Лабораторная работа №6: Автоматизация тестирования

**Цель работы**

Приобрести навыки создания автотестов для проверки функциональности программной системы с использованием ранее разработанной тестовой документации (тест-кейсов, тест-сьютов, планов тестирования).

**Инструменты и подходы автотестов**

Для создания автотестов был выбран Python с использованием библиотек unittest для структурированных модульных тестов и PyQt6 для эмуляции пользовательского интерфейса. Это позволяет тестировать функциональность программы, взаимодействуя с её элементами, как это сделал бы реальный пользователь.

**Основные библиотеки:**

* unittest: Для создания и запуска модульных тестов.
* PyQt6: Для эмуляции взаимодействия с элементами пользовательского интерфейса.
* pandas, openpyxl: Для работы с файлами и данными.

**Код автотестов**

**Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание**

**Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание**

**Отчет о тестировании**

Тест-кейс 1: Тестирование загрузки файла

Время выполнения: 20.05.2024, 10:00-10:30

Идентификатор: TC-01

Результат: Неуспешно

Шаги к воспроизведению:

* Открыть приложение.
* Нажать на кнопку "Загрузить файл".
* Выбрать файл "test\_data.xlsx".
* Нажать "Открыть".

Результат: Появляется ошибка "Файл не может быть загружен".

Ожидаемый результат: Файл загружается и данные корректно отображаются в системе.

Автотест: test\_file\_upload

Описание автотеста: Автотест пытается загрузить файл "test\_data.xlsx" и проверяет, что DataFrame не пустой. Ошибка возникает из-за несовместимости формата файла или неправильной обработки данных.

Тест-кейс 2: Тестирование ввода даты

Время выполнения: 20.05.2024, 10:30-11:00

Идентификатор: TC-02

Результат: Успешно

Описание: Даты введены корректно, система принимает все допустимые форматы. Проверка была проведена для нескольких форматов даты.

Шаги к воспроизведению:

* Открыть приложение.
* Ввести дату в поле ввода даты.
* Нажать "Сохранить".
* Результат: Дата сохраняется и отображается корректно.

Ожидаемый результат: Дата сохраняется и отображается в правильном формате.

Автотест: test\_date\_input

Результат автотеста: Успешно

Описание автотеста: Автотест проверяет, что введенная дата валидна и правильно отображается в календаре.

Тест-кейс 3: Тестирование ввода периодичности

Время выполнения: 20.05.2024, 11:00-11:30

Идентификатор: TC-03

Результат: Успешно

Описание: Периодичность введена без ошибок, все возможные варианты принимаются системой. Проверка была проведена для нескольких вариантов периодичности.

Шаги к воспроизведению:

* Открыть приложение.
* Выбрать периодичность из выпадающего списка.
* Нажать "Сохранить".

Результат: Периодичность сохраняется и отображается корректно.

Ожидаемый результат: Периодичность сохраняется и отображается в правильном формате.

Автотест: test\_periodicity\_input

Результат автотеста: Успешно

Описание автотеста: Автотест проверяет, что выбранная периодичность сохраняется и отображается корректно.

**Выводы по работе**

В ходе выполнения лабораторной работы были разработаны и протестированы автотесты для проверки функциональности программной системы. Тесты успешно покрыли основные аспекты взаимодействия с пользовательским интерфейсом, включая загрузку файлов, ввод дат и периодичности. Результаты тестирования подтверждают корректность работы большинства функций, но указывают на необходимость доработок для обеспечения полной функциональности и устойчивости системы.

**Список литературы**

1. <https://thecode.media/tested/>
2. [unittest — Unit testing framework — Python 3.12.3 documentation](https://docs.python.org/3/library/unittest.html)
3. [Делаем простые автотесты на Python — Журнал «Код» программирование без снобизма (thecode.media)](https://thecode.media/unittest/)