**Домашнє завдання до уроку 3**

**Перший рівень**

|  |  |  |  |
| --- | --- | --- | --- |
| **Назва методології** | **Сильні сторони** | **Слабкі сторони** | **Для якої галузі є доцільною** |
| **Waterfall** | Проста і зрозуміла за структурою методологія, конкретні результати уже є на кожній стадії. | Неможливе раннє тестування, а отже висока вартість виявлених дефектів.  Не гнучка до зміни вимог. | Підходить для проектів державного замовлення, наприклад для оборонного відомства. |
| **V-модель** | Проста у використанні та розумінні.  Тестування проводяться на кожному етапі розробки - дефекти можна виявити на ранніх стадіях. | Не гнучка до зміни вимог.  При внесенні змін до вимог, потребує зміни тестової документації. | Схожа модель до каскадної, важливою є документація. Наприклад, для виготовлення медичного обладнання. |
| **Екстремальне програмування** | Основна увага приділяється залученню клієнтів. Встановлюються графіки. Дуже гнучка до змін. | Ефективність залежить від залучених людей. Необхідність надмірних змін у розробці Через часті зміни майбутні результати точно не відомі. | Може застосовуватись тільки в області розробки програмного забезпечення. |
| **Спіральна модель** | Ітеративна модель - кожна ітерація проходить оцінку ризиків перед продовженням роботи.  Покращений аналіз ризиків.  Хороша документація процесу розробки. | Дорога у використанні, так як потребує висококласних спеціалістів для управління ризиками. Не підходить для малих проектів. | Підходить для дуже ризикованих проектів з різними бізнес-потребами. |
| **Scrum** | Командний проект. Розподіл обов’язків сприяє підвищеній продуктивності й економії часу. Усі учасники відповідають за результат, що підвищує мотивацію. Є часові обмеження - спринти. | Не підходить для великих проектів. Потрібна висококваліфікована команда. Постійні зміни можуть призвести до розширення меж проекту та затримок. Нестабільна вартість. | Ідеально підходить для керування одним масштабним проектом тривалістю від трьох місяців, який має вичерпну специфікацію і вимоги перед початком розробки. |
| **Kanban** | Команда фокусується на найважливішому. Не потрібно проводити щоденні наради, як у Scrum. | Немає часових обмежень, а отже у команди може не виникнути відчуття нагальності завершити завдання. | Підходить для довгострокових проектів, в яких відсутня чітка специфікація і завдання формуються в процесі розробки.  Kanban підійде командам, які працюють в маркетингу, при розробці програмного забезпечення або створенні контенту. |

**Другий рівень  
Чому з’явився Agile-маніфест?**

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

**Agile** (agile software development, від англ. Agile — швидкий) **— *це методологія гнучкої розробки***, яка дозволяє клієнтам швидше отримувати якісне програмне забезпечення.

**Agile-маніфест - *маніфест гнучкої системи розробки ПЗ.*** У цьому документі описані основні принципи (12) та цінності гнучкої системи розробки ПЗ.

Еволюція в технологічному розвитку, еволюція в мисленні не могла залишити в стороні підходи до розробки програмного забезпечення.

Основною проблемою, яку бачили розробники маніфесту: компанії настільки зосереджені на плануванні своїх циклів розробки ПЗ, що забули про головне — про те, що потрібно приносити радість клієнтам. Отже, Agile придумали для того, щоб встигати за змінами на ринку, швидко задовольняти потреби замовника та швидко реагувати на зміни.

За Agile усе відбувається значно швидше: тестування, доставка самого програмного продукту. Тоді як традиційні підходи до проектів досить повільні і часто дорогі, не заохочуються зміни, а зворотний зв’язок, в основному, ігнорується до повного закінчення проекту.

Використання методики Agile завоювало популярність і поза сферою програмного забезпечення. Ця методологія актуальна і сьогодні. Приваблює можливість контролювати розвиток та впровадження, одночасно сприяючи інноваціям та креативності. Компанії, які практикують використання технологій Agile зуміли легше адаптуватись до віддаленої роботи під час карантину.

*Маніфест розробки програмного забезпечення з методології Agile (цінності):*

***“Ми постійно відкриваємо для себе більш досконалі методи розробки програмного забезпечення, займаючись розробкою безпосередньо та допомагаючи іншим”***

Завдяки виконаній роботі ми змогли зрозуміти наступне.

**Люди та взаємодія** важливіші за процеси та інструменти.

**Працюючий продукт** важливіший за вичерпну документацію.

**Співпраця із замовником** важливіша за узгодження умов контракту.

**Готовність до змін** важливіша за проходження плану.

*“Тобто, не заперечуючи важливості того, що праворуч, ми таки більше* ***цінуємо те, що ліворуч****”.*

**Третій рівень**

**Для розробки мобільного застосунку для обміну світлинами котиків я, як власниця стартапу,**  обрала б гнучку методологію, наприклад, **Scrum.** Підходить те, що є часові рамки, тобто бачення, коли продукт буде готовий. Також тісний взаємозв'язок між розробником і командою - можна точніше передати свої вимоги і швидко відреагувати на зміни, якщо такі будуть. Це командна робота, а отже всі зацікавлені у реалізації продукту. Підходить для невеликих команд.