“Software Engineering”

Course

a.a. 2019-2020

Lecturer: Prof. Henry Muccini (henry.muccini@univaq.it)

Progetto 6
Mettimi in contatto con…

Date	<15/01/2019>
Deliverable	Milestone #2
Team (Name)	Flop Team

Team Members		
Nome e cognome	**N# di matricola**	**E-mail**
Alessandro D’Orazio	251811	alessandro.dorazio2@student.univaq.it
Lorenzo Cilli	253121	lorenzo.cilli@student.univaq.it

Table of Contents of this deliverable

1. Specifiche del progetto

UniChat è un servizio di messaggistica disponibile per contesti universitari. Permette infatti di effettuare lo scambio di messaggi tra studenti universitari, docenti e personale amministrativo. Ogni università ha un database differente, con lo stesso schema.

Le conversazioni sono raggruppate in stanze. Le stanze possono essere di tre tipi:

1. Chat diretta, in cui la stanza contiene solamente due partecipanti, ed entrambi i partecipanti possono inviare messaggi. La comunicazione è multidirezionale.
2. Gruppo, in cui la stanza contiene due o più partecipanti e tutti i partecipanti possono inviare messaggi. La comunicazione è multidirezionale.
3. Feed, in cui la stanza contiene due o più partecipanti, suddivisi in amministratori e lettori. Gli amministratori possono inviare e leggere messaggi, mentre i lettori possono esclusivamente leggerli. La comunicazione è monodirezionale.

2. Challenging/Risky Requirements or Tasks

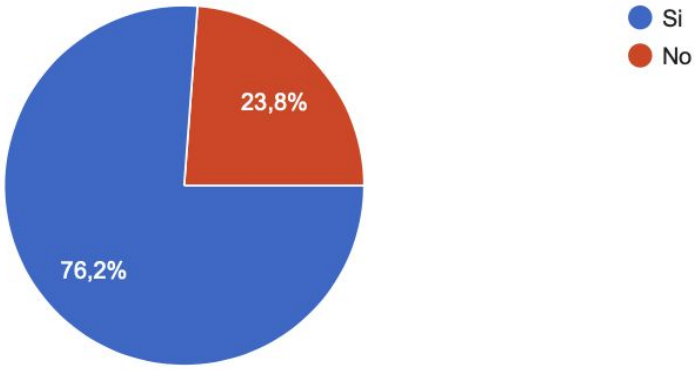
Challenging Task	Date the task is identified	Date the challenge is resolved	Explanation on how the challenge has been managed
Gestione di chat diretta, gruppi e feed	09/12/2019	11/12/2019	Chat diretta, gruppi e feed vengono viste come stanze dal sistema. In questo modo, queste tre entità diventano una tipologia di stanza
Ottimizzazione invio messaggi	19/12/2019	21/12/2019	Uso di WebSocket
Verifica di un utente	14/01/2020	14/01/2020	Un utente del personale amministrativo riceve i documenti e lo verifica
Gestione spam	14/01/2020	14/01/2020	Sistema automatizzato + segnalazioni

3. Stato dell'Arte

È stato realizzato un questionario, condiviso in un gruppo Facebook di studenti universitari.
Questi sono i risultati:

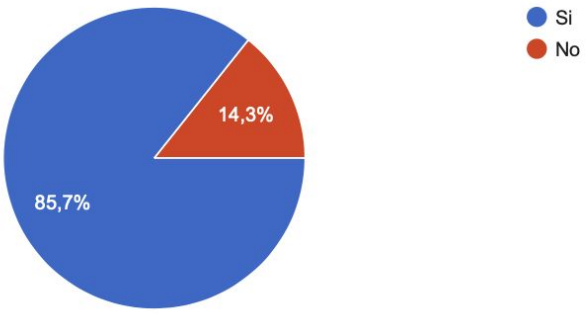
Ti piacerebbe avere una versione desktop del software?

21 risposte



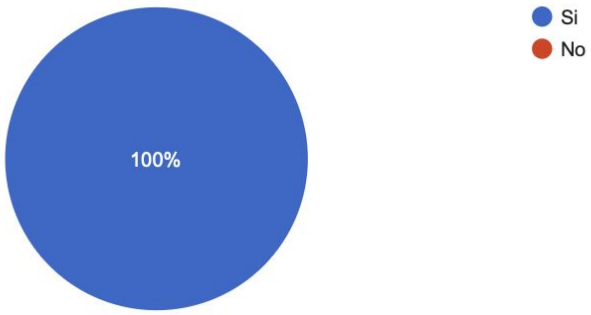
Ti piacerebbe avere una sezione con tutti i messaggi importanti che hai contrassegnato o che ti sono stati contrassegnati da admin di feed/gruppi?

21 risposte



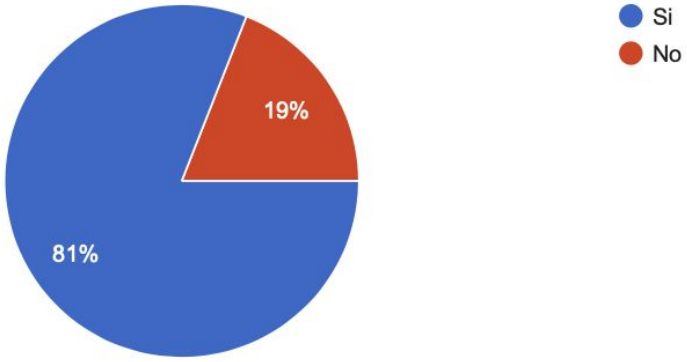
Ti piacerebbe comunicare direttamente con la segreteria tramite un app di messaggistica?

21 risposte



Ti piacerebbe la possibilità di eseguire sondaggi all'interno di feed/gruppi?

21 risposte



Quale sarebbe per te una funzione fondamentale?

9 risposte

- Risposte immediate da parte dei professori/segreterie
- Boh
- Comunicare direttamente con la segreteria tramite un'app di messaggistica
- Possibilità di inviare allegati
- La visualizzazione dei messaggi inviati
- Non so
- Sapere entro quando visualizzeranno il mio messaggio
- Un'app tipo WhatsApp solo per comunicare con i prof data l'inadeguatezza delle mail
- Comunicare con gli insegnati e sveltire le pratiche della segreteria studenti

3.1 Schoology

Schoology è un social network e un ambiente di apprendimento virtuale.
Il software permette la condivisione di documenti in ambito universitario, infatti, accendendo tramite la propria università ed iscrivendosi ai corsi di interesse, è possibile comunicare con il docente, tramite un sistema di messaggistica, ed usufruire del materiale da lui caricato.
La condivisione dei documenti però, non è vincolata solamente ai docenti, anche uno studente può caricare i propri file, creare le proprie cartelle e farle vedere a qualunque utente sfogli il proprio profilo.
Il sistema, tuttavia, non permette la comunicazione tra studenti.
La sezione, nella homepage, delle attività recenti presenta un ottimo spunto per la gestione dei feed, infatti si presenta come una pagina nella quale un solo utente, gli/l'admin del corso, può effettuare comunicazioni che verranno inviate ad ogni utente iscritto.

3.2 Edmodo

Edmodo è un software che permette la comunicazione,collaborazione e coaching per scuole e insegnanti.
Il punto forte di Edmodo sono i differenti tipi di account, ne esistono tre: insegnante, studente e genitore.

Edmodo è molto concentrato sull'insegnante, infatti per accedere come studente o genitore bisogna avere il codice di accesso del corso da seguire/ seguito dal proprio figlio.
L'insegnante, come ogni piattaforma del genere, può condividere materiale, contattare uno o più studenti, assegnare compiti o quiz.
Lo studente, invece, per registrarsi ha bisogno del codice di accesso che solo un insegnante può fornire; una volta entrato nel corso può comunicare direttamente con il docente, usufruire dei materiali e svolgere i compiti da lui caricati.
L'utente genitore invece ha una funzione più da spettatore, questo infatti può vedere l'andamento del proprio figlio, sempre utilizzando il codice del corso fornito dall'insegnante, i suoi voti e pesino comunicare con il docente per qualsiasi eventualità.

3.3 Telegram

Telegram è un servizio di messaggistica basato su cloud, che permette lo scambio di messaggi,documenti,immagini, ecc. in una conversazione o in gruppi; anch'esso nasce come servizio mobile per poi espandersi anche su PC con il software Telegram Web.
I messaggi inviati sono salvati sul cloud di Telegram, così da garantire la sincronizzazione istantanea, il risultato consente all'utente di poter accedere ai messaggi da diversi dispositivi contemporaneamente.
Una delle caratteristiche principali di Telegram che lo distingue dal resto dei suoi concorrenti sono le chat (classiche e segrete), i canali e i bot.
Le chat, come già detto, si dividono in due tipi, le chat classiche, che utilizzano una cifratura client-server ovvero è cifrata dal dispositivo fino al server quindi la conversazione rimane salvata in maniera cifrata sui server, e le chat segrete, che utilizza una cifratura end-to-end ossia è cifrata fra i due dispositivi coinvolti nella conversazione, questa chat ha un'ulteriore funzione: l'autodistruzione; questa funzione permette di impostare un timer nella chat e di eliminare tutti i contenuti alla fine del countdown aumentando la sicurezza per la privacy.
I canali, chat in cui chiunque sia amministratore può inviare messaggi ai membri del canale, anche se questi ultimi non possono rispondere né commentare.
I Bot sono degli account Telegram, gestiti da un programma, che offrono molteplici funzionalità con risposte immediate e completamente automatizzate.

3.4 Slack

Slack è una piattaforma per la collaborazione tra team. Nonostante essa non rappresenti un servizio molto simile a quello richiesto dal progetto, offre vari spunti interessanti da poter implementare nel nostro prodotto.

Slack integra un sistema di workspace, cioè spazi di lavoro differenti per ogni ambiente in cui si lavora. Ogni workspace viene organizzato in canali multidirezionali, e la conversazione, oltre ad avere un sistema di messaggistica tradizionale, integra un sistema basato sui thread, come nei forum, in modo tale da poter discutere riguardo un determinato topic evitando la perdita di informazioni.
Il sistema di ricerca dei messaggi è molto interessante dal punto di vista dell'usabilità, poiché tramite un'unica barra di ricerca, è possibile ritrovare messaggi, file, canali ed utenti.
È presente anche una funzione per contrassegnare dei messaggi come importanti, visibili cliccando sull'apposita icona (stella).

La sicurezza è garantita dall'autenticazione a due fattori e da un meccanismo di data encryption.

Vengono stimati circa 10 milioni di utenti. Effettuando ricerche in merito alle tecnologie utilizzate, emerge che il sistema di messaging è scritto in Java, mentre il front end del sito Web è scritto con React (Javascript).

3.5 Whatsapp

WhatsApp è un'applicazione che permette lo scambio di messaggi di testo, immagini, viedeo e file audio, informazioni sulla pusizione, documenti e informazioni di contatto tra due persone o più tramite gruppi, il tutto gratuitamente (anche se inizialmente aveva un prezzo annuale), questa è proprio il punto di forza del servizio che è arrivato nel giro di poco tempo a sostituire del tutto i normali messaggi.
Nasce inizialmente come applicazione mobile per poi essere sviluppato anche una versione su desktop chiamata WhatsApp Web.
Tuttavia un problema noto per quanto riguarda la privacy è che WhatsApp richieda l'accesso alla rubrica di ogni utente per poter connettere fra loro i contatti facenti uso dell'applicazione, questo metodo è molto utile ma allo stesso tempo fornisce una copia dei propri dati replicati sui server della società per questo le informazioni relative ai numeri telefonici sono memorizzate mediante hash e non forniscono altre informazioni che possano identificare l'utente.
Nel 2016 WhatsApp introduce la crittografia end-to-end che garantisce che solo il mittente e il destinatario possano leggere ciò che viene inviato, e nessun altro, nemmeno WhatsApp.
Ciascuna delle tue chat ha un proprio codice di sicurezza. Questo codice serve a verificare che le tue chiamate e i messaggi inviati in una determinata chat siano crittografati end-to-end.
I messaggi sono protetti con dei lucchetti, e solo i due interlocutori avranno le chiavi necessarie per poterli aprire e leggere i messaggi. Per una maggiore protezione, ciascun messaggio inviato ha un proprio lucchetto e una propria chiave

4. Raffinamento dei Requisiti

4.1 Servizi (con prioritizzazione)

Importanza alta = 3 / Importanza media = 2 / Importanza bassa = 1
Complessità alta = 3 / Complessità media = 2 / Complessità bassa = 1

1. **Stanza**
   - 1.1. Visualizzazione delle stanze. I=3 / C=1
     L'utente può visualizzare le stanze in cui è partecipante o amministratore, in modo tale da poter inviare e visualizzare i messaggi relativi alla stanza, e nel caso sia amministratore, gestire la stanza.

   - 1.2. Accesso ad una stanza. I=2 / C=2
     L'accesso ad una stanza in un gruppo o feed può avvenire in due modalità
     - 1.2.1. Ricerca della stanza.
       L'utente cerca la stanza tramite l'apposito pannello, premendo il pulsante "entra"
     - 1.2.2. Link di invito.
       Un utente può condividere agli altri utenti un gruppo o un feed utilizzando l'apposito strumento di condivisione. Permette di condividere un link ad altri utenti, che cliccando sul link potranno entrare nel gruppo/feed

     L'accesso ad una chat diretta non è possibile a seguito della creazione della chat. Infatti, durante la creazione della chat diretta, viene prima

selezionato l'altro utente che farà parte della chat (oltre all'utente che la crea), e successivamente nessun utente potrà entrare nella stanza.

1.3. Comunicazioni importanti. I=2 / C=2
L'utente amministratore di un gruppo o feed può mettere in evidenza dei messaggi. Evidenziando un messaggio, esso sarà sempre visibile per un periodo di tempo stabilito dall'amministratore, e sarà inviata una notifica ai partecipanti.
Nelle chat dirette entrambi i partecipanti possono evidenziare i messaggi, e gli effetti sono i medesimi dei messaggi evidenziati in gruppi e feed.

1.4. Informazioni della stanza. I=2 / C=1
Sarà possibile visionare i partecipanti della stanza e, se presenti, nome e amministratori di essa.

1.5. Archiviazione di una stanza. I=1/C=1
L'utente può archiviare una stanza di cui fa parte tramite l'apposito pulsante "archivia stanza". Per visualizzare l'elenco delle stanze archiviate, l'utente può aprire l'elenco cliccando sul pulsante presente nella home. La chat sarà archiviata solo per l'utente che archivia la chat.

1.6. **Gestione dello SPAM**. I=2 / C=2

1.6.1. Sistema antiflood. I=2 / C=3
L'utente non può inviare un messaggio con lo stesso contenuto per tre volte di fila nell'arco di un minuto. In questo modo, la conversazione risulterà più pulita e funzionale.

1.6.2. Segnalazioni. I=2 / C=1
L'utente di una stanza può segnalare un messaggio.
La segnalazione arriverà all'amministratore tramite notifica, che potrà decidere se silenziare o meno il mittente del messaggio.

2. Chat diretta

2.1. Creazione di una chat diretta. I=3 / C=1
L'utente che vuole comunicare tramite chat diretta con un altro utente può creare una stanza apposita, nel caso in cui non esista già. Se la stanza esiste già, la comunicazione tra i due utenti mediante chat diretta viene effettuata nella stanza già esistente.

3. Gruppo

3.1. Creazione di un gruppo. I=3 / C=1
All'atto di creazione di un gruppo, viene creata una stanza dove

l'utente creatore viene nominato amministratore. Deve essere assegnato un nome al gruppo.

3.2. Gli <u>amministratori</u> di un gruppo possono modificare il nome del gruppo, aggiungere/rimuovere partecipanti, nominare altri amministratori ed evidenziare i messaggi importanti.

3.3. <u>Gestione del ban</u> I=2 / C=2

3.3.1. Gli amministratori di una stanza possono silenziare un utente all'interno di essa per un periodo di tempo scelto dall'amministratore. Prima del ban, l'amministratore può consultare uno storico relativo all'utente, che può utilizzare come parametro per stabilire la durata del ban.

3.3.2. Il ban può essere rimosso da un amministratore manualmente, oppure viene rimosso automaticamente alla scadenza del periodo di tempo

3.4. <u>Ban automatico</u>. I=2 / C=3
È presente un dizionario di parole, stabilito dal personale amministrativo, in cui sono presenti tutte le parole vietate.

3.4.1. Nel caso in cui venga inviato un messaggio con una parola vietata, il messaggio non viene inviato e viene inviata una notifica al mittente, contenente "il messaggio è stato eliminato per parole vietate". La terza volta che uno stesso utente utilizza una parola vietata, esso viene silenziato per un giorno.

3.4.2. Gli sviluppatori del software forniscono una lista di parole vietate, per agevolare i compiti del personale amministrativo, che si occupa di inserire e rimuovere le parole nel dizionario.

3.5. Motivazioni. Un utente può essere bannato per linguaggio scorretto o per spam.

4. Feed

4.1. <u>Creazione</u> di un feed. I=3 / C=1
All'atto di creazione di un feed, viene creata una stanza dove l'utente creatore viene nominato amministratore. Deve inoltre essere assegnato un nome al feed.

4.2. Gli <u>amministratori</u> di un feed possono modificare il nome del feed, aggiungere/rimuovere partecipanti, nominare altri amministratori ed evidenziare i messaggi importanti.

5. Gestione degli utenti

5.1. Visualizzazione degli utenti presenti nella piattaforma. I=3 / C=1
UniChat prevede un sistema per la visualizzazione degli utenti registrati alla stessa università dell'utente autenticato. Qui è possibile vedere le informazioni di contatto degli utenti

5.2. Ban dalla piattaforma. I=2 / C=2
Qualora l'utente effettui molteplici volte delle azioni non consentite, come lo spam di messaggi e l'utilizzo di vocaboli inadeguati, è dovere del personale amministrativo di bannare l'utente dalla piattaforma per un periodo di tempo prefissato. In questo modo, l'utente non potrà temporaneamente inviare messaggi. Nel caso si tratti di un errore, l'utente può inviare una mail per effettuare un "ricorso", in modo tale da rimuovere il ban prima che il tempo scada.

6. **Filtraggio degli utenti**. I=2 / C=2
Gli utenti possono essere filtrati per tipologia,, facoltà e dipartimento. Tramite questi filtri è possibile inviare messaggi, creare gruppi e feed.

7. **Invio e ricezione di messaggi**. I=3 / C=2
Questo servizio consiste nello scambio di messaggi all'interno di una stanza. I messaggi devono essere scambiati real-time.
   7.1. Invio di messaggi a più stanze. È possibile inviare lo stesso messaggio a più stanze tramite un sistema di filtraggio che mostra tutte le stanze dell'utente.

   7.2. Thread. È possibile commentare un messaggio, per facilitare l'esperienza d'uso nel caso ci siano molti messaggi.

8. **Registrazione**. I=3 / C=1
Gli utenti si possono registrare fornendo nome, cognome, email, password, tipo, università, anno di corso e facoltà. La registrazione viene effettuata autonomamente dall'utente.
   8.1. Tipologie di utenti: studente, docente o personale dell'amministrazione.

   8.2. Verifica degli utenti. I=3 / C=1
   I docenti e il personale dell'amministrazione devono essere verificati per motivi di sicurezza, altrimenti qualsiasi utente potrebbe registrarsi come docente o personale amministrativo.
   La verifica avviene mediante l'invio di un'email ad un utente del personale amministrativo dalla casella universitaria, contenente una scansione dei propri documenti. Il personale amministrativo dovrà poi confermarlo tramite un apposito pannello.

9. **Autenticazione**. I=3 / C=1
   Gli utenti effettuano il login inserendo email e password
   9.1. Reset della password. I=3 / C=1
   Gli utenti possono autonomamente reimpostare la propria password nel caso in cui sia stata dimenticata.

10. Comunicazioni con servizi esterni.
   10.1. Visualizzazione del numero telefonico. I=2 / C=1
   Un utente può visualizzare il numero telefonico di un altro utente, per effettuare chiamate e inviare SMS.
   10.2. Email. I=2 / C=1
   L'utente può visualizzare l'email di un altro utente per avviare una conversazione. La conversazione avviene al di fuori di UniChat
   10.3. App di messaggistica. I=1 / C=1
   L'utente può impostare i propri contatti delle app di messaggistica, come Skype e Twitter.

11. **Ricerca dei messaggi.** I=2 / C=3
   L'utente può effettuare una ricerca intelligente dei messaggi, che permette di visualizzare i messaggi contenenti una determinata frase (o parola), oppure i messaggi inviati in una determinata data.

12. **Gestione delle informazioni dell'università**. I=2 / C=2
   Il personale amministrativo può inserire e rimuovere dipartimenti e corsi di laurea.
   12.1. Inserimento di un'università. I=3 / C=1
   È compito del team di sviluppo di inserire, modificare ed eliminare le università, tramite un apposito pannello.

4.2 Requisiti non Funzionali (ISO/IEC 25010)

Idoneità funzionale

Functional completeness

- Il software dovrà essere completo di tutti i servizi proposti al punto 4.1

Functional appropriateness

- Il software facilita la comunicazione tra studenti, docenti e personale amministrativo

Performance o efficienza

Time behaviour

- I tempi di risposta e di elaborazione devono essere minimi

Compatibilità

Co-existence

- Il software può essere installato su una macchina virtuale, in modo tale da poter assegnare un limite di risorse.

Interoperability

- Il software non ha bisogno di comunicare con altri prodotti per il suo funzionamento. E' comunque possibile invocare delle API per prelevare delle informazioni presenti su UniChat.
- UniChat garantisce lo scambio di 1000 messaggi al secondo, mediante un sistema real-time sviluppato tramite WebSocket

Usabilità

Appropriateness recognizability

- UniChat fornirà le funzionalità di cui studenti, docenti e personale amministrativo hanno bisogno, senza uscire dai suoi confini.

Learnability

- Gli studenti e i docenti saranno in grado di utilizzare al meglio la piattaforma dopo un training di 2 ore, mentre il personale amministrativo dopo un training di 3 ore, date le funzionalità aggiuntive a loro disposizione

Operability, User interface aesthetics

- Ogni operazione presente in UniChat deve essere possibile con il minor numero di click possibili e tramite un'interfaccia pulita

Accessibility

- Dovrà essere presente un sistema di lettura dei messaggi, permettendo ai non vedenti di usufruire di UniChat. Saranno presenti anche funzionalità relative all'ingrandimento del testo.
- Il sistema deve essere facile da usare e deve essere organizzato in modo tale che gli errori commessi dall'utente siano ridotti al minimo.

Affidabilità

Availability

- UniChat deve garantire uno scambio di messaggi sempre disponibile, poiché la perdita di comunicazioni potrebbe causare problematiche di natura logistica e organizzativa. L'up-time deve essere del 99,8%.

Fault tolerance

- Il sistema richiede due macchine per garantire l'operatività a seguito di guasti, che vengono risolti, in base al tipo di guasto, da ripresa a caldo o ripresa a freddo

Recoverability

- I messaggi saranno salvati sul server e non localmente su ogni dispositivo

Sicurezza

Confidentiality

- Gli studenti non possono visualizzare il numero di telefono di docenti e personale amministrativo, poiché potrebbero utilizzarlo in modo inappropriato.
- Solo il personale amministrativo ha accesso a determinate funzionalità,

Integrity

- La password viene codificata mediante un sistema "Bcrypt", per garantire la sicurezza della password in caso di Data Breach

Accountability

Authenticity

- Il sito Web di UniChat sarà dotato di un certificato SSL, per garantire la connessione sicura
- Alla richiesta di reset della password, viene inviata un'email all'utente per notificare quest'operazione
- I docenti e il personale amministrativo vengono verificati dagli utenti del personale amministrativo tramite un apposito pannello
- L'università dovrà comunicare al team di UniChat, tramite un modello Excel, quali saranno gli utenti del personale amministrativo
- I dati devono essere trattati seguendo le normative GDPR
- Inizialmente dovrà essere presente un utente master per ogni università, che si occuperà di verificare il personale amministrativo e i docenti. Le credenziali dell'utente master saranno relative ad un membro del personale amministrativo, scelto dall'università.

Manutenibilità

Modularity

- Il sistema di autenticazione può essere modificato senza compromettere le funzionalità del software

Reusability

- Il riuso del codice avviene mediante l'utilizzo di plugin e classi già esistenti

Analysability

- UniChat dovrà poter essere analizzato tramite strumenti di verifica statica del codice.

Testability

- Il software viene testato tramite Test Driven Development, verificando che ogni componente del software funzioni correttamente

Portabilità

- UniChat può essere utilizzato su una macchina contenente i software necessari a far funzionare uno stack LAMP e Laravel 6.0

4.3 Scenari d'uso dettagliati

1. Giorgio è uno studente di biologia. Utilizza ogni giorno UniChat tramite smartphone per scambiare messaggi relativi al progetto a cui sta lavorando, mediante il gruppo che ha creato lui stesso. Non ci ha messo molto a cercare gli altri studenti, poiché ha utilizzato il sistema dei filtri per facoltà. Fissa una scadenza per la consegna del progetto ed evidenzia il messaggio, in questo modo gli altri riceveranno una notifica per leggere il messaggio.

2. Mattia, studente di chimica, deve sostenere un esame orale di chimica, ma non può presentarsi all'orario deciso dal docente, quindi tramite la chat diretta informa il professore di questa problematica.

3. Carlo, studente di ingegneria informatica, ha ricevuto qualche settimana fa un messaggio in cui era previsto un cambio aula per la lezione di fisica, ma non riesce a ricordarne i dettagli. Utilizza quindi la funzione di ricerca dei messaggi, filtrando tra i messaggi dell'ultimo mese, quelli in cui è presente il testo "cambia aula".

4. Daniele è un docente di matematica, e ogni mattina, dopo aver bevuto il suo caffè, accende il computer ed apre UniChat. È un docente molto attivo, infatti possiede un feed dedicato all'insegnamento di "Analisi 1", da cui invia le comunicazioni agli studenti. Daniele sa che giovedì prossimo non sarà presente a lezione, quindi tramite i messaggi in evidenza, invia una notifica a tutti gli studenti per informarli. Non deve preoccuparsi di inserire i nuovi studenti nel proprio feed, poiché nella sua pagina personale ha inserito un link per partecipare al feed di "Analisi 1", di cui è docente insieme a due collaboratori, entrambi nominati da Daniele. I tre condividono un gruppo per fissare gli appuntamenti e gli orari delle esercitazioni.

5. Stefano lavora nel personale amministrativo, e si occupa dei bandi pubblici. È appena uscito un bando pubblico per una borsa di studio riservata agli studenti di medicina. Stefano quindi, tramite il sistema di ricerca integrato in UniChat, invia la comunicazione a tutti gli studenti di medicina, senza doverli cercare manualmente.
   Inoltre, ha un nuovo collega di nome Matteo nel personale amministrativo, da cui riceve una mail contenente una scansione dei documenti, per verificarlo nel sistema UniChat. La verifica avviene in meno di 30 secondi!

6. Mario è un ragazzo che si prende gioco degli altri. Prova a registrarsi all'interno di UniChat come docente, per inviare comunicazioni false, ma quando gli viene chiesto di inviare una mail dalla casella universitaria del docente contenente la scansione dei documenti, fallisce nel suo tentativo. Si registra quindi come studente, ed inizia a inviare parole offensive a tutti gli studenti, che però non ricevono nulla, poiché il sistema di SPAM non ha permesso l'invio dei messaggi. Arriva una comunicazione al personale amministrativo, che silenzia Mario dalla piattaforma.

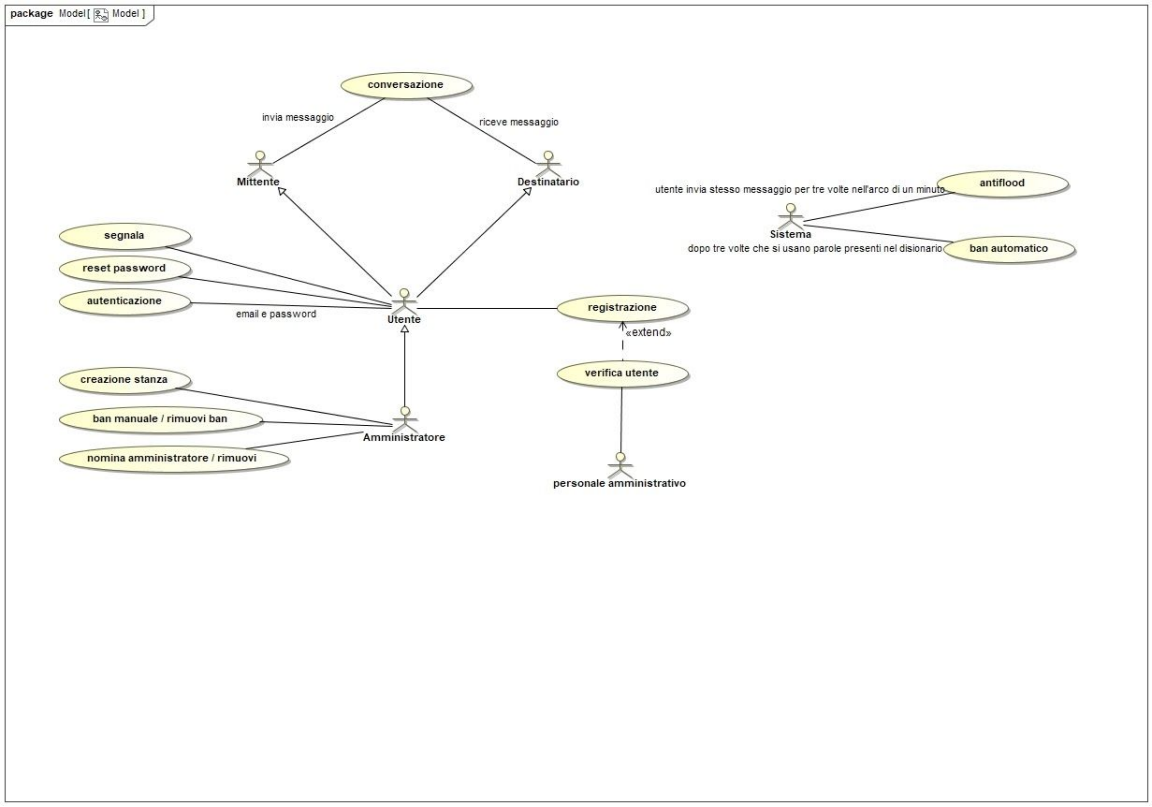
4.4 Excluded Requirements

1. Accesso con SPID o credenziali universitarie. Non viene gestito poiché allungherebbe le tempistiche, e non è detto che gli utenti abbiano delle credenziali di questi due servizi. È possibile gestire questa tipologia di accesso come modalità secondaria, ma non è stato fatto per motivi di tempo.
2. Utilizzo di allegati. Non possono essere inviati allegati, poiché, richiederebbe uno studio di fattibilità riguardo le performance e lo spazio. Nulla esclude che possano essere gestite in una futura release.
3. L'anno di corso non viene gestito dal software, poiché andrebbero analizzate situazioni come i fuori corso, l'avanzamento automatico e altre casistiche, che per motivi di tempo non sono state analizzate.
   L'anno di corso viene selezionato dall'utente
4. Segreteria. La segreteria può essere inserita creando un nuovo utente, di tipo "personale amministrativo", la sua realizzazione infatti chiederebbe altre analisi di fattibilità.

4.5 Assunzioni

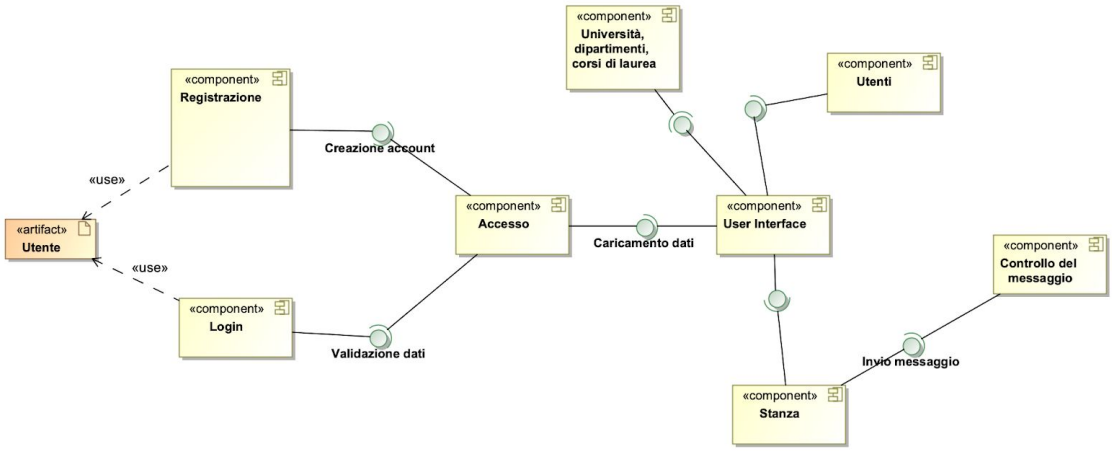
1. I permessi di ogni tipologia di utente sono gli stessi, quindi un utente del personale amministrativo ha gli stessi permessi di tutti gli altri utenti del personale amministrativo.
2. Ogni docente o membro del personale amministrativo ha una mail universitaria
3. Ogni docente creerà un feed per ogni suo insegnamento e condividerà il link dei suoi studenti. In questo modo si riducono le tempistiche di realizzazione del software, poiché non deve essere il software a creare i feed per ogni insegnamento

4.6 Use Case Diagrams



5. Architettura Software

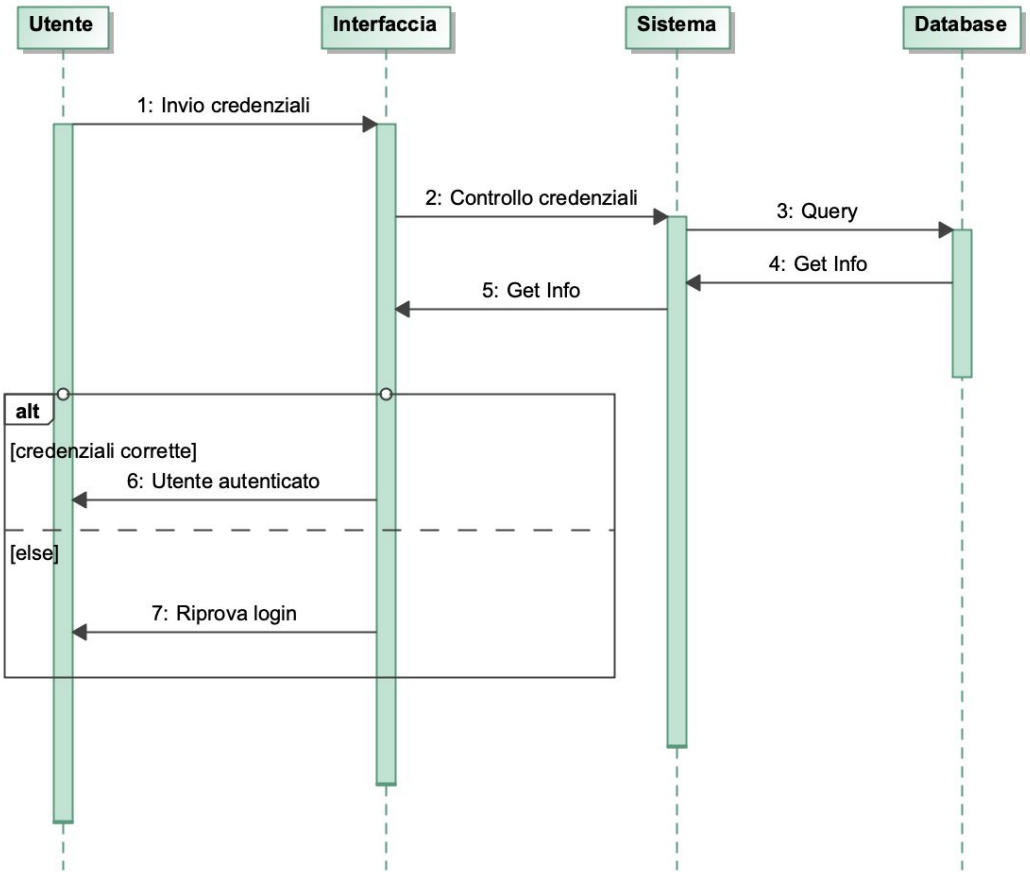
5.1 Vista statica del sistema: Component Diagram



L'utente è la persona fisica che vuole utilizzare UniChat.
La componente dell'autenticazione prevede la registrazione e il login, per la creazione o il login dell'account. Tramite la componente che regola l'accesso, vengono inviati i dati alla user interface, da cui si possono creare, visualizzare, modificare ed eliminare università, dipartimenti, corsi di laurea, utenti, stanze
Il controllo del messaggio verifica che il messaggio non contenga parole vietate e che rispetti le regole antiflood.

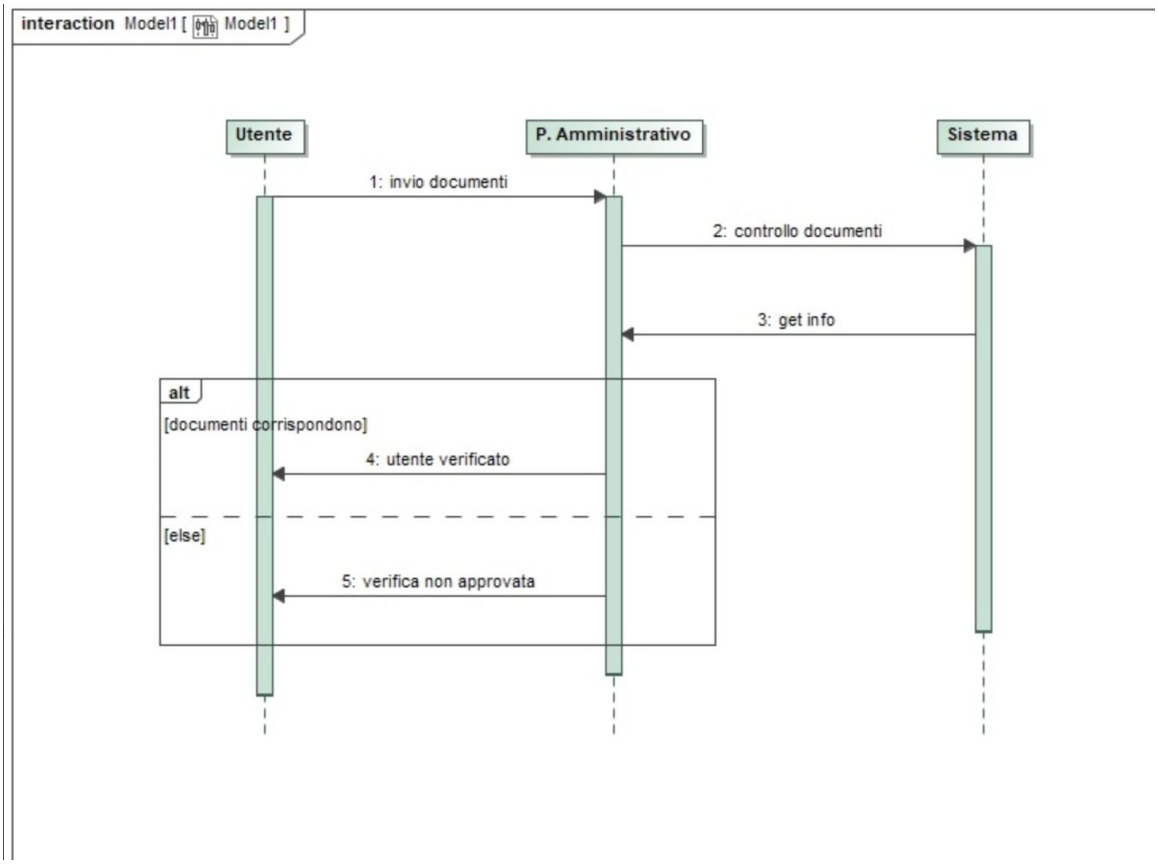
5.2 Vista dinamica del sistema: Sequence Diagram

5.3.1 Autenticazione



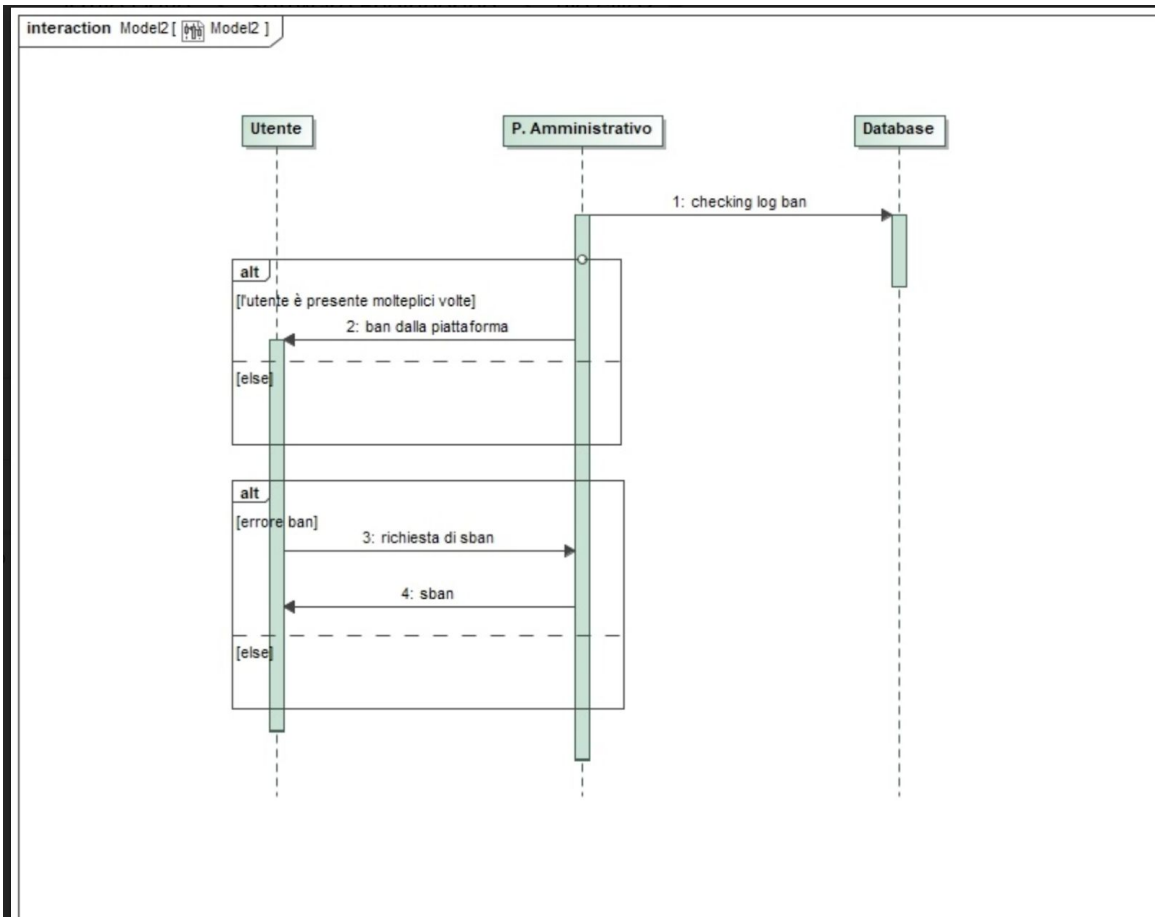
L'autenticazione avviene tramite l'invio delle credenziali dell'utente tramite l'interfaccia che, tramite il sistema collegato al database, verifica se le credenziali sono corrette, e comunica all'utente il risultato. Se le credenziali sono corrette, l'utente risulterà autenticato, altrimenti dovrà riprovare il login.

5.3.2 Verifica di un utente



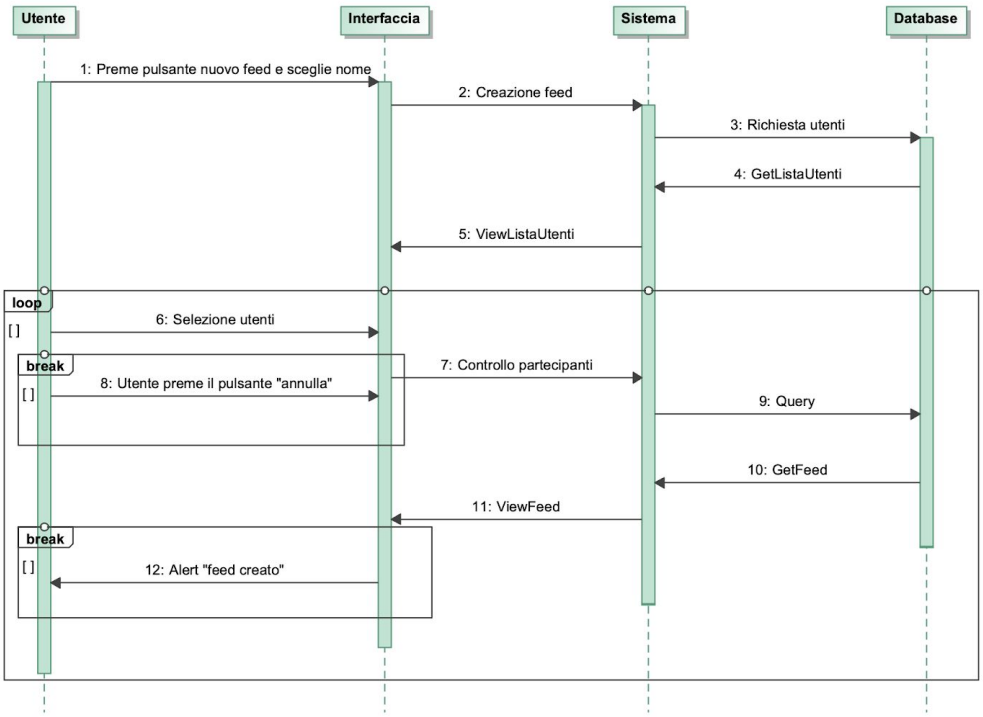
L'utente che vuole verificarsi invia i documenti al personale amministrativo che a sua volta li controlla nel sistema che fornisce le info richieste. A questo punto se le info corrispondono ai documenti inviati l'utente viene riconosciuto altrimenti la verifica viene annullata.

5.3.3 Ban manuale di un utente



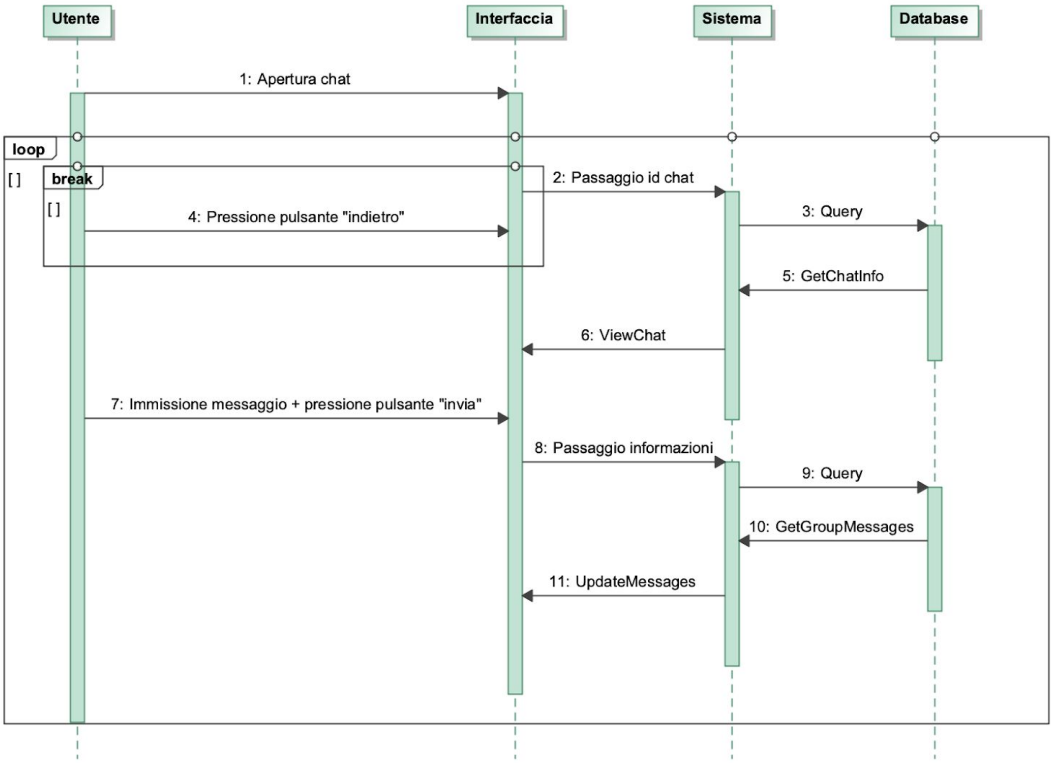
Qui il personale amministrativo controlla il log dei ban dei vari utenti; se un utente risulta presente per tre o più volte il personale amministrativo banna definitivamente l'utente dalla piattaforma.
Può comunque succedere di un errore da parte del personale di fatti è presente, tramite messaggio "richiesta di sban", l'opzione per contattare il personale amministrativo per farsi sbannare

5.3.4 Creazione di un feed



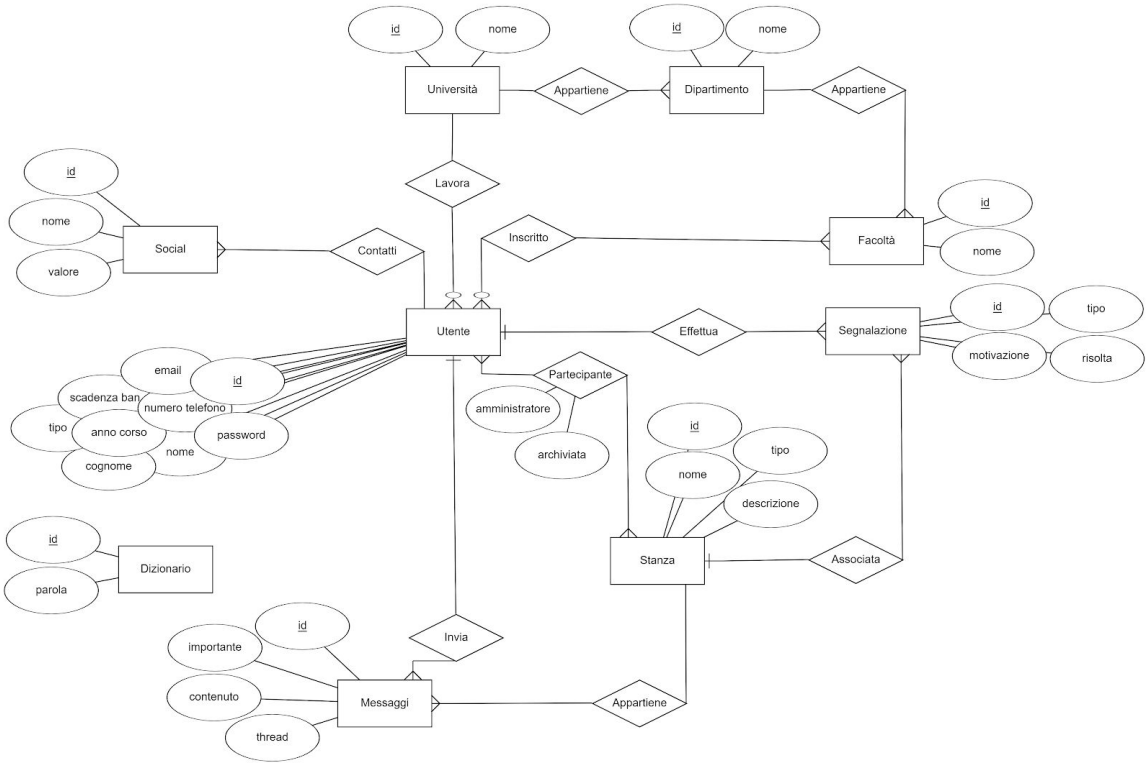
L'utente crea un feed premendo il pulsante "nuovo" e scegliendo un nome. L'interfaccia quindi crea il feed tramite il sistema, che a sua volta richiede gli utenti della stessa università tramite una query al database. In questo modo, l'utente può selezionare gli utenti che faranno parte di quel feed. Dopo la selezione degli utenti, viene creato il feed e l'utente riceve un alert.

5.3.5 Invio di un messaggio



L'invio di un messaggio avviene tramite l'apertura della chat, e successivamente tramite l'immissione del messaggio e la pressione del pulsante per l'invio da parte dell'utente. Il sistema richiama la query e aggiorna i messaggi recenti.

6. Dati e loro modellazione



Chiave primaria *Chiave esterna*

Dizionario (ID, parola)

Università (ID, nome)
Dipartimento (ID, nome, *università_id*)
Facoltà (ID, nome, *dipartimento_id*)

Utente (ID, nome, cognome, email, password, tipo, verificato, numero_di_telefono, scadenza_ban, *facoltà_id*, *università_id*)
L'utente, nel caso sia uno studente, è associato al suo corso di laurea, mentre se è un docente o fa parte del personale amministrativo, è associato all'università.
Tipo: studente=1 docente=2 personale amministrativo=3

Social (ID, nome, valore, *utente_id*)
Viene data totale autonomia all'utente dei canali da mostrare all'interno della piattaforma

Stanza (ID, tipo, nome, descrizione, *università_id*)
Tipo: chat diretta/gruppo/feed

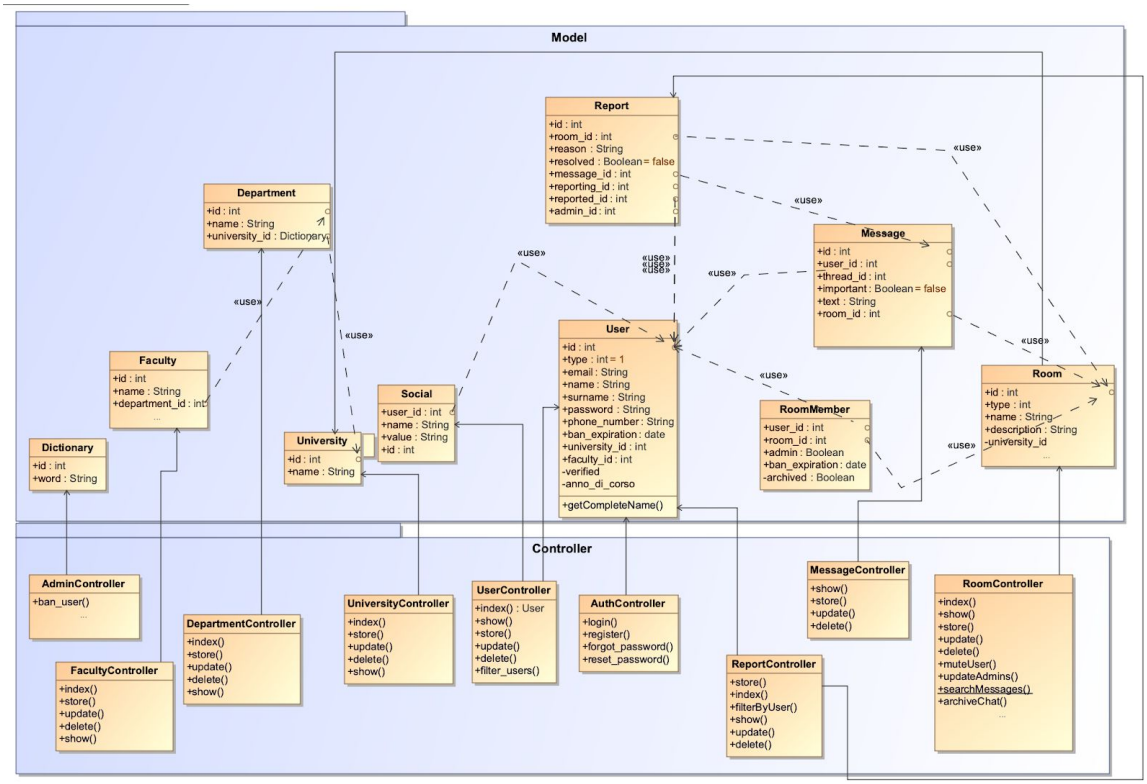
Messaggi (ID, contenuto, importante,thread, *stanza_id*, *utente_id*)
Partecipante(*utente_id*, *stanza_id*,)

Segnalazione (ID, motivazione, risolta,tipo, *utente_id*, *messaggio_id*)

7. Design Decisions

1. Gestione di chat diretta, gruppi e feed tramite un sistema di gestione a stanze. Questo permette di generalizzare i tre concetti in un'unica entità, riducendo le tempistiche
2. Autenticazione nel sistema tramite credenziali salvate nel database di UniChat, anziché l'utilizzo di servizi esterni come SPID o credenziali universitarie
3. La verifica degli utenti è manuale. In questo modo la verifica degli utenti è responsabilità del personale amministrativo, che si presume conosca la struttura universitaria, eliminando le problematiche legate ad una verifica di tipo automatica, che potrebbe verificare utenti malintenzionati.
4. Il ban automatico si basa sul dizionario, poiché è una soluzione efficace e non richiede molto tempo per la sua implementazione, mentre il ban manuale è un compito del personale amministrativo, il quale, avendo l'elenco di utenti, con eventualmente il numero di ban ottenuti, si occuperà del ban alla piattaforma
5. Per limitare lo spam esiste un sistema automatico, l'antiflood, che blocca l'invio dello stesso messaggio per più di tre volte nell'arco di un minuto

8. Design di Basso Livello



Il Class Diagram mostra la strutturazione di UniChat, evidenziando un pattern basato su modelli e controller. I modelli si occupano di definire le entità presenti nel sistema, mentre i controller gestiscono le operazioni effettuabili.

Le operazioni principali sono di tipo CRUD: create, read, update, delete, poiché sono le operazioni basilari per il funzionamento del software. Le altre operazioni degne di nota sono quelle presenti nella parte di autenticazione (AuthController) e di gestione del gruppo (mute_user e updateAdmins)

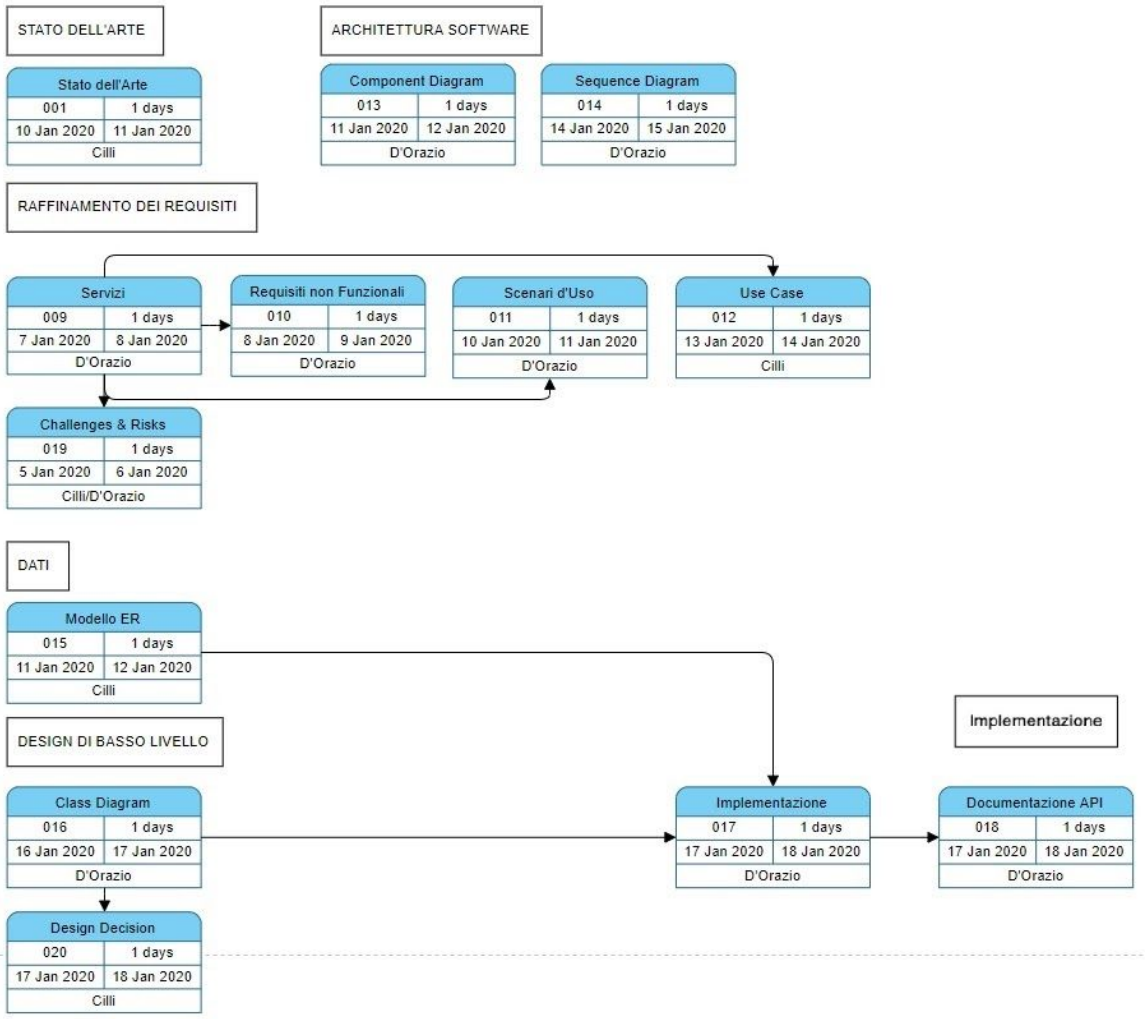
Sono stati aggiunte la funzione “archiveChat” e la variabile “archived” in “RoomMember”, per gestire l’archiviazione di una chat e l’attributo “anno_di_corso” a “User”

9. Explain how the FRs and the NFRs are satisfied by design

Le chat dirette, i gruppi e i feed sono gestiti tutti e tre come stanze dato che, in fondo, le operazioni che differiscono tra le tre tipologie di conversazione sono minime.
Nella chat diretta i messaggi vengono scambiati direttamente tra due persone; il gruppo invece è una conversazione con più utenti; infine il feed è una via di mezzo tra le due dove c'è un unico mittente e molti destinatari.
Affrontiamo adesso un argomento a tratti cruciale per il nostro software, il ban.
Abbiamo due tipi di ban, il primo automatico che si limita a silenziare l'utente per un periodo limitato; il secondo in vece manuale che prevede un ban a tempo indeterminato.
Il primo ban, da non confondere con il sistema di Antiflood (un sistema per limitare lo spam che annulla l'invio dello stesso messaggio per più di tre volte nell'arco di un minuto), è appunto un ban automatico gestito direttamente dal sistema, questo infatti, mediante l'uso di un dizionario delle parole proibite (scritto da un personale amministrativo o addirittura da uno sviluppatore) scannerizza ogni parola del messaggio da inviare e controlla se è presente uno o più termini presenti nel dizionario.
Appena ne trova uno il sistema non invia il messaggio e lo elimina contrassegnando l'utente; se l'utente viene contrassegnato tre volte viene silenziato per un giorno.
Il secondo ban è manuale, infatti direttamente il personale amministrativo controlla tramite un log quante volte un utente è stato silenziato; al terzo silenziamento si viene bannati dalla piattaforma.
È tuttavia possibile insorgere in qualche errore da parte del personale amministrativo; infatti l'utente può inviare una email cosi da permettere la correzione da parte dal personale amministrativo e rimuovere il ban.
Gli utenti si possono registrare fornendo nome, cognome, email, password, tipo, università e facoltà. La registrazione viene effettuata autonomamente dall'utente.
Tuttavia per registrarsi come docente o personale amministrativo bisogna essere verificati da un altro personale amministrativo fornendo i documenti alla registrazione.
Il personale competente della verifica degli utenti controllerà nell'università in questione e se i documenti risultano l'utente verrà verificato.

10. Effort Recording

10.1 PERT



Nonostante le tempistiche non fossero a nostro favore, ci siamo applicati al massimo per ottenere un software ingegnerizzato nel modo migliore possibile. Siamo soddisfatti di aver rispettato le tempistiche.

10.2 Logging

LogTemplate Milestone #2

Team (number and name): Flop team
Student name: Alessandro D'Orazio, Lorenzo Cilli
Student number: 251811, 253121
Email: alessandro.dorazio2@student.univaq.it, lorenzo.cilli@student.univaq.it

When		Time spent (min)	Partners	Brief Description of the performed task	Category	Sub-Category
12	30	15	AL	Call di allineamento	Doing	Studio del progetto
12	30	30	A	Organizzazione task per Milestone #2	Doing	Studio del progetto
1	5	15	AL	Challenging/Risky Requirements or Tasks	Doing	Challenging/Risky Requirements or Tasks
1	6	45	L	Stato dell'Arte	Doing	Stato dell'arte
1	7	45	A	Servizi	Doing	Raffinamento dei requisiti
1	8	120	A	Requisiti non funzionali	Doing	Raffinamento dei requisiti
1	9	15	A	Requisiti esclusi e assunzioni	Doing	Raffinamento dei requisiti
1	10	15	A	Scenari d'uso dettagliato	Doing	Raffinamento dei requisiti
1	11	120	A	Component Diagram	Doing	Architettura software
1	12	150	A	Sequence Diagram	Doing	Architettura software
1	13	90	L	Use-Case Diagram	Doing	Raffinamento dei requisiti
1	14	60	L	Modello ER	Doing	Dati e loro modellazione
1	15	15	A	Class Diagram	Doing	Design di basso livello
1	17	45	A	Modifiche API e lista routes	Doing	Implementazione
1	17	90	L	Design Decision	Doing	Design di basso livello
1	18	60	L	Studio Pert	Learning	Effort Recording
1	18	60	L	Pert Chart	Doing	Effort Recording
1	18	30	AL	Rifinitura documento	Doing	

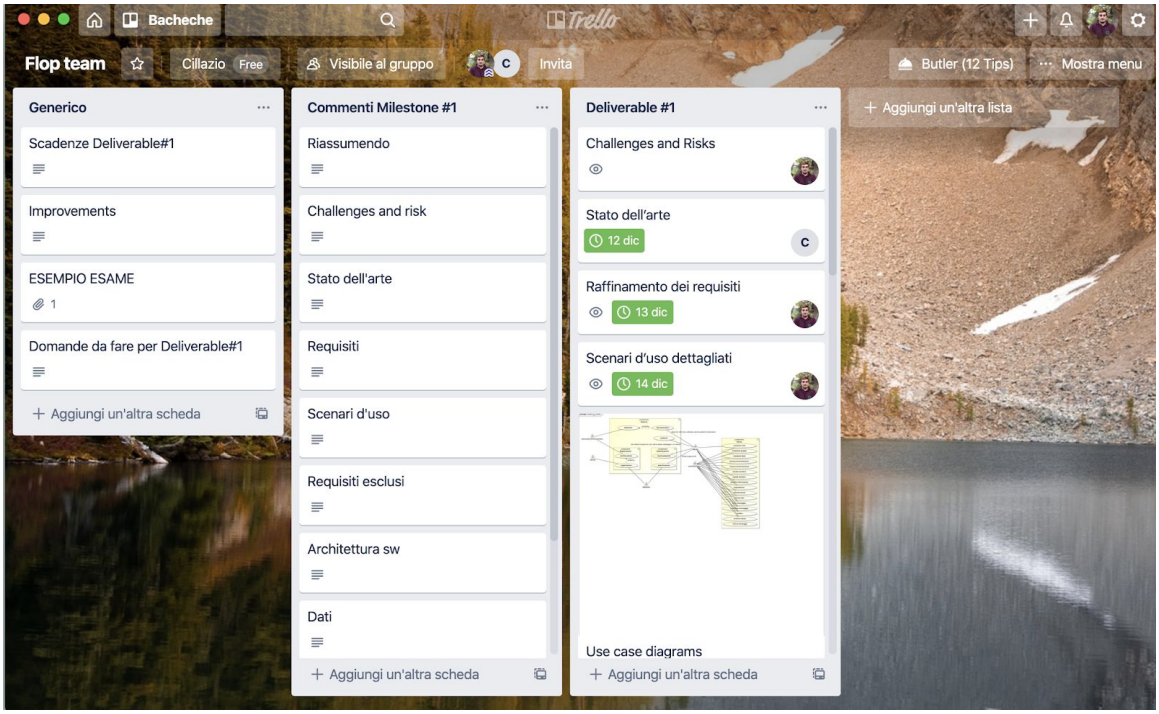
Ore Alessandro	9,3
Ore Lorenzo	6,8
Ore in comune	1,0
Ore totali	**18,0**
Ore Learning	1
Ore Doing	16

Ore Alessandro: 9.3
Ore Lorenzo: 6.8

Ore in comune: 1

Ore Totali: 18

L'organizzazione è avvenuta tramite chiamate Skype, incontri e la gestione di una bacheca su Trello.



11. Appendix. Prototype

Il prototipo, sviluppato con il framework Laravel, si basa su un pattern MVC-like. È presente sia la componente dei modelli che quella dei controller, ma non è stata effettuata un'implementazione delle viste.

Il prototipo segue la definizione dello schema del Class Diagram, includendo una classe PHP (sviluppata da Alessandro D'Orazio in un contesto lavorativo) che gestisce il ritorno dei dati in JSON, ottimizzando la scrittura del codice e le tempistiche grazie al riuso del codice.

Le operazioni permesse sono le CRUD (create, read, update, delete) per le seguenti entità: utenti, università, dipartimenti, facoltà, stanze e messaggi. I social vengono inseriti tramite l'update dell'utente.

La sicurezza viene garantita tramite JSON Web Token con scadenza di 1 ora e possibilità di refresh del token avendo il token precedente. Questo consente di non utilizzare le sessioni, ed offre un'elasticità maggiore rispetto ai metodi tradizionali per la gestione dell'autenticazione.

Quando si invia un messaggio, avviene un controllo dal server che, prima di inserire il messaggio nel db, verifica se l'utente è silenziato, oppure se ha utiliizzato una parola vietata.

Le API sono strutturate seguendo le linee guida di Laravel, utilizzando quindi un pattern {url}/api/{entity}/{action}

La fase di testing viene effettuata tramite le factory di Laravel, e una breve sessione di testing con casi limite. Abbiamo provato a caricare centinaia di migliaia di messaggi, contenuti in migliaia di stanze, senza ottenere problematiche di performance.

Nota per la documentazione mediante Postman
Poiché la preferenza del team è di sviluppare un prodotto creando i componenti, testandoli ed estendendoli, alcune funzionalità, come l'implementazione del Web Sockets non sono state introdotte. Riteniamo che la soluzione migliore sia di fornire dei componenti funzionanti al 100%, piuttosto che un componente sviluppato in poco tempo ma non funzionante.
Le API di un utente autenticato possiedono nell'header una regola "X-Authorization" contenente il token dell'utente autenticato. In questo modo è possibile sapere le informazioni dell'utente e gestire correttamente i permessi.

Sono presenti tutte le chiamate fornite da Laravel come resource (index, show, create, update, store, delete) per università, dipartimenti, corsi di laurea, utenti e stanze.
Sono stati oscurati gli id.

Documentazione mediante Postman
https://documenter.getpostman.com/view/8518915/SWLYAW1g?version=latest

Nella documentazione Postman non sono presenti tutte le API presenti nel sistema. Per motivi di tempo, non abbiamo potuto verificare la correttezza di ogni singola API.

Lista delle routes

METHOD	URI	NAME	ACTION
POST	api/auth/register	register	App\Http\Controllers\AuthController@register
GET	api/auth/login	login	App\Http\Controllers\AuthController@login
POST	api/auth/login	login	App\Http\Controllers\AuthController@login
POST	api/auth/logout	logout	App\Http\Controllers\AuthController@logout
GET	api/auth/info	info	App\Http\Controllers\AuthController@me
GET	api/users	users.index	App\Http\Controllers\UserController@index
POST	api/users	users.store	App\Http\Controllers\UserController@store
GET	api/users/{user}	users.show	App\Http\Controllers\UserController@show
PUT	api/users/{user}	users.update	App\Http\Controllers\UserController@update
POST	api/users/filter		App\Http\Controllers\UserController@filterUsers
GET	api/universities	universities.index	App\Http\Controllers\UniversityController@index
POST	api/universities	universities.store	App\Http\Controllers\UniversityController@store
GET	api/universities/{university}	universities.show	App\Http\Controllers\UniversityController@show
PUT	api/universities/{university}	universities.update	App\Http\Controllers\UniversityController@update
GET	api/departments	departments.index	App\Http\Controllers\DepartmentController@index
POST	api/departments	departments.store	App\Http\Controllers\DepartmentController@store
GET	api/departments/{department}	departments.show	App\Http\Controllers\DepartmentController@show
PUT	api/departments/{department}	departments.update	App\Http\Controllers\DepartmentController@update
GET	api/faculties	faculties.index	App\Http\Controllers\FacultyController@index
POST	api/faculties	faculties.store	App\Http\Controllers\FacultyController@store
GET	api/faculties/{faculty}	faculties.show	App\Http\Controllers\FacultyController@show
PUT	api/faculties/{faculty}	faculties.update	App\Http\Controllers\FacultyController@update
GET	api/rooms	rooms.index	App\Http\Controllers\RoomController@index
POST	api/rooms	rooms.store	App\Http\Controllers\RoomController@store
GET	api/rooms/{room}	rooms.show	App\Http\Controllers\RoomController@show
PUT	api/rooms/{room}	rooms.update	App\Http\Controllers\RoomController@update
POST	api/rooms/{room_id}/user/{user_id}/mute	rooms.mute_user	App\Http\Controllers\RoomController@muteUser
POST	api/rooms/{room_id}/admins/update	rooms.update_admins	App\Http\Controllers\RoomController@updateAdmins
POST	api/rooms/find	rooms.find	App\Http\Controllers\RoomController@find
POST	api/rooms/{room_id}/messages	messages.store	App\Http\Controllers\MessageController@store
GET	api/rooms/{room_id}/messages/{message}	messages.show	App\Http\Controllers\MessageController@show
PUT	api/rooms/{room_id}/messages/{message}	messages.update	App\Http\Controllers\MessageController@update
POST	api/admin/ban	admin.ban_user	App\Http\Controllers\AdminController@ban_user
GET	api/reports	reports.index	App\Http\Controllers\ReportController@index
POST	api/reports	reports.store	App\Http\Controllers\ReportController@store
GET	api/reports/{report}	reports.show	App\Http\Controllers\ReportController@show
PUT	api/reports/{report}	reports.update	App\Http\Controllers\ReportController@update
GET	api/reports/user/{user_id}/all	reports.filter_by_user	App\Http\Controllers\ReportController@filterByUser