**Ролі в IT проєкті**

   У сучасному світі ІТ-проекти стають все більш важливими для підтримки бізнесу, розвитку нових технологій та впровадження інновацій. У цьому контексті відіграє важливу роль професійна команда, що складається з різних спеціалістів, які вкладають свої знання та навички в процес створення та реалізації ІТ-проектів.

     Роль у ІТ Ми розглянемо різні ролі, що зустрічаються у ІТ-проектах. Від проектних менеджерів, аналітиків та розробників програмного забезпечення до тестувальників, системних адміністраторів та технічних письменників. Кожна роль має свої відповідальності, завдання та вклад у загальний успіх проекту.

    Будучи обізнаними з ролями в ІТ-проектах, ви зможете краще розуміти свої власні цілі та потреби, а також сприяти успіху проектів, в яких ви берете участь.

   Ми розглянемо різні ролі, що зустрічаються у ІТ-проектах. Від проектних менеджерів, аналітиків та розробників програмного забезпечення до тестувальників, системних адміністраторів та технічних письменників. Кожна роль має свої відповідальності, завдання та вклад у загальний успіх проекту.

**Project manager**

    Проектний менеджер є ключовою фігурою в управлінні ІТ-проектами. Він або вона відповідають за планування, керування та контроль за процесами, що стосуються реалізації проекту. Роль проектного менеджера вимагає вміння працювати з різними стейкхолдерами, управляти ресурсами та виконувати проектні цілі згідно з графіком, бюджетом та обсягом робіт.

﻿ **Основні обов'язки проектного менеджера:**

**Планування проекту:** Розробка проектного плану, включаючи визначення мети, обсягу робіт, графіка, ресурсів та ризиків. Встановлення пріоритетів та розподіл завдань між учасниками команди.

**Управління командою**: Керування та мотивація команди, забезпечення взаємодії між учасниками проекту. Розподіл завдань, навчання нових членів команди та вирішення конфліктів.

**Моніторинг та контроль:** Відстеження прогресу проекту, забезпечення дотримання графіка, бюджету та якості робіт. Виявлення й управління ризиками та змінами в проекті.

**Спілкування та звітність**: Взаємодія зі стейкхолдерами, включаючи клієнтів, команду, керівництво та інші відділи організації. Забезпечення регулярного звітності про стан проекту та досягнення поставлених цілей.

**Ресурсне управління**: Планування та координація використання ресурсів, включаючи людські, фінансові та матеріальні ресурси. Забезпечення ефективного використання ресурсів та оптимізація витрат.

**Висновок:**

      Проектний менеджер грає критичну роль у успіху ІТ-проектів. Його або її обов'язки включають планування, керування, контроль, комунікацію та ресурсне управління. Здатність лідерства, комунікації, планування та аналітичних навичок є ключовими для виконання цих обов'язків.

**Product manager**

Продуктовий менеджер є важливою фігурою в розробці та управлінні продуктами або послугами компанії. Він або вона відповідають за стратегічне планування, розробку та впровадження продуктів або функціоналу, які задовольняють потреби клієнтів та досягають бізнес-цілей.

**Основні обов'язки продуктового менеджера:**

**Стратегічне планування**: Визначення стратегії розвитку продукту або функціоналу, включаючи аналіз ринку, вимог клієнтів та конкурентів. Встановлення мети та визначення стратегічних напрямків розвитку продукту.

**Розробка продукту**: Управління життєвим циклом продукту, включаючи збір та аналіз вимог, розробку функціональності, планування релізів та тестування. Співпраця з командою розробки для забезпечення успішної реалізації продукту.

**Управління продуктом**: Визначення пріоритетів функціоналу та вимог, планування релізів та зв'язок зі зацікавленими сторонами.    Забезпечення вирішення проблем продукту та забезпечення відповідності продукту стратегії компанії.

**Маркетинг та комунікація**: Розробка стратегії маркетингу та спілкування з клієнтами, включаючи позиціонування продукту, маркетингові матеріали та комунікацію про продукт на ринку. Взаємодія зі збутовими відділами, маркетинговими командами та клієнтами.

**Аналіз та стратегія**: Моніторинг ринку та трендів, аналіз конкурентів та зворотний зв'язок від клієнтів. Розробка стратегій для забезпечення конкурентних переваг та вдосконалення продукту.

**Developer**

Розробник (developer) є ключовим учасником в процесі створення програмного забезпечення або веб-додатків. Це технічний спеціаліст, який виконує програмування, розробку і тестування програмного коду, щоб реалізувати функціональність продукту.

**Основні обов'язки розробника:**

**Програмування та розробка**: Розробники створюють програмний код, використовуючи відповідні мови програмування, фреймворки та інструменти. Вони реалізовують функціональність продукту згідно з вимогами та дизайном.

**Тестування та відлагодження**: Розробники виконують тестування свого коду, виявляють та виправляють помилки (баги) та відлагоджують програмне забезпечення для забезпечення правильної роботи.

**Участь у розробці архітектури**: Розробники співпрацюють з архітекторами та іншими учасниками команди для розробки технічної архітектури продукту. Вони приймають участь у проектуванні компонентів, модулів та структури програмного забезпечення.

**Співпраця з іншими учасниками команди**: Розробники спілкуються зі замовниками, менеджерами проекту, дизайнерами та іншими членами команди для забезпечення взаєморозуміння та спільної роботи над проектом.

**Підтримка та оновленн**я: Розробники забезпечують підтримку продукту, виявляють та виправляють помилки, а також вносять оновлення та покращення, щоб забезпечити оптимальну роботу програмного забезпечення.

Також треба зауважити що розробники можуть поділятися на FronEnd та BackEnd. Front-end та back-end розробники є двома основними типами технічних спеціалістів у сфері розробки програмного забезпечення або веб-додатків.

**Front-end** розробники відповідають за розробку та побудову користувацького інтерфейсу (UI) та взаємодії з користувачем.

**Back-end** розробники відповідають за розробку серверної частини програмного забезпечення та забезпечення обробки даних та бізнес-логіки.

**Full stack** розробники це поэднання Front-end та Back-end

**Різниця між front-end та back-end розробниками:**

**﻿** **Front-end** розробники фокусуються на створенні інтерфейсу користувача, забезпеченні його взаємодії та візуальному вигляді, тоді як back-end розробники працюють з серверною частиною, обробкою даних та бізнес-логікою.

Front-end розробники використовують мови програмування, такі як HTML, CSS, JavaScript, тоді як back-end розробники використовують мови, такі як Java, Python, PHP, Ruby, C# тощо.

Front-end розробники забезпечують інтерактивність та візуальний вигляд веб-додатків, тоді як back-end розробники відповідають за логіку та обробку даних.

Front-end розробники працюють на боку клієнта (користувача), тоді як back-end розробники працюють на боку сервера.

**Front-end** розробники забезпечують користувацький досвід та інтерфейс, тоді як back-end розробники забезпечують функціональність та обробку даних за лаштунками.

**Висновок:**

  Розробник є ключовим учасником у процесі створення програмного забезпечення або веб-додатків. Його обов'язки включають програмування, розробку, тестування та відлагодження, участь у розробці архітектури, співпрацю з іншими учасниками команди та підтримку продукту. Розробники повинні мати вміння програмування, розуміння архітектури, аналітичні навички, комунікативність та навички командної роботи, а також бути відкритими до навчання та саморозвитку.

**System administrator**

Системний адміністратор є ключовою фігурою в управлінні та підтримці інформаційних систем і комп'ютерних мереж в організації. Вони відповідають за налагодження, конфігурацію, забезпечення безпеки та ефективної роботи систем.

**Основні обов'язки системного адміністратора:**

**Установка та налаштування систем**: Системний адміністратор встановлює та налаштовує операційні системи, сервери, мережеві пристрої та інші компоненти інфраструктури, забезпечуючи їх належну роботу та взаємодію.

**Моніторинг та підтримка**: Системний адміністратор відповідає за постійний моніторинг системи, виявлення та усунення виникнутих проблем, виконання резервного копіювання даних та забезпечення доступності системи для користувачів.

**Забезпечення безпеки**: Системний адміністратор встановлює та конфігурує заходи безпеки, контроль доступу, шифрування даних, антивірусний захист та моніторинг вразливостей.

**Резервне копіювання та відновлення**: Системний адміністратор регулярно виконує резервне копіювання даних та налаштовує процедури відновлення для забезпечення безпечності та надійності інформації.

**Встановлення та підтримка програмного забезпечення**: Системний адміністратор встановлює та підтримує програмне забезпечення, включаючи оновлення, патчі та налаштування робочих середовищ для користувачів.

**Технічна підтримка**: Системний адміністратор відповідає на запити користувачів, допомагає вирішувати технічні проблеми, надає консультації щодо використання системи та навчання нових користувачів.

    Системний адміністратор грає важливу роль в управлінні, підтримці та безпеці інформаційних систем і комп'ютерних мереж. Їх обов'язки включають установку, налаштування, підтримку, забезпечення безпеки та ефективної роботи систем. Системний адміністратор повинен мати глибокі знання технологій, вміння вирішувати проблеми, керувати проектами та мають комунікаційні навички. Вони важливі для забезпечення стабільної та безперебійної роботи інформаційних систем у підприємстві.

**Devops**

  DevOps-інженер є ключовим фахівцем у сфері розробки програмного забезпечення, який поєднує в собі компетенції розробника (developer) та системного адміністратора (system administrator). Він забезпечує інтеграцію, автоматизацію та управління процесами розробки та експлуатації програмного забезпечення.

**Основні обов'язки DevOps-інженера:**

**Конфігурація та управління інфраструктурою**: DevOps-інженер налаштовує та керує хмарними сервісами, віртуалізованими серверами, мережами та іншими компонентами інфраструктури. Він використовує інструменти, такі як Docker, Kubernetes, Terraform, для автоматизації управління інфраструктурою.

**Автоматизація процесів розробки та випуску**: DevOps-інженер використовує інструменти, такі як CI/CD (Continuous Integration/Continuous Deployment), для автоматизації процесів збирання, тестування, розгортання та випуску програмного забезпечення.

**Моніторинг та логуванн**я: DevOps-інженер налаштовує системи моніторингу, щоб відстежувати стан та продуктивність інфраструктури та додатків. Він аналізує дані моніторингу та логів, виявляє проблеми та приймає відповідні заходи.

**Безпека та доступ до даних**: DevOps-інженер виконує заходи забезпечення безпеки, встановлює механізми авторизації та аутентифікації, керує доступом до даних та забезпечує захист від зовнішніх загроз.

**Співпраця з командою**: DevOps-інженер співпрацює з розробниками, QA-інженерами та іншими учасниками команди для забезпечення безперебійної роботи процесів розробки та експлуатації програмного забезпечення.

**Вирішення проблем та підтримка**: DevOps-інженер виявляє та вирішує проблеми, що виникають під час розробки та експлуатації програмного забезпечення. Він надає підтримку та допомогу у вирішенні технічних питань.

   DevOps-інженер виконує ключову роль у побудові та управлінні інфраструктурою програмного забезпечення, автоматизації процесів розробки та експлуатації. Він повинен мати глибокі знання інфраструктури, навички програмування, розуміння мережевих протоколів та добрі аналітичні та комунікаційні навички. Роль DevOps-інженера допомагає покращити ефективність розробки, забезпечує автоматизацію та безперебійну роботу програмного забезпечення, а також сприяє співпраці між розробниками та іншими учасниками команди.

**Хто такий QA ?**

     QA-інженер (Quality Assurance Engineer) є фахівцем з контролю якості програмного забезпечення. Він відповідає за встановлення процесів тестування, виявлення дефектів та забезпечення якості продукту під час розробки.

﻿ **Основні обов'язки QA-інженера:**

**Розробка тестових планів**: QA-інженер планує тестові активності, визначає стратегію тестування, складає тестові плани та визначає критерії успішності.

**Виконання тестування:** QA-інженер проводить різні типи тестування, включаючи функціональне, інтеграційне, системне, регресійне тестування тощо. Він перевіряє, чи відповідає програмне забезпечення вимогам та виявляє дефекти.

**Написання тестових сценаріїв та скриптів:** QA-інженер розробляє тестові сценарії, включаючи вхідні дані, очікувані результати та послідовність дій. Він також може автоматизувати тестування, створюючи скрипти виконання тестів.

**Виявлення та відстеження дефектів**: QA-інженер виявляє дефекти та помилки в програмному забезпеченні, описує їх деталі та створює звіти. Він також відстежує процес виправлення дефектів та перевіряє їх коректність.

**Співпраця з командою розробки**: QA-інженер співпрацює з розробниками, бізнес-аналітиками та іншими учасниками команди для розуміння вимог до програмного забезпечення та вирішення питань, пов'язаних з якістю.

**Аналіз та звітність:** QA-інженер аналізує результати тестування, підготовляє звіти про якість програмного забезпечення, вказує на виявлені проблеми та рекомендує вдосконалення процесів тестування.

Треба намалювати roadMap QA

﻿Зображення, що містить текст, знімок екрана

Автоматично згенерований опис

**QA Trainee (стажер з контролю якості)** - це особа, яка починає свою кар'єру в області тестування програмного забезпечення. QA Trainee є новачком у галузі і виконує свої обов'язки під наглядом більш досвідчених QA-інженерів або керівників QA.

**Обовязки**

**Вивчення основ тестування:** QA Trainee отримує базові знання про методології, процеси тестування та інструменти, що використовуються у QA.

**Виконання тестових завдань:** QA Trainee бере участь у виконанні тестових завдань, включаючи тестування функціональності, виявлення дефектів та регресійне тестування під наглядом більш досвідчених членів команди QA.

**Вивчення інструментів та технологій:** QA Trainee має можливість ознайомитися з різними інструментами тестування, автоматизації та іншими технологіями, які використовуються в QA.

Метою QA Trainee є отримання практичного досвіду у сфері тестування, покращення навичок та знань у цій області. Це перша ступінь у кар'єрному шляху QA-спеціаліста, і з часом QA Trainee може розвиватися в більш високорівневі ролі, такі як QA-інженер або QA-аналіти

**QA Junior (молодший QA-інженер)** - це фахівець з контролю якості програмного забезпечення, який має базові знання та навички в області тестування. QA Junior вже має певний рівень досвіду та розуміння процесів та методологій QA, але ще знаходиться на початковій стадії своєї кар'єри.

**Основні характеристики та обов'язки QA Junior включають:**

**Виконання тестування:** QA Junior виконує тестові завдання, такі як функціональне тестування, тестування регресії та виявлення дефектів. Він проводить тестові сценарії, перевіряє відповідність програмного забезпечення вимогам та стандартам якості.

**Використання тестових інструментів:** QA Junior володіє базовими навичками використання різних інструментів тестування, таких як тестові фреймворки, засоби автоматизації тестування, системи відстеження дефектів тощо.

**Аналіз результатів тестування:** QA Junior виконує аналіз результатів тестування, реєструє та документує виявлені дефекти, стежить за їх виправленням та перевіряє правильність виправлень.

QA Junior є проміжним рівнем між стажером та повноцінним QA-інженером. Це етап, на якому він набуває практичного досвіду, уточнює та розширює свої знання у сфері тестування та готується до подальшого кар'єрного росту. За часом та здобутим досвідом QA Junior може розвиватися у ролі QA-інженера, спеціаліста з автоматизованого тестування або взяти участь у більш складних проектах та викликах у галузі QA.

**QA Middle (середній QA-інженер)** - це фахівець з контролю якості програмного забезпечення, який має достатній рівень досвіду та компетенцій для самостійного виконання завдань з тестування. QA Middle володіє глибшими знаннями у сфері QA і може виконувати більш складні завдання та вести проекти незалежно.

**Основні характеристики та обов'язки QA Middle включають:**

**Планування та стратегія тестування:** QA Middle бере участь у плануванні тестових активностей та визначенні стратегій тестування. Він розробляє тестові плани, визначає обсяг тестування та необхідні ресурси.

**Виконання тестових сценаріїв**: QA Middle проводить різні типи тестування, включаючи функціональне, інтеграційне, навантажувальне та інші. Він перевіряє роботу програмного забезпечення відповідно до вимог та стандартів.

**Автоматизація тестування:** QA Middle має досвід у використанні інструментів автоматизації тестування, таких як фреймворки для тестування, інструменти для написання та виконання тестових скриптів. Він розробляє автоматизовані тестові набори та виконує їх.

**Управління дефектами:** QA Middle виявляє, документує та відстежує дефекти у програмному забезпеченні. Він спілкується з розробниками та іншими учасниками команди для вирішення дефектів та підтримує їх виправлення.

**Участь у розробці тестових стратегій:** QA Middle бере активну участь у визначенні та розробці тестових стратегій, процесів та практик у команді. Він допомагає вдосконалювати тестові процеси та підвищувати якість програмного забезпечення.

**Менторство та підтримка:** QA Middle може бути відповідальним за підтримку і навчання менших рівнів QA-спеціалістів, таких як стажери або Junior-інженери, надаючи їм допомогу та керівництво.

QA Middle має достатній досвід у сфері QA та може працювати більш самостійно, брати на себе більш складні проекти та вирішувати проблеми, пов'язані з якістю програмного забезпечення. Він продовжує розвиватися в області QA, набуває глибших знань та досвіду, з метою досягнення більш високого рівня, яким є QA Senior.

**QA Senior (старший QA-інженер)** - це висококваліфікований фахівець з контролю якості програмного забезпечення, який має значний досвід та глибокі знання в області QA. QA Senior володіє високим рівнем компетенцій і виконує складні завдання у сфері тестування та забезпечення якості продукту.

**Основні характеристики та обов'язки QA Senior включають:**

**Планування та керівництво тестовим процесом:** QA Senior бере участь у розробці стратегій тестування, планує ресурси та керує тестовим процесом в команді. Він визначає пріоритети та встановлює стандарти для досягнення високої якості продукту.

**Архітектура тестування:** QA Senior розробляє архітектуру тестування, включаючи вибір інструментів та технологій, створення тестових наборів та створення тестових скриптів. Він гарантує ефективність та надійність тестового набору.

**Впровадження автоматизації:** QA Senior має глибокі знання та досвід у використанні автоматизованих засобів тестування. Він впроваджує автоматизацію тестування, створює та підтримує тестові скрипти та набори для автоматичного виконання тестів.

**Ведення та аналіз тестової документації:** QA Senior відповідає за створення та підтримку тестової документації, включаючи тестові плани, сценарії, звіти та інші важливі документи. Він аналізує результати тестування та надає рекомендації для покращення якості продукту.

**Керівництво та навчання:** QA Senior може виступати на ролі наставника для менших рівнів QA-спеціалістів, надаючи їм керівництво, підтримку та навчання. Він сприяє розвитку команди, ділиться своїм досвідом та навичками.

**Взаємодія з командою та клієнтами:** QA Senior активно спілкується з іншими членами команди розробки, продуктовими менеджерами та клієнтами, забезпечуючи високу якість та задоволення вимог.

    QA Senior володіє високим рівнем експертизи в області QA, має глибокі знання та досвід, які дозволяють йому керувати тестовими процесами та забезпечити високу якість продукту. Він виступає важливим ланком у команді розробки та сприяє досягненню успіху проекту.

**QA Lead (керівник QA)** - це висококваліфікований фахівець з контролю якості програмного забезпечення, який відповідає за керівництво та організацію роботи відділу QA. QA Lead має значний досвід у сфері тестування та глибокі знання QA-процесів, і він виконує ключову роль у забезпеченні високої якості продукту та вдосконаленні тестових практик.

**Основні характеристики та обов'язки QA Lead включають:**

**Керівництво відділом QA:** QA Lead відповідає за керівництво і координацію роботи команди QA. Він планує, організовує та контролює діяльність QA-спеціалістів, забезпечуючи виконання тестових завдань та вимог до якості.

**Стратегічне планування QA:** QA Lead розробляє стратегії тестування, встановлює стандарти та норми QA, спрямовує розвиток тестових процесів та методологій у команді.

**Управління проектами тестування:** QA Lead веде проекти тестування, визначає обсяг робіт, розподіляє ресурси, контролює виконання завдань та забезпечує вчасну доставку якісного продукту.

**Визначення та впровадження найкращих практик:** QA Lead вдосконалює тестові процеси та впроваджує найкращі практики у сфері QA, забезпечуючи ефективність, автоматизацію та стандартизацію.

**Аналіз та звітність:** QA Lead аналізує результати тестування, готує звіти про якість, виявлені проблеми та рекомендації для вдосконалення процесів тестування та продукту.

**Навчання та наставництво:** QA Lead надає керівництво, навчання та наставництво менш досвідченим членам команди QA, допомагаючи розвивати їх навички та вміння.

**Співпраця з командою розробки:** QA Lead взаємодіє з іншими членами команди розробки, включаючи розробників, бізнес-аналітиків та менеджерів, для забезпечення взаєморозуміння, вирішення питань якості та покращення процесів.

    QA Lead має розширені знання в області QA, володіє стратегічним та оперативним мисленням, та здатністю ефективно керувати командою QA. Він є ключовою фігурою в забезпеченні високої якості програмного забезпечення та ефективного функціонування відділу QA. QA Lead виконує стратегічні завдання, керує процесами, координує роботу команди та сприяє досягненню поставлених цілей.

**QA Manager (менеджер QA)**

**QA Manager (менеджер QA)** - це вищий рівень керівного посадового становища у сфері контролю якості програмного забезпечення. QA Manager відповідає за управління відділом QA та забезпечення високої якості продукту, відповідає перед вищим рівнем управління і координує роботу всієї QA-команди.

**Основні характеристики та обов'язки QA Manager включають:**

**Управління відділом QA**: QA Manager керує всіма аспектами діяльності відділу QA, включаючи планування, організацію, контроль та керівництво персоналом. Він встановлює стратегічні цілі, визначає ролі та обов'язки у команді, розподіляє ресурси та керує бюджетом.

**Стратегічне планування QA:** QA Manager розробляє стратегії тестування та поліпшення якості продукту. Він встановлює норми та процедури QA, розробляє політики та стандарти, спрямовані на забезпечення високої якості програмного забезпечення.

**Управління проектами тестування:** QA Manager веде та керує проектами тестування, забезпечуючи виконання тестових завдань вчасно та в межах бюджету. Він визначає стратегію тестування, розробляє плани, встановлює пріоритети та забезпечує відповідну документацію.

**Керівництво та розвиток команди:** QA Manager керує та розвиває членів команди QA. Він забезпечує навчання, наставництво, оцінку роботи та кар'єрне зростання. QA Manager стимулює співпрацю та ефективну комунікацію в команді.

**Аналіз та звітність:** QA Manager аналізує результати тестування, готує звіти про якість, виявлені проблеми та рекомендації для вдосконалення QA-процесів та продукту. Він забезпечує високу якість звітності та інформує стейкхолдерів про стан якості.

**Взаємодія з іншими відділами:** QA Manager співпрацює з розробниками, продуктовим менеджментом та іншими відділами компанії. Він встановлює ефективну комунікацію, вирішує питання якості та сприяє досягненню загальних цілей організації.

      QA Manager володіє великим досвідом та глибокими знаннями в області QA, володіє стратегічним мисленням та керівними навичками. Він є важливим ланком в створенні та забезпеченні високої якості програмного забезпечення, впроваджує кращі практики та сприяє досягненню успіху організації.

**QA Director (директор QA)**

**QA Director (директор QA)** - це вищий рівень керівного посадового становища у сфері контролю якості програмного забезпечення. QA Director відповідає за стратегічне керівництво та розвиток департаменту QA в організації. Він має широкий огляд над всіма аспектами QA і виступає відповідальним за впровадження стратегій, політик та практик, спрямованих на забезпечення високої якості програмного забезпечення.

**Основні характеристики та обов'язки QA Director включають:**

**Стратегічне планування QA:** QA Director визначає стратегічні цілі департаменту QA, розробляє та впроваджує стратегії тестування та поліпшення якості. Він встановлює довгострокові плани розвитку QA та забезпечує їх виконання.

**Керівництво та організація:** QA Director керує всією командою QA та забезпечує її ефективну роботу. Він розподіляє ролі та обов'язки, встановлює стандарти та процедури, сприяє розвитку та підвищенню кваліфікації членів команди.

**Стандартизація та нормативні вимоги:** QA Director встановлює нормативні вимоги, стандарти та політики, що стосуються QA. Він впроваджує процеси стандартизації, забезпечує дотримання правил та регуляторних вимог в області QA.

**Управління ризиками та якістю:** QA Director аналізує ризики, пов'язані з якістю програмного забезпечення, та розробляє стратегії для їх управління. Він встановлює системи контролю якості та моніторингу, щоб забезпечити високу якість продукту.

**Взаємодія зі стейкхолдерами:** QA Director спілкується з вищим рівнем управління, відділами розробки, продуктовим управлінням та іншими стейкхолдерами. Він представляє департамент QA, забезпечує звітність про роботу та результати.

**Навчання та розвиток:** QA Director забезпечує навчання, розвиток та підвищення кваліфікації членів команди QA. Він впроваджує програми навчання та стажування, сприяє професійному розвитку та кар'єрному зростанню.

**Дослідження та інновації:** QA Director веде дослідження та аналіз нових технологій, методів тестування та підходів до забезпечення якості. Він сприяє впровадженню інноваційних рішень та покращенню ефективності QA процесів.

   QA Director володіє високим рівнем експертизи в областіQA, має стратегічні та лідерські навички, і виконує важливу роль у забезпеченні високої якості програмного забезпечення в організації. Він займає важливе місце в управлінській команді та сприяє досягненню успіху в області QA.

**Manual and Automation QA**

Говорячи далі про QA, можно також зробити поділ на QA manual та QA automation. Ручний тестувальник і автоматизатор - це два різних типи QA-спеціалістів, які мають свої особливості та обов'язки. Основна різниця між ними полягає в тому, як вони виконують процес тестування та використовують інструменти.

**Ручний тестувальник** виконує тестування програмного забезпечення вручну, без використання автоматизованих інструментів. Він виконує тест-кейси та сценарії, які заздалегідь розроблені для перевірки функціональності, виконання бізнес-логіки та інших вимог до продукту.    Ручний тестувальник активно спілкується з розробниками та іншими членами команди, вносить звіти про знайдені помилки та проблеми та сприяє їх вирішенню. Він має велику експертизу в області тестування, розуміє методи та підходи до проведення тестів та використовує свої знання для забезпечення високої якості продукту.

**Автоматизатор** використовує автоматизовані інструменти та програми для виконання тестів. Він розробляє та підтримує тестові скрипти та набори, які автоматично виконуються для перевірки функціональності та відповідності вимогам продукту. Він має глибокі знання програмування та розуміє мови програмування, такі як Java, Python або C#, для розробки автоматизованих тестів. Автоматизатор спрямований на автоматизацію повторюваних, рутинних та складних тестів, що дозволяє прискорити процес тестування та забезпечити більшу покриття функціональності. Він співпрацює з іншими членами команди розробки та QA, надає підтримку в розробці функціональності з можливістю автоматизації, а також аналізує результати тестування та звітує про них. Основна перевага ручного тестування полягає в здатності здійснювати складні, нестандартні та користувальницькі тестування, які складно автоматизувати. Ручний тестувальник може здійснювати оцінку інтерфейсу користувача, взаємодію з програмним продуктом та інші аспекти, які потребують відчуття та сприйняття людиною.

    З іншого боку, автоматизація тестування дозволяє зекономити час та ресурси, забезпечує швидкість та повторюваність тестів. Автоматизатори зосереджуються на розробці та підтримці автоматизованих тестів, що дозволяє виконувати їх на різних платформах та конфігураціях.

 В загальному, обидва типи спеціалістів важливі для успішного тестування програмного забезпечення. Вибір між ручним тестуванням і автоматизацією залежить від конкретних вимог проекту, обсягу тестування та доступних ресурсів. Часто використовується комбінація ручного та автоматизованого тестування для досягнення оптимального результату.

**Що таке тестування та його цілі ?**

**Тестування** - це процес перевірки та оцінки програмного забезпечення з метою виявлення дефектів, помилок та недоліків. Воно виконується з метою забезпечення якості продукту та впевненості в його відповідності вимогам та очікуванням користувачів. Тестування є невід'ємною частиною розробки програмного забезпечення та включає різні методики, підходи та процедури.

**Цілі тестування**

**Оцінка робочого продукту, вимог, дизайну, коду:** Ціль тестування полягає в оцінці якості робочого продукту, вимог, дизайну та коду. Тестування допомагає встановити, наскільки продукт відповідає встановленим вимогам, чи виконує необхідні функції, чи є коректним у відображенні дизайну та наскільки ефективним та безпомилковим є код програми.

**Перевірити, чи були виконані вимоги:** Одна з цілей тестування - переконатися, що всі вимоги до продукту були виконані. Тестування допомагає перевірити, чи відповідає програмний продукт усім встановленим вимогам та специфікаціям, що дозволяє забезпечити високу якість та відповідність продукту очікуванням користувачів.

**Підвищити впевненість у рівні якості об'єкту, що тестується:** Тестування спрямоване на підвищення рівня впевненості у якості об'єкту, що тестується. Шляхом проведення тестів та перевірок, тестувальники отримують інформацію про функціональність, надійність та відповідність продукту вимогам, що дозволяє встановити його якість та готовність до випуску.

**Запобігання дефектам:** Одна з основних цілей тестування - запобігання дефектам. Тестування допомагає виявляти та усувати помилки та дефекти на ранніх етапах розробки, що дозволяє зменшити ризики появи проблем у продукті після випуску та забезпечити високу якість програмного забезпечення.

**Знаходження дефектів:** Ціль тестування - виявити дефекти та помилки в програмному забезпеченні. Шляхом проведення тестів та аналізу результатів, тестувальники знаходять та документують дефекти, що дозволяє розробникам виправити їх та забезпечити високу якість продукту.

**Зменшення рівня ризиків:** Тестування сприяє зменшенню рівня ризиків у програмному забезпеченні. Шляхом проведення тестів та аналізу результатів, тестувальники виявляють потенційні проблеми та ризики, що дозволяє прийняти відповідні заходи для зменшення впливу цих ризиків на продукт.

**Дотримання правових, нормативних вимог та стандартів:** Тестування допомагає перевірити, чи відповідає програмне забезпечення правовим, нормативним вимогам та стандартам, встановленим для конкретної галузі чи регуляторного середовища. Це важливо для забезпечення відповідності продукту вимогам законодавства, безпеки та якості.

**А якщо продукт має помилки, невірно написаний код або щось інше як це називається ?**

**Error –** дія людини, яка призводить до неправильного результату роботи програмного забезпечення та може призвести до появи Defects (несправності чи помилки) у програмному коді. Помилка може бути результатом неправильної логіки, невірного алгоритму або неправильного виконання програмного коду. Наприклад, в програмі для обчислення суми двох чисел, якщо вона неправильно додає числа, це може бути помилкою.

**Defects –** недосконалість або нестача робочого продукту, якщо він не відповідає його вимогам та специфікаціям. Дефекти можуть бути виявлені під час тестування або після випуску програмного продукту. Наприклад, якщо в програмі для введення даних неправильно обробляються особливі символи, це може бути дефектом.

**Failure –** відхилення компонента чи системи від очікуваного виконання, експлуатації чи результату. Відмова відбувається, коли дефект призводить до некоректної роботи програми під час її використання реальними користувачами. Наприклад, якщо в програмі для відкриття файлів не відбувається правильне завантаження файлу і програма не може коректно працювати з ним, це може бути відмовою.

**А звідки беруться помилки ?**

    Важливо розуміти, що помилки є невід'ємною частиною процесу розробки програмного забезпечення. Важливо виявляти та виправляти помилки на ранніх етапах розробки, використовуючи ефективні методики тестування та контролю якості, щоб забезпечити якісне та надійне ПО.

﻿

**Вартість дефекту** - це кількісний показник, що визначає економічний збиток або втрату, яку спричиняє дефект у програмному забезпеченні. Вартість дефекту може включати як прямі, так і непрямі витрати, пов'язані з виявленням, усуненням та наслідками дефекту.

**Оцінка вартості дефекту може варіюватися залежно від різних факторів, таких як:**

**Вплив на користувача:** Якщо дефект призводить до серйозної незручності або неправильної роботи продукту, це може мати значний вплив на користувачів, що може позначитися на їхній продуктивності або задоволеності. Чим більша ймовірність виникнення проблеми та її вплив на користувачів, тим вища вартість дефекту.

**Потенційні втрати:** Дефекти, які призводять до втрати даних, недоступності сервісу або інших критичних проблем, можуть мати значний фінансовий вплив на компанію або користувачів. Наприклад, у випадку системи електронної комерції, дефект, який призводить до непрацездатності платіжної системи, може призвести до втрати продажів та негативного впливу на репутацію компанії.

**Затрати на виправлення:** Вартість дефекту також включає витрати на виявлення та виправлення проблеми. Це може включати витрати на аналіз, налагодження, перевірку та тестування, а також витрати на розробку та розгортання виправлення.

**Затрати на відновлення довіри:** Якщо дефект має негативний вплив на сприйняття продукту або на репутацію компанії, можуть знадобитися додаткові зусилля та ресурси для відновлення довіри користувачів. Це може включати витрати на комунікацію, підтримку клієнтів та маркетингові активності.

    Важливо враховувати, що вартість дефекту може бути різною для різних організацій і проектів. У кожному конкретному випадку варто провести оцінку вартості дефекту, щоб краще зрозуміти його вплив та прийняти рішення щодо пріоритетів у виправленні дефектів.

Також дуже важливо враховувати що вартість дефекту може змінюватися залежно від стадії розробки програмного забезпечення (ПО).

**Оцінка вартості дефекту може варіюватися на різних етапах життєвого циклу розробки**

**Вартість дефекту на етапі раннього проектування:** На початкових етапах проектування вартість дефекту може бути відносно низькою, оскільки зміни та виправлення ще легко впроваджуються. У цей період дефекти, виявлені вимоги та дизайн, можуть бути виправлені з меншими затратами часу та ресурсів.

**Вартість дефекту на етапі розробки:** Під час фази розробки, коли програмне забезпечення вже виробляється, вартість дефекту може почати зростати. Невиявлені дефекти на цьому етапі можуть призвести до поширення помилок та проблем в інших частинах ПО. Витрати на виправлення дефектів на етапі розробки можуть бути високими, оскільки це може вимагати переробки коду та змін у вже встановлених компонентах.

**Вартість дефекту на етапі тестування:** Під час фази тестування вартість дефекту може залежати від того, наскільки ефективно виявлені проблеми та як швидко вони виправляються. Незалежно від того, чи це ручне тестування чи автоматизоване тестування, виявлення дефектів на цьому етапі може допомогти запобігти їх поширенню та зменшити наслідки.

**Вартість дефекту на етапі експлуатації:** Після випуску програмного забезпечення, вартість дефекту може стати найвищою. Недоліки та проблеми, виявлені після випуску, можуть мати значний вплив на користувачів, бізнес та репутацію компанії. Витрати на виправлення та підтримку виробленого ПО можуть бути значними.