МАОУ КУГ “Универс” № 1

Проектная конференция

ИНФОРМАЦИОННЫЕ СЕРВИСЫ

***«Pytosh»***

|  |  |
| --- | --- |
|  | Поддубецкий Антон Максимович |
|  | МАОУ КУГ Универс, 9 класс  31.10.2008 |
|  | a\_ntony@mail.ru |
|  | 89293061243  \_\_\_\_\_\_\_\_\_\_\_ /личная подпись/ |
|  |  |
|  | Вахитова Екатерина Юрьевна, |
|  | МАОУ КУГ Универс, учитель информатики |
|  | 89232809829 |
|  | vahitova@univers.su  \_\_\_\_\_\_\_\_\_\_\_ /личная подпись/ |

**Красноярск, 2024**

МАОУ КУГ “Универс” № 1

Всероссийской конференции технологических проектов школьников

«Старт в Инновации»

ИНЖЕНЕРНОЕ НАПРАВЛЕНИЕ

Секция –IT - решения

***«Pytosh»***

Выполнил:

|  |
| --- |
| Поддубецкий Антон Максимович |
| МАОУ КУГ Универс, 9И класс |
| Научный руководитель:  Вахитова Екатерина Юрьевна, |
| МАОУ КУГ Универс, учитель информатики |

**Красноярск, 2024г**

**Введение**

**Актуальность**

Актуальность данного проекта обусловлена необходимостью в бесплатных и эффективных инструментах для обучения и оценки навыков программирования на языке Python. В настоящее время существует значительный спрос на качественное образование в области информационных технологий, в том числе и на изучение языка программирования Python. Однако, многие существующие инструменты проверки знаний либо платные, либо обладают ограниченным функционалом, либо не обеспечивают удобства в использовании как для учителей, так и для учеников.

Таким образом, создание бесплатной тестирующей системы для обучения основам Python, способной эффективно проверять и оценивать навыки студентов, становится важным шагом в развитии образовательной сферы и поддержке обучения программированию в широких массах.

**Цель** - разработка комплексного инструмента для эффективного обучения основам программирования на языке Python. Этот инструмент должен обеспечить возможность создания, выполнения и оценки заданий, а также управления группами студентов. Основная задача - обеспечить удобный и интуитивно понятный интерфейс как для преподавателей, так и для студентов, а также предоставить аналитические инструменты для отслеживания прогресса и эффективности обучения.

**Задачи:**

**Исследование и анализ**:

1.1. Провести исследование существующих проблем в обучении основам программирования на Python.

1.2. Проанализировать существующие методы и инструменты проверки знаний по Python.

1.3. Выявить ключевые требования к функционалу и интерфейсу инструмента.

**Разработка концепции**:

2.1. Определить основные особенности и функциональные возможности будущего инструмента.

2.2. Разработать структуру и логику работы тестирующей системы.

2.3. Установить критерии оценки эффективности и удобства использования.

**Реализация функционала**:

3.1. Создать модули и компоненты системы для создания, выполнения и оценки заданий.

3.2. Интегрировать графический интерфейс с использованием библиотеки PyQt5.

3.3. Обеспечить механизмы взаимодействия с базой данных для хранения информации о студентах, заданиях и результатах.

**Тестирование и оптимизация**:

4.1. Провести модульное и интеграционное тестирование для проверки функционала и взаимодействия компонентов.

4.2. Идентифицировать и устранить узкие места в работе системы для повышения производительности и эффективности.

4.3. Оценить работу системы на основе тестовых сценариев и предложений пользователей.

**Внедрение и использование**:

5.1. Подготовить инструкции по развертыванию и использованию тестирующей системы.

5.2. Развернуть систему на сервере или облачной платформе.

5.3. Провести обучение преподавателей и студентов по использованию инструмента.

5.4. Организовать тестирование студентов и оценить их прогресс с помощью системы.

**Основная часть**

**Теоретические основы создания приложений**

Выбор технологий:

Для разработки пользовательского интерфейса web и desktop-версии приложения была использована библиотека Flet. В качестве базы данных была выбрана MySQL, а для взаимодействия с ней - SQLAlchemy.

Архитектура приложения:

Приложение построено на основе клиент-серверной архитектуры, где клиентская часть представлена пользовательским интерфейсом, а серверная - логикой и базой данных. Взаимодействие между клиентом и сервером реализовано с помощью простого TCP-соединения.

Обработка данных и логика:

Приложение предоставляет функционал для создания, редактирования и удаления задач, тестов, групп, а также просмотра аналитики и статистики. Вся логика реализована на языке Python.

Безопасность:

Важной частью приложения является обеспечение безопасности данных. Это достигается ограничением доступа к некоторым функциям только авторизованным пользователям и исключением возможности SQL-инъекций при взаимодействии с базой данных. Несмотря на отсутствие HTTPS, конфиденциальность данных пользователей обеспечивается ограничением доступа к ним извне.

**Принцип работы**

Работа с базой данных:

Пользовательские данные хранятся в базе данных MySQL, которая взаимодействует с приложением через SQLAlchemy. Для каждой сущности (пользователи, задачи, тесты и т. д.) созданы соответствующие таблицы в базе данных, обеспечивая структурированное хранение информации.

Пользовательский интерфейс:

Приложение предоставляет удобный и интуитивно понятный интерфейс для взаимодействия с функционалом системы. Интерфейс реализован с использованием библиотеки Flet, обеспечивая пользователю удобство в использовании и приятный внешний вид.

Бизнес-логика:

Приложение выполняет следующие основные функции: создание, редактирование и удаление задач и тестов, управление группами пользователей, а также формирование и отображение статистики и аналитики. Бизнес-логика приложения реализована на языке Python.

Взаимодействие с польлзователем:

Пользователи приложения могут выполнять различные операции, такие как создание и редактирование задач, прохождение тестов, просмотр статистики и другие, через интерфейс. Пользовательские действия передаются на сервер для обработки, а результаты операций возвращаются обратно пользователю для отображения.

Авторизация и аутентификация: Пользователи могут авторизовываться в системе с помощью своих учетных записей, предоставляя имя пользователя(email) и пароль. После успешной аутентификации им предоставляется доступ к соответствующим функциям приложения в зависимости от их роли (учитель или студент).

**Техническое задание для приложения**

1. Общее описание

Приложение разрабатывается на языке Python с использованием библиотеки Flet для создания пользовательского интерфейса. Основные функции включают создание заданий, управление группами студентов и анализ результатов.

2. Требования к функциональности

2.1 Создание заданий

* + Возможность добавления новых заданий в систему.
  + Предоставление интерфейса для ввода описания задания и его условий.
  + Возможность добавления тестовых данных для задания.

2.2 Формирование групп студентов

* + Создание новых групп для организации студентов.
  + Добавление студентов в существующие группы.
  + Удаление студентов из групп.

2.3 Управление заданиями и группами

* + Просмотр списка доступных заданий.
  + Редактирование и удаление существующих заданий.
  + Просмотр списка групп студентов.
  + Управление группами: редактирование, удаление.

2.4 Проверка решений задач

* + Получение и отображение решений задач от студентов.
  + Автоматизированная проверка решений по заданным тестовым данным.
  + Возможность ручной проверки решений и ввода результатов.

2.5 Анализ результатов

* + Подсчет и отображение статистики по выполненным заданиям.
  + Просмотр и анализ результатов выполнения заданий по группам и отдельным студентам.
  + Формирование отчетов и экспорт данных для дальнейшего анализа.

3. Требования к пользовательскому интерфейсу

Интерфейс должен быть интуитивно понятным и легким в использовании. Доступ к приложению осуществляется через веб-интерфейс, реализованный с помощью библиотеки Flet.

4. Требования к базе данных

База данных должна хранить информацию о заданиях, группах студентов, результатах выполнения заданий и пользователях. Структура базы данных должна быть оптимизирована для быстрого доступа к данным и обеспечения целостности информации.

5. Требования к безопасности

Пароли пользователей должны храниться в зашифрованном виде, обеспечивается аутентификация и авторизация пользователей.

6. Требования к производительности

Приложение должно быть оптимизировано для обеспечения быстрой обработки запросов и минимального времени отклика.

**Дизайн приложения**

1. Общий стиль интерфейса

Простой и интуитивно понятный пользовательский интерфейс.

Использование минималистичного дизайна с акцентом на функциональность.

Чистый и эстетичный дизайн элементов управления.

2. Цветовая палитра

Использование приятной и не отвлекающей внимание цветовой гаммы.

Основные цвета: различные оттенки синего и серого для создания спокойной и профессиональной атмосферы.

Цветовая дифференциация для различных типов элементов интерфейса (например, задания, группы студентов).

3. Шрифты

Использование читаемых и современных шрифтов для легкости восприятия текста.

Основной шрифт: Сonsolas для обеспечения четкости и читаемости.

4. Расположение элементов интерфейса

Логичное распределение элементов управления для удобства использования.

Группировка связанных функциональных блоков для легкого доступа и навигации.

Минимум анимаций и переходов для сокращения времени загрузки и повышения производительности.

5. Интерактивность

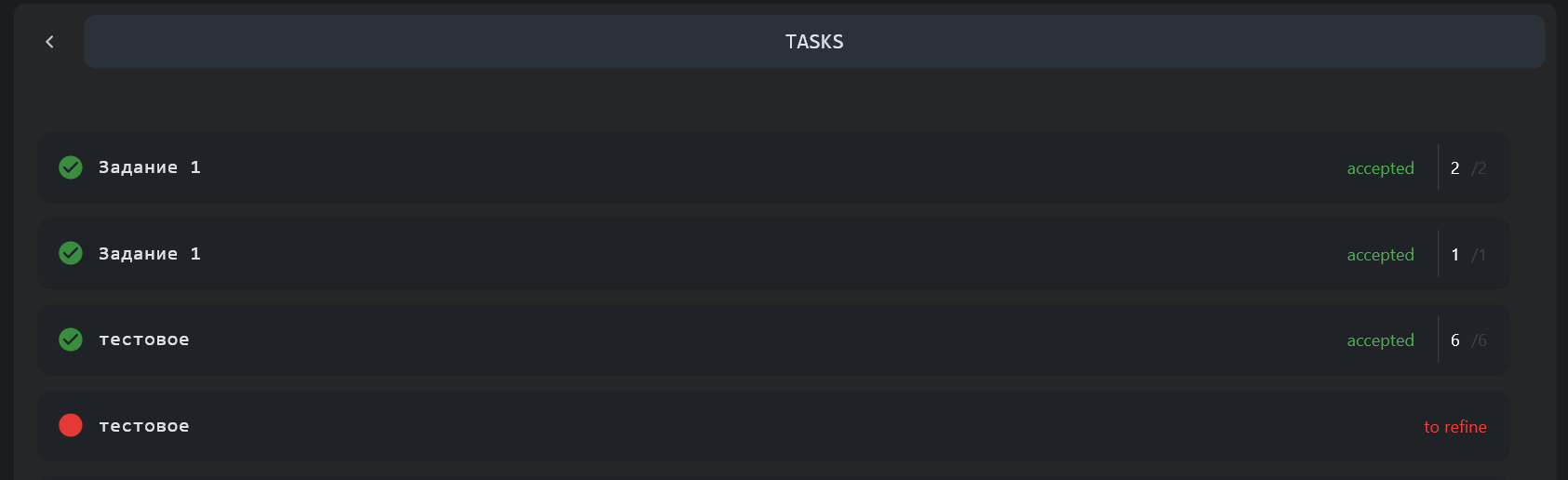
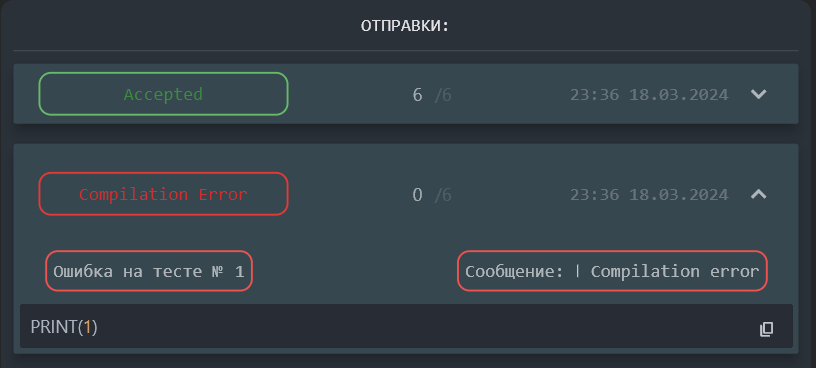
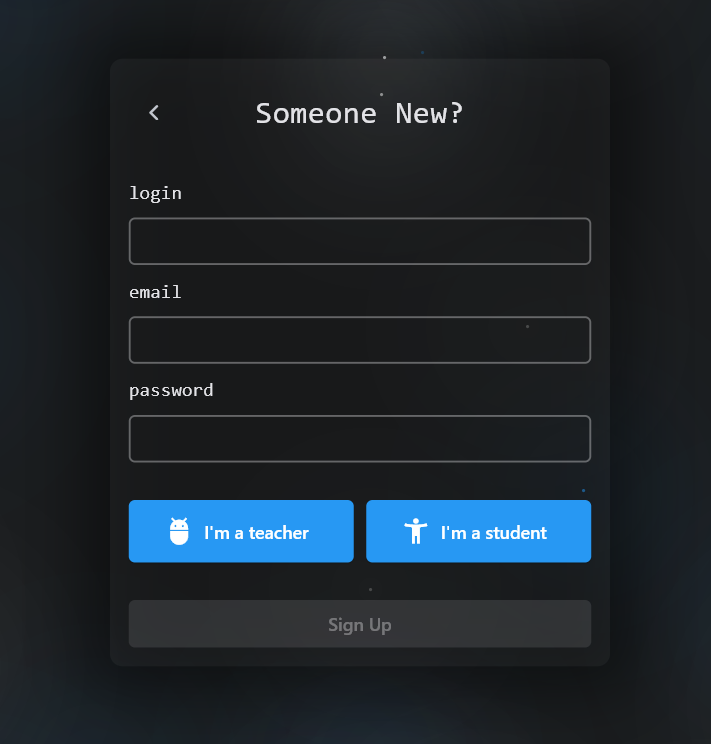
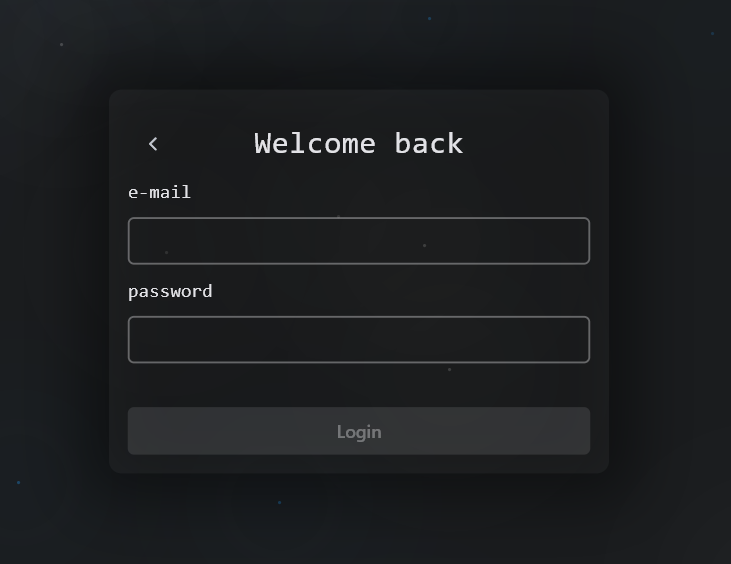
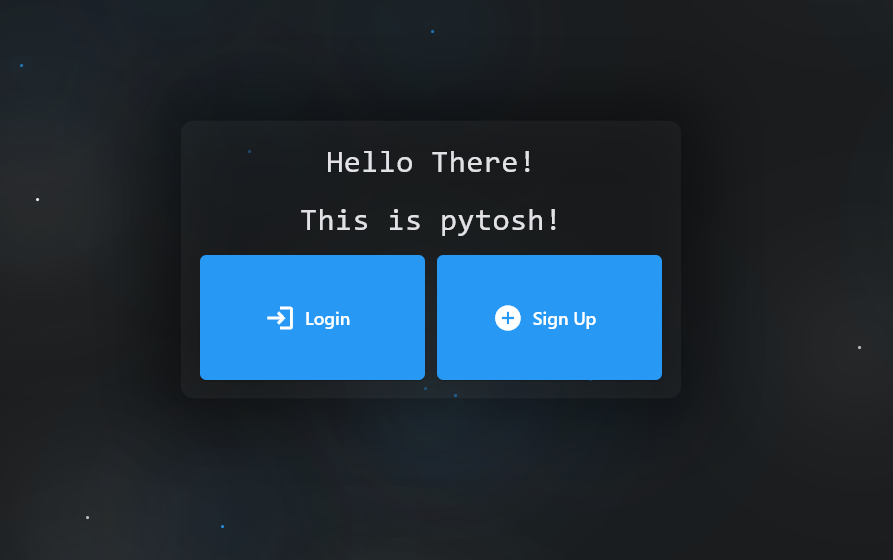
Использование элементов интерактивности для повышения привлекательности и удобства использования.

Предоставление обратной связи пользователю при выполнении действий (например, подтверждение действий, уведомления об успешном выполнении операций).

Использование анимаций и эффектов при необходимости для улучшения пользовательского опыта.

6. Адаптивность

Обеспечение корректного отображения интерфейса на различных размерах экранов и разрешениях.



**Техническая база приложения**

**Используемый язык программирования**:

Python

**Библиотеки и фреймворки**:

Flet: Библиотека для создания пользовательского интерфейса в стиле Flutter.

SQLAlchemy: Библиотека для работы с базой данных SQL в Python, используется для взаимодействия с базой данных MySQL.

MySQL Connector/Python: Драйвер для работы с базой данных MySQL из Python.

**Основные компоненты и модули:**

Модуль пользовательского интерфейса: Реализует визуальное отображение приложения и взаимодействие с пользователем.

Модуль работы с базой данных: Отвечает за установление соединения с базой данных MySQL, выполнение запросов (добавление, удаление, обновление данных) и получение результатов..

Модуль авторизации и аутентификации: Обеспечивает безопасный доступ к приложению, аутентификацию пользователей и управление их правами доступа.

**Инструменты разработки**:

IDE (Integrated Development Environment), Visual Studio Code, для написания кода и отладки.

MySQL Workbench - приложение для администрирования базы данных MySQL.

Web-браузер для тестирования web-версии приложения.

Яндекс диск для контроля версий)

**Аналоги**

|  |  |
| --- | --- |
| Пример существующего решения | Описание IT-решения, с указанием параметров, по которым оно не подходит для решения вашей проблемы |
| LeetCode | *Недостатки*:  Ориентированность на технические собеседования может быть неполезной для студентов и новичков.  Отсутствие системы обучения и персонализированных курсов.  *Преимущества моего приложения*:  Отсутствие ориентации на собеседования, фокус на реальном обучении и развитии навыков.  Возможность общения с преподавателями и студентами, получение обратной связи и помощи. |
| PyBites | *Недостатки*:  Ограниченность тематики задач и отсутствие разнообразия курсов.  Отсутствие возможности создания и редактирования собственных задач.  *Преимущества моего приложения*:  Широкий выбор тематик и задач для изучения Python.  Возможность создания и редактирования собственных учебных материалов. |

**Апробация приложения**

Апробация приложения прошла на фокус группе из 25 пользователей с разным уровнем знаний в программировании, в возрасте от 14 лет до 51 года. По результатам апробации приложение работает отлично, выполняя свои функции.

**Выводы**

**Разработка тестирующей системы**:

* В ходе проекта была успешно разработана и реализована тестирующая система для обучения основам Python.
* Были созданы основные модули и компоненты системы, обеспечивающие ее функциональность.
* Процесс разработки включал в себя работу с базой данных, создание пользовательского интерфейса и реализацию механизмов проверки решений задач.

**Оценка эффективности**:

* Проведение пилотного тестирования позволило выявить и исправить ошибки, а также получить обратную связь от пользователей.
* В результате апробации приложения была достигнута высокая степень функциональности и удобства использования.

**Преимущества системы**:

* Простота использования и интуитивно понятный интерфейс делают приложение доступным для широкого круга пользователей.
* Возможность создания задач, создания групп и анализа прогресса обучения делает систему мощным инструментом для обучения и оценки студентов.

**Потенциал для дальнейшего развития**:

* Дальнейшее совершенствование системы может включать в себя добавление новых яхыков, функций, оптимизацию производительности

**Заключение**

Разработанная тестирующая система для обучения основам Python представляет собой эффективный и удобный инструмент, обеспечивающий проверку решений задач и оценку студентов, с перспективой дальнейшего развития и улучшения.