Use Cases: *User management*

David Bialik, Julian Fleischer, Hagen Mahnke,
Konrad Reiche, André Zoufahl

March 4, 2011

	Admin creates a new user account
Actors	A user with sufficient rights (typically called admin)
Scope	
Precondition	The system is up and running, the user who wants to create a new user account is logged in
Postcondition	
Postcondition on Success	The new user account is created
Basic Course of Events	**1.** The user navigates to the "create user" page. **2.** A form is displayed which offers to create a new user, including first-name, middle-name, last-name, login-name, email-address, password, whether it is a student or a lecturer, etc. Most importantly it should also be possible to select roles and privileges from a list to assign to the user. **3.** The form is submitted. **4.** The new user account is created. The user is shown another empty form for the creation of another user.
Alternative Paths	**1a.** With insufficient rights, the user should not be able to navigate there at all. **4a.** The new account is not created, due to malformed data (bad syntax, invalid values). In this case the same form with hints on what went wrong should be displayed again. **4b.** A database exception occurs. The user should be informed about it, the data should not be lost, i.e. the same form should be shown again (maybe in a way that it can easily be retried).
Open Questions	**Q:** Should we extend a user account by a "home department"?
Solved issues	**Q:** Should we extend a user account by the option of making him a super-user (super-user do have all rights) **A:** Yes.
Implementation Notes	-
Implementation Status	• Implemented, except for checking of rights.

	Edit another user account
Actors	A user with sufficient rights
Scope	
Precondition	The system is up and running, the user who wants to edit is logged in.
Postcondition	
Postcondition on Success	The user account to be edited gets updated by the values provided by the editing user. A page informaing about the success is shown.
Basic Course of Events	**1.** The user selects a user account for editing. This is typically achived by clicking an "edit"-link in an appropriate list (search result, user listing, etc.). **2.** The edit-user form is shown, with the current account properties filled in. However, certain special fields, i.e. "password" are specially protected. If the user does also have the right to edit those fields, these fields should also be shown. If the user does not, these should simply not be shown at all. **3.** The user changes the values. **4.** The form is checked by the application according to the definition set up for it (see Forms). **5.** If the information is not outdated (e.g. a newer record has been saved in between), the new values are to be written to the database.
Alternative Paths	**1a.** The user does not have sufficient rights to edit another users account. Edit links should not be displayed at all then. **2a.** If the user has gotten *somehow* to that page (e.g. by copying a link) and does not have sufficient rights to do so, a page that explains that the user does not have sufficient rights should be shown. **5a.** If the information *is* outdated, it should not be written to the database. However, the information should not get lost. Thus is would be best to display the form whose values could not be updated again. **5b.** It would be even better to show the differences and assist in merging the two (Wikipedia does have such a feature for so-called "edit conflicts").
Open Questions	-
Solved issues	-
Implementation Notes	• A transaction should be used to save the values. • The "timekey" attribte has special support to identify outdated records. • Currently there are two privileges "edit_user_accounts", "edit_user_passwords".
Implementation Status	• Implement, rights are party respected already.

	Create a Role
Actors	A user with sufficient rights
Scope	
Precondition	The system is up and running, the user who wants to create the role is logged in.
Postcondition	
Postcondition on Success	The new role is created.
Basic Course of Events	**1.** The user navigates to the "create role" form. **2.** The user enters the name of the role and selects privileges which should be associated with this role. **3.** The user submits the form. **4.** The system displays the successful creation of the role and association of privileges.
Alternative Paths	**4a.** If a role with the same name already exists, the role is not created but a hint on the issue is shown. **4b.** If some error occurs (like connection to the database is lost), a message informing the user about the error is shown. **4c.** The system detects that the user does not have sufficient rights to create a role. While a user should not be able to navigate to pages he does not have the right to access, it is (in corner-cases) possible to happen so; for example if a user navigated to such a page and the right to use that page is revoked while he is working on that form.
Open Questions	-
Solved issues	-
Implementation Notes	• Creation of the role and creation of the roleImpliesPrivilege-relations should happen within a single transaction. If one of these fails, the users request can not be completed as a whole and therefor nothing should be done at all. This way we do not have half-baked relations within the database.
Implementation Status	-

	Edit a Role
Actors	
Scope	GUI
Precondition	The system is running and the user is logged in with the rights of an administrator.
Postcondition	The system is still running, the user is still logged in.
Postcondition on Success	The role is altered.
Basic Course of Events	**1.** User opens the role panel. **2.** User clicks on edit role. **3.** User changes the data of the role. **4.** User clicks on submit. **5.** GUI updates the role into the database.
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Assign a Privilege to a Role
Actors	Administrator
Scope	GUI
Precondition	The system is running and the user is logged in with the rights of an administrator.
Postcondition	The system is still running, the user is still logged in.
Postcondition on Success	The role has a new privilege assigned.
Basic Course of Events	**1.** User opens the role panel. **2.** User clicks on edit role. **3.** User selects a privilege. **4.** User clicks on submit. **5.** GUI inserts the new association between privilege and role into the database.
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Assign an Attribute to a Role
Actors	Administrator, GUI, Database
Scope	GUI
Precondition	The system is running and the user is logged in with the rights of an administrator.
Postcondition	The system is still running, the user is still logged in.
Postcondition on Success	The role has a new attribute assigned.
Basic Course of Events	**1.** User opens the role panel. **2.** User clicks on edit role. **3.** User selects an attribute. **4.** User clicks on submit. **5.** GUI inserts the new association between attribute and role into the database.
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Assign a Role to a User
Actors	Administrator, GUI, Database
Scope	GUI
Precondition	The system is running and the user is logged in with the rights of an administrator.
Postcondition	The system is still running, the user is still logged in.
Postcondition on Success	The user has a new role assigned.
Basic Course of Events	**1.** User opens the user panel. **2.** User clicks on edit user. **3.** User selects a role. **4.** User clicks on submit. **5.** GUI inserts the new association between user and role into the database.
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Remove a Role form a User
Actors	Administrator, GUI, Database
Scope	GUI
Precondition	The system is running and the user is logged in with the rights of an administrator.
Postcondition	The system is still running, the user is still logged in.
Postcondition on Success	The user has a new role assigned.
Basic Course of Events	**1.** User opens the user panel. **2.** User clicks on edit user. **3.** User clicks on delete role for a defined role. **4.** GUI deletes the association between user and role into the database.
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Delete a Role
Actors	Administrator, GUI, Database
Scope	GUI
Precondition	The system is running and the user is logged in with the rights of an administrator.
Postcondition	The system is still running, the user is still logged in.
Postcondition on Success	The selected role is deleted.
Basic Course of Events	**1.** User opens the role panel. **2.** User clicks on delete role. **3.** User selects a role. **4.** User clicks on submit. **5.** GUI deletes the role from the database.
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Assign a Privilege to a User
Actors	Administrator, GUI, Database
Scope	GUI
Precondition	The system is running and the user is logged in with the rights of an administrator.
Postcondition	The system is still running, the user is still logged in.
Postcondition on Success	The user has a new privilege assigned.
Basic Course of Events	**1.** User opens the user panel. **2.** User clicks on edit user. **3.** User selects an privilege. **4.** User clicks on submit. **5.** GUI inserts the new association between user and privilege into the database.
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-