Ревью индивидуального PR – Andrei Chenchik

Компоненты	Базовый вариант	Великолепный вариант	Комментарий
1. Вёрстка. Auto Layout	✅ Добавлены все констрейнты, чтобы все элементы располагались согласно макетам и корректно масштабировались на разных телефонах (на всех) без потери критической информации вроде цены и названия; ✅ Текст должен быть читаемым, без лишних констрейнтов, без конфликтов. Допущения: - Максимум один конфликт констрейнтов или приоритетов; - 2-3 отступа слегка отличаются от макета (например, если расположение на 10 пикселей отличается от макета)		Отлично!
2A. Вёрстка в коде	✅ Реализована вёрстка экранов в коде без использования Interface Builder ✅ Использованы best-practices для реализации вёрстки, такие как Auto Layout и Stack Views. Допущение: Можно использовать SnapKit или другие сторонние библиотеки для упрощения вёрстки	Реализованы множественные переиспользуемые UI компоненты там, где это возможно – 🍏	Очень крепкое понимание верстки на стеках. Есть кастомные вьюшки (рейтинг).
2B. Вёрстка в сторибордах	–		–
3. Вёрстка таблицы	✅ Корректно свёрстана ячейка; ✅ Продемонстриовано понимание принципов делегата и Data source.	Реализовано кастомное поведение для коллекций, такое как pull-to-refresh, swipe-to- delete и т.д.– 🍏	Использован DiffableDataSource
4. Верстка коллекции	✅ Ячейки свёрстаны корректно, содержимое не вылезает за границы ячейки; ✅ Корректно реализовано обновление и нет падений при обновлении таблицы; ✅ Правильно отображается содержание пустой коллекции; ✅ Пагинация в необходимых местах работает корректно — нет множественных запросов с одними и теми же параметрами, показывается «крутилка».	Кастомный layout – 🍏 Реализовано кастомное поведение для коллекций, такое как pull-to-refresh, swipe-to- delete и т.д. – 🍏	Использован DiffableDataSource. Пагинация и пустая коллекция в этом эпике избыточны – критерии засчитаны по-умолчанию
5. Соответствие макету Figma	✅ Экраны и навигация выполнены верно. Допущение: В одном-двух местах во всём проекте шрифт может быть неправильного размера, картинка криво экспортирована	Правильный размер и сам шрифт, правильные картинки и разрешение в них, правильная навигация между экранами – 🍏 «Пиксель пёрфект» попадание по дизайну – макеты и вёрстка идеально совпадают – 🍏	Верстка скорее Пиксель пеерфект, отлично!
6. UIKit, Foundation	✅ UI обновляется только в главном потоке; ✅ При сетевых запросах блокируется интерфейс; ✅ Почти не используются depricated методы (не более одного-двух раз за весь проект); ✅ Для работы с датами использован date formatter. ✅ Нет retain cycles и утечек памяти.	Когда пользователь переходит на следующий экран при множественном нажатии на кнопку или элемент интерфейса не происходит множественного открытия экрана, на который переходят. – 🍏	Отлично!
7. Работо-способность приложения	✅ Проект компилируется, допустимо минимальное количество некритичных предупреждений (не приводят к падению или некорректной работе приложения). Максимум 3 предупреждения. ✅ Нет падений при разных входных данных; ✅ Приложение не содержит критических багов, которые могут привести к падению или неработоспособности; ✅ Баги, которые возникают, не влияют на работу приложения в целом и не приводят к нарушению его логики.	Приложение не содержит никаких багов. – 🍏	Баги не найдены – будем приравнивать это к великолепному варианту (как тестировщик, я бы отметила, что нельзя утверждать что багов нет :)

Компоненты	Базовый вариант	Великолепный вариант	Комментарий
8. Архитектура. Одна из MVP, MVC, MVVM	✅ В работе компоненты архитектуры имеют правильные «ответственности». Например, не названо ViewModel то, что должно быть презентером. Небольшие кусочки кода могут остаться во вью и при этом не переехать в презентер ✅ Последовательно использует выбранную архитектуру. Экраны содержат все заявленные архитектурой элементы. ✅ Правильно называется классы, продемонстрированы на практике связи между компонентами архитектуры.	В работе используются конечные автоматы. – 🍏	В работе все еще есть места, где путаются ответственности между View/ViewModel. Их не много, поэтому пункт засчитан. Enum-ы Destination зачту за конечный автомат.
9. Работа с сетевыми запросами	✅ Реализована обработка ошибок, таких как отсутствие интернет-соединения, невалидные данные и другие. ✅ Реализованы различные типы запросов, такие как GET, POST, у некоторых эпиков PUT, DELETE		Благодаря ошибкам в моковых данных, наглядно продемонстрированы ошибки :) Отлично
10. Использование многопоточности	✅ В работе определены потенциальные race conditions и реализована защита от них. ✅ Реализована обработка ответов на сетевые запросы в фоновом потоке ✅ Реализовано правильное использование многопоточности, чтобы избежать race conditions и deadlocks.	Реализовано кэширование ответов на запросы, чтобы уменьшить количество сетевых запросов. – 🍏	Отлично!
11. Хранение данных	✅ Реализована работа с локальным хранилищем, таким как UserDefaults или Keychain.		Продемонстрировано на фейковой корзине.
12. Автотетестирование	Автотестов нет	Используются разнообразные типы тестов 🍏 Все критические участки полностью (бизнес-логика) покрыты тестами; 🍏 покрытие достаточно полное, тесты запускаются и работают быстро; 🍏 Есть скриншотные и Unit-тесты. 🍏	–
13. Чистота кода	✅ В большинстве случаев корректно названы переменные (понятно и логично), ✅ используется единый принцип именования camelCase, ✅ код сервисов вынесен в отдельные классы, ✅ нет серьёзных нарушений принципов SOLID, KISS, DRY, YAGNI; ✅ Используются паттерны, ✅ функции нормального размера с нормальными названиями; ✅ Дублирование кода сведено к минимуму; ✅ Корректно расставлены отступы, нет лишних неинформативных комментариев; ✅ Корректная работа с классами и структурами (нет ошибок с референс и велью семантикой) ✅ нет неиспользуемого кода; ❌ Не используется Forced unwrap.	Корректно названы переменные, код сервисов вынесен в отдельные классы, 🍏 Корректная работа с классами и структурами, 🍏 Использует SwiftLint 🍏	Еще остались форс анврапы, местами лишние пробелы. В целом код достаточно чистый. Предупреждений от SwiftLint по коду эпика – нет.
14. Работа с Git	✅ Названия коммитов понятны и логичны ✅ Помечена связь с тасками в таск- трекере в описании пулл-реквеста; ✅ Прописана выбранная архитектура и способ вёрстки.	Адекватный размер коммитов 🍏	Работа с гитом, ПР, с ревью – на высоком уровне. Навыки годны для работы в команде!
Подведение итогов	Не засчитан только пункт про Force unwrap	10 🍏 из 18	Работа на хорошем уровне. Автор явно может круче. Рекомендация: не спешить, быть внимательным к требованиям. Продуктовые задачи решены отлично, технические от верстки до запросов тоже отлично. С архитектурой было много спешки, но в итоге на отлично поправлено. Рекомендация: доправить оставшиеся проблемы. Вернуть работу на последнее ревью, когда будут смерджены все эпики проекта. Рекомендация придерживаться модели работы с ветками gitflow или его упрощенные версии;