

Описание проекта

Проект URL Shortener - это веб-сервис, который предоставляет возможность создавать сокращенные URL и перенаправлять на исходные длинные URL. Сервис включает два основных API-эндпоинта:

- **POST /shorten:** принимает JSON-запрос с длинным URL и возвращает сокращенный URL.
- **GET /{shortURL}:** перенаправляет на исходный длинный URL.

Структура проекта

url-shortener/

```
├─ main.go
├─ main_test.go
├─ go.mod
└─ go.sum
```

Основные компоненты

1. **main.go:** основной файл с логикой приложения.
2. **main_test.go:** файл с тестами для проверки корректности работы сервиса.
3. **go.mod** и **go.sum:** файлы, управляющие зависимостями Go.

Установка и запуск

Требования

- Go 1.18 или выше
- MariaDB/MySQL сервер

Настройка базы данных

Создайте базу данных urlshortener и таблицу urls:

```
CREATE DATABASE urlshortener;
```

```
USE urlshortener;
```

```
CREATE TABLE urls (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    original_url TEXT NOT NULL,
```

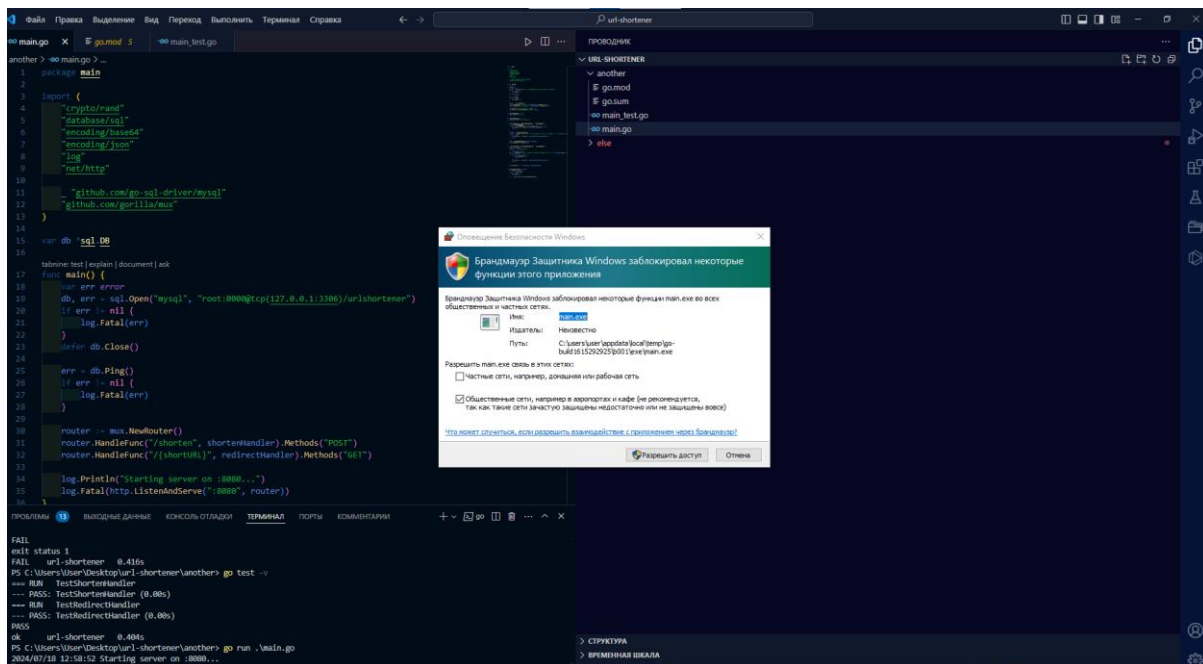
```
short_url VARCHAR(255) NOT NULL UNIQUE  
);
```

```
short_url VARCHAR(255) NOT NULL UNIQUE  
);
```

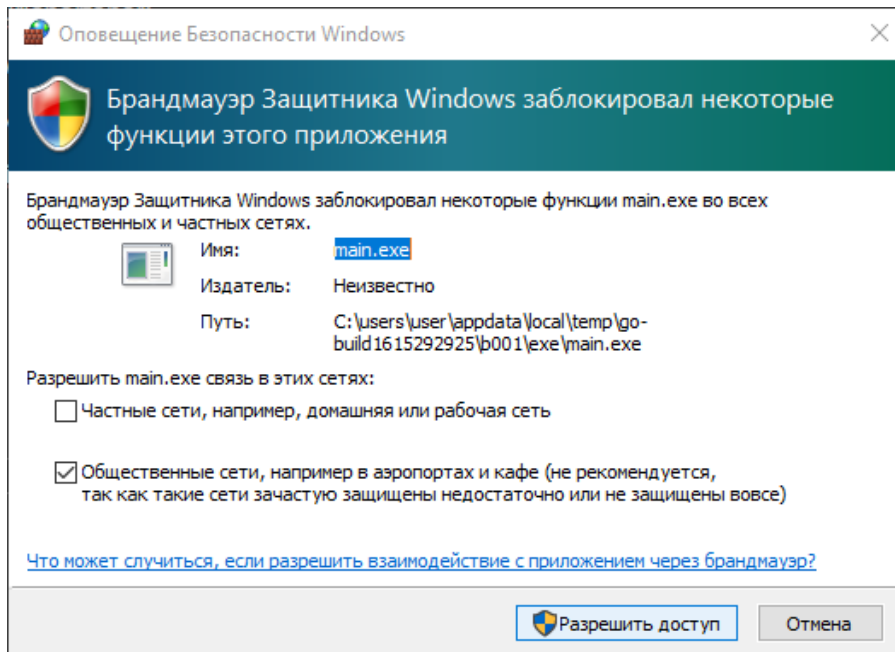
Настройка и запуск приложения

1. Склонируйте репозиторий
git clone https://github.com/yourusername/url-shortener.git
cd url-shortener
2. Установите зависимости:
go mod tidy
3. Запустите приложение:
go run main.go

Запускаем main.go с помощью команды go run .\main.go



Запускаем процесс



Отображение рабочего процесса

```
2024/07/18 12:58:52 Starting server on :8080...  
[]
```

API-эндпоинты

POST /shorten

Описание: Принимает JSON-запрос с длинным URL и возвращает сокращенный URL.

Пример запроса:

```
{  
  "url": "http://example.com"  
}
```

Пример ответа:

```
{  
  "short_url": "randomShortURL"  
}
```

Пример использования с Postman

POST /shorten

В Postman создайте новый запрос типа POST.

Установите URL на <http://localhost:8080/shorten>.

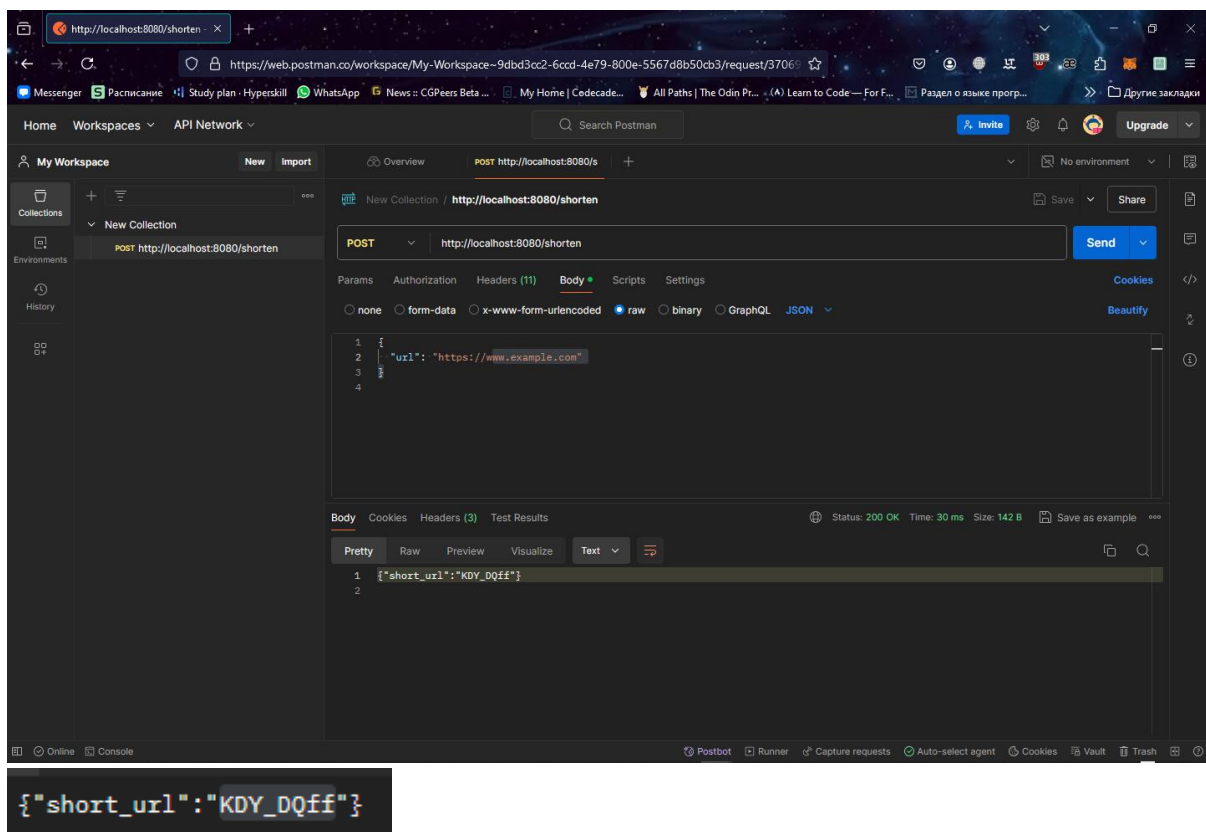
Перейдите на вкладку Body, выберите raw и установите формат JSON.

Введите JSON:

Делаем POST запрос и получаем **KDY_DQff** случайно сгенерированный с помощью функции generateShortURL

```
{
  "url": "http://example.com"
}
```

Нажмите Send. В ответе вы получите сокращенный URL.



Обновляем базу данных и видим отображение в ней

Unnamed/urlshortener/urls - HeidiSQL 12.6.0.6765

Файл Редактировать Поиск Запрос Инструменты Переход Помощь

Хост: 127.0.0.1 База данных: urlshortener Таблица: urls Данные Запрос.sql Запрос #2.sql X Запрос #3.sql X Запрос #4.sql X

Фильтр баз данных Фильтр таблиц

Unnamed

- information_schema
- mysql
- performance_schema
- sys
- urlshortener 32,0 KiB
 - urls 32,0 KiB

1 SELECT * FROM urls;

2

Filter ...

- Столбцы urls
- Функции SQL
- Ключевые слова SQL

#	id	original_url	short_url	created_at
1	1	https://www.example.com	20240718100250	2024-07-18 10:02:50
2	2	https://www.example.com	20240718102355	2024-07-18 10:23:55
3	3	https://www.example.com	H4C4H9W	2024-07-18 10:42:42
4	4	https://www.example.com	9IesaNfe	2024-07-18 10:42:46
5	5	https://www.example.com	E7UJ1N+S	2024-07-18 10:46:24
6	6	https://www.example.com	JwIAuHh	2024-07-18 10:46:27
7	7	https://www.example.com	G-ON0Sxb	2024-07-18 12:15:01
8	8	https://www.example.com	KDY_DQff	2024-07-18 13:00:31

49 /* Запрошено строк: 0 Найденные строки: 7 Предупреждения: 0 Длительность 1 запрос: 0,000 сек. */

50 SELECT * FROM urls;

51 /* Запрошено строк: 0 Найденные строки: 7 Предупреждения: 0 Длительность 1 запрос: 0,000 сек. */

52 SELECT * FROM urls;

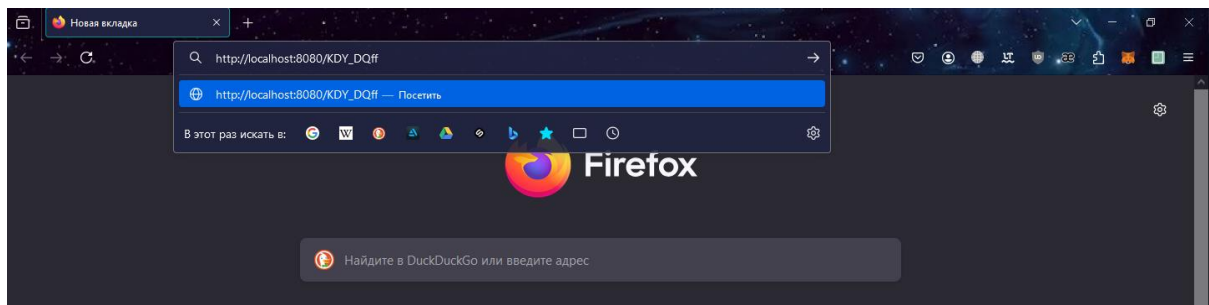
53 /* Запрошено строк: 0 Найденные строки: 8 Предупреждения: 0 Длительность 1 запрос: 0,000 сек. */

Подключено: 00:50 MariaDB 11.4.2 Время работы: 01:41 h Серверное время: 1 Ожидание.

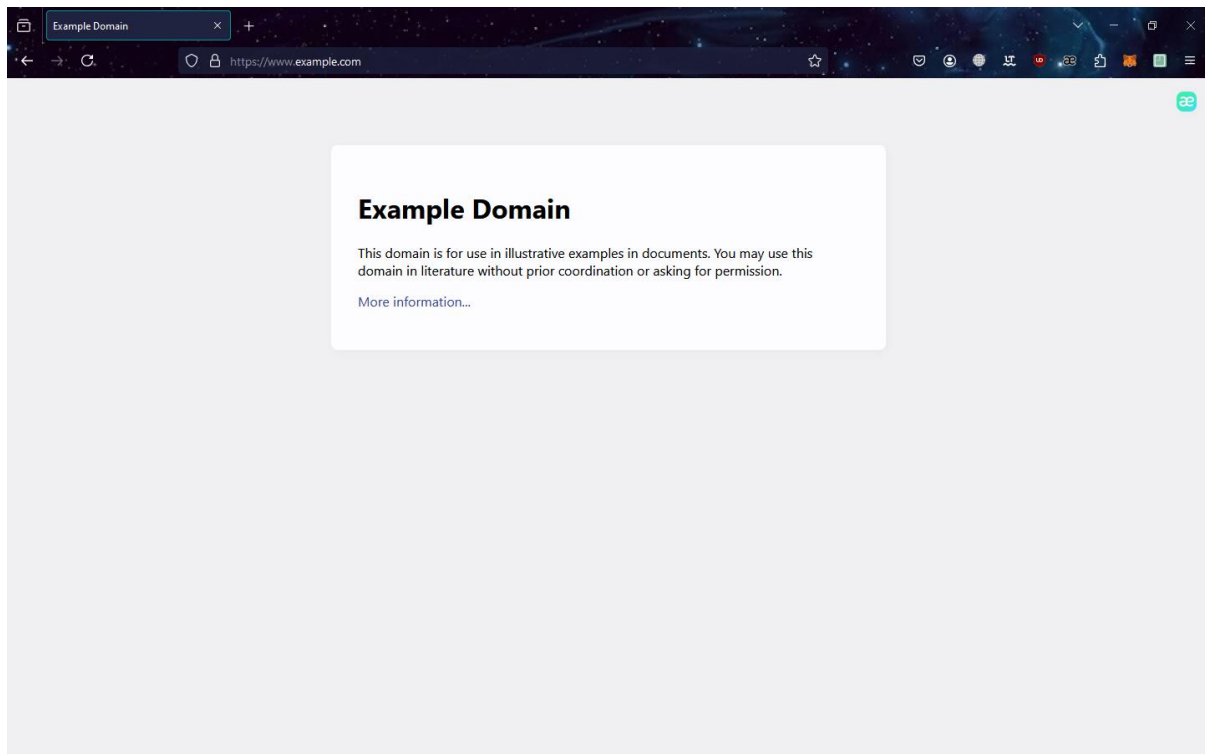
8	8	https://www.example.com	KDY_DQff	2024-07-18 13:00:31
---	---	-------------------------	----------	---------------------

Переходим на локально запущенный сервер по адресу 8080

http://localhost:8080/shorten/KDY_DQff

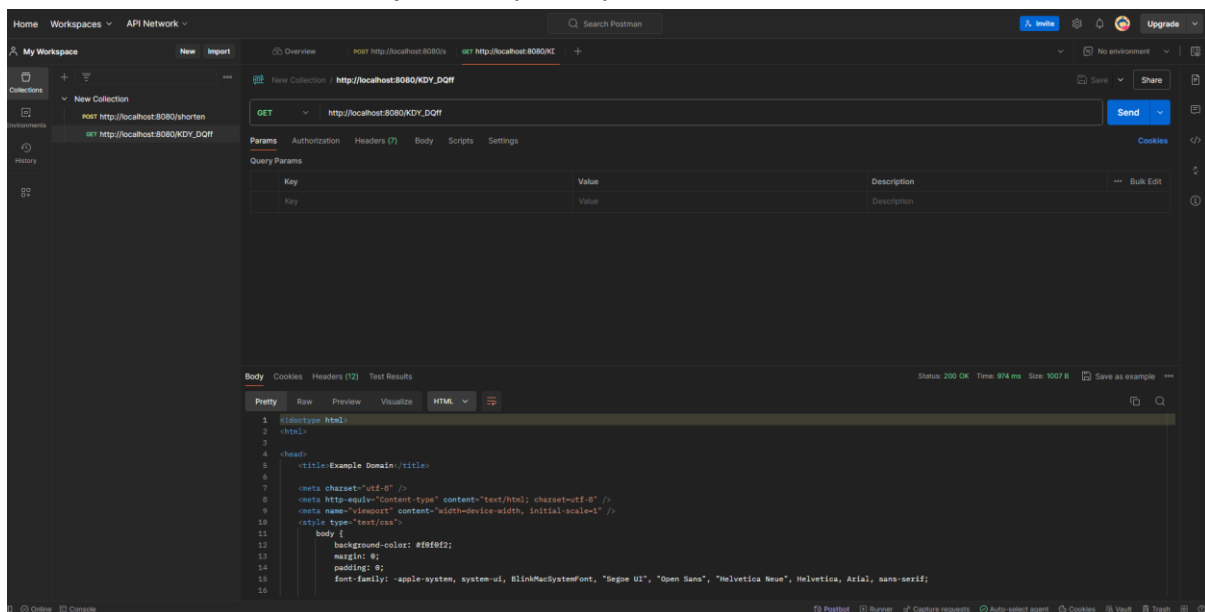


Получаем в ответ страничку о использовании заглушки example



GET /{shortURL}

- В Postman создайте новый запрос типа GET.
- Установите URL на <http://localhost:8080/{shortURL}>, где {shortURL} замените на сокращенный URL, полученный ранее к примеру **KDY_DQff**
- Нажмите Send. Вы будете перенаправлены на исходный длинный URL



Тестирование

Для запуска тестов используйте команду:

```
go test -v
```

Выполняем тесты

```
PS C:\Users\User\Desktop\url-shortener\another> go test -v
=== RUN   TestShortenHandler
--- PASS: TestShortenHandler (0.00s)
=== RUN   TestRedirectHandler
--- PASS: TestRedirectHandler (0.00s)
PASS
ok      url-shortener  0.404s
```

Тестирование функции **TestShortenHandler**.

Эта функция обрабатывает POST-запрос к конечной точке /shorten, которая принимает JSON-полезную нагрузку с URL-адресом и возвращает сокращенную версию этого URL-адреса.

Тест использует пакет sqlmock для создания моковой базы данных и устанавливает ожидания для операций с базой данных.

В этом случае ожидается, что будет выполнено инструкция INSERT с указанным URL-адресом и сгенерированным коротким URL-адресом.

Тест создает POST-запрос с JSON-полезной нагрузкой, содержащей URL-адрес для сокращения. Затем он настраивает запись ответа для захвата HTTP-ответа. Функция `http.HandlerFunc(shortenHandler)` вызывается для обработки запроса и записи ответа в `rr`.

После этого выполняются проверки с помощью пакета `assert`: ожидается, что код ответа будет `http.StatusOK (200)`, и возвращенный короткий URL-адрес не должен быть пустым. Наконец, тест проверяет, были ли выполнены все ожидаемые ожидания с помощью `mock.ExpectationsWereMet()`.

Если есть невыполненные ожидания, тест завершается с ошибкой.

Тестовая функция **TestRedirectHandler** для обработчика HTTP-запросов `redirectHandler`.

Эта функция отвечает за перенаправление пользователей на исходный URL-адрес на основе короткого URL-адреса, указанного в пути запроса.

В тесте используется пакет `sqlmock` для создания моковой базы данных и ожиданий для SQL-запросов, которые будут выполняться во время теста.

Объект `mockDB` используется для замены фактического подключения к базе данных в коде приложения. Ожидается, что при выполнении запроса GET к маршруту `/shortURL123` обработчик `redirectHandler` вернет ответ с кодом состояния 302 (Found) и

заголовком `Location` со значением `"http://example.com"`, которое указывает на исходный URL-адрес.

После выполнения запроса и получения ответа тест проверяет, были ли выполнены все ожидаемые ожидания с помощью `mock.ExpectationsWereMet()`.

Если есть невыполненные ожидания, тест выдает ошибку.