регіонів AWS по всьому світу. Тобто ви можете допомогти своїм клієнтам знизити затримки та підвищити якість обслуговування з мінімальними витратами.

1.3 Spring Framework. Spring Boot.

Spring - це платформа з відкритим початковим кодом та середовище розробки додатків на мові програмування - Java. Фреймворк був створений для вирішення складнощів розробки корпоративних застосувань. Однією з основних переваг фреймворка є його багаторівнева архітектура, яка дозволяє користувачам вибирати, які компоненти використати, забезпечуючи при цьому інтегрований фреймворк для розробки додатків J2EE. Використання Spring не обмежується розробкою на стороні сервера. Будь-який Java додаток може виграти від Spring з точки зору простоти, тестируемости і слабкій зв'язаності. Ядром Spring є інверсія управління і аспектно-орієнтоване програмування (AOP).

Spring Framework містить в собі модулі, які надають наступні послуги:

- Spring Core Container: це базовий модуль, який надає головні контейнери - BeanFactory та ApplicationContext
- Аспектно-орієнтоване програмування: можливість реалізації скрізних завдань.

- Аутентифікація та авторизація: налаштовані процеси безпеки, які підтримують різні протоколи, інструменти та практики, як частина підпроекту Spring Security.
- Доступ до даних: робота з системами управління реляційними базами даних на платформі Java з використанням засобів JDBC та ORM та репозиторіїв NoSQL.

Spring Boot.

Для спрощення роботи, розробники Spring створили додаткові підпрограми, які автоматизують процедуру настройки й пришвидшують процес створення Spring-додатків. Однією з таких підпрограм було створено – Spring Boot.

Spring Boot - це проект, метою якого є спрощення створення програм на основі Spring. Він дозволяє найпростішим та найшвидшим способом створити web-додаток, вимагаючи від розробників мінімум зусиль з його настроювання та написання коду. Проект має великий функціонал, але його найбільш значущими особливостями є управління залежностями, автоматична конфігурація та вбудовані контейнери.

1.4 Telegram Bot API

Telegram – це один із найпопулярніших месенджерів. Ним користується вже кілька сотень мільйонів людей у всьому світі.

Telegram Bot – це стороння програма, яка працює всередині Telegram. Користувач може взаємодіяти з ботом, надсилаючи йому повідомлення, команди та вбудовані запити. Керувати ботом можно використовуючи HTTPS-запити до Telegram Bot API.

Функції ботів:

Отримання сповіщень та новин.

Інтеграція зі сторонніми сервісами (YouTube, GIF).

Приймання платежів від користувачів.

Створення ігор.

Створення соціальних служб.

Telegram Bot – це обліковий запис, для створення якого не потрібно додатковий номер телефону.

РОЗДІЛ 2. ТЕХНОЛОГІЇ ДЛЯ РОЗРОБКИ ДОДАТКУ

2.1 Мова програмування

Мова програмування – Java. Було обрано саме цю мову, тому що завдяки цій мові програмування можна реалізувати всі вивчені теоретичні навички на практиці. Java – об’єктно-орієнтована мова програмування з універсальними функціями програмування, які широко використовуються для розробки веб-додатків та мобільних додатків. Має кросплатформу підтримку. Додатки, які були написані на Java можуть бути запущені на будь-яких операційних системах, без встановлення додаткових плагінів.

Не дивлячись на те, що мова програмування досить стара, але її активно підтримують, додаючи додаткові іновації, аби мова не втрачала своєї актуальності та популярності.

2.2 Середовище розробки

Середовищем для розробки веб-додатку було обрано IntelliJ IDEA. Дане середовище дуже зручне для написання програмного коду, має підтримку не тільки мови програмування Java, але і таких мов, як: JavaScript, Kotlin, Scala, Groovy.

IntelliJ IDEA – інтегроване середовище розробки програмного забезпечення для багатьох мов програмування, але загалом для Java.

Має широкий спектр інтегрованих інструментів для рефакторингу, які дозволяють швидко реорганізувати програмістам - код.

Також має інструмент для створення графічного інтерфейсу користувача.

Починаючи з останніх версій – середовище доступно у двох варіантах: *Community Edition* та *Ultimate Edition*.

Community Edition – повністю довільна, безкоштовна версія додатку, яка доступна під ліцензією Apache 2.0. Реалізована повна підтримка Java SE, Kotlin, Scala, Groovy, а також існує інтеграція з найвідомішими та найпопулярнішими системами управління версія , таких як GIT. Існує підтримка інструментів для роботи з базами даних та файлами SQL.

Ultimate Edition – комерційна версія додатку, яка містить в собі додаткові можливості для написання, рефакторингу програмного коду. Реалізована повна підтримка Java EE, UML- діаграм, перевірка покриття коду, статичні аналізатори коду та підтримка фреймворків, наприклад Spring. Інтеграція з серверами для запуску додатків: Tomcat, TomEE, JBoss, GlassFish.

Наразі, IntelliJ IDEA є найпопулярнішим середовищем для розробки, як веб-додатків, так і десктопних додатків.

2.3 База даних

Для збереження даних мною було обрано – PostgreSQL. Сьогодні це одна з найпопулярніших баз даних з відкритим кодом, якщо не найпопулярніша.
PostgreSQL – система управління базами даних з відкритим кодом. Підримує, як SQL для реляційних, так і JSON для нереляційних запитів.
СУБД пропонує розробникам безліч функцій, які допомагають розробникам у створенні продуктів, адміністраторам у створенні відмовостійкого середовища, забезпечивши цілісність та надійність даних.

Переваги:

- Для використання PostgreSQL непотрібно багато тренуватись, він простий у використанні.
- Адміністрування з мінімальними експлуаційними витратами.
- PostgreSQL підтримує роботу з географічними об’єктами. Можна використовувати для сервісів на основі визначення місцезнаходження.
- PostgreSQL підтримує роботу з географічними об’єктами і може бути використано як сховище для зберігання географічних координат.
- Відкритий вихідний код. Кожен користувач, у якого є ліцензія може використовувати вихідний код: додавати, змінювати, видаляти в залежності від потреб бізнесу

Недоліки:

- Повільніший ніж MySQL
- Багато додатків можуть підтримувати MySQL, але не підтримувати PostgreSQL
- Додавання змін для більшої продуктивності, швидкості займають багато часу.

2.4 Сервіси AWS

Під час розробки практичної частини, були використані наступні Amazon Web Services:Amazon S3, Amazon SQS, AWS Lambda.

Amazon S3 (Amazon Simple Storage Service) – сервіс, який надає можливість зберігання і отримання будь-якого об’єму даних в будь-який час та з будь-якої

точки мережі; файловий хостинг.

Amazon S3 використовується багатьма іншими сервісами для зберігання файлів, наприклад Dropbox. S3 містить в собі Buckets.

Buckets – це контейнери для об'єктів, що зберігаються у Amazon S3. Користувач може зберігати будь-яку кількість об'єктів у кошику і обліковий запис може мати до 100 кошиків.

Amazon SQS (Amazon Simple Queue Service) – сервіс черг повідомлень, за допомогою якрнр можна ізолювати та масштабувати мікросервіси, розподілені системи та безсерверні програми. Завдяки SQS можна відправляти, зберігати та отримувати повідомлення компонентів програмного забезпечення в будь-якому масштабі без втрати повідомлень.

SQS пропонує два типи черг повідомлень. Стандартні черги забезпечують максимальну пропускну здатність, оптимальне впорядкування та доставку повідомлень за принципом: «хоча б один раз».

Черги FIFO SQS мають обмежену пропускну здатність і гарантують, що повідомлення будуть оброблятися лише один раз та виключно дотримуючись порядку відправлення.

AWS Lambda – сервіс безсервісних обчислень, який запускає програмний код у відповідь на певні дії, такі як HTTP-запити, зміна об’єктів у S3 і відповідає за автоматичне виділення необхідних обчислювальних ресурсів.

AWS Lambda можна використовувати для створення нових сервісів, програм, які будуть активовані на вимогу за допомогою інтерфейсу прикладного програмування (API) Lambda. Події оброблюються за допомогою Lambda, а не на пристрої клієнта, дозволяючи не залежати від операційних систем, знижуючи енергоспоживання на стороні клієнта.

РОЗДІЛ 3. ОПИС ТА СТРУКТУРА ПРОЕКТУ

3.1 Опис створеного додатку та порівняння з існуючими.

Онлайн веб лотерея дозволяє користувачам витрачати гроші на їхнє хоббі та отримувати задоволення від відкриття кейсів під назвою BaxtyDrop.

Дуже багато геймерів, які грають в гру Counter Strike Global Offensive, мають в своєму інвентарі предмети – зброю. Але не завжди їх задовільняє якість, ціна та рідкість їхнього інвентарю, тому вони йдуть до таких додатків і за свої гроші відкривають кейси, очікуючі на дорогий приз.

При авторизації новий користувач має 10 ключів, завдяки яким може відкрити будь-які кейси на цю сумму.

Кожен користувач може підписатись на телеграм бот і отримувати повідомлення щодо реферальних промо кодів на наступний депозит, який додасть певний відсоток на їх рахунок.

Для опису сценаріїв, за якими може працювати даний сервіс - було створено мережу Петрі.

Завдяки розробленій мережі Петрі, можна побачити поведінку додатка в залежності від описаних дій користувача.

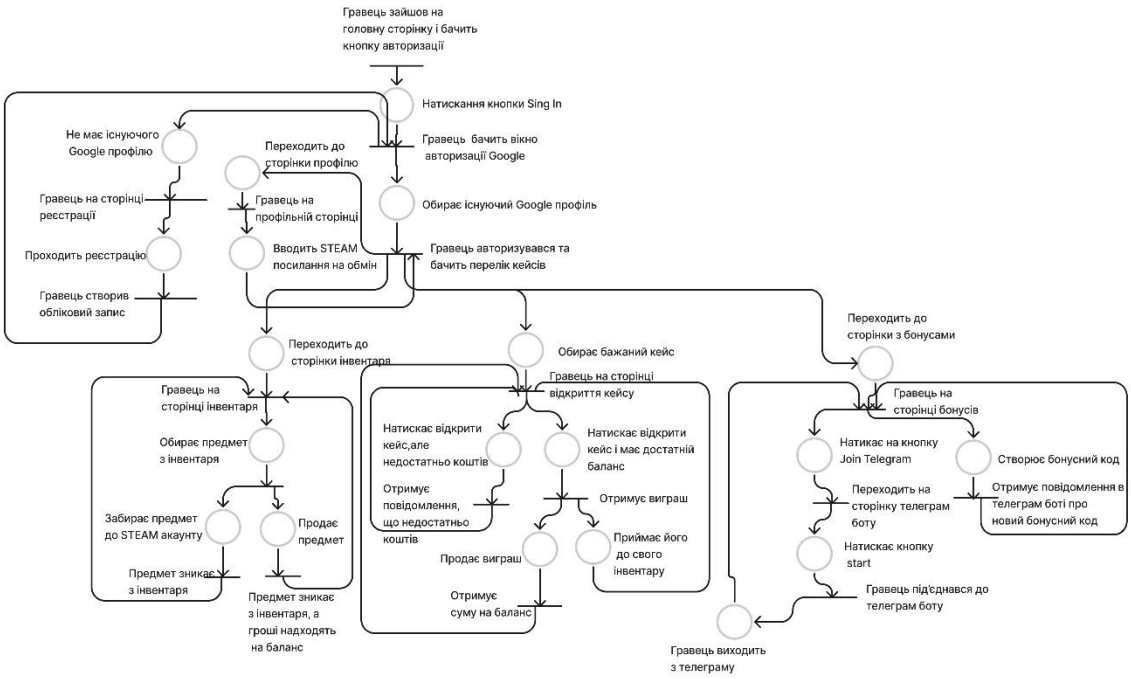

Рис 1. Опис сценаріїв за допомогою мережі Петрі

Порівняно з вже існуючими додатками – BaxtyDrop має перевагу у швидкості, надійності та функціональності.

Завдяки тому, що були використані зовнішні сервіси хмарних обчислень AWS, ми маємо стабільність та надійність даних.

Надійність полягає в тому, що при зломі додатку, шахраї не зможуть видалити статичні дані, бо вони зберігаються на сервісі AWS S3. Тобто, всі дані будуть збережені, так як зламати AWS набагато важче. Користувачі, не проходять зовнішню реєстрацію, де вводять самостійно логін та пароль. Було використано підхід OAuth2.0, який надає повноцінну безпеку при реєстрації через сервіси: Google, GitHub, Instagram, Pinterest.

OAuth2.0 шифрує наші дані таким чином, що ми отримуємо лише кінцевий токен, який через деякий час може бути вичерпаний і доведеться проходити авторизацію знов.

BaxtyDrop має перевагу у функціональності та підтримки користувачів, аби вони отримували якомога більше задоволення. Було розроблено реферальну

систему з використанням сервісів AWS: SQS, Lambda та Telegram Bot. Завдяки такому архітектурному підходу, користувачі зможуть дізнаватись про нові, створені реферальні коди, якомога швидше.

3.2 Авторизація.

Авторизація користувачів проходить з використанням протоколу авторизації OAuth 2.0. Специфікація OAuth 2.0 надає клієнтам безпечний доступ до ресурсів користувача на сервіс-провайдері.

OAuth протокол має підтримку авторизації через соціальні мережі. Користувач авторизується через Google.

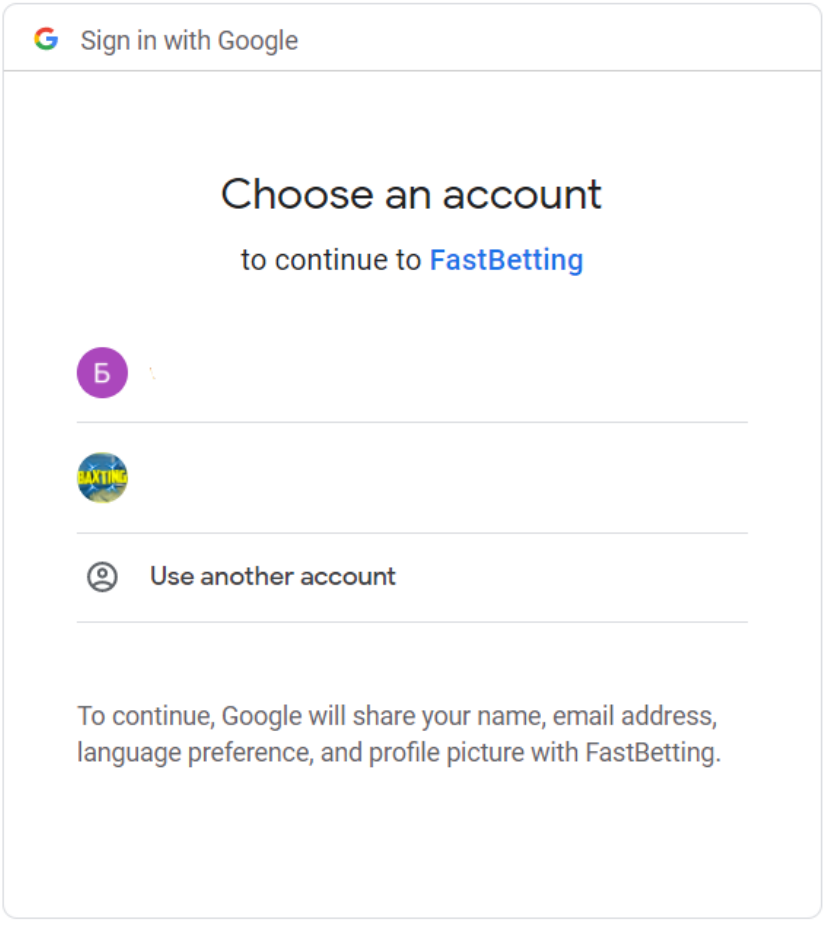

Рис 2. Авторизація через Google

Результатом авторизації є access token , завдяки якому ми можемо переходити до авторизованих ресурсів. Під час наступних запитів, access token буде

доданий до заголовку нашого запиту, тим самим буде розширений доступ для авторизованого користувача до захищених сервісів.

Після авторизації отримуємо OAuth2AuthenticationToken, який містить в собі усі дані про користувача, які можуть бути поширені на інші сервіси: унікальний ідентифікатор, ім’я, зображення профілю, поштова адреса, дата реєстрації. Попередні дані були отримані безпосередньо з аккаунту Google, з якого відбулась авторизація.

Якщо користувач вперше проходить авторизацію – він буде збережений до бази даних, за допомогою свого унікального ідентифікатору. Якщо користувач вже раніше авторизувався – його актуальні дані будуть завантажені після проходження ідентифікації і він не буде знову зберігатись.

У випадку, коли користувач захоче зробити запит на захищений URL він буде переадресований на сторінку авторизації, а запит не буде виконаний.

3.3 Перелік кейсів.

Після успішної авторизації, користувач бачить перед собою головну сторінку – перелік кейсів.

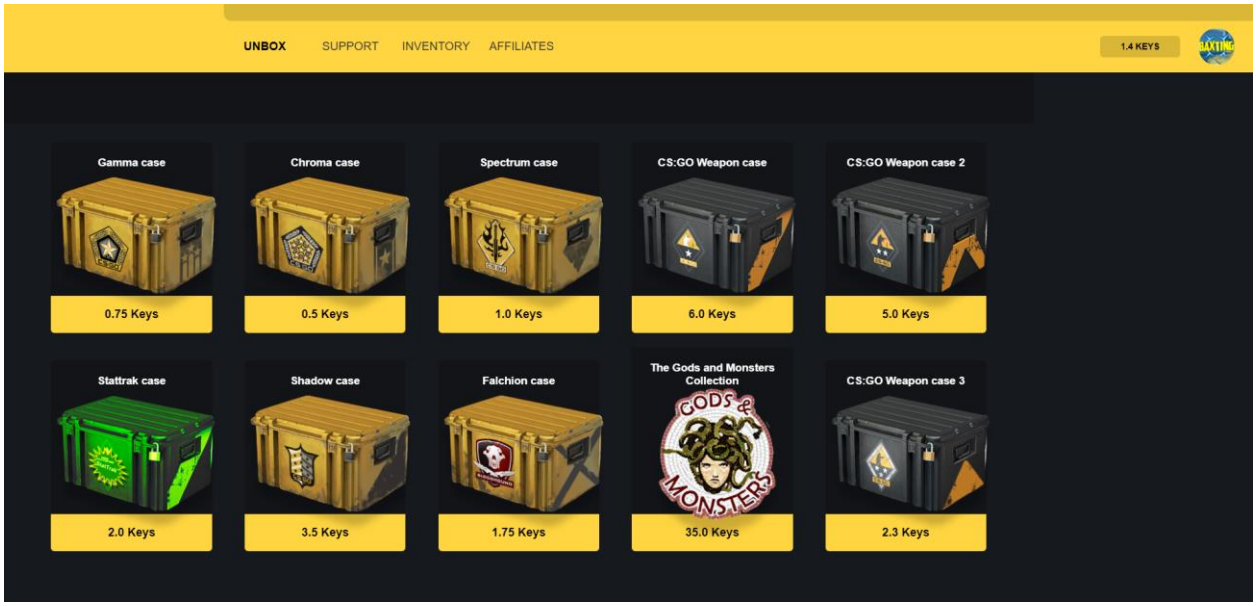

Рис 3 Перелік кейсів.

Кейси не є статичним контентом. Кейси завантажуються через зовнішній сервіс – AWS S3. Було обрано саме S3 тому що цей сервіс дає можливість зберігати файли будь-якого типу, будь-якого рівня, з високим рівнем надійності та доступності. Сервіс надає змогу передбачити випадки, коли на мій сервер може бути виконана кібератака і усі статичні дані, які зберігалися лише на сервері можуть бути знищені, наприклад – зображення. Але з використанням хмарного сервісу, дані будуть зберігатись на віддалених серверах, не матимуть відношення до нашого серверу і будуть надійно захищені.

Для збереження даних у S3, для цього був створений кошик (bucket) під назвою all-cases-csgo.

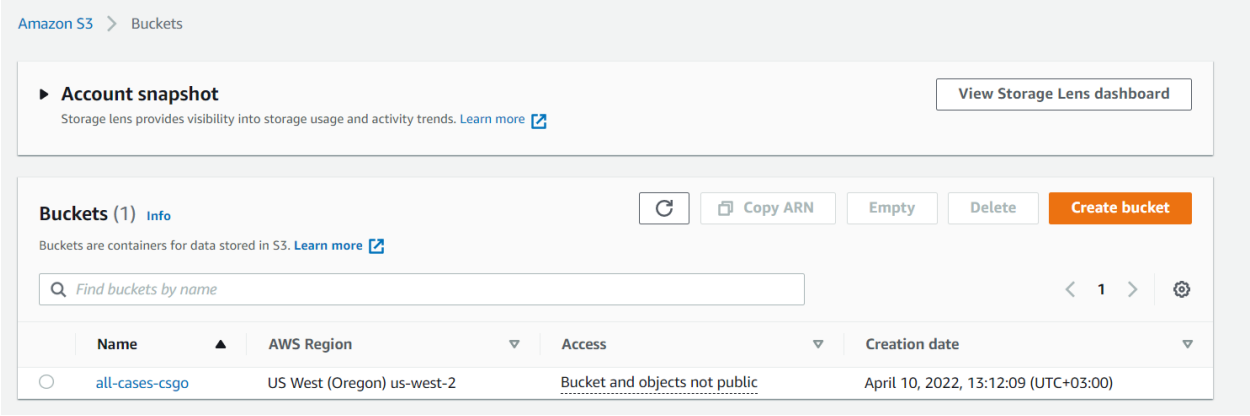

Рис 4 Кошик S3

Кошик може містити статичні дані будь-якого формату: txt, json, png, jpeg і т.д. У кожного кошику має бути свій регіон, для якого він буде доступний і через який до нього можна буде покласти, видалити дані. Якщо регіони будуть різні, то кошик може бути відсутнім у іншому регіоні.

У моєму додатку, S3 використовується як сховище для переліку кейсів.

Кожен кейс представлений у вигляді JSON і має наступну структуру:

```
{
   "name": "Gamma case",
   "price": 0.75,
   "classType": "standard",
```

```
    "caseType": "gamma",
    "type": "case",
    "url": "https://asld.com/csgo/images/Gammacase.png"
}
```


У кожного кейсу є своя назва, ціна, тип, класс та посилання на зображення. Зображення кожного кейсу зберігаються також на окремому, зовнішньому сервері. Кошик зберігає масив кейсів.

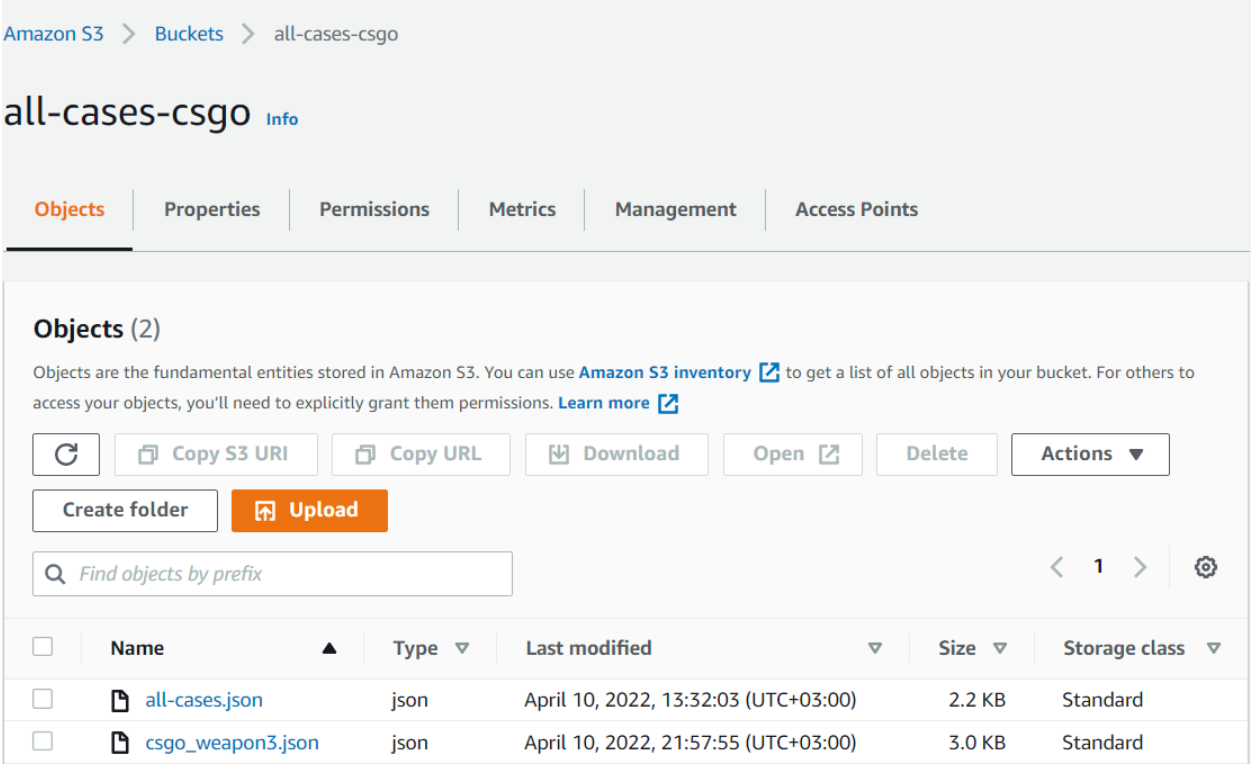

Рис. 5 Вміст кошику

Інколи запит може займати дуже багато часу через деякі причини: сервіс має тимчасові ремонтні роботи, навантаження на зовнішній сервіс надто високе, має обмежену кількість запитів в секунду. Через такі недоліки, користувачі можуть не дочекатись повного завантаження і вирішуть покинути додаток. Задля уникнення таких випадків, вирішено було додатково зробити кешування даних – переліку кейсів та їх контенту.

Кожен кейс та його контент – статичні, незмінні дані і в зв’язку з цим, вони можуть бути закешовані.

Кешування відбувається наступним чином:

1. Користувач робить запит відповідно до зовнішнього сервісу
2. Отримує відповідь від сервісу
3. Дані кешуються в пам’ять
4. Наступний раз коли користувач буде робити цей саме запит, дані будуть діставатись з пам’яті.

3.4 Відкриття кейсу.

Кожен кейс містить в собі предмети різної вартості, рідкості, якості та різного шансу випадання. Існує 5 типів якості зброї:

- Загартоване у боях – найгірша якість у грі, сильно поношений вид скінів; найдешевша вартість.
- Поношене - якість скінів трохи покращується, подряпин і дефектів поменше, але все одно забагато;
- Після польових випробувань - "середня" кількість потертостей;
- Трохи поношене - пошкоджень мало, але вони все ще помітні;
- Прямо з заводу – найкраща якість скінів і найдорожча вартість.

Окрім якості є ще 5 типів рідкості, які визначаються кольором:

- синій – армійська рідкість
- фіолетове – заборонена рідкість
- рожеве – засекречена рідкість
- червоне – тайна рідкість
- жовте – дуже рідкісне

Після обрання кейсу, який хоче відкрити користувач, в нього з’являється вікно з вмістом кейсу (скінами). Кожен користувач, перед придбанням кейсу, тобто

перед його відкриттям має змогу подивитись вміст і що він може отримати, як виграш.

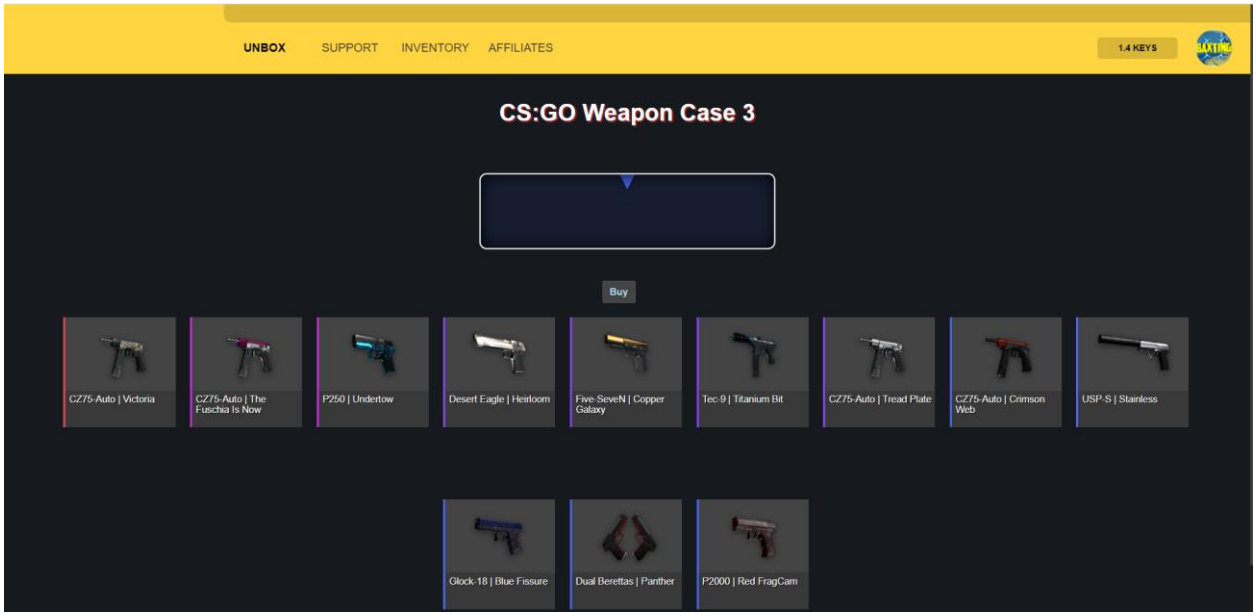

Рис. 6 Вміст кейсу

Кейс містить перелік скінів різної якості та рідкості. Скін має наступну структуру:

```
{
   "name": "CZ75-Auto | Victoria",
   "price": 2.46,
   "classType": "covert",
   "caseType": "csgo_weapon3 stattrak4",
   "type": "skin",
   "quality": "Factory New",
   "url": "https://asld.com/images/CZ75-AutoVictoria.png"
  }
```

Кожний скін має назву, ціну, рідкість, якість, тип. Зображення кожного скіну зберігаються також на окремому, зовнішньому сервері. Масив скінів для кожного кейсу зберігаються у вигляді JSON у кошику S3. (Рис.4.)

Далі, користувач може відкрити кейс, якщо в нього достатньо ключів.

Рандомним чином генерується виграш і крутиться барабан:

GIF 1 Випадання виграшу

Під час отримання виграшу користувач має 2 варіанти:

1. Забрати предмет собі у інвентар
2. Продати предмет за його встановленою додатком вартістю і отримати відповідну суму на баланс

У випадку, коли у користувача недостатньо коштів для відкриття кейсу – він отримає повідомлення аби зробити депозит.

3.5 Бонусні коди.

Кожен користувач, має можливість створити свій власний бонусний, реферальний код для наступного депозиту і поділитись з ним з друзями.
У додатку існує можливість підключення до мережі телеграм.
Користувачі, які мають соціальну мережу телеграм можуть підключити себе до додатку через створений Telegram Bot. Підключення виконується наступним чином:

1. Гравець переходить по посиланню до телеграм боту
2. Натискає кнопку старт і автоматично підключається до додатку завдяки chat_id і зберігається в базу даних.

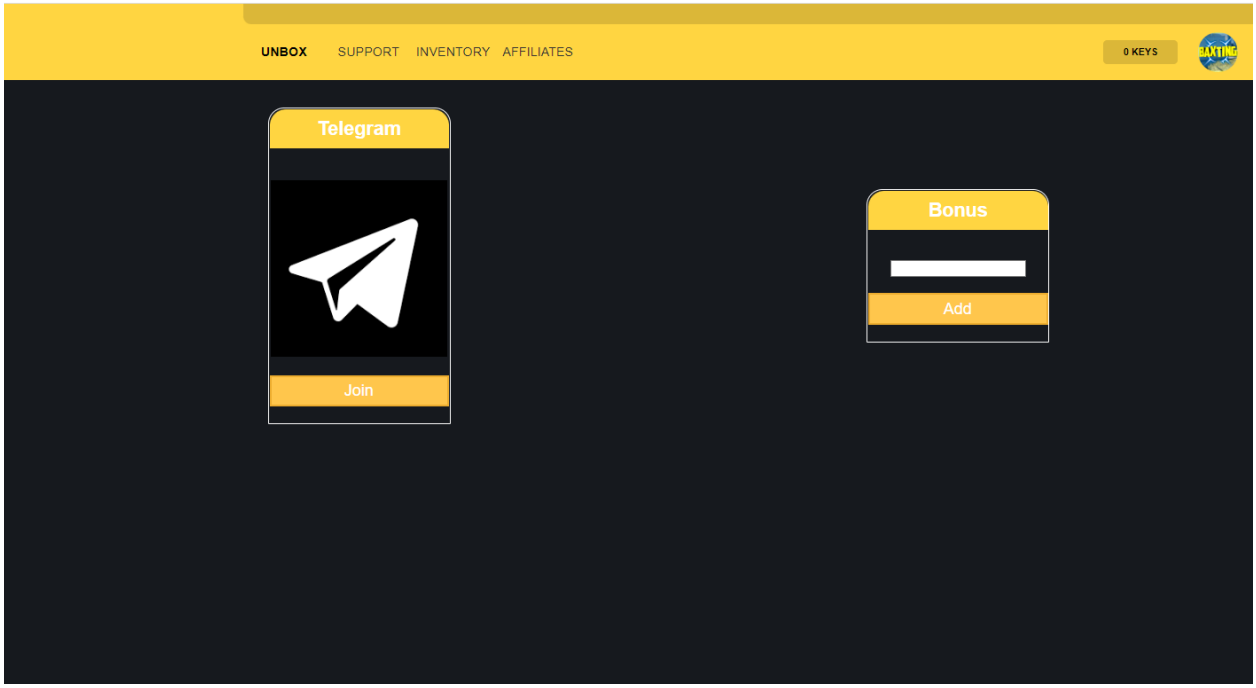

Рис. 7 Підключення до Telegram

3.6 Notifier Bot.

Під час опитування гравців, які використовують подібні сервіси – багато гравців скаржились, що не встигають вводити бонусні коди для додаткових коштів на депозит. Вони не отримують повідомлень, коли був опублікований бонусний код і кількість його введень.

Створено було бот відправки нотифікацій для усіх користувачів, які приєднались до додатку через бота. Коли користувач під’єднується до боту, він може отримувати повідомлення з реферальними посиланнями, бонусними кодами для депозиту, новинами. Також в майбутній розробці можуть бути додані багато різноманітних функцій для підтримки користувачів.

Після старту, гравець робить підписку на бота і коли з’являться нові бонусні коди, новини усі підписники одразу отримають повідомлення і зможуть використати їх.

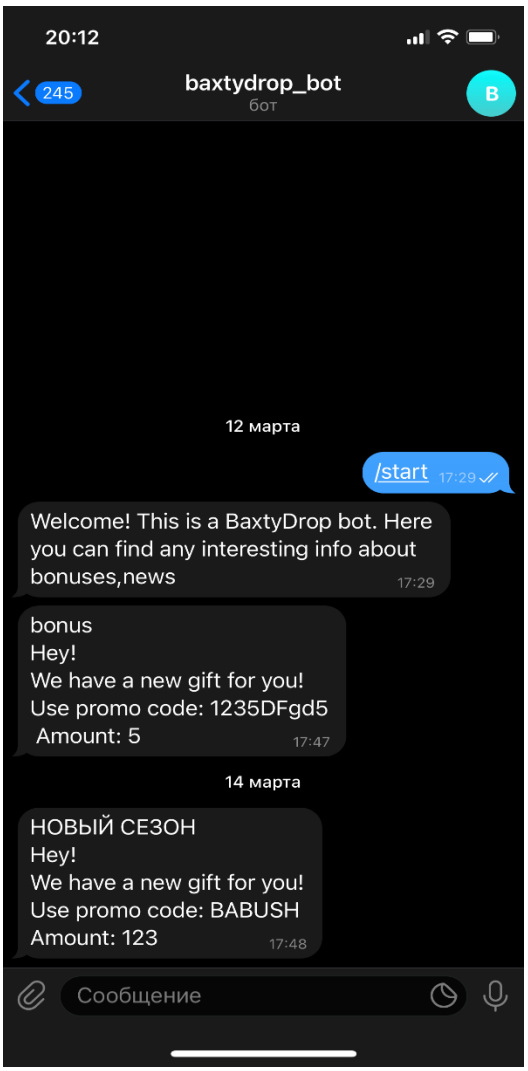

Рис. 8 Notifier Bot

Для надсилання повідомлень через Telegram Bot були використані сервіси: AWS SQS + AWS Lambda.

3.7 Створення повідомлень.

Повідомлення про бонусні коди, реферальні посилання, новини мають структуру JSON.

Коли користувач додає бонусний код – він перетворюється в JSON. Після чого, структура повідомлення надсилається на тимчасове зберігання до SQS.

Мною було обрано зовнішній сервіс SQS через його надійність, безпечність та доступність, завдяки його можливостям, користувачі можуть бути впевнені про те, що їх дані дійдуть до кінцевої точки й не загубляться.

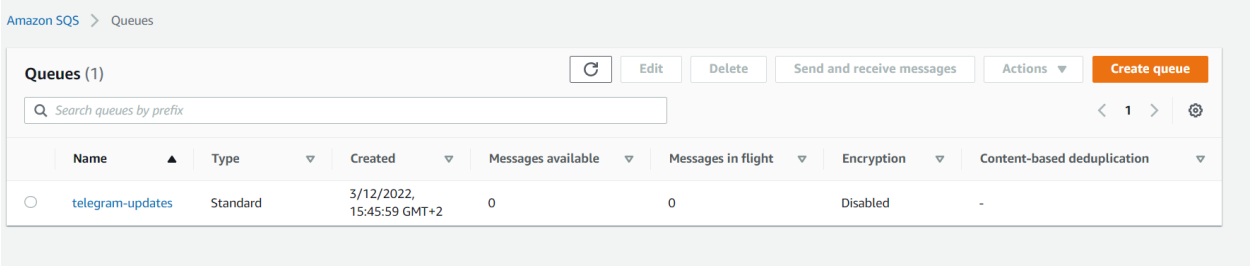

Рис. 9 Черга для повідомлень

Черга повідомлень відіграє роль тимчасового сховища для зберігання. Такий підхід має більшу перевагу, аніж створення додаткової бази даних для зберігання повідомлень. Повідомлення – тимчасовий об'єкт, актуальність якого зберігається певний період, після чого може бути видалена. Не потрібно зберігати бонусні коди, якщо вони скінчились, вони не будуть актуальними. Виходячи з такого принципу, було обрано Simple Queue Service.

3.8 Відправлення повідомлень.

Після того, як користувач додав повідомлення до SQS на тимчасове зберігання, потрібно щоб хтось зчитував ці повідомлення, для цього було обрано сервіс AWS Lambda.

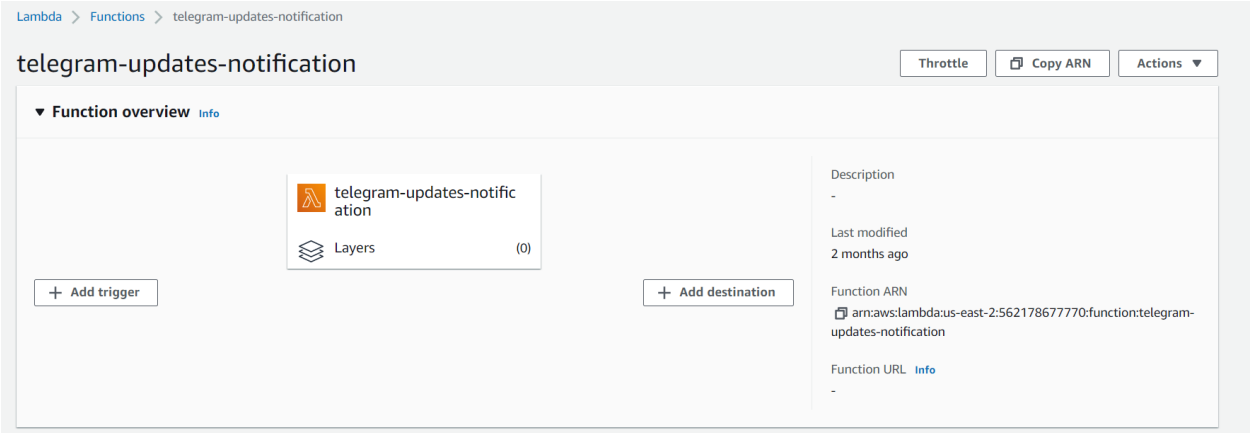

Рис. 10 Lambda для відправки повідомлень

Роль лямбди в додатку має дуже велику цінність, бо вона робить одразу дві речі: зчитує повідомлення і надсилає їх.

Лямбда складається з трьох компонентів:

1. Layers. Це файловий архів, який може містити додатковий код чи дані. Шар може містити бібліотеки, середовище користувача , дані або файли конфігурації. Шари сприяють спільному використанню коду та поділу відповідальності, щоб можна було швидше виконувати ітерації під час написання бізнес-логіки. Для мови Java – це може бути .jar архів з кодом.
2. Function Environment. У коді повинна обов'язково знаходитись функція, яка безпосередньо виконуватиметься при кожному запуску лямбди (Handler). А перед нею знаходиться її середовище, яке ми ставимо. Управління ресурсами відбувається таким чином, що це середовище зберігається окремо від функції якийсь час після її завершення. І при наступному старті, відновлюється, не витрачаючи час та ресурси на ініціалізацію. Таким чином, все що можливо, треба проініціалізувати до самої функції: конфігурації, підключення бібліотек.
3. Handler – безпосередньо сам код для виконання, в залежності від мови програмування, визначається по-різному.

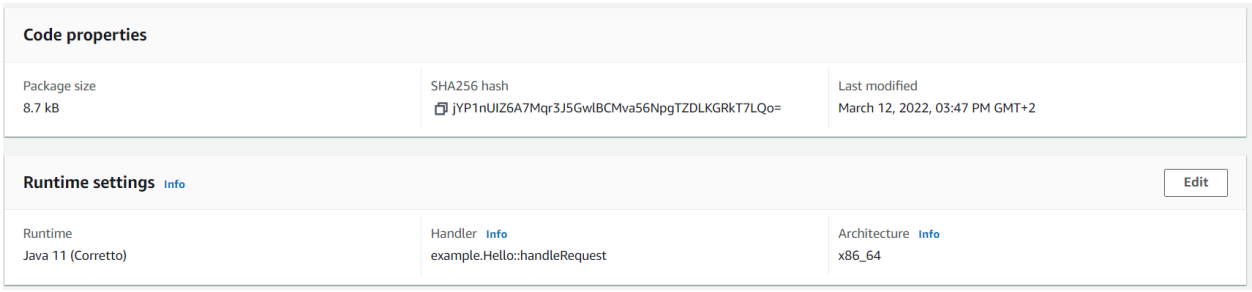

Рис. 11 Налаштування лямбди

Після завантаження лямбди на AWS вона може бути налаштована на автоматичну роботи за таймером. Кожні 10 хвилин, вона буде зчитувати повідомлення з SQS і надсилати їх до телеграм боту конкретного користувача, який підписався.

ВИСНОВКИ.

В результаті розробки розважальної веб лотереї були проаналізовані та порівняні між собою існуючі веб-лотереї та був підібраний вигідний та продуктивний підхід до архітектури, розроблено повноцінний додаток , що представляє собою мікросервісну взаємодію, використання публічного Telegram API, використання зовнішніх сервісів хмарних обчислень.

Додаток представляє собою повноцінні та незалежні модулі бекенд частини, сервіс-функцію AWS Lambda та сервіс телеграм бота.

У системі впроваджено можливість авторизації, реєстрації, отримання та відкриття кейсів, зберігання у тимчасовому інвентарі, можливість продавати виграш, створення реферальних посилань, під’єднання до Telegram боту, отримання додаткової інформації: новини, бонуси в Telegram.

В ході роботи було проаналізовано вже існуючі системи веб лотерей, існуючи підходи для створення архітектури, проаналізовані доступні сервіси хмарних обчислень AWS, порівняні технології для продуктивної розробки системи, створена була архітектура додатку та прототип веб-лотереї. Створений додаток має перевагу серед вже існуючих у тому, що він більш надійний, швидкий, функціональний завдяки використанню AWS ресурсів.

У роботі докладно описано, яким чином розроблялось програмне рішення, та з яких частин воно складається. Розроблено програмне забезпечення, що надає можливість витрачати додаткові ресурси для отримання предметів з гри.

СПИСОК ПОСИЛАНЬ.

1. JetBrains IntelliJ IDEA | Опис продукту [Електронний ресурс]. – Режим доступу: URL: - https://itpro.ua/product/jetbrains-intellij-idea/?tab=description
2. Кейси CS:GO – Що це таке | Counter-Strike: Global Offensive [Електронний ресурс]. – Режим доступу: URL: - https://cyber.sports.ru/cs/1095287799.html
3. Що таке PostgreSQL | CoderLessons [Електронний ресурс]. – Режим доступу: URL: - https://coderlessons.com/tutorials/bazy-dannykh/uchebnik-postgresql/1-chto-takoe-postgresql
4. What is Amazon S3 | Amazon Web Services Documentation [Електронний ресурс]. – Режим доступу: URL: - https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html
5. What is Amazon Lambda | Amazon Web Services Documentation [Електронний ресурс]. – Режим доступу: URL: - https://docs.aws.amazon.com/lambda/latest/dg/welcome.html
6. What is Amazon Simple Queue Service | Amazon Web Services Documentation [Електронний ресурс]. – Режим доступу: URL: - https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/welcome.html
7. Bots: An introduction for developers | Telegram Bot API [Електронний ресурс]. – Режим доступу: URL: - https://core.telegram.org/bots
8. Spring Boot 2.7.0 | SpringIo [Електронний ресурс]. – Режим доступу: URL: - https://spring.io/projects/spring-boot
9. Сучасний підручник з JavaScript | JavascriptInfo [Електронний ресурс]. – Режим доступу: URL: - https://uk.javascript.info/