

# UNIVERSITÀ DEGLI STUDI DI PALERMO DIPARTIMENTO DI INGEGNERIA

CORSO DI LAUREA TRIENNALE IN INGEGNERIA INFORMATICA

# **Object Design Document**

Team GEST-Y

INGEGNERIA DEL SOFTWARE

Castelli Giovanni

Nevaloro Giovanni

Renda Raoul

Costa Nicolò Gabriele



1. Introduzione	2
1.1.Prestazione VS costi	2
1.2.Interfacce	2
1.3.Tempi di risposta	2
1.4.Linee guida per la documentazione delle interfacce	2
1.5.Linee guida per la documentazione del DBMS	2
2. Packages	3
2.1.Com.GEST	3
2.2.Java	6
2.3.MySQL-Connector-Java(JDBC)	6
3. OBJECT DESIGN UML	7

#### 1. INTRODUZIONE

#### 1.1. Prestazione VS costi

Per la realizzazione del software sono state utilizzate librerie Open Source al fine di minimizzare il tempo e i costi ma permettendo di ottenere prestazioni più che soddisfacenti.

#### 1.2. Interfacce:

Le interfacce del software, grazie alla facilità d'utilizzo e al layout semplice e minimale di JavaSwing, permettono un'interazione ottimale, anche per gli utenti meno esperti.

# 1.3. Tempi di risposta:

I tempi di risposta tra server e software sono pressoché assenti grazie al lavoro di ottimizzazione svolto su ogni interrogazione SQL e alla leggerezza del sistema stesso.

Per garantire un sistema scalabile, di facile implementazione e per la semplice ricognizione degli errori si è deciso di realizzare un sistema modulare.

Il sistema rispetta una architettura di tipo Repository, essa permette la disgiunzione dei sottosistemi [vedi SSD – scomposizione in sottosistemi]

La DBMSBoundary si occupa di gestire le comunicazioni con il database.

#### 1.4. Linee guida per la documentazione delle interfacce

Per la realizzazione delle interfacce si è deciso di utilizzare il pacchetto di JavaSwing, esso permette di realizzare interfacce grafiche immediate, essenziali ed efficaci, permettendo all'utilizzatore un facile utilizzo del software proprio per la sua intuitività.

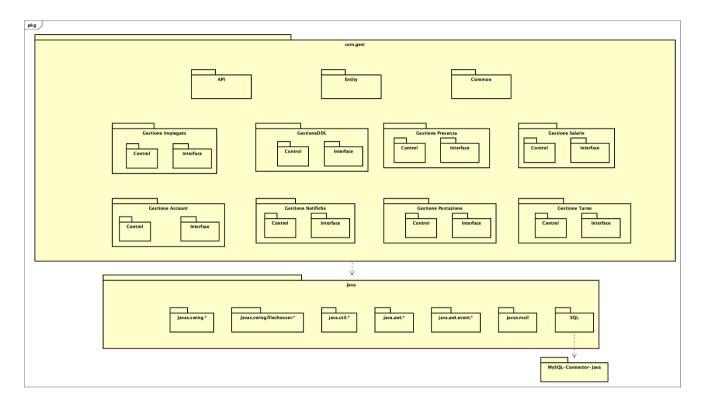
# 1.5. Linee guida per la documentazione del DBMS

Per i DBMS si è utilizzato un database derivato da MySQL, nello specifico, abbiamo usato MariaDB.

Per l'interazione tra il sistema e il DBMS si è deciso di utilizzare MySQL-Connector-Java (Java DataBase Connectivity – JDBC).

I driver di JDBC garantiscono, per ogni tipo di DBMS, la persistenza dei dati sulle basi di dati, oltre all'accesso e alla gestione di esse per mezzo di Java.

# 2. PACKAGES



# **2.1. COM.GEST**

Il package principale dell'applicazione

# 2.1.1. API

Contiene classi wrapper per API, il pacchetto è specializzato per il sistema e quindi poco riutilizzabile in altri contesti.

#### 2.1.2. Entity

Contiene le classi che modellano le entità di cui è necessaria la rappresentazione nel sistema.

#### **2.1.3.** Commons

Contiene classi comuni agli 8 sottosistemi [vedi SDD].

#### 2.1.4. Gestione Account

Vi risiedono tutti i pacchetti necessari che devono essere installati sul sistema, nello specifico è presente l'interfaccia che contiene i file necessari alla gestione delle interfacce grafiche e Control che contiene le classi collegate ad esse, che gestiscono la logica dell'applicazione.

Nello specifico permette di autenticarsi, recuperare le credenziali e modificare i dati del proprio profilo.

#### 2.1.5. Gestione Impiegato

Vi risiedono tutti i pacchetti necessari che devono essere installati sul sistema, nello specifico è presente l'interfaccia che contiene i file necessari alla gestione delle interfacce grafiche e Control che contiene le classi collegate ad esse, che gestiscono la logica dell'applicazione.

Nello specifico permette di cercare un impiegato e di visualizzare e modificare i dati di un impiegato selezionato. Fornisce l'interfaccia ListaImpiegatiBnd per permettere ai package **Gestione Turno** e **Gestione Salario** di visualizzare inoltre i turni e le buste paga dell'impiegato selezionato.

#### 2.1.6. Gestione Turno

Vi risiedono tutti i pacchetti necessari che devono essere installati sul sistema, nello specifico è presente l'interfaccia che contiene i file necessari alla gestione delle interfacce grafiche e Control che contiene le classi collegate ad esse, che gestiscono la logica dell'applicazione.

Nello specifico permette di visualizzare i turni di un impiegato selezionato dalla ListaImpiegatiBnd e di visualizzare il calendario giornaliero di presa di servizio. Permette inoltre agli impiegati di visualizzare il proprio calendario personale e di richiedere un periodo di astensione dal lavoro. Contiene inoltre le classi per poter gestire la richiesta di astensione.

#### 2.1.7. Gestione Salario

Vi risiedono tutti i pacchetti necessari che devono essere installati sul sistema, nello specifico è presente l'interfaccia che contiene i file necessari alla gestione delle interfacce grafiche e Control che contiene le classi collegate ad esse, che gestiscono la logica dell'applicazione.

Nello specifico permette di visualizzare lo storico delle buste paga emesse in un intervallo mesi.

#### 2.1.8. Gestione Postazione

Vi risiedono tutti i pacchetti necessari che devono essere installati sul sistema, nello specifico è presente l'interfaccia che contiene i file necessari alla gestione delle interfacce grafiche e Control che contiene le classi collegate ad esse, che gestiscono la logica dell'applicazione.

Nello specifico permette di registrare la presa di servizio dalla postazione in azienda (firmare ingresso e uscita). Contiene inoltre le classi che permettono di notificare un eventuale ritardo (ingresso ed uscita) ed uscita, calcolare gli stipendi e di generare il calendario trimestrale.

#### 2.1.9. Gestione Presenza

Vi risiedono tutti i pacchetti necessari che devono essere installati sul sistema, nello specifico è presente l'interfaccia che contiene i file necessari alla gestione delle interfacce grafiche e Control che contiene le classi collegate ad esse, che gestiscono la logica dell'applicazione.

Nello specifico permette di rilevare la presenza dal terminale e per visualizzare la lista delle presenze e assenze effettuate.

#### 2.1.10. Gestione Notifiche

Vi risiedono tutti i pacchetti necessari che devono essere installati sul sistema, nello specifico è presente l'interfaccia che contiene i file necessari alla gestione delle interfacce grafiche e Control che contiene le classi collegate ad esse, che gestiscono la logica dell'applicazione.

Nello specifico permette di visualizzare comunicazioni e notifiche.

#### 2.1.11. Gestione DDL

Vi risiedono tutti i pacchetti necessari che devono essere installati sul sistema, nello specifico è presente l'interfaccia che contiene i file necessari alla gestione delle interfacce grafiche e Control che contiene le classi collegate ad esse, che gestiscono la logica dell'applicazione.

Nello specifico permette di inviare comunicazioni, gestire eventuali licenziamenti, assunzioni e modifica ruolo. Contiene inoltre le classi che gli permetto di approvare le ferie richieste dagli impiegati e di pagare le buste paga calcolate dal sistema.

#### **2.2. JAVA**

Vi sono tutte le librerie ed i package standard di Java.

#### Tra cui:

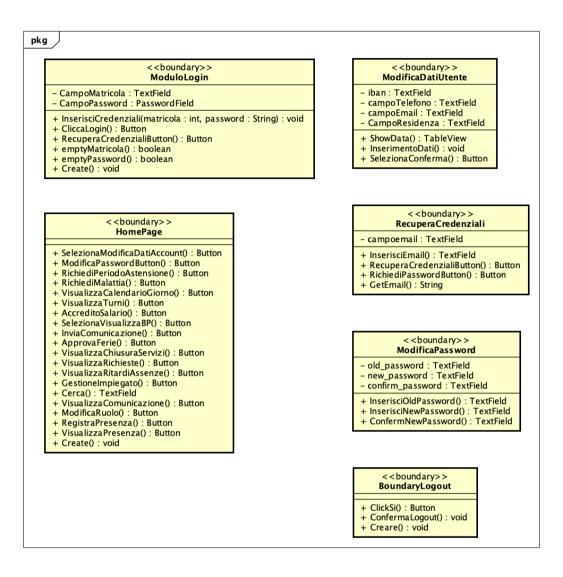
- Utils: Presenta tutte le strutture dati utilizzate per la realizzazione del sistema;
- **SQL:** Gestisce le comunicazioni con i DBMS, attraverso la DBMSBoundary che esegue le query al fine di recuperare o inserire informazioni all'interno del DBMS;
- **Time:** API per l'utilizzo e la gestione di date e orari al fine di realizzare la turnazione ed effettuare le opportune verifiche sui turni per il controllo di ritardi, assenze o firme mancate;
- **Javax.mail:** API che consente comunicazioni SMTP, nello specifico viene utilizzata per l'invio delle email al fine di effettuare comunicazioni su eventuali cambio turno, approvazione di ferie o mail di recupero password per eventuali dimenticanze da parte dell'utilizzatore.

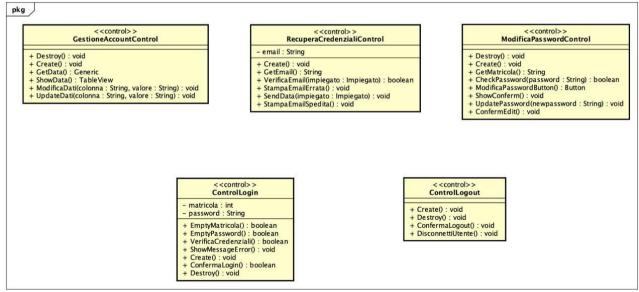
# 2.3. MySQL-Connector-Java(JDBC)

Nel JDBC vi sono i driver per le interazioni con il DBMS.

#### 3. OBJECT DESIGN UML

#### 3.1. Gestione Account



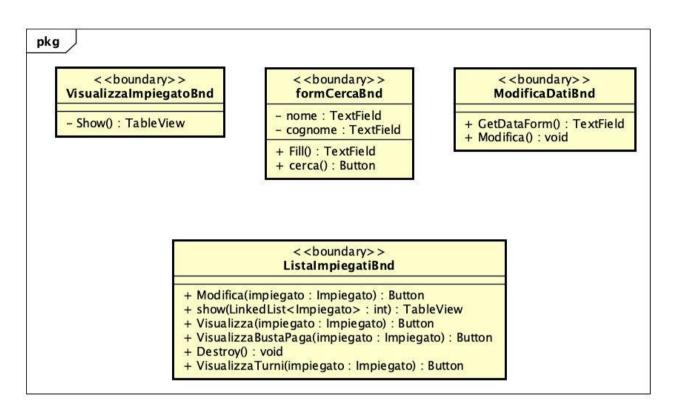


#### 3.2. Gestione Turni

#### pkg <<body><br/><<br/>boundary>></br/> <<body><br/><<br/>boundary>></br/> <<body><br/><<br/>boundary>></br/> VisualizzaTurniBnd **FormRichiestaROL** FormInserisciGiorno - CampoGiorno(): Date + ShowLista() : TableView + RichiediROL() : Button Inserisci(): void + GetDataForm(): void + SelezionaGiorno(): void GetGiornoSelezionato(): Date <<body><<body><<br/><<br/> <<body><<body><<br/><<br/> **FormRichiediAstensione** ListalmpiegatiBnd - CampoCertificato: URL + Modifica(): void CampoGiorno: LinkedList<Date> + Visualizza(): void - CampoTipoCongedo : String + VisualizzaBustaPaga(): TableView - CampoTipoAstensione: String + Destroy() : void + VisualizzaTurni() : TableView + RichiediCongedoFamiliare(): Button + CaricaCertificato() : Button + SelezionaTipoCongedo(): ChoiceBox<String> + GetDataForm() : Generic + SelezionaGiorno() : ChoiceBox<Date> + CaricamentoDocumento() : UploadField

#### pkg <<control>> RichiediAstensioneCtl - DatiForm : TextField - GiornoRilevato : LinkedList<Date> + Destroy() : void + Create() : void + GetDataForm() : Generic + GetMatricola() : int <<control>> GestioneCalendarioCtl - GiornoRilevato : LinkedList<Date> GetGiorniRimanenti(): int - impiegato : Impiegato - Indisponibilità : LinkedList<Date> - GetCurrentDate() : Date - GetCurrentDate() : Date - GetEmail() : String - SendEmail(destinatario : String, oggetto : String, corpo : String) : void - GetFerieRichieste() : LinkedList<Ferie> + GetListaImpiegati() : LinkedList<Impiegato> GetPeriodiAstensione(): LinkedList<PeriodoAstensione> InserisciCalendario(date: LinkedList<Date>): void InserisciComunicazione(comunicazione: String): void + InsertRichiesta(richiesta : Richiesta) : void + RichiediROL(): void + Destroy() : void + GetTurno() : Turno Create(): void GetDataForm(): Generic <<control>> + GetDatarorm() : Generic + UpdateOrario() : void + GetListaTurni() : LinkedList<Turno> + ShowLista() : TableView + GetMatricola() : int GestioneAssenzaCtl - Richiesta Astensione : Richiesta + GetMatricola(): int + GetDay(): Date AggiornamentoIndisp(notifica : Notifica) : void GetTurno() : Turno CheckImpiegatoDisponibile(impiegato: Impiegato): boolean Checkimplegato isponibile (implegato: implegato): boolean Sostituiscilmplegato (implegato): void send Mail (destinatario: String, oggetto: String, corpo: String): void VerifyServizio(turno: Turno): boolean Rimuovilmplegato(matricola: int): void CheckPriorita(turno: Turno): boolean InsertNotificachiusura Turno(notifica: Notifica): void EliminaTurno(id\_turno : int) : void GetImpiegatoServizioBasso(matricola : int) : Impiegato CambiaTurnoServizio(id\_turno: int): void

# 3.3. Gestione Impiegato



#### pkg

# <<control>> GestioneImpiegatoCtI

- Listalmpiegati : LinkedList<Impiegato>
- + Create() : void + Destroy() : void
- + GetDataForm() : DataPicker
- + ExistImpiegato(matricola: int): boolean
- + Error(): void
- + InsertImpiegato(): void + GetMatricola(): String

# <<control>>

#### CercalmpiegatoCtl

- Listalmpiegati : LinkedList<Impiegato>
- + Create() : void + Destroy() : void
- + GetFormCercaData(): void
- + GetListaImpiegatiFiltrati(nome: String, congnome: String): LinkedList<Impiegato>
- + Show(): TableView
- + Error(): void

# 3.4. Gestione Salario

# pkg <<body><<body> <<body><br/><<br/>boundary>></br/></br/> VisualizzaBustaPaga ListalmpiegatiBnd - bustapaga : BustaPaga + Modifica(): void + Show(): TableView + Visualizza() : TableView + InserisciMesi(): void + VisualizzaBustaPaga(): TableView + Destroy(): void + VisualizzaTurni(): TableView <<body><br/><<br/>boundary>></br/>> FormIntervalloMesi + InserisciMesi(meseinizio: TextField, mesefine: TextField): void + GetMesi(): LinkedList<int> + SelectMesi(LinkedList<int>: int): void + GetIntervalloDate(): LinkedList<Date>

# pkg

# <<control>> GestioneSalarioControl

OrarioRilevato : HourMeselnizio : int

- MeseFine : int - turno : Turno

- ListaBustaPaga : LinkedList<BustaPaga>

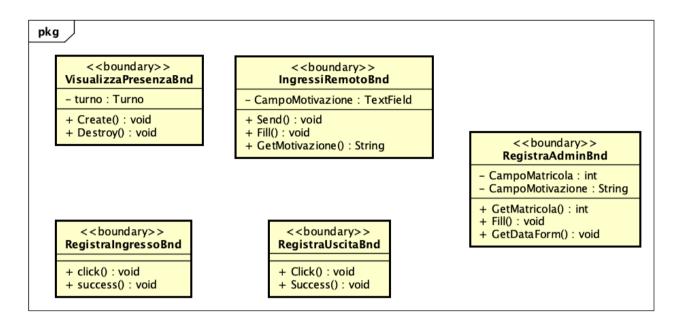
+ GetListaImpiegati(): LinkedList<Impiegato>

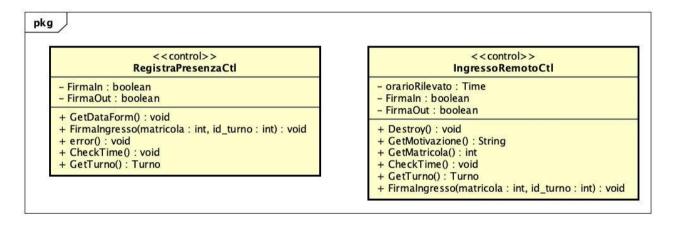
+ CheckTime(): void + EffettuaCalcolo(): void + Updatetotale(): void

+ InserisciBustaPaga(bustapaga : BustaPaga) : void

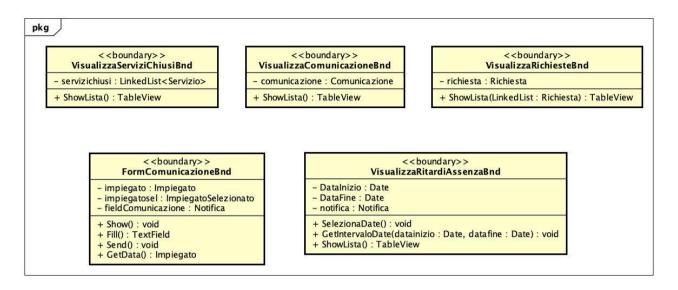
+ InserisciComunicazione(comunicazione : String) : void

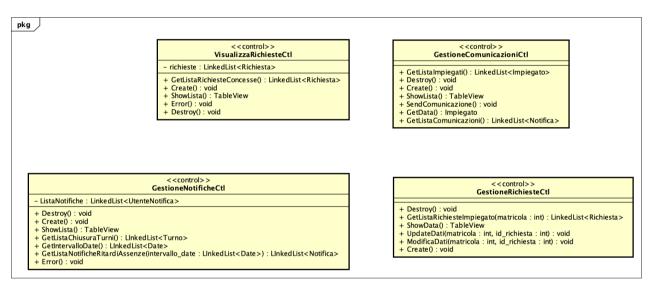
# 3.5. Gestione presenza





# 3.6. Gestione Notifiche



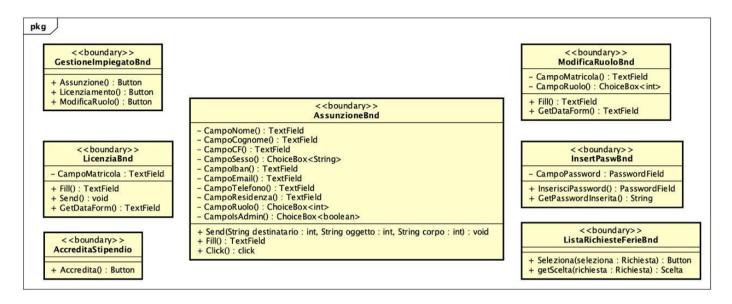


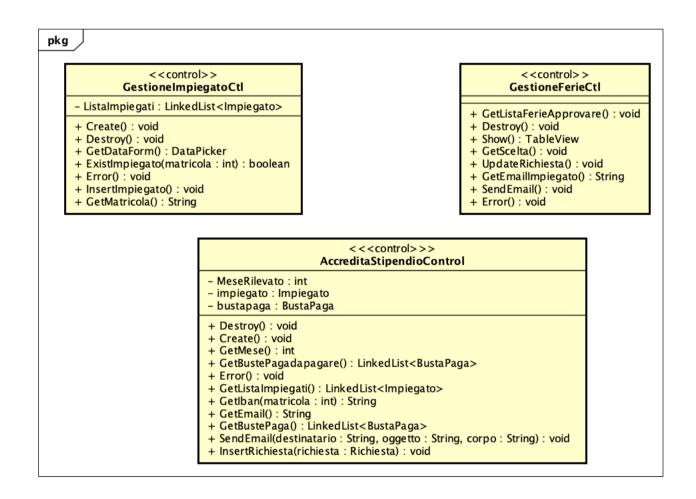
#### 3.7. Gestione Postazione

# c<boundary>> FormPostazioneBnd - CampoNome : String - CampoCognome : String - CampoMatricola : int + Fill() : void + invia() : void + GetDataForm() : TextField

#### pkg <<control>> NotificaRitardoCtl <<control>> GestioneCalendarioCtl - GiornoRilevato : LinkedList<Date> - TempoRilevato : Time impiegato : Impiegato + CheckTime(): void + CheckTurniFiniti(data: Date): LinkedList<Turno> - Indisponibilità : LinkedList<Date> GetListalmpiegati(): LinkedList<Impiegato> GetPeriodiAstensione(): LinkedList<PeriodoAstensione> InserisciCalendario(date: LinkedList<Date>): void + CheckRitardoUscita(matricola : int, id\_turno : int) : boolean + NotifyRitardoUscita(matricola : int, id\_turno : int) : void GetEmailImpiegato(): String Inserisci Comunicazione (comunicazione: String): voidSendEmail(destinatario: String, oggetto: String, corpo: String): void InserisciComunicazione(c RichiediROL() : void Destroy() : void GetTurno() : Turno Create() : void GetDataForm() : Generic + Sende mail(destinatario : String, oggetto : String, corpo : String + InserisciFirma() : void + UpdateAssenza(matricola : int, mese : int, anno : int) : void + CheckRitardoIngresso(matricola : int, id\_turno : int) : boolean + CheckFirmaIngresso(matricola : int, id\_turno : int) : boolean NotifyRitardoIngresso(matricola : int, id\_turno : int) : void UpdateNumRitardo(matricola : int, mese : int, anno : int) : void UpdateOrario() : void GetListaTurni() : LinkedList<Turno> ShowLista() : TableView GetMatricola() : int GetDay(): Date <<control>> <<control>> CalcoloSalarioControl RegistraPresenzaCtl - Firmaln : boolean + GetListaImpiegati(): LinkedList<Impiegato> - FirmaOut : boolean CheckTime(): void EffettuaCalcolo(salario: float): void Updatetotale(salario: float): void InserisciBustaPaga(bustapaga: BustaPaga): void InserisciComunicazione(comunicazione: Notifica): void + GetDataForm() : void + Firmalngresso(matricola : int, id\_turno : int) : void error(): void + CheckTime() : void + GetTurno() : Turno

# 3.8. Gestione DDL





#### 3.9. Common

#### pkg Email <<body><<body><<br/> MessageFrame + sendMail(destinatario : String, oggetto : String, corpo String : int) : void + Click(): void < <boundary> > <<body><br/><<br/>boundary>></br/> <<body><br/>< boundary>></br/>> **Boundary Tempo** <<bod><<br/>boundary>> AvvisoConferma AvvisoError AvvisoSuccesful OraRilevata: int GiornoRilevato: int + Create() : void + okButton() : Button + Create() : void + okButton() : Button + create() : void + okButton() : Button - DataRilevata: Date + Check(): void + Destroy(): void + Destroy(): void + Destroy() : void Create(): void <<body><br/><<br/>boundary>></br/> DRMS + VerificaCredenziali(matricola : int, password : String) : boolean + UpdateDati(colonna : String, valore String : int) : void + UpdatePassword(matricola : int, password : String) : void + VerificaEmail(email : String) : boolean + GetListaImpiegato(): LinkedList<Impiegato> + GetPeriodiAstensione(): LinkedList < Date > + InserisciComunicazione(comunicazione : Comunicazione)() : void + AggiornamentoIndisp(matricola: int, data\_form: Date): void + InserisciRichiesta (matricola : int, tipologia : String, datainizio : Date, datafine : Date, motivazione : String) : void + GetTurno(giorno : Date, matricola : int) : Turno + CheckImpiegatoDisponibile(turno : Turno, data\_richiesta : Date) : boolean + SostituisciImpiegato(impiegato\_disp: Impiegato, turno: Turno): void + VerifyServizio(turno : Turno) : boolean + Rimuovilmpiegato(turno: Turno): void + CheckPriorità(turno : Turno, giorno : Date) : boolean + GetImpiegatoServizioBasso(giorno : Date) : Impiegato + CambiaTurnoServizio(new\_sostituto : Impiegato) : void + GetGiorniRimanenti(matricola: int): int + GetFerieRichieste(matricola : int) : LinkedList<Richiesta> + UpdateOrario(turno : Turno, Data\_form : Date) : void + GetListaTurni() : LinkedList<Turno> + InsertNotificaChiusuraTurno(notifica : Notifica) : void + GetBustePagaDaPagare(mese: int): LinkedList<BustaPaga> + GetBustaPaga(impiegato: Impiegato): LinkedList<BustaPaga> + UpdateListalmpiegati(): void + UpdateTotale(matricola: int, totale: int, mese: int, anno: int): void + InserisciBustaPaga(bustapaga: BustaPaga): void + SendComunicazione(comunicazione : String) : void + CheckTurniFiniti(data : Date) : LinkedList<Turno> + UpdateAssenza(matricola: int, mese: int, anno: int): void + GetEmailImpiegato(): String + checkTurni(data : Date) : LinkedList<Turno> + UpdateNumRitardo(matricola : int, mese : int, anno : int) : void + InserisciFirma(matricola : int, id\_turno : int) : void + GetListaComunicazioni(matricola: int, id\_turno: int): LinkedList<Comunicazione> + GetListaRichiesteConcesse(): LinkedList<Richiesta> + GetListaNotificaRitardiAssenze(): LinkedList<Utente\_Notifica> + GetChiusuraTurni(): LinkedList<Utente Notifica> + GetListaFerieDaApprovare(): LinkedList<Richiesta> + GetMatricola(): int + GetListaImpiegatiFiltrati(nome: String, congnome: String): LinkedList<Impiegato> + UpdateRuolo(ruolo: Ruolo, matricola: int): void + GetBustePaga(impiegato : Impiegato) : LinkedList<BustaPaga> + InserisciCalendario(turni : LinkedList<Turno>) : void + InserisciComunicazione(comunicazione : String, matricola : int) : void

# **3.10.** Entity

