```
/*
 * KerberosAuthService.java
 *
 * Created on August 29, 2006, 3:12 PM
 * Copyright (c) Massachusetts Institute of Technology
 * 77 Massachusetts Avenue, Cambridge, MA 02139-4307
 * All rights reserved.
 */

package edu.mit.coeus.user.auth;

import edu.mit.coeus.bean.CoeusMessageResourcesBean;
import edu.mit.coeus.bean.LoginBean;
import edu.mit.coeus.bean.ValidateUserTxnBean;
import edu.mit.coeus.brokers.RequesterBean;
import edu.mit.coeus.exception.CoeusException;
import edu.mit.coeus.utils.UtilFactory;
import edu.mit.coeus.utils.dbengine.DBException;
import edu.mit.coeuslite.irb.form.MyLogonForm;
import java.io.IOException;
import java.util.Hashtable;
import java.util.Properties;
import javax.naming.AuthenticationException;
import javax.naming.Context;
import javax.naming.NamingEnumeration;
import javax.naming.NamingException;
import javax.naming.directory.Attributes;
import javax.naming.directory.DirContext;
import javax.naming.directory.InitialDirContext;
import javax.naming.directory.SearchControls;
import javax.naming.directory.SearchResult;
//DARTMOUTH
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import edu.mit.coeus.utils.dbengine.DBEngineImpl;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
//DARTMOUTH

/**
 *
 * @author  Geo Thomas
 */
public class LDAPAuthService extends AuthServiceServerBase{
    private Hashtable data;
```

```
    private Properties props;
    private static final String LDAP_CONTEXT_FACTORY =
"com.sun.jndi.ldap.LdapCtxFactory";
    private static final String LDAP_PROVIDER_URL = "ldap://loclhost:123/";
    private static final String LDAP_SECURITY_AUTH_MODE = "simple";
    private static final String LDAP_SECURITY_PROTOCOL = "ssl";
    private static final String LDAP_FILTER_UID_NAME = "samAccountName";
    private static final String LDAP_DN_KEY = "distinguishedName";
    private static DirContext ctx;

    /** Creates a new instance of KerberosAuthService */
    public LDAPAuthService() {
    }
    private void createInitialDirContext() throws CoeusException{
//      if(ctx!=null) return;
        try{
            UtilFactory.log("Entering ldap auth");
            // Set up the environment for creating the initial context
            Hashtable env = new Hashtable();
            env.put(Context.INITIAL_CONTEXT_FACTORY,
props.getProperty("LDAP_CONTEXT_FACTORY",LDAP_CONTEXT_FACTORY));
            env.put(Context.PROVIDER_URL,
props.getProperty("LDAP_PROVIDER_URL",LDAP_PROVIDER_URL));
//
            env.put(Context.SECURITY_AUTHENTICATION,
props.getProperty("LDAP_SECURITY_AUTH_MODE",LDAP_SECURITY_AUTH_M
ODE));
            env.put(Context.SECURITY_PROTOCOL,
props.getProperty("LDAP_SECURITY_PROTOCOL",LDAP_SECURITY_PROTOCO
L));

//            System.setProperty("javax.net.debug", "all");
//            System.setProperty("javax.net.ssl.trustStore",
CoeusProperties.getProperty("TRUSTSTORE_FILE",TRUSTSTORE_FILE));
            //DARTMOUTH
            //System.setProperty("javax.net.ssl.trustStore",
CoeusProperties.getProperty("TRUSTSTORE_FILE",TRUSTSTORE_FILE));
            System.setProperty("javax.net.ssl.trustStore", "/misc_apps/coeus/truststore");
            //System.setProperty("javax.net.ssl.trustStorePassword",
CoeusProperties.getProperty("TRUSTSTORE_PASSWORD",TRUSTSTORE_PASSW
ORD));
            System.setProperty("javax.net.ssl.trustStorePassword", "password");
            UtilFactory.log("LDAP Authentication successful -- truststore");
            //DARTMOUTH
          if(props.getProperty("LDAP_DEBUG")!=null)
              System.setProperty("javax.net.debug", props.getProperty("LDAP_DEBUG"));
```

```
            if(props.getProperty("COEUS_LDAP_ADMIN_DN")!=null)
                env.put(Context.SECURITY_PRINCIPAL,
props.getProperty("COEUS_LDAP_ADMIN_DN"));
            if(props.getProperty("COEUS_LDAP_ADMIN_CREDENTIALS")!=null)
                env.put(Context.SECURITY_CREDENTIALS,
props.getProperty("COEUS_LDAP_ADMIN_CREDENTIALS"));

            UtilFactory.log("going to create admin ldap context");
            // Create the initial context
            ctx = new InitialDirContext(env);
            UtilFactory.log("created admin ldap context");
        }catch(NamingException nEx){
            nEx.printStackTrace();
            UtilFactory.log(nEx.getMessage(),nEx,"PurdueLDAPAuthService",
"createInitialDirContext");
            throw new CoeusException("exceptionCode.100003");
        }
    }
    public boolean authenticate() throws CoeusException {

        createInitialDirContext();
        RequesterBean requester =
(RequesterBean)data.get(RequesterBean.class.getName());
        String userId;
        //DARTMOUTH
        String userName;
        //DARTMOUTH
        String password;
        if(requester==null){
            MyLogonForm logonForm =
(MyLogonForm)data.get(MyLogonForm.class.getName());
            if(logonForm==null) throw new CoeusException("Not able to receive login form
details");
            userId = logonForm.getUsername();
            password = logonForm.getPassword();
            //DARTMOUTH
            userName = userId;
            userId = toCoeusUserid(userId);
            //DARTMOUTH
        }else{
            LoginBean loginBean = (LoginBean)requester.getDataObject();
            userId = loginBean.getUserId();
            password = loginBean.getPassword();
            //DARTMOUTH
            userName = userId;
            userId = toCoeusUserid(userId);
```

```
            //DARTMOUTH

        }
        if(userId==null || userId.trim().equals("")) throw new CoeusException("User id is
empty");
        if(password==null || password.trim().equals("")) throw new
CoeusException("Password is empty");
        CoeusMessageResourcesBean cmrb = new CoeusMessageResourcesBean();
        String errMsg = "";

        try{
            //String filter =
"(&("+props.getProperty("LDAP_FILTER_UID_NAME",LDAP_FILTER_UID_NAME
)+"="+userId+"))";
            // DARTMOUTH
             String filter =
"(&("+props.getProperty("LDAP_FILTER_UID_NAME",LDAP_FILTER_UID_NAME
)+"="+userName+"))";
            // DARTMOUTH
            SearchControls ctls = new SearchControls();
            ctls.setSearchScope(SearchControls.SUBTREE_SCOPE);
            NamingEnumeration answer =
ctx.search(props.getProperty("LDAP_DOMAIN_BASE"), filter,ctls);

            String userDN = null;
            Attributes attrs = null;
            while (answer.hasMore()) {
                SearchResult sr = (SearchResult)answer.next();
                if(sr!=null)
                    attrs = sr.getAttributes();
                if(attrs!=null){
//                    userDN =
(String)attrs.get(props.getProperty("LDAP_DN_KEY",LDAP_DN_KEY)).get();
                    userDN = sr.getName() + ","
+props.getProperty("LDAP_DOMAIN_BASE");
                    break;
                }
            }

            if(userDN!=null){

                Hashtable envUser = new Hashtable();
                envUser.put(Context.INITIAL_CONTEXT_FACTORY,
props.getProperty("LDAP_CONTEXT_FACTORY"));
                if(props.getProperty("LDAP_PROVIDER_URL")!=null)
```

```
                envUser.put(Context.PROVIDER_URL,
props.getProperty("LDAP_PROVIDER_URL"));
            if(props.getProperty("LDAP_SECURITY_AUTH_MODE")!=null)
                envUser.put(Context.SECURITY_AUTHENTICATION,
props.getProperty("LDAP_SECURITY_AUTH_MODE"));
            envUser.put(Context.SECURITY_PRINCIPAL, userDN);
            envUser.put(Context.SECURITY_CREDENTIALS, password);
            try{
                DirContext userCtx = new InitialDirContext(envUser);
            }catch(AuthenticationException authEx){

UtilFactory.log(authEx.getMessage(),authEx,"LDAPAuthService","authenticate");
                String msg = authEx.getExplanation();
                if(msg.indexOf(" 52e")!=-1){
                    errMsg = cmrb.parseMessageKey("exceptionCode.100002");
                    throw new CoeusException(errMsg);
                }else{
                    throw new CoeusException(authEx.getMessage());
                }
            }
            UtilFactory.log("User "+userId+"logged in");
        }else{
            errMsg = cmrb.parseMessageKey("exceptionCode.100000");
            throw new CoeusException(errMsg);
        }
    }catch(NamingException nEx){

        UtilFactory.log(nEx.getMessage(),nEx,"LDAPAuthService","authenticate");
        if(nEx.getMessage().indexOf(" 525")!=-1){
            errMsg = cmrb.parseMessageKey("exceptionCode.100000");
            throw new CoeusException(errMsg);
        }else{
            errMsg = cmrb.parseMessageKey("exceptionCode.100002");
            throw new CoeusException(errMsg);
        }
    }
    if(requester==null)
        setResponseForWeb(userId);//for web
    else
        setResponse(userId);//for swing

    UtilFactory.log("LDAP Authentication successful");
    return true;
  }
  public void init(java.util.Properties props) {
    this.props = props;
```

```
    }

    public void setParams(java.util.Hashtable data) {
       this.data = data;
    }

   //DARTMOUTH
   private static String toCoeusUserid(String name) {
     if (name == null)
       return null;
     String uid = name;
     int i = uid.indexOf('@');
     if (i > 0)
       uid = uid.substring(i+1);
     Connection conn = null;
     DBEngineImpl dbEngine = null;
     try {
       dbEngine = new DBEngineImpl();
       conn = dbEngine.beginTxn();
       PreparedStatement ps = conn.prepareStatement("select user_id from osp$user where
user_name = ?");
       ps.setString(1, name);
       ResultSet rs = ps.executeQuery();
       if (rs.next()) {
         String s = rs.getString(1);
         if (s != null && s.length() != 0)
           uid = s;
       }
     } catch (Exception ex) {
       System.out.println(ex.getMessage());
     } finally {
       try {
         dbEngine.endTxn(conn);
       } catch (Exception ex) {
       }
     }
     return uid;
   }
  //DARTMOUTH

}
```