Laurea Triennale in informatica-Università di Salerno
Corso di *Ingegneria del Software*-Prof.ssa F. Ferrucci–Prof. F. Palomba

Object Design Document

BeeHave

Riferimento	C14_ODD_ver1.0
Versione	1.0
Data	25/01/2023
Destinatario	Prof.ssa Filomena Ferrucci, Prof. Fabio Palomba
Presentato da	C14 Team A.P.Hive
Approvato da	Gianmario Voria, Antonio Della Porta

Revision History

Data	Versione	Descrizione	Autori
06/12/2022	0.1	Prima stesura	GV, ADP
09/12/2022	0.2	Definizione Packages	LB, MLF
10/12/2022	0.3	Specifica Class Interfaces: GU, GA	LB, IG
11/12/2022	0.4	Specifica Class Interfaces: GV, GAU	MLF, GR, TDP
20/01/2023	0.5	Revisione per consegna	MLF
25/01/2023	1.0	Revisione del Documento	Tutto il Team

Team Members

Nome	Ruolo	Acronimo	Contatto
Gianmario Voria	Project Manager	GV	g.voria6@studenti.unisa.it
Antonio Della Porta	Project Manager	ADP	a.dellaporta26@studenti.unisa.it
Luigi Bacco	Team Member	LB	l.bacco2@studenti.unisa.it
Irene Gaita	Team Member	IG	i.gaita1@studenti.unisa.it
Maria Lucia Fede	Team Member	MLF	m.fede1@studenti.unisa.it
Gianluca Ronga	Team Member	GR	g.ronga5@studenti.unisa.it
Thomas De Palma	Team Member	TDP	t.depalma@studenti.unisa.it

Sommario

1. Introduzione

BeeHave punta a diventare una piattaforma che permetta una più facile interazione tra gli apicoltori ed i propri clienti, col fine ultimo di supportare la diffusione delle api a livello nazionale.

In questa prima sezione del documento verranno elencate: le linee guida per la fase di implementazione, la nomenclatura e la documentazione.

1.1. Linee Guida per la stesura del codice

Le linee guida includono una lista di standard che gli sviluppatori devono seguire per la progettazione delle interfacce. Di seguito viene presentata una lista di link alle documentazioni ufficiali:

- https://google.github.io/styleguide/htmlcssguide.html
- https://google.github.io/styleguide/pyguide.html

1.2. Definizioni, Acronimi e Abbreviazioni

- **GU:** Gestione Utente
- **GAU:** Gestione Assistenza Utente
- **GV:** Gestione Vendita
- **GA:** Gestione Adozioni
- **ODD:** Object Design Document
- **HTML:** HyperText Markup Language
- **CSS:** Cascading Style Sheets
- **JS:** JavaScript

1.3. Riferimenti

Libro: -- Object-Oriented Software Engineering (Using UML, Patterns, and Java) Third Edition
Autori: -- Bernd Bruegge & Allen H.

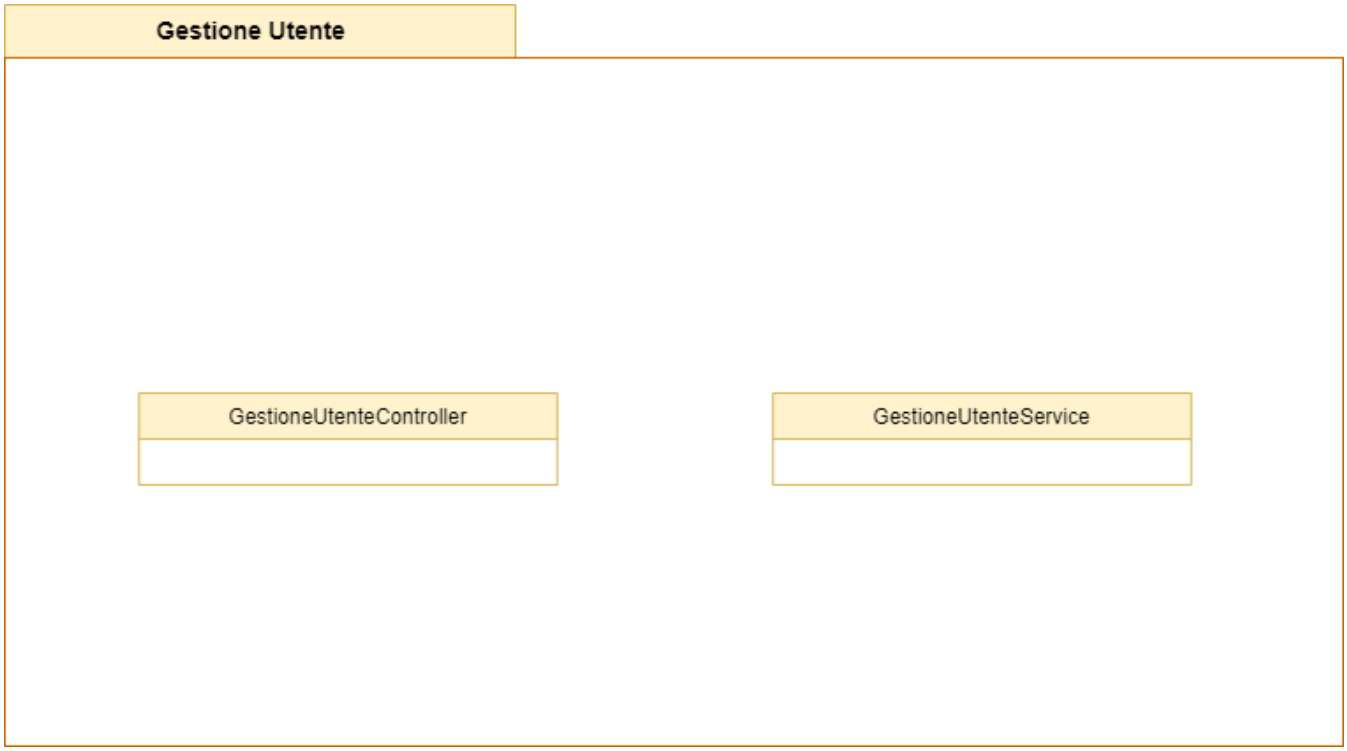
- Di seguito una lista di riferimenti ad altri documenti utili durante la lettura:
  - Statement Of Work;
  - System Design Document;
  - Test Plan;
  - Test Case Specification;
  - Matrice di tracciabilità;
  - Requirements Analysis Document;
  - Manuale Utente;
  - Manuale di Installazione;
  - Test Incident Report;
  - Test Summary Report;

2. Packages del sistema

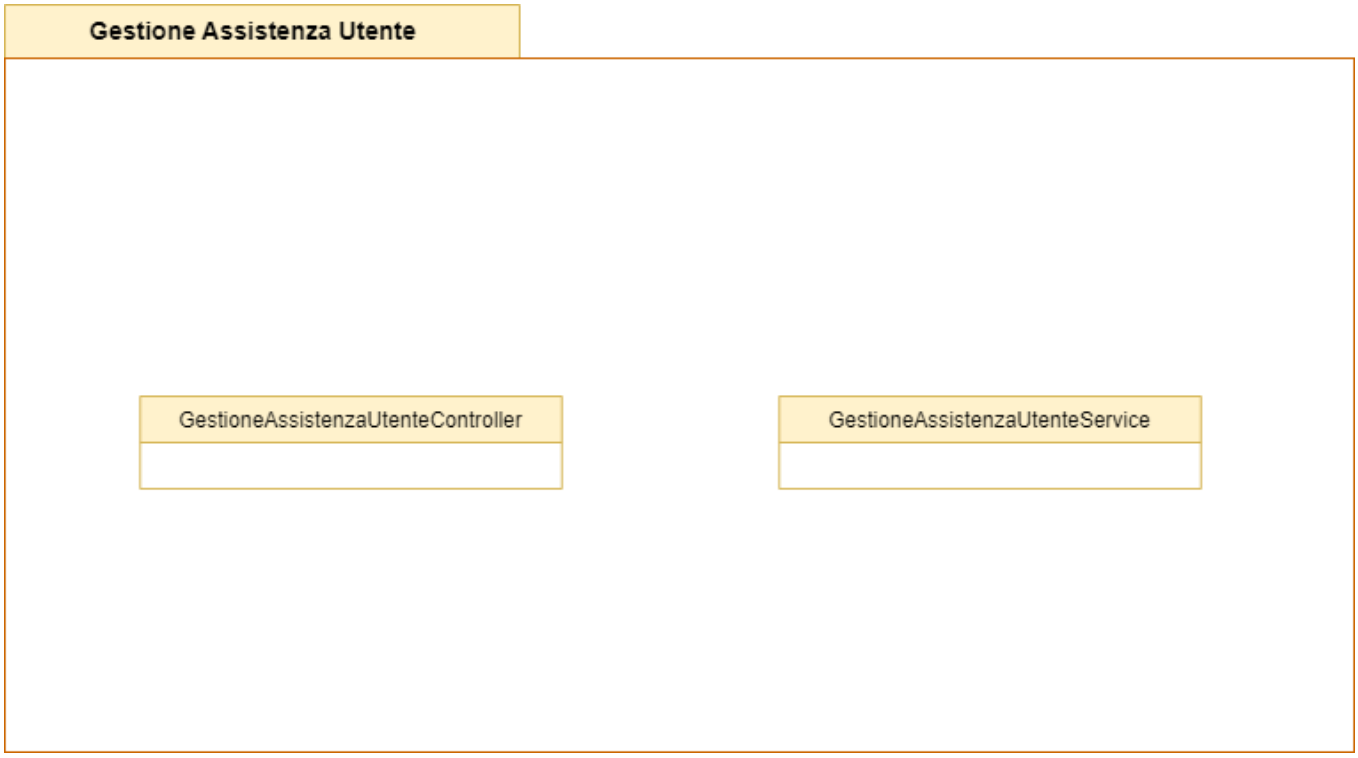
In questa sezione viene mostrata la suddivisione del sistema in package, in base a quanto definito nel documento di System Design. Tale suddivisione è motivata dalle scelte architetturali prese e ricalca la struttura di directory standard definita da Flask.

- **flaskr/**, una cartella che contiene tutti i packages e files del codice sorgente
  - **templates/**, una cartella contenente tutti le pagine html cartelle per l'interfaccia grafica
  - **static/**, una cartella contenente le cartelle contenenti file CSS, JS ed immagini utilizzati per l'interfaccia grafica
- **tests/**, una cartella che contiene i moduli di test
- **venv/**, un ambiente virtuale dove sono installati Flask e altre librerie

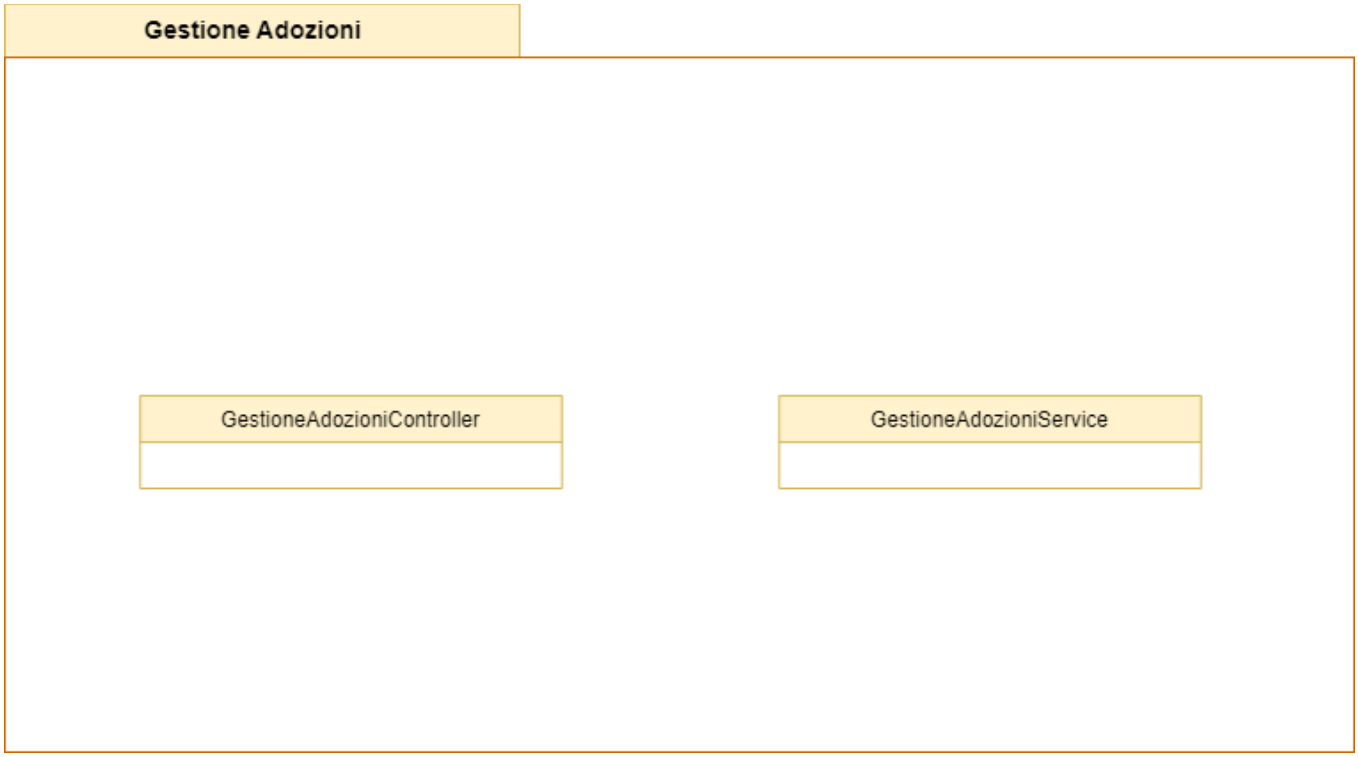
PACKAGE GESTIONE UTENTE



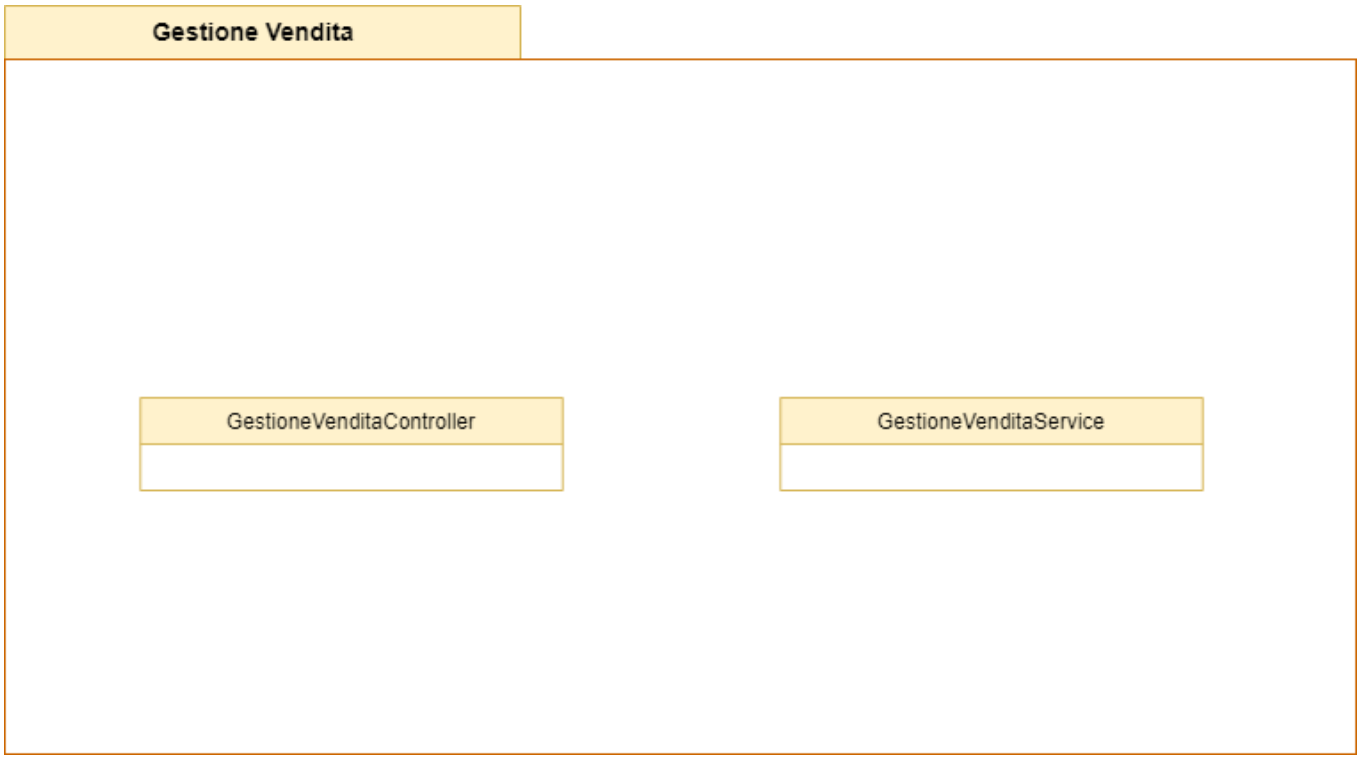
PACKAGE GESTIONE ASSISTENZA UTENTE



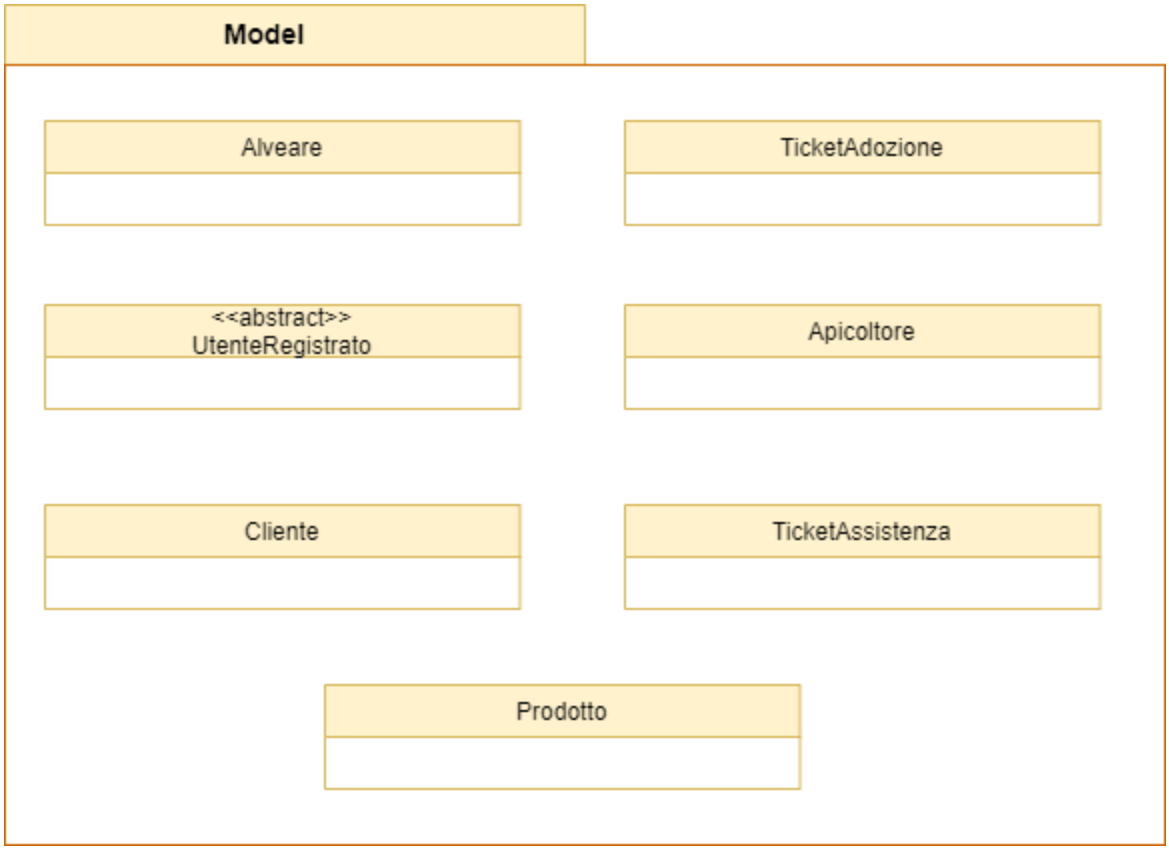
PACKAGE GESTIONE ADOZIONI



PACKAGE GESTIONE VENDITA



PACKAGE MODEL



3. Class Interfaces

PACKAGE GESTIONE VENDITA

Nome Classe	GestioneVenditaService
Descrizione	Questa classe implementa il sistema di gestione vendita, permettendo l'acquisto e la vendita dei prodotti. Inoltre, gestisce anche i metodi riguardanti la gestione del catalogo dei prodotti.
Metodi	+**inserisci_prodotto**(String nome, String descrizione, String località, String peso, String tipologia, String prezzo, String quantità, Apicoltore apicoltore): Boolean +**cancella_prodotto**(int prodotto_id): Boolean +**get_tutti_prodotti():** List Prodotto +**get_prodotto_by_id**(int id_prodotto): Prodotto

	+**acquisto_prodotto**(Cliente cliente, int id_prodotto, String quantità): Boolean +**get_prodotti_by_apicoltore**(int id_apicoltore): List Prodotti +**decrementa_quantita**(int id_prodotto, int quantita): Boolean
Invariante di classe	/

Nome Metodo	**inserisci_prodotto**(String nome, String descrizione, String località, String peso, String tipologia, String prezzo, String quantità, Apicoltore apicoltore): Boolean
Descrizione	Questo metodo consente ad un apicoltore di inserire un prodotto da mettere in vendita.
Pre-condizione	context: GestioneVenditaService::inserisci_prodotto(nome, descrizione, località, peso, tipologia, prezzo, quantità, apicoltore) pre: apicoltore is not None
Post-condizione	/
Nome Metodo	**cancella_prodotto**(int prodotto_id): Boolean
Descrizione	Questo metodo consente ad un apicoltore di cancellare un prodotto.
Pre-condizione	context: GestioneVenditaService::cancella_prodotto(prodotto_id) pre: get_prodotto_by_id(prodotto_id) is not None
Post-condizione	context: GestioneVenditaService::cancella_prodotto(prodotto_id) post: get_prodotto_by_id(prodotto_id).quantita==0
Nome Metodo	**get_tutti_prodotti**(): List Prodotto

Descrizione	Restituisce tutti i prodotti presenti nel db.
Pre-condizione	/
Post-condizione	/
Nome Metodo	**get_prodotto_by_id**(int id_prodotto): Prodotto
Descrizione	Restituisce un prodotto dato l'id
Pre-condizione	/
Post-condizione	/
Nome Metodo	**acquisto_prodotto**(Cliente cliente, int id_prodotto, String quantità): Boolean
Descrizione	Questo metodo consente ad un cliente di acquistare un prodotto.
Pre-condizione	context: GestioneVenditaService::acquista_prodotto(cliente, id_prodotto, quantita) pre: cliente is not None
Post-condizione	/
Nome Metodo	**get_prodotti_by_apicoltore**(int id_apicoltore): List Prodotti

Descrizione	Restituisce tutti i prodotti di un apicoltore.
Pre-condizione	context: GestioneVenditaService::get_prodotti_by_apicoltore(id_apicoltore) pre: get_apicoltore_by_id(id_apicoltore) is not None
Post-condizione	/
Nome Metodo	**decrementa_quantita**(int id_prodotto, int quantita): Boolean
Descrizione	Questo metodo permette ad un apicoltore di decrementare la quantità di un prodotto dopo che è stato acquistato.
Pre-condizione	context: GestioneVenditaService::decrementa_quantita(id_prodotto, quantita) pre: get_prodotto_by_id(prodotto_id) is not None and get_prodotto_by_id(prodotto_id).quantita >= quantita
Post-condizione	context: GestioneVenditaService::decrementa_quantita(id_prodotto, quantita) post: @pre.get_prodotto_by_id(int(id_prodotto)).quantita==get_prodotto_by_id(int(id_prodotto)).quantita - quantita

PACKAGE GESTIONE ASSISTENZA UTENTE

Nome Classe	**GestioneAssistenzaUtenteService**
Descrizione	Questa classe implementa il sistema di gestione dei ticket assistenza e di tutto ciò che concerne il supporto, da parte degli apicoltori, ai clienti.
Metodi	+**richiedi_assistenza(**String Nome, String descrizione, String id_apicoltore, Cliente Cliente): Boolean +**get_ticket_assistenza_cliente**(int id_cliente): List TicketAssistenza +**get_ticket_assistenza_apicoltore**(int id_apicoltore): List TicketAssistenza

	+**inserisci_area_assistenza**(String descrizione, Apicoltore Apicoltore): Boolean +**get_assistenti()**: List Apicoltore +**get_numero_ticket_assistenza_apicoltore**(int id_apicoltore): int +**get_numero_ticket_assistenza_cliente**(int id_cliente): int +**get_ticket_by_id**(int id_ticket): TicketAssistenza +**controlla_apicoltore**(int id_apicoltore): Boolean
Invariante di classe	/

Nome Metodo	**richiedi_assistenza(**String Nome, String descrizione, String id_apicoltore, Cliente Cliente): Boolean
Descrizione	Questo metodo consente la creazione di un ticket assistenza da parte di un cliente.
Pre-condizione	context: GestioneAssistenzaUtenteService::richiedi_assistenza(nome, descrizione, id_apicoltore, cliente) pre: cliente is not None
Post-condizione	/
Nome Metodo	**get_ticket_assistenza_cliente**(int id_cliente): List TicketAssistenza
Descrizione	Restituisce tutti i ticket di assistenza di un cliente.
Pre-condizione	context: GestioneAssistenzaUtenteService::get_ticket_assistenza_cliente(id_cliente) pre: get_cliente_by_id(id_cliente) is not None
Post-condizione	/

Nome Metodo	**get_ticket_assistenza_apicoltore**(int id_apicoltore): List TicketAssistenza
Descrizione	Restituisce tutti i ticket di assistenza di un apicoltore.
Pre-condizione	context: GestioneAssistenzaUtenteService::get_ticket_assistenza_apicoltore(id_apicoltore) pre: get_apicoltore_by_id(id_apicoltore) is not None
Post-condizione	/
Nome Metodo	**inserisci_area_assistenza**(String descrizione, Apicoltore Apicoltore): Boolean
Descrizione	Gestisce la creazione dell'area di assistenza da parte di un apicoltore
Pre-condizione	context: GestioneAssistenzaUtenteService::inserisci_area_assistenza(descrizione, apicoltore) pre: apicoltore is not None
Post-condizione	context: GestioneAssistenzaUtenteService::inserisci_area_assistenza(descrizione, apicoltore) post: apicoltore.assistenza==True
Nome Metodo	**get_assistenti()**: List Apicoltore
Descrizione	Questo metodo restituisce la lista degli assistenti.
Pre-condizione	/
Post-condizione	/

Nome Metodo	**get_numero_ticket_assistenza_apicoltore**(int id_apicoltore): int
Descrizione	Restituisce il numero di ticket di assistenza di un apicoltore.
Pre-condizione	context: GestioneAssistenzaUtenteService::get_numero_ticket_assistenza_apicoltore(id_apicoltore) pre: get_apicoltore_by_id(id_apicoltore) is not None
Post-condizione	/
Nome Metodo	**get_numero_ticket_assistenza_cliente**(int id_cliente): int
Descrizione	Questo metodo permette la creazione dell'area assistenza da parte dell'apicoltore.
Pre-condizione	context: GestioneAssistenzaUtenteService::get_numero_ticket_assistenza_cliente(id_cliente) pre: get_cliente_by_id(id_acliente) is not None
Post-condizione	/
Nome Metodo	**get_ticket_by_id**(int id_ticket): TicketAssistenza
Descrizione	Restituisce un ticket assistenza dato l'id.
Pre-condizione	/
Post-condizione	/

Nome metodo	**controlla_apicoltore**(int id_apicoltore): Boolean
Descrizione	Questo metodo consente di controllare se l'apicoltore esiste ed è disponibile a fornire assistenza
Pre-condizione	/
Post-condizione	/

PACKAGE GESTIONE UTENTE

Nome Classe	GestioneUtenteService
Descrizione	Questa classe implementa i metodi per la gestione degli utenti permettendo la registrazione, il login e la modifica dei dati personali
Metodi	+**registra_utente**(String nome, String cognome, String indirizzo, String citta, String cap, String telefono, String email, String password, String conferma_password, String is_apicoltore): Boolean +**modifica_profilo_personale**(String nome, String cognome, String email, String telefono, String citta, String cap, String indirizzo, String password, String conferma_password): Boolean +**get_apicoltore_by_email**(String email): Apicoltore +**get_cliente_by_email**(String email): Cliente +**get_apicoltore_by_id**(int id_apicoltore): Apicoltore +**get_cliente_by_id**(int id_cliente): Cliente +**controlla_email_esistente**(String email): Boolean +**controlla_campi**(String nome, String cognome, String indirizzo, String citta, String cap, String telefono, String email): Boolean

	+**controlla_password**(String password, String conferma_password): Boolean +**controllo_caratteri_speciali**(String password): Boolean +**controlla_numeri**(String password): Boolean
Invariante di classe	/

Nome Metodo	**registra_utente**(String nome, String cognome, String indirizzo, String citta, String cap, String telefono, String email, String password, String conferma_password, String is_apicoltore): Boolean
Descrizione	Questo metodo consente la registrazione di un utente alla piattaforma.
Pre-condizione	/
Post-condizione	context: GestioneUtenteService::registra_utente(nome, cognome, indirizzo, citta, cap, telefono, email, password, conferma_password, is_apicoltore) post: session['isApicoltore']==is_apicoltore and (get_apicoltore_by_email(email) is not None or get_cliente_by_email(email) is not None)
Nome Metodo	**modifica_profilo_personale**(String nome, String cognome, String email, String telefono, String citta, String cap, String indirizzo, String password, String conferma_password): Boolean
Descrizione	Questo metodo consente la registrazione di un cliente al sistema.
Pre-condizione	/
Post-condizione	/

Nome Metodo	**get_apicoltore_by_email**(String email): Apicoltore
Descrizione	Restituisce un apicoltore data l'email.
Pre-condizione	/
Post-condizione	/
Nome Metodo	**get_cliente_by_email**(String email): Cliente
Descrizione	Restituisce un cliente data l'email.
Pre-condizione	/
Post-condizione	/
Nome Metodo	**get_apicoltore_by_id**(int id_apicoltore): Apicoltore
Descrizione	Restituisce un apicoltore dato l'id.
Pre-condizione	/
Post-condizione	/

Nome Metodo	**get_cliente_by_id**(int id_cliente): Cliente
Descrizione	Restituisce un cliente dato l'id.
Pre-condizione	/
Post-condizione	/
Nome Metodo	**controlla_email_esistente**(String email): Boolean
Descrizione	Controlla se l'email è già presente nel database.
Pre-condizione	/
Post-condizione	/
Nome Metodo	**controlla_campi**(String nome, String cognome, String indirizzo, String citta, String cap, String telefono, String email): Boolean
Descrizione	Effettua il controllo dei campi del form della registrazione.
Pre-condizione	/
Post-condizione	/

Nome Metodo	**controlla_password**(String password, String conferma_password): Boolean
Descrizione	Effettua il controllo della password nel form di registrazione.
Pre-condizione	/
Post-condizione	/
Nome Metodo	**controllo_caratteri_speciali**(String password): Boolean
Descrizione	Effettua il controllo dei caratteri speciali della password nel form di registrazione.
Pre-condizione	/
Post-condizione	/
Nome Metodo	**controlla_numeri**(String password): Boolean
Descrizione	Effettua il controllo dei numeri della password nel form di registrazione.
Pre-condizione	/
Post-condizione	/

PACKAGE GESTIONE ADOZIONI

Nome Classe	GestioneAdozioniService
Descrizione	Questa classe implementa i metodi per la gestione delle adozioni
Metodi	+**inserisci_alveare**(String nome, String produzione, String numero_api, String tipo_miele, String prezzo, String tipo_fiore, Apicoltore apicoltore): Boolean +**get_alveare_by_id** (int alveare_id): Alveare +**get_alveari** (): List Alveari +**drecementa_percentuale**(Alveare alveare, String percentuale): void +**adozione_alveare**(Alveare alveare, Cliente cliente, String tempo_adozione, String percentuale): Boolean +**aggiorna_stato** (Alveare alveare, String covata_compatta, String popolazione, String polline, String stato_cellette, String stato_larve): Boolean +**get_alveari_from_apicoltore**(int apicoltore_id): List Alveare +**get_ticket_adozione**(int cliente_id): List Alveare +**get_ticket_from_alveare**(int id_alveare): List TicketAdozioneser
Invariante di classe	/

Nome Metodo	**inserisci_alveare**(String nome, String produzione, String numero_api, String tipo_miele, String prezzo, String tipo_fiore, Apicoltore apicoltore): Boolean

Descrizione	Questo metodo consente ad un apicoltore di mettere a disposizione alveari da adottare.
Pre-condizione	context: GestioneAdozioniService::inserisci_alveare(nome, produzione, numero_api, tipo_miele, prezzo, tipo_fiore, apicoltore) pre: apicoltore is not None
Post-condizione	context: GestioneAdozioniService::inserisci_alveare(nome, produzione, numero_api, tipo_miele, prezzo, tipo_fiore, apicoltore) post: get_alveare_by_id(alveare.id) is not None
Nome Metodo	**get_alveare_by_id** (int alveare_id): Alveare
Descrizione	Restituisce l'alveare dato l'id.
Pre-condizione	/
Post-condizione	/
Nome Metodo	**get_alveari** (): List Alveari
Descrizione	Questo metodo restituisce la lista di tutti gli alveari.
Pre-condizione	/
Post-condizione	/
Nome Metodo	**drecementa_percentuale**(Alveare alveare, String percentuale): void

Descrizione	Decrementa la percentuale disponibile dell'alveare.
Pre-condizione	context: GestioneAdozioniService::decrementa_percentuale(alveare, percentuale) pre: alveare is not None and alveare.percentuale_disponibile >= percentuale
Post-condizione	context: GestioneAdozioniService::decrementa_percentuale(alveare, percentuale) post: alveare is not None and @pre.alveare.percentuale_disponibile == .alveare.percentuale_disponibile - percentuale
Nome Metodo	**adozione_alveare**(Alveare alveare, Cliente cliente, String tempo_adozione, String percentuale): Boolean
Descrizione	Questo metodo consente l'adozione di un alveare da parte di un cliente.
Pre-condizione	context: GestioneAdozioniService::adozione_alveare(alveare, cliente, tempo_adozione, percentuale) pre: alveare is not None and cliente is not None
Post-condizione	context: GestioneAdozioniService::adozione_alveare(alveare, cliente, tempo_adozione, percentuale) post: get_ticket_adozione(cliente.id).get(alveare) is not None
Nome Metodo	**aggiorna_stato** (Alveare alveare, String covata_compatta, String popolazione, String polline, String stato_cellette, String stato_larve): Boolean
Descrizione	Questo metodo consente l'aggiornamento dello stato di un alveare da parte di un apicoltore.
Pre-condizione	context: GestioneAdozioniService::aggiorna_stato(alveare, covata_compatta, tempo_popolazione, polline, stato_cellette, stato_larve) pre: alveare is not None
Post-condizione	/

Nome Metodo	**get_alveari_from_apicoltore**(int apicoltore_id): List Alveare
Descrizione	Restituisce gli alveari appartenenti ad un apicoltore.
Pre-condizione	context: GestioneAdozioniService::get_alveari_from_apicoltore(apicoltore_id) pre: get_apicoltore_by_id(apicoltore_id) is not None
Post-condizione	/
Nome Metodo	**get_ticket_adozione**(int cliente_id): List Alveare
Descrizione	Restituisce gli alveari adottati da un cliente.
Pre-condizione	context: GestioneAdozioniService::get_ticket_adozione(cliente_id) pre: get_cliente_by_id(cliente_id) is not None
Post-condizione	/
Nome Metodo	**get_ticket_from_alveare**(int id_alveare): List TicketAdozione
Descrizione	Restituisce i ticket di adozione di un alveare.
Pre-condizione	context: GestioneAdozioniService::get_ticket_adozione(cliente_id) pre: get_alvearee_by_id(id_alveare) is not None
Post-condizione	/

4. Elementi di Riuso

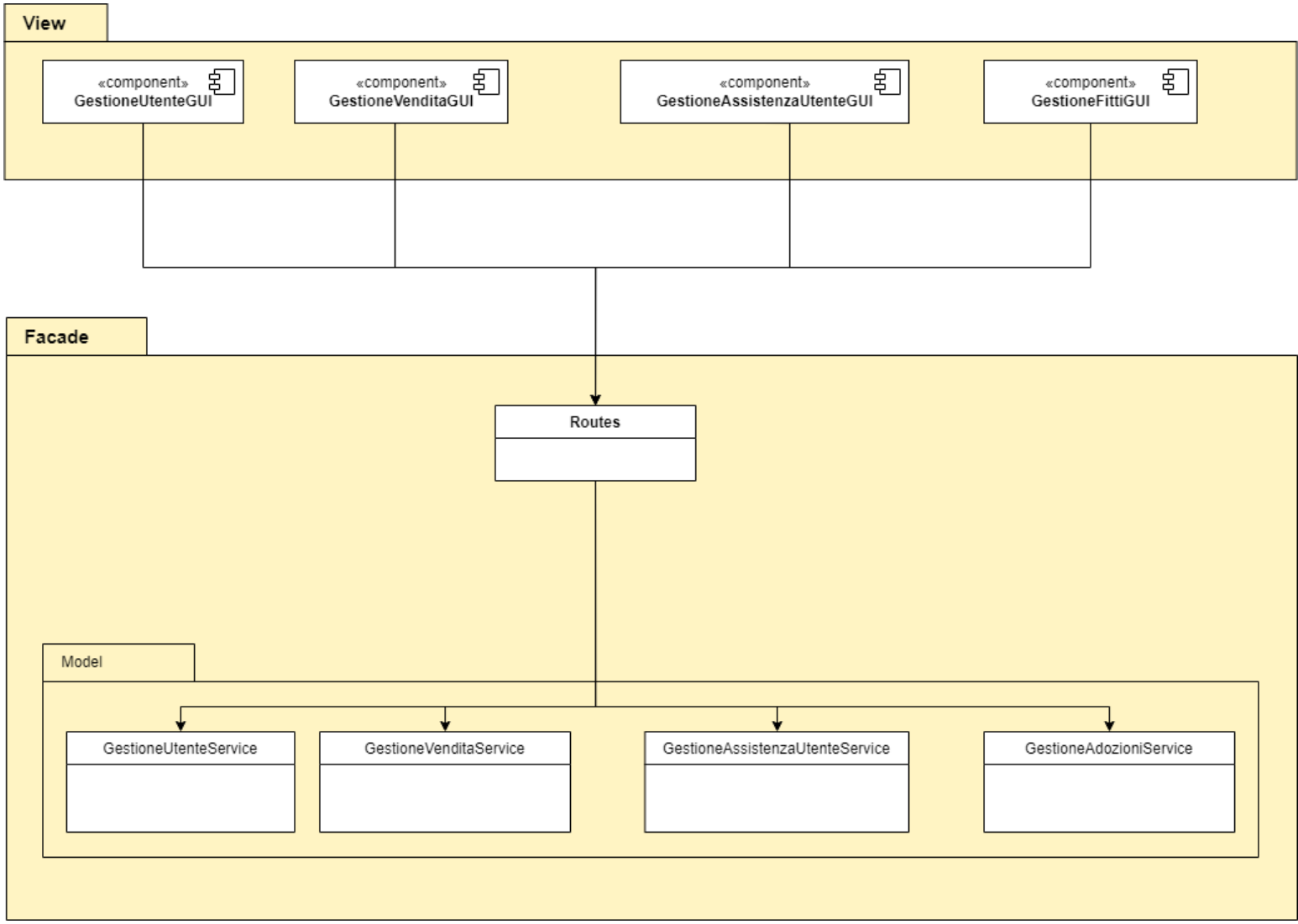
4.1. Design Pattern

FACADE

Il Facade è un design pattern che permette, implementando una interfaccia semplificata, di accedere a sottosistemi più complessi. In questo modo si può nascondere al sistema la complessità delle librerie, dei framework o dei set di classi che si stanno usando. Si garantisce così un alto disaccoppiamento e si rende la piattaforma più manutenibile e più aggiornabile, poiché basterà cambiare l'implementazione dei metodi dell'interfaccia per implementare le modifiche.

BeeHave essendo un sistema molto complesso, sfrutta il design pattern Facade per implementare parte della sua logica di business e rendere più facile l'interfacciarsi con essa.

Di seguito un esempio di Facade nel nostro sistema:


View
«component»
GestioneUtenteGUI
«component»
GestioneVenditaGUI
«component»
GestioneAssistenzaUtenteGUI
«component»
GestioneFittiGUI
Facade
Routes
Model
GestioneUtenteService
GestioneVenditaService
GestioneAssistenzaUtenteService
GestioneAdozioniService

5. Glossario

Terminologia	Definizione
Design Pattern	Template di soluzioni da applicare per la risoluzione di un problema che si può presentare in diverse situazioni durante le fasi di progettazione e sviluppo del software.
Framework	Architettura logica di supporto sulla quale un software può essere progettato e realizzato fornendo un'infrastruttura generale lasciando al programmatore il contenuto vero e proprio dell'applicazione.
Flask	Framework basato su Python utilizzato per sviluppare applicazione web-based.
Facade	Design Pattern utilizzato per fornire, al client, un'interfaccia semplificata di un sottosistema complesso.
Packages	I Packages servono ad aiutare lo sviluppatore ad organizzare modelli UML per aumentarne la leggibilità.

HTML, CSS, JS, Python	Tecnologie utilizzate per lo sviluppo del sistema.