**Краткое описание основной концепции разрабатываемого проекта**

Проект веб-приложения “Domus” представляет собой инновационный маркетплейс для продажи NFT (неподдельных токенов). “Domus” предоставляет уникальную возможность создать и развивать свой собственный NFT бизнес, обеспеченный реальными активами.

“Domus” стремится создать инновационную платформу, где цифровые творческие работы приобретают новый уровень ценности и демократизируется доступ к миру NFT.

**Список требований в формате product backlog**

1.Регистрация и аутентификация:

Реализовать систему регистрации и входа для пользователей

2.Профиль для пользователя

Создать страницу профиля пользователя

3.Каталог NFT

Разработать страницу с каталогом NFT

4.Страницы продуктов NFT

Предоставить страницы с подробной информацией о каждом NFT

5.Оформление заказа

Реализовать удобный процесс оформления заказа с возможностью выбора способа оплаты

6.Интеграция с платёжными системами

Обеспечить интеграцию с популярными платежными системами

7.Кошелек криптовалюты

Реализовать встроенный кошелек для криптовалют для удобной оплаты и получения средств от продаж NFT

8.Интерактивность

Реализация функций взаимодействия с NFT

9.Локализация

Подготовка веб-приложения для локализации на различные языки

10.Тестирование

Разработка тестов для проверки функционала и корректности работы веб-приложения

11.Документация

Написание документации

**Список пользовательских историй из product backlog**

1.Как пользователь, я хочу иметь возможность регистрации

Как пользователь, я должен иметь возможность заполнить форму регистрации и отправить ее

2.Как пользователь, я хочу видеть продукты NFT

Как пользователь, я должен видеть все продукты NFT

3.Как пользователь, я хочу оформить заказ

Как пользователь, я должен иметь возможность оформить заказ

Как пользователь, я должен видеть, что мой заказ оформлен и оплачен

4.Как пользователь, я хочу иметь кошелек криптовалюты

Как пользователь, я должен видеть свой встроенный кошелек криптовалюты

5.Как пользователь, я хочу взаимодействовать с продуктом NFT

Как пользователь, я должен иметь кнопку для выполнения действия с продуктом NFT

**Release1/Sprint1: разработка первой версии веб-приложения Domus**

Задача: регистрация и аутентификация

Подзадача: реализовать систему регистрации и входа для пользователей

Задача: профиль для пользователя

Подзадача: создать страницу профиля пользователя

Задача: каталог NFT

Подзадача: разработать страницу с каталогом NFT

Задача: страницы продуктов NFT

Подзадача: предоставить страницы с подробной информацией о каждом NFT

Задача: оформление заказа

Подзадача: реализовать удобный процесс оформления заказа с возможностью выбора способа оплаты

Задача: интеграция с платёжными системами

Подзадача: обеспечить интеграцию с популярными платежными системами

Задача: кошелек криптовалюты

Подзадача: реализовать встроенный кошелек для криптовалют для удобной оплаты и получения средств от продаж NFT

Задача: интерактивность

Подзадача: реализация функций взаимодействия с NFT

Задача: локализация

Подзадача: подготовка веб-приложения для локализации на различные языки

Задача: тестирование

Подзадача: разработка тестов для проверки функционала и корректности работы веб-приложения

Задача: документация

Подзадача: написание документации

**Вопросы**

1. Что такое жизненный цикл разработки программного обеспечения?

Жизненный цикл разработки программного обеспечения – это период времени, который начинается с момента принятия решения о необходимости создания ПО и заканчивается в момент полного его изъятия из эксплуатации.

1. Какие модели жизненного цикла разработки ПО вы знаете?

Каскадная модель.

Инкрементная модель.

Спиральная модель.

Модель V-образа.

Модель гибкой разработки.

3. Какими способами можно организовать общение с заказчиком?

Общение с заказчиком можно организовать через встречи, видеоконференции, электронную почту, обмен документами, создание прототипов, использование систем отслеживания ошибок и т.д.

4. С какой целью выполняется планирование разработки ПО, составление технического задания?

Планирование разработки ПО и составление технического задания проводится для определения объема работ, сроков выполнения, распределения ресурсов и создания основы для последующего процесса разработки.

5. Поясните, как выполняется детальный анализ предметной области и принимается окончательное решения о необходимости создания ПО.

Детальный анализ предметной области включает в себя изучение требований заказчика, выявление проблем и определение, как программное решение может их решить. Окончательное решение о создании ПО принимается на основе оценки выгод и затрат.

6. Перечислите технологии быстрой разработки программного обеспечения

Технологии быстрой разработки включают в себя RAD (Rapid Application Development), Agile (гибкие методологии разработки, такие как Scrum и Kanban), XP (Extreme Programming) и другие.

7. Что такое технология экстремального программирования?

Технология экстремального программирования (XP) — это методология разработки программного обеспечения, акцентирующая внимание на быстром реагировании на изменения, тесном взаимодействии с заказчиком и тестировании.

8. Что такое SCRUM-технология?

Scrum — это гибкая методология разработки ПО, основанная на итеративном и инкрементальном подходах. Основные элементы включают в себя итерации (спринты), роли (Product Owner, Scrum Master, Team), и постоянную обратную связь.

9. Что такое Kanban-технология?

Kanban — это метод управления разработкой, при котором акцент делается на визуализации процесса работы, ограничении рабочего запаса и непрерывной оптимизации производства.

10.Какие преимущества и недостатки технологий быстрой разработки программного обеспечения?

Преимущества: быстрые результаты, легкость внесения изменений. Недостатки: не всегда подходит для крупных проектов, требует высокой вовлеченности команды.

11.Как организована коллективная работа над проектом при использовании технологий быстрой разработки?

Преимущества: быстрые результаты, легкость внесения изменений. Недостатки: не всегда подходит для крупных проектов, требует высокой вовлеченности команды.

12.Назначение диаграммы вариантов использования.

Диаграмма вариантов использования описывает взаимодействие пользователя с системой, позволяя понять, как система будет использоваться в реальных сценариях.

13.Назначение этапа внедрение и сопровождение программных продуктов.

Этап внедрения включает в себя развертывание программного продукта на живую среду. Сопровождение включает в себя обеспечение корректной работы продукта, устранение ошибок и обновление.

14.Назовите основные задачи, решаемые на этапе внедрения.

адачи включают в себя установку программы на серверы, обучение пользователей, конфигурирование системы под нужды заказчика.

15.Как выполняется процесс устранения ошибок на этапе внедрения.

Обнаружение ошибок:

Ошибки могут быть обнаружены различными способами, такими как тестирование продукта, обратная связь от конечных пользователей, мониторинг работы системы в реальной среде.

Логирование ошибок:

Выявленные ошибки фиксируются в виде логов или отчетов, содержащих информацию о характере ошибки, условиях ее возникновения и других сведениях.

Приоритизация ошибок:

Ошибки классифицируются по их важности и воздействию на работу системы. Определяется, какие из них требуют немедленного внимания, а какие могут быть отложены.

Исправление ошибок:

Разработчики приступают к исправлению обнаруженных ошибок. Это включает в себя изменение исходного кода, проведение тестирования исправленной версии и подготовку к выпуску обновления.

Тестирование исправлений:

После внесения изменений продукт проходит повторное тестирование, чтобы убедиться, что исправления успешно устранили обнаруженные ошибки и не вызвали новых проблем.

Релиз обновления:

Исправленная версия программы выпускается в виде обновления. Это может включать в себя предоставление пользователям новой версии программного обеспечения или внесение изменений в онлайн-сервис.

Обратная связь и мониторинг:

После релиза обновления важно следить за обратной связью от пользователей и мониторить работу системы в реальной среде. Это помогает выявить возможные новые проблемы или подтвердить успешность устранения ошибок.

Документирование:

Все шаги по устранению ошибок и внесению изменений должны быть документированы. Это помогает команде и будущим разработчикам лучше понимать историю изменений в продукте.

16.Какие решения могут приниматься при выявлении ошибок на этапе внедрения.

Исправление ошибок:

Разработчики могут провести анализ и внести необходимые изменения в исходный код для устранения обнаруженных ошибок.

Выпуск патчей или обновлений:

Если ошибки требуют немедленного внимания, команда может решить выпустить патчи или обновления для пользователей, чтобы исправить проблемы.

Переработка функциональности:

В случае, если обнаруживается серьезная проблема в определенной функциональности, разработчики могут принять решение о переработке этой части продукта.

Откат к предыдущей версии:

Если новая версия программы вызывает слишком много проблем, команда может решить временно откатиться к предыдущей стабильной версии.

Внесение изменений в документацию:

Если ошибка связана с документацией, требуется ее исправление. Это может включать в себя обновление руководств пользователя, инструкций по установке и т.д.

Уточнение требований:

Иногда обнаружившиеся ошибки могут быть следствием неполного или недостаточно четкого понимания требований. В этом случае, команда может обратиться к заказчику для уточнения требований и предоставления ясных указаний.

Улучшение процесса тестирования:

При частых обнаружениях ошибок команда может принять решение об улучшении процесса тестирования, добавлении тестовых сценариев, автоматизации тестирования и т.д.

Обучение пользователей:

Если ошибка связана с неправильным использованием продукта пользователями, может потребоваться обучение или предоставление дополнительной документации.

17.Назначение документирования программного обеспечения.

Понимание и улучшение кода:

Документация помогает разработчикам понимать структуру, логику и особенности кода. Это особенно важно при работе с чужим кодом или при возвращении к собственному коду после длительного времени.

Облегчение сопровождения:

Документация упрощает задачу сопровождения и обновления программного обеспечения. Сведения о том, какие изменения были внесены и почему, помогают новым разработчикам и текущей команде поддержки быстро ориентироваться в коде.

Управление проектом:

Документация играет важную роль в управлении проектом. Она включает в себя планы, расписания, отчеты о статусе, анализ требований, архитектурные документы и др. Эти документы обеспечивают прозрачность в процессе разработки и помогают принимать информированные решения.

Согласование с требованиями:

Документация описывает требования к программному продукту, включая функциональные и нефункциональные характеристики. Это позволяет разработчикам и заказчикам согласовывать ожидания и обеспечивать соответствие результатов разработки требованиям.

Тестирование и отладка:

Документация о том, как должен работать программный продукт, служит основой для создания тестовых сценариев. Тестировщики могут использовать документацию для проверки соответствия продукта требованиям и выявления ошибок.

Обучение пользователей:

Документация, включая руководства пользователя, помогает конечным пользователям понять, как использовать программный продукт. Это особенно важно для сложных или многозадачных систем.

Безопасность и конфиденциальность:

В документации могут быть указаны меры безопасности и требования по обеспечению конфиденциальности данных. Это важно для того, чтобы предотвратить возможные угрозы безопасности.

Создание базы знаний:

Документация становится основой для создания базы знаний в организации. Это улучшает передачу знаний между членами команды и обеспечивает непрерывность в случае изменения состава команды.