Содержание

## 1. Введение

Целью составления данного Тест Плана является описание процесса тестирования клиент-серверного приложения "Портовая диспетчерская система" и проверка функциональных возможностей приложения. Документ позволяет получить представление о плановых работах по тестированию проекта. Этот документ описывает, содержит всю информацию о подходах и методологиях, ресурсах, необходимых для достижения поставленных целей.

## 2. Объект тестирования

Объектом тестирования является клиент-серверное приложение "Портовая диспетчерская система". Данное приложение предоставляет возможность быстро и легко решать проблемы распределения нагрузки между причалами и приоритетов между кораблями, позволяет избегать возникновения очередей в портах, в составе имеет клиентскую и серверную часть. Приложение предназначено для работы в средах операционных систем Linux, Windows. Для написания приложения был использован язык программирования Java. Среда разработки - IntelliJ IDEA 2019.3.2. GUI клиента и сервера реализуются с помощью графической библиотеки Swing. Сетевая часть приложения реализована с помощью класса Socket технологии java.net. Связь между клиентом и сервером реализуется с помощью протокола TCP.

Атрибуты качества:

1. Доступ ко всем операциям в одном окне;
2. Все функциональные элементы пользовательского интерфейса имеют названия, описывающие действие, которое произойдет при выборе элемента;
3. Приложение надёжно. Доступно только компаниям, работающим с ним;
4. Заявки на обслуживание кораблей будут обработаны в течении 10-20 мин;
5. Приложение характеризуется высокой производительностью;
6. Удобство в использовании;
7. Невысокая начальная стоимость системы;
8. Низкие эксплуатационные расходы;
9. Поддержка системы разработчиком;
10. Стабильность функционирования;
11. Наличие качественной документации.

## 3. Риски

Условия, которые могут повлиять на качество продукта:
1. Интернет (Работоспособность данного приложения напрямую зависит от интернета, при отсутствии интернета оно перестанет функционировать)
2. Замедление работы системы (При большом количестве кораблей обработка будет проходить до 50 с по времени (при нормальной работе 20-30с)).
3. Риски, связанные со скоростью разработки и выделенным бюджетом (при доработке или разработке следующей версии)

## 4. Аспекты тестирования

Предметы тестирования:
1. Пользовательский интерфейс. Тесты на всевозможные поля ввода-вывода, кнопки, выпадающие списки и т.д.
2. Функциональность. Тесты на все собственные функции. Цель тестирования функций - убедиться, что программное обеспечение работает должным образом и соответствует всем предполагаемым спецификациям. Если какое-либо программное обеспечение или приложение не может выполнять требования клиента или выполнять выбранные функции с точностью, это может привести к значительной потере денег и усилий. Поскольку большое количество программного обеспечения используется организациями вместе, поломка любого программного обеспечения также может стать препятствием для правильного функционирования другого программного обеспечения.
3. Клиент-серверное взаимодействие. Тесты на взаимодействие клиента и сервера: проверка доставки сообщений для обоих сторон, проверка обмена информацией.

## 5. Подходы к тестированию

Каждый тест должен соответствовать принципу FIRST и обладать следующими характеристиками:

1. Быстрота (Fast): тесты должны выполняться быстро.

2. Независимость (Independence): результаты выполнения одного теста не должны быть входными данными для другого. Все тесты должны выполняться в произвольном порядке, поскольку в противном случае при сбое одного теста каскадно “накроется” выполнение целой группы тестов.

3. Повторяемость (Repeatable): тесты должны давать одинаковые результаты независимо от среды выполнения.

4. Очевидность (Self-Validating): результатом выполнения теста должно быть булево значение.

5. Своевременность (Timely): тесты должны создаваться своевременно.

Правила формирования тестов:

1. В одном тесте проверяем только один параметр на корректность. Остальные аргументы подаем заведомо корректными.
2. Не проверяются ненулевые Handle и при этом имеющие некорректное значение т.к. это контроль памяти программистом.
3. Тесты в которых время выполнения превышает 3600 с не выполняются
4. Тесты не предполагают проверку на наличие предварительных инициализаций.

Применяемые техники тест дизайна:

1. Эквивалентное Разделение (Equivalence Partitioning)
2. Анализ Граничных Значений (Boundary Value Analysis)
3. Предугадывание ошибки (Error Guessing)
4. Причина / Следствие (Cause/Effect)

## 6. Представление результатов

### Проверка команды установки веса груза

`public boolean setWeight(Double weight);`

Данная группа тестов проверяет, что функция установки веса работает корректно.

| № теста | Имя теста | Параметры | Последовательность действий | Ожидаемый результат (pass_condition) |
|---|---|---|---|---|
| 1 | Установка нулевого веса | NULL | Передаём в параметры функции нулевой вес | result == false |
| 2 | Установка корректного веса | Любое число типа double, меньшее максимально допустимого значения | Передаём в параметры функции корректное значение | result == true |

| | | | | |
|---|---|---|---|---|
| 3 | Попытка установки буквенного веса | Любой символ типа char или строка | Попытка передачи в параметры функции некорректного типа данных | result == false |
| 4 | Попытка установки отрицательного веса | Любое отрицательное число типа double | Передача в параметры функции отрицательного числа | result == false |
| 5 | Проверка максимально допустимого веса | Максимльно допустимое число double MAX_DOUBLE_SIZE | Передаём в параметры функции максимльно допустимое число double | result == true |
| 6 | Проверка недопустимого по размеру веса | Максимльно допустимое число double +1 MAX_DOUBLE_SIZE+1 | Передаём в параметры функции недопустимое по размеру число double | result == false |

```
// 1, 2, 3, 4, 5, 6
test_check_weight(double param)
{
    bool result;
    result = setWeight(param);
    TEST_ASSERT(pass_condition);
}
```

ВСЕ ОСТАЛЬНЫЕ ФУНКЦИИ ТЕСТИРУЮТСЯ АНАЛОГИЧНО

Тесты на пользовательский интерфейс

| № теста | Имя теста | Последовательность действий | Ожидаемый результат (pass_condition) |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 7 | Проверка работоспособности кнопки «В порт» (программно не выполняется) | Войти в программу, ввести имя корабля, заполнить таблицу грузов, нажать кнопку «В порт» | Отправка корабля в порт(на сервер) |
| 8 | Проверка работоспособности выпадающего списка «Загрузка/Разгрузка» (программно не выполняется) | Войти в программу, ввести имя корабля, заполнить таблицу грузов, выбрать из выпадающего списка Загрузку или Разгрузку | Удостовериться, что программа не зависла |

ВСЕ ОСТАЛЬНЫЕ ЭЛЕМЕНТЫ ПОЛЬЗОВАТЕЛЬСКОГО ИНТРФЕЙСА ВЫПОЛНЯЮТСЯ АНАЛОГИЧНО

## 7. Вывод

Функциональные требования соблюдены в полном масштабе, все функции, которые должно выполнять разрабатываемое ПО оно выполняет. Нефункциональные требования также соблюдены: система демонстрирует свою работоспособность, удобно в сопровождении, расширяема, надёжна, показывает высокую производительность. Есть некоторые ограничения, которые в также соблюдены в полной мере.

Данный продукт обладает всеми требуемыми атрибутами качества, т.к. были проведены всевозможные тесты, большинство из которых завершились успешно. Тесты, которые не выполнились, будут учтены при разработке следующей версии данного приложения, все моменты будут доработаны и протестированы снова.