UNIVERSITÀ DI TRENTO

Dipartimento di Ingegneria e Scienza dell'Informazione

Progetto:

Il Ricettacolo

Titolo del documento:

Documento di architettura

Informazioni documento:

Nome documento:	D3–T06.pdf		
Membri:	Alberto Cimmino	Salvatore Gilles Cassarà	Sara Tait

Indice

1. Diagramma delle classi

In questo capitolo vengono presentate le classi previste nell'ambito del progetto "Il Ricettacolo". Ogni attore e sistema diventano uno o più classi, caratterizzate da un nome, una lista di attributi e una serie di comportamenti (metodi). Una classe può essere associata ad altre classi tramite un'associazione, fornendo informazioni su come queste classi si relazionano tra loro. Vengono riportate di seguito le classi individuate a partire dal diagramma di contesto e il diagramma dei componenti.

1. Utenti

La componente "Interazione" è stata tradotta in tre classi: "UtenteNonAutenticato", "UtenteAutenticato" ed "UtentePremium".

UtenteNonAutenticato
Indica un utente che non si è autenticato.

Ha i seguenti attributi:

- UID : string
  - Contiene la combinazione di lettere e numeri che identifica univocamente l'utente.
- nomeUtente : string
  - Contiene il nome utente corrispondente allo UID.
- password : string
  - Contiene la password corrispondente allo UID.
- email : string
  - Contiene l'email corrispondente allo UID.
- auth : autenticazione
  - Riferimento alla classe "Autenticazione". Permette di effettuare l'autenticazione e controllarne lo stato.

Ha i seguenti metodi:

- registrazione(): *in seguito attributi devono essere inizializzati*
  - Mostra un form da compilare nel quale inserire: nomeUtente, email, password e conferma password. I dati vengono passati alla classe "Autenticazione" che, superato un controllo, ritorna il token corrispondente all'istanza di autenticazione e viene inizializzata un'istanza della sottoclasse "UtenteAutenticato" con quei dati. I dati dell'utente vengono inoltre aggiunti al database e gli viene assegnato un UID;
  - Se non viene superato il controllo, viene mostrato un messaggio di errore.
- login(): *same*
  - Mostra un form da compilare con la combinazione nomeUtente e password o email e password associati al proprio account. Questi vengono passati alla classe "Autenticazione" che verifica i dati e, se corretti, ritorna il token corrisponde all'istanza di autenticazione e viene inizializzata un'istanza della classe "UtenteAutenticato" con quei dati. Nel caso di mancata corrispondenza, viene mostrato un messaggio di errore.
- getUID():
  - Ritorna l'UID dell'istanza della classe "UtenteNonAutenticato" che lo ha chiamato.
  - Se non sono stati effettuati login o registrazione, ritorna NULL;

- getNomeUtente():
    - Ritorna il nomeUtente dell'istanza della classe "UtenteNonAutenticato" che lo ha chiamato.
    - Se non sono stati effettuati login o registrazione, ritorna NULL;
- getEmail():
    - Ritorna l'email dell'istanza della classe "UtenteNonAutenticato" che lo ha chiamato.
    - Se non sono stati effettuati login o registrazione, ritorna NULL;
- getPassword():
    - Ritorna la password dell'istanza della classe "UtenteNonAutenticato" che lo ha chiamato.
    - Se non sono stati effettuati login o registrazione, ritorna NULL;
- recuperoPassword(): *perhaps??*
    - Mostra un form in cui inserire l'email del proprio account, alla quale verrà inviata un'email, dalla classe "GestioneEmail", che permette di recuperare la password dell'account legato all'email inserita.

UtenteAutenticato
Sottoclasse di "UtenteNonAutenticato" che indica un utente che ha effettuato l'autenticazione.

Ha i seguenti attributi:
- punteggio : long int
    - contiene il punteggio dell'utente corrispondente all'attributo UID.
- ranking : int
    - Contiene la posizione in classifica in base al punteggio dell'utente corrispondente all'attributo UID.
- pfp : string
    - Contiene il percorso dell'immagine impostata come foto profilo dell'utente corrispondente all'attributo UID.

Ha i seguenti metodi:
- getPunti()
    - ritorna il punteggio dell'istanza della classe "UtenteNonAutenticato" che lo ha chiamato;
- getRank()
    - ritorna il ranking dell'istanza della classe "UtenteNonAutenticato" che lo ha chiamato;
- getPfp()
    - ritorna la stringa pfp dell'istanza della classe "UtenteNonAutenticato" che lo ha chiamato;
- setPfp(pfp : string): *il fatto che avviene la modifica*
    - permette di impostare un'immagine profilo passando come parametro il percorso dell'immagine e modifica l'attributo "pfp".
- cambiaNomeUtente():
    - mostra un form dove inserire un nuovo nomeUtente. Se questo non è già presente nel database, viene impostato come nomeUtente corrispondente all'utente dell'attributo UID, altrimenti viene mostrato un messaggio di errore;

- cambiaPassword():
    - Mostra un form dove inserire la password corrente, una nuova password e la conferma della nuova password. Se queste rispettano il criterio di sicurezza delle password, viene impostata la nuova password dell'utente corrispondente all'attributo UID, altrimenti viene mostrato un messaggio di errore;
- cambiaEmail():
    - Mostra un form dove inserire una nuova email e la password. Se la password non corrisponde ne impedisce il proseguimento. Altrimento controlla se l'email non è già presente nel database, nel caso, viene impostata come nuova email corrispondente allo UID dell'utente che ha chiesto la modifica e aggiorna l'attributo corrispondente. Nel caso in cui uno dei due passaggi dovesse fallire viene mostrato un messaggio di errore.
- iscrizionePremium(): *non deve già essere premium + modifica stato premium*
    - Avvia la procedura di iscrizione all'abbonamento Premium. Se la procedura viene completata con successo, viene inizializzata un'istanza della sottoclasse "UtentePremium" impostando le date corrispondenti al rinnovo ed alla scadenza dell'abbonamento e viene inviata un'email di avviso tramite "GestioneEmail". Nel caso di fallimento dell'operazione, viene mandata un'email di avviso tramite "GestioneEmail".
- isPremium():
    - metodo che ritorna true se l'utente appartiene alla classe "UtentePremium", false altrimenti.
- logout(): *modifica*
    - chiama la classe "Autenticazione" affinche cancelli il token corrispondente all'istanza di autenticazione, cancellando così anche l'istanza dell'"Utente Autenticato", che torna ad essere un "UtenteNonAutenticato".

UtentePremium
sottoclasse di "UtenteAutenticato". Indica un utente che ha sottoscritto l'abbonamento Premium.

Ha i seguenti attributi:
- indirizzo : string
    - Contiene l'indirizzo di consegna.
- dataRinnovo : date
    - Contiene la data in cui verrà effettuato il rinnovo mensile dell'abbonamento.
- dataScadenza : date
    - Contiene la data in cui terminerà il piano di sottoscrizione all'abbonamento Premium scelto dall'utente.

Ha i seguenti metodi:
- getIndirizzo(): ritorna l'indirizzo di consegna corrispondente allo UID;
- setIndirizzo(indirizzo : string): imposta l'attributo indirizzo con il valore passato per parametro;
- rinnovoPremium(): metodo chiamato automaticamente quando la data attuale corrisponde con dataRinnovo. Effettua il rinnovo dell'abbonamento.
  Se il rinnovo fallisce, viene richiamato il metodo altre due volte a distanza di un'ora una dall'altra. Se tutti i tentativi falliscono, viene chiamato il metodo rimuoviPremium();

- rimuoviPremium(): metodo chiamato automaticamente quando la data attuale corrisponde con dataScadenza o nel caso in cui rinnovoPremium() fallisca per tre volte. Viene cancellata l'istanza dello "UtentePremium", che torna ad essere un UtenteAutenticato.

2. Autenticazione

Traduzione della componente "Autenticazione" nella classe "Autenticazione". Si occupa dell'interazione con Auth0 che controlla la possibilità di effettuare la registrazione ed il login e crea un token corrispondente all'istanza di autenticazione. Inoltre, si occupa di effettuare il logout cancellando il token.

Ha i seguenti attributi:

- token: string
  - Combinazione di lettere e numeri. Contiene il token corrispondente all'istanza di autenticazione di un dato utente.

Ha i seguenti metodi:

- creaToken(nomeUtente : string, password : string)
  - interagisce con Auth0 inviandogli i parametri nomeUtente e password. Se viene trovata una corrispondenza con un utente iscritto al sito, viene ritornato un token che viene poi impostato come l'attributo token.
- creaToken(email : string, password : string)
  - metodo che interagisce con Auth0 inviandogli i parametri email e password. Se viene trovata una corrispondenza con un utente iscritto al sito, viene ritornato un token che viene poi impostato come l'attributo token;
- creaToken(nomeUtente : string, email : string, password : string)
  - metodo che interagisce con Auth0 inviandogli i parametri nomeUtente, email e password. Se non vengono trovate corrispondenze con il nomeUtente e la mail di un utente iscritto al sito, viene ritornato un token che viene poi impostato come l'attributo token;
- ottieniToken()
  - ritorna il token impostato come attributo.
  - Ritorna NULL se l'utente non ha effettuato l'autenticazione.
- rimuoviToken()
  - imposta l'attributo token a NULL, effettuando così il logout.

3. Ricetta

La componente “Gestione ricetta” del diagramma delle componenti è stata tradotta in due classi, “Ricetta” e “Gestione ricetta”.
“Ricetta” rappresenta le ricette caricate dagli utenti e che è possibile visualizzare nel sito.

Ha i seguenti attributi:

- id : string
  - Rappresenta in modo univoco la ricetta.
- nome : string
  - Rappresenta il titolo della ricetta che viene scelto dall’utente al momento della creazione.
- autoreUID : string
  - Identifica l’utente creatore della ricetta.
- procedimento : string
  - Il testo scritto dall’utente al momento della creazione della ricetta per descriverne la preparazione.
- immagini : list<string>
  - Lista contenente il percorso delle immagini inserite al momento della creazione della ricetta.
  - Ritorna NULL nel caso in cui non ci siano immagini.
- giudizi: list<string>
  - Lista contenente le recensioneID delle recensioni lasciate dagli utenti sulla ricetta.
  - Ritorna NULL nel caso in cui non ci siano recensioni.

Ha i seguenti metodi:

- initRicetta() *piu che init, i vari set dovrebbero dare errore se il campo che leggono è vuoto (eccetto immagini)*
  - Mostra un form e lo utilizza per inizializzare gli attributi di una nuova istanza della classe “Ricetta”. Per farlo, chiama i metodi setNome(), setProcediemento() e setImmagini(). Questa istanza è poi caricata nel database. Viene chiamato il metodo setPunti() dalla classe “Classifica” per aggiornare il punteggio dell’utente.
- setNome(nome: string)
  - Imposta l’attributo nome in “Ricetta” con il parametro ad esso passato.
- setProcediemento(procediemento: string)
  - Imposta l’attributo procedimento in “Ricetta” con il parametro ad esso passato.
- setImmagini(immagini : list<string>)
  - Imposta l’attributo immagini in “Ricetta” con il parametro ad esso passato.
- getNome()
  - Restituisce l’attributo nome dell’istanza di “Ricetta” da cui il metodo è chiamato.
- getAutore()
  - Restituisce l’attributo autoreUID dell’istanza di “Ricetta” da cui il metodo è chiamato.
- getProcediemento()
  - Restituisce l’attributo procedimento dell’istanza di “Ricetta” da cui il metodo è chiamato.
- getImmagini()
  - Restituisce l’attributo immagini dell’istanza di “Ricetta” da cui il metodo è chiamato.
- getID()
  - Restituisce l’attributo ID dell’istanza di “Ricetta” da cui il metodo è chiamato.

4. Recensione

Ha i seguenti attributi:

- recensioneID : string
  - Identifica in modo univoco una recensione. Viene utilizzata dalla classe “Ricetta” per tenere traccia delle recensioni associate alle varie ricette.
- autoreUID : string
  - UID che identifica lo ”UtenteAutenticato” autore della recensione.
- numeroStelle : int
  - Intero compreso fra 1 e 5 corrispondente al numero di stelle che l’utente ha scelto di attribuire alla ricetta.
- commento : string
  - Testo scritto dall’utente come giudizio alla ricetta.
  - Ritorna NULL nel caso in cui non ci sia il commento.

Ha i seguenti metodi:

- initRecensione() *stesso discorso di init ricetta*
  - Crea una nuova istanza di “Recensione”. Si serve dei metodi, setStelle(stelle : int) e setCommento(commento : string) con le quali inizializza i rispettivi attributi. Una volta creata l’istanza, i suoi dati sono caricati nel database. Inoltre viene chiamato il metodo setPunti() della classe “Classifica” per aggiornare il punteggio dell’utente.
- setStelle(stelle : int) *check che il nuemero sia da 1 a 5*
  - *Inizializza* l’attributo stelle con il parametro ‘Stelle’ ad esso passato
- setCommento(commento : string)
  - Imposta l’attributo commento con il parametro ad esso passato.
- getAutore()
  - Restituisce l’attributo AutoreUID dell’istanza di “Recensione” da cui il metodo è chiamato.
- getStelle()
  - Restituisce l’attributo stelle dell’istanza di “Recensione” da cui il metodo è chiamato.
- getCommento()
  - Restituisce l’attributo commento dell’istanza di “Recensione” da cui il metodo è chiamato.

5. Classifica

La componente “Classifica” è stata tradotta nella classe “Classifica”, che si occupa della gestione della classifica degli utenti.

Ha i seguenti attributi:

- ranking : Hashmap <string, long int>
  - Hashmap contenente gli UID e i punteggi corrispondenti ai vari utenti.

Ha i seguenti metodi:

- setPunti(UID : string , punti : long int)
  - Controlla se nell’attributo ranking sia già presente la chiave UID: in caso affermativo, somma il valore di punti ai punti già associati all’utente, altrimenti aggiunge alla Hashmap la coppia <UID, punti>. Inoltre ordina l’Hashmap in maniera decrescente secondo i valori contenuti in punti ed aggiorna i dati nel database.

- getRank(UID : string)
    - Restituisce la posizione in ranking dell'utente corrispondente al parametro UID.
- getClassifica()
    - Restituisce l'attributo ranking.
    - Se vuoto, ritorna NULL.

6. GestioneEmail

Traduzione della componente "Gestione Email", si occupa di spedire email agli utenti nel caso di recupero password, per avvisare dello stato dei pagamenti e del termine dell'abbonamento Premium dell'utente.

Ha i seguenti metodi:

- sendRecuperoPassword(indirizzo : string)
    - Invia all'indirizzo passato come parametro un'email contenente un link che porta alla pagina dove è possibile reimpostare la propria password.
- sendPagamentoRiuscito(indirizzo : string)
    - Invia all'indirizzo passato come parametro un'email che conferma all'utente la riuscita del pagamento relativo all'abbonamento Premium.
- sendPagamentoFallito(indirizzo : string)
    - Invia all'indirizzo passato come parametro un'email che avvisa l'utente il fallimento del pagamento relativo all'abbonamento Premium.
- sendFinePremium(indirizzo : string)
    - Invia all'indirizzo passato come parametro un'email che avvisa l'utente della scadenza dell'abbonamento Premium.

7. Traduzione

Traduzione dalla componente "Traduzione", si occupa di tradurre il sito nella lingua di sistema.

Ha i seguenti attributi:

- linguaSistema : string
    - la stringa rappresenta la lingua di sistema.

Ha i seguenti metodi:

- getLinguaSistema()
    - Rileva la lingua di sistema ed imposta l'attributo linguaSistema.
- setLingua()
    - Chiede conferma all'utente se voglia il sito tradotto nella lingua del suo sistema. Nel caso affermativo, invia la richiesta di traduzione a Google Traduttore.

8. Gestione Ricetta

La componente "Gestione ricetta" del diagramma delle componenti è stata tradotta in due classi, "Ricetta" e "GestioneRicetta".

Ha i seguenti attributi:

- **TMP** : ricetta
    - Riferimento alla ricetta su cui si sta lavorando.

- IDRicettaOracolo : string
  - ID della ricetta caricata dal metodo oracolo().

- data : date
  - Giorno in cui è stato caricato l'attributo IDRicettaOracolo.
- recensione : recensione
  - Riferimento alla recensione su cui si sta lavorando.

Ha i seguenti attributi:

- caricaRicetta(ricettaID : string)
  - inizializza un'istanza della classe "Ricetta" con i dati letti dal database relativi alla ricetta identificata dal parametro ricettaID.
- mostraRicetta()
  - Mostra a schermo i seguenti attributi della classe "Ricetta": nome, procedimento, giudizi, le immagini ed il nome utente dell'autore, ricavato tramite l'attributo autoreUID.
- modificaRicetta(UID:string) *controllo del parametro UID*
  - Controlla se il parametro UID corrisponde all'UID della classe "Ricetta". Nel caso in cui corrispondano, permette all'utente la modifica degli attributi della ricetta.
- oracolo() *il fatto che la data deve corrispondere*
  - Controlla se l'attributo data corrisponde alla data odierna. Nel caso in cui siano uguali, chiama la funzione caricaRicetta(IDRicettaOracolo), passandole l'attributo idRicettaOracolo. Senon sono uguali, aggiorna il valore dell'attributo data con la data attuale ed imposta un nuovo ricettaID, scelto tra quelli messi a disposizione per quel periodo, che andrà a salvare in IDRicettaOracolo. Viene poi chiamati i metodi caricaRicetta(IDRicettaOracolo) e mostraRicetta().
- scriviRecensione() *in seguito alla funzione, giudizi deve contenere la recensione*
  - Chiama il metodo initRecensione() presente nella classe "Recensione", creando una nuova istanza di recensione. In seguito salva la recensioneID della recensione appena creata nell'attributo giudizi dell'istanza della ricetta corrispondente.
- modificaRecensione(UID:string, recensioneID:string) *same as above*
  - Controlla se il parametro UID corrisponde all'attributo autoreUID dell'istanza della classe "Recensione" corrispondente al recensioneID passato come parametro. Se corrispondono permette di modificare gli attributi dell'istanza della recensione.

9. ListaSpesa

La componente "Lista della Spesa" nel diagramma delle componenti è stata tradotta nella classe "ListaSpesa".

Ha i seguenti attributi:

- listaSpesa : Hashmap <string, int>
  - Un'Hashmap che prende come chiave una stringa che corrisponde al nome dell'ingrediente che si desidera aggiungere e come valore un intero che ne rappresenta la quantità.

Ha i seguenti metodi:

- getLista()
  - restituisce l'attributo listaSpesa dell'istanza "ListaSpesa" da cui il metodo è stato chiamato.

- ritorna NULL se la lista è vuota.

- addIngrediente(ingrediente:string, quantita:int) *in seguito la lista deve contenere l'ingrediente*
  - Controlla se nell'attributo listaSpesa sia già presente la chiave ingrediente: in caso affermativo, somma il valore di quantità alla quantità già associata all'ingrediente, altrimenti aggiunge alla Hashmap la coppia <ingrediente, quantità>.
- deleteIngrediente (nome : string) *deve essere tolto*
  - Controlla che nell'Hashmap sia presente la chiave corrispondente al valore nome passato come parametro. Se la chiave è presente, la coppia <ingrediente, quantità> corrispondente viene rimossa.
- esportaLista()
  - Esporta in formato pdf i contenuti dell'attributo listaSpesa, scrivendo un solo ingrediente e rispettiva quantità per riga.

10. Preferiti

La componente "Preferiti" del diagramma delle componenti è stata tradotta nella classe "Preferiti".

Ha i seguenti attributi:

- preferiti : list <string>
  - Lista che contiene gli IDRicetta delle ricette salvate come preferite dall'utente.

Ha i seguenti metodi:

- getPreferiti(UID:string)
  - Carica nell'attributo preferiti la lista delle ricette preferite dell'utente, il quale si ricava attraverso il parametro "UID".
- removePreferiti(idRicetta:string) *same as above, deve essere inserito, mentre sotto deve essere tolto*
  - rimuove dall'attributo "preferiti" l'id che corrisponde al parametro idRicetta. (se non esiste?)
- addPreferiti(idRicetta:string)
  - Se non già presente, aggiunge il parametro idRicetta all'attributo "preferiti".

11. Follower

La componente "Follower" del diagramma delle componenti è stata tradotta nella classe "Follower".
Ha i seguenti attributi:

- followers: list <string>
  - Una lista che contiene gli UID della classe "Utente" seguiti dall'utente.

Ha i seguenti metodi:

- getFollowers(UID:string)
  - Carica nell'attributo "followers" la lista degli utenti seguiti dell'utente, il quale si ricava attraverso il parametro "UID".
- removeFollowers(UID:string) *same as above, deve essere inserito, mentre sotto deve essere tolto*

- rimuove dall'attributo "followers" l'id che corrisponde al parametro UID (se non esiste?)

- addFollowers(UID:string)
  - Se non già presente, aggiunge il parametro UID all'attributo "followers".