2.4 PROJECT APIs

2.4.1 Resource extraction from Class Diagram

Analizzando il Class Diagram, siamo arrivati alla conclusione che le API utilizzate dal nostro sistema sono quelle visibili in foto, però una volta iniziata l’implementazione del front-end ci siamo accorti che alcuni API le potevamo implementare direttamente in quest’ultimo, dunque le API che il sistema utilizzerà effettivamente sono quelle nella seconda foto. Di queste abbiamo deciso di sceglierne di implementarne solo alcune. Le API che abbiamo effettivamente implementato sono le API della classe ‘Ricerca’, quindi le API ‘link’, ‘docente’ e ‘database’.

L’API ‘docente’ è una GET, cioè utilizza il metodo GET per recuperare la rappresentazione di una risorsa, nel nostro caso la pagina url del docente.

L’API ‘link’, anch’essa una GET, è la risorsa che ritorna l’url di una pagina la quale contiene le informazioni richieste.

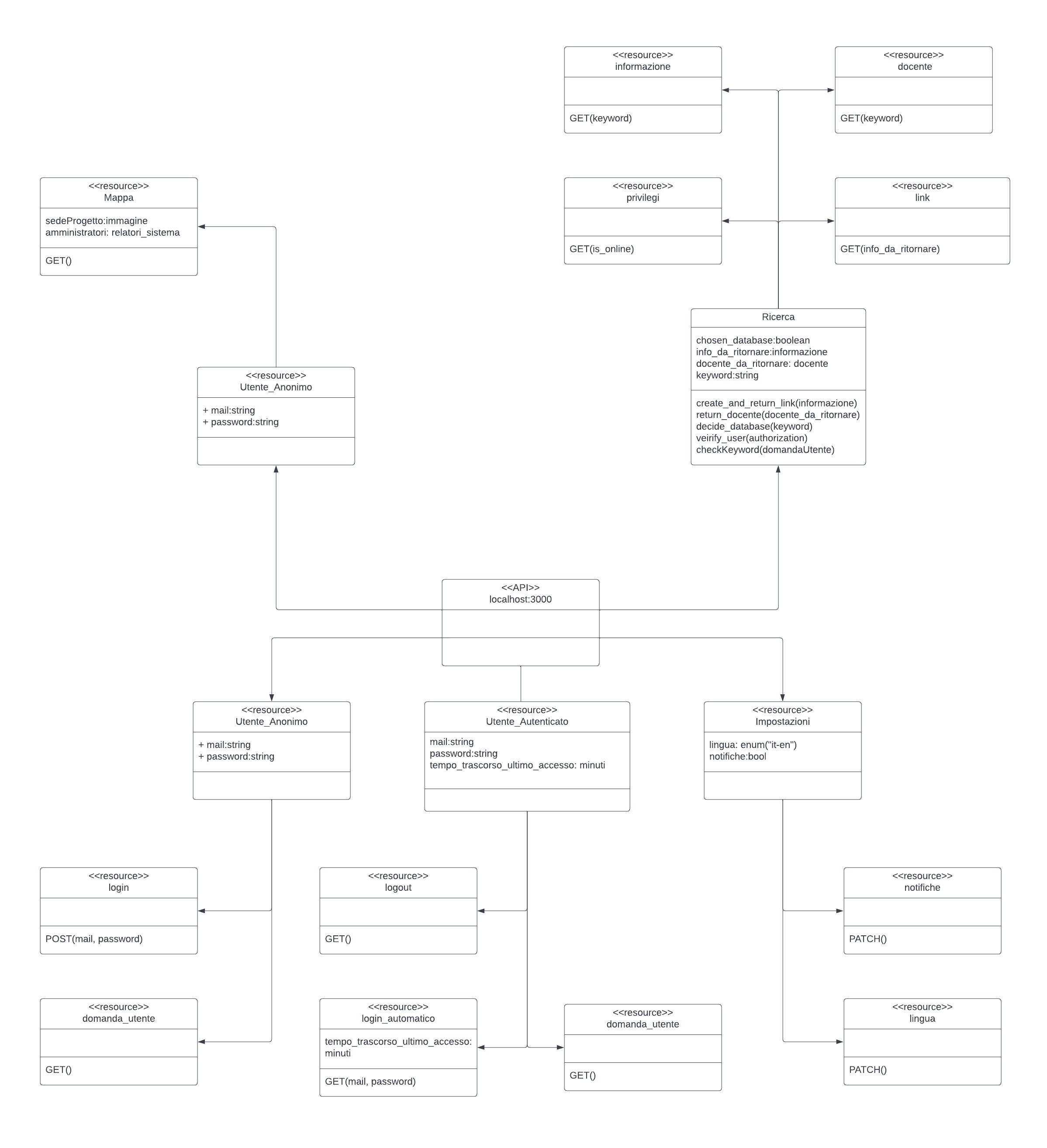
L’API ‘informazione, anch’essa una GET, è la risorsa che ritorna un informazione richiesta in base alle keyword inserite dall’utente.

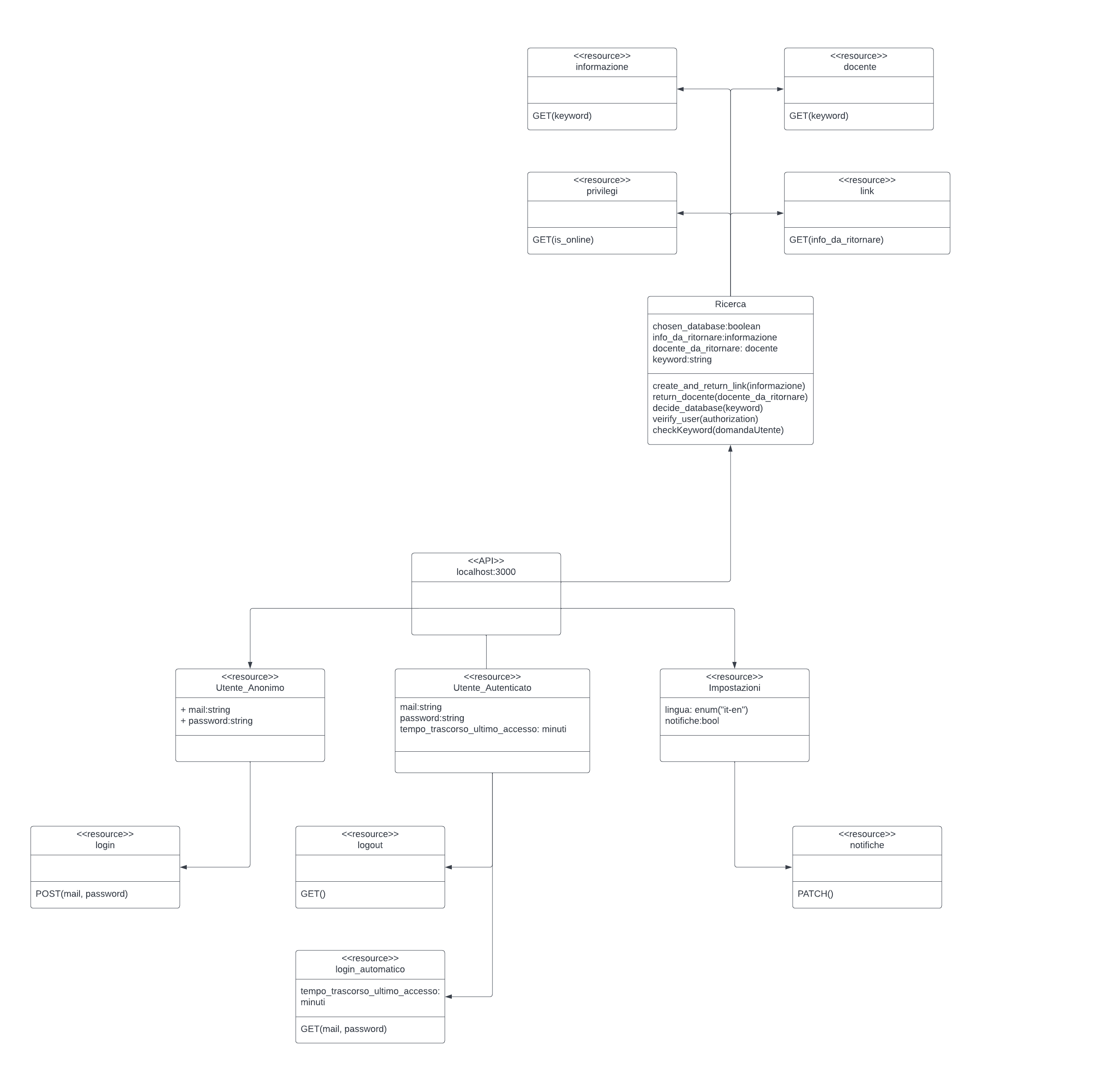
Nelle API ‘informazione e ‘docente’ il dato di partenza su cui si basano le loro funzionalità è una keyword. Invece nell’API ‘link’ il dato di partenza è cosa verrà ritornato dall’API ‘informazione’  
L’API non implementata della classe ‘ricerca’ è ‘privilegi’ e anch’essa è una GET e ritorna se l’utente è attualmente autenticato o no.

Analizzando anche il resto del diagramma notiamo che sono presenti altre classi con le loro relative API. L’API della classe ‘Utente\_Anonimo’ è ‘login’. Anch’essa è una GET e permette all’utente anonimo di diventare un utente autenticato. Questa API prende come informazioni l’email e la password dell’utente.

Le API della classe ‘Utente\_autenticato’ sono ‘logout’ e ‘login\_automatico’. L’API ‘logout’ permette all’utente autenticato di poter diventare un utente anonimo e l’API ‘login\_automatico’ permette ad un utente di loggarsi automaticamente. Per fare ciò all’API serve avere l’informazione ‘tempo\_trascorso\_ultimo\_accesso’, in quanto l’utente può effettuare il login automatico se e solo se il tempo passato dall’ultimo logout è inferiore a 15 minuti. I dati di partenza su cui si basa questa funzionalità sono l’email e la password dell’utente, le quali verranno poi ad essere prese da MongoDB attraverso il metodo GET.

La classe ‘impostazione’ ha come sua unica API ‘notifiche’. Questa API permetta ad un utente autenticato di poter aggiornare la propria preferenza notifiche, la quale di default viene disattivata. Dovendo semplicemente aggiornare una risorsa il metodo di questa API è PATCH.ù





2.4.2 Resorces models

Anche nel Resource Models Diagram abbiamo descritto delle API che il sistema utilizza, ma che abbiamo deciso di non implementare: le due API che abbiamo deciso di implementare sono quelle di ricerca ‘docenti’ e ‘informazioni’. Entrambe sono del tipo GET ed entrambe hanno come URI ‘ricerca’. Alle due API viene passato come body request la risorsa ‘ricerca’ la quale ha come parametri ‘keyword’ di tipo stringa, ‘is\_online’ di tipo booleano e ‘chosen\_database’ anch’esso booleano. ‘chosen\_database’ abbiamo deciso di inizializzarlo come booleano in quanto avendo solo due database sui quali possiamo andare a reperire le informazioni abbiamo determinato che questa fosse la soluzioni più logica in quanto abbiamo attribuito ‘true’ al database dei docenti e ‘false’ al database delle informazioni. La response body di queste API possono essere quattro: ‘found’, ‘not\_found’, ‘unathorized’ e ‘forbidden’. ‘found’ significa che l’API è riuscita a trovare ciò che le è stato richiesto e il codice HTTP che ritorna è 200 OK; ‘not\_found’ significa che l’API non è riuscita a trova l’informazione o il docente richiesto e il codice HTTP che ritorna è 404 NOT FOUND; ‘unauthorized’ significa che l’informazione richiesta non può essere ritornata in quanto per accedervi bisogna essere un utente autenticato (cfr D1 obiettivi) e il codice HTTP che ritorna è 401 UNAUTHORIZED. ‘forbidden’ significa che l’API non permette agli utenti di poter accedere a quelle specifiche informazioni in quanto sono in manutenzione da parte degli amministratori del sistema. In tutti e quattro i casi viene inviato un messaggio all’utente.  
Le altri API presenti nel Resource Models sono ‘login\_automatico’, ‘logout’, entrambe di tipo GET, ‘login’, di tipo POST, e ‘notifiche’ di tipo PATCH. L’API ‘login’ ha come body request la risorsa ‘utente’ la quale ha come parametri mail e password, entrambi di tipo stringa. Le reponse body che questa API può ritornare è una tra queste 3: ‘access’, ‘error’ e ‘server\_error’. ‘access’ significa che la mail e la password inserite dall’utente sono corrette e che quindi il login è stato effettuato; ‘error’ significa che o la mail o la password inserite sono errate e ‘server\_error’ significa che vi è presente un errore nel server dell’università. L’API ‘login\_automatico’, anch’essa di tipo GET, ha come body request la risorsa ‘utente’ la quale ha come parametri mail, password e tempo\_trascorso\_ultimo\_accesso. Mail e password sono di tipo string mentre tempo\_trascorso\_ultimo\_accesso è di tipo int. Le responde body fornite da questa API sono due: ‘access’ e ‘error’. ‘access’ significa che il login\_automatico è stato effettuato ed ha come codice HTTP 200 OK, ‘error’ significa che il login non è stato effettuato e questo viene fatto notare all’utente attraverso un messaggio di tipo stringa. Il codice HTTP di questa responde body è 400 BAD REQUEST.  
L’API ‘logout’ ha come body request la risorsa ‘utente’ la quale ha come parametri ‘is\_online’ di tipo booleano e ‘mail’ di tipo stringa. Le responde body di questa API sono ‘logout’, ‘error’ e ‘unauthorized’. ‘logout’ significa che il logout è avvenuto correttamente e il codice di questa responde è 200 OK, ‘error’ significa che vi è stato un errore durante l’operazione di logout, ‘unauthorized’ significa che un utente non autenticato ha cercato di effettuare il logout.

L’ultima API presente nel resource model diagram è ‘notifiche’. Quest’API ha come request body la risorsa ‘utente’ la quale ha come parametri ‘is\_online’ di tipo booleano, ‘mail’ di tipo stringa e ‘preferenza\_notifiche’ anch’essa di tipo stringa. Sono due le response body di questa API: ‘updated’ e ‘unauthorized’. ‘updated’ significa che la richiesta di aggiornare la propria preferenza notifiche è stata aggiornata: il codice HTTP è 200 OK. ‘unauthorized’ significa che un utente non autenticato ha cercato di aggiornare la propria preferenza mail e il codice HTTP è 401 UNAUTHORIZED. Entrambe queste risposte ritornato all’utente un messaggio di tipo stringa

