Universität Leipzig

Softwaretechnik-Praktikum

Entwurfsbeschreibung Gruppe cz17a - Gamification

Lisa Vogelsberg, Felix Fink, Michael Fritz, Thomas Gerbert, Steven Lehmann, Fabian Ziegner, Willy Steinbach, Christian Schlecht

supervised by
Dr. Christian Zinke, Julia Friedrich, Christian Frommert

22. Januar 2018

Inhaltsverzeichnis

1 Allgemeines

Dieses Projekt setzt sich mit der Entwicklung eines gamifizierten Wissenquiz auseinander, welches in Form von betrieblicher Weiterbildung zum Einsatz kommen soll. Es steht also das Aneignen neuen Wissens im Vordergrund, jedoch soll der als "Last" empfundene Lernprozess durch Gamification so unterdrückt werden, dass freiwillig auf diese Methode zurückgegriffen wird. Somit soll durch das Spielen des Quiz der Lernprozess Spaß bereiten. Das Quiz wird als App für Android-Endgeräte verfügbar gemacht.

2 Produktübersicht

Die Quiz-App wird für mobile Android - Endgeräte entwickelt. Ein Spieler loggt sich unter Angabe von Nutzernamen und Passwort ein und kann dann entweder sein Dashboard oder sein Spielerprofil ansehen oder ein Spiel starten. Wenn er ein Spiel startet, wird ihm

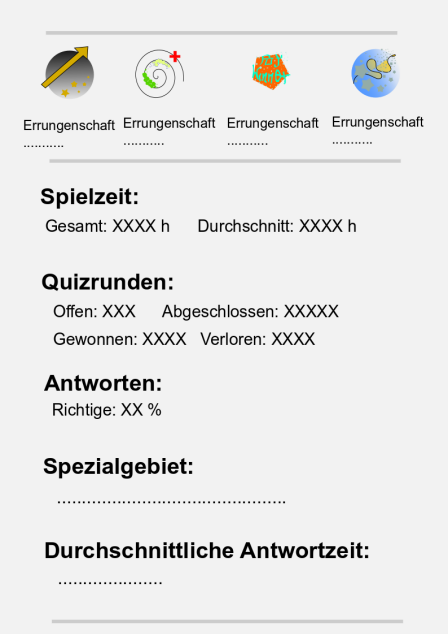

Abbildung 2.1: Dashboardansicht für den User

eine Übersicht über verfügbare Themen gegeben, von denen er eine auswählt, um in die Spiel-Lobby zu gelangen. Sobald mindestens fünf Spieler in der Lobby sind, startet das Spiel. Dabei erfolgt das Spielen anonym, das heißt die Spieler wissen untereinander nicht, gegen wen sie antreten. Es werden darauffolgend die Fragen und Auswahlmöglichkeiten dargestellt, von denen der Spieler eine Antwort auswählt und eine Rückmeldung darüber erhält, ob seine Antwort richtig oder Falsch war. Ggf. wird die richtige Antwort angezeigt. In den Abbildungen 2.1, 2.2 und 2.3 sind beispielhafte Ansichten dargestellt.
Zusätzlich wird eine Administrationsoberfläche in Form einer Webanwendung geben, in welcher der Administrator Fragen bearbeiten und alle relevanten Einstellungen durchführen kann. Die Grafiken werden bei Implementierung durch Screenshots der echten User Interfaces ersetzt.

3 Grundsätzliche Struktur - und Entwurfsprinzipien

Die gewählte Architektur ist eine Server-Client-Architektur. Sie ist unter Abbildung 3.1 schematisch dargestellt. Die erste Schicht bildet die PostgreSQL Datenbank inkl. Datenbankserver. Die Middleware besteht aus einem Server, welcher die Spiellogik händelt

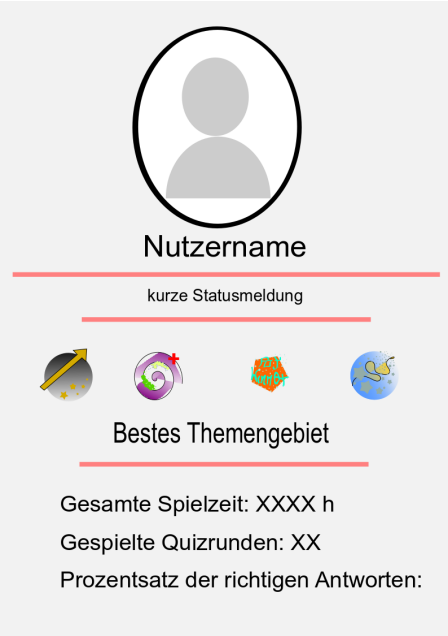

Abbildung 2.2: Profilansicht für den User

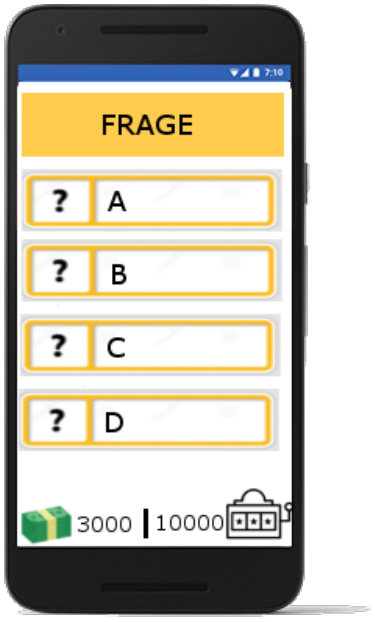

Abbildung 2.3: Fragenansicht für den User

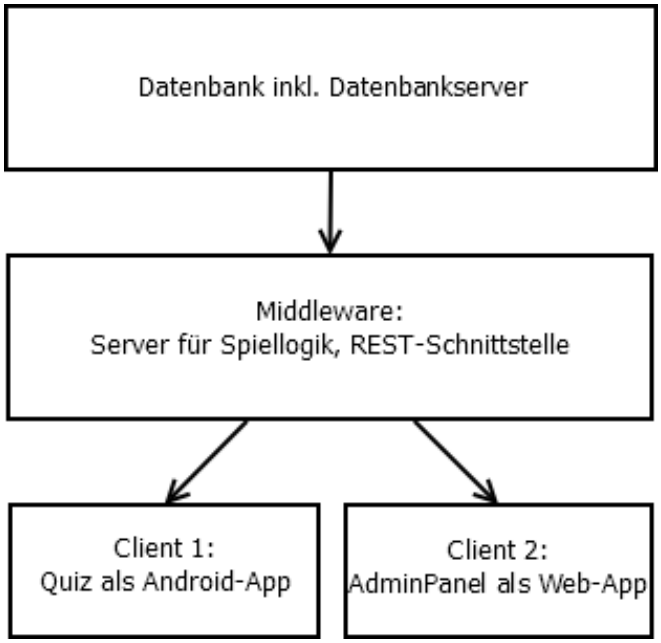

Abbildung 3.1: Architektur der Anwendung

und einer REST-Schnittstelle. Der Client, also die eigentliche App und die Administrationsoberfläche, kommuniziert ausschließlich über REST mit Server und Datenbank. Der Server hingegen kann über ein objektrelationales Datenbankmapping mittels Hibernate direkt auf die Datenbank zugreifen. Dazu werden spezielle Klassen (DataAccessObjects, DAO) verwendet.

4 Struktur - und Entwurfsprinzipien einzelner Pakete

4.1 Arbeitspaket 1 - Vorprojekt

4.1.1 Server

Das UML Klassendiagramm ist zu groß, um es innerhalb dieses Dokuments leserlich einzubinden und ist daher separat verfügbar.
Der Server übernimmt die zentrale Verarbeitung aller Vorgänge im Quiz.

REST-Schnittstelle

Der Server verfügt über eine REST-Schnittstelle zur Kommunikation mit dem Client. Es steht diverse GET- und POST-Requests zur Verfügung. Es können User, Fragen und Antworten, Runden, verfügbare Quiz' und eine Anfrage des Clients, ein Spiel zu starten, angefragt werden. Die REST-Schnittstelle ist unter der Webseite des Praktikums [1] mit einer Übersicht über verfügbare Anfragen einsehbar. Die Clients kommunizieren ausschließlich über diesen Webserver.

Game - Server

Den zweiten Teil des Servers bildet die Einheit zum Handling der Spiellogik. Da im Vorprojekt noch keine solche Logik existiert, werden bereits implementierte Funktionen noch nicht verwendet, jedoch für die nächsten Arbeitspakete benötigt. Da der Game-Server ohne REST auf die Daten zugreifen kann, wurden um Übersicht zu wahren, Data-Access-Object-Klassen eingerichtet. Diese agieren mittles Hibernate auf den Daten. Andere Klassen sollen nicht selbst auf die Datenbank zugreifen können, da sonst schwer behebbare Fehler und unübersichtliche Zugriffsstrukturen entstehen.

4.1.2 Client - Quizapp

Der Client stellt per REST eine Anfrage an den Server, um eine Quizfrage zu bekommen. Diese enthält auch Informationen über die Antwortmöglichkeiten und welche richtig ist. Um Datenredundanz zu vermeiden, ist in der Liste der Antworten stets die erste Antwort die Richtige. Bei Anzeige der Möglichkeiten werden die Antworten durchgemischt, um nicht auch hier stets die erste Antwortmöglichkeit als die Richtige identifizieren zu könnnen. Nach Auswahl durch den User erscheint eine Rückmeldung, ob die gegebene Antwort richtig oder falsch ist und ggf. wird die korrekte Antwort angezeigt. Es werden noch keine Speichervorgänge für die Statistiken durchgeführt.

4.1.3 Client - AdminPanel

Das AdminPanel ist eine separate Web Applikation. In diese kann sich der Administrator über Nutzername und Passwort einloggen. Dort sind alle verfügbaren Einstellungen vorzunehmen (spätere Arbeitspakete) und es können die Fragen in die Datenbank importiert werden. Das AdminPanel wurde logisch als Einheit getrennt, um bereits eine Art Access Control zu haben. Die Spiele-Clients haben somit generell keine Möglichkeit des Zugriffs auf Daten und Funktionen, welche nicht für sie vorgesehen sind.

[1] `http://pcai042.informatik.uni-leipzig.de:1810/restserver`

5 Datenmodell

6 Glossar