**Проект №11 — Финальный проект.**

Привет, участник Школы 21!  
Вот он — финальный проект! В прошлых проектах ты уже изучил все основные моменты ручного тестирования, теперь мы объединим это всё вместе, поговорим про системное тестирование и составим тестовую стратегию.

**Instructions**

Напоминаем, что все отчёты по результатам выполнения заданий тебе нужно оформлять в файлах с расширением .md. Если они уже созданы, то пересоздавать или удалять их не нужно (просто отредактируй этот файл). Все созданные отчёты и файлы тебе нужно будет загрузить в папку src/ в корне проекта (обязательно в ветку *develop*).

**Contents**

1. [Chapter I](https://edu.21-school.ru/project/64096/task#chapter-i)  
   1.1. [Общая инструкция](https://edu.21-school.ru/project/64096/task#%D0%BE%D0%B1%D1%89%D0%B0%D1%8F-%D0%B8%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%86%D0%B8%D1%8F)
2. [Chapter II](https://edu.21-school.ru/project/64096/task#chapter-ii)  
   1.1. [Стратегия тестирования](https://edu.21-school.ru/project/64096/task#%D1%81%D1%82%D1%80%D0%B0%D1%82%D0%B5%D0%B3%D0%B8%D1%8F-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)  
   1.2. [Задание №1](https://edu.21-school.ru/project/64096/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-1-%D1%81%D0%BE%D1%81%D1%82%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5-%D1%82%D0%B5%D1%81%D1%82%D0%BE%D0%B2%D0%BE%D0%B9-%D1%81%D1%82%D1%80%D0%B0%D1%82%D0%B5%D0%B3%D0%B8%D0%B8)
3. [Chapter III](https://edu.21-school.ru/project/64096/task#chapter-iii)  
   2.1. [Системное тестирование](https://edu.21-school.ru/project/64096/task#%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%BE%D0%B5-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)  
   2.2. [Типы системного тестирования](https://edu.21-school.ru/project/64096/task#%D1%82%D0%B8%D0%BF%D1%8B-%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)  
   2.3. [Условия начала системного тестирования](https://edu.21-school.ru/project/64096/task#%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-%D0%BD%D0%B0%D1%87%D0%B0%D0%BB%D0%B0-%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)  
   2.4. [Подходы к системному тестированию](https://edu.21-school.ru/project/64096/task#%D0%BF%D0%BE%D0%B4%D1%85%D0%BE%D0%B4%D1%8B-%D0%BA-%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%BE%D0%BC%D1%83-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8E)  
   2.5. [Задание №2](https://edu.21-school.ru/project/64096/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-2-%D0%BF%D1%80%D0%BE%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5-%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)

**Chapter I**

**Общая инструкция**

Методология Школы 21 может быть не похожа на тот образовательный опыт, который случался с тобой ранее. Её отличает высокий уровень автономии: у тебя есть задача, ты должен её выполнить. По большей части тебе нужно будет самому добывать знания для её решения. Второй важный момент — это peer-to-peer обучение. В образовательном процессе нет менторов и экспертов, перед которыми ты защищаешь свой результат. Ты это делаешь перед таким же учащимися, как и ты сам. У них есть чек-лист, который поможет им качественно выполнить приемку вашей работы.

Роль Школы 21 заключается в том, чтобы обеспечить через последовательность заданий и оптимальный уровень поддержки такую траекторию обучения, при которой ты не только освоишь hard skills, но и научишься самообучаться.

* Не доверяй слухам и предположениям о том, как должно быть оформлено ваше решение. Этот документ является единственным источником, к которому стоит обращаться по большинству вопросов;
* твое решение будет оцениваться другими учащимися;
* подлежат оцениванию только те файлы, которые ты выложил в GIT (ветка develop, папка src);
* в твоей папке не должно быть лишних файлов — только те, что были указаны в задании;
* не забывай, что у вас есть доступ к интернету и поисковым системам;
* обсуждение заданий можно вести и в Rocket.Chat;
* будь внимателен к примерам, указанным в этом документе — они могут иметь важные детали, которые не были оговорены другим способом;
* и да пребудет с тобой Сила!

**Chapter II**

**Стратегия тестирования**

**Стратегия тестирования (test strategy)** — описание общего подхода к тестированию и целей тестирования. Правильно составленная стратегия тестирования даёт четкое представление о том, что команда тестирования будет делать для всего проекта.

Рассмотрим примерные шаги на пути ее формирования:

*Шаг 1*: Определить какие типы тестирования буду применяться на проекте и почему. На всех ли уровнях будет проводиться тестирование? Необходимо ли будет многократно проводить регрессионное тестирование? Важно ли будет тестирование производительности или безопасности? На каких платформах надо проводить тестирование? На эти и прочие подобные им вопросы стоит ответить в самом начале формулировки тестовой стратегии.

*Шаг 2*: Определить приоритеты. Чаще всего у тестировщиков нет возможности спокойно и размеренно провести все возможные тесты, так что стоит сразу определиться с приоритетами. Какие тесты писать раньше, какие автоматизировать (и автоматизировать ли вообще), где критически важно обеспечить полное покрытие и где им можно пренебречь.

*Шаг 3*: Определиться с окружением для работы.

*Шаг 4*: Работа и обязанности тестировщика. На разных проектах требования к команде тестирования и ее работникам могут отличаться, так что стоит их уточнить. Например, деплой новых версий может попадать как под компетенцию тестеров, так и быть частью работы разработчиков. Стоит обсудить эти моменты, чтобы тестировщики четко понимали что от них требуется.

*Шаг 5*: Тестовая документация на проекте. Аналогично прошлому шагу, формат тестовой документации может радикально отличаться от проекта к проекту, так что стоит обсудить и договориться что именно, когда, и в каком виде в нее включается.

*Шаг 6*: Каким образом генерируются тестовые сценарии. Условия и сценарии использования ПО также стоит уточнить, так как от этого зависит, среди прочего, какими техниками тест-дизайна стоит пользоваться при написании тест-кейсов, а также с чей перспективы рассматривать ПО.

*Шаг 7*: Порядок работы с трекером. Существует целый ряд баг-трекеров, в каждом из которых реализовано великое множество функций. Очень редко все они на деле нужны при работе с проектом, так что стоит решить каким баг-трекером и в каком виде будет пользоваться команда.

*Шаг 8*: Критерии начала и окончания тестирования. Тестировщикам необходимо понимать, когда можно браться за задачу и когда стоит отметить ее как решенную и двигаться дальше.

*Шаг 9*: Оценка качества проекта и инструменты мониторинга. Для тестировщика важно понимать метрики для оценки качества ПО, так что стоит их обсудить.

Как ты мог заметить, полноценная тестовая стратегия составляется не только одним тестировщиком. В её создании принимает участие менеджер проекта и остальная часть команды. Однако, чтобы упростить тебе задачу, мы решили дать задание на составление упрощённой версии тестовой стратегии.

**Задание №1. Составление тестовой стратегии**

Представь, что ты идёшь устраиваться на должность "Junior QA Engineer" в небольшую компанию, например, в "Acoola", но пока у них нет отдела контроля качества, потому ты можешь стать первым тестировщиком. Вычитав пару статей в интернете и проанализировав твоё резюме, HR просит тебя написать тестовую стратегию для веб-версии чата [Rocket.Chat](https://www.rocket.chat/). Также тебе дополнительно дают следующие сведения: приложение нужно протестировать полностью, это последняя версия.

В папке src мы уже подготовили для тебя шаблон тестовой стратегии (файл exercise1.doc). Осталось только лишь его заполнить! Внимательно составляй стратегию, так как она напрямую связана со следующим заданием.

*P.S.: в файле materials/instruction.md мы подготовили для тебя инструкцию, которая поможет получить персональную облачную версию чата.*

**Chapter III**

**Системное тестирование**

**Системное тестирование ПО** — это тестирование программного обеспечения, выполняемое на полной, интегрированной системе, с целью проверки соответствия системы исходным требованиям. Системное тестирование проводится методом чёрного ящика, и, следовательно, не требует знаний о внутреннем устройстве системы.

На уровень системного тестирования приходится сравнительно небольшой процент от общего количества проводимых тестов. Однако не стоит думать, что это делает его проще или менее важным! Тестирование системы требует крайней осторожности, а любая ошибка в его проведении особенно критична, так как её нахождение и исправление в среднем займет больше времени, чем в любом другом уровне тестирования.

Основная задача системного тестирования — в выявлении дефектов, связанных с работой системы в целом, таких как неверное использование ресурсов системы, непредусмотренные комбинации данных пользовательского уровня, несовместимость с окружением, непредусмотренные сценарии использования, отсутствующая или неверная функциональность, неудобство в применении и тому подобное.

Теперь давай остановимся на типах системного тестирования!🤗

**Типы системного тестирования**

Категории тестов системного тестирования:

* функциональное тестирование;
* тестирование производительности;
* нагрузочное или стрессовое тестирование;
* тестирование конфигурации;
* тестирование безопасности;
* тестирование надежности и восстановления после сбоев;
* тестирование удобства использования.

Отметим, что в ходе системного тестирования проводятся чаще всего далеко не все из перечисленных видов тестирования — конкретный их набор зависит от тестируемой системы.

Поскольку системное тестирование проводится на пользовательских интерфейсах, может показаться, что построение системы автоматизации тестирования не всегда необходимо. На самом же деле объемы данных на этом уровне таковы, что зачастую более эффективным подходом является полная или частичная автоматизация тестирования.

Стоит понимать, что перед тем, как приступить к системному тестированию, необходимо выполнить ряд условий. Давай рассмотрим их подробнее.

**Условия начала системного тестирования**

Системное тестирование проводится при следующих условиях:

* модульное и интеграционное тестирование успешно завершено;
* разработка ПО в соответствии с требованиями спецификации завершена;
* создана соответствующая тестовая среда для проведения системного тестирования.

При этом существуют 2 основых подхода к системному тестированию, давай разберемся, в чем же они состоят.

**Подходы к системному тестированию**

1. *На базе требований (requirements based)*. Для каждого требования пишутся тест-кейсы, проверяющие выполнение данного требования.
2. *На базе случаев использования (use case based)*. На основе представления о способах использования продукта создаются случаи использования системы (Use Cases). По конкретному случаю использования можно определить один или более сценариев. На проверку каждого сценария пишутся тест-кейсы, которые должны быть протестированы.

**Задание №2. Проведение системного тестирования**

Системное тестирование требует навыков, которые ты уже освоил в ходе прохождения этого курса! Всё то же функциональное тестирование, только строить мы его будем на основе случаев использования (use case based). Так как приложение находится на стадии эксплуатации, то все предыдущие уровни тестирования уже пройдены.

В файле exercise2.usecases.md опиши случаи использования веб-версии рокетчата (структуру Use Case можно подглядеть в предыдущих проектах). На основе сценариев использования составь тест-кейсы и заведи их в TestIT, выгрузи получившийся список тест-кейсов и назови файл exercise2.testcases.xlsx.

В TestIT создай тест-план и проведи функциональное системное тестирование. Результаты тестирования (результаты прогона тестов) выгрузи в формате xlsx и назови файл exercise2.results.xlsx.

*P.S.: не забудь, что части системы и техники тест-дизайна, которые используются для составления тест-кейсов, обязательно должны быть описаны в тестовой стратегии.*

**Double-check**

Перед загрузкой выполненного проекта в репозиторий перепроверь наличие всех необходимых файлов, которые требовалось создать во время его выполнения:

exercise1.doc

exercise2.usecases.md

exercise2.testcases.xlsx

exercise2.results.xlsx