The QuoteServices App Case Study: Zippy Microservice Structure & Functionality (Part 2)

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science

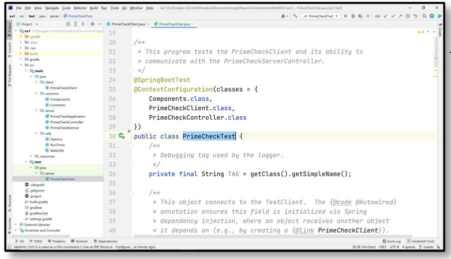
Institute for Software
Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

- Understand the structure & functionality of the ZippyQuoteRepository that uses the Jakarta Persistence API (JPA)

QuoteDriver

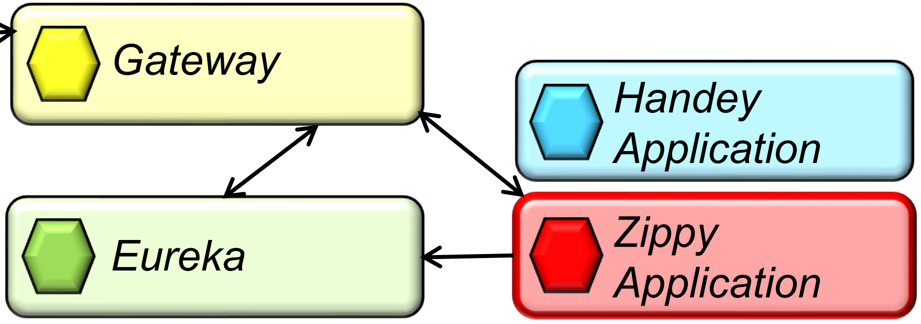


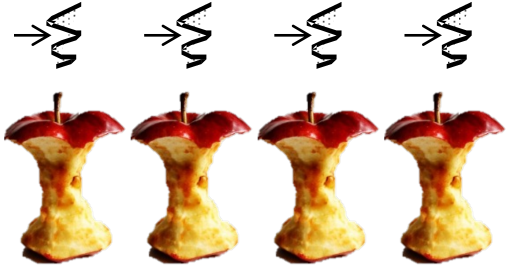
```
/**
 * This program tests the PrimeCheckClient and its ability to
 * communicate with the PrimeCheckServerController.
 */
@SpringBootTest
@ContextConfiguration(classes = {
    Components.class,
    PrimeCheckClient.class,
    PrimeCheckController.class
})
public class PrimeCheckTest {
    /**
     * Debugging tag used by the logger.
     */
    private final String TAG = getClass().getSimpleName();

    /**
     * This object connects to the TestClient.  The {@code @Autowired}
     * annotation ensures this field is initialized via Spring
     * dependency injection, where an object receives another object
     * it depends on (e.g., by creating a {@link PrimeCheckClient}).
```

Asynchronous HTTP GET/POST requests/ responses

Microservice-based Quotes App





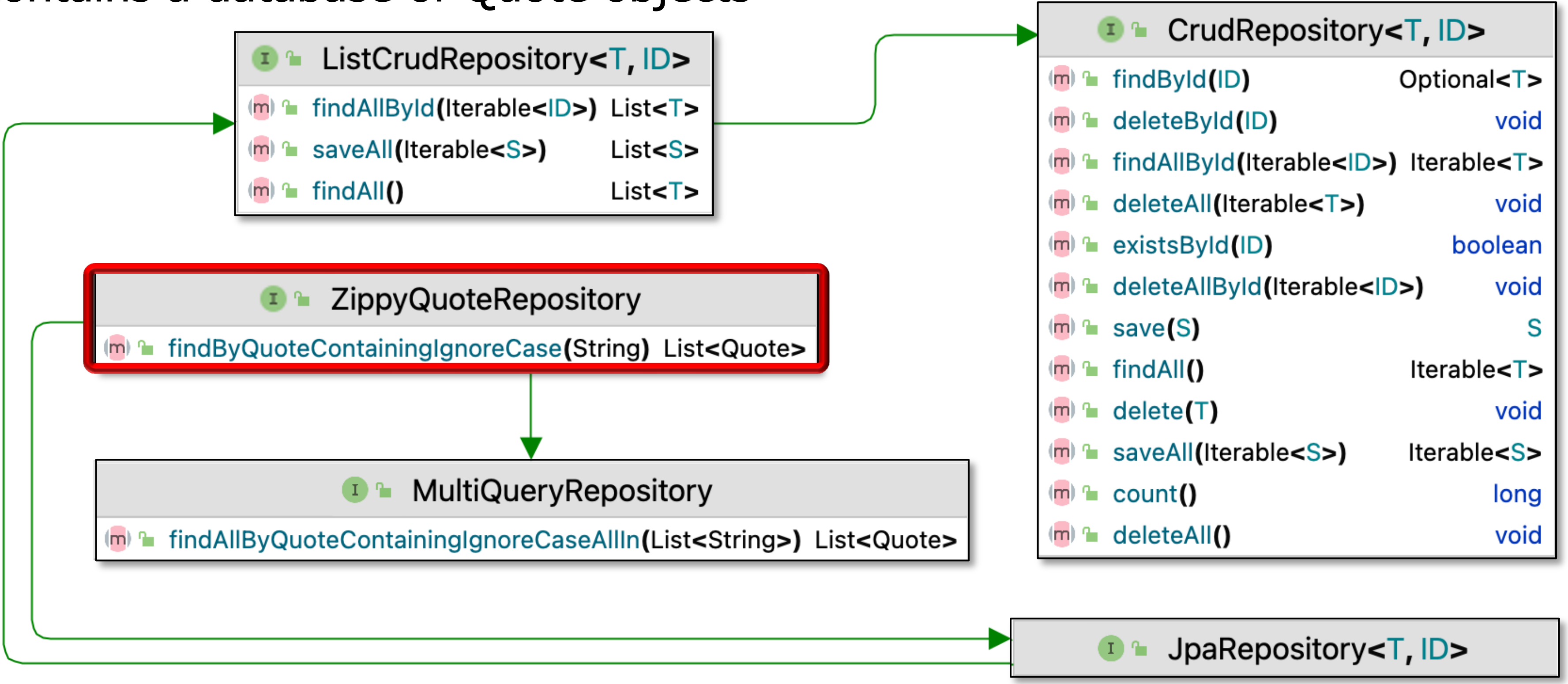
JPA

This microservice also returns Project Reactor reactive types (Flux)

Structure & Functionality of the ZippyQuoteRepository

Structure & Functionality of the ZippyQuoteRepository

- ZippyQuoteRepository is a Jakarta Persistence API (JPA)-based repository that contains a database of Quote objects

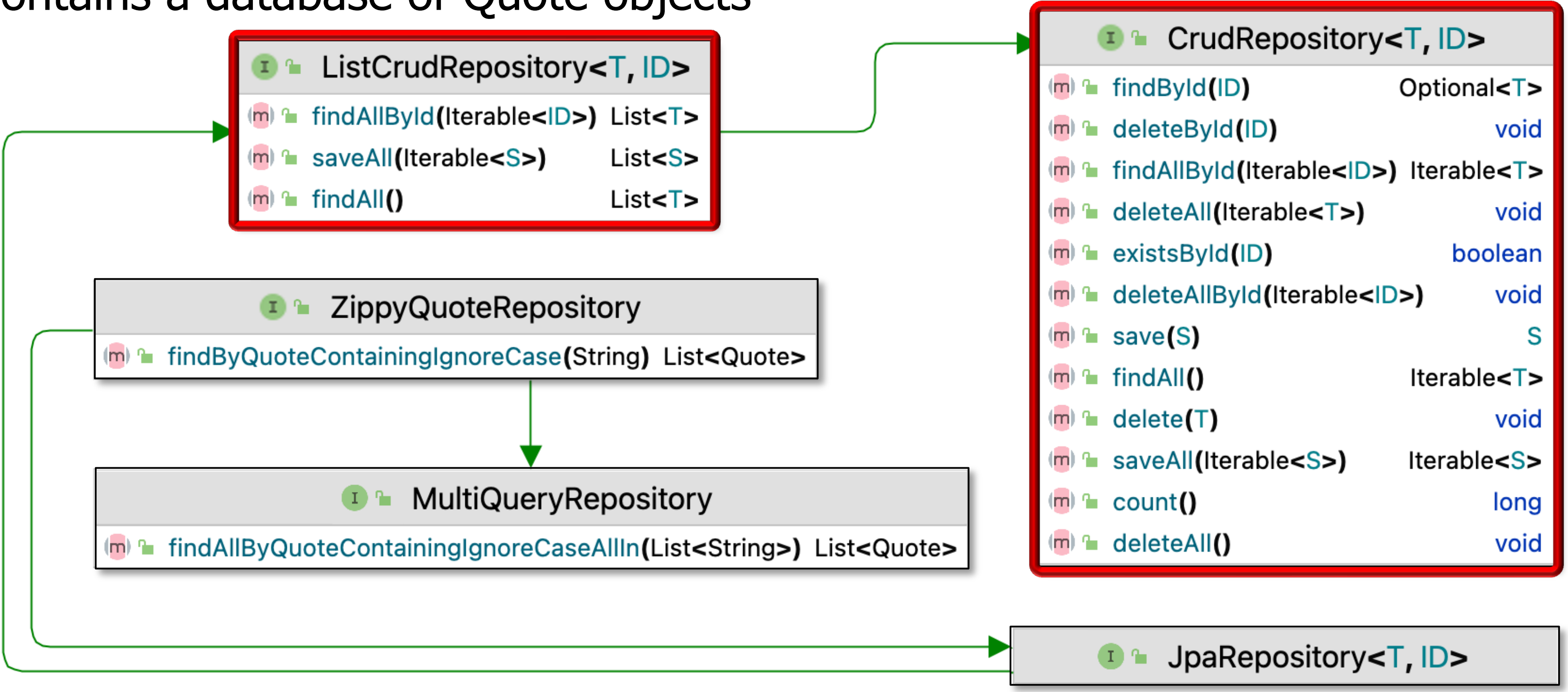


ListCrudRepository<T, ID>
findAllById(Iterable<ID>) List<T>
saveAll(Iterable<S>) List<S>
findAll() List<T>
CrudRepository<T, ID>
findById(ID) Optional<T>
deleteById(ID) void
findAllById(Iterable<ID>) Iterable<T>
deleteAll(Iterable<T>) void
existsById(ID) boolean
deleteAllById(Iterable<ID>) void
save(S) S
findAll() Iterable<T>
delete(T) void
saveAll(Iterable<S>) Iterable<S>
count() long
deleteAll() void
ZippyQuoteRepository
findByQuoteContainingIgnoreCase(String) List<Quote>
MultiQueryRepository
findAllByQuoteContainingIgnoreCaseAllIn(List<String>) List<Quote>
JpaRepository<T, ID>

See en.wikipedia.org/wiki/Jakarta_Persistence

Structure & Functionality of the ZippyQuoteRepository

- ZippyQuoteRepository is a Jakarta Persistence API (JPA)-based repository that contains a database of Quote objects



ListCrudRepository<T, ID>

findAllById(Iterable<ID>)	List<T>
saveAll(Iterable<S>)	List<S>
findAll()	List<T>

CrudRepository<T, ID>

findById(ID)	Optional<T>
deleteById(ID)	void
findAllById(Iterable<ID>)	Iterable<T>
deleteAll(Iterable<T>)	void
existsById(ID)	boolean
deleteAllById(Iterable<ID>)	void
save(S)	S
findAll()	Iterable<T>
delete(T)	void
saveAll(Iterable<S>)	Iterable<S>
count()	long
deleteAll()	void

ZippyQuoteRepository

findByQuoteContainingIgnoreCase(String)	List<Quote>

MultiQueryRepository

findAllByQuoteContainingIgnoreCaseAllIn(List<String>)	List<Quote>

JpaRepository<T, ID>

ZippyQuoteRepository inherits many reusable data access methods

Structure & Functionality of the ZippyQuoteRepository

- ZippyQuoteRepository is a Jakarta Persistence API (JPA)-based repository that contains a database of Quote objects

```
@Repository
public class ZippyQuoteRepository
       extends JPARepository<Quote, Integer>,
               MultiQueryRepository {




  List<Quote> findByQuoteContainingIgnoreCase(String query);
}
```

See quoteservices/microservices/zippy/ZippyQuoteRepository.java

Structure & Functionality of the ZippyQuoteRepository

- ZippyQuoteRepository is a Jakarta Persistence API (JPA)-based repository that contains a database of Quote objects

```
@Repository
public class ZippyQuoteRepository
       extends JPARepository<Quote, Integer>,
               MultiQueryRepository {

  List<Quote> findByQuoteContainingIgnoreCase(String query);
}
```

This annotation indicates the ZippyQuoteRepository class provides mechanisms for many storage, retrieval, search, update, & delete (CRUD) operation on objects

See springframework/stereotype/Repository.html

Structure & Functionality of the ZippyQuoteRepository

- ZippyQuoteRepository is a Jakarta Persistence API (JPA)-based repository that contains a database of Quote objects

```
@Repository
public class ZippyQuoteRepository
        extends JPARepository<Quote, Integer>,
                MultiQueryRepository {
```

Provides methods for basic CRUD operations (e.g., save(), findAll(), findById(), findAllById(), etc.) that enable interactions with a database while writing no SQL

```
  List<Quote> findByQuoteContainingIgnoreCase(String query);
}
```

See springframework/data/jpa/repository/JpaRepository.html

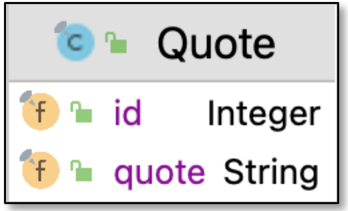
Structure & Functionality of the ZippyQuoteRepository

- ZippyQuoteRepository is a Jakarta Persistence API (JPA)-based repository that contains a database of Quote objects

```
@Repository
public class ZippyQuoteRepository
       extends JPARepository<Quote, Integer>,
               MultiQueryRepository {

  List<Quote> findByQuoteContainingIgnoreCase(String query);
}
```

'Quote' is the entity type managed by the repository & 'Integer' is the type of the primary key of the entity



Quote

id	Integer
quote	String

See [quoteservices/common/Quote.java]()

Structure & Functionality of the ZippyQuoteRepository

- ZippyQuoteRepository is a Jakarta Persistence API (JPA)-based repository that contains a database of Quote objects

```
@Repository
public class ZippyQuoteRepository
       extends JPARepository<Quote, Integer>,
               MultiQueryRepository



  List<Quote> findByQuoteContainingIgnoreCase(String query);
}
```

The SQL query generated by the Spring Data API selects all Quote rows in the database that contain the 'query' param (ignoring case)

See part 4 of this lesson on "*Zippy Microservice Structure & Functionality*"

Structure & Functionality of the ZippyQuoteRepository

- ZippyQuoteRepository is a Jakarta Persistence API (JPA)-based repository that contains a database of Quote objects

```
@Repository
public class ZippyQuoteRepository
       extends JPARepository<Quote, Integer>,
               MultiQueryRepository





  List<Quote> findByQuoteContainingIgnoreCase(String query);
}
```

```
                    'Yow! Am I cleansed yet?!'
                    'Yow! Am I in Milwaukee?'
       "yow"      { 'Yow! Are we in the perfect mood?'
                    'Yow! Is my fallout shelter termite proof?'
                    ...
```

The SQL query is "SELECT * FROM QUOTE WHERE LOWER(quote) LIKE LOWER('%{query}%')"

Structure & Functionality of the ZippyQuoteRepository

- ZippyQuoteRepository is a Jakarta Persistence API (JPA)-based repository that contains a database of Quote objects

```
@Repository
public class ZippyQuoteRepository
       extends JPARepository<Quote, Integer>,
               MultiQueryRepository {

  List<Quote> findByQuoteContainingIgnoreCase(String query);
}
```

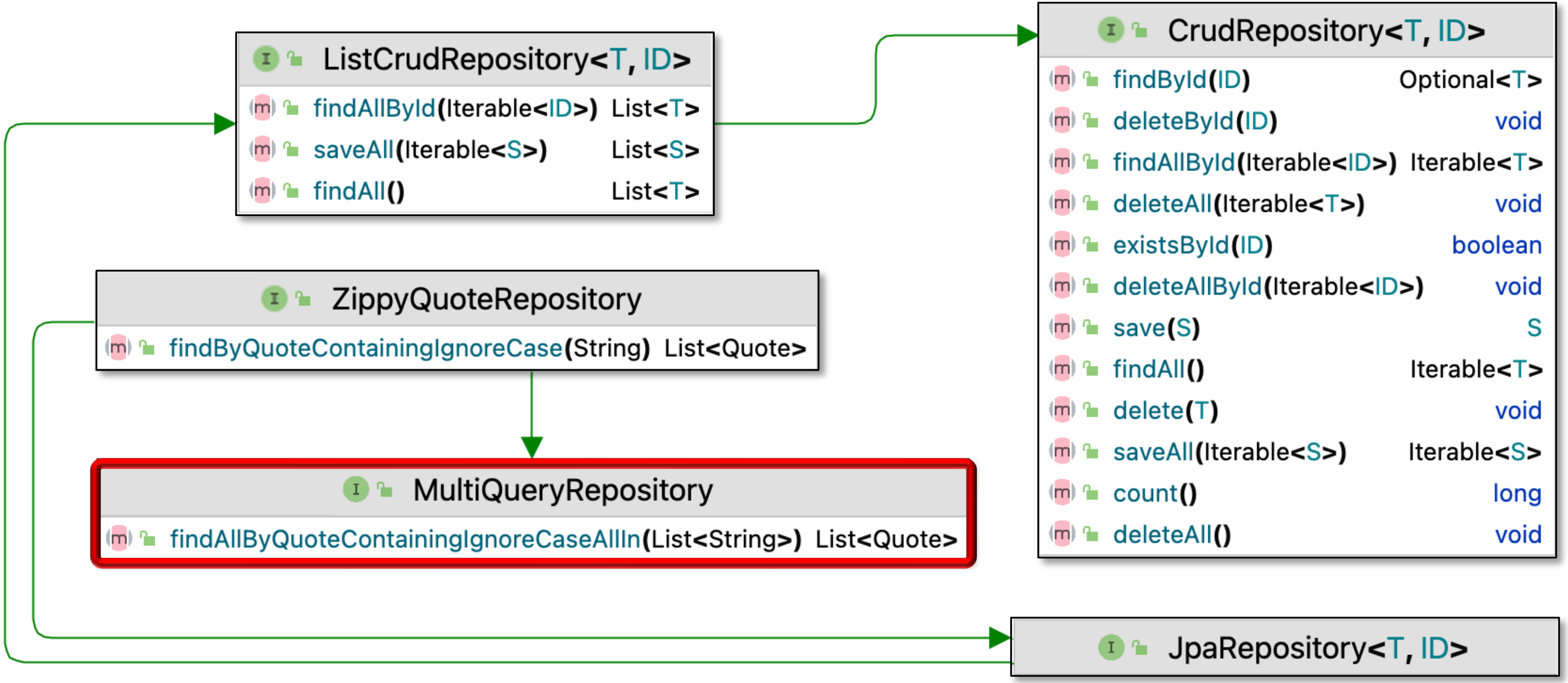
Defines a method that returns a List of Quote objects matching all the queries in a List of String objects

See part 3 of this lesson on "*Zippy Microservice Structure & Functionality*"

Structure & Functionality of the MultiQueryRepository*

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepository defines an interface for creating a custom query



ListCrudRepository<T, ID>
findAllById(Iterable<ID>) List<T>
saveAll(Iterable<S>) List<S>
findAll() List<T>
CrudRepository<T, ID>
findById(ID) Optional<T>
deleteById(ID) void
findAllById(Iterable<ID>) Iterable<T>
deleteAll(Iterable<T>) void
existsById(ID) boolean
deleteAllById(Iterable<ID>) void
save(S) S
findAll() Iterable<T>
delete(T) void
saveAll(Iterable<S>) Iterable<S>
count() long
deleteAll() void
ZippyQuoteRepository
findByQuoteContainingIgnoreCase(String) List<Quote>
MultiQueryRepository
findAllByQuoteContainingIgnoreCaseAllIn(List<String>) List<Quote>
JpaRepository<T, ID>

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepository defines an interface for creating a custom query

```
public interface MultiQueryRepository {
  ...
```

This interface defines enables clients to perform multiple queries at once on the Zippy Quote database

See microservices/src/main/java/edu/vandy/quoteservices/common/MultiQueryRepository.java

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepository defines an interface for creating a custom query

```
public interface MultiQueryRepository {
   ...

  List<Quote> findAllByQuoteContainingIgnoreCaseAllIn
                        (List<String> queries);
}
```

Find a List of Quote objects in the database containing all of the queries (ignoring case)

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepository defines an interface for creating a custom query

```
public interface MultiQueryRepository {
  ...




  List<Quote> findAllByQuoteContainingIgnoreCaseAllIn
                     (List<String> queries);
}
```

```
                       'Yow! Am I cleansed yet?!'
"yow" && "yet"    {    'Yow! Am I having fun yet?'
                       'Yow! Are we laid back yet?’
                       ...
```

SQL query is "SELECT * FROM QUOTE WHERE LOWER(quote) LIKE '%{yow}%' AND LOWER(quote) LIKE '%{yet}%'"

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepository defines an interface for creating a custom query

```
public interface MultiQueryRepository {
  ...



  List<Quote> findAllByQuoteContainingIgnoreCaseAllIn
                    (List<String> queries);
}
```

This method can't be generated automatically by the Spring Data API

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepositoryImpl class implements the custom query

```
public class MultiQueryRepositoryImpl
       implements MultiQueryRepository {
   ...
}
```

Applies the Spring "Repository Implementation" pattern

See chat.openai.com/share/4d3fc741-5d09-4d80-8b61-ef038d68e74a

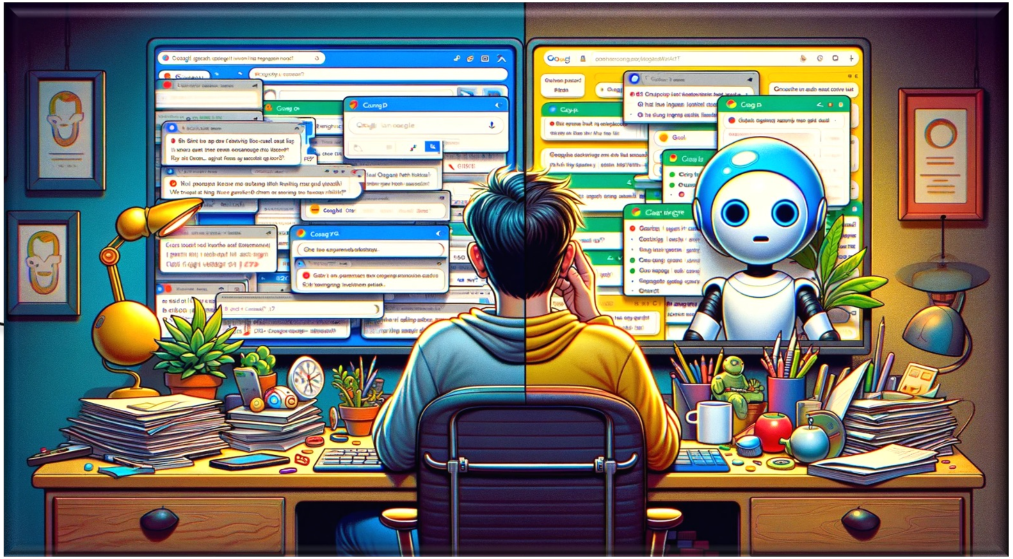
Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepositoryImpl class implements the custom query

```
public class MultiQueryRepositoryImpl
             implements MultiQueryRepository {
    ...


}
```

Again, good luck finding this detailed explanation using a Google search!



See chat.openai.com/share/4d3fc741-5d09-4d80-8b61-ef038d68e74a

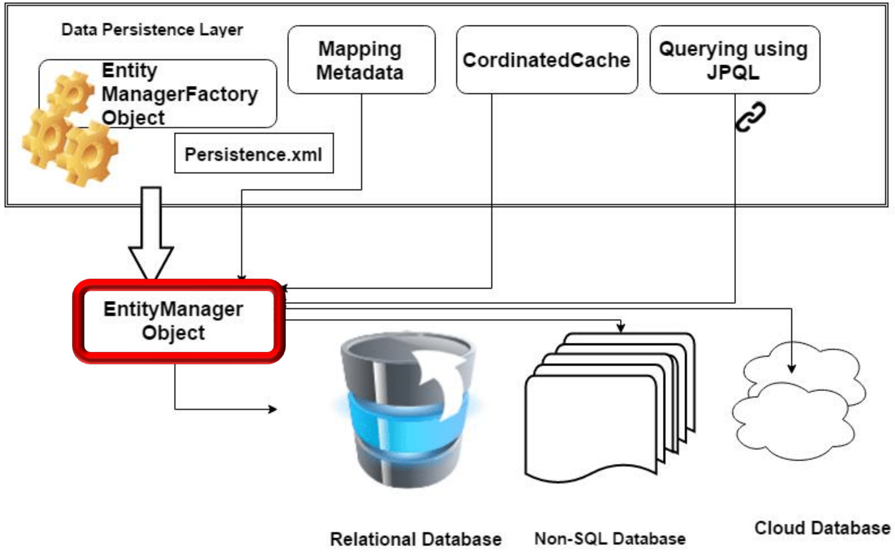
Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepositoryImpl class implements the custom query

```
public class MultiQueryRepositoryImpl
       implements MultiQueryRepository {
   @PersistenceContext
   private EntityManager mEntityManager;
   ...

}
```

Defines a database session for performing CRUD operations & querying the SQL database



See docs.oracle.com/javaee/7/api/javax/persistence/EntityManager.html

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepositoryImpl class implements the custom query

```
public class MultiQueryRepositoryImpl
       implements MultiQueryRepository {
   ...
 List<Quote> findAllByQuoteContainingIgnoreCaseAllIn
                (List<String> queries) {
     var criteriaBuilder = mEntityManager.getCriteriaBuilder();
     var criteriaQuery = criteriaBuilder
       .createQuery(Quote.class);
     var quote = criteriaQuery.from(Quote.class);
     var idExpression = criteriaBuilder
        .lower(quote.get("quote"));
     var andPredicate = ...
     return getQueryResults(criteriaQuery, andPredicate);
  ...
```

Find Quote objects containing all queries (ignoring case)

See upcoming lesson on “*Implementing the Zippy Microservice (Part 1)*”

End of the QuoteServices App Case Study: Zippy MicroService Structure & Functionality (Part 2)