﻿**Projeto Em CRUD com Go Lang e PostgreSQL**

### **Requiserar projetos simples, vamos criar um exemplo de projeto em CRUD (Create, Read, Update, Delete) utilizando a linguagem Go Lang versão 1.22.5 e o banco de dados PostgreSQL.**

**Requisitos prévios:**

* Instalar o Go Lang na sua máquina
* Instalar o pacote database/sql para interagir com o PostgreSQL
* Criar um novo projeto em Go Lang

**Instalação dos pacotes necessários:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | |  |  | | --- | --- | | bash |  | | | |  | | --- | | go get -u github.com/lib/pq | | |

**Criando a estrutura do banco de dados**

Vamos criar uma tabela chamada users no nosso banco de dados com as seguintes colunas:

|  |  |
| --- | --- |
| **Coluna** | **Tipo** |
| id | int (chave primária) |
| nome | varchar(255) |
| email | varchar(255) |

**Criando o arquivo** main.go**:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | |  |  | | --- | --- | | go |  | | | |  | | --- | | package main  import (  "database/sql"  "fmt"  \_ "github.com/lib/pq"  )  // User representa um usuário do sistema  type User struct {  ID int `json:"id"`  Nome string `json:"nome"`  Email string `json:"email"`  }  func criarConexao() (\*sql.DB, error) {  strConn := fmt.Sprintf("user=%s password=% host=localhost dbname=meu\_banco sslmode=disable",  "seu\_usuario", "sua\_senha")  return sql.Open("postgres", strConn)  } | | |

**Criando as funções de CRUD:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | |  |  | | --- | --- | | go |  | | | |  | | --- | | func criarUsuario(db \*sql.DB, nome string, email string) (int64, error) {  query := `  INSERT INTO users(nome,email)  VALUES($1,$2)  RETURNING id;  `  var usuario User  err := db.QueryRow(query, nome, email).Scan(&usuario.ID)  return usuario.ID, err  }  func lerUsuarios(db \*sql.DB) ([]User, error) {  query := `  SELECT \*  FROM users;  `  rows, err := db.Query(query)  if err != nil {  return nil, err  }  defer rows.Close()  var usuarios []User  for rows.Next() {  err = rows.Scan(&usuarios[0].ID, &usuarios[0].Nome, &usuarios[0].Email)  if err != nil {  return nil, err  }  usuarios = append(usuarios, User{ID: usuarios[0].ID})  }  return usuarios, nil  }  func atualizarUsuario(db \*sql.DB, id int64, nome string) error {  query := `  UPDATE users SET nome=$1 WHERE id=$2;  `  \_, err := db.Exec(query, nome, id)  if err != nil {  return err  }  return nil  }  func deletarUsuario(db \*sql.DB, id int64) error {  query := `  DELETE FROM users WHERE id=$1;  `  \_, err := db.Exec(query, id)  if err != nil {  return err  }  return nil  } | | |

**Criando o arquivo** main.go **(continuação):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | |  |  | | --- | --- | | go |  | | | |  | | --- | | func main() {  db, err := criarConexao()  if err != nil {  fmt.Println(err)  return  }  defer db.Close()  nome := "João"  email := "joao@example.com"  id, err := criarUsuario(db, nome, email)  if err == nil {  fmt.Printf("Usuário criado com sucesso! ID: %d\n", id)  } else {  fmt.Println(err)  }  usuarios, err := lerUsuarios(db)  if err != nil {  fmt.Println(err)  } else {  for \_, usuario := range usuarios {  fmt.Printf("%+v\n", usuario)  }  }  nome = "Maria"  atualizarUsuario(db, id, nome)  deletarUsuario(db, id)  } | | |

**Executando o projeto:**

Para executar o projeto, você precisará criar um arquivo main.go com as instruções acima e executá-lo usando o comando:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | |  |  | | --- | --- | | bash |  | | | |  | | --- | | go run main.go | | |

Isso criará uma conexão com o banco de dados PostgreSQL e realizará as operações CRUD.

**Observação:**

Lembre-se de substituir os valores "seu\_usuario" e "sua\_senha" pelo seu usuário e senha reais do PostgreSQL. Além disso, certifique-se de que você tenha criado a tabela users no banco de dados com as colunas corretas.

**Copia para um documento .docx:**

Você pode copiar todas as instruções acima em uma única vez e criar um arquivo .docx para armazenar o projeto.