Міністерство освіти і науки України  
Національний технічний університет України

«Київський політехнічний інститут  ім. І. Сікорського»

Кафедра інженерії програмного забезпечення в енергетиці

Лабораторна робота №5

з курсу: «Розробка програмного забезпечення мобільних пристроїв»

Виконав:

Студент 4-го курсу,

Групи ТВ-11

Гойчук Олександр Володимирович

Посилання на GitHub репозиторій: https://github.com/Hoychuk/Mob\_dev.git

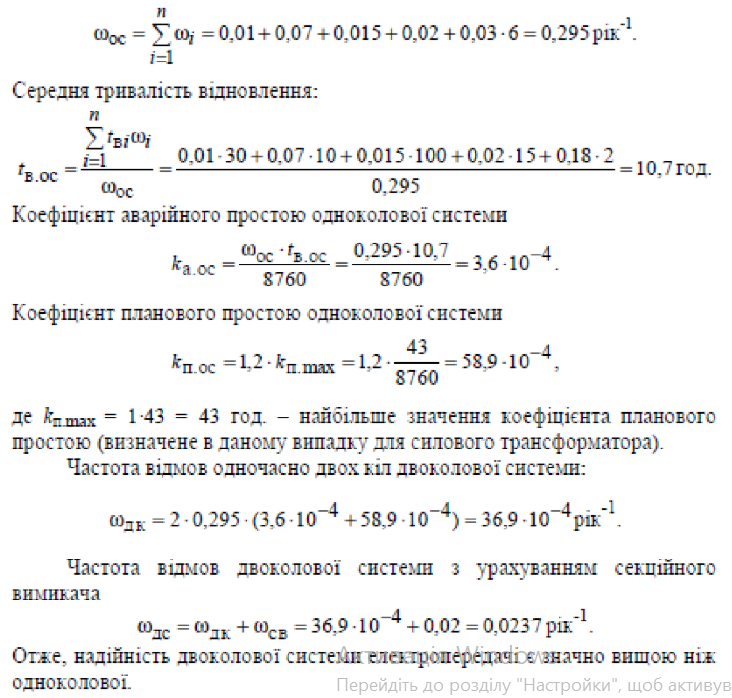
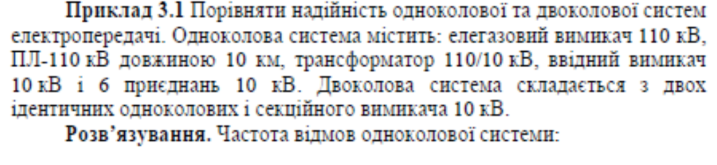
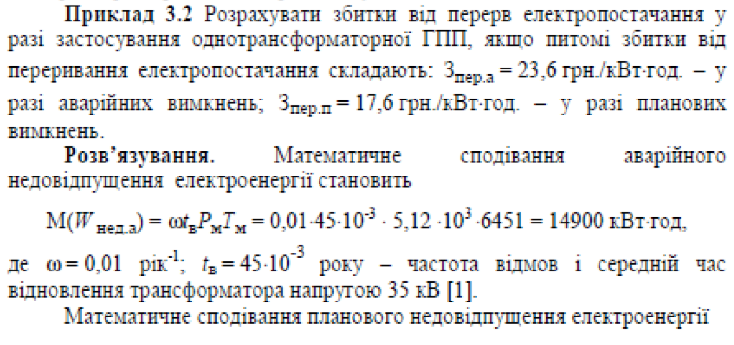
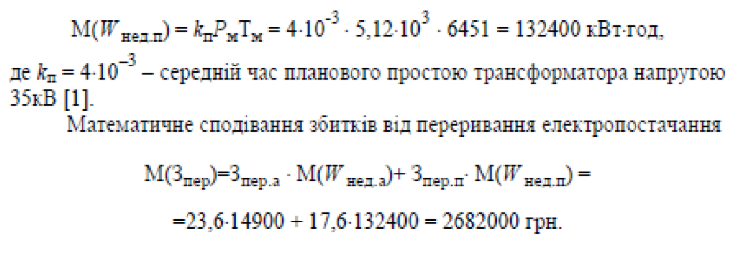
Перевірив:

Недашківський О.Л.

Київ 2024/2025

Лабораторна робота №5

**Теоретичний матеріал:**

**Завдання:**

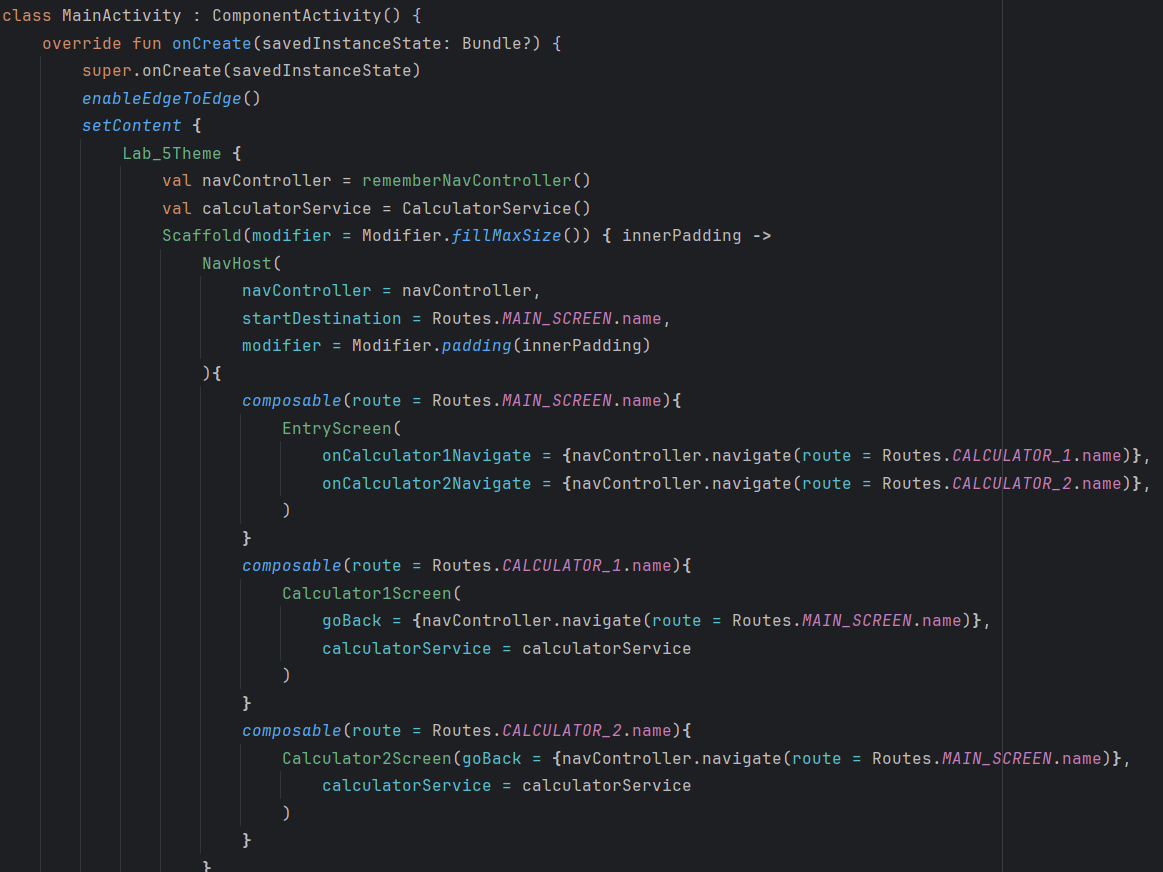
Створіть мобільний калькулятор для порівняння надійності одноколової та двоколової систем електропередачі та розрахунку збитків від перерв електропостачання у разі застосування однотрансформаторної ГТП у складі:

1. Порівняти надійність одноколової та двоколової систем електропередачі (див. Приклад 3.1.);

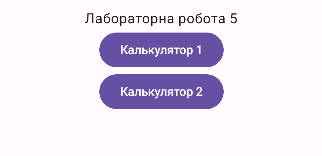
2. Розрахувати збитки від перерв електропостачання у разі застосування однотрансформаторної ГПП (див. Приклад 3.2.).

**Хід виконання:**

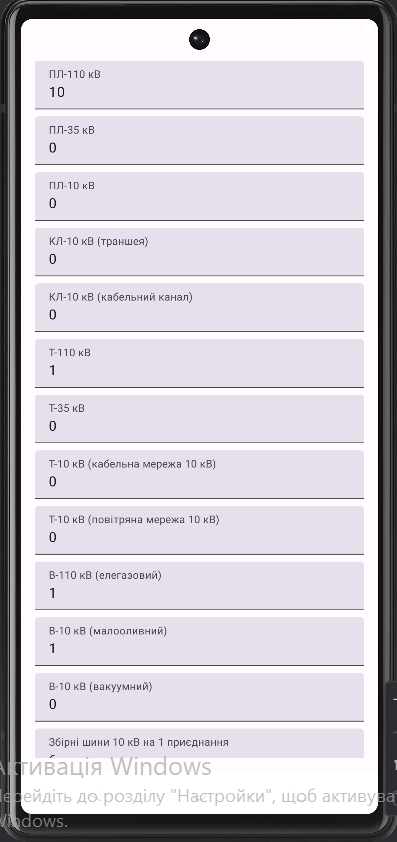
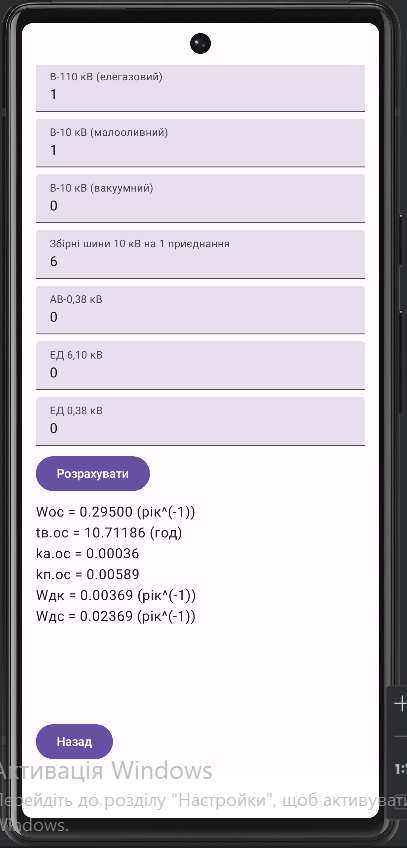
Спочатку реалізуємо навігацію між сторінками, яка дозволить між ними переміщатися:



Після цього реалізуємо дизайн першої сторінки, на якій будуть наявні кнопки для вибору 1 з калькуляторів:

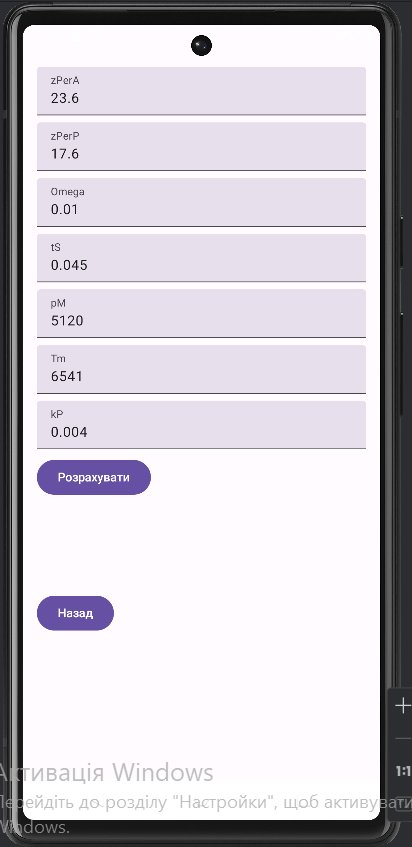
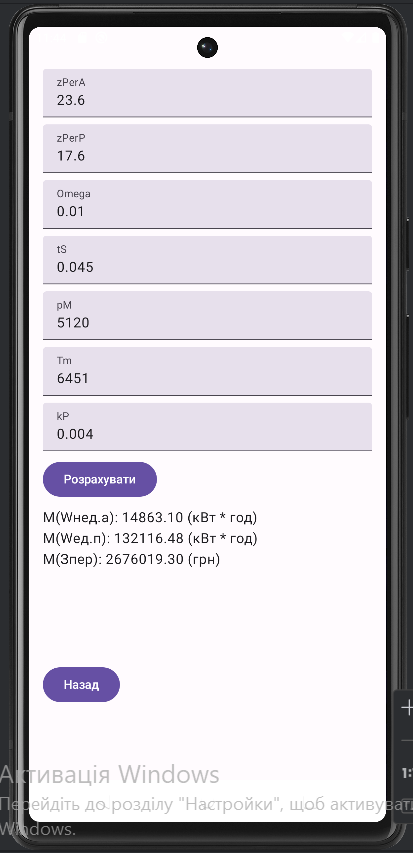
 

Далі реалізуємо дизайн та логіку першої сторінки, на якій будуть наявні поля, де будуть вводитися, потрібні нам коефіцієнти для розрахунку задачі та виводитися результати розрахунку:

package com.example.lab\_5.ui.calculator1  
  
import androidx.compose.foundation.layout.Arrangement  
import androidx.compose.foundation.layout.Box  
import androidx.compose.foundation.layout.Column  
import androidx.compose.foundation.layout.fillMaxSize  
import androidx.compose.foundation.layout.fillMaxWidth  
import androidx.compose.foundation.layout.padding  
import androidx.compose.foundation.rememberScrollState  
import androidx.compose.foundation.verticalScroll  
import androidx.compose.material3.Button  
import androidx.compose.material3.Text  
import androidx.compose.material3.TextField  
import androidx.compose.runtime.Composable  
import androidx.compose.runtime.MutableState  
import androidx.compose.runtime.getValue  
import androidx.compose.runtime.mutableStateOf  
import androidx.compose.runtime.remember  
import androidx.compose.runtime.setValue  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.unit.dp  
import com.example.lab\_5.services.CalculatorService  
  
data class ReliabilityIndicators(  
 val omega: Double,  
 val tV: Double,  
 val mu: Double,  
 val tP: Double  
)  
  
@Composable  
fun Calculator1Screen(  
 goBack: () -> Unit,  
 calculatorService: CalculatorService  
) {  
   
 //вхідні дані  
 val dataIndicators = *mapOf*(  
 "ПЛ-110 кВ" *to* ReliabilityIndicators(0.007, 10.0, 0.167, 35.0),  
 "ПЛ-35 кВ" *to* ReliabilityIndicators(0.02, 8.0, 0.167, 35.0),  
 "ПЛ-10 кВ" *to* ReliabilityIndicators(0.02, 10.0, 0.167, 35.0),  
 "КЛ-10 кВ (траншея)" *to* ReliabilityIndicators(0.03, 44.0, 1.0, 9.0),  
 "КЛ-10 кВ (кабельний канал)" *to* ReliabilityIndicators(0.005, 17.5, 1.0, 9.0),  
 "T-110 кВ" *to* ReliabilityIndicators(0.015, 100.0, 1.0, 43.0),  
 "T-35 кВ" *to* ReliabilityIndicators(0.02, 80.0, 1.0, 28.0),  
 "T-10 кВ (кабельна мережа 10 кВ)" *to* ReliabilityIndicators(0.005, 60.0, 0.5, 10.0),  
 "T-10 кВ (повітряна мережа 10 кВ)" *to* ReliabilityIndicators(0.05, 60.0, 0.5, 10.0),  
 "B-110 кВ (елегазовий)" *to* ReliabilityIndicators(0.01, 30.0, 0.1, 30.0),  
 "B-10 кВ (малооливний)" *to* ReliabilityIndicators(0.02, 15.0, 0.33, 15.0),  
 "B-10 кВ (вакуумний)" *to* ReliabilityIndicators(0.01, 15.0, 0.33, 15.0),  
 "Збірні шини 10 кВ на 1 приєднання" *to* ReliabilityIndicators(0.03, 2.0, 0.167, 5.0),  
 "АВ-0,38 кВ" *to* ReliabilityIndicators(0.05, 4.0, 0.33, 10.0),  
 "ЕД 6,10 кВ" *to* ReliabilityIndicators(0.1, 160.0, 0.5, 0.0),  
 "ЕД 0,38 кВ" *to* ReliabilityIndicators(0.1, 50.0, 0.5, 0.0),  
 )  
  
 //автозаповнення при запуску для спрощення уведення  
 fun createDefaultAmountMap(): Map<String, MutableState<String>> {  
 val defaultValues = *mapOf*(  
 "ПЛ-110 кВ" *to* "10",  
 "ПЛ-35 кВ" *to* "0",  
 "ПЛ-10 кВ" *to* "0",  
 "КЛ-10 кВ (траншея)" *to* "0",  
 "КЛ-10 кВ (кабельний канал)" *to* "0",  
 "T-110 кВ" *to* "1",  
 "T-35 кВ" *to* "0",  
 "T-10 кВ (кабельна мережа 10 кВ)" *to* "0",  
 "T-10 кВ (повітряна мережа 10 кВ)" *to* "0",  
 "B-110 кВ (елегазовий)" *to* "1",  
 "B-10 кВ (малооливний)" *to* "1",  
 "B-10 кВ (вакуумний)" *to* "0",  
 "Збірні шини 10 кВ на 1 приєднання" *to* "6",  
 "АВ-0,38 кВ" *to* "0",  
 "ЕД 6,10 кВ" *to* "0",  
 "ЕД 0,38 кВ" *to* "0"  
 )  
  
 return defaultValues.*mapValues* **{** *mutableStateOf*(**it**.value) **}** }  
  
 val amountMap = remember **{** createDefaultAmountMap() **}** var wOc by remember **{** *mutableStateOf*("") **}** var tVOc by remember **{** *mutableStateOf*("") **}** var kAOc by remember **{** *mutableStateOf*("") **}** var kPOc by remember **{** *mutableStateOf*("") **}** var wDk by remember **{** *mutableStateOf*("") **}** var wDc by remember **{** *mutableStateOf*("") **}** //функція заокруглення  
 fun round(num: Double) = "%.5f".*format*(num)  
  
 fun calculateResults(){//функція розрахунку  
 val result = calculatorService.calculateResult1(amountMap, dataIndicators)  
  
 wOc = result[0].toString()  
 tVOc = result[1].toString()  
 kAOc = result[2].toString()  
 kPOc = result[3].toString()  
 wDk = result[4].toString()  
 wDc = result[5].toString()  
 }  
  
  
 Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(start = 16.*dp*, top = 24.*dp*, end = 16.*dp*, bottom = 16.*dp*)  
 .*verticalScroll*(rememberScrollState()),  
 verticalArrangement = Arrangement.spacedBy(8.*dp*)  
 ) **{**//кожне поле для вводу даних  
 amountMap.*forEach* **{** (key, state) **->** TextField(  
 value = state.value,  
 onValueChange = **{** state.value = **it }**,  
 label = **{** Text(key) **}**,  
 maxLines = 1,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 )  
 **}** Button(//кнопку розрахунку  
 onClick = **{** calculateResults() **}** ) **{** Text("Розрахувати")  
 **}** if (wOc.*isNotEmpty*() && tVOc.*isNotEmpty*() && kAOc.*isNotEmpty*() && kPOc.*isNotEmpty*()  
 && wDk.*isNotEmpty*() && wDc.*isNotEmpty*()) {  
 Text(  
 """  
 Wос = ${round(wOc.*toDouble*())} (рік^(-1))  
 tв.ос = ${round(tVOc.*toDouble*())} (год)  
 kа.ос = ${round(kAOc.*toDouble*())}  
 kп.ос = ${round(kPOc.*toDouble*())}  
 Wдк = ${round(wDk.*toDouble*())} (рік^(-1))  
 Wдс = ${round(wDc.*toDouble*())} (рік^(-1))  
 """.*trimIndent*().*format*()  
 )  
 }  
 Box(//поле виведення  
 modifier = Modifier.*padding*(top = 100.*dp*)  
 ) **{** Button(//кнопка повернення  
 onClick = goBack  
 ) **{** Text("Назад")  
 **}  
 }  
 }**}

Після цього реалізуємо дизайн та логіку другої сторінки, на якій також будуть наявні поля, де будуть вводитися, потрібні нам коефіцієнти для розрахунку задачі наступної та виводитися результати розрахунку:

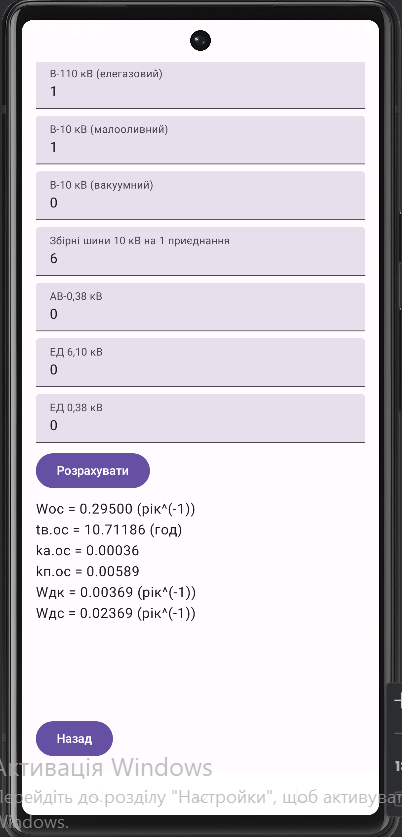
 

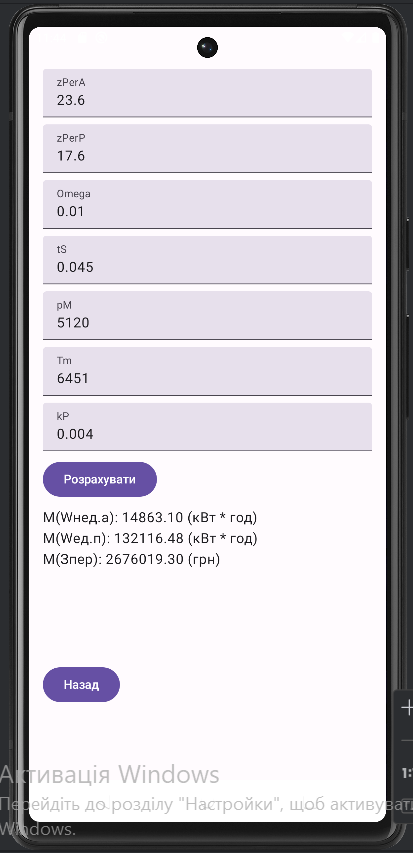
package com.example.lab\_5.ui.calculator2  
  
import androidx.compose.foundation.layout.Arrangement  
import androidx.compose.foundation.layout.Box  
import androidx.compose.foundation.layout.Column  
import androidx.compose.foundation.layout.fillMaxSize  
import androidx.compose.foundation.layout.fillMaxWidth  
import androidx.compose.foundation.layout.padding  
import androidx.compose.foundation.rememberScrollState  
import androidx.compose.foundation.verticalScroll  
import androidx.compose.material3.Button  
import androidx.compose.material3.Text  
import androidx.compose.material3.TextField  
import androidx.compose.runtime.Composable  
import androidx.compose.runtime.getValue  
import androidx.compose.runtime.mutableStateOf  
import androidx.compose.runtime.remember  
import androidx.compose.runtime.setValue  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.unit.dp  
import com.example.lab\_5.services.CalculatorService  
  
@Composable  
fun Calculator2Screen(  
 goBack: () -> Unit,  
 calculatorService: CalculatorService  
) {//автозаповнення для спрощення уведення  
 var zPerA by remember **{** *mutableStateOf*("23.6") **}** var zPerP by remember **{** *mutableStateOf*("17.6") **}** var omega by remember **{** *mutableStateOf*("0.01") **}** var tV by remember **{** *mutableStateOf*("0.045") **}** var Pm by remember **{** *mutableStateOf*("5120") **}** var Tm by remember **{** *mutableStateOf*("6541") **}** var kP by remember **{** *mutableStateOf*("0.004") **}** var mWnedA by remember **{** *mutableStateOf*("") **}** var mWnedP by remember **{** *mutableStateOf*("") **}** var mZper by remember **{** *mutableStateOf*("") **}** //функція заокруглення  
 fun round(num: Double) = "%.2f".*format*(num)  
   
 //виклик функції розранку  
 fun calculateResult(){  
 val formattedZPerA = zPerA.*toDoubleOrNull*() ?: 0  
 val formattedZPerP = zPerP.*toDoubleOrNull*() ?: 0  
 val formattedOmega = omega.*toDoubleOrNull*()?: 0  
 val formattedTV = tV.*toDoubleOrNull*()?: 0  
 val formattedPm = Pm.*toDoubleOrNull*()?: 0  
 val formattedTm = Tm.*toDoubleOrNull*()?: 0  
 val formattedKP = kP.*toDoubleOrNull*()?: 0  
  
 val result = calculatorService.calculateResult2(formattedZPerA.toDouble(),  
 formattedZPerP.toDouble(),  
 formattedOmega.toDouble(),  
 formattedTV.toDouble(),  
 formattedPm.toDouble(),  
 formattedTm.toDouble(),  
 formattedKP.toDouble())  
 mWnedA = result[0].toString()  
 mWnedP = result[1].toString()  
 mZper = result[2].toString()  
 }  
  
 Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(start = 16.*dp*, top = 24.*dp*, end = 16.*dp*, bottom = 16.*dp*)  
 .*verticalScroll*(rememberScrollState()),  
 verticalArrangement = Arrangement.spacedBy(8.*dp*)  
 ) **{**//текстові поля для уведення  
 TextField(  
 value = zPerA,  
 onValueChange = **{** zPerA = **it }**,  
 label = **{** Text("zPerA") **}**,  
 maxLines = 1,  
 modifier = Modifier.*fillMaxWidth*()  
 )  
 TextField(  
 value = zPerP,  
 onValueChange = **{** zPerP = **it }**,  
 label = **{** Text("zPerP") **}**,  
 maxLines = 1,  
 modifier = Modifier.*fillMaxWidth*()  
 )  
 TextField(  
 value = omega,  
 onValueChange = **{** omega = **it }**,  
 label = **{** Text("Omega") **}**,  
 maxLines = 1,  
 modifier = Modifier.*fillMaxWidth*()  
 )  
 TextField(  
 value = tV,  
 onValueChange = **{** tV = **it }**,  
 label = **{** Text("tS") **}**,  
 maxLines = 1,  
 modifier = Modifier.*fillMaxWidth*()  
 )  
 TextField(  
 value = Pm,  
 onValueChange = **{** Pm = **it }**,  
 label = **{** Text("pM") **}**,  
 maxLines = 1,  
 modifier = Modifier.*fillMaxWidth*()  
 )  
 TextField(  
 value = Tm,  
 onValueChange = **{** Tm = **it }**,  
 label = **{** Text("Tm") **}**,  
 maxLines = 1,  
 modifier = Modifier.*fillMaxWidth*()  
 )  
 TextField(  
 value = kP,  
 onValueChange = **{** kP = **it }**,  
 label = **{** Text("kP") **}**,  
 maxLines = 1,  
 modifier = Modifier.*fillMaxWidth*()  
 )  
 Button(//кнопка розрахунку  
 onClick = **{** calculateResult()**}** ) **{** Text("Розрахувати")  
 **}** if (mWnedA.*isNotEmpty*() && mWnedP.*isNotEmpty*() && mZper.*isNotEmpty*()) {  
 Text( """  
 M(Wнед.а): ${round(mWnedA.*toDouble*())} (кВт \* год)  
 M(Wед.п): ${round(mWnedP.*toDouble*())} (кВт \* год)  
 M(Зпер): ${round(mZper.*toDouble*())} (грн)  
 """.*trimIndent*().*format*(mWnedA, mWnedP, mZper)  
 )  
 }  
 Box(//вікно результатів  
 modifier = Modifier.*padding*(top = 100.*dp*)  
 ) **{** Button(//кнопка повернення  
 onClick = goBack  
 ) **{** Text("Назад")  
 **}  
 }  
 }**}

На кінець реалізовуємо клас, який відповідатиме за математичні розрахунки обох прикладів:

package com.example.lab\_5.services  
  
import androidx.compose.runtime.MutableState  
import com.example.lab\_5.ui.calculator1.ReliabilityIndicators  
  
class CalculatorService {  
 fun calculateResult1(amountMap: Map<String, MutableState<String>>,  
 dataIndicators: Map<String, ReliabilityIndicators>): List<Double>{  
 var wOc = 0.0  
 var tVOc = 0.0  
  
 amountMap.*forEach* **{** (key, value) **->** val amount = value.value.*toIntOrNull*() ?: 0  
 val indicator = dataIndicators[key] ?: return@forEach  
  
 if (amount > 0) {  
 wOc += amount \* indicator.omega  
 tVOc += amount \* indicator.tV \* indicator.omega  
 }  
 **}** tVOc /= wOc  
 val kAOc = (tVOc \* wOc) / 8760  
 val kPOs = 1.2 \* 43 / 8760  
 val wDk = 2 \* wOc \* (kAOc + kPOs)  
 return *listOf*(wOc, tVOc, kAOc, kPOs, wDk, wDk + 0.02)  
 }  
  
 fun calculateResult2(formattedZPerA: Double,formattedZPerP: Double,formattedOmega: Double,  
 formattedTV: Double,formattedPm: Double,formattedTm: Double,  
 formattedKP: Double): List<Double>{  
 val mWnedA = formattedOmega \* formattedTV \* formattedPm \* formattedTm  
 val mWnedP = formattedKP \* formattedPm \* formattedTm  
 val mZper = formattedZPerA \* mWnedA + formattedZPerP \* mWnedP  
  
 return *listOf*(mWnedA, mWnedP, mZper)  
 }  
  
}

**Результат виконання контрольних прикладів:**

**** ****

****

0,01\*45\*10-3\*5,12\*103\*6451=14863,1, якщо розраховувати стороннім способом, тобто розрахунки вірні.

**Висновок:**

У результаті виконання лабораторної роботи №5 було реалізовано калькулятор, для порівняння надійності одноколової та двоколової систем електропередачі та розрахунку збитків від перерв електропостачання у разі застосування однотрансформаторної ГТП, по введених значеннях, за допомогою мови Kotlin. Також було отримано навички з використання засобу розробки jetpack composer.