



SAPIENZA
UNIVERSITÀ DI ROMA

HiSchool: sviluppo di un'applicazione per il supporto didattico per la scuola secondaria

Facoltà di Ingegneria dell'Informazione, Informatica e Statistica
Corso di Laurea in Ingegneria Informatica e Automatica

Candidato
Silvia del Piano
Matricola 1759992

Relatore
Prof. Roberto Beraldì

Anno Accademico 2018/2019

Tesi non ancora discussa

HiSchool: sviluppo di un'applicazione per il supporto didattico per la scuola secondaria

Tesi di Laurea. Sapienza – Università di Roma

© 2019 Silvia del Piano. Tutti i diritti riservati

Questa tesi è stata composta con L^AT_EX e la classe Sapthesis.

Versione: today

Email dell'autore: delpiano.1759992@studenti.uniroma1.it

Da vedere

Indice

1 Raccolta dei requisiti	1
1.1 Descrizione della realtà di interesse	1
1.2 Dati gestiti dal sistema e principali funzionalità	2
1.2.1 MockUp e User Stories	2
1.2.2 Lo-Fi MockUp	3
1.2.3 User Stories	6
2 Tecnologie e Metodologie utilizzate	11
2.1 Rails e il paradigma Model-View-Controller	11
2.1.1 Autenticazione locale	11
2.1.2 Autenticazione tramite Google OAuth	12
2.1.3 Google Drive	13
2.1.4 Altre gemme	13
2.2 Agile per lo sviluppo software	15
2.2.1 Tool per applicare al meglio Agile	15
3 Analisi Concettuale del Sistema	17
3.1 Schema Entità-Relazione	17
3.2 Elenco delle entità e delle relazioni	18
3.2.1 Generalizzazione User	18
3.2.2 Entità School e Class	18
3.2.3 Entità Post	19
4 Progettazione del sistema	20
4.1 Archittura del sistema	20
4.1.1 Architettura dei Controller	20
4.1.2 Architettura dei Model	21
4.1.3 Architettura delle View	22
4.2 Progettazione logica del Data Layer	23
5 Realizzazione del sistema	26
5.1 Application Layer	26
5.2 Presentation Layer	30
5.2.1 Aspetto dell'applicazione	33

6 Dispiegamento e Validazione	40
6.1 Installazione	40
6.2 Test	40

Capitolo 1

Raccolta dei requisiti

1.1 Descrizione della realtà di interesse

Un giorno parlando con i miei colleghi universitari, si è accennato alla comodità offerta dai siti relativi ai singoli corsi di studio. Essi infatti sono un porto sicuro per ogni studente iscritto, frequentante e non: è possibile trovarvi il materiale su cui studiare, informazioni e contatti dei professori a cui sono affidate le lezioni, loro comunicazioni e spesso molto altro. Tutti sanno che per qualsiasi dubbio possa sorgere a proposito di uno specifico corso il primo posto dove andare a guardare è il sito ad esso dedicato. Tutto ciò mi ha invogliato a realizzare qualcosa di simile per la scuola secondaria: una piattaforma che i principali attori coinvolti (studenti, insegnanti e genitori) possano usare per facilitare la formazione dei ragazzi ed il suo monitoraggio.

I servizi fondamentali per la didattica offerti sono i seguenti:

- Possibilità di visionare e gestire (per gli insegnanti) il materiale didattico.
- Avere sempre a portata di mano i corsi frequentati/insegnati e le informazioni sui relativi docenti.
- Ricevere/dare (per gli insegnanti) comunicazioni tramite post alle classi di interesse per gli utenti (le quali saranno la classe frequentata per l'alunno, la classe di cui fa parte il figlio per il genitore, e le classi affidate all'insegnante).
- Avere a disposizione un calendario dove saranno presenti le comunicazioni importanti (secondo quanto specificato al momento della loro pubblicazione).

In quanto pensata per affiancarsi alle piattaforme già esistenti per la scuola, nell'applicazione vengono memorizzati i dati anagrafici degli utenti, in modo che in futuro i servizi offerti possano essere integrati o usati parallelamente a quelli istituzionali già presenti. Sempre in quest'ottica, gli unici che possono inserire una scuola con le relative classi nel database dell'applicazione sono gli sviluppatori.

1.2 Dati gestiti dal sistema e principali funzionalità

Essa prevede tre tipi di utenti: gli studenti (Student), gli insegnanti (Teacher) e i genitori (Parent). Di ogni utente sono memorizzati in un database il nome, il cognome, il codice fiscale, la data e il luogo di nascita, il sesso, la scuola (tra quelle registrate sulla piattaforma dagli sviluppatori), la classe (tra quelle registrate per ciascuna scuola), il ruolo (tra Studente, Insegnante e Genitore). Saranno memorizzati anche una mail e una password con cui si potrà effettuare il login al sito, e nel caso del Genitore, anche il nome e il cognome del figlio iscritto alla scuola. Ad ogni modo, per agevolare l'accesso alla piattaforma sarà fornito anche il servizio di autenticazione tramite Google. Ogni utente ha una propria Dashboard da dove, a seconda del ruolo, può usufruire di diverse funzionalità. Tutti gli utenti possono vedere le informazioni fondamentali associate al proprio account, il calendario e i post contenenti comunicazioni relative alla classe di interesse pubblicate dagli insegnanti; le comunicazioni possono avere associata anche una scadenza che verrà mostrata sul calendario. I post saranno memorizzati nell'applicazione. Lo Studente può navigare nell'applicazione per consultare le singole pagine relative alle materie studiate, dove è caricato il materiale di cui può fare il download. L'Insegnante può creare i post; togliere, fare il download e l'upload delle risorse relative alla materia che insegna.

Per semplicità nell'implementazione si assume che ogni Insegnante abbia affidata una sola materia, che ogni Genitore possa avere solo un figlio iscritto alla scuola, che ogni materia sia spiegata da un solo Insegnante e che abbia senso definire una scuola senza classi; tutto ciò considerando la classe frequentata dall'alunno. Tuttavia, lo schema del database è pensato per una maggiore scalabilità, e il codice l'applicazione è stato scritto in modo che essa sia facilmente modificabile o ampliabile.

Tutti i file che costituiscono risorse per gli Studenti sono conservati sul Google Drive associato all'account dell'applicazione.

1.2.1 MockUp e User Stories

Le funzionalità e l'aspetto che l'applicazione deve avere sono meglio specificate rispettivamente nelle user stories e nei lo-fi mockup.

Questi due strumenti sono fondamentali nella fase di raccolta dei requisiti e progettazione: fungono da ponte tra il team che si occuperà di relizzare il software e i committenti, così che le aspettative di questi ultimi siano pienamente soddisfatte. Essi infatti sono abbastanza generici in modo che chiunque possa comprenderli, ma allo stesso tempo sono sufficientemente specifici da poter essere implementati dagli sviluppatori.

1.2.2 Lo-Fi MockUp

Di seguito i lo-fi mockup, realizzati con carta e penna, rappresentano ciò che l'utente vedrà sulla piattaforma e come potrà navigarla.

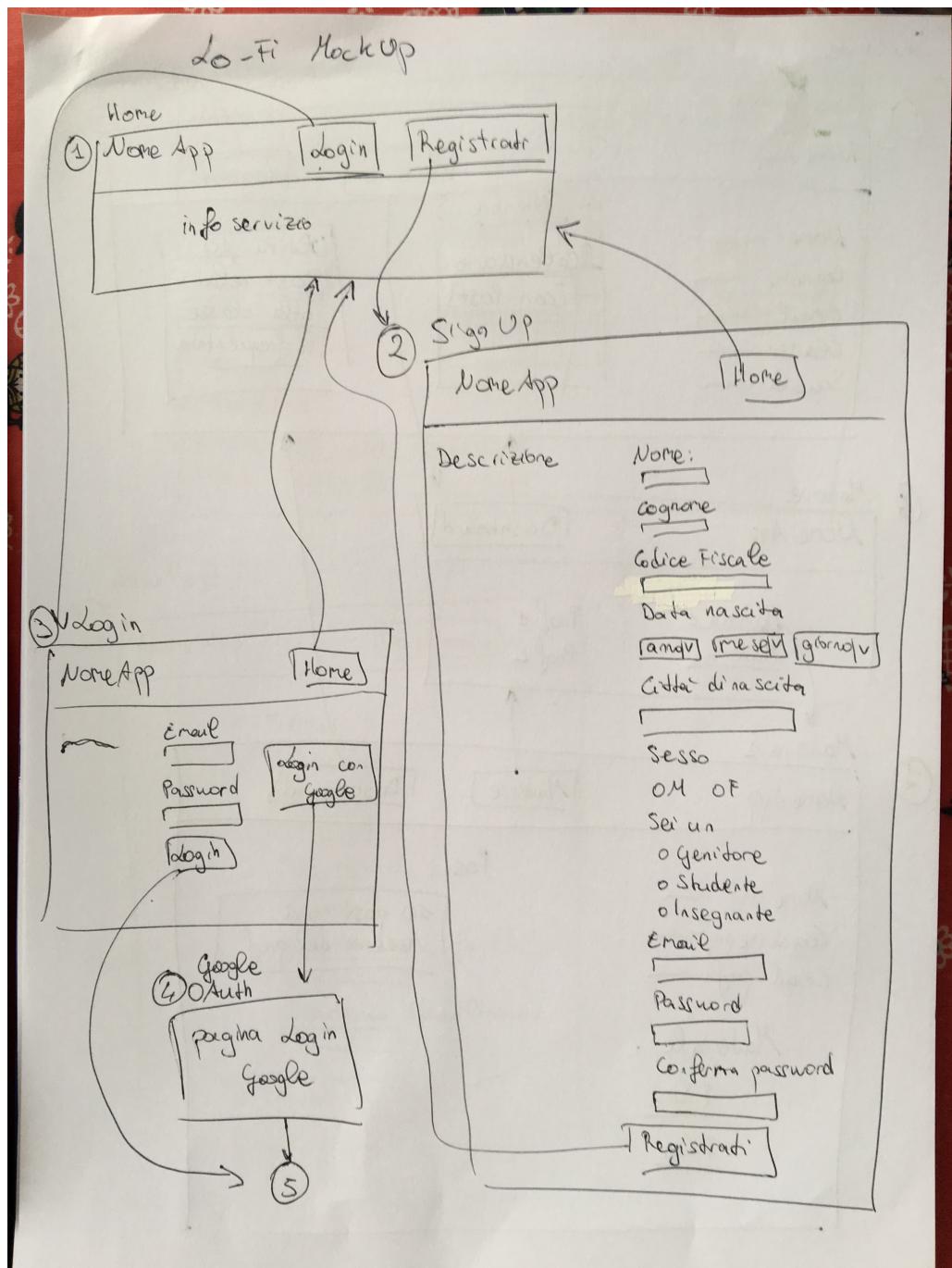


Figura 1.1. Lo-Fi-Mockup 1

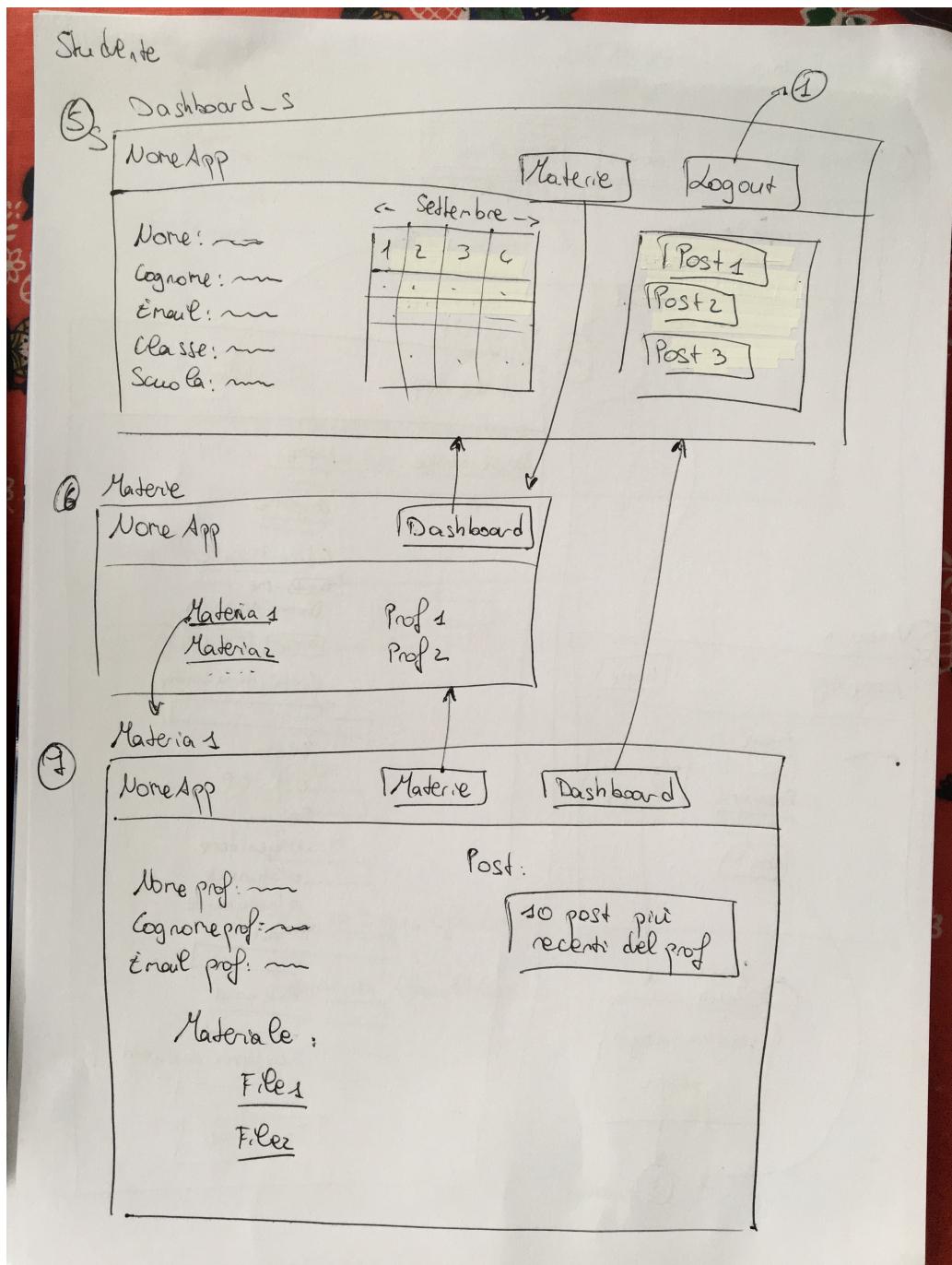


Figura 1.2. Lo-Fi-Mockup 2

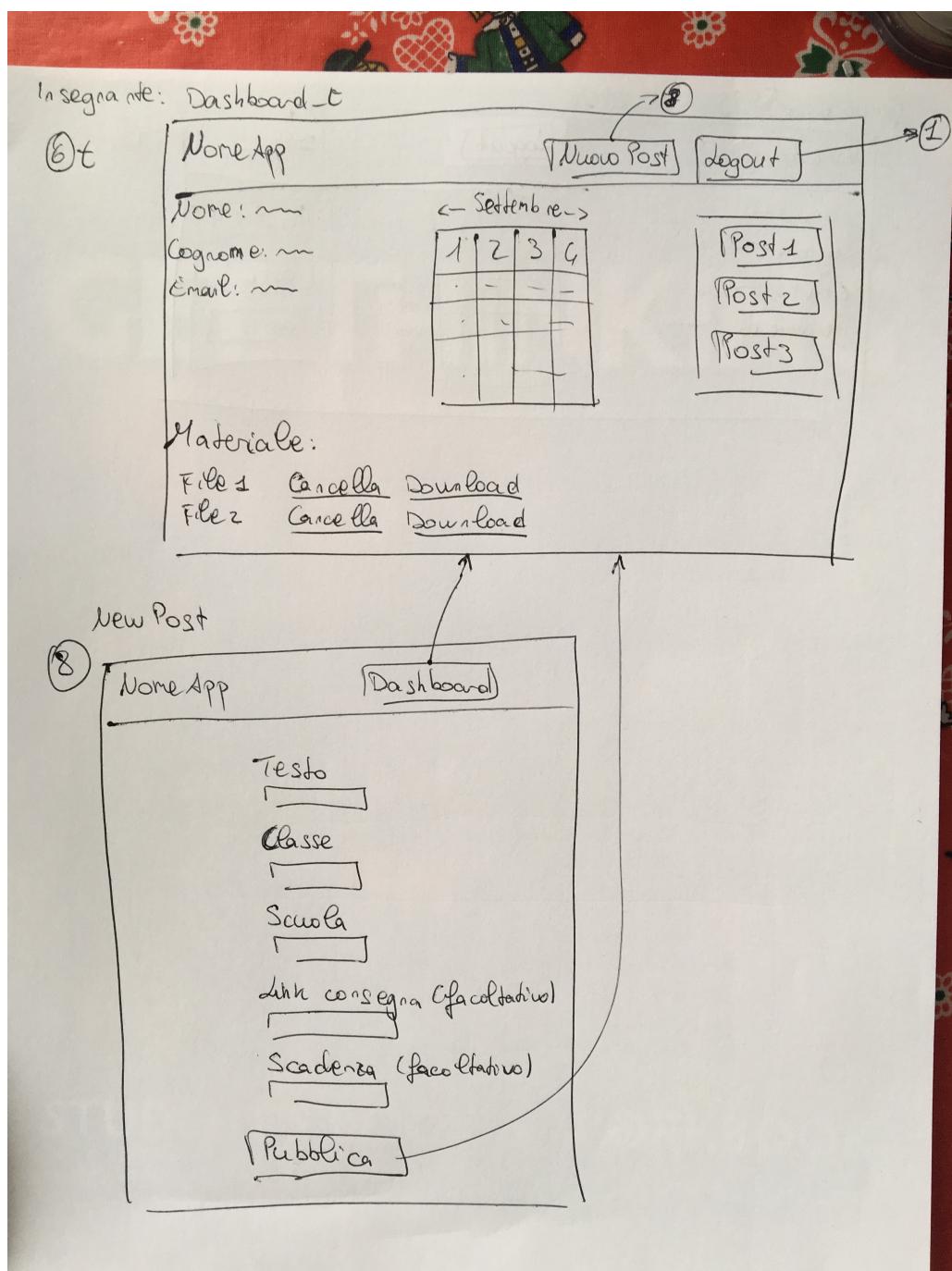


Figura 1.3. Lo-Fi-Mockup 3

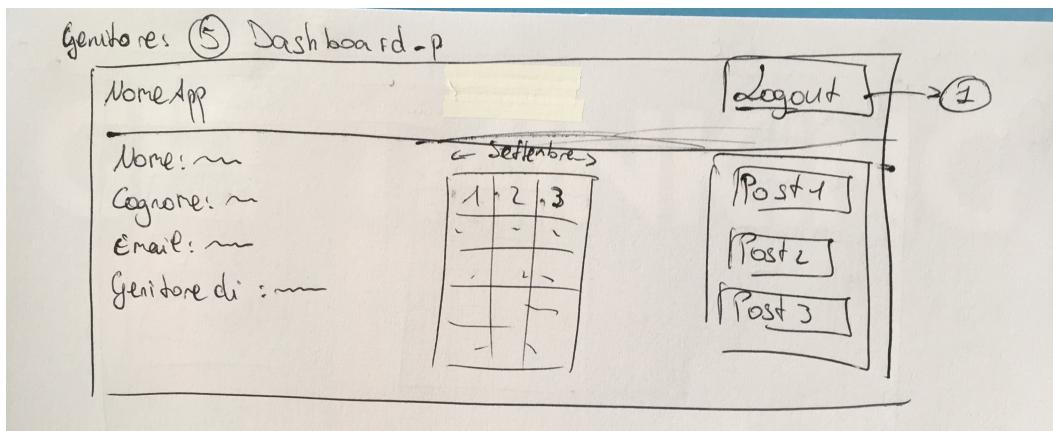


Figura 1.4. Lo-Fi-Mockup 4

1.2.3 User Stories

Di seguito si riporta l'elenco delle user stories: descrivono brevemente ma in maniera efficace le azioni che verranno implementate fornendo i servizi precedentemente descritti agli utenti.

1. Go to the "SignUp" page from the "Home" page

As a user
So that I can sign up
Given I'm in the "Home" page
When I press the button "SignUp"
Then I should be in the "SignUp" page

2. SignUpAs a user

So that I can use the app's services
Given I'm in the "SignUp" page
When I fill the form correctly
and I press the button "SignUp"
Then I should be registered in the app
and I should be in the "Home" page

3. Return to the "Home" page from the "SignUp" page

As a user
So that I can return to the "Home" page without signing up
Given I'm in the "SignUp" page
When I press the button "Home"
Then I should be in the "Home" page

4. Go to the "LogIn" page from the "Home" page

As a user

So that I can login
Given I'm in the "Home" page
When I press the button "Login"
Then I should be in the "Login" page

5. LogIn without Google

As a user
So that I can use the app's services
Given I'm in the "LogIn" page
and that I've signed up as a "Parent"
When I fill the form correctly
and I press the button "LogIn"
Then I should be logged in as a "Parent" and I should be in the "Dashboard_g" page

6. LogIn with Google

As a user
So that I can use the app's services
Given I've signed up as a "Student" or as a "Teacher"
I want to log in with Google OAuth

7. Go to the "Home" page from the "LogIn" page

As a user
So that I can return to the "Home" page without logging in
Given I'm in the "LogIn" page
When I press the button "Home"
Then I should be in the "Home" page

8. LogOut

As a user
So that I can stop using the app's services
Given I'm in the correct "Dashboard_%" page
When I press the button "LogOut"
Then I should have logged out correctly
and I should be in the "Home" page

9. Calendar

As a user
So that I can know the current date and organize my work properly
Given I'm in the correct "Dashboard_%" page
Then I should see the current date

10. Posts seen by "Student" As a "Student"

So that I can see the posts all my "Teacher"s made
Given I'm on the "Dashboard_s" page
Then I should see the 10 most recent posts

11. Go to the "Dashboard_%" page

As a user
So that I can go to the "Dashboard_%" whenever I need it
When I press the button "Dashboard"
Then I should be in the correct "Dashboard_%" page

12. Materie

As a "Student"
So that I can see the subjects teached at school
Given I'm on the "Dashboard_s" page
When I press the button "Materie"
Then I should be on the "Materie" page

13. Subject info

As a "Student"
So that I can have more information about a subject
Given I'm in the "Materie" page
When I click che link corresponding to subject *
Then I should be in the page "Materia*" corresponding to subject *

14. Go to the "Materie" page from "Materia*" page

As a "Student"
So that I can go to the "Materie" page from the "Materia*" page
Given I'm in the "Materia*" page
When I press the button "Materie"
Then I should be in the "Materie" page

15. Resources for a specific subject

As a "Student"
So that I can use the resources on subject*
Given I'm in the "Materia*" page
and I see the files that are the resources
When I click the link corresponding to a file
Then I should be able to download that file

16. "Teacher"'s posts

As a "Student"
So that I can have a focus on the specific subject*

Given I'm in the "Materia*" page

Then I should see the 5 most recent posts made by the "Teacher" corresponding to that subject

17. Posts seen by "Teacher"

As a "Teacher"

So that I can see understand the workload my "Student"s have

Given I' in the "Dahsboard_t" page

Then I should see the 10 most recent posts made by me and the other "Teacher"s who teach my "Student"s

18. Remove a file

As a "Teacher"

So that I can remove files

Given I'm in the "Dashboard_t" page

When I click on the "Cancella" link corresponding to the file I want to remove

Then the file corresponding to the link I clicked should be removed

19. Download a file

As a "Teacher"

So that I can download files

Given I'm in the "Dashboard_t" page

When I clicked on the "Download" link corresponding to the file I want to download

Then the file corresponding to the link I clicked should be downloaded

20. Upload a file

As a user

So that I can upload a file

Given I'm in the "CaricoFile" page

And that I've specified what file I want to upload

When I press the button "Aggiungi"

Then the file should be uploaded correctly

21. Go to the "NewPost" page from the "Dashboard_t" page

As a "Teacher"

So that I can communicate with my "Student"s and assign homework to them

Given I'm in the "Dashboard_t" page

When I press the button "Nuovo Post"

Then I should be in the "NewPost" page

22. Publish a post

As a "Teacher"

So that I can communicate with my "Student"s and assign homework to them
Given I'm in the "NewPost" page
and that I filled the form correctly
When I press the button "Pubblica"
Then a new post should be published correctly

23. Posts seen by a "Parent"

As a "Parent"
So that I can be updated on my son's scholastic situation
Given I'm in the "Dashboard_p" page
Then I should see the posts made by my son's "Teacher"s

Capitolo 2

Tecnologie e Metodologie utilizzate

2.1 Rails e il paradigma Model-View-Controller

L'applicazione è stata realizzata con Ruby on Rails, seguendo i principi su cui esso è basato, e varie gemme in esso contenuto.

In particolare, Rails è un framework per lo sviluppo di applicazioni web SaaS (Software as a Service) scritto in Ruby, ed è stato sviluppato tenendo a mente le più importanti linee guida nella progettazione e programmazione di questo tipo di software. Due dei principi fondamentali sono: Don't Repeat Yourself (non ci deve essere codice ridondante, ma una singola scrittura efficace delle varie funzionalità e caratteristiche) e Conventions Over Configurations (bisogna specificare delle configurazioni solo nel caso in cui non si seguano quelle raccomandate naturalmente dal framework). Tutto ciò è implementato col paradigma Model-View-Controller: i modelli servono per interagire con i dati del sistema, le view li visualizzano in maniera appropriata rendendo possibile l'interazione con gli utenti, mentre i controller ricevono i comandi di questi (solitamente attraverso le view e il browser) e li eseguono.

2.1.1 Autenticazione locale

Per i processi di autenticazione e autorizzazione è stata utilizzata la gemma Canard, per implementare un modello di Role-Based Access Control, insieme alla gemma Devise. Devise è uno strumento molto utile per ogni sviluppatore Rails: è organizzata in moduli, per cui è possibile costruire dal più semplice meccanismo di identificazione e verifica, a quello più complesso con diversi servizi aggiuntivi. In questa applicazione sono stati usati soprattutto i seguenti moduli:

- **Database Authenticatable:** memorizza in maniera sicura e cifrata (grazie al supporto della gemma Bcrypt) la password dell'utente nel database del sistema
- **Registrable:** permette agli utenti di registrarsi
- **Rememberable:** gestisce le sessioni degli utenti tramite i cookie

- **Validatable:** convalida le email e le password
- **Omniauthable:** gestisce autenticazione tramite OAuth (ulteriori dettagli nella sezione successiva).

Ulteriori comodità di questa gemma sono le route e le view da essa create automaticamente per tutti i processi relativi all'autenticazione, personalizzabili in maniera opportuna

2.1.2 Autenticazione tramite Google OAuth

OAuth è un protocollo di rete basato su HTTP che consente di delegare l'autenticazione ad una terza entità (quale potrebbe essere Google, Facebook, Twitter...). In esso sono definiti quattro ruoli principali:

1. **Utente (Resource Owner):** è il possessore dell'account, e deve autorizzare l'accesso alle informazioni contenute in esso e alle risorse protette conservate nel Resource Server
2. **Client (Consumer):** è l'applicazione che vorrebbe accedere alle informazioni e risorse dell'Utente, di cui chiede l'autorizzazione
3. **Resource Server:** il server dove si trovano le risorse protette dell'Utente
4. **Authorization Server:** è il server che si occupa di richiedere all'Utente di autenticarsi (con le credenziali di quest'ultimo) per autorizzare l'accesso alle sue risorse

Lo schema di funzionamento è descritto dalla seguente immagine:

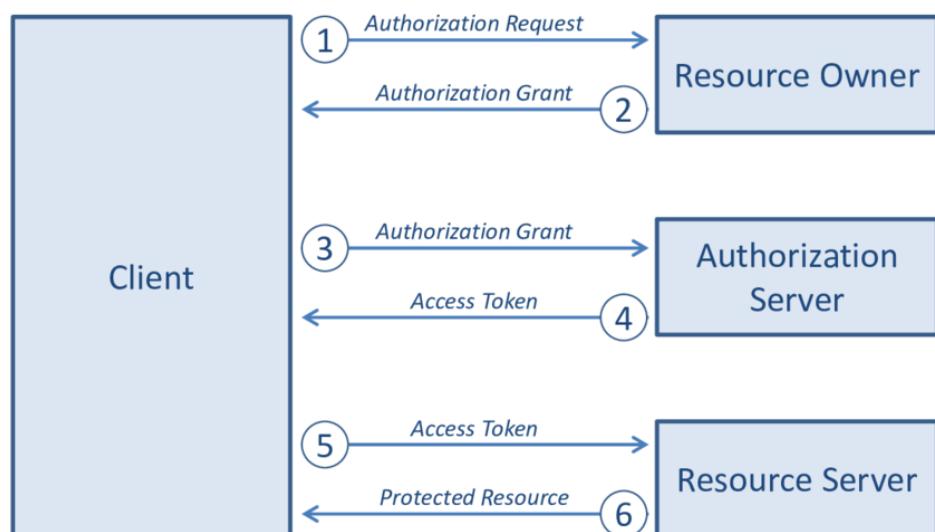


Figura 2.1. OAuth Flow

In questo caso, l'Authorization Server e il Resource Server coincidono (Google), mentre il ruolo del client è svolto dalla nostra applicazione.

Per implementare tutto ciò ci si è serviti, come accennato in precedenza, del modulo `Omniauthable` di Devise in aggiunta alla gemma `omniauth-google-oauth2`.

2.1.3 Google Drive

Per memorizzare le risorse a disposizione degli studenti si è usato Google Drive. La prima cosa da fare è stato associare all'applicazione un service account dalla Google API Console. Un service account è un particolare account Google che rappresenta un' entità non umana che necessita di accedere a delle risorse protette. Di conseguenza all'applicazione in esame sono stati associate una email, una password e un ID per Google.

Per gestire i file (upload, download) ed interagire con questo servizio è stata usata la gemma `google-api-client`.

2.1.4 Altre gemme

Per una più completa trattazione dello sviluppo si è cominciato ad impostare il deploy su Heroku, tramite le gemme `rails_12factor`(per configurare l'ambiente di produzione dell'applicazione), `pg` (in quanto il database di Heroku è basato su PostgreSQL e non su SQLite usato per lo sviluppo) e `figaro` (per gestire le chiavi e gli ID segreti negli ambienti di sviluppo e produzione).

Per realizzare il calendario nella Dashboard degli utenti si è usata `simple_calendar`, mentre per fare i diagrammi UML dei controller e dei model si è utilizzata `railroady`. Di seguito è riportato il `Gemfile` con tutte le gemme utilizzate:

```

1 source 'https://rubygems.org'
2
3 git_source(:github) do |repo_name|
4   repo_name = "#{repo_name}/#{repo_name}" unless repo_name.include?("/")
5   "https://github.com/#{repo_name}.git"
6 end
7
8 # Fix Ruby version
9 ruby '2.4.1'
10 # Bundle edge Rails instead: gem 'rails', github: 'rails/rails'
11 gem 'rails', '~> 5.1.7'
12 # Use Puma as the app server
13 gem 'puma', '~> 3.7'
14 # Use SCSS for stylesheets
15 gem 'sass-rails', '~> 5.0'
16 # Use Uglifier as compressor for JavaScript assets
17 gem 'uglifier', '>= 1.3.0'
18 # See https://github.com/rails/execjs#readme for more supported runtimes
19 # gem 'therubyracer', platforms: :ruby
20
21 # Use HAML to render the views
22 gem 'haml'
23
24 # Bootstrap 3 for styling

```

```
25 gem 'bootstrap-sass'
26
27 # Calendar
28 gem "simple_calendar", "~> 2.0"
29
30 # Google Drive
31 gem 'google-api-client', '~> 0.11'
32
33 # Use CoffeeScript for .coffee assets and views
34 gem 'coffee-rails', '~> 4.2'
35 # Turbolinks makes navigating your web application faster. Read more:
36     https://github.com/turbolinks/turbolinks
37 gem 'turbolinks', '~> 5'
38 # Build JSON APIs with ease. Read more: https://github.com/rails/jbuilder
39 gem 'jbuilder', '~> 2.5'
40 # Use Redis adapter to run Action Cable in production
41 # gem 'redis', '~> 4.0'
42 # Use ActiveModel has_secure_password
43 gem 'bcrypt', '~> 3.1.7'
44
45 # Security help
46 gem 'devise', '~> 4.7.1'
47 gem 'canard', '~> 0.5.0.pre'
48 gem 'omniauth-google-oauth2'
49
50 # Use Capistrano for deployment
51 # gem 'capistrano-rails', group: :development
52
53 gem 'figaro'
54
55 group :production do
56     # Use PostgreSQL in production (Heroku)
57     gem 'pg'
58     # Heroku-specific production settings
59     gem 'rails_12factor'
60 end
61
62 group :development, :test do
63     # Call 'byebug' anywhere in the code to stop execution and get a debugger console
64     gem 'byebug', platforms: [:mri, :mingw, :x64_mingw]
65     # Use sqlite3 as the database for Active Record
66     gem 'sqlite3'
67     gem 'coveralls', :require => false # to use Coveralls.io for test coverage
68     gem 'simplecov', :require => false
69     #gem 'jasmine-rails'           # to test JavaScript/CoffeeScript, not sure if it
90         will be used
70 end
71
72 group :development do
73     # Access an IRB console on exception pages or by using <%= console %> anywhere in
74         the code.
75     gem 'web-console', '>= 3.3.0'
76     gem 'listen', '>= 3.0.5', '< 3.2'
77     # Spring speeds up development by keeping your application running in the
90         background. Read more: https://github.com/rails/spring
78     gem 'spring'
79     gem 'spring-watcher-listen', '~> 2.0.0'
```

```

78  # Security testing
79  gem 'brakeman'
80  #gem 'guard-brakeman'
81 end
82
83 group :test do
84   gem 'gemsurance'           # to check gems' security
85   gem 'rspec-rails'
86   gem 'guard-rspec'
87   gem 'cucumber-rails', :require => false
88   gem 'cucumber-rails-training-wheels' # some pre-fabbed step definitions
89   gem 'database_cleaner'          # to clean Cucumber's test database between runs
90   gem 'capybara'                # lets Cucumber pretend to be a web browser
91   gem 'launchy'                  # a useful debugging aid for user stories
92   gem 'factory_bot_rails'
93 end
94
95 # Windows does not include zoneinfo files, so bundle the tzinfo-data gem
96 gem 'tzinfo-data', platforms: [:mingw, :mswin, :x64_mingw, :jruby]

```

Listing 2.1. Gemfile

Una discussione degli strumenti usati per i test si può trovare nella sezione TODO

2.2 Agile per lo sviluppo software

Nel realizzare questa applicazione mi sono ispirata alla metodologia Agile. Essa è in realtà pensata per gruppi di persone che lavorano insieme allo stesso software: le interazione tra i vari membri di un singolo team e tra i responsabili di questi hanno un'enorme importanza, e sono il vero punto di forza di questa metodologia, insieme al contatto con il committente.

Nonostante ciò, in questo lavoro ho voluto tenere fede ai principi di Behavior-Driven Design e Test-Driven Development. Di conseguenza, prima dello sviluppo del codice, nella fase di progettazione, sono stati realizzati i lo-fi mockup e le user stories per fare chiarezza su cosa doveva essere realizzato. Successivamente, durante lo sviluppo, per ogni user story veniva scritto un test e il codice per soddisfarlo, in modo che tutto fosse fatto in modo corretto. Ponendo l'attenzione sulla qualità e sui contenuti da subito si sono evitati lunghi e tortuosi processi di debug, oltre all'implementazione di funzionalità inutili o non richieste.

2.2.1 Tool per applicare al meglio Agile

Per applicare al meglio la metologia Agile e lavorare efficientemente ho utilizzato i seguenti strumenti:

- **Git:** per il versioning control. La repository su cui è conservato il progetto è la seguente: <https://github.com/SilviadelPiano/HiSchool.git>
- **Visual Studio Code** come editor
- **Draw.io** per la realizzazione degli schemi ER e UML

- **Pivotal Tracker** per il tracking degli sprint e per monitorare i progressi e l'andamento del lavoro. Il link alla pagina corrispondente all'applicazione in esame è il seguente: <https://www.pivotaltracker.com/n/projects/2393952>

Capitolo 3

Analisi Concettuale del Sistema

3.1 Schema Entità-Relazione

In questa sezione viene riportato lo schema ER del database. Come si può vedere esso rappresenta i requisiti descritti nella sezione 1.2, assieme ad altri vincoli necessari per lo sviluppo di questa applicazione. Sono indicati anche gli identificatori principali.

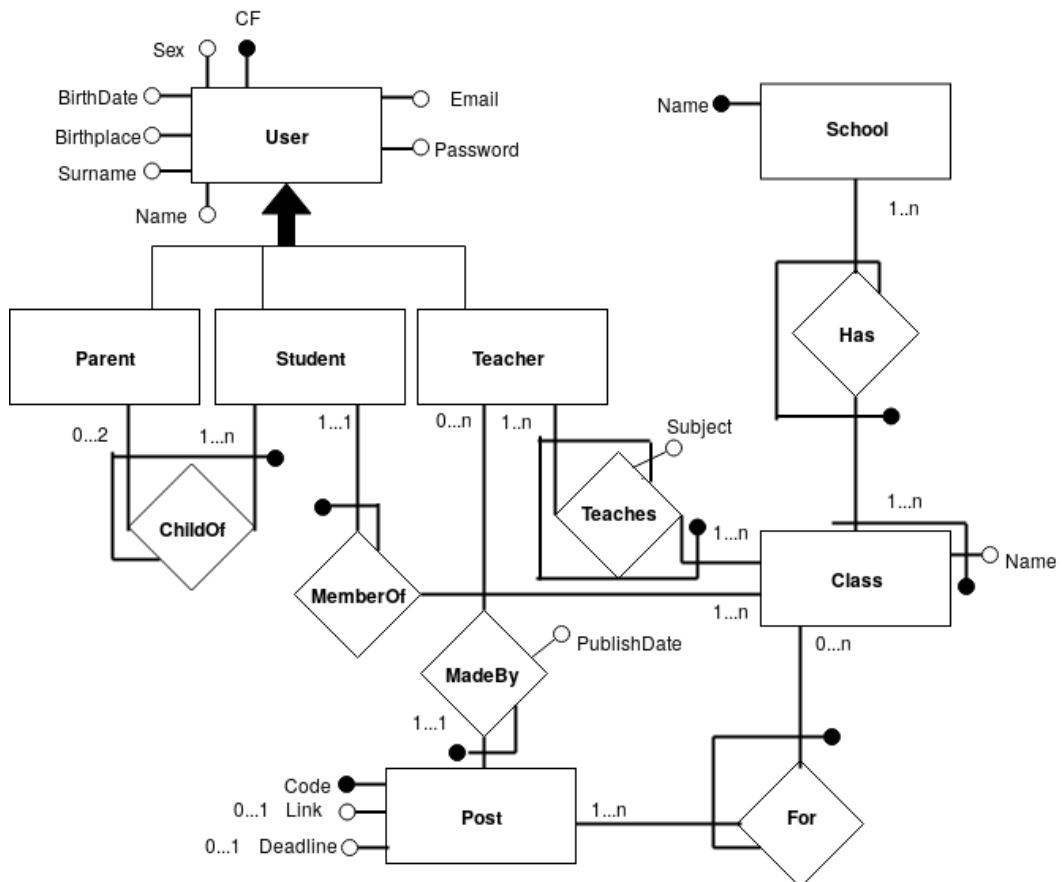


Figura 3.1. Schema ER

3.2 Elenco delle entità e delle relazioni

In questa sezione verranno presentate in maniera più approfondita le varie entità e relazioni situate nello schema.

3.2.1 Generalizzazione User

User è l'entità che rappresenta l'utente che andrà ad usufruire dell'applicazione: si può notare infatti la generalizzazione che differenzia i tre tipi di utilizzatori che si avranno.

Parent e **Student**, cioè le entità che rappresentano il genitore e lo studente, sono legate dalla relazione **ChildOf**: essa esprime la parentela tra lo studente iscritto e il genitore (potrebbe essere anche il tutore legale del ragazzo in mancanza di parenti più stretti) dello stesso. Un Parent per registrarsi sulla piattaforma deve avere almeno un figlio Student registrato su di essa. D'altro canto, uno Student non deve per forza avere un Parent ad esso associato per registrarsi, ma può averne al massimo 2 associati ad esso.

Durante la ristrutturazione Parent, Student e Teacher sono state accorpate a User per generare un'unica entità omonima, dove i vari ruoli sono distinti da una bitmask, tramite l'ausilio della gemma Canard.

3.2.2 Entità School e Class

L'entità **School** rappresenta la scuola registrata nel database dell'applicazione, ed è legata a **Class** tramite la relazione **Has**. Class è infatti identificata dal proprio nome e dalla Scuola a cui appartiene. Per esempio, la classe 1^a del liceo scientifico sarà presente in tutti i licei scientifici, ed almeno in 1, altrimenti non esisterebbe, da qui l'associazione 1...n. Una Scuola deve avere almeno una classe.

La relazione **Teaches** esprime l'incarico di insegnamento conferito ad un **Teacher** (insegnante) nei confronti della classe, ed è presente anche la materia (subject) oggetto di lezione. Ogni classe può essere presa in carico da diversi insegnanti, ma ne deve avere almeno 1 (si pensi solo alle varie materie presenti in un liceo) ed un singolo insegnante, per essere tale, deve fare lezione ad almeno un classe, ma può averne affidate più di una.

La relazione **MemberOf** invece esprime il fatto che ogni classe debba essere frequentata da almeno uno studente, e questo per essere tale deve, e può, essere iscritto ad una sola classe.

Nella ristrutturazione Class e Has sono state accorpate per generare un'unica entità che tenga traccia del nome della classe e della scuola ad essa associata. School rimane singolarmente, in modo che nel futuro sia possibile associare ad essa più classi. Inoltre, al momento della registrazione all'utente viene chiesto di specificare una classe ed una scuola (in diversi punti dei form da compilare a seconda del ruolo), e sono effettuati due controlli: si verifica l'esistenza della singola scuola (School) nel database, e poi si verifica che per questa sia registrata la classe dichiarata nel form tramite l'entità appena creata con l'accorpamento. Quest'ultima è utile anche per motivi di efficienza del codice.

3.2.3 Entità Post

L'entità **Post** rappresenta le comunicazioni che possono essere fatte sulla piattaforma. Ogni Post deve essere creato da un singolo insegnante secondo quanto espresso dalla relazione **MadeBy**, mentre un insegnante non ha l'obbligo di creare Post, e non c'è limite a quanti ne può fare. L'attributo PublishDate registra la data della pubblicazione. Da notare gli attributi facoltativi Link e Deadline. Essi sono stati inseriti per fornire maggiori opzioni al fine di rendere più efficaci le comunicazioni: come insegnante vorrei poter mettere in evidenza un link dove trovare materiale utile oppure una scadenza o una data importante relativa alla comunicazione appena pubblicata (che sarà segnata sul calendario).

Un Post deve essere associato ad almeno una classe, secondo quanto indicato dalla relazione **For**, in modo che i suoi membri possano vederlo.

Capitolo 4

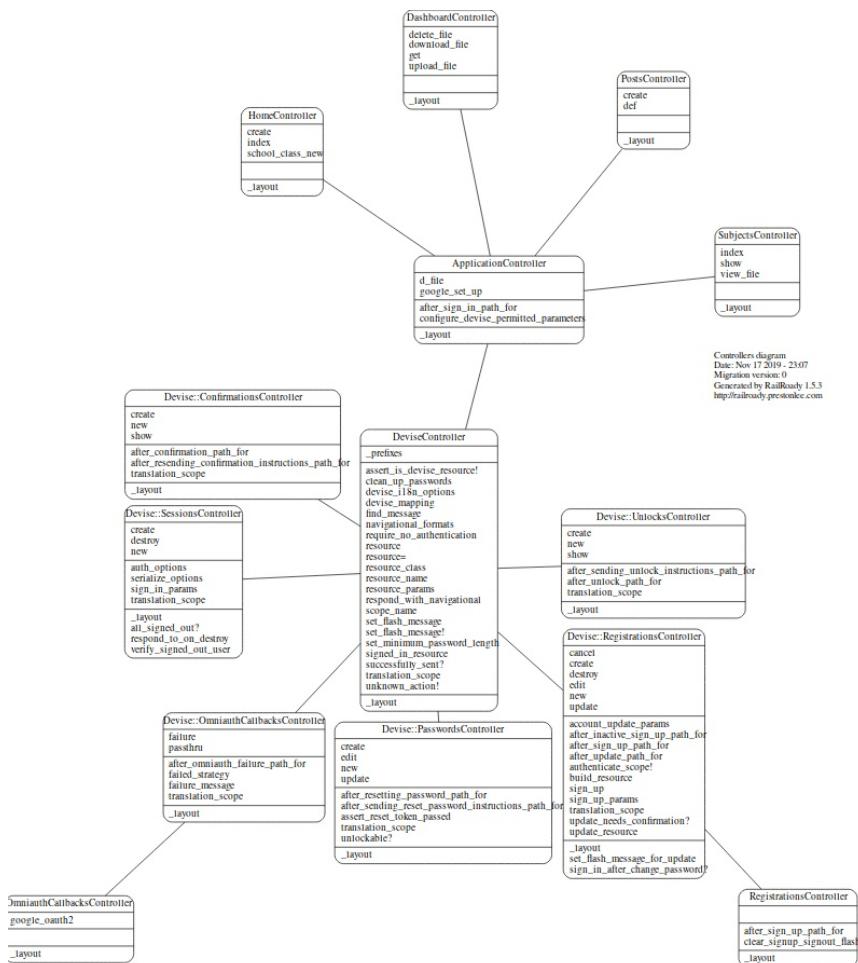
Progettazione del sistema

Questa applicazione è basata su tre layer principali: il Data Layer, l'Application Layer e il Presentation Layer. Di seguito sono esposti gli schemi architetturali per una maggiore comprensione del sistema.

4.1 Archittura del sistema

4.1.1 Architettura dei Controller

In questa sezione è riportato lo schema UML dei controller. (Figura 4.1).

**Figura 4.1.** Schema UML Controllers

4.1.2 Architettura dei Model

In questa sezione invece è riportato lo schema UML dei model. (Figura 4.2).

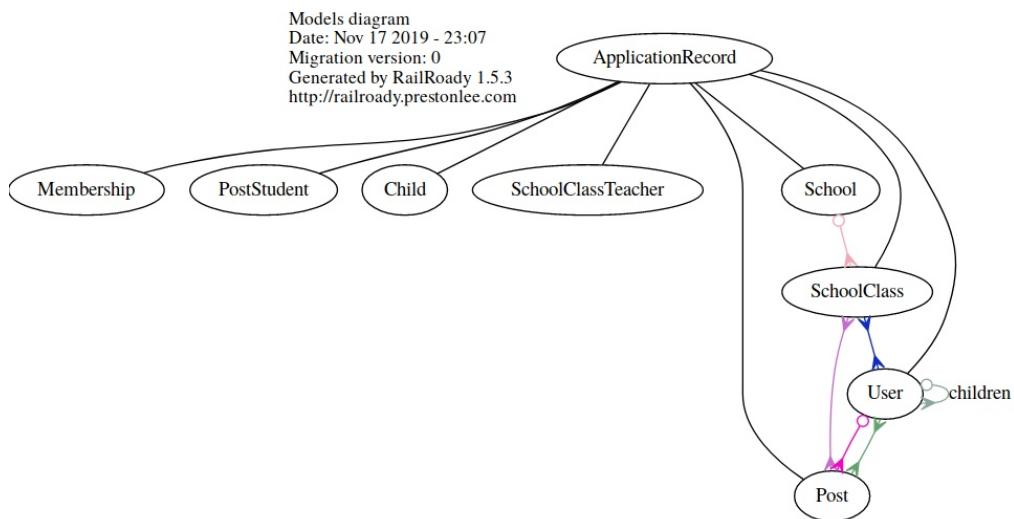


Figura 4.2. Schema UML Models

4.1.3 Architettura delle View

In questa sezione infine è possibile visionare il diagramma UML relativo alle views. (Figura 4.3).

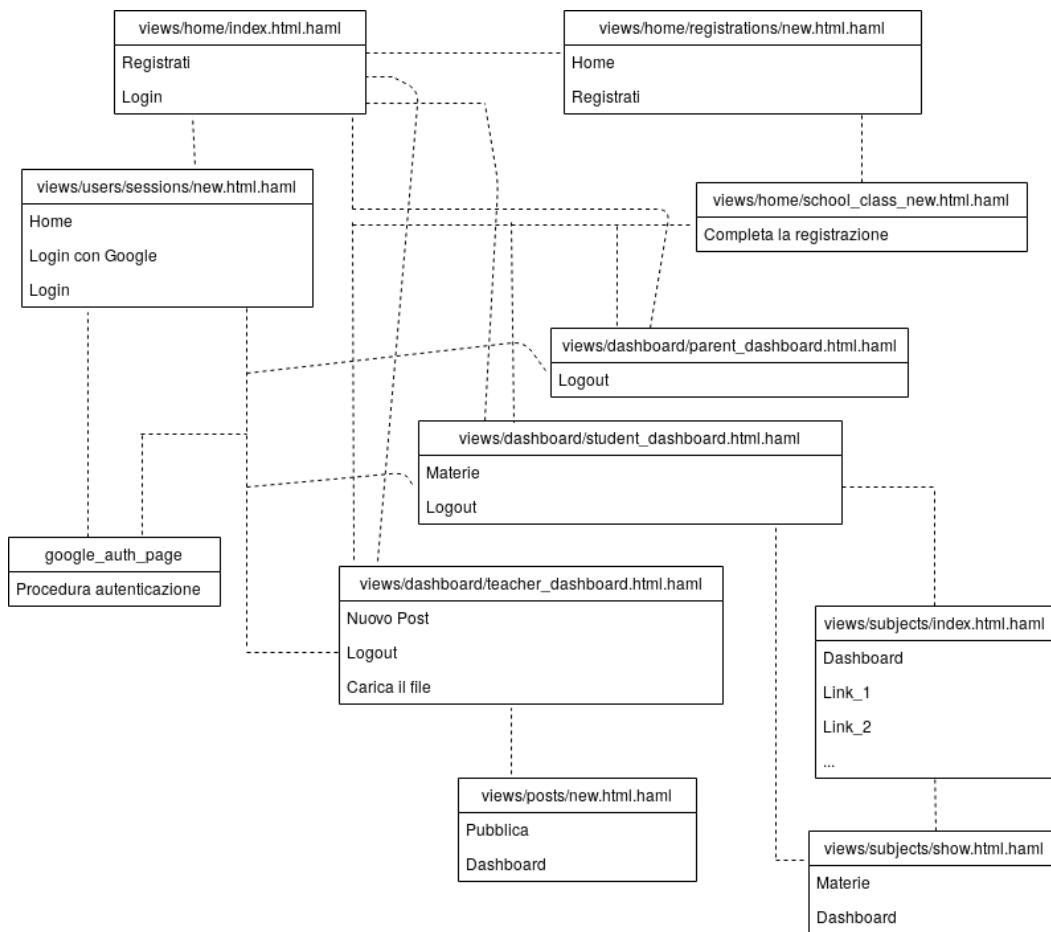


Figura 4.3. Schema UML Views

4.2 Progettazione logica del Data Layer

Come spiegato nella sezione 2.1 Rails è basato sul paradigma Model-View-Controller, quindi nel Data-Layer sono conservati i dati che verranno usati dalle altre componenti dell'applicazione. Tutto ciò è possibile, nell'ambiente di sviluppo, grazie alla gemma *sqlite3*.

Rails offre al programmatore un buon livello di automatizzazione di diversi compiti, tra cui la creazione e la modifica del database: lo schema è conservato nel file `app/db/schema`, il quale viene modificato tramite altri file detti migration, raccolti nella cartella `app/db/migrate`. In questo modo la probabilità di commettere errori umani (distrazione, battitura, ecc.) è minimizzata e si aumenta l'efficienza del lavoro. Di seguito il file `app/db/schema.rb`:

```

1 # This file is auto-generated from the current state of the database. Instead
2 # of editing this file, please use the migrations feature of Active Record to
3 # incrementally modify your database, and then regenerate this schema definition.
4 #
5 # Note that this schema.rb definition is the authoritative source for your
6 # database schema. If you need to create the application database on another
  
```

```
7 # system, you should be using db:schema:load, not running all the migrations
8 # from scratch. The latter is a flawed and unsustainable approach (the more
9 # migrations
10 # you'll amass, the slower it'll run and the greater likelihood for issues).
11 #
12 # It's strongly recommended that you check this file into your version control
13 # system.
14
15 ActiveRecord::Schema.define(version: 20191014161620) do
16
17   create_table "children", force: :cascade do |t|
18     t.integer "parent_id"
19     t.datetime "created_at", null: false
20     t.datetime "updated_at", null: false
21     t.bigint "child_id"
22     t.index ["child_id"], name: "index_children_on_child_id"
23     t.index ["parent_id"], name: "index_children_on_parent_id"
24   end
25
26   create_table "memberships", force: :cascade do |t|
27     t.integer "user_id"
28     t.integer "schoolClass_id"
29     t.datetime "created_at", null: false
30     t.datetime "updated_at", null: false
31     t.index ["schoolClass_id"], name: "index_memberships_on_schoolClass_id"
32     t.index ["user_id"], name: "index_memberships_on_user_id"
33   end
34
35   create_table "posts", force: :cascade do |t|
36     t.string "link"
37     t.date "deadline"
38     t.integer "user_id"
39     t.datetime "created_at", null: false
40     t.datetime "updated_at", null: false
41     t.text "description"
42     t.bigint "school_class_id"
43     t.index ["user_id"], name: "index_posts_on_user_id"
44   end
45
46   create_table "school_classes", force: :cascade do |t|
47     t.string "name"
48     t.integer "school_id"
49     t.datetime "created_at", null: false
50     t.datetime "updated_at", null: false
51     t.index ["school_id"], name: "index_school_classes_on_school_id"
52   end
53
54   create_table "school_classes_teachers", force: :cascade do |t|
55     t.integer "school_class_id", limit: 8
56     t.integer "teacher_id", limit: 8
57     t.string "subject"
58     t.index ["school_class_id"], name:
59       "index_school_classes_teachers_on_school_class_id"
60     t.index ["teacher_id"], name: "index_school_classes_teachers_on_teacher_id"
61   end
62
63   create_table "schools", force: :cascade do |t|
```

```
61   t.string "name"
62   t.datetime "created_at", null: false
63   t.datetime "updated_at", null: false
64 end
65
66 create_table "users", force: :cascade do |t|
67   t.string "email", default: "", null: false
68   t.string "encrypted_password", default: "", null: false
69   t.string "reset_password_token"
70   t.datetime "reset_password_sent_at"
71   t.datetime "remember_created_at"
72   t.datetime "created_at", null: false
73   t.datetime "updated_at", null: false
74   t.integer "roles_mask"
75   t.string "name", default: "", null: false
76   t.string "surname", default: "", null: false
77   t.string "birth_place", default: "", null: false
78   t.date "birth_date", default: "2019-09-04", null: false
79   t.string "sex", default: "", null: false
80   t.string "CF", default: "", null: false
81   t.index ["CF"], name: "index_users_on_CF", unique: true
82   t.index ["email"], name: "index_users_on_email", unique: true
83   t.index ["reset_password_token"], name: "index_users_on_reset_password_token",
84     unique: true
85 end
86 end
```

Listing 4.1. app/db/schema.rb

Da notare gli attributi *index*: essi indicano delle foreign key, cioè riferimenti ad altre tabella.

Capitolo 5

Realizzazione del sistema

5.1 Application Layer

Nell' Application Layer risiede la logica dell'applicazione, implementata attraverso i controller. In particolare, le configurazioni principali sono definite nel file *app/controllers/application_controller.rb* (listing 5.1), che contiene il codice dell'Application Controller. Esso è lo scheletro su cui poggia tutta la struttura, e le caratteristiche qui definite saranno ereditate da tutti gli altri controller.

In generale, i controller si occupano di processare le chiamate fatte dal browser verso l'applicazione, usando i dati contenuti nel Data Layer, esposti dai model, e presentandoli attraverso le view. Grazie al file *config/routes.rb*, le richieste HTTP sono mappate nei corrispondenti metodi definiti nei controller. Nella seguente tabella sono elencate le principali routes dell'applicazione.

Metodo HTTP	URI Pattern	Controller#Action	Effetto
GET	/users/sign_in (.:format)	devise/sessions#new	Restituisce la pagina con la form per il login
POST	/users/sign_in (.:format)	devise/sessions#create	Dopo aver registrato o loggato l'utente restituisce la pagina di default per esso
GET POST	/users/auth /google_oauth2 (.:format)	omniauth_callbacks #passthru	Restituisce la pagina relativa a OAuth di Google
GET POST	/users/auth /google_oauth2 /callback (.:format)	omniauth_callbacks #google_oauth2	Una volta completato il login restituisce la dashboard relativa all'utente

GET	/users/sign_up (:format)	registrations#new	Restituisce la pagina di registrazione dell'utente
POST	/users(:format)	registrations#create	Registra l'utente nel database
GET	/home/index (:format)	home#index	Restituisce la pagina iniziale dell'applicazione
GET	/home /school_class_new (:format)	home#school_class_new	Restituisce la seconda form per completare la registrazione
POST	/home /school_class_new (:format)	home#create	Registra la classe e la scuola di interesse per l'utente e lo reindirizza sulla sua dashboard se la registrazione è andata a buon fine.
GET	/dashboard (:format)	dashboard#get	Restituisce la dashboard dell'utente loggato
GET	/dashboard /delete_file (:format)	dashboard#delete_file	Cancella il file il cui link è stato selezionato
GET	/dashboard /download_file (:format)	dashboard#download_file	Fa il download del file il cui link è stato selezionato
POST	/dashboard (:format)	dashboard#upload_file	Fa l'upload del file il cui link è stato selezionato
GET	/subjects(:format)	subjects#index	Restituisce la pagina dove sono mostrate le materie di interesse
GET	/subjects/:id (:format)	subjects#show	Restituisce la pagina relativa alla singola materia il cui link è stato selezionato
GET	/subjects/:id /download_file (:format)	subjects#view_file	Fa il download del file il cui link è stato selezionato dalla pagina di una singola materia

POST	/posts(:format)	posts#create	Crea un nuovo post e reindirizza l'utente sulla dashboard
GET	/posts/new(:format)	posts#new	Restituisce la pagina con la form per creare un nuovo post
GET	/	home#index	Definisce la pagina iniziale dell'applicazione come root

Nel listing 5.2 è mostrato anche il file *app/controllers/dashboard_controller.rb*, l'altro principale controller su cui si basa l'applicazione.

```

1 class ApplicationController < ActionController::Base
2
3   protect_from_forgery with: :exception
4
5   before_action :configure_devise_permitted_params, if: :devise_controller?
6   before_action :google_set_up
7
8   #after_action :pass_params, if: :devise_controller?
9
10  def google_set_up
11    require 'google/apis/drive_v2'
12
13    scopes = 'https://www.googleapis.com/auth/drive'
14    authorization = Google::Auth.get_application_default(scopes)
15    #Drive = Google::Apis::DriveV2 # Alias the module
16    @drive = Google::Apis::DriveV2::DriveService.new
17    @drive.authorization = authorization
18  end
19
20  def d_file
21    file_id = params[:file_id]
22    @file_name = params[:file_name]
23    @drive.get_file(file_id, download_dest: '/home/biar/Downloads/'+@file_name)
24  end
25
26  protected
27
28  def after_sign_in_path_for(resource)
29    stored_location_for(resource) || dashboard_path
30  end
31
32  def configure_devise_permitted_params
33    registration_params = [ :name, :surname, :CF, :sex, :birth_place, :birth_date,
34                           :email, :password, roles:[] ]
35
36    if params[:action] == 'create'
37      devise_parameter_sanitizer.permit(:sign_up) {
38        |u| u.permit(registration_params)
39      }
40    end
41  end
42
43  def sign_out_path
44    root_path
45  end
46
47  def after_sign_out_path_for(resource)
48    root_path
49  end
50
51  def sign_in_params
52    devise_parameter_sanitizer.permit(:sign_in) do |u|
53      u.permit(:email, :password)
54    end
55  end
56
57  def sign_in_with_http_auth(assoc)
58    user = User.find_by_email(params[:email])
59    if user&.valid_password?(params[:password])
60      sign_in user, scope: assoc
61      sign_in_with_scope user
62    else
63      user.errors.add(:password, I18n.t('errors.messages.password.incorrect'))
64    end
65  end
66
67  def sign_in_with_scope user
68    session[:user_id] = user.id
69    user
70  end
71
72  def sign_in_with_scope user
73    session[:user_id] = user.id
74    user
75  end
76
77  def sign_in_with_scope user
78    session[:user_id] = user.id
79    user
80  end
81
82  def sign_in_with_scope user
83    session[:user_id] = user.id
84    user
85  end
86
87  def sign_in_with_scope user
88    session[:user_id] = user.id
89    user
90  end
91
92  def sign_in_with_scope user
93    session[:user_id] = user.id
94    user
95  end
96
97  def sign_in_with_scope user
98    session[:user_id] = user.id
99    user
100 end
101
102  def sign_in_with_scope user
103    session[:user_id] = user.id
104    user
105  end
106
107  def sign_in_with_scope user
108    session[:user_id] = user.id
109    user
110  end
111
112  def sign_in_with_scope user
113    session[:user_id] = user.id
114    user
115  end
116
117  def sign_in_with_scope user
118    session[:user_id] = user.id
119    user
120  end
121
122  def sign_in_with_scope user
123    session[:user_id] = user.id
124    user
125  end
126
127  def sign_in_with_scope user
128    session[:user_id] = user.id
129    user
130  end
131
132  def sign_in_with_scope user
133    session[:user_id] = user.id
134    user
135  end
136
137  def sign_in_with_scope user
138    session[:user_id] = user.id
139    user
140  end
141
142  def sign_in_with_scope user
143    session[:user_id] = user.id
144    user
145  end
146
147  def sign_in_with_scope user
148    session[:user_id] = user.id
149    user
150  end
151
152  def sign_in_with_scope user
153    session[:user_id] = user.id
154    user
155  end
156
157  def sign_in_with_scope user
158    session[:user_id] = user.id
159    user
160  end
161
162  def sign_in_with_scope user
163    session[:user_id] = user.id
164    user
165  end
166
167  def sign_in_with_scope user
168    session[:user_id] = user.id
169    user
170  end
171
172  def sign_in_with_scope user
173    session[:user_id] = user.id
174    user
175  end
176
177  def sign_in_with_scope user
178    session[:user_id] = user.id
179    user
180  end
181
182  def sign_in_with_scope user
183    session[:user_id] = user.id
184    user
185  end
186
187  def sign_in_with_scope user
188    session[:user_id] = user.id
189    user
190  end
191
192  def sign_in_with_scope user
193    session[:user_id] = user.id
194    user
195  end
196
197  def sign_in_with_scope user
198    session[:user_id] = user.id
199    user
200  end
201
202  def sign_in_with_scope user
203    session[:user_id] = user.id
204    user
205  end
206
207  def sign_in_with_scope user
208    session[:user_id] = user.id
209    user
210  end
211
212  def sign_in_with_scope user
213    session[:user_id] = user.id
214    user
215  end
216
217  def sign_in_with_scope user
218    session[:user_id] = user.id
219    user
220  end
221
222  def sign_in_with_scope user
223    session[:user_id] = user.id
224    user
225  end
226
227  def sign_in_with_scope user
228    session[:user_id] = user.id
229    user
230  end
231
232  def sign_in_with_scope user
233    session[:user_id] = user.id
234    user
235  end
236
237  def sign_in_with_scope user
238    session[:user_id] = user.id
239    user
240  end
241
242  def sign_in_with_scope user
243    session[:user_id] = user.id
244    user
245  end
246
247  def sign_in_with_scope user
248    session[:user_id] = user.id
249    user
250  end
251
252  def sign_in_with_scope user
253    session[:user_id] = user.id
254    user
255  end
256
257  def sign_in_with_scope user
258    session[:user_id] = user.id
259    user
260  end
261
262  def sign_in_with_scope user
263    session[:user_id] = user.id
264    user
265  end
266
267  def sign_in_with_scope user
268    session[:user_id] = user.id
269    user
270  end
271
272  def sign_in_with_scope user
273    session[:user_id] = user.id
274    user
275  end
276
277  def sign_in_with_scope user
278    session[:user_id] = user.id
279    user
280  end
281
282  def sign_in_with_scope user
283    session[:user_id] = user.id
284    user
285  end
286
287  def sign_in_with_scope user
288    session[:user_id] = user.id
289    user
290  end
291
292  def sign_in_with_scope user
293    session[:user_id] = user.id
294    user
295  end
296
297  def sign_in_with_scope user
298    session[:user_id] = user.id
299    user
300  end
301
302  def sign_in_with_scope user
303    session[:user_id] = user.id
304    user
305  end
306
307  def sign_in_with_scope user
308    session[:user_id] = user.id
309    user
310  end
311
312  def sign_in_with_scope user
313    session[:user_id] = user.id
314    user
315  end
316
317  def sign_in_with_scope user
318    session[:user_id] = user.id
319    user
320  end
321
322  def sign_in_with_scope user
323    session[:user_id] = user.id
324    user
325  end
326
327  def sign_in_with_scope user
328    session[:user_id] = user.id
329    user
330  end
331
332  def sign_in_with_scope user
333    session[:user_id] = user.id
334    user
335  end
336
337  def sign_in_with_scope user
338    session[:user_id] = user.id
339    user
340  end
341
342  def sign_in_with_scope user
343    session[:user_id] = user.id
344    user
345  end
346
347  def sign_in_with_scope user
348    session[:user_id] = user.id
349    user
350  end
351
352  def sign_in_with_scope user
353    session[:user_id] = user.id
354    user
355  end
356
357  def sign_in_with_scope user
358    session[:user_id] = user.id
359    user
360  end
361
362  def sign_in_with_scope user
363    session[:user_id] = user.id
364    user
365  end
366
367  def sign_in_with_scope user
368    session[:user_id] = user.id
369    user
370  end
371
372  def sign_in_with_scope user
373    session[:user_id] = user.id
374    user
375  end
376
377  def sign_in_with_scope user
378    session[:user_id] = user.id
379    user
380  end
381
382  def sign_in_with_scope user
383    session[:user_id] = user.id
384    user
385  end
386
387  def sign_in_with_scope user
388    session[:user_id] = user.id
389    user
390  end
391
392  def sign_in_with_scope user
393    session[:user_id] = user.id
394    user
395  end
396
397  def sign_in_with_scope user
398    session[:user_id] = user.id
399    user
400  end
401
402  def sign_in_with_scope user
403    session[:user_id] = user.id
404    user
405  end
406
407  def sign_in_with_scope user
408    session[:user_id] = user.id
409    user
410  end
411
412  def sign_in_with_scope user
413    session[:user_id] = user.id
414    user
415  end
416
417  def sign_in_with_scope user
418    session[:user_id] = user.id
419    user
420  end
421
422  def sign_in_with_scope user
423    session[:user_id] = user.id
424    user
425  end
426
427  def sign_in_with_scope user
428    session[:user_id] = user.id
429    user
430  end
431
432  def sign_in_with_scope user
433    session[:user_id] = user.id
434    user
435  end
436
437  def sign_in_with_scope user
438    session[:user_id] = user.id
439    user
440  end
441
442  def sign_in_with_scope user
443    session[:user_id] = user.id
444    user
445  end
446
447  def sign_in_with_scope user
448    session[:user_id] = user.id
449    user
450  end
451
452  def sign_in_with_scope user
453    session[:user_id] = user.id
454    user
455  end
456
457  def sign_in_with_scope user
458    session[:user_id] = user.id
459    user
460  end
461
462  def sign_in_with_scope user
463    session[:user_id] = user.id
464    user
465  end
466
467  def sign_in_with_scope user
468    session[:user_id] = user.id
469    user
470  end
471
472  def sign_in_with_scope user
473    session[:user_id] = user.id
474    user
475  end
476
477  def sign_in_with_scope user
478    session[:user_id] = user.id
479    user
480  end
481
482  def sign_in_with_scope user
483    session[:user_id] = user.id
484    user
485  end
486
487  def sign_in_with_scope user
488    session[:user_id] = user.id
489    user
490  end
491
492  def sign_in_with_scope user
493    session[:user_id] = user.id
494    user
495  end
496
497  def sign_in_with_scope user
498    session[:user_id] = user.id
499    user
500  end
501
502  def sign_in_with_scope user
503    session[:user_id] = user.id
504    user
505  end
506
507  def sign_in_with_scope user
508    session[:user_id] = user.id
509    user
510  end
511
512  def sign_in_with_scope user
513    session[:user_id] = user.id
514    user
515  end
516
517  def sign_in_with_scope user
518    session[:user_id] = user.id
519    user
520  end
521
522  def sign_in_with_scope user
523    session[:user_id] = user.id
524    user
525  end
526
527  def sign_in_with_scope user
528    session[:user_id] = user.id
529    user
530  end
531
532  def sign_in_with_scope user
533    session[:user_id] = user.id
534    user
535  end
536
537  def sign_in_with_scope user
538    session[:user_id] = user.id
539    user
540  end
541
542  def sign_in_with_scope user
543    session[:user_id] = user.id
544    user
545  end
546
547  def sign_in_with_scope user
548    session[:user_id] = user.id
549    user
550  end
551
552  def sign_in_with_scope user
553    session[:user_id] = user.id
554    user
555  end
556
557  def sign_in_with_scope user
558    session[:user_id] = user.id
559    user
560  end
561
562  def sign_in_with_scope user
563    session[:user_id] = user.id
564    user
565  end
566
567  def sign_in_with_scope user
568    session[:user_id] = user.id
569    user
570  end
571
572  def sign_in_with_scope user
573    session[:user_id] = user.id
574    user
575  end
576
577  def sign_in_with_scope user
578    session[:user_id] = user.id
579    user
580  end
581
582  def sign_in_with_scope user
583    session[:user_id] = user.id
584    user
585  end
586
587  def sign_in_with_scope user
588    session[:user_id] = user.id
589    user
590  end
591
592  def sign_in_with_scope user
593    session[:user_id] = user.id
594    user
595  end
596
597  def sign_in_with_scope user
598    session[:user_id] = user.id
599    user
600  end
601
602  def sign_in_with_scope user
603    session[:user_id] = user.id
604    user
605  end
606
607  def sign_in_with_scope user
608    session[:user_id] = user.id
609    user
610  end
611
612  def sign_in_with_scope user
613    session[:user_id] = user.id
614    user
615  end
616
617  def sign_in_with_scope user
618    session[:user_id] = user.id
619    user
620  end
621
622  def sign_in_with_scope user
623    session[:user_id] = user.id
624    user
625  end
626
627  def sign_in_with_scope user
628    session[:user_id] = user.id
629    user
630  end
631
632  def sign_in_with_scope user
633    session[:user_id] = user.id
634    user
635  end
636
637  def sign_in_with_scope user
638    session[:user_id] = user.id
639    user
640  end
641
642  def sign_in_with_scope user
643    session[:user_id] = user.id
644    user
645  end
646
647  def sign_in_with_scope user
648    session[:user_id] = user.id
649    user
650  end
651
652  def sign_in_with_scope user
653    session[:user_id] = user.id
654    user
655  end
656
657  def sign_in_with_scope user
658    session[:user_id] = user.id
659    user
660  end
661
662  def sign_in_with_scope user
663    session[:user_id] = user.id
664    user
665  end
666
667  def sign_in_with_scope user
668    session[:user_id] = user.id
669    user
670  end
671
672  def sign_in_with_scope user
673    session[:user_id] = user.id
674    user
675  end
676
677  def sign_in_with_scope user
678    session[:user_id] = user.id
679    user
680  end
681
682  def sign_in_with_scope user
683    session[:user_id] = user.id
684    user
685  end
686
687  def sign_in_with_scope user
688    session[:user_id] = user.id
689    user
690  end
691
692  def sign_in_with_scope user
693    session[:user_id] = user.id
694    user
695  end
696
697  def sign_in_with_scope user
698    session[:user_id] = user.id
699    user
700  end
701
702  def sign_in_with_scope user
703    session[:user_id] = user.id
704    user
705  end
706
707  def sign_in_with_scope user
708    session[:user_id] = user.id
709    user
710  end
711
712  def sign_in_with_scope user
713    session[:user_id] = user.id
714    user
715  end
716
717  def sign_in_with_scope user
718    session[:user_id] = user.id
719    user
720  end
721
722  def sign_in_with_scope user
723    session[:user_id] = user.id
724    user
725  end
726
727  def sign_in_with_scope user
728    session[:user_id] = user.id
729    user
730  end
731
732  def sign_in_with_scope user
733    session[:user_id] = user.id
734    user
735  end
736
737  def sign_in_with_scope user
738    session[:user_id] = user.id
739    user
740  end
741
742  def sign_in_with_scope user
743    session[:user_id] = user.id
744    user
745  end
746
747  def sign_in_with_scope user
748    session[:user_id] = user.id
749    user
750  end
751
752  def sign_in_with_scope user
753    session[:user_id] = user.id
754    user
755  end
756
757  def sign_in_with_scope user
758    session[:user_id] = user.id
759    user
760  end
761
762  def sign_in_with_scope user
763    session[:user_id] = user.id
764    user
765  end
766
767  def sign_in_with_scope user
768    session[:user_id] = user.id
769    user
770  end
771
772  def sign_in_with_scope user
773    session[:user_id] = user.id
774    user
775  end
776
777  def sign_in_with_scope user
778    session[:user_id] = user.id
779    user
780  end
781
782  def sign_in_with_scope user
783    session[:user_id] = user.id
784    user
785  end
786
787  def sign_in_with_scope user
788    session[:user_id] = user.id
789    user
790  end
791
792  def sign_in_with_scope user
793    session[:user_id] = user.id
794    user
795  end
796
797  def sign_in_with_scope user
798    session[:user_id] = user.id
799    user
800  end
801
802  def sign_in_with_scope user
803    session[:user_id] = user.id
804    user
805  end
806
807  def sign_in_with_scope user
808    session[:user_id] = user.id
809    user
810  end
811
812  def sign_in_with_scope user
813    session[:user_id] = user.id
814    user
815  end
816
817  def sign_in_with_scope user
818    session[:user_id] = user.id
819    user
820  end
821
822  def sign_in_with_scope user
823    session[:user_id] = user.id
824    user
825  end
826
827  def sign_in_with_scope user
828    session[:user_id] = user.id
829    user
830  end
831
832  def sign_in_with_scope user
833    session[:user_id] = user.id
834    user
835  end
836
837  def sign_in_with_scope user
838    session[:user_id] = user.id
839    user
840  end
841
842  def sign_in_with_scope user
843    session[:user_id] = user.id
844    user
845  end
846
847  def sign_in_with_scope user
848    session[:user_id] = user.id
849    user
850  end
851
852  def sign_in_with_scope user
853    session[:user_id] = user.id
854    user
855  end
856
857  def sign_in_with_scope user
858    session[:user_id] = user.id
859    user
860  end
861
862  def sign_in_with_scope user
863    session[:user_id] = user.id
864    user
865  end
866
867  def sign_in_with_scope user
868    session[:user_id] = user.id
869    user
870  end
871
872  def sign_in_with_scope user
873    session[:user_id] = user.id
874    user
875  end
876
877  def sign_in_with_scope user
878    session[:user_id] = user.id
879    user
880  end
881
882  def sign_in_with_scope user
883    session[:user_id] = user.id
884    user
885  end
886
887  def sign_in_with_scope user
888    session[:user_id] = user.id
889    user
890  end
891
892  def sign_in_with_scope user
893    session[:user_id] = user.id
894    user
895  end
896
897  def sign_in_with_scope user
898    session[:user_id] = user.id
899    user
900  end
901
902  def sign_in_with_scope user
903    session[:user_id] = user.id
904    user
905  end
906
907  def sign_in_with_scope user
908    session[:user_id] = user.id
909    user
910  end
911
912  def sign_in_with_scope user
913    session[:user_id] = user.id
914    user
915  end
916
917  def sign_in_with_scope user
918    session[:user_id] = user.id
919    user
920  end
921
922  def sign_in_with_scope user
923    session[:user_id] = user.id
924    user
925  end
926
927  def sign_in_with_scope user
928    session[:user_id] = user.id
929    user
930  end
931
932  def sign_in_with_scope user
933    session[:user_id] = user.id
934    user
935  end
936
937  def sign_in_with_scope user
938    session[:user_id] = user.id
939    user
940  end
941
942  def sign_in_with_scope user
943    session[:user_id] = user.id
944    user
945  end
946
947  def sign_in_with_scope user
948    session[:user_id] = user.id
949    user
950  end
951
952  def sign_in_with_scope user
953    session[:user_id] = user.id
954    user
955  end
956
957  def sign_in_with_scope user
958    session[:user_id] = user.id
959    user
960  end
961
962  def sign_in_with_scope user
963    session[:user_id] = user.id
964    user
965  end
966
967  def sign_in_with_scope user
968    session[:user_id] = user.id
969    user
970  end
971
972  def sign_in_with_scope user
973    session[:user_id] = user.id
974    user
975  end
976
977  def sign_in_with_scope user
978    session[:user_id] = user.id
979    user
980  end
981
982  def sign_in_with_scope user
983    session[:user_id] = user.id
984    user
985  end
986
987  def sign_in_with_scope user
988    session[:user_id] = user.id
989    user
990  end
991
992  def sign_in_with_scope user
993    session[:user_id] = user.id
994    user
995  end
996
997  def sign_in_with_scope user
998    session[:user_id] = user.id
999    user
1000 end

```

```

38     }
39   end
40 end
41
42 #def pass_params
43   # @codice_fiscale = params[:CF]
44   #@ruolo = params[:roles_mask]
45 #end
46
47 end

```

Listing 5.1. app/controllers/application_controller.rb

Si noti il codice necessario per far funzionare correttamente Devise, i servizi Google Drive e il protocollo OAuth.

```

1 class DashboardController < ApplicationController
2
3 # ci arrivo anche dai login, quindi mi conviene fare la query
4
5 def get
6   if !user_signed_in?
7     redirect_to user_session_path and return
8   end
9   @c_user = current_user
10  if @c_user.blank?
11    redirect_to user_session_path
12  elsif @c_user.roles_mask == 1
13    son_id = Child.find_by(parent_id: @c_user.id).child_id
14    @son = User.find(son_id)
15    son_class_id = Membership.find_by(user_id: @son.id).schoolClass_id
16    @posts = Post.where(school_class_id: son_class_id).limit(10)
17    render 'parent_dashboard'
18  elsif @c_user.roles_mask == 2
19    class_and_school = Membership.find_by(user_id: @c_user.id).schoolClass_id
20    @school_class = SchoolClass.find(class_and_school)
21    @school = School.find(@school_class.school_id)
22    @posts = Post.where(school_class_id: @school_class.id).limit(10)
23    render 'student_dashboard'
24  else
25
26    #metadata = Google::Apis::DriveV2::File.new(title: 'My document')
27    #metadata = drive.insert_file(metadata, upload_source:
28    #  'app/assets/files/test.txt', content_type: 'text/plain')
29
30    # Search for files in Drive (first page only)
31    @files = @drive.list_files
32    @files.items.each do |file|
33      #puts file.title
34    end
35
36    @posts = Post.joins("INNER JOIN school_classes_teachers ON
37      school_classes_teachers.school_class_id =
38      posts.school_class_id").where("school_classes_teachers.teacher_id"
39      => @c_user.id).limit(10)
40    render 'teacher_dashboard'

```

```

37     end
38   end
39
40   def upload_file
41     @new_file_name = params[:g_file_title]
42     @new_file_source = params[:g_file_path]
43
44     metadata = Google::Apis::DriveV2::File.new(title: @new_file_name)
45     metadata = @drive.insert_file(metadata, upload_source:
46       "#{@new_file_source}", content_type: 'text/plain')
47     flash[:notice] = "Caricamento effettuato!"
48     redirect_to '/dashboard'
49
50   end
51
52   def delete_file
53     file_id = params[:file]
54     @drive.delete_file(file_id)
55     flash[:notice] = "Il file è stato cancellato con successo!"
56     redirect_to '/dashboard'
57   end
58
59   def download_file
60     d_file
61     flash[:notice] = "Download effettuato!"
62     redirect_to '/dashboard'
63   end
64 end

```

Listing 5.2. app/controllers/dashboard_controller.rb

Nel Dashboard controller è invece possibile vedere il codice che si occupa di processare i principali contenuti della dashboard: come prima cosa si controlla che l'utente si sia autenticato, altrimenti viene rimandato alla pagina di login. Successivamente si recuperano le principali informazioni su di esso, sui file e sui post.

5.2 Presentation Layer

Il Presentation Layer è costituito dall'insieme delle view che vengono presentate all'utente. Esse sono state realizzate mediante il linguaggio HAML, e si trovano nella cartella *app/views*. Per lo styling sono stati usati i file SCSS definiti nella cartella *app/assets/stylesheets*. Nella cartella *app/assets/images* si trovano le immagini rese nel browser (in questo caso il logo HiSchool).

Nel rispetto del principio Don't Repeat Yourself, sono stati creati i file *app/views/layouts/application.html.haml* (Listing 5.3) e *app/assets/stylesheets/application.css.scss* (Listing 5.4, che essendo ereditati dall'intera applicazione forniscono lo styling di base generale).

Inoltre si è fatto spesso uso di partials.

```

1  !!!
2  %html
3    %head

```

```

4      %meta{:content => "text/html; charset=UTF-8", "http-equiv" =>
5          "Content-Type"}/
6      %meta{:content => "width=device-width", :name => "viewport"}/
7      %meta{:content => "height=device-height", :name => "viewport"
8          }/
9      %title HiSchool
10     = csrf_meta_tags
11     = stylesheet_link_tag    'application', media: 'all', 'data-
12         turbolinks-track': 'reload'
13     = javascript_include_tag 'application', 'data-turbolinks-track
14         ': 'reload'
15     %body
16         #main-container
17             .nav.navbar-default.navbar-fixed-top
18                 .container
19                     .navbar-header
20                         = image_tag("HiSchoolLogo.png")
21                     %ul.nav.navbar-nav.navbar-right
22                         = yield :nav_top
23                     .container-fluid
24                         .row
25                             .col-xs-12.col-md-4.center-block
26                             = yield :left_col
27                             .col-xs-12.col-md-4.center-block
28                             = yield :center_col
29                             .col-xs-12.col-md-4.center-block
30                             = yield :right_col
31             .nav.navbar-default.navbar-fixed-bottom
32                 .container
33                     %ul.nav.navbar-nav.navbar-left
34                         = yield :nav_bottom

```

Listing 5.3. app/views/layouts/application.html.haml

```

1  /*
2   * This is a manifest file that'll be compiled into
3   * application.css, which will include all the files
4   * listed below.
5   *
6   * Any CSS and SCSS file within this directory, lib/assets/
7   * stylesheets, or any plugin's
8   * vendor/assets/stylesheets directory can be referenced here
9   * using a relative path.
10  *
11  * You're free to add application-wide styles to this file and
12  * they'll appear at the bottom of the
13  * compiled file so the styles you add here take precedence over
14  * styles defined in any other CSS/SCSS
15  * files in this directory. Styles in this file should be added
16  * after the last require_* statement.
17  * It is generally better to create a new file per style scope.
18  */
19
20 *= require simple_calendar
21 *= require_tree .
22
23
24 @import "bootstrap-sprockets";
25 @import "bootstrap";

```

```
19
20   label{
21     font-size: medium;
22   }
23
24   .my-label{
25     font-weight: 400;
26   }
27
28   .my-container{
29     height: 400px;    // all'occorrenza si può aumentare
30     width: 450px;
31     border-color: aquamarine;
32     overflow: auto;
33   }
34
35   .google-btn{
36     margin-top: 100px;
37     margin-left: 100px;
38   }
39
40   select{
41     background-color: #d6fffe1;
42   }
43
44   #main-container{
45     max-width: 1200px;
46     margin: 0 auto;
47     padding: 10px;
48   }
49
50   .navbar-default{
51     padding-top:10px;
52     padding-bottom: 10px;
53     background-color: #d6fffe1;
54   }
55
56   #main-container .navbar-default {
57     min-height: 100px;
58   }
59
60   .container-fluid{
61     padding-top: 110px;
62     padding-bottom: 80px;
63   }
64
65   @media screen and (max-width: 768px){
66
67     *[class*='col-xs']{
68       padding:50px;
69     }
70
71   }
```

Listing 5.4. app/assets/stylesheets/application.css.scss

5.2.1 Aspetto dell'applicazione

In questa sezione verranno presentate le view dell'applicazione, le dimensioni sono state riadattate per una migliore impaginazione. Nella figura 5.1 si può vedere la schermata che accoglie l'utente avviata la piattaforma:



Figura 5.1. app/views/home/index.html.haml

Una volta premuto il bottone **Registrati** ci si ritrova nella pagina di registrazione, mostrata nella figura 5.2.

The screenshot shows a web browser window with the URL 'localhost:3000/users/sign_up'. The title bar says 'HiSchool'. The main content area has a green header with the 'HiSchool' logo. On the left, there is a section titled 'Benvenuto nella pagina di registrazione!' with instructions for using the service. On the right, there is a form titled 'Registrazione:' with fields for 'Nome' (Name), 'Cognome' (Surname), 'Codice fiscale' (Tax code), 'Data di nascita' (Date of birth), 'Città di nascita' (City of birth), 'Sesso' (Gender) with options 'M' and 'F', 'Sei un' (You are) with options 'Genitore', 'Studente', or 'Insegnante', 'Email' (Email), 'Password' (Password), and 'Conferma password' (Confirm password). A 'Registrati' button is at the bottom right.

Figura 5.2. app/views/home/registrations/new.html.haml

A seconda del ruolo selezionato, se tutti i campi della form sono stati compilati correttamente, premendo il bottone **Registrati** si arriva a pagine dall'aspetto diverso. Esse sono però definite tutte nello stesso file di codice: *app/views/home/-*

`school_class_new.html.haml`. Se viene selezionato il ruolo di *Genitore* si arriverà alla pagina in figura 5.3, se si è selezionato *Studente* a quella in figura 5.4, e infine se si è selezionato *Insegnante* ci si ritroverà una schermata come quella in figura 5.5.

The screenshot shows a web browser window with the URL `localhost:3000/home/school_class_new/` in the address bar. The page has a light green header with the **HiSchool** logo. The main content area contains two columns. On the left, there are two warning messages: "Hai quasi finito!" and "Attenzione!". "Hai quasi finito!" instructs the user to complete the registration by filling in the tax code field. "Attenzione!" warns that entering an invalid tax code will result in being redirected to this page. On the right, there is a text input field labeled "Genitore di (inserisci il codice fiscale di tuo figlio)" and a green "Completa la registrazione" button.

Figura 5.3. `app/views/home/school_class_new.html.haml`

The screenshot shows a web browser window with the URL `localhost:3000/home/school_class_new/` in the address bar. The page has a light green header with the **HiSchool** logo. The main content area contains two columns. On the left, there are two warning messages: "Hai quasi finito!" and "Attenzione!". "Hai quasi finito!" instructs the user to enter the class and school information. "Attenzione!" warns that entering an invalid class or school will result in being redirected to this page. On the right, there are three input fields: "Classe" (with placeholder "Inserisci la classe e la scuola che frequenti."), "Scuola" (empty), and a green "Completa la registrazione" button.

Figura 5.4. `app/views/home/school_class_new.html.haml`

The screenshot shows a web browser window with the URL `localhost:3000/home/school_class_new/` in the address bar. The page has a light green header with the **HiSchool** logo. The main content area contains two columns. On the left, there are two warning messages: "Hai quasi finito!" and "Attenzione!". "Hai quasi finito!" instructs the user to enter the class, school, and subject information. "Attenzione!" warns that entering an invalid class or school will result in being redirected to this page. On the right, there are three input fields: "Classe" (with placeholder "Inserisci la classe, la scuola e la materia che insegni."), "Scuola" (empty), and "Materia" (empty). Below these fields is a green "Completa la registrazione" button.

Figura 5.5. `app/views/home/school_class_new.html.haml`

In tutti e tre i casi, una volta compilati gli ultimi campi e premuto il bottone **Completa la registrazione**, l'utente risulta registrato e verrà reindirizzato alla sua dashboard. Anch'essa è differente a seconda dei ruoli, ed è ottenuta combinando diverse partial (`app/views/dashboard/_common.html.haml` e `app/views/dashboard/_posts.html.haml`) in diversi file. Nella figura 5.6 è mostrata quella per il Genitore, nella 5.7 quella per lo Studente e nella 5.10 quella per l'Insegnante.

Si noti come in tutte le dashboard sono presenti i dati fondamentali dell'utente, il calendario (dove è segnato in verde acceso il giorno corrente e in giallo le scadenze delle comunicazioni) e i post relativi alla classe di interesse. In tutte è inoltre presente il bottone **Logout** per tornare alla pagina iniziale e terminare la sessione.

The screenshot shows the HiSchool dashboard for a Parent (Genitore). At the top, there's a header with the HiSchool logo and a 'Logout' button. Below the header, a user profile section displays the following information:

- Nome:** Odette
- Cognome:** Cigno
- Email:** odette@gmail.com
- Genitore di:** Clara Schiaccianoci

Below the profile, there's a calendar for November 2019. The 30th is highlighted in yellow with the text "[Italiano] Altro post di prova. Deadline il 30!". The current date, November 4th, is highlighted in green. To the right of the calendar, a sidebar lists posts from students:

- [Italiano, 2LS, Peano] Studiate tanto! Post 1 (Silvia) - 2019-09-22 07:06:01 UTC
- [Storia, 2LS, Peano] Post2, Pinco Pallo! - 2019-09-24 16:01:47 UTC
- [Storia, Post con deadline] Pinco Pallo - 2019-09-30 16:29:53 UTC
Link per la consegna
link1
Entro:
2019-10-02

Figura 5.6. `app/views/dashboard/parent_dashboard.html.haml`

The screenshot shows the HiSchool dashboard for a Student (Studente). The layout is similar to Figure 5.6, featuring a header with 'HiSchool', a 'Logout' button, and a user profile section. The user profile information is as follows:

- Nome:** Odette
- Cognome:** Cigno
- Email:** odette@gmail.com
- Classe:** 2LS
- Scuola:** Peano

A calendar for November 2019 is displayed, with the 30th highlighted in yellow and the current date, November 4th, highlighted in green. A sidebar on the right shows student posts:

- [Italiano, 2LS, Peano] Studiate tanto! Post 1 (Silvia) - 2019-09-22 07:06:01 UTC
- [Storia, 2LS, Peano] Post2, Pinco Pallo! - 2019-09-24 16:01:47 UTC
- [Storia, Post con deadline] Pinco Pallo - 2019-09-30 16:29:53 UTC
Link per la consegna
link1
Entro:
2019-10-02

Figura 5.7. `app/views/dashboard/student_dashboard.html.haml`

Nella pagina della Studente è presente anche il bottone **Materie**. Cliccandolo si potrà accedere ad un'altra schermata dove verranno visualizzati i link alle pagine corrispondenti alle materie insegnate dai professori registrati. (Figura 5.8).

The screenshot shows a web browser window with the URL 'localhost:3000/subjects'. The page has a green header with the 'HiSchool' logo. Below the header, there are two sections: 'Materie' on the left and 'Insegnanti' on the right. Under 'Materie', there are links for 'Italiano' and 'Storia'. Under 'Insegnanti', there are links for 'Silvia del Piano' and 'Pinco Pallo'. A 'Dashboard' button is located in the top right corner.

Figura 5.8. app/views/subjects/index.html.haml

Se si clicca sul bottone **Dashboard** si ritorna nell'omonima pagina, mentre se si segue uno dei link riguardanti le materie, per esempio *Italiano*, ci si ritrova alla pagina dedicata, mostrata in figura 5.9. In essa si potrà trovare il materiale presente su Google Drive ed i post riguardanti solo quella specifica disciplina.

The screenshot shows a web browser window with the URL 'localhost:3000/subjects/4'. The page has a green header with the 'HiSchool' logo. Below the header, there are two sections: 'Italiano' on the left and a main content area on the right. The 'Italiano' section contains information about the teacher: Nome prof: Silvia, Cognome prof: del Piano, Email prof: silvia.delpiano27@gmail.com. It also lists 'Materiale:' with links to files like 'Italiano_test_cancellazione1', 'Italiano_test.txt', etc. A message at the bottom says 'Signed in successfully.' The main content area shows three posts from the teacher: '[Italiano, 2LS, Peano] Studiate tanto! Post 1 (Silvia)' (2019-09-22 07:06:01 UTC), '[Italiano] Post di prova. Ciao allunni!' (2019-10-20 07:02:56 UTC), and '[Italiano] Altro post di prova. Deadline il 30!' (2019-10-20 07:06:09 UTC). There are 'Materie' and 'Dashboard' buttons in the top right corner.

Figura 5.9. app/views/subjects/show.html.haml

Cliccando sul bottone **Materie** si ritorna alla pagina in figura 5.8.

Nella dashboard dell'Insegnante è presente, sulla sinistra, anche il materiale caricato su Google Drive per gli studenti, con i relativi link per la cancellazione ed il download. In basso a destra invece vi è la form per farne l'upload. Seguendo inoltre il bottone **Nuovo Post** si avrà a disposizione la pagina con la form che permetterà pubblicare, per l'appunto, un nuovo post. (Figura 5.11).

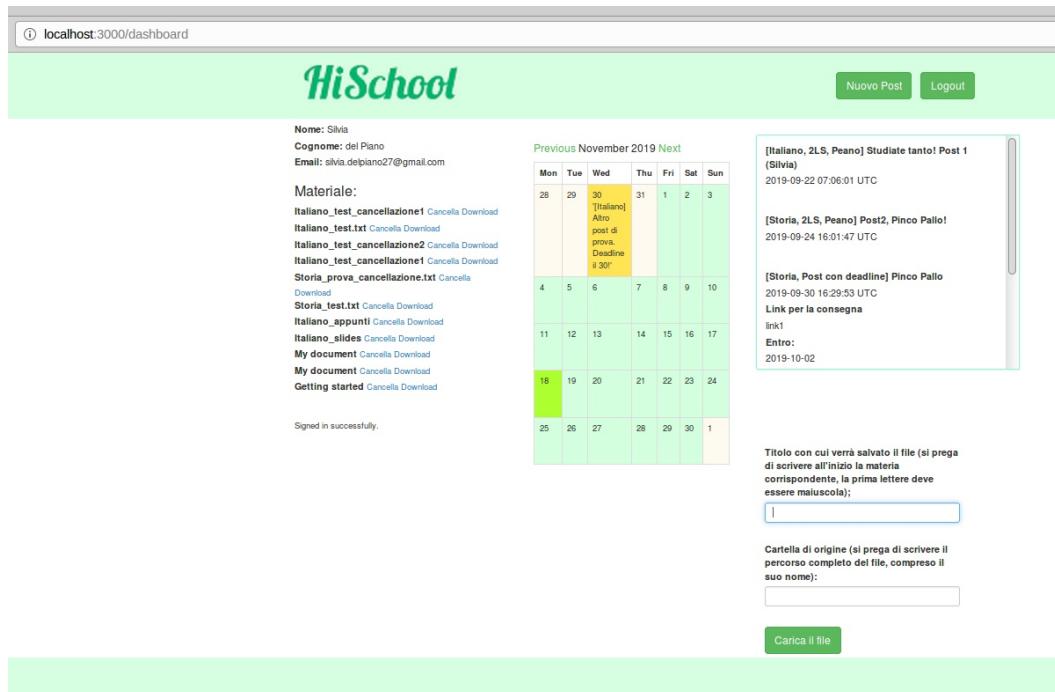


Figura 5.10. app/views/dashboard/teacher_dashboard.html.haml

Cliccando il tasto **Pubblica**, previa adeguata compilazione dei campi, verrà creata e pubblicata la comunicazione, e si sarà reindirizzati alla dashboard.

The screenshot shows the HiSchool post creation form at localhost:3000/posts/new. At the top, there's a header with the HiSchool logo and a 'Dashboard' button. The main area contains several input fields:

- Testo**: A large text input field containing a single character 'I'.
- Classe**: A text input field.
- Scuola**: A text input field.
- Link consegna (facoltativo)**: A text input field.
- Scadenza (facoltativo)**: A text input field containing 'gg / mm / aaaa'.

At the bottom, there's a green 'Pubblica' button.

Figura 5.11. app/views/posts/new.html.haml

Infine, cliccando il bottone **Login** dalla pagina iniziale (figura 5.1) si andrà a finire nell'omonima pagina (figura 5.12). Si può scegliere se compilare i campi ed

accedere con le credenziali locali (quelle impostate al momento della registrazione) oppure se accedere tramite Google sfruttando i servizi OAuth, venendo reindirizzati alla pagina corrispondente (figura 5.13). A login effettuato ci si ritroverà sulla dashboard corrispondente al proprio ruolo.

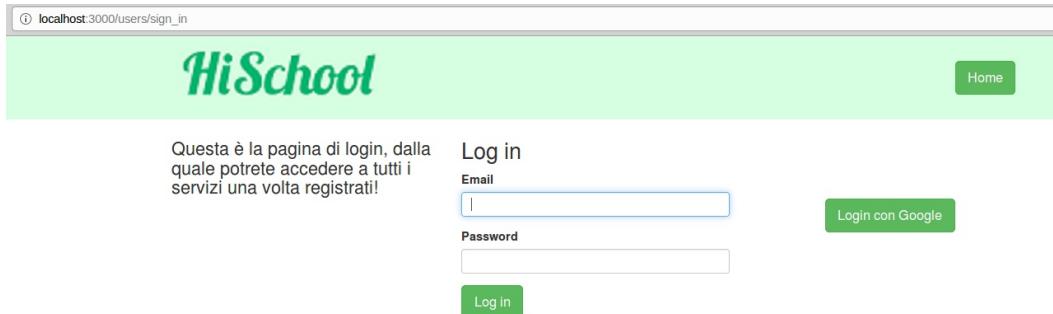


Figura 5.12. app/views/users/sessions/new.html.haml

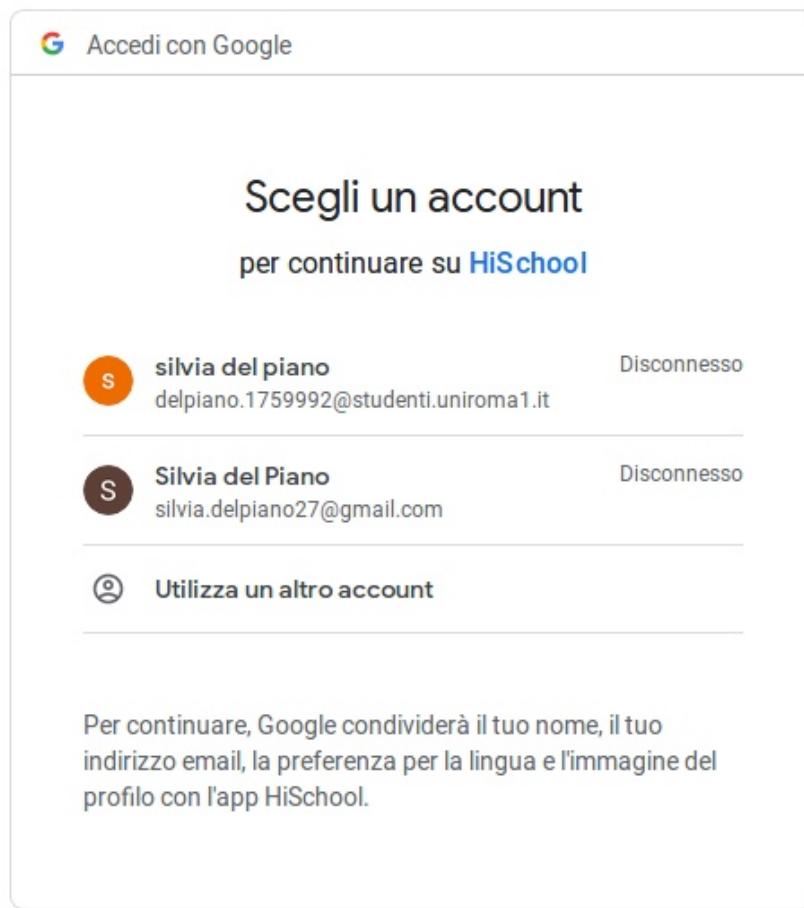
[Italiano ▾](#)[Guida](#)[Privacy](#)[Termini](#)

Figura 5.13. Login con Google

Capitolo 6

Dispiegamento e Validazione

6.1 Installazione

Per installare l'applicazione è necessario posizionarsi nella directory di lavoro ed eseguire i seguenti comandi:

1. `git clone https://github.com/SilviadelPiano/HiSchool.git`
2. `bundle install`
3. `rake db:migrate`

A questo punto perchè l'applicazione funzioni occorre registrare almeno una scuola e una classe, si apra quindi la console di Rails con il comando

- `rails console`

e si registrino i dati necessari con questa utilizzando questa sintassi:

- `> School.create(name: "nome_scuola")`
- `> SchoolClass.create(name: "nome_classe", school_id: "id_scuola (appena ritornato dalla console)")`

Se le transazioni sono andate a buon fine si può far partire l'applicazione con il comando

- `rails server`

6.2 Test

Metodo Agile -> Test con cucumber e capybara (mostra i file più importanti) + file rspec con oauth. brakeman e gembulance per i test di sicurezza. byebug per il debugging durante lo sviluppo