**Документация**

**Описание:**

Приложение предназначено для получения товаров по списку заказов из базы данных и вывода их информации в консоль.

**Структура:**

* **main.py**: Основной файл, отвечающий за запуск приложения.
* **logginApp/loggingApp.py**: Файл, отвечающий за настройку и работу логгера.
* **database/database.py**: Файл, отвечающий за подключение к базе данных и выполнение запросов.
* **console/console.py**: Файл, отвечающий за вывод информации в консоль.
* **config/config.py**: Файл, отвечающий за чтение и обработку конфигурационного файла.

**Зависимости:**

* **psycopg2:** Адаптер для работы с PostgreSQL базами данных в Python.

**Использование**:

1. Установка виртуального окружения

Для начала необходимо установить виртуальное окружение. Это можно сделать с помощью команды:

**pip install virtualenv**

1. Создание виртуального окружения

Следующим шагом необходимо создать виртуальное окружение для проекта. Для этого нужно выполнить команду:

**virtualenv venv**

Эта команда создаст директорию **venv** в текущей директории, которая будет содержать виртуальное окружение.

1. Активация виртуального окружения

Для активации виртуального окружения необходимо выполнить команду:

**source venv/bin/activate**

После активации виртуального окружения в командной строке появится префикс **(venv),** который указывает, что вы работаете в виртуальном окружении.

1. Установка зависимостей

Для установки зависимостей необходимо выполнить команду:

**pip install -r requirements.txt**

Эта команда установит все необходимые библиотеки, указанные в файле **requirements.txt**.

1. Запуск программы

После установки зависимостей можно запустить программу. Для этого нужно выполнить команду:

**python main.py <orders>**

где **<orders>** - список заказов, разделенных пробелом.

1. Деактивация виртуального окружения

После завершения работы с программой необходимо деактивировать виртуальное окружение. Для этого нужно выполнить команду:

**deactivate**

После деактивации виртуального окружения префикс **(venv)** исчезнет из командной строки.

**Файлы логов:**

* **loggingApp/info.log**: Файл, содержащий информационные сообщения логгера.
* **loggingApp/error.log**: Файл, содержащий сообщения об ошибках логгера.

**Функционал:**

* **main.py**: Запускает приложение, получает список заказов из командной строки, читает конфигурационный файл, получает товары по заказам из базы данных и выводит их информацию в консоль.
* **logginApp/loggingApp.py**: Настраивает логгер, записывает информационные и ошибочные сообщения в соответствующие файлы.
* **database/database.py**: Устанавливает соединение с базой данных, выполняет запрос на получение товаров по списку заказов и возвращает результат.
* **console/console.py**: Выводит информацию о товарах в консоль.
* **config/config.py**: Читает и обрабатывает конфигурационный файл, предоставляет объект конфигурации для дальнейшего использования.

**Конфигурационный файл:**

Файл **config/config.json** должен содержать следующие настройки:

{

"DB": {

"host": "<hostname>",

"user": "<username>",

"password": "<password>",

"database": "<database>",

"port": "<port>"

}

}

Здесь <hostname>**,** <username>**,** <password>**,** <database>и<port> - соответствующие параметры подключения к базе данных.

**Схема таблиц базы данных**

Схема таблиц базы данных:

1. Таблица products (продукты):

* id - уникальный идентификатор продукта (первичный ключ, автоинкремент);
* name - название продукта (строка, не может быть пустой, уникально);
* description - описание продукта (текст);
* price - цена продукта (десятичное число с двумя знаками после запятой, не может быть отрицательным или равным нулю);
* quantity - количество продукта на складе (целое число, не может быть отрицательным или равным нулю).

1. Таблица shelves (полки):

* id - уникальный идентификатор полки (первичный ключ, автоинкремент);
* name - название полки (строка, не может быть пустой, уникально).

1. Таблица orders (заказы):

* id - уникальный идентификатор заказа (первичный ключ, автоинкремент);
* created\_at - дата и время создания заказа (автоматически устанавливается текущим временем);
* status\_id - идентификатор статуса заказа (внешний ключ, ссылается на таблицу status).

1. Таблица product\_orders (продукты в заказе):

* id - уникальный идентификатор строки заказа (первичный ключ, автоинкремент);
* created\_at - дата и время создания строки заказа (автоматически устанавливается текущим временем);
* product\_id - идентификатор продукта (внешний ключ, ссылается на таблицу products);
* order\_id - идентификатор заказа (внешний ключ, ссылается на таблицу orders);
* quantity - количество продукта в заказе (целое число, не может быть отрицательным или равным нулю).

1. Таблица shelves\_product (продукты на полках):

* id - уникальный идентификатор строки (первичный ключ, автоинкремент);
* created\_at - дата и время создания строки (автоматически устанавливается текущим временем);
* product\_id - идентификатор продукта (внешний ключ, ссылается на таблицу products);
* shelf\_id - идентификатор полки (внешний ключ, ссылается на таблицу shelves);
* is\_main - флаг, указывающий, является ли полка основной для продукта (булево значение).

1. Таблица status (статусы заказов):

* id - уникальный идентификатор статуса (первичный ключ, автоинкремент);
* name - название статуса (строка, не может быть пустой, уникально).

Ограничения:

* Внешние ключи должны ссылаться на существующие записи в соответствующих таблицах;
* Поля created\_at должны автоматически устанавливаться текущим временем;
* Поле price не может быть отрицательным или равным нулю;
* Поля quantity не могут быть отрицательными или равными нулю;
* Поле is\_main должно быть булевым значением;
* Поля name и description не могут быть пустыми;
* Поле name в таблице shelves должно быть уникальным;
* Поле name в таблице status должно быть уникальным.

В таблице status должны быть добавлены начальные значения ('created', 'in\_process', 'completed').

В таблице orders должно быть добавлено ограничение внешнего ключа, ссылающегося на таблицу status.