**Developer**

Abaixo está um exemplo de um projeto CRUD simples em Go (versão 1.22.5) utilizando o PostgreSQL. Vou descrever os passos necessários e fornecer o código que você pode copiar para um documento .docx.

### **Exemplo de Projeto CRUD em Go com PostgreSQL**

#### **1. Pré-requisitos**

* Go instalado (versão 1.22.5)
* PostgreSQL instalado e rodando
* Bibliotecas Go necessárias:
* **github.com/lib/pq** para conectar ao PostgreSQL
* **github.com/gorilla/mux** para roteamento

#### **2. Criando o Banco de Dados**

Primeiro, você precisa criar um banco de dados e uma tabela. Conecte-se ao PostgreSQL e execute os seguintes comandos:

CREATE DATABASE exemplo\_crud;

\c exemplo\_crud;

CREATE TABLE usuarios (

id SERIAL PRIMARY KEY,

nome VARCHAR(100) NOT NULL,

email VARCHAR(100) NOT NULL

);

#### **3. Estrutura do Projeto**

Crie uma nova pasta para o seu projeto e a seguinte estrutura de arquivos:

/exemplo-crud

├── main.go

#### **4. Código do main.go**

Cole o código abaixo em **main.go**:

package main

import (

"database/sql"

"encoding/json"

"net/http"

"github.com/gorilla/mux"

\_ "github.com/lib/pq"

)

type Usuario struct {

ID int `json:"id"`

Nome string `json:"nome"`

Email string `json:"email"`

}

var db \*sql.DB

var err error

func inicializarDB() {

connStr := "user=seu\_usuario dbname=exemplo\_crud sslmode=disable"

db, err = sql.Open("postgres", connStr)

if err != nil {

panic(err)

}

}

func criarUsuario(w http.ResponseWriter, r \*http.Request) {

var usuario Usuario

json.NewDecoder(r.Body).Decode(&usuario)

sqlStatement := `INSERT INTO usuarios (nome, email) VALUES ($1, $2) RETURNING id`

id := 0

err = db.QueryRow(sqlStatement, usuario.Nome, usuario.Email).Scan(&id)

if err != nil {

w.WriteHeader(http.StatusInternalServerError)

return

}

usuario.ID = id

w.Header().Set("Content-Type", "application/json")

json.NewEncoder(w).Encode(usuario)

}

func obterUsuarios(w http.ResponseWriter, r \*http.Request) {

rows, err := db.Query("SELECT id, nome, email FROM usuarios")

if err != nil {

w.WriteHeader(http.StatusInternalServerError)

return

}

defer rows.Close()

var usuarios []Usuario

for rows.Next() {

var usuario Usuario

if err := rows.Scan(&usuario.ID, &usuario.Nome, &usuario.Email); err != nil {

w.WriteHeader(http.StatusInternalServerError)

return

}

usuarios = append(usuarios, usuario)

}

w.Header().Set("Content-Type", "application/json")

json.NewEncoder(w).Encode(usuarios)

}

func obterUsuario(w http.ResponseWriter, r \*http.Request) {

vars := mux.Vars(r)

id := vars["id"]

var usuario Usuario

sqlStatement := `SELECT id, nome, email FROM usuarios WHERE id=$1`

row := db.QueryRow(sqlStatement, id)

err = row.Scan(&usuario.ID, &usuario.Nome, &usuario.Email)

if err != nil {

w.WriteHeader(http.StatusNotFound)

return

}

w.Header().Set("Content-Type", "application/json")

json.NewEncoder(w).Encode(usuario)

}

func atualizarUsuario(w http.ResponseWriter, r \*http.Request) {

vars := mux.Vars(r)

id := vars["id"]

var usuario Usuario

json.NewDecoder(r.Body).Decode(&usuario)

sqlStatement := `UPDATE usuarios SET nome=$1, email=$2 WHERE id=$3`

\_, err = db.Exec(sqlStatement, usuario.Nome, usuario.Email, id)

if err != nil {

w.WriteHeader(http.StatusInternalServerError)

return

}

w.WriteHeader(http.StatusNoContent)

}

func deletarUsuario(w http.ResponseWriter, r \*http.Request) {

vars := mux.Vars(r)

id := vars["id"]

sqlStatement := `DELETE FROM usuarios WHERE id=$1`

\_, err = db.Exec(sqlStatement, id)

if err != nil {

w.WriteHeader(http.StatusInternalServerError)

return

}

w.WriteHeader(http.StatusNoContent)

}

func main() {

inicializarDB()

defer db.Close()

r := mux.NewRouter()

r.HandleFunc("/usuarios", criarUsuario).Methods("POST")

r.HandleFunc("/usuarios", obterUsuarios).Methods("GET")

r.HandleFunc("/usuarios/{id}", obterUsuario).Methods("GET")

r.HandleFunc("/usuarios/{id}", atualizarUsuario).Methods("PUT")

r.HandleFunc("/usuarios/{id}", deletarUsuario).Methods("DELETE")

http.ListenAndServe(":8000", r)

}

#### **5. Executando o Projeto**

1. Abra o terminal e navegue até a pasta do projeto.
2. Execute o comando para instalar as dependências:

go get github.com/lib/pq

go get github.com/gorilla/mux

3.Inicie o servidor:

go run main.go

1. O servidor estará rodando em **http://localhost:8000**.

#### **6. Testando as Rotas**

* **Criar Usuário:**

curl -X POST -H "Content-Type: application/json" -d '{"nome":"João","email":"joao@example.com"}' <http://localhost:8000/usuarios>

* **Obter Todos os Usuários:**

curl http://localhost:8000/usuarios

* **Obter Usuário por ID:**

curl http://localhost:8000/usuarios/1

* **Atualizar Usuário:**

curl -X PUT -H "Content-Type: application/json" -d '{"nome":"João Silva","email":"joaosilva@example.com"}' <http://localhost:8000/usuarios/1>

* **Deletar Usuário:**

curl -X DELETE <http://localhost:8000/usuarios/1>