Storyentwicklung und Testkonzept

1 Allgemeines

1.1 Storyplanung

Um die in den nächsten 5 Wochen anstehenden Implementierungsarbeiten möglichst effizient in der Gruppe und über den Zeitraum zu verteilen, wird die Entwicklung des Produktes in kleinere Aufgaben gegliedert.
Diese „Stories“ sollten möglichst unabhängig voneinander und in Gruppen von je 2 Personen (Pair Programming) implementier - und testbar sein. Da es leicht zu Verzögerungen durch plötzlich auftretende Probleme bei der Implementierung kommen kann, werden die Stories so verteilt, dass die gesamte Funktionalität bereits nach der 4. Woche verfügbar ist und die letzte Woche lediglich als Zeitpolster und zum Finden von Fehlern sowie kleineren Verbesserungen genutzt wird.

1.2 Testkonzept

Da bei der Entwicklung der einzelnen Stories eine ganze Reihe von logischen oder semantischen Fehlern auftreten können, ist es unumgänglich jede Story bereits während der Implementierung ausführlich zu testen. Dazu werden JUnit-Testsuiten verwendet, deren Inhalt bereits vorher möglichst genau definiert wird, um testgetriebenes Entwickeln zu ermöglichen.
Nach Abschluss der Arbeiten an den Stories wird außerdem ein Test des erweiterten Gesamtsystems nötig sein, um Fehler bei der Integration der Teile in den bestehenden Code und vorallem OLAT auszuschließen. Ein besonderer Aspekt beim Testen der Komponenten und des Gesamtsystems liegt auf der Threadsicherheit bezüglich eines Mehrbenutzerbetriebs.

2 Storyplanung und Testkonzepte

2.1 1.Woche(23.5.-30.5.):

- 1. Rechtevergabe über Administrator und Prüfungsamt
- 2. Rollenmanagement verwalten
- 3. Menutree und Menutreeverwaltung in Maincontoller
- 4. Tab in der Sitedefinition / Erstellen des Extension-Grundgerüsts

Innerhalb der ersten Woche wird das Extension-Grundgerüst für das Projekt geschaffen (3. & 4.), mittels dessen später die gesamte Funktionalität in Olat verfügbar gemacht wird. Zudem wird bereits versucht, das Rollenkonzept bereits soweit zu implementieren, dass die Unterscheidung der Rollen und ihrer verschiedenen Funktionen bereits im UserInterface sichtbar wird.

Testfälle:

- testen ob alle Events richtig geworfen werden und die Seiten richtig verlinkt sind
- korrekte Einbindung der Extension in Olat, die Funktionen sollten für Gäste nicht sichtbar sein

- die Rechtevergabe muss funktionieren und die Rolle eines Nutzers sollte jederzeit bekannt sein
- Test, ob auch nur die Funktionen im MenuTree/View angezeigt werden, die zu der jeweiligen Rolle gehören

2.1.1 2.Woche(31.5.-6.6.):

- 5. Ressourcenverwaltung für Termine und Prüfungen
- 6. XML-Persistierung der Daten
- 7. Editor um Prüfungen anzulegen / ändern für Prüfer
- 8. „Meine Prüfungen“ für Prüfer

Die gesamte Entwicklung der Persistenzschicht sollte in der 2. Woche ablaufen, da diese bereits für spätere Funktionen benötigt wird. Hierzu zählen also die Ressourcenverwaltung für Termine und Prüfungen und die Speicherung der dazugehörigen Daten in XML (5. und 6.). Es wird außerdem bereits ein Editor entickelt, der lediglich von Prüfern genutzt werden kann und das anlegen bzw. editieren von mündlichen/ schriftlichen Prüfungen ermöglicht. Zusammen mit der zu implementierenden „Meine Prüfungen“-Ansicht des Prüfers sollte somit ein erster Funktionszyklus zum anschauen und testen entstehen.(7. & 8.)

Testfälle:

- alle Tests aus der 1.Woche müssen weiterhin positiv sein
- die Zuordnung von Prüfungen/ Terminen zu Ressourcen/ Securitygroups muss funktionieren und eindeutig sein
- Fehler bei der Eingabe von Prüfungs oder Termindaten müssen ausgeschlossen oder abgefangen werden
- die Daten müssen korrekt in der XML persisitiert werden und auslesbar sein
- eine Ressource und die dazugehörige XML-Datei müssen zueinanderpassen, d.h. sie müssen konsistent gehalten werden
- Testen des Ressourcenmanagements anhand einiger Beispiele mittels des Editors und der „Meine Prüfungen“-Ansicht des Prüfers

2.1.2 3.Woche(7.6.-12.6.):

- 9. „Alle Prüfungen“-View
- 10. „Prüfungs-Details“-View
- 11. „Meine Prüfungen“-View für Studenten
- 12. An-/Abmeldung von Studenten zu einer schriftlichen / mündlichen Prüfung

Nachdem in den ersten zwei Wochen die Grundlagen geschaffen wurden, soll in der 3. Woche vor allem der größte Teil der Funktionalität implementiert werden. Darunter fallen also die Ansicht aller Prüfungen (mit Einschränkung der einzelnen Rollen, 9.), die Detailsansicht zu einer bestimmten Prüfung (10.) und der Prüfungsan- bzw. abmeldevorgang für Studenten (12.). All diese Funktionen müssen ausreichend auf Funktionstüchtigkeit, Fehlertoleranz und Mehrbenutzerbetrieb getestet werden.

Testfälle:

- alle Tests aus der 1.Woche müssen weiterhin positiv sein, das Ressourcenmanagement aus der 2.Woche muss weiterhin reibungslos funktionieren
- Einzeltests der implementierten Klassen/Funktionen anhand einiger konkreter Beispiele
- die Unterscheidung der Rollen innerhalb der Views muss korrekt funktionieren
- innerhalb eines Views müssen alle Daten richtig angezeigt werden
- Test, ob Änderungen der Daten durch ein View richtig gespeichert werden
- Test auf Threadsicherheit der implementierten Funktionen des Prüfungsmoduls

2.1.3 4.Woche(13.6.-19.6.):

- 13.Validierungsfunktion des Prüfungsamts
- 14. Freischaltung von Studenten die sich in eine schriftliche Pruefung eingeschrieben haben

Innerhalb der 4. Woche sollten die letzten Funktionen implementiert werden, hierzu zählt die Validierungsfunktion des Prüfungsamt (13.) und die Anmeldebestätigung für Studenten (14.)

Testfälle:

- sämtliche Funktionen aus den vorherigen Wochen müssen weiterhin korrekt laufen und alle Tests positiv sein
- Studenten können nicht gleichzeitig in allen BGs sein
- testen des Zusammenspiels der einzelnen Funktionen im Gesamtsystem

2.1.4 5.Woche(20.6.-25.6.):

- 15. ausführliches Testen aller Funktionen
- 16. Fehler finden
- 17. Design verbessern
- 18. Nutzerfreundlichkeit erhöhen

Um das ganze Projekt in einen auslieferbaren Zustand zu bringen, wird die gesamte 5. Woche für Verbesserungen jeglicher Art genutzt. Dazu zählt insbesondere die Suche nach Bugs, logischen Fehlern aber auch die Erhöhung der Nutzbarkeit durch eventuelle Anpassungen des äußeren Designs oder kleinere Anpassungen der Funktionen.

Testfälle:

- alle bisher gemachten Tests werden wiederholt und müssen weiterhin positiv sein
- ausgiebige Tests des gesamten Projetes in Bezug auf das richtige Zusammenspiel aller Teile, v.a. im Mehrbenutzerbetrieb
- Testen auf Benutzerfreundlichkeit
- Durchspielen einiger größerer Testszenarien

3 Verteilung der Stories auf die Projektgruppe

Als Grundlage der Planung stehen 4 jeweils wechselnde Zweierpaare, die jeweils eine Story übernehmen. Die folgende Planung sollte auf alle Fälle eine erste Richtlinie sein, kann sich aber u.U. den Gegebenheiten anpassen.

	Team 1		**Team 2**		**Team 3**		**Team 4**	
Woche 1	MC,SH	1	SK,DG	2	MH,SM	3	JF,ML	4
Woche 2	MC,ML	8	SK,SH	6	MH,DG	7	JF,SM	5
Woche 3	MC,SM	9	SK,ML	10	MH,SH	11	JF,DG	12
Woche 4	MC,DG	13	SK,SM	13	MH,ML	14	JF,SH	14
Woche 5	MC,SH	15	SK,DG	16	MH,SM	17	JF,ML	18