**Docker**

**Definition:**

Docker ist eine Plattform, mit der man Prozesse in Container verpacken kann. Solch ein Container agiert wie eine kleine effizientere virtuelle Maschine. Dies hilft dabei Anwendungen reibungslos über verschiedenste Umgebungen hinweg auszuführen.

**Docker-Commands:**

* docker pull <image:version>: Zieht ein Docker-Image aus dem Repository.
* docker run -it -rm <image> <application>: startet einen Container mit einer interaktiven Terminalverbindung, welcher nach dem Schließen automatisch gelöscht wird.
* docker images: Listet alle auf dem System verfügbaren Images auf.
* docker ps: Zeigt laufende Docker-Prozesse an.
* docker logs <container/id>: Überprüft die Logs eines Containers
* docker stop <container/id>: Stoppt einen laufenden Docker-Container.
* Docker kill <container/id>: Beendet einen Docker-Container ohne Wartezeit.
* Docker rm <container/id>: Entfernt einen Docker-Container.
* Docker rmi <image>: Entfernt ein Docker-Image vom System.
* Docker tag <image:version> <new\_image\_name>: benennt Images um

**MongoDB**

**Definition:**

MongoDB ist eine dokumentenorientierte NoSQL-Datenbank, die für ihre Flexibilität, Skalierbarkeit und Leistungsfähigkeit bekannt ist.

**MongoDB-Commands:**

* use restaurants: In Datenbank wechseln
* db.restaurants.find(): alle Daten ausgeben
* db.restaurants.find( { }, { name: 1}): nur Name anzeigen
* db.restaurants.find({}, {\_id: 0, name: 1}): Name anzeigen, \_id explizit nicht anzeigen
* db.restaurants.find({}, {"address.zipcode": 1,}): zipcode (im Objekt address) anzeigen
* db.restaurants.find({borough: 'Bronx'}): nach bestimmten Namen Filtern
* db.restaurants.find({borough: 'Bronx'}).limit(5): die Ausgabe auf 5 Einträge limitieren
* db.restaurants.find({borough: 'Bronx'}).skip(5): die 5 ersten Einträge überspringen
* db.restaurants.find({"grades.score": {$gt:90}}): nach größe sortieren
* db.restaurants.find( { "address.coord.0": { "$lt": -95.754168 } }): nach größe in array sortieren
* db.restaurants.find({"cuisine": { "$ne": "American " } }): not equal
* db.restaurants.find("$or": [ { "cuisine": "American " }, { "cuisine": "Chinese" } ] }: ODER
* db.restaurants.find({ "borough": { "$in": ["Staten Island","Queens","Bronx","Brooklyn"] } }}): enthalten in
* db.restaurants.find({ "borough": {"$nin": ["Staten Island","Queens","Bronx","Brooklyn"] } }): NICHT enthalten in