-Die Aufgabe im Bild unten wird im Projekt behandelt. Da es keine direkte Anleitung zu Parametern wie Sicherheit und Datenbanken in CSV Daten und Code gibt, sind die von mir entworfenen Add-Ons unten ausgeführt, und am Ende ist einen weiteren Hinweis als "fortgeschrittene Studien" hinzugefügt worden.Graphical user interface, text, application, email

Description automatically generated

-CSV-Daten wurden untersucht und es wurde entschieden, mit "Set Data Struktur" zu arbeiten, da die Werte in der RL100-Code-Spalte eindeutig sind und die Suche durch diese Spalte bearbeitet wird.

-Es gibt leere Werte in der Spalte mit dem Namen Typ Kurz, ich wollte Informationsverlust verhindern, indem ich in diesen Abschnitten Typ Lang verwende.

Table

Description automatically generated

Graphical user interface, text, application

Description automatically generated

-Ich habe eine oberflächliche Unit-Test-Methode über die OperationOfficeFactory-Klasse geschrieben, aus der wir die CSV Daten einlesen. Obwohl es nicht direkt mit dem Konzept des Testens zusammenhängt, wurde die Methodenvielfalt durch die Gestaltung von Validierungsbestimmungen mit Unit-Test-Methoden erhöht, und gleichzeitig wurden "Stream-Operationen" erwähnt.

Text, chat or text message

Description automatically generated

Graphical user interface, text

Description automatically generated

-Integrationstest wurde mit einer Datenbank auf AWS durchgeführt (Details folgen später).

Graphical user interface, text, application

Description automatically generated

- Dank der während "Request" laufenden Validierungen wollte ich die Verschwendung von Ressourcen minimieren. Text

Description automatically generated with low confidence

- Collection link in Postman: <https://www.getpostman.com/collections/a259b863777eadeb1f6b>

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

- Im ersten und einfachen Szenario werden csv-Dateien unabhängig von einer Datenbank mit dem Pojo-Objekt namens „OperationOffice“ verarbeitet. Diese Anfrage zielt darauf ab, die gewünschte Basisleistung im Projekt darzustellen, ohne dass eine Sicherheit verlangt wird.

A screenshot of a computer

Description automatically generated

-Statt Fehlerdetails wie "timestamp-status-error-trace" usw. Durch "Exception Handling"-Mechanismen werden dem Benutzer verschiedene Fehlermeldungen angezeigt.

A screenshot of a computer

Description automatically generated with medium confidence

-Im zweiten Szenario wird dieselbe OperationOffice-Klasse als Entität mit einem anderen Ansatz dargestellt, indem die Datenbank-, Authentifizierungs- und Autorisierungsprozesse in einer mehrschichtigen Architektur durchlaufen werden.

-Eine einfache Benutzerregistrierung und ein Login-Code wurden entwickelt. Das Passwort ist verschlüsselt und die APIs Connection wird mit JWT bereitgestellt.

A screenshot of a computer

Description automatically generated with medium confidenceGraphical user interface, text, application

Description automatically generatedGraphical user interface, text, application

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated

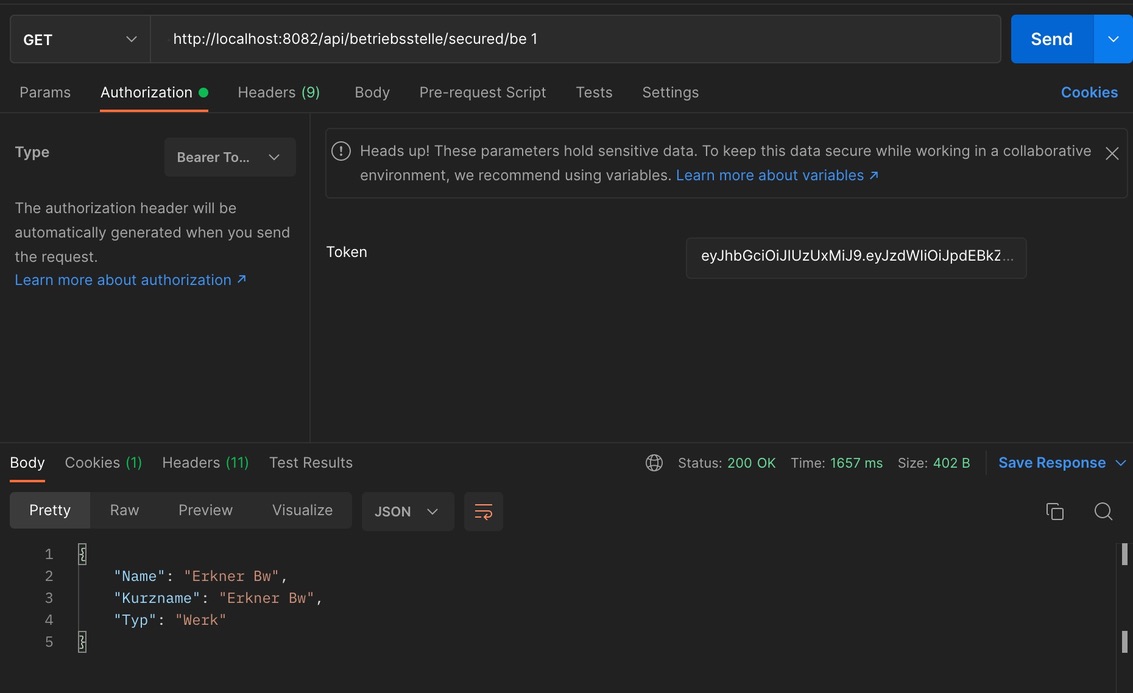
A screenshot of a computer

Description automatically generated with medium confidence

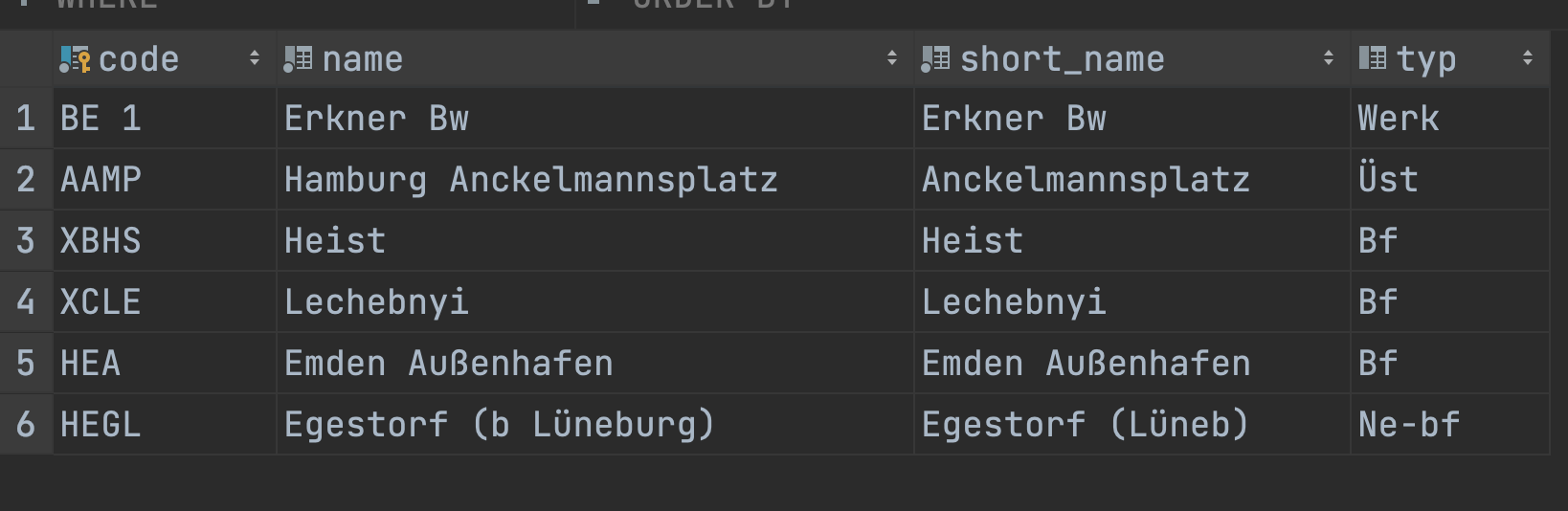
-Der Benutzer kann ohne das Token nicht mit dem Server kommunizieren. Als Ergebnis erfolgreicher Anmeldungen mit dem Token zurückgegebene Objekte werden in einer Datenbank gespeichert, die sich in der AWS-Cloud befindet! Ich erwähnte, dass ich einen einfachen Integrationstest über diese Datenbank geschrieben habe!

A screenshot of a computer

Description automatically generated



-Da es sich um den Primärschlüssel Code handelt, wird Daten Wiederholung vermieden! Also selbst wenn der Benutzer dieselbe Anfrage 100-mal zu unterschiedlichen Zeiten schreibt, wird nur 1 aufgenommen!



zusätzliche Verbesserungen

* versioning and documentation of APIs
* Dockerizing Jar files and run on AWS ECS
* Different Grant Types Using Oauth2
* Logging Operations
* Spring MVC + Angular