|  |  |
| --- | --- |
| **Nome Classe** | **AutenticationDAO** |
| **Descrizione** | Questa classe gestisce le funzionalità relative all’autenticazione e la registrazione nel sistema. |
| **Pre-condizione** | **context** AutenticationDAO**:** login (String email, String password);  **pre:** String email != null && String password != null  **context** AutenticationDAO: logout ():  **pre:** login=True  **context** AutenticationDAO**:** register (String email, String password, String nome, String cognome, String nickname)  **pre:** request.getParameter(“email”) != null &&  request.getParameter(“password”) != null &&  request.getParameter(“nome”) != null &&  request.getParameter(“cognome”) != null &&  request.getParameter(“nickname”) != null; |
| **Post-condizione** | **context** AutenticationDAO: login (String email, String password);  **post:** login=True  **context** AutenticationDAO: logout ():  **post:** login=False  **context** AutenticationDAO**:** register (String email, String password, String nome, String cognome, String nickname)  **post:** register=True |
| **Invarianti** |  |

|  |  |
| --- | --- |
| Nome Classe | **UserDAO** |
| Descrizione | Questa classe rappresenta l’entità tabellare User contenuta all’interno del nostro DBMS relazionale. Ci permette di interfacciarci con il DBMS relazionale e di interrogarlo tramite dei metodi specifici. |
| Pre-condizione | **context** UserDAO**:** removeUser(String nickname);  **pre:** register=True  **context** UserDAO**:** editProfile(User user);  **pre:** login=True |
| Post-condizione | **context** UserDAO**:** removeUser(String nickname);  **post:** removeUser=True  **context** UserDAO**:** editProfile(User user);  **pre:** editProfile=True |
| Invarianti |  |

|  |  |
| --- | --- |
| Nome Classe | **PublicationDAO** |
| Descrizione | Questa classe rappresenta l’entità tabellare Publication contenuta all’interno del nostro DBMS relazionale. Ci permette di interfacciarci con il DBMS relazionale e di interrogarlo tramite dei metodi specifici. |
| Pre-condizione | **context** PublicationDAO**:** insertPublication(Publication p);  **pre: if p.getTipo != null && p.getTipo == Topic:**  p.getCodicepubblicazione != null &&  p.getAutore != null &&  p.getTitolo != null &&  p.getDescrizione != null &&  p.getCategoria != null &&  p.getData != null  **if p.getTipo != null && p.getTipo == Recensione:**  p.getCodicepubblicazione != null &&  p.getAutore != null &&  p.getTitolo != null &&  p.getDescrizione != null &&  p.getCategoria != null &&  p.getData != null &&  p.getGameplay != null &&  p.getTrama != null &&  p.getGrafica != null &&  p.getVotocomplessivo != null  **context** PublicationDAO**:** getPublication(String codicePubblicazione);  **pre:** codicePubblicazione != null  **context** PublicationDAO**:** removePublication(String codicePubblicazione);  **pre:** codicePubblicazione != null |
| Post-condizione | **context** PublicationDAO**:** insertPublication(Publication p);  **post:** insertPublication=True  **context** PublicationDAO**:** getPublication(String codicePubblicazione);  **post:** return Publication p  **context** PublicationDAO**:** removePublication(String codicePubblicazione);  **post:** removePublication=True |
| Invarianti |  |

|  |  |
| --- | --- |
| Nome Classe | **NewsDAO** |
| Descrizione | Questa classe rappresenta l’entità tabellare News contenuta all’interno del nostro DBMS relazionale. Ci permette di interfacciarci con il DBMS relazionale e di interrogarlo tramite dei metodi specifici. |
| Pre-condizione | **context** NewsDAO**:** insertNews(News n);  **pre:** n.getAutore != null &&  n.getTitolo != null &&  n.getDescrizione != null &&  n.getData != null  **context** NewsDAO**:** getNews(String autore, String titolo, Date data);  **pre:** autore != null && titolo != null && data != null  **context** NewsDAO**:** removeNews(String autore, String titolo, Date data);  **pre:** autore != null && titolo != null && data != null |
| Post-condizione | **context** NewsDAO**:** insertNews(News n);  **post:** insertNews=True  **context** NewsDAO**:** getNews(String autore, String titolo, Date data);  **post:** return News n  **context** NewsDAO**:** removeNews(String autore, String titolo, Date data);  **post:** return True |
| Invarianti |  |

|  |  |
| --- | --- |
| Nome Classe | **VideoGameDAO** |
| Descrizione | Questa classe rappresenta l’entità tabellare VideoGame contenuta all’interno del nostro DBMS relazionale. Ci permette di interfacciarci con il DBMS relazionale e di interrogarlo tramite dei metodi specifici. |
| Pre-condizione | **context** VideoGameDAO**:** insertVideoGame(VideoGame v);  **pre:** v.getGenere != null &&  v.getTitolo != null &&  v.getDescrizione != null  **context** VideoGameDAO**:** getVideoGame(String titolo);  **pre:** titolo != null  **context** VideoGameDAO**:** removeVideoGame(String titolo);  **pre:** titolo != null |
| Post-condizione | **context** VideoGameDAO**:** insertVideoGame(VideoGame v);  **post:** insertVideoGame=True  **context** VideoGameDAO**:** getVideoGame(String titolo);  **post:** return Videogame v  **context** VideoGameDAO**:** removeVideoGame(String titolo);  **post:** removeVideoGame=True |
| Invarianti |  |

|  |  |
| --- | --- |
| Nome Classe | **BugDAO** |
| Descrizione | Questa classe rappresenta l’entità tabellare Bug contenuta all’interno del nostro DBMS relazionale. Ci permette di interfacciarci con il DBMS relazionale e di interrogarlo tramite dei metodi specifici. |
| Pre-condizione | **context** BugDAO**:** insertBug(Bug b);  **pre:** b.getTitolo != null &&  b.getCategoria != null &&  b.getData != null &&  b.getTesto != null &&  b.getVideogioco != null &&  b.getAutore != null  **context** BugDAO**:** removeBug(String autore, String videogioco, Date data);  **pre:** autore != null && videogioco != null && data != null  **context** BugDAO**:** getBug(String autore, String videogioco, Date data);  **pre:** autore != null && videogioco != null && data != null  **context** BugDAO**:** approvedBug(String autore, String videogioco, Date data);  **pre:** autore != null && videogioco != null && data != null  **context** BugDAO**:** disapprovedBug(String autore, String videogioco, Date data);  **pre:** autore != null && videogioco != null && data != null |
| Post-condizione | **context** BugDAO**:** insertBug(Bug b);  **post:** insertBug=True  **context** BugDAO**:** removeBug(String autore, String videogioco, Date data);  **post:** removeBug=True  **context** BugDAO**:** getBug(String autore, String videogioco, Date data);  **post:** return Bug b  **context** BugDAO**:** approvedBug(String autore, String videogioco, Date data);  **post:** approvedBug=True  **context** BugDAO**:** disapprovedBug(String autore, String videogioco, Date data);  **post:** disapprovedBug=True |
| Invarianti |  |

|  |  |
| --- | --- |
| Nome Classe | **CommentDAO** |
| Descrizione | Questa classe rappresenta l’entità tabellare Comment contenuta all’interno del nostro DBMS relazionale. Ci permette di interfacciarci con il DBMS relazionale e di interrogarlo tramite dei metodi specifici. |
| Pre-condizione | **context** commentDAO**:** insertComment(Comment c);  **pre:** c.getCodicepubblicazione != null &&  c.getAutore != null &&  c.getData != null &&  c.getTesto != null  **context** commentDAO**:** removeComment(String codicePubblicazione, String autore, Date data);  **pre:** codicePubblicazione != null && autore != null && data != null  **context** commentDAO**:** getListComment(String codicePubblicazione);  **pre:** codicePubblicazione != null |
| Post-condizione | **context** commentDAO**:** insertComment(Comment c);  **post:** insertComment=True  **context** commentDAO**:** removeComment(String codicePubblicazione, String autore, Date data);  **post:** removeComment=True  **context** commentDAO**:** getListComment(String codicePubblicazione);  **post:** return ArrayList<Comment> |
| Invarianti |  |

|  |  |
| --- | --- |
| Nome Classe | **PublicationReportDAO** |
| Descrizione | Questa classe rappresenta l’entità tabellare PublicationReport contenuta all’interno del nostro DBMS relazionale. Ci permette di interfacciarci con il DBMS relazionale e di interrogarlo tramite dei metodi specifici. |
| Pre-condizione | **context** PublicationReportDAO**:** insertPublicationReport(PubblicationReport pr);  **pre:** pr.getCodicePubblicazione != null &&  pr.getAutoreSegnalazione != null &&  pr.getData != null &&  pr.getCategoria != null &&  pr.getDescrizione != null  **context** PublicationReportDAO**:** removePublicationReport(String autoreSegnalazione, String codicePubblicazione);  **pre:** autoreSegnalazione != null && codicePubblicazione != null  **context** PublicationReportDAO**:** approvePublicationReport(String autoreSegnalazione, String codicePubblicazione);  **pre:** autoreSegnalazione != null && codicePubblicazione != null  **context** PublicationReportDAO**:** disapprovePublicationReport(String autoreSegnalazione,  String codicePubblicazione);  **pre:** autoreSegnalazione != null && codicePubblicazione != null |
| Post-condizione | **context** PublicationReportDAO**:** insertPublicationReport(PubblicationReport pr);  **post:** insertPublicationReport=True  **context** PublicationReportDAO**:** removePublicationReport(String autoreSegnalazione, String codicePubblicazione);  **post:** removePublicationReport=True  **context** PublicationReportDAO**:** approvePublicationReport(String autoreSegnalazione, String codicePubblicazione);  **post:** approvePublicationReport=True  **context** PublicationReportDAO**:** disapprovePublicationReport(String autoreSegnalazione,  String codicePubblicazione);  **post:** disapprovePublicationReport=True |
| Invarianti |  |