1. В даному проєкті впроваджений модульний підхід, який легше підтримувати та розширювати, наприклад, порівнюючи з BDD підходом.
2. У проєкті використовуються патерни проектування, такі як Builder, DTO, Listener. З їх допомогою зменшується кількість повторюваного коду та покращується читабельність.
3. Також використовується Lombok – хоч цей фреймворк і зменшує читабельність коду та вимагає певного часу для пристосування, але при цьому значно покращує його організованість та структуру, прибирає необдхідність явного оголошення get-ерів, set-ерів та конструкторів.
4. Тести та методи розділені по папках, test та main відповідно.
5. Директорія java в папці main розділена на окремі субдиректорії:
   core, котра містить загальні java методи, які можна використовувати як для UI, так і для API тестування, ці методи мають бути загальнодоступними в спільному пакеті.
6. В папці Enum знаходяться перелічувані об'єкти. Зокрема, важливим є Enum TestCache, котрий являє собою HashMap. Оскільки в деяких тестах виникає потреба зберегти значення змінних, які потрібно використати за межами методу, що їх ініціалізує. На даний момент тести запускаються в один потік, але за рахунок того, що CacheManager створює екземпляр InheritableThreadLocal, в майбутньому це дозволить мати окремі кеші для кожного потоку, котрі не будуть перетинатись.
7. Були додані кастомні винятки, оскільки основні ризики на даному етапі пов'язані з обробкою та кастуванням JSON формату, був доданий CustomJsonProcessingException. На даний момент він передає лише повідомлення про помилку, але є можливість розширити та покращити виняток, наприклад, додати причину чи детальніше логування.
8. В папці API знаходиться вся структура API методів: основні конфігурації, конкретні методи, що стосуються АПІ колів (контролери та білдери). Окремо в процесі автоматизації виникла потреба в retry опції. Часто retry опція приносить ризики пропустити багу. Але в даній ситуації, на мою думку, це виправдано, оскільки ми викликаємо сторонній сервер, який працює нестабільно та періодично повертає помилки.
9. На рівні з папками імплементації знаходиться папка models, котра містить DTO, що відповідають основним респонсам.
10. Оскільки авторизація для апішних запитів виконується перед кожним із тестів (в BeforeMethod), то для негативних ТС, де потрібно викликати ендпоінт без авторизації, було використано кастомну анотацію, котра за рахунок listener-а викликає видалення авторизації в окремо взятих тестах (анотація @Non_Authorized).
11. В тести були додані основні перевірки - позитивні сценарії, та кілька негативних (видалення/отримання неіснуючого ордера, запит без авторизації і т.д.)
12. В ресурсах розміщено xml файл з конфігураціями для логування, використовується logback та slf4j, як одні з найбільш популярних та ефективних.
13. Для розуміння, які інструменти та технології вимагає даний проєкт, було додано README файл з посиланням на офіційну документацію та основним переліком команд.
14. Репортинг було налаштовано через Allure, оскільки це безкоштовний, популярний інструмент, який легко інтегрувати в фреймворку. Також він

забезпечує гарну візуалізацію тестових результатів (актуальний репорт прикріплено нижче)