Приватний заклад вищої освіти

Одеський технологічний університет «ШАГ»

Кафедра інформаційних технологій та фундаментальної підготовки

Випускна кваліфікаційна робота бакалавра

«ІНФОРМАЦІЙНА СИСТЕМА КЕРУВАННЯ АВТОРСЬКИМИ МІКРОБЛОГАМИ»

на здобуття бакалаврського рівня вищої освіти
зі спеціальності «122 Комп'ютерні науки»

Виконавці проекту

№	П.І.Б.	Група
1	Петров В. Е.	КН-А-201
2		
3		
4		
5		
6		

Автор звіту

_____Петров Всеволод Едуардович

Одеса – 2024

АНОТАЦІЯ

Розробка інтернет магазину. Клон Розетки: Автореферат випускної кваліфікаційної роботи на здобуття бакалаврського рівня вищої освіти зі спеціальності «122 Комп’ютерні науки». - Одеса: ОТУШ, 2022 - 13 с.

ЗМІСТ

Головним моїм завданням як devops розробника була реалізація області для роботи програмістів над нашим проектом. Не довго думаючи мною було реалізовано три основні сервери для роботи. Для реалізації серверів і підключення до них для роботи знадобилося реалізувати систему контролю версії на базі git. Далі завдання підійшли до реалізації CI/CD - це поєднання безперервної інтеграції та безперервного розгортання програмного забезпечення в процесі розробки. Після всіх завдань залишилося реалізувати моніторинг серверів для перегляду навантаження і завдань на сервери.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

16

ВСТУП

Розвиток і поширення інформаційних технологій супроводжується перенесенням великої кількості послуг у цифровий простір. Комунікаційні технології стають необхідним інструментом соціального прогресу та одним з основних чинників інноваційного розвитку IT індустрії. У цьому контексті виникають нові інформаційні системи, які сприяють ефективному обміну інформацією та комунікації в цифровому світі.

Одним із цікавих проектів, що ґрунтується на вищезазначених принципах, є наш проект Perry - сайт, який є аналогом Amazon. Perry надає користувачам можливість відчути привабливість і потужність популярного інтернет-магазину, пропонуючи широкий асортимент товарів і послуг, забезпечуючи зручність і комфорт онлайн-шопінгу.

У цій роботі ми розглянемо проект Perry, його реалізацію з точки зору DevOps. Детально опишемо загальну характеристику системи, проаналізуємо її технічне завдання та дослідження в галузі інформаційної безпеки, розгортання та оновлення.

ТЕХНІЧНЕ ЗАВДАННЯ ДО ПРОЕКТУ

Загальний опис
Програмне забезпечення Perry призначене для поширення інформації шляхом здійснення онлайн-покупок і взаємодії з іншими користувачами.

Призначення
Надання можливості зручного та ефективного шопінгу в мережі, подібно до Amazon. Perry забезпечує збереження даних, конфіденційність користувачів, а також безпеку та захист особистої інформації.Вимоги до функціоналу:

Реєстрація та авторизація

Реєстрація
Підтвердження особистості
Авторизація

Особистий акаунт користувача

Особистий профіль
Зміна особистих даних
Зміна паролю
Огляд товару та купівля
Відвідування товару який цікавить
Швидка покупка і коментар до товару

Архітектура:
S3 bucket - Використовується для хранения картинок із сайту, всі фото товару і користувачів хранятся саме там.
Сервер бази даних SQL - серверне програмне забезпечення, що забезпечує збереження та обробку поточних даних, та інформацію про товар і користувачів.
Сервер GitLab – Сервер GitLab нам потрібен для налаштування CI/CD, який в рази прискорить процес розробки і розгортання сайту в маси. Так само GitLab є локальним і підключення до нього тільки через VPN WireGuard, що робить наш проект максимально секюрним. Працює з контейнера Докер.
Сервер Prod - на цьому сервері підніматимуться і запускатимуться контейнери з нашим сайтом, як-от backend, frontend, програми для збору інформації та стабільності роботи проекту. Так само працює з контейнерів Докер.

РОЗДІЛ 1

ЗАГАЛЬНА ХАРАКТЕРИСТИКА СИСТЕМИ

Архітектура платформи складається з таких частин:

«Frontend»
«Design»
«Backend»

«DevOps»

«FRONTEND»

Головна мета фронтенду Perry полягає у створенні зручного та інтуїтивно зрозумілого інтерфейсу для користувачів. Цей інтерфейс має бути добре організованим, з урахуванням принципів UI/UX дизайну, щоб забезпечити легку навігацію та взаємодію з функціями платформи. Важливо, щоб інтерфейс був адаптивним до різних типів пристроїв.

«DESIGN»

Секція "Дизайн" складається з колекції моделей та детальних описів, які відображають поведінку користувачів у системі. Ці моделі охоплюють різні типи інтерфейсів взаємодії між користувачами та інформаційною системою, а також показують шляхи переходів між різними інтерфейсами. При створенні цих інтерфейсів ми дотримувалися принципів розумної навігації і використовували передові підходи в галузі UI/UX дизайну для створення зручного та естетичного користувацького досвіду. Застосування принципів UI/UX дизайну дозволило ретельно розрахувати всі елементи інтерфейсів, забезпечуючи зрозумілу та зручну взаємодію з системою. Кольорові рішення були уважно підібрані для створення гармонійного та привабливого зовнішнього вигляду. Особлива увага була приділена ергономіці мобільних пристроїв, щоб забезпечити легкий доступ та зручне використання інтерфейсів на різних пристроях.

«BACKEND»

Частина «Backend» Perry є невід'ємною складовою нашої платформи і відповідає за обробку та збереження даних, які створюються та модифікуються в процесі роботи системи. Вона забезпечує ефективне зберігання та організацію інформації, а також надає програмний інтерфейс (API), який забезпечує доступ до основних функцій платформи. Backend Perry гарантує стабільну роботу системи, обробку користувацьких запитів і підтримку безперебійного обміну даними між клієнтською частиною та сервером.

«DEVOPS»

«DevOps» відіграє важливу роль у розробці проекту. Його функції включають реалізацію постійної інтеграції та безперервного розгортання елементів додатка на платформі GitLab. Ми використовуємо приватні сервери нашого університету з підключенням VPN WireGuard для створення інфраструктури, що дає нам змогу забезпечити надійну та масштабовану роботу проєкту.

Ми також використовуємо сервіси Prometheus і Grafana для створення моніторингу системи серверів, як-от GitLab і сервер Prod, на який відбувається розгортання проєкту, що дає нам змогу слідкувати за споживанням ресурсів, аналізувати дані та вчасно виявляти будь-які проблеми або незвичні відхилення в роботі системи.

РОЗДІЛ 2

РЕАЛІЗАЦІЯ DEVOPS ЧАСТИНИ

Ми розпочали з визначення меж проекту в контексті застосування практики DevOps. Під час обговорення з командою ми визначили, які компоненти і процеси повинні бути включені до області практики DevOps, щоб досягти найкращих результатів:

Монолітний додаток
Хмарні технології
Постійна інтеграція та безперервне розгортання (CI/CD)
Моніторинг системи
Безпека та захист даних

Ми обирали, яким хмарним сервісом користуватися. Наразі на ринку присутні 3 найрозвиненіші компанії, тому ми обирали з них. S3 і MySQL Workbench були на ринку довше, що дало їм змогу набрати більшу популярність і розгорнути центри обробки даних по всьому світу, забезпечуючи ширшу географічну доступність.

Сервіси які будуть використовуватися:

GitLab - це веб-сервіс для керування проектами та контролю версій на основі

системи Git. Він надає інструменти для спільної роботи розробників над програмним забезпеченням, включаючи управління задачами, зберігання коду, збір та тестування програм, а також управління версіями коду. GitLab має ряд функцій, таких як інтеграція з CI/CD.

Docker - Це популярний інструмент для розгортання, управління та масштабування додатків у контейнерних середовищах. Він дозволяє упаковувати додатки разом з усіма їхніми залежностями у стандартизовані та ізольовані контейнери. Кожен контейнер є автономним та легковагим середовищем, що містить все необхідне для роботи програми, включаючи код, бібліотеки, залежності та конфігурацію. Docker надає можливість швидко та легко розгортати додатки, запускаючи контейнери з однаковим середовищем на різних серверах або хмарних платформах.

S3 - це веб-сервіс, який забезпечує масштабування та надійне зберігання даних у вигляді об'єктів у веб-хмарі.

WireGuard - це сучасний VPN-протокол, відомий своєю простотою, швидкістю та безпекою.

Постійна інтеграція та безперервне розгортання (CI/CD)
Для реалізації цього етапу було обрано сервіс GitLab. Головною причиною стало те, що раніше ми вже обрали GitLab, як систему контроля версій. Також було цікаво обрати нову систему для вивчення. Все запущено в Docker контейнерах для зручності та впорядкування процесу роботи.

Було створено два головних сценария для CI/CD:

- build - для збору проекту в контейнери
- deploy - переадресовує контейнер на prod сервер і автоматичний починав свою роботу

Структура пайплайну для складання та deploy проекту на сервер Прод

- build

```
stages:
  - publish
  - deploy

variables:
  TAG_LATEST: $CI_REGISTRY_IMAGE:$CI_COMMIT_REF_SLUG-latest
  TAG_COMMIT:
$CI_REGISTRY_IMAGE:$CI_COMMIT_REF_SLUG-$CI_COMMIT_SHORT_SHA

publish:
  image: docker:latest
  stage: publish
  services:
    - docker:dind
  script:
    - pwd
    - docker build -t $TAG_COMMIT -t $TAG_LATEST .
    - docker login -u $CI_REGISTRY_USER -p $CI_REGISTRY_PASSWORD $CI_REGISTRY
    - docker push $TAG_COMMIT
    - docker push $TAG_LATEST
  tags:
    - runner
```

- deploy

```
deploy:
  stage: deploy
  before_script:
```

```
  # Setup SSH deploy keys
  - 'which ssh-agent || ( apt-get install -y -qq openssh-client git )'
  - eval $(ssh-agent -s)
  - mkdir -p ~/.ssh
  - ssh-add <(echo "$ID_RSA")
  - echo -e "Host *\n\tStrictHostKeyChecking no\n\n" > ~/.ssh/config
  script:
    - ssh $DEPLOY_USER@$SERVER_IP "docker login -u $CI_REGISTRY_USER -p $CI_REGISTRY_PASSWORD $CI_REGISTRY && docker pull $TAG_COMMIT && docker container rm -f my-app || true && docker run -d -p 8081:8081 -p 8080:8080 --name my-app $TAG_COMMIT"
```

Також, ми використовуємо технологію Docker для нашого проекту. Це дозволяє створювати та швидко оновлювати нові версії додатку на віддаленому сервері. Всі нові образи зберігаються на GitLab.

Конфігурація Docker-compose для GitLab и runner для него:

Моніторинг системи

Для моніторингу, ми використовуємо Grafana та Prometheus. Ця технологія дозволяє слідкувати за показниками наших систем.

Сервіси Prometheus збирають метрики серверів, що дозволяє відображати їх на панелі приладів.

Усі конфіденційні дані для бази даних та авторизації, що використовуються в GitLab, зберігаються в розділі "CI/CD Variables".

Конфігурація проксі-nginx:

РОЗДІЛ 3
ОЦІНКА

На основі створеного проекту можна стверджувати, що він реалізований відповідно до ключових принципів та практик DevOps, що забезпечує ефективність та надійність процесів розробки та розгортання програмного забезпечення.

Також, створення неперервної інтеграції та неперервного розгортання (CI/CD) для автоматизації тестування, збирання та розгортання програмного забезпечення. Це дозволяє забезпечити швидкий і безперервний процес розгортання, зменшуючи час та зусилля, необхідні для впровадження змін.

Крім того, використання моніторингу системи та реагування на події допомагає забезпечити стабільну роботу проекту та вчасно виявляти проблеми. Описані вище розділи також зазначають важливість збереження конфіденційних даних та застосування заходів безпеки, таких як використання проксі-серверів та шифрування SSL-з'єднань.

Узагалі, застосування цих практик сприяє покращенню продуктивності, якості та безпеки процесу розробки та розгортання програмного забезпечення. Проект є прикладом не поганої імплементації DevOps методології та може служити початковим прикладом для подальших проектів у сфері розробки програмного забезпечення.

ВИСНОВКИ

Ми розглянули DevOps-практику для розробки та розгортання додатку. Обрали систему контролю версій GitLab і створили два репозиторії для поділу проекту на фронтенд і бекенд.

Для реалізації неперервної інтеграції та неперервного розгортання (CI/CD) ми використовували пайплайн едитор. Було створено два сценарії: build та deploy, які автоматизували процеси збирання, тестування та розгортання додатку. Ці сценарії дозволяють перевіряти код, створювати Docker образи.

Для моніторингу системи ми використовували Prometheus Grafana, що дозволяє відстежувати показники системи і надсилати повідомлення про стан системи. Для збереження конфіденційних даних, таких як дані авторизації та доступу до бази даних, ми використовували ми хранимо на нашому сервері MySQL.

В цілому, ми успішно впровадили DevOps практики для розробки та розгортання нашого додатку. Використання GitLab, Docker, MySQL Grafana, AWS та інших інструментів дозволило нам автоматизувати процеси, підвищити ефективність розробки та забезпечити високу якість та безпеку додатку.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

№	Ресурс
1.	digitalocean Розробник: digitalocean, Inc. https://digitalocean.com/
2.	docs.GitLab Розробник: GitLab, Inc. https://docs.gitlab.com/
3.	Docker Розробник: Docker, Inc. https://docs.docker.com/
4.	Amazon Web Services Розробник: AWS, Inc. https://docs.aws.amazon.com/

5.	Grafana Розробник: Grafana, Inc. https://grafana.com/
6.	sql workbench Розробник: sql workbench, Inc. https://www.mysql.com/
7.	VPN WireGuard Розробник: WireGuard, Inc. https://www.wireguard.com/