LAMPIRAN

- **Source Packages**

```
<defaul package>
/*
 * To change this template, choose Tools | Templates
 * and open the template in  the editor.
 */
import java.util.ArrayList;
import java.util.List;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;
/**
 *
 * @author bimMargera
 */
@ManagedBean
@RequestScoped
public class ImageSwitchBean {
private List<String> images;
     public ImageSwitchBean() {
        images = new ArrayList<String>();
        images.add("iniesta.jpg");
        images.add("leo.jpg");
        images.add("tiago.jpg");
        images.add("xavi.jpg");
     }
     public List<String> getImages() {
        return getfloated();
     }
   /**
    * @return the images
    */
   public List<String> getfloated() {
      return images;
   }
   /**
    * @param images the images to set
    */
   public void setfloated(List<String> images) {
      this.images = images;
   }
  }
```

```
    /**
     * Creates a new instance of ImageSwitchBean
     */
```

- **Database.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package Database;
import java.sql.Connection;
import java.sql.DriverManager;
/**
 *
 * @author Fuck
 */
public class Database {
    private  Connection con;
    public static Connection getConnection() {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/sp_hormon",
                    "root", "");
            return con;
        } catch (Exception ex) {
            System.out.println("Database.getConnection() Error -->" +
ex.getMessage());
            return null;
        }
    }
    public static void close(Connection con) {
        try {
            con.close();
        } catch (Exception ex) {
        }
    }
}
```

- **AbtractController.java**

```
package beans;
import control.AbstractFacade;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.faces.event.ActionEvent;
```

```
import javax.faces.model.SelectItem;
import javax.ejb.EJB;
/**
 * Represents an abstract shell of to be used as JSF Controller to be used in
 * AJAX-enabled applications. No outcomes will be generated from its
methods
 * since handling is designed to be done inside one page.
 */
public abstract class AbstractController<T> {
    private AbstractFacade<T> ejbFacade;
    private Class<T> itemClass;
    private T selected;
    private List<T> items;
    public AbstractController() {
    }
    public AbstractController(Class<T> itemClass) {
        this.itemClass = itemClass;
    }
    protected AbstractFacade<T> getFacade() {
        return ejbFacade;
    }
    protected void setFacade(AbstractFacade<T> ejbFacade) {
        this.ejbFacade = ejbFacade;
    }
    public T getSelected() {
        return selected;
    }
    public void setSelected(T selected) {
        this.selected = selected;
    }
    protected void setEmbeddableKeys() {
        // Nothing to do if entity does not have any embeddable key.
    };
    protected void initializeEmbeddableKey() {
        // Nothing to do if entity does not have any embeddable key.
    }
    /**
     * Returns all items in a List object *
     * @return
     */
    public List<T> getItems() {
        if (items == null) {
            items = this.ejbFacade.findAll();
        }
```

```
        return items;
    }
    public void save(ActionEvent event) {
        if (selected != null) {
            this.setEmbeddableKeys();
            this.ejbFacade.edit(selected);
        }
    }
    public void saveNew(ActionEvent event) {
        save(event);
        items = null; // Invalidate list of items to trigger re-query.
    }
    public void delete(ActionEvent event) {
        if (selected != null) {
            this.ejbFacade.remove(selected);
            selected = null; // Remove selection
            items = null; // Invalidate list of items to trigger re-query.
        }
    }
    /**
     * Creates a new instance of an underlying entity and assigns it to Selected
     * property.
     *
     * @return
     */
    public T prepareCreate(ActionEvent event) {
        T newItem;
        try {
            newItem = itemClass.newInstance();
            this.selected = newItem;
            initializeEmbeddableKey();
            return newItem;
        } catch (InstantiationException ex) {
Logger.getLogger(AbstractController.class.getName()).log(Level.SEVERE,
null, ex);
        } catch (IllegalAccessException ex) {
    Logger.getLogger(AbstractController.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return null;
    }
    public SelectItem[] getItemsAvailableSelectMany() {
        return getSelectItems(ejbFacade.findAll(), false);
    }
```

```
    public SelectItem[] getItemsAvailableSelectOne() {
        return getSelectItems(ejbFacade.findAll(), true);
    }
    private SelectItem[] getSelectItems(List<T> entities, boolean selectOne) {
        int size = selectOne ? entities.size() + 1 : entities.size();
        SelectItem[] items = new SelectItem[size];
        int i = 0;
        if (selectOne) {
            items[0] = new SelectItem("", "---");
            i++;
        }
        for (Object x : entities) {
            items[i++] = new SelectItem(x, x.toString());
        }
        return items;
    }
}
```

- **Adminbean.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package beans;
import dao.AdminDao;
import java.util.ArrayList;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
/**
 *
 * @author Fuck
 */
@ManagedBean(name = "customer")
@SessionScoped
public class AdminBean {
private String user;
private String pass;
    public String getUser() {
        return user;
    }
    public void setUser(String user) {
        this.user = user;
    }
    public String getPass() {
        return pass;
```

```
    }

    public void setPass(String pass) {
        this.pass = pass;
    }
    /**
     * Creates a new instance of AdminBean
     */
    public ArrayList<AdminBean> getMessages() {
        return AdminDao.getCustomer();
    }
    }
```

- **Gejalacontroller.java**

```
package beans;
import model_crud_prime.Gejala;
import control.GejalaFacade;
import java.io.Serializable;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ViewScoped;
@ManagedBean(name = "gejalaController")
@ViewScoped
public class GejalaController extends AbstractController<Gejala> implements
Serializable {
    @EJB
    private GejalaFacade ejbFacade;
    public GejalaController() {
        super(Gejala.class);
    }
    @PostConstruct
    public void init() {
        super.setFacade(ejbFacade);
    }
}
```

- **Loginbean.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package beans;
import dao.UserDao;
import java.io.Serializable;
import javax.faces.application.FacesMessage;
```

```
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
import javax.faces.context.FacesContext;
import javax.servlet.http.HttpSession;
import model.Util;
/**
 *
 * @author Fuck
 */
@ManagedBean (name = "loginBean")
@SessionScoped
public class LoginBean implements Serializable {
    private static final long serialVersionUID = 1L;
    private String uname;
    private String password;
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String getUname() {
        return uname;
    }
    public void setUname(String uname) {
        this.uname = uname;
    }
    public String loginProject() {
        boolean result = UserDao.login(uname, password);
        if (result) {
          FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_INFO,"login
berhasil",""));
              return "menu_konsultasi";
        } else {
            FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_WARN, " Login
gagal! ,maaf anda belum terdaftar sebagai user",""));
            return "login";
        }
    }
    public String logout() {
```

```
        HttpSession session = Util.getSession();
        session.invalidate();
        return "index";
    }
}
```

- **Logincontroller.java**

```
package beans;
import model_crud_prime.Login;
import control.LoginFacade;
import java.io.Serializable;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ViewScoped;
@ManagedBean(name = "loginController")
@ViewScoped
public class LoginController extends AbstractController<Login> implements
Serializable {
    @EJB
    private LoginFacade ejbFacade;
    public LoginController() {
        super(Login.class);
    }
    @PostConstruct
    public void init() {
        super.setFacade(ejbFacade);
    }
}
```

- **Penyakitcontroller.java**

```
package beans;
import model_crud_prime.Penyakit;
import control.PenyakitFacade;
import java.io.Serializable;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ViewScoped;
@ManagedBean(name = "penyakitController")
@ViewScoped
public class PenyakitController extends AbstractController<Penyakit>
implements Serializable {
    @EJB
    private PenyakitFacade ejbFacade;
    public PenyakitController() {
```

```
        super(Penyakit.class);
    }
    @PostConstruct
    public void init() {
        super.setFacade(ejbFacade);
    }
}
```

- **Rulebasedcontroller.java**

```
package beans;
import model_crud_prime.RuleBased;
import control.RuleBase dFacade;
import java.io.Serializable;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ViewScoped;
@ManagedBean(name = "ruleBasedController")
@ViewScoped
public class RuleBasedController extends AbstractController<RuleBased>
implements Serializable {
    @EJB
    private RuleBasedFacade ejbFacade;
    public RuleBasedController() {
        super(RuleBased.class);
    }
    @PostConstruct
    public void init() {
        super.setFacade(ejbFacade);
    }
}
```

- **Gejala_bean.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package beans;
import dao.gejala_Dao;
import java.util.ArrayList;
import javax.faces.application.ConfigurableNavigationHandler;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
import javax.faces.context.FacesContext;
/**
 *
```

```
 * @author leomessi10
 */
@SessionScoped
@ManagedBean(name = "gejala")
public class gejala_bean {
  private String id,
nama,jenispemeriksaan,kd_penyakit,kd_gejala,HASIL_pk,trapi_penye,DES_pe
nye,Gej_penye,CF_penye,NM_penye;
  private Double cf_penyakit;
    public String getHASIL_pk() {
        return HASIL_pk;
    }
    public void setHASIL_pk(String HASIL_pk) {
        this.HASIL_pk = HASIL_pk;
    }
    public String getTrapi_penye() {
    return new gejala_Dao().Terapi_penyakit(selectedGejala2);
    }
    public void setTrapi_penye(String trapi_penye) {
        this.trapi_penye = trapi_penye;
    }
    public String getDES_penye() {
    return new gejala_Dao().Deskripsi_penyakit(selectedGejala2);
    }
    public void setDES_penye(String DES_penye) {
        this.DES_penye = DES_penye;
    }
    public String getGej_penye() {
     return new gejala_Dao().Gejala_penyakit(selectedGejala2);
    }
    public void setGej_penye(String Gej_penye) {
        this.Gej_penye = Gej_penye;
    }
    public String getCF_penye() {
       return new gejala_Dao().CF_penyakit(selectedGejala2);
    }
    public void setCF_penye(String CF_penye) {
        this.CF_penye = CF_penye;
    }
    public String getNM_penye() {
       return new gejala_Dao().nama_penyakit(selectedGejala2);
    }
    public void setNM_penye(String NM_penye) {
        this.NM_penye = NM_penye;
```

```
}
public String getKd_penyakit() {
    return kd_penyakit;
}
public void setKd_penyakit(String kd_penyakit) {
    this.kd_penyakit = kd_penyakit;
}
public String getKd_gejala() {
    return kd_gejala;
}
public void setKd_gejala(String kd_gejala) {
    this.kd_gejala = kd_gejala;
}
public Double getCf_penyakit() {
    return cf_penyakit;
}
public void setCf_penyakit(Double cf_penyakit) {
    this.cf_penyakit = cf_penyakit;
}
public String getId() {
    return id;
}
public void setId(String id) {
    this.id = id;
}
public String getNama() {
    return nama;
}
public void setNama(String nama) {
    this.nama = nama;
}
public String getJenispemeriksaan() {
    return jenispemeriksaan;
}
public void setJenispemeriksaan(String jenispemeriksaan) {
    this.jenispemeriksaan = jenispemeriksaan;
}
private gejala_bean selectedGejala;
private gejala_bean[] selectedGejala2;
    public ArrayList<gejala_bean> getMessages() {
    return gejala_Dao.getCustomer();
}
public gejala_bean getSelectedGejala() {
    return selectedGejala;
```

```
    }
    public void setSelectedGejala(gejala_bean selectedGejala) {
        this.selectedGejala = selectedGejala;
    }
    public gejala_bean[] getSelectedGejala2() {
        return selectedGejala2;
    }
    public void setSelectedGejala2(gejala_bean[] selectedGejala2) {
        this.selectedGejala2 = selectedGejala2;
    }
   public void onRowSelect() {
        ConfigurableNavigationHandler configurableNavigationHandler =
            (ConfigurableNavigationHandler)FacesContext.
              getCurrentInstance().getApplication().getNavigationHandler();
        configurableNavigationHandler.performNavigation("hasil
Diagnosa?faces-redirect=true");
      }
}
    /**
     * Creates a new instance of gejala_bean
     */
```

- **Perhitungan_cf.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package beans;
/**
 *
 * @author aris-sugi
 */
public class perhitungan_CF {
}
```

- **Abtractfacade.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package control;
import java.util.List;
import javax.persistence.EntityManager;
/**
 *
 * @author leomessi10
```

```
 */
public abstract class AbstractFacade<T> {
    private Class<T> entityClass;
    public AbstractFacade(Class<T> entityClass) {
        this.entityClass = entityClass;
    }
    protected abstract EntityManager getEntityManager();
    public void create(T entity) {
        getEntityManager().persist(entity);
    }
    public void edit(T entity) {
        getEntityManager().merge(entity);
    }
    public void remove(T entity) {
        getEntityManager().remove(getEntityManager().merge(entity));
    }
    public T find(Object id) {
        return getEntityManager().find(entityClass, id);
    }
    public List<T> findAll() {
        javax.persistence.criteria.CriteriaQuery cq =
getEntityManager().getCriteriaBuilder().createQuery();
        cq.select(cq.from(entityClass));
        return getEntityManager().createQuery(cq).getResultList();
    }
    public List<T> findRange(int[] range) {
        javax.persistence.criteria.CriteriaQuery cq =
getEntityManager().getCriteriaBuilder().createQuery();
        cq.select(cq.from(entityClass));
        javax.persistence.Query q = getEntityManager().createQuery(cq);
        q.setMaxResults(range[1] - range[0]);
        q.setFirstResult(range[0]);
        return q.getResultList();
    }
    public int count() {
        javax.persistence.criteria.CriteriaQuery cq =
getEntityManager().getCriteriaBuilder().createQuery();
        javax.persistence.criteria.Root<T> rt = cq.from(entityClass);
        cq.select(getEntityManager().getCriteriaBuilder().count(rt));
        javax.persistence.Query q = getEntityManager().createQuery(cq);
        return ((Long) q.getSingleResult()).intValue();
    }
}
```

- **Gejalafacade.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package control;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import model_crud_prime.Gejala;
/**
 *
 * @author leomessi10
 */
@Stateless
public class GejalaFacade extends AbstractFacade<Gejala> {
    @PersistenceContext(unitName = "sistem_pakar_hormonPU")
    private EntityManager em;
    @Override
    protected EntityManager getEntityManager() {
        return em;
    }
    public GejalaFacade() {
        super(Gejala.class);
    }
}
```

- **Loginfacade.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package control;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import model_crud_prime.Login;
/**
 *
 * @author leomessi10
 */
@Stateless
public class LoginFacade extends AbstractFacade<Login> {
    @PersistenceContext(unitName = "sistem_pakar_hormonPU")
    private EntityManager em;
    @Override
```

```
    protected EntityManager getEntityManager() {
        return em;
    }
    public LoginFacade() {
        super(Login.class);
    }
}
```

- **Penyakitfacade.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package control;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import model_crud_prime.Penyakit;
/**
 *
 * @author leomessi10
 */
@Stateless
public class PenyakitFacade extends AbstractFacade<Penyakit> {
    @PersistenceContext(unitName = "sistem_pakar_hormonPU")
    private EntityManager em;
    @Override
    protected EntityManager getEntityManager() {
        return em;
    }
    public PenyakitFacade() {
        super(Penyakit.class);
    }
}
```

- **Rulebasedfacade.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package control;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import model_crud_prime.RuleBased;
/**
```

```
 *
 * @author leomessi10
 */
@Stateless
public class RuleBasedFacade extends AbstractFacade<RuleBased> {
    @PersistenceContext(unitName = "sistem_pakar_hormonPU")
    private EntityManager em;
    @Override
    protected EntityManager getEntityManager() {
        return em;
    }
    public RuleBasedFacade() {
        super(RuleBased.class);
    }
}
```

- **Userfacade.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package control;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import model.User;
/**
 *
 * @author leomessi10
 */
@Stateless
public class UserFacade extends AbstractFacade<User> {
    @PersistenceContext(unitName = "sistem_pakar_hormonPU")
    private EntityManager em;
    @Override
    protected EntityManager getEntityManager() {
        return em;
    }
    public UserFacade() {
        super(User.class);
    }
}
```

- **Gej alaconverter.java**

```
package convert;
import model_crud_prime.Gejala;
```

```
import control.GejalaFacade;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.convert.FacesConverter;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.ServletContext;
@FacesConverter("gejalaConverter")
public class GejalaConverter implements Converter {
    @Override
    public Object getAsObject(FacesContext facesContext, UIComponent
component, String value) {
        if (value == null || value.length() == 0) {
            return null;
        }
        ServletContext servletContext = (ServletContext)
facesContext.getExternalContext().getContext();
        Context ctx = null;
        GejalaFacade facade = null;
        try {
            ctx = new InitialContext();
        } catch (NamingException ex) {
         Logger.getLogger(GejalaConverter.class.getName()).log(Level.SEVERE,
null, ex);
        }
        if (ctx != null) {
            try {
                String lookupString;
                if (servletContext != null) {
                    lookupString = "java:global" + servletContext.getContextPath() +
"/" + GejalaFacade.class.getSimpleName();
                } else {
                    lookupString = "java:global/" +
GejalaFacade.class.getSimpleName();
                }
                facade = (GejalaFacade) ctx.lookup(lookupString);
            } catch (NamingException ex) {
        Logger.getLogger(GejalaConverter.class.getName()).log(Level.SEVERE,
null, ex);
            }
```

```
        }
        if (facade != null) {
            return facade.find(getKey(value));
        }
        return null;
    }
    java.lang.String getKey(String value) {
        java.lang.String key;
        key = value;
        return key;
    }
    String getStringKey(java.lang.String value) {
        StringBuffer sb = new StringBuffer();
        sb.append(value);
        return sb.toString();
    }
    @Override
    public String getAsString(FacesContext facesContext, UIComponent
component, Object object) {
        if (object == null) {
            return null;
        }
        if (object instanceof Gejala) {
            Gejala o = (Gejala) object;
            return getStringKey(o.getIdGej());
        } else {
            Logger.getLogger(this.getClass().getName()).log(Level.SEVERE,
"object {0} is of type {1}; expected type: {2}", new Object[]{object,
object.getClass().getName(), Gejala.class.getName()});
            return null;
        }
    }
}
```

- **Loginconverter.java**

```
package convert;
import model_crud_prime.Login;
import control.LoginFacade;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.convert.FacesConverter;
import javax.naming.Context;
```

```
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.ServletContext;
@FacesConverter("loginConverter")
public class LoginConverter implements Converter {
    @Override
    public Object getAsObject(FacesContext facesContext, UIComponent
component, String value) {
        if (value == null || value.length() == 0) {
            return null;
        }
        ServletContext servletContext = (ServletContext)
facesContext.getExternalContext().getContext();
        Context ctx = null;
        LoginFacade facade = null;
        try {
            ctx = new InitialContext();
        } catch (NamingException ex) {
         Logger.getLogger(LoginConverter.class.getName()).log(Level.SEVERE,
null, ex);
        }
        if (ctx != null) {
            try {
                String lookupString;
                if (servletContext != null) {
                    lookupString = "java:global" + servletContext.getContextPath() +
"/" + LoginFacade.class.getSimpleName();
                } else {
                    lookupString = "java:global/" +
LoginFacade.class.getSimpleName();
                }
                facade = (LoginFacade) ctx.lookup(lookupString);
            } catch (NamingException ex) {
             Logger.getLogger(LoginConverter.class.getName()).log(Level.SEVERE,
null, ex);
            }
        }
        if (facade != null) {
            return facade.find(getKey(value));
        }
        return null;
    }
    java.lang.Integer getKey(String value) {
        java.lang.Integer key;
```

```
        key = Integer.valueOf(value);
        return key;
    }
    String getStringKey(java.lang.Integer value) {
        StringBuffer sb = new StringBuffer();
        sb.append(value);
        return sb.toString();
    }
    @Override
    public String getAsString(FacesContext facesContext, UIComponent
component, Object object) {
        if (object == null) {
            return null;
        }
        if (object instanceof Login) {
            Login o = (Login) object;
            return getStringKey(o.getId());
        } else {
            Logger.getLogger(this.getClass().getName()).log(Level.SEVERE,
"object {0} is of type {1}; expected type: {2}", new Object[]{object,
object.getClass().getName(), Login.class.getName()});
            return null;
        }
    }
}
```

- **Penyakitconverter.java**

```
package convert;
import model_crud_prime.Login;
import control.LoginFacade;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.convert.FacesConverter;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.ServletContext;
@FacesConverter("loginConverter")
public class LoginConverter implements Converter {
    @Override
    public Object getAsObject(FacesContext facesContext, UIComponent
component, String value) {
```

```
        if (value == null || value.length() == 0) {
            return null;
        }
        ServletContext servletContext = (ServletContext)
facesContext.getExternalContext().getContext();
        Context ctx = null;
        LoginFacade facade = null;
        try {
            ctx = new InitialContext();
        } catch (NamingException ex) {
          Logger.getLogger(LoginConverter.class.getName()).log(Level.SEVERE,
null, ex);
        }
        if (ctx != null) {
            try {
                String lookupString;
                if (servletContext != null) {
                    lookupString = "java:global" + servletContext.getContextPath() +
"/" + LoginFacade.class.getSimpleName();
                } else {
                    lookupString = "java:global/" +
LoginFacade.class.getSimpleName();
                }
                facade = (LoginFacade) ctx.lookup(lookupString);
            } catch (NamingException ex) {
         Logger.getLogger(LoginConverter.class.getName()).log(Level.SEVERE,
null, ex);
            }
        }
        if (facade != null) {
            return facade.find(getKey(value));
        }
        return null;
    }
    java.lang.Integer getKey(String value) {
        java.lang.Integer key;
        key = Integer.valueOf(value);
        return key;
    }
    String getStringKey(java.lang.Integer value) {
        StringBuffer sb = new StringBuffer();
        sb.append(value);
        return sb.toString();
    }
```

```
    @Override
    public String getAsString(FacesContext facesContext, UIComponent
component, Object object) {
        if (object == null) {
            return null;
        }
        if (object instanceof Login) {
            Login o = (Login) object;
            return getStringKey(o.getId());
        } else {
            Logger.getLogger(this.getClass().getName()).log(Level.SEVERE,
"object {0} is of type {1}; expected type: {2}", new Object[]{object,
object.getClass().getName(), Login.class.getName()});
            return null;
        }
    }
}
```

- **Rulebasedconverter.java**

```
package convert;
import model_crud_prime.RuleBased;
import control.RuleBasedFacade;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.convert.FacesConverter;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.ServletContext;
@FacesConverter("ruleBasedConverter")
public class RuleBasedConverter implements Converter {
    @Override
    public Object getAsObject(FacesContext facesContext, UIComponent
component, String value) {
        if (value == null || value.length() == 0) {
            return null;
        }
        ServletContext servletContext = (ServletContext)
facesContext.getExternalContext().getContext();
        Context ctx = null;
        RuleBasedFacade facade = null;
        try {
```

```
            ctx = new InitialContext();
        } catch (NamingException ex) {
  Logger.getLogger(RuleBasedConverter.class.getName()).log(Level.SEVERE,
null, ex);
        }
        if (ctx != null) {
            try {
                String lookupString;
                if (servletContext != null) {
                    lookupString = "java:global" + servletContext.getContextPath() +
"/" + RuleBasedFacade.class.getSimpleName();
                } else {
                    lookupString = "java:global/" +
RuleBasedFacade.class.getSimpleName();
                }
                facade = (RuleBasedFacade) ctx.lookup(lookupString);
            } catch (NamingException ex) {
Logger.getLogger(RuleBasedConverter.class.getName()).log(Level.SEVERE,
null, ex);
            }
        }
        if (facade != null) {
            return facade.find(getKey(value));
        }
        return null;
    }
    java.lang.String getKey(String value) {
        java.lang.String key;
        key = value;
        return key;
    }
    String getStringKey(java.lang.String value) {
        StringBuffer sb = new StringBuffer();
        sb.append(value);
        return sb.toString();
    }
    @Override
    public String getAsString(FacesContext facesContext, UIComponent
component, Object object) {
        if (object == null) {
            return null;
        }
        if (object instanceof RuleBased) {
            RuleBased o = (RuleBased) object;
```

```
            return getStringKey(o.getIdRule());
        } else {
            Logger.getLogger(this.getClass().getName()).log(Level.SEVERE,
"object {0} is of type {1}; expected type: {2}", new Object[]{object,
object.getClass().getName(), RuleBased.class.getName()});
            return null;
        }
    }
}
```

- **Admindao.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package dao;
import Database.Database;
import beans.AdminBean;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
/**
 *
 * @author Fuck
 */
public class AdminDao {
    public static ArrayList<AdminBean> getCustomer() {
        try {
            Connection con = Database.getConnection();
            PreparedStatement ps = con.prepareStatement("select * from user");
            ArrayList<AdminBean> al = new ArrayList<AdminBean>();
            ResultSet rs = ps.executeQuery();
            boolean found = false;
            while (rs.next()) {
                AdminBean e = new AdminBean();
                e.setUser(rs.getString("username"));
                e.setPass(rs.getString("password"));
                al.add(e);
                found = true;
            }
            rs.close();
            if (found) {
                return al;
            } else {
```

```
            return null; // no entires found
        }
    } catch (Exception e) {
        System.out.println("Error In getCustomer() -->" + e.getMessage());
        return (null);
    }
  }
}
```

- **Userdao.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package dao;
import Database.Database;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
/**
 *
 * @author Fuck
 */
public class UserDao {
    public static boolean login(String user, String password) {
        Connection con = null;
        PreparedStatement ps = null;
        try {
            con = Database.getConnection();
            ps = con.prepareStatement(
                    "select username, password from user where username= ? and
password= ? ");
            ps.setString(1, user);
            ps.setString(2, password);
            ResultSet rs = ps.executeQuery();
            if (rs.next()) // found
            {
                System.out.println(rs.getString("username"));
                return true;
            }
            else {
                return false;
            }
        } catch (Exception ex) {
            System.out.println("Error in login() -->" + ex.getMessage());
```

```
            return false;
        } finally {
            Database.close(con);
        }
    }
}
```

- **Gejala_dao.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package dao;
import Database.Database;
import beans.gejala_bean;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.DecimalFormat;
import java.util.ArrayList;
/**
 *
 * @author leomessi10
 */
public class gejala_Dao {
    public static ArrayList<gejala_bean> getCustomer() {
        try {
            Connection con = Database.getConnection();
            PreparedStatement ps = con.prepareStatement("SELECT
konsultasi.id_gej, gejala.nama, gejala.Jenis_pemeriksaan\n" +
"FROM konsultasi\n" +
"INNER JOIN gejala ON konsultasi.id_gej = gejala.id_gej");
            ArrayList<gejala_bean> al = new ArrayList<gejala_bean>();
            ResultSet rs = ps.executeQuery();
            boolean found = false;
            while (rs.next()) {
                gejala_bean e = new gejala_bean();
                e.setId(rs.getString("konsultasi.id_gej"));
                e.setNama(rs.getString("gejala.nama"));
                  e.setJenispemeriksaan(rs.getString("gejala.jenis_pemeriksaan"));
                al.add(e);
                found = true;
            }
```

```
            rs.close();
            if (found) {
                return al;
            } else {
                return null; // no entires found
            }
        } catch (Exception e) {
            System.out.println("Error In getCustomer() -->" + e.getMessage());
            return (null);
        }
    }
       private gejala_bean selectedGejala;
        private gejala_bean[] selectedGejala2;
        public ArrayList<gejala_bean> getMessages() {
        return gejala_Dao.getCustomer();
    }
    public gejala_bean getSelectedGejala() {
        return selectedGejala;
    }
    public void setSelectedGejala(gejala_bean selectedGejala) {
        this.selectedGejala = selectedGejala;
    }
    public gejala_bean[] getSelectedGejala2() {
        return selectedGejala2;
    }
    public void setSelectedGejala2(gejala_bean[] selectedGejala2) {
        this.selectedGejala2 = selectedGejala2;
    }
 // <editor-fold defaultstate="collapsed" desc="RUMUS CF Sistem Pakar">
    public   ArrayList<gejala_bean> Hasil_penyakit(gejala_bean []
selectedGejala2){
        int a =selectedGejala2.length;
        String [] gejala=new String[a];
        for (int i = 0; i < gejala.length; i++)
        gejala[i]=selectedGejala2[i].getId();
        ArrayList<gejala_bean> jn_penyakit=Jenis_penyakit(gejala);
        ArrayList<gejala_bean> CF_Penyakit =Jenis_penyakit(jn_penyakit);
            return CF_Penyakit;
    }
    public  String CF_penyakit(gejala_bean [] selectedGejala2){
 ArrayList<gejala_bean> CF_Penyakit=Hasil_penyakit(selectedGejala2);
 double [] CF=new double[CF_Penyakit.size()];
        for (int i = 0; i < CF.length; i++) {
            CF[i]=CF_Penyakit.get(i).getCf_penyakit();
```

```
        }
        String KD_py="";
CF=sort(CF);
        for (int i = 0; i < CF_Penyakit.size(); i++) {
           if(CF[0]==CF_Penyakit.get(i).getCf_penyakit()){
            KD_py=CF_Penyakit.get(i).getKd_penyakit();
            break;
           }
        }
      return ""+CF[0]+" / "+(int)(CF[0]*100)+"%";
    }
    public  String nama_penyakit(gejala_bean [] selectedGejala2){
 ArrayList<gejala_bean> CF_Penyakit=Hasil_penyakit(selectedGejala2);
 double [] CF=new double[CF_Penyakit.size()];
        for (int i = 0; i < CF.length; i++) {
            CF[i]=CF_Penyakit.get(i).getCf_penyakit();
        }
        String KD_py="";
CF=sort(CF);
        for (int i = 0; i < CF_Penyakit.size(); i++) {
           if(CF[0]==CF_Penyakit.get(i).getCf_penyakit()){
            KD_py=CF_Penyakit.get(i).getKd_penyakit();
            break;
           }
        }
        String NM_py="";
  Statement st = null;
                ResultSet rs = null;
                 Connection con=(Connection) new Database().getConnection();
        try {
            st =  (Statement) con.createStatement();
             rs = st.executeQuery("SELECT nama FROM penyakit WHERE
id_pen='"+ KD_py+"'");
                  while(rs.next()){
                 NM_py=rs.getString(1);
                  }
        } catch (SQLException ex) {
        }
      return NM_py;
    }
    public  String Terapi_penyakit(gejala_bean [] selectedGejala2){
 ArrayList<gejala_bean> CF_Penyakit=Hasil_penyakit(selectedGejala2);
 double [] CF=new double[CF_Penyakit.size()];
        for (int i = 0; i < CF.length; i++) {
```

```
            CF[i]=CF_Penyakit.get(i).getCf_penyakit();
        }
        String KD_py="";
CF=sort(CF);
        for (int i = 0; i < CF_Penyakit.size(); i++) {
           if(CF[0]==CF_Penyakit.get(i).getCf_penyakit()){
            KD_py=CF_Penyakit.get(i).getKd_penyakit();
            break;
           }
        }
        String NM_py="";
  Statement st = null;
                ResultSet rs = null;
                 Connection con=(Connection) new Database().getConnection();
        try {
            st =  (Statement) con.createStatement();
             rs = st.executeQuery("SELECT terapi FROM penyakit WHERE
id_pen='"+ KD_py+"'");
                  while(rs.next()){
                 NM_py=rs.getString(1);
                  }
        } catch (SQLException ex) {
        }
      return NM_py;
    }
    public  String Gejala_penyakit(gejala_bean [] selectedGejala2){
 ArrayList<gejala_bean> CF_Penyakit=Hasil_penyakit(selectedGejala2);
 double [] CF=new double[CF_Penyakit.size()];
        for (int i = 0; i < CF.length; i++) {
            CF[i]=CF_Penyakit.get(i).getCf_penyakit();
        }
        String KD_py="";
CF=sort(CF);
        for (int i = 0; i < CF_Penyakit.size(); i++) {
           if(CF[0]==CF_Penyakit.get(i).getCf_penyakit()){
            KD_py=CF_Penyakit.get(i).getKd_penyakit();
            break;
           }
        }
        String NM_py="";
  Statement st = null;
                ResultSet rs = null;
                 Connection con=(Connection) new Database().getConnection();
        try {
```

```
            st =  (Statement) con.createStatement();
             rs = st.executeQuery("SELECT gejala FROM penyakit WHERE
id_pen='"+ KD_py+"'");
                    while(rs.next()){
                   NM_py=rs.getString(1);
                    }
        } catch (SQLException ex) {
        }
      return NM_py;
    }
    public  String Deskripsi_penyakit(gejala_bean [] selectedGejala2){
 ArrayList<gejala_bean> CF_Penyakit=Hasil_penyakit(selectedGejala2);
 double [] CF=new double[CF_Penyakit.size()];
        for (int i = 0; i < CF.length; i++) {
            CF[i]=CF_Penyakit.get(i).getCf_penyakit();
        }
        String KD_py="";
CF=sort(CF);
        for (int i = 0; i < CF_Penyakit.size(); i++) {
           if(CF[0]==CF_Penyakit.get(i).getCf_penyakit()){
            KD_py=CF_Penyakit.get(i).getKd_penyakit();
            break;
           }
        }
        String NM_py="";
  Statement st = null;
                 ResultSet rs = null;
                  Connection con=(Connection) new Database().getConnection();
        try {
            st =  (Statement) con.createStatement();
             rs = st.executeQuery("SELECT deskripsi FROM penyakit WHERE
id_pen='"+ KD_py+"'");
                    while(rs.next()){
                   NM_py=rs.getString(1);
                    }
        } catch (SQLException ex) {
        }
      return NM_py;
    }
   public static double[] sort( double a[] ){
  int i, j;double t=0;
  for(i = 0; i < a.length; i++){
  for(j = 1; j < (a.length-i); j++){
  if(a[j-1] < a[j]){
```

```
  t = a[j-1];
  a[j-1]=a[j];
  a[j]=t;
  }
  }
  }
  return a;
  }
   private static ArrayList<gejala_bean>
Jenis_penyakit(ArrayList<gejala_bean> penyakit){
          Statement st = null;
                ResultSet rs = null;
         ArrayList<gejala_bean> penyakit_CF=new ArrayList<gejala_bean>();
              try {
                  Connection con=(Connection) new Database().getConnection();
                    st =  (Statement) con.createStatement();
        for (int i = 0; i <  penyakit.size(); i++) {
             String [] gejala= penyakit.get(i).getKd_gejala().split(",");
             if(gejala.length>1){
                    Double cf_penyakit = null;
                      rs = st.executeQuery("SELECT cf FROM penyakit WHERE
id_pen='"+ penyakit.get(i).getKd_penyakit()+"'");
                   while(rs.next())
                   cf_penyakit=Double.parseDouble(rs.getString(1));
                   Double []cf_gejala = new Double[gejala.length];
                      for (int j = 0; j < gejala.length; j++) {
                        rs = st.executeQuery("SELECT cf_gejala FROM konsultasi
where id_gej='"+gejala[j]+"'");
                        while(rs.next())
                         cf_gejala[j]=Double.parseDouble(rs.getString(1));
                      }
                  gejala_bean civ=new gejala_bean();
                  civ.setKd_penyakit(penyakit.get(i).getKd_penyakit());
                  civ.setCf_penyakit(CF_penyakit(cf_penyakit,cf_gejala));
                 penyakit_CF.add(civ);
             }else{
                 Double cf_penyakit = null;
                 Double cf_gejala = null;
             rs = st.executeQuery("SELECT cf FROM penyakit WHERE
id_pen='"+ penyakit.get(i).getKd_penyakit()+"'");
             while(rs.next())
             cf_penyakit=Double.parseDouble(rs.getString(1));
            rs = st.executeQuery("SELECT cf_gejala FROM konsultasi where
id_gej='"+ penyakit.get(i).getKd_gejala()+"'");
```

```
            while(rs.next())
             cf_gejala=Double.parseDouble(rs.getString(1));
            // gejala_bean civ=new
gejala_bean(penyakit.get(i).getKd_penyakit(),CF_penyakit(cf_penyakit,cf_geja
la));
             gejala_bean civ=new gejala_bean();
                  civ.setKd_penyakit(penyakit.get(i).getKd_penyakit());
                  civ.setCf_penyakit(CF_penyakit(cf_penyakit,cf_gejala));
             penyakit_CF.add(civ);
             }
              }
        } catch (SQLException ex) {
                 }
    return penyakit_CF;
    }
    private static ArrayList<gejala_bean> Jenis_penyakit(String [] gejala){
        ArrayList<gejala_bean> jenis_py=new ArrayList<gejala_bean>();
      try {
          Statement st = null;
                 ResultSet rs = null;
             Connection con=(Connection) new Database().getConnection();
                    st =  (Statement) con.createStatement();
                    for (int i = 0; i < gejala.length; i++) {
                 rs = st.executeQuery("SELECT * from rule_based where
id_gej='"+gejala[i]+"'");
                 while(rs.next()){
                 gejala_bean peny=new gejala_bean();
                 peny.setKd_penyakit(rs.getString(2));
                 peny.setKd_gejala(rs.getString(3));
                    if(jenis_py.size()>0){
                       for (int j = 0; j < jenis_py.size(); j++) {
if(jenis_py.get(j).getKd_penyakit().equalsIgnoreCase(peny.getKd_penyakit()))
{
                       String awal_gejala=jenis_py.get(j).getKd_gejala();
                       jenis_py.remove(j);
                    //  cf py_k=new
cf(peny.getKd_penyakit(),awal_gejala+","+peny.getKd_gejala());
                      gejala_bean py_k=new gejala_bean();
                 py_k.setKd_penyakit(peny.getKd_penyakit());
                 py_k.setKd_gejala(awal_gejala+","+peny.getKd_gejala());
                      jenis_py.add(py_k);
                      break;
                        }else{
                           jenis_py.add(peny);
```

```
                    break;
                    }
                }
            }else{
                jenis_py.add(peny);
            }
        }
    }
    } catch (SQLException ex) {
//          Logger.getLogger(gejala_bean.class.getName()).log(Level.SEVERE,
null, ex);
    }
 return jenis_py;
  }
  public static Double CF_penyakit(Double cf_penyakit,Double cf_gejala){
Double CF=(cf_gejala*cf_penyakit);
 DecimalFormat df = new DecimalFormat(".##");
return Double.valueOf(df.format(CF));
}
 public static Double CF_penyakit(Double cf_penyakit,Double [] cf_gejala){
DecimalFormat df = new DecimalFormat(".##");
Double [] CF_perpenyakit=new Double[cf_gejala.length];
    for (int i = 0; i < CF_perpenyakit.length; i++) {
     CF_perpenyakit[i]=cf_penyakit*cf_gejala[i];
    }
  if(CF_perpenyakit.length>2){
      Double
CF_sementara=Double.valueOf(df.format(CF_kombinasi(CF_perpenyakit[0],
CF_perpenyakit[1])));
      Double CF_hasil=null;
      for (int i = 2; i < CF_perpenyakit.length; i++) {
        CF_hasil=CF_kombinasi(CF_sementara, CF_perpenyakit[i]);
        CF_sementara=CF_hasil;
      }
  return Double.valueOf(df.format(CF_hasil));
  }else{
  return Double.valueOf(df.format(CF_kombinasi(CF_perpenyakit[0],
CF_perpenyakit[1])));
  }
}
 public static Double CF_kombinasi(Double CF1, Double CF2){
 Double CF_kombinasi=CF1+CF2*(1-CF1);
 return  CF_kombinasi;
 }
```

```
// </editor-fold>
}
```

- **Adminbean1.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package login_code;
import beans.AdminBean;
import dao.AdminDao;
import java.util.ArrayList;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
/**
 *
 * @author leomessi10
 */
@ManagedBean(name = "customer1")
@SessionScoped
public class AdminBean1 {
 private int id;
private String user;
private String pass;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getUser() {
        return user;
    }
    public void setUser(String user) {
        this.user = user;
    }
    public String getPass() {
        return pass;
    }
    public void setPass(String pass) {
        this.pass = pass;
    }
    /**
     * Creates a new instance of AdminBean
     */
```

```
    public ArrayList<AdminBean1> getMessages() {
        return AdminDao1.getCustomer();
    }
    }
    /**
     * Creates a new instance of AdminBean1
     */
```

- **Admindao1.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package login_code;
import Database.Database;
import beans.AdminBean;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
/**
 *
 * @author leomessi10
 */
public class AdminDao1 {
    public static ArrayList<AdminBean1> getCustomer() {
        try {
            Connection con = Database.getConnection();
            PreparedStatement ps = con.prepareStatement("select * from login");
            ArrayList<AdminBean1> al = new ArrayList<AdminBean1>();
            ResultSet rs = ps.executeQuery();
            boolean found = false;
            while (rs.next()) {
                AdminBean1 e = new AdminBean1();
                e.setUser(rs.getString("username"));
                e.setPass(rs.getString("password"));
                al.add(e);
                found = true;
            }
            rs.close();
            if (found) {
                return al;
            } else {
                return null; // no entires found
            }
```

```
        } catch (Exception e) {
            System.out.println("Error In getCustomer() -->" + e.getMessage());
            return (null);
        }
    }
}
```

- **Loginbean1.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package login_code;
import dao.UserDao;
import java.io.Serializable;
import javax.faces.application.FacesMessage;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
import javax.faces.context.FacesContext;
import javax.servlet.http.HttpSession;
import model.Util;
/**
 *
 * @author leomessi10
 */
@ManagedBean (name = "loginBean1")
@SessionScoped
public class LoginBean1 implements Serializable {
private static final long serialVersionUID = 1L;
    private String uname;
    private String password;
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String getUname() {
        return uname;
    }
    public void setUname(String uname) {
        this.uname = uname;
    }
    public String loginProject1() {
        boolean result = UserDao1.login(uname, password);
```

```
        if (result) {

          FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_INFO,"login
berhasil",""));
            return "index";
        } else {
           FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_WARN, " Login
gagal!",""));
           return "login_admin";
        }
    }
    public String logout() {
      HttpSession session = Util.getSession();
      session.invalidate();
      return "index";
   }
}
    /**
     * Creates a new instance of LoginBean1
     */
```

- **Userdao1.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package login_code;
import Database.Database;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
/**
 *
 * @author leomessi10
 */
public class UserDao1 {
    public static boolean login(String user, String password) {
        Connection con = null;
        PreparedStatement ps = null;
        try {
            con = Database.getConnection();
```

```
            ps = con.prepareStatement(
                    "select username, password from login where username= ? and
password= ? ");
            ps.setString(1, user);
            ps.setString(2, password);

            ResultSet rs = ps.executeQuery();
            if (rs.next()) // found
            {
                System.out.println(rs.getString("username"));
                return true;
            }
            else {
                return false;
            }
        } catch (Exception ex) {
            System.out.println("Error in login() -->" + ex.getMessage());
            return false;
        } finally {
            Database.close(con);
        }
    }
}
```

- **User.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package model;
import java.io.Serializable;
import javax.persistence.Basic;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Lob;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import javax.xml.bind.annotation.XmlRootElement;
/**
 *
 * @author leomessi10
```

```
 */
@Entity
@Table(name = "user")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "User.findAll", query = "SELECT u FROM User
u"),
    @NamedQuery(name = "User.findByUsername", query = "SELECT u
FROM User u WHERE u.username = :username"),
    @NamedQuery(name = "User.findByPassword", query = "SELECT u
FROM User u WHERE u.password = :password"),
    @NamedQuery(name = "User.findByNama", query = "SELECT u FROM
User u WHERE u.nama = :nama"),
    @NamedQuery(name = "User.findByUmur", query = "SELECT u FROM
User u WHERE u.umur = :umur"),
    @NamedQuery(name = "User.findByJenisKelamin", query = "SELECT u
FROM User u WHERE u.jenisKelamin = :jenisKelamin")})
public class User implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 25)
    @Column(name = "username")
    private String username;
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 25)
    @Column(name = "password")
    private String password;
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 50)
    @Column(name = "nama")
    private String nama;
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 10)
    @Column(name = "umur")
    private String umur;
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 25)
    @Column(name = "jenis_kelamin")
```

```
    private String jenisKelamin;
    @Basic(optional = false)
    @NotNull
    @Lob
    @Size(min = 1, max = 65535)
    @Column(name = "alamat")
    private String alamat;
    public User() {
    }
    public User(String username) {
        this.username = username;
    }
    public User(String username, String password, String nama, String umur,
String jenisKelamin, String alamat) {
        this.username = username;
        this.password = password;
        this.nama = nama;
        this.umur = umur;
        this.jenisKelamin = jenisKelamin;
        this.alamat = alamat;
    }
    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
        this.nama = nama;
    }
    public String getUmur() {
        return umur;
    }
    public void setUmur(String umur) {
        this.umur = umur;
```

```
    }
    public String getJenisKelamin() {
        return jenisKelamin;
    }
    public void setJenisKelamin(String jenisKelamin) {
        this.jenisKelamin = jenisKelamin;
    }
    public String getAlamat() {
        return alamat;
    }
    public void setAlamat(String alamat) {
        this.alamat = alamat;
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (username != null ? username.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are
not set
        if (!(object instanceof User)) {
            return false;
        }
        User other = (User) object;
        if ((this.username == null && other.username != null) || (this.username !=
null && !this.username.equals(other.username))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "model.User[ username=" + username + " ]";
    }
}
```

- **Util.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package model;
```

```
import javax.faces.context.FacesContext;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
/**
 *
 * @author Fuck
 */
public class Util {
  public static HttpSession getSession() {
       return (HttpSession)
         FacesContext.
         getCurrentInstance().
         getExternalContext().
         getSession(false);
     }
     public static HttpServletRequest getRequest() {
      return (HttpServletRequest) FacesContext.
         getCurrentInstance().
         getExternalContext().getRequest();
     }
     public static String getUserName()
     {
       HttpSession session = (HttpSession)
FacesContext.getCurrentInstance().getExternalContext().getSession(false);
       return  session.getAttribute("username").toString();
     }
     public static String getUserId()
     {
       HttpSession session = getSession();
       if ( session != null )
           return (String) session.getAttribute("userid");
       else
           return null;
     }
}
```

- **Gejala.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package model_crud_prime;
import java.io.Serializable;
import java.util.Collection;
import javax.persistence.Basic;
```

```
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Lob;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.OneToMany;
import javax.persistence.Table;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlTransient;
/**
 *
 * @author leomessi10
 */
@Entity
@Table(name = "gejala")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Gejala.findAll", query = "SELECT g FROM
Gejala g"),
    @NamedQuery(name = "Gejala.findByIdGej", query = "SELECT g FROM
Gejala g WHERE g.idGej = :idGej"),
    @NamedQuery(name = "Gejala.findByNama", query = "SELECT g FROM
Gejala g WHERE g.nama = :nama")})
public class Gejala implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 10)
    @Column(name = "id_gej")
    private String idGej;
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 100)
    @Column(name = "nama")
    private String nama;
    @Basic(optional = false)
    @NotNull
    @Lob
    @Size(min = 1, max = 65535)
```

```
    @Column(name = "Jenis_pemeriksaan")
    private String jenispemeriksaan;
    @OneToMany(cascade = CascadeType.ALL, mappedBy = "idGej")
    private Collection<RuleBased> ruleBasedCollection;
    public Gejala() {
    }
    public Gejala(String idGej) {
        this.idGej = idGej;
    }
    public Gejala(String idGej, String nama, String jenispemeriksaan) {
        this.idGej = idGej;
        this.nama = nama;
        this.jenispemeriksaan = jenispemeriksaan;
    }
    public String getIdGej() {
        return idGej;
    }
    public void setIdGej(String idGej) {
        this.idGej = idGej;
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
        this.nama = nama;
    }
    public String getJenispemeriksaan() {
        return jenispemeriksaan;
    }
    public void setJenispemeriksaan(String jenispemeriksaan) {
        this.jenispemeriksaan = jenispemeriksaan;
    }
    @XmlTransient
    public Collection<RuleBased> getRuleBasedCollection() {
        return ruleBasedCollection;
    }
    public void setRuleBasedCollection(Collection<RuleBased>
ruleBasedCollection) {
        this.ruleBasedCollection = ruleBasedCollection;
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (idGej != null ? idGej.hashCode() : 0);
```

```
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are
not set
        if (!(object instanceof Gejala)) {
            return false;
        }
        Gejala other = (Gejala) object;
        if ((this.idGej == null && other.idGej != null) || (this.idGej != null &&
!this.idGej.equals(other.idGej))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "model_crud_prime.Gejala[ idGej=" + idGej + " ]";
    }
}
```

- **Login.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package model_crud_prime;
import java.io.Serializable;
import javax.persistence.Basic;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import javax.xml.bind.annotation.XmlRootElement;
/**
 *
 * @author leomessi10
 */
@Entity
@Table(name = "login")
```

```
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Login.findAll", query = "SELECT l FROM Login
l"),
    @NamedQuery(name = "Login.findById", query = "SELECT l FROM
Login l WHERE l.id = :id"),
    @NamedQuery(name = "Login.findByUsername", query = "SELECT l
FROM Login l WHERE l.username = :username"),
    @NamedQuery(name = "Login.findByPassword", query = "SELECT l
FROM Login l WHERE l.password = :password")})
public class Login implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @NotNull
    @Column(name = "id")
    private Integer id;
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 25)
    @Column(name = "username")
    private String username;
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 25)
    @Column(name = "password")
    private String password;
    public Login() {
    }
    public Login(Integer id) {
        this.id = id;
    }
    public Login(Integer id, String username, String password) {
        this.id = id;
        this.username = username;
        this.password = password;
    }
    public Integer getId() {
        return id;
    }
    public void setId(Integer id) {
        this.id = id;
    }
    public String getUsername() {
```

```
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (id != null ? id.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are
not set
        if (!(object instanceof Login)) {
            return false;
        }
        Login other = (Login) object;
        if ((this.id == null && other.id != null) || (this.id != null &&
!this.id.equals(other.id))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "model_crud_prime.Login[ id=" + id + " ]";
    }
}
```

- **Penyakit.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package model_crud_prime;
import java.io.Serializable;
import java.math.BigDecimal;
```

```
import java.util.Collection;
import javax.persistence.Basic;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Lob;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.OneToMany;
import javax.persistence.Table;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlTransient;
/**
 *
 * @author leomessi10
 */
@Entity
@Table(name = "penyakit")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Penyakit.findAll", query = "SELECT p FROM
Penyakit p"),
    @NamedQuery(name = "Penyakit.findByIdPen", query = "SELECT p
FROM Penyakit p WHERE p.idPen = :idPen"),
    @NamedQuery(name = "Penyakit.findByNama", query = "SELECT p
FROM Penyakit p WHERE p.nama = :nama"),
    @NamedQuery(name = "Penyakit.findByCf", query = "SELECT p FROM
Penyakit p WHERE p.cf = :cf")})
public class Penyakit implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 10)
    @Column(name = "id_pen")
    private String idPen;
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 100)
    @Column(name = "nama")
    private String nama;
```

```
    @Basic(optional = false)
    @NotNull
    @Lob
    @Size(min = 1, max = 65535)
    @Column(name = "deskripsi")
    private String deskripsi;
    @Basic(optional = false)
    @NotNull
    @Lob
    @Size(min = 1, max = 65535)
    @Column(name = "gejala")
    private String gejala;
    @Basic(optional = false)
    @NotNull
    @Lob
    @Size(min = 1, max = 65535)
    @Column(name = "terapi")
    private String terapi;
    // @Max(value=?)  @Min(value=?)//if you know range of your decimal
fields consider using these annotations to enforce field validation
    @Basic(optional = false)
    @NotNull
    @Column(name = "cf")
    private BigDecimal cf;
    @OneToMany(cascade = CascadeType.ALL, mappedBy = "idPen")
    private Collection<RuleBased> ruleBasedCollection;
    public Penyakit() {
    }
    public Penyakit(String idPen) {
        this.idPen = idPen;
    }
    public Penyakit(String idPen, String nama, String deskripsi, String gejala,
String terapi, BigDecimal cf) {
        this.idPen = idPen;
        this.nama = nama;
        this.deskripsi = deskripsi;
        this.gejala = gejala;
        this.terapi = terapi;
        this.cf = cf;
    }
    public String getIdPen() {
        return idPen;
    }
    public void setIdPen(String idPen) {
```

```
        this.idPen = idPen;
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
        this.nama = nama;
    }
    public String getDeskripsi() {
        return deskripsi;
    }
    public void setDeskripsi(String deskripsi) {
        this.deskripsi = deskripsi;
    }
    public String getGejala() {
        return gejala;
    }
    public void setGejala(String gejala) {
        this.gejala = gejala;
    }
    public String getTerapi() {
        return terapi;
    }
    public void setTerapi(String terapi) {
        this.terapi = terapi;
    }
    public BigDecimal getCf() {
        return cf;
    }
    public void setCf(BigDecimal cf) {
        this.cf = cf;
    }
    @XmlTransient
    public Collection<RuleBased> getRuleBasedCollection() {
        return ruleBasedCollection;
    }
    public void setRuleBasedCollection(Collection<RuleBased>
ruleBasedCollection) {
        this.ruleBasedCollection = ruleBasedCollection;
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (idPen != null ? idPen.hashCode() : 0);
```

```
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are
not set
        if (!(object instanceof Penyakit)) {
            return false;
        }
        Penyakit other = (Penyakit) object;
        if ((this.idPen == null && other.idPen != null) || (this.idPen != null &&
!this.idPen.equals(other.idPen))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "model_crud_prime.Penyakit[ idPen=" + idPen + " ]";
    }
}
```

- **Rulebased.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package model_crud_prime;
import java.io.Serializable;
import javax.persistence.Basic;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import javax.xml.bind.annotation.XmlRootElement;
/**
 *
 * @author leomessi10
 */
```

```
@Entity
@Table(name = "rule_based")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "RuleBased.findAll", query = "SELECT r FROM
RuleBased r"),
    @NamedQuery(name = "RuleBased.findByIdRule", query = "SELECT r
FROM RuleBased r WHERE r.idRule = :idRule")})
public class RuleBased implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 10)
    @Column(name = "id_rule")
    private String idRule;
    @JoinColumn(name = "id_gej", referencedColumnName = "id_gej")
    @ManyToOne(optional = false)
    private Gejala idGej;
    @JoinColumn(name = "id_pen", referencedColumnName = "id_pen")
    @ManyToOne(optional = false)
    private Penyakit idPen;
    public RuleBased() {
    }
    public RuleBased(String idRule) {
        this.idRule = idRule;
    }
    public String getIdRule() {
        return idRule;
    }
    public void setIdRule(String idRule) {
        this.idRule = idRule;
    }
    public Gejala getIdGej() {
        return idGej;
    }
    public void setIdGej(Gejala idGej) {
        this.idGej = idGej;
    }
    public Penyakit getIdPen() {
        return idPen;
    }
    public void setIdPen(Penyakit idPen) {
        this.idPen = idPen;
```

```
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (idRule != null ? idRule.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are
not set
        if (!(object instanceof RuleBased)) {
            return false;
        }
        RuleBased other = (RuleBased) object;
        if ((this.idRule == null && other.idRule != null) || (this.idRule != null &&
!this.idRule.equals(other.idRule))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "model_crud_prime.RuleBased[ idRule=" + idRule + " ]";
    }
}
```

- **Usermanagedbean.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package session;
import java.util.List;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
import model.User;
/**
 *
 * @author leomessi10
 */
@ManagedBean (name ="UserManagedBean")
@SessionScoped
public class UserManagedBean {
```

```
    @EJB
    private UserSessionBean userSB;
    private String nama;
    private List<User> userData = null;
    /**
     * Creates a new instance of UserManagedBean
     */
    public UserManagedBean() {
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
        this.nama = nama;
    }
    public List<User> getUserData() {
        userData = userSB.cariUser(nama);
        return userData;
}
}
```

- **Usersessionbean.java**

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package session;
import java.io.Serializable;
import java.util.List;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.persistence.Query;
import model.User;
/**
 *
 * @author leomessi10
 */
@Stateless
public class UserSessionBean implements Serializable {
@PersistenceContext(unitName = "sistem_pakar_hormonPU")
    private EntityManager em;
    // Add business logic below. (Right-click in editor and choose
    // "Insert Code > Add Business Method")
    public List<User> cariUser(String firstNameInput) {
```

```
        Query query = em.createNamedQuery("User.findByNama");
        query.setParameter("nama",firstNameInput);
        List<User> daftarUser = query.getResultList();
        return daftarUser;
    }
}
```

- **Usercontroller.java**

```
package user;
import model.User;
import user.util.JsfUtil;
import user.util.PaginationHelper;
import control.UserFacade;
import java.io.Serializable;
import java.util.ResourceBundle;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.convert.FacesConverter;
import javax.faces.model.DataModel;
import javax.faces.model.ListDataModel;
import javax.faces.model.SelectItem;
@ManagedBean(name = "userController")
@SessionScoped
public class UserController implements Serializable {
    private User current;
    private DataModel items = null;
    @EJB
    private control.UserFacade ejbFacade;
    private PaginationHelper pagination;
    private int selectedItemIndex;
    public UserController() {
    }
    public User getSelected() {
        if (current == null) {
            current = new User();
            selectedItemIndex = -1;
        }
        return current;
    }
    private UserFacade getFacade() {
        return ejbFacade;
```

```
    }
    public PaginationHelper getPagination() {
        if (pagination == null) {
            pagination = new PaginationHelper(10) {
                @Override
                public int getItemsCount() {
                    return getFacade().count();
                }
                @Override
                public DataModel createPageDataModel() {
                    return new ListDataModel(getFacade().findRange(new
int[]{getPageFirstItem(), getPageFirstItem() + getPageSize()}));
                }
            };
        }
        return pagination;
    }
    public String prepareList() {
        recreateModel();
        return "List";
    }
    public String prepareView() {
        current = (User) getItems().getRowData();
        selectedItemIndex = pagination.getPageFirstItem() +
getItems().getRowIndex();
        return "View";
    }
    public String prepareCreate() {
        current = new User();
        selectedItemIndex = -1;
        return "Create";
    }
    public String create() {
        try {
            getFacade().create(current);
JsfUtil.addSuccessMessage(ResourceBundle.getBundle("/Bundle").getString("
UserCreated"));
            return prepareCreate();
        } catch (Exception e) {
            JsfUtil.addErrorMessage(e,
ResourceBundle.getBundle("/Bundle").getString("PersistenceErrorOccured"));
            return null;
        }
    }
```

```
    public String prepareEdit() {
        current = (User) getItems().getRowData();
        selectedItemIndex = pagination.getPageFirstItem() +
getItems().getRowIndex();
        return "Edit";
    }
    public String update() {
        try {
            getFacade().edit(current);
JsfUtil.addSuccessMessage(ResourceBundle.getBundle("/Bundle").getString("
UserUpdated"));
            return "View";
        } catch (Exception e) {
            JsfUtil.addErrorMessage(e,
ResourceBundle.getBundle("/Bundle").getString("PersistenceErrorOccured"));
            return null;
        }
    }
    public String destroy() {
        current = (User) getItems().getRowData();
        selectedItemIndex = pagination.getPageFirstItem() +
getItems().getRowIndex();
        performDestroy();
        recreatePagination();
        recreateModel();
        return "List";
    }
    public String destroyAndView() {
        performDestroy();
        recreateModel();
        updateCurrentItem();
        if (selectedItemIndex >= 0) {
            return "View";
        } else {
            // all items were removed - go back to list
            recreateModel();
            return "List";
        }
    }
    private void performDestroy() {
        try {
            getFacade().remove(current);
JsfUtil.addSuccessMessage(ResourceBundle.getBundle("/Bundle").getString("
UserDeleted"));
```

```
        } catch (Exception e) {
            JsfUtil.addErrorMessage(e,
ResourceBundle.getBundle("/Bundle").getString("PersistenceErrorOccured"));
        }
    }
    private void updateCurrentItem() {
        int count = getFacade().count();
        if (selectedItemIndex >= count) {
            // selected index cannot be bigger than number of items:
            selectedItemIndex = count - 1;
            // go to previous page if last page disappeared:
            if (pagination.getPageFirstItem() >= count) {
                pagination.previousPage();
            }
        }
        if (selectedItemIndex >= 0) {
            current = getFacade().findRange(new int[]{selectedItemIndex,
selectedItemIndex + 1}).get(0);
        }
    }
    public DataModel getItems() {
        if (items == null) {
            items = getPagination().createPageDataModel();
        }
        return items;
    }
    private void recreateModel() {
        items = null;
    }
    private void recreatePagination() {
        pagination = null;
    }
    public String next() {
        getPagination().nextPage();
        recreateModel();
        return "List";
    }
    public String previous() {
        getPagination().previousPage();
        recreateModel();
        return "List";
    }
    public SelectItem[] getItemsAvailableSelectMany() {
        return JsfUtil.getSelectItems(ejbFacade.findAll(), false);
```

```
    }

    public SelectItem[] getItemsAvailableSelectOne() {
        return JsfUtil.getSelectItems(ejbFacade.findAll(), true);
    }
    @FacesConverter(forClass = User.class)
    public static class UserControllerConverter implements Converter {
        public Object getAsObject(FacesContext facesContext, UIComponent
component, String value) {
            if (value == null || value.length() == 0) {
                return null;
            }
            UserController controller = (UserController)
facesContext.getApplication().getELResolver().
                    getValue(facesContext.getELContext(), null, "userController");
            return controller.ejbFacade.find(getKey(value));
        }
        java.lang.String getKey(String value) {
            java.lang.String key;
            key = value;
            return key;
        }
        String getStringKey(java.lang.String value) {
            StringBuffer sb = new StringBuffer();
            sb.append(value);
            return sb.toString();
        }
        public String getAsString(FacesContext facesContext, UIComponent
component, Object object) {
            if (object == null) {
                return null;
            }
            if (object instanceof User) {
                User o = (User) object;
                return getStringKey(o.getUsername());
            } else {
                throw new IllegalArgumentException("object " + object + " is of type
" + object.getClass().getName() + "; expected type: " + User.class.getName());
            }
        }
    }
}
```

- **Jfsutil.java**

```
package user.util;
```

```
import java.util.List;
import javax.faces.application.FacesMessage;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.model.SelectItem;
public class JsfUtil {
    public static SelectItem[] getSelectItems(List<?> entities, boolean selectOne)
{
        int size = selectOne ? entities.size() + 1 : entities.size();
        SelectItem[] items = new SelectItem[size];
        int i = 0;
        if (selectOne) {
            items[0] = new SelectItem("", "---");
            i++;
        }
        for (Object x : entities) {
            items[i++] = new SelectItem(x, x.toString());
        }
        return items;
    }
    public static void addErrorMessage(Exception ex, String defaultMsg) {
        String msg = ex.getLocalizedMessage();
        if (msg != null && msg.length() > 0) {
            addErrorMessage(msg);
        } else {
            addErrorMessage(defaultMsg);
        }
    }
    public static void addErrorMessages(List<String> messages) {
        for (String message : messages) {
            addErrorMessage(message);
        }
    }
    public static void addErrorMessage(String msg) {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext.getCurrentInstance().addMessage(null, facesMsg);
    }
    public static void addSuccessMessage(String msg) {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext.getCurrentInstance().addMessage("successInfo", facesMsg);
    }
```

```
    public static String getRequestParameter(String key) {
        return
FacesContext.getCurrentInstance().getExternalContext().getRequestParameter
Map().get(key);
    }
    public static Object getObjectFromRequestParameter(String
requestParameterName, Converter converter, UIComponent component) {
        String theId = JsfUtil.getRequestParameter(requestParameterName);
        return converter.getAsObject(FacesContext.getCurrentInstance(),
component, theId);
    }
}
```

- **Paginationhelper.java**

```
package user.util;
import javax.faces.model.DataModel;
public abstract class PaginationHelper {
    private int pageSize;
    private int page;
    public PaginationHelper(int pageSize) {
        this.pageSize = pageSize;
    }
    public abstract int getItemsCount();
    public abstract DataModel createPageDataModel();
    public int getPageFirstItem() {
        return page * pageSize;
    }
    public int getPageLastItem() {
        int i = getPageFirstItem() + pageSize - 1;
        int count = getItemsCount() - 1;
        if (i > count) {
            i = count;
        }
        if (i < 0) {
            i = 0;
        }
        return i;
    }
    public boolean isHasNextPage() {
        return (page + 1) * pageSize + 1 <= getItemsCount();
    }
    public void nextPage() {
        if (isHasNextPage()) {
            page++;
        }
```

```
    }
    public boolean isHasPreviousPage() {
        return page > 0;
    }
    public void previousPage() {
        if (isHasPreviousPage()) {
            page--;
        }
    }
    public int getPageSize() {
        return pageSize;
    }
}
```