|  |
| --- |
| Entity |
| // @Entity  @Data  public class User      implements EntityInterface<User, Long>  {      private Long id;      private String username;      private String password;      private String email;      private String sodienthoai;      private Boolean enabled;        @Override      public void updateNotId(User o2) {          this.id = o2.getId();          this.username = o2.getUsername();          this.password = o2.getPassword();          this.email = o2.getEmail();          this.sodienthoai = o2.getSodienthoai();          this.enabled = o2.getEnabled();      }      public User(String username, String password, String email, String sodienthoai, Boolean enabled) {          this.username = username;          this.password = password;          this.email = email;          this.sodienthoai = sodienthoai;          this.enabled = enabled;      }      public User(Long id, String username, String password, String email, String sodienthoai, Boolean enabled) {          this.id = id;          this.username = username;          this.password = password;          this.email = email;          this.sodienthoai = sodienthoai;          this.enabled = enabled;      }      public User() {      }    } |

|  |
| --- |
| RowMapper |
| public class UserRowMapper      implements RowMapper<User>  {      @Override      @Nullable      public User mapRow(ResultSet ketQuaTraVe, int sttHang) throws SQLException {          Long id             = ketQuaTraVe.getLong   ("id");          String username     = ketQuaTraVe.getString ("username");          String password     = ketQuaTraVe.getString ("password");          String email        = ketQuaTraVe.getString ("email");          String sodienthoai  = ketQuaTraVe.getString ("sodienthoai");          Boolean enabled     = ketQuaTraVe.getBoolean("enabled");          User user1 = new User(id, username, password, email, sodienthoai, enabled);            return user1;      }  } |

|  |
| --- |
| UserRepo |
| @Repository  public class UserRepo      implements RepoInterface<User, Long>  {      @Autowired      private JdbcTemplate jdbcTemplate;      private UserRowMapper userRowMapper = new UserRowMapper();      @Override      public User findById(Long tKey) {          String sql= "SELECT \* FROM [User] WHERE id= ? ";          PreparedStatementCreator psCreator = con->{              PreparedStatement ps = con.prepareStatement(sql);              ps.setLong(1, tKey);              return ps;          };          return jdbcTemplate.query(psCreator, this.userRowMapper).stream().findFirst().orElse(null);      }      @Override      public List<User> getList(Integer sttPage, Integer sizePage) {          String sql =              "SELECT "              + " DISTINCT "              + " \* "              + " FROM [User] "              + " WHERE 1=1 "              + " ORDER BY id DESC "              + " OFFSET ? ROW "              + " FETCH NEXT ? ROW ONLY ";            PreparedStatementCreator psCreator = con -> {              PreparedStatement ps = con.prepareStatement(sql);              ps.setInt(1, ( (sizePage-1)\*sttPage ) );              ps.setInt(2, sizePage);              return ps;          };          return jdbcTemplate.query(psCreator, userRowMapper);      }      @Override      public User create(User t) {          t.setId(null);          String sql=              " INSERT INTO [User] "              + " (username, password, email, sodienthoai, enabled)"              + " VALUES ( ?, ?, ?, ?, ? )";          KeyHolder keyHolder = new GeneratedKeyHolder();          PreparedStatementCreator psCreator = con -> {              PreparedStatement ps = con.prepareStatement(sql, Statement.RETURN\_GENERATED\_KEYS);              // điền tham số vào dấu ?              ps.setString(1, t.getUsername());              ps.setString(2, t.getPassword());              ps.setString(3, t.getEmail());              ps.setString(4, t.getSodienthoai());              ps.setBoolean(5, t.getEnabled());              return ps;          };          jdbcTemplate.update(psCreator, keyHolder);          Long id = keyHolder.getKey().longValue();            return this.findById(id);      }      @Override      public User update(Long tKey, User t) {          t.setId(tKey);            String sql =              " UPDATE [User] "              + " SET "                  + " username = ? , "                  + " password = ? , "                  + " email = ? , "                  + " sodienthoai = ? , "                  + " enabled = ? "              + " WHERE id = ? ";          PreparedStatementCreator psCreator = con -> {              PreparedStatement ps = con.prepareStatement(sql);              ps.setString(1, t.getUsername());              ps.setString(2, t.getPassword());              ps.setString(3, t.getEmail());              ps.setString(4, t.getSodienthoai());              ps.setBoolean(5, t.getEnabled());              ps.setLong(6, t.getId());              return ps;          };          jdbcTemplate.update(psCreator);          return this.findById(t.getId());      }      @Override      public Boolean delete(Long tKey) {          String sql = "DELETE FROM [User] WHERE id = ? ";          PreparedStatementCreator psCreator = con->{              PreparedStatement ps = con.prepareStatement(sql);              ps.setLong(1, tKey);              return ps;          };          jdbcTemplate.update(psCreator);            return true;      }      // 🟩🟩🟩🟩🟩🟩hết phần implements🟩🟩🟩🟩🟩🟩🟩 //      public User findByUsername(String username){          String sql= "SELECT \* FROM [User] WHERE username= ? ";          PreparedStatementCreator psCreator = con -> {              PreparedStatement ps = con.prepareStatement(sql);                ps.setString(1, username);              return ps;          };          return this.jdbcTemplate.query(psCreator, this.userRowMapper).stream().findFirst().orElse(null);      } |

|  |
| --- |
| UserService |
| @Service  public class UserService      implements ServiceInterface<User, Long>  {        @Autowired      private UserRepo userRepo;      @Override      public User findById(Long tKey) {          return this.userRepo.findById(tKey);      }      @Override      public List<User> getList(Integer sttPage, Integer sizePage) {          return this.userRepo.getList(sttPage, sizePage);      }      @Override      public User create(User t) {          t.setId(null);            User user1 = this.findByUsername(t.getUsername());          if(user1 != null){              return null;          }          return this.userRepo.create(t);      }      @Override      public User update(Long tKey, User t) {          User user1 = this.findById(tKey);          if(user1==null){              return null;          }          return this.userRepo.update(tKey, t);      }      @Override      public Boolean delete(Long tKey) {          return this.userRepo.delete(tKey);      }      // 🟩🟩🟩🟩🟩🟩hết phần implements🟩🟩🟩🟩🟩🟩🟩 //      public User findByUsername(String username){          return this.userRepo.findByUsername(username);      }  } |