Anleitung: Baqend im Praktikum

`Aktuelle Version dieses Dokuments: baqend.com/files/praktikum.pdf`

Bitte folgende Schritte ausführen, um das Baqend Tutorial lokal auf den **Pool-Rechnern** auszuführen und weiterzuentwickeln:

1. Unter Windows mit Nutzerkennung anmelden
2. Falls nichts vorhanden Git Portable installieren: https://git-scm.com/download/win
3. Freigabe \\bq0\Shared im Explorer öffnen und *node.zip* in den Home Ordner *(%USERPROFILE%*) entpacken
4. Skript *env.bat* (auch in der Freigabe) ausführen
5. Neue Variable "PATH", Wert: %PATH%;%USERPROFILE%\node
6. Git Bash öffnen und *git clone https://github.com/Baqend/todo.git* ausführen
7. Kommandozeile (*cmd*) öffnen und Befehl *npm install* aufrufen
8. In Kommandozeile *grunt* ausführen, um einen Entwicklungsserver zu starten

Falls ihr das Projekt **lokal auf eurem Rechner** aufsetzt, ist folgendes notwendig:

1. Git installieren: https://git-scm.com/
2. Node.js & Npm installieren: https://nodejs.org/
3. Grunt global installieren: *npm install –g grunt-cli*
4. *git clone https://github.com/Baqend/todo.git*
5. Im Projekt: *npm install* und anschließend *grunt* ausführen (Entwicklungsserver)

Wenn Ihr auch einen lokalen Baqend Server installieren wollt: http://www.baqend.com/#download

Wenn ihr IntelliJ benutzen wollt, sprecht uns an, wir haben Lizenzen für euch.

Wichtige Links & Referenzen

Was	**Wo**
Baqend Guide (How To)	http://www.baqend.com/guide/
JavaScript API	http://www.baqend.com/js-sdk/latest/baqend.html
Baqend Tutorial	http://www.baqend.com/#tutorial

Aufgaben & Tutorials zur Einarbeitung (Tag 1-3)

Hier ein paar Vorschläge, um sich in Baqend und die Standardtechniken der Web-Entwicklung einzuarbeiten.

Web-Development Basics:

- JavaScript Kurs (falls nötig): https://www.codecademy.com/en/tracks/javascript
- Handlebars (Template Engine) Tutorial: http://handlebarsjs.com/

- JQuery Tutorial: http://www.w3schools.com/jquery/
- CSS Tutorial: http://www.w3schools.com/css/default.asp
    - Vertiefung: Less (http://lesscss.org/) oder Sass (http://sass-lang.com/)
- JavaScript Promises: http://www.html5rocks.com/en/tutorials/es6/promises/?redirect_from_locale=de
- Bootstrap Tutorial: http://www.w3schools.com/bootstrap/ Docs: http://getbootstrap.com/getting-started/

Gelesenes & gelerntes am besten direkt anwenden und Todo-Listen App weiterentwickeln (optional ist auch direkt eine eigene Anwendung möglich):

Zum Nachschlagen immer offen haben: http://www.baqend.com/guide/

Dashboard:

- **Ziel**: Tutorial von gemeinsamen Baqend Server auf eigene Instanz migrieren
- Schema.json aus dem Git Projekt öffnen und Inhalt kopieren
- In eurem Dashboard einloggen
- API Explorer öffnen > Schema Sektion > POST Schema > In Body Feld kopierten Inhalt einfügen & ausführen > F5 (Browser Refresh)
- Zurück zu Dashboard wechseln und Todo Klasse öffnen > Im Data Tab ein neues Todo einfügen
- In Schema-Tab wechseln
    - Index auf listId anlegen
    - Neues Feld hinzufügen, z.B. „extras“ von Typ String
    - F5 (Browser Refresh)
- Handler-Tab > onDelete: JavaScript Handler schreiben, der Löschen nur erlauben, wenn Todo-Item als erledigt markiert ist
- In Data-Tab wechseln > mit F12 Browser-Entwicklungs-Tools öffnen:
    - *DB.Todo.load('057b….[Objekt-ID]').then(function(obj) { console.log(obj); });*
    - Statt log nun das Objekt löschen -> testen ob Handler funktioniert
- Neue Baqend Method „helloworld“ anlegen
    - In Methodenrumpf einfügen: *return { message : "hello world" };*
    - Entwicklerkonsole: *DB.modules.get('helloworld').then(function(response) { console.log(response); });* um Methode zu überprüfen

Selbstständige Weiterentwicklung:

- Registrierung und Log-In für Todo-Listen-Benutzer
- Listenverwaltung: mehrere Listen pro User (Erweiterung des User Schemas)
- Suchfunktion über Query API realisieren
    - Primitiv: Regex-Queries
    - Attributbasiert: über Tags, Datum, etc.
    - Type-Ahead: Echtzeit-Vorschläge während Tippen („anchored Regex-Queries“
- Like-Buttons für Todos
    - Verschiedene Realisierungsmöglichkeiten; Invariante: nur ein Like pro User pro Todo
    - Eine Möglichkeit: Baqend-Method die Counter inkrementiert und Likes als Set über User-Referenzen speichert
- Metadaten für Listen (z.B. Titel, Tags, Description)

- Todos mit Nutzerzuordnung (wer erledigt was)
- Private Listen die mit anderen Usern geteilt werden können (über Access Control Lists)
- OAuth Login & Social Media Integration

Vertiefungen:

- Komplexere Template-Engine oder MVC Framework als Ersatz für Handlebars & JQuery:
    - Angular.js: https://angularjs.org/
      Umfangreiches Tutorial: https://www.codeschool.com/courses/shaping-up-with-angular-js
    - React.js http://facebook.github.io/react/
    - Ember.js http://emberjs.com/
    - Thymeleaf: http://www.thymeleaf.org/
- Statt Webanwendung hybride App:
    - Ionic Framework: http://docs.ionic.io/
        - Basiert auf Angular.js: https://angularjs.org/
        - Und Cordova: https://cordova.apache.org/
    - Cordova & PhoneGap
        - https://cordova.apache.org/ http://phonegap.com/

Eigene Anwendung (Tag 4-15)

Ideen:

- Blog
- Twitter- oder Facebook-Klon
- Online-Shop / Ecommerce
- Online Game
- Uber-Klon
- Kalender
- QA-Platform à la Quora oder Stackoverflow
- Notizen-App à la Evernote
- Meetup/Verabredungs-App
- Preisvergleichsplattform
- Auktionsplattform
- Splitwise-Klon