(RETL) Roman emperor tier list

# Projektteam

Gregory Reiter, Tim Freitag

# Wahl der Projekt-Variante

Welche Variante setzen Sie um?

Applikation mit anderen Technologien

# Projektidee

Man kann alle römischen Herrscher bewerten und Kommentare dazu abgeben.

Man kann die Römer in der Datenbank mit ihren Bewertungen und ihren Kommentaren speichern.

# NoSQL-Datenbank-Wahl

MongoDBs flexibles Datenmodell ist ideal für Tierlist-Apps: hohe Leistung, Skalierbarkeit und Echtzeit-Updates.

# Datenmodell

## Herrscher

{  
  "\_id": "6492d5bfdc97107ffaf303da",  
  "index": 3,  
  "name": "Caligula",  
  "name\_full": "GAIVS IVLIVS CAESAR AVGVSTVS GERMANICVS",  
  "birth": "12-08-31",  
  "death": "41-01-24",  
  "birth\_cty": "Antitum",  
  "birth\_prv": "Italia",  
  "rise": "Birthright",  
  "reign\_start": "37-03-18",  
  "reign\_end": "41-01-24",  
  "cause": "Assassination",  
  "killer": "Senate",  
  "dynasty": "Julio-Claudian",  
  "era": "Principate",  
  "notes": "assassination may have only involved the Praetorian Guard",  
  "verif\_who": "Reddit user zonination",  
  "image": "[https://upload.wikimedia.org/wikipedia/commons/7/7b/Caligula\_Rekonstru…"](https://upload.wikimedia.org/wikipedia/commons/7/7b/Caligula_Rekonstru…%22)  
}

## Bewertung

{

"\_id": "6492bf85416fd199024a8c93",

"emperorId": 3,

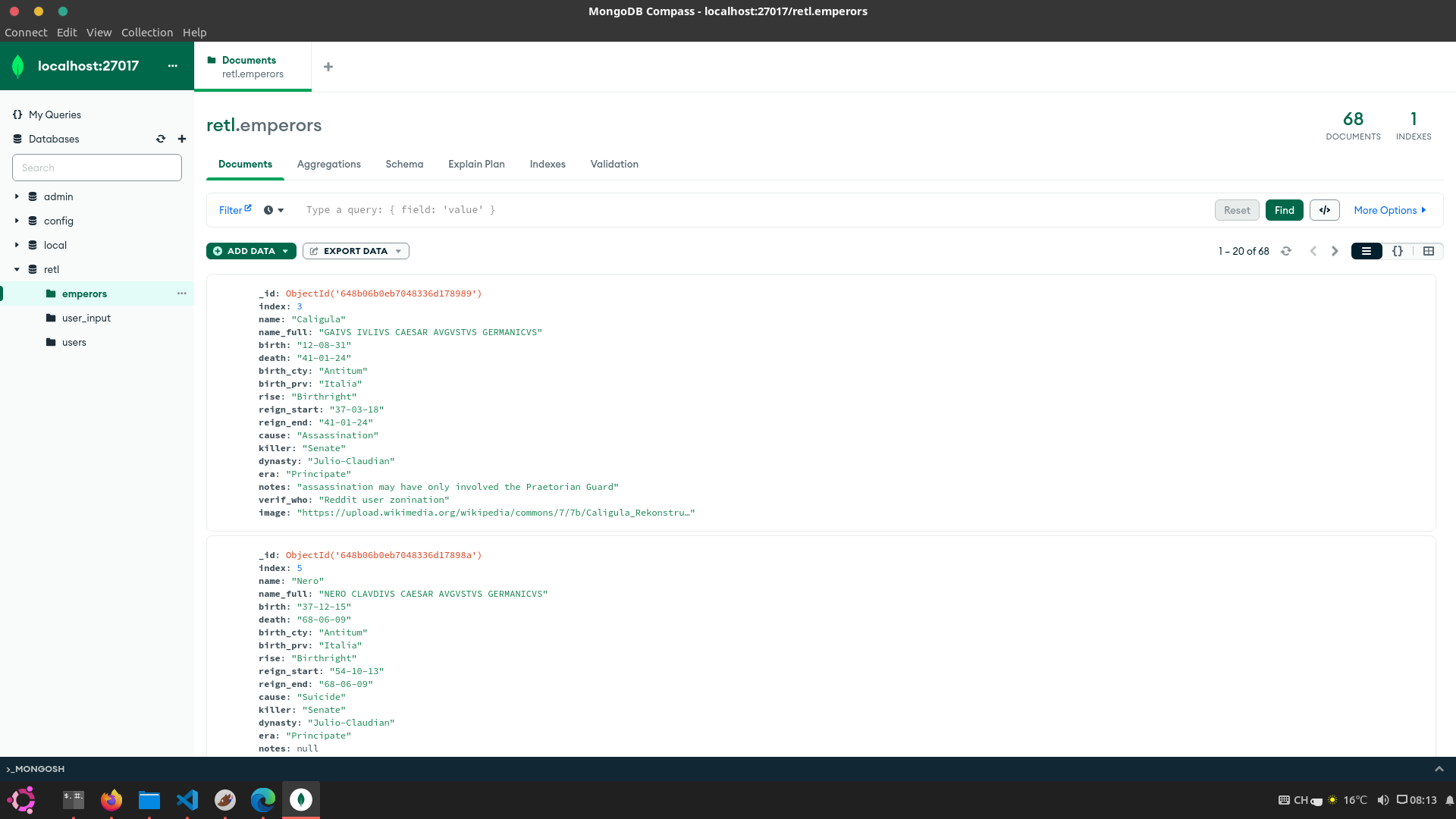
"user": "greg",

"review": "Very NICE",

"rating": 5

}

# Screenshots mit Daten

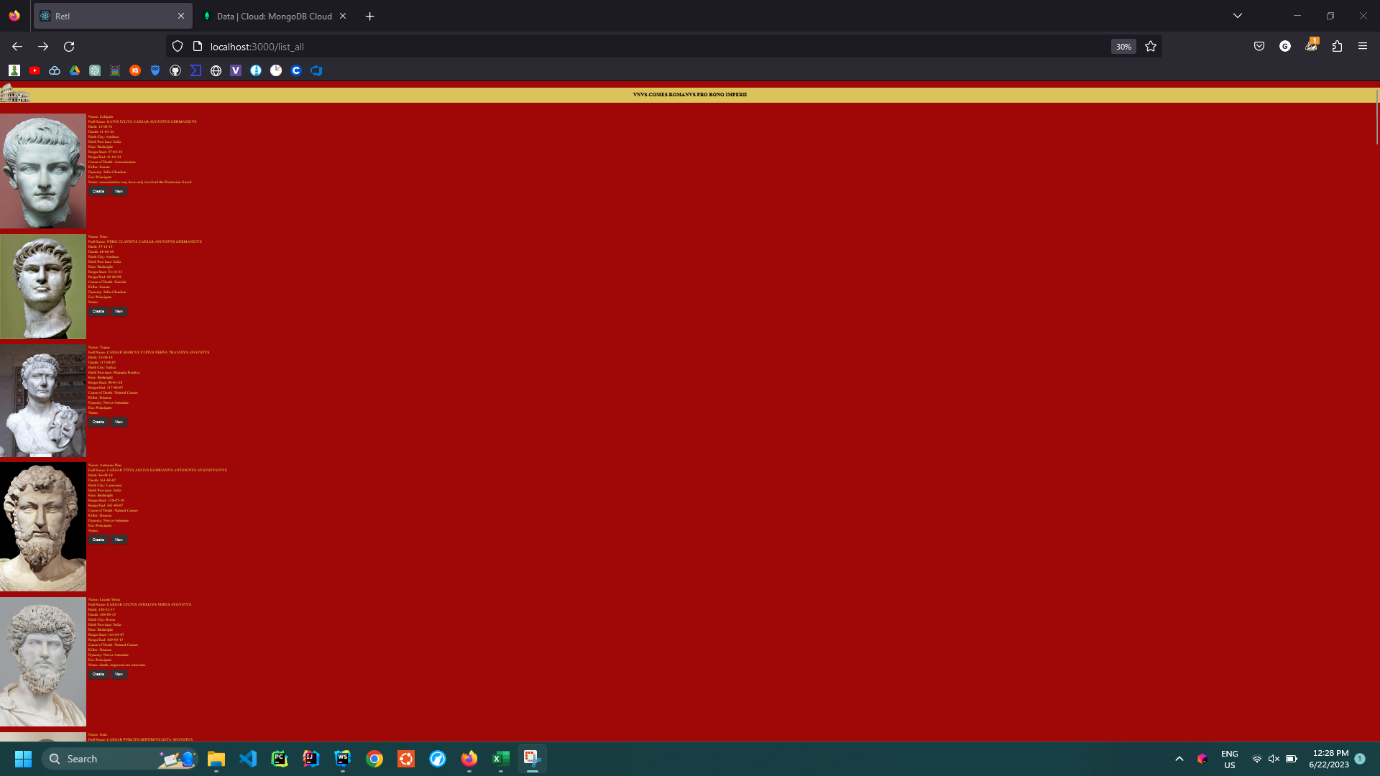


# Screenshots Applikation

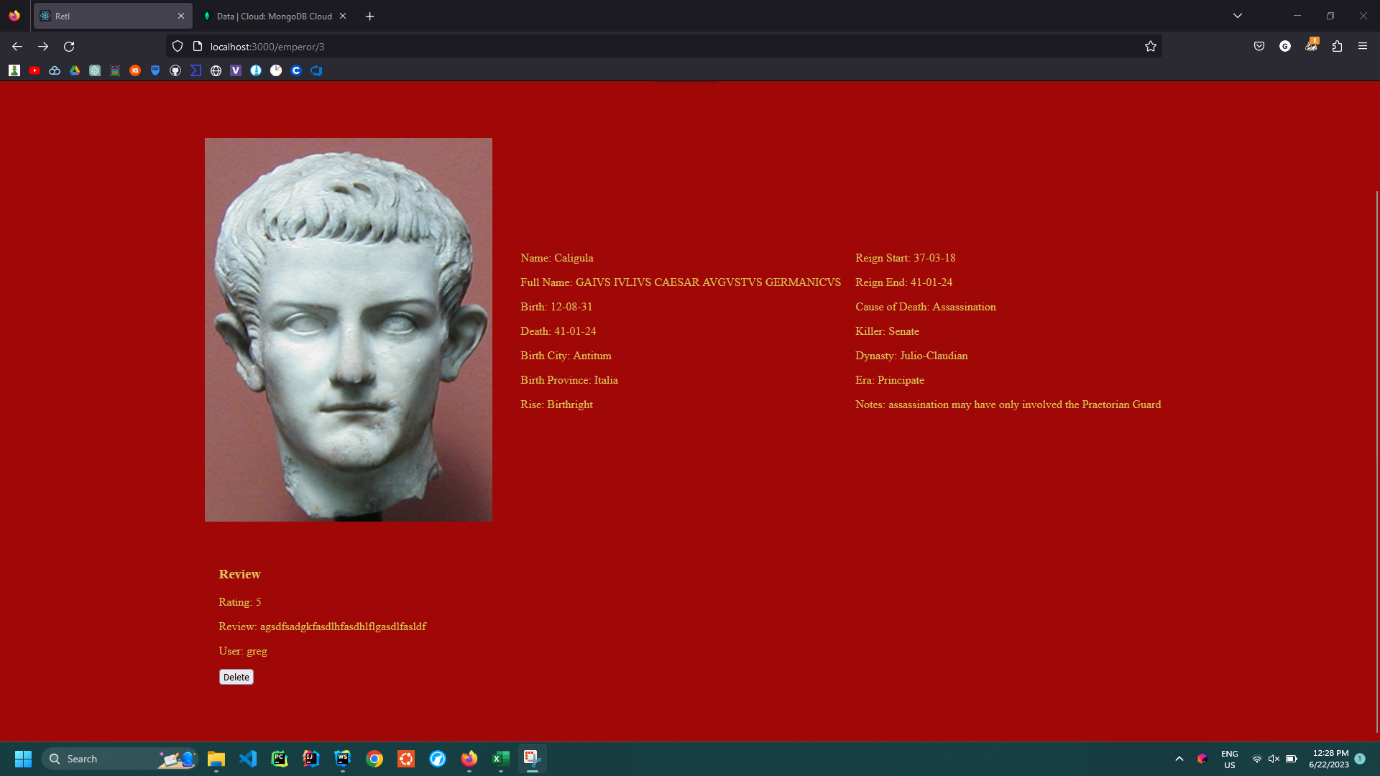
Die Startseite unserer Anwendung hat zwei Knöpfe. Der erste Knopf ist der grosse «SPQR» Knopf, welcher auf die Liste der Herrscher führt. Der Zweite Knopf ist das Kolosseum der von jeder Seite wieder auf die Startseite führt.



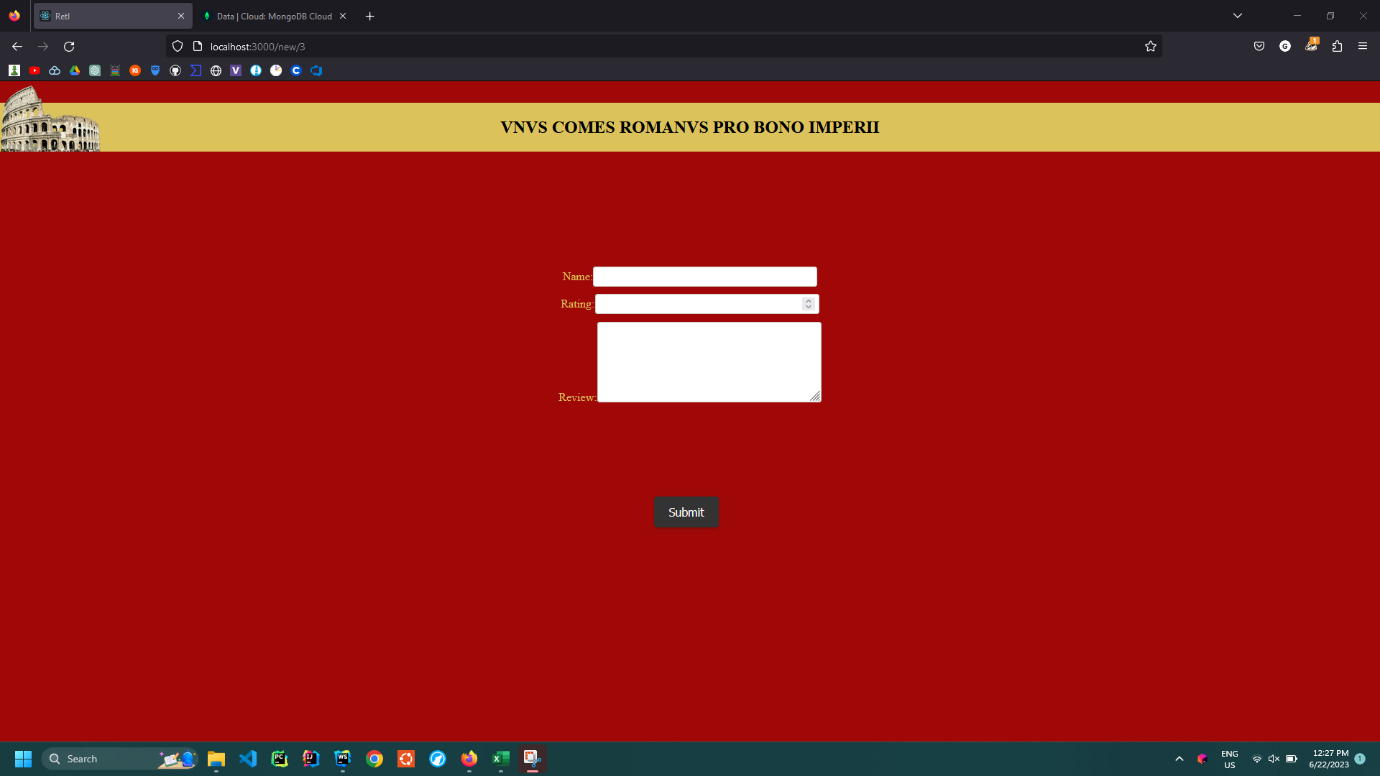
Dies ist die Liste der römischen Herrscher die auf Knopfdruck angezeigt wird. Jeder Eintrag enthält alle Informationen des jeweiligen Herrschers. Dazu kann man auch einzelne Einträge mit «view» im Detail anschauen, oder den jeweiligen Eintrag mit «delete» löschen.



Dies ist die Detailansicht eines römischen Herrschers. Hier kann man zusätzlich noch Benutzerbewertungen und Kommentare zum angezeigten Römer sehen. Mit «delete» kann man die eigene Bewertung bei Bedarf wieder löschen.



Dies ist die Bewertungsseite. Hier kann man den Benutzernamen angeben, einen Kommentar verfassen und eine Bewertung abgeben. Mit «submit» wird die Bewertung gespeichert.



# Zusatz Datenbank

## Transaction

Für die Transaction haben wir eine Python-Datei erstellt, welche eine Transaktion durchführt. Zuerst wird in einem Try-Catch überprüft ob man sich mit der Datenbank retl über mongodb://localhost:27017 verbinden kann. Als nächstes wird eine Sitzung mit client.start\_session() gestartet in welcher die Transaktion mit session.start\_transaction() dann gestartet wird. Darauffolgend wird ein try gestartet, das fehlschlägt, wenn die darauffolgenden Befehle nicht richtig ausgeführt werden können. Wenn der try fehlschlägt wird die Transaktion mit der Nachricht «Transaction aborted» abgebrochen. Wenn das try aber erfolgreich ausgeführt wird, wird die Transaktion mit session.commit\_transaction() «commitet».

from pymongo import MongoClient

from pymongo.errors import ConnectionFailure

try:

client = MongoClient('mongodb://localhost:27017/')

db = client['retl']

except ConnectionFailure:

print("Failed to connect to MongoDB.")

with client.start\_session() as session:

with session.start\_transaction():

try:

collection = db['user\_input']

collection.insert\_one({"rating": 1, "comment": "toll!", "user\_id": 1, "emp\_id": 1})

collection.insert\_one({"rating": 1, "comment": "toll!", "user\_id": 2, "emp\_id": 1})

collection.update\_one({"user\_id": 2}, {"$set": {"rating": 9}})

session.commit\_transaction()

print("Transaction committed successfully.")

except Exception as e:

print("Transaction aborted:", str(e))

## Backup

Mit mongodump kann man alle Datenbanken auf einem System in eine bson-Dateien speichern. Mit dem Befehl mongorestore kann man dann alle Datenbanken a wiederherstellen.

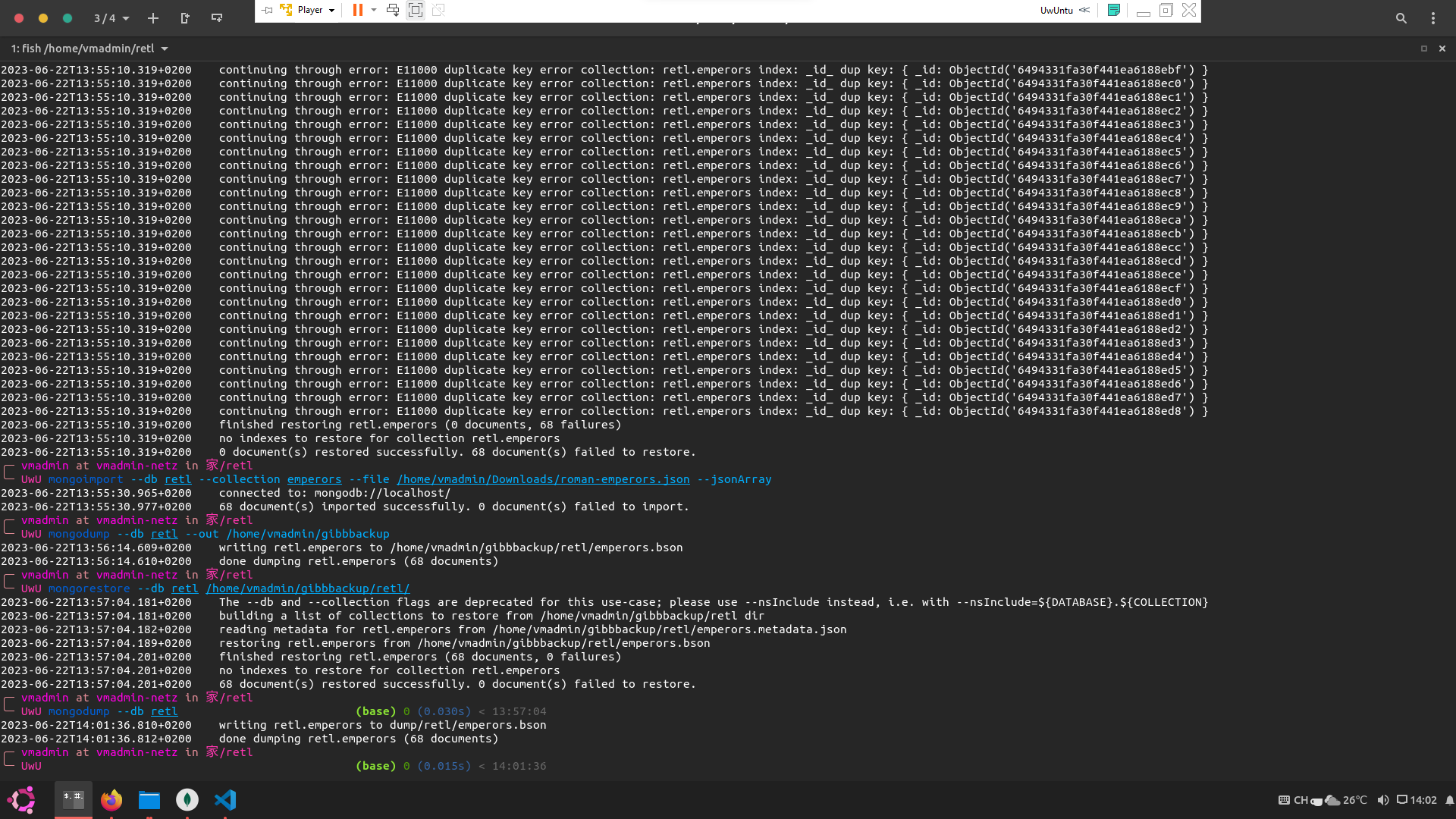
Wenn man eine spezifische Datenbank an einen spezifischen Ort «dumpen» will, kann man mongodump --db «DB-Name» --out «Speicherort». Wenn man dann diese Datei wiederherstellen möchte, kann man dies mit mongorestore --db «DB-Name» «Pfad zur BSON-Datei».

Die BSON-Datei wird am angegebenen Pfad in einem Ordner mit dem DB-Namen gespeichert.

In unserem Beispiel wären die Befehle so:

mongodump --db retl --out /home/vmadmin/gibbbackup

mongorestore --db retl /home/vmadmin/gibbbackup/retl



# Arbeitsjournal / Reflexionen

|  |
| --- |
| **Donnerstag, 8. Juni 2023** |
| ***Wir haben die React-Seite schon aufgesetzt und müssen diese nur noch erweitern. Die MongoDB Datenbank wurde auch schon mit einer dazugehörigen Collection aufgesetzt.*** |
|  |
| ***Es gab immer wieder Schwierigkeiten beim Importieren der JSON-Datei. Bing hat das Problem lösen können.*** |
|  |
| ***Wir müssen noch ein funktionierendes Backend für die Datenbank machen und die Website fertigstellen.*** |
|  |

|  |
| --- |
| **Donnerstag, 15. Juni 2023** |
| *Heute haben wir unseren Plan für die Datenbank abgeschlossen und in MongoDB umgesetzt. Wir haben auch alle Kompetenzen für den Datenbankteil abgeschlossen.* |
|  |
| *Wir hatten viele Probleme damit, wie wir unser Schema erstellen sollten.* |
|  |
| *Das nächste, was wir tun müssen, ist, es mit der Front-End-Anwendung zu verbinden und dafür zu sorgen, dass es gut aussieht.* |
|  |

|  |
| --- |
| **Donnerstag, 22. Juni 2023** |
| ***Heute erstellen wir ein Backup unserer Datenbank für zukünftige Wiederherstellungen und aus Gründen der Redundanz. Wir auch die Front-End- und Back-End-Arbeiten abgeschlossen.*** |
|  |
| ***Die Verbindung des Front-Ends zur Datenbank hat uns einige Probleme bereitet, auch das Back-End hatte Schwierigkeiten bei der Verbindung zur Datenbank. die ordnungsgemässe Ausführung der REST-Anfragen bereitete uns Probleme.*** |
|  |
| ***Wir sind fertig 😊🗿*** |
|  |