Гайд для команды Frontend

1-ы месяц: реализация MVP, можно не полностью реализовать стили и эффекты и опустить второстепенный функционал.

1-я неделя:

Задача 1. Инициализация рабочего пространства

- создать репозиторий
- инициализировать структуру файлов (решаем использовать FSD в проекте или нет, но как минимум нужно разделять компоненты, ui-kit и вспомогательный код такой как сервисы, утилиты, хуки, контекст)
- настроить сборку (webpack\vite\rollup\CRA по выбору, SCSS применять обязательно)
- настроить тесты (jest)
- установить и настроить storybook (в .storybook/preview.js добавить общие с проектом зависимости)
- установить и настроить инструментарий (eslint, prettier, stylelint, husky + lint-staged)

Задача 2. Подготовка базового кода (до получения макета, важно именование и расположение, содержание пока не важно)

- определить переменные окружения
- определить базовые константы в коде
- определить SCSS (и если надо CSS) переменные
- разметить структуру под будущий код, выделить фичи если работаем по FSD
- в ui-kit выделить как минимум типографику, формы, кнопки, базовые контейнеры, завести в истории
- в компоненты app, header, footer и layout, завести в истории

- в сервисы базовый API, если планируется работа с localStorage это тоже сюда, нельзя в компонентах напрямую работать с окружением для переиспользуемости кода
- если используется роутер, то определяем общую структуру маршрутов, подключаем в app и создаем компоненты под будущие страницы. Layout может как обрамлять страницы если общий для всех, так и быть верхнеуровневым контейнером страницы если нужна вариативность.
- если будем использовать глобальный менеджер состояний, то настраиваем и подключаем его, создаем в структуре требуемые под него файлы.

> 💡 РЕВЬЮ: проверка базовой структуры проекта, тулсет настроен, сборка работает

2-я и 3-я неделя, задачи выполняются параллельно по мере поступления требуемых вводных:

Задача 3. Начинаем реализацию с UI-kit (уже должны быть вайрфреймы или черновик макета)

- создаем в структуре файлы под требуемые компоненты по макетам
- по мере готовности макета реализуем компоненты ui-kit в storybook, тестируем их и сверяемся с дизайнерами по реализации (могут быть правки)
- по готовности нужных компонентов в ui-kit начинаем накидывать компоненты (секции и блоки страниц), компонент начинаем только когда готовы все ui-kit элементы для него (не хардкодим! переиспользуем код)
- по готовности нужных компонент собираем страницы, данные мокаем через сервисы, где вместо запроса к API пока для отладки сразу возвращаем данные (Promise.resolve)
- Важно: собранный storybook опубликовать на gh-pages или на сервере проекта чтобы к нему был доступ у дизайнеров, тестировщиков и других участников команды.
- Совет: Разделяйте отображение и бизнес логику (если проще: в одном компоненте верстка, в другом подключение к данным и обработчики событий,

эффектов и т.д.)

Задача 4. Интеграция с сервером (требуется OpenAPI\Swagger контракт с бэкендом)

- начинать интеграцию нужно с авторизации если она есть
- обработку данных для соответствия требования фронтенда лучше проводить на уровне API-клиента, чтобы в компонент данные поступали уже в нужном виде.
- реализуем методы API по мере готовности на бекенде. Проверяем сначала в Postman и потом уже через код нашего API клиента. Можно автоматизировать через jest тестами.

> 💡 РЕВЬЮ: проверка базовой реализации, есть сторибук и в нем как минимум ui-kit, даже если еще нет интеграции, то данные не захардкожены, а вынесены в сервисы

4-я неделя:

Задача 5. Собираем все вместе

- Настраиваем сборку и деплой на сервер (деплоит бэк, мы даем исходные данные и по возможности создаем Dockerfile в своем репозитории)
- по мере готовности API и страниц можно подключать данные к страницам. Подключение данных без глобального менеджера состояний делаем на уровне страницы, а с глобальным менеджером на уровне экшенов. Все методы работы с окружением также спускаем через пропсы.
- Финализируем внешний вид и функциональность приложения
- Пишем интеграционные тесты для проверки страниц в сборе (с моками)
- Фиксим ошибки по баг-репортам тестирования

> 💡 РЕВЬЮ: проверка опубликованного на сервере приложения

2-й месяц: Доработка, пред-релизная подготовка

1-я и 2-я неделя:

Задача 6. Доработка всей функциональности

- реализуем в том числе второстепенную функциональность не сделанную в первый месяц
- добиваем все стили и эффекты
- Производим локальный рефакторинг в требуемых компонентах и страницах если есть возможность уменьшить кодовую базу и более эффективно переиспользовать код
- оптимизируем производительность по необходимости, добавляем ленивую загрузку страниц, добавляем кэш на API запросы чтения, оптимизируем assets
- удаляем отладочный код и вывод в консоль (если нужно оставить в критических местах, можно скрыть через console.debug или обернуть в debug и выводить только при установленной в localStorage переменной)

3-я и 4-я неделя:

Задача 7. Тестируем, Фиксим, повторяем...

- Фиксим, доделываем
- Заполняем документацию: стэк, команда, инструкция по подготовке и запуску проекта, верхнеуровневая архитектура, ключевые компоненты, конфигурация (где лежат наши переменные, константы, как ключить логирование и т.д.), что сделано\что нет, известные проблемы или потенциальные места для рефакторинга и улучшения если известны.
- Тесты нужны хотя бы на уровне снэпшотов, чтобы понимать где что поменялось в случае изменения компонент.

РЕВЬЮ: финальное

Финальное демо:

1. Собираемся, демонстрируем задеплоенное приложение, показываем работу основного функционала.
2. Объясняем какие решения применили (FSD, глобальный менеджер состояний, настройки тулсета, структура компонент) и почему
3. Рефлексия — что получилось, что не очень, какой опыт вынесли и насколько удалось освоиться с инструментарием и удержаться в рамках заданного процесса.
4. Что можно было бы улучшить, какие советы дали бы себе на старте с уже имеющимся опытом.

После завершения проекта:

В портфолио добавить ссылку на задеплоенный проект и репозиторий, приложить скриншоты по возможности.

Добавить описание проекта, какая у вас была роль и какие задачи делали. Что было интересным, вызывающим, новым. Что хотелось бы улучшить в проекте.