Федеральное государственное образовательное бюджетное учреждение

высшего профессионального образования

**Финансовый университет при Правительстве Российской Федерации**

**(Финансовый университет)**

**Департамент анализа данных и машинного обучения**

**Факультета информационных технологий и анализа больших данных**

**Лабораторная работа №6**

**по дисциплине «Управление качеством программных систем»**

Направление подготовки - 09.03.09 «Прикладная информатика»

на тему: Автоматизация тестирования.

**Подготовила**

студентка группы ЗБ-ПИ20-2

Булдакова Кира Александровна

(Ф.И.О.)

**Проверил**:

ассистент ДАДиМО ФИТиАБД,

Клочков Евгений Юрьевич

(Ф.И.О.)

Москва 2024

# **Цель работы:**

Целью данной лабораторной работы является, основываясь на ранее разработанной тестовой документации, разработать автотесты для проверки функциональности программной системы для прогнозирования цен на акции в соответствии с требованиями.

# **Описание реализованных автотестов - инструменты, подходы:**

**Инструменты:**

* JUnit/TestNG: Это фреймворки для написания и запуска автоматических тестов на языке Java. Они предоставляют большое количество аннотаций и утверждений, что делает написание тестов более простым и понятным.
* Selenium: Selenium - это инструмент для автоматизации тестирования веб-приложений. Он позволяет имитировать действия пользователя (например, нажатие кнопок, ввод текста) и проверять результаты на веб-странице.
* Cucumber: Cucumber - это инструмент для тестирования поведения, который позволяет описывать тесты на естественном языке. Это очень полезно для сценариев приемочного тестирования.
* Mockito: Mockito - это фреймворк для создания макетов в Java. Это полезно, когда требуется изолировать тест от внешних зависимостей.

**Подходы:**

* Unit Testing: Это тестирование отдельных единиц кода (например, функций или методов). Это помогает убедиться, что каждая часть приложения работает правильно в изоляции.
* Integration Testing: Это тестирование взаимодействия между различными частями приложения. Это помогает убедиться, что различные компоненты работают правильно вместе.
* End-to-End Testing: Это тестирование приложения в целом, с начала до конца. Это обычно включает в себя имитацию действий реального пользователя и проверку, что приложение ведет себя ожидаемо.
* Regression Testing: Это повторное проведение тестов после внесения изменений в код, чтобы убедиться, что существующая функциональность не была нарушена.
* Behavioural Testing: Это тестирование, которое сосредоточено на том, как приложение ведет себя с точки зрения пользователя. Это включает в себя проверку, что приложение ведет себя ожидаемо при различных сценариях использования.

# **Код автотестов:**

В качестве примера мы будем использовать Python и его библиотеку для тестирования pytest.

**1. Unit Testing:**

Unit-тестирование подразумевает тестирование отдельных единиц кода (например, функций или методов).

Изображение выглядит как Шрифт, снимок экрана, текст, Графика

Автоматически созданное описание

**2. Integration Testing:**

Интеграционное тестирование проверяет взаимодействие между различными частями системы.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

**3. End-to-End Testing:**

End-to-end тестирование проверяет работу системы в целом.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

\*password="123456"

**4. Regression Testing:**

Регрессионное тестирование проверяет, что существующая функциональность осталась неизменной после внесения изменений.

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

**5. Behavioural Testing:**

Поведенческое тестирование сосредоточено на том, чтобы убедиться, что система ведет себя ожидаемо при различных сценариях использования.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

# **Отчет о тестировании:**

| **№** | **Шаги тест-кейса** | **Ожидаемый**  **результат** | **Фактический**  **результат** | **Статус** |
| --- | --- | --- | --- | --- |
| 1 | Проверить, что функция регистрации создает нового пользователя | Функция возвращает объект нового пользователя | Программа возвращает объект нового пользователя | Успешно |
| 2 | Проверить, что после регистрации пользователя, мы можем аутентифицировать этого пользователя | Функция аутентификации возвращает объект зарегистрированного пользователя | Функция аутентификации возвращает объект зарегистрированного пользователя | Успешно |
| 3 | Проверить, что мы можем зарегистрировать и аутентифицировать пользователя | Функция аутентификации возвращает объект зарегистрированного пользователя после регистрации | Функция аутентификации возвращает объект зарегистрированного пользователя после регистрации | Успешно |
| 4 | Проверить, что функции регистрации и аутентификации все еще работают после внесения изменений в код | Функции регистрации и аутентификации все еще работают, как ожидалось | Функции регистрации и аутентификации все еще работают, как ожидалось | Успешно |
| 5 | Проверить, что функция регистрации не создает нового пользователя, если введены некорректные данные | Функция регистрации возвращает ошибку | Функция регистрации возвращает ошибку | Успешно |

# **Выводы по работе:**

На основе проведенных тестов можно сделать следующие выводы:

**Unit Testing**: Функция регистрации работает корректно, так как она успешно создает новых пользователей.

**Integration Testing**: Функции регистрации и аутентификации корректно работают вместе, так как после регистрации пользователя, мы можем успешно аутентифицировать его.

**End-to-End Testing**: Весь процесс регистрации и аутентификации пользователя работает корректно, так как мы можем зарегистрировать и аутентифицировать пользователя без проблем.

**Regression Testing**: Функции регистрации и аутентификации продолжают работать корректно после внесения изменений в код, что говорит о стабильности кода.

**Behavioural Testing**: Функции регистрации и аутентификации ведут себя ожидаемо при различных сценариях использования. Защита от некорректных данных работает корректно, так как функция регистрации не создает нового пользователя при введении некорректных данных.

Тестирование показало, что функции регистрации и аутентификации работают корректно и стабильно, а также корректно реагируют на различные сценарии использования.

# **Список используемых источников:**

1. Учебное пособие «Основы управления качеством программных средств»
2. С. А. Фиросва, А. В. Пакшин «Организация процесса автоматизированного тестирования».