Engineer

DevOps

44 досягнення

Поєднайте частини системи

Роботи за темою «DevOps» показують одну частину дієвої системи. Ми можемо допомогти пов'язати її з рештою вашої діяльності. Моніторинг і сповіщення, які попереджають, поки збій ще малий, і цифри, що показують, як часто так і залишається.

Переглянути наші послуги Зв'язатися з нами →

Знайдіть схоже завдання

Відкрийте приклад, схожий на вашу ситуацію. Огляд пояснює рішення, що привели до результату, а не лише називає технологію. Запустила перший дашборд спостережуваності — перший огляд власних систем компанії

Запустила перший дашборд спостережуваності Зв'язатися з нами →

Запустила перший дашборд спостережуваності

Розробила та запустила перший у компанії дашборд спостережуваності, що забезпечував аналітику продуктивності системи в реальному часі та візуалізацію даних на великому офісному телевізорі.

Підтримувала 10 серверів PostgreSQL і MS SQL

Спроєктувала, розгорнула та підтримувала 10 серверів PostgreSQL і MS SQL на Ubuntu Linux VPS, забезпечивши оптимальну продуктивність і надійність серверів.

Написала 1 000 правил RBAC для Linux-серверів

Посилила безпеку даних, впровадивши 1 000 правил RBAC для розробників, інстансів застосунків, PostgreSQL, MS SQL та інших Linux-серверів, запобігши несанкціонованому доступу; задокументувала їх як автоматизацію в Ansible.

Автоматизувала розгортання GIS SaaS через CI/CD

Автоматизувала розгортання GIS SaaS-застосунку, обробку даних та систему звітності за допомогою GitHub Actions CI/CD, Python, Bash та SQL.

Автоматизувала розгортання 20 GIS-пайплайнів

Автоматизувала розгортання 20 пайплайнів GIS-даних та ETL-процесів даних застосунку, оптимізувавши автоматизацію інфраструктури та звітність.

Автоматизувала 100 критичних резервних копій

Автоматизувала 100 критично важливих резервних копій даних за допомогою Barman, Google Cloud, Bash та Python, забезпечивши цілісність даних в усіх базах.

Вела 20 застосунків Docker у Google Cloud і AWS

Розгорнула та підтримувала 20 контейнеризованих застосунків Docker, усуваючи несправності за допомогою Podman і Kubernetes, а також керувала застосунками на R у Google Cloud та AWS.

Керувала 30 інстансами Ubuntu Linux VPS

Керувала 30 інстансами Ubuntu Linux VPS, впровадивши стратегії аварійного відновлення та забезпечивши оптимальні конфігурації мережі.

Знизила операційні ризики з Grafana і Prometheus

Знизила операційні ризики, впровадивши дашборд моніторингу на базі Grafana та Prometheus, підвищивши надійність системи.

Збільшила дохід GIS-застосунку в 10 разів

Очолила розробку програмного забезпечення GIS-картографічного застосунку, збільшивши дохід у 10 разів та позиціонувавши продукт як основний актив даних.

Автоматизувала SSL/TLS для 100 застосунків Docker

Автоматизувала створення сертифікатів SSL/TLS для 100 застосунків Docker, забезпечивши безпечні з'єднання на хостах Ubuntu Linux.

Оптимізувала CI/CD, заощадивши 4 000 годин

Оптимізувала процеси CI/CD, заощадивши 4 000 годин завдяки впровадженню автоматизації в пайплайнах розробки програмного забезпечення.

Заощадила 4 000 годин з CI/CD на GitHub і GitLab

Оптимізувала процеси аналізу даних та розробки програмного забезпечення, заощадивши 4 000 годин завдяки впровадженню практик CI/CD на базі GitHub, GitLab, Bash та Python.

Спрощена інфраструктура поза AWS

Перебудувала інфраструктуру міжнародного клієнта, вилучивши невикористані служби, скоригувавши потужність і перенісши навантаження з AWS. За свідченням засновниці, витрати на хмару скоротилися у 10 разів без втрати продуктивності.

Згенерувала контракт API із самої бази даних

Згенерувала контракт API з бази даних назовні — OpenAPI, типізований TypeScript-клієнт на 44 076 рядків, 61 mock-обробник та обмеження, які застосовує UI, — з перевіркою на кожній ланці, що падає в разі розбіжностей.

Розгорнула інфраструктуру Azure як код

Розгорнула інфраструктуру Azure як код за допомогою Bicep — Container Apps, PostgreSQL Flexible Server, Front Door/WAF та мережу — у середовищах development, staging і production.

Побудувала пайплайни CI/CD на GitHub Actions

Побудувала пайплайни CI/CD на GitHub Actions з distroless-образом продакшн-фронтенду та просуванням релізів між кількома середовищами.

Написала 578 цілей автоматизації go-task

Написала 578 цілей автоматизації go-task, що охоплюють нативний, Docker- та HTTPS-режими розробки, лінтинг, тестування, базу даних та розгортання.

Відповідала за розгортання платформи в Azure

Відповідала за наскрізне розгортання платформи в Azure, керуючи релізами в середовищах development, staging і production.

Налаштувала резервні копії та відновлення як код

Налаштувала зберігання резервних копій бази даних у вигляді інфраструктури як коду (infrastructure-as-code), провела аудит готовності до відновлення й задокументувала процедуру відновлення — чітко назвавши наявні прогалини, а не залишивши їх на час інциденту.

Запровадила нуль попереджень у шести мовах

Встановила стандарт якості без жодних попереджень для шести мов — Go, TypeScript, SQL, Python, Shell і Markdown — дотримання якого забезпечували pre-commit hooks.

Заклала засадничі технічні рішення платформи

Заклала засадничі рішення платформи в тижні після відкриття репозиторію в листопаді 2025 року — поділ на рівні, доступ до даних передусім через базу даних та планку без жодних попереджень — і вони досі чинні через дев'ять місяців.

Побудувала всю базову інфраструктуру з нуля

Як одна з перших співробітниць, спроєктувала та побудувала всю базову інфраструктуру та супутні процеси з нуля для SaaS-стартапу в зеленій енергетиці, заклавши фундамент для швидкого зростання.

Автоматизувала командну співпрацю

Автоматизувала командну співпрацю, керування паролями, управління завданнями та часом, а також створила напівавтоматичну систему демонстрації проєктів, підвищивши продуктивність команди.

Побудувала тести для Go, SQL і браузера

Побудувала багаторівневий набір автоматизованих тестів — 981 тест Go, 543 фронтендні та браузерні специфікації, 494 поведінкові тести SQL — з мутаційним тестуванням, property-based тестами та обов'язковою перевіркою доступності.

Побудувала механізм із 268 перевірок репозиторію

Побудувала механізм перевірок репозиторію — 268 зареєстрованих перевірок коміту, 277 правил лінтингу та 15 власних правил ESLint — а також 146 тестів самих перевірок, щоб стандарт тримала збірка, а не код-рев'ю.

Побудувала власні моніторинг помилок і трасування

Побудувала власний моніторинг помилок і трасування OpenTelemetry, а не купила готові — очищення payload, виявлення сплесків і регресій, символікацію та синтетичний heartbeat — за 11 операторськими поданнями.

Тримала схему узгодженою впродовж 1 022 міграцій

Тримала схему узгодженою впродовж 1 022 міграцій за допомогою перевірки CI, яка збирає базу даних обома шляхами — чиста інсталяція та інсталяція плюс усі міграції — і падає, коли вони розходяться.

Запровадила безперервне сканування безпеки

Запровадила безперервну програму безпеки — сканування коду, DAST, перевірки залежностей і вразливостей, генерацію SBOM, пошук секретів та actions, закріплені за SHA, — поряд із 21 письмовим аудитом безпеки.

Інфраструктура, описана в репозиторії

Побудувала власну інфраструктуру компанії у вигляді 19 плейбуків Ansible і 34 ролей обсягом 12 065 рядків YAML, які приводять робочий хост до задекларованого стану, причому кожен play ідемпотентний.

Домоглася, щоб check mode в Ansible казав правду

Домоглася, щоб check mode показував правду на всій платформі, після того як знайшла шість перевірок, що ухвалювали рішення за значенням, якого хост ніколи не повертав: модуль command в Ansible у режимі --check звітує про успіх, узагалі не виконуючи команду.

Додала секундну попередню перевірку секретів

Додала попередній плей (preflight), який приблизно за секунду проганяє той самий код, що й converge, на локальних секретах оператора, — після того як продакшн-запуск обірвався на дев'ятому завданні, застосувавшись лише наполовину, а налаштування swap уже було записано на робочий хост.

Ваш код на вашому власному сервері

Розгорнула власний git-сервер компанії на Soft Serve, приватний за замовчуванням і без вебпанелі, з портом SSH, прив'язаним до loopback за проміжним хостом (jump host), і зробила цільову сторінку перед ним артефактом збірки основного сайту, а не копією, яку підтримують вручну.

Запустила контейнери на Podman і Quadlet

Розгорнула контейнерну платформу на Podman і Quadlet під systemd, а не на Docker, бо типове публікування портів Docker у мережі bridge може обходити правила UFW, — а для розгортань виділила непривілейованого користувача з однією фіксованою командою замість root.

Побудувала перевірку комітів із 22 лінтерів

Побудувала перевірку перед комітом із 22 лінтерів, кожен з яких описується одним рядком, і ще п'ятьох, що заслуговують окремого абзацу, без рівня попереджень і без дозволу на вбудоване вимкнення правил, яка охоплює HTML, CSS, JavaScript, Python, YAML, Markdown, shell, посилання, орфографію, секрети й типографіку.

Страховка, що більше не забирає пів години

Скоротила браузерну перевірку якості сайту з 1 636 секунд до 615, запускаючи перевірки від найдовших до найкоротших через пул воркерів із максимум чотирма одночасними перевірками, після того як виміряла, що алфавітний порядок коштував 320 секунд проти 224.

Перевірила лінтером 10 242 рядки перевірки якості

Охопила форматувальником і лінтером 10 242 рядки JavaScript у перевірках якості, з'ясувавши, що це найбільший обсяг коду в репозиторії і єдиний, який ніщо не перевіряло, — виправила 13 зауважень і не вимкнула жодного.

Забезпечила генератору 981 тест і 92% покриття

Покрила генератор 981 тестовим випадком із мінімальним порогом покриття гілок 92%, де попередження вважаються помилками, і перевірила ідемпотентність, двічі запустивши весь пайплайн збірки з порожнього файлу та вимагаючи, щоб другий прохід нічого не змінював.

Увімкнула всі правила лінтера Python як помилки

Увімкнула всі правила лінтера Python як помилки й довела 1 815 зауважень до нуля; кожен із небагатьох винятків має письмове обґрунтування, а два з них підкріплені перевіркою, а не коментарем.

Виявила розбіжність між хуками й гейтом якості

Виявила, що хуки комітів і гейт якості виконують різні перевірки, хоча документ обіцяв, що вони однакові, порівнявши два переліки в тесті, — п'ять перевірок, які запускалися лише вручну, були саме тими, що читали текст CV.

Перевірила CI-хук git-сервера, знайшла 2 дефекти

Відрепетирувала CI-хук на боці git-сервера й знайшла два дефекти, яких не виявити читанням файлу: запасний варіант, що передавав на вхід хука аргумент, який неможливо розв'язати, і власну змінну середовища git, яка потрапляла слідом за гейтом у checkout і робила 19 тестів червоними.

Пришвидшила набір тестів до 8,9 секунди

Скоротила набір тестів від шести тестів, що перевищували ліміт у шістдесят секунд, до 135 успішних за 8,9 секунди, профілюючи головний потік і прибравши два виклики, у яких він перебував 3 989 із 4 017 вибірок.

Знайте ліцензію всього, що випускаєте

Запровадила перевірку сумісності ліцензій 644 крейтів Rust під час кожної збірки й довела, що вона спрацьовує, переписавши ліцензію одного крейта та змістивши зафіксовану ревізію ядра без перегенерації.

Привела 4 репозиторії до одного стандарту історії

Привела чотири репозиторії до одного стандарту історії — conventional-коміти, без емодзі, без рядків атрибуції, з контролем через хук повідомлень комітів, — поряд із 48 документами інструкцій, що визначають, як виконується робота.

Спрямуйте інженерні зусилля

Опишіть шлях розгортання, експлуатаційні ризики та чергову команду. Ми можемо визначити зміну з перевірками, шляхом відновлення та чіткою відповідальністю.

Дивитися, скільки це коштує Зв'язатися з нами →

Розгляньте тему у своєму контексті

Розкажіть, як «DevOps» стосується вашої системи. Ми допоможемо обрати відповідну послугу та визначити обсяг роботи.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://engineer.company/uk/categories/devops/