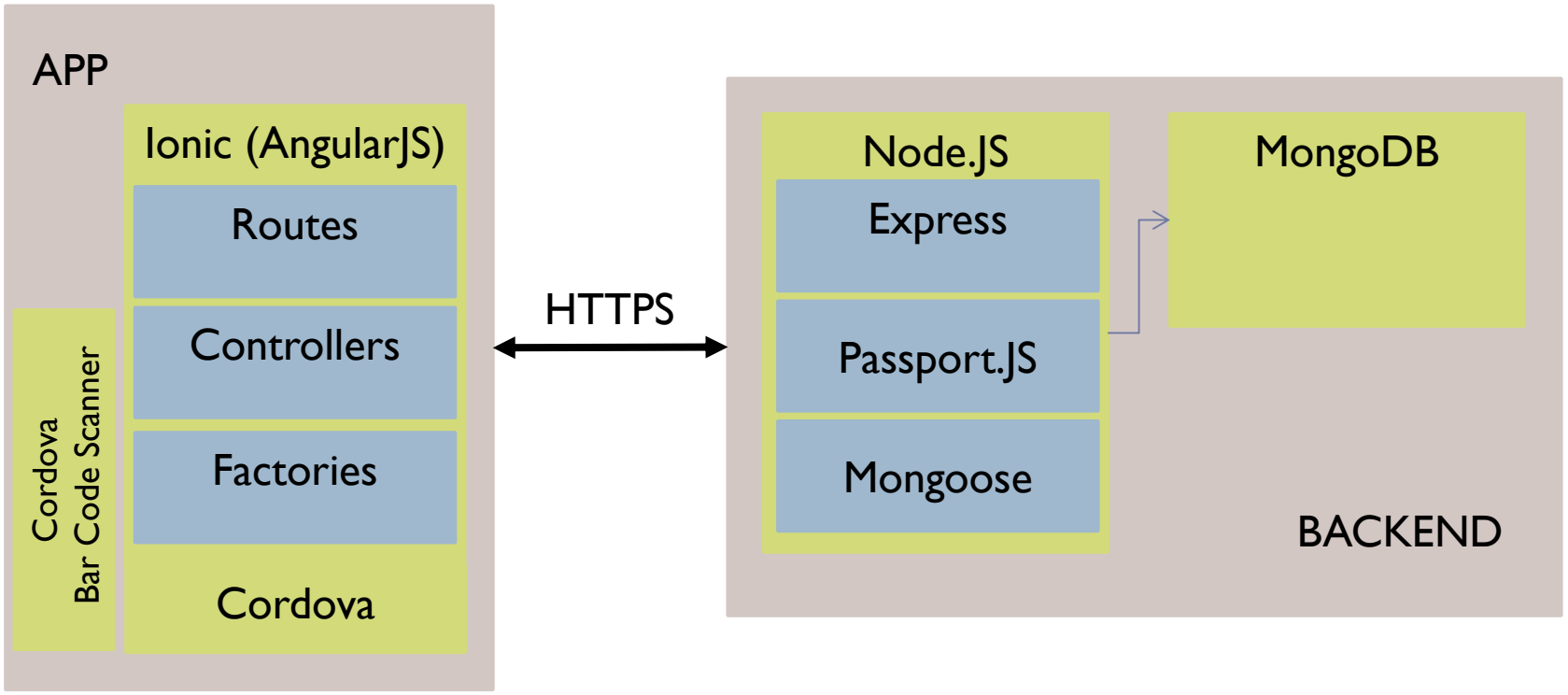
Architektur – Einkaufsapp

Sebastian Kiepsch, 12.12.2015

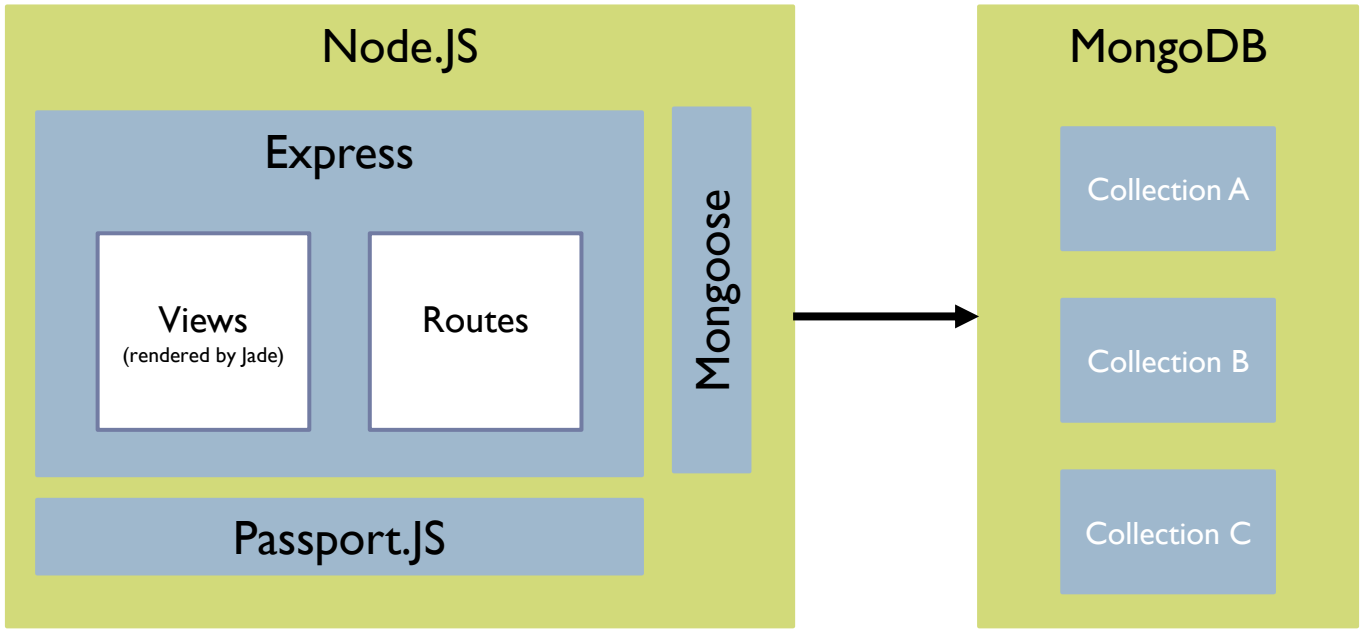
Agenda

1. Architektur Übersicht
2. Architektur Backend
    1. Übersicht
    2. Begründung der Komponentenauswahl
    3. Details
    4. Zusammenhang MVC - Backend
    5. Datenbank
    6. Hosting
3. Architektur Hybrid App
    1. Übersicht
    2. Begründung der Komponentenauswahl
    3. Details
    4. Zusammenhang MVC – App
    5. Tools
4. Quellen

1. Architektur - Übersicht



2.1 Architektur Backend - Übersicht



Begriffe

- Express: Webframework für Node.JS
- Jade: Rendering Engine für Views – generiert HTML
- Passport.JS: Modul für die Authentifizierung (Login, Registrierung)
- Mongoose: Objektmodellierungstool für Node.JS – Übernimmt Datenbankanbindung sowie damit verbundene Businesslogik

2.2 Architektur Backend – Begründung der Komponentenauswahl

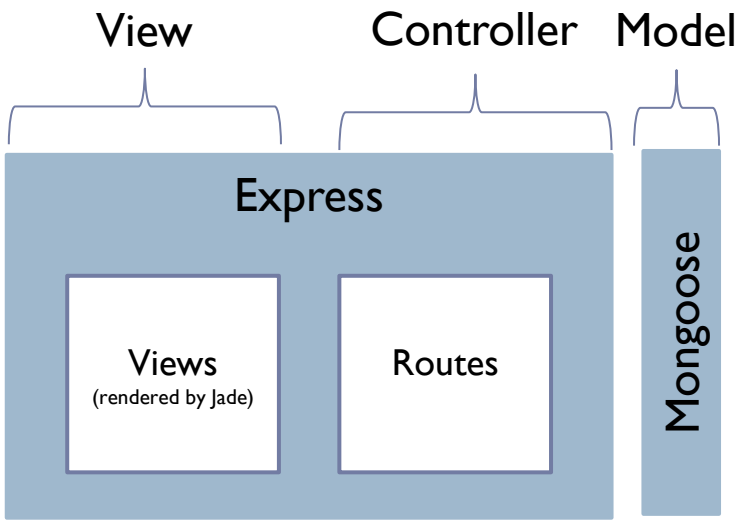
- Node.JS
  - Neutrale Sprache – läuft auf allen OS
  - Hoher throughput und Skalierbarkeit
  - Perfekt geeignet für Anwendung mit viel IO wie WebAPIs ohne großen Rechenaufwand
- Express Webframework
  - „Erwachsendstes“ und flexibelstes Webframework für Node.JS
  - Erweiterbar durch große Auswahl an verfügbarer Middleware (wie z.B. bei uns mit Passport.JS)
- Passport.JS
  - Riesige Auswahl an Authentification Providern (300+ Authentifizierungsstrategien)
- MongoDB
  - Flexibilität: Datenmodell schnell und flexibel änderbar (Stichwort „iterative Entwicklung“)
  - Hohe Scaliberbarkeit (auto-sharding)
  - Einfache Implemtierung mit Node.JS über Mongoose

2.3 Architektur Backend - Details

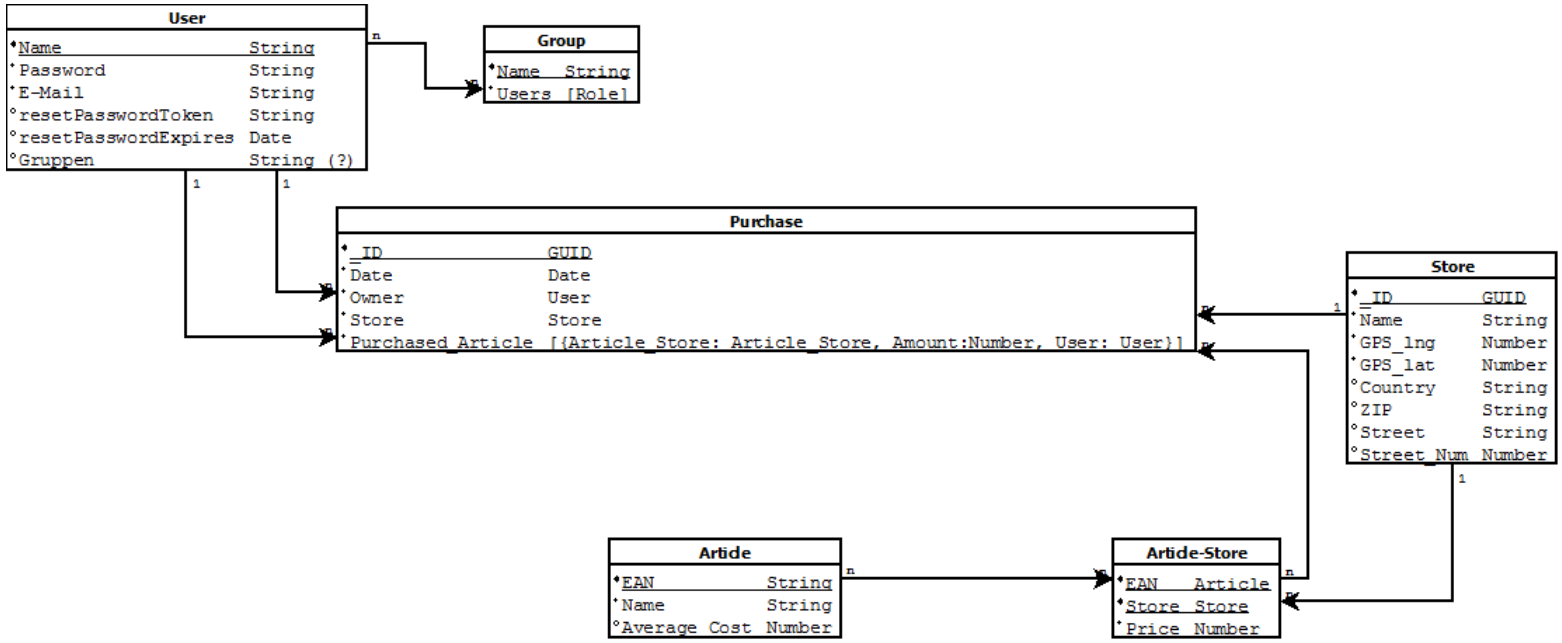
- basiert auf Node.JS, eine Runtime-Umgebung für JavaScript.
- Node.JS ermöglicht es JavaScript Applikationen zu entwickeln → können als Desktop bzw. Kommandozeilenapplikationen entwickelt werden
- Node.JS hat eigenen Packetmanager (npm) → viele fertige Module für eigene Anwendungen, u.a. auch für Serveranwendungen
- Bei unserem Projekt:
  - Express.JS als Webframework
    →stellt Schnittstellen sowie Programmierparadigmen für die Entwicklung von Webanwendungen und APIs bereit
    →Routen: Definition von Endpunkten sowie der Businesslogik für deren Behandlung
    →Views: Stellen überreichte Inhalte aus Routen dar (gerendert via Engine, in unserem Fall „Jade“)
  - Passport.JS als Authentifizierungsprovider
    →stellt Tools für die Authentifizierung von Benutzern in Express bereit
    →ermöglicht verschiedene Authentifizierungsprovider und ist damit erweiterbar und flexibel
    →wir verwenden die „lokale Authentifizierung“, d.h. den Abgleich von Username und Passwort mit den Werten einer lokalen DB (Mongo DB)
  - Mongoose als DB Modelling Tool
    →stellt Tools für die Datenbankmodellierung in MongoDB sowie ihren Zugriff bereit
    →definierte Modelle erstellen automatisch Collections in der DB
    →Zugriff über Mongoose eigene Funktionen wie z.B. find, create etc.
- MongoDB: Verwendete Datenbank
  - NoSQL Datenbank
  - Collections statt Tabellen

2.4 Zusammenhang MVC - Backend

- Model – View – Controller liegt adaptiert vor
- Mongoose schafft Datenmodelle, Controller greifen darauf zu und bestimmen was im View dargestellt wird



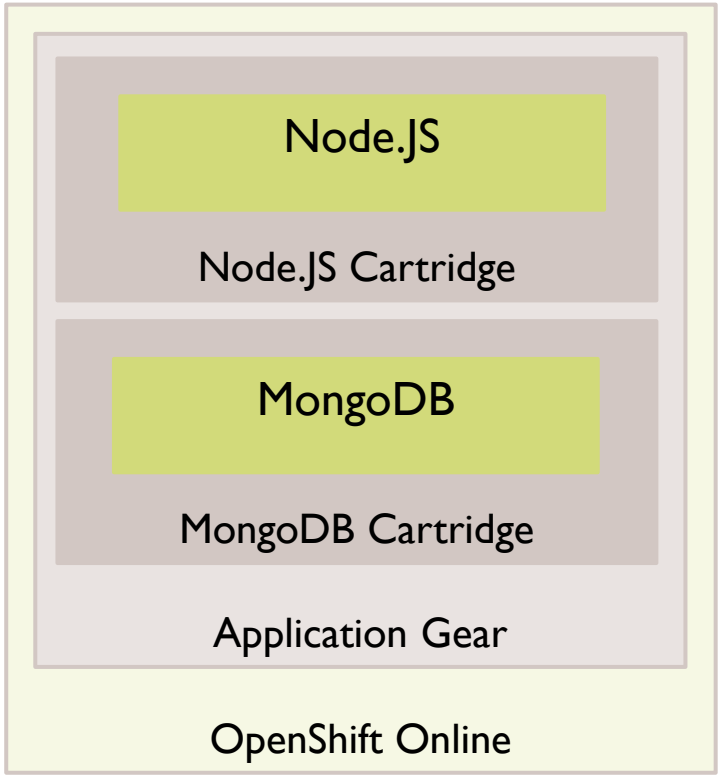
2.5 Datenbank



Erklärung

- Benutzer können in n Gruppen sein
- Benutzer können n Einkäufe für sich selbst und Gruppen durchführen
- Jeder Einkauf kann in nur einem Geschäft durchgeführt werden, hat ein Datum und eine Reihe an gekauften Artikeln
- Diese Artikel haben je eine EAN, einen Namen (und für den Spezifischen Shop einen Preis – das macht die Tabelle „Article-Store“)
- Der Preis kann auch ein Angebot sein (Bild muss hierhingehend noch geupdatet werden, sorry dafür)
- … und so einfach alle Tabellen beschreiben, die Eigenschaften die jede Tabelle hat seht ihr im Inneren ;)

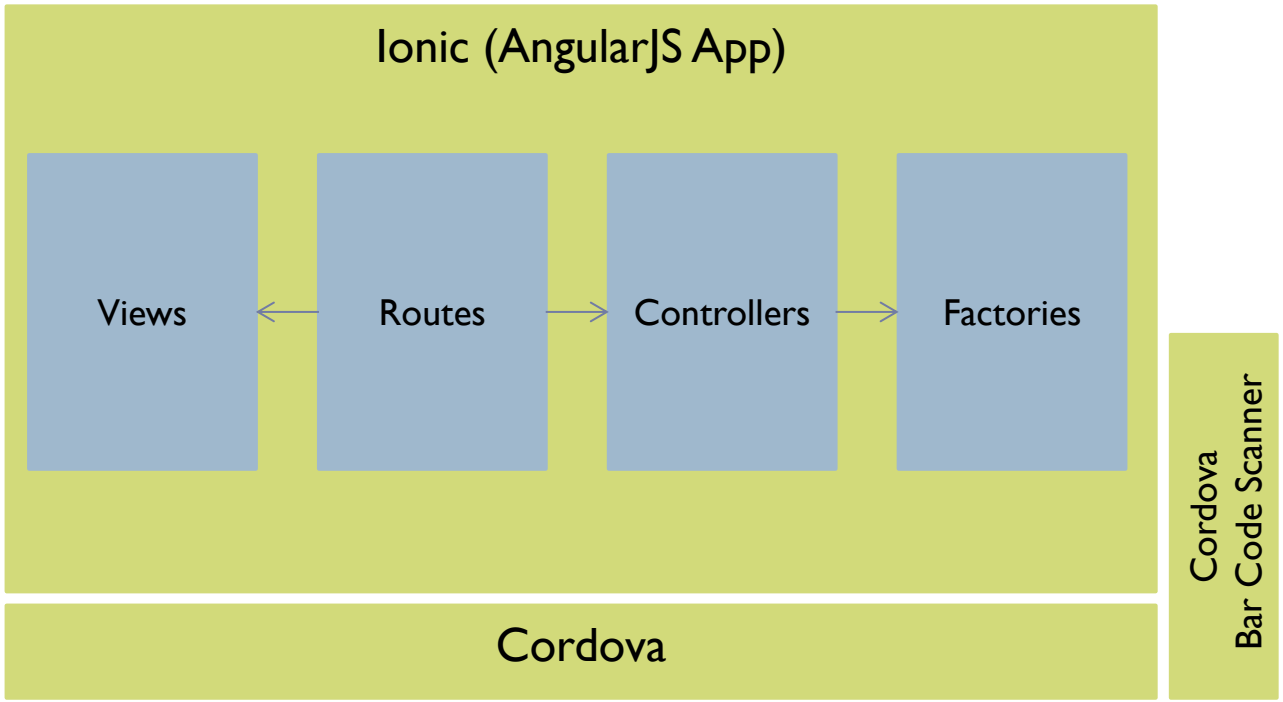
2.6 Hosting



Warum Openshift?

- OpenShift – cloud hosting Plattform
- Ermöglicht die (kostenfreie) Erstellung von „Application Stacks“, d.h. kompletten Umgebungen von Anwendungen
- PaaS, also keine Servereinrichtung mehr
- Anwendungen sind über Cartridges erweiterbar (wie z.B.bei uns durch MongoDB Cartridge)
- Code wird via Git auf die Cloud deployed
- Command Line Tools & Web Console ermöglichen einfaches Management der Anwendung
- Automatische HTTPS Verschlüsselung mit gültigen Zertifikat – ohne Kosten!
- Erwähnenswerte Tools: RockMongo (http://einkaufsapp-hftlswe.rhcloud.com/rockmongo/) – Web DB Management Tool für den direkten Zugriff auf die MongoDB Cartridge. Rockmongo selber ist ebenfalls eine hinzugefügte Cartridge, welche aber, da sie mit unserer Applikation nichts zu tun hat im Bild links nicht vorkommt.

3.1 Architektur Hybrid App - Übersicht



Begriffe

- Ionic: Framework mit HTML, CSS und JS Komponenten zur Entwicklung von Hybriden Apps
- Cordova: Development Framework für die Entwicklung in Javascript anstatt mit den nativen Sprachen der Plattformen
- Cordova Bar Code Scanner: verwendetes Plugin um Bar Codes zu scannen
- Views: HTML Ansichten; Routes: Navigation und Addressierung in der App; Controllers: Business Logik der App; Factories: Datenbankanbindung

3.2 Architektur Hybrid App – Begründung der Komponentenauswahl

- Ionic
  - Open source
  - Einfache Implementierung und viele vorgefertigte Shapes und Tools
  - Möglichkeit auf verschiedene Plattformen zu entwickeln (iOS & Android)
- Cordova
  - Abhängigkeit von Ionic, ist damit unweigerlich Basis unserer Anwendung
  - Einfaches Pluginsystem

3.3 Architektur Hybrid App - Details

- [Hybride Apps Input habe ich an euch weitergeleitet, siehe Mail ;)] – also „Warum hybride Apps?“ könnte hier als Einstieg beantwortet werden
- Ionic implementiert eine AngularJS App → aufgezeigten Elemente sind Bestandteile einer klassischen AngularJS App
- Was ist AngularJS ?
  - Open Source Framework für die Erstellung von Webanwendungen
  - Hauptidee: Databinding in 2 Richtungen – vom View zum Controller und umgekehrt – in Realtime
  - Stellt Programmiermodel bereit, erlaubt neben Binding auch Navigation (siehe Routes)
  - Basis bei der Implementierung einer Ionic App
- Factories (bei uns hier: js/factories.js)
  - Stellen Verbindung zu den in der API definierten Endpunkten her
  - Machen Endpunkte und ihre Funktionen in der App verfügbar (z.B. GET /arcticles/ean/[ean] im Backend würde durch eine Funktion in der Factory abgebildet (function getArticleByEAN(ean){})
- Controllers (bei uns hier: js/controllers.js)
  - Legen fest was auf dem View passiert
  - Beinhalten Funktionen und benötigte Variablen
  - Views greifen auf Funktionen und Variablen zu um Aktionen auszuführen
- Routes (bei uns hier js/app.js)
  - Verknüpfen die Views mit den Controllern
  - Jeder View hat eine spezifische URL sowie einen Namen, auf welche zum Zwecke der Navigation vom Controller aus verwiesen werden kann
  - Jedem View kann ein Controller zugewiesen werden, damit entsprechende Logik auf ihm ausgeführt wird
- Views (bei uns hier index.html & templates/*)
  - Stellen Daten aus Controllern dar und bieten Input zum Ausführen von Funktionen in den Controllern
  - Sind pures HTML mit AngularJS Bindings und Directives
- Cordova BarCode Scanner Plugin
  - CordovanG bietet direkt für AngularJS Plugins an, da Ionic auf Cordova basiert können wir diese ebenfalls verwenden und wird als Modul eingebunden

3.5 Tools

- Ionic eigene CLI („Command Line Interface“) Tools (zu installieren via „npm install ionic cordova“)
  - Ionic serve – Debugging der Anwendung im lokalen Browser
  - Ionic build [plattform] – Erstellen eines Debug APKs für die angegebene Plattform
  - Ionic run [plattform] – Ausführen des APKs auf der angebenen Plattform
- Ionic View – App zum Betrachten und Ausführen der App auf allen Plattformen
- Ionic Creator – Webanwendung zum Entwerfen von Ionic Views – hiermit wurden die meisten unserer Views final designed
- Chrome Inspector – ermöglicht es die Anwendung auf dem Android Gerät zu debuggen
- Text-Editoren: Brackets & Atom
  - Beides Texteditoren mit einem weitreichenden Plugin System für spezielles Syntax-Highlightening, Git Upload
- Git CLI – zum Upload auf Github

4. Quellen

- Openshift Features- https://www.openshift.com/features/
- Ionic - http://ionicframework.com/
- NodeJS - https://nodejs.org/en/
- Why MongoDB - https://www.mongodb.com/blog/post/why-mongodb-popular
- AngularJS - https://de.wikipedia.org/wiki/AngularJS