Communication Tool “Slack light”

Erstellen Sie eine Webanwendung für ein Communication-Tool (wie bspw.: http://www.slack.com). In Ihrer Anwendung können registrierte und angemeldete Benutzer miteinander kommunizieren, dazu können die Benutzer unterschiedliche Themen / Channels nutzen.

1 Funktionale Anforderungen

1.1 Posting

Die Postings werden zumindest mit folgenden Attributen erfasst:

- Channel / Thema (1 : n)
- Titel
- Beschreibung (Inhalt)
- Verfasser (Benutzer)
- ....

Nach erfolgreichem Login werden dem Benutzer seine Nachrichten in chronologischer Form, geordnet / kategorisiert nach Channels angezeigt. Sobald ein Benutzer eine Nachricht gelesen hat (= angezeigt bekommen hat), wird diese als gelesen markiert. Achten Sie auch darauf, ungelesene Nachrichten zu markieren. Zustätzlich soll es möglich sein, dass ein Benutzer gewisse Posts als “wichtig” markiert - diese erscheinen in der Auflistung gesondert markiert und werden an erster Stelle gereiht. Die Nachrichten können je nach Channel gefiltert werden, sodass die Kommunikation auch nur innerhalb eines Channels stattfinden kann.

Benutzer können nur eigene Nachrichten bearbeiten und löschen, aber nur dann, wenn noch keine Antworten bzw. nachfolgenden Posts auf den jeweiligen Eintrag existieren.

WICHTIG: innerhalb eines Channels / Themas können mehrere Benutzer miteinander kommunizieren - verschiedene Stati der Postings (gelesen / ungelesen / wichtig) gelten also je eingeloggtem Benutzer!

Sie können für Ihre Anwendung vordefinierte Channels nutzen, eine eigene Channel-Verwaltung ist nicht notwendig - es sollten jedoch zumindest 2 Channels existieren. Bei der Neuregistrierung kann der Benutzer auswählen, welche Channels er nutzen möchte.

1.2 Benutzer / Rollen

Standardmäßig existieren folgende Rollen:

1. **Anonyme Besucher**
   Können sich für die Anwendung durch Vergabe eines Benutzernamens und Passwort neu registrieren und einen oder mehreren Channels beitreten.
2. **Registrierte Nutzer**
   Nach erfolgter Registrierung kann die Kommunikation in den ausgewählten Channels stattfinden.

2 Non-funktionale Anforderungen

2.1 Programmierung Server

Umsetzung mittels PHP > 5.4 (OOP) – achten Sie auf einen sauberen Aufbau und eine Trennung der einzelnen Anwendungsschichten. Für die Anbindung der Datenquelle benutzen Sie die Bibliotheken *mysqli* oder *PDO*. Sichern Sie Ihre Anwendung gegen SQL-Injection und XSS-Angriffe durch serverseitige Prüfung der Eingaben und Parameter.
Bei Benutzeraktionen werden zusätzlich Daten zur Identifizierung mitgespeichert werden (bspw. IP-Adresse des Benutzers)

2.2 Datenbank

Die Datenverwaltung erfolgt mittels MySQL 5 (InnoDB). Achten Sie auf die Grundregeln des Datenbank-Designs (Datentypen, Normalisierung, sinnvolle Constraints,...). Hinweis: Datensätze, die von den Usern gelöscht wurden, sollten nicht direkt aus der DB gelöscht werden, sondern lediglich mit einem entsprechenden Flag gekennzeichnet werden.

2.3 Client / Frontend

Die Verwendung von Javascript (auch Frameworks, aber lediglich UI Frameworks wie bspw. JQuery UI) ist erlaubt. Der HTML-Code sollte je nach DOCTYPE (XHTML oder HTML5) valide sein, sämtliche Formatierungen via CSS umgesetzt werden. Die Benutzeroberfläche sollte intuitiv bedienbar sein und bei neuen Nutzern der Anwendung keinerlei Erklärungsbedarf erfordern. Achten Sie insbesondere auf Validierung von Eingaben sowie die Fehlerausgabe. Auch eine sinnvolle Verwendung von AJAX ist möglich, sofern diese für Sie sinnvoll erscheint (ist jedoch nicht zwingend Voraussetzung).
Hinweis: das „Design“ sollte einfach, aber funktional sein – wichtig ist die Bedienbarkeit und die logische Führung des Benutzers durch die Website (Navigationselemente, Feedback bei Benutzeraktionen, Formular-Validierungen & -Layout,...). Achten Sie insbesondere auf eine übersichtliche Darstellung, wenn die Threads länger werden.

3 Form der Abgabe

3.1 Dokumentation

Erstellen Sie für Ihr Datenbank-Modell ein ER- oder UML-Diagramm, welches die Entitäten und Beziehungen visualisiert. Für die Anwendung erstellen Sie bitte eine textuelle Beschreibung und / oder eine dokumentierte Skizze der Architektur.

3.2 Testfälle

Führen Sie ausführliche Tests Ihrer Anwendung durch und dokumentieren Sie diese. Testen Sie auch fehlerhafte Eingaben! Ihre Testfälle sollten auf jeden Fall folgende Zustände testen:

- Benutzerregistrierung (neu)
- Benutzerlogin (inkl. Fehlerüberprüfung)
- Darstellung des Nachrichten-Feeds inkl. Status & Favoriten
- Benutzer erfasst neuen Eintrag
- Benutzer editiert / löscht bestehenden Eintrag (nur, wenn noch keine Antwort)
- Benutzer markiert Beiträge als Favoriten

- Zweiter Benutzer kommuniziert mit erstem Benutzer und führt ebenfalls obige Schritte durch

Erstellen Sie die Dokumentation und die Testfälle in Form eines PDF-Dokuments.

3.3 Anwendung

- **Datenbank**
  SQL – Dump inkl. Testdaten und CREATE DATABASE statement

  Datenbankname: **fh_2015_scm4_[IHRE_MATRIKELNUMMER]**
  Benutzername: **fh_2015_scm4**
  Passwort: **fh_2015_scm4**

- **Code**
  Zip-Archiv, Startdokument = index.php, muss auf Standard – XAMPP Installation lauffähig sein. Die Dokumentation speichern Sie bitte im Unterverzeichnis „/doc“.

3.4 Deadline

Laden Sie Ihre Anwendung als **eine Datei im ZIP-Format** in der entsprechenden Abgabe via Moodle hoch. Den konkreten Termin entnehmen Sie bitte dem Abgabe-Modul in Moodle.

3.5 Beurteilung

Die Beurteilung der Übung erfolgt im Rahmen einer Präsentation vor der eigenen Gruppe. Bei dieser Präsentation stellen Sie bitte Ihre Lösung vor, demonstrieren die Testfälle und stehen für technische und inhaltliche Fragen bzw. Code-Reviews zur Verfügung.

Der Termin wird, wie in der LVA bereits angesprochen, von Ihrem Studiengangsprecher mit der Administration abgeklärt und findet am Beginn des WS 2015 statt. Der genaue Termin wird noch kommuniziert.