Завдання 1. Beet Seed

1.1. Порівняльна таблиця найбільш поширених методологій з обґрунтуванням

№	Назва методології	Сильні сторони	Слабкі сторони	Для якої галузі є доцільною
1.	**Waterfall**	- *Структурований підхід:* Методологія Waterfall дотримується лінійного та послідовного підходу, що робить її легкою для розуміння та впровадження. - *Чітка документація:* Кожна фаза моделі Waterfall має специфічні вимоги до документації, що забезпечує ретельне документування проекту на кожному етапі. - *Передбачуваність:* Завдяки заздалегідь визначеним часовим рамкам і результатам стає легше відстежувати прогрес і ефективно управляти ресурсами. Водоспад підштовхує ретельно оцінювати та опрацьовувати ризики прямо «на березі».	- *Обмежена гнучкість:* Лінійна природа Waterfall не дозволяє вносити зміни після завершення фази, що ускладнює адаптацію до мінливих вимог. - *Високий ризик:* будь-які непорозуміння або зміни у вимогах можуть призвести до дорогого перероблення, оскільки на пізніх етапах вносити зміни складніше. Ціна помилки може бути дуже велика. - *Запізнілий зворотній зв'язок:* Зацікавлені сторони можуть бачити лише кінцевий продукт, що призводить до того, що зворотній зв'язок та коригування вносяться на занадто пізньому етапі процесу. Процес розробки може тривати довгий час. Як результат відставання від ринку	Методологія Waterfall найкраще підходить для проектів з чітко визначеними та стабільними вимогами, де зміни малоймовірні. Наприклад: - Аерокосмічна та оборонна промисловість: Розробка програмного забезпечення для критично важливих систем в аерокосмічній та оборонній галузях, де вимоги визначені заздалегідь, а зміни є дорогими та ризикованими. - Охорона здоров'я: Створення програмного забезпечення для медичних пристроїв або систем охорони здоров'я, які вимагають ретельної документації, валідації та відповідності нормативним вимогам.
2.	**V-Model**	- *Структурований підхід:* V-модель дотримується систематичного та структурованого підходу, забезпечуючи ретельне планування та тестування на кожному етапі. - *Раннє виявлення дефектів:* Включаючи тестування на кожному етапі, V-Model сприяє ранньому виявленню та усуненню дефектів, що призводить до створення більш якісного програмного забезпечення. Випуск дуже якісної продукції	- Залучення спеціалістів профільної освіти - *Жорсткість:* V-модель часто критикують за її жорсткість і недостатню гнучкість у пристосуванні до змін під час процесу розробки. - *Високий рівень попереднього планування:* Детальне планування та документація, що вимагаються у V-моделі, можуть призвести до вищих авансових витрат і довших термінів реалізації проекту.	V-модель зазвичай використовується в проектах, де вимоги чітко визначені і стабільні, і де систематичний і комплексний підхід до тестування є критично важливим, наприклад, в галузях зі строгими регуляторними вимогами, таких як охорона здоров'я та аерокосмічна промисловість.
3.	**Scrum**	- *Ітеративний підхід:* Скрам дотримується ітеративного підходу до розробки, що забезпечує гнучкість та адаптивність до мінливих вимог. Гнучкий малоконтрольований метод, який передбачає постійні оновлення. - *Командна співпраця:* Скрам сприяє ефективній командній співпраці, комунікації та самоорганізації, що призводить до підвищення продуктивності та інновацій. Прийняття рішень знаходиться в руках команди. - *Підвищена прозорість:* Скрам робить акцент на прозорості завдяки регулярному спілкуванню, оглядам спринтів та щоденним стенд-ап зустрічам, завдяки чому всі члени команди отримують інформацію про хід виконання проекту. - *Швидке реагування:* Короткі спринти дають змогу швидко адаптуватися до змін і вимог клієнта. - *Участь замовника:* Замовник активно залучений до процесу розробки, що сприяє точній реалізації вимог.	- *Складність:* Скрам може бути складним у впровадженні і потребує досвідчених скрам-майстрів та членів команди для ефективного дотримання фреймворку. - *Управління залежностями:* Скрам може зіткнутися з проблемами в управлінні залежностями між завданнями або командами, що може призвести до затримок або ускладнень в процесі розробки. Ефективна координація та комунікація є важливими для усунення цього недоліку. - *Надмірна увага до зустрічей:* Різні церемонії Скраму, такі як щоденні стендапи, планування спринту, огляд та ретроспективні зустрічі, можуть забирати багато часу та потенційно порушувати фокус команди. - Не підходить для великих проєктів. Scrum орієнтований на роботу групами з 6 — 9 спеціалістів. Якщо учасників значно більше, методологія починає збоїти.	Скрам зазвичай використовується в проектах з розробки програмного забезпечення для поступової реалізації та ефективного реагування на мінливі вимоги. Розробка продуктів: Скрам також підходить для проектів з розробки продуктів, які вимагають постійного зворотного зв'язку, ітерацій та швидкої доставки функцій. Скрам широко використовується в проектах з розробки програмного забезпечення для створення додатків, веб-сайтів та цифрових продуктів. А також принципи Скрам застосовуються в маркетингових проектах, які включають планування, проведення та оптимізацію кампаній.

4.	**Kanban**	- *Візуальний робочий процес:* Канбан забезпечує візуальне представлення робочого процесу, що дозволяє легко виявляти вузькі місця і визначати пріоритетність завдань. - *Постійне вдосконалення:* Канбан заохочує постійне вдосконалення за допомогою циклів зворотного зв'язку, що дозволяє командам оптимізувати свої процеси з плином часу. - *Гнучкість і адаптація:* Легко адаптуватися до змін у потребах клієнта або ринку. - *Зменшення затримок:* Ліміти в процесі знижують затримки і прискорюють виконання завдань.	- *Обмежене планування:* Орієнтація Канбану на безперервний потік і гнучкість може призвести до проблем у довгостроковому плануванні та прогнозуванні, що ускладнює встановлення чітких термінів і етапів проекту. - *Управління залежностями:* Канбан може не справлятися зі складними взаємозалежностями між завданнями або командами, що потенційно може призвести до проблем з координацією та затримок у робочому процесі.	- *Проекти технічного обслуговування:* Канбан підходить для поточних проектів технічного обслуговування, де завдання потрібно розставляти за пріоритетами, виходячи з терміновості та потреб клієнтів. - *Команди підтримки:* Канбан може бути ефективним для команд підтримки, які керують вхідними запитами та інцидентами, допомагаючи їм оптимізувати робочий процес та ефективно реагувати на них. - *Розробка програмного забезпечення:* Канбан часто застосовується в проектах з розробки програмного забезпечення для управління робочими процесами, відстеження завдань і візуалізації прогресу в розробці. - *Виробництво:* Принципи Канбан використовуються у виробничих процесах для управління запасами, контролю виробництва та оптимізації ефективності робочих процесів.
5.	**Crystal Methods**	- *Гнучкість:* Crystal Methods пропонує гнучкий підхід, який можна адаптувати до конкретних потреб проекту або команди. - *Зосередженість на людях:* Crystal Methods підкреслює важливість командної роботи, комунікації та співпраці, що призводить до підвищення морального духу та продуктивності. - *Швидша доставка* - фреймворк дозволяє команді швидше доставляти робоче програмне забезпечення, що може допомогти отримати конкурентну перевагу на ринку. - Команди мають багато автономії, щоб працювати так, як вони вважають найбільш ефективним. - *Підвищення задоволеності клієнтів* - фреймворк сприяє залученню клієнтів, дозволяючи команді створювати продукти, які відповідають потребам клієнтів, що призводить до підвищення рівня задоволеності клієнтів.	- *Складність:* Кристалічні методи можуть бути складними у впровадженні і вимагають глибокого розуміння методології, що може стати проблемою для деяких команд. - *Масштабованість:* Кристалічні методи можуть зіткнутися з проблемами масштабування для великих і складних проектів, оскільки ефективність методології може знизитися в таких сценаріях. - *Брак передбачуваності* - акцент фреймворку на адаптивності та гнучкості може призвести до браку передбачуваності, що ускладнює планування та оцінку термінів і бюджетів проєктів. - *Брак документації* - акцент на комунікації та співпраці може призвести до браку документації, що ускладнює відстеження прогресу та ведення обліку прийнятих рішень. - *Залежність від досвіду команди* - фреймворк значною мірою покладається на досвід і навички команди розробників, що може бути неприйнятним для команд з обмеженим досвідом або знаннями. - *Брак чіткості щодо ролей та обов'язків* - акцент фреймворку на самоорганізацію команд може призвести до браку чіткості щодо ролей та обов'язків, що може призвести до плутанини та втрати фокусу.	Crystal Methods може підходити для малих і середніх проектів у різних галузях, де гнучкість і командна співпраця є важливими. *Веб-розробка:* Crystal Methods можна використовувати у проектах з веб-розробки для покращення командної комунікації та координації, що призводить до ефективної розробки веб-сайтів та веб-додатків. *Стартапи:* У стартап-середовищі, де гнучкість та адаптивність є важливими, Crystal Methods може допомогти командам долати невизначеності та ітеративно досягати цілей проекту.

6.	**Extreme Programming**	- *Залучення клієнтів:* XP робить сильний акцент на залученні клієнтів та зворотному зв'язку протягом усього процесу розробки, що призводить до підвищення задоволеності клієнтів. - *Безперервне тестування:* XP виступає за безперервне тестування, що забезпечує високу якість коду та раннє виявлення помилок. - *Комунікація:* В XP процес комунікації простий, надійний і досить прозорий. Кожен з членів команди залежить одне від одного і ділиться знаннями всередині команди, що означає, що всі знають про обов'язки одне одного. - *Простота:* Оскільки сам етап комунікації починається з простого і прозорого підходу, простота гарантується на всіх інших етапах. Більш того, в цьому контексті простота відноситься до реалізації підходу, за якого ви скорочуєте все неважливе і включаєте тільки необхідну інформацію. - *Зворотний зв'язок:* Завдяки зворотному зв'язку легше відстежити області, які можна поліпшити поряд з модифікацією застосовуваних в ній процесів, щоб забезпечити якість продукції.	- *Ресурсоємність:* XP може вимагати значних ресурсів і відданості від членів команди, що робить його складним для впровадження в організаціях з обмеженими ресурсами. - *Крива навчання:* Впровадження XP може вимагати від членів команди стрімкої кривої навчання, особливо якщо вони не знайомі з його практиками та принципами. - Потрібні часті зустрічі з розробки, що збільшує загальні витрати. - Необхідність надмірних змін у розробці. - Майбутні можливості і результати точно не відомі.	Розробка програмного забезпечення: XP широко використовується у проектах з розробки програмного забезпечення, особливо у тих, де вимоги швидко змінюються і є потреба у частих ітераціях та циклах зворотного зв'язку.

Завдання 2. Beet Sprout

2.1 Чому з'явився Agile-маніфест? Які проблеми він мав вирішити і чи це вдалося?

Маніфест безперечно став революційним документом. Він був написаний як відповідь на обмеження та проблеми, пов'язані з традиційними методологіями розробки програмного забезпечення, такими як Waterfall. Автори маніфесту зійшлися в думках про основну проблему: компанії були настільки зосереджені на надмірному плануванні та документуванні своїх циклів розробки програмного забезпечення, що забули про головне — про те, що потрібно приносити радість клієнтам. Треба було це змінити. Треба було звернути увагу на те, що необхідний більш гнучкий, спільний та адаптивний підхід до розробки програмного забезпечення. Це повинно було вирішити такі проблеми:
- Швидка зміна вимог: Традиційні методології не намагалися пристосуватися до мінливих вимог у динамічному бізнес-середовищі.
- Відсутність залучення клієнтів: Клієнти часто не брали активної участі в процесі розробки, що призводило до розбіжностей між кінцевим продуктом та очікуваннями клієнтів.
- Нездатність реагувати на відгуки: Традиційні методології не наголошували на ітеративній розробці та швидких ітераціях на основі зворотного зв'язку, що заважало адаптуватися до мінливих потреб.

Маніфест Agile спрямований на просування таких цінностей, як індивідуальна взаємодія, реагування на зміни, співпраця та залучення клієнтів, щоб вирішити ці проблеми та сприяти більш ефективному та результативному процесу розробки програмного забезпечення.

Метою Маніфесту Agile є просування набору керівних принципів для розробки програмного забезпечення, які визначають пріоритети:
- Люди та співпраця важливіші за процеси та інструменти
- Працюючий продукт важливіший за вичерпну документацію
- Співпраця із замовником важливіша за обговорення умов контракту
- Готовність до змін важливіша за дотримання плану

Зосереджуючись на цих цінностях та принципах, Agile Маніфест мав на меті покращити ефективність та результативність процесів розробки програмного забезпечення, підвищити рівень задоволеності клієнтів та дати можливість командам адаптуватися до мінливих вимог та пріоритетів.

Маніфест Agile мав значний вплив на індустрію розробки програмного забезпечення з моменту його створення у 2001 році. Мета маніфесту Agile - знайти заміну старим методологіям та процесам управління проектами, які показали свою непрацездатність у сучасних проектах. Цей короткий і виразний документ назавжди змінив розробку програмного забезпечення. За майже два десятиліття, що пройшли з моменту його створення, його слова були прийняті (в тій чи іншій мірі) величезною кількістю людей, команд і компаній.
З роками Agile-методології набули широкого розповсюдження і довели свою ефективність у вирішенні багатьох проблем, з якими стикаються традиційні підходи до розробки. Загалом, можна сказати, що програмісти значною мірою досягли своєї мети, написавши Маніфест Agile, оскільки принципи та практики Agile продовжують сприйматися та впроваджуватися організаціями по всьому світу для підвищення якості, швидкості та адаптивності процесів розробки програмного забезпечення.

Завдання 3. Mighty Beet

Я засновниця стартапу і планую випустити на ринок мобільний застосунок для обміну світлинами котиків. Котики – це моя слабкість. І маю

переконання, що нас таких котолюбів дуже багато. І було б чудово мати можливість обмінюватися цією мімішністю один з одним по всьому світу. Впевнена, що це зробить світ добрішим та кращим.

Маючи дуже сентиментальну натуру і надихаючись безмірним обожнюванням та любов'ю до мого котика, названо мій застосунок буде на його честь - **MASIK**.

В мене буде невелика команда і для процесу розробки свого застосунку я обрала беззаперечливо гнучку методологію, а саме гібридну методологію **Scrumban**. Я зробила саме такий вибір, бо Scrumban об'єднав у себе найкраще з таких ефективних методологій як Scrum и Kanban. Його буде легше впровадити, аніж Scrum, наприклад, але це все одно вимагатиме зусиль, це факт. Процес Scrumban вільніший і більше схожий на Kanban. В результаті моя команда зможе навчатися та адаптуватися до нього швидше. Тобто робота вже зі старту буде більш ефективною та зрозумілою. Перш за все, Scrumban поєднує гнучкість Kanban зі структурою Scrum, дозволяючи стартапу адаптуватися до мінливих вимог, зберігаючи при цьому рівень необхідної передбачуваності. Для стартапу пріоритетними є швидкість виходу на ринок, швидкі ітерації та швидка адаптація до мінливих ринкових умов. Я вважаю, що саме ця обрана модель здатна забезпечити швидкість і маневреність, необхідні для стартапу, де ми будемо швидко приймати рішення і зосереджуватися на результатах. Scrumban забезпечує ітеративну розробку, швидкі ітерації, отримання зворотнього зв'язку та постійне вдосконалення, що має вирішальне значення для мене як засновника стартапу, бо я прагну ітеративно створювати та вдосконалювати свій веб-додаток, щоб задовольнити потреби користувачів. В результаті ми маємо свободу та гнучкість Kanban, що частково врівноважується правилами Scrum.

У Scrumban зазвичай визначають чотири ключові аспекти в процесі — це:
- тригер планування;
- дошка Kanban;
- ліміти часу для незавершених завдань;
- планування в довгостроковій перспективі.

Важливою для розробки мого проекту є візуалізація процесу. І для цього ми будемо використовувати Канбан дошку, де можна буде побачити, які перед нами стоять задачі і оцінити загальну картину цих завдань. Візуальне представлення завдань і робочих процесів полегшує відстеження прогресу, виявлення вузьких місць і розстановку пріоритетів у функціях, пов'язаних з обміном фотографіями.

Також важливо, що у Scrumban є ліміти на кількість робочих завдань. Цей показник залежить від учасників команди. Це дасть змогу кожному співробітнику зосередитися на конкретному елементі проєкту, що позитивно вплине на індивідуальну та загальну ефективність — кожен учасник робить свою роботу, за яку й відповідає.

В той же час організуємо роботу в спринтах, в яких зможемо гнучко розподіляти завдання та коригувати завантаження щодо ситуації. Спринт планується на підставі показників минулих результатів, що дає змогу правильно визначити початок наступної ітерації. (Це щодо тригеру планування) Зі скраму ми візьмемо проводити традиційні стендапи, спринт-рев'ю та інші настановні зустрічі, які допомагають краще координувати дії. Команда гарантовано вносить покращення, просуваючи свій робочий процес – постійне покращення. Це для меня також надважливо. І звісно важлива постійна взаємодія з замовником проекту, тобто зі мною.
Для мого стартапу з випуску мобільного застосунку адаптація до змін є також необхідною и важливою, а такий підхід до управління процесом дозволяє швидко реагувати на вимоги, що змінюються.

Scrumban — це інноваційний Agile-метод і я вважаю, що він буде найдоцільнішим для мого стартапу, бо дає змогу працювати над проєктом ефективніше та справді розв'язувати поставлені завдання, а не просто концентруватися на них. Гібридний підхід допомагає швидше визначати ключові проблеми та знаходити шляхи їхнього вирішення, краще планувати та контролювати навантаження команди, швидко реагувати на зміни.