

EXAMPLE

<<Pacote>> NomeDaClasse

+ atributo: TipoAtributo

+ metodo(parametro: TipoParametro) : TipoMétodo;

+ metodo1(parametro: TipoParametro);

TAPE

COMMENT

<<*Entity*>> Tape <<Entity>>
Comment

.

ROLE

USER

<<Entity>> Role <<*Entity*>> Usuario

	< <enum>> TapeGender</enum>	
< <enum>> TapeStatus AILABLE: Enum; ENTED: Enum;</enum>	+ ACTION: Enum; + ADVENTURE: Enum; + ANIMATED: Enum; + COMEDY: Enum; + HORROR: Enum; + DRAMA: Enum; + ROMANTIC: Enum; + FANTASY: Enum; + MUSICAL: Enum;	

+ id: UUID;
+ name: String;
+ code: String;
+ status: TapeSt
+ gender: TapeG
+ users: List<Us

+ Tape();
+ Tape(id: UUID
+ getID(): UUID;
+ setID(id: UUID
+ getName(): Str
+ setName(nam
+ getCode(): Str
+ setCode(code
+ getStatus(): Ta
+ setStatus(tape
+ getTapeGende
+ setTapeGende
+ getUsers(): Lis
+ setUsers(user:
+ isRented(): bo

+ isRented(): bo + isAvailable(): b + hashCode(): ir

+ equals(obj: Ob

```
+ descricao: String
                                                                                                                             + tape: Tape;
                                                                                                                             + user: User:
atus;
Sender;
                                                                                                                             + Comment(id: UUID, descricao: String, tape: Tape, user: User);
                                                                                                                             + Comment();
                                                                                                                             + Comment(id: UUID);
name: String, code: String, status: TapeStatus, gender: TapeGender, user: List<User>);
                                                                                                                             + getId(): UUID;
                                                                                                                             + setId(id: UUID): Void;
): Void;
                                                                                                                             + getDescricao(): String;
                                                                                                                             + setDescricao(descricao: String): Void;
e: String): Void;
                                                                                                                             + getTape(): Tape;
                                                                                                                             + setTape(tape: Tape): void;
ing;
                                                                                                                             + getUser(): User;
: String): Void;
peStatus;
                                                                                                                             + setUser(user: User): Void
Status: TapeStatus): Void;
er(): TapeGender;
er(tapeGende: TapeGender): Void;
st<User>:
s: List<User>);
olean;
                                                                                                                                                    <<Controller>>
oolean;
                                                                                                                                                  CommentController
jeto): boolean;
                                                                                                                       + createTapeCommentService: CreateTapeCommentService
                                                                                                                       + create(createTapeCommentDTO: CreateTapeCommentDTO): Comment;
                                                                                                                       + list(): List<Comment>;
                               <<Controller>>
                               TapeController
                                                                                                                                                      <<Service>>
                                                                                                                                              CreateTapeCommentService
        + tapeService: TapeService;
        + createRentTapeService: CreateRentTapeService;
                                                                                                                        + tapeRepository : TapeRepository
        + deliverRentedTapeService: DelieverRentedTapeService;
                                                                                                                        + commentRepository : CommentRepository
                                                                                                                        + userRepository: UserRepository
        + create(tape: Tape): Tape;
        + rent(createRentTapeService: CreateRentTapeService): Tape;
        + deliver(tape: Tape): Tape;
                                                                                                                        + execute(createTapeCommentDTO : CreateTapeCommentDTO);
        + list(): List<Tape>;
                                                                                                                        + listAll(): List<Comment>;
                                                                                                                                                       <<DTO>>
                                 <<DTO>>
                                                                                                                                               CreateTapeCommentDTO
                            CreateTapeRentDTO
                                                                                                                                    + idUser: UUID
                                                                                                                                    + idTape: UUID
                  + idTape : UUID
                                                                                                                                    + idComment: UUID
                  +idsUser : List<UUID>
                                                                                                                                    + getIdUser(): UUID;
                                                                                                                                    + setIdUser(idUser: UUID): Void;
                  +getIdTape(): UUID;
                                                                                                                                    + getIdTape(): UUID;
                  +setIdTape(idTape: UUID): Void;
                                                                                                                                    + setIdTape(idTape: UUID): Void;
                  +getidsUser(): UUID;
                                                                                                                                    + getIdComment(): UUID;
                  +setIdUser(idsUser: List<UUID>): Void;
                                                                                                                                    + setIdComment(idComment: UUID): Void;
                  +hascode(): int;
                                                                                                                                    + hashCode(): int;
                  +equals(obj. Object): hoolean:
```

+ id: UUID;

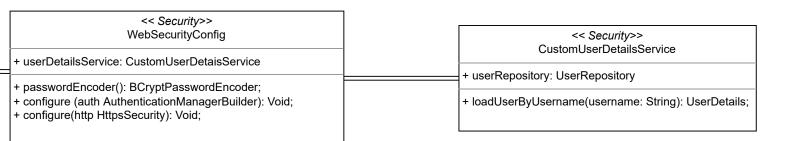
```
+ id: UUID
       + name: String
       + Role(id: UUID);
       + Role();
       + Role(id: UUID, name: String);
       + getId(): UUID;
       + setId(id: UUID): Void;
       + getName(): String;
       + setName(name: String): Void;
       + hashCode(): int;
       + equals(obj: Object): boolean;
                         <<DTO>>
                   CreateUserRoleDTO
        + idUser : UUID
        + idsRoles : List<UUID>
        +getidUser(): UUID;
        +setIdUser(idUser : UUID): Void;
        +getIdsRoles(): List<UUID>;
        +setIdsRoles(idsRoles : List<UUID>): Void;
        +hascode(): int;
        +equals(obj: Object): boolean;
                       <<Service>>
                  CreateRoleUserService
+ userRepository : UserRepository
+execute(createUserRoleDTO : CreateUserRoleDTO): User;
```

+ id: UUID + name: String

+ User();

+ email: String

```
+ username: String
+ password: String
+ roles : List<Role>
+ User(id: UUID);
+ User(id: UUID, name:String, email: String, username: String, password: String, roles: List<Role>);
+ getId(): UUID;
+ setId(id: UUID): Void;
+ getName(): String;
+ setName(name: String): Void;
+ getEmail(): String;
+ setEmail(email: String): Void;
+ getUsername(): String;
+ setUsername(username: String): Void;
+ getPassword(password: String): String;
+ getRoles(): List<Role>;
+ setRoles(roles: List<Role>): Void;
+ hashCode(): int;
+ equals(obj: Object): boolean;
                                          << Security>>
                                          UserPrincipal
                  + serialVersionUID: final long
                  + username : String
                  + password : String
                  + authorities : Collection
                  + UserPrincipal(username: User);
                  + create (user User): Static;
                  + getAutorities(): Collection<? extends GrantedAuthority>;
                  + getPassword(): String;
                  + getUsername(): String;
                 + isAccountNonExpired(): boolean;
                  + isAccountNonLocked(): boolean:
                  + isCredentialsNonExpired(): boolean;
                  + isEnabled(): boolean;
                                          <<Controller>>
                                          UserController
                   + userService: UserService;
                   + createRoleUserService : CreateRoleUserService
                   + create(user : User): User;
                   + role(createUserRoleDTO : CreateUserRoleDTO): User;
                   + list(): List<User>;
```



			Deli	< <service>> iverRentedTape</service>
				epository : Tape
RentDTO): Tape;		+ execut	te(tape : Tape):
₽R	entDTO	entDTO): Tape;	entDTO): Tape;	+ tapeR

