Pagination in Spring JPA – 2022

**Entity Layer**

@Entity(name = "Tutorial")

@Table(name = "tutorials")

@Getter

@Setter

**public** **class** Tutorial {

@Id

@GeneratedValue

**private** **long** id;

@Column(name = "title")

**private** String title;

@Column(name = "description")

**private** String description;

**public** Tutorial() {

}

**public** Tutorial(String title, String description) {

**this**.title = title;

**this**.description = description;

}

}

@Data

**public** **class** PageTutorials {

**private** **long** totalElements;

**private** List<Tutorial> tutorials;

}

**Repository Layer**

@Repository

**public** **interface** PaginatedRepo **extends** **PagingAndSortingRepository<Tutorial, Long>** {

**@Query(value = "SELECT \* FROM tutorials WHERE title = ?1",**

**countQuery = "SELECT count(\*) FROM tutorials WHERE title = ?1",**

**nativeQuery = true)**

Page<Tutorial> getInfoByTitle(String title, Pageable pageable);

Page<Tutorial> findByTitle(String title, Pageable pageable);

}

**Service Layer**

**public** **interface** TutorialService {

**void** saveAllTutotials(List<Tutorial> tutorials);

PageTutorials getAllTutorialsByPage(**int** page, **int** size);

PageTutorials getAllTutorialsByTitlePage(String title, **int** page, **int** size);

}

**Service Implementation Layer**

@Service

**public** **class** TutorialServiceImpl **implements** TutorialService {

@Autowired

**private** PaginatedRepo repo;

@Override

**public** **void** saveAllTutotials(List<Tutorial> tutorials) {

repo.saveAll(tutorials);

}

@Override

**public** PageTutorials getAllTutorialsByPage(**int** page, **int** size) {

PageTutorials pagedTutorials = **new** PageTutorials();

**Pageable pageable = PageRequest.*of*(page, size);**

Page<Tutorial> pageTutorial = repo.findAll(pageable);

**long totalNoOfData = pageTutorial.getTotalElements();**

**pagedTutorials.setTotalElements(totalNoOfData);**

**if (pageTutorial.hasContent()) {**

**pagedTutorials.setTutorials(pageTutorial.getContent());**

**}**

**return** pagedTutorials;

}

@Override

**public** PageTutorials getAllTutorialsByTitlePage(String title, **int** page, **int** size) {

PageTutorials pagedTutorials = **new** PageTutorials();

Pageable pageable = PageRequest.*of*(page, size);

// Pageable pageable = PageRequest.of(page, size, Sort.DEFAULT\_DIRECTION, "title");

// Page<Tutorial> pageTutorial = repo.findByTitle(title, pageable);

Page<Tutorial> pageTutorial = repo.findByTitle(title, pageable);

**long** totalNoOfData = pageTutorial.getTotalElements();

pagedTutorials.setTotalElements(totalNoOfData);

**if** (pageTutorial.hasContent()) {

pagedTutorials.setTutorials(pageTutorial.getContent());

}

**return** pagedTutorials;

}

}

**Spring Main Application**

@SpringBootApplication

**public** **class** MainApp {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(MainApp.**class**, args);

}

}

**AutoRun Component**

@Component

**public** **class** AutoRun {

@Autowired

**private** TutorialService service;

@EventListener(ApplicationReadyEvent.**class**)

**public** **void** afterStartup() {

List<Tutorial> tutorials = **new** ArrayList<>();

**for** (**int** i = 0; i < 100; i++) {

Tutorial tutorial = **new** Tutorial("Title "+i, "Description "+i);

tutorials.add(tutorial);

}

service.saveAllTutotials(tutorials);

}

}