**Перший рівень**

**Завдання 1**

| Методи розробки ПО | Переваги | Недоліки | Використовують |
| --- | --- | --- | --- |
| Waterfall | -Кожна стадія має чіткий перевіряємий результат;(кожен етап детально пропрацьовується)  -В кожен момент часу команда виконує один вид роботи; | -Повна неспроможність адаптувати проект до змін в вимогах;  -Вкрай пізнє створення працюючого продукту; | Для створення інженерних конструкцій (споруд, літаків, мостів тощо). |
| V-model | - Проста і зручна у використанні;  - Увагу тестуванню приділяють із першої ж стадії ( чим раніше ми будемо тестувати, тим менше ми витратимо грошей і часу)  - Добре працює для проектів зі стабільними вимогами.  - Добре працює для невеликих проектів; | -Дуже жорстка і найменш гнучка;  -Ранні прототипи ПЗ не створюються;  - Складність усунення проблем, пропущених на ранніх стадіях розвитку проекту. | Для малих та середніх проектів, де вимоги чітко визначені та зафіксовані; |
| Ітераційна | Високорівневий дизайн додатку, перш ніж ми фактично почнемо будувати проект;  Створюємо та вдосконалюємо продукт крок за кроком; Відстеження дефектів на ранніх стадіях;  Надійний зворотній зв’язок від користувачів;  Менше часу на документування, більше на проектування. | Кожна ітерація є жорсткою і не перетинається;  Можуть виникнути дорогі проблеми з архітектурою або дизайном системи.  Випуск ПЗ з яким можно працювати і котрий можно продавати затягується на довготривалий термін. | WEB-продукти, чат-боти, мессенджери. |
| Інкрементальна | Більш гнучка, менш затратна до змін обсягу та вимог;  Легше тестувати та налагоджувати протягом меншої ітерації;  Замовник може реагувати на кожну збірку;  Легше управляти ризиками;  Створюється робоче ПЗ швидко а на ранній стадії життєвого циклу ПЗ. | Потребує хорошого планування та проектування;  Потребує чіткого і повного визначення всієї системи, перш ніж її можно буде розбити на частини і будувати поетапно;  Загальна вартість вища, ніж у водоспаду. | Наприклад, соціальні мережі. |
| Спіраль | Глибокий аналіз ризиків;  Підходить для великих проектів;  Досить раннє прототипування. | Високі витрати;  Складність застосування для невеликих проектів;  Висока залежність успіху від якості аналізу ризиків. | приклад, Розробка системи «Розумний дім» |
| Agile (гнучкі) методології | Задоволення потреб клієнтів шляхом швидкого, безперервного постачання ПЗ;  Акцент на людях та взаємодії;  Замовники, розробники, тестувальники постійно взаємодіють.  Робоче ПЗ поставляється часто;  Постійна увага до технічної досконалості та гарного дизайну;  Готовність до змін важливіше слідування первинному плану. | Важко оцінити зусилля, які необхідні на початку життєвого циклу ПЗ;  Недостатньо уваги приділяється необхідному проектуванню та документації;  Проект може зійти з дистанції якщо замовник не має чіткого уявлення про кінцевий результат; | mobile app, web, але не тільки IT, а і для любого матеріального продукта - автомобіля, приборів. |

**Другий рівень**

**Завдання 2**

**1.Чому з’явився Agile-маніфест?**

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

Почнемо з історії розвитку програмування і організації роботи програмістів. Починаючи з 1950 року програмістами були математики, інженери - дорослі люди яким не потрібен менеджмент, відповідальні були до строків і домовленостям.

70-ті роки Комп’ютерів і програмістів дедалі стає більше, їх час стає менш цінним. Програмісти з одного боку мали реалізувати всі перечисленні вимоги в строк, а замовник, з іншого боку, уже не міг змінювати вимоги без слідування процедури додаткових угод до контракту. 90-ті роки велика кількість програмістів, підхід до розробки ставав все більш регламентованим та heavyweight, вимагали детальних специфікацій, звіт, розподіл ролей. Паралельно є деякі ініціативні професіонали які експерементували з більш lightweight способами взаємодії. Вони бачили ефективність в і прагнули до живого спілкування, меншої кількості і більш коротким регламентам, і виконували роботу ітераціями.

І вже в 2000 роках зібрались представники різних методологій, щоб знайти загальне, що об’єднує їх підходи/методи , які в той час вже називали lightweight.

Ціль була зафіксувати ці загальні характеристики.

Так і дійшли до того, що було створено 4 принципа:

* Люди і 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 дотримання початкового плану.

Покладені в основу гнучкої моделі підходи є логічним розвитком і продовженням усього того, що було за десятиліття створено й випробувано у водоспадній, v-подібній, ітераційній , інкрементальній, спіральній та інших моделях. Причому тут уперше було досягнуто відчутного результату в зниженні бюрократичної складової та максимальної адаптації процесу розроблення ПЗ до миттєвих змін ринку і вимог замовника.

**Третій рівень**

**Завдання 2**

2. Для того щоб випустити на ринок мобільний застосунок ( для обміну світлинами котиків) маємо розуміти, що розробка мобільний додатків має важливі характеристики, що належать до більшості фаз життєвого циклу:

* Мобільний проект часто схильний до внутрішніх або зовнішніх невизначеностей, таким як нечітко визначені вимоги, або часта зміна потреб користувачів; необхідність вносити зміни по ходу розробки.
* Доставка продукта на ринок якомога раніше і в подальшому додавати оновлення;
* Велика кількість користувачів з усього світу;
* Високий темп технічної еволюції - нові пристрої , релізи ОС, і т.д.

Тому для того, щоб знизити ступінь ризику, і впорядкувати процес мобільної розробки - я виберу методологію Agile з її адаптивним ( допустимість частих змін), ітеративно-інкрементальним ( зворотній зв’язок з замовником на кожній ітерації, і безліччю релізів), кооперативним ( щільне співробітництво розробників, замовника і кінцевих користувачів) і простим ( легко зрозуміти, змінювати і покращувати) підходом до розробки.