



ОНЛАЙН-ОБРАЗОВАНИЕ



Курс «Разработчик на Spring Framework»

Проектная работа:

«Human Resources rEactive Files»

<https://github.com/vskurikhin/href-on-spring>



Содержание



Легенда

Легаси

Ракета многоступенчатая

Первое приближение



CI/CD

Нас много он один

Многоступенчатая реакция

Бесконечная бесконечность

Кормчий и Jenkins



Особенности и не только

Аналогия и разница

Один к одному, раз

Многое к одному, два

Баги и другие насекомые



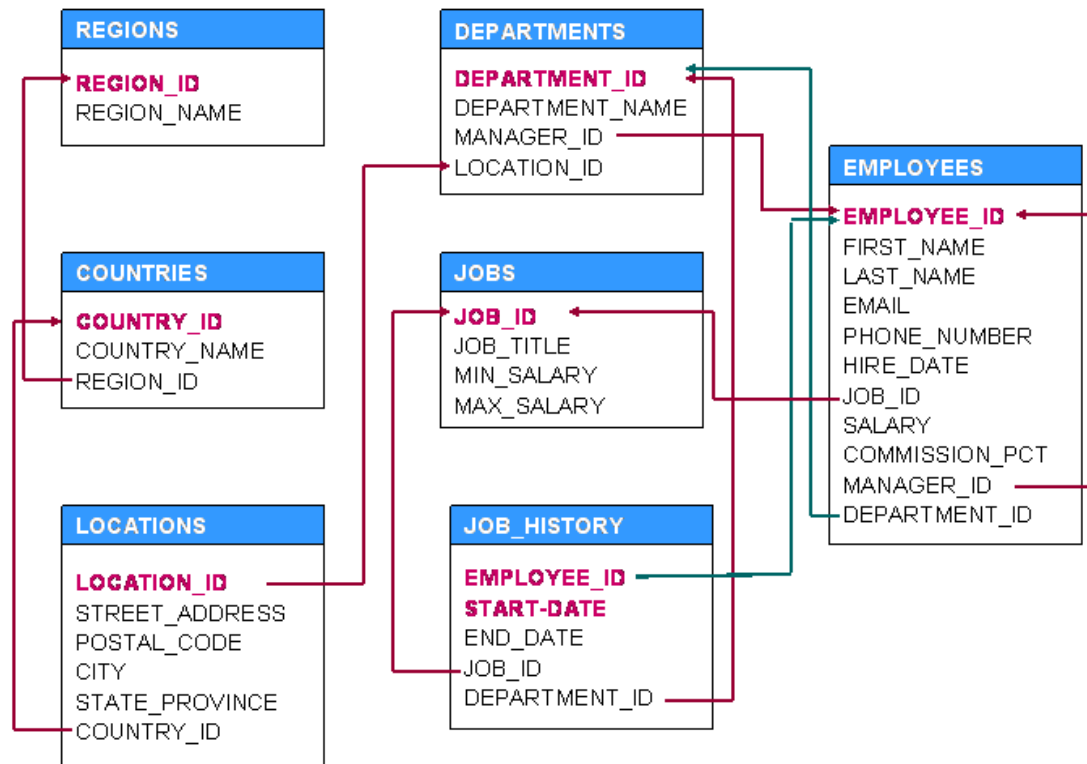
TODO

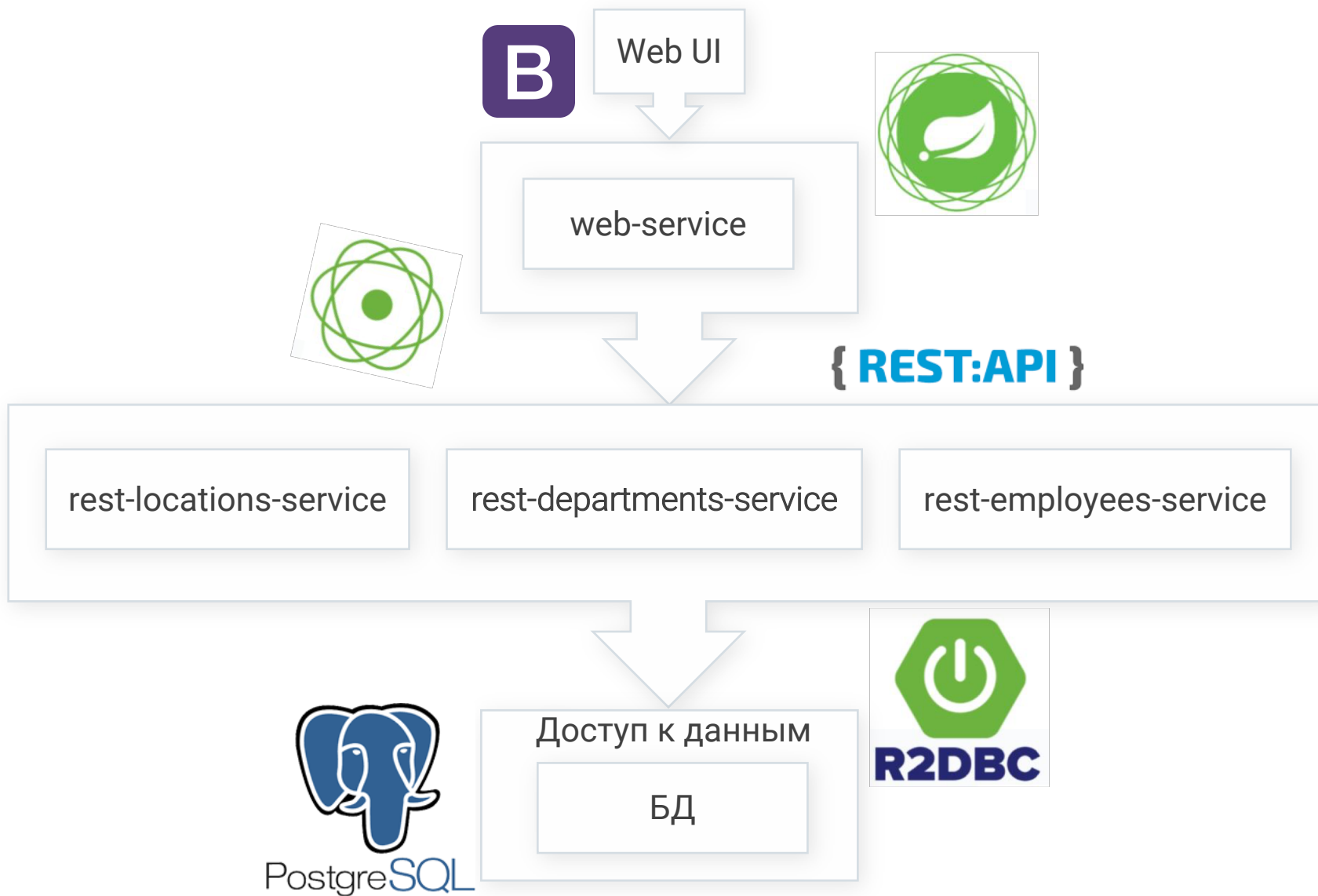
Ох уж эта безопасность

Технически должен

Вечный двигатель или ТУДУ

Легаси





```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-webflux</artifactId>
</dependency>
<dependency>
  <groupId>io.projectreactor</groupId>
  <artifactId>reactor-core</artifactId>
  <version>3.2.6.RELEASE</version>
</dependency>
<dependency>
  <groupId>io.projectreactor.addons</groupId>
  <artifactId>reactor-extra</artifactId>
  <version>3.2.2.RELEASE</version>
</dependency>
<!-- R2DBC -->
<dependency>
  <groupId>org.springframework.data</groupId>
  <artifactId>spring-data-r2dbc</artifactId>
  <version>1.0.0.M1</version>
</dependency>
<dependency>
  <groupId>io.r2dbc</groupId>
  <artifactId>r2dbc-spi</artifactId>
  <version>1.0.0.M7</version>
</dependency>
<dependency>
  <groupId>io.r2dbc</groupId>
  <artifactId>r2dbc-postgresql</artifactId>
  <version>1.0.0.M7</version>
</dependency>
```

```
package su.svn.href.dao;

import org.springframework.data.r2dbc.repository.query.Query;
import org.springframework.data.repository.reactive.ReactiveCrudRepository;
import reactor.core.publisher.Flux;
import su.svn.href.models.Region;

public interface RegionDao extends ReactiveCrudRepository<Region, Long>
{
    @Query("SELECT * FROM regions WHERE region_name = $1")
    Flux<Region> findByRegionName(String regionName);
}
```

```
public static String SELECT =
    "SELECT location_id, street_address, postal_code, city, state_province,"
    + " l.country_id, country_name, c.region_id, region_name"
    + " FROM locations l"
    + " JOIN countries c ON c.country_id = l.country_id"
    + " JOIN regions r ON r.region_id = c.region_id";

public Flux<LocationDto> findAll(
    int offset, int limit, String sortBy, boolean descending)
{
    String direction = descending ? " DESC" : " ASC";
    String orderBy = sortBy != null ? " ORDER BY " + sortBy : "";

    return databaseClient.execute()
        .sql(SELECT + orderBy + direction + " OFFSET $1 LIMIT $2")
        .bind("$1", offset)
        .bind("$2", limit)
        .fetch()
        .all()
        .map(LocationDto::collectFromMap);
}
```


начало...

```
public static String SELECT =  
"SELECT d.department_id, department_name, d.manager_id,"  
+ " m.first_name, m.last_name, m.email, m.phone_number,"  
+ " d.location_id, l.street_address, l.postal_code,"  
+ " l.city, l.state_province, l.country_id"  
+ " FROM departments d"  
+ " LEFT JOIN employees m ON m.employee_id = d.manager_id"  
+ " LEFT JOIN locations l ON l.location_id = d.location_id";
```

```
private Mono<DepartmentDto> departmentDtoMono(DepartmentDto departmentDto)  
{  
    return employeeDao  
        .findByDepartmentId(departmentDto.getId())  
        .collectList()  
        .map(e -> {  
            departmentDto.setEmployees(e);  
            return departmentDto;  
        });  
}
```

продолжение

```
@Override
public Mono<List<DepartmentDto>> findAll(
    int offset, int limit, String sortBy, boolean descending)
{
    String direction = descending ? " DESC" : " ASC";
    String orderBy = sortBy != null ? " ORDER BY " + sortBy : "";

    Comparator<DepartmentDto> comparator =
        Comparator.comparingLong(DepartmentDto::getId);

    return databaseClient.execute()
        .sql(SELECT + orderBy + direction + " OFFSET $1 LIMIT $2")
        .bind("$1", offset)
        .bind("$2", limit)
        .fetch()
        .all()
        .map(DepartmentDto::collectFromMap)
        .flatMap(this::departmentDtoMono)
        .collectSortedList(comparator);
}
```

```
@Override
public Flux<DepartmentDto> findAll(
    int offset, int limit, String sortBy, boolean descending)
{
    String direction = descending ? " DESC" : " ASC";
    String orderBy = sortBy != null ? " ORDER BY " + sortBy : "";

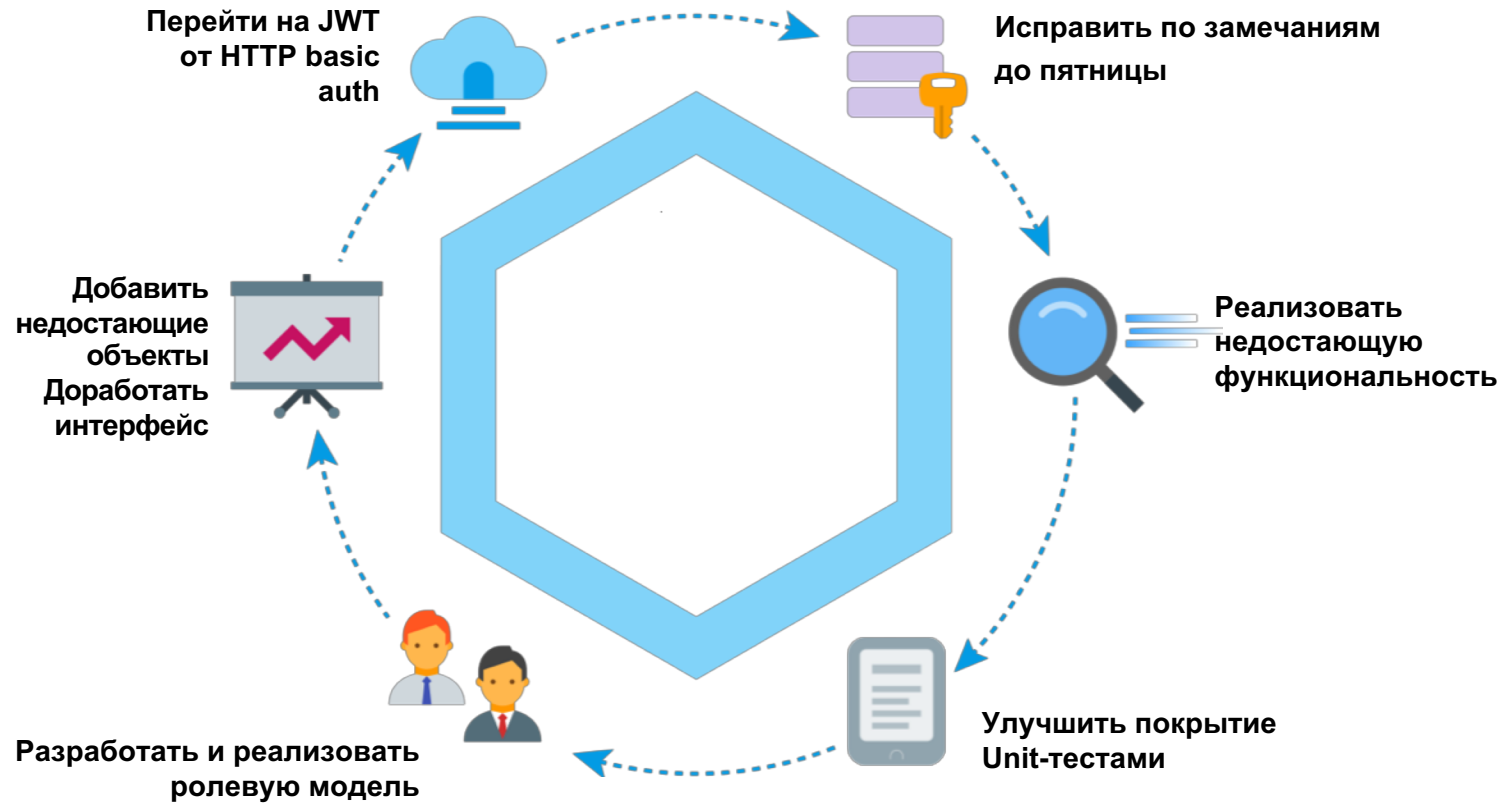
    return databaseClient.execute()
        .sql(SELECT + orderBy + direction + " OFFSET $1 LIMIT $2")
        .bind("$1", offset)
        .bind("$2", limit)
        .fetch()
        .all()
        .map(DepartmentDto::collectFromMap)
        .flatMap(departmentDto -> employeeDao
            .findByDepartmentId(departmentDto.getId())
            .collectList().map(employees -> {
                departmentDto.setEmployees(employees);
                return departmentDto;
            }));
}
```

```
public static String SELECT =  
+ " e.commission_pct, e.manager_id, e.department_id,"  
+ " m.employee_id AS manager_id, m.first_name AS manager_first_name, "  
+ " m.last_name AS manager_last_name,"  
+ " m.email AS manager_email, m.phone_number AS manager_phone_number,"  
+ " d.department_id, d.department_name, d.manager_id AS dept_manager_id, "  
+ " d.location_id"  
+ " FROM employees e"  
+ " LEFT JOIN employees m ON m.employee_id = e.manager_id"  
+ " LEFT JOIN departments d ON d.department_id = e.department_id";
```



- Jenkins Maven project
- Jenkins как pod в Kubernetes
- Plugin интеграции Jenkins с kubernetes
- Настройка Cloud Kubernetes
- Jenkins + Pipeline скрипт для kubernetes

Вечный двигатель или ТУДУ



- Исправить по замечаниям до пятницы
- Улучшить покрытие Unit-тестами
- Добавить недостающие объекты модели
- Реализовать недостающую функциональность для моделей
- Доработать интерфейс редактирования (JavaScript)
- Реализовать интерфейс добавления новых объектов
- Разработать и реализовать ролевую модель (Security)
- Перейти на JWT от HTTP basic auth
- Что дальше?

Вопросы?

**Спасибо
за внимание!**

