Dokumentacion I metodave qe perdor.

1) Filter Generic me Criteria Builder → August 2021 IkubINFO

|  |
| --- |
| public class *Filters* {   private final *EntityManager* entityManager;  *@Autowired* public Filters(*EntityManager entityManager*) {  this.entityManager = *entityManager*; }  public *List* genericFilter(*Class c*, *SearchCriteria searchCriteria*, *String fetch*) {   *CriteriaBuilder* cb = entityManager.getCriteriaBuilder();  *CriteriaQuery* cq = cb.createQuery(*c*);  *Root* root = cq.from(*c*);  *List*<*Predicate*> predicateList = new ArrayList<>();  *List*<*Field*> fields = *Arrays*.*asList*(*c*.getDeclaredFields());   if (isStringValid(*fetch*)) {  if (isFieldPresent(fields, *fetch*)) {  root.fetch(*fetch*, *JoinType*.INNER);  }  }  if (*searchCriteria* == null) {  return entityManager.createQuery(cq.select(root)).getResultList();  }   if (*searchCriteria*.getFieldValuePair() != null) {  for (*FieldValuePair* fieldValuePair : *searchCriteria*.getFieldValuePair()) {  if (isFieldPresent(fields, fieldValuePair.getField())) {  if (!isStringValid(fieldValuePair.getOperation())) {  if (fieldValuePair.getField() != null && fieldValuePair.getValue() != null) {  log.info("Field {} is present ", fieldValuePair.getField());  predicateList.add(  isBool(fieldValuePair.getValue()) ?  cb.equal(root.get(fieldValuePair.getField()), *Boolean*.*valueOf*(fieldValuePair.getValue()))  :  cb.equal(root.get(fieldValuePair.getField()), fieldValuePair.getValue())  );  }  } else {  if (!isBool(fieldValuePair.getValue())) {  if (fieldValuePair.getOperation().compareToIgnoreCase("greaterThanOrEqualTo") == 0) {  predicateList.add(cb.greaterThanOrEqualTo(root.get(fieldValuePair.getField()), fieldValuePair.getValue()));  } else if (fieldValuePair.getOperation().compareToIgnoreCase("lessThanOrEqualTo") == 0) {  predicateList.add(cb.lessThanOrEqualTo(root.get(fieldValuePair.getField()), fieldValuePair.getValue()));  }  }   }   }  }  }  cq.select(root).where(predicateList.toArray(new Predicate[predicateList.size()]));   log.info("Order by {} Sort dir {}", *searchCriteria*.getOrderBy(), *searchCriteria*.getSortDirection());   if (isStringValid(*searchCriteria*.getOrderBy())) {  log.info("Order by ain't blank {} " + *searchCriteria*.getOrderBy());  if (isFieldPresent(fields, *searchCriteria*.getOrderBy())) {  log.info("Field is present {}" + *searchCriteria*.getOrderBy());  if (isStringValid(*searchCriteria*.getSortDirection())) {  log.info("Sort direction isn't blank");  cq.orderBy(  isDirAsc(*searchCriteria*.getSortDirection())  ?  cb.asc(root.get(*searchCriteria*.getOrderBy()))  :  cb.desc(root.get(*searchCriteria*.getOrderBy()))  );  } else {  cq.orderBy(  cb.desc(root.get(*searchCriteria*.getOrderBy()))  );  }   }  }  return entityManager.createQuery(cq).getResultList(); }  private boolean isFieldPresent(*List*<*Field*> *existing*, *String name*) {  for (*Field* field : *existing*) {  if (field.getName().compareTo(*name*) == 0) {  return true;  }  }  return false; }  private boolean isStringValid(*String s*){  return *s* != null && !*s*.isBlank(); }  private boolean isBool(*String s*){  return *s*.compareToIgnoreCase("true")==0 || *s*.compareToIgnoreCase("false")==0; }  private boolean isDirAsc(*String s*){  return *s*.compareToIgnoreCase("asc")==0; }  } |

Klasa Search Criteria:

|  |
| --- |
| *@Getter* *@Setter* *@NoArgsConstructor* *@ToString* public class *SearchCriteria* {  private *List*<*FieldValuePair*> fieldValuePair;  private *String* orderBy;  private *String* SortDirection;  } |

Klasa FieldValuePair:

|  |
| --- |
| *@Getter* *@Setter* *@NoArgsConstructor* public class *FieldValuePair* {  private *String* field;  private *String* value;  private *String* operation;  } |

FieldValuePair perdoret per te mbajtur ciftet key:value per te bere pjesen where ne query si dhe operacionin (veprimin qe do kryhet mbi to pra greater than a value or less than a value). Pra where name=’joan’. Pjesa Search Criteria mban liste me field value pair dhe parametrat e nevojshem per sortimin.

Metodat private ne klase kane per qellim berjen e kodit me te lexueshem me te mirembajtshem me generic dhe me te riperdorshem.

Metoda e filtrit eshte gjenerike mund te perdoret me cdo klase me cdo entitetet pavaresisht tipit te klaseses apo projektit. Mjafton te merret e te vihet keshtu sic eshte dhe te perdoret.

Fetch perdoret per te bere fetch ato qe jane lazily initialized nqs kerkohet.

Workflow:

1) Merret criteria builder nga entitiy manager (menaxhon persistance conetext)

2) Krjohet query per klasen e kaluar si argument ( Smthing.class kalohet si argument, nga kjo mund te merren fushat qe ka)

3) Krijohet rooti I querit (root in from clause)

4) Krijohet nje arrayliste qe do te mbaje nje numer te percaktuar nga perdoruesi predicates ose kushtet. Perdoret per te krijuar where clauses

5) Meren fushat e klases

6) Kontrollohet nqs fushat qe ka dhene useri gjenden te kjo klase dhe nqs po jane valide apo jo (not null not blank)

7) Nqs cdo gje ok, shtohet fusha ne predicate list ne formen enje predicate,. (Nqs eshte booleane ka nevoje per konvertim per te shmangur exceptionat ne runtime). Nqs kemi dhe fushen operation qe nuk eshte null atehere kontrollohet me cfare ngjason operacioni greater then, less then dhe me pas shtohet. Kjo pjese mund te shtohet hiqet ne kod sipas kerkeses.

8) krijohet query me predicated e deritanishem

cq.select(root).where(predicateList.toArray(new Predicate[predicateList.size()]));

e nevojshme te konvertohet ne array sepse ate tip pranon select.where

9) E njejta procedure ndiqet edhe per orderat. Check if valid check cila eshte asc apo desc dhe shtoja querit.

cq.orderBy(  
 isDirAsc(*searchCriteria*.getSortDirection())  
 ?  
 cb.asc(root.get(*searchCriteria*.getOrderBy()))  
 :  
 cb.desc(root.get(*searchCriteria*.getOrderBy()))  
);

Nqs eshte zgejdhur nje order by por asnje sort, by default zgjedhim descending sidrejtim cb.desc.Ne fund ekzkeutohet query nga entity manager dhe merret rezultari