# The purpose of this exercise is to add EJB, JPA and H Exerci se: Working with ernateJPA and Hibibernate support to the existing lab exercise.

# Part I: Add JDBC support and data source to Wildfly

First we need to add MySql JDBC driver and connector to Wildfly

## Add Mysql JDBC driver

* Go to wildfly-10.1.0.Final\modules\system\layers\base\com
* Create folder like “sql\mysql\main\” and go to sql\mysql\main\
* Place mysql connector jar to main folder
* Place module.xml file to main folder

|  |
| --- |
| **module**.xml |
| <?xml version=**"1.0"** encoding=**"UTF-8"**?>  <module xmlns=**"urn:jboss:module:1.1"** name=**"com.sql.mysql"**>  <resources>  <resource-root path=**"mysql-connector-java-5.0.4-bin.jar"**/>  </resources>  <dependencies>  <module name=**"javax.api"**/>  </dependencies>  </module> |

## Add datasource to wildfly configuration file

* Go to wildfly-10.1.0.Final\standalone\configuration
* Open standalone.xml file (we assume you are using standalone.xml as server configuration file)
* Add following code to datasources section

|  |
| --- |
| standalone.xml |
| <datasources>  <datasource jndi-name=**"java:jboss/datasources/ExampleDS"** pool-name=**"ExampleDS"** enabled=**"true"** use-java-context=**"true"**>  <connection-url>**jdbc:h2:mem:test;DB\_CLOSE\_DELAY=-1;DB\_CLOSE\_ON\_EXIT=FALSE**</connection-url>  <driver>**h2**</driver>  <security>  <user-name>**sa**</user-name>  <password>**sa**</password>  </security>  </datasource>  <datasource jta=**"true"** jndi-name=**"java:/SoftUniDS"** pool-name=**"SoftUniDS"** enabled=**"true"** use-java-context=**"true"**>  <connection-url>**jdbc:mysql://localhost:3306/blog?characterEncoding=utf8**</connection-url>  <driver>**mysql**</driver>  <pool>  <min-pool-size>**5**</min-pool-size>  <max-pool-size>3**0**</max-pool-size>  <prefill>**true**</prefill>  </pool>  <security>  <user-name>**root**</user-name>  </security>  </datasource>  <drivers>  <driver name=**"h2"** module=**"com.h2database.h2"**>  <xa-datasource-class>**org.h2.jdbcx.JdbcDataSource**</xa-datasource-class>  </driver>  <driver name=**"mysql"** module=**"com.mysql.driver"**>  <driver-class>**com.mysql.jdbc.Driver**</driver-class>  </driver>  </drivers>  </datasources> |

# Part II: Register the datasource in the application

## Create persistence.xml, located in /SoftUniJEEEJB/ejbModule/META-INF

|  |
| --- |
| persistence.xml |
| <?xml version=**"1.0"** encoding=**"UTF-8"**?>  <persistence xmlns=**"http://java.sun.com/xml/ns/persistence"**  xmlns:xsi=**"http://www.w3.org/2001/XMLSchema-instance"**  xsi:schemaLocation=**"http://java.sun.com/xml/ns/persistence**  **http://java.sun.com/xml/ns/persistence/persistence\_1\_0.xsd"**  version=**"1.0"**>  <persistence-unit name=**"SoftUnitJEE"**>  <jta-data-source>**java:/SoftUniDS**</jta-data-source>  <properties>  <property name=**"jboss.entity.manager.factory.jndi.name"** value=**"java:/Manager1Factory"** />  <property name=**"hibernate.dialect"** value=**"org.hibernate.dialect.MySQL5InnoDBDialect"** />  <!-- <property name="hibernate.show\_sql" value="true" /> -->  <!-- comment the below in case of production deploy -->  <!-- <property name="hibernate.hbm2ddl.auto" value="update" /> -->    </properties>  </persistence-unit>    </persistence> |

# Part III: Create Entity beans

## Add email column to USERS DB table and split fullname column to two columns named first\_name and last\_name

## Create BaseDomainObject class in EJB project

|  |
| --- |
| BaseDomainObject.java |
| package bg**.**softuni**.**entity**.**base**;**  **import** java**.**io**.**Serializable**;**  **import** javax**.**persistence**.**GeneratedValue**;**  **import** javax**.**persistence**.**GenerationType**;**  **import** javax**.**persistence**.**Id**;**  **import** javax**.**persistence**.**MappedSuperclass**;**  /\*\*  \* Base model class, all entities should extend this one.  \*/  @MappedSuperclass  public class BaseDomainObject **implements** Serializable **{**  private static final long serialVersionUID **=** 1L**;**  protected Long id**;**  @Id  @GeneratedValue**(**strategy **=** GenerationType**.**IDENTITY**)**  public Long getId**()** **{**  **return** id**;**  **}**  public void setId**(**Long id**)** **{**  **this.**id **=** id**;**  **}**  @Override  public int hashCode**()** **{**  final int prime **=** 31**;**  int result **=** 1**;**  result **=** prime **\*** result **+** **((**id **==** **null)** **?** 0 **:** id**.**hashCode**());**  **return** result**;**  **}**  @Override  public boolean equals**(**Object obj**)** **{**  **if** **(this** **==** obj**)** **{**  **return** **true;**  **}**  **if** **(**obj **==** **null)** **{**  **return** **false;**  **}**  **if** **(**getClass**()** **!=** obj**.**getClass**())** **{**  **return** **false;**  **}**  BaseDomainObject other **=** **(**BaseDomainObject**)** obj**;**  **if** **(**id **==** **null)** **{**  **if** **(**other**.**id **!=** **null)** **{**  **return** **false;**  **}**  **}** **else** **if** **(!**id**.**equals**(**other**.**id**))** **{**  **return** **false;**  **}**  **return** **true;**  **}**  **}** |

## Create Entity Beans for User and Post objects

|  |
| --- |
| UserModel.java |
| package bg**.**softuni**.**entity**;**  **import** java**.**io**.**Serializable**;**  **import** java**.**util**.**List**;**  **import** javax**.**persistence**.**CascadeType**;**  **import** javax**.**persistence**.**Column**;**  **import** javax**.**persistence**.**Entity**;**  **import** javax**.**persistence**.**FetchType**;**  **import** javax**.**persistence**.**OneToMany**;**  **import** javax**.**persistence**.**Table**;**  **import** bg**.**softuni**.**entity**.**base**.**BaseDomainObject**;**  @Entity  @Table**(**name **=** "USERS"**)**  public class UserModel **extends** BaseDomainObject **implements** Serializable **{**  private static final long serialVersionUID **=** 1L**;**  private String username**;**  private String password**;**  private String firstName**;**  private String lastName**;**  private String email**;**  private List**<**PostModel**>** posts**;**  public UserModel**()** **{**  **}**  public UserModel**(**Long id**,** String username**,** String password**,** String first**,** String last**,** String email**)** **{**  **this.**id **=** id**;**  **this.**username **=** username**;**  **this.**password **=** password**;**  **this.**firstName **=** first**;**  **this.**lastName **=** last**;**  **this.**email **=** email**;**  **}**  @Column**(**name **=** "username"**,** length **=** 45**,** nullable **=** **false)**  public String getUsername**()** **{**  **return** username**;**  **}**  public void setUsername**(**String username**)** **{**  **this.**username **=** username**;**  **}**  @Column**(**name **=** "password"**,** length **=** 200**,** nullable **=** **false)**  public String getPassword**()** **{**  **return** password**;**  **}**  public void setPassword**(**String password**)** **{**  **this.**password **=** password**;**  **}**  @Column**(**name **=** "first\_name"**,** length **=** 45**,** nullable **=** **true)**  public String getFirstName**()** **{**  **return** firstName**;**  **}**  public void setFirstName**(**String firstName**)** **{**  **this.**firstName **=** firstName**;**  **}**  @Column**(**name **=** "last\_name"**,** length **=** 45**,** nullable **=** **true)**  public String getLastName**()** **{**  **return** lastName**;**  **}**  public void setLastName**(**String lastName**)** **{**  **this.**lastName **=** lastName**;**  **}**  @Column**(**name **=** "email"**,** length **=** 45**,** nullable **=** **true)**  public String getEmail**()** **{**  **return** email**;**  **}**  public void setEmail**(**String email**)** **{**  **this.**email **=** email**;**  **}**    @OneToMany**(**cascade **=** CascadeType**.**ALL**,** mappedBy **=** "author"**,** fetch **=** FetchType**.**LAZY**)**  public List**<**PostModel**>** getPosts**()** **{**  **return** posts**;**  **}**  public void setPosts**(**List**<**PostModel**>** posts**)** **{**  **this.**posts **=** posts**;**  **}**  @Override  public int hashCode**()** **{**  int hash **=** 0**;**  hash **+=** **(**id **!=** **null** **?** id**.**hashCode**()** **:** 0**);**  **return** hash**;**  **}**  @Override  public boolean equals**(**Object object**)** **{**  **if** **(!(**object **instanceof** UserModel**))** **{**  **return** **false;**  **}**  UserModel other **=** **(**UserModel**)** 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** "Users[ id=" **+** id **+** " ]"**;**  **}**  **}** |

|  |
| --- |
| PostModel.java |
| package bg**.**softuni**.**entity**;**  **import** java**.**io**.**Serializable**;**  **import** java**.**util**.**Date**;**  **import** javax**.**persistence**.**Column**;**  **import** javax**.**persistence**.**Entity**;**  **import** javax**.**persistence**.**JoinColumn**;**  **import** javax**.**persistence**.**ManyToOne**;**  **import** javax**.**persistence**.**Table**;**  **import** javax**.**persistence**.**Temporal**;**  **import** javax**.**persistence**.**TemporalType**;**  **import** bg**.**softuni**.**entity**.**base**.**BaseDomainObject**;**  @Entity  @Table**(**name **=** "POSTS"**)**  public class PostModel **extends** BaseDomainObject **implements** Serializable **{**  private static final long serialVersionUID **=** 1L**;**  private String title**;**  private String content**;**  private Date date**;**  private UserModel author**;**  @Column**(**name **=** "title"**,** length **=** 100**,** nullable **=** **false)**  public String getTitle**()** **{**  **return** title**;**  **}**  public void setTitle**(**String title**)** **{**  **this.**title **=** title**;**  **}**  @Column**(**name **=** "content"**,** length **=** 4000**,** nullable **=** **false)**  public String getContent**()** **{**  **return** content**;**  **}**  public void setContent**(**String content**)** **{**  **this.**content **=** content**;**  **}**  @Column**(**name **=** "date"**)**  @Temporal**(**TemporalType**.**DATE**)**  public Date getDate**()** **{**  **return** date**;**  **}**  public void setDate**(**Date date**)** **{**  **this.**date **=** date**;**  **}**  @JoinColumn**(**name **=** "author\_id"**,** referencedColumnName **=** "id"**)**  @ManyToOne  public UserModel getAuthor**()** **{**  **return** author**;**  **}**  public void setAuthor**(**UserModel author**)** **{**  **this.**author **=** author**;**  **}**  **}** |

# Part IV: Create EJBs for manipulating the data

## Create UserService EJB

|  |
| --- |
| UserService.java |
| package bg**.**softuni**.**service**;**  **import** java**.**util**.**List**;**  **import** javax**.**ejb**.**Local**;**  **import** bg**.**softuni**.**entity**.**UserModel**;**  @Local  public interface UserService **{**  List**<**UserModel**>** findAllUsers**();**  UserModel save**(**UserModel entity**);**  UserModel update**(**UserModel entity**);**  void delete**(**UserModel entity**);**  UserModel findById**(**Long id**);**  UserModel loginUser**(**String aUsername**,** String aPassword**);**  UserModel checkUserExists**(**String username**,** Long id**);**  **}** |

|  |
| --- |
| UserServiceImpl.java |
| package bg**.**softuni**.**service**;**  **import** java**.**util**.**List**;**  **import** javax**.**ejb**.**Stateless**;**  **import** javax**.**persistence**.**EntityManager**;**  **import** javax**.**persistence**.**NoResultException**;**  **import** javax**.**persistence**.**PersistenceContext**;**  **import** javax**.**persistence**.**Query**;**  **import** bg**.**softuni**.**entity**.**UserModel**;**  /\*\*  \* Session Bean implementation class UserServiceImpl  \*/  @Stateless  public class UserServiceImpl **implements** UserService **{**  @PersistenceContext  protected EntityManager entityManager**;**  public UserServiceImpl**()** **{**  **super();**  **}**  @SuppressWarnings**(**"unchecked"**)**  @Override  public List**<**UserModel**>** findAllUsers**()** **{**  String query **=** "select model from UserModel model where model.isActive = true order by upper(model.username) asc"**;**  Query q **=** entityManager**.**createQuery**(**query**);**  **return** q**.**getResultList**();**  **}**  @Override  public UserModel save**(**UserModel entity**)** **{**  entityManager**.**persist**(**entity**);**  **return** entity**;**  **}**  @Override  public UserModel update**(**UserModel entity**)** **{**  entityManager**.**merge**(**entity**);**  entityManager**.**flush**();**  **return** entity**;**  **}**  @Override  public void delete**(**UserModel entity**)** **{**  entityManager**.**remove**(**entity**);**  **}**  @Override  public UserModel findById**(**Long id**)** **{**  **try** **{**  UserModel instance **=** entityManager**.**find**(**UserModel**.**class**,** id**);**  **return** instance**;**  **}** **catch** **(**RuntimeException re**)** **{**  **throw** re**;**  **}**  **}**  @Override  public UserModel loginUser**(**String aUsername**,** String aPassword**)** **{**  StringBuffer query **=** **new** StringBuffer**(**  "select model from UserModel model where model.username = :em and model.password = :p"**);**  Query q **=** entityManager**.**createQuery**(**query**.**toString**());**  q**.**setParameter**(**"em"**,** aUsername**);**  q**.**setParameter**(**"p"**,** aPassword**);**  **try** **{**  **return** **(**UserModel**)** q**.**getSingleResult**();**  **}** **catch** **(**NoResultException nre**)** **{**  // the user doesn't exist  **return** **null;**  **}**  **}**  @Override  public UserModel checkUserExists**(**String username**,** Long id**)** **{**  StringBuffer query **=** **new** StringBuffer**(**  "select model from UserModel model where upper(model.username) = upper(:em)"**);**  **if** **(**id **!=** **null)** **{**  query**.**append**(**" and model.id <> "**).**append**(**id**);**  **}**  Query q **=** entityManager**.**createQuery**(**query**.**toString**());**  q**.**setParameter**(**"em"**,** username**);**  **try** **{**  **return** **(**UserModel**)** q**.**getSingleResult**();**  **}** **catch** **(**NoResultException nre**)** **{**  // the user doesnt exist  **return** **null;**  **}**  **}**  **}** |

# Part V: Add Password Encryption

## Create GeneralUtils class in Web project

|  |
| --- |
| GeneralUtils.java |
| package bg**.**softuni**.**web**.**utils**;**  **import** java**.**io**.**UnsupportedEncodingException**;**  **import** java**.**math**.**BigInteger**;**  **import** java**.**security**.**MessageDigest**;**  **import** java**.**security**.**NoSuchAlgorithmException**;**  **import** java**.**text**.**DateFormat**;**  **import** java**.**text**.**SimpleDateFormat**;**  **import** java**.**util**.**Date**;**  **import** javax**.**servlet**.**http**.**HttpServletRequest**;**  **import** org**.**bouncycastle**.**util**.**encoders**.**Hex**;**  **import** bg**.**softuni**.**entity**.**UserModel**;**  /\*\*  \* Utility class for helping various operations  \*  \*/  public class GeneralUtils **{**    public static UserModel getLoggedUser**(**Object request**)** **{**  **if** **(**request **instanceof** HttpServletRequest**)** **{**  HttpServletRequest req **=** **(**HttpServletRequest**)** request**;**  UserModel loggedUser **=** **(**UserModel**)** req**.**getSession**().**getAttribute**(**"LOGGED\_USER"**);**  **return** loggedUser**;**  **}**  **return** **null;**  **}**    public static String encodeSha256Password**(**String aPlainTextPassword**)** **{**  **try** **{**  MessageDigest digest **=** MessageDigest**.**getInstance**(**"SHA-256"**);**  byte**[]** hash **=** digest**.**digest**(**aPlainTextPassword**.**getBytes**(**"UTF-8"**));**  **return** **new** String**(**Hex**.**encode**(**hash**));**  **}** **catch** **(**NoSuchAlgorithmException e**)** **{**  **throw** **new** RuntimeException**(**"No encoding algorythm found"**,** e**);**  **}** **catch** **(**UnsupportedEncodingException e**)** **{**  **throw** **new** RuntimeException**(**"No encoding support"**,** e**);**  **}**  **}**    public static String encodeMd5**(**String aPlainText**)** **{**  **try** **{**  MessageDigest digest **=** MessageDigest**.**getInstance**(**"MD5"**);**  digest**.**update**(**aPlainText**.**getBytes**(**"UTF-8"**),** 0**,** aPlainText**.**length**());**  **return** **new** BigInteger**(**1**,** digest**.**digest**()).**toString**(**16**);**  **}** **catch** **(**NoSuchAlgorithmException e**)** **{**  **throw** **new** RuntimeException**(**"No encoding algorythm found"**,** e**);**  **}** **catch** **(**UnsupportedEncodingException e**)** **{**  **throw** **new** RuntimeException**(**"No encoding support"**,** e**);**  **}**  **}**        public static String covertDateToString**(**Date aDate**,** String pattern**)** **{**  DateFormat format **=** **new** SimpleDateFormat**(**pattern**);**  **return** format**.**format**(**aDate**);**  **}**      **}** |

## Add bcprov-jdk14-138.jar to Web project class path

# Part VI: Add EJBs to the Web project

## Inject UserService in LoginBean and remove simulated storage usage

|  |
| --- |
| LoginBean.java |
| package bg**.**softuni**.**web**;**  **import** java**.**io**.**Serializable**;**  **import** javax**.**annotation**.**PostConstruct**;**  **import** javax**.**ejb**.**EJB**;**  **import** javax**.**faces**.**application**.**FacesMessage**;**  **import** javax**.**faces**.**bean**.**ManagedBean**;**  **import** javax**.**faces**.**bean**.**ManagedProperty**;**  **import** javax**.**faces**.**bean**.**RequestScoped**;**  **import** javax**.**faces**.**context**.**FacesContext**;**  **import** javax**.**inject**.**Inject**;**  **import** javax**.**servlet**.**http**.**HttpServletRequest**;**  **import** org**.**apache**.**commons**.**lang3**.**StringUtils**;**  **import** bg**.**softuni**.**dto**.**UserDto**;**  **import** bg**.**softuni**.**entity**.**UserModel**;**  **import** bg**.**softuni**.**service**.**UserService**;**  **import** bg**.**softuni**.**web**.**beans**.**UsersBean**;**  **import** bg**.**softuni**.**web**.**utils**.**MessageUtils**;**  /\*\*  \* The purpose of this class is to provide backend functionality of the login  \*/  @ManagedBean**(**name **=** "loginBean"**)**  @RequestScoped  public class LoginBean **implements** Serializable **{**  private static final long serialVersionUID **=** 1L**;**  // beans.xml need to be added in WEB-INF in order CDI to work  @Inject  private HttpServletRequest request**;**  private String username**;**  private String password**;**  //removed  //@ManagedProperty("#{usersBean}")  //private UsersBean usersBean;    //added  @EJB  UserService userService**;**  private static final String SUCCESS\_LOGIN\_REDIRECT **=** "/page/listUsers?faces-redirect=true"**;**  private static final String LOGIN\_PAGE\_REDIRECT **=** "/page/login?faces-redirect=true"**;**  @PostConstruct  public void init**()** **{**  // TODO  **}**  /\*\*  \* Implement application login logic  \*  \* **@return**  \*/  public String login**()** **{**  String cryptedPassword = GeneralUtils.*encodeSha256Password*(password);    //added  UserModel userModel **=** userService**.**loginUser**(**username**,** cryptedPassword**);**    //removed  //UserDto user = usersBean.validateUser(username, password);  **if** **(null** **==** userModel**)** **{**  MessageUtils**.**addErrorMessage**(**"login.error.invalid.credentials"**);**  **return** ""**;**  **}** **else** **{**  request**.**getSession**().**setAttribute**(**"LOGGED\_USER"**,** userModel**);**  **return** SUCCESS\_LOGIN\_REDIRECT**;**  **}**  **}**  /\*\*  \* Implement application logout logic  \*  \* **@return**  \*/  public String logout**()** **{**  request**.**getSession**().**invalidate**();**  **return** LOGIN\_PAGE\_REDIRECT**;**  **}**  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 UsersBean getUsersBean**()** **{**  **return** usersBean**;**  **}**  public void setUsersBean**(**UsersBean usersBean**)** **{**  **this.**usersBean **=** usersBean**;**  **}**  **}** |

## Update the users in the database

Set the following password to all of them: a665a45920422f9d417e4867efdc4fb8a04a1f3fff1fa07e998e86f7f7a27ae3

This is SHA-256 representation of the default password : 123

## Modify ListUsersBean.java and listUsers.xhtml to use the UsersService (replace the users collection and start passing id instead of name when editing)

|  |
| --- |
| ListUsersBean.java |
| package bg**.**softuni**.**web**.**beans**;**  **import** java**.**util**.**List**;**  **import** javax**.**annotation**.**PostConstruct**;**  **import** javax**.**ejb**.**EJB**;**  **import** javax**.**faces**.**bean**.**ManagedBean**;**  **import** javax**.**faces**.**bean**.**ManagedProperty**;**  **import** javax**.**faces**.**bean**.**ViewScoped**;**  **import** bg**.**softuni**.**entity**.**UserModel**;**  **import** bg**.**softuni**.**service**.**UserService**;**  @ManagedBean**(**name **=** "listUsersBean"**)**  @ViewScoped  public class ListUsersBean **{**  //removed  // @ManagedProperty("#{usersBean}")  // private UsersBean usersBean;  // added  @EJB  UserService userService**;**  @PostConstruct  public void init**()** **{**  **}**  public String editAction**()** **{**  **return** "/page/editUser"**;**  **}**  public String createAction**()** **{**  **return** "/page/createUser"**;**  **}**  //removed  // public UsersBean getUsersBean() {  // return usersBean;  // }  //  // public void setUsersBean(UsersBean usersBean) {  // this.usersBean = usersBean;  // }  public List**<**UserModel**>** getAllUsers**()** **{**  **return** userService**.**findAllUsers**();**  **}**  **}** |

## Modify EditUserBean.java

|  |
| --- |
| EditUserBean.java |
| package bg**.**softuni**.**web**.**beans**;**  **import** java**.**util**.**Iterator**;**  **import** java**.**util**.**regex**.**Pattern**;**  **import** javax**.**annotation**.**PostConstruct**;**  **import** javax**.**ejb**.**EJB**;**  **import** javax**.**faces**.**application**.**FacesMessage**;**  **import** javax**.**faces**.**bean**.**ManagedBean**;**  **import** javax**.**faces**.**bean**.**ManagedProperty**;**  **import** javax**.**faces**.**bean**.**ViewScoped**;**  **import** javax**.**faces**.**context**.**FacesContext**;**  **import** javax**.**inject**.**Inject**;**  **import** javax**.**servlet**.**http**.**HttpServletRequest**;**  **import** org**.**apache**.**commons**.**lang3**.**StringUtils**;**  **import** bg**.**softuni**.**dto**.**UserDto**;**  **import** bg**.**softuni**.**entity**.**UserModel**;**  **import** bg**.**softuni**.**service**.**UserService**;**  **import** bg**.**softuni**.**web**.**utils**.**MessageUtils**;**  @ManagedBean**(**name **=** "editUserBean"**)**  @ViewScoped  public class EditUserBean **{**  @Inject  HttpServletRequest request**;**  // removed  // @ManagedProperty("#{usersBean}")  // private UsersBean usersBean;  // added  @EJB  UserService userService**;**    // removed  // private UserDto user;  private UserModel user**;**  private static final String EMAIL\_PATTERN **=** "^[\_A-Za-z0-9-]+(\\."  **+** "[\_A-Za-z0-9-]+)\*@[A-Za-z0-9]+(\\.[A-Za-z0-9]+)\*(\\.[A-Za-z]{2,})$"**;**  private Pattern pattern**;**  @PostConstruct  public void init**()** **{**  // removed  /\*  \* String username = request.getParameter("username");  \*  \* for (UserDto currUser : usersBean.getUsers()) { if  \* (currUser.getUsername().equals(username)) { user = currUser; break; }  \* }  \*/  String id **=** request**.**getParameter**(**"id"**);**  **if** **(**StringUtils**.**isNotBlank**(**id**)** **||** StringUtils**.**isNumeric**(**id**))** **{**  user **=** userService**.**findById**(**Long**.**valueOf**(**id**));**  **}**  **}**  public String updateAction**()** **{**  **if** **(!**validate**())** **{**  **return** **null;**  **}**    String cryptedPassword = GeneralUtils.*encodeSha256Password*(user.getPassword());  user.setPassword(cryptedPassword);  userService**.**update**(**user**);**  **return** "/page/listUsers?faces-redirect=true"**;**  **}**  // added  public UserModel getUser**()** **{**  **return** user**;**  **}**  public void setUser**(**UserModel user**)** **{**  **this.**user **=** user**;**  **}**  // removed  /\*  \* public UsersBean getUsersBean() { return usersBean; }  \*  \* public void setUsersBean(UsersBean usersBean) { this.usersBean =  \* usersBean; }  \*  \* public UserDto getUser() { return user; }  \*  \* public void setUser(UserDto user) { this.user = user; }  \*/  protected boolean validate**()** **{**  boolean hasErrors **=** **false;**  **if** **(**StringUtils**.**isBlank**(**user**.**getUsername**()))** **{**  MessageUtils**.**addErrorMessage**(**"username"**,** "error.required.username"**);**  hasErrors **=** **true;**  **}**  **if** **(**StringUtils**.**isBlank**(**user**.**getPassword**()))** **{**  MessageUtils**.**addErrorMessage**(**"error.required.password"**);**  hasErrors **=** **true;**  **}**  **if** **(**StringUtils**.**isBlank**(**user**.**getFirstName**()))** **{**  MessageUtils**.**addErrorMessage**(**"error.required.firstname"**);**  hasErrors **=** **true;**  **}**  **if** **(**StringUtils**.**isBlank**(**user**.**getLastName**()))** **{**  MessageUtils**.**addErrorMessage**(**"error.required.lastname"**);**  hasErrors **=** **true;**  **}**  **if** **(**StringUtils**.**isBlank**(**user**.**getEmail**()))** **{**  MessageUtils**.**addErrorMessage**(**"error.required.email"**);**  hasErrors **=** **true;**  **}**  pattern **=** Pattern**.**compile**(**EMAIL\_PATTERN**);**  **if** **(!**pattern**.**matcher**(**user**.**getEmail**()).**matches**())** **{**  MessageUtils**.**addErrorMessage**(**"error.invalid.email"**);**  hasErrors **=** **true;**  **}**  **if** **(**hasErrors**)** **{**  **return** **false;**  **}**  **return** **true;**  **}**  /\*\*  \* Verifies if a error messages are present in the context  \*/  public boolean hasErrors**()** **{**  Iterator**<**FacesMessage**>** messages **=** FacesContext**.**getCurrentInstance**().**getMessages**();**  **for** **(;** messages**.**hasNext**();)** **{**  FacesMessage message **=** messages**.**next**();**  **if** **(**message**.**getSeverity**().**compareTo**(**FacesMessage**.**SEVERITY\_ERROR**)** **==** 0**)** **{**  **return** **true;**  **}**  **}**  **return** **false;**  **}**  **}** |

## Modify CeateUserBean.java

|  |
| --- |
| CreateUserBean.java |
| package bg**.**softuni**.**web**.**beans**;**  **import** java**.**util**.**Iterator**;**  **import** java**.**util**.**regex**.**Matcher**;**  **import** java**.**util**.**regex**.**Pattern**;**  **import** javax**.**annotation**.**PostConstruct**;**  **import** javax**.**ejb**.**EJB**;**  **import** javax**.**faces**.**application**.**FacesMessage**;**  **import** javax**.**faces**.**bean**.**ManagedBean**;**  **import** javax**.**faces**.**bean**.**ManagedProperty**;**  **import** javax**.**faces**.**bean**.**ViewScoped**;**  **import** javax**.**faces**.**context**.**FacesContext**;**  **import** javax**.**inject**.**Inject**;**  **import** javax**.**servlet**.**http**.**HttpServletRequest**;**  **import** org**.**apache**.**commons**.**lang3**.**StringUtils**;**  **import** bg**.**softuni**.**dto**.**UserDto**;**  **import** bg**.**softuni**.**entity**.**UserModel**;**  **import** bg**.**softuni**.**service**.**UserService**;**  **import** bg**.**softuni**.**web**.**utils**.**GeneralUtils**;**  **import** bg**.**softuni**.**web**.**utils**.**MessageUtils**;**  @ManagedBean**(**name **=** "createUserBean"**)**  @ViewScoped  public class CreateUserBean **{**  @Inject  HttpServletRequest request**;**  // added  @EJB  UserService userService**;**  // @ManagedProperty("#{usersBean}")  // private UsersBean usersBean;  // private UserDto user;  // added  private UserModel user**;**  private static final String EMAIL\_PATTERN **=** "^[\_A-Za-z0-9-]+(\\."  **+** "[\_A-Za-z0-9-]+)\*@[A-Za-z0-9]+(\\.[A-Za-z0-9]+)\*" **+** "(\\.[A-Za-z]{2,})$"**;**  private Pattern pattern**;**  @PostConstruct  public void init**()** **{**  user **=** **new** UserModel**();**  **}**  public String createAction**()** **{**  **if** **(!**validate**())** **{**  **return** **null;**  **}**  // removed  // usersBean.getUsers().add(user);    String cryptedPassword **=** GeneralUtils**.**encodeSha256Password**(**user**.**getPassword**());**  user**.**setPassword**(**cryptedPassword**);**    userService**.**save**(**user**);**  **return** "/page/listUsers?faces-redirect=true"**;**  **}**  /\*  \* public UsersBean getUsersBean() { return usersBean; }  \*  \* public void setUsersBean(UsersBean usersBean) { this.usersBean =  \* usersBean; }  \*  \* public UserDto getUser() { return user; }  \*  \* public void setUser(UserDto user) { this.user = user; }  \*/  public UserModel getUser**()** **{**  **return** user**;**  **}**  public void setUser**(**UserModel user**)** **{**  **this.**user **=** user**;**  **}**  protected boolean validate**()** **{**  boolean hasErrors **=** **false;**  **if** **(**StringUtils**.**isBlank**(**user**.**getUsername**()))** **{**  MessageUtils**.**addErrorMessage**(**"error.required.username"**);**  hasErrors **=** **true;**  **}**  **if** **(**StringUtils**.**isBlank**(**user**.**getPassword**()))** **{**  MessageUtils**.**addErrorMessage**(**"error.required.password"**);**  hasErrors **=** **true;**  **}**  **if** **(**StringUtils**.**isBlank**(**user**.**getFirstName**()))** **{**  MessageUtils**.**addErrorMessage**(**"error.required.firstname"**);**  hasErrors **=** **true;**  **}**  **if** **(**StringUtils**.**isBlank**(**user**.**getLastName**()))** **{**  MessageUtils**.**addErrorMessage**(**"error.required.lastname"**);**  hasErrors **=** **true;**  **}**  **if** **(**StringUtils**.**isBlank**(**user**.**getEmail**()))** **{**  MessageUtils**.**addErrorMessage**(**"error.required.email"**);**  hasErrors **=** **true;**  **}**  pattern **=** Pattern**.**compile**(**EMAIL\_PATTERN**);**  **if** **(!**pattern**.**matcher**(**user**.**getEmail**()).**matches**())** **{**  MessageUtils**.**addErrorMessage**(**"error.invalid.email"**);**  hasErrors **=** **true;**  **}**  **if** **(**hasErrors**)** **{**  **return** **false;**  **}**  **return** **true;**  **}**  /\*\*  \* Verifies if a error messages are present in the context  \*/  public boolean hasErrors**()** **{**  Iterator**<**FacesMessage**>** messages **=** FacesContext**.**getCurrentInstance**().**getMessages**();**  **for** **(;** messages**.**hasNext**();)** **{**  FacesMessage message **=** messages**.**next**();**  **if** **(**message**.**getSeverity**().**compareTo**(**FacesMessage**.**SEVERITY\_ERROR**)** **==** 0**)** **{**  **return** **true;**  **}**  **}**  **return** **false;**  **}**  **}** |

## Delete UsersBean and UserDto. They will no longer be used

## Modify AuthenticationFilter to use UserModel instead of UserDto

## Display posts count in ListUsers grid