Професионална гимназия по високи технологии „А. С. Попов“



Доклад

по Бази данни:

“ Свързване на база данни с приложение”

Изготвил: Виктор Христов

гр. София

2024

# Измисляне на тема

За да има върху какво да работим първо ни трябва тема. Моята е сайт за портфолия, като всеки участник има следните неща – първо и последно име, имейл, снимка за аватар, телефонен номер и две опции за адрес и LinkedIn профил.

# Решаване на база данни

За моята база ще използвам онлайн сървър – Supabase. Това е платформа за бекенд като услуга, която предоставя инструменти за лесно изграждане и управление на бекенд приложения. Тя е отворен код и предлага функционалности, подобни на Firebase, но базирани на SQL бази данни (PostgreSQL).

# Настройване на базата данни

След като имаме идея и база данни е време да си направим схемата. Тя ще бъде съставена от 2 таблици. Една, която ще пази информацията за притежателя на портфолиото и една за адресите на хората.

## User table – MainBase

Първата таблица е User table или както аз съм я кръстил – MainBase. В нея ще има информация за потребителя - първо и последно име, имейл, снимка за аватар, телефонен номер. Всички тези данни ще са от тип string (низ). Кода за това е

create table

  public."MainBase" (

    id bigint generated by default as identity,

    email text null,

    "firstName" text null,

    "phoneNum" text null,

    "addressId" bigint null,

    "linkedInProfile" text null,

    "lastName" text null,

    avatar text null,

    constraint MainBase\_pkey primary key (id),

    constraint MainBase\_addressId\_fkey foreign key ("addressId") references "SecondBase" (id)

  ) tablespace pg\_default;

С този код създаваме публична таблица MainBase. Тя има основен идентификатор Id, който е използван за основен (primary key). След това имаме имейл, първо, последно име, телефонен номер и linkedIn profile, които са от тип текст, защото това е string типа. Освен това може да видим addressId е от тип bigint или int8. Това е типа за цифри и id-та. Накрая може да видим два constraint-а. Първия е за primary key или основен ключ, а втория е за foreign key, този който ще използваме за достъп до втората таблица. Foreign key addressId се свързва с id от втората таблица SecondBase.

* 1. Address table – SecondBase

Втора таблица е Address table или както аз съм я кръстил – SecondBase. В нея ще

се съдържа информацията за адресите на потребителите. Ще има две поета – адрес и град, и двете от тип text (string). Те също си имат Id, което е идентификатора им за достъп както и primary key. Кода за създаването й изглежда по следния начин:

create table

  public."SecondBase" (

    id bigint generated by default as identity,

    address text null,

    city text null,

    constraint SecondBase\_pkey primary key (id)

  ) tablespace pg\_default;

# Структура

След като вече имаме идея и готова база ни трябва място където ще я вградим,

тоест проект. Аз реших че ще го правя под формата на сайт. За него ще използваме HTML, CSS, JavaScript.

## Създаване

За да започнем изобщо да изписваме данни, първо трябва да ги създадем, за това ще

Използваме форма A screenshot of a contact form

Description automatically generated. В тази форма записваме данните за човека, които после ще изпишем. След кликане на бутона submit, ние изпращаме post заявка на сървъра. Ако използваме готовата библиотека ще изглежда по следния начин - await supabase.from('MainBase').insert(formData), където formData е обект с данни, ако го искаме с sql код ще изглежда така –

INSERT INTO MainBase (email, firstName, lastName, avatar, linkedInProfile, phoneNum, address)

VALUES ('user@example.com', 'John', 'Doe', 'avatar\_url\_here', 'linkedin.com/in/johndoe', '123-456-7890', '123 Main St'); (примерни данни)

## Каталог

Каталогът е просто мястото където ще визуализираме всички хора. При него заявката

е най-проста. Тя е просто SELECT \* FROM MainBase, а ако е с библиотека - await supabase.from('MainBase').select(). След това тези получени данни просто ги визуализираме

A screenshot of a cartoon character

Description automatically generated

## Детайли

Детайлите са просто по-специфичен каталог. Тяхната заявка просто ще търси човек,

чието ID ние подадем.

SELECT \*

FROM MainBase

WHERE id = id;

На чист SQL e така, с библиотека - supabase.from('MainBase').select().eq('id', id);

A screenshot of a cartoon character

Description automatically generated

## Редактиране

Редактирането е може би най-трудното, защото трябва да се направят 2 заявки, една

за да се намери, кой човек редактираме и една за самата редакция. Първата заяква е едно към едно с тази за детайлите.

SELECT \*

FROM MainBase

WHERE id = id;

На чист SQL e така, с библиотека - supabase.from('MainBase').select().eq('id', id);

С данните попълваме таблицата за редакция - A screenshot of a computer

Description automatically generated  
След като се submit-не формата, заявка подобна на тази за създаването се пуска, изглежда по следния начин -   
UPDATE MainBase

SET

email = data.email,

firstName = data.firstName,

lastName = data.lastName,

avatar = data.avatar,

linkedInProfile = data. linkedInProfile,

phoneNum = data.phoneNum,

address = data.address

WHERE id = data.id;

Или supabase.from('MainBase').update(formData).eq('id', idConst), където formData е обект с данни подобни на тези в sql кода, а idConst е променлива за id. След редакция, ние отиваме в каталога.

## Изтриване

Последното нещо което ни трябва е изтриването или delete. За delete заяката е още

по-проста.

DELETE FROM MainBase

WHERE id = id;

Ако го пишем с библиотеката е - supabase.from('MainBase').delete().eq('id', id).

# Край

Тъй като проектът е по час бази данни, не мисля че е уместно да говоря за кода и

функциите като цяло. Няма нужда да споменавам и навигация и това как нещата се рендерират в кода. Така че това май е основното.