**Рівень 1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Waterfall | V-Model | Scrum | Kanban | XP |
| Плюси: Стійкість до змін;  Визначений початок і кінець проекту;  Детально описані вимоги на початку; Великий розмір команди;  Чітко визначений набір функцій; Перехід до наступного етапу можливий лише після повного завершення попереднього. | Плюси: Планування на ранніх стадіях розробки системи чи проекту;  Тестування відбувається паралельно розробці (тобто на кожному етапі);  Простота у використанні і розумінні;  В порівнянні з каскадною моделлю простіше відстежувати хід розробки. | Плюси: наявність спрінтів (2-4 тижні, що дає можливість вибрати конкретний пул задач);  присутній скрам мастер (допомагає краще розуміти процес, вирішити конфліктні питання, але він немає влади!;  наявне планування, що дозволяє нам дати відповідь на питання що ми маємо зробити? І як ми маємо це зробити?  присутня ретроспектива (це допомагає покращити роботу у майбутньому);  Невеликі команди від 3 до 9 людей задіяні в проекті (простіше спілкуватися один з одним)  Реліз після кожного спринта;  Часті мітинги команди;  Задачы набираються у спрінт беклог за пріорітетом з беклогу;  Задачі не можуть змінюватися під час спрінта. | Немає таймбоксів ні на задачі і спринт;  Обмежена кількість задач на певному етапі пропорційно до виконавців;  контроль термінів виконання певної задачі;  Одна дошка з задачами на весь процес роботи (в той час, як наприклад в скрамі на кожну ітерацію мати бути своя окрема дошка)ю | Плюси: Дозволяє оперативно реагувати на часті зміни у вимогах клієнта;  Колективне володіння кодом;  часті релізи (хоч кожного дня);  Стійкий темп роботи, але не більше, ніж 40 годин на тиждень;  Безпосередня участь замовника чи його представника у процесі розробки;  Парне програмування;  Замовник сам пише автотести, а команда їх проганяє. |
| Мінуси: Тут зміна є багом (приходиться починати роботу спочатку і це обходиться дуже дорого);  слабка участь клієнта у розробці;  великий розмір команди (важче спілкуватися) | Мінуси:  Відсутність ітерацій між фазами;  пізні терміни тестування вимог в життєвому циклі;  неможливість передбачити ризики у майбутній розробці. | Мінуси:  Ризик не дійти до закінчення проекту;  Якщо член команди виходить з спрінта, коли він вже запущений, то спрінт може не завершитися;  Важкість до адаптації деяких компаній при роботі з скрамом. | Мінуси:  Повна самоорганізація команди, відсутність зовнішнього контролю, пріоритизації чи оптимізації задач;  відсутність мітингів (їх може і не бути);  Канбан не призначений для довгострокового планування; | Мінуси: Нестабільні вимоги замовника;  Успіх проекту залежить від залучення замовника;  Через нестачу структури чи документації такі проекти невеликі (прості). |

**Рівень 2**  
  
Agile маніфест з’явився у зв’язку з тим, що було необхідно адаптувати програмне забезпечення до змін, які поступали від замовників. Тобто, якщо, наприклад, у Waterfall методології зміна є багом, то тут зміна виступає особливістю. Agile це гнучка розробка ПЗ, тобто це найкращий спосіб для підвищення продуктивності розробників ПЗ. Agile ніколи не завершиться, якщо в аналізі, дизайні, кодуванні, тестуванні залишиться ще хоч 1 фіча для розробки, так, як всі ці процеси йдуть паралельно.  
  
Основні ідеї:  
1. Особистості та їхні взаємодії є важливішими, ніж процеси та інструменти.  
  
  
2. Робоче ПЗ важливіше, ніж вичерпна документація.  
  
Документація, процеси чи інструменти безумовно важливі, але важливіше те, як люди розуміють один одного, їхнє націлення на результат є важливішим, ніж просте слідування процесам чи новим інструментам. (Це можна віднести і до пункту 1 також).  
  
3. Співпраця з замовником важливіше, ніж контрактні зобов’язання.  
Ми можемо прописати 10 платіжних методів, які ми хочемо використати, але, через відсутність спілкування з клієнтом, ситуація могла помінятися, і виявиться, що в кінці йому потрібні лише 2 методи. Тобто, ми просто потратили час на 8 непотрібних методів оплати.  
  
4. Реакція на зміни важливіша, ніж дотримання плану.  
Безумовно, якщо в нас не буде плану, ми зафейлимся, проте, якщо бізнес приходить до нас і каже, що нам потрібно до кінця місяця зробити нову фічу, а ми відповідаємо, що в нас на неї 2 місяці, ми все рівно повинні на це реагувати і внести зміни.  
  
**РІВЕНЬ 3.**Безумовно, я виберу Scrum. На мою думку, він найбільш точно дозволяє нам реагувати на зміни і вимоги клієнта. Саме скрам передбачає постійне спілкування з Product Owner, спілкування між членами команди на пленінгу, демо-мітингу, чи ретроспективі. Скрам дозволяє Project Manager вести контроль над тим, хто чим займається (Daily Standup), дозволяє вибирати пул задач з беклогу в спрінт беклог за пріорітетом і оцінити, скільки максимально з них ми можемо зробити (estimation), тут також наявний скрам-мастер, який покращує процес роботи команди. Скрам також дозволяє задокументувати кожен спрінт, вказати (goals, epics, critical bugs) у документі кожного спрінта.  
Скрам допомагає в даному випадку мобільному додатку розвиватися поступовою, тобто, на кожен спринт ми закладаємо певний пул задач.