DEPLOY CHAINCODE—

package main

import (

"fmt"

"github.com/hyperledger/fabric-chaincode-go/shim"

"github.com/hyperledger/fabric-protos-go/peer"

)

type LogChaincode struct {

}

func (t \*LogChaincode) Init(stub shim.ChaincodeStubInterface) peer.Response {

return shim.Success(nil)

}

func (t \*LogChaincode) Invoke(stub shim.ChaincodeStubInterface) peer.Response {

function, args := stub.GetFunctionAndParameters()

if function == "storeLog" {

return t.storeLog(stub, args)

}

return shim.Error("Invalid function name")

}

func (t \*LogChaincode) storeLog(stub shim.ChaincodeStubInterface, args []string) peer.Response {

if len(args) != 2 {

return shim.Error("Expecting 2 arguments: logID and logData")

}

logID := args[0]

logData := args[1]

// Save the log data to the ledger

err := stub.PutState(logID, []byte(logData))

if err != nil {

return shim.Error(fmt.Sprintf("Failed to store log: %s", err.Error()))

}

return shim.Success(nil)

}

func main() {

err := shim.Start(new(LogChaincode))

if err != nil {

fmt.Printf("Error starting LogChaincode: %s", err)

}

}