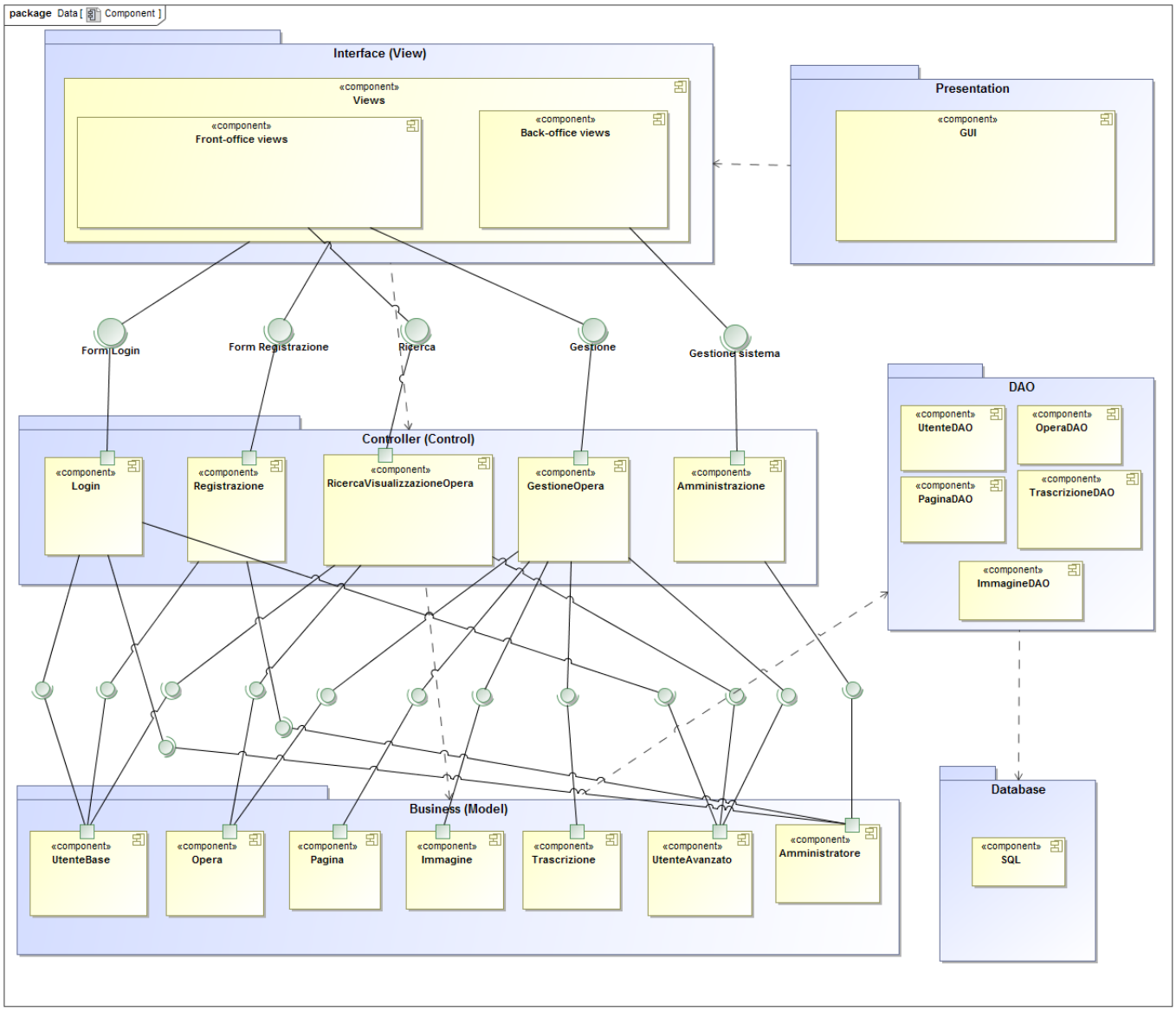
Di Giosaffatte Valentino 232411
Di Prinzio Riccardo Armando 229032
Furia Flavio 229034

System Design

***Component diagram*: modello dell'architettura software (2.1):**



Descrizione dell'architettura (2.2):

L'architettura di questo sistema software presenta diversi package, primo fra tutti il package "Presentation" che contiene la componente "GUI", ovvero l'interfaccia grafica generale del programma. "GUI" è direttamente dipendente dal package "Interface", l'insieme delle pagine del sistema tramite le quali l'utente potrà interfacciarsi con le funzioni principali del programma che si trovano nel package "Controller".

Il package "Interface" è composto da una componente "Views" divisa in due sottocomponenti, una "Front-office views" (insieme delle pagine del Front-End), utilizzata dagli utenti e contenente le funzionalità fondamentali dell'applicazione, ed una "Back-office views" (insieme delle pagine della Back-End), utilizzata dall'amministratore per la gestione del sistema e dei dati nel database.

La componente "Views" richiede le seguenti interfacce offerte dal package "Controller":

- Login: L'utente già registrato fa l'accesso e acquisisce i propri privilegi.
- Registrazione: L'utente base non precedentemente iscritto, si registra nel sistema e acquisisce i privilegi di utente avanzato.

Il "Front-office views" richiede le diverse interfacce offerte dal package "Controller" e le componenti di quest'ultimo restituiscono i metodi che implementano tali interfacce:

- RicercaVisualizzazioneOpera: L'utente può cercare un'opera presente nel sistema e in base ai privilegi che possiede, avrà in output le seguenti tipologie di risultati:
    - Utente base: Verranno restituiti solo i titoli delle opere.
    - Utente avanzato: Verrà restituito un elenco di titoli di opere complete di immagini e trascrizioni le quali potranno essere sfogliate dall'utente.
- GestioneOpera: Gli utenti avanzati addetti all'acquisizione, trascrizione e revisione possono gestire (caricare, validare, eliminare, …) le opere del sistema.

Il "Back-office views" richiede l'interfaccia "Amministrazione" offerta dal package "Controller", che fornirà le funzioni e i metodi per gestire il database e il sistema.

Le componenti all'interno del package "Controller" hanno bisogno di maneggiare oggetti e dati persistenti per fornire i servizi per cui sono stati pensati. Per questo motivo, necessitano di interfacce contenute nel package "Business", contenente gli oggetti principali.

Nel component diagram possiamo infatti notare che le componenti "Login" e "Registrazione" si interfacciano con le componenti "Business" "UtenteBase", "UtenteAvanzato" e "Amministratore". "RicercaVisualizzazioneOpera" e "GestioneOpera", invece, utilizzano rispettivamente "Opera" e "Pagina". Inoltre "RicercaVisualizzazioneOpera" utilizza sia "UtenteBase" sia "UtenteAvanzato". "GestioneOpera" utilizza le componenti "Immagine" e "Trascrizione". Il gestore del sistema continua ad interfacciarsi con una sola componente, "Amministrazione".

Sarà possibile garantire la persistenza di tutti i dati e gli oggetti del sistema mediante un database comune (presente nel package "Database"), accessibile unicamente tramite funzionalità presenti nel package "DAO", che avrà appunto il compito di salvare nel database gli oggetti del "Business" senza che essi debbano accedere direttamente al DBMS.

Descrizione delle scelte e strategie adottate (2.3):

L'architettura software descritta dal component diagram precedente è stata ottenuta in seguito ad alcune decisioni di design analizzate e discusse dal team per risolvere alcuni problemi applicativi. Partendo dal package "Interface", si è deciso di fornire un componente "Views" che raggruppa due interfacce grafiche distinte all'interno della "GUI": La Front-office per tutti gli utenti che hanno come unico scopo quello di visualizzare e gestire le opere, la Back-office, totalmente differente, utilizzata unicamente dall'amministratore per gestire il database e i permessi dati agli utenti.

I permessi utente hanno lo scopo di fornire o negare funzionalità e visualizzazione di intere pagine dell'interfaccia grafica, e sono ottenuti durante l'autenticazione di un utente a seguito della login. Ogni riga del database contenente le informazioni di ogni singolo utente, conterrà anche una colonna, denominata "Permessi", che ammetterà solo valori di tipo intero compresi tra 0 e 6, corrispondenti al grado di potere dell'utente stesso. Un utente non registrato potrà accedere al sistema come "ospite"; così facendo, verrà salvata una riga nella tabella "Utente" con valore "Permessi" uguale a 0. Questo utente potrà unicamente visualizzare i titoli delle opere. Quando egli chiuderà il programma, la riga verrà eliminata automaticamente. Un utente che ha precedentemente effettuato la registrazione, avrà una riga persistente nel database con valore della colonna "Permessi" uguale a 1. Ciò significa che, oltre a visualizzare l'elenco dei titoli delle opere, egli potrà accedervi in toto. Tutti i permessi speciali verranno forniti unicamente dall'amministratore agli utenti addetti alla acquisizione, trascrizione e revisione. Gli altri attributi e le proprietà degli utenti verranno definite in seguito.

Per modellare l'oggetto "Opera" è stato deciso di creare tre oggetti:

- "Immagine": corrisponde ad una singola acquisizione effettuata da un acquisitore;
- "Trascrizione": è il testo digitale di un'immagine; non tutte le immagini avranno una trascrizione associata;
- "Pagina": contiene, oltre a dati di varia natura, una coppia "Immagine", "Trascrizione".

Un'istanza dell'oggetto "Opera" sarà dunque costituita da un array di oggetti di tipo "Pagina". Nel database, ognuno di questi tipi di oggetto avrà una tabella contenente informazioni aggiuntive.

Di Giosaffatte Valentino 232411
Di Prinzio Riccardo Armando 229032
Furia Flavio 229034

L'architettura software descritta segue il design pattern DAO per interfacciarsi con i dati persistenti nel database e per la suddivisione delle logiche viene utilizzato il design pattern MVC. In particolare le logiche del sistema sono state suddivise nel modo seguente:

- **View:** Il package Interface.
- **Control:** Il package Controller.
- **Model:** Il package Business.

La scelta di dividere le logiche in questo modo ha semplificato notevolmente la complessità nelle classi del sistema.

Eventuali decisioni di design successive verranno analizzate e comunicate nel prossimo deliverable.