Package com.kimberly.location;

Import mongodb.WriteResult;

Import org.springframework.beans.factory.annotation.Autowired;

Import org.springframework.data.mongodb.core.MongoTemplate;

Import org.springframework.data.mongodb.core.query.Criteria;

Import org.springframework.data.mongodb.core.query.Query;

Import org.springframework.data.mongodb.core.query.Update;

Public class LocationDomainRepositoryImpl implements LocationDomainRepository {

@Autowired

LocationRepository locationRepository;

Public Boolean addLocation(Location location) {

// MongoClient mongoClient = new MongoClient();

//MongoDatabase db = mongoClient.getDatabase(databaseName);

MongoCollection<Document> collection = db.getCollection(“Location”);

ObjectMapper mapper = new ObjectMapper();

String json;

Try {

Json = mapper.writeValueAsString(location);

Document document = Document.parse(json);

collection.insertOne(document);

} catch (MonoWriteException ex) {

Status = false;

} catch (IOException io) {

Status = false;

}

Return status;

}

@Override

Public Boolean updateById(Location location) {

MongoClient mongoClient = new MongoClient();

MongoDatabase db = mongoClient.getDatabase(databaseName);

MongoCollection<Document> collection = db.getCollection(“Location”);

Bson filter = null;

Bson query = null;

Try {

BasicDBObject document

= BaseicDBObject.parse(new ObjectMapper().writeValueAsString(location));

Bson newDocument = new Document(“$set”, document);

filter = eq(“ID”, location.getID());

UpdateResult result = collection.updateOne( filter,

newDocument,

(new UpdateOptions()).upsrt(true));

} catch (MongoException ex) {

Status = false;

} catch (IOException io) {

Status = false;

}

Return status;

}

@Override

Public Boolean deleteLocationById(int id) {

Try {

//MongoClient mongoClient = new MongoClient();

// MongoDatabase db = mongoClient.getDatabase(databaseName);

MongoCollection<Document> collection = db.getCollection(“Location”);

Bson filter = eq(“id”, id);

DeleteResult result = collection.deleteOne(filter);

If (result.getDeletedCount() == 1) {

Return true;

} else {

Return false;

}

} catch (MongoException e) {

System.output.print(“Mongogo exception message:” +e);

}

}

@Override

Public List<Location> findAllByCurrLocation(double paramLat, double paramLng) {

MongoClient mongoClient = new MongoClient();

MongoDatabase db = mongoClient.getDatabase(databaseName);

MongoCollection<Document> collection = db.getCollection(“Location”);

db.collection.find( {

$and : [

{ $and : [ { lat: { $gte: paramLat }, { $lte: 1Mile} ] },

{ $and : [{ lng: {$gte: paramLng}, {$lte: 1Miles} ] }

]

})

Return LocationDao.findAllLocationsByCurrLocation( lat, lng);

}

}