Razvoj mobilne programske podrške za izračun provjesa neizoliranih energetskih vodova

Mihalj, Matko

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
ELEKTROTEHNIČKI FAKULTET

Preddiplomski studij računarstva

RAZVOJ MOBILNE PROGRAMSKE PODRŠKE ZA IZRAČUN PROVJESA NEIZOLIRANIH ENERGETSKIH VODOVA

Završni rad

Matko Mihalj

Osijek, 2021

1.UVOD

Tema ovog završnog je izrada mobilne aplikacije za Android koja služi za izračun provjesa neizoliranih energetskih vodova. U prvom dijelu završnog rada prikazuje se način na koji se računa provjes neizoliranih energetskih vodova, prikazuju se kritične temperature na kojima se izračunava provjes, također se predstavljaju i potrebne formule za izračun, te svi parametri koji su potrebni. U drugom dijelu opisane su karakteristike tehnologije i primjeri koda koje se koriste prilikom izrade same android aplikacije, te prilikom izračuna provjesa neizoliranih energetskih vodova. Treći dio završnog rada sadrži opis algoritma za izračun provjesa neizoliranih energetskih vodova koji je napisan za Android u programskom jeziku Java. Opisan je izgled aplikacije, funkcionalnosti aplikacije i njena struktura. U četvrtom dijelu se prikazuju dobiveni rezultati koje izračunava izrađena aplikacija s pomoću metoda koja su opisane u prvom dijelu. U posljednjem petom dijelu analiziraju se dobiveni rezultati i provjeravaju da li aplikacija i ručni izračun daju iste rezultate.

1.1 Zadatak završnog rada

Razvoj mobilne programske podrške za izračun provjesa neizoliranih energetskih vodova. Potrebno je izraditi android mobilnu aplikaciju i algoritam za izračun provjesa neizoliranih energetskih vodova na osnovu svojstava vodiča, temperature okoliša, te mjerenjem visine stupova na kojima su vodovi.

2. IZRAČUN PROVJESA NEIZOLIRANIH ENERGETSKIH VODOVA

Mehanički proračun neizoliranih vodiča je glavni dio dalekovoda kojima se određuje sigurnost neizoliranog vodiča, također s pomoću mehaničkog proračuna određuje se naprezanje samog vodiča i provjes pri kritčnim temperaturama. Dopušteno naprezanje određenih neizoliranih vodiča određuje se pri najnepovoljnijim uvjetima, tada dolazi do najvećeg naprezanja.

2.1. Parametri potrebni za izračun mehaničkog proračuna

Prvi korak prilikom izračuna mehaničkog proračuna je sakupljanje svih parametara vodiča, zato što ponašanje neizoliranog vodiča ovisi o svojstvima materijala od kojeg je izgrađen, te način na koji je izrađen. Prema [1] parametri vodiča koji su nama potrebni za izračun mehaničkog proračuna su :

- Iznimno dozvoljeno istezanje- σi [N/mm2]
- Normalno dozvoljeno istezanje – σd [N/mm2]
- Modul elastičnosti – E [N/mm2]
- Koeficijent linearnog toplinskog istezanja – β [1/°C]
- Uzdužna masa – m [kg/m]
- Promjer neizoliranog vodiča – d [mm]
- Presjek neizoliranog vodiča – A [mm2]

Osim parametara o vodiču, potrebno je znati i podatke o visini i rasponu samih stupova između kojih računamo provjes neizoliranih energetskih vodiča:

- a – raspon [m]
- $h12$ – denivelacija(razlika u visini stupova) [m]
- a' – spojnica(udaljenost između hvatišta dvaju stupova) [m]
- $h2$ – visina 2. Stupa [m]
- $h1$ – visina 1. Stupa [m]
- k-faktor normalnog dodatnog tereta

2.2. Formule za izračun provjesa

U ovom ulomku opisan je postupak i matematičke formule koje se koriste prilikom izračuna provjesa neizloranih energetskih vodova. Prema [2] dan je izračun za reduciranu težinu neizoliranog vodiča i leda. Reduciranu težinu neizoliranog vodiča dobiva se po formuli:

$$go = \frac{Go}{A} = \frac{m*g}{A}\ [\frac{N}{m*mm^2}] \tag{2-1}$$

Gdje su:

- Go - težina neizoliranog vodiča $[\frac{N}{m}]$
- A – presjek neizoliranog vodiča $[mm^2]$
- m – jedinična masa neizoliranog vodiča $[\frac{Kg}{m}]$
- go - reducirana težina neizoliranog vodiča$[\frac{N}{m*mm^2}]$
- g - gravitacija sila- 9.81 $[\frac{m}{s}]$

Osim težine neizoliranog vodiča, na ukupnu težinu neizoliranog vodiča utječe još i dodatna opterećenja kao što su snijeg i led koji imaju veliki utjecaj pri samom izračunu provjesa. Prema [2] Normalno dodatno opterećenje dobiva se formulom:

$$G_{10} = 0.18 * \sqrt{d} * g\ [\frac{N}{m}] \tag{2-2}$$

Gdje je:

- d- promjer neizoliranog energetskog vodiča [m]

Prilikom izračuna dodatnog opterećenja u obzir se mora uzeti i k (koeficijent hidrometeoroloških podataka) koji vrijedi za izgradnju električnog voda na nekom geološkom području. Najpoznatije vrijednosti koeficijenta hidrometeoroloških podataka-k su: k=1.6, k=1.0, k=3.0, k=4.0, k=2.5. Zbog toga što se koeficijent hidrometeoroloških podataka mora uzeti u obzir dobiva se formula:

$$G_l = k * G_{10} \left[\frac{N}{m}\right] \tag{2-3}$$

Razradom izraza (2-1) i (2-2) dobiva se slijedeći izraz koji predstavlja reduciranu težinu zaleđenog neizoliranog vodiča:

$$G_z = g_0 + G_l \quad \left[\frac{N}{m*mm^2}\right] \tag{2-4}$$

Slijedeći korak tijekom izračuna je usporedba idealnog i kritičnog raspona neizoliranog vodiča. Potrebno je odrediti njihov odnos kako bi se moglo odrediti početno stanje neizoliranog vodiča koje je potrebno za daljnje računanje provjesa. Formula s pomoću koje možemo dobiti kritični raspon je:

$$a_k = \sigma_{max} * \sqrt{(360 * \beta)/(gz^2 - g0^2)} \quad \text{[m]} \tag{2-5}$$

Gdje je:

σ_{max} - maksimalno naprezanje neizoliranog vodiča na -5 °C sa dodatnim teretom ili na -20°C te je prema [1] uvijek jednako ili manje normlanom dopuštenom rastezanju neizoliranog vodiča.

Idealni raspon dobiva se po formuli:

$$a_{idealno} = \sqrt{\frac{\sum_{i=1}^{n} ai^3}{\sum_{i=1}^{n} \frac{ai'^2}{ai}}} * \frac{\sum_{i=1}^{n} \frac{ai'^3}{ai^2}}{\sum_{i=1}^{n} \frac{ai'^2}{ai}} \quad \text{[m]} \tag{2-6}$$

Nakon izračuna kritičnog(2-4) i idealnog raspona(2-5) rezultati se uspoređuju kako bi se dobilo željeno početno stanje vodiča. Imamo dvije mogućnosti:

1) $a_{idealno} < a_k$

U ovom slučaju za početno stanje postavlja se -20°C bez opterećenja tj. postavljaju se slijedeće veličine za izračun:

- $\sigma 1 = \sigma max \left[\frac{N}{mm^2}\right]$

- $\theta 1 = -20°C$
- $g1 = g0\ [\frac{N}{m*mm^2}]$

2) $a_{idealno} > a_k$

U ovom slučaju za početno stanje postavlja se -5°C sa dodatnim opterećenjem, tj. postavljaju se slijedeće veličine za izračun:

- $\theta 1 = -5\ °C$
- $g1 = gz\ \ [\frac{N}{m*mm^2}]$
- $\sigma 1 = \sigma max\ [\frac{N}{mm^2}\ ]$

Nakon što se odredi o tome kakvo će biti početno stanje, potrebno je odrediti horizontalno naprezanje na pojedinim temperaturama. Ako ima visinske razlike između stupova(denivelacije), potrebno je izračunati nadomjesno naprezanje:

$$\bar{\sigma 1} = \sigma 1 * \frac{\sum_{i=1}^{n} \frac{ai'^3}{ai^2}}{\sum_{i=1}^{n} \frac{ai'^2}{ai}} \qquad [\frac{N}{mm^2}\ ] \tag{2-7}$$

U slučaju da nema visinske razlike između stupova(denivelacija) onda je trasa u potpunosti horizontalna i tada vrijedi formula:

$$\bar{\sigma}1 = \ \sigma 1 = \sigma\text{max} \qquad [\frac{N}{mm^2}\ ] \tag{2-8}$$

Do traženog horizontalnog naprezanja $\sigma 2\ [\frac{N}{mm^2}\ ]$ koji ovisi o temperaturi dolazi se slijedećim izrazom:

$$\frac{\bar{\sigma}1 - \bar{\sigma}2}{E} + \beta(\theta 1 - \theta 2) = \frac{aidealno^2}{24} * \left(\frac{g1^2}{\bar{\sigma}1^2} - \frac{g2^2}{\bar{\sigma}2^2}\right) \qquad [\frac{N}{mm^2}\ ] \tag{2-9}$$

Gdje je:

- $\theta 2$ – temperatura pri kojoj se računa provjes [$°C$]
- E - module elastičosti [$\frac{N}{mm^2}$]
- g2 – ima ovisnost o temperaturi $\theta 2$, za $\theta 2 = -20$ °C i za $\theta 2 = -40$ °C vrijedi g2=g0, a za slučaj $\theta 2 = -5$ °C vrijedi da je g2=gz
- β – linearni toplinski koeficijent [1/°C]
- $\overline{\sigma}2$ - nadomjesno naprezanje [$\frac{N}{mm^2}$]

Rješavanjem jednadžbe dolazi se do izračuna za nadomjesno naprezanje ($\overline{\sigma}2$) [$\frac{N}{mm^2}$] pri našoj temperarturi na kojoj se računa .Nakon izračuna nadomjesnog naprezanja, kreče se prema računanju stvarnog naprezanja po formuli:

$$\sigma_2 = \overline{\sigma_2} * \frac{\sum_{i=1}^{n} \frac{ai'^2}{ai}}{\sum_{i=1}^{n} \frac{ai'^3}{ai^2}} \qquad [\frac{N}{mm^2}] \tag{2-10}$$

Cijeli proces se ponavlja za 3 različite temperature:

1) +40 $°C$.

2) −5 $°C$ sa dodatnim teretom

3)−20 $°C$ bez dodatnog tereta

Kao zadnji, ali ne i manje bitan korak je izračun samog provjesa na odabranom rasponu i odabranoj temperaturi:

I. Horizontalni raspon- ukoliko nema razlike u visini stupova

$$f = \frac{a^2 * g_2}{8 * \sigma_2} \quad [\text{m}] \tag{2-11}$$

II. Vertikalni raspon- u slučaju da postoji visinska razlika među stupovima

$$f' = f * \frac{a'}{a} = \frac{a^2 * g_2}{8 * \sigma_2} * \frac{a'}{a} \quad [\text{m}] \tag{2-12}$$

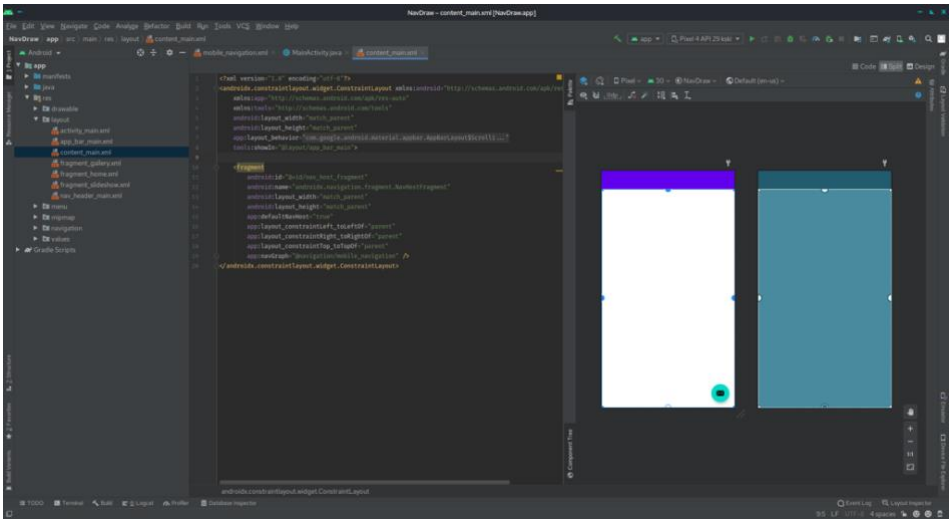
3.TEHNOLOGIJE KORIŠTENE PRI IZRADI APLIKACIJE

3.1. Android studio

Android studio je najpopularnije razvojno okruženje za razvoj Android aplikacija, daje podršku za sve mobilne uređaje ,tablete, TV uređaje, itd. Omogućuje instatno pokretanje koda na uređaju ili emulatoru. Za razvoj android aplikacije zahtijeva instaliranje JDK(eng. *Java Development Kit*). Ima mogućnost korištenja na računalima imaju operacijski sustav *Linux, MAC, Windows*. Na slici 1. prikazano je razvojno okruženje Android studija. Kao što vidimo s lijeve strane android studija imamo strukturu datoteka, jedna od temeljnih datoteka jest Manifest datototeka.Riječ je o datoteci pisanoj XML(eng. *Extensible Markup Language*) opisnim jezikom koja sadrži metapodatke o aplikaciji, definira strukturu aplikacije, sve aktivnosti, servise, pružatelje sadržaja, dozvole, itd. Kod android aplikacije razdvaja se od resursa (izgleda, slika i drugih podataka) zato što olakšava prilagodbu različitim uređajima, omogučuje lakše izmjene. Resursi aplikacije smješteni su u posebnoj mapi koja se naziva res mapa. Imamo dva načina stvaranja korisničkog sučelja a to su:

1)*drag-and-drop* - omogućuje programeru da prenosi elemente koje želi korisiti

2) *XML*(*eng. Extensible Markup Language*) - pisani kod koji nam omogućuje stvaranju vizualnog dijela aplikacije



Slika 3.1. razvojno Okruženje Android Studio [6]

3.2. Java

James Gosling je izvorno razvio Javu u tvrtki pod nazivom *Sun Microsystems Inc* koja je kasnije preuzeta od strane tvrtke pod nazivom *Oracle Corporation* . Razvoj objektno orijentiranog programskog jezika java započeo je 1991.godine u projektu *Green* koji je objavljen 1995.godine. Java je razvojno okruženje koje je besplatno svima na korištenje iako tvrtka *Sun* ima prava na ime *Java*. Java programi se izvode u *bytecode* na svim sustavima koja posjeduju JVM(eng. *Java Virtual Machine*). *Java* iako je nastala po uzoru na C-programskom jeziku daje puno veću sigurnost zahvaljujući VM (eng. *Virtual Machine*) i razvojnom okolišu u kojem se programi izvode. Neke od verzija Java-e su: *JDK Beta*(1995), *JDK 1.0*(1996), *Java SE 7*(2011), *Java SE 11*(2018), *Java SE 14*(2020), *Java SE 15*(2020), *Java SE 16*(2021). Slika 2. prikazuje kod napisan u orijentirano programskom jeziku *Java*. Tijekom stvaranja *Java* bilo je pet primarnih ciljeva a to su:

1.mora biti jedostavna,objektno-orijentirana

2. mora biti sigurna

3.mora biti arhitektonski neutralna i prenosiva

4.mora se izvršiti s visokih perfomansi

5.mora biti dinamična

```
package rentalStore;
import java.util.Enumeration;
import java.util.Vector;

class Customer {
    private String _name;
    private Vector<Rental> _rentals = new Vector<Rental>();

    public Customer(String name) {
        _name = name;
    }
    public String getMovie(Movie movie) {
        Rental rental = new Rental(new Movie("", Movie.NEW_RELEASE), 10);
        Movie m = rental._movie;
        return movie.getTitle();
    }
    public void addRental(Rental arg) {
        _rentals.addElement(arg);
    }
    public String getName() {
        return _name;
    }
```

Slika 3.2. Primjer koda napisan u „Java“ [7]

3.3 Operacijski sustav-Android

Android operacijski sustav je mobilni operacijski sustav koji je zasnovan je na modificiranoj verziji *Linux*, dizajniran je ponajprije za mobitele i tablete koji imaju ekran osjetljiv na čovjekov dodir , ali zbog svoje velike raspostranjenosti može se korisiti i na TV uređajima, automobilima i pametnim satovima. *Android* operacijski sustav predstavljen je u studenom 2007.godine na prvom komercijalnom android uređaju koji je bio *HTC Dream*. Andy Rubin, Chris White, Rich Miner, Nick Sears su osnovali *Android Inc.* 2003.godine. *Android* je besplatan softver otvorenog koda čiji je izvorni kod poznat kao *AOSP* (eng. *Android Open Source Project)*. 70% *Android* pametnih telefona pokreće *Google* ekosustav. *Android* kod služio je pri razvoju nizu velikih elektronskih uređaja kao što su: digitalni fotoaparati, prijenosni medijski playeri i računala. *Android* operacijski sustav je naprodavaniji među pametnim telefonima na svijetu još od 2011.godine., a na tabletima od 2013.godine. *Android* operacijski sustav ima preko tri milijarde korisnika aktivnih mjesečno. Slikovni formati koje podržava *Android* su: *.jpg(eng. Joint Photographic Experts Group)*, *.gif(eng. The Graphics Interchange Format),* itd. Od dodatnih ugrađenih podrški Android ima ugrađeno *Gps(eng. The Global Positioning System),* žiroskop i grafičku 3D akceleraciju. Neke od najpoznatijih verzija Androida su: Cupcake(2009), Froyo(2010), Honeycomb(2011), Jelly Bean(2012), Kitkat(2013), Lollipop(2014), Oreo(2017), Android 10(2019) i Android 11(2020). Arhitektura Androida promatra se u više razina. Najniža razina Androida je *Linux* jezgra koja sadrži drivere: driver za zaslon, driver za kameru, driver za wifi, driver za tipkovnicu, driver za zvuk, upravljač napajanja. Iznad jezgre nalazi se knjižnica koji su pisani u objektno orijentiranom jeziku C. Knjižnica se sastoji od: crtanja grafičkog sučelja , 2D knjižnica, sigurnosnih komunikacija, engine za web preglednike , baze podataka i iscrtavanje fontova. *Android Runtime* sloj sastoji se od*: Core Libraries* i *Dalvik Virtaul Machine* koje nam služe za pokretanje aplikacije. Slijedeći sloj nama je aplikacijski sloj koji se sastoji od mehanizama koji nama pomažu pri razvijanju aplikacija. Aplikacijski sloj nama omogućuje korištenje API(eng. *Application Programming Interface*). Aplikacijski sloj sadrži: upravitelj paketa(eng. *Package Manager)*, upravitelj lokacija (eng. *Location Manager)* i davatelj sadržaja*(eng. Content provider).* Na samom vrhu arhitekture Androida nalaze se Aplikacije koje se mogu pronaći u *Android Market*, te je aplikacijski sloj vidljiv svim korisnicima *Android*.

4. IZRADA ANDROID APLIKACIJE

U prošlim poglavljima obrađen je način na koji se računa provjes i formule potrebne za izračun . U ovom poglavlju izrađuje se aplikacija koja će nam omogućiti brže i lakše računanje. Formule su dosta velike i zahtjevne i ima dosta izračuna. Aplikacija je razvijana u Android Studio-u u programskom jeziku Java. Sama svrha aplikacije je ubrzanje procesa računanja provjesa neizoliranih energetskih vodova

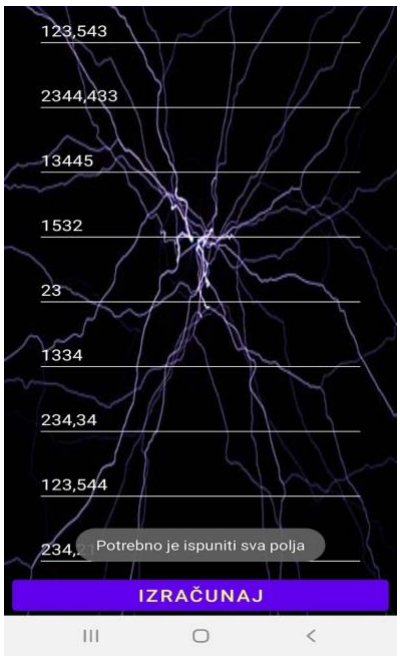
4.1. Korisničko sučelje Android aplikacije

Korisničko sučelje andorid aplikacije omogućava korisniku upravljanje aplikacijom. Kao što je vidljivo na Slici 3. Aplikacija je koncipirana kao kalkulator, unose se vrijednosti odgovarajućih parametara u odgovarajuće polje . Pritiskom na gumb „Izračunaj“ aplikacija izvršava algoritam napisan u programskom jeziku Javi koji izračunava provjes neizoliranih energetskih vodova.

Slika 4.1. Korisničko sučelje aplikacije

Kako bi se izbjegle poteškoće u radu same aplikacije postavljeni su uvjeti koji moraju biti zadovoljeni kako bi aplikacija radila. Na slici 4. Vid se prvi uvjet koji kaže da aplikacija mora imat unesena sva polja kako bi se mogao pritisnuti gumb „Izračunaj“ te kako bi se mogao izvesti algoritam aplikacije za izračun. Na algoritmu ispod teksta vidi se uvjet napisan u programskom jeziku Java.

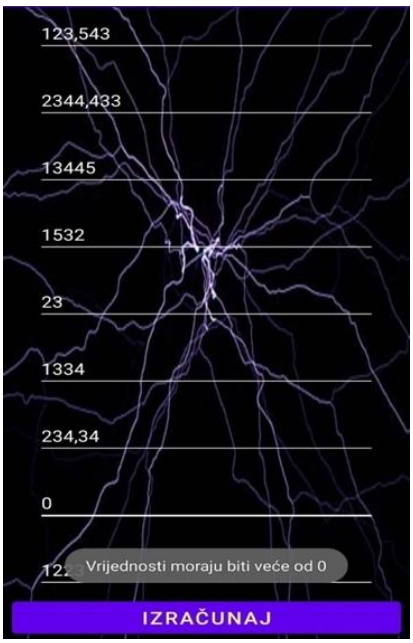
```
private Boolean checkInputFieldsAreEmpty() {
        return presjekVodicaUnos.getText().toString().equals("")
                || promjerVodicaUnos.getText().toString().equals("")
                || masaVodicaUnos.getText().toString().equals("")
                || koefToplinskogIstezanjaUnos.getText().toString().equals("")
                || faktorNormDodatnogTeretaUnos.getText().toString().equals("")
                || moduleElasticnostiUnos.getText().toString().equals("")
                || normalnoDozvoljenoNaprezanjeUnos.getText().toString().equals("")
                || iznimnoDozvoljenoNaprezanjeUnos.getText().toString().equals("")
                || maksimalnoNaprezanjeUnos.getText().toString().equals("")
                || rasponStupovaUnos.getText().toString().equals("")
                || denivelacijaUnos.getText().toString().equals("");
    }
//    }
}
```



Slika 3.2. Korisničko sučelje s uvjetom

Drugi uvjet aplikacije je da sve vrijednosti moraju biti veće od 0 (x>0),osim za visinsku razliku između stupova koja može biti jednaka 0. Na slici 5. vidi se upozorenje za uvjet(vrijednosti >0) u korisničkom sučelju. Na algoritmu ispod teksta vidi se uvjet(unesene vrijednosti >0) napisan u programskom jeziku „Java“.

```
if(saveCalculations(calculations.getJsonResult())){
            Intent intent = new Intent(this, Results.class);
            startActivity(intent);
        }else{
            Toast.makeText(MainActivity.this, "Nesto nije u redu, probaj ponovno!",
                    Toast.LENGTH_SHORT).show();
        }
    } catch (IllegalArgumentException exception) {
        Toast.makeText(MainActivity.this, "Vrijednosti moraju biti veće od 0",
                Toast.LENGTH_SHORT).show();
    } catch (JSONException e) {
        e.printStackTrace();
    }
}
```



Slika 4.3. Uvjet da sve vrijednosti moraju biti >0

Pritiskom na gumb aplikacije „Izračunaj“ pokazuje se rješenje korisniku aplikacije za paramtere koji su unešeni .Aplikacija pamti paramtere unešene tako da ako je došlo do pogreške u zadnjem izračunu samo pritisnemo gumb za nazad i vrati se na korisničko sučelje gdje se unose parametri. Korisničko sučelje je jedini dio Android aplikacije koji je napisan u XML-u. Na algoritmu ispod teksta može se vidjeti dio koda napisan u XML-u. Vidi se da su postavljena *EditText* polja za unos vrijednosti parametara, te polje Button koji će biti okidač za sumaciju svim vrijednosti i njihov izračun.

```
<EditText
    android:id="@+id/et_PresjekVodiča"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_marginStart="32dp"
    android:layout_marginTop="32dp"
    android:layout_marginEnd="32dp"
    android:hint="Presjek Vodiča"
    android:textColor="@color/white"
    android:textColorHint="@color/white"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    android:backgroundTint="@color/white"
    />


    <Button
    android:id="@+id/mybutton"
    android:onClick="mybutton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:layout_marginLeft="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginRight="8dp"
    android:layout_marginBottom="8dp"
    android:text="IZRAČUNAJ"
    app:layout_constraintTop_toBottomOf="@+id/et_Denivelacija"
    android:textSize="20sp"
    android:textColor="@color/colorEt2"
    />
```

4.2. Povezivanje korisničkog sučelja i varijabli android aplikacije

Nakon definiranja XML-a i polja za unos parametara, potrebno je povezati korisničko sučelje s glavnim dijelom aplikacije koji se izvodi u programskom jeziku Java-i. U glavnom dijelu aplikacije primjenjene su formule napisane u potpoglavlju 2.2 koje služe za izračun provjesa neizoliranih energetskih. Algoritmom prikazanim ispod pokazuje se kako se dodjeljuju vrijednosti parametara koja su unešena u polja varijablama u glavnom dijelu koda aplikacije.

```
@Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      initEditTexts();
      initFileDataBase();
   }

   private void initFileDataBase() {
      fileDataBaseReaderWriter = new FileDataBaseReaderWriter(this);
   }

   private void initEditTexts() {
      presjekVodicaUnos = (EditText) findViewById(R.id.et_PresjekVodiča);
      promjerVodicaUnos = (EditText) findViewById(R.id.et_PromjerVodiča);
      masaVodicaUnos = (EditText) findViewById(R.id.et_LinijskaMasaVodiča);
      koefToplinskogIstezanjaUnos = (EditText)
findViewById(R.id.et_koefToplinskogIstezanja);
      faktorNormDodatnogTeretaUnos = (EditText)
findViewById(R.id.et_faktorNormDodatnogTereta);
      moduleElasticnostiUnos = (EditText) findViewById(R.id.et_ModulElastičnosti);
      normalnoDozvoljenoNaprezanjeUnos = (EditText)
findViewById(R.id.et_NormDozvoljenoNaprezanje);
      iznimnoDozvoljenoNaprezanjeUnos = (EditText)
findViewById(R.id.et_IznimnoDozvoljenoNaprezanje);
      maksimalnoNaprezanjeUnos = (EditText)
findViewById(R.id.et_MaksimalnoNaprezanje);
      rasponStupovaUnos = (EditText) findViewById(R.id.et_RasponStupova);
      denivelacijaUnos = (EditText) findViewById(R.id.et_Denivelacija);
   }
```

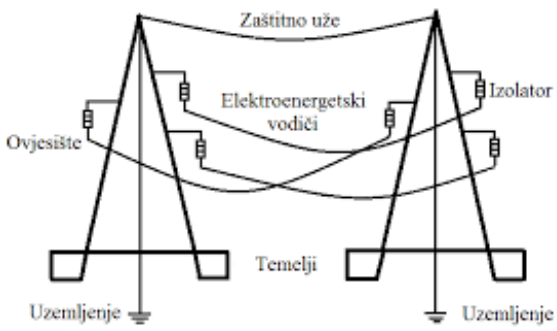
```
public void mybutton(View v) {

    if (checkInputFieldsAreEmpty()) {
       Toast.makeText(MainActivity.this, "Potrebno je ispuniti sva polja",
             Toast.LENGTH_SHORT).show();
       return;
    }
    try {
       Calculations calculations;

       calculations = new Calculations(
             Double.parseDouble(presjekVodicaUnos.getText().toString()),
             Double.parseDouble(promjerVodicaUnos.getText().toString()),
             Double.parseDouble(masaVodicaUnos.getText().toString()),
             Double.parseDouble(koefToplinskogIstezanjaUnos.getText().toString()),
             Double.parseDouble(faktorNormDodatnogTeretaUnos.getText().toString()),
             Double.parseDouble(moduleElasticnostiUnos.getText().toString()),
             Double.parseDouble(normalnoDozvoljenoNaprezanjeUnos.getText().toString()),
             Double.parseDouble(iznimnoDozvoljenoNaprezanjeUnos.getText().toString()),
             Double.parseDouble(maksimalnoNaprezanjeUnos.getText().toString()),
             Double.parseDouble(rasponStupovaUnos.getText().toString()),
             Double.parseDouble(denivelacijaUnos.getText().toString())
       );
       if(saveCalculations(calculations.getJsonResult())){
          Intent intent = new Intent(this, Results.class);
          startActivity(intent);
       }else{
          Toast.makeText(MainActivity.this, "Nesto nije u redu, probaj ponovno!",
                Toast.LENGTH_SHORT).show();
       }
```

4.3. Formule za izračun napisane u Java programskom jeziku

Nakon definiranja korisničkog sučelja napisanog u XML , povezujemo ga s glavnim kodom aplikacije. Varijable koje su poprimile vrijednost unešenih parametara uvrštuju se u formule koje su opisane u drugom poglavlju. Na Algoritmu napisanom ispod teksta vidi se kako formule od (2-2-1) do (2-2-11) izgledaju napisane u programskom jeziku Java.

```
double directDistance = Math.sqrt(Math.pow(denivelacija, 2) + Math.pow(rasponStupova,
2));
double reducedWeight = masa * 9.81 / presjek;

      double reducedConductorWeight =
            reducedWeight + (faktorDodatnogTereta * 0.18 * Math.sqrt(promjer) * 9.81) /
presjek;

      double idealSpan = Math.sqrt((Math.pow(rasponStupova, 3) / (Math.pow(directDistance,
2) / rasponStupova))) * ((Math.pow(directDistance, 3) / Math.pow(rasponStupova, 2))
            / (Math.pow(directDistance, 2) / rasponStupova));

      double strain = maksimalnoNaprezanje * ((Math.pow(directDistance, 3) /
Math.pow(rasponStupova, 2)) /
            (Math.pow(directDistance, 2) / rasponStupova));


      double criticalSpan = maksimalnoNaprezanje * Math.sqrt((360 * koefIstezanja)
            / (Math.pow(reducedConductorWeight, 2) - Math.pow(reducedWeight, 2)));
```

Slijedeći korak u računanju je usporedba kritičkog i idealnog raspona, kako bi sama aplikacija imali uvid o tome koje su osnove vrijednosti potrebne za izračun jednadžbe stanja, te prema tim paramaterima radi izračun za svaku temperaturu. Pošto varijable u formulama ovise o početnom stanju i temperaturi imati ćemo dva slučaja . Prvi slučaj je vidljiv na algoritmu napisanom ispod teksta kada je idealni raspon veći od kritičog raspona.



Slika 4.4. Prikaz prijenosa Električne energije [9]

```
if (idealSpan > criticalSpan) {
        double temperature = -5.0;
        double secCo1 = -(strain - (((Math.pow(idealSpan, 2) / 24) *
              (Math.pow(reducedConductorWeight, 2)
                    / Math.pow(strain, 2)) - (koefIstezanja * (temperature - (-20)))) *
              modulElasticnosti));
        double secCo2 = -(strain - (((Math.pow(idealSpan, 2) / 24) *
              (Math.pow(reducedConductorWeight, 2)
                    / Math.pow(strain, 2)) - (koefIstezanja * (temperature - (-5)))) *
modulElasticnosti));
        double secCo3 = -(strain - (((Math.pow(idealSpan, 2) / 24) *
              (Math.pow(reducedConductorWeight, 2)
                    / Math.pow(strain, 2)) - (koefIstezanja * (temperature - 40))) *
modulElasticnosti));
        double thirdCo1 = -((Math.pow(idealSpan, 2) * Math.pow(reducedWeight, 2)) / 24) *
              modulElasticnosti;
        double thirdCo2 = -((Math.pow(idealSpan, 2) * Math.pow(reducedConductorWeight,
2)) / 24) * modulElasticnosti;
        double thirdCo3 = -((Math.pow(idealSpan, 2) * Math.pow(reducedWeight, 2)) / 24) *
              modulElasticnosti;
        Cubic cub = new Cubic();
        cub.solve(1.0, secCo1, 0.0, thirdCo1);
        double strainRaw1 = cub.x1;
        Cubic cub2 = new Cubic();
        cub2.solve(1.0, secCo2, 0.0, thirdCo2);
        double strainRaw2 = cub2.x1;
        Cubic cub3 = new Cubic();
        cub3.solve(1.0, secCo3, 0.0, thirdCo3);
        double strainRaw3 = cub3.x1;
        double strain20 = strainRaw1 * ((Math.pow(directDistance, 2) / idealSpan) /
              ((Math.pow(directDistance, 3) / Math.pow(idealSpan, 2))));
        provjes20 = ((Math.pow(idealSpan, 2) * reducedWeight) / (8 * strain20)) *
              (directDistance / idealSpan);
        double strain5 = strainRaw2 * ((Math.pow(directDistance, 2) / idealSpan) /
              ((Math.pow(directDistance, 3) / Math.pow(idealSpan, 2))));
        provjes5 = ((Math.pow(idealSpan, 2) * reducedConductorWeight) / (8 * strain5)) *
              (directDistance / idealSpan);
        double strain40 = strainRaw3 * ((Math.pow(directDistance, 2) / idealSpan) /
              ((Math.pow(directDistance, 3) / Math.pow(idealSpan, 2))));
        provjes40 = ((Math.pow(idealSpan, 2) * reducedWeight) / (8 * strain40)) *
              (directDistance / idealSpan);
                Log.d("provjes40", String.valueOf((provjes40)));
    }
```

A drugi slučaj je vidljiv na slijedećem algoritmu koji je napisan i to je slučaj kada je idealni raspon manji nego kritični raspon.

```
else {
        double temperature = -20.0;
        double secCo1 = -(strain - (((Math.pow(idealSpan, 2) / 24) *
            (Math.pow(reducedConductorWeight, 2) / Math.pow(strain, 2)) - (koefIstezanja *
(temperature - (-
            20)))) * modulElasticnosti));
        double secCo2 = -(strain - (((Math.pow(idealSpan, 2) / 24) *
            (Math.pow(reducedConductorWeight, 2) / Math.pow(strain, 2)) - (koefIstezanja *
(temperature - (-
            5)))) * modulElasticnosti));
        double secCo3 = -(strain - (((Math.pow(idealSpan, 2) / 24) *
            (Math.pow(reducedConductorWeight, 2) / Math.pow(strain, 2)) - (koefIstezanja *
(temperature -
            40))) * modulElasticnosti));
        double thirdCo1 = -((Math.pow(idealSpan, 2) * Math.pow(reducedWeight, 2)) / 24) *
            modulElasticnosti;
        double thirdCo2 = -((Math.pow(idealSpan, 2) * Math.pow(reducedConductorWeight,
2)) /
            24) * modulElasticnosti;
        double thirdCo3 = -((Math.pow(idealSpan, 2) * Math.pow(reducedWeight, 2)) / 24) *
            modulElasticnosti;
        Cubic cub = new Cubic();
        cub.solve(1.0, secCo1, 0.0, thirdCo1);
        double strainRaw1 = cub.x1;
        Cubic cub2 = new Cubic();
        cub2.solve(1.0, secCo2, 0.0, thirdCo2);
        double strainRaw2 = cub2.x1;
        Cubic cub3 = new Cubic();
        cub3.solve(1.0, secCo3, 0.0, thirdCo3);
        double strainRaw3 = cub3.x1;

        double strain20 = strainRaw1 * ((Math.pow(directDistance, 2) / idealSpan) /
            ((Math.pow(directDistance, 3) / Math.pow(idealSpan, 2))));
        provjes20 = ((Math.pow(idealSpan, 2) * reducedWeight) / (8 * strain20)) *
            (directDistance / idealSpan);
        double strain5 = strainRaw2 * ((Math.pow(directDistance, 2) / idealSpan) /
            ((Math.pow(directDistance, 3) / Math.pow(idealSpan, 2))));
        provjes5 = ((Math.pow(idealSpan, 2) * reducedConductorWeight) / (8 * strain5)) *
            (directDistance / idealSpan);
        double strain40 = strainRaw3 * ((Math.pow(directDistance, 2) / idealSpan) /
            ((Math.pow(directDistance, 3) / Math.pow(idealSpan, 2))));
        provjes40 = ((Math.pow(idealSpan, 2) * reducedWeight) / (8 * strain40)) *
            (directDistance / idealSpan);

    }
```

Kao što se vidi na prijašnja dva algoritma koristili smo class-u *Cubic* koja omogućuje izračun kubnih jednadžbi i olakšava sami proces računanja u aplikaciji. Također na tim algoritmima

može se primjetiti da klasa *Cubic* ima metodu *„solve()“* koja prima koeficijente kubne jednažbe. Na algoritmu ispod teksta vidi se dio koda klase *Cubic*.

```
public class Cubic {
   private static final double TWO_PI = 2.0 * Math.PI;
   private static final double FOUR_PI = 4.0 * Math.PI;

   public int nRoots;
   public double x1;
   public double x2;
   public double x3;


   public Cubic()
   {
   }


   public void solve
         (double a,
          double b,
          double c,
          double d)
   {
      double denom = a;
      a = b/denom;
      b = c/denom;
      c = d/denom;


      double a_over_3 = a / 3.0;
      double Q = (3*b - a*a) / 9.0;
      double Q_CUBE = Q*Q*Q;
      double R = (9*a*b - 27*c - 2*a*a*a) / 54.0;
      double R_SQR = R*R;
      double D = Q_CUBE + R_SQR;
```

Nakon dodjeljivanje vrijednosti varijablama, određivanja odnosa kritičnog i idealnog raspona zadnji korak izračuna odnosi se na sami izračun provjesa neizoliranog energetskog vodiča, prema formuli (2-10) i (2-11) ovisno o tome dali postoji razlika u visini stupova tzv. denivelacija. Na slijedećem algoritmu ispod teksta vidi se krajnja formula izračuna provjesa pri našim kritičnim temperaturama.

```
double strain20 = strainRaw1 * ((Math.pow(directDistance, 2) / idealSpan) /
        ((Math.pow(directDistance, 3) / Math.pow(idealSpan, 2))));
    provjes20 = ((Math.pow(idealSpan, 2) * reducedWeight) / (8 * strain20)) *
        (directDistance / idealSpan);
    double strain5 = strainRaw2 * ((Math.pow(directDistance, 2) / idealSpan) /
        ((Math.pow(directDistance, 3) / Math.pow(idealSpan, 2))));
    provjes5 = ((Math.pow(idealSpan, 2) * reducedConductorWeight) / (8 * strain5)) *
        (directDistance / idealSpan);
    double strain40 = strainRaw3 * ((Math.pow(directDistance, 2) / idealSpan) /
        ((Math.pow(directDistance, 3) / Math.pow(idealSpan, 2))));
    provjes40 = ((Math.pow(idealSpan, 2) * reducedWeight) / (8 * strain40)) *
        (directDistance / idealSpan);
```

5. IZRAČUN RUČNOM METODOM I PUTEM APLIKACIJE

U petom poglavlju prikazan je izračun putem aplikacije i ručni izračun koristeći formule objašnjenje u drugom poglavlju. Prema [3] tablici 1. navedeni su sve vrijednosti i podaci o neizoliranom vodiču koji će nam biti potreban za izračun provjesa.

Tablica 1. vrijednosti karakteristika Al/Fe 265/35 [11]

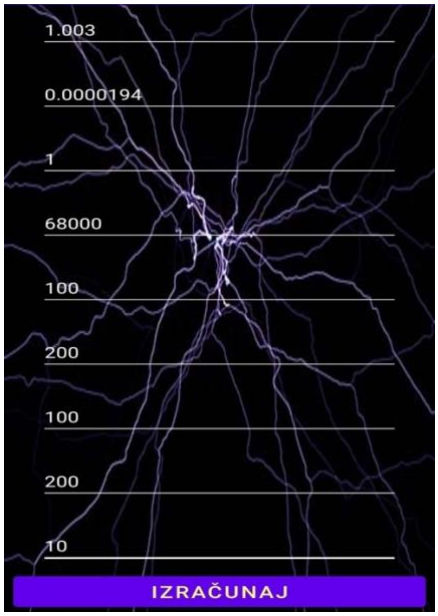
Naziv vodiča	Al/Fe 265/35
Koeficijent dodatnog tereta - k	1
Iznimno dozvoljeno naprezanje - σi [N/mm2]	200
Modul elastičnosti - E [N/mm2]	68000
Uzdužna masa - m [kg/m]	1,003
Računski presjek - A [mm2]	297,8
Promjer - d [mm]	22,4
Koeficijent linearnog toplinskog rastezanja - β [1/°C]	$19{,}4 \cdot 10{-}6$
Normalno dozvoljeno naprezanje - σd [N/mm2]	100
Maksimalno dozvoljeno naprezanje – σmax [N/mm2]	100

Osim podataka o neizoliranom vodiču, potrebne su vrijednosti denivelacije(h12) i raspon (a)stupova što će u našem slučaju iznositi:

- h12- 10 m
- a- 200m

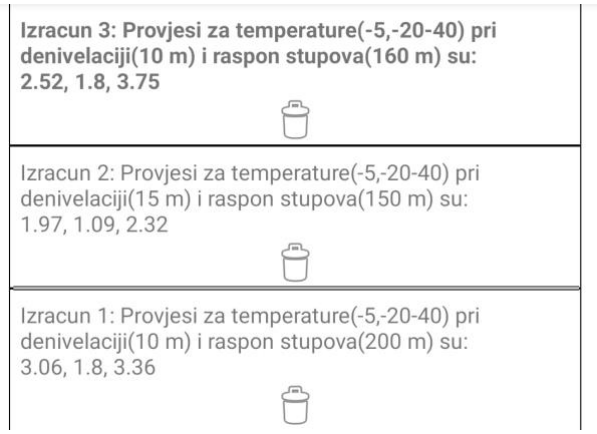
5.1. Računanje putem aplikacije

Kako bi se odredila valjanost aplikacije moraju se unijeti vrijednosti koje se koriste prilikom ručnog izračuna provjesa neizoliranog energetskog vodiča. Vrijednosti koje se unose su vrijednosti upisane u Tablica 1. Na slici 7. vidi se unos vrijednosti.



Slika 5.1. Unos vrijednosti

Nakon pritiska gumb "Izračunaj" aplikacija unešene vrijednosti unese u algoritam formula, zatim se izračuna vrijednost provjesa te u novom prozorčiću izbaci rezultate izračuna što je vidljivo na slici 8.



Slika 5.2. Rezultati izračuna provjesa

5.2.Ručni izračun provjesa neizoliranih energetskih vodova

Prvi korak u ručnom računanju je izračun za vlastitu težinu neizoliranog vodiča:

$$G_0 = m1 * g = 1{,}003 * 9{,}81 = 9{,}83943 \ \frac{N}{M}$$

Nakon izračuna težine neizoliranog vodiča potrebno je naći njegovu reduciranu težinu po formuli:

$$g_0 = \frac{G_0}{A} = \frac{9{,}83943}{297{,}8} = 0{,}03304 \ \frac{N}{m * mm^2}$$

Zatim se računa dodatno opterećenje:

$$G_{lo} = 0.18 * \sqrt{d} * g = 0.18 * \sqrt{22{,}4} * 9{,}81 = 8{,}3573 \ \frac{N}{m}$$

Slijedeći korak je određivanje stvarnog dodatnog opterećenja:

$$G_l = k * G_{lo} = 1 * 8{,}3573 = \frac{N}{m}$$

Nakon što se izračuna stvarno dodatno opterećenje i vlastita težina vodiča, računa se reducirana težina zaleđenog vodiča prema formuli:

$$g_z = \frac{G_0 + G_l}{A} = \frac{8{,}3573 + 9{,}83943}{297{,}8} = 0{,}061104 \ \frac{N}{m * mm^2}$$

Nakon izračuna reducirane težine zaleđenog vodiča, slijedeći korak u računanju provjesa je računanje kritičkog raspona i idealnog raspona kako bi se znalo osnovno stanje neizoliranog vodiča .

Postoji samo jedan raspon koji nam je zadan , pa se taj raspon uzima u jednadžbi kao idealni raspon, pa po tome vrijedi:

$$a_{idealno} = a = 200m$$

Formula za računanje kritičkog raspon je:

$$a_{k=\sigma_{max}} * \sqrt{\frac{360*\beta}{gz^2 - g0^2}} = 100 * \sqrt{\frac{360*19,4*10-6}{0,061104^2 - 0,03304^2}} = 162,585 \text{ m}$$

Nakon što se izračuna kritični raspon i odredi idealni raspon, određuje se uvjet za osnovno stanje jednadžbe:

$$a_{idealno} > a_k$$

Iz uvjeta vidi se da je idealni raspon veći nego kritični raspon, zbog toga početno stanje je:

- $\sigma 1 = \sigma max = 100 \frac{N}{m*mm^2}$
- $g1 = g_z = 0,061104 \frac{N}{m*mm^2}$
- $\theta_1 = -5° \text{ C}$

Pošto u zadanim vrijednostima je postavljena denivelacija koja iznosi 10m, mora se izračunati nadomjesno naprezanje:

$$\bar{\sigma 1} = \sigma 1 * \frac{\sum_{i=1}^{n} \frac{ai'^3}{ai^2}}{\sum_{i=1}^{n} \frac{ai'^2}{ai}} = 100 * \frac{\frac{200,25^3}{200^2}}{\frac{200,25^2}{200}} = 100,125 \frac{N}{mm^2}$$

Slijedeći korak je određivanje stvarnog nadomjesnog naprezanja za svaku temperaturu:

1) $\theta 2$ = -5° C

$$g_2 = g_z = 0{,}061104\,\frac{N}{m*mm^2}$$

Nadomjesno naprezanje pri temperaturi -5° C dobiti ćemo formulom:

$$\frac{\bar{\sigma}1-\bar{\sigma}2}{E} + \beta(\theta 1 - \theta 2) = \frac{aidealno^2}{24} * \left(\frac{g1^2}{\bar{\sigma}1^2} - \frac{g2^2}{\bar{\sigma}2^2}\right)\,\frac{N}{mm^2}$$

$$\frac{100{,}125-\bar{\sigma}2}{68000} + 19{,}4 * 10^{-6}\,(-5-(-5)) = \frac{200^2}{24} * \left(\frac{0{,}061104^2}{100.125^2} - \frac{0{,}061104^2}{\bar{\sigma}2^2}\right)$$

$$\frac{100{,}125-\bar{\sigma}2}{68000} = 6{,}207303 * 10^{-4}\ - \frac{6{,}22283}{\bar{\sigma}2^2}$$

$$\bar{\sigma}2^3 + 57{,}9154\bar{\sigma}2^2 + 423154{,}44$$

Nadomjesno naprezanje: $\bar{\sigma}2 = 100{,}1250293\,\frac{N}{mm^2}$

Stvarno nadomjesno naprezanje: $\sigma 2 = \bar{\sigma}2 * \frac{\frac{200{,}25^2}{200}}{\frac{200{,}25^3}{200^2}} = 100.2501856\,\frac{N}{mm^2}$

2) $\theta 2$ = -20° C

$$g_2 = g_o = 0{,}03304\,\frac{N}{m*mm^2}$$

Nadomjesno naprezanje pri temperaturi -20° C dobiti ćemo formulom:

$$\frac{\bar{\sigma}1-\bar{\sigma}2}{E} + \beta(\theta 1 - \theta 2) = \frac{aidealno^2}{24} * \left(\frac{g1^2}{\bar{\sigma}1^2} - \frac{g2^2}{\bar{\sigma}2^2}\right)$$

$$\frac{100{,}125-\bar{\sigma}2}{68000} + 19{,}4 * 10^{-6}\,(-5-(-20)) = \frac{200^2}{24} * \left(\frac{0{,}061104^2}{100.125^2} - \frac{0{,}03304^2}{\bar{\sigma}2^2}\right)$$

$$\frac{100{,}125-\bar{\sigma}2}{68000} + 2.91 * 10^{-4} = 6{,}207303 * 10^{-4}\ - \frac{1{,}8194}{\bar{\sigma}2^2}$$

$$\bar{\sigma}2^3 + 77{,}7033396\,\bar{\sigma}2^2 + 123719{,}2$$

Nadomjesno naprezanje: $\bar{\sigma}2 = 92{,}24340733\,\frac{N}{mm^2}$

Stvarno nadomjesno naprezanje: $\sigma 2 = \bar{\sigma}2 * \frac{\frac{200{,}25^2}{200}}{\frac{200{,}25^3}{200^2}} = 92{,}1282\,\frac{N}{mm^2}$

1) $\theta 2$ = 40° C

$$g_2 = g_o = 0{,}03304\,\frac{N}{m*mm^2}$$

Nadomjesno naprezanje pri temperaturi -20° C dobiva se formulom:

$$\frac{\bar{\sigma}1-\bar{\sigma}2}{E} + \beta(\theta 1 - \theta 2) = \frac{aidealno^2}{24} * \left(\frac{g1^2}{\bar{\sigma}1^2} - \frac{g2^2}{\bar{\sigma}2^2}\right)$$

$$\frac{100{,}125-\bar{\sigma}2}{68000} + 19{,}4 * 10^{-6}\,(-5-(40)) = \frac{200^2}{24} * \left(\frac{0{,}061104^2}{100.125^2} - \frac{0{,}03304^2}{\bar{\sigma}2^2}\right)$$

$$\frac{100{,}125-\bar{\sigma}2}{68000} - 8{,}73 * 10^{-4} = 6{,}207303 * 10^{-4} - \frac{1{,}8194}{\bar{\sigma}2^2}$$

$$\bar{\sigma}2^3 - 1{,}4486\,\bar{\sigma}2^2 + 123719{,}38$$

Nadomjesno naprezanje: $\bar{\sigma}2 = 49{,}35045\,\frac{N}{mm^2}$

Stvarno nadomjesno naprezanje: $\sigma 2 = \bar{\sigma}2 * \frac{\frac{200{,}25^2}{200}}{\frac{200{,}25^3}{200^2}} = 49{,}2883895\,\frac{N}{mm^2}$

Nakon što se izračuna stvarno nadomjesno rastezanje pri svim temperaturama , slijedeći korak nam je računanje provjesa pri svakoj temperaturi:

1) Provjes neizoliranog energetskog vodiča pri temperaturi -5° C :

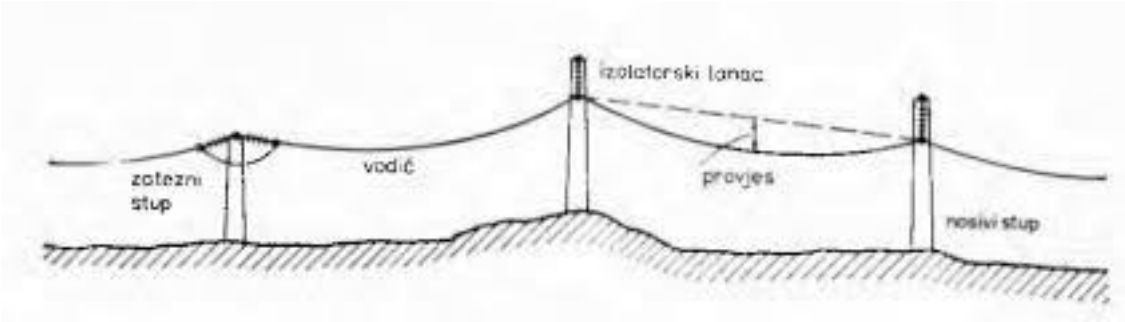
$$f' = f * \frac{a'}{a} = \frac{a^2 * g_2}{8 * \sigma_2} * \frac{a'}{a} = \frac{200^2 * 0{,}061104}{8 * 100{,}1501856} * \frac{200{,}25}{200} = 3{,}05138m$$

2) Provjes neizoliranog energetskog vodiča pri temperaturi -20° C :

$$f' = f * \frac{a'}{a} = \frac{a^2 * g_2}{8 * \sigma_2} * \frac{a'}{a} = \frac{200^2 * 0{,}03304}{8 * 92{,}1282} * \frac{200{,}25}{200} = 1{,}795396m$$

3) Provjes neizoliranog energetskog vodiča pri temperaturi 40° C :

$$f' = f * \frac{a'}{a} = \frac{a^2 * g_2}{8 * \sigma_2} * \frac{a'}{a} = \frac{200^2 * 0{,}03304}{8 * 49{,}28883895} * \frac{200{,}25}{200} = 3{,}3558611\ m$$



Slika 5.3. Elektroenergetski Sistem [10]

5.3 Usporedba rezultata

Nakon svih izračuna putem aplikacije i ručnim putem, može se vidjeti u tablici 2. da se su dobile približne vrijednosti zbog različitog zaokruživanja vrijednosti pri izračunu putem aplikacije i ručnim putem.

Tablica 2. Usporedba rezultata

Kritične Temperature	Ručni Izračun	Aplikacija
40° C	3,356 m	3,356m
-5° C	3,06 m	3,06m
-20° C	1,795m	1,795m

6. Zaključak

Neizolirani nadzemni vodiči su jakobitni za ljudsku svakodnevnicu i važan su dio u elektroenergetskom sustavu. Glavni zadatak ovoga rada bilo je određivanje provjesa neizoliranih energetskih vodova. Važnost u izračunu provjesa neizoliranih energetskih vodova je što olakšava projektiranje i postavljanje električnih nadzemnih vodova. Kako bi ubrzali taj proces izrađena je aplikacija koja pomaže pri tome i štedi nam vrijeme, umjesto ručnog izračuna napravljen je algoritam kojem su potrebne vrijednosti koje ona uvrštava u formule i onda izračuna provjes neizoliranih energetskih vodiča. Kao što je bilo predstavljeno u prošlim poglavljima vidimo da je aplikacija jako jednostavna i lagana za korištenje, što omogućuje korisniku lakše snalaženje u aplikaciji. Aplikacija je razvijana u objektno-orijentiranom programskom jeziku Java u razvojnom okruženju Android studio. Usporedbom rezultata dobivenih računanjem s pomoću aplikacije i ručnom metodom primjetili smo da aplikacija daje iste rezultate kao i ručna metoda računanja.

LITERATURA

[1] Tehnička enciklopedija, Leksikografski zavod Miroslav Krleža, dostupno na: https://tehnika.lzmk.hr/tehnickaenciklopedija/dalekovodi.pdf , 13.9.2019

[2] Brkić, N.: Mehanički proračun vodiča, Sveučilište u Rijeci, Tehnički fakultet, Rijeka. 2015.

[3] „Aluminijsko-čelično uže za nadzemni vod“, http://www.tim-kabel.hr/content/view/273/342/lang,hrvatski/

[4]“Java(programski jezik)“, https://en.wikipedia.org/wiki/Java_(programming_language)

[5] „Android(operacijski sustav)“, https://hr.wikipedia.org/wiki/Android_(operacijski_sustav)

[6] Razvojno okruženje „Android Studio“, https://commons.wikimedia.org/wiki/File:Android_Studio_4.1_screenshot.png

[7] Primjer koda iz „Java“, https://www.researchgate.net/figure/Source-Code-Example-Customerjava-Partial_fig2_317401664

[8] Antonio Plasaj(2016), https://repozitorij.vuka.hr/islandora/object/vuka%3A538/datastream/PDF/view

[9] Prijenos i razdjela Električne energije, https://www.fer.unizg.hr/_download/repository/PRIJENOS_I_RAZDJELA_ELEKTRICNE_ENERGIJE.pdf

[10] „Elektroenergetski sistem“, https://www.google.com/search?q=provjes+&tbm=isch&ved=2ahUKEwj_9fTcmfTyAhUNMRoKHSCHDiMQ2-cCegQIABAA&oq=provjes+&gs_lcp=CgNpbWcQAzIHCCMQ7wMQJzIHCCMQ7wMQJzoFCAAQgAQ6BAgAEB46CggjEO8DEOoCECdQhU1YuXNg_3hoCXAAeACAAVmIAfYEkgEBOJgBAKABAaoBC2d3cy13aXotaW1nsAEKwAEB&sclient=img&ei=1DM7Yf-7Ho3iaKCOupgC&bih=911&biw=1920#imgrc=IjoCR0XMTDcohM

[11] vrijednosti karakteristika Al/Fe 265/35, http://www.tim-kabel.hr/images/stories/katalog/datasheetHRV/0602_ACSR.pdf

[12]Bruno Kapular (2018), https://core.ac.uk/download/pdf/197872145.pdf?fbclid=IwAR1Un6bl7HvdTS9W1DjlmV9dQE9CWnoGUu5gtqpcmjEsFPhnoc2VmWz0GSE

SAŽETAK

Cilj završnog rada je razvijanje Android aplikacije za računanje provjesa neizoliranih energetskih vodova. U ovome radu su spomenute formule koje su korištene tijekom samog procesa, te tehnologije koje su se koristile pri samoj izradi aplikacije. Najvažniji cilj u razvijanju aplikacije je taj da smo htjeli doći do toga da aplikacija ubrzava i olakšava sami ručni izračun provjesa neizoliranih energetskih vodova kroz algoritme koje smo napisali u samoj aplikaciji u objektno-orijentiranom programskom jeziku Java. Aplikacija je razvijana u Android Studio. Rezultati aplikacije pokazali su se kao dobar pokazatelj, te nema velikih odstupanja od ručnog izračuna.

Ključne riječi: Provjes neizoliranih energetskih vodiča, *Android*, *Java*, aplikacija

ABSTRACT

The aim of the final work is to develop an Android application for calculating the sagging of uninsulated power lines. This paper mentions the formulas that were used during the process, and the technologies that were used in the development of the application. The most important goal in developing the application is to make the application speed up and facilitate the manual calculation of uninsulated power lines through algorithms that we wrote in the application in the object-oriented Java programming language. The application was developed in Android Studio. The results of the application proved to be a good indicator, and there are no major deviations from the manual calculation.

Keywords: Android,Java,application,uninsulated conductors sag

ŽIVOTOPIS

Matko Mihalj rođen je u Đakovu 21.veljače.2000.godine. Pohađao je osnovnu školu „Josipa Antuna Ćolnića“ u Đakovu. Završetkom osnovne škole, upisuje gimnaziju „Antun Gustav Matoš“ u Đakovu, smjer Matematičko-prirodoslovni. Srednju školu završava 2018.godine te trenutno pohađa preddiplomski smjer računarstva na „Fakultetu elektrotehnike, računarstva i informacijskih tehnologija“ u Osijeku.

PRILOZI

1.CD/DVD medij sljedećeg sadržaja:

-Završni rad u formatu .pdf

-Projekt izrađene mobilne aplikacije