LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN
PENERAPAN *GRAPHICAL USER INTERFACE (GUI)* PADA APLIKASI KALKULATOR

Disusun Oleh:

Haliya Isma Husna Putri Ahmadi
2511532002

Dosen Pengampu:
Wahyudi. Dr. S.T.M.T

Asisten Praktikum:
Muhammad Zaky Al Hafiz

DEPARTEMEN INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

2025

KATA PENGANTAR

Puji dan Syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa karena atas Rahmat dan karunia-Nya, penulis dapat menyelesaikan lapora praktikum Algoritma dan Pemrograman pekan 9 dengan judul "*Penerapan Graphical User Interface (GUI)* pada Aplikasi Kalkulator" dalam bahasa *Java* tepat pada waktunya.

Penyusunan laporan ini bertujuan untuk memenuhi salah satu tugas dalam mata kuliah Praktikum Algoritma dan Pemrograman, sekaligus sebagai sarana pembelajaran bagi penulis dalam memahami pembuatan aplikasi menggunakan *Graphical User Interface (GUI)* pada *Java*. Pada praktikum ini, penulis mempelajari cara mengimplementasikan komponen–komponen *GUI* seperti. *JFrame, Jbutton*, dan *JTextField*. Serta memanfaatkan *event handling* melalui *ActionListener* untuk mengelola *input button* dan menjalankan operasi aritmatika dalam proses perhitungan kalkulator.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, penulis sangat terbuka terhadap kritik dan saran yang membangun demi perbaikan di masa yang akan datang.

Ucapan terima kasih yang sebesar-besarnya penulis sampaikan kepada dosen pengampu, asisten praktikum, serta semua pihak yang terlibat dalam pelaksanaan praktikum dan penyusunan laporan ini. Semoga laporan ini dapat memberikan manfaat dan menambah wawasan bagi penulis maupun pembaca.

Padang, 24 November 2025

Penulis

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dalam perkembangan teknologi perangkat lunak, kebutuhan akan aplikasi yang interaktif dan mudah digunakan semakin meningkat. Salah satu cara untuk memenuhi kebutuhan tersebut adalah dengan menerapkan *Graphical User Interface (GUI)* pada aplikasi. Berbeda dengan program berbasis teks (*console*), *GUI* memungkinkan pengguna berinteraksi melalui elemen visual seperti tombol, kotak teks, jendela, dan menu, sehingga pengalaman penggunaan menjadi lebih intuitif dan efisien.

Melalui praktikum ini, mahasiswa diharapkan dapat memahami cara merancang dan mengimplementasikan *GUI* menggunakan *Java Swing*, mulai dari pembuatan jendela utama (*JFrame*), penambahan komponen seperti *JButton*, *JLabel*, dan *JTextField*, hingga pemanfaatan *event handling* untuk mengatur interaksi antara pengguna dan program.

1.2 Tujuan

Tujuan dari pelaksanaan praktikum ini adalah sebagai berikut:

1.2.1 Memahami konsep dasar *Graphical User Interface (GUI*) dalam pemrograman *Java*.

1.2.2 Mampu menggunakan komponen-komponen *GUI* seperti *JFrame, JButton, JLabel, JTextField*, dan lainnya

1.2.3 Menguasai penggunaan *WindowBuilder* dalam merancang tampilan *GUI* secara efisien.

1.3 Manfaat

Manfaat yang diharapkan dari pelaksanaan praktikum ini antara lain:

1.3.1 Memberikan pemahaman mengenai pembuatan antarmuka grafis menggunakan *Java Swing*.

1.3.2 Memberikan pengalaman langsung dalam merancang dan mengimplementasikan komponen *GUI* pada aplikasi atau program *Java* .

BAB II
PEMBAHASAN

2.1 Dasar Teori

Dalam bahasa pemrograman *Java*, *Graphical User Interface (GUI)* merupakan antarmuka yang memungkinkan *user* berinteraksi dengan aplikasi/ program melalui elemen visual seperti tombol, jendela, ikon, dan *text box*. Dalam pemrograman *Java*, *GUI* biasanya dibuat menggunakan *Java Swing*, yaitu *library* yang menyediakan berbagai komponen untuk menyusun tampilan aplikasi. Dengan adanya *GUI*, aplikasi menjadi lebih mudah digunakan dibandingkan aplikasi berbasis teks (*console*), karena pengguna dapat melakukan *input* dan navigasi menggunakan komponen visual.

Pada praktikum ini, mahasiswa mempelajari cara merancang dan mengimplementasikan *GUI* menggunakan *Windows Application WindowBuilder* pada *Java*, yaitu *plugin visual* editor untuk membuat tampilan aplikasi secara *drag-and-drop* di *Eclipse*. Pemahaman terhadapkomponen *GUI*, serta mekanisme *event handling* menjadi dasar penting dalam membangun aplikasi/ program *Java* yang interaktif.

2.1.1 WindowBuilder

WindowBuilder adalah sebuah *GUI designer* untuk *Java* yang tersedia sebagai *plugin* pada *IDE Eclipse*. Dengan *WindowBuilder*, dapat merancang tampilan *GUI* secara visual menggunakan metode *drag-and-drop*, tanpa harus menulis semua kode secara manual. *WindowBuilder* mendukung pengembangan antarmuka menggunakan *Swing*, *SWT*, maupun *GWT*.

Berikut beberapa fungsi dari *WindowBuilder:*

1. Sinkronisasi otomatis antara editor visual dan kode.
2. Editor visual untuk mendesain layout.
3. Mempercepat proses pembuatan antarmuka *Java*

2.1.2 Java Swing

Java Swing adalah *library GUI* yang termasuk dalam *Java Foundation Classes (JFC*). Swing digunakan untuk membuat aplikasi desktop dengan tampilan yang dapat dikustomisasi. *Library* ini menyediakan banyak komponen *GUI*, mulai dari komponen dasar seperti tombol hingga komponen kompleks seperti *tabel* dan *dialogs*.

Keunggulan *Java Swing:*

1. Bersifat *lightweight* (tidak bergantung pada sistem operasi).
2. Memiliki banyak komponen bawaan.
3. Memungkinkan pembuatan *layout* yang fleksibel.

2.1.3 Application Window

Pada praktikum pekan ini, pembuatan antarmuka dilakukan menggunakan *Application Window*, yaitu template bawaan dari *Swing Designer* (*WindowBuilder*) yang digunakan untuk membangun aplikasi *GUI* berbasis *Java Swing*. Meskipun menggunakan template yang berbeda, *Application Window* tetap berbasis pada kelas *JFrame*, sehingga konsep dan struktur kontainer utama tidak berubah.

2.1.4 Component GUI Java Swing

Komponen *GUI* adalah elemen visual yang digunakan pengguna untuk berinteraksi dengan program. Beberapa komponen dasar yang digunakan pada praktikum ini antara lain:

1. *JLabel*

   Digunakan untuk menampilkan teks atau informasi statis. Fungsi: memberi petunjuk, judul, atau deskripsi *input*.
2. *JTextField*

Komponen *input* untuk menerima masukan berupa teks dari pengguna. Biasanya digunakan untuk formulir, *input* nilai, atau data sederhana.

3. *Jbutton*

   Tombol yang dapat ditekan pengguna untuk melakukan aksi tertentu. Umumnya digunakan bersama *ActionListener*.

4. *JPanel*

   Wadah tambahan yang digunakan untuk menata komponen secara rapi berdasarkan layout tertentu. Membantu modularisasi tampilan aplikasi.

5. *JComboBox*

   Drop-down menu untuk memilih salah satu opsi dari daftar yang dibuat.

6. *JCheckBox & JRadioButton*

   Komponen input pilihan:

   i. *JCheckBox* (pilihan bisa lebih dari satu)

   ii. *JRadioButton* (pilihan tunggal dalam satu grup)

2.1.5 Event Handling

Event handling adalah mekanisme untuk menangani aksi pengguna, seperti menekan tombol, mengetik teks, atau memilih opsi. Dalam *Java Swing*, *event handling* dilakukan menggunakan *listener*, seperti:

i. *ActionListener*, menangani aksi tombol

ii. *KeyListener,* menangani *input keyboard.*

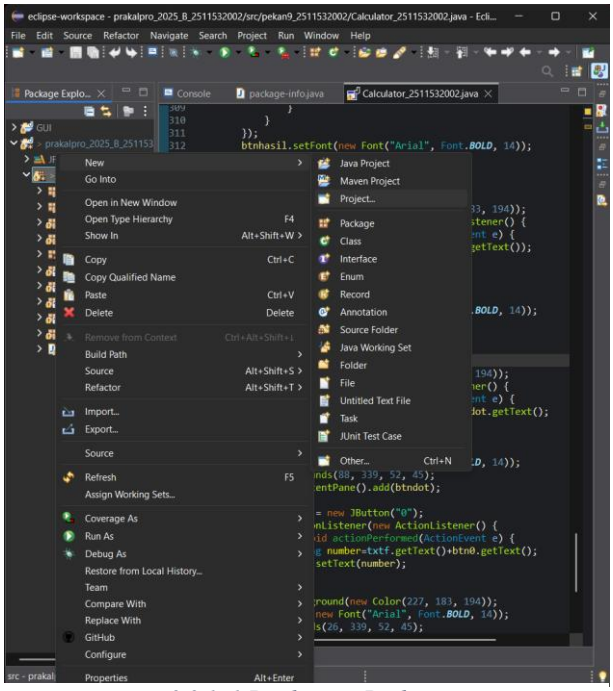
iii. *MouseListener*, menangani klik *mouse*.

Event handling memastikan aplikasi merespon interaksi pengguna dengan benar dan dinamis.

2.2 Langkah Kerja

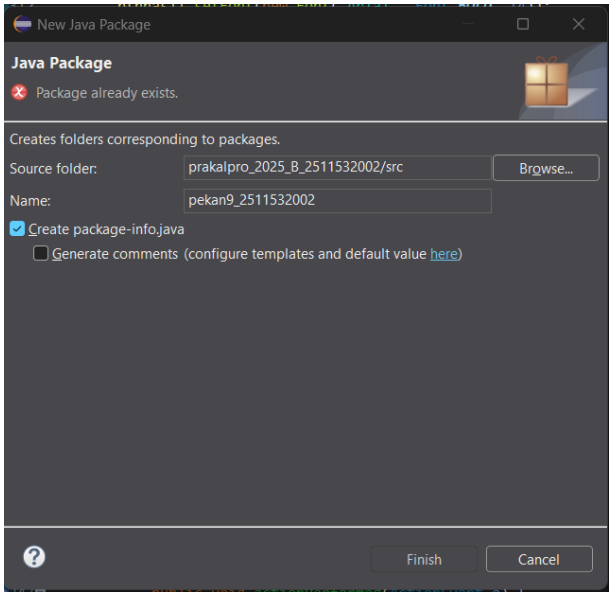
2.2.1 Pembuatan *Package* dan *Class*

1. Sebelum membuat suatu program *user* perlu untuk membuat *package* terlebih dahulu. Klik kanan pada *src* lalu pilih "*New"* dan klik "*Package"*.



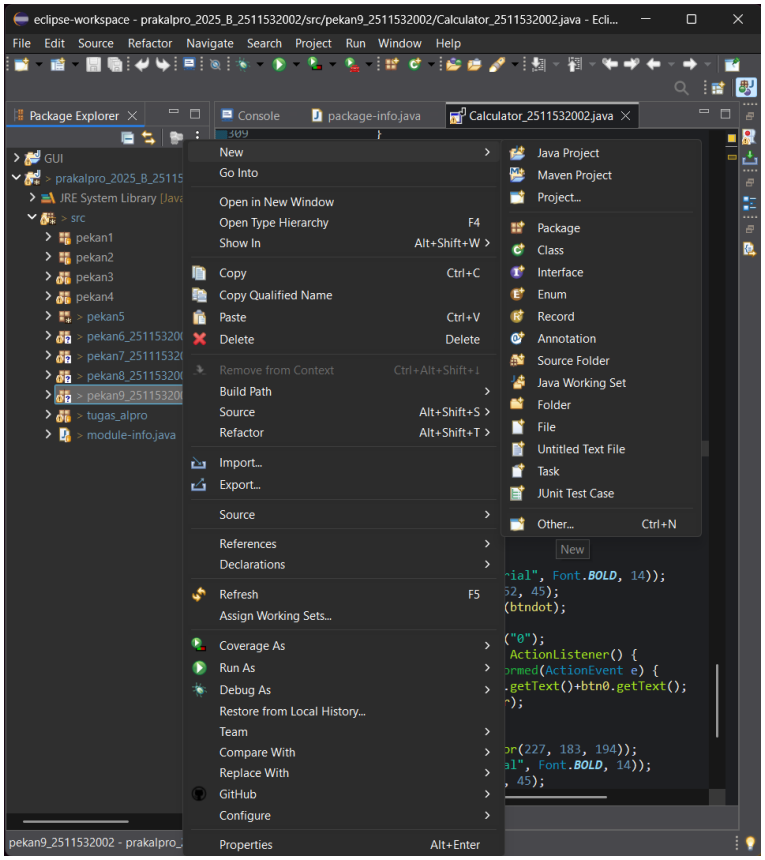
2.2.1. 1 Pembuatan Package

2. Setelah itu akan muncul *pop up* "*Java Package*", buat nama *package* dengan ketentuan tanpa *capslock, space*, dan karakter khusus lainnya. Lalu klik "*finnish*".



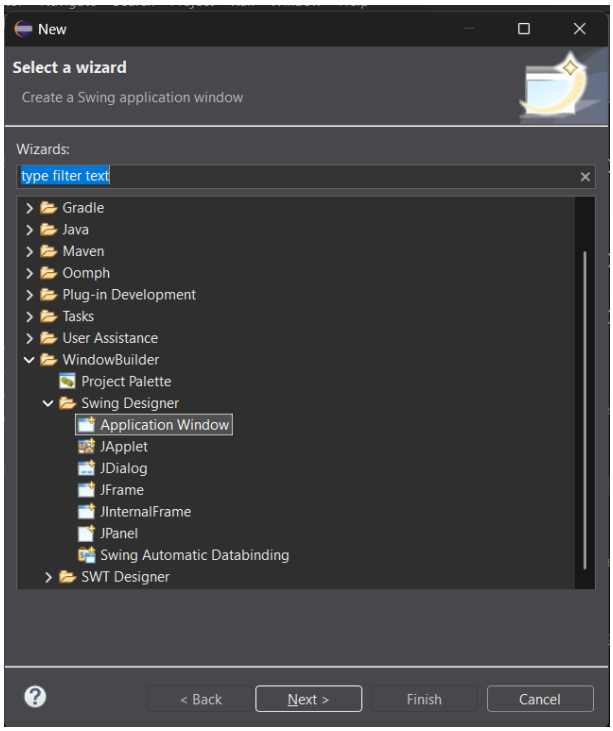
2.2.1. 2 Pembuatan Pakager Program pekan9_2511532002

3. Selajutnya adalah membuat membuat *class*. pada package yang telah dibuat tapi klik kanan lalu pilih "*New*" dan klik "*Other*".



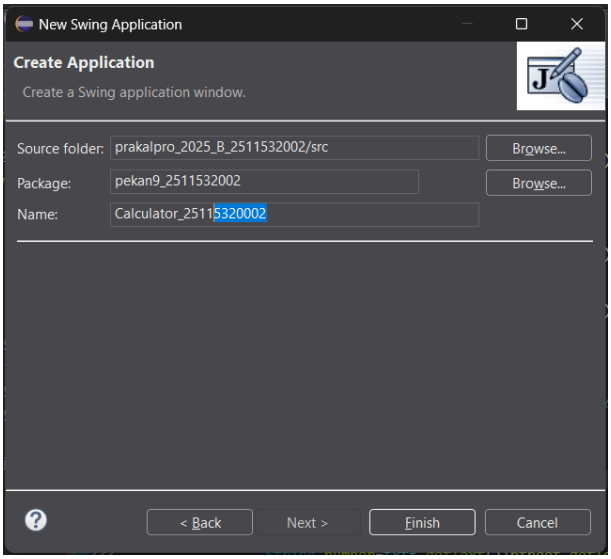
2.2.1. 3 Pembuatan Class Calculator_2511532002

4. Setelah itu akan muncul menu *pop up "Selected a Wizard"*, lalu pilih *Window Builder* dan *Swing Designer*. Pada menu *swing designer* pilih *"Application Window"*. Lalu klik "*Next"*.



2.2.1. 4 Pembuatan Class Calculator_2511532002

5. Setelah itu pada menu "*Creat Application"* buat nama *class*, dengan ketentuan nama harus *capslock* diawal kalimat dan tanpa spasi. Dan klik "*Finish"*.

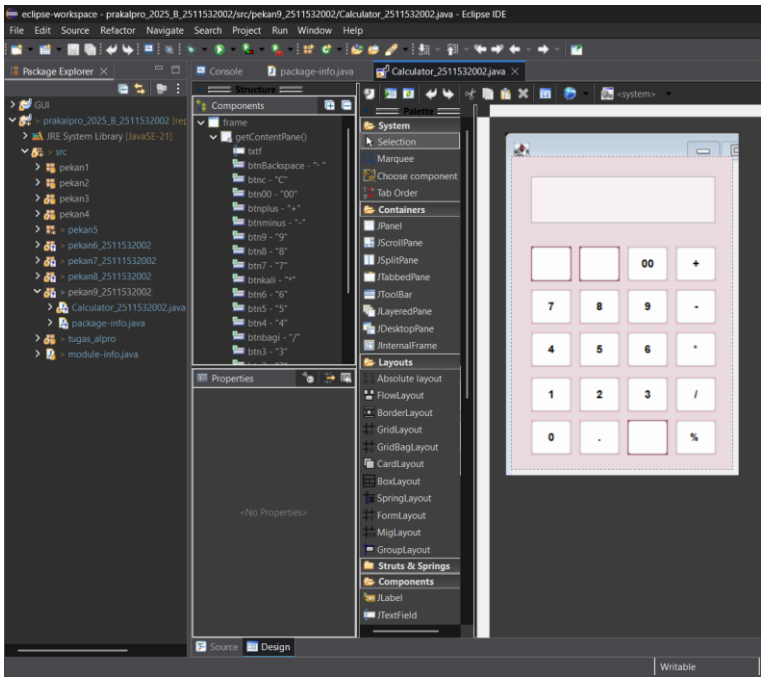


2.2.1. 5 Pembuatan class Calculator_2511532002

Maka akan kelas baru, dengan tab *"Source"* dan "Design".

2.2.2 Pembuatan Program

6. Lalu, disini *user* dapat mulai membuat desain tampilan antar muka pogram dengan *drag and drop.*

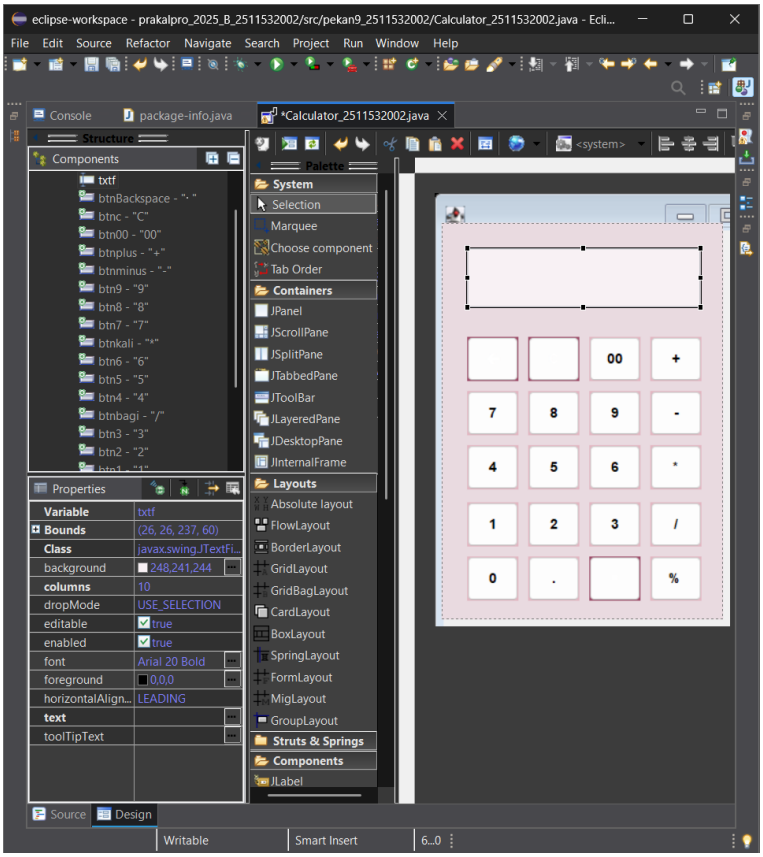


2.2.1. 6 Pembuatan Program Calculator_2511532002

Atur *size,* warna *background*, dan *layout* panel sesuai yang dingingkan. Dan pastikan *layout* dari panel dalam mode "*Absolute Layout",* agar programmer lebih mudah untuk mengatur/ mendesain tampilan.

7. Menambahkan Komponen *GUI*
    1) Setelah menyiapkan panel, *user* dapat mulai mendesain tampilan kalkulator.

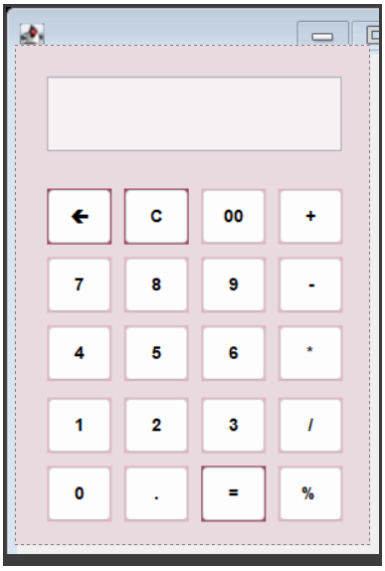
2) Disini saya memulai dengan menambahkan *"JTextFiled"* untuk kotak *input* dan *output,* caranya pada "*Palette*", pilih *component "JTextFiled" drag and drop* ke panel. Maka akan muncul *text box.*



2.2.1. 7 Menambahkan Component JTextField

Ganti nama *component* atau *variabel,* disini saya Ganti dengan "txtf" agar mudah dan mempercepat dalam menulis program nantinya. Lalu atur size, *font, allignment,* dan lainnya dari kolom tersebut.

3) Selanjutnya menambahkan tombol *button* menggunakan *Jbutton* untuk nomor, *backspace*, *C (clear*), dan hasil. Sehingga menghasilkan tampilan seperti berikut.



2.2.1. 8 Desain Tampilan Calculator

User dapat mengatur property dari tiap komponen, seperti memilih *font, alignment,* warna dan lainnya.

8. Menambahkan Kode Program

Selanjutnya agar kalkulator dapat dioperasikan, *user* perlu meambahkan beberapa *syntax* tambahan untuk *button* dan . Deklarasi variabel pada bagian atas *class*, agar komponen dapat diakses oleh seluruh *method* dalam *class*.

Berikut adalah pejelasan lebih lanjut untuk kode program:

```
public class Calculator_2511532002 {

    private JFrame frame;
    private JTextField txtf;

    double first;
    double second;
    double result;
    String operation;
    String answer;
```

2.2.1. 9 Penjelasan Kode Program

a) *public class Calculator_2511532002 {*

Deklarasi kelas utama program kalkulator, dimana semua komponen dan logika ditulis di dalam kelas ini.

b) *Variabel*

1) *private JFrame frame;*

   Variabel frame sebagai jendela utama (*window)* aplikasi *Java Swing*.

2) *private JTextField txtf;*

   Variabel txtf adalah tempat *input/output* angka dan hasil perhitungan.

3) *Double first*

   Uuntuk menyimpan angka pertama yang di *input user.*

4) *Double second*

   Untuk menyimpan angka kedua.

5) *Double Result*

   Untuk menyimpan hasil perhitungan (pnjumlahan, pengurangan, dll).

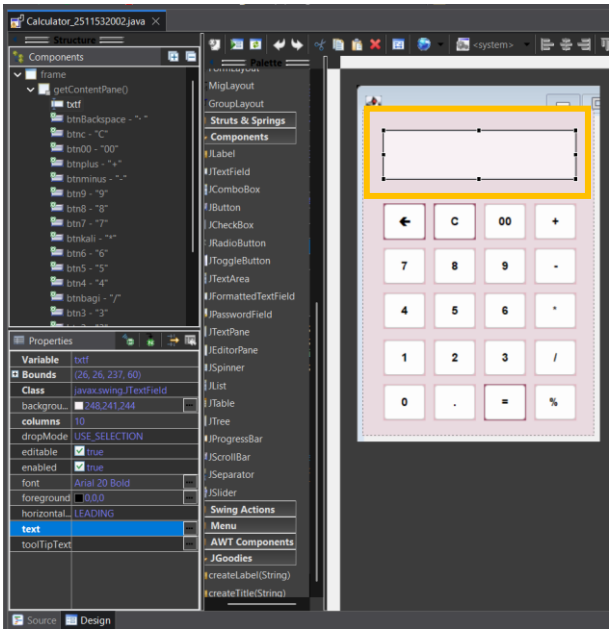
6) *String operation*

   Untuk menyimpan jenis operasi yang dipilih *user (*contoh*: "+", "-", "*", "/")*.

7) *String answer*

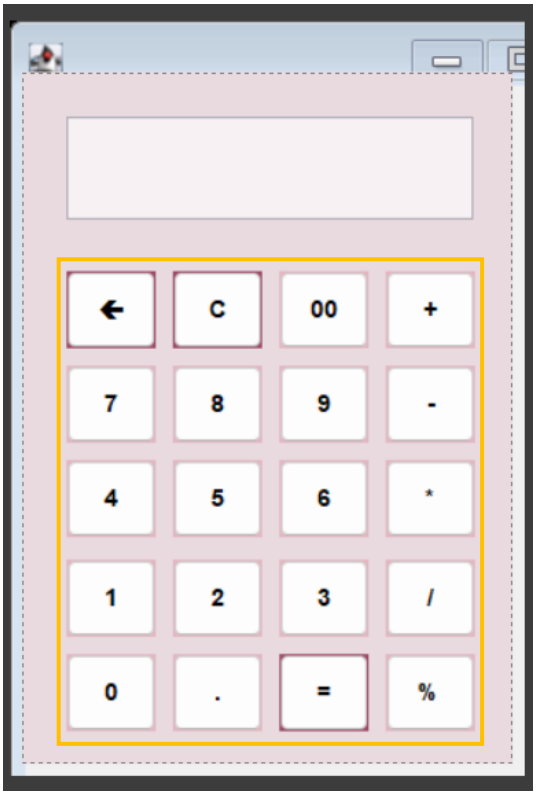
   Untuk menyimpan hasil dalam bentuk *String* agar bisa ditampilkan di *JTextField.*

c) *JTextField*



2.2.1. 10 Penjelasan Syntax JTextField

d) *JButton*



2.2.1. 11 Penjelasan Syntax JButton

1) *Button Backspace*

Tombol *backspace* untuk menghapus satu karakter terakhir pada *textfield.*

```
JButton btnBackspace = new JButton("\uF0E7");
btnBackspace.setForeground(new Color(0, 0, 0));
btnBackspace.setBackground(new Color(157, 77, 105));
btnBackspace.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        String backSpace=null;
        if (txtf.getText().length()>0)
        {
            StringBuilder str=new StringBuilder(txtf.getText());
            str.deleteCharAt(txtf.getText().length()-1);
            backSpace=str.toString();
            txtf.setText(backSpace);
        }

    }
```

2.2.1. 12 Penjelasan Button Backspace

Keterangan:

i. String backSpace = null;

Menyiapkan variabel untuk menyimpan teks setelah karakter terakhir dihapus.

ii. if (txtf.getText().length() > 0)

Mengecek apakah *textfield* tidak kosong, *backspace* hanya boleh jalan jika ada karakter.

iii. str.deleteCharAt(txtf.getText().length() - 1);

Menghapus satu karakter terakhir dari teks.

2) *Button Clear (C/AC)*

```
JButton btnc = new JButton("C");
btnc.setForeground(new Color(0, 0, 0));
btnc.setBackground(new Color(157, 77, 105));
btnc.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        txtf.setText(null);
    }
});
```

2.2.1. 13 Penjelasan Button Clear

Keterangan:

i. txtf.setText(null);

Berarti menghapus semua isi di *JTextField*.

3) *Button Angka (00, 0-9)*

```
JButton btn9 = new JButton("9");
btn9.setBackground(new Color(227, 183, 194));
btn9.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String number=txtf.getText()+btn9.getText();
        txtf.setText(number);
    }
```

2.2.1. 14 Penjelasan Button Angka

Setiap tombol angka menambahkan angka/ text nya ke *textfield*.

Keterangan:

i. *txtf.getText()*

Mengambil teks yang sudah ada.

ii. *btn00.getText()*

Mengambil teks di tombol (misal "00", "8", "9").

4) *Button Operator (+, -, , /, %)*

```
JButton btnplus = new JButton("+");
btnplus.setBackground(new Color(227, 183, 194));
btnplus.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        first = Double.parseDouble(txtf.getText());
        txtf.setText("");
        operation="+";
    }
});
```

2.2.1. 15 Penjelasan Button Operator

Keterangan:

i. *First=Double.parseDouble(txtf.getText());*

Mengambil angka yang sedang ditampilkan di *JTextField*.

ii. *txtf.setText(“”);*

Mengosongkan *textfield* untuk mengetik angka kedua.

iii. operation = "+";

Menyimpan jenis operasi yang dipilih *user*.

5) *Button Hasil (=)*

```
JButton btnhasil = new JButton("=");
btnhasil.setBackground(new Color(157, 77, 105));
btnhasil.setForeground(new Color(0, 0, 0));
btnhasil.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String answer;
        second=Double.parseDouble(txtf.getText());
        if(operation=="+")
        {
            result=first+second;
            answer=String.format("%.2f", result);
            txtf.setText(answer);
        }
        else if (operation=="-")
        {
            result=first-second;
            answer=String.format("%.2f", result);
            txtf.setText(answer);
        }
        else if (operation=="*")
        {
                result=first*second;
            answer=String.format("%.2f", result);
            txtf.setText(answer);
        }
        else if (operation=="/")
        {
            result=first/second;
        answer=String.format("%.2f", result);
        txtf.setText(answer);
        }
        else if (operation=="%")
        {
            result=first%second;
        answer=String.format("%.2f", result);
        txtf.setText(answer);
        }
    }
});
```

2.2.1. 16 Pejelasan Button Hasil

Keterangan*:*

iv. *second=Double.parseDouble(txtf.getText());*

Mengambil angka kedua.

v. *if(operation == "+") {*

Mengecek operator apa yang dipakai.

vi. *result = first + second;*

Melakukan operasi untuk bilangan *input first* dan *second* lalu memasukan ke *result.*

vii. *answer = String.format("%.2f", result);*

Hasil/ *answer* dari *result* dengan dibuat 2 angka di belakang koma.

viii. *txtf.setText(answer);*

Menampilkan hasil/ *answer* ke *textfiled.*

2.2.3 Uji Coba Hasil Program

Setelah membuat program, maka *user* dapat menguji coba program. Klik tombol *run* pada menu *toolbar*, maka akan muncul *pop up* program *calculator,* disini *user* dapat mulai menguji coba program. Disini dapat dilihat bahwa *program "Calculator"* telah berjalan.

Berikut hasil analisis dari uji coba lainnya:

a) Uji Coba 1: Penjumlahan

2.2.3. 1 Hasil Uji Coba ke-1

Keterangan:

Input ke-1 = 10

Input ke-2 = 7

Dapat dilihat bahwa program “Calculator” telah berjalan dengan benar dan menghasilkan hasil perhitungan yang sesuai.

b) Uji Coba 2: Pengurangan

2.2.3. 2 Hasil Uji Coba ke-2

Keterangan:

Input ke-1 = 10

Input ke-2 = 3

Dapat dilihat bahwa program “Calculator” telah berjalan dengan benar dan menghasilkan hasil perhitungan yang sesuai.

c) Uji Coba 3: Perkalian

2.2.3. 3 Hasil Uji Coba ke-3

Keterangan:

Input ke-1 = 5

Input ke-2 = 2

Dapat dilihat bahwa program "*Calculator*" telah berjalan dengan benar dan menghasilkan hasil perhitungan yang sesuai

d) Uji Coba 4: Pembagian

2.2.3. 4 Uji Coba ke-4

Keterangan:

Input ke-1 = 10

Input ke-2 = 2

Dapat dilihat bahwa program "*Calculator*" telah berjalan dengan benar dan menghasilkan hasil perhitungan yang sesuai

e) Uji Coba 5: Persen

2.2.3. 5 Uji Coba ke-5

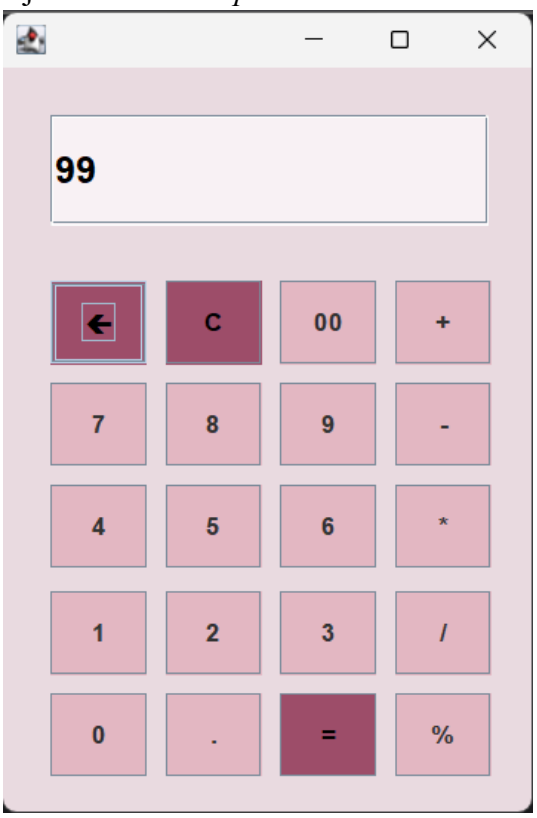
Keterangan:

Input ke-1 = 15

Input ke-2 = 4

Dapat dilihat bahwa program "*Calculator*" telah berjalan dengan benar dan menghasilkan hasil perhitungan yang sesuai

f) Uji Coba 6: *Backspase*

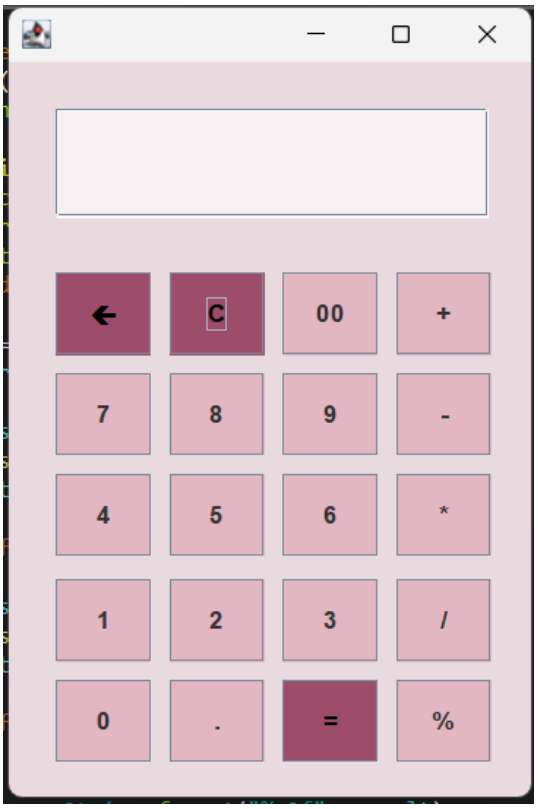


2.2.3. 6 Uji Coba ke-6

Untuk uji coba *Backspace button*, sudah di uji selama proses perhitungan sebelumnya, disini hanya untuk pembuktian. *User input* awal 999, setelah *backspace* menjadi 99.

Dapat dilihat bahwa program "Calculator" telah berjalan dengan benar dan menghasilkan hasil perhitungan yang sesuai

g) Uji Coba 7: *Clear*



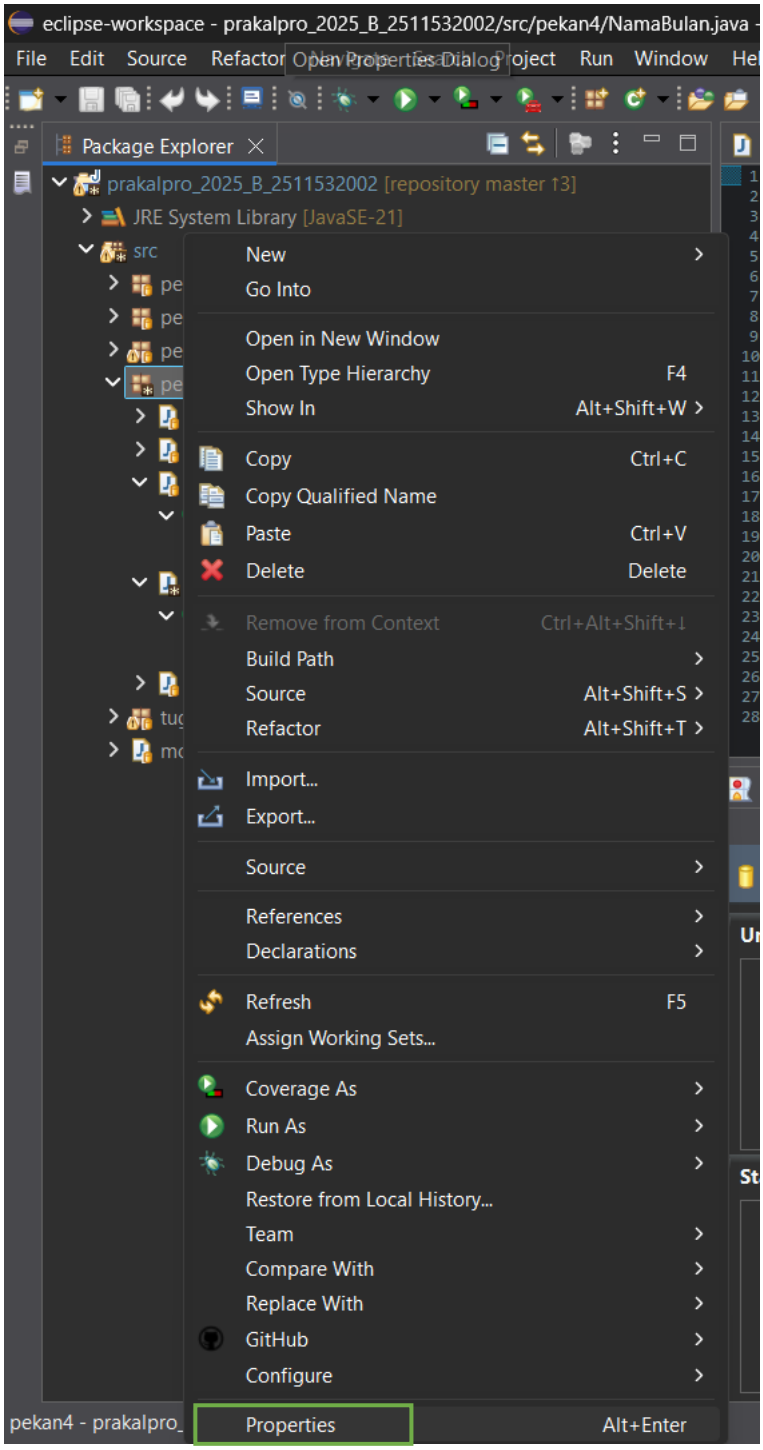
2.2.3. 7 Uji Coba ke-7

Untuk uji coba *Clear button*, sudah di uji selama proses perhitungan sebelumnya, disini hanya untuk pembuktian. User *input* awal 99, setelah *clear* menjadi kosong.

Dapat dilihat bahwa program "*Calculator*" telah berjalan dengan benar dan menghasilkan hasil perhitungan yang sesuai.

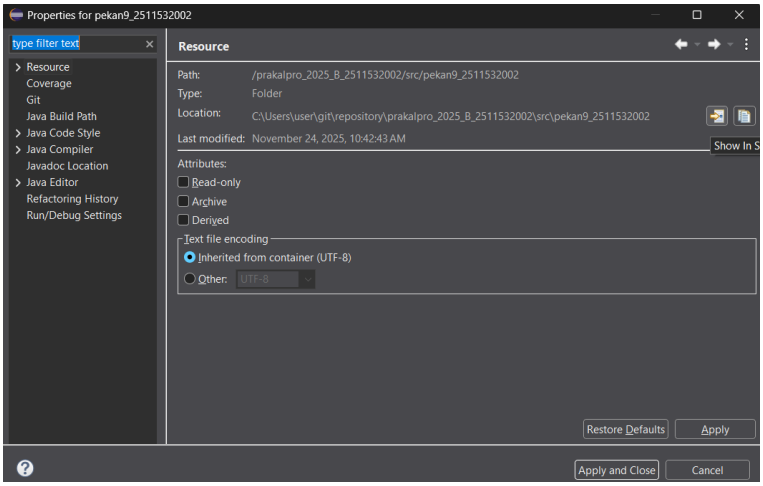
2.2.4 *Upload* ke GitHub

1. Setelah semua program selesai dibuat, selanjutnya adalah *user* perlu untuk memasukan program yang dibuat di Eclips ke *repository* GitHub. Selain dengan cara *"Team"* atau *"Commit and Push"*. Bisa juga dengan cara *upload* manual. Caranya klik kanan pada *package,* lalu pilih *properties.*



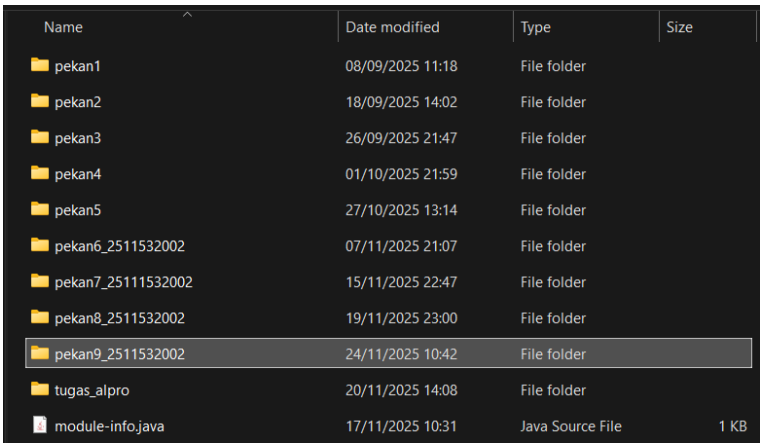
2.2.4. 1 Package Properties of pekan9_2511532002

2. Maka akan muncul *pop up menu "Properteis for pekan9_2511532002"*. Lalu klik *icon "Show in System Folder"*.



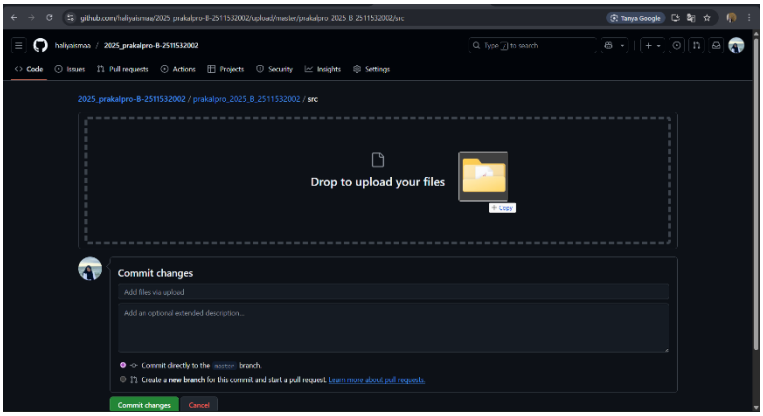
2.2.4. 2 Package Properties of pekan9_2511532002

3. Maka akan menunjukan lokasi *file "pekan9_2511532002"* pada *File Explorer computer.*



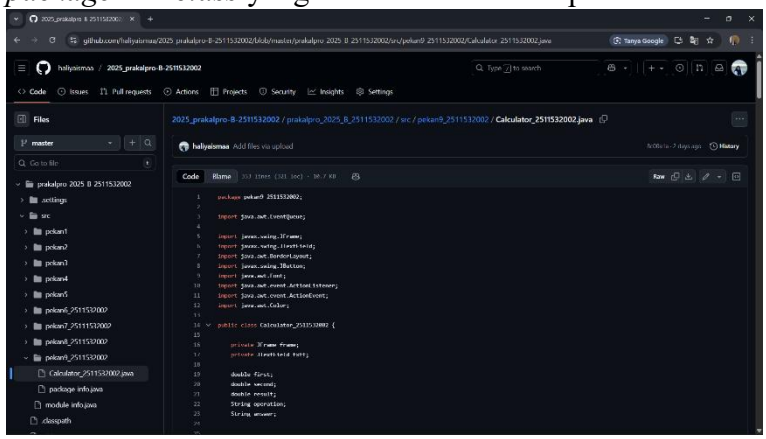
2.2.4. 3 Lokasi File pakage pekan9_2511532002

4. Setelah itu *drag and drop file* "*pekan9_2511532002*" ke GitHub. Lalu tambahkan keterangan jika perlu. Setelah itu klik *upload.*



2.2.4. 4 Drag and Drop Package Pekan9_2511532002

5. Maka dapat dilihat pada *repository GitHub* bahwa file "*pekan9_2511532002*" telah terunggah via *Upload.*
Dapat kita lihat disini jika kita buka, maka akan muncul *file package* dan *class* yang telah di buat di Eclipse.



2.2.4. 5 Hasil Upload Package Pekan9_2511532002

BAB III
KESIMPULAN

3.1 Kesimpulan

Berdasarkan praktikum pekan ke-9 dengan topik *"Pembuatan Program Kalkulator Berbasis GUI Menggunakan Java"*, dapat disimpulkan bahwa penggunaan *Application Window* pada *Swing Designer* sangat mempermudah proses pengembangan antarmuka aplikasi. Komponen seperti *JButton* dan *JTextField* dapat ditambahkan secara *drag and drop* dan diatur secara visual, sehingga penyusunan *layout* menjadi lebih cepat dan efisien tanpa perlu menulis seluruh kode secara manual.

Program berjalan dengan baik, di mana *user* dapat memasukkan angka, memilih operasi (+,-, *, /, dan %), dan memperoleh hasil perhitungan secara otomatis. Setiap tombol berhasil memicu event yang sesuai, termasuk fitur pendukung seperti *Clear* dan *Backspace*. Secara keseluruhan, praktikum ini membantu memahami dasar pembuatan *GUI* di *Java* serta penerapannya dalam pembuatan aplikasi sederhana yang fungsional dan *user-friendly*.

3.2 Saran

Saran untuk praktikum selanjutnya adalah agar mahasiswa lebih teliti dalam memahami penulisan *component* yang tepat sesuai dengan kebutuhan program, terutama saat mengatur komponen *GUI* dan menangani *input* dan *output* pengguna. Selain itu, penting untuk membedakan dengan benar antara tipe data *String* dan *numerik* agar tidak terjadi kesalahan logika pada proses perhitungan maupun tampilan *output*.

DAFTAR PUSTAKA

[1] Eclipse Foundation, “WindowBuilder – Eclipse Project.” [Daring]. Tersedia pada: https://eclipse.dev/windowbuilder/. [Diakses: 11-Nov-2025].

[2] E. H. Chua, “Java GUI Programming – Swing (JFC).” [Daring]. Tersedia pada: https://www3.ntu.edu.sg/home/ehchua/programming/java/J4a_GUI.html. [Diakses: 11-Nov-2025].

[3] GeeksforGeeks, “Introduction to Java Swing.” [Daring]. Tersedia pada: https://www.geeksforgeeks.org/java/introduction-to-java-swing/. [Diakses: 11-Nov-2025].

[4] Tech Projects, “GUI Calculator in JAVA using eclipse ide,” YouTube, 2020. [Daring]. Tersedia pada: https://www.youtube.com/watch?v=mZVD5OssHQM. [Diakses: 25-Nov-2025].