PENJELASAN PROGRAM GUI FORM DENGAN JAVA

“PERSEWAAN BUKU”

Pengembang :

NAUFAL HAIDAR RAUF A11.4423 - A11.2019.12342

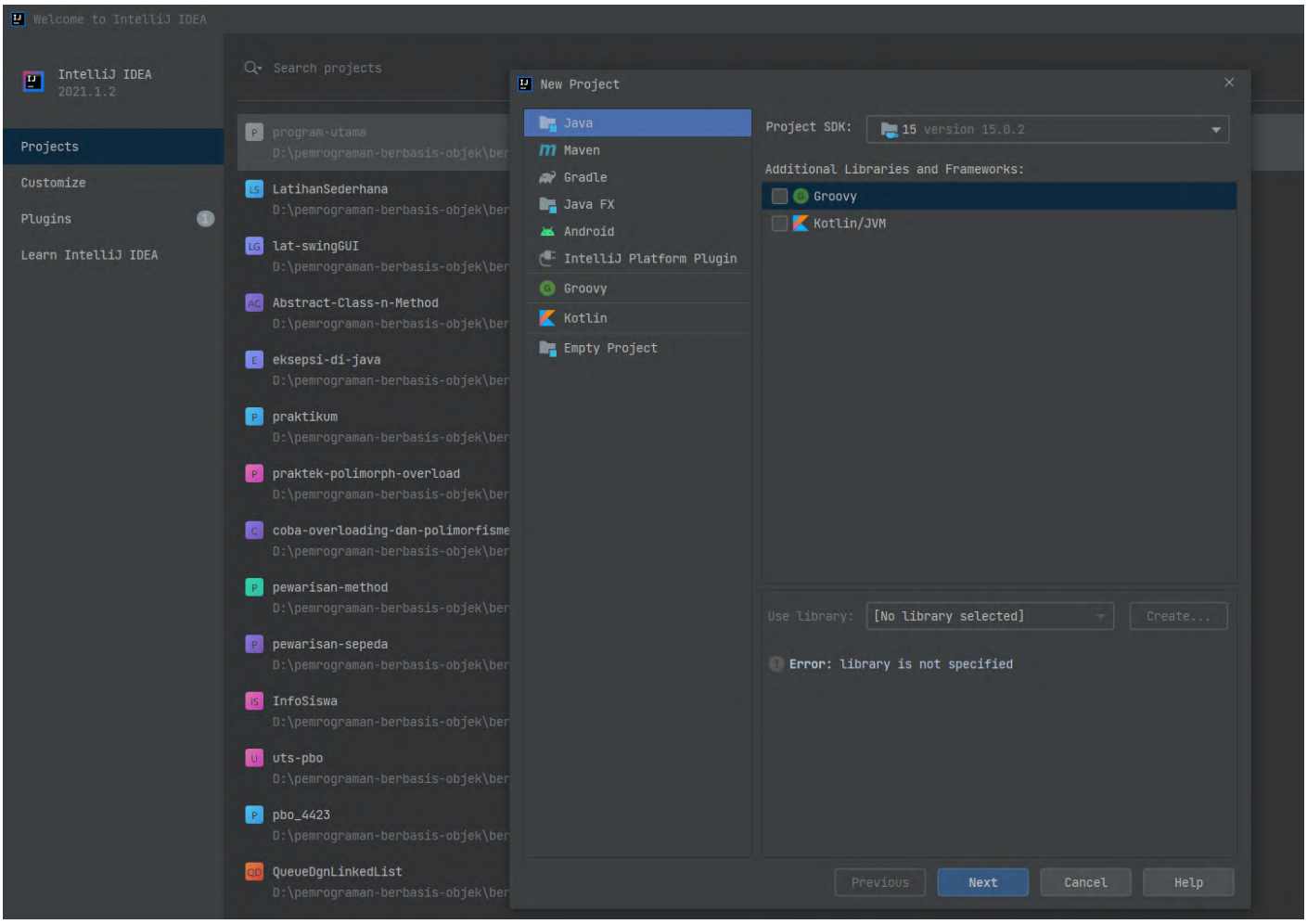
Alat yang digunakan

- *jetBrains IntelliJ IDEA Community Version*
- *MySQL Connector J Windows Version* (diunduh dari https://dev.mysql.com/downloads/connector/j/)
- *JDK 15*
- *MySQL* dari *XAMPP*
- *phpMyAdmin* dari *XAMPP*

Persiapan membuat *project*

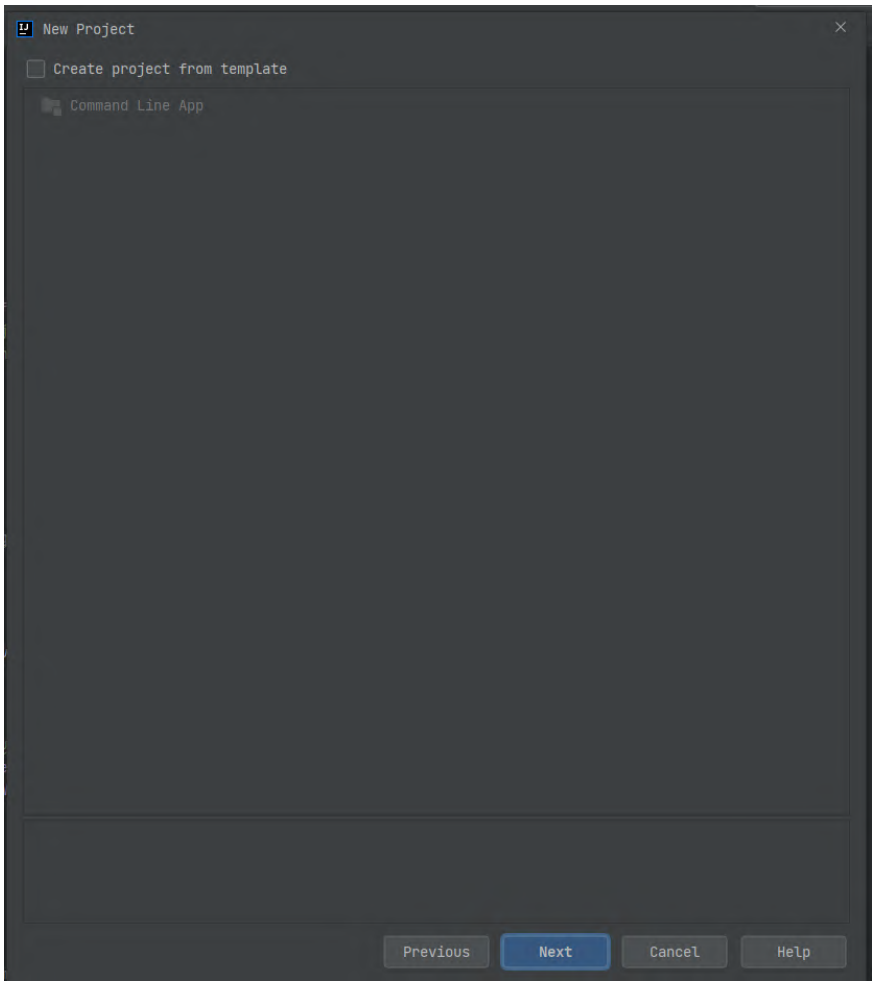
1. Klik *File* kemudian pilih *New Project*. Pada pilihan di panel kiri, pilih *Java*. Kali ini, kita akan membuat *GUI Form* secara manual, bukan menggunakan *framework*.

   NB: *JavaFX* pada opsi tersebut adalah *framework*.



Klik *Next* untuk melanjutkan.

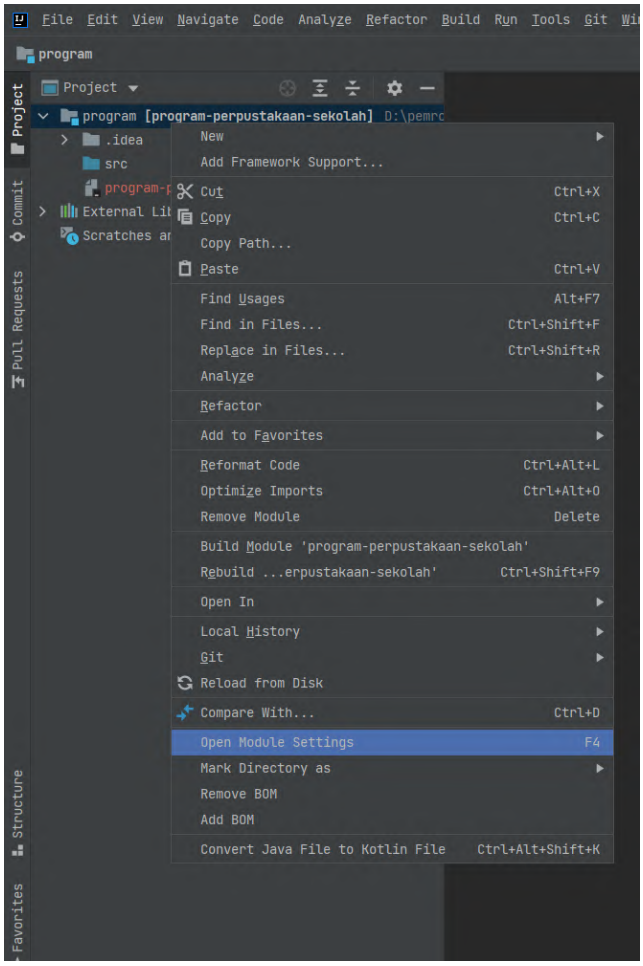
2. Pada bagian selanjutnya, lanjut klik *Next* . Biarkan pilihan *Create project from template* tidak tercentang.



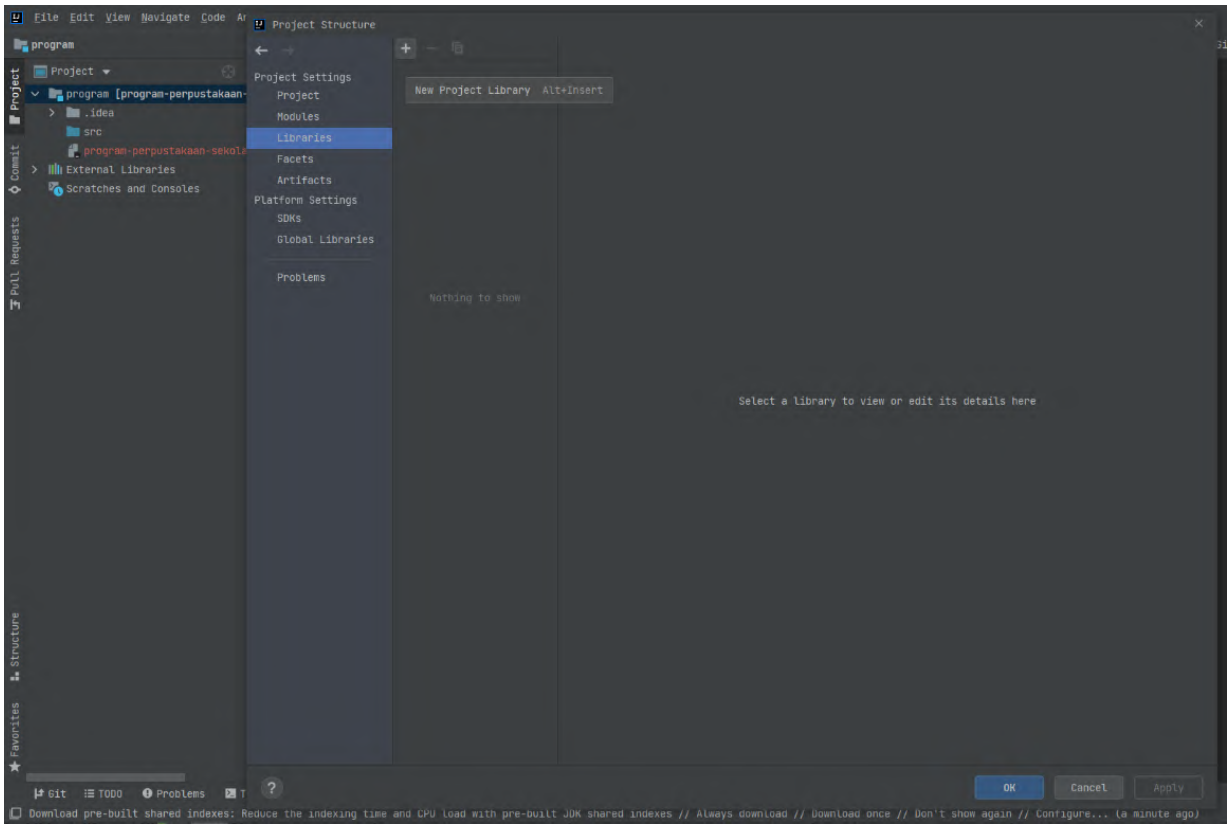
Selanjutnya, pilih folder kosong yang akan digunakan oleh *IntelliJ* untuk menyimpan *project* kita.

Menambahkan *MySQL driver* untuk *Java*

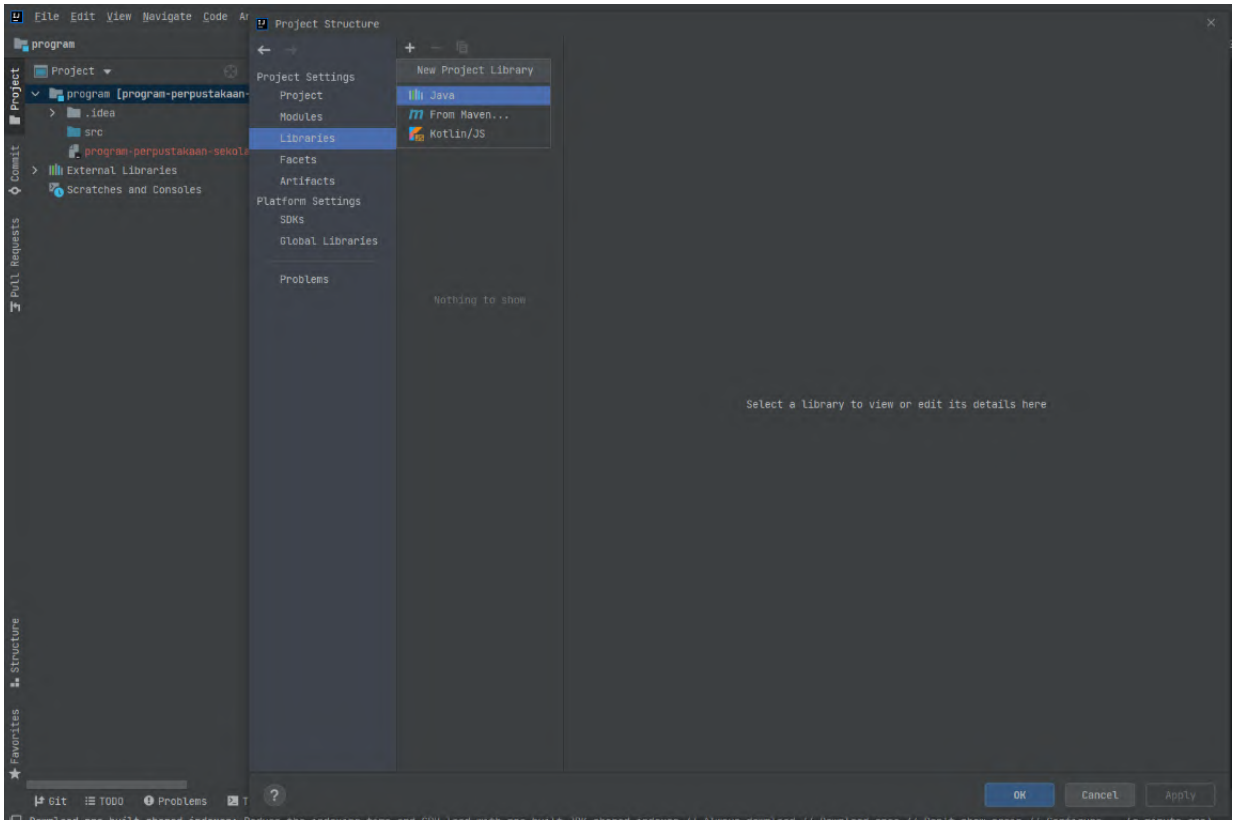
1. Klik kanan pada nama *project* yang dibuat, lalu pilih *Open Module Settings*



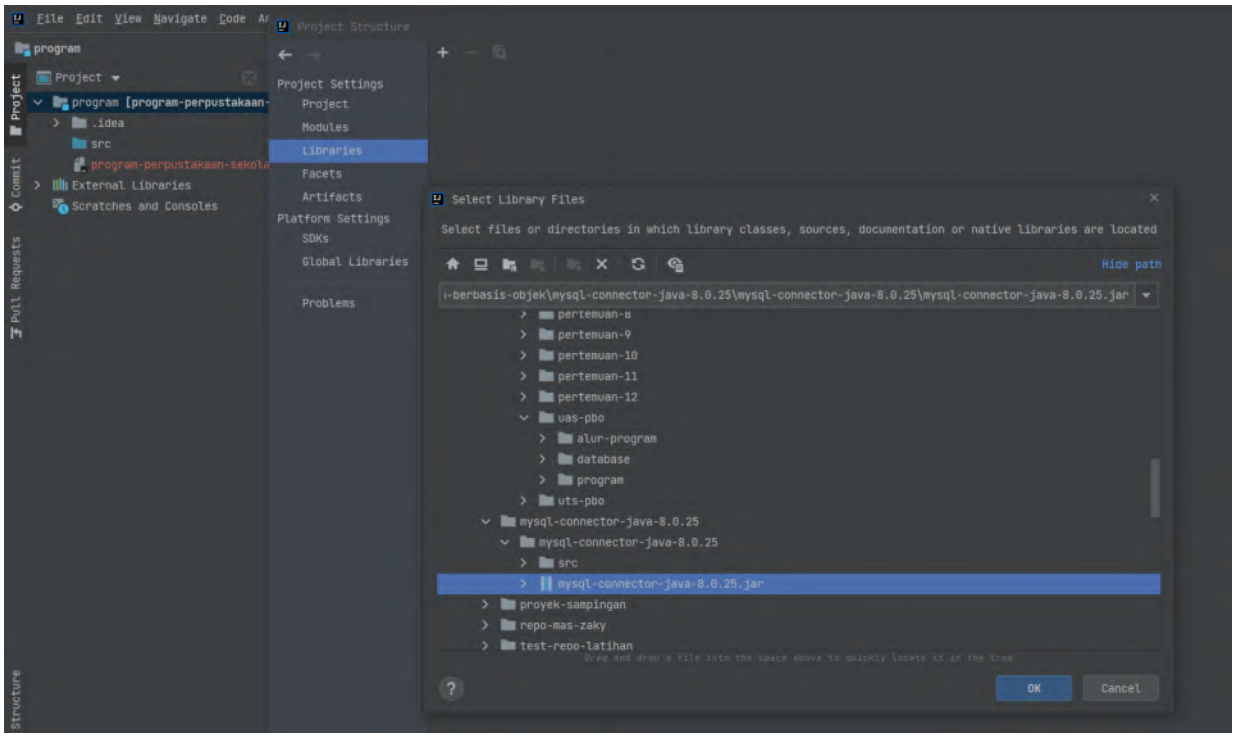
2. Pada panel *Project settings*, pilih *Libraries*



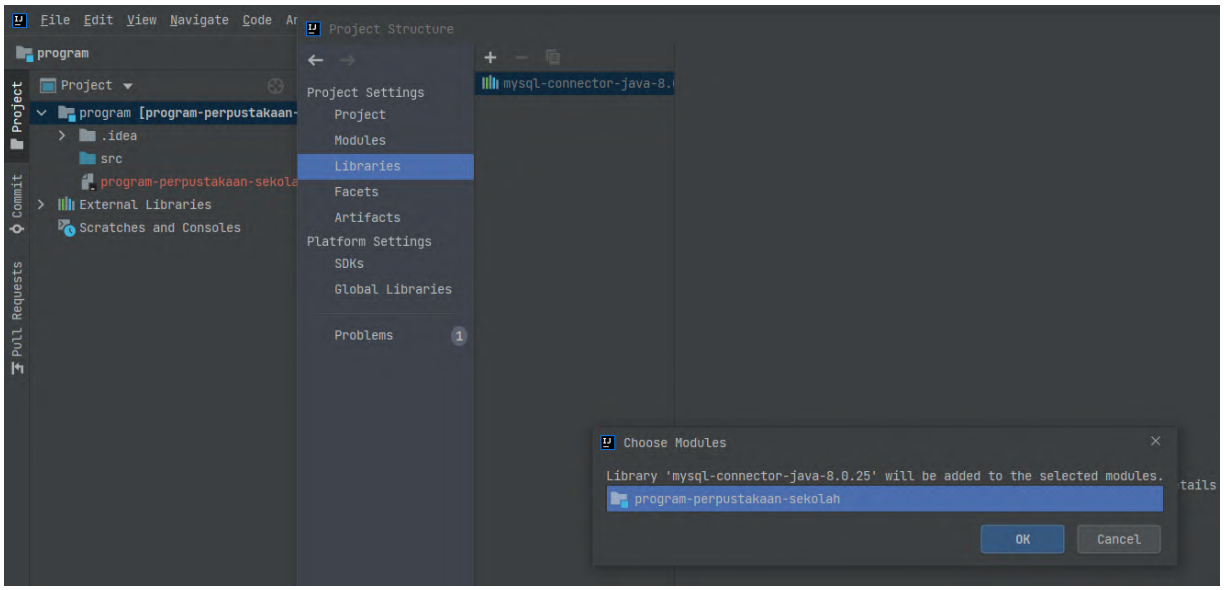
3. Pilih yang *Java*. Artinya, kita harus menambahkan *file* yang bertipe *.jar* (*java archive)*.



4. Arahkan ke lokasi hasil *extract* dari proses mengunduh *MySQL connector J* sebelumnya. Ingat, pilih yang *extension*-nya *.jar*

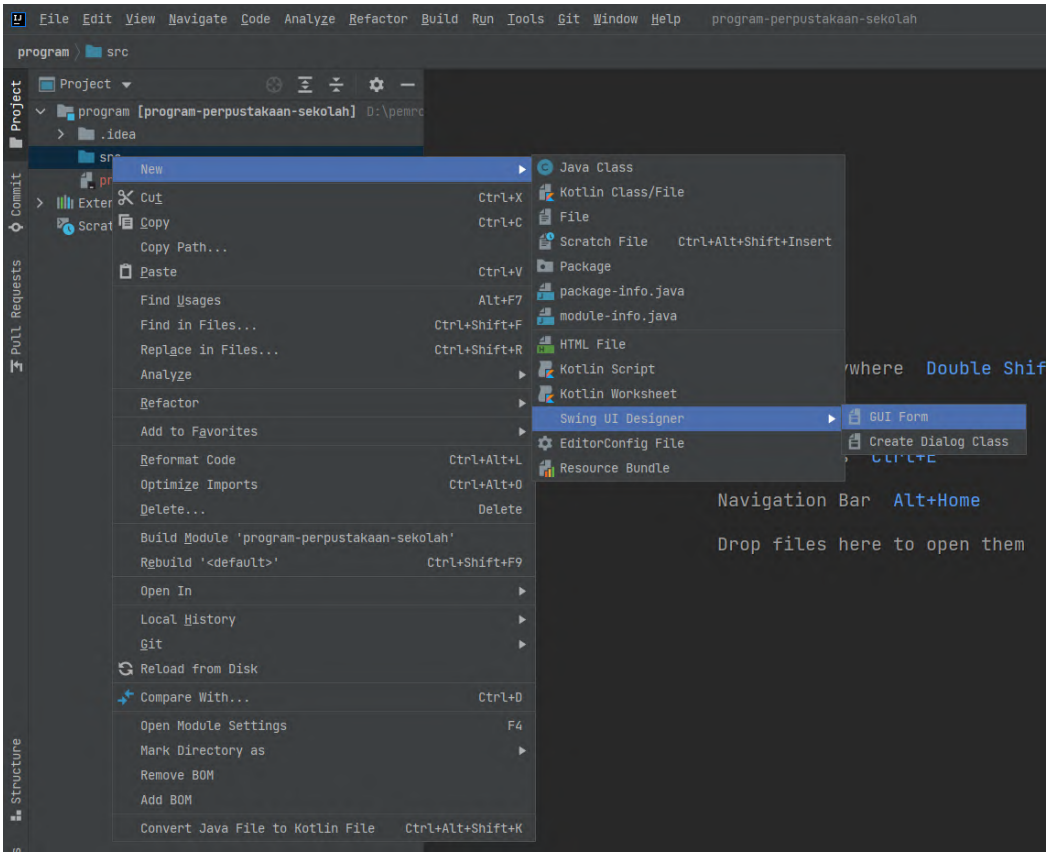


5. Konfirmasi penambahan *library* baru dengan klik *OK*. Terakhir, klik *Apply* lalu klik *OK* untuk keluar dari *Module settings*.

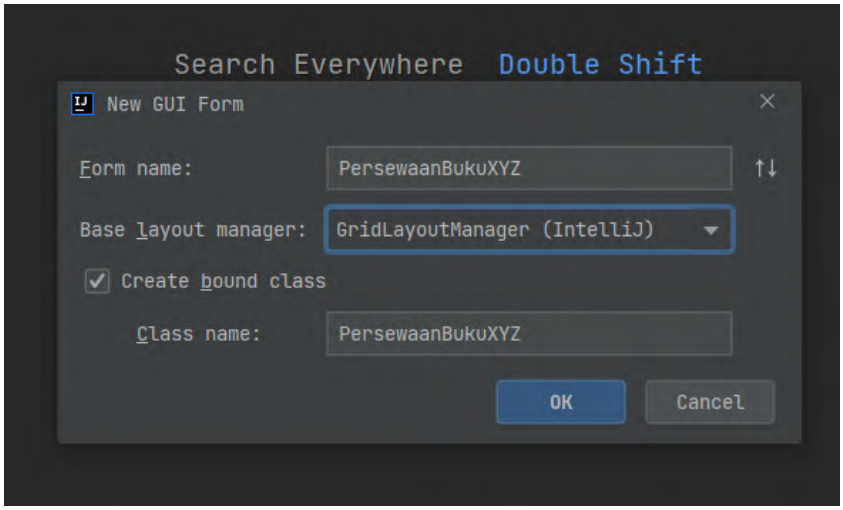


Membuat GUI Form di *IntelliJ IDEA*

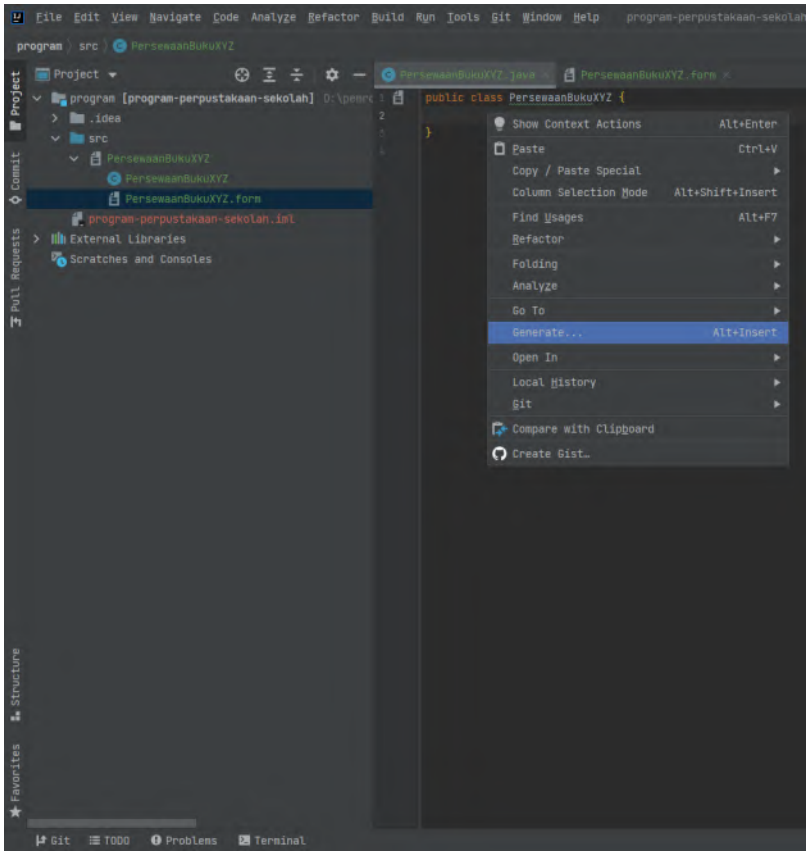
1. Klik kiri pada folder *src* , lalu klik kanan pada folder tersebut lalu pilih *New…* pada opsi yang muncul. Selanjutnya pilih *Swing UI Designer*. Terakhir, pilih *GUI Form*.



2. Beri nama *form* sesuai keinginan. Nama *form* ini juga akan digunakan untuk memberi nama *class* (file bertipe *.java*) milik *form* tersebut. Biarkan saja centang pada opsi *Bound to class* .

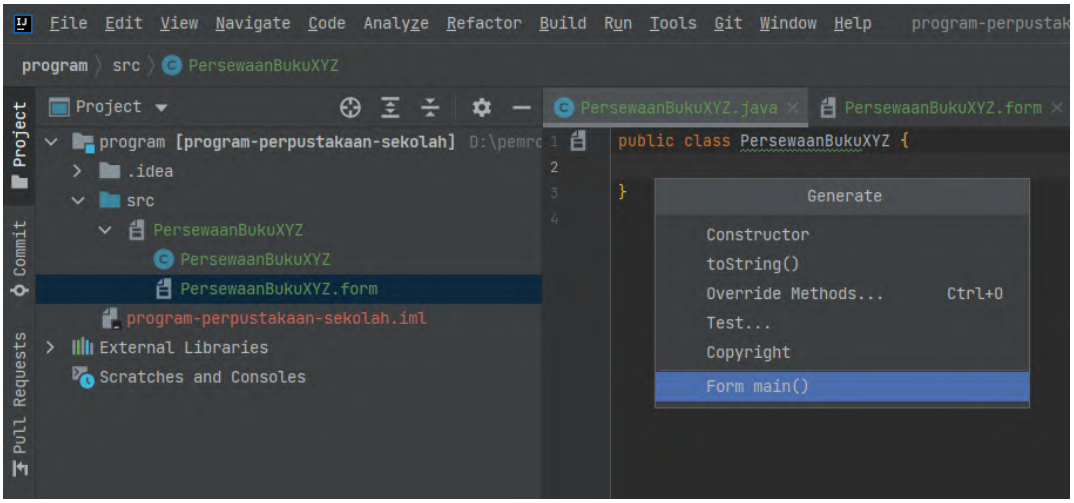


3. Untuk membuat *main method*, klik kanan di dalam *class* yang baru saja dibuat, lalu pilih *Generate…*



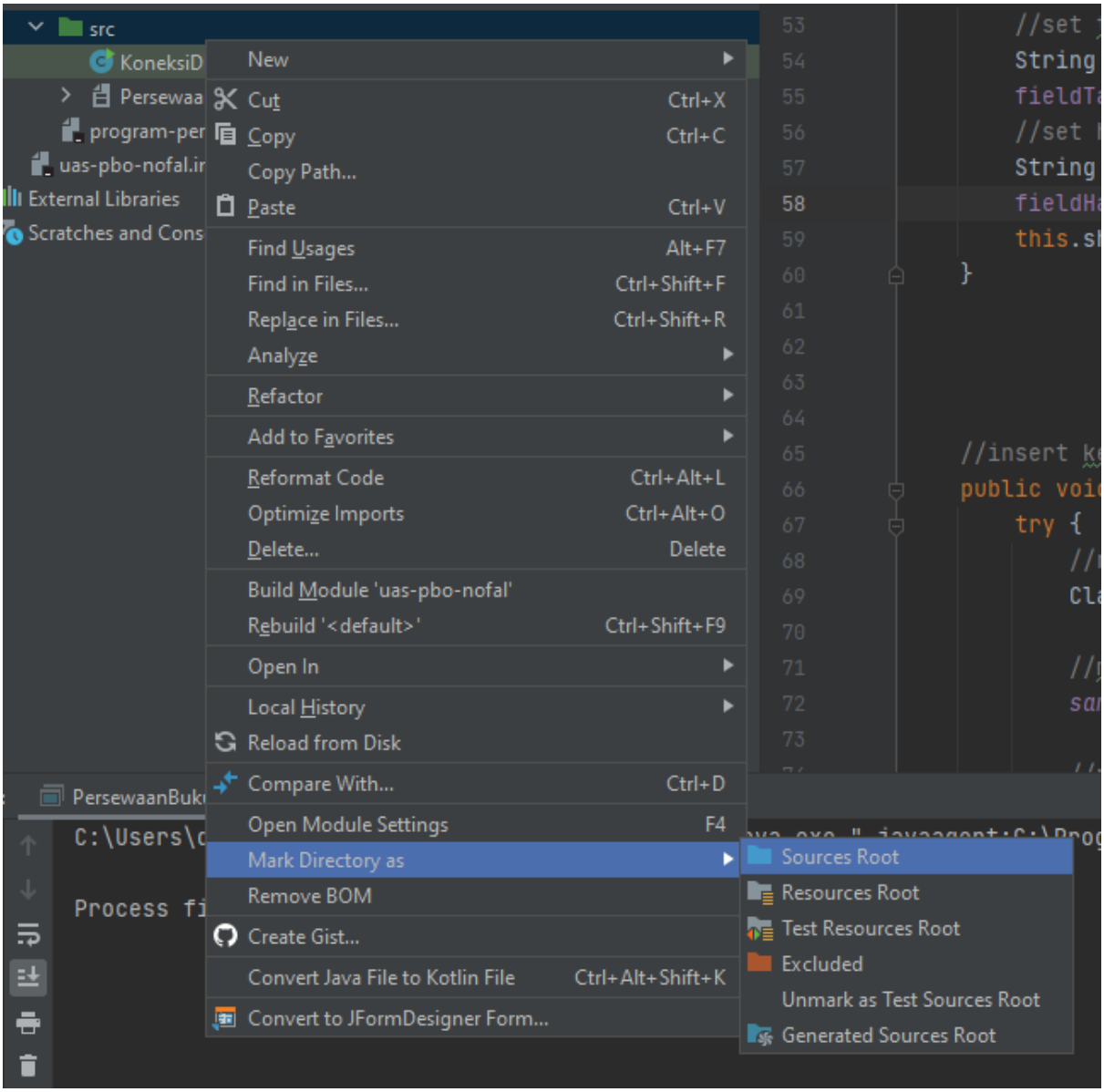
Selanjutnya, pilih *Form main()*

NB: *Panel* yang dibuatkan oleh *IntelliJ* (di dalam file *.form*) harus diberi nama pada kotak *field name*



Tambahan :

Jika tombol *Play* (tombol segitiga hijau) tidak muncul setelah *main method* di-*generate* , klik kanan pada folder *src*, lalu pilih *Mark Directory as Sources Root*


src
KoneksiD
Persewaa
program-per
uas-pbo-nofal.ir
External Libraries
Scratches and Cons
New
Cut Ctrl+X
Copy Ctrl+C
Copy Path...
Paste Ctrl+V
Find Usages Alt+F7
Find in Files... Ctrl+Shift+F
Replace in Files... Ctrl+Shift+R
Analyze
Refactor
Add to Favorites
Reformat Code Ctrl+Alt+L
Optimize Imports Ctrl+Alt+O
Delete... Delete
Build Module 'uas-pbo-nofal'
Rebuild '<default>' Ctrl+Shift+F9
Open In
Local History
Reload from Disk
Compare With... Ctrl+D
Open Module Settings F4
Mark Directory as
Remove BOM
Create Gist...
Convert Java File to Kotlin File Ctrl+Alt+Shift+K
Convert to JFormDesigner Form...
Sources Root
Resources Root
Test Resources Root
Excluded
Unmark as Test Sources Root
Generated Sources Root
PersewaanBuku
C:\Users\
Process fi
53 //set
54 String
55 fieldTa
56 //set
57 String
58 fieldHa
59 this.sh
60 }
65 //insert ke
66 public voi
67 try {
68 //
69 Cla
71 //
72 sa

Penjelasan isi *PenyewaanBuku.java*

```
//kebutuhan untuk menyambungkan database ke gui form kita
static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";
static final String DB_URL = "jdbc:mysql://localhost/perpus_sekolah";
static final String USER="root";
static final String PASS="";

static Connection sambungkan;
static Statement statmt;
static ResultSet setHasil;
```

- *JDBC_DRIVER* adalah nama *driver* yang kita pakai. Versi lama biasanya menggunakan *com.mysql.jdbc.Driver*
- *DB_URL* memiliki urutan
  jdbc:mysql://lokasidatabase/nama-database

  karena *database* saya ada di komputer sendiri/ *local* , maka saya isi dengan *localhost*. nama *database* yang saya gunakan adalah *perpus_sekolah*
- *USER* merupakan nama *user* yang kita gunakan ketika masuk *database*. Karena saya menggunakan bawaan *xampp,* maka nama *user* adalah *root*
- *PASS* merupakan *password* dari *database* kita. Karena saya menggunakan bawaan *xampp,* maka *password* nya adalah kosong.
- *Connection, Statement, ResultSet* merupakan *method* bawaan *java.sql*

Main method

```
public static void main(String[] args) {
    JFrame frame = new JFrame("PenyewaanBuku");
    frame.setContentPane(new PenyewaanBuku().panelUtama);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.pack();
    frame.setVisible(true);
    frame.setSize(1024,720);
}
```

- *method* beserta isinya merupakan hasil *generate* yang kita lakukan sebelumnya. *PenyewaanBuku* merupakan nama *form* sekaligus nama *class* yang kita buat. *panelUtama* merupakan nama *panel* dari *panel* yang diberikan oleh *IntelliJ* ketika membuat *GUI Form*
- saya menambahkan *setSize()* untuk mengatur ukuran *window* dari *form* nya. *parameter* dari method *setSize()* adalah *int width, int height* . Maka, *form* saya akan berukuran *1024x720* .

Constructor

```
public PenyewaanBuku() {
    //menyembunyikan fieldID
    fieldID.setVisible(false);

    //membuat variabel 'tampilkanTanggal' untuk menampung
    //hasil konversi string dari tanggalTerkini
    String tampilkanTanggal=String.valueOf(tanggalTerkini);

    //set value milik textfield 'fieldTanggal'
    //menjadi berisi tanggalTerkini
    fieldTanggal.setText(tampilkanTanggal);

    //membuat variabel 'tampilkanHari' untuk menampung
    //hasil konversi string dari hariTerkini
    String tampilkanHari=String.valueOf(hariTerkini);

    //set value milik textfield 'fieldHari'
    //menjadi berisi hariTerkini
    fieldHari.setText(tampilkanHari);

    //set value milik textfield 'fieldRupiah'
    fieldRupiah.setText("5.000");


    //menampilkan tabel dari database
    show();
```

```
    //ketika tombol simpan di-klik
    buttonSimpan.addMouseListener(new MouseAdapter() {
        @Override
        public void mouseClicked(MouseEvent e) {
            super.mouseClicked(e);
            String tampungJudul=fieldBuku.getText().toString();
            fieldBuku.setText("");

            //tanggap pinjam diambil dari tanggalTerkini
            LocalDate tampungPinjam=tanggalTerkini;

            //tanggal harus kembali , dimana maksimal peminjaman buku
            //adalah 7 hari sejak pinjam
            LocalDate tampungHarusKembali=tanggalTerkini.plusDays(7);

            //memasukkan ke dalam tabel kemudian ditampilkan data yang
            //baru saja dimasukkan
            insert(tampungJudul,tampungPinjam,tampungHarusKembali);
            show();
        }
    });

    //aksi yang akan dijalankan
    // ketika tombol 'buttonKembali' di klik
    buttonKembali.addMouseListener(new MouseAdapter() {
        @Override
        public void mouseClicked(MouseEvent e) {
            super.mouseClicked(e);
        }
    });

    //aksi yang akan dijalankan
    // ketika tombol 'buttonEdit' di klik
    buttonEdit.addMouseListener(new MouseAdapter() {
        @Override
        public void mouseClicked(MouseEvent e) {
            super.mouseClicked(e);
        }
    });

    //aksi yang akan dijalankan
    // ketika tombol 'buttonDelete' di klik
    buttonDelete.addMouseListener(new MouseAdapter() {
        @Override
        public void mouseClicked(MouseEvent e) {
            super.mouseClicked(e);
        }
    });
}
```

Method *insert*

```
//insert ketika tombol 'Simpan' di-klik
public void insert(String judulBuku, LocalDate tanggalPinjam, LocalDate
wajibKembali ){
    try {
        //register driver yang akan dipakai
        Class.forName(JDBC_DRIVER);

        //menyambungkan ke database
        sambungkan= DriverManager.getConnection(DB_URL,USER,PASS);

        //perintah sql-nya
        String sql="INSERT INTO sewabuku (judul, tanggal_pinjam,
tanggal_harus_kembali) VALUES (?, ?, ?)";

        //Prepared statement untuk menghindari sql injection
        PreparedStatement prstmt= sambungkan.prepareStatement(sql);
        prstmt.setString(1, judulBuku);
        prstmt.setString(2, String.valueOf(tanggalPinjam));
        prstmt.setString(3, String.valueOf(wajibKembali));

        prstmt.execute();

        //tutup koneksi
        sambungkan.close();


    }catch (SQLException throwables) {
        throwables.printStackTrace();
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
}
```

- pada method *insert()* tersebut, saya menggunakan *PreparedStatement (method* bawaan dari *java.sql*) untuk menghindari *sql dependency injection* sekaligus membuat proses memasukkan ke *database* menjadi lebih aman dan cepat

Method *show()*

```
//menampilkan isi dari tabel pada database
public void show(){
    try {
        //register driver yang akan dipakai
        Class.forName(JDBC_DRIVER);

        //menyambungkan ke database
        sambungkan= DriverManager.getConnection(DB_URL,USER,PASS);

        //mengatur table head atau nama kolom
        DefaultTableModel kerangkaTabel = new DefaultTableModel();
        kerangkaTabel.addColumn("Nomer");
        kerangkaTabel.addColumn("Judul Buku");
        kerangkaTabel.addColumn("Tanggal Pinjam");
        kerangkaTabel.addColumn("Tanggal Harus Kembali");
        kerangkaTabel.addColumn("Tanggal Kembali");
        kerangkaTabel.addColumn("Denda");
        kerangkaTabel.addColumn("Biaya Sewa");

        //perintah sql nya
        statmt=sambungkan.createStatement();
        String sql = "SELECT * FROM sewabuku";

        //eksekusi perintah sql
        setHasil= statmt.executeQuery(sql);

        while (setHasil.next()){
            kerangkaTabel.addRow(new Object[] {
                    setHasil.getString("id"),
                    setHasil.getString("judul"),
                    setHasil.getString("tanggal_pinjam"),
                    setHasil.getString("tanggal_harus_kembali"),
                    setHasil.getString("tanggal_kembali"),
                    setHasil.getString("denda"),
                    setHasil.getString("biaya_sewa")
            });

        }
        setHasil.close();
        sambungkan.close();
        statmt.close();

        //set table model tadi ke dalam JTables
        tableBuku.setModel(kerangkaTabel);
    }catch (SQLException eksepsi){
        eksepsi.getMessage();
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
}
```

- pada method tersebut, memang tidak menggunakan *PreparedStatement* karena hanya menampilkan isi dari tabel *sewabuku* yang berada di dalam database *perpus_sekolah*

Method *update()*

```
//update ketika tombol 'btnKembaliBuku' di-klik
public void update(LocalDate tanggalKembali, int denda,int biayaSewa, int
id){
    try {
        //register driver yang akan dipakai
        Class.forName(JDBC_DRIVER);

        //menyambungkan ke database
        sambungkan= DriverManager.getConnection(DB_URL,USER,PASS);

        //perintah sqlnya untuk update table
        String sql="UPDATE sewabuku SET tanggal_kembali=? ,denda=? ,
biaya_sewa=? WHERE id=?";

        //prepared statement untuk update
        PreparedStatement prstmt=sambungkan.prepareStatement(sql);
        prstmt.setString(1, String.valueOf(tanggalKembali));
        prstmt.setString(2, String.valueOf(denda));
        prstmt.setString(3, String.valueOf(biayaSewa));
        prstmt.setString(4, String.valueOf(id));

        prstmt.executeUpdate();

        prstmt.close();
        sambungkan.close();



    } catch (SQLException throwables) {
        throwables.printStackTrace();
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
}
```