

ALIAS
SMART LEARNING

Übersicht

1. Wer sind wir?
2. Motivation
3. Was ist Alias?
4. Architektur
5. Frontend
6. Backend
7. Correctness
8. Deployment
9. Fragen erstellen
10. Erfahrungen und Learnings
11. Ausblick

Wer sind wir ?



Patrick Sabau
7. Semester IN



Patrick Schneider
7. Semester IN



Andreas Schüpferling
7. Semester IN

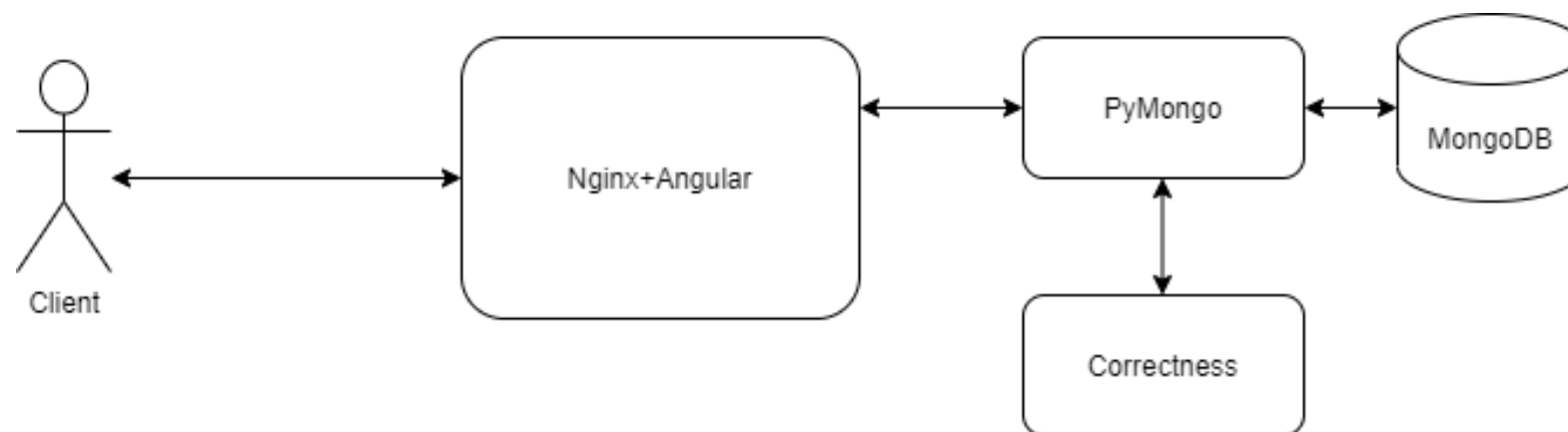
Motivation

- Benutzen selber Karteikarten zum Lernen
- Können wir direktes Feedback in das Lernen integrieren?
- Ist es möglich mit Spracherkennung eine Bewertung zu erzeugen, die der eines Korrektors/einer menschlichen Bewertung entspricht?

Was ist Alias ?

- Smarte Karteikarten-App
- Direktes Feedback der App mittels Sprachverarbeitung:
 - Antworten anschauen und sagen „Das hätte ich gewusst!“ wird eliminiert
- Vergleich mit eigenem Empfinden und dem Empfinden anderer Nutzer
- Analyse des Lernfortschritts
- Fragen werden geteilt unter allen Studenten

Alias - Architektur



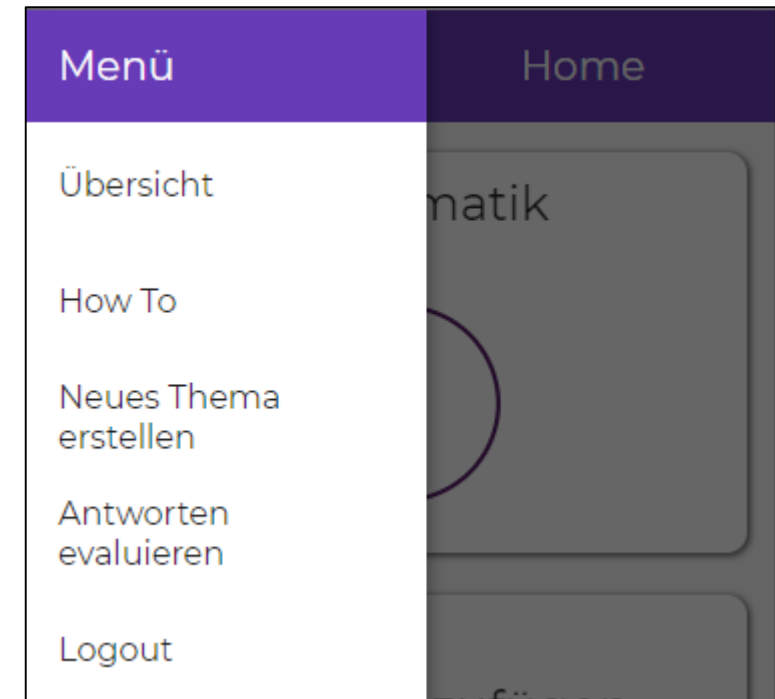
Frontend - Angular

- Frontend-Framework basierend auf Typescript
- Warum Angular?
 - Modernes Design
 - Funktionsreich (Router etc. integriert)
 - Intuitiver als bspw. React
- Zusammensetzung aus (wiederverwendbaren) Komponenten

Frontend – Angular Components



- Zusammensetzung aus (wiederverwendbaren) Komponenten
- HTML, CSS und TS können überall eingebunden werden



Frontend – Angular Components



main-nav-component.ts

```
16 export class MainNavComponent implements OnInit{
17
18     opened: Boolean;
19     location:string="ALIAS";
20
21     constructor(private breakpointObserver: BreakpointObserver
22                 ,private route:ActivatedRoute
23                 ,private router:Router
24                 ,private oauthService:OAuthService) {}
25
26     ngOnInit(){
27         //set the heading in the main nav to the current location/function
28         this.location = this.getLocation(this.route.snapshot['_routerState'].url);
29         //if the route contains ?code=... then do the login process
```

Zeile 16:

- Klassenname zum Einbinden der Component

Zeile 18-19:

- Variablen der Komponente

Zeile 21 ff.:

- Konstruktor mit Injected Moduls

Zeile 26 ff.:

- OnInit Interface, führt Methoden beim starten der Component aus

Frontend – Angular Components



main-nav-component.ts

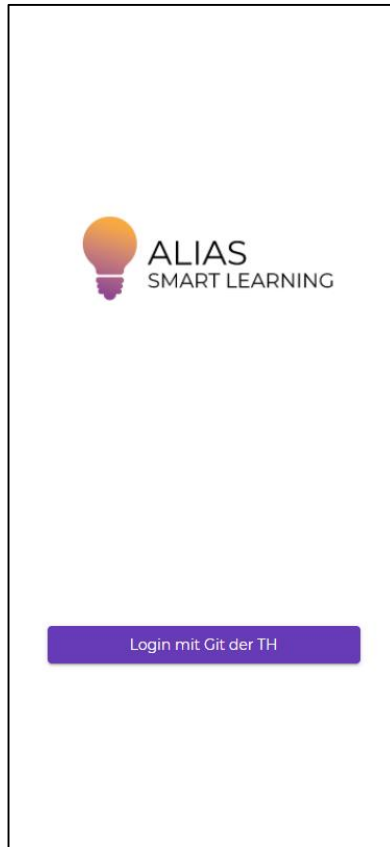
```
getLocation(url:string):string{  
  if(url.startsWith('/home/cards')){  
    return 'Karten'  
  }else if (url.startsWith('/home/create/thema')){  
    return 'Neues Thema erstellen'  
  }else if (url.startsWith('/home/create/card')){  
    return 'Neue Karte erstellen'  
  }else if (url.startsWith('/home/thema')){  
    return 'Themenübersicht'  
  }  
}
```



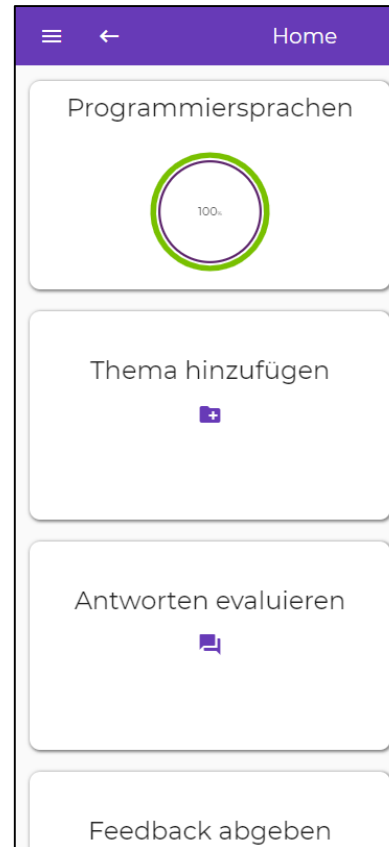
main-nav-component.html

```
<span aria-label="back" class="location">{{ location }}</span>
```

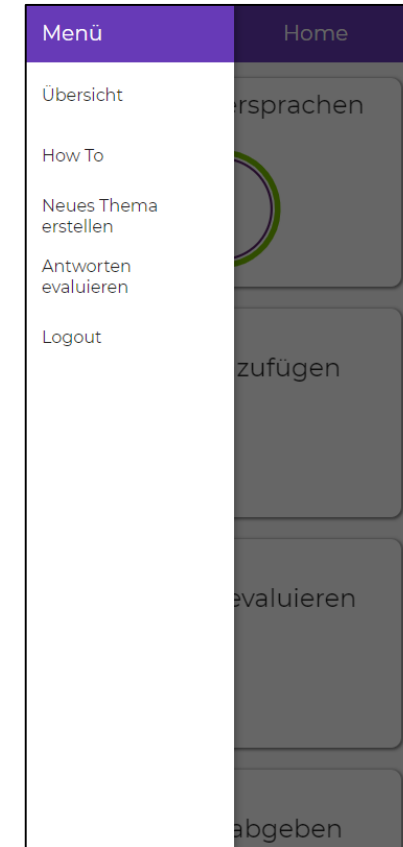
Frontend – Startseite



Login-Screen

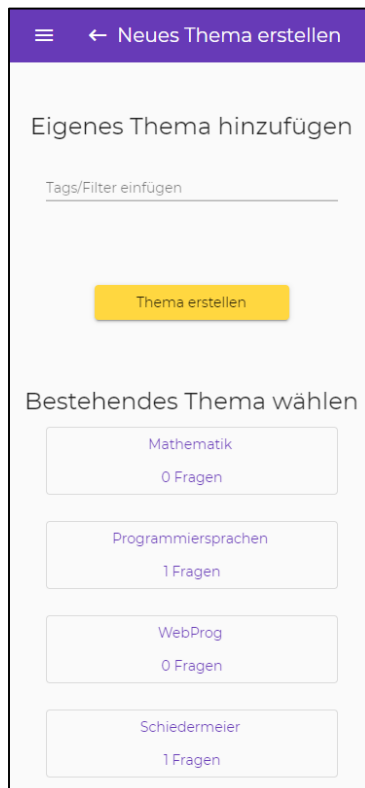


Startseite

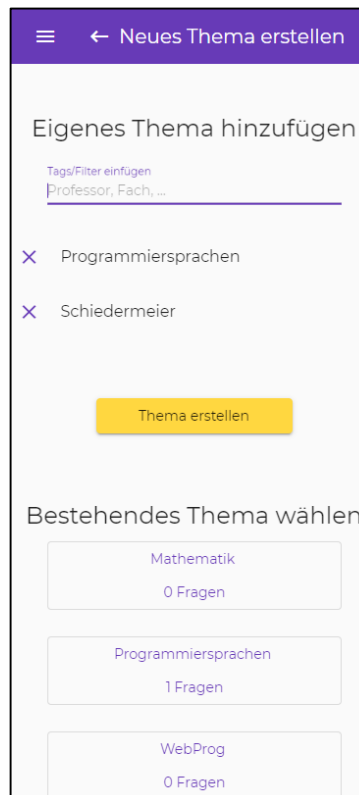


Navigation

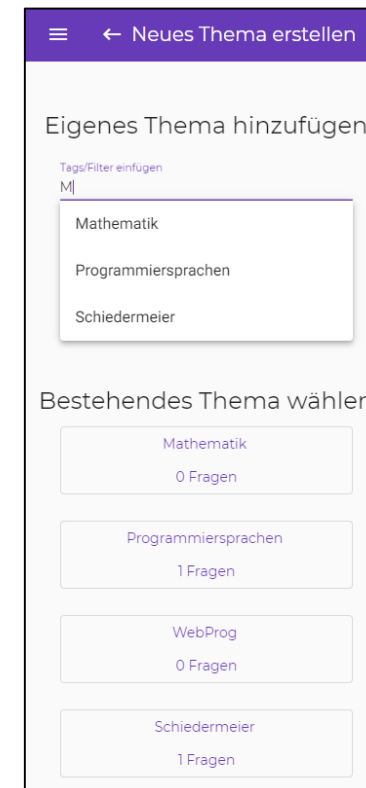
Frontend – Thema hinzufügen



Thema hinzufügen

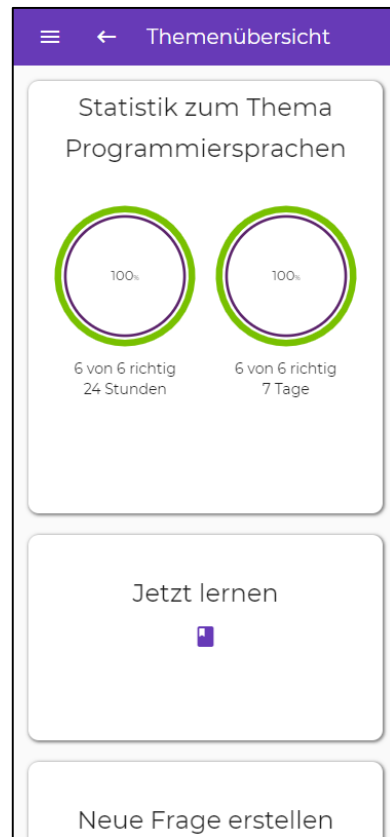


Kombination mehrerer
Tags
(z.B. Fach und Prof)

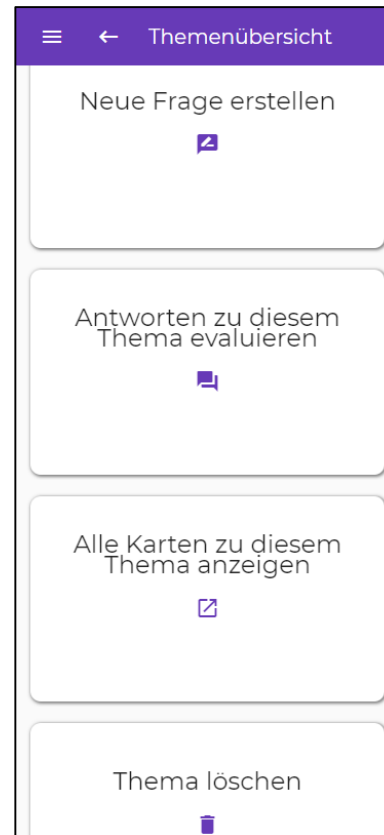


Auto-Completion mit bereit
bestehenden Tags

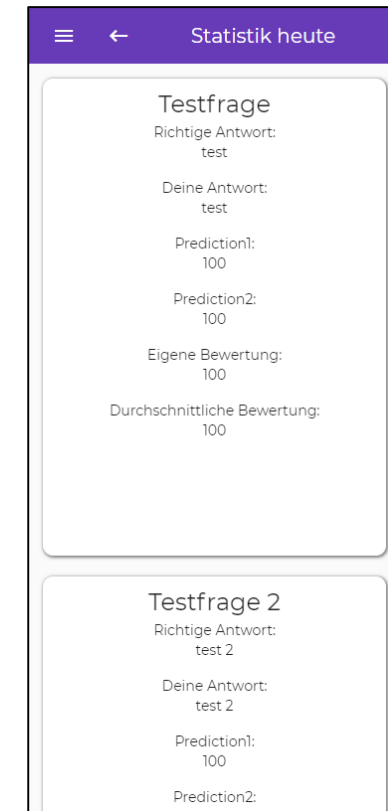
Frontend – Thema



Themenübersicht

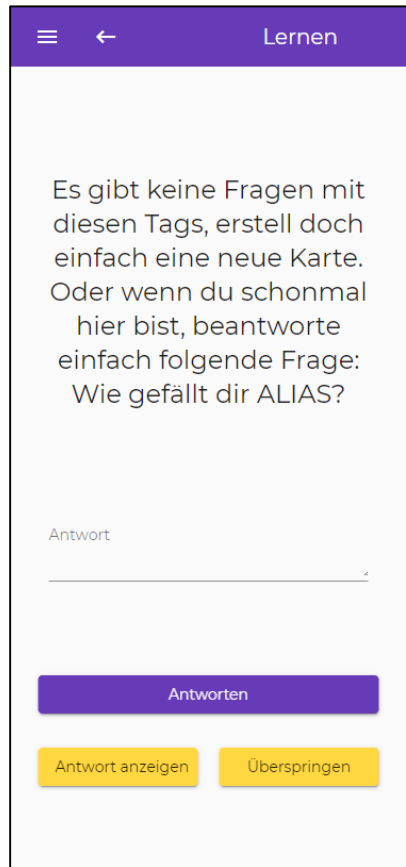


Themenübersicht



Statistik des Themas

Frontend – Lernen



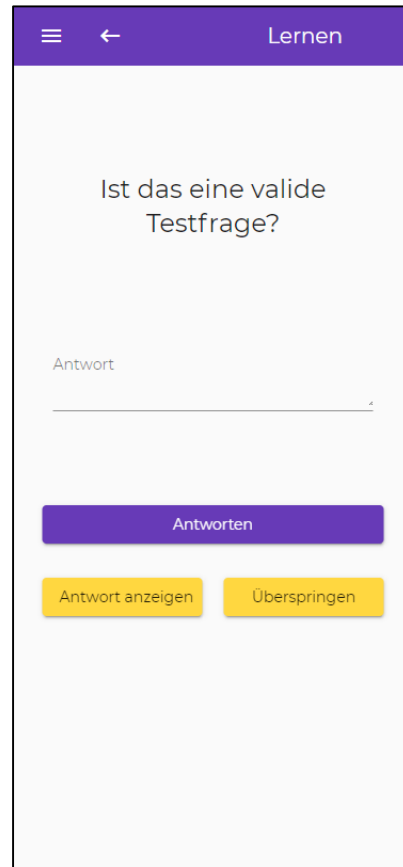
Es gibt keine Fragen mit diesen Tags, erstell doch einfach eine neue Karte. Oder wenn du schonmal hier bist, beantworte einfach folgende Frage: Wie gefällt dir ALIAS?

Antwort

Antworten

Antwort anzeigen Überspringen

Default-Frage, falls es für ein Thema keine Fragen gibt



Ist das eine valide Testfrage?

Antwort

Antworten

Antwort anzeigen Überspringen

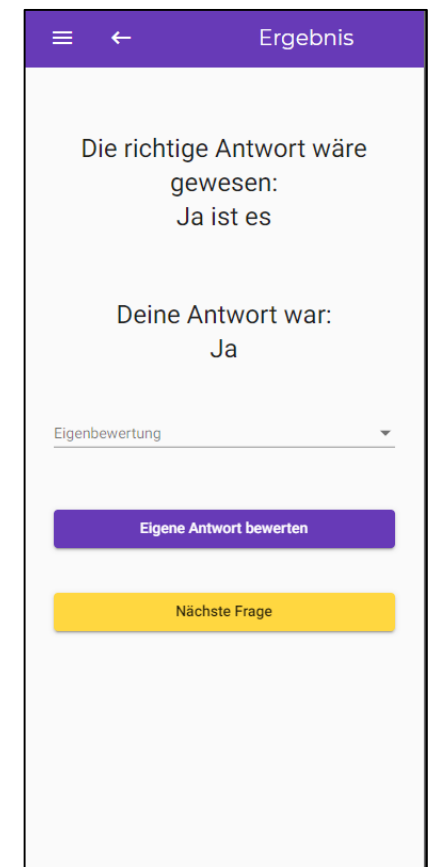
Normale Frage



Die richtige Antwort wäre gewesen: Ja ist es

Nächste Frage

Antwort anzeigen



Die richtige Antwort wäre gewesen: Ja ist es

Deine Antwort war: Ja


Eigenbewertung

Eigene Antwort bewerten

Nächste Frage

Selbstevaluation nach Beantworten der Frage

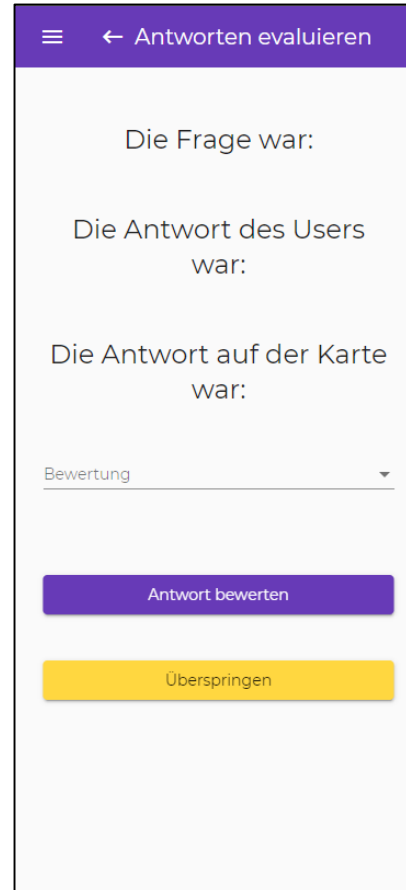
Frontend – Fragen erstellen



The screenshot shows a mobile application interface for creating a new card. At the top, there is a purple header bar with a hamburger menu icon on the left and the text "← Neue Karte erstellen". Below the header, the title "Neue Karte erstellen" is displayed. The form consists of three main sections: a "Frage" (Question) section with a placeholder text "Steht hier eine Frage?" and a text input field; an "Antwort" (Answer) section with a text input field; and a "Zusätzliche Tags einfügen" (Add additional tags) section with a text input field. Below the tags section, there is a list of tags, currently showing "X Programmiersprachen". At the bottom of the form is a yellow button labeled "Karte erstellen".

Fragen erstellen

Frontend – Antworten evaluieren



The screenshot shows a mobile application interface for evaluating answers. At the top is a purple header bar with a hamburger menu icon and the text '← Antworten evaluieren'. Below the header, the screen has a light gray background. It contains three text prompts: 'Die Frage war:', 'Die Antwort des Users war:', and 'Die Antwort auf der Karte war:'. Each prompt is followed by a large, empty text input field. Below these fields is a 'Bewertung' label with a dropdown arrow. At the bottom, there are two buttons: a purple 'Antwort bewerten' button and a yellow 'Überspringen' button.

Antworten evaluieren
(eigene Antworten
ausgeschlossen)

Backend - MongoDB

- documentorientierte NoSQL Datenbank
 - Objekte werden als BSON gespeichert (binary JSON)
 - Collections symbolisieren Tabellen
- Flexibles Datenformat
 - Vorsicht: nicht zwingend einheitlich

Auszug aus Collection „users“

```
1 { "_id" : ObjectId("5ec272bbc624641ecd4604d4"),  
2   "email" : "sabaupa72181@th-nuernberg.de",  
3   "filter" : [ [ "Mathematik" ], [ "BWL", "Badura" ] ] }
```

Backend - PyMongo

- Python-Client für MongoDB
- Python-Dict kann meistens 1:1 eingefügt werden
- Einfügen mit `collection.insert(dict)`

```
cardsCollection.insert_one(dataInsert)
```

- Updaten mit `collection.update({<filter>},{<field>})`

```
cardsCollection.update_one({'_id':ObjectId(oldCardId)},{"$set":{"latest":False}})
```

Backend - PyMongo

- Abfragen mit unterschiedlichen Methoden und Operatoren

```
@app.route('/cards/all', methods=['GET'])
def get_all_cards():
    output = []
    for card in cardsCollection.find():
        card['_id']=str(card['_id'])
        output.append(card)
    return jsonify({'cards':output})
```



collection.find(<filter>)
-> gibt ALLE Objecte aus der
Collection aus

```
card = cardsCollection.find_one({'_id':ObjectId(id)})
```



collection.find_one(<filter>)
-> gibt 1 Object oder "None" aus

```
card = cardsCollection.aggregate([
    {"$match":{"tags":{"$all":[str(tag) for tag in tags]}}},
    {"$match":{"cardId":{"$ne":alias_question_id}}},
    {"$match":{"latest":True}},
    {"$sample":{"size":1}}])
```



collection.aggregate()
-> hat speziellere Operatoren
->{"\$sample":{"size":1}} sorgt dafür
das ein zufälliges Object
ausgegeben wird

Backend - Flask



- Flask ist ein Webframework für WSGI (Web Service Gateway Interface)

```
1  from flask import Flask, jsonify
2  from waitress import serve
3
4  app = Flask(__name__)
5
6  #Get all cards
7  @app.route('/cards/all', methods=['GET'])
8  def get_all_cards():
9      output = []
10     for card in cardsCollection.find():
11         card['_id']=str(card['_id'])
12         output.append(card)
13     return jsonify({'cards':output})
14
15 if __name__ == "__main__":
16     #use waitress server as production server
17     #serve(app,host="0.0.0.0",port=5000)
18
19     #If DB runs from Python script (flask dev server), use:
20     app.run(port=5000)
```

Zeile 4:

- Initialisieren der Flask-App

Zeile 7: @app.route

- Definition einer Route

Zeile 20:

- Starten des Flask-Development Server

Der Flask-Dev-Server sollte nicht produktiv genutzt werden, deshalb:

Zeile 2 & Zeile 17:

- waitress.serve() als Produktions-Server

Backend - Correctness **spaCy**

- spaCy ist eine Python-Library für NLP
- Vortrainierte Modelle
- Für Deutsch: recht eingeschränkt
 - Vor kurzer Zeit: Release eines “großen” News Sprachpaket

```
1  import spacy
2
3  #load the pretrained language pack
4  nlpNews = spacy.load('de_core_news_lg')
```

Backend - Correctness

```
25 #comparison with 'de_core_news_lg'
26 def compareNews(a_str, b_str):
27     try:
28         a = nlpNews(process_text(a_str))
29         b = nlpNews(process_text(b_str))
30         return int(round(a.similarity(b)*100))
31     except:
32         return 50
33
34 #prepare the text for comparison
35 def process_text(text):
36     doc = nlpNews(text)
37     result = []
38     for token in doc:
39         #replace with synonym
40         if token.text in synonyms:
41             result.append(synonyms[token.text])
42             continue
43         #remove stop words
44         if token.is_stop:
45             continue
46         #remove pronouns
47         if token.pos_ == 'PRON':
48             continue
49         result.append(token.lemma_)
50     return " ".join(result)
51
```

Zeile 36:

- Text wird zu Tokens umgewandelt

Zeile 40-41:

- Synonyme werden ersetzt (synonyme werden aus einem Dict gelesen)

Zeile 44-45:

- Stop-Words werden gefiltert (Wörter ohne wirkliche Bedeutung, Füllwörter)

Zeile 47-48:

- Pronomen werden gefiltert (meistens kaum Relevanz für die Bedeutung eines Texts)

Zeile 49:

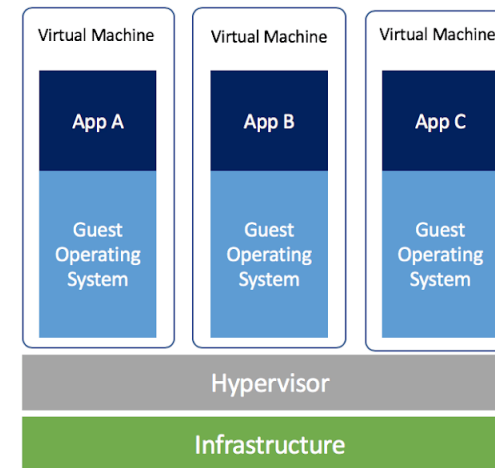
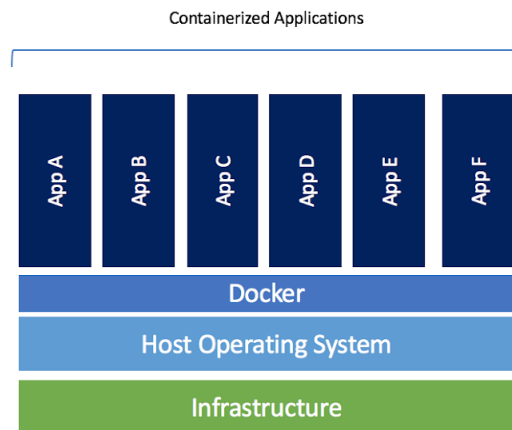
- Wortstamm wird ermittelt

Zeile 30:

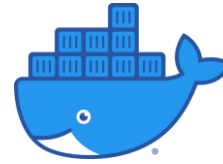
- Gleichheit in Prozent wird ermittelt

Deployment - Docker

- Docker bietet die Möglichkeit Container laufen zu lassen
- Applikationen laufen isoliert (Write once – run everywhere)



Deployment - Docker



Dockerfile

```
1 FROM python:3.7.7-buster
2
3 #First copy the requirments.txt and load them
4 COPY ./requirements.txt /app/requirements.txt
5 WORKDIR /app
6 RUN pip3 install -r requirements.txt
7
8
9 #get the rest of the code
10 COPY ./auth.py /app/auth.py
11 COPY ./backend.py /app/backend.py
12 COPY ./backend_config_dev.cfg /app/backend_config_dev.cfg
13
14
15 #Start the app as flask dev server
16 ENTRYPOINT ["python"]
17 CMD ["/app/backend.py"]
```

Zeile 1:

- Baseimage von Docker-Hub

Zeile 3 ff.:

- Was soll gemacht werden ? Meisten z.B. kopieren von Dateien etc.

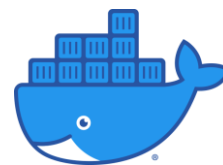
Zeile 16 & 17:

- Wie soll der Container bzw. die Applikation die im Container läuft gestartet werden?

Docker-Container muss vor dem starten gebaut werden:

- `docker build .`
- `docker run <containername> --parameter`

Deployment - Docker



```
1 #####
2 #build the app
3 #####
4 FROM node:12.16.3-buster as build
5
6 WORKDIR /app
7
8 # add `/app/node_modules/.bin` to $PATH
9 ENV PATH /app/node_modules/.bin:$PATH
10
11 WORKDIR /app
12
13 #install and cache app dependencies
14 COPY package.json /app/package.json
15 RUN npm install
16 RUN npm install -g @angular/cli@9.1.4
17
18
19 #generate build
20 COPY . /app
21 RUN ng build --output-path=dist
22
23 #####
24 #serve the app
25 #####
26 FROM nginx
27
28 #copy build from the 'build env'
29 COPY --from=build /app/dist /usr/share/nginx/html
30
31 #Prod config
32 COPY nginx.conf.template /etc/nginx/conf.d/default.conf
33
34 EXPOSE 80
35 EXPOSE 443
36 CMD ["nginx","-g","daemon off;"]
```

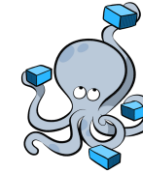
Zeile 1- 21:

- Build-Container, welche die Applikation im Voraus “compiliert” o.ä.

Zeile 26 ff.:

- Starten der Applikation in einem extra Container (hier: nginx)

Deployment – Docker-Compose



docker-compose.yml

```
1  ---
2  version: '3'
3  services:
4  mongodb:
5    image: mongo:4.0.16
6    hostname: mongodb
7    container_name: mongodb
8    volumes:
9      - /data/mongodb:/data/db
10 pymongo:
11   build: ./alias_backend/backend
12   hostname: pymongo
13   container_name: pymongo
14   ports:
15     - "5000:5000"
16   depends_on:
17     - mongodb
18 frontend:
19   build: ./aliasFrontend
20   hostname: frontend
21   container_name: frontend
22   depends_on:
23     - mongodb
24     - pymongo
25   ports:
26     - "80:80"
27     - "443:443"
28   #Volume for cert-mount
29   volumes:
30     - /etc/letsencrypt:/etc/letsencrypt
```

Starten von mehreren Containern gleichzeitig:

Zeile 4-9 , Zeile 10-17, Zeile 18-30:

- Startet jeweils eigene Container mit Parametern

Zeile 8-9:

- Mounten eines Volumes in den Container -> Persistenz

Zeile 14-15:

- Öffnen eines Ports (Port von außen : Port im Container)

Starten mit “docker-compose up”

Under the hood laufen dann Docker-Befehle wie:

- docker network
- docker build
- docker run

Deployment – nginx



nginx.conf

```
1 server {
2     listen 80;
3     server_name alias-learning.de;
4     return 301 https://alias-learning.de$request_uri;
5 }
6
7 server {
8     listen 443 ssl;
9     server_name alias-learning.de;
10    root /usr/share/nginx/html;
11    resolver 127.0.0.11;
12
13    #The location of the certs, created by certbot docker container
14    ssl_certificate /etc/letsencrypt/live/alias-learning.de/fullchain.pem;
15    ssl_certificate_key /etc/letsencrypt/live/alias-learning.de/privkey.pem;
16
17    #proxy for oidc
18    location ~/oidc(.*)$ {
19        proxy_pass https://git.informatik.fh-nuernberg.de$1;
20        proxy_redirect off;
21    }
22
23    #proxy for backend
24    location /api/ {
25        proxy_pass http://pymongo:5000/;
26        proxy_redirect off;
27    }
28
29    #direct all routes to index.html
30    location / {
31        try_files $uri $uri/ /index.html;
32    }
```

nginx ist ein Server, der unter anderem als HTTP Server genutzt wird

Zeile 1-5 :

- Anfragen auf Port 80 werden umgeleitet

Zeile 8-15:

- Configurationen wie Port und SSL

Zeile 17-21:

- Proxy-Configs, um CORS-Probleme bei der Authentifikation zu beheben

Zeile 23-27:

- Proxy/Loadbalancer fürs Backend

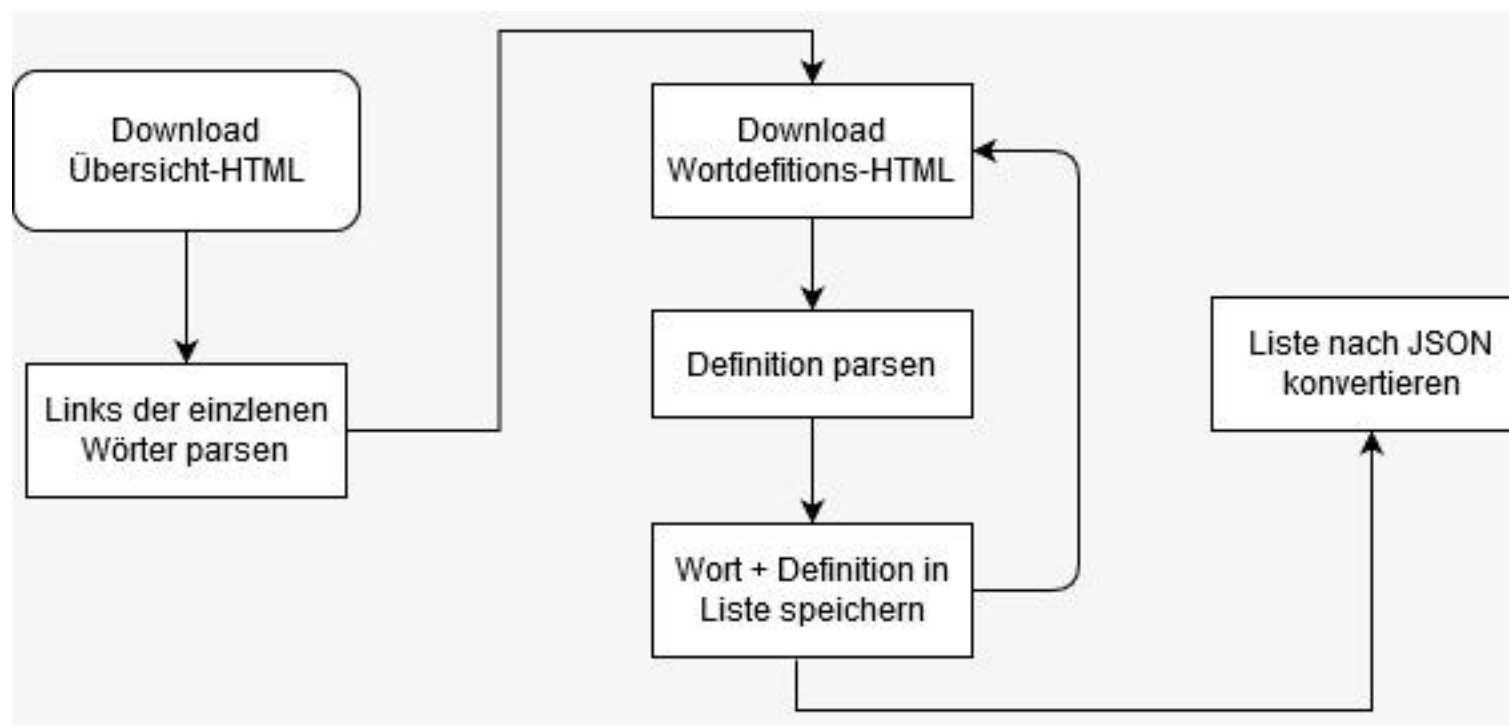
Zeile 29-32:

- ausliefern der Angular-App

Fragen erstellen

- Fächerwahl:
 - Keine Formeln etc.
 - Definitionen
 - Klassische „Karteikartenfächer“
- -> z.B. BWL und Englisch
- Stoff aus den Vorlesungen (Skripte und Moodle-Quizen)
- Skript zur automatischen Generierung

Fragen erstellen



Erfahrungen und Learnings

- Projektmanagement wichtiger als gedacht
 - Kein persönliches Treffen wegen Corona -> Planung vernachlässigt
 - Gesamtüberblick ist essenziell
 - Lieber 1 Tool, anstatt 10 verteilte
 - To-Dos klar definieren -> einheitliches Verständnis

Erfahrungen und Learnings

Problem	Lösung
Projektmanagement	Learnings aus vorheriger Folie
Wenig Nutzerzahlen	<ul style="list-style-type: none">-Falschen Zeitpunkt (Corona -> Auswendig lernen nicht effektiv bei Online-Klausuren)-Mehr direkte Nutzertests mit Feedback vor Veröffentlichung-Mehr Marketing betreiben
Zielgruppe wurde nicht erreicht	<ul style="list-style-type: none">-Feedback-Fächer anders wählen (entsprechend Funktionalität)

Ausblick

- Ohne Nutzer -> keine Daten -> keine sinnvolle Verwendung der App
- Theoretische Möglichkeiten:
 - Sprachverarbeitung optimieren wodurch die Resultate besser werden
 - Vergleich von User-Bewertungen und NLP-Bewertungen
 - Eigenes Sprachmodell trainieren