**Перший рівень**

Склади порівняльну таблицю найбільш поширених методологій:

| № | Назва методології | Сильні сторони | Слабкі сторони | Для якої галузі є доцільною |
| --- | --- | --- | --- | --- |
| 1 | Waterfall | * Чіткий план, дозволяє структурувати роботу і полегшити процес розробки продукту * Ведення детальної документації зменшує ризик виявлення невідповідностей, нечіткостей * Зрозуміла та проста логіка * Стабільність завдань * Реальна оцінка вартості й строків закінчення проєкту | * Помилки можна виправити тільки під час фази, що збільшує вірогідність отримати фінальний продукт із багами * Період тестування проходить досить пізно * Відсутність гнучкості, неможливість корегувати попередні кроки для поліпшення результату * У разі дефіциту ресурсів може постраждати якість продукту, адже оцінка вартості і строків визначається на весь період розробки продукту | * Підходить для проектів, де чітко визначені вимоги |
| 2. | V - model | * Строга етапність у розробці * Фаза тестування йде паралельно з кожною фазою розробки, тобто планування тестування і верифікація системи виробляються на ранніх етапах * Наявне проміжне тестування,що дозволяє виявити дефекти на більш ранній стадії | * Недостатня гнучкість моделі, що унеможливлює роботу з паралельними подіями і можливість динамічного внесення змін * Власне створення програми відбувається на етапі написання коду, тобто вже в середині процесу розробки; * Недостатній аналіз ризиків | * У проектах, в яких існують часові та фінансові обмеження; * Для завдань, які передбачають більш широке, порівняно з каскадної моделлю, тестове покриття. |
| 3 | Spiral model | * Ризик-орієнтована модель * Розвиток відбувається швидко * Більші проекти / програмне забезпечення створюються та обробляються стратегічно * Контроль на всіх фазах розвитку. * Все більше і більше функцій додаються систематично. * Програмне забезпечення виробляється рано. * Має місце для зворотного зв'язку із клієнтами, і зміни впроваджуються швидше. | * Аналіз ризиків є важливою фазою, тому потрібні досвідчені люди. * Невигідний для менших проектів. * Більше документації, оскільки є проміжні фази. * Для менших проектів це дорого. | * Коли проект великий. * Там, де програмне забезпечення потребує постійної оцінки ризиків. * Вимоги дещо складні і вимагають постійного уточнення. * Програмне забезпечення вимагає значних змін. * Де достатньо часових рамок, щоб отримати відгук кінцевого користувача. * Там, де випуски повинні бути частими. |
| 4 | Scrum | * Прозорість моделі * Гнучкість * Scrum-команди самоорганізуються. Це означає, що вони самі керують своїм часом та роботою в межах спринту. * Взаємодія у колективі, тобто відкритий обмін інформацією, знаннями, проблемами * Замовник завжди в курсі процесу робіт, вносить правки, отримує достовірну інформацію про терміни здачі проєкту. * Дозволяє отримувати чергову версію продукту частіше, регулярно підтримувати зворотній зв’язок та швидко доопрацьовувати продукт, поліпшуючи процес роботи | * Неможливо змінити план робіт під час дії спринту * Не підходить для проєктів з нечіткими вимогами до кінцевого продукту, тому що замовник може нарощувати функціонал до нескінченності * Загальний успіх проєкту почасти залежить від професіоналізму однієї людини – скрам-майстра. | * Застосовують там, де є продукт, що має цінність для користувачів та замовників * Коли потрібен високий рівень контролю над результатами роботи, Scrum — хороший вибір. Коли вам потрібно знати, що саме буде готово через певний строк, щоб повідомити про це зацікавленим особам та клієнтам. |
| 5 | Kanban | * Візуалізація, яка допомагає зосередитись на процесах * Гнучкість планування * Менша тривалість циклу розробки * Обмежена кількість завдань в процесі,зменшує ймовірність перенавантаження * у Kanban можна інтегрувати інші принципи управління проектами для покращення процесу розробки | * Можна “втратити” задачі з високим пріоритетом, оскільки співробітники самі беруть завдання в роботу, * Вимагає постійної наявності завдань, як на конвеєрі. * Немає часових обмежень. В Kanban немає строків, для виконання завдання дається стільки часу, скільки потрібно. Це може створити проблеми з дедлайнами проєкту. | * Коли ви маєте справу з різноманітною або непередбачуваною роботою, коли пріоритети часто змінюються або є багато завдань різного об’єму. Kanban добре працює в таких випадках, тому що його легко підлаштувати під власні потреби. |

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

**Другий рівень**

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?

На мою думку, Agile-маніфест виник як наслідок еволюційного процесу розвитку методологій, які попередньо використовувались для розробки програмного забезпечення. З розвитком технологій, технік, методів, ресурсів попередні методології почали явно відставати від того рівня. Agile виник на противагу методологіям, які почали втрачати свою ефективність і з кожним роком почали мати все більше негативних рис.Тобто не гнучкі методології, які використовувалися при розробці ПЗ не покривали усіх тих необхідних вимог для того, щоб випустити якісний продукт. Agile мав вирішити ці проблеми, а саме: розробка продукту, що матиме високу цінність для користувачів, пришвидшити процес розробки ПЗ, зробивши його якіснішим, з можливістю адаптації до змін, які будуть вносити як замовники так і вимагатимуть користувачі.

Звичайно Agile покрив усі ті проблемні місця, які мали інші методології, і зміг привнести новий спосіб мислення для команд розробників. Саме завдяки Agile працюючий продукт став важливішим за документацію, саме цей фактор пришвидшив процес розробки продукту, адже для створення детальної документації витрачалось чимало часу. Далі великою відмінністю Agile стало те, що готовність до змін займала вищий пріоритет за дотримання чіткого плану. Зараз ми живемо в дуже мінливий час, коли технології розвиваються стрімко і необхідно йти в ногу з часом, тому треба швидко реагувати на зміни для випуску актуального продукту. Також співпраця із замовником у Agile-маніфесті важливіша за умови контракту. Замовник завжди в курсі процесу робіт, вносить правки, отримує достовірну інформацію про терміни здачі проект, саме тому зменшується ризик того, що розроблюваний продукт не буде відповідати тим уявленням, які має замовник. І останнім не менш важливим принципом Agile є люди та співпраця важливіші за процеси та інструменти. Даний принцип покриває проблему продуктивності команди. Адже він заохочує відкритий обмін інформацією, знаннями, проблемами між членами команди, що допомагає команді швидко справлятися з труднощами, що виникають на різних етапах розробки і розвиватися всім членам комади.

**Третій рівень**

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 Scrum - метод. Scrum використовують там, де є продукт, що має цінність для користувачів та замовників.

За даної методології замовник має доступ до всієї інформації, вносить правки та в курсі, що відбувається на кожному етапі розробки. Тобто в кінці кожного спринту я, як замовник, матиму можливість бачити робочі частинки продукту (інкременти) і далі зможу вносити зміни в той момент, коли це буде необхідно. Мені імпонує дана методологія тим, що весь процес розробки розділений на спринти - часові проміжки, де чітко визначені завдання і їх пріоритети, що вносить ясність і чіткість у процес розробки.