Engineer

Автоматизація та CI/CD

82 досягнення

Поєднайте частини системи

Роботи за темою «Автоматизація та CI/CD» показують одну частину дієвої системи. Ми можемо допомогти пов'язати її з рештою вашої діяльності. Цифри з'являються там, де потрібні, за розкладом, і ніхто не копіює таблицю наприкінці місяця.

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. Один інфраструктурний план для 14 підрозділів — 14 дослідницьких підрозділів, охоплених спільним інфраструктурним планом

Один інфраструктурний план для 14 підрозділів Зв'язатися з нами →

30 днів даних про електроенергію за 180 мс

Розробила варіанти погодинної агрегації споживання електроенергії на Python, SQL і Bash + Jq; під час вимірювання Python обробив один 30-денний набір даних JSONL за 180 мс.

Усунула 1 000 проблем у географічних даних

Усунула 1 000 проблем у географічних даних та часових рядах, використовуючи GDAL, ArcGIS, PostGIS, Mapbox, QGIS, SQL (PL/pgSQL, Transact-SQL), Bash, забезпечивши високоякісну обробку великих даних.

Спроєктувала й підтримувала 6 пайплайнів ETL/ELT

Спроєктувала, реалізувала та адмініструвала 6 пайплайнів ETL/ELT, використовуючи Google BigQuery, MSSQL, PostgreSQL, shell-скрипти, PL/pgSQL та Transact-SQL, інтегрувавши дані для ефективної обробки через Python API.

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 доступом у M365 і SSO

Запобігла порушенням безпеки, очоливши ініціативи з керування доступом, використовуючи M365, 1Password, Red Hat SSO та OKTA SSO.

Знизила операційні ризики з Grafana і Prometheus

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

Покращила комунікацію та співпрацю в команді

Покращила комунікацію та співпрацю в команді, впровадивши Slack, Mattermost, 1Password та Jira, заощадивши 8 000 людино-годин.

Реорганізувала процеси, заощадивши 8 000 годин

Реорганізувала внутрішні процеси, заощадивши 8 000 годин завдяки вдосконаленню архітектури програмного забезпечення, систем та ефективності планування.

Адмініструвала мережеву інфраструктуру

Адмініструвала мережеву інфраструктуру для понад 1 000 серверів, забезпечивши оптимальне розгортання систем, безпеку та усунення несправностей.

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.

Розробила систему аналітичної звітності

Розробила систему аналітичної звітності, збільшивши квартальний дохід від програмного забезпечення на 400% завдяки PDF-звітам на базі Python.

Автоматизувала обробку даних на Python і PL/pgSQL

Автоматизувала завдання обробки даних за допомогою shell-скриптів, PL/pgSQL, Python та Transact-SQL, підвищивши продуктивність і ефективність.

Заощадила 150 годин на місяць у 30 проєктах

Підвищила ефективність 30 проєктів: оптимізувала робочі процеси й управління ресурсами та заощадила 150 годин на місяць.

Розробила 600 пайплайнів ETL/ELT на PL/pgSQL

Розробила 600 пайплайнів ETL/ELT на основі PL/pgSQL для оптимізації складних робочих процесів обробки даних у кількох середовищах розробки та продакшну PostgreSQL.

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.

Створила англо-данську систему i18n

Створила систему інтернаціоналізації для англійської та данської мов, що містить 12 027 повідомлень на локаль у 458 файлах просторів імен, зі словником, дотримання якого забезпечував лінтер, та бюджетом у 400 рядків на файл.

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.

Спроєктувала систему обліку часу та звітності

Спроєктувала систему управління часом та звітності, яка роками залишалася в промисловій експлуатації без суттєвих змін.

Автоматизувала командну співпрацю

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

Впровадила корпоративний менеджер паролів

Інтегрувала загальнокорпоративну систему керування паролями, посиливши безпеку та оптимізувавши керування доступом.

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.

Перенесла повільні операції в чергу завдань River

Перенесла повільну роботу зі шляху запиту в чергу завдань River — 15 модулів воркерів, 8 запланованих задач та 20 завдань pg_cron — щоб запит повертався, поки робота за ним триває.

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.

Посилила SSH до 24 перевірених директив

Посилила захист SSH: 24 директиви, кожну з яких перевіряє assert, і триетапна валідація — файл-кандидат, зібрана конфігурація, а потім зворотне зчитування самим демоном, — після того як таке зчитування виявило, що запущений сервер мовчки перевизначав дві з двадцяти чотирьох.

Довела, що блокування вторгнень працює щоразу

Довела працездатність усього ланцюжка блокування вторгнень під час кожного запуску посилення захисту: блокувала зарезервовану тестову адресу, зчитувала створене правило ядра й знімала блокування в гарантованому блоці очищення, тож jail, який перестав працювати, обриває запуск помилкою, а не звітує про справність.

Перевірила правила фаєрвола за позицією

Перевірила правила фаєрвола за позицією, а не лише за наявністю, зчитуючи пронумерований перелік правил і чинний ланцюжок фільтрації пакетів, бо наявне правило ще не означає, що до нього доходить хоч один пакет.

Автоматизація резервування з явним способом відновлення

Реалізувала автоматизацію резервування restic з політикою зберігання, перевіркою цілісності та щомісячним тестом відновлення; задокументувала активацію сховища й перевірку справжнього відновлення як окремі операційні кроки.

Побудувала моніторинг хоста з dead man's switch

Побудувала моніторинг за принципом dead man's switch, який надсилає пінг, лише поки пам'ять і диск у нормі, тож хост у погіршеному стані здіймає тривогу своїм мовчанням, — і виявила шість імен змінних зі словом «free» там, де перевірка правильно вимірювала «available»: на машині з 464 МБ ці значення різняться на порядок.

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.

Описала зону DNS із 20 записів як код

Налаштувала декларативне узгодження зони DNS із 20 записів з API Cloudflare, з окремими точками входу для аудиту та експорту в BIND, а побудувавши проксі CDN, знову вимкнула його з міркувань приватності.

Посилила ізоляцію systemd для кожного юніта

Знизила оцінку відкритості (exposure) у пісочниці systemd для кожного юніта, який встановлює платформа: служба dead man's switch — з 9,6 UNSAFE до 1,5, відкритий в інтернет git-сервер — з 8,3 EXPOSED до 1,5, — і додала перевірку парсером під час converge, після того як знайшла у трьох шаблонах юнітів директиву з орфографічною помилкою, яку systemd мовчки ігнорував.

Побудувала 7 ролей звітності лише для читання

Побудувала сім ролей звітності лише для читання, які описують робочий хост у Markdown — факти, доступ, git, метрики, трафік, безпека та інвентар провайдера, — за правилом: не виводиться жодне число, якого запуск не виміряв.

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.

Підготувала другий сервер із коду

Автоматизувала підготовку (provisioning) другого сервера в другого хмарного провайдера, створюючи фаєрвол раніше за машину, щоб вона від самого початку була за ним, а фаєрволи обох провайдерів налаштовуються безпосередньо через їхні REST API, без сторонньої колекції.

Записала правило меж після витоку облікових даних

Записала в репозиторій правило про межі його вмісту після того, як під час реструктуризації туди потрапили інвентар, дозволи фаєрвола та тексти іншої компанії, — і залишила відокремлений у карантин залишок під наглядом сканера секретів, а не виключила його.

Розділила всі операційні секрети за хостами

Розділила чотири операційні секрети так, щоб кожен хост мав власні, з'ясувавши, що два хости зі спільним dead man's switch сповіщають про збої рідше, ніж два окремі перемикачі, а не частіше, і що спільна парольна фраза резервних копій може відкрити доступ до копій обох хостів навіть в окремих репозиторіях, якщо хтось отримає її та зможе прочитати зашифровані дані й файли ключів обох репозиторіїв.

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.

Замінила 27 розмірів шрифту шкалою з 6 кроків

Замінила 27 фактичних розмірів шрифту, сусідні значення яких різнилися лише на 0,6%, шестиступеневою шкалою Major Third і написала лінтер, який відхиляє двадцять восьмий.

Вади доступності, яких не бачить ваша автоматична перевірка

Знайшла й виправила 11 дефектів доступності, які браузер вважав справними: вісім посилань у підвалі, що лишалися в порядку табуляції попри вимкнену взаємодію через pointer-events, часову шкалу прокрутки (scroll timeline), яка обрізала колофон на трьох сторінках, і рушій правил, що виконував лише 70 зі своїх 105 правил.

Обмежила палітру сайту 28 кольорами

Обмежила систему кольорів 28 задокументованими кольорами й написала лінтер, який відхиляє значення, що застосоване, але не задокументоване, задокументоване, але не застосоване, хибно виміряне, переписане як літерал або розташоване в межах перцептивної відстані 0,02 від уже наявного.

Знайшла, що світлій темі бракує цілого шару

Виявила, що у світлій темі від самого її створення бракувало повносторінкового шару накладання, додавши перевірку, яка вимагає, щоб обидві теми малювали кожну багатошарову поверхню однаковою кількістю шарів.

Віддзеркалила сайт у Gemini та Gopher

Побудувала вихідні формати Gemini та Gopher з того самого джерела сайту; запис досягнення фіксує по 777 документів на тому етапі та нуль змінених байтів HTML від додавання форматів.

Замінила 963 блоки метаданих одним графом

Замінила 963 анонімні блоки структурованих даних, які 1 671 раз повторно описували компанію, одним зв'язаним графом із 16 типів, створеним на основі стабільних ідентифікаторів походження.

Скоротила пакунок стилів із 87 КБ до 31 КБ

Скоротила пакунок стилів із 87 КБ до 31 КБ, вийнявши з нього шрифт у base64, і позбулася 756 КБ у 22 файлах, які публікувалися під час кожного розгортання і на які ніщо не посилалося.

Виправила мапу сайту з однією позначкою часу

Виправила мапу сайту, де 172 з 176 URL мали одну позначку часу зміни, взявши дату з історії git, після того як з'ясувала, що експорт переписує кожен файл під час кожного запуску.

Підключила до документації окремий лінтер

Підключила до документації окремий лінтер — ліміт рядків для кожного файлу, який можна лише знижувати, вимогу індексу та перевірку справжніх шляхів, — який знайшов шість із дев'яти планів, позначених виконаними, попри невідмічені пункти, і чотирнадцять завдань перевірки перед комітом, не згаданих у жодному документі.

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.

Побудувала генератор резюме на основі бази даних

Побудувала на Python генератор CV, рекомендацій, портфоліо та супровідних листів на основі бази даних — 41 модуль, 10 580 рядків, — що відтворює шість форматів виводу з одного джерела SQLite на 23 таблиці, зібраного ідемпотентним пайплайном із 19 кроків.

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.

PDF/UA-1: пройдено 106 зі 106 перевірок профілю

Зафіксувала перевірку PDF/UA-1 для 9 документів: успішно пройдено 106 зі 106 налаштованих перевірок профілю доступності, тоді як архівний варіант не пройшов одне зі 146 правил.

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.

Написала тести для самих перевірок

Написала тести для самих перевірок, з'ясувавши, що перевірка, яка отримує лише чисті вхідні дані, одного дня звітуватиме про чистоту, бо нічого не прочитала: підклала помилку правопису, щоб підтвердити, що перевірка орфографії її знаходить, і взяла діапазон ідентифікаторів із бази даних, а не з числа в тесті.

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.

Побудувала міжмовну перевірку чисел

Побудувала міжмовну перевірку контенту, яка падає, коли переклад губить число, вказане англійською, або коли мова вживає не притаманну їй нотацію, — і знайшла два данські описи без метрики та шістнадцять українських фрагментів, що цитували на англійський лад.

Написала парсер, що перевіряє кожен виклик FFI

Написала парсер, який читає справжній C-заголовок на 7 308 рядків і перевіряє кожне місце виклику, кожну константу переліку й те, що кожен клас-власник вказівника є final, після того як через написаний вручну заголовок-заглушку виклики трьох вилучених функцій компілювалися, лінкувалися й аварійно завершувалися.

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.

Назвала всі 17 піктограм для зчитувачів екрана

Дала назви для зчитувачів екрана всім елементам керування, що складаються лише з піктограми, після того як виявила, що кнопка надсилання оголошувалася як «arrow up circle, button», і написала лінтер, який вимагає підпису в межах восьми рядків від піктограми.

Побудувала дизайн-систему macOS із 48 токенів

Побудувала дизайн-систему з 48 токенів на основі власного скляного матеріалу платформи й написала лінтер, який відхиляє магічне число, жорстко заданий розмір шрифту чи анімацію, що ігнорує налаштування зменшеного руху.

Задіяла невикористану половину ядра

Задіяла ту половину ядра обміну повідомленнями, якою застосунок ніколи не користувався, — передавання резервної копії, зникомі повідомлення, редагування та повторне надсилання, підтверджені запрошення, проксі та політику шифрування, — проганяючи кожен тест на справжній бібліотеці без заглушок.

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.

Створила айдентику з 2 малюнків

Створила набори піктограм і фавіконів компанії на основі двох намальованих вручну зображень; кожен похідний ресурс перегенеровується скриптом, а перевірка падає, якщо похідний файл закомічено раніше за його джерело.

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.

Виміряла 13 адрес без трекінгу

Виміряла тринадцять адрес лише за власним журналом доступу вебсервера — без кукі, без трекерів і без сторонніх сервісів — маскуючи мережу клієнта в момент запису рядка та згортаючи результат у постійний архів із 21 денного показника та 20 місячних фасетів.

Змусила шість збирачів звітувати про пропущене

Змусила шість збирачів звітів повідомляти ту частину свого входу, яку вони так і не прочитали, — файли, що лишилися закритими, невстановлені зонди, нерозпізнані рядки журналу, — і підігнала пороги до 116 виміряних зразків, з'ясувавши, що сліпий збір і справний хост дають однакову сторінку.

Спрямуйте інженерні зусилля

Надайте репрезентативний набір даних, запитання до нього та поточне вузьке місце. Узгодимо вимірювання якості, пропускної здатності та відновлення.

Дивитися, скільки це коштує Зв'язатися з нами →

Розгляньте тему у своєму контексті

Розкажіть, як «Автоматизація та CI/CD» 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/automation/