**1.Сравните между собой все изученные методологии. Сделайте таблицу-сравнение этих методологий.**

В **таблице** приведены отличия этих методологий от традиционных подходов разработки.

| **Характеристики процесса создания продукта** | **Традиционные подходы (каскадная)** | **Гибкие методологии (Agile)** |
| --- | --- | --- |
| Подход | Прогнозирующий | Адаптивный |
| Критерии успеха | Следование плану | Ценность для бизнеса |
| Риски | Определены | Не определены |
| Контроль | Легко контролировать | Зависит от профессионального уровня специалистов |
| Заказчики | Низкая вовлеченность | Высокая вовлеченность |
| Документация | Детальная с начала проекта | Доработка по мере развития проекта |
| Требования | Известны заранее, стабильны | Легко изменяемы, не всегда известны заранее |
| Команда проекта | Включение новых специалистов на любом этапе | Опытные специалисты, стабильный состав |
| Рефакторинг | Дорого | Недорого |

**2. Определите, какая из методологий, по вашему мнению, больше всего подойдет для решения поставленной задачи и опишите, почему.**

**Задача: вам сообщили, что компания получила предложение заняться разработкой мобильного приложения для крупного банка.**

Я выбрала бы методологию Agile для решения данной задачи, по сравнению с традиционной методологией, например, каскадной. В силу того, что рынок мобильных приложений в банковском секторе быстро развивается, то при разработки продукта необходимо с заказчиком оговаривать сроки разработки, тестирования, обязательного версионирования продукта, те. итеративных скоростных релизов раз в день (например, внедрении новой фичи и т.д), список необходимых требований, например, какие опции должны быть включены в продукт, какой дизайн.

Если заказчик попросит сделать мобильное приложение с минимальными функциями и без дальнейшей разработки по новым релизам, то можно воспользоваться Kanban. Продакт менеджер со своей командой определят общий список задач и расформируют их на разделы: что нужно сделать, что в процессе и что уже готово. Но в практике такое редко встречается. Все же для разработки мобильного приложения для банка лучше использовать scrum. Заранее формируется команда из разных ролей, например, разработчик, devops, дизайнер, аналитик, продакт-менеджер, скрам-мастер. Весть процесс делится на спринты, обычно одно-недельные и каждый день скрам-мастер проводит с командой 15-минутный стенд-ап, чтобы команда синхронизировалась и понмал, что и когда они делают, а также какие трудности возникают, чтобы можно было моментально их решить, например, брейнштормом или обращением к эксперту, кто ранее решал аналогичные проблемы. В конце каждого спринта проводится ретро, на котором участники делятся что у них хорошо получилось, а что плохо для того, чтобы улучшить в следующем спринте, и бэклог несделанных задач переносится на следующую итерацию. Также в конце каждой итерации участники могут продемонстрировать как работает разработанная ими функциональность в тестовой среде. В целом, такой процесс выглядит непрерывным, гладким, что позволяет команде понимать что происходит с продуктом на каждой ветке ее разработки, а также сокращается время переработок участников и траты времени впустую, чтобы люди не выгорали.

**3. Черновик плана по внедрению выбранной вами методологии.**

10.09.2022. Сделать родмап проекта - определить:

* артефакты проекта (какие технические требования, аналитика, приоритет, условия приемки)
* сроки выполнения проекта (начало, спринты, тестирование, релиз-передача заказчику)
* всех участников проекта.

Итак, 29 августа стартует проект. Дата сдачи заказчику: 15.09. В команде 1 продакт-менеджер, 1 проджект-менеджер, 2 разработчика, 1 девопс, 2 тестировщика, 1 дизайнер и 1 технический писатель (если заказчик закажет приложение на 23 международных языках, то возьмем в команду 3 инженеров по локализации).

**29.08.2022** скрим-митинг, на котором менеджер проговаривает с командой всю информацию -сроки, объем, условия приемки, где и какая информация в каком воркспейсе будет находиться для прозрачности, фиксирование кто какую роль выполняет и в какие сроки. Договорились одно-недельных спринтах.

**30.08** - начало 1-го спринта. Поделили задачи, кто какую фичу разрабатывает, кто в какой среде тестирует.

**31.08** - стендап 15 мин. разработчики делятся с какими трудностями столкнулись. Подключили девопса для исправления.

**митинги…разработка/доработка/документирование**

**02.09** - передача реализованного функционала на тестирование- нахождение багов- багфикс разработчиками-регресс до минимизации количества багов. В идеале конечно отладить тестирование сразу же после того, как разработчик выкатит более-менее рабочую фичу.

**05.09 митинги…разработка/доработка/документирование.**

**12-14.09.** тестирование, багфиксы. Возможно заказчик что-то еще попросит сделать, но по договоренности весь скоуп работ и сроки зафиксированы. Новые фичи можно либо в следующем релизе выкатить по согласованию с заказчиком, либо внедрить до текущего релиза, но это обойдется дорого, тк команде потребуется в выходные работать.

**15.09.** Фиксируем готовность продукта текущей версии и передаем заказчику. Ожидаем фидбек по улучшению и т.д. Поздравляем команду с успешным выпуском.

**23.09**. Празднуем всей командой релиз -тимбилдинг.