**Отчет по анализу программного проекта "Онлайн автосервис на Spring Boot"**

**a. Цель работы**

Целью данной работы является анализ программного проекта "Онлайн автосервис на Spring Boot" с точки зрения различных видов, типов, методов и уровней тестирования. В рамках работы будут рассмотрены и классифицированы подходы к тестированию, а также приведены примеры характерных дефектов для каждого вида тестирования. Итогом станет обоснование рекомендаций по проведению тестирования данного проекта.

**b. Описание программного проекта**

Проект "Онлайн автосервис на Spring Boot" представляет собой веб-приложение, предоставляющее пользователям возможность записываться на обслуживание автомобилей, управлять заказами, а также получать информацию о предоставляемых услугах. Основные компоненты проекта включают:

1. **Веб-интерфейс**: доступный пользователям интерфейс для взаимодействия с системой.
2. **Серверная часть**: реализованная на Spring Boot логика приложения, обрабатывающая запросы пользователей.
3. **База данных**: хранение информации о пользователях, заказах и услугах.
4. **API**: интерфейс для взаимодействия с другими системами и мобильными приложениями.

**c. Классификация видов, типов, методов, уровней тестирования для проекта**

**Виды тестирования**

1. **Функциональное тестирование**:
   * Тестирование функциональности веб-интерфейса.
   * Тестирование API.
2. **Нефункциональное тестирование**:
   * Производительность.
   * Безопасность.
   * Юзабилити.

**Типы тестирования**

1. **Черный ящик**:
   * Тестирование пользовательских сценариев.
   * Тестирование API без знания внутренней реализации.
2. **Белый ящик**:
   * Юнит-тестирование серверной логики.
   * Тестирование интеграций между компонентами.

**Методы тестирования**

1. **Ручное тестирование**:
   * Проверка пользовательского интерфейса.
   * Тестирование юзабилити.
2. **Автоматизированное тестирование**:
   * Юнит-тесты.
   * Интеграционные тесты.
   * Тесты производительности.

**Уровни тестирования**

1. **Модульное тестирование**:
   * Юнит-тесты для отдельных методов и классов.
2. **Интеграционное тестирование**:
   * Тесты взаимодействия между различными модулями и сервисами.
3. **Системное тестирование**:
   * Полное тестирование всей системы перед релизом.
4. **Приемочное тестирование**:
   * Проверка выполнения требований заказчика перед внедрением.

**d. Примеры дефектов, характерных для каждого вида тестирования**

**Функциональное тестирование**

1. **Тестирование веб-интерфейса**:
   * Дефект: Кнопка "Отправить" не работает.
   * Пример: Пользователь нажимает на кнопку, но форма не отправляется.
2. **Тестирование API**:
   * Дефект: Некорректный ответ на запрос.
   * Пример: Запрос на получение списка заказов возвращает пустой массив при наличии данных в базе.

**Нефункциональное тестирование**

1. **Производительность**:
   * Дефект: Время отклика сервера превышает допустимые значения.
   * Пример: При большом количестве запросов сервер начинает сильно замедляться.
2. **Безопасность**:
   * Дефект: Уязвимость к SQL-инъекциям.
   * Пример: Ввод вредоносного SQL-кода в поля ввода приводит к утечке данных.
3. **Юзабилити**:
   * Дефект: Плохая навигация по сайту.
   * Пример: Пользователи не могут быстро найти нужные разделы из-за сложной структуры меню.

**Черный ящик**

1. **Тестирование пользовательских сценариев**:
   * Дефект: Некорректная обработка ошибок.
   * Пример: При введении неверных данных в форму регистрации система не показывает сообщение об ошибке.
2. **Тестирование API**:
   * Дефект: API не обрабатывает неправильные запросы.
   * Пример: Отправка неверного формата данных приводит к сбою сервера.

**Белый ящик**

1. **Юнит-тестирование**:
   * Дефект: Метод возвращает неправильное значение.
   * Пример: Метод расчета стоимости услуг возвращает неверный результат.
2. **Интеграционное тестирование**:
   * Дефект: Неправильное взаимодействие между модулями.
   * Пример: Модуль авторизации не передает правильные токены модулю управления заказами.

**Ручное тестирование**

1. **Проверка пользовательского интерфейса**:
   * Дефект: Несоответствие дизайна.
   * Пример: Цвета и шрифты не соответствуют утвержденным макетам.
2. **Тестирование юзабилити**:
   * Дефект: Сложный процесс регистрации.
   * Пример: Пользователи бросают процесс регистрации из-за большого количества шагов.

**Автоматизированное тестирование**

1. **Юнит-тесты**:
   * Дефект: Неполное покрытие кода.
   * Пример: В критическом методе отсутствуют тесты для граничных условий.
2. **Интеграционные тесты**:
   * Дефект: Некорректная конфигурация окружения.
   * Пример: Тесты не проходят из-за неверных настроек базы данных в тестовом окружении.
3. **Тесты производительности**:
   * Дефект: Система не справляется с нагрузкой.
   * Пример: При стресс-тестировании выявлено, что сервер падает при 1000 одновременных запросах.

**e. Выводы по работе**

Анализ показал, что для проекта "Онлайн автосервис на Spring Boot" необходимо применять различные виды и уровни тестирования для обеспечения высокого качества продукта. Особое внимание следует уделить функциональному и нефункциональному тестированию, используя как ручные, так и автоматизированные методы. Примеры дефектов показывают, что ошибки могут возникать на всех уровнях тестирования, поэтому важно иметь комплексный подход к тестированию.

**f. Список использованных источников**

1. Официальная документация Spring Boot: <https://spring.io/projects/spring-boot>
2. Руководство по тестированию ПО: <https://www.istqb.org/>
3. Книги по тестированию ПО:
   * "Совершенное программное обеспечение" (Джеймс Уиттакер)
   * "Тестирование DOT COM приложений" (Роман Савин)
4. Онлайн-курсы по тестированию:
   * Coursera: Software Testing and Automation Specialization
   * Udemy: The Complete Guide to Software Testing