V Doku Martin Christian
V Impl Christian Würker

Softwaretechnik Praktikum
Gruppe ueb22
Aufgabenblatt Ueb-4

erstellt am 09.06.2003

Designbeschreibung

- Skizze -

1.Überblick

Funktionen

Diese Anwendung soll den Übungsbetrieb zu einer universitären Lehrveranstaltung unterstützen.
Dazu stellt es folgende Nutzerfunktionen zur Verfügung:

- Erfassung von Personen und Zuteilung von Rollen
- Erstellen und Einteilen der Übungsgruppen
- Up- (Dozent) und Download (Student) von Materialien / Aufgaben
- Up- (Student) und Download (Tutor) von Lösungen, mit Terminprüfung
- Upload der Korrekturergebnisse (Tutor) und anonymisierte Einsicht (Student).
- Anmeldung zur Klausur (Student)
- Erstellung von Klarlisten (Dozent)
- Einstellen und lesen (alle angemeldeten Nutzer) von Nachrichten im Diskussionsforum
- Moderation des Diskussionsforums (Dozent)

Der Administrator hat weiterhin noch die Möglichkeit zu jeder Zeit die Daten im System zu archivieren.

Nutzer

Das System kennt 5 Rollen in denen ein Benutzer auftreten kann:

a) Administrator:
Installiert und deinstalliert die Anwendung; sichert Daten während und nach Ende des Übungsbetriebes

b) Dozent:
Verantwortlich für den Ablauf des Übungsbetriebes. Er veröffentlicht Übungsaufgaben und Dokumente, organisiert die Übungsgruppen, legt Termine fest und moderiert das Diskussionsforum.

c) Tutor:
Korrigiert und bewertet die Lösungen der Studenten.

d) Student:
Sendet Lösungen zu Übungsaufgaben ein und kann sich Korrekturergebnisse anschauen.

e) Gast:
Kann allgemeine Informationen abrufen und Diskussionsbeiträge lesen.

2.Paketstruktur

Aufgrund des geringen Umfanges wurde auf eine weitere Paketunterteilung verzichtet. Die Anwendung besteht nur mehr aus einem Paket.

3.Klassenstruktur

Es gab bisher nur ein Treffen zur Festlegung der Grobstruktur. Eine genauere Ausarbeitung oder gar Klassendiagramme lagen mir heute leider noch nicht vor. Daher kann ich die Klassenstruktur nur ganz grob vorstellen:

Servlet:
Ist der Einsprungpunkt über den der Servletcontainer HTTP-Anfragen weitergibt. Das Servlet extrahiert den nachgefragten Dienst und überprüft mit Hilfe der Authentifzierungs-Klasse die Berechtigung des Nutzers. Daraufhin wird die entsprechende Validate-Klasse aufgerufen und der Rückgabestream an den Webserver weitergeleitet.

Authentifzierung:
Diese Klasse stellt eine Schnittstelle zur Authentifzierung von Nutzern zur Verfügung. Die zugrundeliegenden Informationen bezieht sie aus der XML-Datei *users.xml*. Beim Start der Anwendung wird eine Authentifizierungs-Matrix erstellt mit den Dimensionen:

Navigations-Item x Rolle

Validate:
Die abstrakte Oberklasse, aus der alle funktionalen Klassen abgeleitet werden.
Die abgeleiteten Validate-Klassen sind mit einem Screen assoziiert, der den Benutzer zur Eingabe von Informationen oder zum Ausführen von Aktionen auffordert. Die zurückgelieferten Informationen werden auf ihre Validität überprüft und entsprechende Aktionen ausgeführt. Das Ergebnis der Aktion, bzw. die Fehlermeldung wird dann von der Screen-Klasse erzeugt.
Informationen über jeweils benötigten Felder lesen die Klassen aus einer Property-Datei ein.

Screen:
Die abstrakte Oberklasse aus der die Ein- und Ausgabe Bildschirme der jeweiligen Validate-Klassen abgeleitet werden.
Die abgeleiteten Screen-Klassen erzeugen HTML-Code der den Nutzer zur Eingabe von Daten auffordert oder eine Rückmeldung auf Aktionen bringt.

DB-Manipulation:
Ein Interface legt die Methoden fest, auf die Validate und Authorization Klassen zugreifen. Die Methoden sind dem SQL/92 Standard nachempfunden. Jede Klasse, die als Schnittstelle zum Datenbestand dienen soll, muss dieses Interface implementieren.
Im Rahmen unseres Projektes wird nur eine Schnittstelle zu einem XML-Datenbestand zur Verfügung gestellt.
Die Zugangsdaten werden in *users.xml* gespeichert. Zur Speicherung der Studentendaten wird eine Verzeichnisstruktur aufgebaut, die für jeden Studenten ein Unterverzeichnis vorsieht, worin die Korrekturergebnisse als XML-Datei in *results.xml* und die abgegebenen Lösungen als PDF-Datei

gespeichert werden.
Für den Dozenten wird ebenfalls ein Verzeichnis angelegt.

4.Funktionsweise

Über die Methoden *doGet()*, *doPost()* und *doPut()* werden HTTP/Get/Post/Put-Anfragen der Anwendung übergeben. Allerdings werden *doPost()* und *doPut()* Aufrufe einfach an *doGet()* weitergeleitet, so dass die Reaktion auf Anfragen zentral gesteuert wird.
Dort wird dann der genaue Dienst extrahiert. Ist der Benutzer zu dieser Anfrage berechtigt, was vom Servlet über die Authentifizierungs-Klasse geprüft wird, dann wird das entsprechende Validate Objekt erzeugt. Dieses wiederum instanziiert den zugehörigen Screen.
Authentifizierungs- und Validate-Klassen greifen über ein DB-Manipulations-Interface auf den Datenbestand zu.