Praktik Kerja Lapangan Universitas Muhammadiyah Prof. Dr Hamka 2023

LAPORAN PERKEMBANGAN WEBSITE PKL

Disusun Oleh :

Nama : Bintang Gede Hartono

NISN : 0055863048

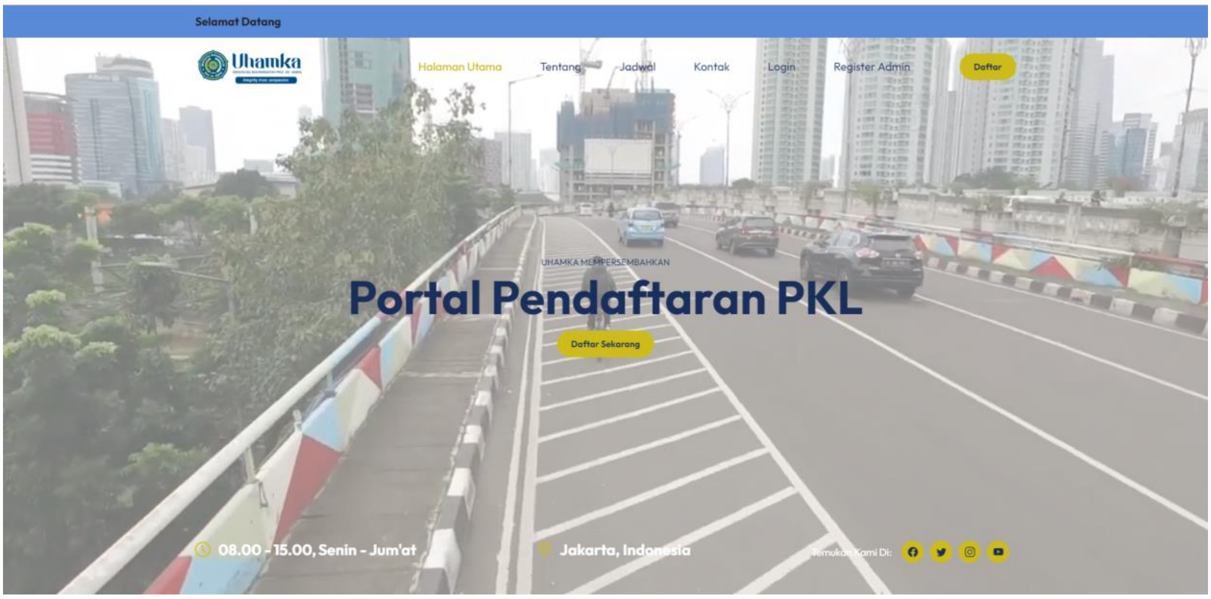
Kelas : XII Rekayasa Perangkat Lunak 2

Berikut adalah Laporan mengenai Perkembangan Website Praktik Kerja Lapangan(PKL) UHAMKA per Senin, 4 September 2023 yang memiliki fitur sebagai berikut:

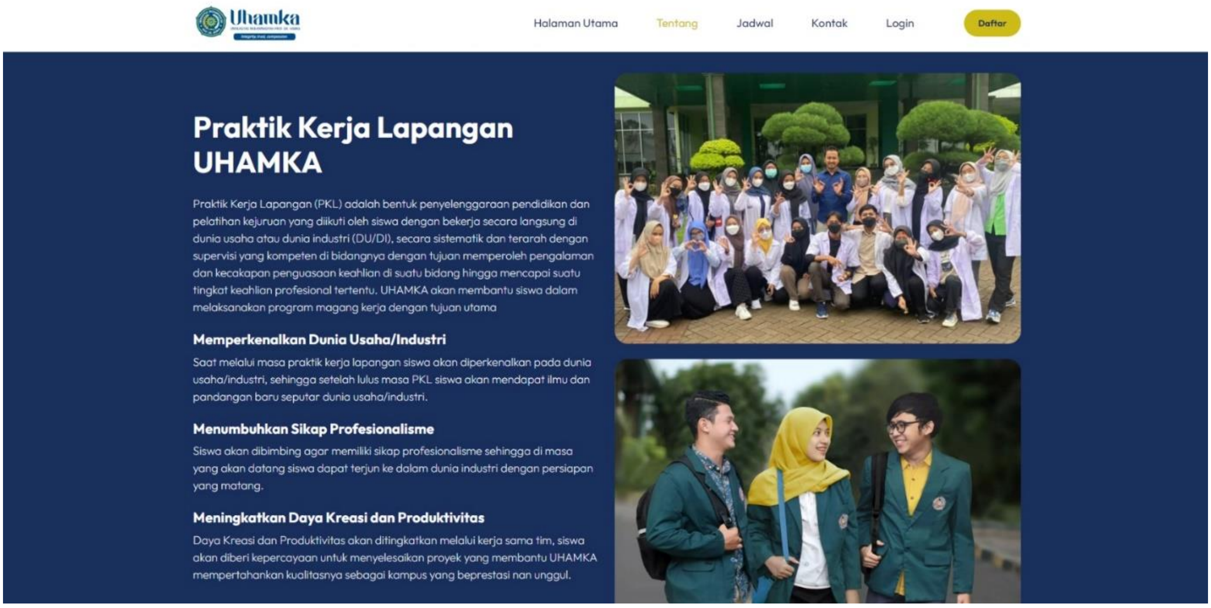
- Home
- Login/Logout
- Tentang
- Register
- Edit as Admin/User
- Social Media Links
- Profile Admin/User

Saya juga mengerjakan bagian login, register, home_admin, home_user, dan edit profile

Home



About



Praktik Kerja Lapangan UHAMKA

Praktik Kerja Lapangan (PKL) adalah bentuk penyelenggaraan pendidikan dan pelatihan kejuruan yang diikuti oleh siswa dengan bekerja secara langsung di dunia usaha atau dunia industri (DU/DI), secara sistematik dan terarah dengan supervisi yang kompeten di bidangnya dengan tujuan memperoleh pengalaman dan kecakapan penguasaan keahlian di suatu bidang hingga mencapai suatu tingkat keahlian profesional tertentu. UHAMKA akan membantu siswa dalam melaksanakan program magang kerja dengan tujuan utama

Memperkenalkan Dunia Usaha/Industri

Saat melalui masa praktik kerja lapangan siswa akan diperkenalkan pada dunia usaha/industri, sehingga setelah lulus masa PKL siswa akan mendapat ilmu dan pandangan baru seputar dunia usaha/industri.

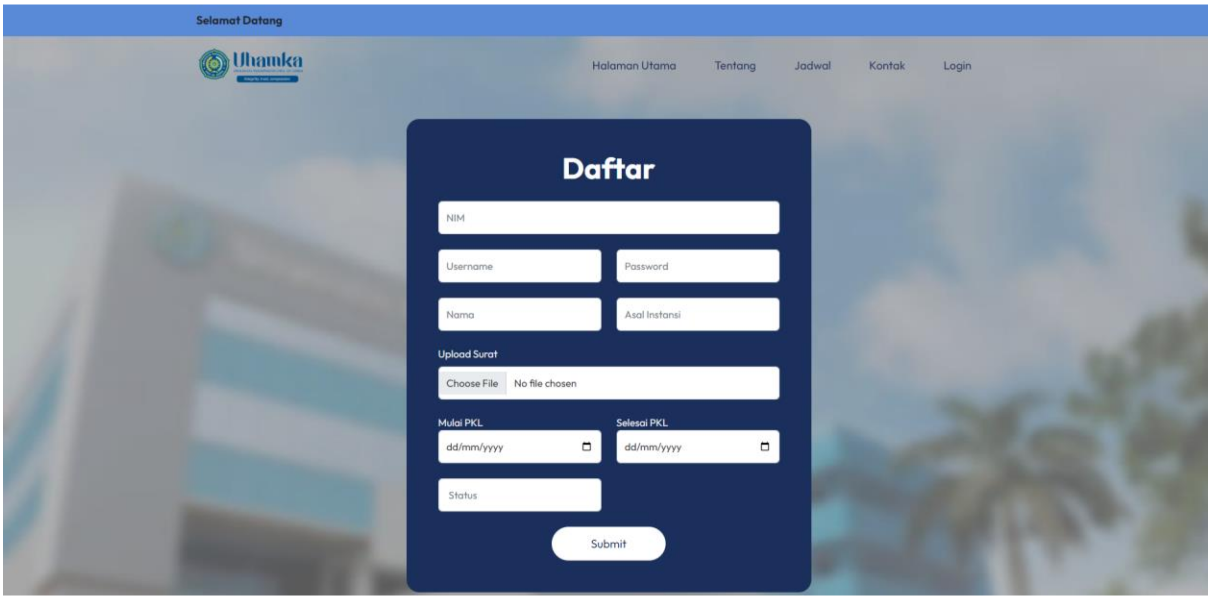
Menumbuhkan Sikap Profesionalisme

Siswa akan dibimbing agar memiliki sikap profesionalisme sehingga di masa yang akan datang siswa dapat terjun ke dalam dunia industri dengan persiapan yang matang.

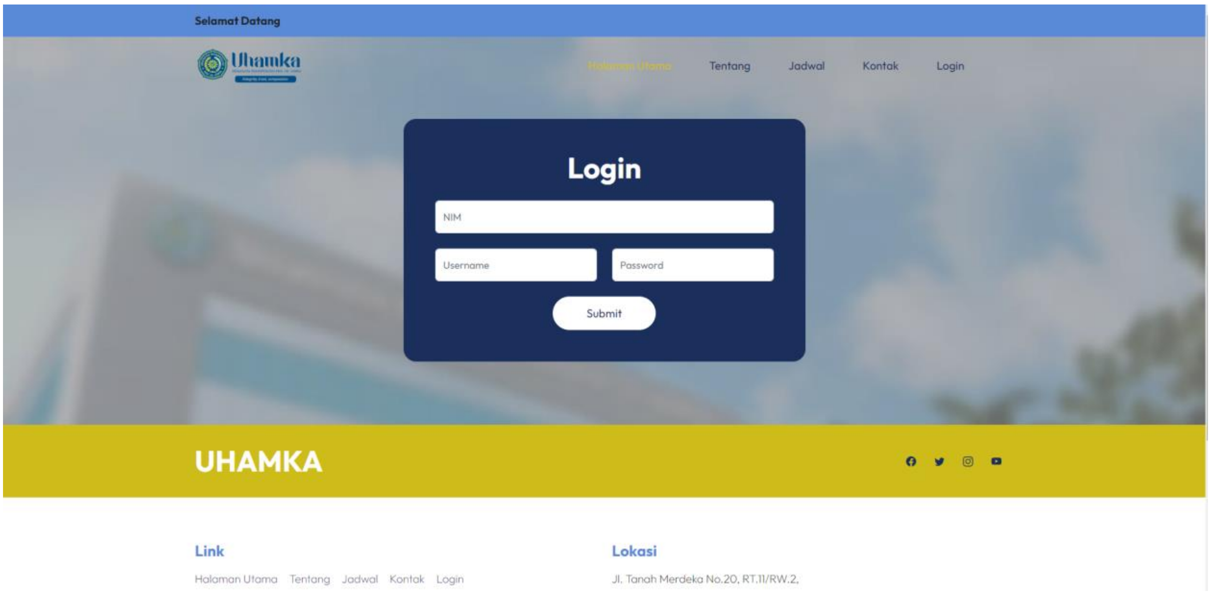
Meningkatkan Daya Kreasi dan Produktivitas

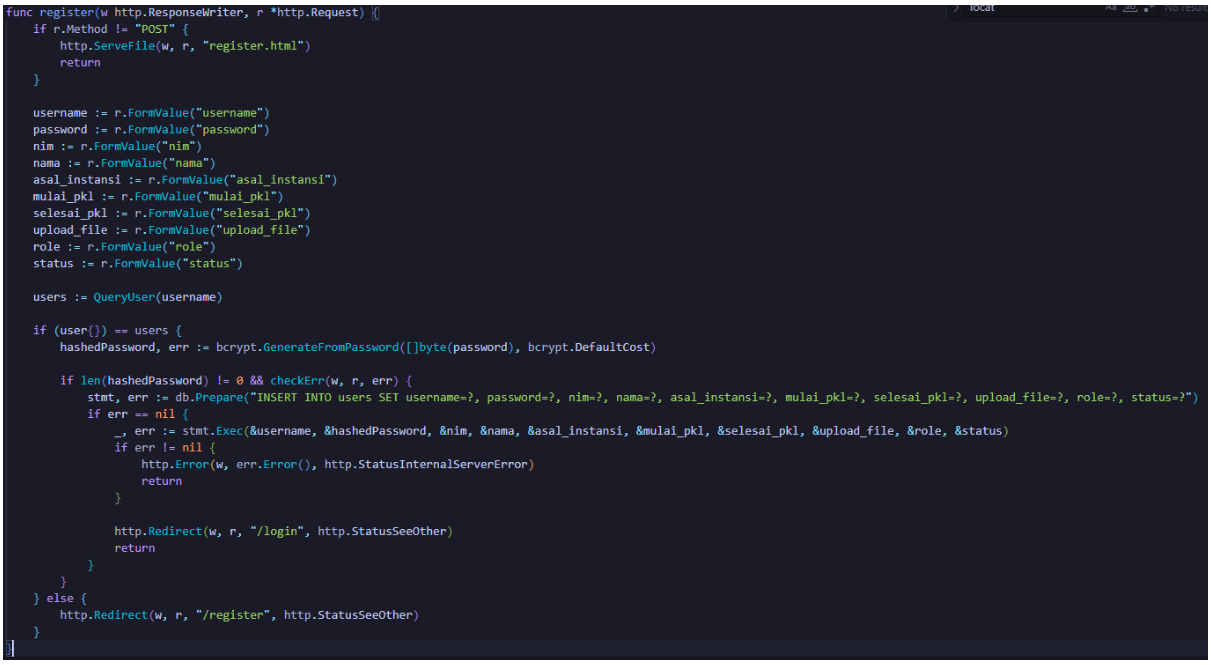
Daya Kreasi dan Produktivitas akan ditingkatkan melalui kerja sama tim, siswa akan diberi kepercayaan untuk menyelesaikan proyek yang membantu UHAMKA mempertahankan kualitasnya sebagai kampus yang beprestasi nan unggul.

Register

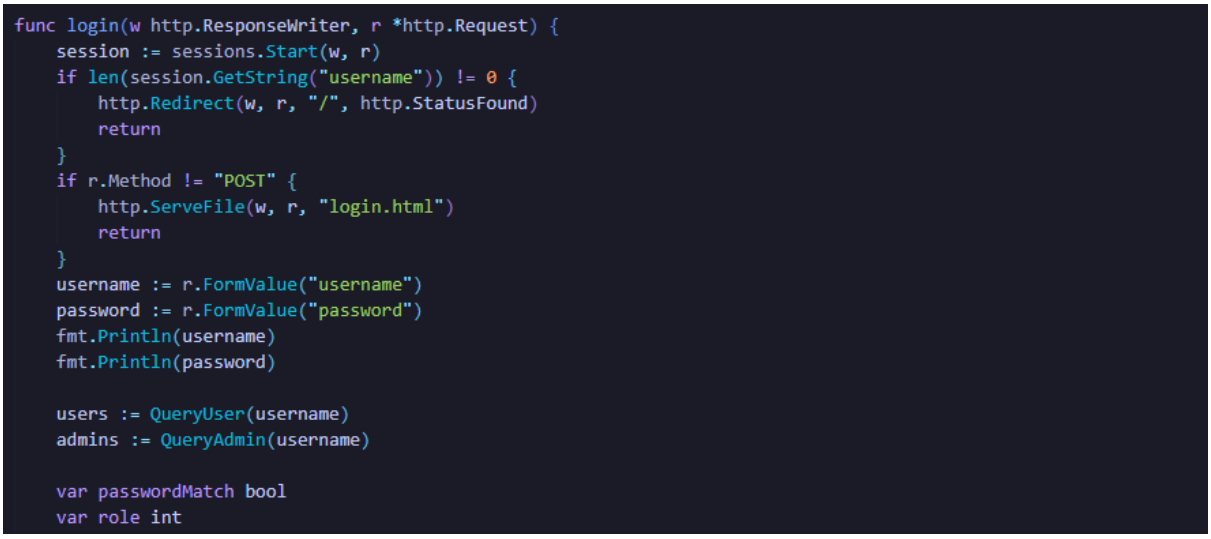


Login



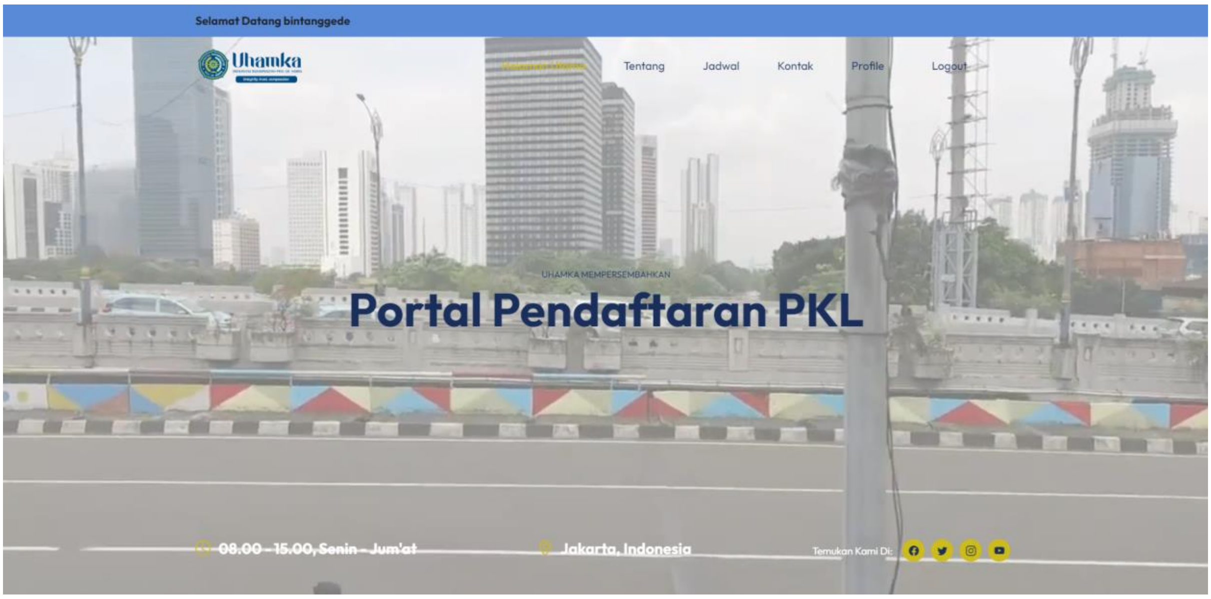


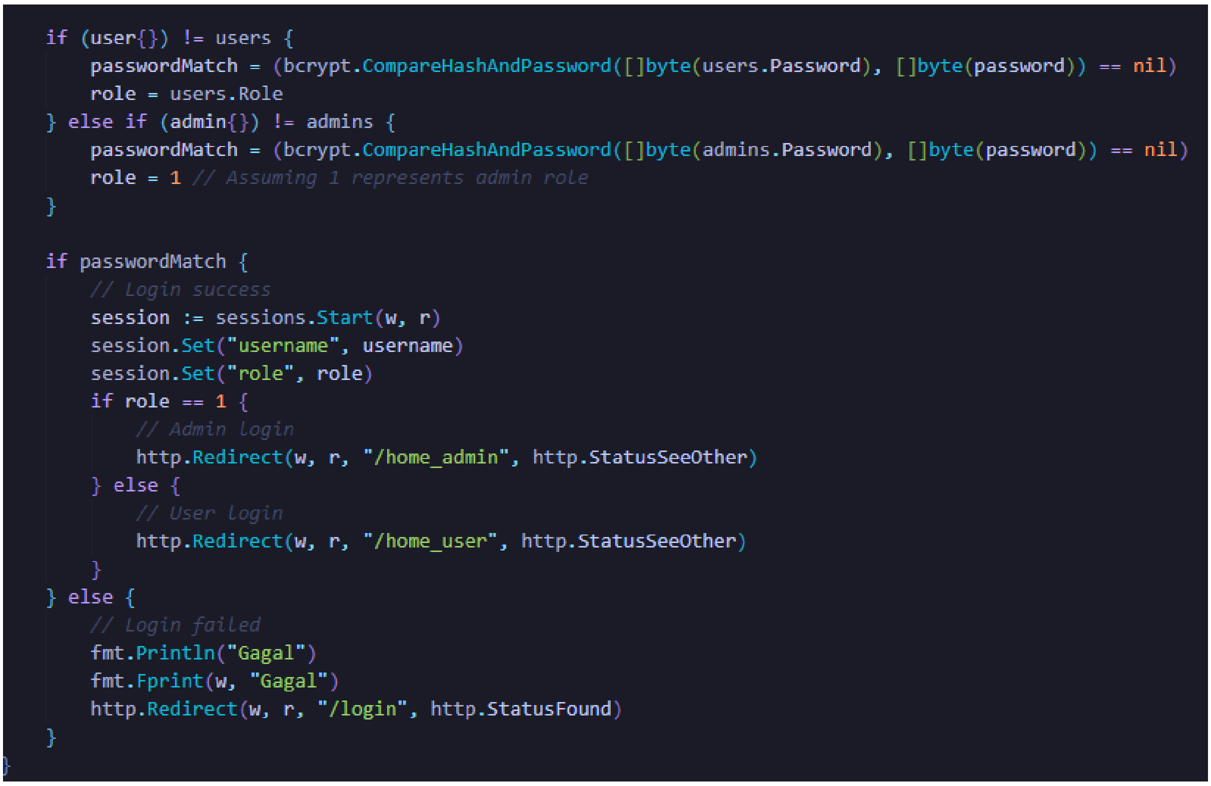
```go
func register(w http.ResponseWriter, r *http.Request) {
    if r.Method != "POST" {
        http.ServeFile(w, r, "register.html")
        return
    }

    username := r.FormValue("username")
    password := r.FormValue("password")
    nim := r.FormValue("nim")
    nama := r.FormValue("nama")
    asal_instansi := r.FormValue("asal_instansi")
    mulai_pkl := r.FormValue("mulai_pkl")
    selesai_pkl := r.FormValue("selesai_pkl")
    upload_file := r.FormValue("upload_file")
    role := r.FormValue("role")
    status := r.FormValue("status")

    users := QueryUser(username)

    if (user{}) == users {
        hashedPassword, err := bcrypt.GenerateFromPassword([]byte(password), bcrypt.DefaultCost)

        if len(hashedPassword) != 0 && checkErr(w, r, err) {
            stmt, err := db.Prepare("INSERT INTO users SET username=?, password=?, nim=?, nama=?, asal_instansi=?, mulai_pkl=?, selesai_pkl=?, upload_file=?, role=?, status=?")
            if err == nil {
                _, err := stmt.Exec(&username, &hashedPassword, &nim, &nama, &asal_instansi, &mulai_pkl, &selesai_pkl, &upload_file, &role, &status)
                if err != nil {
                    http.Error(w, err.Error(), http.StatusInternalServerError)
                    return
                }

                http.Redirect(w, r, "/login", http.StatusSeeOther)
                return
            }
        }
    } else {
        http.Redirect(w, r, "/register", http.StatusSeeOther)
    }
}
```



```go
func login(w http.ResponseWriter, r *http.Request) {
    session := sessions.Start(w, r)
    if len(session.GetString("username")) != 0 {
        http.Redirect(w, r, "/", http.StatusFound)
        return
    }
    if r.Method != "POST" {
        http.ServeFile(w, r, "login.html")
        return
    }
    username := r.FormValue("username")
    password := r.FormValue("password")
    fmt.Println(username)
    fmt.Println(password)

    users := QueryUser(username)
    admins := QueryAdmin(username)

    var passwordMatch bool
    var role int
```

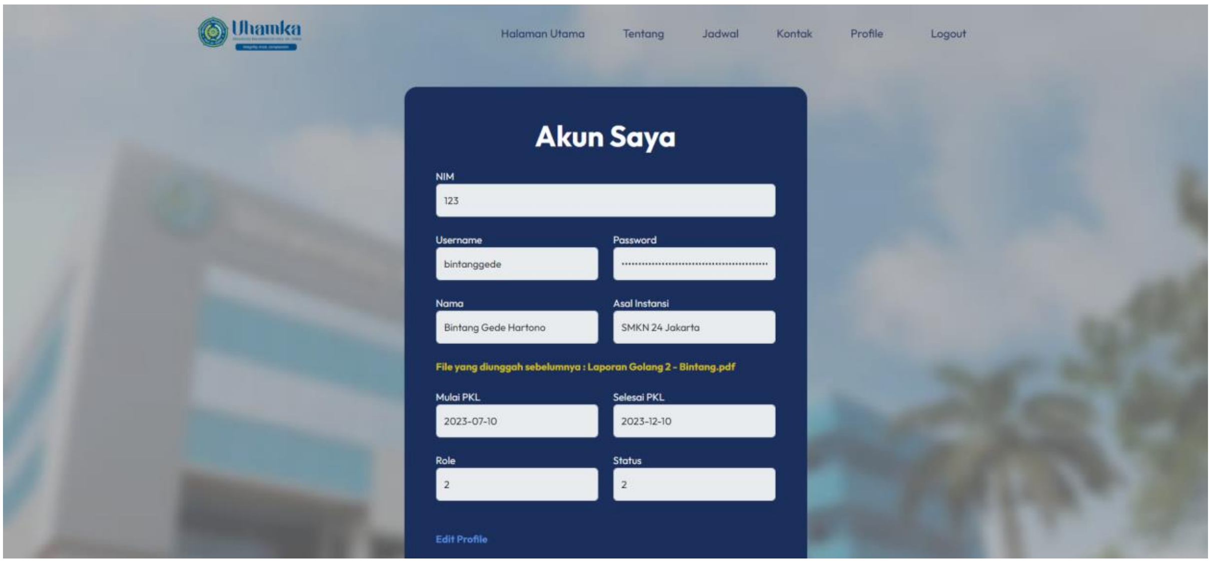
User page





```go
    if (user{}) != users {
        passwordMatch = (bcrypt.CompareHashAndPassword([]byte(users.Password), []byte(password)) == nil)
        role = users.Role
    } else if (admin{}) != admins {
        passwordMatch = (bcrypt.CompareHashAndPassword([]byte(admins.Password), []byte(password)) == nil)
        role = 1 // Assuming 1 represents admin role
    }

    if passwordMatch {
        // Login success
        session := sessions.Start(w, r)
        session.Set("username", username)
        session.Set("role", role)
        if role == 1 {
            // Admin login
            http.Redirect(w, r, "/home_admin", http.StatusSeeOther)
        } else {
            // User login
            http.Redirect(w, r, "/home_user", http.StatusSeeOther)
        }
    } else {
        // Login failed
        fmt.Println("Gagal")
        fmt.Fprint(w, "Gagal")
        http.Redirect(w, r, "/login", http.StatusFound)
    }
}
```

Profil



Akun Saya

NIM: 123

Username: bintanggede

Password: ••

Nama: Bintang Gede Hartono

Asal Instansi: SMKN 24 Jakarta

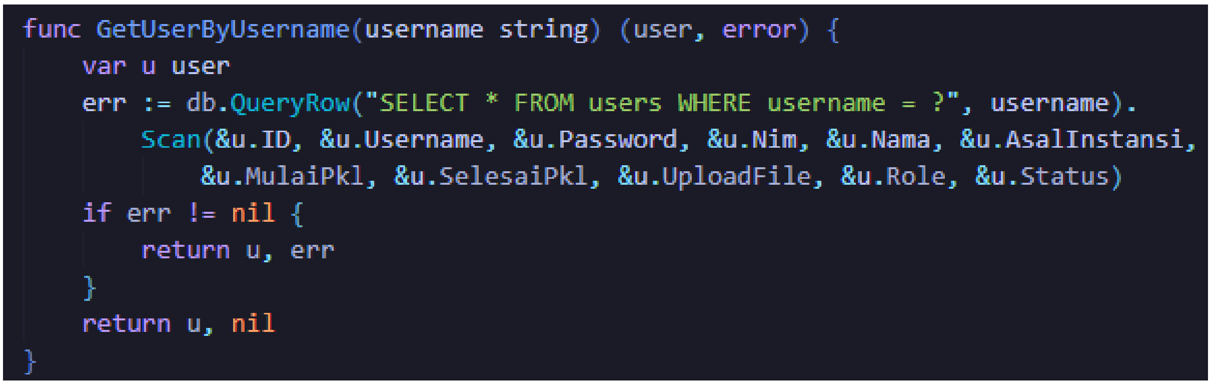
File yang diunggah sebelumnya : Laporan Golang 2 - Bintang.pdf

Mulai PKL: 2023-07-10

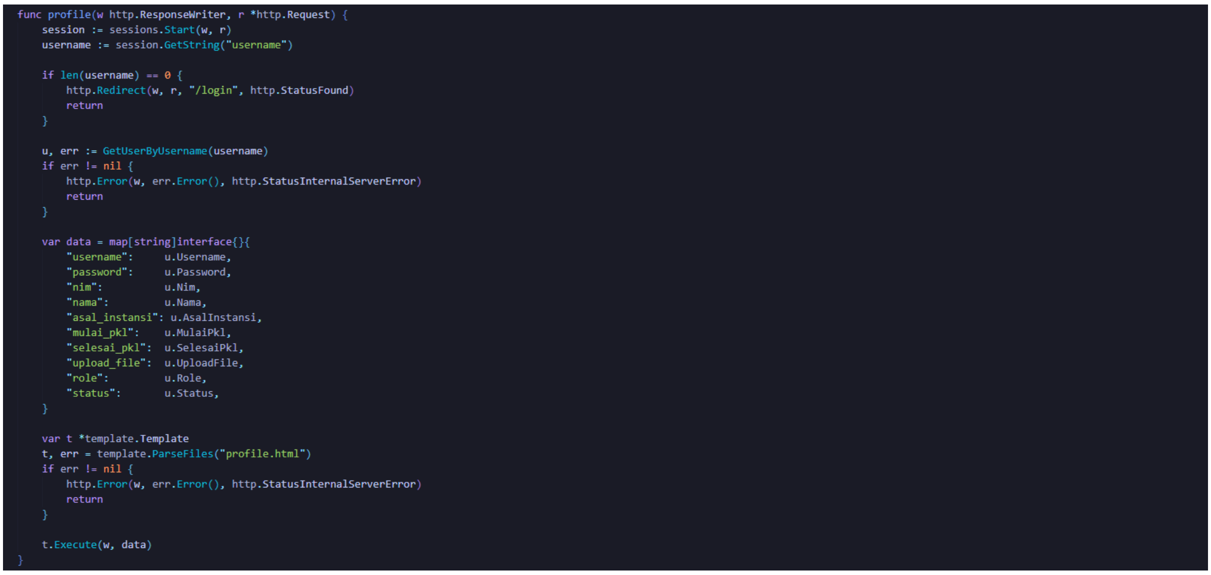
Selesai PKL: 2023-12-10

Role: 2

Status: 2

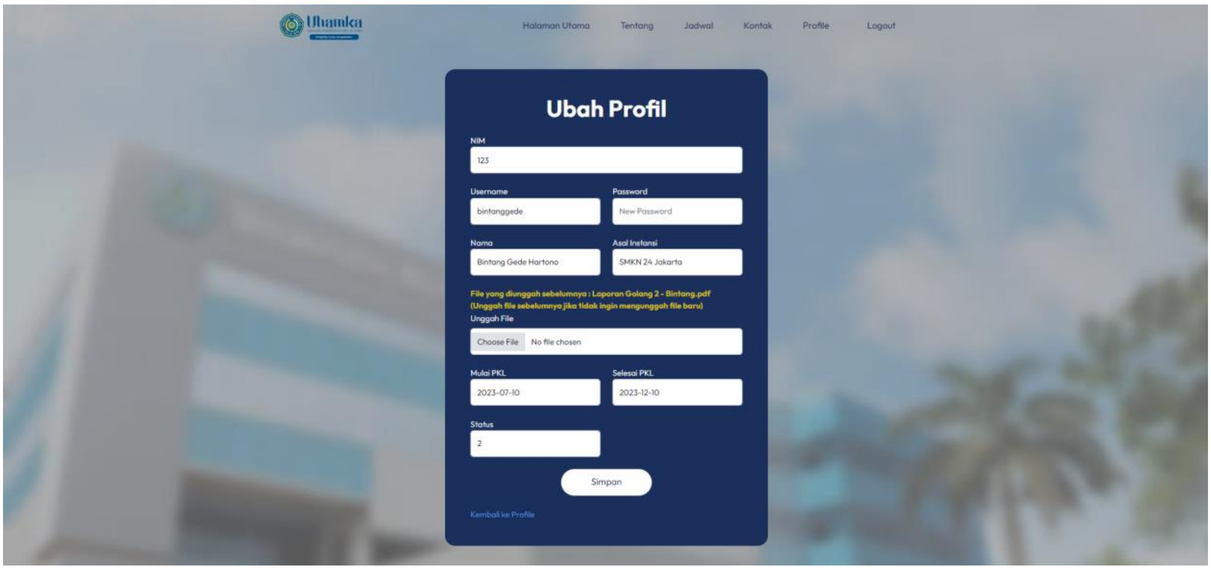


```go
func GetUserByUsername(username string) (user, error) {
    var u user
    err := db.QueryRow("SELECT * FROM users WHERE username = ?", username).
        Scan(&u.ID, &u.Username, &u.Password, &u.Nim, &u.Nama, &u.AsalInstansi,
            &u.MulaiPkl, &u.SelesaiPkl, &u.UploadFile, &u.Role, &u.Status)
    if err != nil {
        return u, err
    }
    return u, nil
}
```



```go
func profile(w http.ResponseWriter, r *http.Request) {
    session := sessions.Start(w, r)
    username := session.GetString("username")

    if len(username) == 0 {
        http.Redirect(w, r, "/login", http.StatusFound)
        return
    }

    u, err := GetUserByUsername(username)
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }

    var data = map[string]interface{}{
        "username":     u.Username,
        "password":     u.Password,
        "nim":          u.Nim,
        "nama":         u.Nama,
        "asal_instansi": u.AsalInstansi,
        "mulai_pkl":    u.MulaiPkl,
        "selesai_pkl":  u.SelesaiPkl,
        "upload_file":  u.UploadFile,
        "role":         u.Role,
        "status":       u.Status,
    }

    var t *template.Template
    t, err = template.ParseFiles("profile.html")
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }

    t.Execute(w, data)
}
```

Edit profil



Ubah Profil

NIM: 123

Username: bintanggede

Password: New Password

Nama: Bintang Gede Hartono

Asal Instansi: SMKN 24 Jakarta

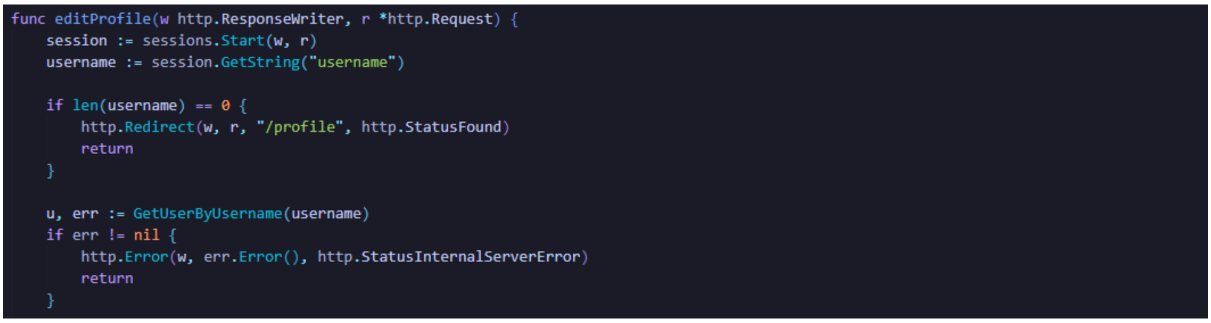
File yang diunggah sebelumnya : Laporan Golang 2 - Bintang.pdf
(Unggah file sebelumnya jika tidak ingin mengunggah file baru)

Unggah File: Choose File No file chosen

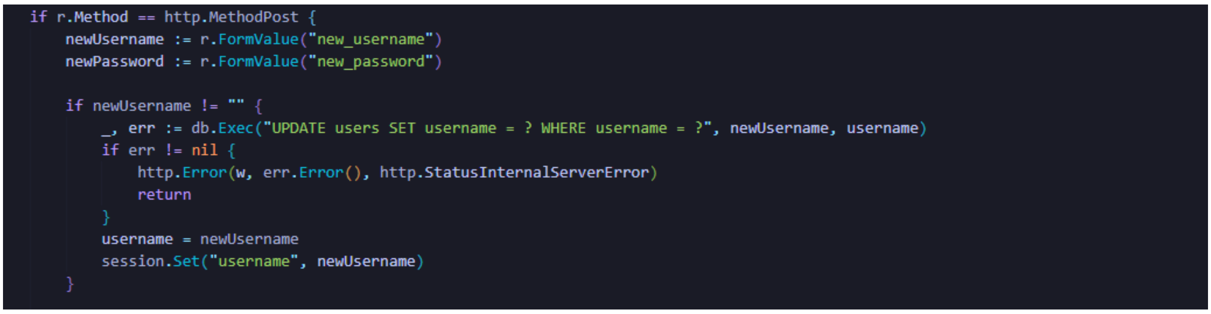
Mulai PKL: 2023-07-10

Selesai PKL: 2023-12-10

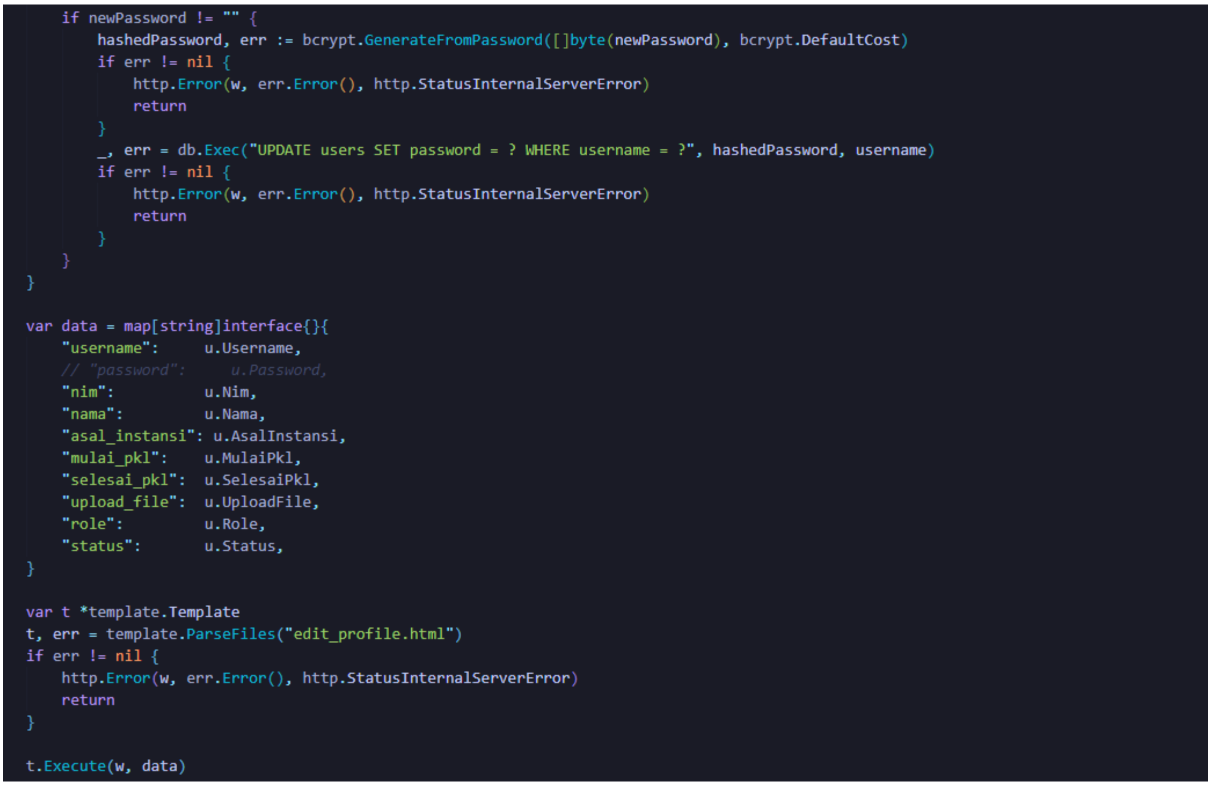
Status: 2



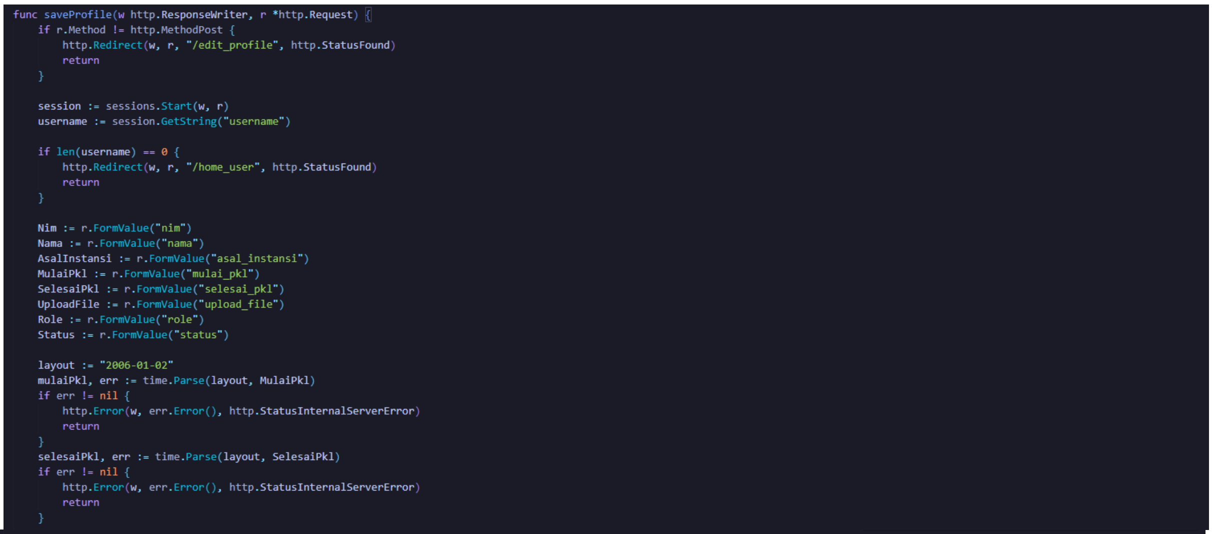
```go
func editProfile(w http.ResponseWriter, r *http.Request) {
    session := sessions.Start(w, r)
    username := session.GetString("username")

    if len(username) == 0 {
        http.Redirect(w, r, "/profile", http.StatusFound)
        return
    }

    u, err := GetUserByUsername(username)
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
```



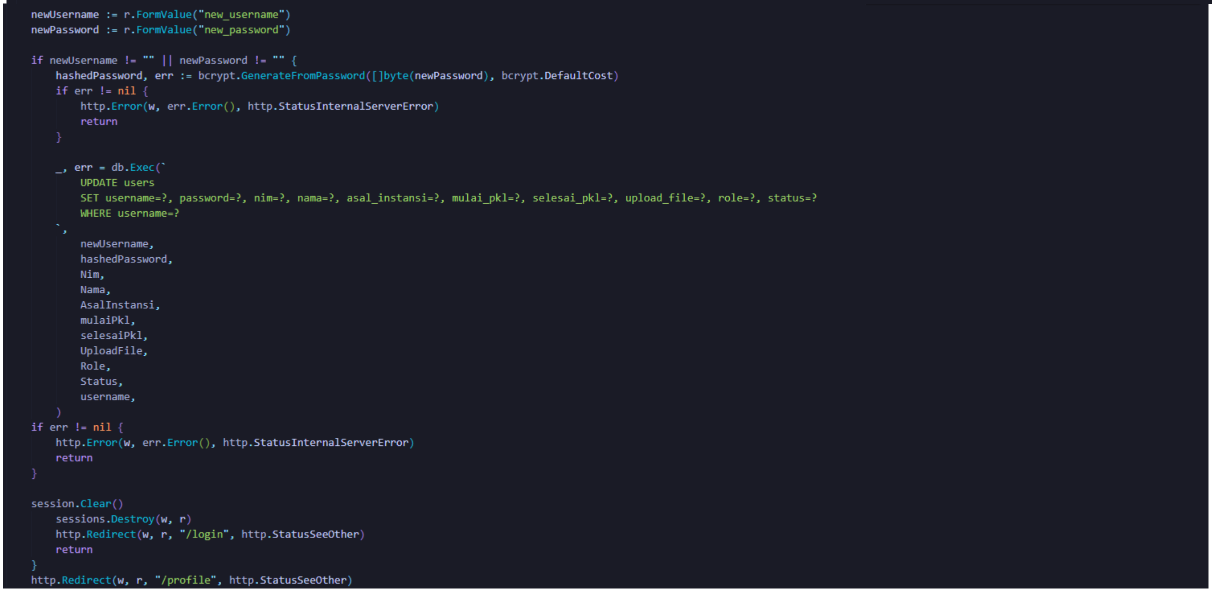
```go
    if r.Method == http.MethodPost {
        newUsername := r.FormValue("new_username")
        newPassword := r.FormValue("new_password")

        if newUsername != "" {
            _, err := db.Exec("UPDATE users SET username = ? WHERE username = ?", newUsername, username)
            if err != nil {
                http.Error(w, err.Error(), http.StatusInternalServerError)
                return
            }
            username = newUsername
            session.Set("username", newUsername)
        }
```



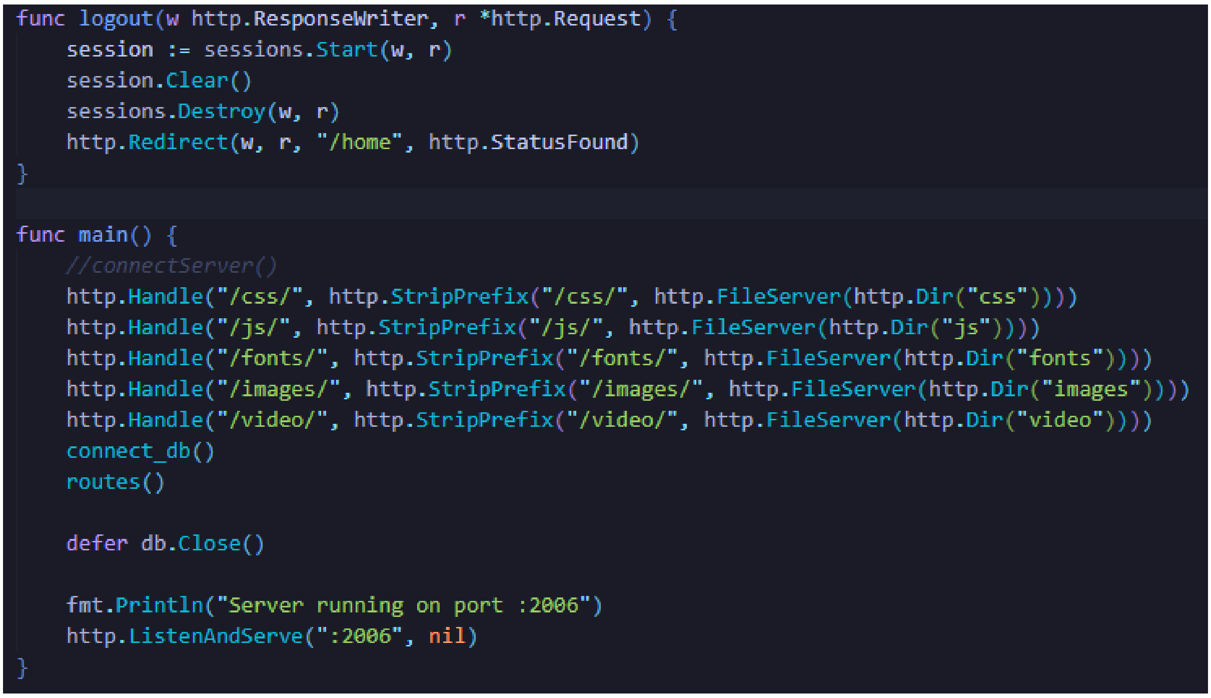
```
        if newPassword != "" {
            hashedPassword, err := bcrypt.GenerateFromPassword([]byte(newPassword), bcrypt.DefaultCost)
            if err != nil {
                http.Error(w, err.Error(), http.StatusInternalServerError)
                return
            }
            _, err = db.Exec("UPDATE users SET password = ? WHERE username = ?", hashedPassword, username)
            if err != nil {
                http.Error(w, err.Error(), http.StatusInternalServerError)
                return
            }
        }
    }

    var data = map[string]interface{}{
        "username":      u.Username,
        // "password":      u.Password,
        "nim":           u.Nim,
        "nama":          u.Nama,
        "asal_instansi": u.AsalInstansi,
        "mulai_pkl":     u.MulaiPkl,
        "selesai_pkl":   u.SelesaiPkl,
        "upload_file":   u.UploadFile,
        "role":          u.Role,
        "status":        u.Status,
    }

    var t *template.Template
    t, err = template.ParseFiles("edit_profile.html")
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }

    t.Execute(w, data)
```



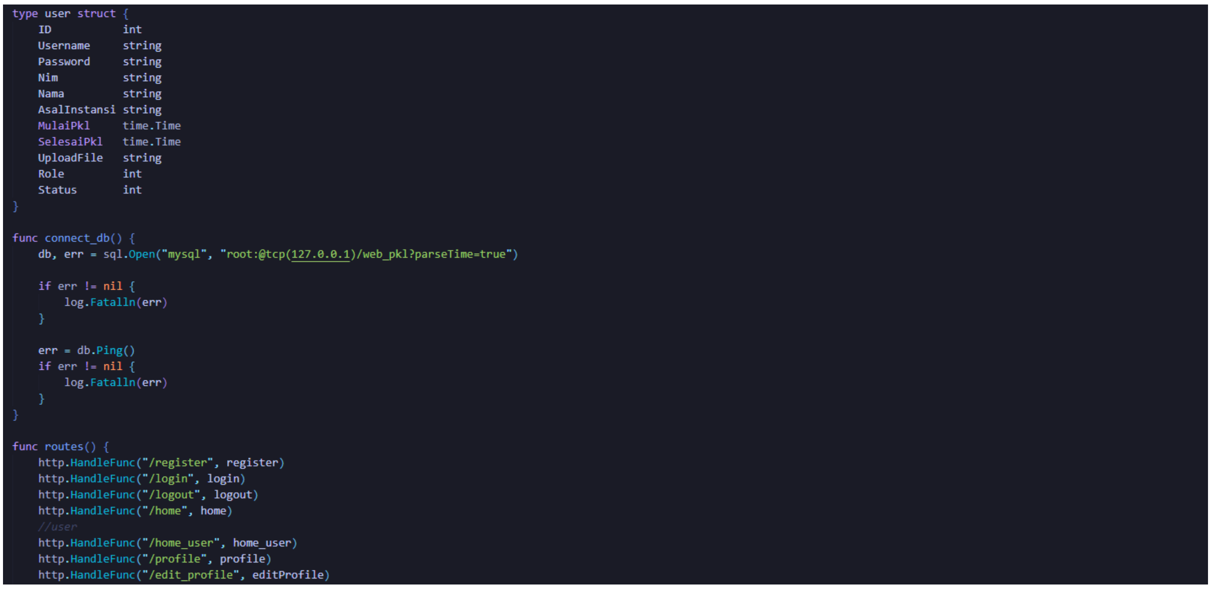
```
func saveProfile(w http.ResponseWriter, r *http.Request) {
    if r.Method != http.MethodPost {
        http.Redirect(w, r, "/edit_profile", http.StatusFound)
        return
    }

    session := sessions.Start(w, r)
    username := session.GetString("username")

    if len(username) == 0 {
        http.Redirect(w, r, "/home_user", http.StatusFound)
        return
    }

    Nim := r.FormValue("nim")
    Nama := r.FormValue("nama")
    AsalInstansi := r.FormValue("asal_instansi")
    MulaiPkl := r.FormValue("mulai_pkl")
    SelesaiPkl := r.FormValue("selesai_pkl")
    UploadFile := r.FormValue("upload_file")
    Role := r.FormValue("role")
    Status := r.FormValue("status")

    layout := "2006-01-02"
    mulaiPkl, err := time.Parse(layout, MulaiPkl)
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    selesaiPkl, err := time.Parse(layout, SelesaiPkl)
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
```



```
    newUsername := r.FormValue("new_username")
    newPassword := r.FormValue("new_password")

    if newUsername != "" || newPassword != "" {
        hashedPassword, err := bcrypt.GenerateFromPassword([]byte(newPassword), bcrypt.DefaultCost)
        if err != nil {
            http.Error(w, err.Error(), http.StatusInternalServerError)
            return
        }

        _, err = db.Exec(`
            UPDATE users
            SET username=?, password=?, nim=?, nama=?, asal_instansi=?, mulai_pkl=?, selesai_pkl=?, upload_file=?, role=?, status=?
            WHERE username=?
        `,
            newUsername,
            hashedPassword,
            Nim,
            Nama,
            AsalInstansi,
            mulaiPkl,
            selesaiPkl,
            UploadFile,
            Role,
            Status,
            username,
        )
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }

    session.Clear()
        sessions.Destroy(w, r)
        http.Redirect(w, r, "/login", http.StatusSeeOther)
        return
    }
    http.Redirect(w, r, "/profile", http.StatusSeeOther)
```



```go
func logout(w http.ResponseWriter, r *http.Request) {
    session := sessions.Start(w, r)
    session.Clear()
    sessions.Destroy(w, r)
    http.Redirect(w, r, "/home", http.StatusFound)
}

func main() {
    //connectServer()
    http.Handle("/css/", http.StripPrefix("/css/", http.FileServer(http.Dir("css"))))
    http.Handle("/js/", http.StripPrefix("/js/", http.FileServer(http.Dir("js"))))
    http.Handle("/fonts/", http.StripPrefix("/fonts/", http.FileServer(http.Dir("fonts"))))
    http.Handle("/images/", http.StripPrefix("/images/", http.FileServer(http.Dir("images"))))
    http.Handle("/video/", http.StripPrefix("/video/", http.FileServer(http.Dir("video"))))
    connect_db()
    routes()

    defer db.Close()

    fmt.Println("Server running on port :2006")
    http.ListenAndServe(":2006", nil)
}
```



```go
type user struct {
    ID           int
    Username     string
    Password     string
    Nim          string
    Nama         string
    AsalInstansi string
    MulaiPkl     time.Time
    SelesaiPkl   time.Time
    UploadFile   string
    Role         int
    Status       int
}

func connect_db() {
    db, err = sql.Open("mysql", "root:@tcp(127.0.0.1)/web_pkl?parseTime=true")

    if err != nil {
        log.Fatalln(err)
    }

    err = db.Ping()
    if err != nil {
        log.Fatalln(err)
    }
}

func routes() {
    http.HandleFunc("/register", register)
    http.HandleFunc("/login", login)
    http.HandleFunc("/logout", logout)
    http.HandleFunc("/home", home)
    //user
    http.HandleFunc("/home_user", home_user)
    http.HandleFunc("/profile", profile)
    http.HandleFunc("/edit_profile", editProfile)
```