

REALIZZAZIONE DI UN SOFTWARE DI PLANNING AZIENDALE IN JAVA

Autori: Andrea Lemmo N86003262, Davide Soldatini N86002862, Michele Zurlo N86003219

Indice

| | |
|--------------------------------------|----|
| Descrizione del problema | 5 |
| 1. Class Diagram | 6 |
| 2. CRC Cards..... | 7 |
| 2.1 Entità..... | 7 |
| Dipendente | 7 |
| Meeting | 7 |
| Progetto | 8 |
| LuogoNascita | 8 |
| Skill..... | 8 |
| SalaRunione | 9 |
| PartecipazioneMeeting..... | 9 |
| CollaborazioneProgetto..... | 10 |
| AmbitoProgetto | 10 |
| 2.2 Controller..... | 11 |
| ControllerStart..... | 11 |
| ControllerAccesso | 11 |
| ControllerGestioneProfilo..... | 12 |
| ControllerProgetto..... | 12 |
| ControllerMeeting | 13 |
| ControllerDipendentiSegreteria | 13 |
| ControllerPartecipantiProgetto | 14 |
| ControllerPartecipantiMeeting | 14 |
| ControllerProgettiSegreteria | 15 |
| ControllerMeetingSegreteria..... | 15 |
| ControllerAreaSegreteria..... | 16 |
| 2.3 Elementi GUI..... | 17 |
| iPlanner..... | 17 |
| Login | 17 |
| Home | 18 |
| MioAccount | 18 |
| MieiProgetti..... | 19 |

| | |
|-------------------------------------|-----------|
| MieiMeeting | 19 |
| GestioneProgettiDipendente..... | 20 |
| GestioneProgettiSegreteria | 21 |
| GestioneMeetingDipendente | 21 |
| GestioneSaleSegreteria | 22 |
| GestioneMeetingSegreteria | 22 |
| AreaSegreteria | 23 |
| AutenticazioneSegreteria | 23 |
| InserisciPartecipantiProgetto | 24 |
| InserisciPartecipantiMeeting..... | 24 |
| GestioneDipendentiSegreteria | 25 |
| MeetingListRenderer | 25 |
| DipendentiTableModel..... | 26 |
| MeetingTableModel | 26 |
| PartecipantiTableModel | 27 |
| ProgettoTableModel..... | 27 |
| InvitatiListRenderer | 28 |
| PartecipantiListRenderer..... | 28 |
| ProgettoListRenderer | 28 |
| ProgettoDiscussoListRenderer | 29 |
| CustomScrollBarUI..... | 29 |
| DataComparator | 30 |
| OrarioComparator | 30 |
| DefaultLookManager..... | 31 |
| ErroreDialog..... | 31 |
| 2.4 InterfacceDAO | 32 |
| DipendenteDAO..... | 32 |
| ProgettoDAO | 32 |
| MeetingDAO | 33 |
| AmbitoProgettoDAO..... | 33 |
| LuogoNascitaDAO | 34 |
| SalaRiunioneDAO..... | 34 |
| SkillDAO | 35 |
| 2.5 ImplementazioniDAO | 36 |
| AmbitoProgettoDAOPSQL | 36 |
| DipendenteDAOPSQL | 36 |

| | |
|----------------------------|----|
| LuogoNascitaDAOPSQL..... | 37 |
| MeetingDAOPSQL..... | 37 |
| ProgettoDAOPSQL | 38 |
| SalaRiunioneDAOPSQL | 38 |
| SkillDAOPSQL..... | 39 |
| 2.6 dbManager | 40 |
| ManagerConnessioneDB | 40 |
| CostruttoreDB..... | 41 |
| 2.7 Starter..... | 42 |
| 3. Sequence Diagrams | 43 |

Descrizione del problema

La traccia ricevuta (n.2) era testualmente la seguente:

"Si sviluppi un sistema informativo, composto da una base di dati relazionale e un applicativo Java dotato di GUI (Swing o JavaFX), per la gestione di progetti in un'azienda. Si tenga traccia dei partecipanti al progetto, identificando i ruoli per ognuno di essi (per ogni progetto ci sarà solo un project manager). Ad ogni progetto è associato una tipologia ("Ricerca di base", "Ricerca Industriale", "Ricerca sperimentale", "Sviluppo Sperimentale", ...) ed uno o più ambiti (Economia, Medicina, ...). Il sistema dovrà permettere anche l'organizzazione di meeting fisicamente, in sale riunioni, o telematicamente su una piattaforma di videoconferenza. Si dovrà tenere traccia delle partecipazioni ai progetti ed ai meeting, ai fini della valutazione del singolo partecipante. In fase di creazione di un nuovo progetto, i partecipanti dovranno essere selezionati in base a criteri di ricerca che includono anche il salario medio e la valutazione aziendale del partecipante, oltre alla tipologia di progetti cui ha preso parte. Ad ogni partecipante sarà associata una lista di skill. Inoltre, in fase di creazione di un nuovo progetto, i partecipanti potranno essere scelti in funzione anche delle loro skill. In fase di registrazione di un partecipante, inserire le skill e se non presente nel DB, crearne una nuova."

Nelle prossime pagine di questo documento troverete:

- Un Class diagram di progettazione dell'intero software.
- CRC cards di tutte le classi presenti, per comprendere al meglio il compito di ognuna di queste nell'intero progetto.
- Sequence diagrams di due tra le funzionalità cardine dell'intero software: la creazione di un nuovo progetto e l'eliminazione di un meeting da parte del dipendente.

1. Class Diagram

Nella pagina successiva troverete una rappresentazione del class diagram, completamente ridimensionabile e senza perdita di qualità

- Nella parte inferiore potrete osservare tutte le entità di base del problema, con le relative associazioni che verranno poi tradotte in maniera corretta all'interno del codice Java.
- Al di sopra delle entità troviamo le implementazioni DAO per PostgreSQL , il DBMS utilizzato per gestire la persistenza dei dati.

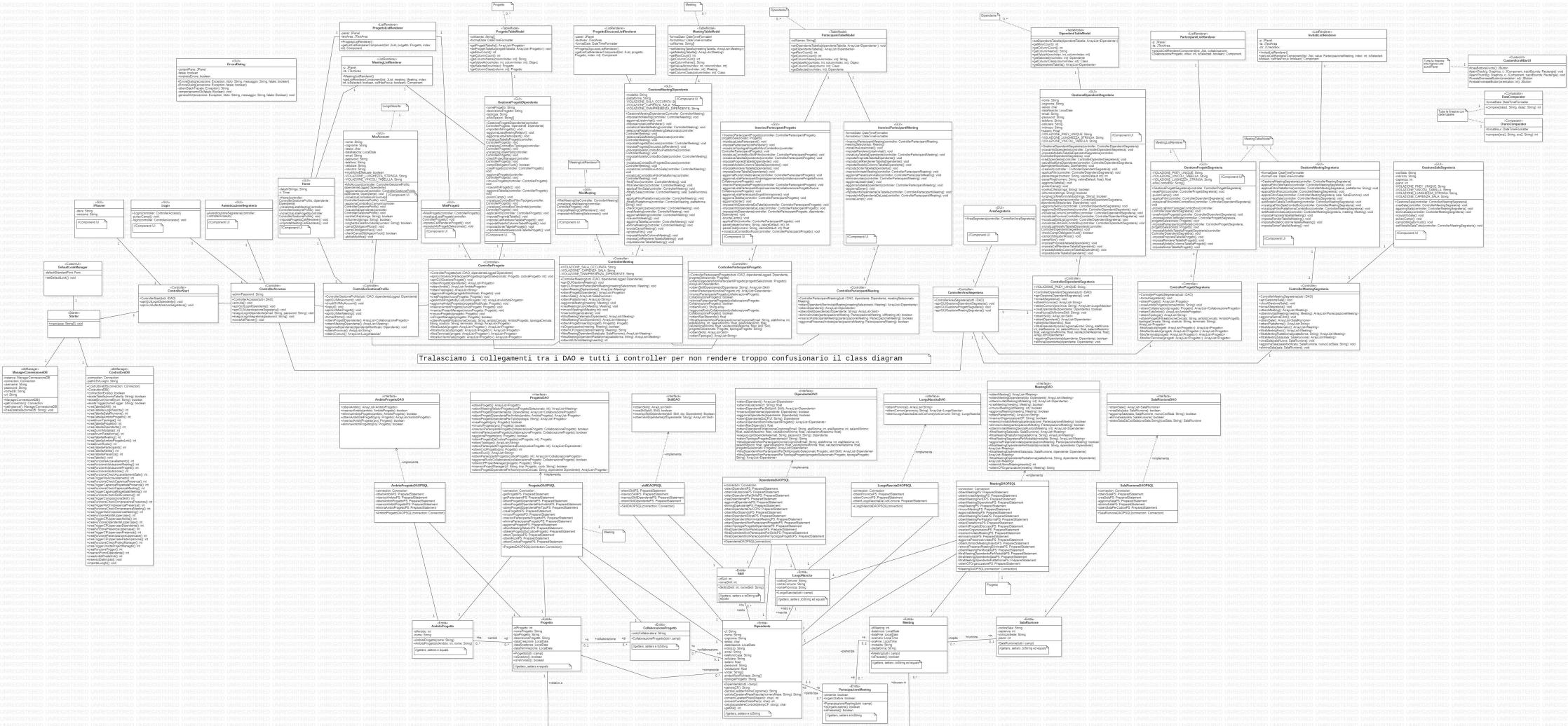
Queste classi hanno il compito di comunicare con il Database, restituendo dei tipi compatibili con Java, in questo modo tutte le altre classi si manterranno indipendenti dalla persistenza di dati e continueranno a funzionare anche nel caso questa dovesse cambiare.

- Successivamente troviamo le InterfacceDAO, che vengono utilizzate come tipo astratto e vanno a "regolare" il comportamento delle classi DAO che le implementano.

In questo modo, nel caso fossero presenti più implementazioni degli stessi DAO (ad esempio dei DAO Oracle), potremmo continuare ad utilizzare gli stessi metodi, che per forza di cose saranno presenti anche nei DAO oracle

- Troviamo poi una serie di controller che si occupano di recuperare le informazioni necessarie dai DAO e di gestire il collegamento tra le relative finestre
- Al di sopra dei controller sono presenti tutte le GUI dell'intero software, con le quali gli utenti andranno ad interfacciarsi
- Sono presenti anche una serie di ListRenderer e TableModel, che si occupano rispettivamente di far visualizzare correttamente le entità all'interno della JList e delle Jtable

In alcuni casi in cui le linee delle associazioni avrebbero dovuto seguire percorsi troppo lunghi, andando quindi a compromettere la leggibilità del class diagram, abbiamo inserito delle note con tanto di cardinalità per simulare i vari collegamenti.



2. CRC Cards

2.1 Entità

Dipendente

| Dipendente | |
|---|---|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">- Contiene tutte le informazioni anagrafiche del dipendente- Contiene tutte le informazioni aziendali del dipendente (salario, valutazione e credenziali di accesso)- Tiene traccia dei meeting, progetti e skill associati al dipendente- Si occupa di generare il codice fiscale di un dipendente a partire dai suoi dati anagrafici | <ul style="list-style-type: none">-LuogoNascita-Skill-PartecipazioneMeeting-CollaborazioneProgetto |

Meeting

| Meeting | |
|--|--|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">-Contiene tutte le informazioni necessarie per rappresentare un meeting (dataInizio, dataFine, orarioInizio, orarioFine, Piattaforma ecc..)-Tiene traccia della sala riunione in cui viene ospitato il meeting, nel caso questo sia fisico-Tiene traccia del progetto relativo che sarà discusso nel meeting-Tiene traccia di tutti i partecipanti al meeting | <ul style="list-style-type: none">-SalaRiunione-Progetto-PartecipazioneMeeting |

Progetto

| Progetto | |
|--|--|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Contiene tutte le informazioni necessarie per rappresentare un progetto (nomeProgetto, descrizioneProgetto, dataCreazione ecc..) -Tiene traccia degli ambiti relativi al progetto -Tiene traccia dei meeting in cui si discute di questo progetto -Tiene traccia dei dipendenti che collaborano al progetto con un determinato ruolo | -AmbitoProgetto -Meeting -CollaborazioneProgetto |

LuogoNascita

| LuogoNascita | |
|---|---------------|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Rappresenta il luogo di nascita di un dipendente attraverso il nome , la provincia ed il codice del comune | -Dipendente |

Skill

| Skill | |
|---|---------------|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Rappresenta le skill lavorative associate ad un dipendente | -Dipendente |

SalaRunione

| SalaRiunione | |
|--|--|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Contiene tutte le informazioni per rappresentare le sale riunioni aziendali , che potranno essere utilizzate per ospitare dei meeting fisici (Codice sala, indirizzo, piano e capienza) -Tiene traccia dei meeting che vengono ospitati nella sala | <ul style="list-style-type: none"> -Meeting |

PartecipazioneMeeting

| PartecipazioneMeeting | |
|--|---|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Si occupa di mantenere le informazioni relative alla partecipazione di un dipendente ad un meeting -Tiene traccia della presenza del partecipante al relativo meeting -Specifica se il partecipante assuma il ruolo di organizzatore o meno del meeting | <ul style="list-style-type: none"> -Meeting -Dipendente |

CollaborazioneProgetto

| CollaborazioneProgetto | |
|---|------------------------------|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Si occupa di mantenere le informazioni relative ai collaboratori in un progetto -Tiene traccia del ruolo che un dipendente può assumere all'interno di un progetto (Project Manager, Team member..) | -Progetto -Dipendente |

AmbitoProgetto

| AmbitoProgetto | |
|---|---------------|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Rappresenta gli ambiti che un progetto può interessare -Tiene traccia di tutti i progetti che hanno un particolare ambito | -Progetto |

2.2 Controller

ControllerStart

| ControllerStart | |
|--|---|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -È il primo controller che viene inizializzato all'avvio del programma , ha il compito di mostrare la finestra iniziale e di indirizzare l'utente verso la finestra di accesso della segreteria o di login del dipendente. | -Iplanner (GUI) -ControllerAccesso -Tutti i DAO |

ControllerAccesso

| ControllerAccesso | |
|---|--|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Si occupa di gestire l'accesso dell'utente al sistema, sia per il lato dipendente che per il lato segreteria -Nel caso in cui le credenziali inserite siano giuste richiama ControllerGestioneProfilo o ControllerAreaSegreteria -Nel caso in cui l'utente decida di annullare l'accesso richiama ControllerStart che mostra di nuovo la prima schermata | -Login (GUI) -AutenticazioneSegreteria(GUI) -Dipendente -ControllerGestioneProfilo -ControllerStart -ControllerAreaSegreteria -Tutti i DAO |

ControllerGestioneProfilo

| ControllerGestioneProfilo | |
|--|---|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Si occupa di gestire la Home page e la finestra del proprio account -Ricava le informazioni per popolare le liste di progetti e meeting relativi al dipendente -Nella schermata Il mio account consente di aggiornare le info del dipendente nel database -Nel caso in cui l utente prema sui buttoni “Miei Progetti” oppure “Miei Meeting” richiama i rispettivi ControllerProgetto e ControllerMeeting, che mostreranno le relative finestre | <ul style="list-style-type: none"> -Home (GUI) -MioAccount (GUI) -Dipendente -CollaborazioneProgetto -Meeting -ControllerProgetto -ControllerMeeting -Tutti i DAO |

ControllerProgetto

| ControllerProgetto | |
|--|---|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Mostra le schermate Miei Progetti e Gestione Progetti in cui un dipendente puo visualizzare, modificare e creare progetti di cui è il Project Manager -Ha una serie di metodi per ricavare le informazioni di un progetto(getPartecipanti , getMeetingRelativi ecc...) -Ha una serie di metodi per modificare, aggiungere e rimuovere progetti dal database | <ul style="list-style-type: none"> -MieiProgetti (GUI) -GestioneProgettiDipendente (GUI) -Dipendente -CollaborazioneProgetto -AmbitoProgetto -Meeting -Tutti i DAO |

ControllerMeeting

| ControllerMeeting | |
|--|---|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Mostra le schermate Miei Meeting e Gestione Meeting in cui un dipendente puo visualizzare, modificare e creare meeting -Gestisce la schermata di inserimento dei partecipanti in un meeting -Ha una serie di metodi per ricavare le informazioni di un meeting (ottieneSale, ottienePiattaforme, ottieneInvitati ecc...) -Ha una serie di metodi per modificare, aggiungere e rimuovere meeting dal database | <ul style="list-style-type: none"> -MieiMeeting (GUI) -GestioneMeetingDipendente (GUI) -InserisciPartecipantiMeeting (GUI) -Dipendente -Meeting -SalaRiunione -Skill -Tutti i DAO |

ControllerDipendentiSegreteria

| ControllerDipendentiSegreteria | |
|---|---|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Si occupa di gestire le finestre e le funzionalità relative alla segreteria -Mostra la finestra di gestione dipendenti oppure torna indietro alla finestra iniziale richiamando ControllerAreaSegreteria -Consente di creare un nuovo account ad un dipendente inserendolo nel database -Consente di creare nuove skills da associare ai dipendenti | <ul style="list-style-type: none"> -GestioneDipendentiSegreteria (GUI) -ControllerAreaSegreteria -LuogoNascita -Skill -Tutti i DAO |

ControllerPartecipantiProgetto

| ControllerPartecipantiProgetto | |
|---|--|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Si occupa di gestire la finestra InserisciPartecipantiProgetto -Ha una serie di metodi per ottenere le info dei partecipanti come ottieniRuoli(), ottieniSkillDipendente()... -Ha dei metodi per eliminare ed inserire dei partecipanti nel progetto | -InserisciPartecipantiProgetto (GUI) -Dipendente -Progetto -Skill -Tutti i DAO |

ControllerPartecipantiMeeting

| ControllerPartecipantiMeeting | |
|--|--|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Si occupa di gestire la finestra InserisciPartecipantiMeeting -Ha una serie di metodi per ottenere le info dei partecipanti come ottieniInvitati()... -Ha dei metodi per eliminare ed inserire dei partecipanti nel meeting | -InserisciPartecipantiMeeting (GUI) -Dipendente -Meeting -Skill -Tutti i DAO |

ControllerProgettiSegreteria

| ControllerProgettiSegreteria | |
|--|--|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <p>-Si occupa di gestire la finestra della segreteria GestioneProgettiSegreteria</p> <p>-Ha una serie di metodi per ottenere tutte le info riguardanti i progetti e le collaborazioni come ottieniAmbitiProgetto() e ottieniCollaborazioni()</p> | <p>-GestioneProgettiSegreteria (GUI)</p> <p>-ControllerAreaSegreteria</p> <p>-AmbitoProgetto</p> <p>-CollaborazioneProgetto</p> <p>-Progetto</p> <p>-Tutti i DAO</p> |

ControllerMeetingSegreteria

| ControllerMeetingSegreteria | |
|---|---|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <p>-Si occupa di gestire le finestre della segreteria GestioneMeetingSegreteria e GestioneSale</p> <p>-Ha dei metodi per passare da una finestra all'altra come apriGestioneSale() e tornaAiPlanner()</p> <p>-Ha una serie di metodi che consentono di ottenere informazioni sui Meetings e sulle Sale Riunione</p> <p>-Ha una serie di metodi che consentono di filtrare le informazioni come filtraMeetingFisici() e filtraMeetingPiattaforma(String piattaforma)</p> | <p>-GestioneMeetingSegreteria (GUI)</p> <p>-ControllerAreaSegreteria</p> <p>-GestioneSaleSegreteria(GUI)</p> <p>-Meeting</p> <p>-PartecipazioneMeeting</p> <p>-SalaRiunione</p> <p>-Tutti i DAO</p> |

ControllerAreaSegreteria

| ControllerAreaSegreteria | |
|---|--|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">-Si occupa di gestire la finestra dell'area iniziale della segreteria.-Ha una serie di metodi per passare alla finestra dei dipendenti, dei meeting e dei progetti | <ul style="list-style-type: none">-AreaSegreteria(GUI)-ControllerDipendentiSegreteria-ControllerProgettiSegreteria-ControllerMeetingSegreteria-Tutti i DAO |

2.3 Elementi GUI

iPlanner

| iPlanner | |
|--|------------------|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -È la prima finestra che si apre all'avvio del programma in cui l'utente sceglie se accedere all'area dipendenti o all'area segreteria -In base alla scelta fatta l'utente verrà indirizzato alla relativa finestra di accesso. | -ControllerStart |

Login

| Login | |
|---|--------------------|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -È la finestra in cui il dipendente andrà ad inserire i dati per fare l'accesso -Ha un metodo login() che, oltre a richiamare verificaCredenziali() nel controller, si occupa di evidenziare in rosso i campi non inseriti e di stampare eventuali messaggi di errore -Ha un metodo svuotaCampi() che viene chiamato nel caso il dipendente sbagli ad inserire le credenziali | -ControllerAccesso |

Home

| Home | |
|---|---|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <p>-È la home page dell'intero programma in cui il dipendente ha una panoramica di tutti i progetti e meeting che lo riguardano</p> <p>-Da questa schermata, tramite i buttoni in alto a destra, è possibile passare alle finestre relative all'account , ai meeting ed ai progetti</p> | -ControllerAccesso -Dipendente -Meeting -Progetto -MeetingListRenderer -ProgettoListRenderer |

MioAccount

| MioAccount | |
|--|--|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <p>-È la finestra in cui il dipendente può visualizzare le sue informazioni personali ed aziendali (valutazione , salario e skill)</p> <p>-Il dipendente può anche modificare le sue info personali , mentre quelle aziendali sono modificabili solo dalla segreteria</p> <p>-Ha un metodo updateAccount() che va a ricavare tutte le informazioni dai campi e con queste richiama il metodo aggiornaInfoDipendente() nel controller</p> | -ControllerGestioneProfilo -Dipendente -Skill -LuogoNascita |

MieiProgetti

| MieiProgetti | |
|---|--|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -È la finestra in cui il dipendente può visualizzare nel dettaglio le informazioni dei suoi progetti -Premendo sul tasto Inserisci/Modifica Progetto il dipendente verrà indirizzato nella finestra di gestione progetti | -ControllerProgetto -Progetto -AmbitoProgetto -ProgettoListRenderer -CustomScrollBarUI |

MieiMeeting

| MieiMeeting | |
|---|--|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -È la finestra in cui il dipendente può visualizzare nel dettaglio le informazioni dei suoi meeting --Premendo sul tasto Inserisci/Modifica Meeting il dipendente verrà indirizzato nella finestra di gestione meeting | -ControllerMeeting -Meeting -MeetingListRenderer -CustomScrollBarUI |

GestioneProgettiDipendente

| GestioneProgettiDipendente | |
|---|--|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">-È la finestra in cui il dipendente può gestire i progetti di cui è il project manager-Tutti i progetti sono organizzati in una tabella in cui vengono mostrate le info-Qui si potranno modificare tutte le info di un progetto, creare nuovi progetti ed eliminare completamente progetti-Premendo sul bottone Inserisci partecipanti il dipendente verrà indirizzato alla finestra InserisciPartecipantiProgetto | <ul style="list-style-type: none">-ControllerProgetto-ProgettoTableModel-Progetto-AmbitoProgetto-MeetingListRenderer-Dipendente-Meeting-CustomScrollBarUI-DataComparator |

GestioneProgettiSegreteria

| GestioneProgettiSegreteria | |
|--|---|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -È la finestra in cui la segreteria può visualizzare le informazioni su tutti i progetti aziendali -Tutti i progetti sono mostrati in una tabella che può essere filtrata in base agli ambiti , alla tipologia, alla scadenza ed alla terminazione -Di ogni progetto vengono mostrati tutti i partecipanti e tutti i meeting relativi in delle liste apposite -In questa finestra è possibile anche creare nuovi ambiti che i dipendenti potranno poi associare ai progetti | <ul style="list-style-type: none"> -ControllerProgettiSegreteria -Progetto -AmbitoProgetto -Meeting -PartecipantiListRenderer -MeetingListRenderer -CustomScrollBarUI -DataComparator -ManagerEccezioniDatiSQLAmbito |

GestioneMeetingDipendente

| GestioneMeetingDipendente | |
|---|---|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -È la finestra in cui il dipendente può gestire i propri meeting -Tutti i meeting sono organizzati in una tabella in cui vengono mostrate le info -Qui si potranno modificare tutte le info di un meeting, creare nuovi meeting ed eliminare completamente dei meeting -Premendo sul bottone Inserisci partecipanti il dipendente verrà indirizzato alla finestra InserisciPartecipantiMeeting | <ul style="list-style-type: none"> -ControllerMeeting -MeetingTableModel -ProgettoDiscussoListRenderer -SalaRiunione -Meeting -Progetto -CustomScrollBarUI -DataComparator -OrarioComparator |

GestioneSaleSegreteria

| GestioneSaleSegreteria | |
|---|---|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -È la finestra in cui la segreteria può aggiungere della Sale Riunioni aziendali, modificare quelle attuali oppure eliminarle -Sulla sinistra vengono mostrate tutte le informazioni della sala (nome, capienza, indirizzo e piano) -Sulla destra vengono mostrate tutte le Sale con relativa capienza in un apposita lista | <ul style="list-style-type: none"> -ControllerMeetingSegreteria -SalaRiunione -CustomScrollBarUI |

GestioneMeetingSegreteria

| GestioneMeetingSegreteria | |
|---|---|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -È la finestra in cui la segreteria può visualizzare le informazioni su tutti i meeting aziendali -Tutti i meeting sono mostrati in una tabella che può essere filtrata in base alla sala, alla piattaforma ed al tipo di meeting(telematico o fisico) -Di ogni meeting vengono mostrati gli invitati in una lista apposita, nella stessa lista l'organizzatore è contrassegnato con una checkbox -Per ogni meeting viene anche mostrato il nome del progetto discusso | <ul style="list-style-type: none"> -ControllerMeetingSegreteria -MeetingTableModel -Meeting -PartecipazioneMeeting -InvitatiListRenderer -CustomScrollBarUI -DataComparator -OrarioComparator |

AreaSegreteria

| AreaSegreteria | |
|---|---------------------------|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -È la finestra che viene aperta dopo aver inserito la password corretta per accedere alla segreteria -Qui l'utente può decidere a quale parte della segreteria accedere -Premendo sui vari label è possibile accedere alla gestione dei dipendenti, dei meeting e dei progetti. | -ControllerAreaSegreteria |

AutenticazioneSegreteria

| AutenticazioneSegreteria | |
|---|--------------------|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -È la finestra in cui l'utente deve inserire la password amministrativa per accedere all'area segreteria. | -ControllerAccesso |

InserisciPartecipantiProgetto

| InserisciPartecipantiProgetto | |
|--|--|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -È la finestra in cui un dipendente può aggiungere ed eliminare partecipanti ad un progetto di cui è il project manager -Nella parte superiore si trovano tutte le info utili del dipendente (salario, valutazione, skill...) e del progetto, mentre nella parte inferiore vengono mostrati tutti i dipendenti in una tabella | -ControllerPartecipantiProgetto -Progetto -PartecipantiTableModel -Dipendente -CustomScrollBarUI |

InserisciPartecipantiMeeting

| InserisciPartecipantiMeeting | |
|---|--|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -È la finestra in cui un dipendente può aggiungere ed eliminare partecipanti ad un meeting --Nella parte superiore si trovano tutte le info utili del dipendente (salario, valutazione, skill...) e del meeting, mentre nella parte inferiore vengono mostrati tutti i dipendenti in una tabella | -ControllerPartecipantiMeeting -Dipendente -Meeting -PartecipantiTableModel -CustomScrollBarUI |

GestioneDipendentiSegreteria

| GestioneDipendentiSegreteria | |
|--|--|
| Superclassi: | JFrame |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -In questa finestra la segreteria può gestire tutte le informazioni legate ai dipendenti -Si possono aggiungere nuovi dipendenti ed eliminare dipendenti già presenti -Si possono modificare tutte le info del dipendente comprese quelle aziendali non modificabili dal dipendente stesso -È possibile inoltre filtrare i dipendenti in base a dei criteri come salario , valutazione ed età | <ul style="list-style-type: none"> -ControllerDipendentiSegreteria -Dipendente -LuogoNascita -Skill -DipendentiTableModel -CustomScrollBarUI |

MeetingListRenderer

| MeetingListRenderer | |
|---|--|
| Superclassi: | Object , IMPLEMENTAZIONI : ListCellRender (Interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Ha lo scopo di far visualizzare i meeting all'interno delle JList in maniera corretta -Fa apparire i meeting di colore rosso quando questi sono scaduti | <ul style="list-style-type: none"> -Meeting |

DipendentiTableModel

| DipendentiTableModel | |
|---|--------------------------------------|
| Superclassi: | AbstractTableModel (Classe astratta) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Ha lo scopo di gestire i dati contenuti nella tabella dei dipendenti -Implementa una serie di metodi ereditati da AbstractTableModel che però sono compatibili con il tipo Dipendente , ad esempio il metodo getSelected() ritorna la riga selezionata sotto forma di un oggetto Dipendente | -Dipendente |

MeetingTableModel

| MeetingTableModel | |
|--|--------------------------------------|
| Superclassi: | AbstractTableModel (Classe astratta) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Ha lo scopo di gestire i dati contenuti nella tabella dei meeting -Implementa una serie di metodi ereditati da AbstractTableModel che però sono compatibili con il tipo Meeting, ad esempio il metodo getValueAt(int colonna,int riga) ritorna un singolo campo del meeting in base alla cella data in input | -Meeting |

PartecipantiTableModel

| PartecipantiTableModel | |
|---|--------------------------------------|
| Superclassi: | AbstractTableModel (Classe astratta) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Ha lo scopo di gestire i dati contenuti nella tabella dei partecipanti di meeting e progetti -Implementa una serie di metodi ereditati da AbstractTableModel che però sono compatibili con il tipo Dipendente | -Dipendente |

ProgettoTableModel

| ProgettoTableModel | |
|---|--------------------------------------|
| Superclassi: | AbstractTableModel (Classe astratta) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Ha lo scopo di gestire i dati contenuti nella tabella dei progetti -Implementa una serie di metodi ereditati da AbstractTableModel che però sono compatibili con il tipo Progetto | -Progetto |

InvitatiListRenderer

| InvitatiListRenderer | |
|---|---|
| Superclassi: | Object , IMPLEMENTAZIONI :ListCellRenderer<PartecipazioneMeeting> (Interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Ha lo scopo di far visualizzare correttamente gli invitati ad un meeting all'interno delle JList -Oltre al nome dell'invitato fa visualizzare un checkbox per indicare se questo sia l'organizzatore del meeting o meno | -PartecipazioneMeeting |

PartecipantiListRenderer

| PartecipantiListRenderer | |
|--|--|
| Superclassi: | Object , IMPLEMENTAZIONI :ListCellRenderer<CollaborazioneProgetto> (Interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Ha lo scopo di far visualizzare correttamente i partecipanti ad un progetto nelle JList -Di ogni partecipante oltre a mostrare il nome, viene mostrato anche il ruolo nel progetto | -CollaborazioneProgetto |

ProgettoListRenderer

| ProgettoListRenderer | |
|---|---|
| Superclassi: | Object , IMPLEMENTAZIONI :ListCellRenderer<Progetto > (Interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Ha lo scopo di far visualizzare correttamente i progetti all'interno delle JList -Di un progetto viene mostrato il nome e la scadenza | -Progetto |

ProgettoDiscussoListRenderer

| ProgettoDiscussoListRenderer | |
|--|---|
| Superclassi: | Object , IMPLEMENTAZIONI :ListCellRenderer<Progetto > (Interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Ha lo scopo di far visualizzare correttamente i progetti discussi in un meeting all'interno delle JList -Di un progetto viene mostrato il nome e le date di creazione, terminazione e scadenza | -Progetto |

CustomScrollBarUI

| CustomScrollBarUI | |
|---|----------------------|
| Superclassi: | BasicScrollBarUI |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Rappresenta una versione personalizzata di una scrollBar, a tema con tutte le altre finestre del software -Viene assegnata a tutte le finestre che hanno uno scrollPane -“Colora” la scrollBar di light gray e il percorso di bianco | |

DataComparator

| DataComparator | |
|--|---|
| Superclassi: | Object , IMPLEMENTAZIONI : Comparator <String > (Interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| -Ha lo scopo di definire un metodo di comparazione corretto tra le LocalDate presenti nel software -Questo verrà poi assegnato ai sorter delle tabelle, In modo che possano ordinare le date in maniera corretta. | |

OrarioComparator

| OrarioComparator | |
|--|---|
| Superclassi: | Object , IMPLEMENTAZIONI : Comparator <String > (Interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| - Ha lo scopo di definire un metodo di comparazione corretto tra i LocalTime presenti nel software -Questo verrà poi assegnato ai sorter delle tabelle, In modo che possano ordinare gli orari in maniera corretta. | |

DefaultLookManager

| DefaultLookManager | |
|---|---------------|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Ha lo scopo di inizializzare alcune proprietà estetiche di default nel programma, come ad esempio il font dei messaggi mostrati nei JOptionPane ed il border dei bottoni -Ha un unico metodo setDefaultLook() che viene chiamato nello starter all'avvio del programma e setta queste proprietà di default per tutte le GUI, utilizzando la classe UIManager. | Starter |

ErroreDialog

| ErroreDialog | |
|--|--|
| Superclassi: | JDialog |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> - È una versione personalizzata di un JDialog di errore che permette di visualizzare anche lo stack trace dell'eccezione lanciata. -L'utente in questo modo, se non sarà in grado di interpretare l'errore, potrà mandare le informazioni agli sviluppatori. -In base al boolean "fatale" in input nel costruttore, permette di mostrare un errore fatale o meno: True: ERRORE FATALE , non appena l'utente andrà a premere su "ok" il programma verrà chiuso false: ERRORE NON FATALE, vengono mostrati comunque l'errore e lo stack trace ma il programma resterà aperto | -tutte le GUI in cui abbiamo bisogno di visualizzare le informazioni dell'errore |

2.4 InterfacceDAO

DipendenteDAO

| DipendenteDAO | |
|--|--|
| Superclassi: | |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">-Contiene le signature dei metodi che andranno ad effettuare operazioni sui dipendenti nel database-Ha lo scopo di “forzare” l’inserimento di questi metodi nelle classi che andranno ad implementarla-Viene utilizzata come tipo astratto indipendentemente dal modo in cui vengono inizializzati i DAO (ad esempio DipendenteDAO può essere sia un DipendenteDAOPSQl che un ipotetico DipendenteDAOORACLE) | <ul style="list-style-type: none">-Dipendente-Skill |

ProgettoDAO

| ProgettoDAO | |
|--|--|
| Superclassi: | |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">-Contiene le signature dei metodi che andranno ad effettuare operazioni sui progetti nel database-Ha lo scopo di “forzare” l’inserimento di questi metodi nelle classi che andranno ad implementarla-Viene utilizzata come tipo astratto indipendentemente dal modo in cui vengono inizializzati i DAO (ad esempio ProgettoDAO può essere sia un ProgettoDAOPSQl che un ipotetico ProgettoDAOORACLE) | <ul style="list-style-type: none">-AmbitoProgetto-CollaborazioneProgetto-Dipendente-Meeting-Progetto |

MeetingDAO

| MeetingDAO | |
|--|---|
| Superclassi: | |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Contiene le signature dei metodi che andranno ad effettuare operazioni sui meeting nel database -Ha lo scopo di “forzare” l’inserimento di questi metodi nelle classi che andranno ad implementarla -Viene utilizzata come tipo astratto indipendentemente dal modo in cui vengono inizializzati i DAO (ad esempio MeetingDAO può essere sia un MeetingDAOPSQl che un ipotetico MeetingDAOORACLE) | <ul style="list-style-type: none"> -Dipendente -Meeting -Progetto -SalaRiunione |

AmbitoProgettoDAO

| AmbitoProgettoDAO | |
|--|--|
| Superclassi: | |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Contiene le signature dei metodi che andranno ad effettuare operazioni sugli Ambiti nel database -Ha lo scopo di “forzare” l’inserimento di questi metodi nelle classi che andranno ad implementarla -Viene utilizzata come tipo astratto indipendentemente dal modo in cui vengono inizializzati i DAO (ad esempio AmbitoProgettoDAO può essere sia un AmbitoProgettoDAOPSQl che un ipotetico AmbitoProgettoDAOORACLE) | <ul style="list-style-type: none"> -AmbitoProgetto -Progetto |

LuogoNascitaDAO

| LuogoNascitaDAO | |
|--|--|
| Superclassi: | |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Contiene le signature dei metodi che andranno ad ottenere informazioni sui Luoghi di nascita dei dipendenti nel database -Ha lo scopo di “forzare” l’inserimento di questi metodi nelle classi che andranno ad implementarla -Viene utilizzata come tipo astratto indipendentemente dal modo in cui vengono inizializzati i DAO (ad esempio LuogoNascitaDAO può essere sia un LuogoNascitaDAOPSQl che un ipotetico LuogoNascitaDAOORACLE) | <ul style="list-style-type: none"> -Dipendente -LuogoNascita |

SalaRiunioneDAO

| SalaRiunioneDAO | |
|---|---|
| Superclassi: | |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Contiene le signature dei metodi che andranno ad effettuare operazioni sulle Sale riunioni nel database -Ha lo scopo di “forzare” l’inserimento di questi metodi nelle classi che andranno ad implementarla -Viene utilizzata come tipo astratto indipendentemente dal modo in cui vengono inizializzati i DAO (ad esempio SalaRiunioneDAO può essere sia un SalaRiunioneDAOPSQl che un ipotetico SalaRiunioneDAOORACLE) | <ul style="list-style-type: none"> -SalaRiunione |

SkillDAO

| SkillDAO | |
|--|--|
| Superclassi: | |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">-Contiene le signature dei metodi che andranno ad effettuare operazioni sulle skill nel database-Ha lo scopo di “forzare” l’inserimento di questi metodi nelle classi che andranno ad implementarla-Viene utilizzata come tipo astratto indipendentemente dal modo in cui vengono inizializzati i DAO (ad esempio SkillDAO può essere sia un SkillDAOPSQL che un ipotetico SkillDAOORACLE) | <ul style="list-style-type: none">-Dipendente-Skill |

2.5 Implementazioni DAO

AmbitoProgettoDAOPSQ

| AmbitoProgettoDAOPSQ | |
|--|--|
| Superclassi: | Object , IMPLEMENTAZIONI: AmbitoProgettoDAO (interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Contiene tutte le implementazioni dei metodi presenti in AmbitoProgettoDAO per postgreSQL -Ha il compito di fare da “tramite” tra il database e le altre classi (non DAO), trasformando il risultato delle query in tipi compatibili con java -Ha una serie di PreparedStatement che permettono di parametrizzare le query oltre che portare vantaggi in termini di efficienza sul database | <ul style="list-style-type: none"> -AmbitoProgetto -Progetto |

DipendenteDAOPSQ

| DipendenteDAOPSQ | |
|--|--|
| Superclassi: | Object , IMPLEMENTAZIONI: DipendenteDAO (interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Contiene tutte le implementazioni dei metodi presenti in DipendenteDAO per postgreSQL -Ha il compito di fare da “tramite” tra il database e le altre classi (non DAO), trasformando il risultato delle query in tipi compatibili con java -Ha una serie di PreparedStatement che permettono di parametrizzare le query oltre che portare vantaggi in termini di efficienza sul database | <ul style="list-style-type: none"> -Dipendente -LuogoNascita -Skill |

LuogoNascitaDAOPSQL

| LuogoNascitaDAOPSQL | |
|--|--|
| Superclassi: | Object , IMPLEMENTAZIONI: LuogoNascitaDAO (interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">-Contiene tutte le implementazioni dei metodi presenti in LuogoNascitaDAO per postgreSQL-Ha il compito di fare da “tramite” tra il database e le altre classi (non DAO), trasformando il risultato delle query in tipi compatibili con java-Ha una serie di PreparedStatement che permettono di parametrizzare le query oltre che portare vantaggi in termini di efficienza sul database | <ul style="list-style-type: none">-Dipendente-LuogoNascita-DipendenteDAO |

MeetingDAOPSQL

| MeetingDAOPSQL | |
|---|--|
| Superclassi: | Object , IMPLEMENTAZIONI: MeetingDAO (interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">-Contiene tutte le implementazioni dei metodi presenti in MeetingDAO per postgreSQL-Ha il compito di fare da “tramite” tra il database e le altre classi (non DAO), trasformando il risultato delle query in tipi compatibili con java-Ha una serie di PreparedStatement che permettono di parametrizzare le query oltre che portare vantaggi in termini di efficienza sul database | <ul style="list-style-type: none">-Dipendente-LuogoNascita-Meeting-Progetto-DipendenteDAO-LuogoNascitaDAO-ProgettoDAO-SalaRiunioneDAO |

ProgettoDAOPSQ

| ProgettoDAOPSQ | |
|--|--|
| Superclassi: | Object , IMPLEMENTAZIONI: ProgettoDAO (interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Contiene tutte le implementazioni dei metodi presenti in ProgettoDAO per postgreSQL -Ha il compito di fare da “tramite” tra il database e le altre classi (non DAO), trasformando il risultato delle query in tipi compatibili con java -Ha una serie di PreparedStatement che permettono di parametrizzare le query oltre che portare vantaggi in termini di efficienza sul database | <ul style="list-style-type: none"> -Dipendente -AmbitoProgetto -Meeting -Progetto -CollaborazioneProgetto -DipendenteDAO -AmbitoProgettoDAO -ProgettoDAO -SalaRiunioneDAO |

SalaRiunioneDAOPSQ

| SalaRiunioneDAOPSQ | |
|--|---|
| Superclassi: | Object , IMPLEMENTAZIONI: SalaRiunioneDAO (interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Contiene tutte le implementazioni dei metodi presenti in SalaRiunioneDAO per postgreSQL -Ha il compito di fare da “tramite” tra il database e le altre classi (non DAO), trasformando il risultato delle query in tipi compatibili con java -Ha una serie di PreparedStatement che permettono di parametrizzare le query oltre che portare vantaggi in termini di efficienza sul database | <ul style="list-style-type: none"> -SalaRiunione |

SkillDAOPSQl

| SkillDAOPSQl | |
|---|--|
| Superclassi: | Object , IMPLEMENTAZIONI: SkillDAO (interfaccia) |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">-Contiene tutte le implementazioni dei metodi presenti in SkillDAO per postgreSQL-Ha il compito di fare da “tramite” tra il database e le altre classi (non DAO), trasformando il risultato delle query in tipi compatibili con java-Ha una serie di PreparedStatement che permettono di parametrizzare le query oltre che portare vantaggi in termini di efficienza sul database | <ul style="list-style-type: none">-Dipendente-Skill |

2.6 dbManager

ManagerConnessioneDB

| ManagerConnessioneDB | |
|--|----------------------|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">-Ha il compito di aprire e chiudere la connessione con il database-Ha una serie di attributi che conservano le proprietà del database cioè username, password, nome ed url-Ha un metodo getInstance() che restituisce la singola istanza del ManagerConnessioneDB richiamando il costruttore privato-Ha un metodo creaDatabase() che va a creare il database nel caso questo non esista già-Ha un metodo getConnection() che restituisce semplicemente la connessione creata nel costruttore privato | |

CostruttoreDB

| CostruttoreDB | |
|---|---------------|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none"> -Contiene la definizione delle tabelle, dei vincoli, dei trigger, e di tutto ciò che è necessario per inizializzare il database prima dell'utilizzo del software -Ha tutta una serie di metodi per andare a creare le singole tabelle ,le funzioni ed i trigger -Ha un metodo generale creaTabelle() che richiama tutti gli altri metodi e va a creare tutte le tabelle -Ha un metodo generale creaFunzioniTrigger() che va a richiamare tutti gli altri metodi per creare tutte le funzioni esterne -Ha un metodo importaLuoghi() che va ad importare nella tabella LuogoNascita tutti i comuni italiani (con nome ,provincia e codice) da un file CSV esterno | |

2.7 Starter

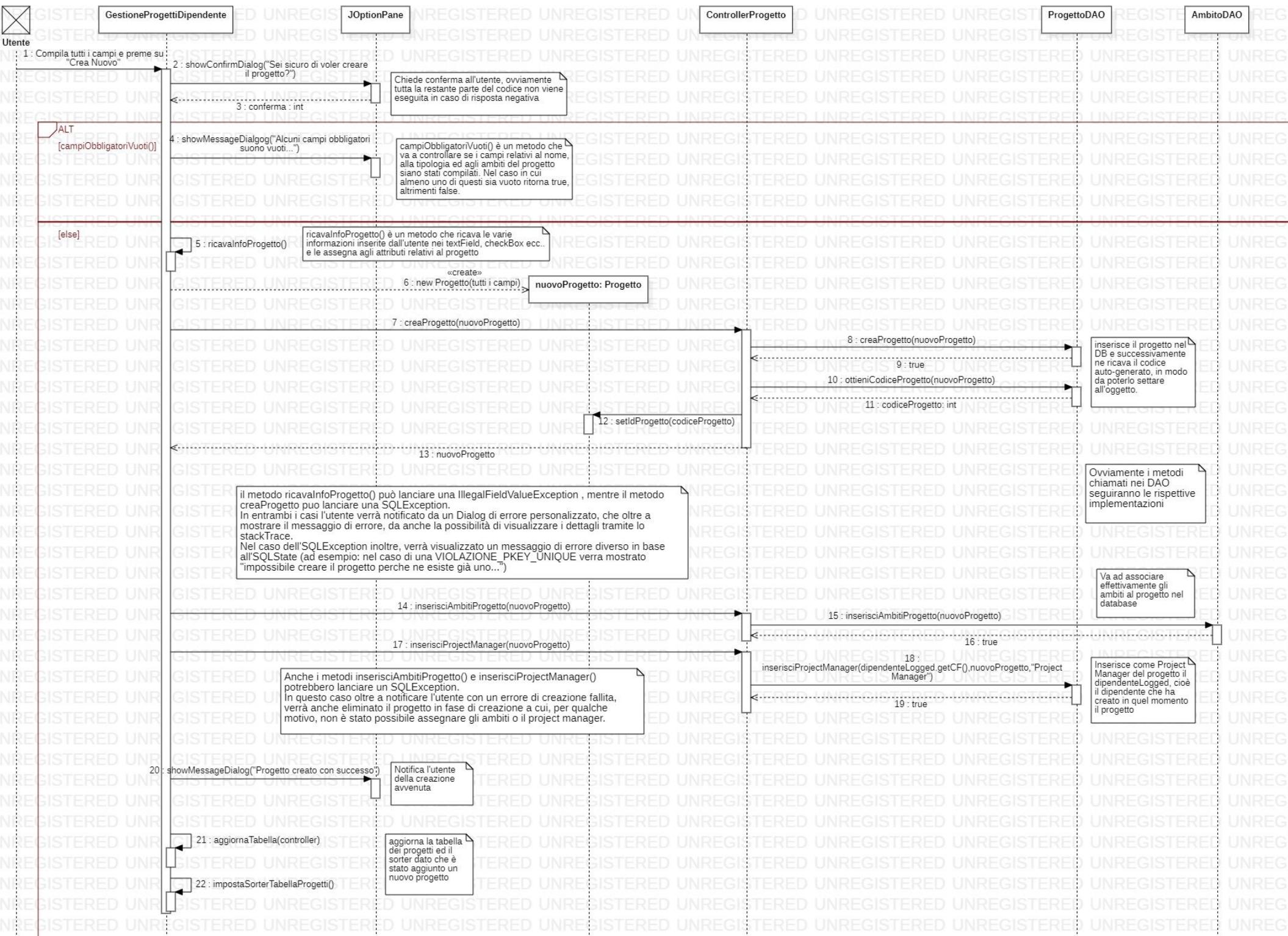
| Starter | |
|---|--|
| Superclassi: | Object |
| Sottoclassi: | |
| Responsabilità | Collaboratori |
| <ul style="list-style-type: none">-È la classe che contiene il main che quindi fa partire l'intero programma-Si connette al database e nel caso questo non esista lo crea, con tutte le tabelle e funzioni-Inizializza tutti i DAO per postgreSQL-Inizializza ControllerStart con tutti i DAO appena creati per avviare la prima schermata del software. | <ul style="list-style-type: none">-ControllerStart-CostruttoreDB-ManagerConnessioneDB-tutti i DAO-DefaultLookManager |

3. Sequence Diagrams

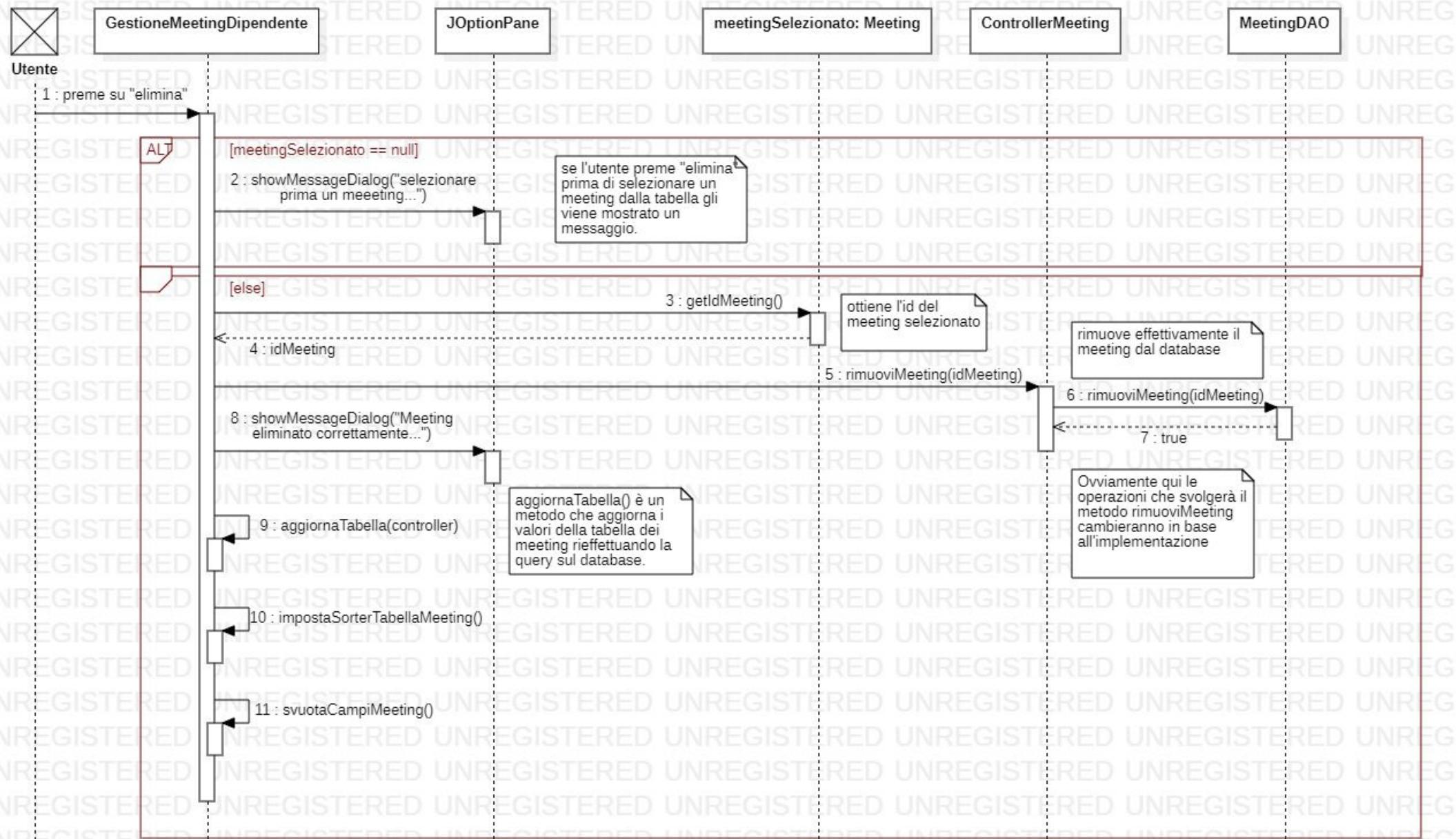
Nelle pagine successive troverete i Sequence Diagram di due funzionalità fondamentali per il funzionamento dell'intero software:

- La creazione di un nuovo progetto da parte del dipendente:
Il sequence diagram descrive l'intera operazione a partire da quando l'utente preme sul bottone "Inserisci/Modifica Progetto" fino a quando il progetto non viene effettivamente inserito nel Database.
- L'eliminazione di un meeting da parte del dipendente:
Il sequence describe l'intera operazione a partire da quando l'utente preme sul bottone "elimina" nella finestra di Gestione Meeting fino a quando il meeting non viene effettivamente eliminato dal database e viene aggiornata la JTable.

Le rappresentazioni dei Sequence Diagram nelle pagine successive possono essere ridimensionate senza perdita di qualità.



sd Eliminazione di un Meeting



rimuoviMeeting() ed aggiornaTabella() possono lanciare una SQLException. Nel caso questa venga lanciata viene mostrato un messaggio di errore all'utente tramite un JDialog di errore personalizzato, ed ovviamente non viene eseguita l'operazione