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

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

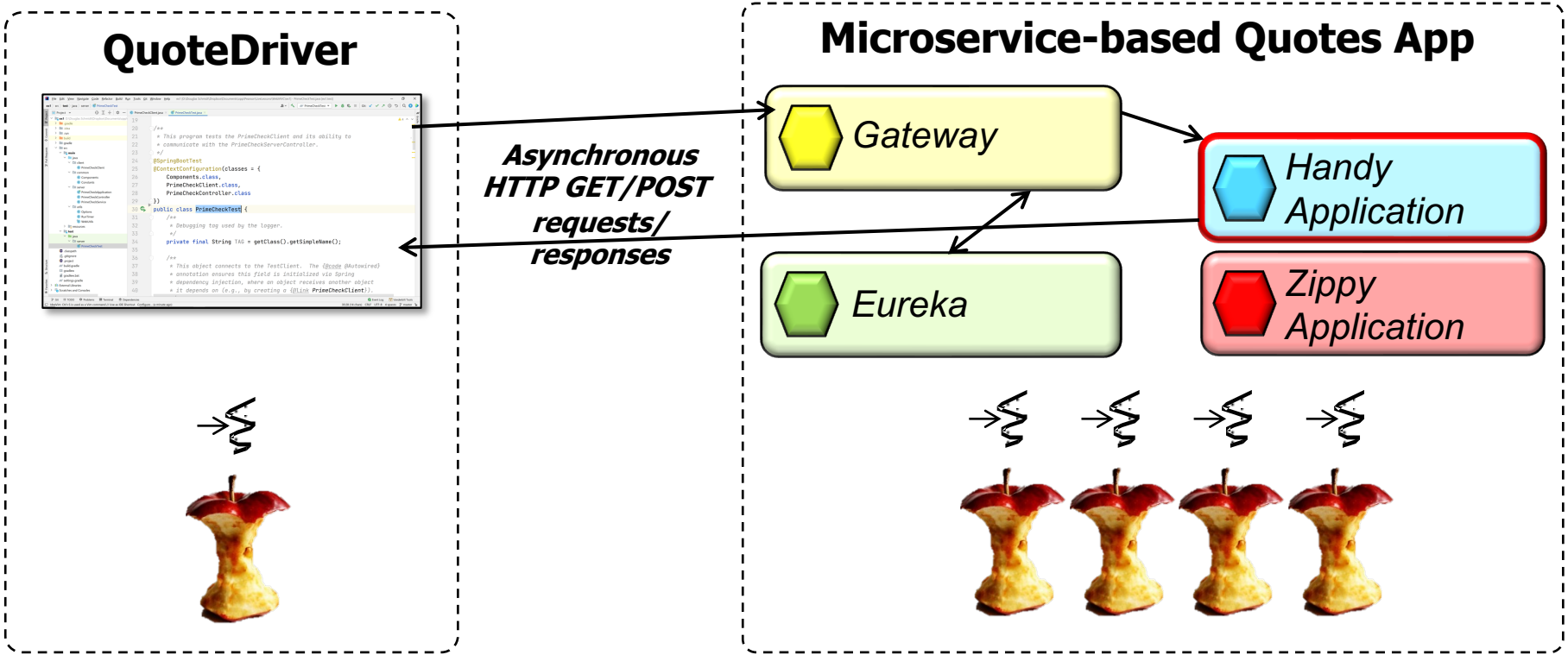
Professor of Computer Science

Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

- Understand the structure & functionality of the ReactiveQuoteRepository, MultiQueryRepository, & MultiQueryRepositoryImpl classes



This microservice uses R2DBC & Project Reactor reactive types

Structure & Functionality of the ReactiveQuoteRepository

Structure & Functionality of the ReactiveQuoteRepository

- Provides a persistent repository containing information about Quote objects using the R2DBC reactive database


```
@Repository
public interface ReactiveQuoteRepository
        extends ReactiveCrudRepository<Quote, Integer>,
                MultiQueryRepository {




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

See r2dbc.io

Structure & Functionality of the ReactiveQuoteRepository

- Provides a persistent repository containing information about Quote objects using the R2DBC reactive database

```
@Repository
public interface ReactiveQuoteRepository
        extends ReactiveCrudRepository<Quote, Integer>,
                MultiQueryRepository {




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

See quoteservices/repository/ReactiveQuoteRepository.java

Structure & Functionality of the ReactiveQuoteRepository

- Provides a persistent repository containing information about Quote objects using the R2DBC reactive database

```
@Repository
public interface ReactiveQuoteRepository
       extends ReactiveCrudRepository<Quote, Integer>,
               MultiQueryRepository {

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

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

See docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/stereotype/Repository.html

Structure & Functionality of the ReactiveQuoteRepository

- Provides a persistent repository containing information about Quote objects using the R2DBC reactive database

```
@Repository
public interface ReactiveQuoteRepository
       extends ReactiveCrudRepository<Quote, Integer>,
               MultiQueryRepository {

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

Interface for generic CRUD operations on a repository for a specific type that follows reactive paradigms & uses Project Reactor types built on top of Reactive Streams

See springframework/data/repository/reactive/ReactiveCrudRepository.html

Structure & Functionality of the ReactiveQuoteRepository

- Provides a persistent repository containing information about Quote objects using the R2DBC reactive database

```
@Repository
public interface ReactiveQuoteRepository
       extends ReactiveCrudRepository<Quote, Integer>,
               MultiQueryRepository {


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

This interface defines the means to perform multiple queries at once on the Quote database

See handeymicroservice/src/main/java/edu/vandy/quoteservices/repository/MultiQueryRepository.java

Structure & Functionality of the ReactiveQuoteRepository

- Provides a persistent repository containing information about Quote objects using the R2DBC reactive database

```
@Repository
public interface ReactiveQuoteRepository
        extends ReactiveCrudRepository<Quote, Integer>,
                MultiQueryRepository {

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

Asynchronously find all Quote rows in the data base that contain the query String (ignoring case)

Structure & Functionality of the ReactiveQuoteRepository

- Provides a persistent repository containing information about Quote objects using the R2DBC reactive database

```
@Repository
public interface ReactiveQuoteRepository
        extends ReactiveCrudRepository<Quote, Integer>,
                MultiQueryRepository {

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

This Spring Data API method implementation is automatically generated as
SELECT * FROM quote WHERE LOWER(quote) LIKE LOWER('%{query}%')

Structure & Functionality of the MultiQueryRepository*

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepository defines an interface for creating a custom query

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

This interface defines the means to perform multiple queries at once on the Quote database

See quoteservices/repository/MultiQueryRepository.java

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepository defines an interface for creating a custom query

```
public interface MultiQueryRepository {
   ...

   Flux<Quote> findAllByQuoteContainingIgnoreCaseAllIn
                                          (List<String> queries);



   Flux<Quote> findAllByQuoteContainingIgnoreCaseAnyIn
                                          (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 {
   ...


   Flux<Quote> findAllByQuoteContainingIgnoreCaseAllIn
                                          (List<String> queries);

   Flux<Quote> findAllByQuoteContainingIgnoreCaseAnyIn
                                          (List<String> queries);
}
```

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

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepository defines an interface for creating a custom query

```
public interface MultiQueryRepository {
   ...

   Flux<Quote> findAllByQuoteContainingIgnoreCaseAllIn
                                          (List<String> queries);

   Flux<Quote> findAllByQuoteContainingIgnoreCaseAnyIn
                                          (List<String> queries);
}
```

These methods 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 "Repository Implementation" pattern

See docs.spring.io/spring-data/jpa/docs/current/reference/html/#repositories.custom-implementations

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepositoryImpl class implements the custom query

```
public class MultiQueryRepositoryImpl
        implements MultiQueryRepository {
    @Autowired
    private DatabaseClient mDatabaseClient;
    ...
```

This field defines a database session that provides the main API for performing CRUD operations & querying the database

See springframework/r2dbc/core/DatabaseClient.html

Structure & Functionality of the MultiQueryRepository*

- The MultiQueryRepositoryImpl class implements the custom query

```
public class MultiQueryRepositoryImpl
       implements MultiQueryRepository {
   ...
 Flux<Quote> findAllByQuoteContainingIgnoreCaseAllIn
                  (List<String> queries) {
   String sql = buildQueryString
     ("SELECT * FROM quote WHERE LOWER(quote) LIKE :params",
      "%' AND LOWER(quote) LIKE '%",
      queries);


   return getQuoteFlux(sql);
}
```

Find Quote objects containing all queries (ignoring case)

See upcoming lesson on "*Implementing the Handey Microservice*"

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