**Internal java Script:**

<!DOCTYPE html>

<html>

<body>

<p id = "para\_id">At the time of Go’s initiation, complexity was the major concern in Software Development.</p>

<script>

document.getElementById("para\_id").innerHTML = "Internal Java Script Example";

</script>

</body>

</html>

**External java Script:**

<!DOCTYPE html>

<html>

<head>

<script type="text/javascript" src="myjs.js"></script>

</head>

<body onload="function\_onload()">

<p id = "para\_id">At the time of Go’s initiation, complexity was the major concern in Software Development.</p>

</body>

</html>

**Myjs.js:**

function function\_onload()

{

var div\_ele = document.createElement("div");

div\_ele.innerHTML = "External Java Script called";

document.body.appendChild(div\_ele);

}

**Inline css:**

<!DOCTYPE html>

<html>

<body>

<p style="color: red; text-align: center; font-size: 50px ;"> At the time of Go’s initiation, complexity was the major concern in Software Development. Usually the development used to be done with C++ or Java but there were some challenges with them like security, memory utilization, speed, efficiency, debugging etc. Software should have the power to function correctly against some potential risk produced by malicious attacks. </p>

</body>

</html>

**Internal css:**

<!DOCTYPE html>

<html>

<head>

<style>

p, div

{

color: red;

text-align: center;

font-size: 50px;

}

</style>

</head>

<body>

<p>At the time of Go’s initiation, complexity was the major concern in Software Development.</p>

<div>

Usually the development used to be done with C++ or Java but there were some challenges with them like security, memory utilization, speed, efficiency, debugging etc.

</div>

<span>

Software should have the power to function correctly against some potential risk produced by malicious attacks.

</span>

</body>

</html>

**External css:**

**HTML file**

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" type="text/css" href="mycss.css">

</head>

<body>

<p>At the time of Go’s initiation, complexity was the major concern in Software Development.</p>

<div>

Usually the development used to be done with C++ or Java but there were some challenges with them like security, memory utilization, speed, efficiency, debugging etc.

</div>

<span>

Software should have the power to function correctly against some potential risk produced by malicious attacks.

</span>

</body>

</html>

**CSS file**

p, div

{

color: red;

text-align: center;

font-size: 20px;

}

span

{

color: blue;

font-size: 50px;

}

**Gopher\_example.go**

package main

import "github.com/gopherjs/gopherjs/js"

func main() {

js.Global.Get("document").Call("write", "Hello I am an example of gopherjs")

}

<!DOCTYPE html>

<html>

<head>

<script type="text/javascript" src="myjs.js"></script>

</head>

<body>

<p id = "para\_id">At the time of Go’s initiation, complexity was the major concern in Software Development.</p>

</body>

</html>

<!DOCTYPE html>

<html>

<body bgcolor="orange">

<div>Enter age:<input type="number" id = "age"><div>

<button id="btn\_id" type="button">Verify</button>

<div id="message"></div>

<script type="text/javascript" src="myjs.js"></script>

</body>

</html>

package main

import "honnef.co/go/js/dom"

func main() {

d := dom.GetWindow().Document()

age\_ele := d.GetElementByID("age").(\*dom.HTMLInputElement)

btn\_ele := d.GetElementByID("btn\_id").(\*dom.HTMLButtonElement)

msg\_ele := d.GetElementByID("message").(\*dom.HTMLDivElement)

btn\_ele.AddEventListener("click", false, func(event dom.Event) {

if age\_ele.Value=="50"{

msg\_ele.SetInnerHTML("Age is valid")

} else{

msg\_ele.SetInnerHTML("Age is not valid")

}

})

}

<!doctype html>

<html lang="en">

<head>

<script src="path\_of\_downloaded\_jquery\_file"></script>

</head>

<body bgcolor="orange">

<button id ="btn">Show / Hide</button>

<div id="msg"> Hello Go</span>

<script src="myjquery.js"></script>

</body>

</html>

package main

import "github.com/gopherjs/jquery"

var jQuery = jquery.NewJQuery

const (

btn = "button#btn"

msg\_ele = "div#msg"

)

func main() {

jQuery(btn).On(jquery.CLICK, func(e jquery.Event) {

jQuery(msg\_ele).Toggle()

})

}

**Build an application:**

**Server.go**

package main

import (

"net/http"

)

func main() {

http.ListenAndServe(":8080", nil)

}

package main

import (

"log"

"html/template"

"net/http"

)

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

t , err := template.ParseFiles("signup.html")

if err != nil {

log.Print("template parsing error: ", err)

}

err = t.Execute(w, nil)

if err != nil {

log.Print("template executing error: ", err)

}

}

func main() {

http.HandleFunc("/", Signup)

http.ListenAndServe(":8080", nil)

}

**Signup.html**

<!DOCTYPE html>

<html style="background-image: url(‘signup.jpg’);background-size: cover;">

<head>

<style type="text/css">

.data\_form{

margin-top: 15%;

margin-left: 29%;

}

.label{

font-size: 25px;

font-weight: 25px;

display: inline-block;

min-width: 22%;

}

.in\_field{

width: 27%;

display: inline-block;

}

.goforlogin{

margin-left: 39%;

}

div{

margin-bottom: 1.5%;

}

</style>

</head>

<body>

<form action="/AddUser" method="post" class="data\_form">

<div><span class="label">First Name:</span><span ><input class="in\_field" type="text" name ="first\_name" required/></span></div>

<div><span class="label">Last Name:</span><span><input class="in\_field" type="text" name ="last\_name"/></span></div>

<div><span class="label">Email Id:</span><span><input class="in\_field" type="Email" name ="email" required /></span></div>

<div><span class="label">Password:</span><span><input class="in\_field" type="password" name ="password" required /></span></div>

<div><span class="label">Phone Number:</span><span><input class="in\_field" type="tel" name ="phone" pattern="[0-9]{10}" required /></span></div>

<input type="submit" style="width: 14%;font-size: 20px;margin: 4% 0% 0% 13%;" value="Confirm"/>

</form>

<a class="goforlogin" href="login.html">Already registered</a>

</body>

</html>

**Server.go**

package main

import (

"log"

"html/template"

"net/http"

"database/sql"

"fmt"

\_ "github.com/go-sql-driver/mysql"

)

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

if err := r.ParseForm(); err != nil {

fmt.Fprintf(w, "ParseForm() err: %v", err)

}

fName := r.FormValue("first\_name")

lName := r.FormValue("last\_name")

emailId := r.FormValue("email")

phoneNo := r.FormValue("phone")

password := r.FormValue("password")

// **Create Database connection**

db, err := sql.Open("mysql", "root:@/go\_web")

if err != nil {

panic(err.Error())

}

// **Insert data**

stmtIns, err := db.Query("INSERT INTO user\_info VALUES(?,?,?,?,?)",fName,lName,emailId,phoneNo,password)

if err != nil {

panic(err.Error()) // proper error handling instead of panic in your app

}

defer stmtIns.Close()

t , err := template.ParseFiles("registrationConfirm.html")

if err != nil {

log.Print("template parsing error: ", err)

}

err = t.Execute(w, nil)

}

func main() {

http.HandleFunc("/", Signup)

http.HandleFunc("/AddUser", AddUser)

http.ListenAndServe(":8080", nil)

}

**registrationConfirm.html**

<html>

<body style="background-color:yellow;">

You have successfully registered.

<a href="/login">Login here</a>

</body>

</html>

package main

import (

"log"

"html/template"

"net/http"

"database/sql"

"fmt"

\_ "github.com/go-sql-driver/mysql"

)

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

t , err := template.ParseFiles("login.html")

// if there is an error

if err != nil {

log.Print("template parsing error: ", err)

}

err = t.Execute(w, nil)

if err != nil {

log.Print("template executing error: ", err)

}

}

func main() {

http.HandleFunc("/", Signup)

http.HandleFunc("/AddUser", AddUser)

http.HandleFunc("/login", Login)

http.ListenAndServe(":8080", nil)

}

**Login.html**

<!DOCTYPE html>

<html style="background-image: url('login.jpg');background-size: cover;">

<head>

<style type="text/css">

.data\_form{

margin-top: 15%;

margin-left: 29%;

}

#msg{

color:red;

font-size: 20px;

}

.label{

font-size: 25px;

font-weight: 25px;

display: inline-block;

min-width: 22%;

}

.in\_field{

width: 27%;

display: inline-block;

}

div{

margin-bottom: 1.5%;

}

</style>

</head>

<body>

<form action="/home" method="post" class="data\_form">

<div><span id="msg"></span></div>

<div><span class="label">Email Id:</span><span><input class="in\_field" type="Email" name ="email" required /></span></div>

<div><span class="label">Password:</span><span><input class="in\_field" type="Password" name ="password" required /></span></div>

<input type="submit" style="width: 14%;font-size: 20px;margin: 4% 0% 0% 13%;" value="Confirm"/>

</form>

<script type="text/javascript">

var vars = {};

var parts = window.location.href.replace(/[?&]+([^=&]+)=([^&]\*)/gi, function(m,key,value) {vars[key] = value;});

if (vars["msg"]!==undefined){

document.getElementById("msg").innerHTML = decodeURI(vars["msg"])

}

</script>

</body>

</html>

**Server.go**

package main

import (

"log"

"strings"

"html/template"

"net/http"

"database/sql"

"fmt"

\_ "github.com/go-sql-driver/mysql"

)

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

if err := r.ParseForm(); err != nil {

fmt.Fprintf(w, "ParseForm() err: %v", err)

}

user\_email := r.FormValue("email")

user\_password := r.FormValue("password")

// Create a Database connection

db, err := sql.Open("mysql", "root:@/go\_web")

fmt.Println(db,err)

if err != nil {

panic(err.Error())

}

type UserInfo struct {

First\_name, Last\_name, Email, Password string

Contact int64

}

var info UserInfo

// **Getting data from Database**

quer := "SELECT\* FROM user\_info WHERE email=?"

errr := db.QueryRow(quer,user\_email).Scan(&info.First\_name, &info.Last\_name, &info.Email, &info.Contact, &info.Password)

if errr != nil {

fmt.Println(errr.Error())

// No such Email exists

if strings.Contains(errr.Error(), "no rows") {

http.Redirect(w, r, "http://localhost:8080/login?msg=Invalid Email", 301)

}

}

db.Close() //Close database connection

// Check user password

if user\_password==info.Password{

t , err := template.ParseFiles("home.html")

if err != nil {

log.Print("template parsing error: ", err)

}

err = t.Execute(w, info)

// if there is an error

if err != nil {

log.Print("template executing error: ", err)

}

} else {

//in case of incorrect password

http.Redirect(w, r, "http://localhost:8080/login?msg=Invalid Password", 301)

}

}

func main() {

http.HandleFunc("/", Signup)

http.HandleFunc("/AddUser", AddUser)

http.HandleFunc("/login", Login)

http.HandleFunc("/home", Home)

http.ListenAndServe(":8080", nil)

}

Here is complete server code below:

**server.go**

package main

import (

"log"

"strings"

"html/template"

"net/http"

"database/sql"

"fmt"

\_ "github.com/go-sql-driver/mysql"

)

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

t , err := template.ParseFiles("signup.html")

if err != nil {

log.Print("template parsing error: ", err)

}

err = t.Execute(w, nil)

if err != nil {

log.Print("template executing error: ", err)

}

}

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

if err := r.ParseForm(); err != nil {

fmt.Fprintf(w, "ParseForm() err: %v", err)

}

fName := r.FormValue("first\_name")

lName := r.FormValue("last\_name")

emailId := r.FormValue("email")

phoneNo := r.FormValue("phone")

password := r.FormValue("password")

// **Create Database connection**

db, err := sql.Open("mysql", "root:@/go\_web")

if err != nil {

panic(err.Error())

}

// **insert data**

stmtIns, err := db.Query("INSERT INTO user\_info VALUES(?,?,?,?,?)", fName, lName, emailId, phoneNo, password)

if err != nil {

panic(err.Error()) // proper error handling instead of panic in your app

}

defer stmtIns.Close()

t , err := template.ParseFiles("registrationConfirm.html")

if err != nil {

log.Print("template parsing error: ", err)

}

err = t.Execute(w, nil)

}

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

t , err := template.ParseFiles("login.html")

if err != nil { // if there is an error

log.Print("template parsing error: ", err)

}

err = t.Execute(w, nil)

if err != nil {

log.Print("template executing error: ", err)

}

}

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

if err := r.ParseForm(); err != nil {

fmt.Fprintf(w, "ParseForm() err: %v", err)

}

user\_email := r.FormValue("email")

user\_password := r.FormValue("password")

// Create a Database connection

db, err := sql.Open("mysql", "root:@/go\_web")

fmt.Println(db,err)

if err != nil {

panic(err.Error())

}

type UserInfo struct {

First\_name, Last\_name, Email, Password string

Contact int64

}

var info UserInfo

// Getting data from Database

quer := "SELECT\* FROM user\_info WHERE email=?"

errr := db.QueryRow(quer,user\_email).Scan(&info.First\_name, &info.Last\_name, &info.Email, &info.Contact, &info.Password)

if errr != nil {

fmt.Println(errr.Error())

// No such Email exists

if strings.Contains(errr.Error(), "no rows") {

fmt.Println("no rows here")

http.Redirect(w, r, "http://127.0.0.1:8080/login?msg=Invalid Email", 301)

}

}

db.Close() //Close database connection

// Check user password

if user\_password==info.Password{

fmt.Println("come",info.First\_name)

t , err := template.ParseFiles("home.html")

if err != nil {

log.Print("template parsing error: ", err)

}

err = t.Execute(w, info)

if err != nil { // if there is an error

log.Print("template executing error: ", err)

}

} else {

//in case of incorrect password

http.Redirect(w, r, "http://127.0.0.1:8080/login?msg=Invalid Password", 301)

}

}

func main() {

http.HandleFunc("/", Signup)

http.HandleFunc("/AddUser", AddUser)

http.HandleFunc("/login", Login)

http.HandleFunc("/home", Home)

http.ListenAndServe(":8080", nil)

}