

**Dokumentation JProject**

*Entwurf und Implementierung einer Projektverwaltung mittels JEE 5 Webtechnologien*

**Team:**

*Andy Klay*

*Michael Koppen*

*Tino Reuschel*

Inhaltsverzeichnis

1. Einführung ..........................................................................................................................4

2. Portal-Funktionen .............................................................................................................5

2.1 Funktionale Sicht ...........................................................................................................5

2.2 Funktionen Detailsicht ................................................................................................7

3. Architektur ........................................................................................................................11

3.1 Schichten .......................................................................................................................11

3.2Komponenten ...............................................................................................................13

3.2.1 GUI .........................................................................................................................14

3.2.2 Controller ..............................................................................................................14

3.2.3 Manager .................................................................................................................14

3.2.4 Model .....................................................................................................................15

3.2.5 DA ..........................................................................................................................16

3.2.6 DAO .......................................................................................................................16

3.2.7 Datenbank .............................................................................................................16

4. Systemvoraussetzungen ..................................................................................................18

4.1 Allgemeine Voraussetzungen .....................................................................................18

5. Anhang ..............................................................................................................................19

A1 Datenbank ERR-Diagramm ..........................................................................................20

A2 Sequenz-Diagramm ......................................................................................................21

A3 Package-Diagramm ......................................................................................................22

A4 Klassendiagrammübersicht ............................................................................................23

**Dokument-Historie**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Datum** | **Autor** | **Erläuterung** |
| 1.0 | 09.07.2011 | Klay,Koppen |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**1. Einführung**

JProject ist in erster Linie eine allgemeine Projektverwaltungsumgebung die von jedem Rechner, der Internet hat, erreicht werden kann.

JProject wird in 3 Teile unterteilt: JProject, Content und Wiki.

Mit JProject kann eine Benutzergruppe Teams zur Realisierung eines Projektes bilden und sich selbst organisieren. Die User können Projekte erstellen, Dokumente/Sourcecode hochladen und Aufgaben anlegen und zuordnen. Dazu steht ein flexibles Rechtesystem zur Seite.

Zum späteren Funktionsumfang soll eine Umfangreiche Datenbank mit Hilfestellungen, Anleitungen bzw. Tutorials zur Verfügung stehen. Dieser Bereich nennt sich Content.

Des Weiteren soll später ein Bereich namens Wiki eingerichtet werden, in dem die User allgemeine Informationen zum Ablauf und zur Organisation, in deren Herarchie das Projekt entworfen werden soll, finden.

***Funktionaler Umfang der Idee:***

- Rechtesystem für die Projekte

- Öffentlicher Bereich für Projekte

- Download/Upload des Contents (später mit Versionskontrolle)

- Synchrones und Verlustfreies arbeiten am Projekt

- Termin- und Aufgabenverwaltung innerhalb der Projekte



(Abb. 1 Ursprünglicher Ideen-Entwurf)

**2. Portal-Funktionen**

***2.1 Funktionale Sicht/User-Sicht auf das Portal***

In dem Usecase-Diagramm(Abb. 2) ist nur eine starre Konstellation der Rollen zu erkennen.

Implementiert wurde ein Rollensystem welches erlaubt dynamisch Rollen hinzu zufügen.

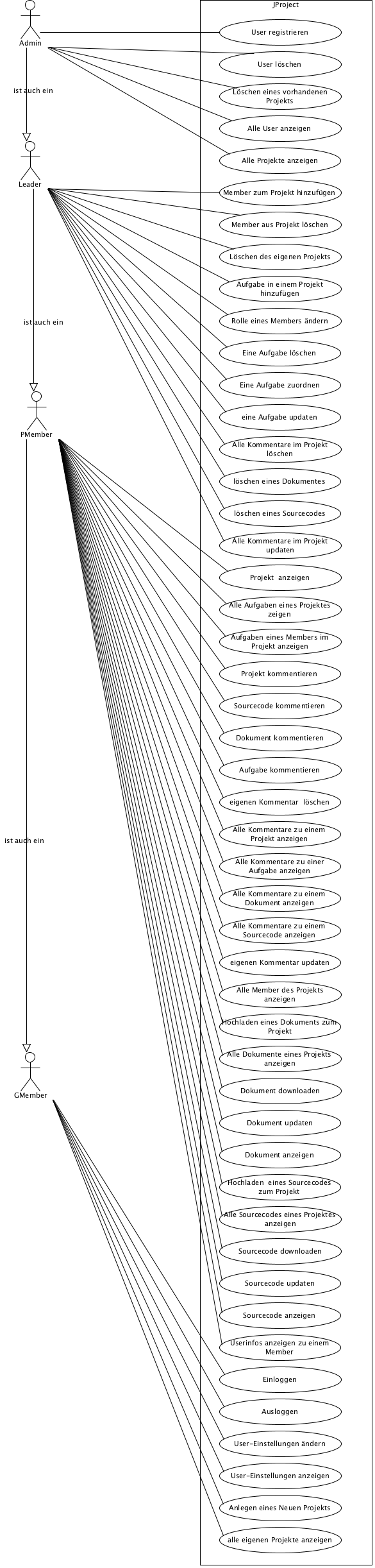
Es gibt ein globales Rollensystem (GlobalRoles). In ihm wird unterschieden welche Aktionen der Benutzer im Allgemeinen tätigen darf. Beispielsweise ist hier im Usecase-Diagramm zu sehen der GMember (User ohne Projekt) und der Admin, dies sind zwei mögliche globale Rollen.

Als Weiteres gibt es aber noch ein Rollensystem nur für Projekte (ProjectRoles). In ihm wird unterschieden welche Aktionen der Benutzer innerhalb eines Projektes tätigen darf. Als Beispiel dafür ist im Usecase-Diagramm der Leader und der PMember zu sehen.

Die globalen Rollen und Projektrollen sind für den Betrieb des Systems zwingend erforderlich d.h. es sind mindestens 2 Rollen global (Admin und Member)und 2 Projektrollen (Leader und Member) erforderlich.

Der Admin hat durch seine globale Rolle eine übergeordnete Funktion und darf grundsätzlich alles was andere Rollen auch dürfen außer Tasks zuordnen, dann kommt der Projekt-Leader, der innerhalb eines Projektes wiederum mehr Rechte hat als ein Projekt-Member. User die in einem Projekt nicht involviert sind, also weder Member noch Leader (GMember) sind und haben daher nur grundlegende Aktionen wie z.B. Einloggen, Ausloggen, Projekt anlegen.

Die Rollen können derzeit von Ersteller des Systems umbenannt und auch deren Rechte dynamisch geändert oder auch komplett neue Rollen hinzugefügt werden.

 (Abb. 2 UseCase-Diagramm)

***2.2 Funktionen Detailsicht***

Es wird hier im Folgenden nicht mehr immer explizit erwähnt, dass alle Aktionen die der Projekt-Leader machen kann, auch der Admin über die AdminConsole tätigen kann. Ausgeschlossen davon ist das zuordnen von Tasks an PMember.

## Zusammenfassung

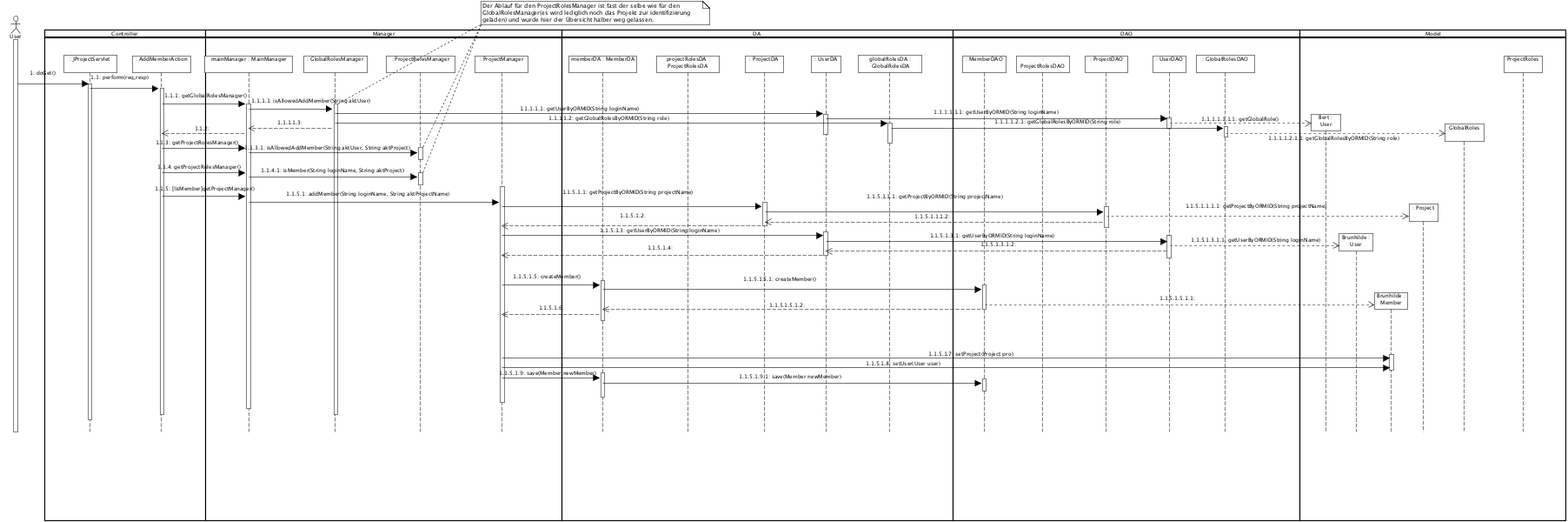
|  |  |
| --- | --- |
| Name | Dokumentation |
| Image1.png Admin | **Admin**= d.h. also ein User mit entsprechenden Rechten, muss nicht Admin heißen. |
| Image2.png User löschen | Über die AdminConsole kann ein Admin einen User aus dem System löschen, damit dieser das System nicht mehr nutzen kann. |
| Image2.png Alle Projekte anzeigen | Nur der Admin kann über die AdminConsole alle Projekte anzeigen. |
| Image2.png Alle User anzeigen | Nur der Admin kann über die AdminConsole alle User anzeigen. |
| Image2.png Löschen eines vorhandenen Projekts | Ein Projekt kann direkt gelöscht werden vom Admin über die AdminConsole. |
| Image2.png User registrieren | Über die AdminConsole kann ein Admin einen User im System erstellen, damit dieser das System nutzen kann. |
| Image1.png Leader | **Projekt-Leader**=d.h. also ein User mit entsprechenden Rechten der in einem Projekt mehr Rechte hat als ein einfacher Teilnehmer(**Projekt-Member**). |
| Image2.pngEine Aufgabe updaten | Der Projekt-Leader kann über das Projekt und über den Reiter „Task“ eine Aufgabe des Projekts updaten. Durch Auswahl des Tasks links, dem Eintragen der neuen Daten rechts und dem Klicken auf Update-Button unten kann dies bewerkstelligt werden. |
| Image2.png Eine Aufgabe zuordnen | Der Projekt-Leader kann über das Projekt und über den Reiter „Task“ eine Aufgabe des Projekts einem Member zuordnen. Durch die Auswahl der vorhandenen Aufgabe links und Auswahl des Members rechts und klicken auf Zuweisen kann dies bewerkstelligt werden. |
| Image2.png Löschen des eigenen Projekts | Ein Projekt kann gelöscht werden, indem der Projekt-Leader alle Member des Projektes löscht und zuletzt sich selbst. |
| Image2.png Member zum Projekt hinzufügen | Der Projekt-Leader kann über das Projekt einen User als Projekt-Member hinzufügen. |
| Image2.pnglöschen eines Sourcecodes | Das Mitglied mit den benötigten Rechten, kann über das Projekt und dann Reiter „Sourcecode“ die Sourcecodes des Projektes anzeigen lassen. Durch Klick auf das Kreuz vor dem Namen des Sourcecodes wird die Sourcecode Datei gelöscht. |
| Image2.png Alle Kommentare im Projekt updaten | Der Projekt-Leader kann alle Kommentare eines Projektes updaten (alle Kommentare zu allen Tasks, allen Dokumenten, allen Sourcecodes und die Kommentare des Projektes selber)durch das Anzeigen eines Kommentars, dem Ändern des Contents im Textfeld und dem Klick auf den Update Button darunter. |
| Image2.png Eine Aufgabe löschen | Der Projekt-Leader kann über das Projekt und über den Reiter „Task“ in der Leiste links eine Aufgabe des Projekts löschen durch klicken auf das Kreuz neben einer Aufgabe. |
| Image2.png Member aus Projekt löschen | Der Projekt-Leader kann über das Projekt einen Projekt-Member aus dem Projekt entfernen. |
| Image2.png Alle Kommentare im Projekt löschen | Der Projekt-Leader kann über das Projekt alle Kommentare zu allen Tasks, allen Dokumenten, allen Sourcecodes und die Kommentare des Projektes selber löschen durch das Anzeigen dieser Kommentare und dem Button Delete darüber. |
| Image2.png Aufgabe in einem Projekt hinzufügen | Der Projekt-Leader kann über das Projekt und über den Reiter „Task“ eine Aufgabe zum Projekt hinzufügen. |
| Image2.pngLöschen eines Dokumentes | Das Mitglied mit den benötigten Rechten, kann über das Projekt und dann Reiter „Dokument“ die Dokumente des Projektes anzeigen lassen. Durch Klick auf das Kreuz vor dem Namen des Dokumentes wird das Dokument gelöscht. |
| Image2.png Rolle eines Members ändern | Der Projekt-Leader kann über das Projekt und über den Reiter „Project“ im zweiten Abschnitt die Rolle der vorhandenen Member im Projekt ändern. Des Weiteren kann der Leader seine eigene Rolle im Projekt ändern, jedoch muss min. ein Projekt-Leader vorhanden sein. |
| Image1.png GMember | Grund-User, der kein Projektteilnehmer ist |
| Image2.png Einloggen | Wenn man die Startseite des Systems aufruft so findet man rechts in der Navigationsleiste ein Eingabefeld für den Benutzernamen dort trägt man auch sein Benutzernamen ein, des Weiteren findet man ein Eingabefeld für das Passwort, wo man sein Passwort einträgt und dann klickt man auf „einloggen“. |
| Image2.png Ausloggen | Wenn ein User eingeloggt ist, so findet er rechts in der Navigationsleiste den Button „ausloggen“. |
| Image2.png User-Einstellungen ändern | Wenn ein User eingeloggt ist so findet er links von Button zum ausloggen, ein Link zu seinen User-Einstellungen, dort kann er Änderungen vornehmen und danach speichern. |
| Image2.png User-Einstellungen anzeigen | Wenn ein User eingeloggt ist so findet er links von Button zum ausloggen, ein Link zu seinen User-Einstellungen, wenn er dort drauf klickt so werden ihm seine User-Einstellungen angezeigt. |
| Image2.png Anlegen eines Neuen Projekts | Wenn ein User sich gerade eingeloggt hat so kann er auf dieser Seiter ein neues Projekt anlegen, indem er rechts den Namen des Projektes einträgt und dann auf „addnew Project“ klickt. |
| Image2.pngalle eigenen Projekte anzeigen | Wenn ein User eingeloggt ist, so werden alle seine Projekte auf der linken Seite unter dem Punkt Projekte angezeigt. |
| Image1.png PMember | Ist ein einfacher Projektteilnehmer(**Projekt-Member**). |
| Image2.png Sourcecode anzeigen | Ein Mitglied eines Projektes kann über Projekt und den Reiter „Sourcecode“ sich alle Sourcecodes anzeigen lassen, wenn er nun auf den Namen eines Dokumentes klickt, so wird ihm das Dokument angezeigt. |
| Image2.png Alle Sourcecodes eines Projektes anzeigen | Ein Mitglied eines Projektes kann über Projekt und den Reiter „Sourcecode“ sich alle Sourcecodes anzeigen lassen. |
| Image2.pngProjekt anzeigen | Sobald der User sich ans System eingeloggt hat, werden seine Projekte auf der linken Seite angezeigt. |
| Image2.png Dokument downloaden | Ein Mitglied eines Projektes kann über Projekt und den Reiter „Dokumente“ sich alle Dokumente anzeigen lassen, wenn er nun auf den Namen eines Dokumentes klickt wird ihm das Dokument angezeigt, dort findet er dann auch den Download Button. |
| Image2.png Aufgabe kommentieren | Wenn ein Mitglied eines Projektes sich eine Aufgabe anzeigen lässt, so findet er am unteren Ende die Kommentare und kann selber ein Kommentar schreiben. |
| Image2.png Sourcecode downloaden | Ein Mitglied eines Projektes kann über Projekt und den Reiter „Sourcecode“ sich alle Sourcecodes anzeigen lassen, wenn er nun auf den Namen eines Sourcecodes klickt wird ihm der Sourcecode angezeigt, dort findet er dann auch den Download Button. |
| Image2.png Alle Kommentare zu einem Dokument anzeigen | Wenn ein Mitglied eines Projektes sich ein Dokument anzeigen lässt, so findet er am unteren Ende alle Kommentare, wenn man auf „show Comments“ klickt. |
| Image2.png Sourcecode updaten | Ein Mitglied eines Projektes kann über Projekt und den Reiter „Sourcecode“ sich alle Sourcecodes anzeigen lassen, wenn er nun auf den Namen eines Sourcecodes klickt wird ihm der Sourcecode angezeigt, dort findet er dann auch den Update Button. |
| Image2.png Dokument kommentieren | Wenn ein Mitglied eines Projektes sich ein Dokument anzeigen lässt, so findet er am unteren Ende die Kommentare und kann selber ein Kommentar schreiben. |
| Image2.png Dokument anzeigen | Ein Mitglied eines Projektes kann über Projekt und den Reiter „Dokumente“ sich alle Dokumente anzeigen lassen, wenn er nun auf den Namen eines Dokumentes klickt, so wird ihm das Dokument angezeigt. |
| Image2.png Dokument updaten | Ein Mitglied eines Projektes kann über Projekt und den Reiter „Dokumente“ sich alle Dokumente anzeigen lassen, wenn er nun auf den Namen eines Dokumentes klickt wird ihm das Dokument angezeigt, dort findet er dann auch den Update Button. |
| Image2.png Aufgaben eines Members im Projekt anzeigen | Ein Member des Projektes sieht auf der Projekt-Seite durch das Anklicken eines Members im Member-Abschnitt unten dann auch dessen zugewiesenden Aufgaben. |
| Image2.png Alle Kommentare zu einem Sourcecode anzeigen | Wenn ein Mitglied eines Projektes sich ein Sourcecode anzeigen lässt, so findet er am unteren Ende alle Kommentare, wenn man auf „show Comments“ klickt. |
| Image2.png Sourcecode kommentieren | Wenn ein Mitglied eines Projektes sich ein Sourcecode anzeigen lässt, so findet er am unteren Ende die Kommentare und kann selber ein Kommentar schreiben. |
| Image2.png Alle Member des Projekts anzeigen | Der Projekt-Leader sieht wenn er auf das Projekt klickt alle Member des Projektes. |
| Image2.png Alle Dokumente eines Projekts anzeigen | Ein Mitglied eines Projektes kann über Projekt und den Reiter „Dokumente“ sich alle Dokumente anzeigen lassen. |
| Image2.pngHochladen eines Sourcecodes zum Projekt | Ein Mitglied eines Projektes kann über Projekt und den Reiter “Sourcecode“ ein Sourcecode hochladen, dazu muss er dann auf „Durchsuche“ klicken und sucht sich die Datei raus die man hochladen möchte. |
| Image2.png Alle Aufgaben eines Projektes zeigen | Jedes Mitglied eines Projektes kann über den Reiter „Task“ sich alle Aufgaben eines Projektes anzeigen lassen. |
| Image2.pngeigenen Kommentar löschen | Wenn ein Mitglied eines Projektes ein Kommentar geschrieben hat, hat er die Möglichkeit diesen Kommentar auch wieder zu löschen, indem er sich den Kommentar anzeigt und auf das dazugehörige Kreuz klickt. |
| Image2.pngeigenen Kommentar updaten | Der Member eines Projektes kann ein eigenen Kommentar updaten durch das Anzeigen eines Kommentars, dem Ändern des Contents im Textfeld und dem Klick auf den Update Button darunter. |
| Image2.png Projekt kommentieren | Klickt ein User auf eines seiner Projekte, so wird ihm das Projekt angezeigt und am unteren Ende findet er alle Kommentare zum Projekt und kann dort auf ein eigenen Kommentar schreiben |
| Image2.png Alle Kommentare zu einer Aufgabe anzeigen | Wenn ein Mitglied eines Projektes sich eine Aufgabe anzeigen lässt, so findet er am unteren Ende alle Kommentare, wenn man auf „show Comments“ klickt. |
| Image2.png Hochladen eines Dokuments zum Projekt | Ein Mitglied eines Projektes kann über Projekt und den Reiter „Dokumente“ ein Dokument hochladen, dazu muss er dann auf „Durchsuche“ klicken und sucht sich die Datei raus die man hochladen möchte. |
| Image2.png Userinfos anzeigen zu einem Member | Wenn man auf den Namen eines Member klickt, so werden Informationen zu diesem Member angezeigt. |
| Image2.png Alle Kommentare zu einem Projekt anzeigen | Wenn ein User auf eines seiner Projekte klickt, so wird ihm das Projekt angezeigt und am unteren Ende kann er über „show Comments“ sich alle Kommentare anzeigen lassen. |

**3. Architektur**

***3.1 Schichten***



(Abb.3 Schichten-Übersicht)



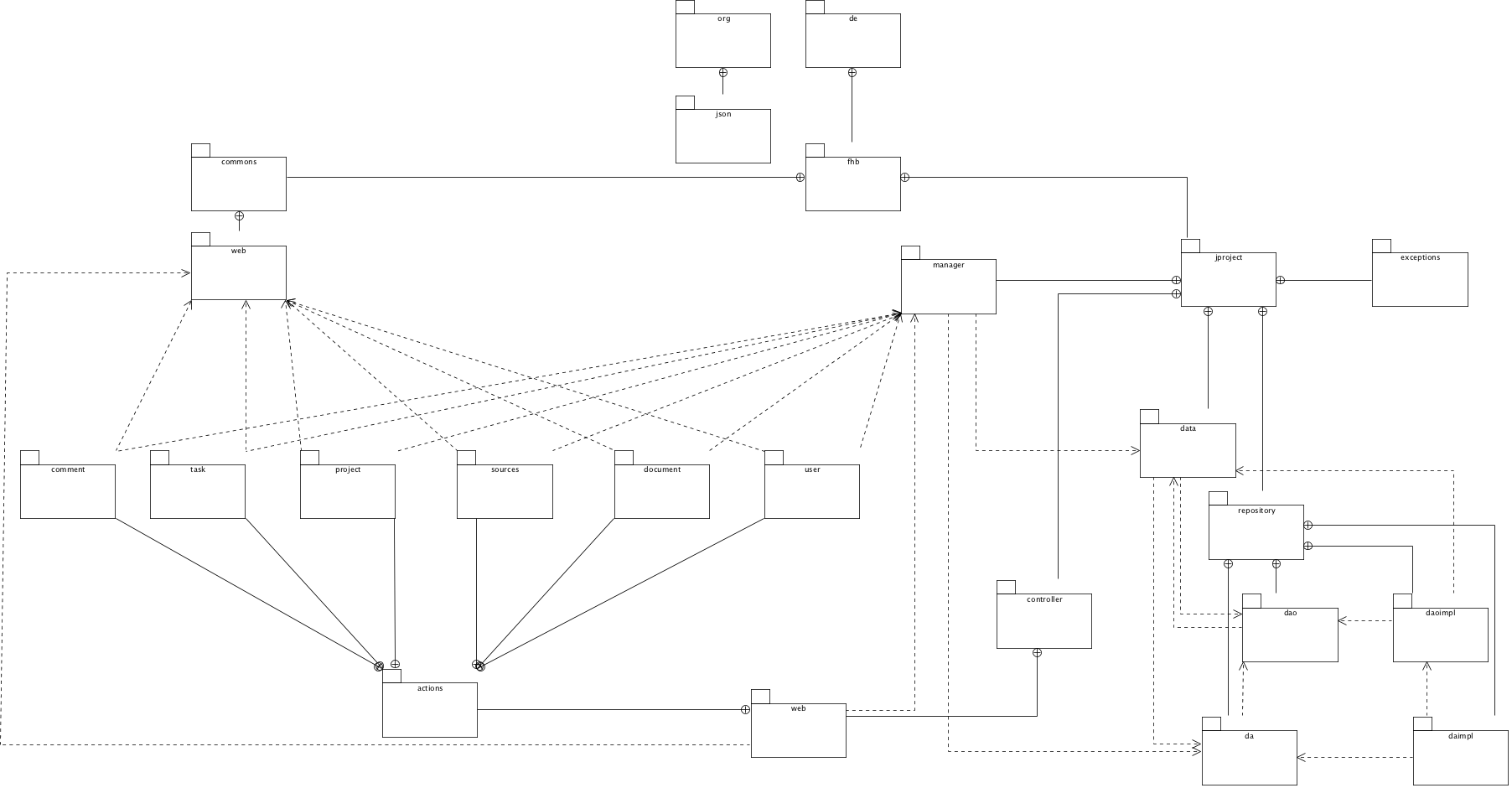
(Abb. 4 Sequenz-Diagramm)

**Beschreibung des Anwendungsfalls mit Sequenzdiagramms**

|  |  |
| --- | --- |
| Name | AddMemberAction |
| Kurzbeschreibung | Dieses Sequenz-Diagramm(Abb. 2) stellt dar, wie der ein Projekt-Leader(Bert: User bzw. Bert : Member) einen Member(Brunhilde : User bzw. Brunhilde : Member) zu seinem Projekt hinzufügt. |
| Akteure | User |
| Auslöser | Eintragung des Usernamen und Klicken auf Button „Member hinzufügen“ |
| Vorbedingung | User eingeloggt, User hat Projekt indem er Leader ist, der hinzu zu fügende User ist nicht bereits im Projekt |
| Ergebnis | User wird dem Projekt zugeordnet(Member wurde erstellt) |
| Nachbedingung | keine |
| Essentieller Ablauf | Ablauf:   * ***1. doGet()*** * Der Request wird an das JProjectServlet gesendet . * Es überprüft ob der User einer Session zugeordnet ist. * Es ordnet dem Request die Aktion „AddMemberAction“ zu. * ***1.1 perform(req, resp)*** * Die Action zieht aus dem Request alle erforderlichen Parameter heraus. * Die Action AddMemberAction überprüft ob der User eingeloggt ist. * ***1.1.1 getGlobalRolesManager()*** und folgende -   Die Action überprüft, ob der User Globale-Rechte hat um diese Action auszuführen.   * ***1.1.3 getProjectRolesManager()*** und folgende   Die Action überprüft, ob der User Projekt-Rechte hat um die Action auszuführen, sofern er nicht schon Globale Rechte hatte.   * ***1.1.4 getProjectRolesManager()*** und folgende -   Die Action überprüft, ob der angegebene User eventuell schon Member in diesem Projekt ist.   * ***1.1.5 getProjectManager()***   Sofern der User nicht Member in diesem Projekt ist, wird der Member hinzugefügt (1.1.5.1)   * ***1.1.5.1.1 getProjectByORMID(String projectName)***und folgende - Das Projekt wird geladen. * 1***.1.5.1.3 getUserByORMID(String loginName)***   und folgende -  Der hinzu zu fügende User wird geladen.   * ***1.1.5.1.5 createMember()*** und folgende -   Ein neuer Member wird erzeugt.   * ***1.1.5.1.7 setProject(Project pro) –*** Das Attribut „Projekt“(PK) im neuen Member wird gesetzt. * ***1.1.5.1.8 setUser(User user) –*** Das Attribut „User“(PK) im neuen Member wird gesetzt. * ***1.1.5.1.9 save(Member newMember)*** und folgende – Der neue Member wird in die Datenbank gespeichert. |

## *3.2 Komponenten*

# ***Paketdiagramm***



(Abb. 5Paket-Diagramm)

## Zusammenfassung

|  |  |
| --- | --- |
| Name | Dokumentation |
| Image12.pngorg | Sprachraum/Organisationform |
| Image12.pngde | Sprachraum/Organisationform |
| Image12.pngjson | beinhaltet Klassen für die JSON-Funktionalität |
| Image12.pngcommons | beinhaltet allgemeine Klassen für immer wiederkehrende Aufgaben |
| Image12.pngfhb | Organisation |
| Image12.pngweb | Allgemeine Klassen bezogen auf das Web |
| Image12.pngjproject | Projektpackage - beinhaltet alles Projektspezifisches |
| Image12.pngexceptions | beinhaltet eigene Exceptions für JProject |
| Image12.pngmanager | beinhaltet alle Manager die für JProject benötigt werden |
| Image12.pngdata | beinhaltet alle Models/VO´s für JProject |
| Image12.pngcomment | beinhaltet alle Actions die sich auf Kommentare beziehen |
| Image12.pngtask | beinhaltet alle Actions die sich auf Tasks beziehen |
| Image12.pngproject | beinhaltet alle Actions die sich auf Projekte beziehen |
| Image12.pngsources | beinhaltet alle Actions die sich auf Sourcecode beziehen |
| Image12.pngdocument | beinhaltet alle Actions die sich auf Dokumente beziehen |
| Image12.pnguser | beinhaltet alle Actions die sich auf den User beziehen |
| Image12.pngrepository | beinhaltet Zugriffsschichten zur Datenbank |
| Image12.pngcontroller | beinhaltet Controller für JProject |
| Image12.pngdao | beinhaltet alle Interfaces und Factory für die DAO-Schicht(Datenbankzugriff),  bietet die Funktionalität für Standardzugriffe auf die Datenbank |
| Image12.pngdaoimpl | beinhaltet alle Implementationen und Factory für die DAO-Schicht(Datenbankzugriff), bietet die Funktionalität für Standardzugriffe auf die Datenbank |
| Image12.pngactions | beinhaltet alle Actions für JProject |
| Image12.pngweb | beinhaltet alle Controller/Servlets die für das Web nötig sind |
| Image12.pngda | beinhaltet alle Interfaces und Factory für die DA-Schicht(Datenbankzugriff),  bietet die Funktionalität für erweiterte Zugriffe auf die Datenbank |
| Image12.pngdaimpl | beinhaltet alle Implementationen und Factory für die DA-Schicht(Datenbankzugriff).  bietet die Funktionalität für erweiterte Zugriffe auf die Datenbank |

**3.2.1 GUI**

Das GUI stellt die Schnittstelle für den User zum System. Das JProjectServlet und AdminServlet leiten alle Anfragen an die „index.jsp“. Diese zieht aus dem Response alle nötigen Informationen um zu wissen welche Informationen dargestellt werden sollen.

**3.2.2 Controller**

Die Controllerschicht mit Servlets und Actions regelt den Zugriff auf das System und regelt was gemacht wird bei welchem Zugriff und greift dann auf die Manager zu.

(weitere Details siehe Klassendiagramm im Anhang A4)

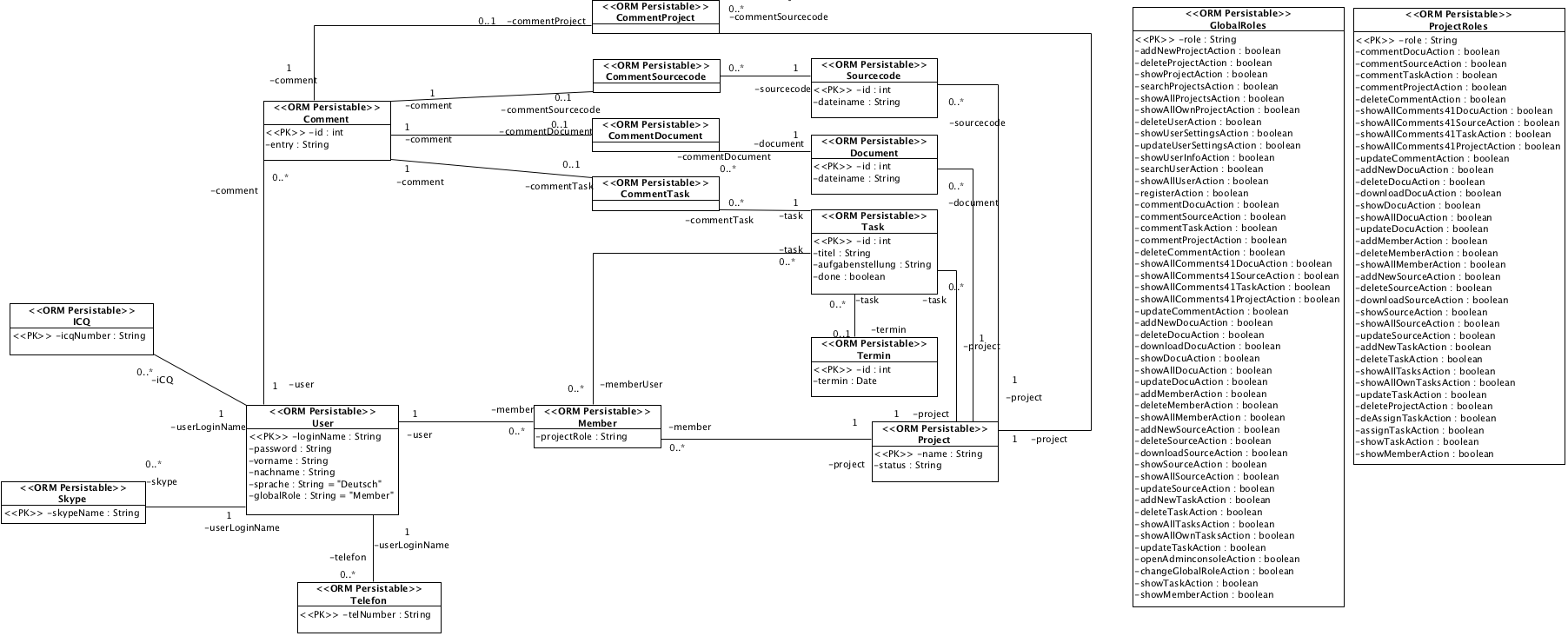
**3.2.3. Manager**

Die Manager regeln wie Zugriffe im System ablaufen und greifen über die DA-Sicht mit Operationen und Zugriffe auf das Model zu.

(weitere Details siehe Klassendiagramm im Anhang A4)

**3.2.4. Model**

# ***Domänenmodell***

(Abb.6Domänenmodell)

## Zusammenfassung

|  |  |
| --- | --- |
| Name | Dokumentation |
| Image5.png CommentProject | spezieller Kommentar zu einem Projekt |
| Image5.png GlobalRoles | Eine GlobalRole ist eine Rolle, die ein User im System einnehmen kann. D.h. welche Aktionen dieser tätigen kann und darf bzw. nicht darf. |
| Image5.pngProjectRoles | Eine ProjectRole ist eine projektspezifische Rolle, die ein User des Systems innerhalb eines Projekts spielen kann. D.h. welche Aktionen dieser tätigen kann und darf bzw. nicht darf in einem Projekt. |
| Image5.png Sourcecode | Ein Sourcecode ist ein in der Informatik, für Menschen lesbare, in einer Programmiersprache geschriebene Text eines Computerprogramms. |
| Image5.png CommentSourcecode | spezieller Kommentar zu einem Sourcecode |
| Image5.png Comment | Kommentar zu Sourcecode/Document/Project/Task. Beinhaltet spezielle Kommentare. |
| Image5.png CommentDocument | spezieller Kommentar zu einem Dokument |
| Image5.png Document | Ein Document ist eine vom Menschen verfasste Datei  mit Informationen, dessen Darstellung unterschiedlich sein kann. |
| Image5.png CommentTask | spezieller Kommentar zu einem Task |
| Image5.png Task | Ein Task ist eine Aufgabe die es in einem Projekt zu erledigen gilt. |
| Image5.png ICQ | ICQ stellt eine Adresse für einen Messenger dar. |
| Image5.png Termin | Ein Termin ist ein festgelegtes Kalenderdatum. |
| Image5.png User | Der User stellt jeglichen Benutzer des System dar mit nötigen Daten zum Einloggen und zum Identifizieren. |
| Image5.png Member | Member ist Assoziationsklasse zwischen User und Project.  Ein Member ist also ein User der in ein Projekt involviert ist. |
| Image5.png Project | Ein bis mehrere User können in einem Project involviert sein zum Lösen einer Problemstellung. |
| Image5.png Skype | Skype stellt eine Adresse für einen Messenger dar |
| Image5.png Telefon | ist eine Telefonnummer |

**3.2.5DA**

DA bietet für erweiterte Zugriffsfunktionalität auf die Datenbank.

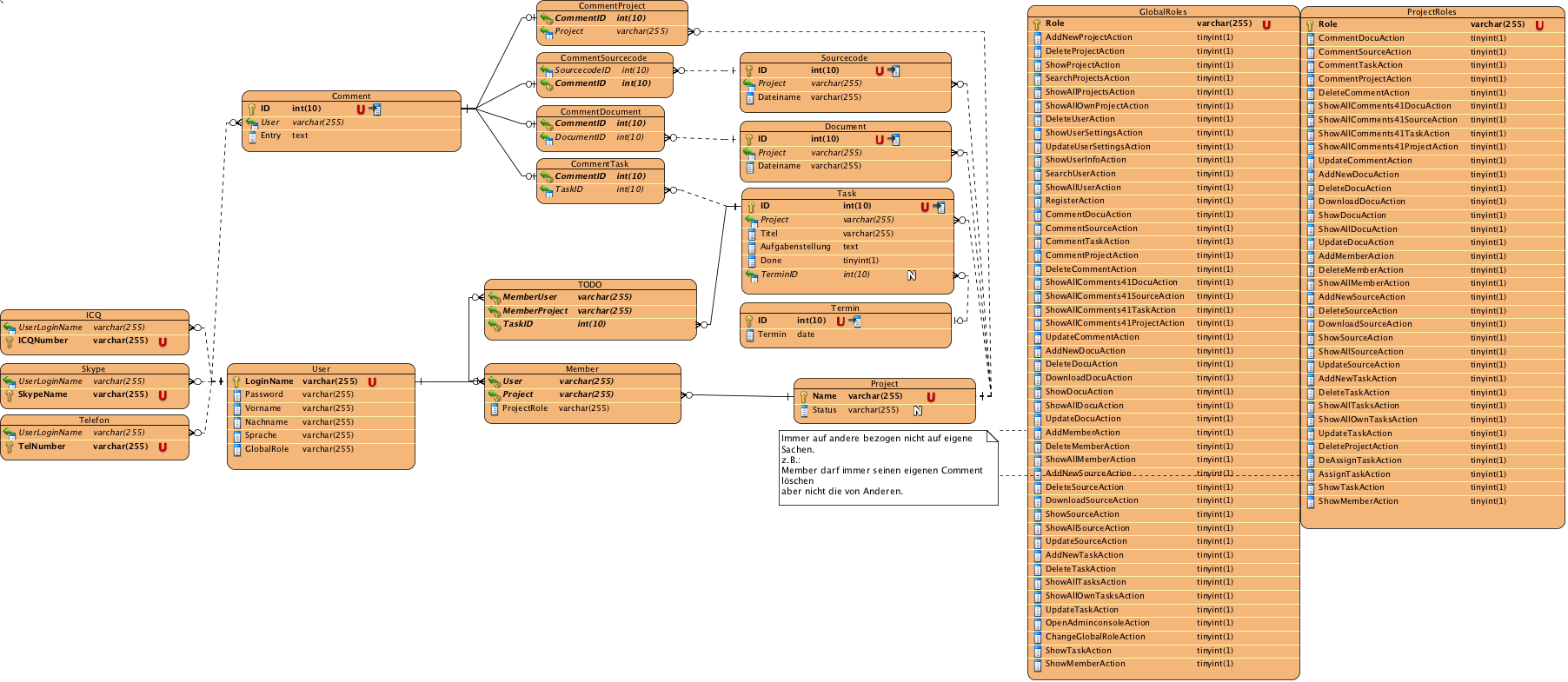
**3.2.6 DAO**

DAO bietet die Grundfunktionalität für Datenbankzugriffe.

**3.2.7Datenbank**

In der Datenbankschicht wird MySQL und Hibernate OR-Mapping genutzt.

# ***Entity-Relationship Diagramm***



(Abb.7 Entity-Relationship Diagramm)

## Zusammenfassung

|  |  |
| --- | --- |
| Name | Dokumentation |
| Image14.png CommentProject | spezieller Kommentar zu einem Projekt |
| Image14.png GlobalRoles | Eine GlobalRole ist eine Rolle, die ein User im System einnehmen kann. D.h. welche Aktionen dieser tätigen kann und darf bzw. nicht darf. |
| Image14.png ProjectRoles | Eine ProjectRole ist eine projektspezifische Rolle, die ein User des Systems innerhalb eines Projekts spielen kann. D.h. welche Aktionen dieser tätigen kann und darf bzw. nicht darf in einem Projekt. |
| Image14.png CommentSourcecode | spezieller Kommentar zu einem Sourcecode |
| Image14.png Sourcecode | Ein Sourcecode ist ein in der Informatik, für Menschen lesbare, in einer Programmiersprache geschriebene Text eines Computerprogramms. |
| Image14.png Comment | Kommentar zu Sourcecode/Document/Project/Task. Beinhaltet spezielle Kommentare. |
| Image14.png CommentDocument | spezieller Kommentar zu einem Dokument |
| Image14.png Document | Ein Document ist eine vom Menschen verfasste Datei  mit Informationen, deren Darstellung unterschiedlich sein kann. |
| Image14.png CommentTask | Spezieller Kommentar zu einem Task |
| Image14.png Task | Ein Task ist eine Aufgabe die es in einem Projekt zu erledigen gilt. |
| Image14.pngTODO | Zuordnung welcher Member welchen Task zu erledigen hat. |
| Image14.png Termin | Ein Termin ist ein festgelegtes Kalenderdatum. |
| Image14.png ICQ | ICQ stellt eine Adresse für einen Messenger dar. |
| Image14.png Skype | Skype stellt eine Adresse für einen Messenger dar |
| Image14.png User | Der User stellt jeglichen Benutzer des System dar mit nötigen Daten zum Einloggen und zum Identifizieren. |
| Image14.png Member | Member ist Assoziationsklasse zwischen User und Project.  Ein Member ist also ein User der in ein Projekt involviert ist. |
| Image14.png Project | Ein bis mehrere User können in einem Project involviert sein zum Lösen einer Problemstellung. |
| Image14.png Telefon | ist eine Telefonnummer |
| Image10.png Notiz zu den Rollen | Immer auf andere bezogen nicht auf eigene Sachen.  z.B.: Member darf immer seinen eigenen Comment löschen  aber nicht die von Anderen. |

**4. Systemvoraussetzungen**

***4.1 Allgemeine Voraussetzungen***

***Voraussetzungen für JProject:***

Das GUI wurde optimiert für Firefox 4 und 5.

Es wird mindestens Java Version 1.5 benötigt.

Es wird ein Container benötigt z.B. Glassfish oder Tomcat.

Des Weiteren wird eine MySQL Datenbank benötigt - ein entsprechender JDBC-Connector liegt bei.

***JProject benutzt folgende Frameworks und APIs:***

- Hibernate

- Log4j

- Visual Paradigm JPA (nur zum Öffnen der VPP-Dateien)

- JSTL

- Apache commons Fileupload& IO

- JavaScript mit Mootools

- MySQL JDBC-Connector

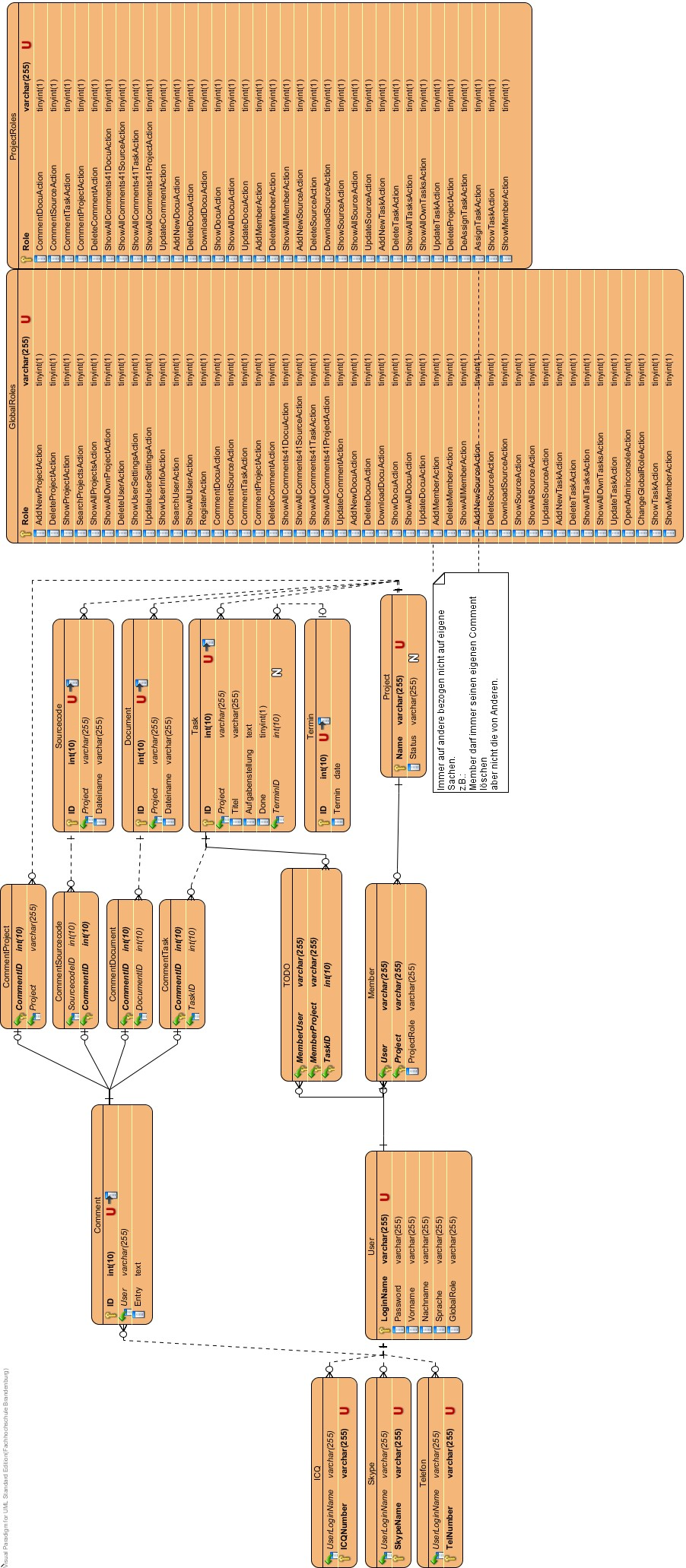
**5. Anhang**

A1 Datenbank ERR-Diagramm

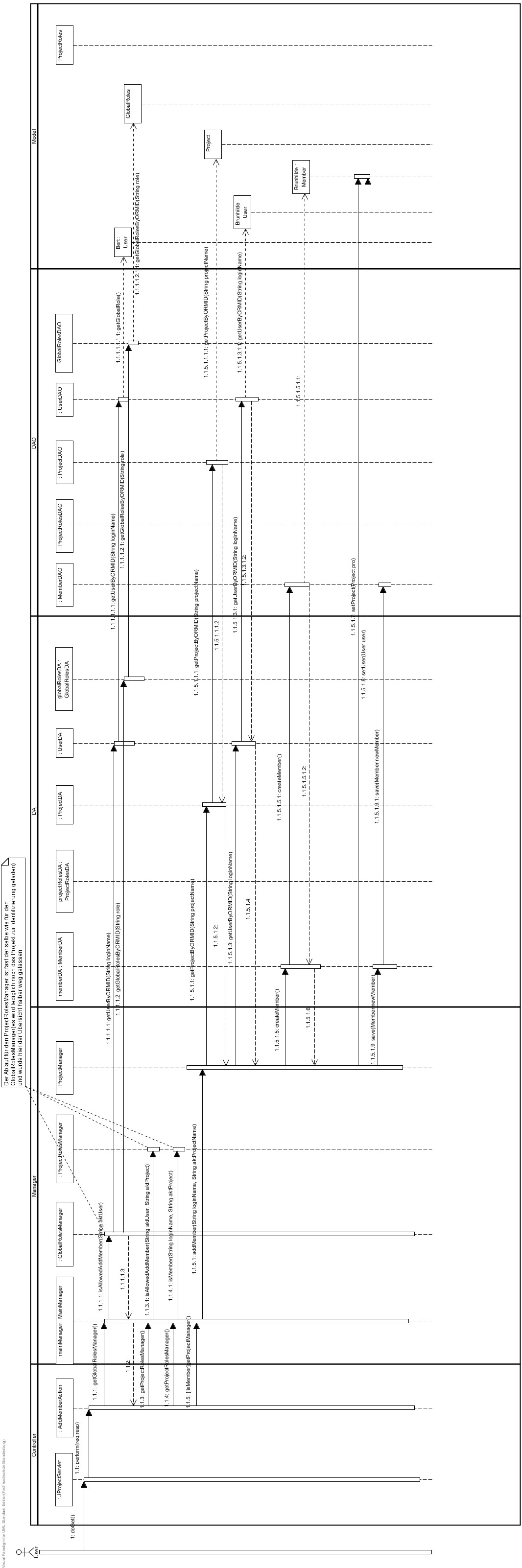
A2 Sequenz-Diagramm(AddMemberAction)

A3 Package-Diagramm

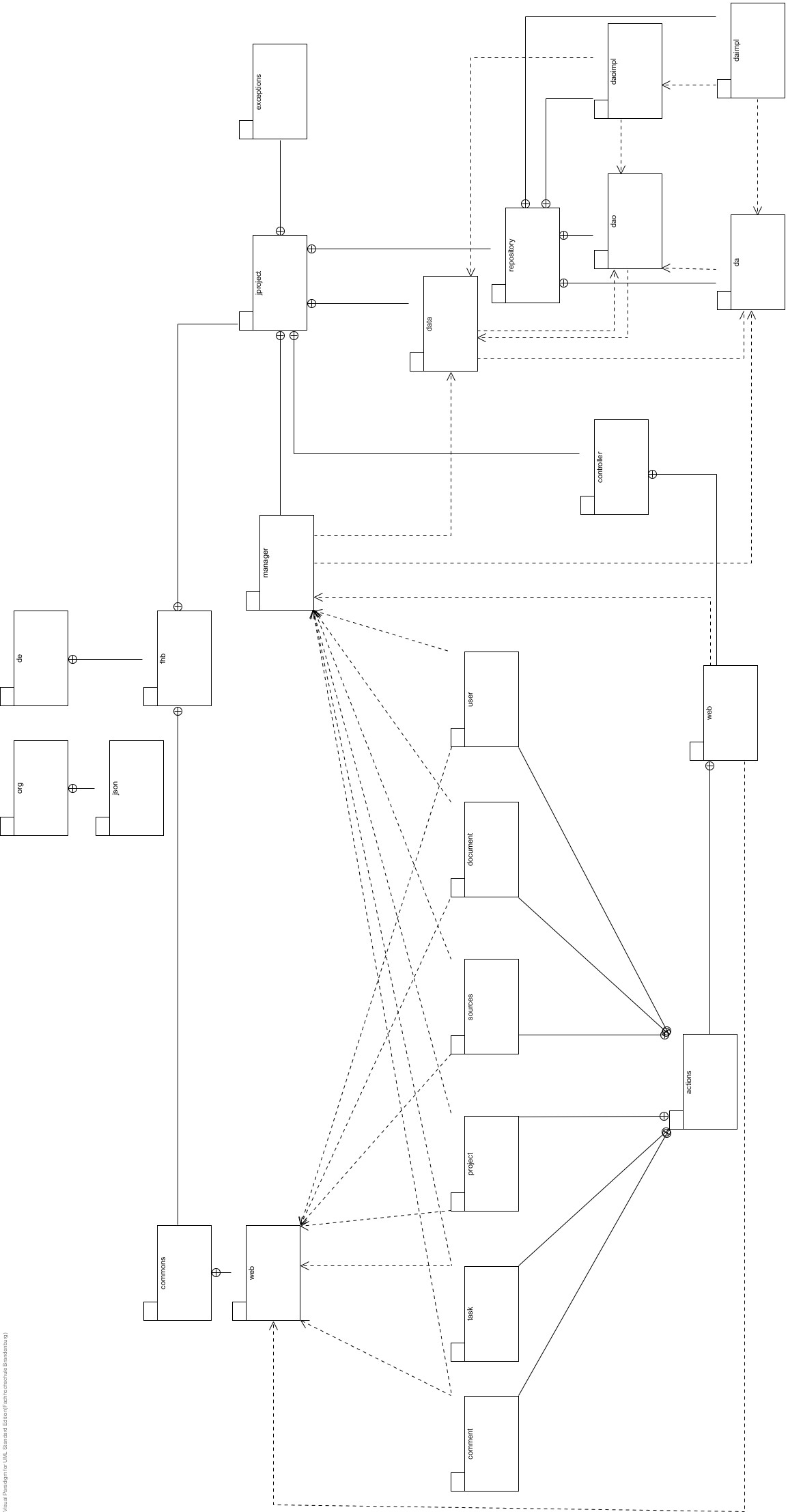
A4 Klassendiagrammübersicht

**A1 Datenbank ERR-Diagramm**

**A2 Sequenz-Diagramm(AddMemberAction)**

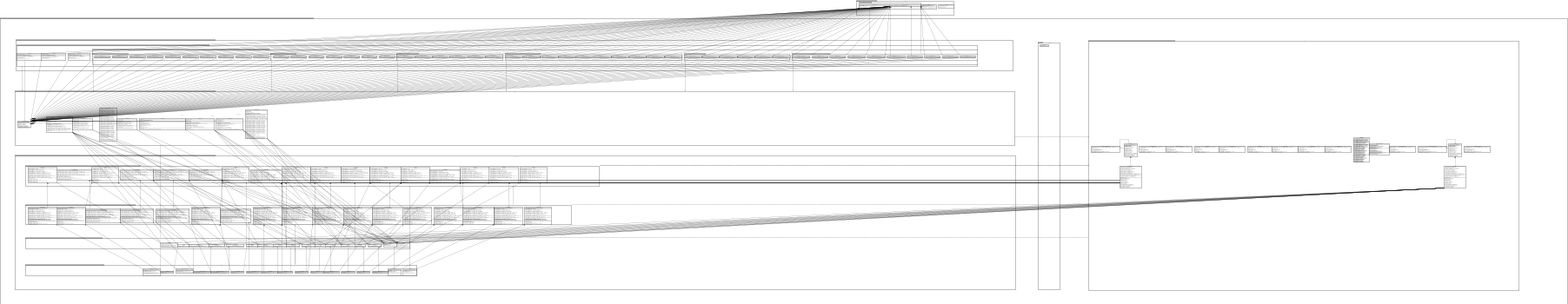


**A3Package-Diagramm**



**A4 Klassen-Diagrammübersicht**

# ***Klassendiagramm***



## Zusammenfassung

|  |  |
| --- | --- |
| Name | Dokumentation |
| Image1.pngcommons | Beinhaltet allgemeine Klassen für immer wiederkehrende Aufgaben |
| Image1.pngweb | Allgemeine Klassen bezogen auf das Web. |
| Image2.png HttpServletControllerBase |  |
| Image2.png HttpRequestActionBase | Aktion die durch Ableitung definiert werden muss.   Die Aktion fuehrt Operationen mit dem Modell aus, und  bereitet die Daten fuer die Ausgabe auf. Als letzte Aktion sollte eine  Aktion zu einer View den Request weiterleiten, wo dann das Ergebnis  der Aktion eingelesen und in eine (HTML-)Seite eingebunden wird. |
| Image2.png CheckString | Diese Klasse ist zum Kontrollieren von Strings. |
| Image2.png HashIt | Diese Klasse generiert einen SHA1- oder MD5-Haswert von einem String. |
| Image1.pngjproject | Projektpackage - beinhaltet alles Projektspezifisches. |
| Image1.pngcontroller | Beinhaltet Controller für jProject. |
| Image1.pngdata | Beinhaltet alle Models/VO´s für jProject. |
| Image1.pngexceptions | Beinhaltet eigene Exceptions für jProject. |
| Image1.pngweb | Beinhaltet alle Controller/Servlets die für das Web nötig sind. |
| Image2.png ProjectException | Programmspezifische Exception zur Allgemeinen Fehlerbehandlung |
| Image1.pngactions | Beinhaltet alle Actions für jProject. |
| Image2.png JProjectServlet | JProjectServlet ist zum Verwalten der GET und POST Requests.  Dieses Servlet is spezialisiert auf alle Anfragen die nicht zu  Dateiübermittlungszwecken oder Adminaktionen dient. |
| Image2.png DataServlet | DataServlet ist zum Verwalten von POST Requests.  Dieses Servlet is spezialisiert auf alle Anfragen die speziell die úbermittlung von Daten/JSONObjekte betreffen. |
| Image2.png AdminServlet | AdminServlet ist zum Verwalten der GET und POST Requests.  Dieses Servlet is spezialisiert auf alle Anfragen die speziell den Admin betreffen. |
| Image1.pnguser | Beinhaltet alle Actions die sich auf den User beziehen. |
| Image1.pngtask | Beinhaltet alle Actions die sich auf Tasks beziehen. |
| Image1.pngsources | Beinhaltet alle Actions die sich auf Sourcecode beziehen. |
| Image1.pngproject | Beinhaltet alle Actions die sich auf Projekte beziehen. |
| Image1.pngdocument | Beinhaltet alle Actions die sich auf Dokumente beziehen. |
| Image1.pngcomment | Beinhaltet alle Actions die sich auf Kommentare beziehen. |
| Image2.png ShowUserInfoAction | Action, die beim Anzeigen der Info eines Users angesprochen wird.   Parameter:  Aktueller User: Session -> aktUser  loginName: request -> loginName    Rechteüberprüfung für GUI:  isAllowedShowUserSettings   Managermethoden:  showUserInfo   Beispiel-Aufruf:  do=showUserInfo&loginName=loginName |
| Image2.png SearchUserAction | Action, die beim Suchen eines Users angesprochen wird.  (UNUSED)   Parameter:  Aktueller User: Session -> aktUser  searchValue: request -> searchValue    Rechteüberprüfung für GUI:  keine   Managermethoden:  searchUser   Beispiel-Aufruf:  do=searchUser&searchValue=loginName |
| Image2.png ShowAllUserAction | Action, die beim Anzeigen aller User angesprochen wird.   Parameter:  Aktueller User: Session -> aktUser    Rechteüberprüfung für GUI:  isAllowedDeleteUserAction   Managermethoden:  showAllUser   Beispiel-Aufruf:  do=showAllUser |
| Image2.png ShowUserSettingsAction | Action, die beim Anzeigen der Settings eines Users angesprochen wird.   Parameter:  Aktueller User: Session -> aktUser  loginName: request -> loginName    Rechteüberprüfung für GUI:  isAllowedUpdateUserSettings   Managermethoden:  showUserSettings   Beispiel-Aufruf:  do=showUserSettings&loginName=loginName |
| Image2.png UpdateUserSettingsAction | Action, die beim √Ñndern von Settings eines Users angesprochen wird.   Parameter:  Aktueller User: Session -> aktUser  loginName: request -> loginName  nachname: request -> nachname  vorname: request -> vorname  neuesPasswortEins: request -> neuesPasswortEins  neuesPasswortZwei: request -> neuesPasswortZwei   Rechteüberprüfung für GUI:  keine   Managermethoden:  updateUserSettings   Beispiel-Aufruf:  do=updateUserSettings&loginName=neuerUser&nachname=Schmidt&vorname=Kurt&neuesPasswortEins=bla&neuesPasswortZwei=bla |
| Image2.png RegisterAction | Action, die beim Hinzufügen eines neuen Users angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  loginName: request -> loginName  passwort: request -> passwort  passwortWdhl: request -> passwortWdhl  nachname: request -> nachname  vorname: request -> vorname    Rechteüberprüfung für GUI:  keine   Managermethoden:  register   Beispiel-Aufruf:  do=Register&loginName=neuerUser&passwort=passwort&passwortWdhl=passwort&nachname=Schmidt&vorname=Kurt |
| Image2.png LoginAction | Action, die beim Login eines Users angesprochen wird.   Parameter:  loginName des Accounts: request ->loginname  password des Accounts: request -> password   Rechteüberprüfung für GUI:  isAllowedOpenAdminconsoleAction    Managermethoden:  login   Beispiel-Aufruf:  do=Login&loginName=Bert&password=berta |
| Image2.png OpenAdminconsole | Action, die beim √ñffnen der AdminConsole angesprochen wird.   Parameter:  Aktueller User: Session -> aktUser   Rechteüberprüfung für GUI:  isAllowedRegister    Managermethoden:  showAllProjects  showAllUser   Beispiel-Aufruf:  do=OpenAdminconsole |
| Image2.png LogoutAction | Action, die beim Logout eines Users angesprochen wird.   Parameter:  keine   Rechteüberprüfung für GUI:  keine    Managermethoden:  logout   Beispiel-Aufruf:  do=Logout |
| Image2.png DeleteUserAction | Action, die beim Löschen eines Users angesprochen wird.   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  loginName(LoginName des Users): request -> loginName    Rechteüberprüfung für GUI:  keine    Managermethoden:  deleteUser   Beispiel-Aufruf:  do=DeleteUser&loginName=Bert |
| Image2.png UpdateTaskAction | Action, die beim Updaten einer Task in einem Projekt angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  taskId(Id des Tasks): request -> taskId  (optional)titel(titel des Tasks): request -> titel  (optional)aufgabenStellung(aufgabenStellung des Tasks): request -> aufgabenStellung  (optional)date(Termin des Tasks): request -> date  (optional)done(Status des Tasks): request -> done   Rechteüberprüfung für GUI:  keine    Managermethoden:  updateTask   Beispiel-Aufruf:  do=UpdateTask&taskId=5&titel=DeineAufgabe&date=2011-06-10&done=true |
| Image2.png ShowAllOwnTasksAction | Action die ausgefuehrt wird, wenn die eigenen Tasks  zu einem Projekt in dem der User teilnimmt angezeigt werden sollen   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  loginName(für den Member): request -> loginName    Rechteüberprüfung für GUI:  isAllowedShowAllOwnTasks   Managermethoden:  deAssignTask   Beispiel-Aufruf:  do=ShowAllOwnTasks |
| Image2.png AssignTaskAction | Action, die beim Zuordnen eines Tasks zu einem Member des Projekt angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  taskId(Id des Tasks): request -> taskId  loginName(für Memberauswahl): request -> loginName    Rechteüberprüfung für GUI:  keine   Managermethoden:  assignTask   Beispiel-Aufruf:  do=AssignTask&loginName=karl&taskId=5 |
| Image2.png DeAssignTaskAction | Action, die beim Abordern eines Member des Projekts von einem Task, angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  taskId(Id des Tasks): request -> taskId  loginName(für Memberauswahl): request -> loginName    Rechteüberprüfung für GUI:  keine   Managermethoden:  deAssignTask   Beispiel-Aufruf:  do=DeAssignTask&loginName=karl&taskId=5 |
| Image2.png AddNewTaskAction | Action, die beim Hinzufügen eines neuen Tasks in einem Projekt angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  titel(Bezeichnung des Tasks): request -> titel  aufgabenStellung(formulierte Aufgabenstellung): request -> aufgabenStellung  date(Datum der Form "yyyy-mm-dd"): request -> date    Rechteüberprüfung für GUI:  keine   Managermethoden:  addNewTask   Beispiel-Aufruf:  do=AddNewTask&titel=TestAufgabe&aufgabenStellung=Tue%20dies%20und%20das!&date=2011-06-02 |
| Image2.png DeleteTaskAction | Action, die beim Löschen eines Tasks angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  taskId(Id des Tasks): request -> taskId    Rechteüberprüfung für GUI:  keine    Managermethoden:  deleteTask   Beispiel-Aufruf:  do=DeleteTask&taskId=1 |
| Image2.png ShowAllTasksAction | Action die angesprochen wird, wenn alle Aufgaben eines Projektes angezeigt werden sollen   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  taskId(Id des Tasks): request -> taskId    Rechteüberprüfung für GUI:  isAllowedUpdateTaskAction,  isAllowedAddNewTaskAction,  isAllowedDeleteTaskAction,  isAllowedShowAllMemberAction,  isAllowedAssignTaskAction,  isAllowedDeAssignTaskAction   Managermethoden:  showAllTasks   Beispiel-Aufruf:  do=ShowAllTasks&taskId=0 |
| Image2.png ShowAllSourceAction | Action, die beim Anzeigen aller Sourcecodes in einem Projekt angesprochen wird,  dabei wird das erste Element detailiert angezeigt falls der Parameter sourceodeId null ist.  Wenn der Parameter nicht null ist wird der entsprechende Sourcecode angezeigt.    Parameter:  Aktueller User: Session ->aktUser  Aktuelles Project: Session -> aktProject  (optional)sourcecodeId: request -> sourcecodeId    Rechteüberprüfung für GUI:  isAllowedDeleteSourceAction    Managermethoden:  showAllSource   Beispiel-Aufruf:  do=ShowAllSourceAction |
| Image2.png UpdateSourceAction | Action, die beim Updaten eines Sourcecodes in einem Projekt angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  sourcecodeId(Id des Sourcecodes): request -> sourcecodeId    Rechteüberprüfung für GUI:  keine    Managermethoden:  updateSource    Beispiel-Aufruf:  do=UpdateSourceAction |
| Image2.png ShowSourceAction | Action, die beim Anzeigen eines Sourcecode in einem Projekt angesprochen wird  (UNUSED)   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  sourcecodeId(Id des Sourcecodes): request -> sourcecodeId    Rechteüberprüfung für GUI:  keine    Managermethoden:  showSource   Beispiel-Aufruf:  do=ShowSourceAction&sourcecodeId=1 |
| Image2.png AddNewSourceAction | Action, die beim Hinzufügen eines neuen Sourcecodes angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  data(List FileItem): request -> data    Rechteüberprüfung für GUI:  keine   Managermethoden:  addNewSource   Beispiel-Aufruf:  do=AddNewSourceAction |
| Image2.png DownloadSourceAction | Action, die beim Download eines Sourcecodes angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  sourcecodeId(Id des Sourcecodes): request -> sourcecodeId    Rechteüberprüfung für GUI:  keine    Managermethoden:  downloadSource   Beispiel-Aufruf:  do=DownloadSourceAction&sourcecodeId=1 |
| Image2.png DeleteSourceAction | Action, die beim Löschen eines Sourcecodes angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  sourcecodeId(Id des Sourcecodes): request -> sourcecodeId    Rechteüberprüfung für GUI:  keine    Managermethoden:  deleteSource   Beispiel-Aufruf:  do=DeleteSourceAction&sourcecodeId=1 |
| Image2.png UpdateMemberAction | Action, die beim Updaten eines Members angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  Loginname(für Memberauswahl): request -> loginName  rolle(Projectrolle): request -> rolle    Rechteüberprüfung für GUI:  keine    Managermethoden:  addMember    Beispiel-Aufruf:  do=UpdateMember&loginName=Michael&rolle=Leader |
| Image2.png ShowAllProjectsAction | Action, die beim Anzeigen aller Projekte angesprochen wird   Parameter:  Aktueller User: Session -> aktUser    Rechteüberprüfung für GUI:  keine    Managermethoden:  showAllProjects   Beispiel-Aufruf:  do=ShowAllProjects |
| Image2.png ShowProjectAction | Action, die das angeklickte Project darstellt.  Es werden allgemeine Informationen zum Project dargestellt und Optionen zur Memberverwaltung bereitgestellt.  Zuvor werden jedoch die Rechte überprüft.   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  Projectname(für Wechsel des aktProject): request -> projectName  Loginname(für Memberauswahl): request -> loginName    Rechteüberprüfung für GUI:  isAllowedAddMemberAction  isAllowedDeleteMemberAction  isAllowedShowAllTasksAction    Managermethoden:  ShowProject,  ShowAllMember,  ShowMember,  showUserInfo   Beispiel-Aufruf:  do=ShowProject&projectName=ProjectName |
| Image2.png ShowAllOwnProjectsAction | Action, die beim Anzeigen aller eigenen Projekte angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject    Rechteüberprüfung für GUI:  keine   Managermethoden:  showAllOwnProjects   Beispiel-Aufruf:  do=ShowAllOwnProjects |
| Image2.png ShowAllMemberAction | Action, die beim Anzeigen aller Member in einem Projekt angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  loginName(loginName des Users): request -> loginName    Rechteüberprüfung für GUI:  keine   Managermethoden:  showMember   Beispiel-Aufruf:  do=ShowAllMember&loginName=Heinz |
| Image2.png DeleteProjectAction | Action, die beim Löschen eines Projektes angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  projectName(Bezeichnung des Projektes): request -> projectName    Rechteüberprüfung für GUI:  keine    Managermethoden:  deleteProject   Beispiel-Aufruf:  do=DeleteProject&projectName=Bezeichnung |
| Image2.png DeleteMemberAction | Action, die beim Löschen eines Members angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  userLoginName(loginname eines Users der Member ist): request -> userLoginName    Rechteüberprüfung für GUI:  keine    Managermethoden:  deleteMember   Beispiel-Aufruf:  do=DeleteMember&userLoginName=karl |
| Image2.png AddNewProjectAction | Action, die beim Hinzufügen eines neuen Projekts angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  projectName(Bezeichnung des Projektes): request -> projectName  status(Projectstatus): request -> status    Rechteüberprüfung für GUI:  keine    Managermethoden:  addNewProject   Beispiel-Aufruf:  do=AddNewProject&projectName=NeuesProjekt&status=beginn |
| Image2.png SearchProjectsAction | Action, die beim Suchen von Projekten angesprochen wird  (UNUSED)   Parameter:  Aktueller User: Session -> aktUser  searchValue(Teil einer Bezeichnung eines gesuchten Projektes): request -> searchValue    Rechteüberprüfung für GUI:  keine    Managermethoden:  searchProjects   Beispiel-Aufruf:  do=SearchProjects&searchValue=a |
| Image2.png AddMemberAction | Action, die beim Hinzufügen eines Members in einem Projekt angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  loginName(loginName eines Users): request -> loginName  rolle(Projectrolle): request -> rolle    Rechteüberprüfung für GUI:  keine    Managermethoden:  addMember    Beispiel-Aufruf:  do=AddMember&loginName=Heinz&rolle=Member |
| Image2.png UpdateDocuAction | Action, die beim Updaten eines Dokuments in einem Projekt angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  documentId(Id des Dokuments): request -> documentId    Rechteüberprüfung für GUI:  keine    Managermethoden:  updateDocu   Beispiel-Aufruf:  do=UpdateDocuAction |
| Image2.png DownloadDocuAction | Action, die beim Download eines Dokuments angesprochen wird    Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  documentId(Id des Documents): request -> documentId    Rechteüberprüfung für GUI:  keine    Managermethoden:  downloadDocu    Beispiel-Aufruf:  do=DownloadDocuAction&documentId=1 |
| Image2.png ShowDocuAction | Action, die beim Anzeigen eines Dokuments in einem Projekt angesprochen wird  (UNUSED)   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  documentId(Id des Dokuments): request -> documentId    Rechteüberprüfung für GUI:  keine    Managermethoden:  showDocuContent    Beispiel-Aufruf:  do=ShowDocuAction&documentId=1 |
| Image2.png ShowAllDocuAction | Action, die beim Anzeigen aller Dokumente in einem Projekt angesprochen wird,  dabei wird das erste Element detailiert angezeigt falls der Parameter documentId null ist.  Andernfalls wird das entsprechende Dokument angezeigt.   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  (optional)documentId: request -> documentId    Rechteüberprüfung für GUI:  isAllowedDeleteDocuAction    Managermethoden:  showAllDocu   Beispiel-Aufruf:  do=ShowAllDocuAction&documentId=0 |
| Image2.png DeleteDocuAction | Action, die beim Löschen eines Dokuments angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  documentId(Id des Documents): request -> documentId    Rechteüberprüfung für GUI:  keine    Managermethoden:  deleteDocu    Beispiel-Aufruf:  do=DeleteDocuAction&documentId=1 |
| Image2.png AddNewDocuAction | Action, die beim Hinzufügen eines neuen Dokuments angesprochen wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  data(List FileItem): request -> data    Rechteüberprüfung für GUI:  keine   Managermethoden:  addNewDocu   Beispiel-Aufruf:  do=AddNewDocuAction |
| Image2.png ShowAllComments41TaskAction | Action die angesprochen wird, wenn zu einem Task alle Kommentare angezeigt werden   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  taskId(Id des Tasks): request -> taskId    Rechteüberprüfung für GUI:  keine    Managermethoden:  ShowAllComments41Task   Beispiel-Aufruf:  do=ShowAllComments41Task&taskId=1 |
| Image2.png UpdateCommentAction | Action die angesprochen wird, wenn ein Kommentar geupdatet werden soll   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  commentId(Id des Comments): request -> commentId  entry(Neuer Inhalt des Comments): request -> entry    Rechteüberprüfung für GUI:  keine    Managermethoden:  updateComment    Beispiel-Aufruf:  do=UpdateComment&commentId=1&entry=aenderungInhalt |
| Image2.png CommentDocuAction | Action die angesprochen wird wenn ein Dokument kommentiert wird   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  documentId(Id des Dokuments): request -> documentId entry(Inhalt des Comments): request -> entry    Rechteüberprüfung für GUI:  keine     Managermethoden:  commentDocu    Beispiel-Aufruf: do=CommentDocu&documentId=1&entry=GutesDokument |
| Image2.png ShowAllComments41ProjectAction | Action die angesprochen wenn zu einem Projekt alle Kommentare angezeigt werden   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject    Rechteüberprüfung für GUI:  keine    Managermethoden:  showAllComments41Project   Beispiel-Aufruf:  do=ShowAllComments41Project |
| Image2.png ShowAllComments41SourceAction | Action die angesprochen wenn zu einem Sourcecode alle Kommentare angezeigt werden   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  sourcecodeId(Id des Sourcecodes): request -> sourcecodeId    Rechteüberprüfung für GUI:  keine    Managermethoden:  showAllComments41Source   Beispiel-Aufruf:  do=ShowAllComments41Source&sourcecodeId=1 |
| Image2.png ShowAllComments41DocuAction | Action die angesprochen wird, wenn zu einem Dokument alle Kommentare angezeigt werden   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  documentId(Id des Dokuments): request -> documentId    Rechteüberprüfung für GUI:  keine    Managermethoden:  showAllComments41Docu   Beispiel-Aufruf:  do=ShowAllComments41Docu&documentId=1 |
| Image2.png DeleteCommentAction | Action die angesprochen wird wenn ein Kommentar geloescht werden soll   Parameter:  Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  commentId(Id des Comments): request -> commentId    Rechteüberprüfung für GUI:  keine    Managermethoden:  deleteComment   Beispiel-Aufruf:  do=DeleteComment&commentId=1 |
| Image2.png CommentTaskAction | Action die angesprochen wird wenn eine Aufgabe kommentiert wird   Parameter: Aktueller User: Session -> aktUser  Aktuelles Project: Session -> aktProject  taskId(Id des Tasks): request -> taskId  entry(Inhalt des Comments): request -> entry    Rechteüberprüfung für GUI:  keine    Managermethoden:  commentTask    Beispiel-Aufruf:  do=CommentSource&taskId=1&entry=DieseAufgabeIstDringend |
| Image2.png CommentSourceAction | Action die angesprochen wird wenn ein Sourcecode kommentiert wird   Parameter:  Aktueller User: Session -> aktUser Aktuelles Project: Session -> aktProject sourcecodeId(Id des Sourcecodes): request -> sourcecodeId entry(Inhalt des Comments): request -> entry   Rechteüberprüfung für GUI: keine    Managermethoden: commentSource   Beispiel-Aufruf: do=CommentSource&sourcecodeId=1&entry=GuterSource |
| Image2.png CommentProjectAction | Action die angesprochen wird wenn in einem Project kommentiert wird   Parameter:  Aktueller User: Session -> aktUser Aktuelles Project: Session -> aktProject projectName(Id des projects): request -> projectName entry(Inhalt des Comments): request -> entry   Rechteüberprüfung für GUI: keine    Managermethoden: commentProject   Beispiel-Aufruf: do=CommentProject&projectName=ProjectName&entry=KickOffStarted |
| Image1.pngmanager | Beinhaltet alle Manager die für jProject benötigt werden. |
| Image2.png GlobalRolesManager | Dieser Manager kontrolliert den Zugriff auf die Globalen Rechte |
| Image2.png ProjectRolesManager | Dieser Manager kontrolliert den Zugriff auf die Projekt-Rechte |
| Image2.png CommentManager | Dieser Manager kontrolliert den Zugriff auf die Kommentare |
| Image2.png DocumentManager | diese Klasse kontrolliert den Zugriff auf die Documente |
| Image2.png SourceManager | diese Klasse kontrolliert den Zugriff auf den SourceCode |
| Image2.png UserManager | Dieser Manager kontrolliert den Zugriff auf die User |
| Image2.png TaskManager | Dieser Manager kontrolliert den Zugriff auf die Tasks |
| Image2.png ProjectManager | Dieser Manager kontrolliert den Zugriff auf die Projekte |
| Image2.png MainManager | Hauptmanager fuer die Aktionen (Haupt BO-Klasse),  besitzt die Untermanager(BOs) zu den einzelnen Teilbereichen  an Abfragen und Aktionen des systems |
| Image1.pngrepository | Beinhatet Zugriffsschichten zur Datenbank. |
| Image1.pngdao | Beinhaltet alle Interfaces und Factory für die DAO-Schicht(Datenbankzugriff).  Dieses Package bietet die Funktionalität für Standardzugriffe auf die Datenbank. |
| Image1.pngdaoimpl | Beinhaltet alle Implementationen und Factory für die DAO-Schicht(Datenbankzugriff).  Dieses Package bietet die Funktionalität für Standardzugriffe auf die Datenbank. |
| Image1.pngda | Beinhaltet alle Interfaces und Factory für die DA-Schicht(Datenbankzugriff).  Dieses Package bietet die Funktionalität für erweiterte Zugriffe auf die Datenbank. |
| Image1.pngdaimpl | Beinhaltet alle Implementationen und Factory für die DA-Schicht(Datenbankzugriff).  Dieses Package bietet die Funktionalität für erweiterte Zugriffe auf die Datenbank. |