

**Gesamtentwurf**

# **Dokumentation**

Matthias Englert, Fabian Schilha,  
Andreas Rottach, Marius Kircher, Julius Friedrich, Lorenz Weiß

WS 2014/15 und SS 2014/15

# Inhaltsverzeichnis

<b>1 Datenbankentwurf</b>	<b>3</b>
1.1 Datenbankdiagramm . . . . .	3
1.2 Beschreibung der Tabellen in der Datenbank . . . . .	4
1.3 Datenbanktabellen . . . . .	4
<b>2 System-Architektur</b>	<b>11</b>
2.1 Klassendiagramm . . . . .	11
2.2 Kommunikationsdiagramme . . . . .	13
2.2.1 BenutzerServlet Diagramme . . . . .	13
2.2.2 AdminServlet Diagramme . . . . .	15
2.2.3 KarteikartenServlet Diagramme . . . . .	16
2.2.4 KommentarServlet Diagramme . . . . .	19
2.2.5 ProfilServlet Diagramme . . . . .	21
2.2.6 NotizenServlet Diagramme . . . . .	23
2.2.7 VeranstaltungServlet Diagramme . . . . .	24
2.3 Methodenbeschreibung . . . . .	29
2.3.1 BenutzerServlet Methoden . . . . .	29
2.3.2 ProfilServlet Methoden . . . . .	30
2.3.3 VeranstaltungsServlet Methoden . . . . .	31
2.3.4 KarteikartenServlet Methoden . . . . .	32
2.3.5 KommentarServlet Methoden . . . . .	32
2.3.6 NotizenServlet Methoden . . . . .	33
2.3.7 AdminServlet Methoden . . . . .	34
2.3.8 DatenbankManager Methoden . . . . .	34
<b>3 Entwicklungsvorgaben</b>	<b>38</b>
3.1 Rollenverteilung . . . . .	38
3.2 Projektplan . . . . .	38
3.3 Programmierkonventionen . . . . .	39
3.3.1 Allgemeine Konventionen . . . . .	39
3.3.2 Java-Konventionen . . . . .	40
<b>4 Benutzerschnittstelle</b>	<b>41</b>
4.1 Startseite . . . . .	41
4.2 Hauptseite . . . . .	42
4.3 Veranstaltungsseite . . . . .	45

# 1 Datenbankentwurf

## 1.1 Datenbankdiagramm

Das folgende Diagramm ist ein ER-Diagramm in min-max Notation.

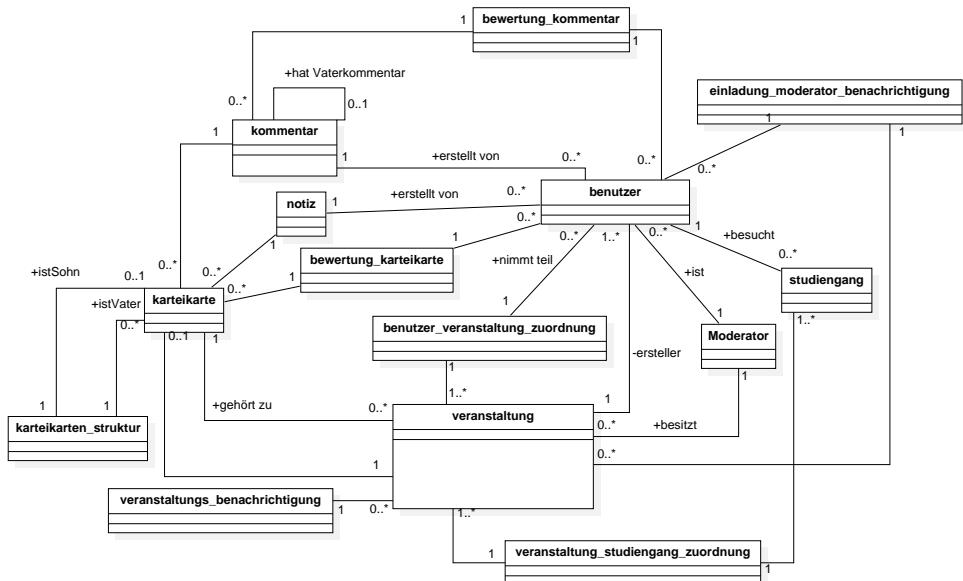


Abbildung 1.1: ER-Diagramm

## 1.2 Beschreibung der Tabellen in der Datenbank

### 1.3 Datenbanktabellen

TABELLE	benutzer
BESCHREIBUNG	Speichert alle Informationen, die zu einem Benutzer gehören.
SCHLÜSSEL	eMail : String
FELD	Vorname : String
FELD	Nachname : String
FELD	Martrikelnummer : Integer
FELD	Studiengang : String
BESCHREIBUNG	Fremdschlüssel, Benutzer ist bestimmtem Studiengang zugeordnet. Dozenten sind hier keinem Studiengang zugeordnet. Referenziert auf Tabelle Studiengang
FELD	Kennwort : String
BESCHREIBUNG	Speichert das verschlüsselte Passwort des Nutzers
FELD	Nutzerstatus : Enum
BESCHREIBUNG	Speichert ob Benutzer ein Student oder Dozent ist.
FELD	NotifyDiskussionen : Enum
BESCHREIBUNG	Speichert wie ein Benutzer über Diskussionen informiert wird.

TABELLE	studiengang
BESCHREIBUNG	alle Studiengänge an der Universität Ulm
SCHLÜSSEL	Name : String

TABELLE	veranstaltung
BESCHREIBUNG	Daten für alle Veranstaltungen (hauptsächlich Vorlesungen)
SCHLÜSSEL	Titel : String
FELD	Beschreibung : String
FELD	Semester : String
BESCHREIBUNG	Eine Veranstaltung gilt nur ein Semester lang
FELD	Kennwort : String
FELD	KommentareErlaubt : Boolean
FELD	BewertungenErlaubt : Boolean
FELD	ModeratorKarteikartenBearbeiten : Boolean
BESCHREIBUNG	Die Moderatoren haben die Erlaubnis, Karteikarten zu bearbeiten.
FELD	Ersteller : String
BESCHREIBUNG	Fremdschlüssel in die Tabelle benutzer. Eine Veranstaltung hat genau einen Ersteller. Ein Benutzer kann mehrere Veranstaltungen erstellen.
FELD	ErsteKarteikarte : Integer
BESCHREIBUNG	Fremdschlüssel in die Tabelle karteikarte. Eine Veranstaltung hat genau eine Startkarteikarte.

TABELLE	moderator
BESCHREIBUNG	Speichert für jede Veranstaltung die Moderatoren
SCHLÜSSEL	ID : Integer
FELD	Benutzer : String
BESCHREIBUNG	Fremdschlüssel in die Tabelle benutzer. Ein benutzer kann Moderator in mehreren Veranstaltungen sein. Ein Moderator einer Veranstaltung ist immer genau einem Benutzer zugeordnet
FELD	Veranstaltung : String
BESCHREIBUNG	Fremdschlüssel in die Tabelle veranstaltung. Eine Veranstaltung kann beliebig viele Moderatoren haben. Ein Moderator gehört immer genau zu einer Veranstaltung

TABELLE	benutzer_veranstaltung_zuordnung
BESCHREIBUNG	Zuordnungstabelle zwischen benutzer und veranstaltung. Ein Benutzer kann mehrere Veranstaltungen besuchen und eine Veranstaltung wird von mehreren Benutzern besucht
SCHLÜSSEL	ID : Integer
FELD	BenutzerMail : String
BESCHREIBUNG	Fremdschlüssel in die Tabelle benutzer
FELD	VeranstaltungsName : String
BESCHREIBUNG	Fremdschlüssel in die Tabelle veranstaltung
TABELLE	veranstaltung_studiengang_zuordnung
BESCHREIBUNG	Zuordnungstabelle zwischen veranstaltung und studiengang. Eine Veranstaltung kann von Studenten unterschiedlicher Studiengänge besucht werden. Ein Studiengang kann zu mehreren Veranstaltungen gehören
SCHLÜSSEL	ID : Integer
FELD	VeranstaltungsName : String
BESCHREIBUNG	Fremdschlüssel in die Tabelle veranstaltung
FELD	Studiengang : String
BESCHREIBUNG	Fremdschlüssel in die Tabelle studiengang
TABELLE	karteikarte
BESCHREIBUNG	Speichert Eigenschaften einer Karteikarte
SCHLÜSSEL	ID : Integer
FELD	Titel : String
FELD	Typ : Enum
BESCHREIBUNG	Eine Karteikarte kann entweder Text, ein Bild oder ein Video enthalten
FELD	Bewertung : Integer
BESCHREIBUNG	Speichert Gesamtbewertung einer Karteikarte
FELD	Aenderungsdatum : Date
FELD	VeranstaltungsName : String
BESCHREIBUNG	Fremdschlüssel in die Tabelle Veranstaltung. Eine Karteikarte gehört genau zu einer Veranstaltung

TABELLE	karteikarten_struktur
BESCHREIBUNG	Speichert die Struktur der Karteikarten. Setzt zwei Karteikarten in eine Vater-Kind Beziehung. Um eine Reihenfolge zwischen den Kindern einer Karteikarte herzustellen, wird für jedes Kind eine Nummer gespeichert
SCHLÜSSEL	ID : Integer
FELD	Position
BESCHREIBUNG	Legt Reihenfolge der Kinder fest
FELD	VaterKarteikarte : Integer
BESCHREIBUNG	Fremdschlüssel in die Tabelle karteikarte. Eine Karteikarte kann öfters Vater sein.
FELD	SohnKarteikarte : Integer
BESCHREIBUNG	Fremdschlüssel in die Tabelle karteikarte. Eine Karteikarte kann Sohn von maximal einer anderen Karteikarte sein

TABELLE	bewertung_karteikarte
BESCHREIBUNG	Zuordnungstabelle zwischen karteikarte und benutzer. Eine Karteikarte kann von mehreren Benutzern bewertet werden. Ein Benutzer kann mehrere karteikarten bewerten
SCHLÜSSEL	ID : Integer
FELD	Bewertung : Integer
BESCHREIBUNG	Bewertung, die ein bestimmter Benutzer dieser Karteikarte gegeben hat. Die Bewertung kann -1 oder 1 sein.
FELD	BenutzerMail : String
BESCHREIBUNG	Fremdschlüssel in die Tabelle benutzer
FELD	KarteikarteID : Integer
BESCHREIBUNG	Fremdschlüssel in die Tabelle karteikarte

TABELLE	kommentar
BESCHREIBUNG	Speichert Attribute für einen Kommentar. Zu einer Karteikarte kann es mehrere Kommentare geben. Ein Kommentar gehört zu genau einer Karteikarte
SCHLÜSSEL	ID : Integer
FELD	Inhalt : String
FELD	Erstelldatum : Date
FELD	Bewertung : Integer
BESCHREIBUNG	Speichert Gesamtbewertung eines Kommentars
FELD	BenutzerMail : String
BESCHREIBUNG	Fremdschlüssel für den Ersteller des Kommentars. Ein Kommentar hat immer genau einen Ersteller
FELD	KarteikarteID : Integer
BESCHREIBUNG	Fremdschlüssel in die Tabelle karteikarte
FELD	VaterkommentarID : Integer
BESCHREIBUNG	Um Kommentare zu Kommentaren repräsentieren zu können, kann ein Kommentar auf einen darüberliegenden verweisen. Wird ein Vaterkommentar gelöscht, so werden alle Kindkommentare mitgelöscht
FELD	UnterhaltungBeendet : Boolean
BESCHREIBUNG	True, wenn der Ersteller/Moderator/Dozent die Unterhaltung beendet hat.

TABELLE	bewertung_kommentar
BESCHREIBUNG	Zuordnungstabelle zwischen kommentar und benutzer. Ein Kommentar kann von mehreren Benutzern bewertet werden. Ein Benutzer kann mehrere kommentare bewerten
SCHLÜSSEL	ID : Integer
FELD	Bewertung : Integer
BESCHREIBUNG	Bewertung, die ein bestimmter Benutzer diesem Kommentar gegeben hat. Die Bewertung kann -1 oder 1 sein.
FELD	Benutzer : String
BESCHREIBUNG	Fremdschlüssel in die Tabelle benutzer
FELD	KommentarID : Integer
BESCHREIBUNG	Fremdschlüssel in die Tabelle kommentar

TABELLE	notiz
BESCHREIBUNG	Notiz, die ein Benutzer für sich selbst zu einer Karteikarte angelegt hat. Eine Notiz gehört genau zu einem Benutzer und eine Notiz gehört genau zu einer Karteikarte.
SCHLÜSSEL	ID : Integer
FELD	Inhalt : String
FELD	BenutzerMail : String
BESCHREIBUNG	Fremdschlüssel für den Ersteller der Notiz
FELD	KarteikarteID : Integer
BESCHREIBUNG	Fremdschlüssel in die Tabelle karteikarte

TABELLE	benachrichtigung
BESCHREIBUNG	Speichert Benachrichtigungen an Benutzer, welche Änderungen an einer Veranstaltungen/Karteikarten/Kommentaren usw. beinhalten.
SCHLÜSSEL	ID : Integer
FELD	Inhalt : String
FELD	Erstelldatum : Date
FELD	Verweis : Int
BESCHREIBUNG	Enthält die ID des Verweis Ziels. Der Zieltyp wird mit Hilfe des Felds „VerweisTyp festgelegt.“
FELD	VerweisTyp: enum
BESCHREIBUNG	Legt fest, in welcher Tabelle sich das VerweisZiel befindet. (Karteikarten, Veranstaltung, ...).

# 2 System-Architektur

## 2.1 Klassendiagramm

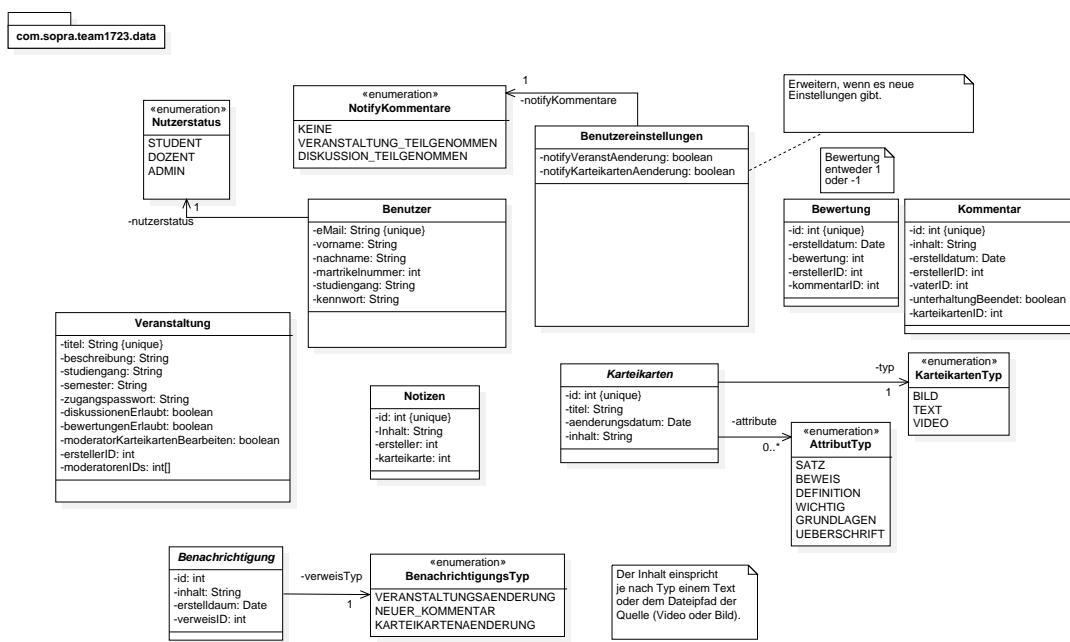


Abbildung 2.1: Klassendiagramm, das die Daten-Klassen enthält

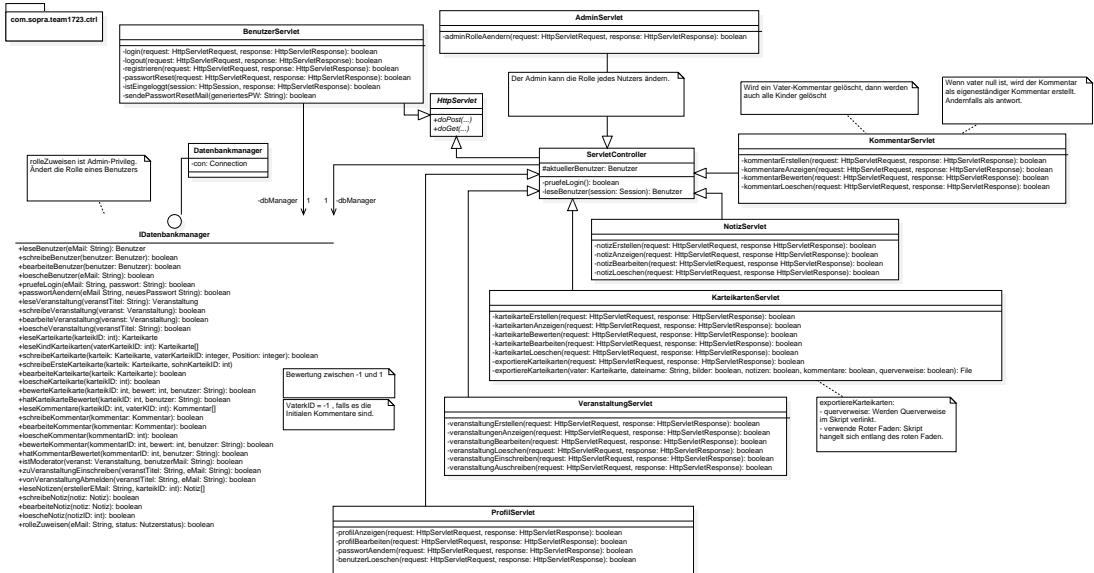


Abbildung 2.2: Klassendiagramm der Systemklassen. Also alle Klassen, die die Steuerung der Website übernehmen

## 2.2 Kommunikationsdiagramme

### 2.2.1 BenutzerServlet Diagramme

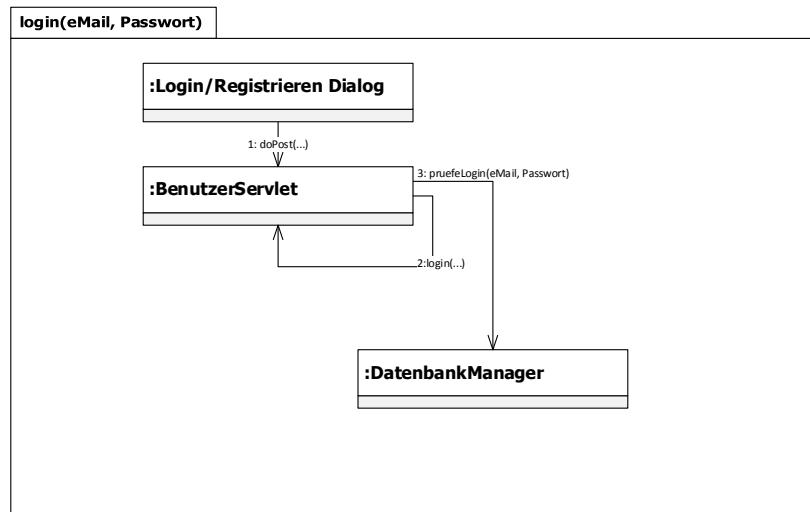


Abbildung 2.3: Login

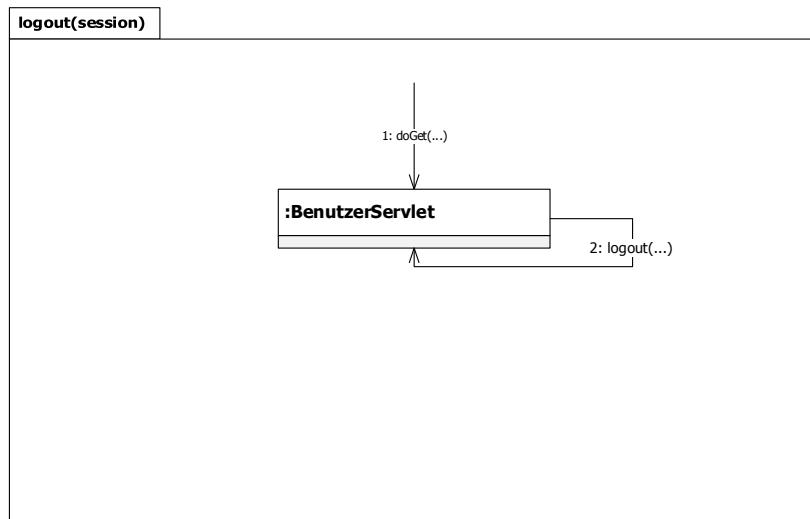


Abbildung 2.4: Logout

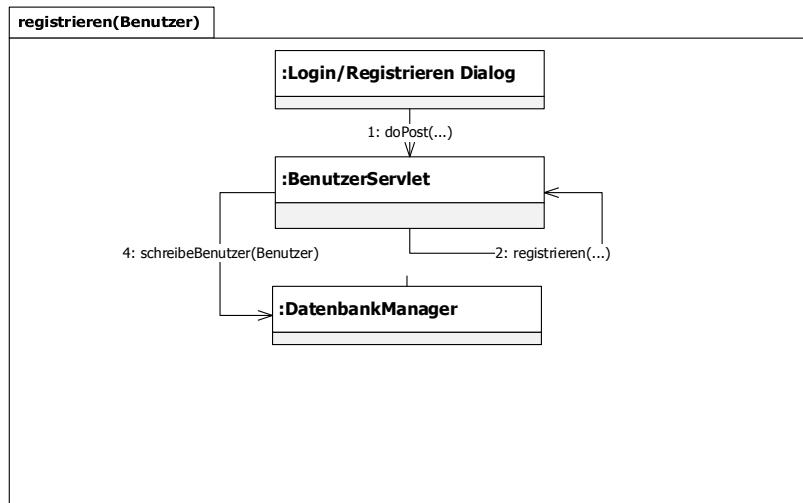


Abbildung 2.5: Registrierung

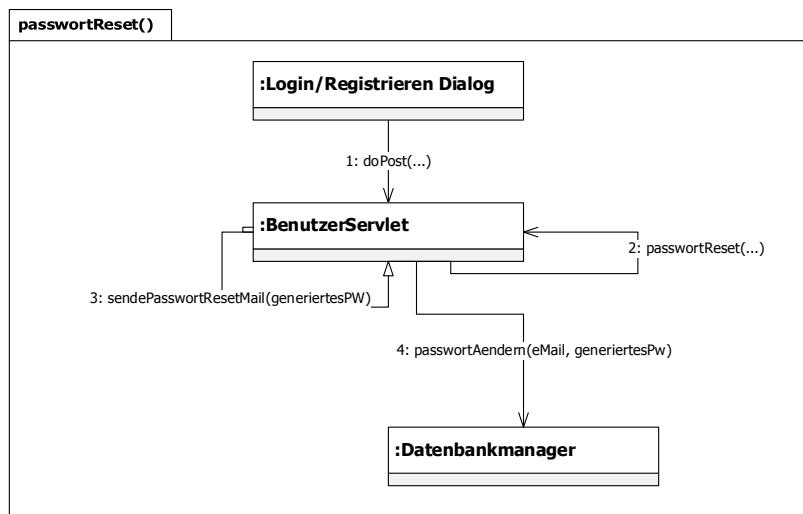


Abbildung 2.6: Passwort zurücksetzen

## 2.2.2 AdminServlet Diagramme

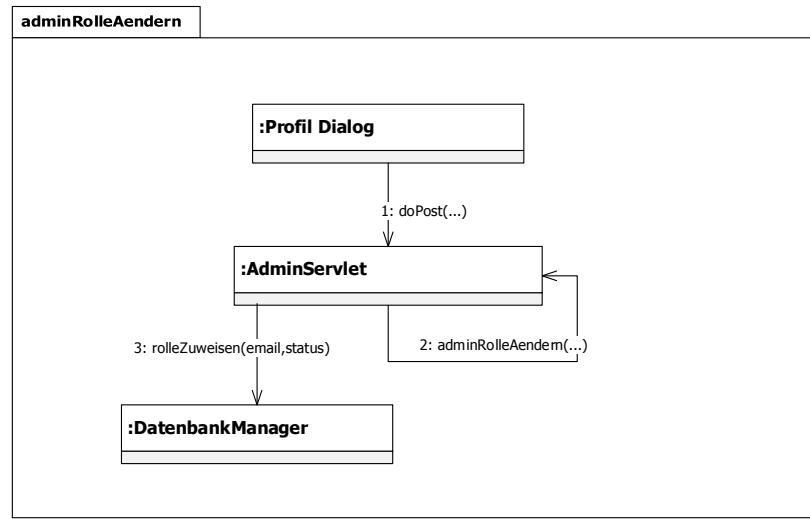


Abbildung 2.7: RolleAendern

### 2.2.3 KarteikartenServlet Diagramme

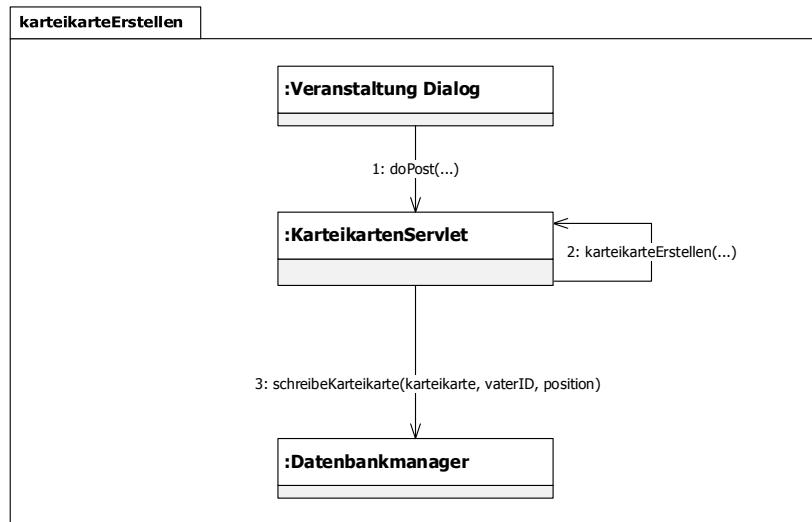


Abbildung 2.8: Karteikarte erstellen

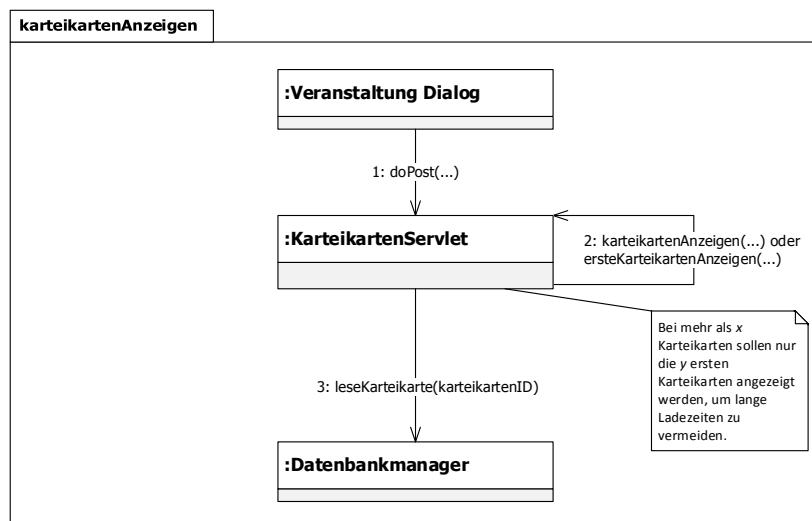


Abbildung 2.9: Karteikarte anzeigen

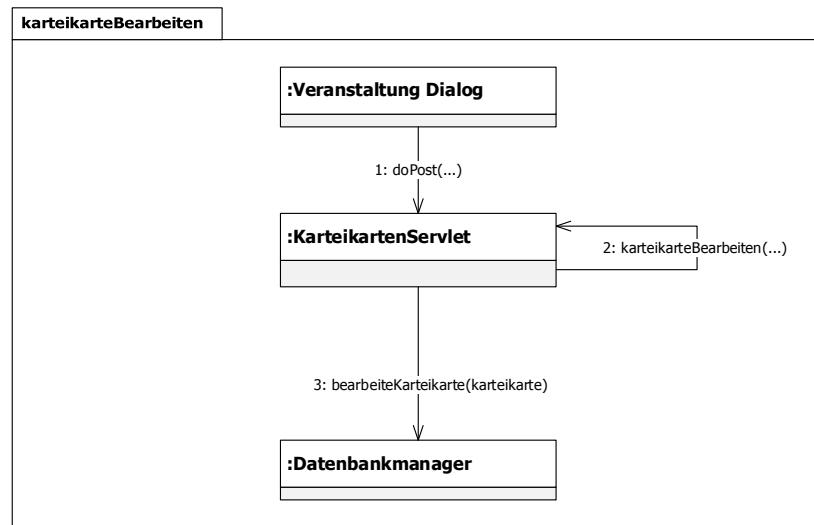


Abbildung 2.10: Karteikarte bearbeiten

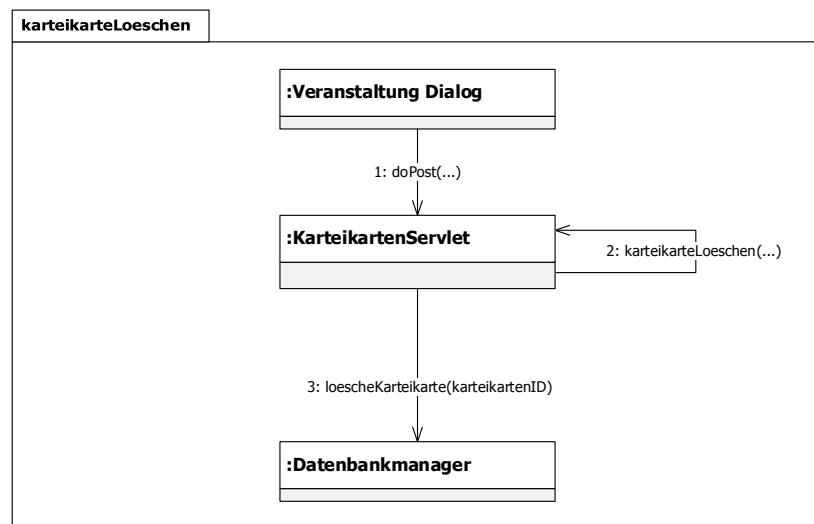


Abbildung 2.11: Karteikarte löschen

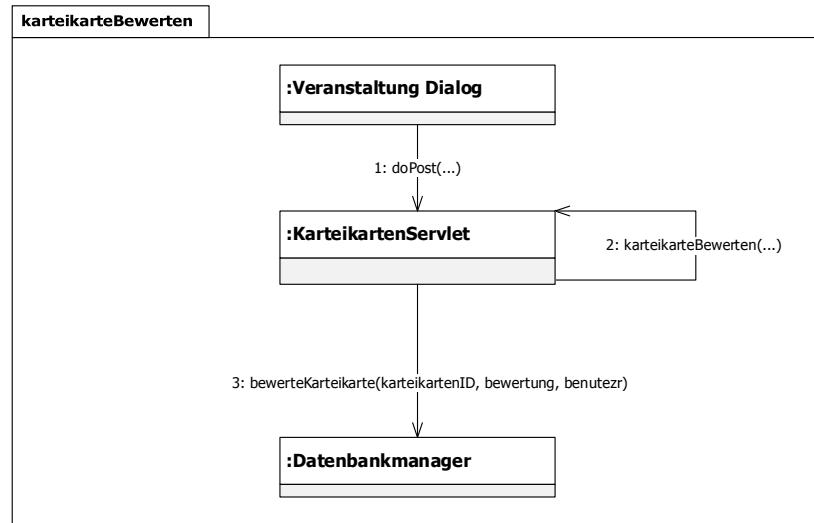


Abbildung 2.12: Karteikarte bewerten

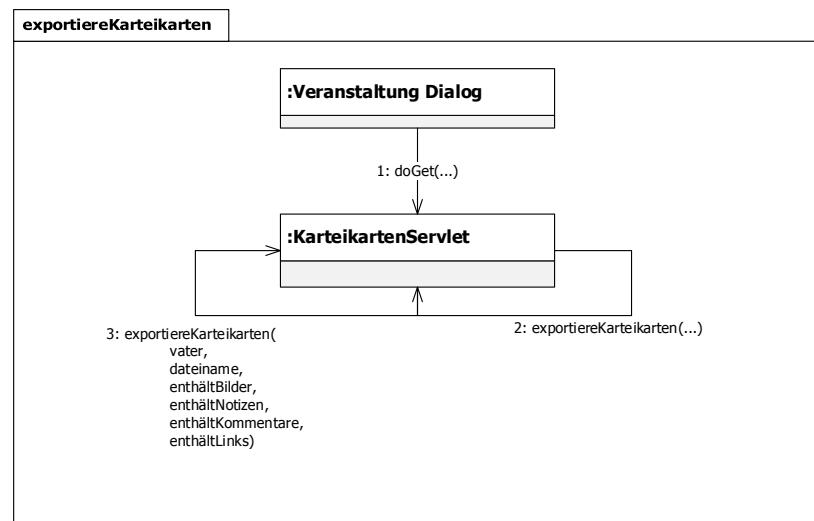


Abbildung 2.13: Export von Karteikarten

## 2.2.4 KommentarServlet Diagramme

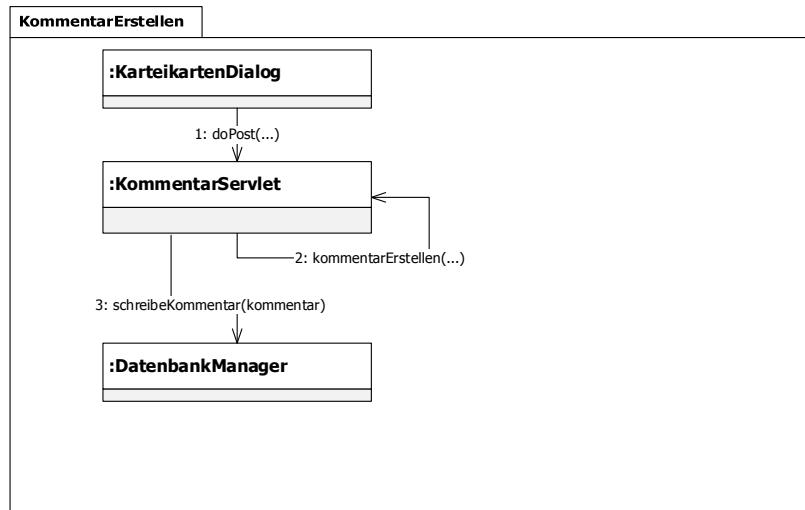


Abbildung 2.14: Kommentar erstellen

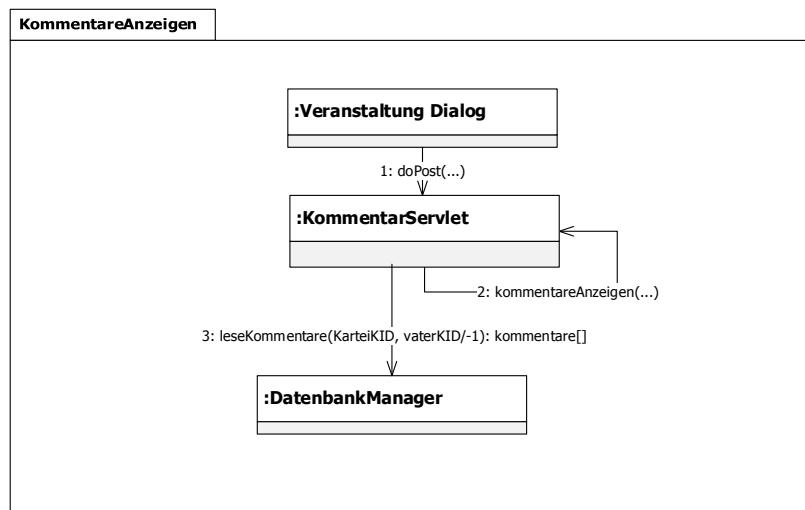


Abbildung 2.15: Kommentar anzeigen

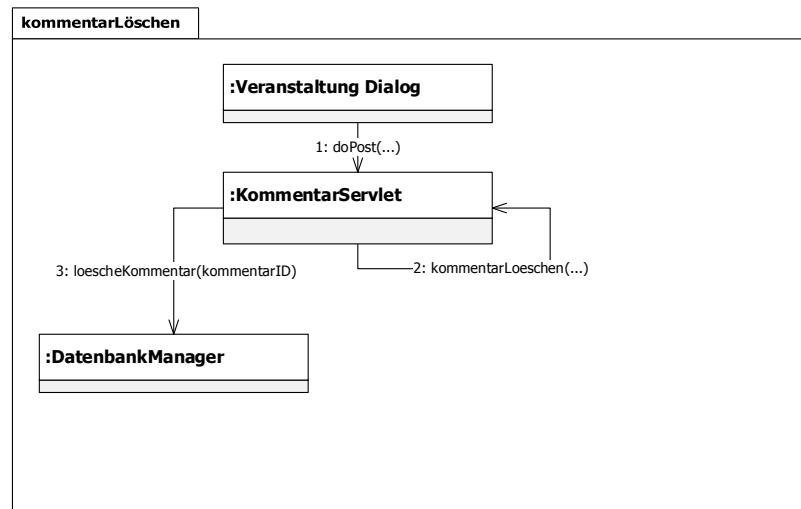


Abbildung 2.16: Kommentar löschen

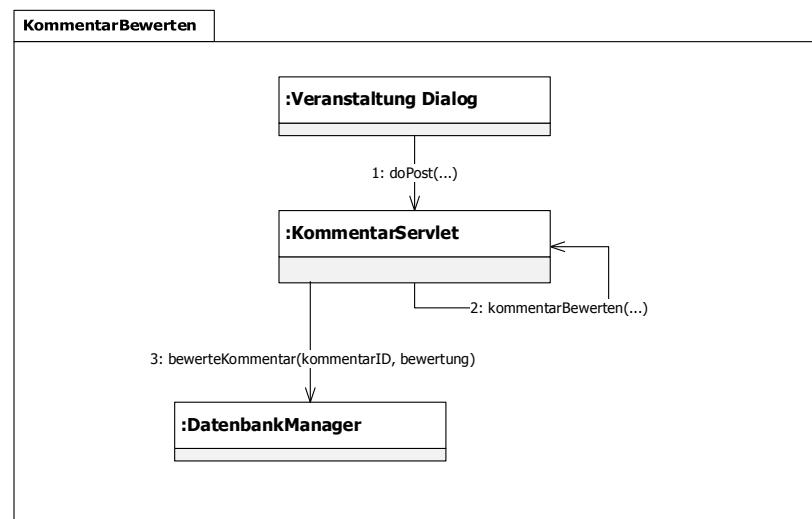


Abbildung 2.17: Kommentar bewerten

## 2.2.5 ProfilServlet Diagramme

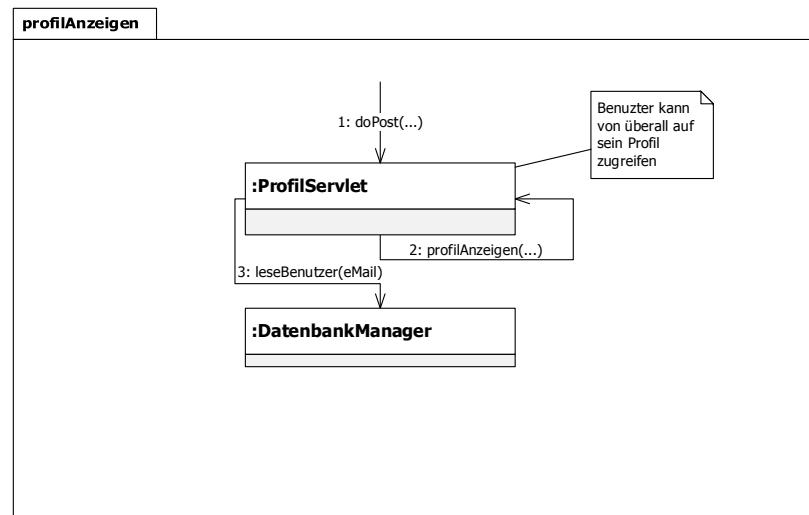


Abbildung 2.18: Profil anzeigen

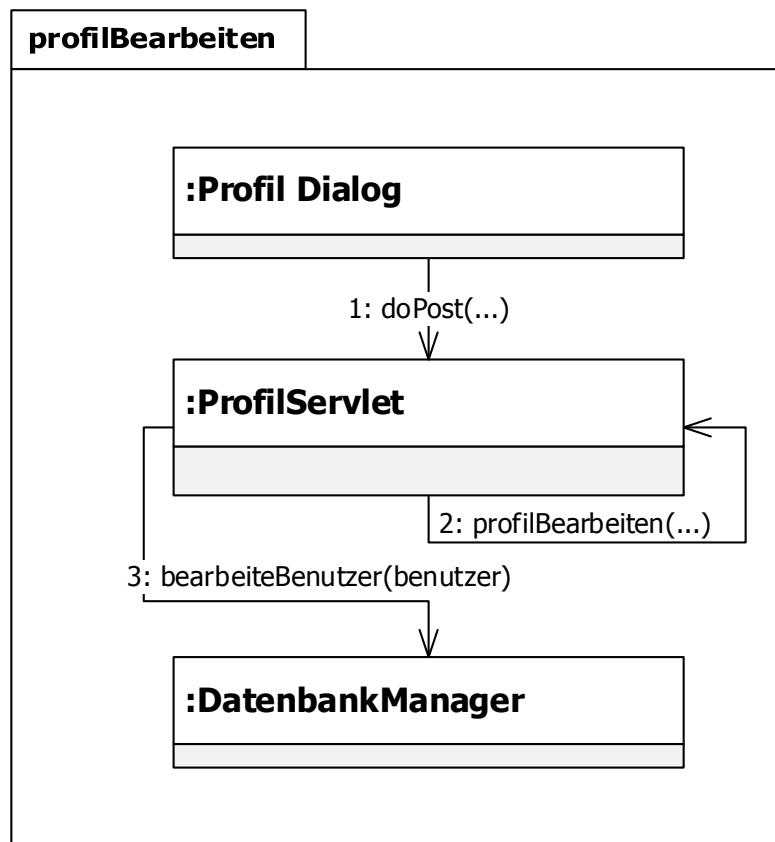


Abbildung 2.19: Profil bearbeiten

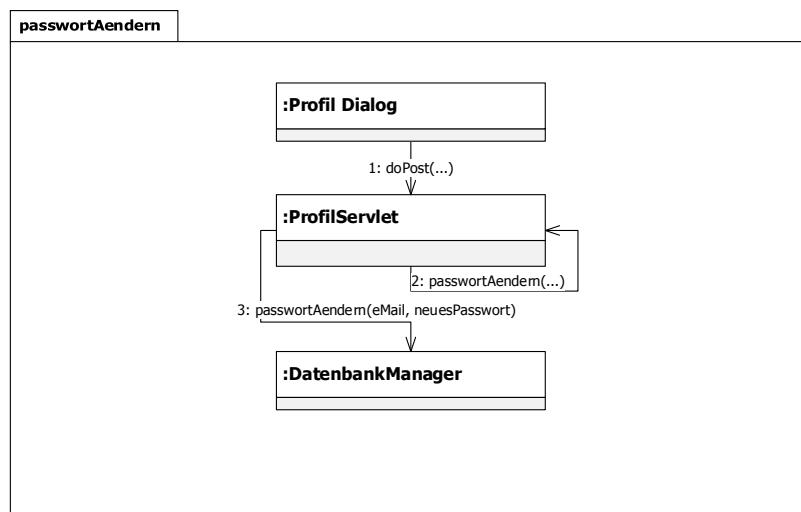


Abbildung 2.20: Passwort ändern

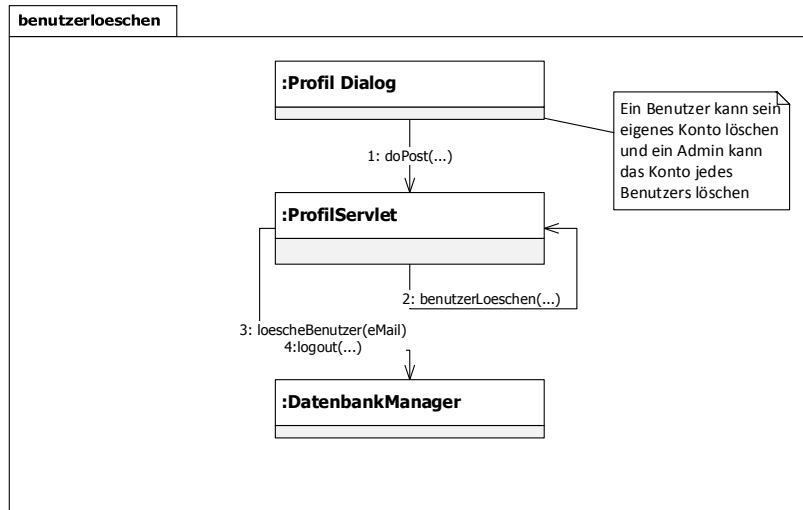


Abbildung 2.21: Benutzer löschen

## 2.2.6 NotizenServlet Diagramme

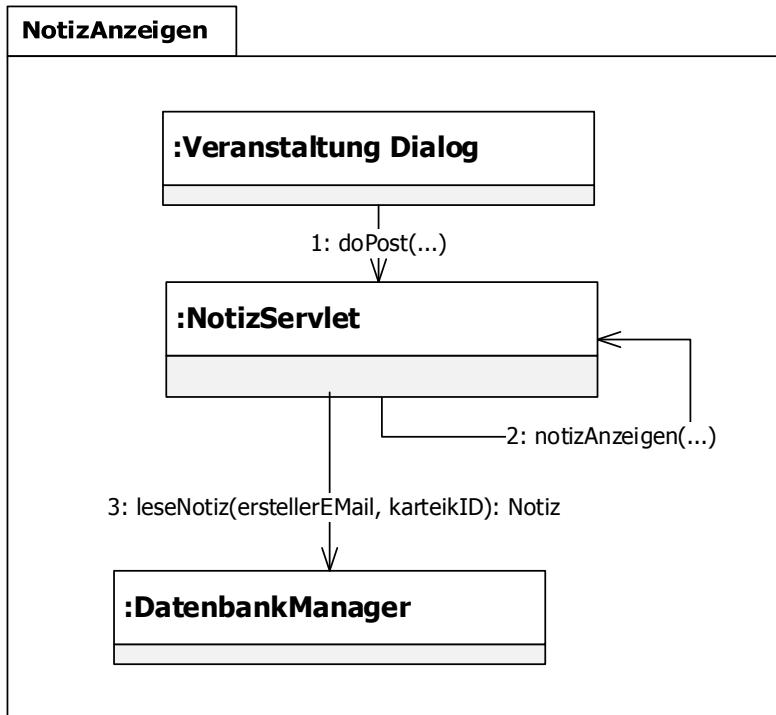


Abbildung 2.22: Notiz anzeigen

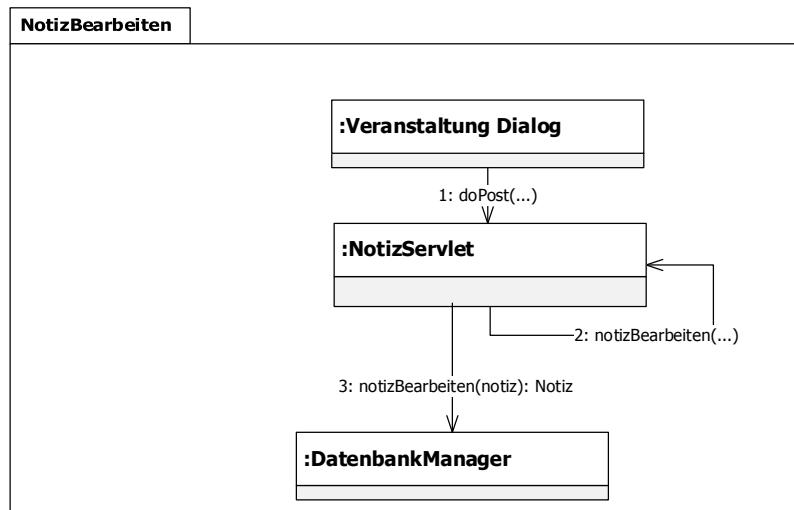


Abbildung 2.23: Notiz bearbeiten

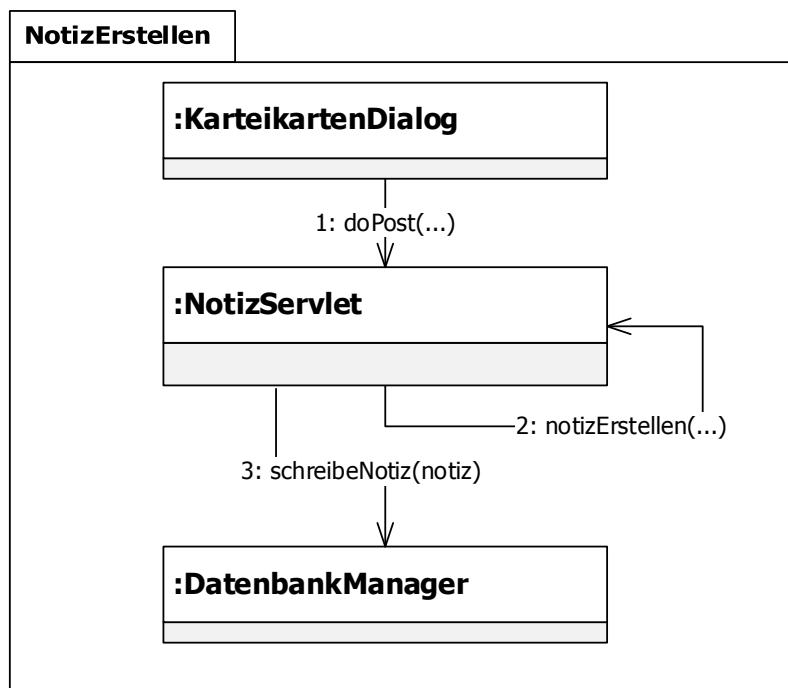


Abbildung 2.24: Notiz erstellen

## 2.2.7 VeranstaltungServlet Diagramme

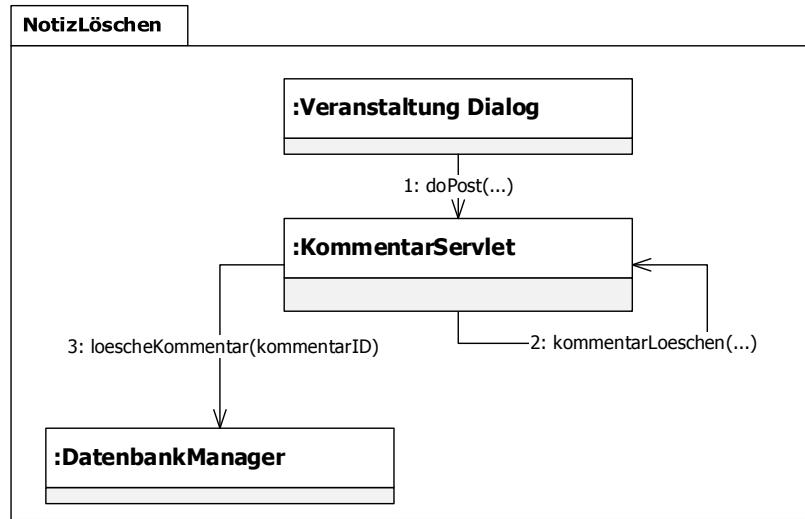


Abbildung 2.25: Notiz löschen

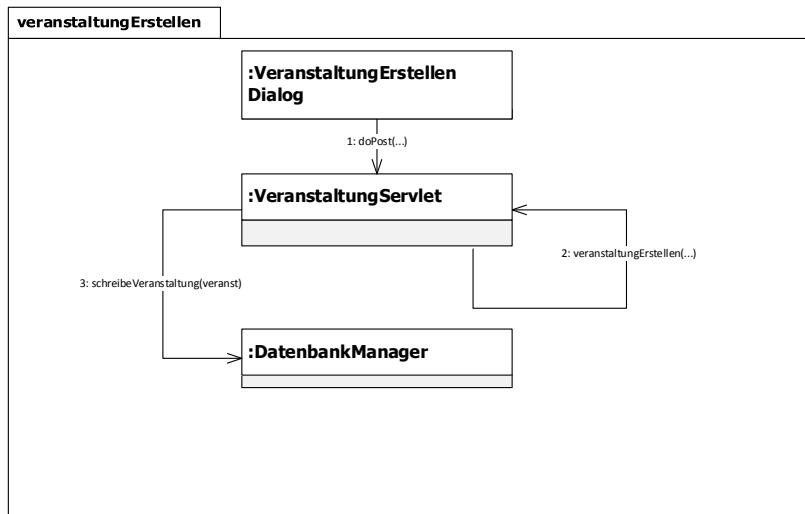


Abbildung 2.26: Veranstaltung erstellen

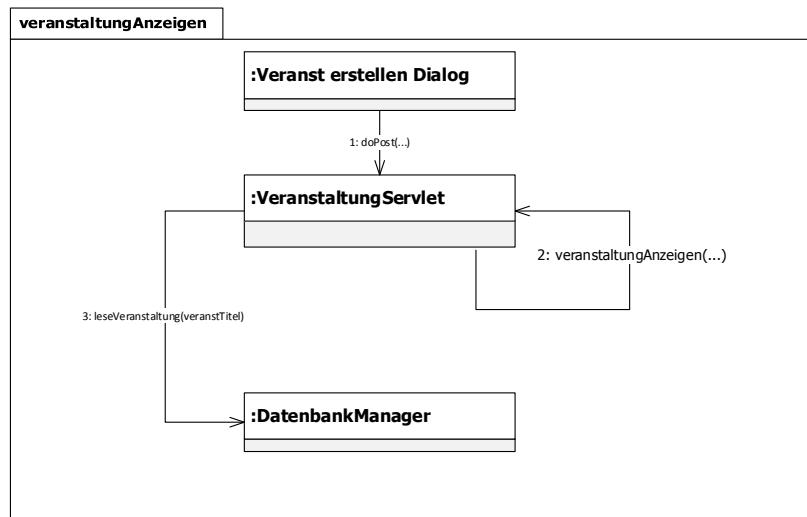


Abbildung 2.27: Veranstaltung anzeigen

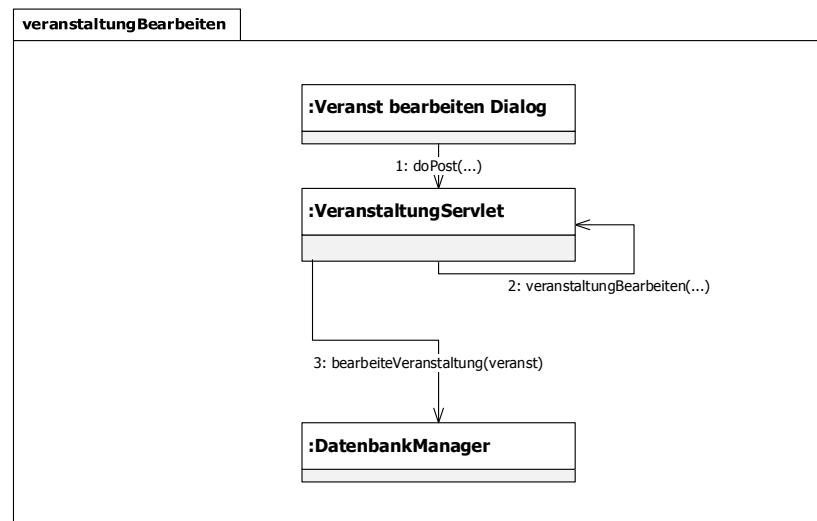


Abbildung 2.28: Veranstaltung bearbeiten

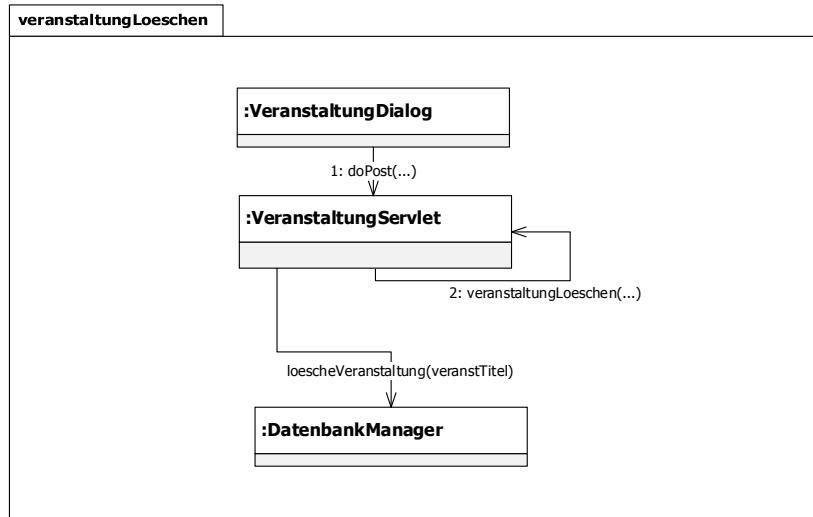


Abbildung 2.29: Veranstaltung löschen

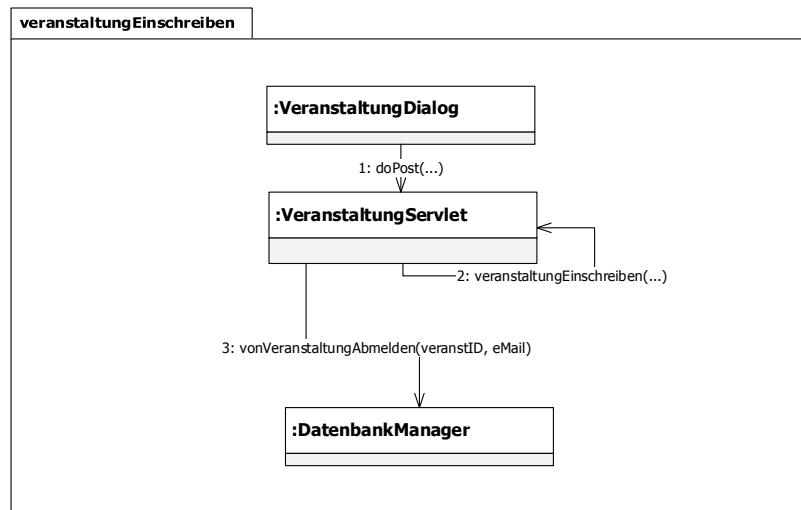


Abbildung 2.30: Veranstaltung einschreiben

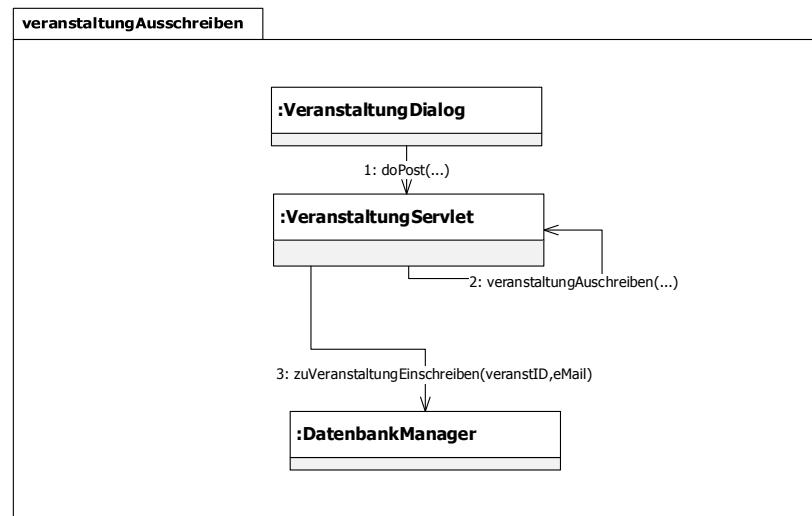


Abbildung 2.31: Veranstaltung ausschreiben

## 2.3 Methodenbeschreibung

### 2.3.1 BenutzerServlet Methoden

Operation	<b>istEingeloggt(session: HttpSession, response: HttpServletResponse): boolean</b>
Beschreibung	Überprüft anhand der übergebenen Session, ob ein Nutzer bereits eingeloggt ist oder nicht. Gibt "true" zurück, wenn die Session diverse Parameter beinhaltet und wenn nicht "false".
Operation	<b>registrieren(request: HttpServletRequest, response: HttpServletResponse): boolean</b>
Beschreibung	Die Daten des Benutzers werden aus der Request gelesen. Diese werden geprüft und dann wird daraus ein Benutzerobjekt erschaffen. Dieses wird an den Datenbankmanager weitergegeben, welcher dann den Benutzer in der DB anlegt. Gibt "true" zurück, wenn bei diesem Ablauf keine Fehler auftreten, und "false" bei Fehlern.
Operation	<b>login(request: HttpServletRequest, response: HttpServletResponse): boolean</b>
Beschreibung	Aus der übergebenen Request werden folgende Emailadresse und Passwort als Parameter ausgelesen. Dann werden sie an die Datenbank weitergegeben. Wenn die Anmeldedaten korrekt sind, wird der Benutzer aus der Datenbank mit der Emailadresse als Identifikator gelesen. Lege neue Session an und schreibe die Benutzerattribute hinein. Schreibt via PrintWriter die Benutzerdaten als JSON in die Response. Gibt "true" zurück, wenn bei diesem Ablauf keine Fehler auftreten, und "false" bei Fehlern.
Operation	<b>passwortReset(request: HttpServletRequest, response: HttpServletResponse): boolean</b>
Beschreibung	Aus der übergebenen Request werden die Parameter zum Zurücksetzen des Passworts herausgelesen. Anschließend wird ein neues Passwort generiert. Dieses wird in die DB geschrieben. Gibt "true" zurück, wenn bei diesem Ablauf keine Fehler auftreten, und "false" bei Fehlern.
Operation	<b>sendePasswortResetMail(generiertesPW: String): boolean</b>
Beschreibung	Das in passwortReset(rq, resp) generierte Passwort, wird an die Emailadresse des Benutzers, der sein Passwort zurücksetzen möchte, gesendet. Gibt "true" zurück, wenn bei diesem Ablauf keine Fehler auftreten, und "false" bei Fehlern.

Operation	<b>logout(HttpSession session): boolean</b>
Beschreibung	Löscht die übergebene HttpSession via invalidate() und somit alle temporär server-seitig gespeicherten Daten und Parameter. Gibt "true" zurück, wenn bei diesem Ablauf keine Fehler auftreten, und "false" bei Fehlern.

### 2.3.2 ProfilServlet Methoden

Operation	<b>profilAnzeigen(request: HttpServletRequest, response: HttpServletResponse): boolean</b>
Beschreibung	Liest aus der Request die ID des Benutzer aus und schickt damit eine Anfrage an den Datenbankmanager, der die Profildaten aus der DB liest. Diese Methode gibt dann via PrintWriter die Profildaten aus. Gibt "true" zurück, wenn bei diesem Ablauf keine Fehler auftreten, und "false" bei Fehlern.

Operation	<b>profilBearbeiten(request: HttpServletRequest, response: HttpServletResponse): boolean</b>
Beschreibung	Aus der Request werden die neuen Profildaten ausgelesen. Diese werden geprüft und wenn diese Prüfung erfolgreich war, werden sie über den Datenbankmanager in der DB geupdated. Gibt "true" zurück, wenn bei diesem Ablauf keine Fehler auftreten, und "false" bei Fehlern.

Operation	<b>passwortAendern(request: HttpServletRequest, response: HttpServletResponse): boolean</b>
Beschreibung	Löschen? siehe Benutzerservlet selbe Methode.

Operation	<b>benutzerLoeschen(request: HttpServletRequest, response: HttpServletResponse): boolean</b>
Beschreibung	Liest aus der Request ID und Verifikationsparameter für diese Handlung aus. Wenn Benutzer existiert, rufe Methode zum Löschen von Benutzern des Datenbankmanagers auf. Gibt "true" zurück, wenn bei diesem Ablauf keine Fehler auftreten, und "false" bei Fehlern.

### 2.3.3 VeranstaltungsServlet Methoden

Operation	<b>veranstaltungErstellen(Veranstaltung boolean)</b>	<b>veranstaltung):</b>
Beschreibung	Ein Dozent erstellt eine Veranstaltung und weist ihr verschiedene Eigenschaften zu. Die Veranstaltung bekommt eine individuelle ID. Außerdem wird geprüft ob es bereits eine Veranstaltung mit dem selben gibt.	
Operation	<b>veranstaltungenAnzeigen(int VeranstaltungsID): boolean</b>	
Beschreibung	Aus der Datenbank wird mit Hilfe der VeranstaltungsID die Informationen bezüglich der Veranstaltung gelesen und zurückgegeben. Falls es diese Veranstaltung in der Datenbank nicht gibt, gibt die Methode false zurück.	
Operation	<b>veranstaltungBearbeiten(int VeranstaltungsID): boolean</b>	
Beschreibung	Aus der Datenbank wird mit Hilfe der VeranstaltungsID die Informationen bezüglich der Veranstaltung gelesen und zurückgegeben. Der Benutzer kann diese nun ändern und die geänderten Daten werden in der Datenbank abgespeichert.	
Operation	<b>veranstaltungLoeschen(int VeranstaltungsID): boolean</b>	
Beschreibung	Aus der Datenbank wird mit Hilfe der VeranstaltungsID die gewünschte Veranstaltung gelöscht. Falls es diese Veranstaltung nicht gibt, gibt die Methode false zurück	
Operation	<b>veranstaltungEinschreiben(Benutzer benutzer,int Veranstaltungsid): boolean</b>	
Beschreibung	Ein Benutzer kann sich in die gewünschte Veranstaltung einschreiben. Seine ID wird dann in die betreffende Datenbank bezüglich der Veranstaltung geschrieben. Falls es diese Veranstaltung nicht gibt, gibt die Methode false zurück	
Operation	<b>veranstaltungAusschreiben(request HttpServletRequest): boolean</b>	
Beschreibung	Ein Benutzer kann sich in die gewünschte Veranstaltung ausschreiben. Seine ID wird dann von der betreffende Datenbank bezüglich der Veranstaltung gelöscht. Falls es diese Veranstaltung nicht gibt, oder er noch nicht eingeschrieben ist, gibt die Methode false zurück	

### 2.3.4 KarteikartenServlet Methoden

Operation	<b>karteikarteErstellen(Karteikarte karteikarte): boolean</b>
Beschreibung	Ein Benutzer erstellt eine Karteikarte und weist ihr verschiedene Eigenschaften zu. Die Karteikarte bekommt eine individuelle ID. Außerdem wird geprüft ob es bereits eine Karteikarte mit dem selben Inhalt gibt.

Operation	<b>karteikartenAnzeigen(int KarteikartenID): boolean</b>
Beschreibung	Aus der Datenbank wird mit Hilfe der KarteikartenID die Informationen bezüglich der Karteikarte gelesen und zurückgegeben. Falls es diese Karteikarte in der Datenbank nicht gibt, gibt die Methode false zurück

Operation	<b>karteikarteBewerten(int KarteikartenID): boolean</b>
Beschreibung	Aus der Datenbank wird die gewünschte Karteikarte gelesen. Der Benutzer kann diese nun mit einer Zahl bewerten. die Bewertung wird dann in der Datenbank gespeichert.Falls es diese Karteikarte in der Datenbank nicht gibt, gibt die Methode false zurück

Operation	<b>karteikarteBearbeiten(int KarteikartenID): boolean</b>
Beschreibung	Aus der Datenbank wird mit Hilfe der KarteikartenID die Informationen bezüglich der Karteikarte gelesen und zurückgegeben. Der Benutzer kann diese nun ändern und die geänderten Daten werden in der Datenbank abgespeichert.Falls es diese Karteikarte in der Datenbank nicht gibt, gibt die Methode false zurück

Operation	<b>karteikarteLoeschen(int KarteikartenID): boolean</b>
Beschreibung	Aus der Datenbank wird mit Hilfe der KarteikartenID die gewünschte Karteikarte gelöscht. Falls es diese Karteikarte nicht gibt, gibt die Methode false zurück

### 2.3.5 KommentarServlet Methoden

Operation	<b>kommentarErstellen(in request:HttpServletRequest, in response:HttpServletResponse): boolean</b>
Beschreibung	Aus der Request werden folgende Parameter ausgelesen: Der Inhalt("content"), Das Erstelltdatum("dateofbirth"),Die ErstellerID("creatorID") und die VaterId("fatherID") Dies wird an den Datenbankmanager weitergegeben.

Operation	<b>kommentarBewerten(in request:HttpServletRequest, in response:HttpServletResponse): boolean</b>
Beschreibung	Aus der Request werden folgende Parameter ausgelesen: Die kommentarid("commentID"), die Nuterid(userID") und die Bewertung("commentrating") Dies wird an den Datenbankmanager weitergegeben.

Operation	<b>kommentareAnzeigen(in request:HttpServletRequest, in response:HttpServletResponse): boolean</b>
Beschreibung	Aus der Request werden folgende Parameter ausgelesen: Die Kommentarid("commentID"). Dies wird an den Datenbankmanager weitergegeben.

Operation	<b>kommentarLoeschen(in request:HttpServletRequest, in response:HttpServletResponse): boolean</b>
Beschreibung	Aus der Request werden folgende Parameter ausgelesen: Die Kommentarid("commentID"). Dies wird an den Datenbankmanager weitergegeben.

### 2.3.6 NotizenServlet Methoden

Operation	<b>notizErstellen(in request:HttpServletRequest, in response:HttpServletResponse): boolean</b>
Beschreibung	Aus der übergebenen Request wird der Inhalt und die ID des Erstellers gelesen. Die Methode zum Notizen schreiben des Datenbankmanagers wird aufgerufen. Wenn der Ablauf fehlerfrei verlief, returned die Methode "true", wenn nicht "false".

Operation	<b>notizBearbeiten(in request:HttpServletRequest, in response:HttpServletResponse): boolean</b>
Beschreibung	Aus der übergebenen Request wird der neue Inhalt, die ID der Notiz und die ID des Erstellers gelesen. Die Methode zum Notizen ändern des Datenbankmanagers wird aufgerufen. Wenn der Ablauf fehlerfrei verlief, returned die Methode "true", wenn nicht "false".

Operation	<b>notizAnzeigen(in request:HttpServletRequest, in response:HttpServletResponse): boolean</b>
Beschreibung	Aus der übergebenen Request wird die ID der Notiz gelesen. Die Methode zum Notizen lesen des Datenbankmanagers wird aufgerufen. Returned den Inhalt via Printwriter. Wenn der Ablauf fehlerfrei verlief, returned die Methode "true", wenn nicht "false".

Operation	<b>notizLoeschen(in request:HttpServletRequest, in response:HttpServletResponse): boolean</b>
Beschreibung	Aus der übergebenen Request wird die ID der Notiz gelesen. Die Methode zum Notizen löschen des Datenbankmanagers wird aufgerufen, die Notiz wird so gelöscht. Wenn der Ablauf fehlerfrei verlief, returned die Methode "true", wenn nicht "false".

### 2.3.7 AdminServlet Methoden

Operation	<b>adminRolleAendern(HttpServletRequest request): boolean</b>
Beschreibung	Liest aus der Request die BenutzerID/Emailadresse und die neue Rolle aus. Anschließend wird mit dem Datenbankmanager die Rolle in der Datenbank geupdated. Der betroffene Benutzer hat nun neue Rechte. Gibt "true" zurück, wenn bei diesem Ablauf keine Fehler auftreten, und "false" bei Fehlern.

### 2.3.8 DatenbankManager Methoden

Operation	<b>leseBenutzer(String email): Benutzer</b>
Beschreibung	Liest Benutzer mit der angegebenen Email aus. Wird kein Benutzer gefunden, so gibt die Methode null zurück

Operation	<b>schreibeBenutzer(Benutzer benutzer): boolean</b>
Beschreibung	Legt neuen Benutzer in der Datenbank an. War das Speichern erfolgreich so gibt die Methode true zurück. Bei Auftreten eines Fehlers oder falls es schon einen Benutzer mit der selben eMail Adresse gibt, wird false zurückgegeben.

Operation	<b>bearbeiteBenutzer(Benutzer benutzer): boolean</b>
Beschreibung	Daten des angegebenen Benutzers werden in der Datenbank geupdated. Bei Erfolg liefert die Methode true zurück (auch wenn der Benutzer gar nicht in der Datenbank vorhanden ist). Bei einem Fehler false.

Operation	<b>loescheBenutzer (String eMail): boolean</b>
Beschreibung	Entfernt den Benutzer aus der Datenbank. Tritt ein Fehler auf gibt die Methode false zurück. Ansonsten true. (Auch wenn der Benutzer gar nicht in der Datenbank vorhanden war.)

Operation	<b>pruefeLogin (String eMail, String passwort): boolean</b>
Beschreibung	Überprüft, ob eMail und passwort zusammenpassen. Falls nicht oder falls ein Fehler in der Datenbank auftritt gibt die Methode false zurück. Ansonsten true.
Operation	<b>passwortAendern (String eMail, String neuesPasswort): boolean</b>
Beschreibung	Ändert das Passwort des angegebenen Benutzers. Liefert bei Erfolg true zurück. (Auch wenn der Benutzer gar nicht in der Datenbank vorhanden ist.) Bei einem Fehler wird false zurückgegeben.
Operation	<b>leseVeranstaltung (String veranstTitel): Veranstaltung</b>
Beschreibung	Holt Veranstaltung mit dem Titel des Parameters aus der Datenbank. Ist keine Veranstaltung mit diesem Titel vorhanden, wird null zurückgegeben.
Operation	<b>schreibeVeranstaltung (Veranstaltung veranst): boolean</b>
Beschreibung	Fügt neue Veranstaltung in die Datenbank ein. Bei Erfolg wird true zurückgegeben. Bei einem Fehler in der Datenbank oder falls es schon eine Veranstaltung mit dem gleichen Titel gibt, wird false zurückgeliefert.
Operation	<b>bearbeiteVeranstaltung (Veranstaltung veranst): boolean</b>
Beschreibung	Daten der angegebenen Veranstaltung werden in der Datenbank geupdatet. Bei Erfolg liefert die Methode true zurück (auch wenn die Veranstaltung gar nicht in der Datenbank vorhanden war). Bei einem Fehler false.
Operation	<b>loescheVeranstaltung (String veranstTitel): boolean</b>
Beschreibung	Entfernt die Veranstaltung aus der Datenbank. Tritt ein Fehler auf gibt die Methode false zurück. Ansonsten true. (Auch wenn die Veranstaltung gar nicht in der Datenbank vorhanden war.)
Operation	<b>leseKarteikarte (Integer karteikID): boolean</b>
Beschreibung	Holt Daten der Karteikarte anhand der ID aus der Datenbank und return
Operation	<b>schreibeKarteikarte (Karteikarte karteik): boolean</b>
Beschreibung	Fügt neue Karteikarte in die Datenbank ein. Bei Erfolg wird true zurückgegeben. Bei einem Fehler in der Datenbank wird false zurückgeliefert.

Operation	<b>bearbeiteKarteikarte (Karteikarte karteik): boolean</b>
Beschreibung	Daten der angegebenen Karteikarte werden in der Datenbank geupdatet. Bei Erfolg liefert die Methode true zurück (auch wenn die Karteikarte gar nicht in der Datenbank vorhanden war). Bei einem Fehler false.
Operation	<b>loescheVeranstaltung (Integer karteikID): boolean</b>
Beschreibung	Entfernt die Karteikarte aus der Datenbank. Tritt ein Fehler auf gibt die Methode false zurück. Ansonsten true. (Auch wenn die Karteikarte gar nicht in der Datenbank vorhanden war.)
Operation	<b>bewerteKarteikarte (Integer karteikID, Integer bewert, String benutzer): boolean</b>
Beschreibung	Speichert die Bewertung, die der Benutzer dieser Karteikarte gegeben hat. Die Gesamtbewertung der Karteikarte wird entsprechend angepasst. Bei einem Fehler wird false zurückgeliefert ansonsten true.
Operation	<b>bewerteKarteikarte (Integer karteikID, Integer bewert, String benutzer): boolean</b>
Beschreibung	Speichert die Bewertung, die der Benutzer dieser Karteikarte gegeben hat. Die Gesamtbewertung der Karteikarte wird entsprechend angepasst. Bei einem Fehler wird false zurückgeliefert ansonsten true.
Operation	<b>hatKarteikarteBewertet (Integer karteikID, String benutzer): boolean</b>
Beschreibung	Gibt true zurück, falls der Benutzer diese Karteikarte bereits bewertet hat. Ansonsten wird false zurückgegeben.
Operation	<b>leseKommentare (Integer karteikID): Kommentar[]</b>
Beschreibung	Gibt alle Kommentare zu einer Karteikarte zurück. Bei einem Fehler wird null zurückgegeben
Operation	<b>schreibeKommentar (Kommentar kommentar): boolean</b>
Beschreibung	Fügt neuen Kommentar in die Datenbank ein. Bei Erfolg wird true zurückgegeben. Bei einem Fehler in der Datenbank wird false zurückgeliefert.

Operation	<b>bearbeiteKommentar (Kommentar kommentar): boolean</b>
Beschreibung	Daten des angegebenen Kommentars werden in der Datenbank geupdated. Bei Erfolg liefert die Methode true zurück (auch wenn der Kommentar gar nicht in der Datenbank vorhanden war). Bei einem Fehler false.
Operation	<b>loescheVeranstaltung (Integer kommentarID): boolean</b>
Beschreibung	Entfernt den Kommentar aus der Datenbank. Tritt ein Fehler auf gibt die Methode false zurück. Ansonsten true. (Auch wenn der Kommentar gar nicht in der Datenbank vorhanden war.)
Operation	<b>bewerteKommentar (Integer kommentarID, Integer bewert, String benutzer): boolean</b>
Beschreibung	Speichert die Bewertung, die der Benutzer diesem Kommentar gegeben hat. Die Gesamtbewertung des Kommentars wird entsprechend angepasst. Bei einem Fehler wird false zurückgeliefert ansonsten true.
Operation	<b>hatKommentarBewertet (Integer kommentarID, String benutzer): boolean</b>
Beschreibung	Gibt true zurück, falls der Benutzer diesen Kommentar bereits bewertet hat. Ansonsten wird false zurückgegeben.
Operation	<b>istModerator (Veranstaltung veranst, String benutzerMail): boolean</b>
Beschreibung	Gibt true zurück, falls der Benutzer ein Moderator dieser Veranstaltung ist. Ansonsten false.
Operation	<b>zuVeranstaltungEinschreiben (String veranstTitel, String eMail): boolean</b>
Beschreibung	Schreibt Benutzer in die Veranstaltung ein. Tritt ein Fehler auf gibt die Methode false zurück. Ansonsten true.
Operation	<b>vonVeranstaltungAbmelden (String veranstTitel, String eMail): boolean</b>
Beschreibung	Meldet Benutzer von der Veranstaltung ab. Tritt ein Fehler auf gibt die Methode false zurück. Ansonsten true.

# 3 Entwicklungs vorgaben

Unser Softwaresystem wird mit in Java, sowie unter Zuhilfenahme von JavaScript/JQuery, Json, CSS, HTML. Als Datenbank wird eine MySQL-Datenbank verwendet, die über das Softwarepaket XAMPP gestartet wird. Als Entwicklungsumgebung wird Eclipse verwendet.

## 3.1 Rollenverteilung

- Projektleiter: Fabian Schilha
- Chefdesigner: Andreas Rottach
- Projektverwalter: Julius Friedrich
- QS-Ingenieur: Lorenz Weiß
- Validierer & Verifizierer: Matthias Englert
- Doku-Spezialist: Marius Kircher

## 3.2 Projektplan

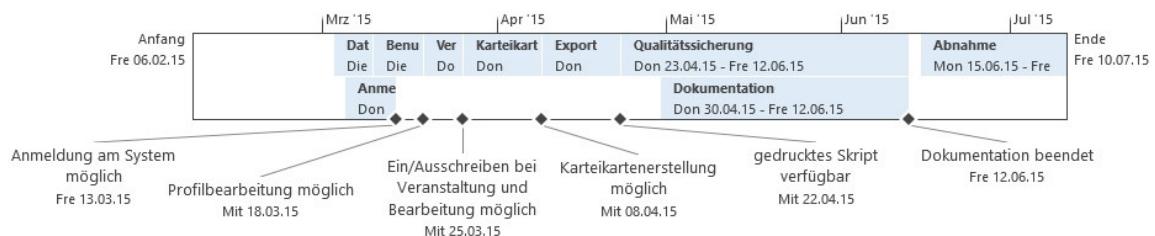
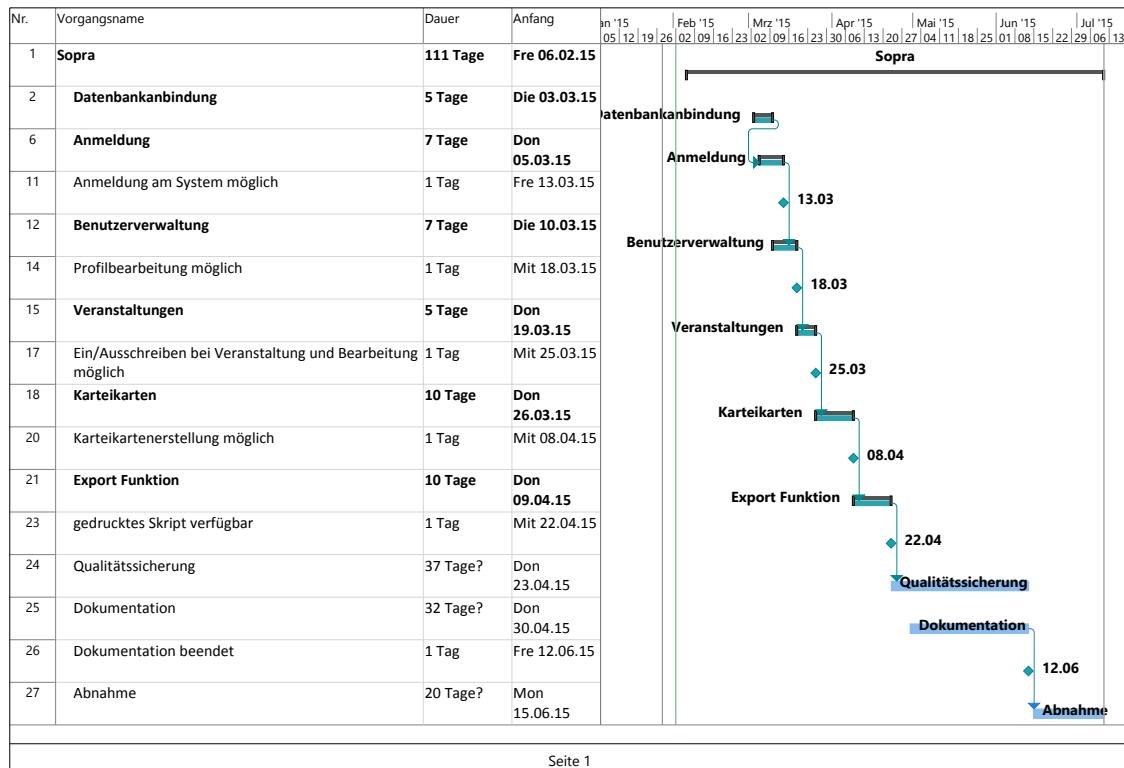


Abbildung 3.1: visueller Plan



Seite 1

Abbildung 3.2: struktureller Plan

### 3.3 Programmierkonventionen

Diese Programmierkonventionen wurden großteils aus der Vorlage aus Moodle übernommen.

#### 3.3.1 Allgemeine Konventionen

- Alle Elemente des Quellcodes erhalten ausschließlich deutsche Bezeichner!
- Jede Datei dokumentiert in den ersten Zeilen den Inhalt, den verantwortlichen Programmautor, den Reviewer/Sachverständigen.
- Jeder Block muss 4 Leerzeichen eingerückt werden (Keine Tabs sondern Spaces, Einstellungen in Eclipse vornehmen).
- Benutzer Sie aussagekräftige Bezeichner.
- Abkürzungen nur verwenden, wenn sie eindeutig und gebräuchlich sind.
- Jeder Programmblock wird kommentiert. Insbesondere Klassen, Methoden und Attribute; aber auch Schleifen und wichtige if then else Konstrukte.

- Jede Änderung muss in den Commits beschrieben werden. Es muss klar sein, mit welchem Commit was geändert wurde.

### 3.3.2 Java-Konventionen

- Keine „\_“ innerhalb von Bezeichnern. Verwenden Sie CamelCase.
- Verwenden Sie Substantive für Werte, Verben für Tätigkeiten und Eigenschaftswörter für Bedingungen, um die Bedeutung von Bezeichnern eindeutig hervortreten zu lassen.
- Getter und Setter beginnen immer mit „get“ und „set“.
- Geschweifte Klammern befinden sich immer am Anfang einer Zeile.
- Namen von Prozeduren, Methoden beginnen mit kleinen Buchstaben
- Namen von Modulen, Klassen, Datentypen und beginnen mit einem großen Buchstaben.
- Konstanten (*final*) werden komplett großgeschrieben.
- Temporäre Elemente wie Parameter und lokale Variablen werden klein geschrieben.

# 4 Benutzerschnittstelle

## 4.1 Startseite

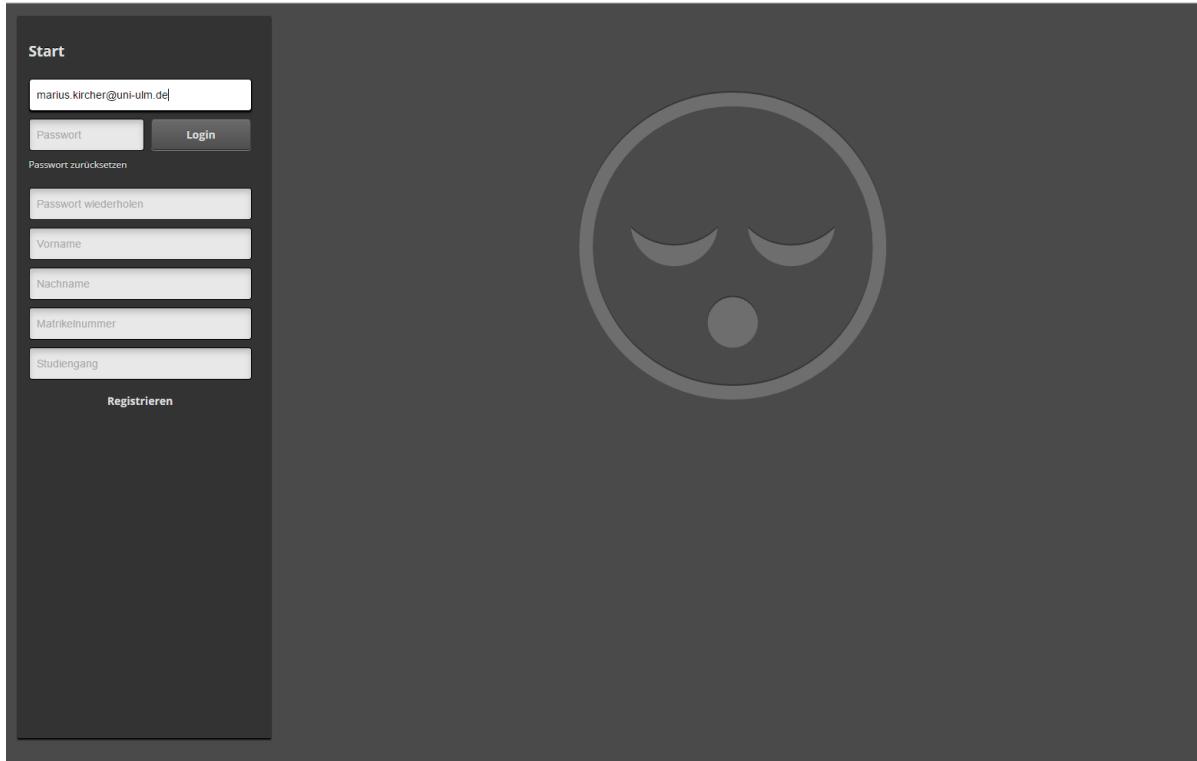


Abbildung 4.1: Startseite mit Formular zum Anmelden und Registrieren

Bevor alle Funktionalitäten des Systems aktiv werden muss sich ein Nutzer einloggen bzw. - wenn er noch kein eingetragenes Konto besitzt - beim System registrieren. Zum Einloggen muss er die ersten beiden Felder auf obigem Screenshot korrekt ausfüllen und den Login-Button betätigen.

Falls der Nutzer sein Passwort vergessen hat, füllt er nur das Feld für die E-Mail-Adresse aus und klickt auf Passwort zurücksetzen. Sofern ein Eintrag mit dieser E-Mail-Adresse in der Datenbank vorhanden ist, wird sein Passwort auf einen zufälligen Wert gesetzt und ihm per E-Mail mitgeteilt.

Zum Registrieren müssen alle Felder ausgefüllt werden und der Registrieren-Button betätigt werden. Danach kann sich der Nutzer mit seinen Daten einloggen.

## 4.2 Hauptseite

The screenshot shows the university's main page. On the left, there is a sidebar with a user profile for 'Marius Kircher' (Student, Abmelden), a section for 'Angepinnte Veranstaltungen' (Analysis I, Algorithmen, Lineare Algebra), and a 'Benachrichtigungen' (Notifications) box containing several recent activity items. The main content area has a search bar ('Veranstaltungen und Personen finden...') and a green button ('+ Veranstaltung erstellen'). It displays a list of events under tabs: 'Meine' (selected), 'Studiengang', 'Semester', and 'Alle'. The first event listed is 'Molekularbiologie I' taught by Prof. Dr. Oberhuber with 60 participants, WSe 14/15. A detailed description of the course is provided, along with a 'Weniger' (less) link and a 'Ausschreiben' (apply) button. Below it are 'Algorithmen' (taught by Prof. Dr. Schöning, 40 participants, WSe 14/15) and 'Lineare Algebra' (taught by Dr. Liebezeit, 30 participants, WSe 14/15). Each event entry includes a 'Mehr' (more) link.

Abbildung 4.2: Hauptseite mit Liste der Veranstaltungen im System

Diese Seite wird dem Nutzer nach erfolgreichem Einloggen angezeigt. Hier kann er alle Veranstaltungen einsehen und sich in diese einschreiben bzw. sich aus diesen ausschreiben.

Die Box auf der linken Seite wird den Nutzer im gesamtem System in dieser Form begleiten. Darin sieht er oben Informationen zu seinem Nutzerkonto. Darunter folgt eine Liste der Veranstaltungen, die der Nutzer angepinnt hat, weil er oft darauf zugreifen muss. Am unteren Rand der Box befindet sich der Benachrichtigungsbereich. Hier werden aktuelle Aktivitäten in Veranstaltungen angezeigt, in die er eingeschrieben ist. Zu den Aktivitäten gehören das Hinzufügen neuer Karteikarten oder Kommentare etc.

Der Hauptbereich rechts enthält oben ein multifunktionales Suchfeld. Dieses wird automatisch fokussiert, sodass der Nutzer sehr schnell und nur per Tastatureingaben Veranstaltungen und Nutzerprofile finden und aufrufen kann. Man kennt das Prinzip von vielen Webseiten.

Oben rechts hat ein Nutzer in der Rolle *Dozent* per Buttondruck die Möglichkeit eine neue Veranstaltung zu erstellen. Der dadurch getriggerte Dialog ist auf der nächsten Seite zu sehen.

Die Liste der Veranstaltungen steht im Mittelpunkt. Sie ist durch Tabs in Kategorien aufgeteilt, um Übersichtlich zu gewinnen.

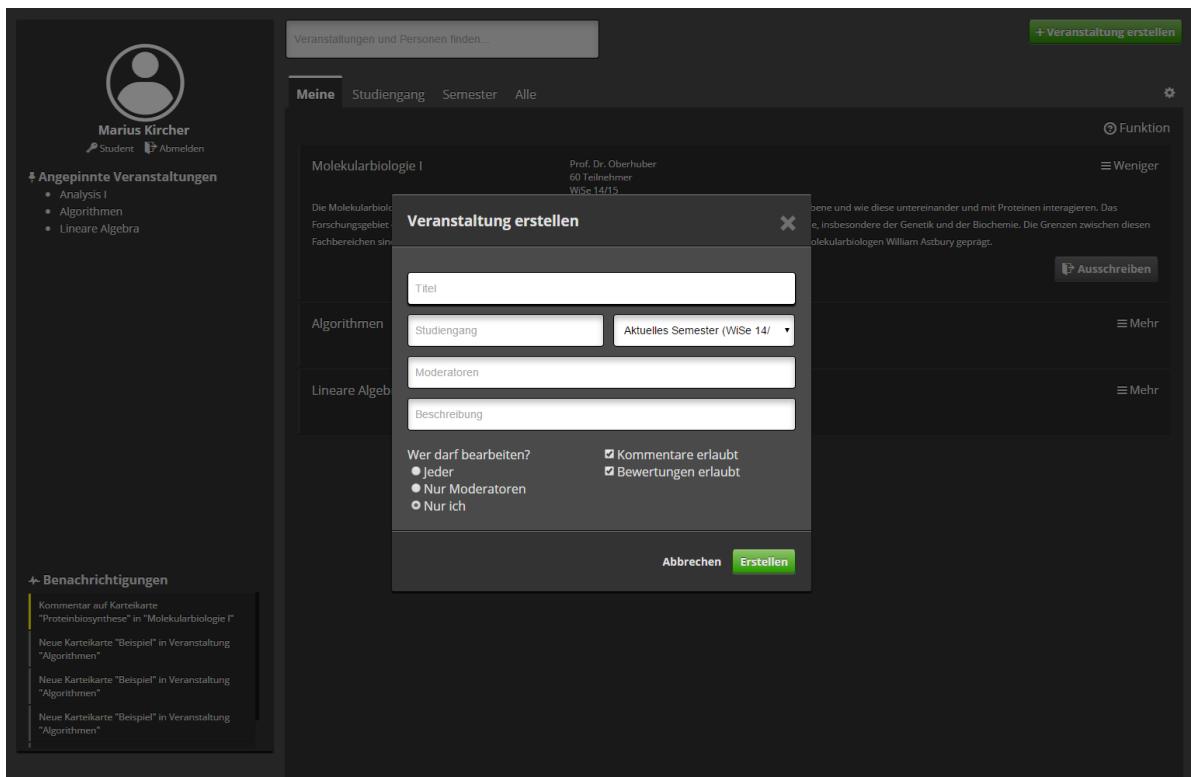


Abbildung 4.3: Modaler Dialog zum erstellen einer neuen Veranstaltung

Beim Erstellen einer Veranstaltung werden über das Formular im Popup Fenster die nötigen Informationen vom Benutzer eingegeben. Nach korrekter Eingabe und erfolgreichem Abschicken des Formulars erscheint die Ansicht, die ab der nächsten Seite vorgestellt wird (natürlich noch ohne jegliche Karteikarten).

The screenshot shows a user profile editing interface. At the top left is a navigation bar with a back arrow, the text "Zurück zu den Veranstaltungen", and a user icon. Below this is a header with the name "Marius Kircher" and the title "Student". On the right is a red button labeled "Konto löschen". The main area contains a form with fields for "E-Mail" (marius.kircher@uni-ulm.de), "Passwort" (\*\*\*\*\*), "Vorname" (Marius), "Nachname" (Kircher), "Matrikelnummer" (123456), "Studiengang" (Medieninformatik), and "Rolle" (Student). A "Speichern" button is at the bottom right. To the left of the form is a sidebar titled "Benachrichtigungen" with several notifications listed.

Abbildung 4.4: Nutzerprofil anzeigen und ändern

Diese Ansicht zeigt Informationen über einen Nutzer. Ist dies derselbe wie der eingeloggte Nutzer oder ist ein Administrator eingeloggt, steht die Option zum Ändern der Daten zur Verfügung. Außerdem kann in diesem Fall über den roten Button oben rechts das Konto komplett gelöscht werden.

## 4.3 Veranstaltungsseite

The screenshot shows a digital note-taking application interface. At the top, there is a navigation bar with icons for Filter, Sortieren, Ansicht, Skript exportieren, and + Karteikarte erstellen. On the right side of the header, it says Dr. Liebezeit Wintersemester 2014/15. Below the header, the title "Analysis I" is displayed. On the left, there is a sidebar with a user profile for Marius Kircher (Student, Abmelden), a list of pinned events ("Angepinnte Veranstaltungen" including Analysis I, Algorithmen, Lineare Algebra), and a list of notifications ("Benachrichtigungen" including comments on cards from "Proteinbiosynthese" and "Algorithmen"). The main content area displays a list of cards. One card is highlighted with a grey background and has a note "Notizen Verstehe ich nicht" written on it. Below the card list, there is a section titled "Kommentare (6)" showing several comments from users like Matthias Englert, Jan Wille, Erik Hintz, and Torsten Tostensson. At the bottom of the screen, there is a footer with a "Neues Thema beginnen..." button.

Abbildung 4.5: Seite einer Veranstaltung

Diese Seite ermöglicht die Interaktion mit den Karteikarten in einer Veranstaltung. Am oberen Rand stehen die vier Werkzeuge *Filtern*, *Sortieren*, *Ansicht* und *Skript exportieren* bereit. Folgende Filter stehen zur Verfügung:

- Keiner (Default)
- Nur als wichtig markierte Karteikarten
- Nur Bilder
- Nur Videos
- Nur Karteikarten, die einen bestimmten Text enthalten

Folgende Sortieroptionen stehen zu Verfügung:

- Vom Ersteller vorgegebene Struktur (Default)
- Nach bestbewerteten Karteikarten absteigend
- Nach jüngstem Änderungsdatum absteigend

Folgende Ansichtsoptionen stehen zur Verfügung:

- Leseansicht (Default)
- Baumansicht
- Graphansicht

Die Funktion *Skript exportieren*, die über den Button getriggert wird, ruft auf dem Server einen Algorithmus auf, der Karteikarteninhalte in LaTeX-Syntax konvertiert. Diese dient dann als Eingabe für den *PDF LaTeX*-Compiler. Das erzeugte PDF Dokument kann letztlich vom Nutzer über einen üblichen Downloaddialog heruntergeladen werden. Der grüne Button rechts oben erzeugt eine leere Karteikarte, die dann wie auf den folgenden Seiten zu sehen mit *CKEditor* bearbeitet werden kann.

Unter der Werkzeugeiste werden Informationen über die Veranstaltung angezeigt. Der Titel ist persistent sichtbar, während Dozent, Semester und Bearbeiten sowie Löschen Buttons nur bei mouseover angezeigt werden. Der Bearbeiten Button triggert den Dialog, der oben schon zum Erstellen verwendet wurde. Der Löschen Button triggert eine *Sind Sie sicher?*-Abfrage, bevor die Veransaltung mitsamt allen Karteikarten gelöscht wird.

Unter der Informationsleiste folgt der Hauptteil: Die Karteikartenansicht. Die hier abgebildete Karteikarte hat den Inhaltstyp *Text*. Weitere Typen *Bild* und *Video* werden unten vorgestellt. Zur Rechten einer Karteikarte befindet sich stets ein Notizfeld, in das private Notizen vom Benutzer geschrieben werden können. Unter Karteikarten befindet sich die Kommentarbox. Kommentare können erstellt, beantwortet, bewertet und gelöscht werden. Zur Rechten von Karteikarten befinden sich Upvotebutton, Downvotebutton und Votecounter. Das Prinzip ist von *StackOverflow* bekannt. Wenn der Mauszeiger in der Nähe ist erscheinen darunter spezielle Optionen für eine Karteikarte. Diese werden unten im Detail beschrieben.

The screenshot shows a digital note-taking application interface. At the top, there's a header bar with various icons and a button labeled '+ Karteikarte erstellen' (Create Note Card). Below the header is a toolbar with text and style options like bold, italic, underline, etc. The main area is divided into sections:

- Notizen**: A large text input field containing the following text:
 

5.19 Definition. Sei  $D \subset \mathbb{R}^m \times \mathbb{R}^d$  und  $F : D \rightarrow \mathbb{R}$  deine Funktion.  
 Eine implizitdefinierte Funktion oder implizite Funktion ist eine  
 Funktion  $y : \mathbb{R}^m \rightarrow \mathbb{R}^d$ , deren Funktionswerte durch die Gleichung  
 $F(x, y(x)) = 0$  definiert sind.

Ein Beispiel findet man auf [dieser Karteikarte](#).  
 Streber können sich diesen [Wikipediaartikel](#) zu Gemüte führen ;)
- Kommentare (6)**: A list of comments from users:
  - Mir ist nicht so ganz klar wie man allgemein die Randkurve von einem Körper bestimmt. Z.B. bei der Aufgabe 1c braucht man ja die Randkurve des Kegels für den Satz von Stokes — Matthias Englert 28.Jan.'15 um 15:29
  - Ist der Papst katholisch? — Jan Willem Liebezeit 28.Jan.'15 um 16:29
  - Welche Farbe hat das weiße Pferd von Napoleon? — Matthias Englert 28.Jan.'15 um 17:31
  - Die Randkurve bestimmt man wie im Skript Satz 13.5 — Erik Hintz 29.Jan.'15 um 12:29
  - Antworten...
  - Ich check hier garnix alder wasn da los? — Torsten Tostansson 30.Jan.'15 um 03:29
  - Hör auf zu labern du Labersack -.- — Jan Willem Liebezeit 30.Jan.'15 um 04:00
  - Antworten...
  - Neues Thema beginnen...

On the left sidebar, there are sections for 'Angepinnte Veranstaltungen' (Pinned Events) and 'Benachrichtigungen' (Notifications), each listing several items.

Abbildung 4.6: Interaktion mit Karteikarten

Hier ist zu sehen wie Karteikarten bearbeitet werden. Erhält das Textfeld einer Karteikarte Fokus erscheint der abgebildete *CKEditor*, ein JavaScript-Framework für WYSIWYG-Editoren, die HTML-Code erzeugen. Dieser ermöglicht das Einfügen von formatiertem Text, Links, mathematischen Formeln (*MathJax*-Widgets), Tabellen usw.

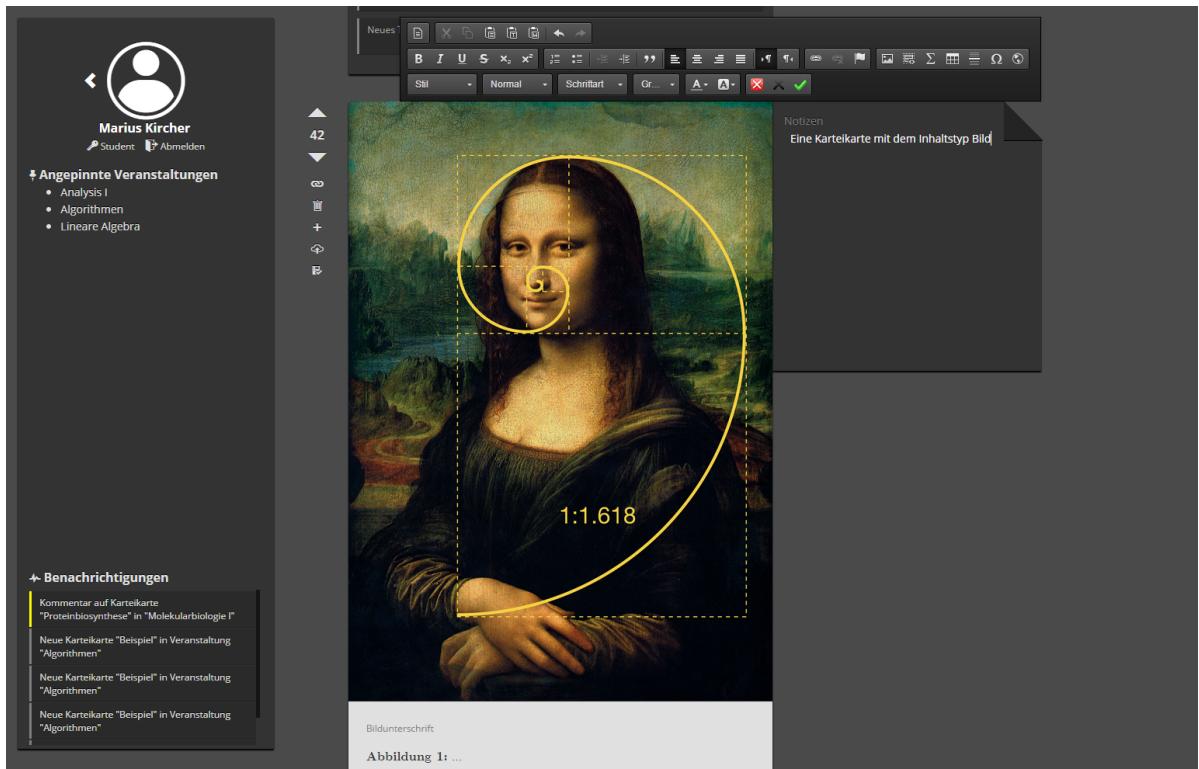


Abbildung 4.7: Karteikarte mit Inhaltstyp *Bild*

Hier ist dargestellt, wie Karteikarten erscheinen, wenn sie Bilder enthalten. Um zusätzliche Informationen zu vermitteln, ist auch die Eingabe von Text über das Feld *Bildunterschrift* möglich. Darunter folgt die bereits bekannt Kommentarbox, die auf dem Screenshot keinen Platz mehr fand.

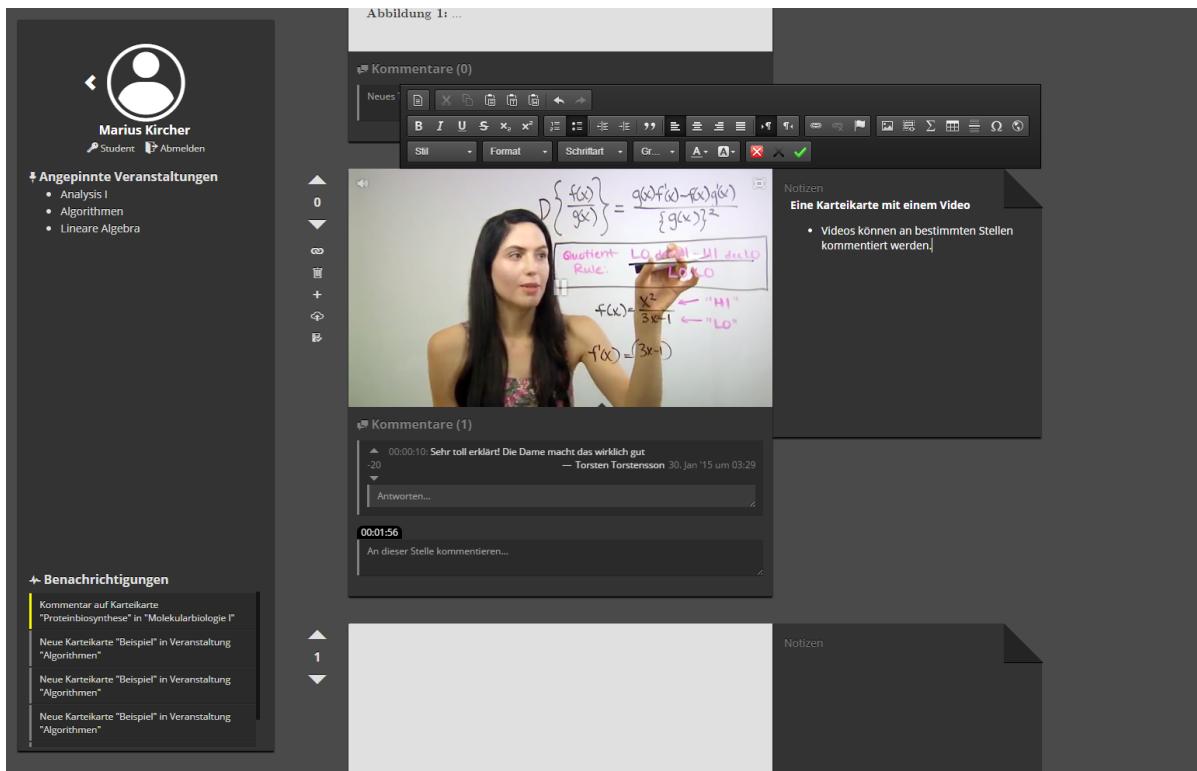


Abbildung 4.8: Karteikarte mit Inhaltstyp *Video*

Karteikarten mit einem Video erscheinen wie oben dargestellt. Kommentare enthalten nun zusätzlich den Zeitpunkt im Videos, zu dem sie erstellt wurden.

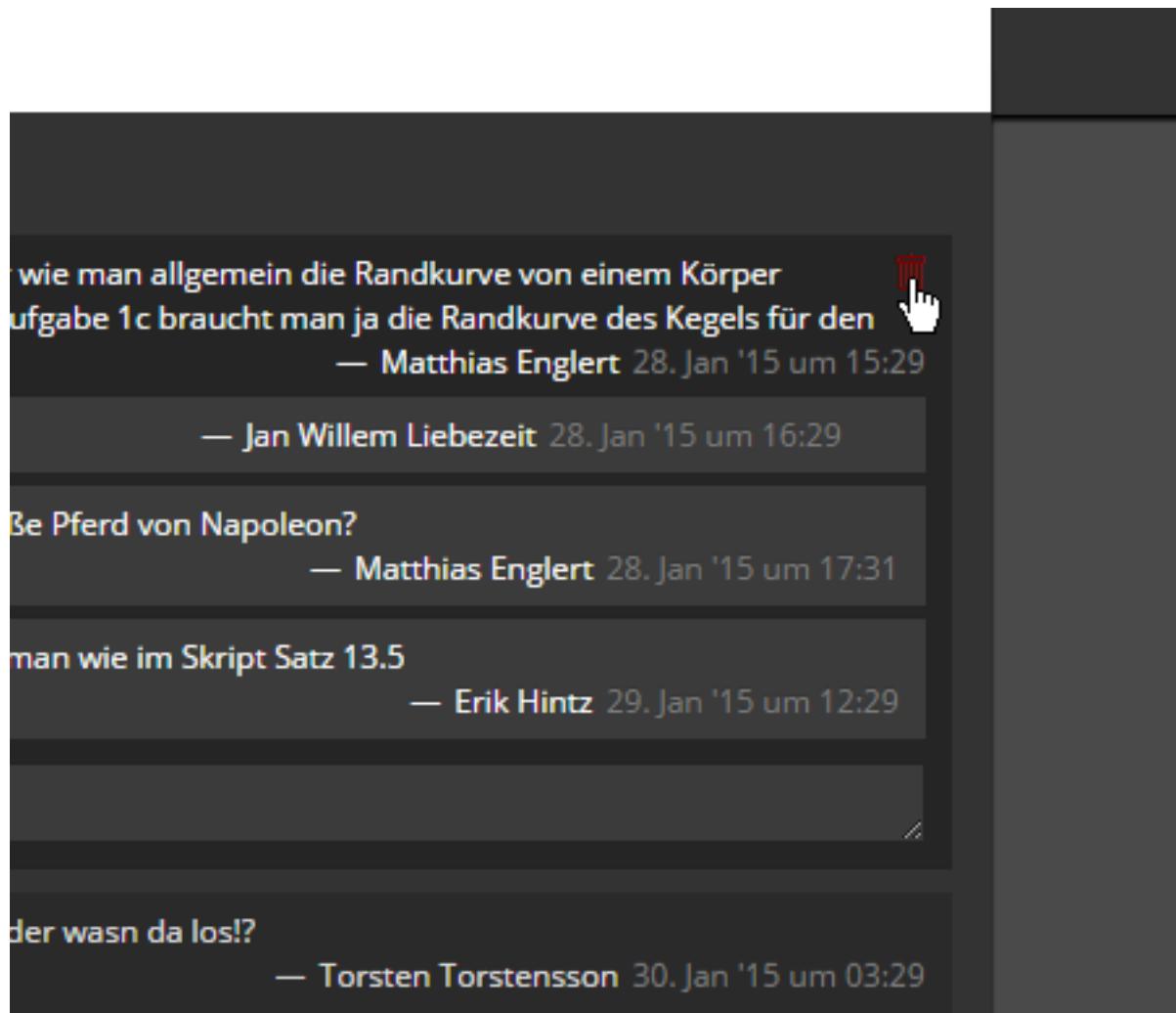


Abbildung 4.9: Löschen von Kommentaren

Das Icon, welches das Löschen von Karteikarten triggert, erscheint nur bei mouseover über dem betreffenden Kommentar.

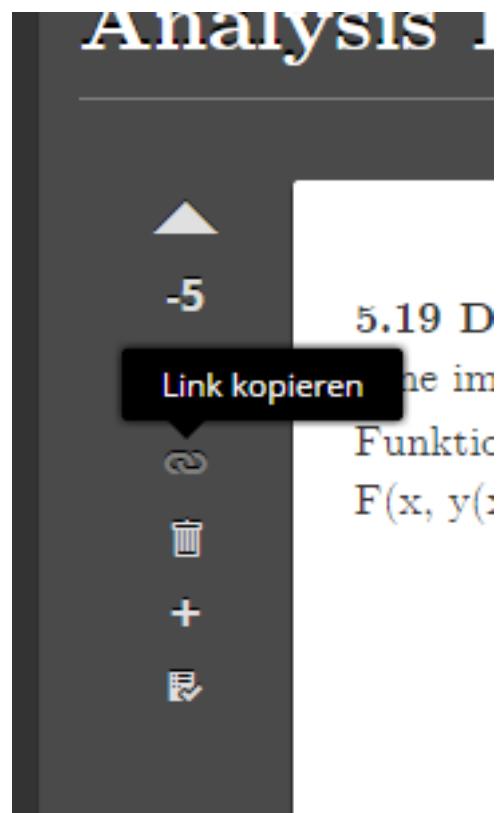


Abbildung 4.10: Spezielle Optionen für Karteikarten

Im Folgenden werden die speziellen Optionen für Karteikarten beschrieben.

**Link kopieren** Schreibt einen permanenten Hyperlink zur betreffenden Karteikarte in die Zwischenablage

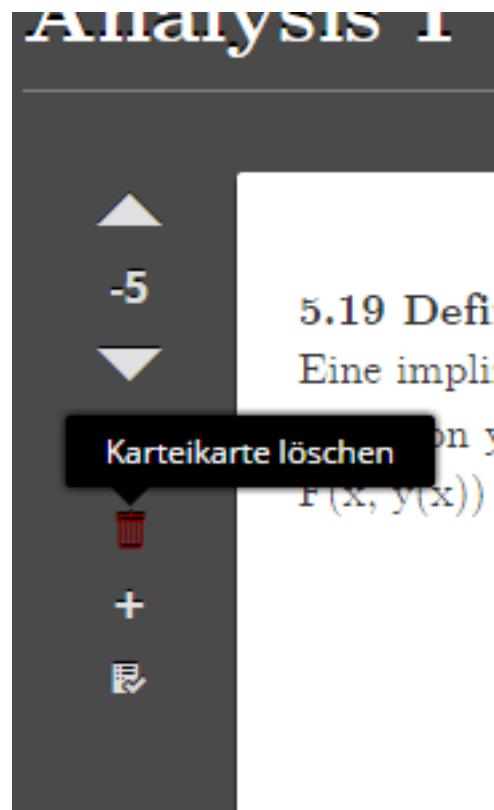


Abbildung 4.11: Spezielle Optionen für Karteikarten

**Karteikarte Löschen** Triggert einen *Sind Sie sicher?*-Dialog bevor die Karteikarte mit-  
samt Kommentaren und Notizen gelöscht wird.

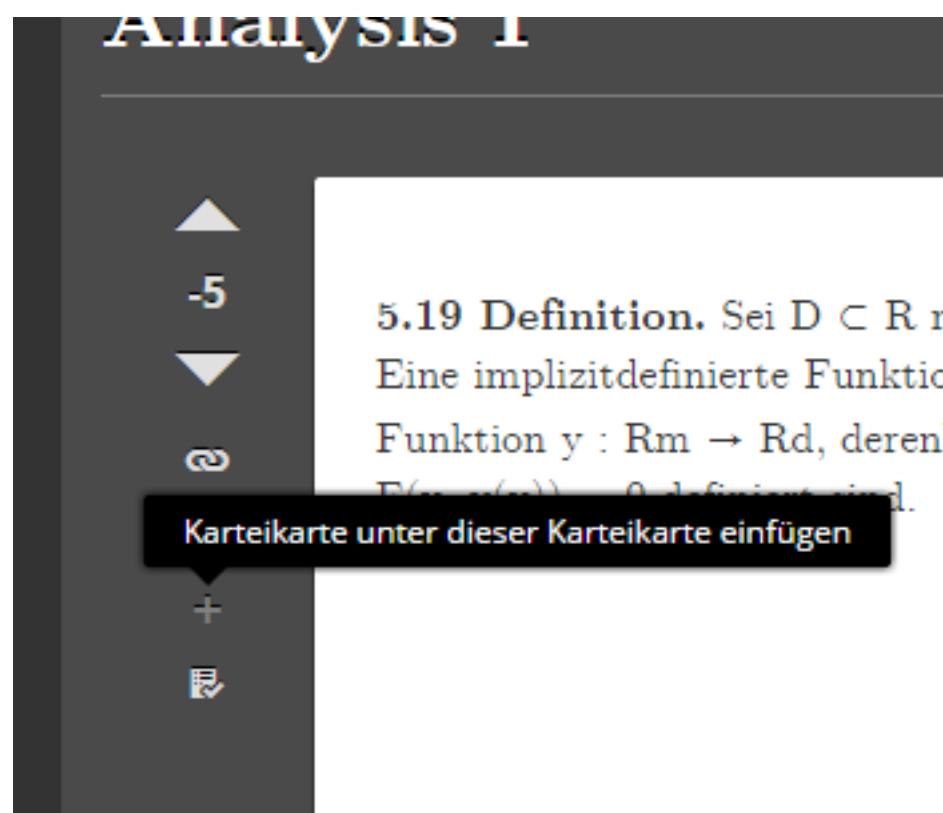


Abbildung 4.12: Spezielle Optionen für Karteikarten

**Karteikarte unter dieser Karteikarte einfügen** Erstellt eine Kindkarteikarte unter der betreffenden Karteikarte.

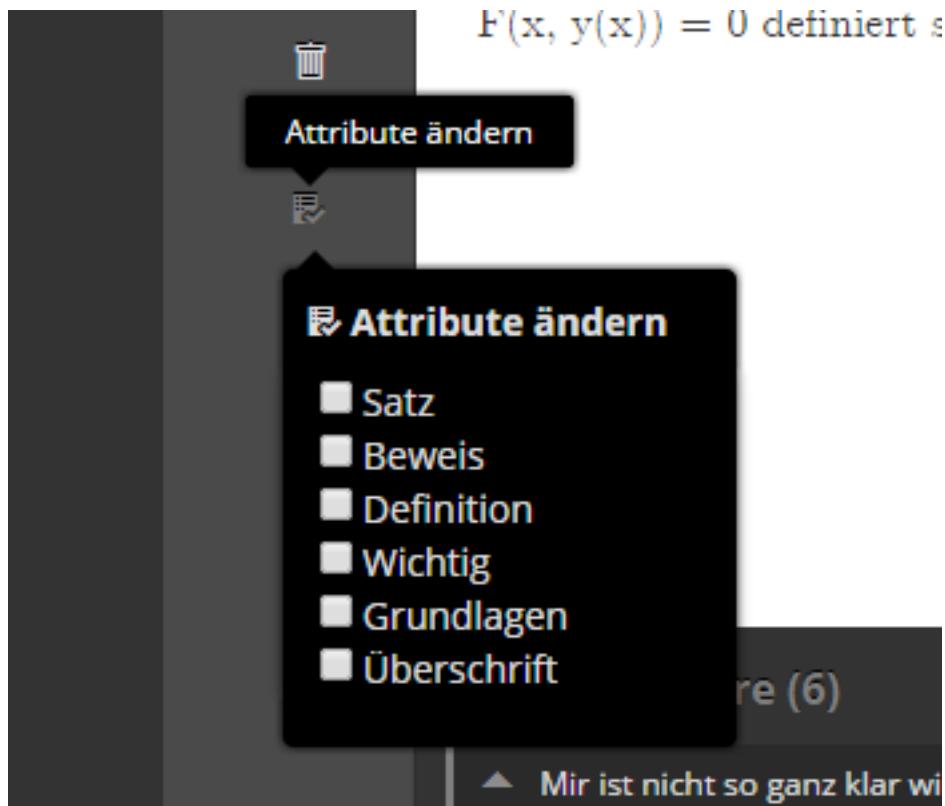


Abbildung 4.13: Spezielle Optionen für Karteikarten

**Attribute ändern** Ermöglicht das Zuweisen von Attributen wie u.a. *wichtig* zur betreffenden Karteikarte.

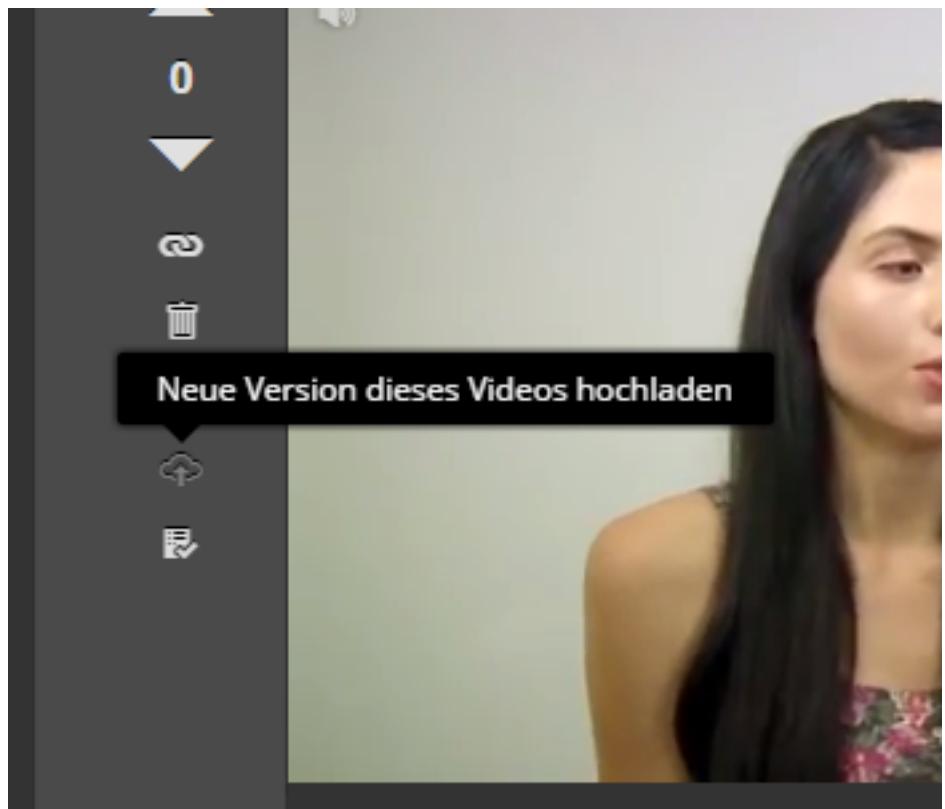


Abbildung 4.14: Spezielle Optionen für Karteikarten

**Neue Version dieses Bildes bzw. Videos hochladen** Triggert einen üblichen Dateibrowser-Dialog, um das Bild bzw. Video gegen ein lokales Bild bzw. Video auszutauschen.

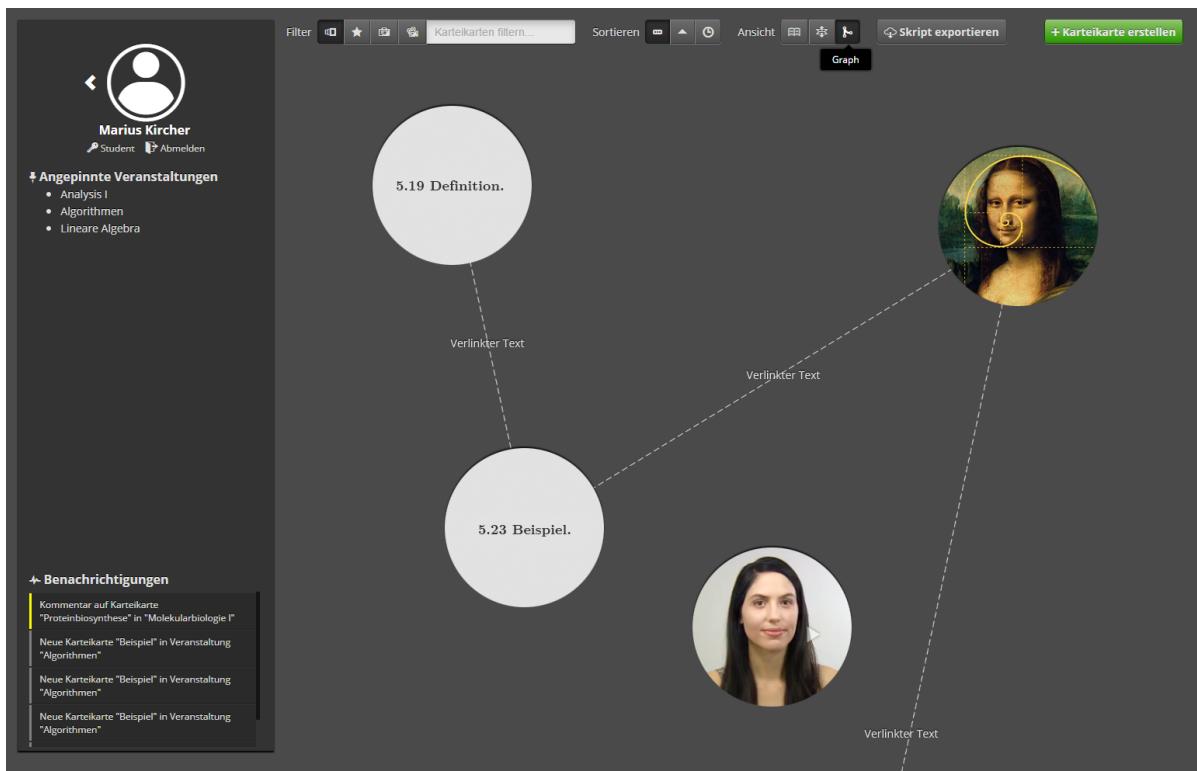


Abbildung 4.15: Karteikarten in Graphansicht

Wie bereits dokumentiert können Karteikarten Links auf andere Karteikarten enthalten. Dadurch entsteht implizit die Struktur eines Graphen. Die Graphansicht stellt dies visuell dar, um dem Benutzer einen besseren Überblick über das Dokument zu verschaffen.