Міністерство освіти і науки України  
Національний технічний університет України

«Київський політехнічний інститут ім. І. Сікорського»

Кафедра інженерії програмного забезпечення в енергетиці

Лабораторна робота № 6

з курсу: «*Програмування вебзастосунків*»

**Виконав:**  
студент 4-го курсу,  
групи ТВ-11  
Гудзовський Марк Юрійович

Посилання на GitHub репозиторій:

<https://github.com/Chivas1717/webapp-go-labs/tree/master/lab6>

**Перевірив:**

Недашківський О.Л.

Київ 2025

Лабораторна робота № 6

Варіант №4

**Хід виконання:**

Завдання 1:

Створіть веб калькулятор на go для розрахунку електричних навантажень об’єктів з використанням методу впорядкованих діаграм.

Цехова мережа складається з трьох типових цехів які під’єднується до трьох різних розподільчих шин (ШР1-ШР3) та кількох крупних електроприймачів (ЕП). Для спрощення приймемо що склад, номенклатура і характеристики ЕП всіх трьох цехів однакові. На основі складу ЕП та їх характеристик необхідно розрахувати силове навантаження цехової мережі.

Код розрахунку основних параметрів:

package main

import (

"fmt"

"html/template"

"log"

"math"

"net/http"

"strconv"

"strings"

)

// Equipment описує один електроприймач (чи групу ідентичних електроприймачів)

type Equipment struct {

Quantity int // Кількість

Power float64 // Потужність одного (кВт)

KV float64 // Коеф. використання

}

// calcLab6 – основна функція розрахунку

func calcLab6(EPList []Equipment) string {

var totalPower, totalKVPower, totalPowerSquare float64

for \_, ep := range EPList {

Pn := float64(ep.Quantity) \* ep.Power // сумарна потужність цієї групи

totalPower += Pn

totalKVPower += Pn \* ep.KV

totalPowerSquare += float64(ep.Quantity) \* math.Pow(ep.Power, 2)

}

// Kv, ne, Kr, Pp, Qp, Sp, Ip

Kv := totalKVPower / totalPower

ne := math.Round(math.Pow(totalPower, 2) / totalPowerSquare)

Kr := 1.25

Pp := Kr \* totalKVPower

Qp := Kv \* totalPower \* 1.57

Sp := math.Sqrt(Pp\*Pp + Qp\*Qp)

Ip := Pp / 0.38

return fmt.Sprintf(`Сумарна потужність: %.2f кВт

Сумарний добуток (P·KV): %.2f

Kv = %.3f

ne = %.0f

Kr = %.2f

Pp = %.2f кВт

Qp = %.2f квар

Sp = %.2f кВА

Ip = %.2f A (при U=0.38 кВ)`,

totalPower, totalKVPower,

Kv, ne, Kr, Pp, Qp, Sp, Ip,

)

}

// Парсер рядка "3,2.5,0.8;4,1.2,0.95" -> []Equipment

func parseEquipmentList(input string) ([]Equipment, error) {

var result []Equipment

// Розділяємо по крапці з комою (або пробілу, кому треба)

groups := strings.Split(input, ";")

for \_, g := range groups {

g = strings.TrimSpace(g)

if g == "" {

continue

}

// Очікуємо формат "Quantity,Power,KV"

parts := strings.Split(g, ",")

if len(parts) != 3 {

return nil, fmt.Errorf("неправильний формат рядка: %s", g)

}

q, err1 := strconv.Atoi(strings.TrimSpace(parts[0]))

p, err2 := strconv.ParseFloat(strings.TrimSpace(parts[1]), 64)

kv, err3 := strconv.ParseFloat(strings.TrimSpace(parts[2]), 64)

if err1 != nil || err2 != nil || err3 != nil {

return nil, fmt.Errorf("помилка перетворення значень у %s", g)

}

result = append(result, Equipment{

Quantity: q,

Power: p,

KV: kv,

})

}

return result, nil

}

// Структура для передачі даних у шаблон

type PageData struct {

Input string

Result string

Error string

}

var tmpl \*template.Template

func init() {

var err error

tmpl, err = template.ParseFiles("templates/index.html")

if err != nil {

log.Fatalf("Помилка завантаження шаблону: %v", err)

}

}

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

data := PageData{

// Дефолтний приклад:

// 3 шт. по 2.5 кВт, KV=0.8; 4 шт. по 1.2 кВт, KV=0.95

Input: "3,2.5,0.8;4,1.2,0.95",

}

if r.Method == http.MethodPost {

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

data.Error = "Помилка парсингу форми"

tmpl.Execute(w, data)

return

}

data.Input = r.FormValue("equip")

EPList, errParse := parseEquipmentList(data.Input)

if errParse != nil {

data.Error = errParse.Error()

} else {

data.Result = calcLab6(EPList)

}

}

tmpl.Execute(w, data)

}

func main() {

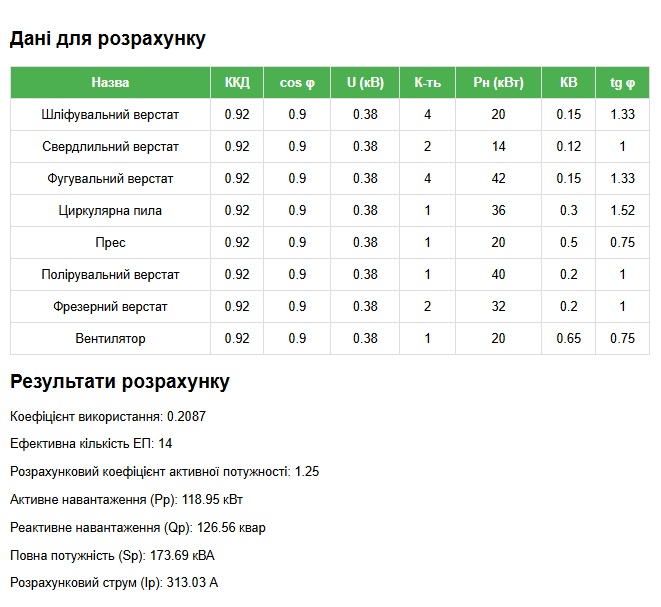
http.HandleFunc("/", handleCalc)

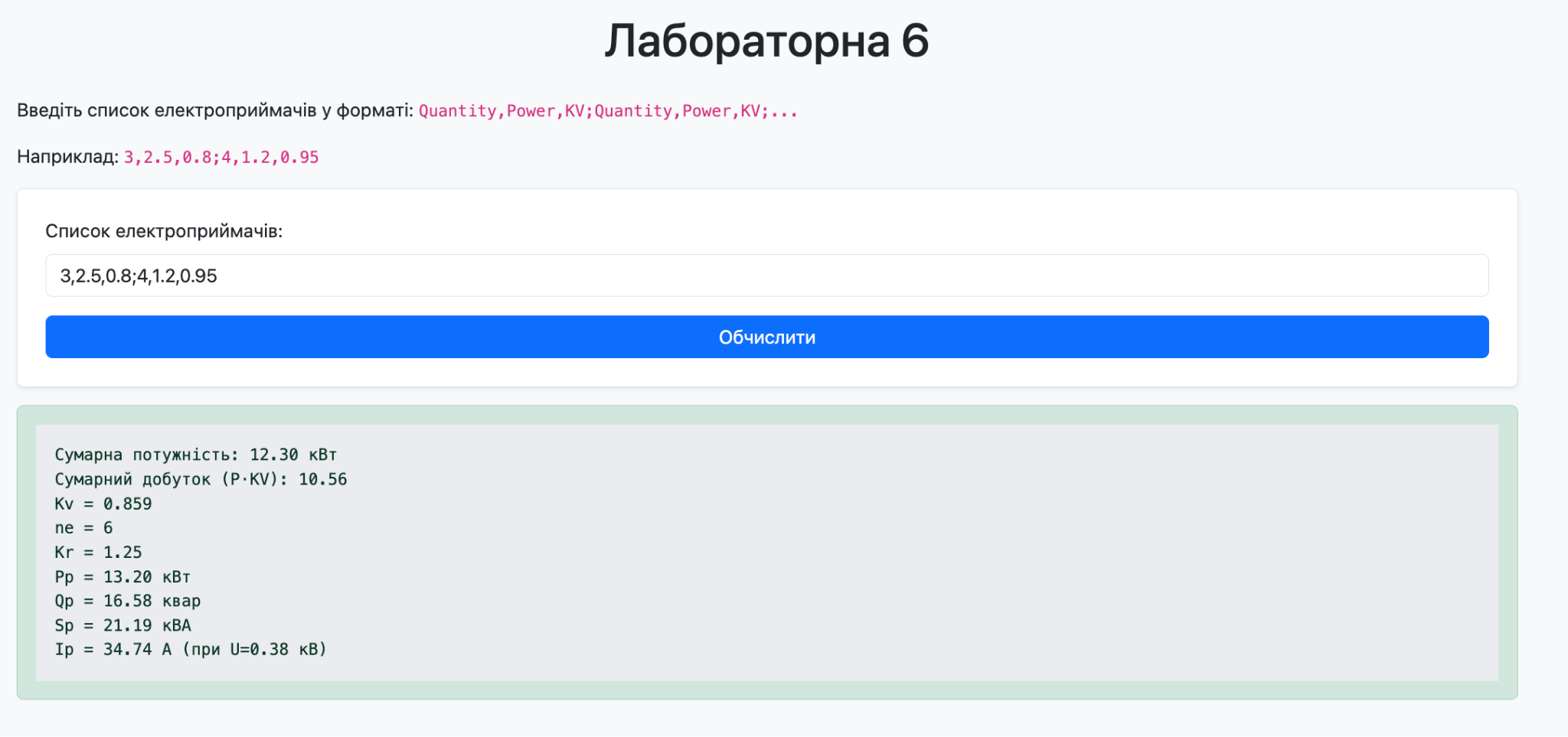
fmt.Println("Сервер запущено на http://localhost:8080/")

log.Fatal(http.ListenAndServe(":8080", nil))

}

Результат перевірки на контрольному прикладі:





**Висновок**

У рамках цієї лабораторної роботи було створено веб-додаток для розрахунку електричних навантажень об'єктів з використанням методу впорядкованих діаграм. Реалізовані алгоритми були перевірені на контрольних прикладах, отримані результати відповідають теоретичним очікуванням та можуть бути використані для подальшого аналізу та моделювання систем електропостачання.