Progetto Object Orientation Wiki

Tammaro Iavarazzo

Angelo Marcone

N86004573

N86004638

Anno Accademico
23/24

Indice

1 Progettazione Concettuale

1.1 Traccia del progetto

Si sviluppi un sistema informativo, composto da una base di dati relazionale e da un applicativo Java dotato di GUI (Swing o JavaFX), per la gestione del ciclo di vita di una pagina di una wiki.

Una pagina di una wiki ha un titolo e un testo. Ogni pagina è creata da un determinato autore. Il testo è composto di una sequenza di frasi. Il sistema mantiene traccia anche del giorno e ora nel quale la pagina è stata creata. Una frase può contenere anche un collegamento. Ogni collegamento è caratterizzato da una frase da cui scaturisce il collegamento e da un'altra pagina destinazione del collegamento.

Il testo può essere modificato da un altro utente del sistema, che seleziona una frase, scrive la sua versione alternativa (modifica) e prova a proporla.

La modifica proposta verrà notificata all'autore del testo originale la prossima volta che utilizzerà il sistema. L'autore potrà vedere la sua versione originale e la modifica proposta. Egli potrà accettare la modifica (in quel caso la pagina originale diventerà ora quella con la modifica apportata), rifiutare la modifica (la pagina originale rimarrà invariata). La modifica proposta dall'autore verrà memorizzata nel sistema e diventerà subito parte della versione corrente del testo. Il sistema mantiene memoria delle modifiche proposte e anche delle decisioni dell'autore (accettazione o rifiuto).

Nel caso in cui si fossero accumulate più modifiche da rivedere, l'autore dovrà accettarle o rifiutarle tutte nell'ordine in ordine dalla più antica alla più recente.

Ad esempio, il testo originale del 3 novembre ore 10 del Prof. Tramontana potrebbe essere "traccia del progetto di OO", mentre il 3 novembre alle ore 11 il Prof. Barra potrà modificarlo in "traccia del progetto di OO e BD" e il Prof. Tramontana potrà accettare la modifica alle ore 12. A quel punto la versione corrente sarà quella proposta dal Prof. Barra. In alternativa il Prof. Tramontana potrebbe, alle ore 12 rifiutare la modifica del Prof. Barra e attuare invece la modifica "traccia del progetto di OO e BD gruppo 2" che diventerà subito parte della versione corrente (essendo la modifica stata disposta dall'autore della pagina).

Gli utenti generici del sistema potranno cercare una pagina e il sistema mostrerà la

versione corrente del testo e i collegamenti.

Gli autori dovranno prima autenticarsi fornendo la propria login e password. Tutti gli autori potranno vedere tutta la storia di tutti i testi dei quali sono autori e di tutti quelli nei quali hanno proposto una modifica.

1.2 Analisi dei requisiti

La nostra traccia richiede di creare un sistema informativo per la gestione del ciclo di vita di una pagina di una wiki. Dalla traccia andiamo a trovare i seguenti requisiti:

1. **Gestione degli utenti:**
   (a) Gli utenti si registrano al sistema fornendo le proprie informazioni come email, nome, cognome e password.
   (b) Gli autori sono utenti speciali che hanno un nome d'arte e un anno di inizio della carriera. Ogni autore è associato a un utente.
2. **Gestione delle pagine wiki:**
   (a) Ogni pagina ha un titolo univoco e un testo associato.
   (b) Ogni pagina è creata da un autore e ha una data e un'ora di creazione.
3. **Gestione delle modifiche:**
   (a) Gli utenti possono proporre modifiche al testo di una pagina.
   (b) Ogni modifica è caratterizzata da un codice univoco, la frase originale, la frase modificata, lo stato della modifica (accettata, rifiutata o in attesa), data e ora della modifica, l'utente che ha proposto la modifica, l'autore della pagina e il codice della frase associata alla modifica.
   (c) Le modifiche proposte vengono notificate all'autore originale della pagina per la revisione.
   (d) L'autore può accettare o rifiutare una modifica. Se accettata, diventa parte della versione corrente del testo della pagina.
   (e) Se la modifica è proposta dall'autore stesso della pagina, allora la modifica verrà effettuata immediatamente

4. **Gestione delle frasi:**

   (a) Il testo di una pagina è composto da una sequenza di frasi.

   (b) Ogni frase può contenere collegamenti ad altre pagine wiki.

5. **Gestione dei collegamenti:**

   (a) I collegamenti collegano una frase della pagina a un'altra pagina wiki.

   (b) Ogni collegamento è caratterizzato da una frase di origine e dalla pagina di destinazione del collegamento.

6. **Autenticazione degli autori:**

   (a) Gli autori devono autenticarsi fornendo la loro email e password per accedere al sistema.

   (b) Gli autori possono accedere alla storia di tutti i testi di cui sono autori e di quelli in cui hanno proposto modifiche.

7. **Ricerca delle pagine:**

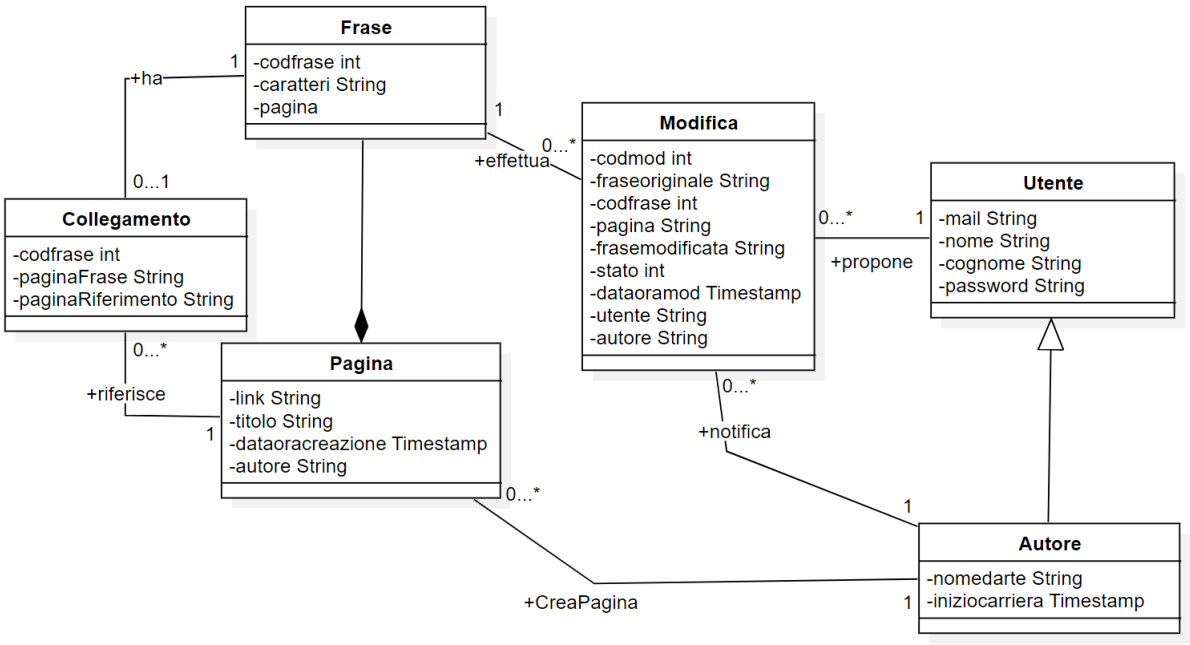
   (a) Gli utenti possono cercare una pagina e visualizzare la versione corrente del testo e i collegamenti associati.

8. **Revisione delle modifiche multiple:**

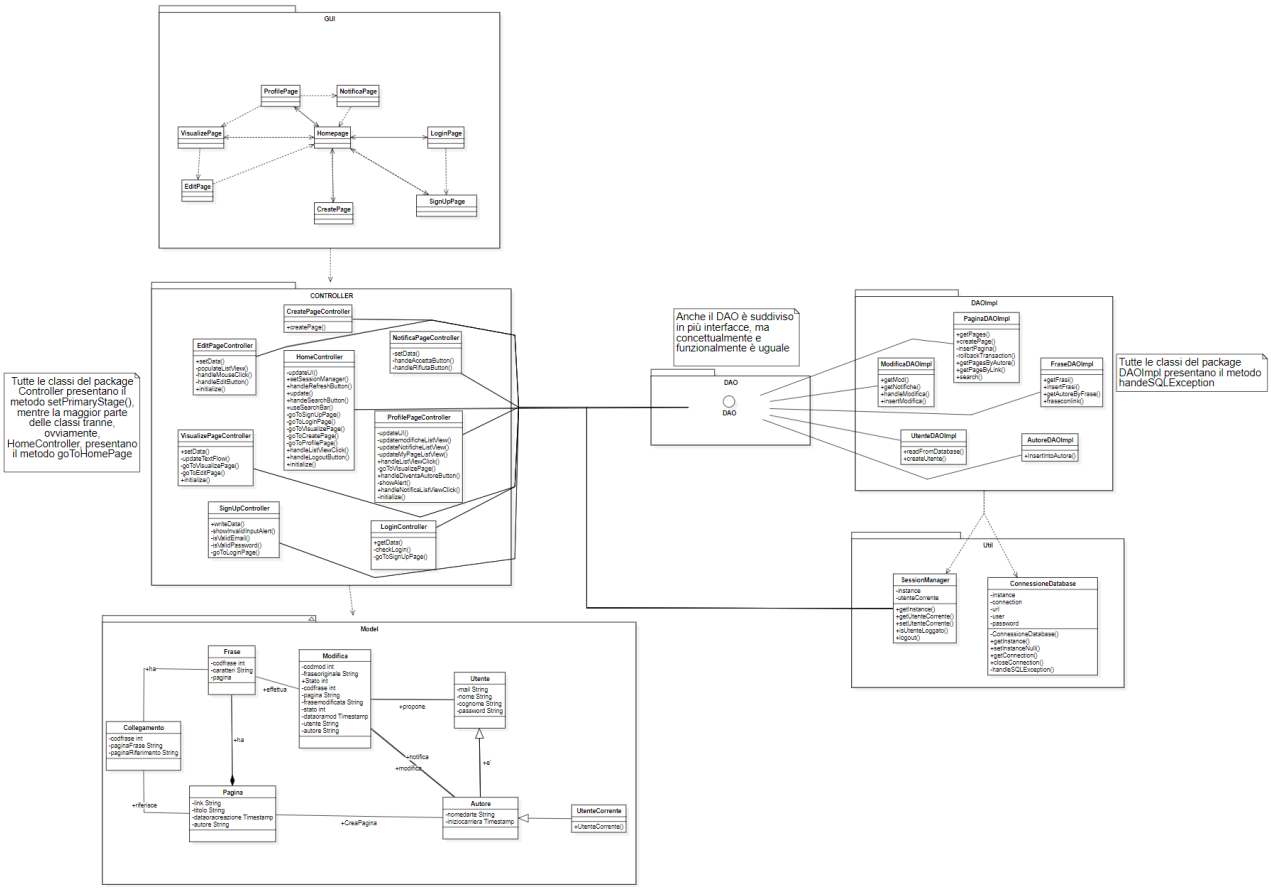
   (a) Se ci sono più modifiche pendenti su una pagina, l'autore deve decidere se accettarle o rifiutarle nell'ordine cronologico delle proposte.

2 Modello Concettuale

2.1 Modello UML del dominio del problema



2.2 Modello UML del dominio della soluzione



3 Analisi del Java Application

Ecco un'analisi dettagliata dell'intera applicazione java fatta con javafx:

1. **Main:**
   (a) **Funzionalità:** Gestisce l'avvio dell'applicazione.
   (b) **Responsabilità:**
      i. Carica la schermata principale dell'applicazione.
      ii. Inizializza il primaryStage.
   (c) **Metodi Principali:**
      i. **start(Stage primaryStage):** Metodo principale per avviare l'applicazione.
2. **ConnessioneDatabase:**
   (a) **Funzionalità:** Gestisce la connessione al database.
   (b) **Responsabilità:**
      i. Fornisce metodi per stabilire una connessione al database.
      ii. Fornisce metodi per eseguire query sul database.
   (c) **Metodi Principali:**
      i. **getConnection():** Restituisce un'istanza di connessione al database.
      ii. Altri metodi per eseguire query, inserimenti e aggiornamenti nel database.
3. **SessionManager:**
   (a) **Funzionalità:** Gestisce lo stato della sessione utente.
   (b) **Responsabilità:**
      i. Mantiene le informazioni di sessione dell'utente, come l'utente corrente.
      ii. Fornisce metodi per il login, il logout e il controllo dello stato della sessione.
   (c) **Metodi Principali:**
      i. **getInstance():** Restituisce l'istanza Singleton del SessionManager.
      ii. **login(UtenteCorrente utenteCorrente):** Esegue il login dell'utente specificato.

iii. **logout():** Esegue il logout dell'utente corrente.

iv. Altri metodi per ottenere e impostare l'utente corrente, verificare lo stato del login, ecc.

4. **HomeController:**

   (a) **Funzionalità:** Gestisce la schermata principale dell'applicazione (HomePage).

   (b) **Responsabilità:**

      i. Aggiorna l'interfaccia utente con le pagine esistenti.

      ii. Gestisce il login e il logout dell'utente.

      iii. Gestisce la ricerca di pagine.

      iv. Naviga verso altre pagine dell'applicazione.

   (c) **Metodi Principali:**

      i. **initialize(URL url, ResourceBundle resourceBundle):** Metodo chiamato durante l'inizializzazione del controller.

      ii. **handleRefreshButton(ActionEvent event):** Gestisce l'azione del pulsante di aggiornamento.

      iii. **handleSearchButton():** Gestisce l'azione del pulsante di ricerca.

      iv. Altri metodi per gestire il login, il logout, la navigazione e la gestione della ListView.

5. **VisualizePageController:**

   (a) **Funzionalità:** Visualizza i dettagli di una singola pagina.

   (b) **Responsabilità:**

      i. Ottiene e visualizza i dettagli della pagina selezionata.

      ii. Gestisce il ritorno alla HomePage.

   (c) **Metodi Principali:**

      i. **initialize(URL url, ResourceBundle resourceBundle):** Metodo chiamato durante l'inizializzazione del controller.

      ii. **setData(Pagina currentPage):** Imposta i dati della pagina da visualizzare.

      iii. **goToHomePage():** Naviga alla HomePage.

6. **SignUpController:**
   (a) **Funzionalità:** Gestisce la registrazione di un nuovo utente.
   (b) **Responsabilità:**
      i. Raccoglie i dati inseriti dall'utente durante la registrazione.
      ii. Utilizza UtenteDAO per registrare il nuovo utente nel database.
   (c) **Metodi Principali:**
      i. **setPrimaryStage(Stage primaryStage):** Imposta il primaryStage per il controller.
      ii. **handleSignUpButton():** Gestisce l'azione del pulsante di registrazione.

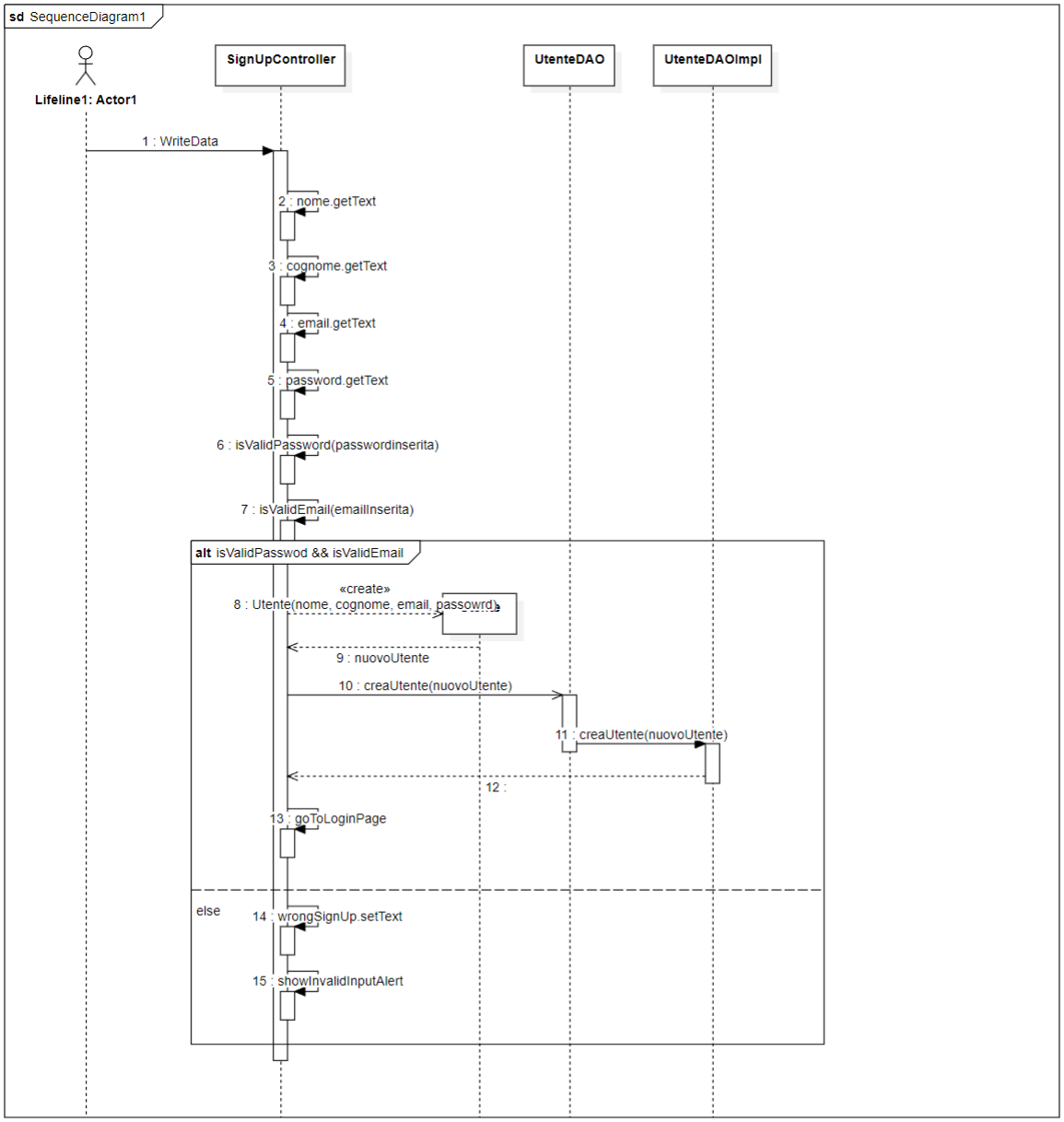
7. **ProfilePageController:**
   (a) **Funzionalità:** Visualizza il profilo dell'utente.
   (b) **Responsabilità:**
      i. Ottiene e visualizza i dettagli del profilo dell'utente corrente.
   (c) **Metodi Principali:**
      i. **setPrimaryStage(Stage primaryStage):** mposta il primaryStage per il controller.

8. **LoginController:**
   (a) **Funzionalità:** Gestisce la schermata di login.
   (b) **Responsabilità:**
      i. Gestisce il login dell'utente.
      ii. Naviga alla HomePage dopo il login.
   (c) **Metodi Principali:**
      i. **setPrimaryStage(Stage primaryStage):** Imposta il primaryStage per il controller.
      ii. **getData(ActionEvent event):** Gestisce l'azione del pulsante di login, controllando se le credenziali siano giuste.

9. **EditPageController:**
    (a) **Funzionalità:** Gestisce la pagina di modifica di una frase.
    (b) **Responsabilità:**
        i. Consente all'utente di selezionare e modificare una frase esistente.
        ii. Registra la modifica nel database e naviga alla HomePage.
    (c) **Metodi Principali:**
        i. **setPrimaryStage(Stage primaryStage):** Imposta il primaryStage per il controller.
        ii. **handleMouseClick():** Gestisce il click del mouse sulla ListView.
        iii. **handleEditButton():** Gestisce l'azione del pulsante di modifica.
10. **CreatePageController:**
    (a) **Funzionalità:** Gestisce la creazione di una nuova pagina.
    (b) **Responsabilità:**
        i. Raccoglie i dati inseriti dall'utente per la nuova pagina.
        ii. Utilizza PaginaDAO per creare la nuova pagina nel database.
        iii. Naviga alla HomePage dopo la creazione della pagina.
    (c) **Metodi Principali:**
        i. **setPrimaryStage(Stage primaryStage):** Imposta il primaryStage per il controller.
        ii. **createPage():** Gestisce l'azione del pulsante per la creazione di una nuova pagina.

4 Sequence Diagram

4.1 Funzione 1



4.2 Funzione 2