To Do App

выполнил: Динтер “RomSTil” Максим

Постановка задачи:
Создание приложения управления списком задач (To-Do App)

Цель:
Разработать веб-приложение для эффективного управления списком задач, позволяющее пользователям добавлять, редактировать, удалять и фильтровать задачи.

Основные требования

Пользователи должны иметь возможность добавлять новые задачи, указывая их название.

Приложение должно позволять пользователям редактировать названия задач.

Пользователи должны иметь возможность удалять задачи из списка.

Приложение должно предоставлять возможность фильтровать задачи по статусу (все задачи, активные, завершённые).

На экране должна отображаться информация о количестве активных задач.

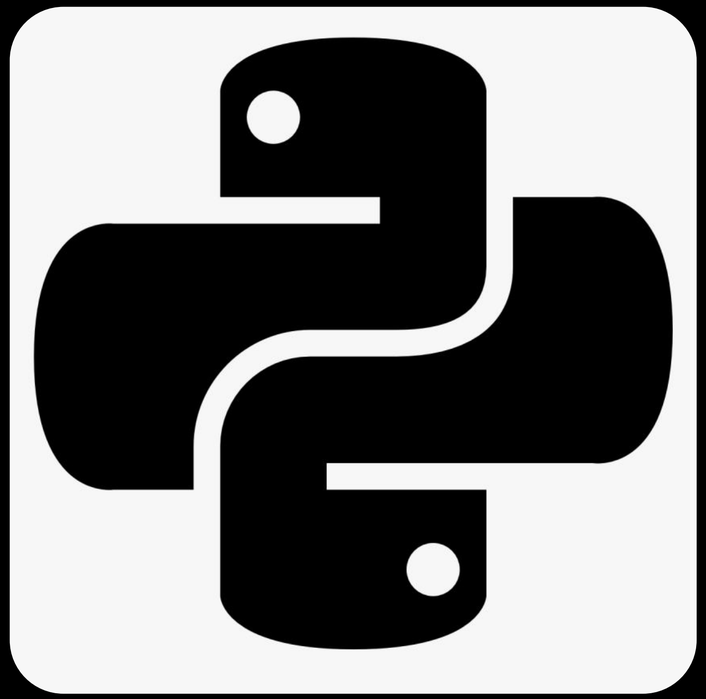
Приложение должно обладать удобным и интуитивно понятным пользовательским интерфейсом.

Приложение должно быть адаптировано для работы на различных устройствах и разрешениях экранов.

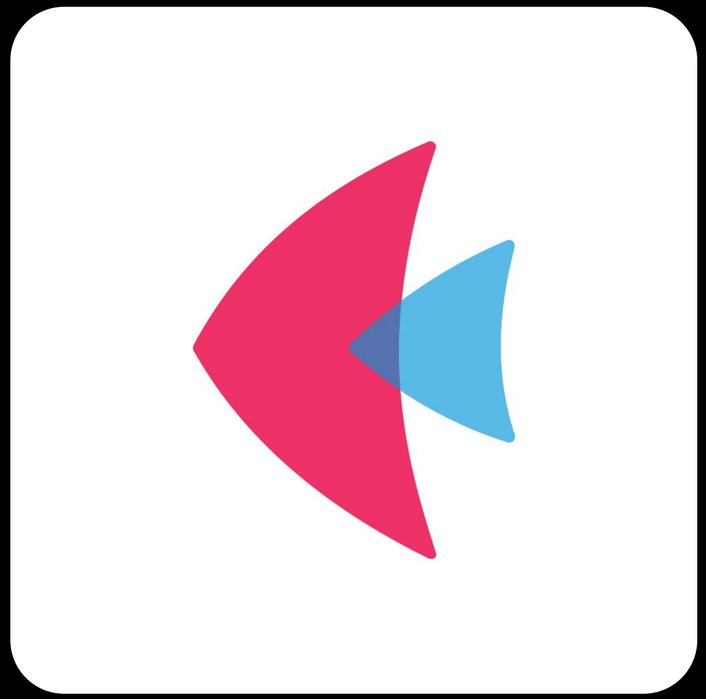
Стек Технологий

Todo App

Python



Flet



Нюансы

Асинхронная обработка

Использование асинхронности позволяет приложению быть отзывчивым и эффективным в обработке запросов пользователей. Это особенно важно при выполнении операций ввода-вывода, таких как обновление интерфейса или обращение к базе данных.

Нюансы

Адаптивный дизайн

Реализация адаптивного дизайна позволяет приложению хорошо выглядеть и удобно работать на устройствах с различными разрешениями экранов. Это обеспечивает лучший пользовательский опыт и увеличивает охват аудитории.

Нюансы

Фильтрация задач

Фильтрация задач по статусу (все, активные, завершённые) позволяет пользователям легко находить нужные задачи и организовывать свою работу более эффективно.

Нюансы

Добавление задачи

Алгоритм: При получении запроса на добавление задачи, создаётся экземпляр класса Task с указанным названием задачи. Этот экземпляр затем добавляется в список задач приложения.

Нюансы

Добавление задачи

Алгоритм: При получении запроса на добавление задачи, создаётся экземпляр класса Task с указанным названием задачи. Этот экземпляр затем добавляется в список задач приложения.

Математический аппарат: Простая операция добавления элемента в список.

```python
async def add_clicked(self, e):
    if self.new_task.value:
        task = Task(self.new_task.value, self.task_status_change, self.task_delete)
        self.tasks.controls.append(task)
        self.new_task.value = ""
        await self.new_task.focus_async()
        await self.update_async()
```

Нюансы

Фильтрация задач

Алгоритм: При изменении выбранной вкладки фильтра, происходит пересмотр списка задач и скрываются те, которые не соответствуют выбранному фильтру.
Математический аппарат: Условные операторы для проверки статуса задачи и её видимости в зависимости от выбранного фильтра.

Нюансы

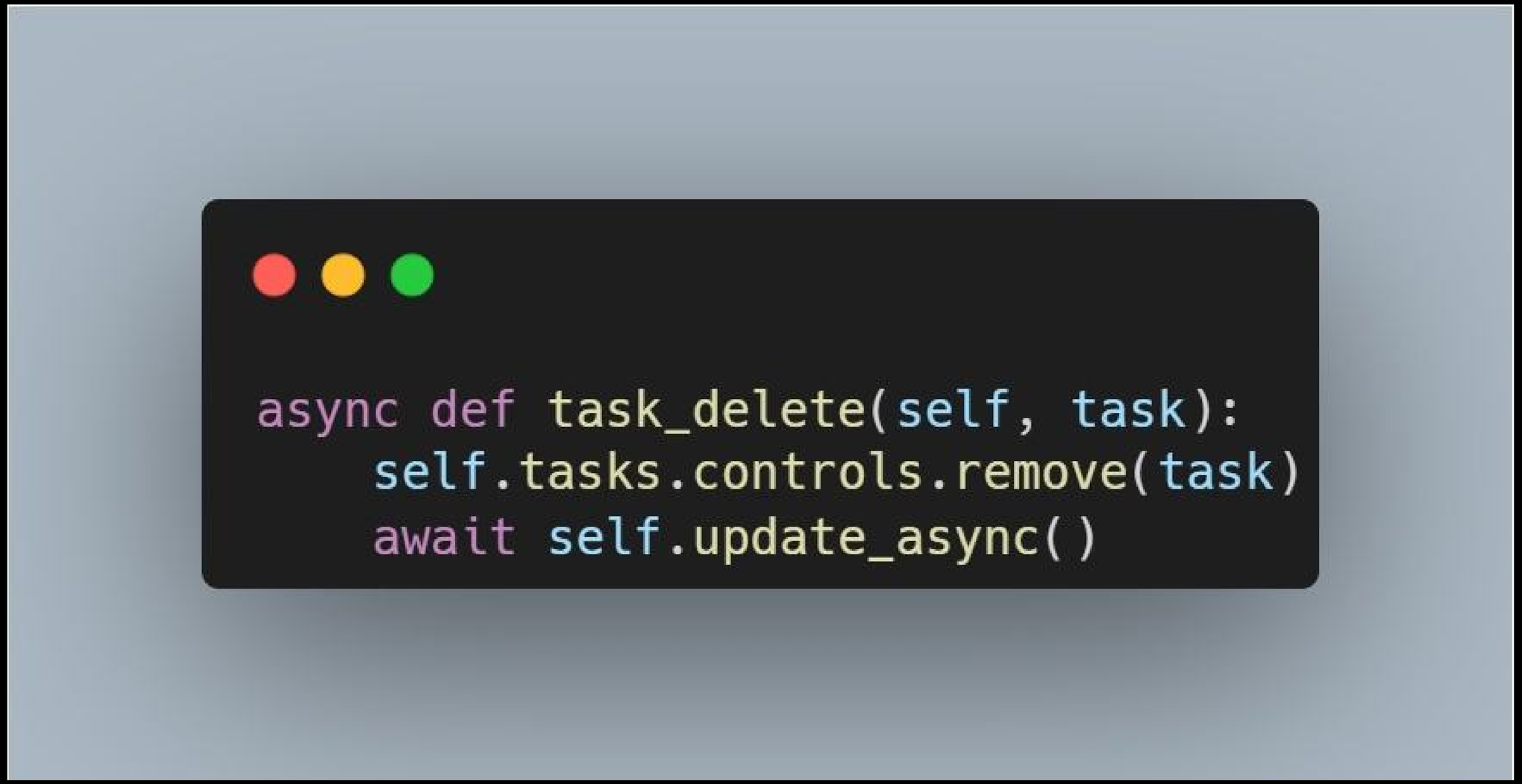
```python
async def update_async(self):
    status = self.filter.tabs[self.filter.selected_index].text
    count = 0
    for task in self.tasks.controls:
        task.visible = (
            status == "all"
            or (status == "active" and task.completed == False)
            or (status == "completed" and task.completed)
        )
        if not task.completed:
            count += 1
    self.items_left.value = f"{count} active item(s) left"
    await super().update_async()
```

Нюансы

Удаление задачи

Алгоритм: При запросе на удаление задачи, соответствующий экземпляр Task удаляется из списка задач приложения.
Математический аппарат: Операции удаления элемента из списка.

Нюансы



```
async def task_delete(self, task):
    self.tasks.controls.remove(task)
    await self.update_async()
```

GitHub

github.com/RomSTil/