**Склади порівняльну таблицю найбільш поширених методологій:**

*а. Надаючи відповіді, обґрунтуй свою думку: чому це саме сильна/слабка сторона/доцільна галузь застосування.*

| **№** | **Назва методології** | **Сильні сторони** | **Слабкі сторони** | **Для якої галузі є доцільною** |
| --- | --- | --- | --- | --- |
| **1.** | **Waterfall** | Кожен етап йде чітко за наступним; чітке планування, терміни та документація; кожен етап починається з певною готовою базою; можлива реалізація в коротких проєктах *(це є сильною стороною, оскільки така структурованість забезпечує передбачуваність, стабільність і полегшує управління проєктами, якщо вимоги не змінюються)*. | Потрібно переплановувати кожен етап, якщо є якась зміна *(негнучкість є слабкою стороною, адже будь-яка зміна на пізніх етапах коштуватиме дорого і буде надто трудомісткою)*. | Проєкти з чіткими та фіксованими вимогами, де зміни малоймовірні *(наприклад, державні або військові розробки)*. |
| **2.** | **Agile** | Гнучкість і швидке реагування на зміни завдяки ітеративному підходу; можливість бути причетним до проєкту і його змін; клієнт постійно залучений до процесу, що забезпечує відповідність кінцевого продукту його потребам і очікуванням; фокус на працюючому продукті, а не на документації *(ці принципи є сильними сторонами, оскільки вони дозволяють створювати продукт, який краще адаптовується до ринку і задовольняє потреби замовника)*. | Більш хаотична модель: проєкти є менш передбачуваними щодо кінцевих термінів виконання та бюджету через постійні зміни вимог; потребує високої самоорганізації та дисципліни команди, а також постійної взаємодії із замовником, що може бути стресово для команди. | Проєкти зі змінними та невизначеними конкретно вимогами, стартапи, розробка програмного забезпечення, що виходить на ринок. |
| **3.** | **Kanban** | Максимальна гнучкість, оскільки відсутні часові обмеження (спринти); візуалізація робочого процесу допомагає швидко виявити і усунути “вузькі місця”, сприяє постійному вдосконаленню *(сильні сторони цієї методології дозволяють легко адаптуватися до будь-якого проєкту і це забезпечує ефективне управління потоком завдань).* | Не підходить для проєктів, де потрібні чіткі кінцеві терміни, оскільки немає жорстких обмежень по часу; не має певної ролі в корпоративній культурі *(без наставництва підготовленого тренера ця методологія не матиме високої ефективності, а також вона є більш складною для починаючих QA).* | Чиста розробка програмного забезпечення; технічне обслуговування, системне адміністрування, маркетинг або продаж. |
| **4.** | **V-model** | Постійне тестування, яке відповідає кожній фазі розробки - поступовість великий +, адже це гарантує високу якість продукту; фокус на детальній документації та вимогах, що допомагає відстежувати зміни *(сильною стороною є гарантія якості, оскільки кожен етап тестування ретельно перевіряє відповідну фазу розробки).* | Негнучкість; зміни в проєкті можуть бути дуже дорогими та складними, оскільки потрібно перероблювати пов’язані етапи тестування та розробки. | Проєкти з високими вимогами до безпеки та надійності, де помилки є критичними (наприклад, медичне обладнання або авіація, тощо) |
| **5.** | **Scrum** | Фокус на частих поставках працюючого продукту через короткі ітерації (спринти); прозора комунікація та швидке вирішення проблем завдяки щоденним зустрічам; сприяє самоорганізації команди *(завдяки спринтам і регулярним зустрічам, команда швидко адаптовується і постійно покращує якість продукту)*. | Не підходить для малих проєктів; можуть виникнути проблеми, якщо вимоги не є стабільними протягом спринту. | Підходить для галузей зі складними, мінливими вимогами, де потрібна гнучкість, швидке реагування на зміни та часта співпраця між членами команди. Scrum успішно застосовується у фінансах, маркетингу, освіті, банківській сфері, телекомунікаціях, фармацевтиці, туризмі та інших галузях для створення продуктів, управління проєктами, оптимізації процесів та підвищення ефективності команд. |
| **6.** | **Spiral** | Ключова перевага підходу - ризик менеджмент; дозволяє включати прототипування; підходить для великих, складних проєктів, де багато невизначеності *(завдяки фокусу на аналізі ризиків, ця модель дозволяє ідентифікувати та усувати потенційні проблеми на ранніх етапах)*. | Дуже висока вартість та складність, адже кожен цикл вимагає детального аналізу ризиків; може бути неефективною для невеликих проєктів. | Масштабні, проєкти з високими ризиками (наприклад, військова або аерокосмічна розробка), науково-дослідні проєкти. |

**Beet Sprout — детальніше заглибся в практику.**

* ~~1. Виконай завдання попереднього рівня.~~

**2. Напиши розгорнуті відповіді (0,5 - 1 сторінки тексту) на такі два питання:**

**На твою думку, чому з’явився Agile-маніфест?**

**Які проблеми він мав вирішити і чи це вдалося?**

Об’єднаю відповідь на ці питання в одну:

На мою думку, Agile-маніфест з’явився тому, що в кінці 90-х і на початку 2000-х років у розробці програмного забезпечення було дуже багато проблем. Тоді більшість компаній користувалися старими підходами, такими як “Waterfall”. Суть його в тому, що проєкт робили етап за етапом: спочатку планування, потім проєктування, далі розробка, тестування і лише наприкінці - готовий продукт. Виглядало це логічно, але на практиці майже завжди виникали труднощі. Замовник міг побачити результат тільки через кілька місяців або навіть років і часто виходило так, що продукт йому не підходив. Змінювати щось у процесі було дуже дорого і складно. Через це багато проєктів затягувались, ставали невдалими або закінчувались провалом.

Agile-маніфест став відповіддю на ці проблеми. Група розробників зібралася, щоб переосмислити підхід і придумати більш гнучкий спосіб роботи. Вони зробили акцент не на документах і суворих планах, а на людях, командній роботі та постійній співпраці з клієнтом. Важливим стало те, щоб замовник бачив результат уже на ранніх етапах і міг впливати на подальше створення та розвиток продукту. По суті, Agile дав розробникам свободу швидко реагувати на зміни й робити програму більш корисною та актуальною.

Цей маніфест мав вирішити кілька основних проблем. По-перше, це негнучкість - раніше план завжди ставився вище за зміни, а тепер зміни стали нормою. По-друге, проблема відриву від замовника: раніше він бачив продукт лише наприкінці, а зараз бере участь у процесі. По-третє, скорочення зайвої бюрократії - ніхто не витрачає купу часу на папери, головне показати працюючу версію програми. Ну і найважливіше - люди в команді. Адже навіть найкращі інструменти нічого не варті, якщо немає довіри, співпраці та нормальної комунікації.

Чи вдалося вирішити ці проблеми? Думаю, що так. Agile став основою для сучасних методів, як-от Scrum чи Kanban, і сьогодні він використовується у багатьох компаніях у світі. Завдяки цьому підходу продукти створюють швидше, вони краще відповідають потребам клієнта, а команди працюють ефективніше. Звичайно, Agile не можна назвати “чарівною пігулкою” - не у всіх компаніях він працює ідеально, бо все залежить від людей і корпоративної культури. Але там, де його принципи реально застосовують, результати набагато кращі, ніж у старих “жорстких” підходах.

Отже, на мою думку, Agile-маніфест з’явився як реакція на кризу старих методів і справді допоміг зробити розробку більш гнучкою та орієнтованою на людину. Він не тільки вирішив проблеми того часу, але й задав новий напрямок розвитку всього ІТ.

**Mighty Beet — різнобічно опануй тематику уроку.**

* ~~1. Виконай завдання двох попередніх рівнів.~~

**2. Ти – засновник/ця стартапу і плануєш випустити на ринок мобільний застосунок для обміну світлинами котиків. Яку методологію ти обереш для процесу розробки і чому? Відповідь текстово обґрунтуй.**

Я б обрала Agile, а саме Scrum. Причина проста: у стартапі завжди багато невідомого і неточного. Ми не знаємо, що саме сподобається користувачам - просто фото, чи ще фільтри, коментарі користувачів, рейтинг котиків. Якщо працювати по жорсткому плану, можна витратити купу часу на непотрібні речі.

Scrum дозволить зробити маленькі кроки: за 1-2 тижні ми вже матимемо щось готове, що зможемо презентувати користувачам. Вони нададуть фідбек і ми швидко вирішимо, що додати або змінити. Це дуже зручно для стартапу з невеликою командою, бо замість великої документації ми робимо акцент на спілкуванні й швидких результатах.

Тому Agile/Scrum, на мою думку, найкраще підходить для такого застосунку - можна швидко перевіряти ідеї та розвивати продукт разом із користувачами.