Dokumentation IBF-Tool

# Vorwort

Diese Dokumentation soll eine Hilfestellung bei der Weiterentwicklung oder Wartung des Tools geben, sowie die grundlegende Funktionsweise des Programms verdeutlichen.

# Installation und Setup

Das Programm ist mithilfe des Zend Frameworks komplett in PHP, HTML und JavaScript geschrieben. Es verwendet teilweise jQuery für Clientseitige Berechnungen und dynamische Events in Experimenten.

## Voraussetzungen

Es braucht relativ wenig um das Tool lauffähig zu kriegen.

* Apache oder LighTTPD Server
* MySQL Datenbank

### Apache-Konfiguration

Folgende Schritte sind nötig um das Tool auf Apache lauffähig zu machen:

* Aktivierung von mod\_rewrite
* Editieren der httpd\_conf:

/ibftool "<PATHOFSERVER>/ibftool/public"

<Directory "<PATHOFSERVER>/ibftool/public">

AllowOverride All

Order Deny,Allow

Allow from all

</Directory>

Natürlich lässt sich das Tool auch mittels vhost konfigurieren.

### LighTTPD-Konfiguration

Folgende Schritte sind nötig um das Tool auf LighTTPD lauffähig zu machen:

* Aktivierung von mod\_redirect und mod\_setenv
* Folgende Linie ans Ende der lighttpd.conf hinzufügen:

$HTTP["host"] == "ibftool" { server.document-root = "PATH\_TO\_IBFTool/public" setenv.add-environment = ( "APPLICATION\_ENV" => "development" ) url.rewrite-once = ( ".\*\.(js|ico|gif|jpg|png|css|html?|eot|docx?|xlsx?|svg).\*$" => "$0", "^/.\*(\?.\*)" => "/index.php$1", "" => "/index.php" ) }

Der Hostname sollte natürlich entsprechend konfiguriert werden (IP oder fixer Hostnamen mit Umleitung in der hosts-Datei.

## Datenbankkonfiguration

Die Konfiguration der Datenbank wird in der Datei unter application/config/db\_config.ini vorgenommen. Je nach Environment lassen sich unterschiedliche Parameter eintragen.

Das ganze DB-Modell wurde mittels der MySQL Workbench erstellt und benutzt InnoDB als relationales Schema, ist zudem komplett UTF-8.

# Allgemeine Funktionsweise

Das ganze Tool hat einen zusammenhängenden Aufbau. Experimente sind grundsätzlich als Treatments anzusehen. Jedes Treatment kann dabei aus einer Anzahl von verschiedenen Modulen bestehen.

Diese Module entsprechen den jeweiligen Controllerklassen im Code (Modul Questionnaire 🡪 QuestionnaireController).

Beim Start eines Treatments werden die dazugehörigen Configs aus der Datenbank geladen und die Module konfiguriert. Jedes Modul hat dabei seine eigene Konfiguration, in welcher z.B. die Seiten-ID (Static-Module für statische Seiten) bzw. die ID der Fragebogen-Seiten eingetragen werden (Questionnaire). Wiederum andere Module erhalten dadurch nur Konfigurationsparameter (Investmentgame) zu Anfangsvermögen oder der Anzahl Runden.

Jedem Treatment können User zugeordnet werden. Diese können sowohl händisch in der DB erfasst werden, es gibt aber auch einen Registrierungsmechanismus mit E-Mail Validierung für den Betrieb im Web (RegistrationController).

# MVC

Das gesamte Projekt implementiert das Model-View-Controller Pattern. Logik und GUI sind somit strikt getrennt. Models sind eng an die Datenbank geknüpft, Controller verknüpfen die Views (in HTML) mit den Models und sorgen für die Applikationslogik.

Modulbeschrieb

# Questionnaire

Das Questionnaire-Modul wird verwendet um Fragebögen zu erstellen. Es gibt diverse verschiedene Arten von erstellbaren Fragen (SingleChoice, MultipleChoice, Experimentspezifische Fragen wie „RS“- … ).

Unter „library/ibftool/Form/Element“ lassen sich alle dazugehörigen Zend\_Form\_Element-Klassen finden.