

Тестовое задание: Разработка приложения "Малина"

Общие сведения:

Срок выполнения: 24 часа (рекомендуемое время работы: 7-8 часов)

Тип приложения: Локальное (оффлайн) Flutter-приложение

Основной фокус: Чистый код, точное выполнение требований

Цель задания:

Разработать мобильное приложение с возможностью:

Локальной авторизации пользователей

Управления корзиной товаров

Добавления товаров через QR-код или ручной ввод

Сохранения данных между сессиями

Основные требования:

1. Система авторизации

- При первом входе с любыми валидными данными пользователь сохраняется локально
- При повторных входах система проверяет соответствие пароля
- После 3 неудачных попыток входа:
 - Пользователь и все его данные удаляются
 - Счетчик попыток не сбрасывается при перезапуске приложения
- Каждый пользователь имеет:
 - Собственные данные корзины
 - Индивидуальные настройки
 - Отдельный счетчик попыток входа

2. Работа с корзиной

- Просмотр списка добавленных товаров
- Добавление товаров двумя способами:
 - Сканирование QR-кода
 - Ручной ввод данных
- Возможность изменения количества товаров:
 - Реализовать debounce-эффект при частых нажатиях (+/-)
 - Визуальное отображение изменений должно быть мгновенным
 - Фактическое сохранение данных - с небольшой задержкой

- При выходе из профиля:
 - Возможность сохранить или удалить данные корзины

3. Валидация QR-кодов

Допустимые форматы кодов:

- Категория "Еда": коды, начинающиеся с food
 - Пример валидного: food////
 - Пример невалидного: foo/sub/name/1/desc
- Категория "Бьюти": коды, начинающиеся с beauty
 - Пример валидного: beauty12345
 - Пример невалидного: beatuy/cream


4. Технические требования

- Навигация: Использование стороннего пакета (go_router, auto_route и т.п.)
- CustomPainter:
 - На экране добавления товара реализовать розовый объект
 - Точное соответствие форме не обязательно
 - Текстовые поля и кнопки должны визуально располагаться внутри объекта
- State Management: BLoC, Provider или Riverpod
- Работа с данными:
 - Хранение данных корзины в JSON-формате
 - Использование моделирования данных (Freezed, JsonSerializable)
- Dependency Injection: Демонстрация работы с get_it
- Адаптивный дизайн:
 - Поддержка горизонтальной ориентации
 - Адаптация под планшетные устройства

5. Тестирование (опционально)

- Unit-тесты: 1-2 кейса
- Widget-тесты: 1-2 кейса
- Integration-тесты: 1-2 кейса

4. Дизайн

 **Figma-макет** (следовать комментариям и подписям).
Внимательно изучить все комментарии и подписи в дизайне

Требования к сдаче


1. Исходный код:

- Ссылка на GitHub-репозиторий (или другой git-хостинг)
- Либо ZIP-архив с полной историей коммитов

2. Документация:

- Четкое описание архитектуры решения
- Пояснения по сложным моментам реализации

3. Презентация

 **Видео-демонстрация** (Google Drive / YouTube) с пояснением функционала.

Рекомендации

- Сосредоточьтесь на качестве кода
- Соблюдайте принципы SOLID и Clean Architecture
- Оптимизируйте производительность приложения
- Обеспечьте плавность пользовательского интерфейса

Желаем успехов в выполнении задания!