package main

import (

"encoding/json"

"fmt"

"io/ioutil" // Added for ioutil.ReadFile

"log"

"net/http" // Added for HTTP server functionality

"github.com/graphql-go/graphql"

"github.com/graphql-go/handler"

)

func main() {

// Schema

// Read the schema from the .graphql file

schemaAsString, err := getSchema("./schema.graphql")

if err != nil {

log.Fatalf("Failed to read schema: %v", err) // Changed panic to log.Fatalf for better error handling

}

// Parse the read schema string into a GraphQL schema

diaSchema := graphql.MustParseSchema(schemaAsString, &resolver.RootResolver{}, graphql.UseFieldResolvers())

// Create a new HTTP multiplexer

mux := http.NewServeMux()

// Serve a simple page at the root URL

mux.HandleFunc("/", func(w http.ResponseWriter, r \*http.Request) {

\_, err := w.Write(page)

if err != nil {

log.Printf("Failed to write response: %v", err) // Log instead of silently returning

return

}

})

// Handle GraphQL queries at /query using the defined schema

mux.Handle("/query", handler.New(&handler.Config{Schema: diaSchema}))

// Start the server

listenPort := ":1111" // Default port if not set in environment

log.Printf("Listening on %s", listenPort)

log.Fatal(http.ListenAndServe(listenPort, mux)) // Utilize the listenPort variable

}