Exemplos de implementações no SpringBoot

Exmplo do uso de DTOs (Service)

Para retornar um elemento

```
public UsuarioResponse buscaUsuarioAtivobyId(int id) {
        Optional <Usuario>  usuario = usuarioRepository.buscarUsuariosAtivosPorId(id);
        if(usuario !=null) {
                UsuarioResponse response = new UsuarioResponse();
                response.setId(Integer.toString(usuario.get().getId()));
                response.setCpf(usuario.get().getCpf());
                response.setEmail(usuario.get().getEmail());
                response.setNomeCompleto(usuario.get().getNomeCompleto());
                response.setTipo(Integer.toString(usuario.get().getTipo()));
                response.setStatus(Integer.toString(usuario.get().getStatus()));
                response.setEmpresaId(Integer.toString(usuario.get().getEmpresa().getId()));
                return response;
        }
        else {
                return new UsuarioResponse();
        }

}
```

Para retornar vários elementos

```
public List<SolicitacaoAnaliseJuridico> buscaSolicitacaoAnaliseJuridicobyTitulo(String titulo) {

	List<SolicitacaoAnaliseJuridico> responses = new ArrayList<SolicitacaoAnaliseJuridico>();

	List<SolicitacaoAnaliseJuridico> solicitacaoAnaliseJuridicos = solicitacaoAnaliseJuridicoRepository.buscarSolicitacaoAnaliseJuridicoAtivosPorTitulo(titulo);

	if(solicitacaoAnaliseJuridicos !=null) {

	for (int i=0; i<solicitacaoAnaliseJuridicos.size(); i++){
		SolicitacaoAnaliseJuridico response = new SolicitacaoAnaliseJuridico();
			response.setId(solicitacaoAnaliseJuridicos.get(i).getId());
			response.setNumero(solicitacaoAnaliseJuridicos.get(i).getNumero());
			response.setDataRegistro(solicitacaoAnaliseJuridicos.get(i).getDataRegistro());
			response.setTitulo(solicitacaoAnaliseJuridicos.get(i).getTitulo());
			response.setObservacao(solicitacaoAnaliseJuridicos.get(i).getObservacao());
			response.setStatus(solicitacaoAnaliseJuridicos.get(i).getStatus());
			response.setSolicitacaoTipo(solicitacaoAnaliseJuridicos.get(i).getSolicitacaoTipo());
			response.setUsuario(solicitacaoAnaliseJuridicos.get(i).getUsuario());
			response.setPrioridade(solicitacaoAnaliseJuridicos.get(i).getPrioridade());

			responses.add(response);
	}
	}
	return responses;
}
```

Exmplo de CRUD completo (Controller)

```java
@GetMapping
public ResponseEntity<List<SolicitacaoPrioridade>> obterSolicitacaoPrioridade(){
        List<SolicitacaoPrioridade> solicitacaoPrioridade = solicitacaoPrioridadeService.buscarSolicitacaoPrioridadeAtivas();
        return new ResponseEntity<>(solicitacaoPrioridade, HttpStatus.OK);
}

@GetMapping("/buscar/{id}")
public ResponseEntity<SolicitacaoPrioridade> obterSolicitacaoPrioridadePorId(@PathVariable("id") Integer id){
        SolicitacaoPrioridade solicitacaoPrioridade = solicitacaoPrioridadeService.buscaSolicitacaoPrioridadebyId(id);
        return new ResponseEntity<>(solicitacaoPrioridade, HttpStatus.OK);
}

@PostMapping("/adicionar")
public ResponseEntity<SolicitacaoPrioridade> adicionarSolicitacaoPrioridade(@RequestBody SolicitacaoPrioridade solicitacaoPrioridade){
        SolicitacaoPrioridade novaSolicitacaoPrioridade = solicitacaoPrioridadeService.adicionarSolicitacaoPrioridade(solicitacaoPrioridade);
        return new ResponseEntity<>(novaSolicitacaoPrioridade, HttpStatus.CREATED);
}

@PutMapping("/atualizar")
public ResponseEntity<SolicitacaoPrioridade> atualizarSolicitacaoPrioridade(@RequestBody SolicitacaoPrioridade solicitacaoPrioridade){
        SolicitacaoPrioridade atualizarSolicitacaoPrioridade = solicitacaoPrioridadeService.atualizarSolicitacaoPrioridade(solicitacaoPrioridade);
        return new ResponseEntity<>(atualizarSolicitacaoPrioridade, HttpStatus.OK);
}

@DeleteMapping("/excluir/{id}")
public ResponseEntity<?> excluirSolicitacaoPrioridade(@PathVariable("id") Integer id){
        solicitacaoPrioridadeService.excluirSolicitacaoPrioridade(id);
        return new ResponseEntity<>(HttpStatus.OK);
}
```

Na **Adição** e **Atualização** precisamos receber um JSON com as informações necessárias

Exemplos de métodos no Service

```java
@Service
public class EmpresaService {
	private final EmpresaRepository empresaRepository;

	@Autowired
	public EmpresaService(EmpresaRepository empresaRepository) {
		this.empresaRepository = empresaRepository;
	}

	public Empresa adicionarEmpresa(Empresa empresa) {
		return empresaRepository.save(empresa);
	}

	public Empresa atualizarEmpresa(Empresa empresa) {
		return empresaRepository.save(empresa);
	}

	public void apagarEmpresa(int id) {
		empresaRepository.apagarEmpresaPorId(id);
	}

	public Empresa buscaEmpresabyId(int id) {
		return empresaRepository.buscarEmpresaAtivasPorId(id)
			.orElseThrow(() -> new EmpresaNotFoundException ("Empresa id "+ id + "não foi encontrada!"));
	}

	public List<Empresa> buscarEmpresasAtivas(){
		return empresaRepository.buscarEmpresasAtivas();
	}
}
```

Exemplos de métodos no Repository

```java
public interface AtividadeRepository extends JpaRepository<Atividade, Long> {

    @Query("SELECT a from Atividade a WHERE a.status != -1")
    List<Atividade> getActiveAtividades();
}

public interface BadgeRepository extends JpaRepository<Badge, Integer> {
    List<Badge> findByBadgeTipoId(Integer badgeTipoId);

    List<Badge> findByBadgeNivelId(Integer badgeNivelId);


}
```

```java
@Query(value = "SELECT * FROM EMPRESA WHERE EMPRESA_STATUS >= 0", nativeQuery = true)
    List<Empresa> buscarEmpresasAtivas();

@Query(value = "SELECT * FROM EMPRESA WHERE EMPRESA_ID = :id AND EMPRESA_STATUS >= 0", nativeQuery = true)
    Optional<Empresa> buscarEmpresaAtivasPorId(int id);

@Modifying
@Transactional
@Query(value = "UPDATE empresa SET empresa_status = -1 WHERE empresa_id = :id", nativeQuery = true)
    void apagarEmpresaPorId(@Param("id") Integer id);

@Modifying
@Transactional
@Query(value = "UPDATE empresa SET empresa_status = 0 WHERE empresa_id = :id", nativeQuery = true)
    void desativarEmpresaPorId(@Param("id") Integer id);
```

```java
@Repository
public interface ArquivoRepository extends JpaRepository<Arquivo, String>{

	@Query(value = "SELECT * FROM arquivo WHERE arquivo_id = :id AND arquivo_status >= 0", nativeQuery = true)
	Optional<Arquivo> buscarArquivoAtivosPorId(int id);

	@Query(value = "SELECT * FROM arquivo WHERE arquivo_status >= 0", nativeQuery = true)
	List<Arquivo> buscarArquivoAtivos();

	@Modifying
	@Transactional
	@Query(value = "UPDATE arquivo SET arquivo_status = -1 WHERE arquivo_id = :id", nativeQuery = true)
	void apagarArquivoPorId(@Param("id") Integer id);
}
```

Delete (Controller)

```java
@DeleteMapping("/apagar/{id}")
public ResponseEntity<?> excluirArquivo(@PathVariable("id") Integer id){
        arquivoService.excluirArquivo(id);
        return new ResponseEntity<>(HttpStatus.OK);
}
```