**Задание 1**

Используя механизм маршрутизации создайте веб-приложение, отображающее разные разделы с информацией о компании. В приложении должны быть такие разделы:

■ Главная;

■ Новости;

■ Руководство компании;

■ О компании;

■ Контакты.

Доступ к разным разделам должен производиться по разным адресам.

Например, если пользователь указал 127.0.0.1:8000, должен открыться раздел Главная.

Если было указано 127.0.0.1:8000/news, должен открыться раздел Новости.

Если было указано 127.0.0.1:8000/management, должен открыться раздел Руководство компании и т. д.

**Реализация проэкта**

Для создания веб-приложения с маршрутизацией в Python можно использовать фреймворк Flask, который прост в освоении и отлично подходит для таких задач:

Установка Flask:

Создайте папку для проекта, откройте ее через VS code и в терминале пропишите следующую команду для установки фреймворка Flask:

**pip install flask**

**Как это работает:**

Маршруты:

@app.route("/") – отображает раздел "Главная".

@app.route("/news") – отображает раздел "Новости".

@app.route("/management") – отображает раздел "Руководство компании".

@app.route("/about") – отображает раздел "О компании".

@app.route("/contacts") – отображает раздел "Контакты".

Функции: Каждая функция возвращает HTML-код, который будет отображаться пользователю при переходе на соответствующий маршрут.

Запуск: Приложение запускается на адресе 127.0.0.1:8000.

**Запуск**

Сохраните код в файл, например, app.py, и запустите его:

**python app.py**

Перейдите по адресам в браузере:

http://127.0.0.1:8000 — Главная

http://127.0.0.1:8000/news — Новости

http://127.0.0.1:8000/management — Руководство компании

http://127.0.0.1:8000/about — О компании

http://127.0.0.1:8000/contacts — Контакты

Код программы в файле – «Код к заданию 1»

**Задание 2**

Добавьте к проекту из первого задания возможность обработки неправильных адресов. Например, при указании 127.0.0.1:8000/news/heyjude или 127.0.0.1:8000/news/yellowsubmarine, или 127.0.0.1:8000/ news/cometogether и т. д., нужно отображать содержимое раздела news. Тот же принцип нужно применить и к остальным разделам.

**Реализация проекта:**

Для обработки неправильных адресов и отображения контента соответствующего раздела, можно использовать параметризованные маршруты в Flask. Это позволяет обрабатывать маршруты с дополнительными частями URL и игнорировать их.

Изменения и пояснения

Обработка дополнительных путей:

В каждом маршруте добавлен дополнительный маршрут вида /<path:subpath>.

subpath — это необязательный параметр, который захватывает все дополнительные части пути после основного маршрута.

Например, если пользователь переходит по адресу 127.0.0.1:8000/news/heyjude, то этот маршрут будет обработан функцией news.

Параметр subpath:

Он используется как параметр в функции, но мы его игнорируем, так как возвращаем только основное содержимое раздела.

Это позволяет игнорировать любые дополнительные подмаршруты.

Общий вид URL:

127.0.0.1:8000/news/anything будет отображать "Новости".

127.0.0.1:8000/management/anything будет отображать "Руководство компании" и так далее.

Код программы в файле – «Код к заданию 2».

После внесения изменений запустить проект с помощью команды:

**python app.py**

Проверьте в браузере:

http://127.0.0.1:8000/news — отобразятся новости.

http://127.0.0.1:8000/news/heyjude — отобразятся новости.

http://127.0.0.1:8000/management/cometogether — отобразится руководство компании.

http://127.0.0.1:8000/about/yellowsubmarine — отобразится "О компании".

Этот подход делает маршруты более гибкими и устойчивыми к ошибкам.

**Задание 3**

Добавьте раздел Филиалы к проекту из первого задания. При указании пути 127.0.0.1:8000/branches должен отображаться раздел с общей информацией обо всех филиалах. Если был указан путь 127.0.0.1:8000/branches/London, отображается информация о филиале в Лондоне. Если был указан путь 127.0.0.1:8000/branches/Paris, отображается информация о филиале в Париже.

**Реализация**

Для добавления раздела "Филиалы" с указанной логикой обработки маршрутов в проект можно использовать параметризованные маршруты. Мы будем динамически отображать информацию о филиалах в зависимости от переданного параметра.

Код программы в файле – «Код к заданию 3».

**Изменения и пояснения**

Добавлен маршрут для общей информации о филиалах:

@app.route("/branches") отображает список филиалов с общей информацией.

Добавлен маршрут для конкретного филиала:

@app.route("/branches/<string:branch\_name>") обрабатывает запросы с именами филиалов, например, /branches/London или /branches/Paris.

**Словарь branch\_info:**

Используется для хранения информации о доступных филиалах.

Если переданное имя филиала (branch\_name) есть в словаре, возвращается соответствующая информация.

Если филиала нет в словаре, отображается сообщение о том, что филиал не найден.

Результаты:

http://127.0.0.1:8000/branches — отображается общая информация обо всех филиалах.

http://127.0.0.1:8000/branches/London — отображается информация о филиале в Лондоне.

http://127.0.0.1:8000/branches/Paris — отображается информация о филиале в Париже.

http://127.0.0.1:8000/branches/Unknown — отображается сообщение о том, что филиал не найден.

Этот подход легко масштабируется: достаточно добавить новые филиалы в словарь branch\_info.

**Задание 4**

Добавьте к проекту из первого задания механизм ссылок, который позволит переходить из раздела в раздел.

**Реализация**

Чтобы добавить механизм ссылок для перехода между разделами, можно использовать HTML-ссылки в каждом разделе. Это создаст простую навигацию по сайту.

**Изменения и пояснения**

Механизм навигации:

Добавлена функция get\_navigation(), которая возвращает HTML-код с ссылками на все разделы.

Ссылки указывают на маршруты приложения: /, /news, /management, /about, /contacts, /branches.

Включение навигации в разделы:

Во всех обработчиках маршрутов (home, news, management и т. д.) добавлен вызов get\_navigation() для отображения ссылок.

Ссылки на филиалы:

В разделе "Филиалы" (/branches) добавлены ссылки на конкретные филиалы, например, /branches/London.

Результаты:

Каждый раздел содержит меню навигации, позволяющее перейти в любой другой раздел.

В разделе "Филиалы" ссылки ведут к информации о конкретных филиалах.

Для проверки работоспособности приложения пропишите в консоли следующую команду:

**python app.py**

Откройте в браузере:

http://127.0.0.1:8000 — Главная с меню.

Перейдите по ссылкам в меню для проверки навигации.