APLIKASI PEMILU LEGISLATIF

GO-PILU

TUGAS BESAR ALGORITMA PEMROGRAMAN

Oleh :

Gede Bagus Krishnanditya Merta - 1301223088 / IF-46-04

Raka Aditya Waluya - 1301220192 / IF-46-04

PROGRAM STUDI S-1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY

2022/2023

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pemilihan umum merupakan salah satu momen penting dalam suatu sistem demokrasi suatu negara. Namun, proses pemilihan umum seringkali dihadapkan pada kendala seperti panjangnya waktu penghitungan suara dan potensi kecurangan. Oleh karena itu, kami memperkenalkan sebuah solusi inovatif berupa Go-Pilu yang bertujuan untuk meningkatkan efisiensi dan transparansi dalam pelaksanaan pemilihan umum.

Go-Pilu yang kami buat memiliki fokus utama pada pemilihan umum legislatif. Aplikasi ini dirancang untuk memberikan kemudahan dalam proses pendaftaran calon legislatif, pendaftaran pemilih, pemungutan suara, dan penghitungan suara.

Dengan adanya Go-Pilu ini, diharapkan partisipasi pemilih akan meningkat, proses pemilihan umum menjadi lebih cepat dan transparan, serta potensi kecurangan dapat diminimalisir.

1.2 Batasan Masalah

Dalam pengembangan Go-Pilu ini, terdapat beberapa batasan masalah yang perlu diperhatikan. Pertama, Go-Pilu ini masih dalam bentuk konsep sederhana dan belum terimplementasi secara menyeluruh, Penggunaan aplikasi ini difokuskan akan pada pemilihan calon legislatif dan partai calon di 3 provinsi tertentu. Hal ini berarti bahwa pengguna aplikasi ini terbatas pada pemilih yang berada di provinsi-provinsi yang ditentukan. Selain itu, pemilih hanya dapat memilih calon yang berasal dari provinsi yang sama dengan mereka, sehingga aplikasi ini membatasi pemilihan berdasarkan provinsi tempat tinggal pemilih.

Selanjutnya, batasan lainnya adalah pengaturan tanggal pemilu dalam aplikasi. Pemilih hanya dapat menggunakan aplikasi ini pada tanggal pemilihan yang telah ditentukan. Di luar tanggal tersebut, pengguna aplikasi hanya dapat melihat daftar calon tanpa dapat melakukan pemilihan.

BAB II
ISI

2.1 Alur Bisnis Aplikasi

Go-Pilu yang kami kembangkan memiliki alur bisnis yang terdiri dari beberapa tahapan yang penting dalam proses pemilihan umum legislatif. Berikut adalah alur bisnis Go-Pilu yang kami buat:

a. Penentuan Tanggal Pemilu oleh KPU:

   Tahapan pertama dalam alur bisnis Go-Pilu ini adalah penentuan tanggal dimulainya dan berakhirnya suatu pemilu oleh Komisi Pemilihan Umum (KPU). KPU bertanggung jawab untuk mengatur jadwal pelaksanaan pemilu sesuai dengan ketentuan yang berlaku.

b. Manajemen Data Calon oleh KPU:

   KPU memiliki akses untuk mengelola data calon yang akan dipilih oleh pemilih melalui aplikasi ini. KPU dapat menambahkan, mengedit, dan menghapus data calon yang terdaftar dalam aplikasi.

c. Waktu Pemilihan:

   Pemilihan hanya dapat dilakukan oleh pemilih dalam masa atau waktu pemilihan yang telah ditentukan oleh KPU.

d. Seleksi Calon berdasarkan Provinsi:

   Pemilih dalam aplikasi ini hanya dapat memilih calon yang memiliki kesamaan provinsi dengan pemilih. Fitur ini memastikan bahwa pemilih hanya memiliki akses ke daftar calon yang relevan dengan wilayah tempat tinggal mereka.

e. Tampilan Hasil Pemilu:

Setelah waktu pemilihan berakhir, hasil pemilu akan ditampilkan melalui aplikasi. Pemilih dapat melihat hasil pemilu secara real-time, termasuk perolehan suara untuk masing-masing calon dan partai calon yang terdaftar dalam aplikasi.

2.2 Rancangan Aplikasi

Dalam merancang Go-Pilu, kami melakukan pembuatan *Flowchart* sebagai langkah awal dalam proses pembuatan aplikasi ini. Flowchart digunakan untuk merancang dan menggambarkan alur bisnis dan logika dari Go-Pilu yang kami kembangkan. Dengan menggunakan flowchart, kami dapat memvisualisasikan langkah-langkah utama dalam proses pemilihan calon legislatif, mulai dari penentuan tanggal pemilu oleh KPU, manajemen data calon, waktu pemilihan, seleksi calon berdasarkan provinsi, hingga tampilan hasil pemilu. Flowchart membantu kami dalam merancang struktur aplikasi, mengidentifikasi input dan output yang diperlukan, serta memahami keterkaitan antara setiap langkah dalam proses tersebut.

Gambar 1 flowchart

Setelah pembuatan *flowchart*, langkah selanjutnya adalah implementasi dengan pembuatan program menggunakan bahasa pemrograman golang. Berikut adalah beberapa potongan kode dari program yang telah kami buat :

1. Konstanta

```
const NMAX = 1000
```

Gambar 2.1 konstanta

Const NMAX adalah konstanta yang kami gunakan untuk membuat isi array sebanyak NMAX.

2. Struktur Data

```
type date struct {
    dd1, mm1, yy1, dd2, mm2, yy2 int
}

type tabDate [NMAX]date

type provinsi struct {
    calon            calon
    pemilih          pemilih
    pilih            int
    totalSuaraCalon  int
    totalSuaraPartai int
    nPA, nPB, nPC    int
}
```

```
type calon struct {
    nama, partai string
}

type pemilih struct {
    nama    string
    pilihan int
}

type arrProvA [NMAX]provinsi
type arrProvB [NMAX]provinsi
type arrProvC [NMAX]provinsi
```

Gambar 2.2 data struct

Potongan kode di atas merupakan data struct yang kami gunakan untuk menampung variabel secara global . *Data struc*t tersebut nantinya akan kami gunakan dalam pembuatan function dalam program.

3. Fungsi Utama

```
func main() {
    var A date
    var TA arrProvA
    var TB arrProvB
    var TC arrProvC
    var nA, nB, nC int
    var nPA, nPB, nPC int

    menuUtama(&A, &TA, &TB, &TC, &nA, &nB, &nC, &nPA, &nPB, &nPC)

}
```

Gambar 2.3 func main

Fungsi utama pada program kami hanya memuat variabel dan pemanggilan fungsi “menuUtama” yang digunakan untuk melanjutkan ke dalam menu pilihan lainnya.

4. Fungsi Menu

```go
func menuUtama(A *date, TA *arrProvA, TB *arrProvB, TC *arrProvC, nA, nB, nC, nPA, nPB, nPC *int) {

    var pilihan int
    fmt.Println()
    fmt.Println("======================================")
    fmt.Println("  *Selamat Datang di Aplikasi PEMILU*  ")
    fmt.Println("======================================")
    fmt.Println()
    fmt.Println("Masuk sebagai : ")
    fmt.Println("1. KPU")
    fmt.Println("2. Pemilih")
    fmt.Println("3. Selesai")

    fmt.Println()
    fmt.Print("Pilihan anda : ")
    fmt.Scan(&pilihan)
    if pilihan == 1 {
        menuKPU(A, TA, TB, TC, nA, nB, nC, nPA, nPB, nPC)
    } else if pilihan == 2 {
        menuPemilih(A, TA, TB, TC, nA, nB, nC, nPA, nPB, nPC)
    } else if pilihan == 3 {
        hasilPEMILU(*TA, *TB, *TC, *nA, *nB, *nC, *nPA, *nPB, *nPC)
        fmt.Println()
        fmt.Println("----------------------------------")
        fmt.Println("Terima Kasih")
    } else {
        fmt.Println("Pilihan salah")
        menuUtama(A, TA, TB, TC, nA, nB, nC, nPA, nPB, nPC)
    }

}
```

Gambar 2.4 func menu utama

Fungsi ini bertujuan untuk menampilkan menu utama yang berisi tampilan awal saat masuk ke Go-Pilu. Pada menu utama terdapat 3 pilihan utama yaitu Masuk sebagai KPU, Pemilih, dan Selesai.

Pada pilihan “KPU”, pengguna akan masuk sebagai KPU. Disini pengguna sebagai KPU dapat mengatur berbagai macam hal yang berkaitan dengan proses pemilu, mulai dari mengatur tanggal hingga penambahan calon.

Pada piliham “Pemilih”, pengguna akan masuk sebagai Pemilih. Disini pengguna sebagai Pemilih dapat memilih calon yang telah didaftarkan KPU.

Terakhir adalah pilihan “Selesai”. Jika pengguna memilih pilihan tersebut, maka hasil pemilu akan ditampilkan dan program akan berhenti berjalan.

5. Fungsi untuk Proses Data

Berikut adalah beberapa potongan kode fungsi yang berfungsi untuk mengolah dan memproses data yang data. Fungsi tersebut contohnya dalah “func isiCalon”, “func isiPemilih”, “func editCalon”, “func hapusCalon”, dan sebagainya.

```
func isiCalon(TA *arrProvA, TB *arrProvB, TC *arrProvC, nA, nB, nC *int) {
    var namaA, partaiA, namaB, partaiB, namaC, partaiC string

    fmt.Println("Masukkan data calon Provinsi A: ")
    fmt.Scan(&namaA)
    for namaA != "STOP" {
        fmt.Scan(&partaiA)
        TA[*nA].calon.nama = namaA
        TA[*nA].calon.partai = partaiA
        *nA++
        fmt.Scan(&namaA)
    }
}
```

Gambar 2.5 func isi calon

```
func editCalonA(TA *arrProvA, noUrut, nA int) {
    var nama, partai, konfirmasi string
    fmt.Println("Masukkan data yang di edit: ")
    fmt.Scan(&nama, &partai)
    TA[noUrut-1].calon.nama = nama
    TA[noUrut-1].calon.partai = partai
    fmt.Println("======== Data yang di edit =========")
    fmt.Println(TA[noUrut-1].calon.nama, TA[noUrut-1].calon.partai)
    fmt.Print("Konfirmasi? (y/n) ")
    fmt.Scan(&konfirmasi)
    if konfirmasi == "y" {
        fmt.Println("Data berhasil di edit!")
        fmt.Println("Data Calon Provinsi A : ")
        cetakCalonA(*TA, nA)
    } else {
        editCalonA(TA, noUrut, nA)
    }
}
```

```
func hapusCalonA(TA *arrProvA, noUrut int, nA *int) {
    var konfirmasi string
    fmt.Println("Apakah anda yakin akan menghapus calon nomor", noUrut, TA[noUrut-1].calon.nama, TA[noUrut-1].calon.partai, " ?")
    fmt.Print("Konfirmasi? (y/n)")
    fmt.Scan(&konfirmasi)
    if konfirmasi == "y" {
        for i := noUrut - 1; i < *nA-(noUrut-1); i++ {
            TA[i].calon = TA[i+1].calon
        }
        *nA = *nA - 1
```

Gambar 2.6 func edit calon

```
func cetakCalonA(TA arrProvA, nA int) {
    fmt.Println("--------------")
    fmt.Println("Provinsi A : ")
    fmt.Println("--------------")
    for i := 0; i < nA; i++ {
        fmt.Printf("%d. %s %s \n", i+1, TA[i].calon.nama, TA[i].calon.partai)
    }
    fmt.Println()
}
```

Gambar 2.8 func cetak calon

```
func searchNamaCalonA(TA arrProvA, nA int) {
    var nama string
    var ketemu bool
    fmt.Print("Masukkan nama yang dicari: ")
    fmt.Scan(&nama)
    ketemu = false
    for i := 0; i < nA; i++ {
        if TA[i].calon.nama == nama {
            fmt.Println(i)
            fmt.Println("=========================================")
            fmt.Println("              ~Data ditemukan!~          ")
            fmt.Println("-----------------------------------------")
            fmt.Printf("%d. %s %s\n", i+1, TA[i].calon.nama, TA[i].calon.partai)
            fmt.Println("=========================================")
            ketemu = true
            i = nA
        }
    }
    if ketemu == false {
        fmt.Println("=========================================")
        fmt.Println("            ~Data tidak ditemukan!~      ")
        fmt.Println("=========================================")
    }

}
```

Gambar 2.9 func searching nama calon

```
func searchPartaiCalonA(TA arrProvA, nA int) {
    var partai string
    var ketemu bool
    fmt.Print("Masukkan partai yang dicari: ")
    fmt.Scan(&partai)
    ketemu = false
    for i := 0; i < nA; i++ {
        if TA[i].calon.partai == partai {
            fmt.Printf("%d. %s %s\n", i+1, TA[i].calon.nama, TA[i].calon.partai)
            ketemu = true
        }
    }
    if ketemu == true {
        fmt.Println("=========================================")
        fmt.Println("              ~Data ditemukan!~          ")
```

```
func cetakPemilihA(TA arrProvA, nPA int) {
    var calon int
    fmt.Println("Provinsi A : ")
    if nPA > 0 {
        for i := 0; i < nPA; i++ {
            calon = TA[i].pemilih.pilihan
            fmt.Printf("%d. Nama: %s \n", i+1, TA[i].pemilih.nama)
            fmt.Printf("   Pilihan: %d. %s - %s \n", TA[i].pemilih.pilihan, TA[calon-1].calon.nama, TA[calon-1].calon.partai)
        }
        fmt.Println("")
    }
}
```

Gambar 2.11 func cetak pemilih

```
func searchPemilihA(TA arrProvA, nPA int) {
    var calon int
    var nama string
    var ketemu bool
    fmt.Print("Masukkan nama yang dicari: ")
    fmt.Scan(&nama)
    ketemu = false
    for i := 0; i < nPA; i++ {
        if TA[i].pemilih.nama == nama {
            fmt.Println(i)
            calon = TA[i].pemilih.pilihan
            fmt.Println("=========================================")
            fmt.Println("              ~Data ditemukan!~          ")
            fmt.Println("-----------------------------------------")
            fmt.Printf("%d. Nama: %s \n", i+1, TA[i].pemilih.nama)
            fmt.Printf("   Pilihan: %d. %s - %s \n", TA[i].pemilih.pilihan, TA[calon-1].calon.nama, TA[calon-1].calon.partai)
            fmt.Println("=========================================")
            ketemu = true
            i = nPA
        }
    }
    if ketemu == false {
        fmt.Println("=========================================")
        fmt.Println("           ~Data tidak ditemukan!~       ")
        fmt.Println("=========================================")
    }
}
```

Gambar 2.12 func search pemilih

```
func hitungSemuaCALON(TA arrProvA, TB arrProvB, TC arrProvC, nA, nB, nC, nPA, nPB, nPC int) {

    //hitungA
    for i := 0; i < nA; i++ {
        for j := 0; j < nPA; j++ {
            if TA[j].pemilih.pilihan == i+1 {
                TA[i].totalSuaraCalon++
            }
        }
    }

    //hitungB
    for i := 0; i < nB; i++ {
        for j := 0; j < nPB; j++ {
            if TB[j].pemilih.pilihan == i+1 {
                TB[i].totalSuaraCalon++
            }
        }
    }

    //hitungC
    for i := 0; i < nC; i++ {
        for j := 0; j < nPC; j++ {
```

Gambar 2.10 func search calon

```
    //hitungPartai A

    fmt.Println("----------------------------------")
    fmt.Println("Total suara PARTAI di Provinsi A: ")

    for i := 0; i < nA; i++ {
        for j := i + 1; j < nA; j++ {
            if TA[i].calon.partai == TA[j].calon.partai {
                TA[i].totalSuaraPartai = TA[i].totalSuaraCalon + TA[j].totalSuaraCalon
                //hapus data indeks j
                for k := j; k < nA; k++ {
                    TA[k] = TA[k+1]
                }
            } else {
                TB[i].totalSuaraPartai = TB[i].totalSuaraCalon
            }
        }
        if TA[i].totalSuaraPartai > 0 {
            fmt.Println(TA[i].calon.partai, " - ", TA[i].totalSuaraPartai, "suara")
        }
    }
```

Gambar 2.14 menghitung total partai

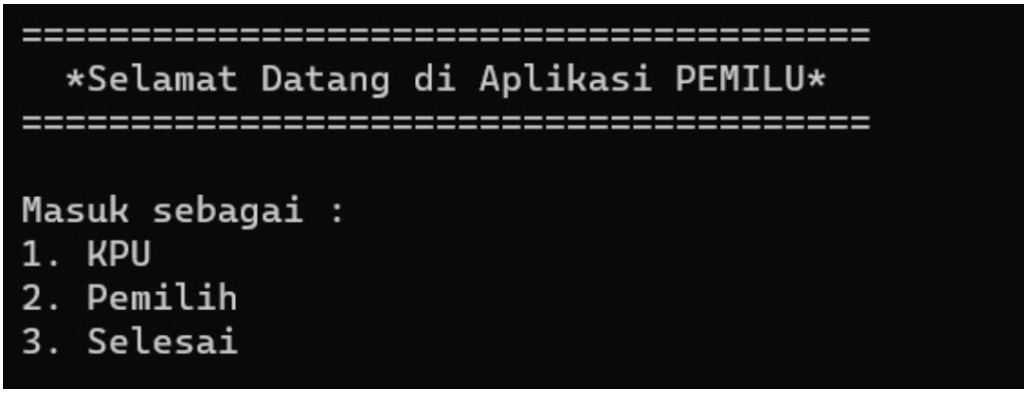
```
func sortingTOTAL(TA arrProvA, TB arrProvB, TC arrProvC, nPA, nPB, nPC int) {
    var tempA, tempB, tempC provinsi

    for i := 0; i < nPA; i++ {
        for j := i + 1; j < nPA; j++ {
            if TA[i].totalSuaraCalon < TA[j].totalSuaraCalon {
                tempA = TA[i]
                TA[i] = TA[j]
                TA[j] = tempA
            }
        }
    }

    for i := 0; i < nPB; i++ {
        for j := i + 1; j < nPB; j++ {
            if TB[i].totalSuaraCalon < TB[j].totalSuaraCalon {
                tempB = TB[i]
                TB[i] = TB[j]
                TB[j] = tempB
            }
        }
    }

    for i := 0; i < nPC; i++ {
        for j := i + 1; j < nPC; j++ {
            if TC[i].totalSuaraCalon < TC[j].totalSuaraCalon {
                tempC = TC[i]
                TC[i] = TC[j]
                TC[j] = tempC
            }
        }
    }
}
```

Gambar 2.13 func hitung calon

Gambar 2.15 func sorting total

Setelah berhasil melakukan pembuatan kode program, Go-Pilu dapat dijalankan. Aplikasi ini dapat dijalankan pada terminal dari komputer. Berikut adalah potongan tampilan dari Go-Pilu :

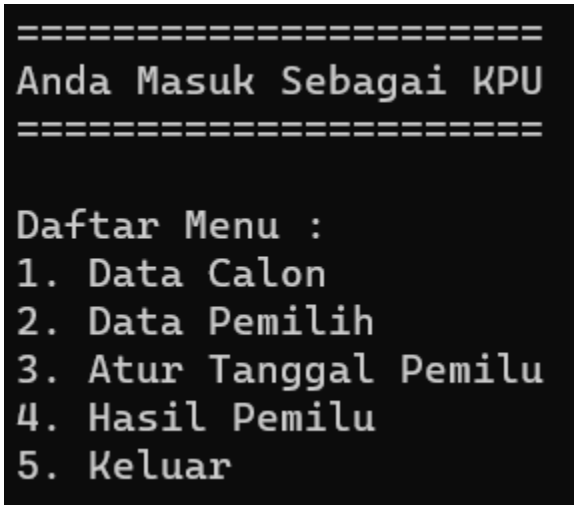
1. Menu Utama



Gambar 3.1 tampilan awal Go-Pilu

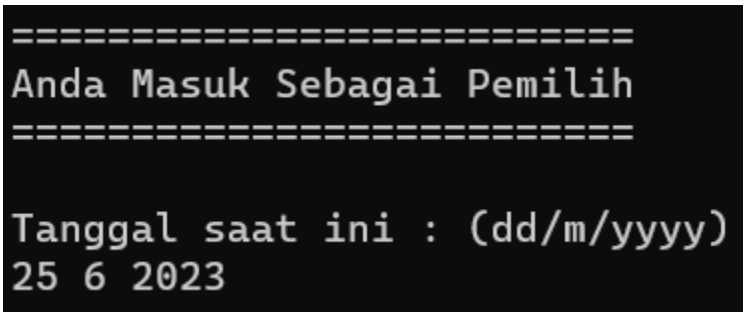
Tampilan menu utama pada Go-Pilu difokuskan kepada 3 poin, yaitu “KPU” dan “Pemilih” sebagai pilihan pengguna dan “Selesai” untuk menampilkan hasil pemilu.

2. Menu KPU



Gambar 3.2 tampilan menu KPU

3.



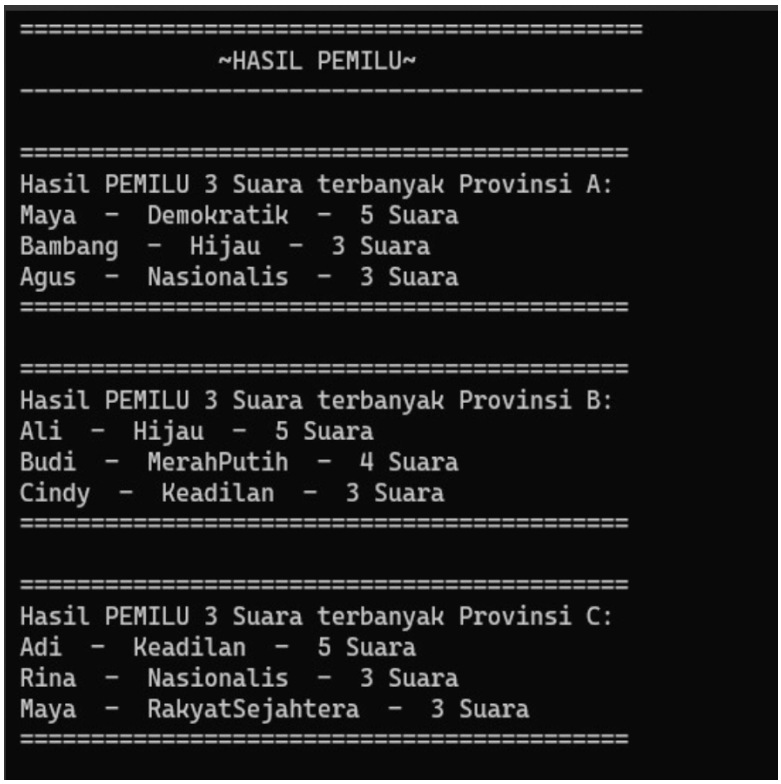
Gambar 3.3 tampilan menu Pemilih

4. Menu Pemilih

Gambar 3.2 memilih calon sebagai pemilih

Pada menu pemilih, pengguna masuk sebagai pemilih kemudian memasukkan tanggal saat ini. Jika tanggal sesuai dengan tanggal berlangsung nya pemilu, maka pemilih akan ditanya asal provinsinya. Setelah memilih asal provinsi, pemilih akan ditampilkan daftar calon yang ada pada provinsi tersebut. Kemudian pemilih dapat memilih dengan memasukkan nama dan nomor urut calon pilihan.

5. Hasil Pemilu



Gambar 3.1 tampilan hasil pemilu.

Berikut adalah tampilan akhir ketika pemilu telah berakhir, yaitu dengan memilih "Selesai" atau "Hasil Pemilu" pada menu yang disediakan.

BAB III
PENUTUP

3.1 Kesimpulan

Go-Pilu ini bertujuan untuk mempermudah proses pemilihan calon legislatif dengan mengutamakan efisiensi dan akurasi data. Go-Pilu ini dapat digunakan oleh KPU dan pengguna sebagai pemilih. Dengan adanya aplikasi ini, diharapkan pemilihan umum dapat dilaksanakan dengan lebih efektif, transparan, dan responsif terhadap kebutuhan pengguna.