**Лабораторная работа №2: Классификация видов тестирования**

**Цель работы**

Целью данной лабораторной работы является анализ и применение различных видов, типов, методов и уровней тестирования к проекту "Андроид приложение для отображения расписания Университета".

**Описание программного проекта**

Проект представляет собой Android-приложение для отображения расписания занятий университета. Основные функции включают отображение текущих и предстоящих занятий, уведомления о предстоящих занятиях, поиск по расписанию и возможность интеграции с календарем Google. Приложение разработано с использованием Kotlin и использует Firebase для хранения данных.

**Анализ и классификация тестирования**

* **Уровни тестирования**
  + **Компонентное тестирование**: Проверка отдельных функций, таких как отображение расписания и отправка уведомлений. Использование JUnit и Mockito для создания юнит-тестов.
  + **Интеграционное тестирование**: Тестирование взаимодействия между компонентами приложения, такими как синхронизация данных с Firebase и отображение этих данных в приложении. Использование Android Instrumentation Tests.
  + **Системное тестирование**: Проверка всего приложения на различных устройствах и эмуляторах для оценки общей производительности и соответствия требованиям. Инструменты: Espresso для автоматизации UI-тестов.
* **Типы тестирования**
  + **Функциональное тестирование**: Проверка корректности работы всех функций приложения, включая поиск и фильтрацию занятий, а также уведомления.
  + **Нефункциональное тестирование**: Нагрузочное тестирование для оценки производительности под нагрузкой, тестирование безопасности для защиты данных пользователей и проверки на уязвимости.
* **Методы тестирования**
  + **Статическое тестирование**: Рецензирование кода и статический анализ с использованием SonarQube. Проверка кода на предмет соответствия стандартам и выявление потенциальных уязвимостей.
  + **Динамическое тестирование**: Методы черного ящика для проверки функциональности без знания внутренней структуры кода. Тестирование на основе спецификаций и пользовательских сценариев.
* **Виды тестирования**
  + **Функциональное тестирование**: Проверка соответствия функциональным требованиям, таких как корректное отображение расписания и работа уведомлений.
  + **Нагрузочное тестирование**: Проверка системы под высокой нагрузкой, например, при одновременном доступе большого числа студентов. Использование инструментов, таких как Firebase Test Lab.
  + **Юзабилити тестирование**: Оценка удобства использования интерфейса приложения. Проведение опросов и интервью с реальными пользователями для сбора обратной связи.
  + **Тестирование безопасности**: Проверка на уязвимости, такие как SQL-инъекции, XSS, и другие распространенные атаки. Использование инструментов, таких как OWASP ZAP.

**Примеры дефектов**

* **Компонентное тестирование**: Ошибки в отображении расписания при изменении данных в Firebase. Например, неправильная сортировка занятий по времени.
* **Интеграционное тестирование**: Проблемы с синхронизацией данных между локальным хранилищем и Firebase. Например, задержка в обновлении расписания после изменения данных.
* **Системное тестирование**: Несоответствие функциональным требованиям, такие как неработающие уведомления о предстоящих занятиях. Проблемы с производительностью при большом объеме данных.
* **Нагрузочное тестирование**: Падение производительности при одновременном использовании приложения большим числом пользователей. Например, увеличенное время загрузки расписания при более чем 1000 активных пользователей.
* **Юзабилити тестирование**: Пользователи испытывают трудности при навигации по приложению. Например, сложность в поиске определенных занятий или неудобное расположение элементов интерфейса.
* **Тестирование безопасности**: Уязвимости, такие как SQL-инъекции или XSS. Например, возможность ввода вредоносного кода в поле поиска расписания.

**Выводы по работе**

Проект "Андроид приложение для отображения расписания Университета" требует комплексного подхода к тестированию для обеспечения качества и надежности системы. Компонентное и интеграционное тестирование помогут выявить дефекты на ранних этапах, тогда как системное тестирование обеспечит соответствие всем требованиям. Нагрузочное и юзабилити тестирование являются критически важными для обеспечения устойчивости и удобства использования системы. Тестирование безопасности необходимо для защиты данных пользователей и предотвращения возможных атак.

**Список использованных источников**

* Учебные материалы по управлению качеством.
* Официальная документация SonarQube.
* Статьи и руководства по тестированию ПО.
* [GitHub репозиторий](https://github.com/madebyhidden/QMoSSS).