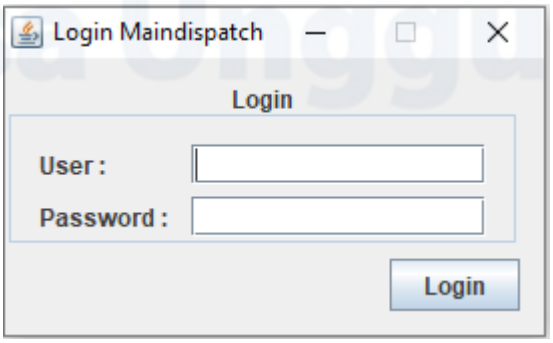
Lampiran 2 Source Code Program

I. Aplikasi desktop client-server sistem monitoring

Login



Setelah Login , petugas admin dapat melengkapi pengisian form input data operator, data armada truk , dan data kontainer yang telah disampaikan pada bab sebelumnya. Dengan Bahasa JAVA, maka program client-server ini dibangun kedalam 3 kategori yaitu : API, Client dan Server. Beberapa modul penulis tampilkan potongan kode programnya dan selengkapnya bisa dilihat softcopynya pada CD.

A. Kode program API

1. Truk.java

```
package maindispatchapi.entity;
import java.io.Serializable;
/* @author albani.soeleman*/
public class Truk implements Serializable
{
  private int IdTruk;
   public int getIdTruk() {return IdTruk;}
   public void setIdTruk(int IdTruk) {this.IdTruk = IdTruk;}
   private String KodeUnit;
   private String Pemilik ;
   private String MerkUnit;
   private String NoPol;
   private String JenisTruk;
   private Long BeratUnit;
   private Long BebanAngkut;
   private String StatusUnit;
     public String getKodeUnit() {return KodeUnit;}
     public void setKodeUnit(String KodeUnit) {this.KodeUnit = KodeUnit;}
     public String getPemilik() {return Pemilik;}
     public void setPemilik(String Pemilik) {this.Pemilik = Pemilik;}
     public String getMerkUnit() {return MerkUnit;}
     public void setMerkUnit(String MerkUnit) {this.MerkUnit = MerkUnit;}
     public String getNoPol() {return NoPol;}
     public void setNoPol(String NoPol) {this.NoPol = NoPol;}
     public String getJenisTruk() {return JenisTruk;}
     public void setJenisTruk(String JenisTruk) {this.JenisTruk = JenisTruk;}
     public Long getBeratUnit() {return BeratUnit;}
     public void setBeratUnit(Long BeratUnit) {this.BeratUnit = BeratUnit;}
     public Long getBebanAngkut() {return BebanAngkut;}
     public void setBebanAngkut(Long BebanAngkut) {this.BebanAngkut = BebanAngkut;}
     public String getStatusUnit() {return StatusUnit;}
     public void setStatusUnit(String StatusUnit) {this.StatusUnit = StatusUnit;}
}
```

2. Operator.java

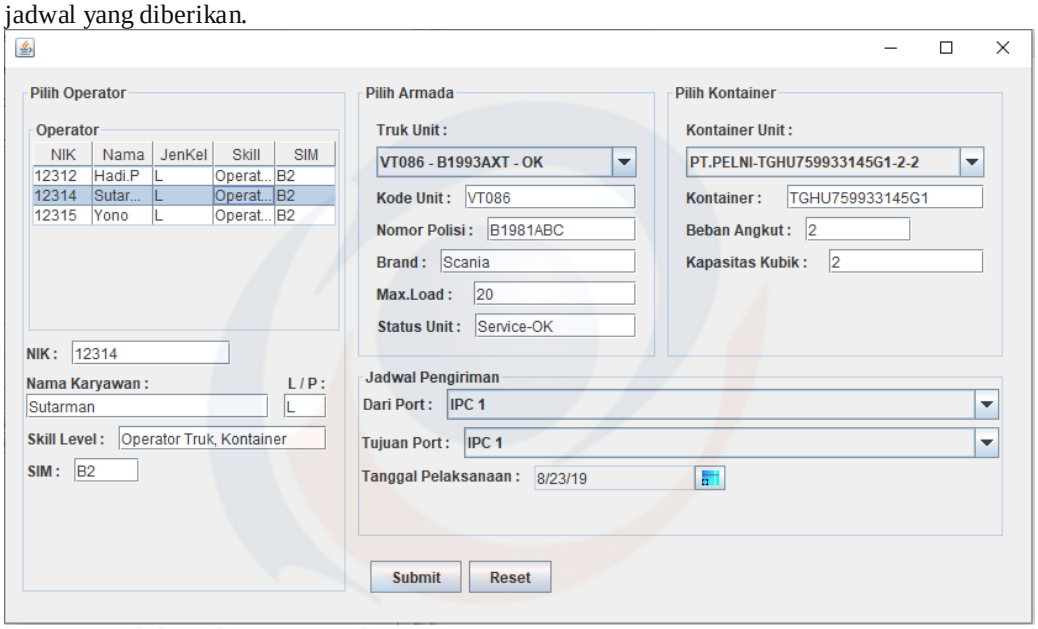
```
package maindispatchapi.entity;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Date;
/* @author albani.soeleman */
public class Operator implements Serializable
{
  private Long id;
  public Long getId() {return id;}
  public void setId(Long id) {this.id = id;}
  private String NIK;
  private String Nama ;
  private String JenKel;
  private String Posisi;
  private String Skill;
  private String SIM;
  private Date ExpSIM;
    public String getNIK() {return NIK;}
    public void setNIK(String NIK) {this.NIK = NIK;}
    public String getNama() {return Nama;}
    public void setNama(String Nama) {this.Nama = Nama;}
    public String getJenKel() {return JenKel;}
    public void setJenKel(String JenKel) {this.JenKel = JenKel;}
    public String getPosisi() {return Posisi;}
    public void setPosisi(String Posisi) {this.Posisi = Posisi;}
    public String getSkill() {return Skill;}
    public void setSkill(String Skill) {this.Skill = Skill;}
    public String getSIM() {return SIM;}
    public void setSIM(String SIM) {this.SIM = SIM;}
    public Date getExpSIM() {return ExpSIM;}
    public void setExpSIM(Date ExpSIM) {this.ExpSIM = ExpSIM;}
}
```

3. Kontainer.java

```
package maindispatchapi.entity;
import java.io.Serializable;
/*@author albani.soeleman*/
public class Kontainer implements Serializable
{
  private Long IdKTR;
  public Long getIdKTR() {return IdKTR;}
  public void setIdKTR(Long IdKTR) {this.IdKTR = IdKTR;}
  private String Pemilik ;
  private String OwnerCode ;
  private String ProductGroup;
  private long Registration;
  private long CheckDigit;
  private String Size;
  private String ContainerCode;
  private long BebanMax;
  private long Payload;
  private long BebanKontainer;
  private long Kapasitas;
    public String getPemilik() {return Pemilik;}
    public void setPemilik(String Pemilik) {this.Pemilik = Pemilik;}
    public String getOwnerCode() {return OwnerCode;}
    public void setOwnerCode(String OwnerCode) {this.OwnerCode = OwnerCode;}
    public String getProductGroup() {return ProductGroup;}
    public void setProductGroup(String ProductGroup) {this.ProductGroup = ProductGroup;}
    public long getRegistration() {return Registration;}
    public void setRegistration(long Registration) {this.Registration = Registration;}
    public long getCheckDigit() {return CheckDigit;}
    public void setCheckDigit(long CheckDigit) {this.CheckDigit = CheckDigit;}
```

```
    public String getSize() {return Size;}
    public void setSize(String Size) {this.Size = Size;}
    public String getContainerCode() {return ContainerCode;}
    public void setContainerCode(String ContainerCode) {this.ContainerCode = ContainerCode;}
    public long getBebanMax() {return BebanMax;}
    public void setBebanMax(long BebanMax) {this.BebanMax = BebanMax;}
    public long getPayload() {return Payload;}
    public void setPayload(long Payload) {this.Payload = Payload;}
    public long getBebanKontainer() {return BebanKontainer;}
    public void setBebanKontainer(long BebanKontainer) {this.BebanKontainer = BebanKontainer;}
    public long getKapasitas() {return Kapasitas;}
   public void setKapasitas(long Kapasitas) {this.Kapasitas = Kapasitas;}
}
```

Selanjutnya, melakukan Assignment Job yaitu proses penunjukan operator yang akan bertugas untuk membawa unit truk dan kontainernya dari lokasi Port asal menuju lokasi Port tujuan sesuai jadwal yang diberikan.



4. Job.java

```
package maindispatchapi.entity;
import java.io.Serializable;
import java.util.Date;
/*@author albani.soeleman*/
public class Job implements Serializable
{
    private Long idJob;
    public Long getIdJob() {return idJob;}
    public void setIdJob(Long idJob) {this.idJob = idJob;}
    private String JobRegister;
    private String Operator;
    private String Truk;
    private String Nopol;
    private String Kontainer;
    private String Pool;
    private String Tujuan;
    private Date TglKirim;
    private String IdNIK;
      public String getJobRegister() {return JobRegister;}
      public void setJobRegister(String JobRegister) {this.JobRegister = JobRegister;}
      public String getOperator() {return Operator;}
      public void setOperator(String Operator) {this.Operator = Operator;}
      public String getTruk() {return Truk;}
      public void setTruk(String Truk) {this.Truk = Truk;}
```

```
    public String getNopol() {return Nopol;}
    public void setNopol(String Nopol) {this.Nopol = Nopol;}
    public String getKontainer() {return Kontainer;}
    public void setKontainer(String Kontainer) {this.Kontainer = Kontainer;}
    public String getPool() {return Pool;}
    public void setPool(String Pool) {this.Pool = Pool;}
    public String getTujuan() {return Tujuan;}
    public void setTujuan(String Tujuan) {this.Tujuan = Tujuan;}
    public Date getTglKirim() {return TglKirim;}
    public void setTglKirim(Date TglKirim) {this.TglKirim = TglKirim;}
    public String getIdNIK(){
        String IdNIK = null;
        return IdNIK;
    }
    public void setIdNIK(String IdNIK) {this.IdNIK = IdNIK;}
}
```

5. JobServices.java

```
package maindispatchapi.Service;
import maindispatchapi.entity.Job;
import java.rmi.Remote;
import java.rmi.RemoteException;
import java.util.List;
/*@author albani.soeleman*/
public interface JobService extends Remote
{
   Job insertJobAssignment(Job job) throws RemoteException;
   void deleteJobAssignment(Long IdJob)throws RemoteException;
   Job getJobAssignment(Long IdJob) throws RemoteException;
    List<Job> getListJob() throws RemoteException;
}
```

6. KontainerService.java

```
package maindispatchapi.Service;
import maindispatchapi.entity.Kontainer;
import java.rmi.Remote;
import java.rmi.RemoteException;
import java.util.List;
/*@author albani.soeleman*/
public interface KontainerService extends Remote
{
  Kontainer insertKontainer(Kontainer kontainer) throws RemoteException;
  void updateKontainer(Kontainer kontainer) throws RemoteException;
  void deleteKontainer(Long IdKTR)throws RemoteException;
   Kontainer getKontainer(Long IdKTR) throws RemoteException;
   List<Kontainer> getKontainer() throws RemoteException;
}
```

7. OperatorService.java

```
package maindispatchapi.Service;
import maindispatchapi.entity.Operator;
import java.rmi.Remote;
import java.rmi.RemoteException;
import java.util.List;
/*@author albani.soeleman */
public interface OperatorService extends Remote
{
    Operator insertOperator(Operator operator) throws RemoteException;
    void updateOperator(Operator operator) throws RemoteException;
    void deleteOperator(Long Id) throws RemoteException;
    Operator getOperator(Long Id) throws RemoteException;
    List<Operator> getOperator()throws RemoteException;
}
```

8. TrukService.java

```
package maindispatchapi.Service;
import maindispatchapi.entity.Truk;
import java.rmi.Remote;
import java.rmi.RemoteException;
import java.util.List;
/*@author albani.soeleman */
public interface TrukService extends Remote
{
   Truk insertTruk(Truk truk) throws RemoteException;
   void updateTruk(Truk truk) throws RemoteException;
   void deleteTruk(int IdTruk)throws RemoteException;
    Truk getTruk(int IdTruk) throws RemoteException;
    List<Truk> getTruk() throws RemoteException;
}
```

B. Kode program Client

1. Assignment_Report.java

```
package maindispatch.client.OperatorForm;
/*@author albani.soeleman*/
public class Assignment_Report extends javax.swing.JInternalFrame
{
   public Assignment_Report() {initComponents();}
   @SuppressWarnings("unchecked")
   // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
  private void initComponents()
 {
    jScrollPane1 = new javax.swing.JScrollPane();
    tblReportJob = new javax.swing.JTable();
    tblReportJob.setModel(new javax.swing.table.DefaultTableModel
   (
        new Object [][]
       {
              {null, null, null, null},
              {null, null, null, null},
              {null, null, null, null},
              {null, null, null, null}
        },
         new String [] {"Title 1", "Title 2", "Title 3", "Title 4"}
     );
      jScrollPane1.setViewportView(tblReportJob);
      javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
      getContentPane().setLayout(layout);
    layout.setHorizontalGroup
   (
       layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
       .addComponent(jScrollPane1, javax.swing.GroupLayout.Alignment.TRAILING,
       javax.swing.GroupLayout.DEFAULT_SIZE, 713, Short.MAX_VALUE)
   );
```

```
 layout.setVerticalGroup
 (
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
       .addGap(0, 0, Short.MAX_VALUE)
       .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 283,
        javax.swing.GroupLayout.PREFERRED_SIZE)
 );
 pack();
}// </editor-fold>//GEN-END:initComponents
   // Variables declaration - do not modify//GEN-BEGIN:variables
      private javax.swing.JScrollPane jScrollPane1;
      private javax.swing.JTable tblReportJob;
   // End of variables declaration//GEN-END:variables
}
```

2. FormOperator.java

```
package maindispatch.client.OperatorForm;
import maindispatch.client.TableModel.TableModelOperator;
import maindispatchapi.entity.Operator;
import maindispatchapi.Service.OperatorService;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;
import java.rmi.RemoteException;
import java.time.Instant;
import java.util.Date;
import java.util.List;
import java.util.Set;
/*@author albani.soeleman */
public class Form extends javax.swing.JFrame {
   private TableModelOperator tmOperator = new TableModelOperator();
   private OperatorService operatorService;
   public Form(OperatorService operatorService) {
      this.operatorService = operatorService;
      try{tmOperator.setData(this.operatorService.getOperator());}
      catch(RemoteException exception){exception.printStackTrace();}
       initComponents();
       tableOperator.setModel(tmOperator);
       tableOperator.getSelectionModel().addListSelectionListener(new ListSelectionListener()
       { @Override
          public void valueChanged(ListSelectionEvent e)
          {
             int row = tableOperator.getSelectedRow();
             if(row != -1){
                Operator operator = tmOperator.get(row);
                txId.setValue(operator.getId());
                txNIK.setText(operator.getNIK());
                txNama.setText(operator.getNama());
                txJenKel.setText(operator.getJenKel());
                txPosisi.setText(operator.getPosisi());
                txSkill.setText(operator.getSkill());
                txSIM.setText(operator.getSIM());
                txExpSIM.setValue(operator.getExpSIM());
             }
          }
       });
   }
@SuppressWarnings("unchecked")
   // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
   private void initComponents() {
      jPanelOp = new javax.swing.JPanel();
      lbNIK = new javax.swing.JLabel();
      lbNama = new javax.swing.JLabel();
      lbJenKel = new javax.swing.JLabel();
      lbPosisi = new javax.swing.JLabel();
```

```
        lbSkill = new javax.swing.JLabel();
        lbSIM = new javax.swing.JLabel();
        lbExpSIM = new javax.swing.JLabel();
        txNama = new javax.swing.JTextField();
        txJenKel = new javax.swing.JTextField();
        txPosisi = new javax.swing.JTextField();
        txSkill = new javax.swing.JTextField();
        txSIM = new javax.swing.JTextField();
        txExpSIM = new javax.swing.JFormattedTextField();
        txId = new javax.swing.JFormattedTextField();
        txNIK = new javax.swing.JTextField();
        jPanel1 = new javax.swing.JPanel();
        btnInsert = new javax.swing.JButton();
        btnUpdate = new javax.swing.JButton();
        btnDelete = new javax.swing.JButton();
        btnRefresh = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        tableOperator = new javax.swing.JTable();
        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
         jPanelOp.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory
         .createEtchedBorder(), "Data Operator"));
        jPanelOp.setName(""); // NOI18N
        lbNIK.setText("NIK :");
        lbNama.setText("Nama :");
        lbJenKel.setText("Jenis Kelamin :");
        lbPosisi.setText("Posisi :");
        lbSkill.setText("Skill :");
        lbSIM.setText("SIM :");
        lbExpSIM.setText("Expired SIM :");
        txExpSIM.setFormatterFactory(new javax.swing.text.DefaultFormatterFactory(new
        javax.swing.text.DateFormatter(new java.text.SimpleDateFormat("dd MMMM YYYY"))));
        txExpSIM.setValue(new java.util.Date());
        txId.setEditable(false);
        txId.setBorder(null);
        txId.setForeground(new java.awt.Color(240, 240, 240));
        txId.setFormatterFactory(new javax.swing.text.DefaultFormatterFactory(new
        javax.swing.text.NumberFormatter(new java.text.DecimalFormat("#0"))));
        txId.setValue(new Long(0L));
```

3. FormJob.java

```
public class FormJob extends javax.swing.JInternalFrame {
    private TableModelOperator tmOperator = new TableModelOperator();
    private OperatorService service;
    private String operator;
    private final TableModelJob tmJob = new TableModelJob();
    private JobService jservice;
    DefaultTableModel dtm;
    Connection conn = null;
    ResultSet rs = null;
    PreparedStatement ps = null;
    public FormJob() {
       this.jservice = jservice;
       this.operator = operator;
        initComponents();
        cbPilihArmada();
        cbPilihKontainer();
        getdatatable();
    }
private void Update_tableOperator(){
try{
DriverManager.getConnection("jdbc:mysql://localhost:3306/dbp2p","root","");
            String sql = "SELECT NIK,Nama,JenKel,Skill,SIM FROM tbloperator";
            ps=conn.prepareStatement(sql);
            rs = ps.executeQuery();
            jtbljob.setModel(DbUtils.resultSetToTableModel(rs));
```

```
        }catch(Exception e){ System.out.println(e); }
    }
public void getdatatable(){
        dtm = new DefaultTableModel();
        jtbljob.setModel(dtm);
        dtm.addColumn("NIK");
        dtm.addColumn("Nama");
        dtm.addColumn("JenKel");
        dtm.addColumn("Skill");
        dtm.addColumn("SIM");
        try{
            Class.forName("com.mysql.jdbc.Driver");
            Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/dbp2p","root","");
            Statement st = conn.createStatement();
            String sql = "SELECT * from tbloperator";
            ResultSet rs = st.executeQuery(sql);
            while(rs.next()){
                Object[] obj = new Object[5];
                obj[0] = rs.getString("NIK");
                obj[1] = rs.getString("Nama");
                obj[2] = rs.getString("JenKel");
                obj[3] = rs.getString("Skill");
                obj[4] = rs.getString("SIM");
                dtm.addRow(obj);
            }
        }catch(SQLException err){
            System.out.println(err);
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(FormJob.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
public void cbPilihArmada() {
        try{
            Class.forName("com.mysql.jdbc.Driver");
            conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/dbp2p","root","");
            String query = "SELECT KodeUnit,NomorPolisi,MerkUnit,Beban_Angkut,StatusUnit FROM tbltruk";
            ps = conn.prepareStatement(query);
            rs = ps.executeQuery();
            cbPilihArmada.addItem("Pilih Armada");
            while (rs.next())
            {
                String data = rs.getString("KodeUnit")+ " - " +rs.getString("NomorPolisi")+ " - "
+rs.getString("StatusUnit");
                cbPilihArmada.addItem(data);
            }
        }catch(Exception e){ System.out.println(e);}
}
public void cbPilihKontainer(){
        try{
            Class.forName("com.mysql.jdbc.Driver");
            Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/dbp2p","root","");
            Statement st = conn.createStatement();
            String query = "SELECT
Pemilik,OwnerCode,ProductGroup,Registration,CheckDigit,Size,Payload,Kapasitas FROM tblkontainer";
            ResultSet rs = st.executeQuery(query);
            cbPilihKontainer.addItem("Pilih Kontainer");
            while (rs.next())
            {
                cbPilihKontainer.addItem(rs.getString("Pemilik")+"-
"+rs.getString("OwnerCode")+rs.getString("ProductGroup")+rs.getString("Registration")+
 rs.getString("CheckDigit")+rs.getString("Size")+"-"+rs.getString("Payload")+"-"+rs.getString("Kapasitas"));    }
            rs.close();
            st.close();
        }catch(ClassNotFoundException | SQLException e){System.out.println(e);}
    }
```

4. FormKontainer.java

```
package maindispatch.client.OperatorForm;
import maindispatch.client.TableModel.TableModelKontainer;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;
import maindispatchapi.entity.Kontainer;
import maindispatchapi.Service.KontainerService;
import java.rmi.RemoteException;
import java.util.List;
import java.util.Set;
public class FormKontainer extends javax.swing.JFrame {
    private TableModelKontainer tmKontainer = new TableModelKontainer();
    private KontainerService kontainerService;
    public FormKontainer(KontainerService kontainerService) {
        this.kontainerService = kontainerService;
        try{
            tmKontainer.setData(this.kontainerService.getKontainer());
        }catch(RemoteException exception){ exception.printStackTrace();}
        initComponents();
        tableKontainer.setModel(tmKontainer);
        tableKontainer.getSelectionModel().addListSelectionListener(new ListSelectionListener(){
            @Override
            public void valueChanged(ListSelectionEvent e){
                int row = tableKontainer.getSelectedRow();
                if(row != -1){
                    Kontainer kontainer = tmKontainer.get(row);
                    txIdKTR.setValue(kontainer.getIdKTR());
                    txPemilik.setText(kontainer.getPemilik());
                    txOwnerCode.setText(kontainer.getOwnerCode());
                    txProductGroup.setText(kontainer.getProductGroup());
                    txRegistration.setValue(kontainer.getRegistration());
                    txCheckDigit.setValue(kontainer.getCheckDigit());
                    txSize.setText(kontainer.getSize());
                    txContainerCode.setText(kontainer.getContainerCode());
                    txBebanMax.setValue(kontainer.getBebanMax());
                    txPayload.setValue(kontainer.getPayload());
                    txBebanKontainer.setValue(kontainer.getBebanKontainer());
                    txKapasitas.setValue(kontainer.getKapasitas());
                }
            }
        });
    }
```

5. FormLogin.java

```
public class FormLogin extends javax.swing.JInternalFrame {
    Connection conn = null;
    ResultSet rs = null;
    Statement st = null;
    public FormLogin() {initComponents();}

    private void initComponents() {
        btnExit = new javax.swing.JButton();
        btnLogin = new javax.swing.JButton();
        lbUser = new javax.swing.JLabel();
        lbPassword = new javax.swing.JLabel();
        pwdPassword = new javax.swing.JPasswordField();
        txtUser = new javax.swing.JTextField();
        setDefaultCloseOperation(javax.swing.WindowConstants.HIDE_ON_CLOSE);
        btnExit.setText("Exit");
        btnExit.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnExitActionPerformed(evt);
            }
        });
```

```
        btnLogin.setText("Login");
        btnLogin.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnLoginActionPerformed(evt);
            }
        });
        lbUser.setText("User Name :");
        lbPassword.setText("Password :");
    }
```

6. FormMain.java

```
    /*@author albani.soeleman*/
    public class FormMain extends javax.swing.JFrame {
        FormLogin frmlogin;
        public FormMain() {
            initComponents();
            this.setLocationRelativeTo(null);
            this.setSize(600, 400);
        }
        @SuppressWarnings("unchecked")
        // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-
    BEGIN:initComponents
        private void initComponents() {
            jMenuItem2 = new javax.swing.JMenuItem();
            dpMain = new javax.swing.JDesktopPane();
            jMenuBar1 = new javax.swing.JMenuBar();
            mFile = new javax.swing.JMenu();
            mLogin = new javax.swing.JMenuItem();
            mExit = new javax.swing.JMenuItem();
            mResources = new javax.swing.JMenu();
            mOperator = new javax.swing.JMenuItem();
            mTruk = new javax.swing.JMenuItem();
            mContainer = new javax.swing.JMenuItem();
            mAssignment = new javax.swing.JMenu();
            mJob = new javax.swing.JMenuItem();
            mLaporan = new javax.swing.JMenu();
            mMonitoring = new javax.swing.JMenu();
            mMap = new javax.swing.JMenuItem();
            jMenuItem2.setText("jMenuItem2");
            setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
            setTitle("Aplikasi Monitoring");
            setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));
            setIconImages(null);
            setLocation(new java.awt.Point(0, 500));
            setResizable(false);
            setType(java.awt.Window.Type.UTILITY);
            javax.swing.GroupLayout dpMainLayout = new javax.swing.GroupLayout(dpMain);
            dpMain.setLayout(dpMainLayout);
            dpMainLayout.setHorizontalGroup(
                dpMainLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGap(0, 924, Short.MAX_VALUE)
            );
            dpMainLayout.setVerticalGroup(
                dpMainLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGap(0, 396, Short.MAX_VALUE)
            );
            getContentPane().add(dpMain, java.awt.BorderLayout.CENTER);
            mFile.setText("File");
            mLogin.setText("Login");
            mLogin.addActionListener(new java.awt.event.ActionListener() {
```

```
    public void actionPerformed(java.awt.event.ActionEvent evt)
     {mLoginActionPerformed(evt);}
});
mFile.add(mLogin);
mExit.setText("Exit");
mExit.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt)
     {mExitActionPerformed(evt);}});
mFile.add(mExit);
jMenuBar1.add(mFile);
mResources.setText("Data");
mResources.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt)
     {mResourcesActionPerformed(evt);}
});
mOperator.setText("Operator");
mOperator.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt)
     {mOperatorActionPerformed(evt);}
});
mResources.add(mOperator);
mTruk.setText("Truk Unit");
mTruk.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt)
     {mTrukActionPerformed(evt);}
});
mResources.add(mTruk);
mContainer.setText("Container");
mContainer.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt)
     {mContainerActionPerformed(evt);}
});
mResources.add(mContainer);
jMenuBar1.add(mResources);
mAssignment.setText("Assignment");
mAssignment.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt)
     {mAssignmentActionPerformed(evt);}
});
mJob.setText("Resources & Job");
mJob.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt)
     {mJobActionPerformed(evt);}
});
mAssignment.add(mJob);
jMenuBar1.add(mAssignment);
mLaporan.setText("Laporan");
jMenuBar1.add(mLaporan);
mMonitoring.setText("Monitoring");
mMonitoring.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt)
     {mMonitoringActionPerformed(evt);}
});
mMap.setText("Map");
mMap.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt)
     {mMapActionPerformed(evt);}
});
```

```
            mMonitoring.add(mMap);
            jMenuBar1.add(mMonitoring);
            setJMenuBar(jMenuBar1);
            pack();
        }// </editor-fold>//GEN-END:initComponents
        // Variables declaration - do not modify//GEN-BEGIN:variables
        // End of variables declaration//GEN-END:variables
}
```

7. FormTruk.java

```
package maindispatch.client.OperatorForm;
import java.awt.Toolkit;
import maindispatch.client.TableModel.TableModelTruk;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;
import maindispatchapi.entity.Truk;
import maindispatchapi.Service.TrukService;
import java.rmi.RemoteException;
import java.util.List;
import java.util.Set;
/*@author albani.soeleman */
public class FormTruk extends javax.swing.JFrame {
    private TableModelTruk tmTruk = new TableModelTruk();
    private TrukService trukService;
    public FormTruk(TrukService trukService) {
        this.trukService = trukService;
        try{tmTruk.setData(this.trukService.getTruk());}
        catch(RemoteException exception){exception.printStackTrace();}
        initComponents();
        tableTruk.setModel(tmTruk);
        tableTruk.getSelectionModel().addListSelectionListener(new ListSelectionListener(){
            @Override
            public void valueChanged(ListSelectionEvent e){
                int row = tableTruk.getSelectedRow();
                if(row != -1){
                    Truk truk = tmTruk.get(row);
                    txIdTruk.setValue(truk.getIdTruk());
                    txKodeUnit.setText(truk.getKodeUnit());
                    txPemilik.setText(truk.getPemilik());
                    txMerkUnit.setText(truk.getMerkUnit());
                    txNopol.setText(truk.getNoPol());
                    txJenisTruk.setText(truk.getJenisTruk());
                    txBeratUnit.setValue(truk.getBeratUnit());
                    txBebanAngkut.setValue(truk.getBebanAngkut());
                    txStatusUnit.setText(truk.getStatusUnit());
                }
            }
        });
    }
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents() {
        jPanelTruk = new javax.swing.JPanel();
        lbKodeUnit = new javax.swing.JLabel();
        lbPemilik = new javax.swing.JLabel();
        lbMerkUnit = new javax.swing.JLabel();
        lbNopol = new javax.swing.JLabel();
        lbJenisTruk = new javax.swing.JLabel();
        lbBeratUnit = new javax.swing.JLabel();
```

```
        lbBebanAngkut = new javax.swing.JLabel();
        txPemilik = new javax.swing.JTextField();
        txMerkUnit = new javax.swing.JTextField();
        txNopol = new javax.swing.JTextField();
        txJenisTruk = new javax.swing.JTextField();
        txIdTruk = new javax.swing.JFormattedTextField();
        txKodeUnit = new javax.swing.JTextField();
        lbStatusUnit = new javax.swing.JLabel();
        txStatusUnit = new javax.swing.JTextField();
        txBeratUnit = new javax.swing.JFormattedTextField();
        txBebanAngkut = new javax.swing.JFormattedTextField();
        jPanelT = new javax.swing.JPanel();
        btnInsert = new javax.swing.JButton();
        btnUpdate = new javax.swing.JButton();
        btnDelete = new javax.swing.JButton();
        btnRefresh = new javax.swing.JButton();
        jScrollPaneT = new javax.swing.JScrollPane();
        tableTruk = new javax.swing.JTable();
        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
jPanelTruk.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory.creat
eEtchedBorder(), "Data Armada Truk"));
        jPanelTruk.setName(""); // NOI18N
        lbKodeUnit.setText("Kode Unit :");
        lbPemilik.setText("Pemilik :");
        lbMerkUnit.setText("Merk Unit :");
        lbNopol.setText("Nomor Polisi :");
        lbJenisTruk.setText("Jenis Truk :");
        lbBeratUnit.setText("Berat Unit :");
        lbBebanAngkut.setText("Beban Angkut :");
        txIdTruk.setEditable(false);
        txIdTruk.setBorder(null);
        txIdTruk.setForeground(new java.awt.Color(240, 240, 240));
        txIdTruk.setFormatterFactory(new javax.swing.text.DefaultFormatterFactory(new
        javax.swing.text.NumberFormatter(new java.text.DecimalFormat("#0"))));
        txIdTruk.setValue(new Long(0L));
        lbStatusUnit.setText("Status Unit :");
        txStatusUnit.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txStatusUnitActionPerformed(evt);}
        });
        tableTruk.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {
                {null, null, null, null},
                {null, null, null, null},
                {null, null, null, null},
                {null, null, null, null}
            },
            new String [] {
                "Title 1", "Title 2", "Title 3", "Title 4"
            }
        ));
    // Variables declaration - do not modify//GEN-BEGIN:variables
    // End of variables declaration//GEN-END:variables
}
```

8. TableModelJob.java

```
package maindispatch.client.TableModel;
import java.util.ArrayList;
import java.util.List;
import javax.swing.table.AbstractTableModel;
import maindispatchapi.entity.Job;
/*@author albani.soeleman */
public class TableModelJob extends AbstractTableModel{
   private List<Job> list = new ArrayList<Job>();
   public TableModelJob(){
   }
   public Job get(int row){
      return list.get(row);
   }
   public void insert(Job job){
      list.add(job);
      fireTableDataChanged();
   }
   public void delete(int row){
      list.remove(row);
      fireTableDataChanged();
   }
   public void setData(List<Job>list){
      this.list = list;
      fireTableDataChanged();
   }
   @Override
   public String getColumnName (int column){
      switch(column){
         case 0: return "IdJob";
         case 1: return "Operator";
         case 2: return "Truk";
         case 3: return "NoPol";
         case 4: return "Kontainer";
         case 5: return "Pool";
         case 6: return "Tujuan";
         case 7: return "Tgl Kirim";
         default: return null;
      }
   }
   @Override
   public int getRowCount(){return list.size();}
   @Override
   public int getColumnCount(){return 9;}
   @Override
   public Object getValueAt(int rowIndex,int colIndex){
      switch(colIndex){
      case 0 : return list.get(rowIndex).getIdJob();
      case 1 : return list.get(rowIndex).getOperator();
      case 2 : return list.get(rowIndex).getTruk();
      case 3 : return list.get(rowIndex).getNopol();
      case 4 : return list.get(rowIndex).getKontainer();
      case 5 : return list.get(rowIndex).getPool();
      case 6 : return list.get(rowIndex).getTujuan();
      case 7 : return list.get(rowIndex).getTglKirim();
      default: return null;
      }
   }}
```

9. TableModelKontainer.java

```
package maindispatch.client.TableModel;
import java.util.ArrayList;
import java.util.List;
import javax.swing.table.AbstractTableModel;
import maindispatchapi.entity.Kontainer;
/*@author albani.soeleman */
public class TableModelKontainer extends AbstractTableModel{
   private List<Kontainer> list = new ArrayList<Kontainer>();
   public TableModelKontainer(){}
   public Kontainer get(int row){return list.get(row);}
   public void insert(Kontainer kontainer){
      list.add(kontainer);
      fireTableDataChanged();
   }
   public void update(int row,Kontainer kontainer){
      list.set(row, kontainer);
      fireTableDataChanged();
   }
   public void delete(int row){
      list.remove(row);
      fireTableDataChanged();
   }
   public void setData(List<Kontainer>list){
      this.list = list;
      fireTableDataChanged();
   }
   @Override
   public String getColumnName(int column){
      switch(column){
         case 0: return "IdKTR";
         case 1: return "Pemilik";
         case 2: return "OwnerCode";
         case 3: return "ProductGroup";
         case 4: return "Registration";
         case 5: return "CheckDigit";
         case 6: return "Size";
         case 7: return "ContainerCode";
         case 8: return "BebanMax";
         case 9: return "Payload";
         case 10: return "BebanKontainer";
         case 11: return "Kapasitas";
         default:  return null;
      }
   }
   @Override
   public int getRowCount() {return list.size();}
   @Override
   public int getColumnCount() {return 12;}
   @Override
   public Object getValueAt(int rowIndex, int colIndex) {
      switch(colIndex){
         case 0 : return list.get(rowIndex).getIdKTR();
         case 1 : return list.get(rowIndex).getPemilik();
         case 2 : return list.get(rowIndex).getOwnerCode();
         case 3 : return list.get(rowIndex).getProductGroup();
         case 4 : return list.get(rowIndex).getRegistration();
         case 5 : return list.get(rowIndex).getCheckDigit();
```

```
            case 6 : return list.get(rowIndex).getSize();
            case 7 : return list.get(rowIndex).getContainerCode();
            case 8 : return list.get(rowIndex).getBebanMax();
            case 9 : return list.get(rowIndex).getPayload();
            case 10 :  return list.get(rowIndex).getBebanKontainer();
            case 11 :  return list.get(rowIndex).getKapasitas();
            default :   return null;
        }
    }
}
```

10. TableModelOperator.java

```
package maindispatch.client.TableModel;
import java.util.ArrayList;
import java.util.List;
import javax.swing.table.AbstractTableModel;
import maindispatchapi.entity.Operator;
public class TableModelOperator extends AbstractTableModel{
    private List<Operator> list = new ArrayList<Operator>();
    public TableModelOperator(){}
    public Operator get(int row){return list.get(row);}
    public void insert(Operator operator){
        list.add(operator);
        fireTableDataChanged();
    }
    public void update(int row, Operator operator){
        list.set(row, operator);
        fireTableDataChanged();
    }
    public void delete(int row){
        list.remove(row);
        fireTableDataChanged();
    }

    public void setData(List<Operator>list){
        this.list = list;
        fireTableDataChanged();
    }
    @Override
    public String getColumnName(int column){
        switch(column){
            case 0: return "Id";
            case 1: return "NIK";
            case 2: return "Nama";
            case 3: return "Jenkel";
            case 4: return "Posisi";
            case 5: return "Skill";
            case 6: return "SIM";
            case 7: return "ExpiredSIM";
            default: return null;
        }
    }
    @Override
    public int getRowCount() {return list.size();}
    @Override
    public int getColumnCount() {return 8;}
    @Override
    public Object getValueAt(int rowIndex, int colIndex) {
        switch(colIndex){
```

```
                case 0 : return list.get(rowIndex).getId();
                case 1 : return list.get(rowIndex).getNIK();
                case 2 : return list.get(rowIndex).getNama();
                case 3 : return list.get(rowIndex).getJenKel();
                case 4 : return list.get(rowIndex).getPosisi();
                case 5 : return list.get(rowIndex).getSkill();
                case 6 : return list.get(rowIndex).getSIM();
                case 7 : return list.get(rowIndex).getExpSIM();
                default : return null;
            }
        }
    }
```

11. TableModelTruk.java

```
package maindispatch.client.TableModel;
import java.util.ArrayList;
import java.util.List;
import javax.swing.table.AbstractTableModel;
import maindispatchapi.entity.Truk;
public class TableModelTruk extends AbstractTableModel{
    private List<Truk> list = new ArrayList<Truk>();
    public TableModelTruk(){}
    public Truk get(int row){ return list.get(row); }
    public void insert(Truk truk){
        list.add(truk);
        fireTableDataChanged();
    }
    public void update(int row,Truk truk){
        list.set(row, truk);
        fireTableDataChanged();
    }
    public void delete(int row){
        list.remove(row);
        fireTableDataChanged();}
    public void setData(List<Truk>list){
        this.list = list;
        fireTableDataChanged();
    }
    @Override
    public String getColumnName(int column){
        switch(column){
            case 0: return "IdTruk";
            case 1: return "KodeUnit";
            case 2: return "Pemilik";
            case 3: return "Merk Unit";
            case 4: return "Nomor Polisi";
            case 5: return "Jenis Truk";
            case 6: return "Berat Unit";
            case 7: return "Beban Angkut";
            case 8: return "Status Unit";
            default: return null;
        }
    }
    @Override
    public int getRowCount() {return list.size();}
    @Override
    public int getColumnCount() {return 9;}
    @Override
    public Object getValueAt(int rowIndex, int colIndex) {
```

```
        switch(colIndex){
            case 0 : return list.get(rowIndex).getIdTruk();
            case 1 : return list.get(rowIndex).getKodeUnit();
            case 2 : return list.get(rowIndex).getPemilik();
            case 3 : return list.get(rowIndex).getMerkUnit();
            case 4 : return list.get(rowIndex).getNoPol();
            case 5 : return list.get(rowIndex).getJenisTruk();
            case 6 : return list.get(rowIndex).getBeratUnit();
            case 7 : return list.get(rowIndex).getBebanAngkut();
            case 8 : return list.get(rowIndex).getStatusUnit();
            default :return null;
        }
    }
}
```

C. Kode program Server

1. JobServiceServer.java

```
package maindispatch.server.Service;
import maindispatch.server.utilities.DBUtilities;
public class JobServiceServer extends UnicastRemoteObject implements JobService{
    public JobServiceServer() throws RemoteException {}
    @Override
    public Job insertJobAssignment(Job job) throws RemoteException {
        System.out.println("Client melakukan insert");
        PreparedStatement st = null;
        try{
            st = DBUtilities.getConnection().prepareStatement(
                "INSERT INTO tbljobassignment (idJob, Operator, Truk, NoPol, Kontainer, Pool,
Tujuan, TglKirim, IdNIK) values ( null, ?, ?, ?, ?, ?, ?, ?,
?)",Statement.RETURN_GENERATED_KEYS );
            st.setString(1, job.getOperator());
            st.setString(2, job.getTruk());
            st.setString(3, job.getNopol());
            st.setString(4, job.getKontainer());
            st.setString(5, job.getPool());
            st.setString(6, job.getTujuan());
            st.setDate(7, new Date(job.getTglKirim().getTime()));
            st.setString(8, job.getIdNIK());
            st.executeUpdate();
            try (ResultSet rs = st.getGeneratedKeys()) {
                if(rs.next()){
                    job.setIdJob(rs.getLong(1));
                }
            }
            return job;
        }catch(SQLException exception){
            exception.printStackTrace();
            return null;
        }finally{
            if(st != null){
                try{
                    st.close();
                }catch(SQLException exception){  }
            }
        }
    }
    @Override
    public void deleteJobAssignment(Long IdJob) throws RemoteException {
        System.out.println("Client melakukan delete");
```

```
    PreparedStatement st = null;
    try{
        st = DBUtilities.getConnection().prepareStatement(
            "DELETE FROM tbljobassignment WHERE IdJob = ?");
        st.setLong(1, IdJob);
        st.executeUpdate();
    }catch(SQLException exception){ exception.printStackTrace();}
   finally{
        if(st != null){
            try{
                st.close();
            }catch(SQLException exception){ exception.printStackTrace();}
        }
    }
 }
@Override
 public void pilihOperator() throws RemoteException{
    PreparedStatement st = null;
    try{
        st = DBUtilities.getConnection().prepareStatement(
                "SELECT NIK,Nama from tbloperator"
        );
        ResultSet rs = st.executeQuery();
        while (rs.next()){
            Operator operator = new Operator();
            operator.setNIK(rs.getString("NIK"));
            operator.setNama(rs.getString("Nama"));
        }
        rs.close();
    }catch(SQLException exception){
        exception.printStackTrace();
    }
 }
 @Override
 public Job getJobAssignment(Long IdJob) throws RemoteException {
        System.out.println("Client mengambil data berdasarkan IdJob");
        PreparedStatement st = null;
        try{
            st = DBUtilities.getConnection().prepareStatement(
                "SELECT * FROM tbljob WHERE IdJob = ?"
                );
            st.setLong(1, IdJob);
            ResultSet rs = st.executeQuery();
            Job job = null;
            if(rs.next()){
                job = new Job();
                job.setIdJob(rs.getLong("IdJob"));
                job.setOperator(rs.getString("Operator"));
                job.setTruk(rs.getString("Truk"));
                job.setNopol(rs.getString("NoPol"));
                job.setKontainer(rs.getString("Kontainer"));
                job.setPool(rs.getString("Pool"));
                job.setTujuan(rs.getString("Tujuan"));
                job.setTglKirim(rs.getDate("TglKirim"));
               //job.setIdNIK(rs.getString("IdNIK"));
            }
            rs.close();
            return job;
```

```
            }catch(SQLException exception){
                exception.printStackTrace();
                return null;
            }finally{
                if (st != null){
                    try{
                        st.close();
                    }catch(SQLException exception){
                        exception.printStackTrace();
                    }
                }
            }
        }
        public List<Job> getListJob() throws RemoteException {
            System.out.println("Client menampilkan semua data");
            Statement st = null;
            try{
                st = DBUtilities.getConnection().createStatement();
                ResultSet rs = st.executeQuery("SELECT * FROM tbljob");
                List<Job> list = new ArrayList<Job>();
                while(rs.next()){
                    Job job = new Job();
                        job.setIdJob(rs.getLong("IdJob"));
                        job.setOperator(rs.getString("Operator"));
                        job.setTruk(rs.getString("Truk"));
                        job.setNopol(rs.getString("NoPol"));
                        job.setKontainer(rs.getString("Kontainer"));
                        job.setPool(rs.getString("Pool"));
                        job.setTujuan(rs.getString("Tujuan"));
                        job.setTglKirim(rs.getDate("TglKirim"));
                    list.add(job);
                }
                rs.close();
                return list;
            }catch(SQLException exception){
                exception.printStackTrace();
                return null;
            }finally{
                if(st != null){
                    try{
                        st.close();
                    }catch (SQLException exception){
                        exception.printStackTrace();
                    }
                }
            }
        }
    }
```

2. **KontainerServiceServer.java**

```
    package maindispatch.server.Service;
    import maindispatch.server.utilities.DBUtilities;
    public class KontainerServiceServer extends UnicastRemoteObject implements KontainerService{
        public KontainerServiceServer() throws RemoteException {}
        @Override
        public Kontainer insertKontainer(Kontainer kontainer) throws RemoteException {
            System.out.println("Client melakukan insert");
            PreparedStatement st = null;
            try{
```

```
        st = DBUtilities.getConnection().prepareStatement(
            "INSERT INTO tblkontainer (IdKTR,Pemilik, OwnerCode, ProductGroup, "
            + "Registration, CheckDigit, Size, ContainerCode, BebanMax, Payload,
BebanKontainer, Kapasitas) "
            + "values ( null, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)",Statement.RETURN_GENERATED_KEYS
);
        st.setString(1, kontainer.getPemilik());
        st.setString(2, kontainer.getOwnerCode());
        st.setString(3, kontainer.getProductGroup());
        st.setLong(4, kontainer.getRegistration());
        st.setLong(5, kontainer.getCheckDigit());
        st.setString(6, kontainer.getSize());
        st.setString(7,
kontainer.getOwnerCode()+kontainer.getProductGroup()+kontainer.getRegistration()+
        kontainer.getCheckDigit()+kontainer.getSize());
        st.setLong(8, kontainer.getBebanMax());
        st.setLong(9, kontainer.getPayload());
        st.setLong(10, kontainer.getBebanKontainer());
        st.setLong(11, kontainer.getKapasitas());
        st.executeUpdate();
        ResultSet rs = st.getGeneratedKeys();
        if(rs.next()){ kontainer.setIdKTR(rs.getLong(1)); }
        rs.close();
        return kontainer;
      }catch(SQLException exception){
        exception.printStackTrace();
        return null;
      }finally{
        if(st != null){
          try{
            st.close();
          }catch(SQLException exception){
          }
        }
      }
    }
    @Override
    public void updateKontainer(Kontainer kontainer) throws RemoteException {
      System.out.println("Client melakukan update");
      PreparedStatement st = null;
      try{
        st = DBUtilities.getConnection().prepareStatement(
            "UPDATE tblkontainer SET Pemilik = ?, OwnerCode = ?, ProductGroup = ?,
Registration = ?" +
            ", CheckDigit = ?, Size = ?, ContainerCode = ?, BebanMax = ?, Payload = ?,
BebanKontainer = ?, Kapasitas = ?" +
            "WHERE IdKTR = ?"
        );
        st.setString(1, kontainer.getPemilik());
        st.setString(2, kontainer.getOwnerCode());
        st.setString(3, kontainer.getProductGroup());
        st.setLong(4, kontainer.getRegistration());
        st.setLong(5, kontainer.getCheckDigit());
        st.setString(6, kontainer.getSize());
        st.setString(7,
kontainer.getOwnerCode()+kontainer.getProductGroup()+kontainer.getRegistration()+
        kontainer.getCheckDigit()+kontainer.getSize());
        st.setLong(8, kontainer.getBebanMax());
```

```
            st.setLong(9, kontainer.getPayload());
            st.setLong(10, kontainer.getBebanKontainer());
            st.setLong(11, kontainer.getKapasitas());;
            st.setLong(12, kontainer.getIdKTR());
            st.executeUpdate();
        }catch(SQLException exception){exception.printStackTrace();}
finally{
            if(st != null){
                try{
                    st.close();
                }catch (SQLException exception){ exception.printStackTrace();}
            }
        }
    }
    @Override
    public void deleteKontainer(Long IdKTR) throws RemoteException {
        System.out.println("Client melakukan delete");
        PreparedStatement st = null;
        try{
            st = DBUtilities.getConnection().prepareStatement(
                "DELETE FROM tblkontainer WHERE IdKTR = ?");
            st.setLong(1, IdKTR);
            st.executeUpdate();
        }catch(SQLException exception){ exception.printStackTrace();}
finally{
            if(st != null){
                try{
                    st.close();
                }catch(SQLException exception){exception.printStackTrace();}
            }
        }
    }
    @Override
    public Kontainer getKontainer(Long IdKTR) throws RemoteException {
            System.out.println("Client mengambil data berdasarkan Container Code");
            PreparedStatement st = null;
            try{
                st = DBUtilities.getConnection().prepareStatement(
                    "SELECT * FROM tblkontainer WHERE IdKTR = ?"
                    );
                ResultSet rs = st.executeQuery();
                Kontainer kontainer = null;
                if(rs.next()){
                    kontainer = new Kontainer();
                    kontainer.setIdKTR(rs.getLong("IdKTR"));
                    kontainer.setPemilik(rs.getString("Pemilik"));
                    kontainer.setOwnerCode(rs.getString("OwnerCode"));
                    kontainer.setProductGroup(rs.getString("ProductGroup"));
                    kontainer.setRegistration(rs.getLong("Registration"));
                    kontainer.setCheckDigit(rs.getLong("CheckDigit"));
                    kontainer.setSize(rs.getString("Size"));
                    kontainer.setContainerCode(rs.getString("ContainerCode"));
                    kontainer.setBebanMax(rs.getLong("BebanMax"));
                    kontainer.setPayload(rs.getLong("Payload"));
                    kontainer.setBebanKontainer(rs.getLong("BebanKontainer"));
                    kontainer.setKapasitas(rs.getLong("Kapasitas"));
                }
                rs.close();
```

```
            return kontainer;
        }catch(SQLException exception){
            exception.printStackTrace();
            return null;
        }finally{
            if (st != null){
                try{
                    st.close();
                }catch(SQLException exception){exception.printStackTrace();}
            }
        }
    }


    @Override
    public List<Kontainer> getKontainer() throws RemoteException {
        System.out.println("Client menampilkan semua data");
        Statement st = null;
        try{
            st = DBUtilities.getConnection().createStatement();
            ResultSet rs = st.executeQuery("SELECT * FROM tblkontainer");
            List<Kontainer> list = new ArrayList<Kontainer>();
            while(rs.next()){
                Kontainer kontainer = new Kontainer();
                    kontainer.setIdKTR(rs.getLong("IdKTR"));
                    kontainer.setPemilik(rs.getString("Pemilik"));
                    kontainer.setOwnerCode(rs.getString("OwnerCode"));
                    kontainer.setProductGroup(rs.getString("ProductGroup"));
                    kontainer.setRegistration(rs.getLong("Registration"));
                    kontainer.setCheckDigit(rs.getLong("CheckDigit"));
                    kontainer.setSize(rs.getString("Size"));
                    kontainer.setContainerCode(rs.getString("ContainerCode"));
                    kontainer.setBebanMax(rs.getLong("BebanMax"));
                    kontainer.setPayload(rs.getLong("Payload"));
                    kontainer.setBebanKontainer(rs.getLong("BebanKontainer"));
                    kontainer.setKapasitas(rs.getLong("Kapasitas"));
                list.add(kontainer);
            }
            rs.close();
            return list;
        }catch(SQLException exception){
            exception.printStackTrace();
            return null;
        }finally{
            if(st != null){
                try{
                    st.close();
                }catch (SQLException exception){ exception.printStackTrace(); }
            }
        }
    }
}
```

3. OperatorServiceServer.java

```
package maindispatch.server.Service;
import maindispatch.server.utilities.DBUtilities;
public class OperatorServiceServer extends UnicastRemoteObject implements OperatorService{
   public OperatorServiceServer() throws RemoteException {
   }
   @Override
   public Operator insertOperator(Operator operator) throws RemoteException {
      System.out.println("Client melakukan insert");
      PreparedStatement st = null;
      try{
         st = DBUtilities.getConnection().prepareStatement(
             "INSERT INTO tbloperator (Id, NIK, Nama, JenKel, Posisi, Skill, SIM, ExpiredSIM)
values ( null, ?, ?, ?, ?, ?, ?, ?)",Statement.RETURN_GENERATED_KEYS );
         st.setString(1, operator.getNIK());
         st.setString(2, operator.getNama());
         st.setString(3, operator.getJenKel());
         st.setString(4, operator.getPosisi());
         st.setString(5, operator.getSkill());
         st.setString(6, operator.getSIM());
         st.setDate(7, new Date(operator.getExpSIM().getTime()));
         st.executeUpdate();
         ResultSet rs = st.getGeneratedKeys();
         if(rs.next()){operator.setId(rs.getLong(1)); }
         rs.close();
         return operator;
      }catch(SQLException exception){
         exception.printStackTrace();
         return null;
      }finally{
         if(st != null){
            try{
               st.close();
            }catch(SQLException exception){ }
         }
      }
   }
   @Override
   public void updateOperator(Operator operator) throws RemoteException {
      System.out.println("Client melakukan update");
      PreparedStatement st = null;
      try{
         st = DBUtilities.getConnection().prepareStatement(
             "UPDATE tbloperator SET NIK = ?, Nama = ?" +
             ", JenKel = ?, Posisi = ?, Skill = ?, SIM = ?, ExpiredSIM = ?" +
             "WHERE Id = ?"
         );
         st.setString(1, operator.getNIK());
         st.setString(2, operator.getNama());
         st.setString(3, operator.getJenKel());
         st.setString(4, operator.getPosisi());
         st.setString(5, operator.getSkill());
         st.setString(6, operator.getSIM());
         st.setDate(7, new Date(operator.getExpSIM().getTime()));
         st.setLong(8, operator.getId());
         st.executeUpdate();
      }catch(SQLException exception){ exception.printStackTrace();}
    finally{
```

```
        if(st != null){
            try{
                st.close();
            }catch (SQLException exception){ exception.printStackTrace();}
        }
    }
}
@Override
public void deleteOperator(Long Id) throws RemoteException {
    System.out.println("Client melakukan delete");
    PreparedStatement st = null;
    try{
        st = DBUtilities.getConnection().prepareStatement(
            "DELETE FROM tbloperator WHERE Id = ?");
        st.setLong(1, Id);
        st.executeUpdate();
    }catch(SQLException exception){exception.printStackTrace();}
    finally{
        if(st != null){
            try{
                st.close();
            }catch(SQLException exception){ exception.printStackTrace();}
        }
    }
}
@Override
public Operator getOperator(Long Id) throws RemoteException {
        System.out.println("Client mengambil data berdasarkan NIK");
        PreparedStatement st = null;
        try{
            st = DBUtilities.getConnection().prepareStatement(
                "SELECT * FROM tbloperator WHERE Id = ?"
                );
            ResultSet rs = st.executeQuery();
            Operator operator = null;
            if(rs.next()){
                operator = new Operator();
                operator.setId(rs.getLong("Id"));
                operator.setNIK(rs.getString("NIK"));
                operator.setNama(rs.getString("Nama"));
                operator.setJenKel(rs.getString("JenKel"));
                operator.setPosisi(rs.getString("Posisi"));
                operator.setSkill(rs.getString("Skill"));
                operator.setSIM(rs.getString("SIM"));
                operator.setExpSIM(rs.getDate("ExpiredSIM"));
            }
            rs.close();
            return operator;
        }catch(SQLException exception){
            exception.printStackTrace();
            return null;
        }finally{
            if (st != null){
                try{
                    st.close();
                }catch(SQLException exception){ exception.printStackTrace();}
            }
        }
```

```
        }
        @Override
        public List<Operator> getOperator() throws RemoteException {
            System.out.println("Client menampilkan semua data");
            Statement st = null;
            try{
                st = DBUtilities.getConnection().createStatement();
                ResultSet rs = st.executeQuery("SELECT * FROM tbloperator");
                List<Operator> list = new ArrayList<Operator>();
                while(rs.next()){
                    Operator operator = new Operator();
                    operator.setId(rs.getLong("Id"));
                    operator.setNIK(rs.getString("NIK"));
                    operator.setNama(rs.getString("Nama"));
                    operator.setJenKel(rs.getString("JenKel"));
                    operator.setPosisi(rs.getString("Posisi"));
                    operator.setSkill(rs.getString("Skill"));
                    operator.setSIM(rs.getString("SIM"));
                    operator.setExpSIM(rs.getDate("ExpiredSIM"));
                    list.add(operator);
                }
                rs.close();
                return list;
            }catch(SQLException exception){
                exception.printStackTrace();
                return null;
            }finally{
                if(st != null){
                    try{
                        st.close();
                    }catch (SQLException exception){ exception.printStackTrace();}
                }
            }
        }
    }
```

4. TrukServiceServer.java

```
package maindispatch.server.Service;
import maindispatch.server.utilities.DBUtilities;
public class TrukServiceServer extends UnicastRemoteObject implements TrukService{
    public TrukServiceServer() throws RemoteException {  }
    @Override
    public Truk insertTruk(Truk truk) throws RemoteException {
        System.out.println("Client melakukan insert");
        PreparedStatement st = null;
        try{
            st = DBUtilities.getConnection().prepareStatement(
                "INSERT INTO tbltruk (IdTruk, KodeUnit, Pemilik, MerkUnit, NomorPolisi, JenisTruk, BeratUnit,
Beban_Angkut, StatusUnit) values ( null, ?, ?, ?, ?, ?, ?, ?, ?)",Statement.RETURN_GENERATED_KEYS );
            st.setString(1, truk.getKodeUnit());
            st.setString(2, truk.getPemilik());
            st.setString(3, truk.getMerkUnit());
            st.setString(4, truk.getNoPol());
            st.setString(5, truk.getJenisTruk());
            st.setLong(6, truk.getBeratUnit());
            st.setLong(7, truk.getBebanAngkut());
            st.setString(8, truk.getStatusUnit());
            st.executeUpdate();
            ResultSet rs = st.getGeneratedKeys();
            if(rs.next()){truk.setIdTruk(rs.getInt(1)); }
            rs.close();
            return truk;
```

```
    }catch(SQLException exception){
       exception.printStackTrace();
       return null;
    }finally{
       if(st != null){
          try{
             st.close();
          }catch(SQLException exception){ }
       }
    }
 }
 @Override
 public void updateTruk(Truk truk) throws RemoteException {
    System.out.println("Client melakukan update");
    PreparedStatement st = null;
    try{
       st = DBUtilities.getConnection().prepareStatement(
          "UPDATE tbltruk SET KodeUnit = ?, Pemilik = ?, MerkUnit = ?, NomorPolisi = ?" +
          ", JenisTruk = ?, BeratUnit = ?, Beban_Angkut = ?, StatusUnit = ?" +
          "WHERE IdTruk = ?"
       );
       st.setString(1, truk.getKodeUnit());
       st.setString(2, truk.getPemilik());
       st.setString(3, truk.getMerkUnit());
       st.setString(4, truk.getNoPol());
       st.setString(5, truk.getJenisTruk());
       st.setLong(6, truk.getBeratUnit());
       st.setLong(7,truk.getBebanAngkut());
       st.setString(8, truk.getStatusUnit());
       st.setInt(9, truk.getIdTruk());
       st.executeUpdate();
    }catch(SQLException exception){ exception.printStackTrace();}
finally{
       if(st != null){
          try{
             st.close();
          }catch (SQLException exception){ exception.printStackTrace();}
       }
    }
 }
 @Override
 public void deleteTruk(int IdTruk) throws RemoteException {
    System.out.println("Client melakukan delete");
    PreparedStatement st = null;
    try{
       st = DBUtilities.getConnection().prepareStatement(
          "DELETE FROM tbltruk WHERE IdTruk = ?");
       st.setInt(1, IdTruk);
       st.executeUpdate();
    }catch(SQLException exception){ exception.printStackTrace();}
finally{
       if(st != null){
          try{
             st.close();
          }catch(SQLException exception){ exception.printStackTrace();}
       }
    }
 }
 @Override
 public Truk getTruk(int IdTruk) throws RemoteException {
       System.out.println("Client mengambil data berdasarkan KodeUnit");
       PreparedStatement st = null;

       try{
          st = DBUtilities.getConnection().prepareStatement("SELECT * FROM tbltruk WHERE IdTruk = ?" );
```

```
            ResultSet rs = st.executeQuery();
            Truk truk = null;
            if(rs.next()){
               truk = new Truk();
               truk.setIdTruk(rs.getInt("IdTruk"));
               truk.setKodeUnit(rs.getString("KodeUnit"));
               truk.setPemilik(rs.getString("Pemilik"));
               truk.setMerkUnit(rs.getString("MerkUnit"));
               truk.setNoPol(rs.getString("NomorPolisi"));
               truk.setJenisTruk(rs.getString("JenisTruk"));
               truk.setBeratUnit(rs.getLong("BeratUnit"));
               truk.setBebanAngkut(rs.getLong("Beban_Angkut"));
               truk.setStatusUnit(rs.getString("StatusUnit"));
            }
            rs.close();
            return truk;
         }catch(SQLException exception){
            exception.printStackTrace();
            return null;
         }finally{
            if (st != null){
               try{
                  st.close();
               }catch(SQLException exception){ exception.printStackTrace();}
            }
         }
      }
      @Override
      public List<Truk> getTruk() throws RemoteException {
         System.out.println("Client menampilkan semua data");
         Statement st = null;
         try{
            st = DBUtilities.getConnection().createStatement();
            ResultSet rs = st.executeQuery("SELECT * FROM tbltruk");
            List<Truk> list = new ArrayList<Truk>();
            while(rs.next()){
               Truk truk = new Truk();
                  truk.setIdTruk(rs.getInt("IdTruk"));
                  truk.setKodeUnit(rs.getString("KodeUnit"));
                  truk.setPemilik(rs.getString("Pemilik"));
                  truk.setMerkUnit(rs.getString("MerkUnit"));
                  truk.setNoPol(rs.getString("NomorPolisi"));
                  truk.setJenisTruk(rs.getString("JenisTruk"));
                  truk.setBeratUnit(rs.getLong("BeratUnit"));
                  truk.setBebanAngkut(rs.getLong("Beban_Angkut"));
                  truk.setStatusUnit(rs.getString("StatusUnit"));
               list.add(truk);
            }
            rs.close();
            return list;
         }catch(SQLException exception){
            exception.printStackTrace();
            return null;
         }finally{
            if(st != null){
               try{
                  st.close();
               }catch (SQLException exception){ exception.printStackTrace();}
            }
         }
      }
   }
```

5. DBUtilities.java

```
package maindispatch.server.utilities;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
public class DBUtilities {
    private static Connection connection;
    public static Connection getConnection(){
        if(connection == null){
            try {
                DriverManager.registerDriver(new com.mysql.jdbc.Driver());
                connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/dbp2p","root","");
            } catch (SQLException ex) {
                Logger.getLogger(DBUtilities.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
        return connection;
    }
}
```

6. MainDispatchServer.java

```
import maindispatch.server.utilities.DBUtilities;
import java.rmi.RemoteException;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import maindispatch.server.Service.OperatorServiceServer;
import maindispatch.server.Service.TrukServiceServer;
import maindispatch.server.Service.KontainerServiceServer;
import maindispatch.server.Service.JobServiceServer;
public class MainDispatchServer {
    public static void main(String[] args) throws RemoteException {
        DBUtilities.getConnection();
        Registry server = LocateRegistry.createRegistry(6789);//port default RMI (1099)
        OperatorServiceServer operatorService = new OperatorServiceServer();
        TrukServiceServer trukService = new TrukServiceServer();
        KontainerServiceServer kontainerService = new KontainerServiceServer();
        JobServiceServer jobService = new JobServiceServer();
        server.rebind("service", operatorService);
        server.rebind("tservice", trukService);
        server.rebind("kservice", kontainerService);
        server.rebind("jservice", jobService);
        System.out.println("Server Berhasil");
    }
}
```

II. Source Code perangkat IoT
1.Modul RFID

```
#include <SPI.h>
#include <MFRC522.h>
#define RST_PIN         9
#define SS_PIN          10

MFRC522 mfrc522(SS_PIN, RST_PIN);
void setup() {
  Serial.begin(9600);
  SPI.begin();
  mfrc522.PCD_Init();
  Serial.println(F("Read personal data on a MIFARE PICC:"));
}
void loop() {
  MFRC522::MIFARE_Key key;
  for (byte i = 0; i < 6; i++) key.keyByte[i] = 0xFF;
  byte block;
  byte len;
  MFRC522::StatusCode status;
  if ( ! mfrc522.PICC_IsNewCardPresent()) {
    return;
  }
  if ( ! mfrc522.PICC_ReadCardSerial()) {
    return;
  }
  Serial.println(F("**Card Detected:**"));
  mfrc522.PICC_DumpDetailsToSerial(&(mfrc522.uid));
  Serial.print(F("Name: "));
  byte buffer1[18];
  block = 4;
  len = 18;
  status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, 4, &key,
&(mfrc522.uid));
  if (status != MFRC522::STATUS_OK) {
    Serial.print(F("Authentication failed: "));
    Serial.println(mfrc522.GetStatusCodeName(status));
    return;
  }
  status = mfrc522.MIFARE_Read(block, buffer1, &len);
  if (status != MFRC522::STATUS_OK) {
    Serial.print(F("Reading failed: "));
    Serial.println(mfrc522.GetStatusCodeName(status));
    return;
  }
  for (uint8_t i = 0; i < 16; i++)
  {
    if (buffer1[i] != 32)
    { Serial.write(buffer1[i]); }
  }
  Serial.print(" ");
  byte buffer2[18];
  block = 1;
  status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, 1, &key,
&(mfrc522.uid));
  if (status != MFRC522::STATUS_OK) {
    Serial.print(F("Authentication failed: "));
    Serial.println(mfrc522.GetStatusCodeName(status));
```

```
    return;
  }
  status = mfrc522.MIFARE_Read(block, buffer2, &len);
  if (status != MFRC522::STATUS_OK) {
    Serial.print(F("Reading failed: "));
    Serial.println(mfrc522.GetStatusCodeName(status));
    return;
  }
  for (uint8_t i = 0; i < 16; i++) {
    Serial.write(buffer2[i] );
  }
  Serial.println(F("\n**End Reading**\n"));
  delay(1000); //change value if you want to read cards faster
  mfrc522.PICC_HaltA();
  mfrc522.PCD_StopCrypto1();
}
int solenoidPin = 9;
void setup()
{ pinMode(solenoidPin, OUTPUT); }
void loop()
{
  digitalWrite(solenoidPin, HIGH);
  delay(1000);
  digitalWrite(solenoidPin, LOW);
  delay(1000);
}
```

3. Modul GPS

```
#include <SoftwareSerial.h>
#define DEBUG true
SoftwareSerial sim808(2,3);
void setup()
{
  Serial.begin(9600);
 sim808.begin(9600);
}
void loop()
{
   getgps();
   while(1)
   {
       sendData( "AT+CGNSINF",1000,DEBUG);
       delay(1000);
   }
}
void getgps(void)
{
   sendData( "AT+CGNSPWR=1",1000,DEBUG);
   sendData( "AT+CGNSSEQ=RMC",1000,DEBUG);
}
String sendData(String command, const int timeout, boolean debug)
{
    String response = "";
    sim808.println(command);
    long int time = millis();
    while( (time+timeout) > millis())
    {
      while(sim808.available())
      {
```

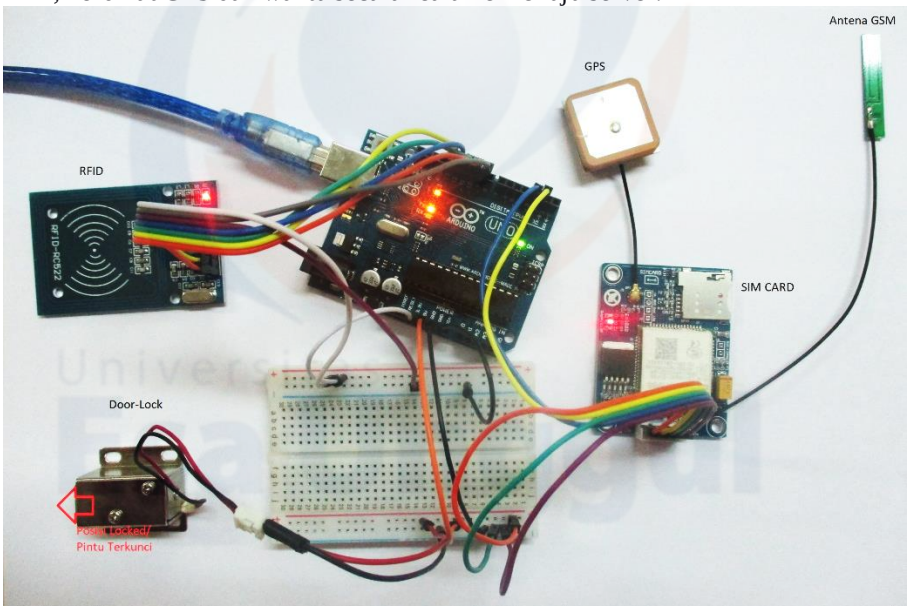
```
      char c = sim808.read();
      response+=c;
    }
  }
  if(debug)
  { Serial.print(response);}
  return response;
}
```

3.Modul kunci gembok (door-lock)

```
int solenoidPin = 9;
void setup()
{  pinMode(solenoidPin, OUTPUT);  }
void loop()
{
 digitalWrite(solenoidPin, HIGH);
 delay(1000);
 digitalWrite(solenoidPin, LOW);
 delay(1000);
}
```

Hasil uji coba IOT Perangkat

Kunci akan terbuka jika operator truk melakukan tapping kartu ke RFID
Kemudian, melalui koneksi GSM , perangkat akan mengirimkan data yang berisi status door-lock, status RFID, kordinat GPS dan waktu secara realtime menuju server.



TUGAS AKHIR

NAMA : Albani Soeleman
NIM : 20160801019

SISTEM MONITORING KEAMANAN EKSPEDISI METODE PORT-TO-PORT BERBASIS SISTEM INFORMASI GEOGRAFIS

2019