UNIVERSITÀ
DI TRENTO

Dipartimento di Ingegneria e Scienza dell'Informazione

Corso di Laurea in
Informatica

Elaborato finale

Titolo

Sottotitolo (alcune volte lungo - opzionale)

Supervisore
Prof. Paolo Giorgini
Prof. Alessandro Tomasi

Laureando
Stefano Dal Mas

Anno accademico 2022/2023

Ringraziamenti

OpenAI

Indice

Sommario

Sommario è un breve riassunto del lavoro svolto dove si descrive l'obiettivo, l'oggetto della tesi, le metodologie e le tecniche usate, i dati elaborati e la spiegazione delle conclusioni alle quali siete arrivati.

Il sommario dell'elaborato consiste al massimo di 3 pagine e deve contenere le seguenti informazioni:

- contesto e motivazioni
- breve riassunto del problema affrontato
- tecniche utilizzate e/o sviluppate
- risultati raggiunti, sottolineando il contributo personale del laureando/a

1 Obbiettivi del progetto

Il progetto ha come obbiettivo la realizzazione di una WebApp sottoforma di modulo da poter aggiungere alla piattaforma DISI Industry. L'applicativo deve permettere alle aziende, enti esterni all'università, di poter formalizzare e proporre delle "Challenge". Esse possono essere definite come gare a lungo termine ove viene richiesto ai gruppi partecipanti di effettuare sia la fase di progettazione che eventualmente la fase implementativa di una soluzione rispetto ad un problema definito, oppure può essere proposta sottoforma di hackaton, ossia una competizione volte alla risoluzione di un problema in un lasso di tempo limitato mettendo in palio un premio per i team vincitori, come nel caso delle Reply Challenge, hackaton nelle quali ai gruppi viene richiesto di risolvere dei problemi di algoritmica e di programmazione.

Questo progetto nasce dalla mia partecipazione al Samsung Innovation Camp, un evento proposto da Samsung in collaborazione con Randstad, nel quale veniva richiesto agli studenti di risolvere una problematica proposta dall'azienda, differenziandola sulla base dell'ateneo scelto, proponendo soluzioni che sfruttassero le nuove tecnologie quali Internet of Things, Intelligenza Artificiale e tutte le nuove tecnologie correlate. La problematica proposta all'ateneo di Trento era inerente alla valorizzazione del patrimonio naturalistico, artistico e culturale mediante l'innovazione digitale, con l'obbiettivo di sostenere il settore della Cultura e del Turismo.

La partecipazione a tale progetto mi ha permesso di conoscere nuove tecnologie, meccaniche aziendali e di lavorare in un team di sviluppo, permettendomi di migliorare le mie capacità di problem solving, di lavoro in team e di gestione del tempo, e mi ha fatto realizzare quanto sia importante avere una piattaforma nella quale gli studenti possano in modo semplice e veloce trovare tutte le informazioni necessarie per partecipare alle Challenge proposte dalle aziende.

1.1 DISI Industry

La scelta dell'integrazione di questo modulo all'interno di una WebApp già esistente è sorta dal fatto che DISI industry nasce già con l'idea di connettere gli studenti con l'industria, in quanto i primi possono cercare delle offerte di lavoro comparabili con le loro competenze apprese durante gli studi, mentre le compagnie possono offrire ruoli di lavoro per trovare il personale più adatto alle loro richieste. Essa può essere dunque definita come una piattaforma di recruiting online per gli studenti del DISI. [3]

Mi è dunque venuta naturale l'idea di implementare il modulo all'interno della WebApp per l'affinità intrinseca tra la mia ideologia e quella proposta dalla piattaforma già esistente, così da permettere agli studenti non solo di poter cercare potenziali offerte di tirocinio o di lavoro, ma anche di poter partecipare a delle Challenge proposte dalle aziende, così da poter mettere in pratica le proprie conoscenze e competenze e di poterle migliorare, oltre che di poter vincere dei premi, il tutto mediante la stessa piattaforma per ridurre al minimo la fatica dovuta alla navigazione di molteplici pagine web e portali.

2 Tecnologie utilizzate

In questo capitolo vengono presentate le tecnologie utilizzate e le motivazioni dietro tali scelte. In particolare, vengono presentate le tecnologie utilizzate per la realizzazione del back-end, del front-end e del database.

2.1 Django

Il framework utilizzato è Django, esso è stato scelto per la sua semplicità di utilizzo e per la sua flessibilità. Django è un framework open-source di alto-livello per lo sviluppo di applicazioni web scritto in Python. Esso è basato sul pattern architetturale Model-View-Template (MVT) e permette di sviluppare applicazioni web in modo rapido e pulito, fornendo un'interfaccia di amministrazione per la gestione dei dat, così da non dover implementare un admin zone mediante un altro linguaggio, ma mantenendo unificata la codebase.

Un altro vantaggio risiede nella scalabilità del framework, il quale permette la creazione di moduli che possono essere inseriti o rimossi con l'aggiunta di poche righe di codice all'interno di un file apposito.

Data la natura del progetto, è stato necessario avere un framework che permettesse la creazione, validazione e modifica dei form, difatti Django offre un componente apposito per la gestione dinamica dei form dalle View, senza dover passare per il codice html evitando dunque inconsistenza in tutta la codebase. Per la visualizzazione dei form dinamicamente e con dello stile unificato viene utilizzato il modulo Crispy Forms per permettere di controllare come il form venga renderizzato dall'utente finale. [2]

2.1.1 Architettura MVT

Il vantaggio dell'utilizzo di un framework basato su architettura MVT permette ad ogni livello, ossia Model, View e Template, di lavorare indipendentemente, il che permette una maggiore indipendenza tra le parti. Esso è conveniente nel caso si decida di scalare l'applicativo web, in quanto permette di aggiungere nuove funzionalità senza dover modificare il codice già esistente. Inoltre, un'architettura MVT permette di rendere più veloce e più seplice la trasmissione di dati via internet.

Vengono presentati in dettaglio le componenti dell'architettura MVT:

- Model : questo componente aiuta la gestione del database ed esso è un livello di accesso dati che contiene i campi ed il comportamento dei dati che stanno venendo gestiti. Esso utilizza un pattern ORM (Object-Relational Mapping) che mappa i campi del database con gli attributi della classe Python. Questo permette di utilizzare un linguaggio di programmazione orientato agli oggetti come Python per la gestione dei dati, permette un livello di astrazione tale da non rendere obbligatoria la stesura di query in linguaggio SQL, inoltre permette una maggiore semplicità durante la manutenzione del codice dovuta al fatto che solamente un linguaggio viene utilizzato. Mediante ORM è anche possibile cambiare database in modo efficace ed efficiente, in quanto non è necessario modificare il codice già esistente. I modelli aiutano ad effettuare le operazioni CRUD (Create, Read, Update, Delete) sul database in modo semplice e veloce. Inoltre, Django permette di definire delle relazioni tra i modelli, in modo da poter accedere ai

dati in maniera semplice e veloce, oltre alla definizione di ciò che viene definita Business Logic del programma (es. validazione dei dati, ecc.). [4]

- View : Esso può essere visto come l'organoesecutivo della Business Logic, in quanto permette di interagire con i modelli definiti dal modulo Model e gestisce la renderizzazione in template. La view ottiene i dati dal Model, dopodichè può modificare i suddetti ed eventualmente renderizzarli in un template. Esso accetta richeste HTTP(HyperText Transfer Protocol), applica la business logic e fornisce risposte HTTP al client. [7]

- Template : Questo modulo è il livello più vicino all'utente, in quanto è un livello di presentazione che gestisce completamente l'interfaccia dell'utente. Essi sono file con codice HTML(HyperText Markup Language) che vengono utilizzati per renderizzare dati ed essi possono contenere dati statici o dinamici. Nel caso di Django viene utilizzato un lingaggio speciale chiamato Django Template, il quale permette la definizione di variabili, cicli, condizioni, ecc. oltre ai comandi classici di HTML. Inoltre, permette di definire un template base, il quale può essere esteso da altri template. [6]

Di seguito viene rappresentato un diagramma esplicativo dell'architettura di Django.

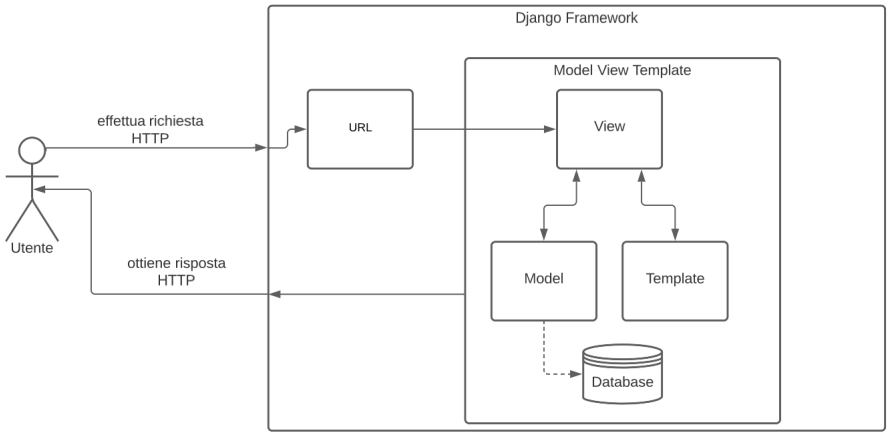

Figura 2.1: Architettura del framework Django

Il processo di una richiesta e risposta HTTP utilizzando il framework Django è il seguente:

1. L'utente effettua una richiesta HTTP dal browser fornendo un URL(Uniform Resource Locator).
2. Django valuta l'URL e controlla che esista una View corrispondente a tale URL.
3. Entrando nel Model View Template la View cattura la richiesta, elabora i dati ed invia una risposta. Possono accadere le seguenti azioni :
   (a) La View richiede l'accesso a dei dati al componente Model : la query espressa sottoforma di linguaggio python viene valutata e trascritta nel linguaggio del database (nel contesto di questo specifico applicativo in SQL) e fornisce il QuerySet(l'insieme dei dati richiesti) alla View.
   (b) Con i dati ottenuti la view effettua delle modifiche a questi dati ed invia una copia al componente Model.
   (c) Se richiesto, utilizzando i dati ottenuti, la view formula una richiesta di visualizzazione di una pagina HTML al componente Template e passa come argomenti gli eventuali dati aggiuntivi ottenuti. Una volta ricevuta la richiesta tale modulo genera i dati HTML necessari e li restituisce alla View
4. Dopo aver fornito una risposta a Django, il componente View si incarica dell'inviare la risposta HTTP della richiesta effettuata mediante URL.

Ovviamente nel caso l'operazione del CRUD pattern sia una Create o una Delete e non una Read o Update, la richiesta di accesso ai dati del Database da parte del componente View sarà gestita in maniera appropriata(fornire l'accesso al dato e salvare l'entry nel database nel caso di una Create, o fornire la possibilità di eliminare una specifica entry nel database nel caso di una Delete).

2.1.2 Sicurezza

Un altro vantaggio del framework Django è la sicurezza, in quanto nativamente supporta dei controlli che possono essere facilmente inclusi sfruttando il livello d'astrazione fornito da un framework. [5]

Oltre ai controlli sull'autenticazione, autorizzazione e gestione delle sessioni, tale framework permette di evitare attacchi tra i quali:

- SQL Injection : Attacco volto all'inserimento di codice SQL in input non controllati, in modo da poter accedere ai dati del database. Django permette di evitare tale attacco in quanto permette di utilizzare un linguaggio di programmazione orientato agli oggetti come Python per la gestione dei dati, applicando ciò che viene definito *input sanitizing*, il quale permette di evitare l'inserimento di codice SQL in input non controllati.

- Cross Site Scripting(XSS) : Tale attacco permette all'utente di inserire codice HTML e JavaScript malevolo che verrà poi visualizzato dagli altri utenti dell'applicazione mediante la pressione di un link o bottone. Tale problema viene risolto mediante input sanitizing e l'escape automatico di caratteri non sicuri.

- Cross Site Request Forgery(CSRF): L'attacco CSRF sfrutta il fatto che molte applicazioni web autenticano gli utenti utilizzando sessioni o token di autenticazione, ma non validano in modo appropriato la fonte delle richieste che vengono inviate. In un attacco CSRF, un malintenzionato crea una pagina web malevola o un messaggio e lo fa visualizzare all'utente target mentre è autenticato in un'altra applicazione web.

  Quando l'utente visualizza questa pagina o questo messaggio, il codice maligno presente al suo interno genera una richiesta HTTP verso l'applicazione web bersaglio, sfruttando la sessione o il token di autenticazione dell'utente. Poiché l'applicazione web bersaglio considera la richiesta legittima, esegue l'azione richiesta, che potrebbe essere, ad esempio, un trasferimento di fondi, una modifica delle impostazioni dell'account o l'invio di un messaggio. Django permette di evitare tale attacco mediante l'utilizzo di un token di sicurezza.

- Clickjacking : Attacco che mira ad ingannare l'utente mediante il click di zone nascoste o trasparenti. Django permette di evitare tale attacco mediante l'utilizzo di un middleware che permette di inserire un'intestazione HTTP.

2.2 Bootstrap

Per poter implementare blocchi di stile in modo efficiente e per rendere l'applicazione responsive viene utilizzato Bootstrap, un framework open-source di sviluppo Front-end e viene considerato il framework CSS più utilizzato per creare applicativi web e Mobile-first.

Tale framework è estremamente personalizzabile, è compatibile con tutti i browser moderni e permette di evitare inconsistenze nel caso vengano utilizzati browser differenti. Precisamente viene utilizzato Bootstrap v5.0 [1]

2.3 SCSS

Per permettere la customizzazione di alcuni elementi di stile, viene utilizzato Sassy Cascading Style Sheets (SCSS), una delle estrensioni del classico CSS più stabili, mature e potenti che permette di utilizzare variabili, annidamento, mixin, import, ereditarietà e altro, rendendo più semplice la scrittura di codice CSS. [8]

2.4 Shibboleth
2.5 Mail

3 Analisi di Contesto e definizione di Challenge

In questo capitolo viene introdotta una definzione formalizzata di Challenge, successivamente alla quale viene introdotta un'analisi di contesto, necessaria per la definizione di attori e come essi interagiscono tra loro.

3.1 Challenge

Viene definita **Challenge** una sfida proposta dall'azienda esterna all'università ed è composta da:

- Nome della Challenge.
- Laboratorio nel quale si svolgerà.
- Descrizione accurata del problema da risolvere.
- Data di inizio.
- Data di fine.
- Orario.
- Autore, ossia il nome dell'azienda.
- Descrizione testuale del premio messo in palio dall'azienda per il gruppo vincitore.
- Limite temporale per l'iscrizione dei gruppi.
- Dimensione di ogni gruppo.
- Numero massimo di gruppi.

Ogni azienda può formare più di una Challenge. Vi possono essere molteplici Challenge all'interno di un laboratorio.

Alla conclusione del limite temporale previsto l'azienda che ha creato la Challenge **deve** selezionare i gruppi che parteciperanno entro un quantitativo di giorni. Gli utenti facenti parte dei gruppi selezionati riceveranno una mail contenente i dati della Challenge con eventuali dettagli aggiuntivi.

Alla conclusione di una Challenge ad ogni studente è richiesto di compilare un form per fornire feedback sull'attività appena svolta per poterlo inviare all'azienda. L'applicativo non prevede uno strumento di comunicazione tra azienda e studente oltre al form sopracitato.

Vi è una diretta correlazione **l'area tematica** della challenge proposta dall'azienda ed il **laboratorio didattico**, in quanto la suddetta può formalizzare una challenge di una specifica area tematica solamente nel laboratorio corrispondente.

Alla creazione della Challenge, essa viene inserita in un Pending state finchè non viene accettata/rifiutata dal moderatore. Dopo tale azione essa entra in uno stato Accettato o Rifiutato in base alla scelta del suddetto.

3.2 Attori

Di seguito vengono riportati gli attori dell'applicativo. Ad ogni attore vengono associati i requisiti funzionali correlati sulla base delle azioni che possono effettuare.

3.2.1 Studente

Uno studente viene definito come una persona iscritta all'Università di Trento e che attualmente ricopre il ruolo di studente del corso di laurea triennale o magistrale, dottorando o ricercatore.

Tale utente può effettuare le seguenti azioni:

- Selezionare le aree tematiche al quale è interessato per poter ricevere notifiche alla validazione di nuove Challenge da parte del moderatore.
  - Lo studente può decidere di non ricevere email inerenti a nuove Challenge facente parte della sua area tematica.
- Visualizzare la lista delle Challenges facenti parte dell'area di interesse selezionata.
- Visualizzare informazioni inerenti alla Challenge selezionata, potendo visualizzare la *Reputazione* della compagnia che ha proposto la Challenge (vedi sezione 3.5).
- Iscriversi alla Challenge formando o entrando a far parte di un gruppo.
- Al termine di una Challenge lo studente può compilare un form per fornire un feedback all'azienda sulla Challenge appena compiuta.
- Al termine di una Challenge lo studente può visualizzare il feedback ricevuto da parte del Company Manager.

Uno studente può avere molteplici aree di interesse e può partecipare a più di una Challenge appartenente al suo settore di interesse.

3.2.2 Gruppo di Studenti

Per poter partecipare ad una Challenge gli studenti devono formare un gruppo. Un gruppo è definito da uno o più studenti ed un Tutor. Alla formazione di un nuovo gruppo il primo nominativo diviene il capogruppo ed ad esso viene richiesto di:

1. Fornire il nome per il gruppo.
2. Selezionare il Tutor del gruppo.
3. Fornire una presentazione testuale del gruppo.

Alla formazione di un gruppo esso è in stato di **Pending**. Esso può cambiare di stato quando:

- Il tutor selezionato visualizza il dettaglio del gruppo ed accetta di farne parte. In tal caso lo stato del gruppo diviene **Tutored**.
- Il tutor selezionato visualizza il dettaglio del gruppo e rifiura di farna parte. In tal caso lo stato del gruppo diviene **Rejected**.
- Il Company Manager visualizza il dettaglio del gruppo e lo accetta per partecipare alla Challenge. In tal caso lo stato del gruppo diviene **Accepted**.
- Il Company Manager visualizza il dettaglio del gruppo e lo rifiuta. In tal caso lo stato del gruppo diviene **Rejected**.

Per partecipare ad una Challenge è richiesto che il gruppo sia in **Accepted** state. Il gruppo ha un tempo di vita limitato alla durata della Challenge, dunque tale entità viene creata per la Challenge.

3.2.3 Moderatore

Tale figura é incaricata di accettare le Challenges che potranno essere effettuate, oltre al modificare i membri dei gruppi. Ogni Laboratorio ha un solo responsabile. Esso viene riferito anche come *responsabile di laboratorio*.

L'utente può effettuare le seguenti azioni:

- Visualizzare tutti i laboratori che gestisce.
- Visualizzare tutte le challenge proposte per quel laboratorio.
- Accettare o rifiutare le Challenges proposte dall'azienda.
- Prenotare il laboratorio nel caso la Challenge venga accettata. Tale azione è esterna all'applicativo.
- Gestire gli account delle aziende(creazione, eliminazione, modifica, etc). Tale azione è esterna all'applicativo.

3.2.4 Company Manager

Questo utente viene creato all'interno del Sistema in seguito alla formulazione ed approvazione di una richiesta esterna all'applicativo.

Tale utente può effettuare le seguenti azioni:

- Creare una Challenge.
- Visualizzare tutte le Challenge che ha effettuato.
- Modificare Challenge non accettate o rifiutate.
- Accedere al dettaglio della Challenge e visualizzare ogni gruppo.
- Accettare o rifiutare l'ingresso di un gruppo ad una determinata Challenge.
- Al termine di una Challenge il Company Manager può fornire il feedback mediante un modulo apposito per il lavoro svolto ad ogni gruppo e studente.
- Al termine di una Challenge il Company Manager può Visualizzare il feedback di tutti gli studenti che hanno partecipato alla Challenge e lo hanno fornito.

3.2.5 Tutor

Tale figura è un membro del DISI ed è incaricata della supervisione di un gruppo durante una Challenge. Alla creazione di un gruppo per partecipare ad una Challenge **deve** essere selezionato.

Esso può svolgere le seguenti attività:

- Visualizzare tutti i gruppi che hanno richiesto la sua partecipazione.
- Accettare di entrare a far parte di un gruppo o rifiutarlo. Si ricordi che all'accettazione da parte del tutor il gruppo entra in **Tutored** state.

3.3 Diagramma di Contesto

Nel seguente elenco è posisbile trovare una descrizione ad alto livello delle relazioni tra le differenti componente esterne al sistema. Bisogna notare che tale rappresentazione viene effettuata a livello logico e non è correlata all'architettura del framework utilizzato.

Il **Company Manager** è un **attore** e può svolgere le seguenti azioni: può effettuare il login, gestire le proprie challenge, accettare o rifiutare i gruppi (già in tutored state, ossia gruppi i quali siano già formati ed accettati da un Tutor), oltre a poter compilare e visualizzare i feedback inerenti alle challenge dell'azienda.

Il **Moderatore** è un **attore** e può svolgere le seguenti azioni: può visualizzare i laboratori che modera, accedere al loro dettaglio ed accettare o rifiutare le Challenge proposte per il laboratorio.

Lo **Studente** è un **attore** e può svolgere le seguenti azioni: può cambiare i propri interessi in modo da vedere solo le challenge che facciano parte della sua area di interesse, può decidere se essere aggiornato o meno sulle nuove Challenge approvate dal moderatore, formare e gestire i gruppi creati appositamente per le Challenge, fornire e visualizzare i feedback relativi alle Challenge effettuate.

Il **Tutor** è un **attore** e può svolgere le seguenti azioni: può visualizzare i gruppi che hanno richiesto la sua partecipazione, accettare o rifiutare di farne parte.

Il **Sistema di invio mail/notifica** è un **subordinato** ed è l'infrastruttura che permette l'invio di email nei seguenti casi ed ai seguenti attori:

- Creazione di una nuova Challenge da parte del Company Manager : viene inviata una mail a tutti i moderatori del laboratorio nella quale la Challenge deve essere svolta.

- Accettazione o rifiuto di una Challenge da parte del Moderatore: viene inviata una mail al Company Manager che ha creato la Challenge, oltre che ad una mail a tutti gli studenti che hanno richiesto di essere aggiornati sulle nuove Challenge e i quali interessi siano un super-set della Challenge accettata.

- Creazione di un nuovo gruppo da parte di uno studente : viene inviata una mail al Tutor selezionato dal capogruppo.

- Accettazione o rifiuto di entrare a far parte di un gruppo da parte del Tutor : viene inviata una mail a tutti i membri del gruppo.

- Accettazione o rifiuto di un gruppo da parte del Company Manager : viene inviata una mail a tutti i membri del gruppo.

Shibboleth è un **subordinato** ed è il modulo che permette di effettuare il login all'interno dell'applicativo. Esso è un software open source che permette l'autenticazione federata e single sign-on.

Il **Database** è un **subordinato** ed è il corrispettivo di ciò che è stato definito come componente Model all'interno dell'architettura di Django. Esso gestisce le transazioni con il Database ed è l'unico modulo in grado di accedere ai dati. Esso permette la traduzione di query da linguaggio python al linguaggio della base di dati selezionata.

Di seguito viene riportato il diagramma di contesto dell'applicativo.

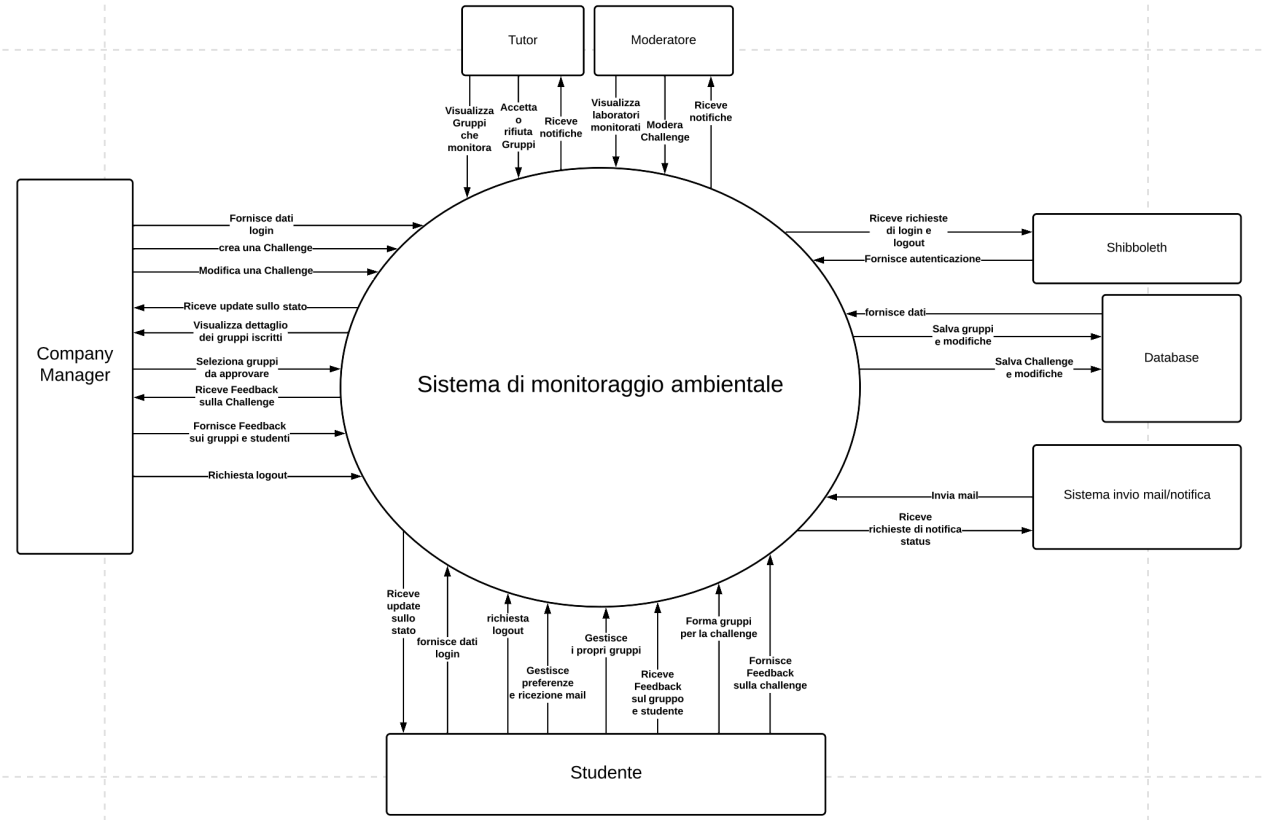

Figura 3.1: Diagramma di contesto

3.4 Processo di creazione e partecipazione ad una Challenge

Data la complessità del processo, viene di seguito utilizzato un diagramma Business Process Model Notation (BPMN) per descrivere il processo di creazione e partecipazione ad una Challenge.

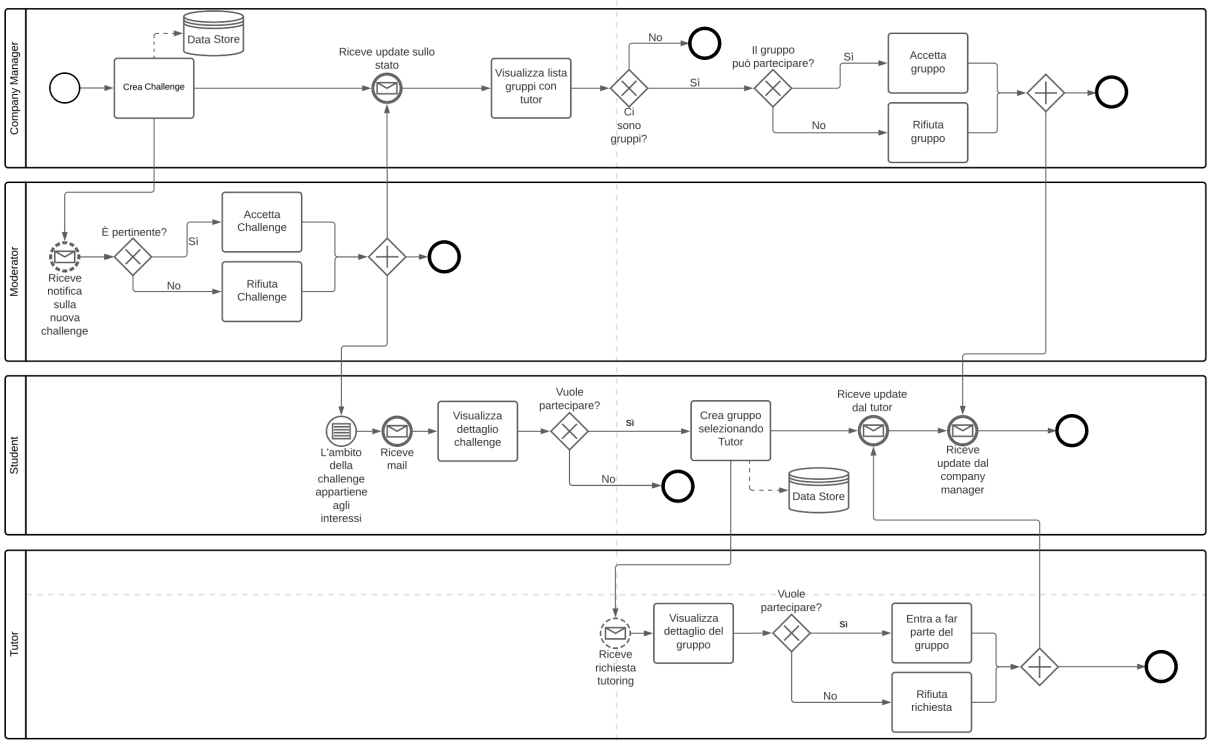

Figura 3.2: BPMN della creazione di una Challenge

Il flow dell'applicativo è il seguente:

1. Il Company Manager crea una Challenge ed una mail viene inviata ai moderatori del laboratorio scelto come luogo per svolgere la Challenge.

2. Il Moderatore riceve la mail di richiesta di accettazione di una Challenge. Accede alla sua area personale, dove può visualizzare i suoi laboratori ed accede al dettaglio della Challenge. Alla conferma o rifiuto della Challenge viene inviata una email al Company Manager ed agli studenti che hanno sia scelto di essere aggiornati sulle nuove Challenge, sia che abbiano interessi che siano un super-set della Challenge accettata.

3. Gli studenti possono visualizzare la challenge approvata e formare dei gruppi o entrare a far parte di gruppi già creati mediante il codice fornito al capo gruppo. Alla creazione di un gruppo il tutor selezionato riceve una email.

4. Il tutor riceve la email e può accettare o rifiutare di entrare a far parte del gruppo. Alla sua decisione viene inviata una mail a tutti i membri del gruppo.

5. Il Company Manager può accedere al dettaglio della Challenge ed accettare o rifiutare di far partecipare i gruppi che sono già stati accettati da un tutor. Alla sua decisione viene inviata una mail a tutti i membri del gruppo.

3.5 Valutazione delle Challenge e dei gruppi

Al termine di una Challenge, il sistema rende disponibile la compilazione di feedback. Un feedback viene rappresentato da un form Django ed esso può essere compilato dai seguenti attori: Company Manager e Studente.

Il Company Manager può compilare il form dando una valutazione al gruppo ed ai singoli studenti, i quali vedranno all'interno della pagina di dettaglio del gruppo formato per la specifica Challenge la valutazione ricevuta.

Lo studente, invece, può compilare il form dando una valutazione alla Challenge che ha effettuato. Il Company Manager vedrà all'interno della pagina di dettaglio della Challenge sia la valutazione media ricevuta, che la versione anonimizzata del feedback fornito da ogni studente.

Assieme al concetto di feedback dello studente, è utile introdurre il conceto di **Reputazione** della compagnia. Tale valore raccoglie la media di tutti i feedback effettuati da tutti gli studenti per ogni Challenge creata dall'azienda ed è utile per poter dare una valutazione oggettiva della compagnia per quanto concerne la creazione di Challenge all'interno del sistema.

4 Conclusioni

Conclusioni

Bibliografia

[1] Bootstrap. https://getbootstrap.com/docs/5.0/getting-started/introduction/.

[2] Crispy forms. https://django-crispy-forms.readthedocs.io/en/latest/.

[3] Disi industry. https://industry.disi.unitn.it.

[4] Django models. https://docs.djangoproject.com/en/4.2/topics/db/models/.

[5] Django security. https://docs.djangoproject.com/en/4.2/topics/security/.

[6] Django template. https://docs.djangoproject.com/en/4.2/topics/templates/.

[7] Django views. https://docs.djangoproject.com/en/4.2/topics/class-based-views/.

[8] Scss. https://sass-lang.com.