Anleitung: Baqend im Praktikum

Aktuelle Version dieses Dokuments: *duvs2017.app.baqend.com/praktikum.pdf*

Wichtige Links & Referenzen

Was	Wo
Dashboard	https://dashboard.baqend.com/register
Baqend Tutorial	https://www.baqend.com/tutorial.html
Allgemeine Infos & weitere Links	https://www.baqend.com
Baqend Guide (How To)	https://www.baqend.com/guide/
JavaScript API	https://www.baqend.com/js-sdk/latest/baqend.html

Aufgaben & Tutorials zur Einarbeitung (Tag 1-3)

Allgemeines:

- Ihr könnt nach Belieben entweder **eigene Rechner** oder die **Pool-Rechner** verwenden. **Hinweis:** Sollte das Baqend CLI (z.B. `baqend login`) am Pool-Rechner nicht funktionieren, könnt ihr es wie folgt aufrufen: `npm run baqend …`)
- Wenn ihr euren eigenen Rechner verwendet, stellt sicher, dass ihr Node 4 mit NPM und Git installiert habt.
- Wenn ihr Hilfe braucht, holt uns einfach aus Raum F-528 oder schreibt uns in dem **On-Site-Chat**. Entweder können wir Eure Frage direkt beantworten oder wir kommen rüber.
- Wenn ihr Bugs findet oder Features vermisst, lasst es uns gerne (auch über den Chat) wissen, wir freuen uns darüber!

Umgang mit Baqend lernen:

1. Interaktives **Tutorial** durcharbeiten, um die API und das Backend-as-a-Service-Konzept zu lernen: http://www.baqend.com/tutorial.html

Legt euch nun pro Team eine App an unter https://dashboard.baqend.com/register . Unter *Collaboration* könnt ihr die anderen Team-Mitglieder im Dashboard als Entwickler hinzufügen. Die erhalten jeweils eine Email und haben dann auch vollen Zugriff auf App und Dashboard.

2. Geht den *Quickstart* im Dashboard durch, um die Funktionen am Beispiel einer zweiten Anwendung konkret kennenzulernen.
3. Setzt das Todo-Projekt (https://github.com/Baqend/todo.git) auf der gleichen Instanz auf. Klont euch dazu zunächst das git-Projekt, installiert seine Node-Module lokal und meldet euch an:

```
git clone https://github.com/Baqend/todo.git
cd todo
npm install
```

`npm run login`

Schaut euch auch die README unter https://github.com/Baqend/todo/blob/master/README.md an, sie enthält alle erforderlichen Kommandos.

4. Übertragt das Schema aus dem lokalen Projekt in eure Cloud-Instanz:
   a. mit CLI:
      `npm run schema upload <app-name>`
   b. oder alternativ über das Dashboard:
      i. Inhalt von Activity.json und Todo.json im Ordner baqend/schema kopieren
      ii. für jede dieser Klassen: Im Dashboard den API Explorer öffnen > Schema Sektion > POST Schema > In Body Feld kopierten Inhalt einfügen, den Klassennamen in Bucket einfügen & ausführen > F5 (Browser Refresh)
5. Öffnet die app.js und ändert in Zeile 156 den App-Namen von „toodle“ zu eurem App-Namen.
6. weitere Aufgaben – **Ziel**: Tutorial auf eure Instanz migrieren und **kennenlernen**
   - Todo unter Data öffnen > Im Data Tab ein neues Todo einfügen
   - Hosting nutzen: Inhalt des dist Folders unter Files > www mit Drag & Drop reinziehen → Anwendung ist nun unter <euer-app-name>.app.baqend.com aufrubar
   - In Schema-Tab wechseln
     - Index auf listId anlegen
     - Neues Feld hinzufügen, z.B. „extras“ von Typ String
   - Handler-Tab > onInsert: JavaScript Handler schreiben, der Anlegen nur erlaubt, wenn Todo-Item eine listId hat
     (siehe https://www.baqend.com/guide/topics/baqend-code/#aborting-requests)
   - In Data-Tab wechseln > mit F12 Browser-Entwicklungs-Tools öffnen:
     - *DB.Todo.load('057b….[Objekt-ID]').then(function(obj) { console.log(obj); });*
     - Im Guide (http://www.baqend.com/guide/) nachsehen, wie man ein Objekt einfügt → prüfen, ob insert Handler funktioniert
   - Neue Baqend Method „helloworld“ anlegen
     - In Methodenrumpf einfügen: *return { message : "hello world" };*
     - Entwicklerkonsole: *DB.modules.get('helloworld').then(console.log);* um Methode zu überprüfen
   - *Optionale* Vertiefung: Todo-Projekt mit Git auschecken und lokal mit grunt (ein JavaScript Task Runner à la Ant/Maven/Gradle) weiterentwickeln:
     - Git installieren: https://git-scm.com/
     - Node.js & Npm installieren: https://nodejs.org/
     - Grunt global installieren: *npm install –g grunt-cli*
     - *git clone https://github.com/Baqend/todo.git*
     - Im Projekt: *npm install* und anschließend *grunt* ausführen (Entwicklungsserver)

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

Web-Development Basics:

- JavaScript-Kurs (falls nötig): https://www.codecademy.com/en/tracks/javascript
- ES6 (neuer JS Standard) Neuerungen: https://github.com/lukehoban/es6features
- 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/
- Wenn ihr **IntelliJ** benutzen wollt, nutzt eine Studentenlizenz:
  https://www.jetbrains.com/student/.
  Bei Problemen sprecht uns an.

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
  - Angular2: https://angular.io/
    Video-Tutorial: https://egghead.io/courses/angular-2-fundamentals
  - React.js http://facebook.github.io/react/
  - Ember.js http://emberjs.com/
  - Aurelia: http://aurelia.io/
  - Vue.js: https://vuejs.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/

Für viele Themen haben wir Spezialliteratur, sprecht uns an, wenn ihr besondere Technologien einsetzt und nach Büchern, etc. sucht.

Das Projekt aufsetzen

Nun könnt ihr euer eigentliches Projekt aufsetzen:

- Für diejenigen, die Cutting-Edge-Technologien probieren wollen: Baqend + Angular 2 Starter Kit als Basis nehmen https://www.baqend.com/guide/starter-kits/
- Wir empfehlen mit Git über Github zu arbeiten, um euren Code im Team zu teilen und zu versionieren.
- Unter http://www.initializr.com/ könnt ihr euch ein Grundprojekt mit ausgewählten Libraries generieren lassen. Wir empfehlen Twitter Bootstrap für Design und Layout zu verwenden (kann dort einfach ausgewählt werden). Ein alternativer Frontend Generator ist Yeoman http://yeoman.io/

Um eure Anwendung auf Baqend zu deployen, müsste ihr zunächst das Baqend SDK installieren:

```
npm install -g baqend
```

Anschließend könnt ihr euch mit euren Zugangsdaten anmelden:

```
baqend login
```

Eure Files und euern Baqend-Code ladet ihr mit folgendem Befehl hoch. Dabei ist zu beachten, dass die Dateien aus dem Ordner 'www' und der Baqend-Code aus 'code' verwendet wird. Die Baqend-

Module heißen wie eure JavaScript-Dateien. Die Handler (insert, update, delete, validate) werden in Unterordner für jede Klasse gepflegt:

```
baqend deploy --app appName
```

Eure Orderstruktur ist also z.B.:

```
Code
-- module.js
-- user
-- -- insert.js
www
-- index.html
-- andere Files, z.B. Bootstrap
```

Eigene Anwendung (Tag 4-15)

Zum Beispiel kollaborative Anwendung mit **Echtzeitsuche**, siehe auch:

- Twoogle:
  - Webseite: https://twoogle.info/
  - Codepen: https://codepen.io/baqend/pen/BWbyVY
- Baqend Realtime Queries:
  - Announcement: https://medium.com/p/3a44a13fde86/
  - Doku: https://www.baqend.com/guide/topics/realtime/

weitere 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

Oder Weiterentwicklung der Todo-App:

- 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“)
- Sich selbst aktualisierende Listen durch Real-Time Queries
- Like-Buttons für Todos
  - Verschiedene Realisierungsmöglichkeiten; Invariante: nur ein Like pro User pro Todo

    - Eine Möglichkeit: Baqend Code der 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