|  |  |
| --- | --- |
| Comprensione del comportamento scorretto nella discussione di COVID19 sui social network e sui media online | |
|  | |
| Studente/i | Relatore |
| Nogara Gianluca | Giordano Cremonese Silvia |
| Correlatore |
| Luceri Luca |
| Committente |
| - |
| Corso di laurea | Modulo |
| Ingegneria Informatica TP | C10344 Progetto di diploma |
| Anno |  |
| 2020/2021 |  |

|  |  |
| --- | --- |
| Data |  |
| - |  |

**Indice**

[Abstract 7](#_Toc71295617)

[Progetto assegnato 9](#_Toc71295618)

[1 Analisi 11](#_Toc71295619)

[1.1 Contesto 11](#_Toc71295620)

[1.2 Fruitori 11](#_Toc71295621)

[1.3 Analisi delle tecnologie 11](#_Toc71295622)

[1.4 Requisiti 12](#_Toc71295623)

[1.4.1 Contenuti 12](#_Toc71295624)

[1.4.2 Funzionalità 12](#_Toc71295625)

[1.4.3 Rispetto della privacy 13](#_Toc71295626)

[1.5 Obiettivi 13](#_Toc71295627)

[2 Studio delle Soluzioni 14](#_Toc71295628)

[2.1 La nostra scelta 14](#_Toc71295629)

[2.2 Moodle plugin 14](#_Toc71295630)

[2.3 L’alternativa 14](#_Toc71295631)

[2.4 Learning Locker 15](#_Toc71295632)

[2.5 Ambienti di sviluppo e linguaggi 16](#_Toc71295633)

[2.5.1 Anaconda e il linguaggio Python 17](#_Toc71295634)

[2.5.2 MariaDB 17](#_Toc71295635)

[2.5.3 Bash 17](#_Toc71295636)

[2.5.4 Postman 17](#_Toc71295637)

[2.6 Strumenti utilizzati 18](#_Toc71295638)

[2.6.1 VMware 18](#_Toc71295639)

[2.6.2 PuTTY 18](#_Toc71295640)

[3 Installazione 19](#_Toc71295641)

[3.1 Installazione custom 19](#_Toc71295642)

[3.1.1 Problemi di versione 20](#_Toc71295643)

[3.2 installazione via script 20](#_Toc71295644)

[3.2.1 problemi MongoDB 21](#_Toc71295645)

[4 Operazioni sul LRS ed API 22](#_Toc71295646)

[4.1 customQuery.py 22](#_Toc71295647)

[4.2 LLSender.py 24](#_Toc71295648)

[4.3 MSTeams 26](#_Toc71295649)

[5 Piani di lavoro 28](#_Toc71295650)

[5.1 Piattaforma SCM 28](#_Toc71295651)

[5.2 Repository 28](#_Toc71295652)

[6 Conclusioni 29](#_Toc71295653)

[6.1 Problemi riscontrati 29](#_Toc71295654)

[6.2 Implementazioni future 30](#_Toc71295655)

[7 Fonti 31](#_Toc71295656)

**Indice delle figure**

[Figura 1: Esempio di uno Statement xAPI 12](#_Toc71295657)

[Figura 2: Esempio di dati anonimizzati nel database di Moodle 13](file:///\\win1.dti.supsi.ch\ProgettiDiStudio2021\ProgettoSem\Info\Moro%20-%20Nogara\Relazione.docx#_Toc71295658)

[Figura 3: Pagina di login di LL 15](#_Toc71295659)

[Figura 4: Pagina principale di LL, sezione Data > Dashboards 16](#_Toc71295660)

[Figura 5: Configurazione PuTTY 18](#_Toc71295661)

[Figura 6: Stato dei servizi di LL dopo l'installazione via script 20](#_Toc71295662)

[Figura 7: Esempio di una post tramite Postman 21](#_Toc71295663)

[Figura 8: Schema della raccolta e normalizzazione dei dati 22](#_Toc71295664)

[Figura 9: Schema del fetiching e inserimento in Learning Locker 24](#_Toc71295665)

[Figura 10: Risultato dell'inserimento visto all'interno di Learning Locker 26](#_Toc71295666)

[Figura 11: Esempio di dati prima del parsing 27](#_Toc71295667)

[Figura 12: Esempio di dati dopo il parsing 27](#_Toc71295668)

[Figura 13: Organizzazione del lavoro 28](#_Toc71295669)

[Figura 14: Sprint Board iterazione-05 29](#_Toc71295670)

Abstract

I social network e i media online (OSNEM) sono tecnologie basate su computer che consentono agli utenti di creare contenuti e intrattenere relazioni sociali.

Le OSNEM sono rapidamente cresciute dall’essere semplicemente un canale di aggregazione all’essere un fenomeno globale che ha suscitato un cambio di paradigma della nostra società, trasformando il modo in cui le persone accedono alle notizie, condividono opinioni, fanno affari e fanno politica.

In uno scenario del genere, l'accuratezza, la veridicità e l'autenticità del contenuto condiviso sono ingredienti necessari per mantenere una sana discussione online.

Tuttavia, negli ultimi tempi, le OSNEM hanno dovuto affrontare una notevole crescita di account e attività dannose, che minano intenzionalmente l'integrità delle conversazioni online condividendo, nelle piattaforme OSNEM, informazioni false e provocatorie per influenzare l'opinione pubblica e creare conflitti su questioni sociali o politiche.

Ciò è stato, e continua ad essere, drammaticamente vero per la discussione sul COVID19, che è stata minata da campagne di manipolazione e disinformazione a livello globale. È quindi necessario studiare i comportamenti peculiari delle entità dannose nel dibattito COVID19 per fornire un cambiamento sociale verso una migliore comprensione (alfabetizzazione) delle reti sociali.

Questo include il rilevamento di malintenzionati, l'identificazione di informazioni di scarsa credibilità e notizie false, la consapevolezza degli utenti e il controllo dei comportamenti scorretti.

Social networks and online media (OSNEMs) are computer-based technologies that allow users to create content and engage in social relationships.

OSNEMs have rapidly grown from being simply a channel of aggregation to being a global phenomenon that has sparked a paradigm shift in our society, transforming the way people access news, share opinions, do business, and make policy.

In such a scenario, accuracy, truthfulness and authenticity of shared content are necessary ingredients to maintain a healthy online discussion.

However, in recent times, OSNEMs have faced a significant growth of malicious accounts and activities, which intentionally undermine the integrity of online conversations by sharing, in OSNEM platforms, false and provocative information to influence public opinion and create conflict on social or political issues.

This has been, and continues to be, dramatically true of the COVID19 discussion, which has been undermined by global campaigns of manipulation and misinformation. Therefore, it is necessary to study the peculiar behaviors of malicious entities in the COVID19 debate to provide social change towards a better understanding (literacy) of social networks.

This includes detection of malicious actors, identification of information of low credibility and fake news, user awareness, and control of misbehavior.

Progetto assegnato

Lo scopo di questo progetto è quello di analizzare entità malevoli presenti in un dataset pubblico su COVID19. L’ OSNEM di riferimento è Twitter e l’attività di analisi e classificazione delle entità malevoli (come bot e trolls) viene fatta tramite diversi algoritmi esistenti.

Il progetto in questione consente quindi di analizzare il comportamento all'interno della discussione e l’impatto che hanno tali entità malevole nella discussione globale.

**COMPITI**

•

•

•

**OBIETTIVI**

L'obiettivo principale di questa tesi è quello di arrivare a distinguere tra entità/azioni corrette e quelle malevoli nel dibattito COVID19.

Questo include il rilevamento di malintenzionati (ad esempio bot e troll), l'identificazione di informazioni di scarsa credibilità e notizie false, la consapevolezza degli utenti e il controllo dei comportamenti scorretti.

**TECNOLOGIE**

• Python

• Complex networks

# 1 Analisi

## 1.1 Contesto

Il progetto si inserisce nel contesto del Dipartimento Tecnologie Innovative della SUPSI e in particolare nel corso di Ingegneria Informatica. Si tratta di un progetto interno alla SUPSI, anche se di grande importanza sociale dal momento che le informazioni raccolte possono essere utilizzate per limitare la propagazione di entità malevoli sui social network.

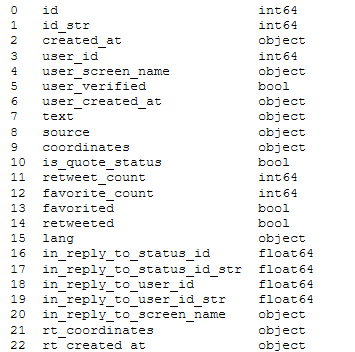
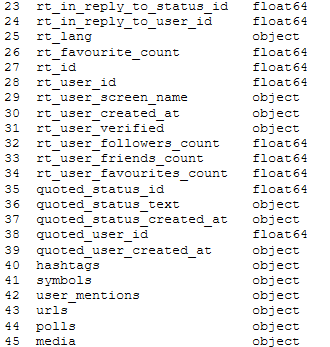
## 1.2 Fruitori

## 1.3 Analisi delle tecnologie

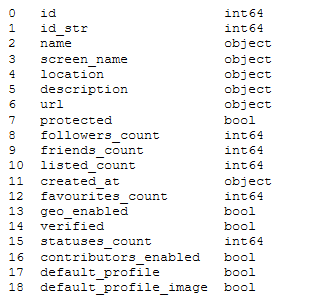
Il primo compito assegnato riguardava l’analisi di un sample dei dati (di soli 50.000 campioni) in modo da poter identificare la struttura del dataset e operare in modo corretto.

I dataset da utilizzare sono di due tipi:

Tweets



E users



## 1.4 Requisiti

### 1.4.1 Contenuti

Il principale requisito a livello contenutisco è quello di avere dei feedback visivi dell’analisi effettuata sui dati, in modo da effettuare operazioni di ulteriore processamento realizzando delle …

### 1.4.2 Funzionalità

Per quando riguarda i requisiti legati all’usabilità del sistema sviluppato possiamo identificare come utente finale sia un’organizzazione che un singolo individuo. Infatti, essendo il progetto open source e completamente personalizzabile, è possibile implementarne funzionalità ad-hoc.

### 1.4.3 Rispetto della privacy

Dal momento che i dati utilizzati dal LRS sono informazioni di studenti e docenti si è deciso di anonimizzare i dati sensibili, come il nome e il cognome di docenti e studenti. Questo ha portato ad avere delle informazioni aggregate utilizzabili ma totalmente privacy-friendly, come riportato nella Figura 2.

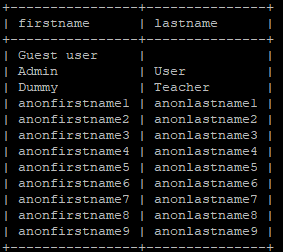


Figura 2: Esempio di dati anonimizzati nel database di Moodle

## 1.5 Obiettivi

Gli obiettivi di questo progetto sono i seguenti:

* Installare e configurare un LRS interagibile su una macchina Unix
* Accedere ai dati di una piattaforma Moodle di prova con dei dati anonimizzati
* Accedere ai dati tool di videoconferenza (p.e. MS Teams)
* Implementare delle API per estrarre i dati e importarli nel LRS
* Visualizzare i dati tramite un tool di visualizzazione commerciale (p.e. Tableau)

# 2 Studio delle Soluzioni

## 2.1 La nostra scelta

La scelta dell’utilizzo di uno standard rispetto ad un altro ha portato via diverse ore in quanto scelta chiave per l’implementazione del progetto.

Per il progetto lo standard utilizzato è stato xAPI, che come anticipato è utilizzato da Learning Locker, un Learning Record Store (LRS) open source iniziato nel 2013 da HT2 Labs (ora Learning Pool) che altro non è che un repository progettato per memorizzare statements di attività di apprendimento generate da differenti sorgenti (quali Moodle, MS Teams ecc…).

La documentazione online è stata trovata piuttosto chiara ed esaustiva, infatti da una prima analisi possiamo notare come tutte le informazioni di cui abbiamo bisogno siano sul sito di Learning Locker (<https://docs.learninglocker.net/welcome/>).

La chiarezza e la presenza di differenti informazioni su Internet ci hanno convinti ad optare per questa soluzione, affidandoci anche alla garanzia di lavorare sull’LRS più installato al mondo.

## 2.2 Moodle plugin

Per automatizzare la raccolta e l’inserimento dei dati all’interno di Learning Locker dal database di Moodle, si è utilizzato un plugin.

Il plugin in questione è stato configurato per inserire automaticamente qualasisi nuovo dato che venisse inserito nel database di prova, in modo da popolare una prima volta il LRS e mantenerlo aggiornato ai dati di Moodle.

## 2.3 L’alternativa

L’altro standard studiato in fase di analisi è stato Caliper, usato da Callisto.

Come già accennato è molto simile a Learning Locker (LL) e ciò ci ha spinti a valutare la documentazione e le informazioni su Internet come parte essenziale per la nostra scelta, possiamo dare uno sguardo al proprio sito web per notare come sia ricco di informazioni (<https://www.imsglobal.org/activity/caliper>).

Purtroppo, la documentazione seppur chiara non è stata abbastanza soddisfacente da convincerci e navigando fuori dal sito non abbiamo trovato molte informazioni, pertanto questa scelta è stata scartata.

## 2.4 Learning Locker

LL è stato installato su una macchina della SUPSI con sistema operativo Ubuntu 18.04. Questa macchina è stata resa disponibile per il progetto dal momento che il servizio dev’essere accessibile per gli studenti che si possono collegare alla rete universitaria tramite VPN.

È possibile accedere al servizio tramite un qualsiasi browser all’indirizzo <http://lrs.supsi.ch/login>, sarà necessario effettuare un login per poter accedere alle funzionalità.

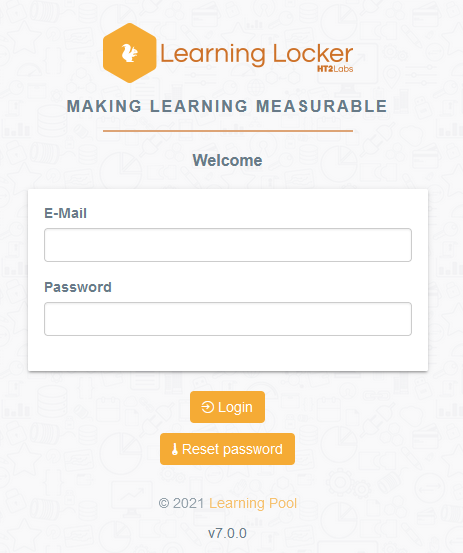


Figura 3: Pagina di login di LL

Una volta entrati, dopo aver scelto la nostra organizzazione (SUPSI), si ottiene la pagina principale, composta da un menù sulla sinistra composto da una serie di sezioni in cui è possibile navigare:

* Data
* People
* Settings

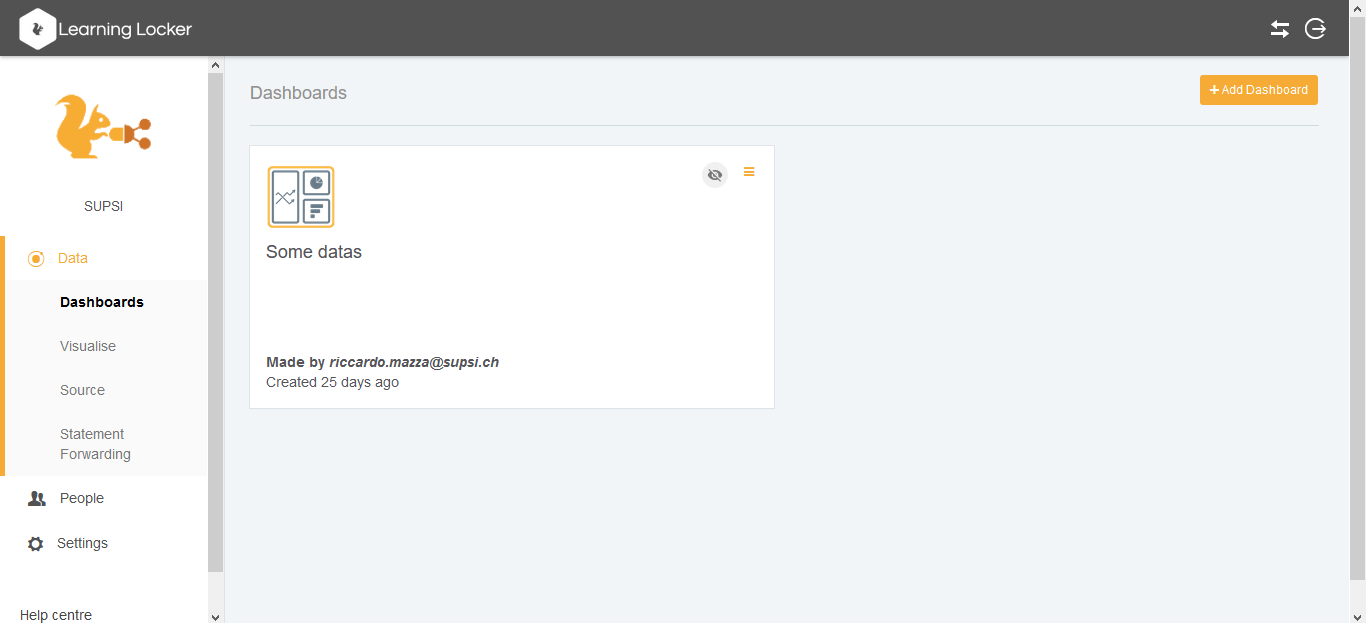
****

Figura 4: Pagina principale di LL, sezione Data > Dashboards

Nella sezione Data possiamo andare a gestire i dati che sono stati inseriti, possiamo visualizzare graficamente le attività (in Dashboard) gestire i dati per creare grafici (in Visualise), vedere la struttura delle statements inserite (in Source) e inserire degli statements (in Statements Forwarding).

Nella sezione People possiamo gestire le persone presenti in LL aggiungendone, eliminandole ed importandole.

La sezione Settings ci permette di gestire la parte logica del nostro repository, infatti per poter funzionare abbiamo bisogno di un Client (endpoint xAPI) su cui poter eseguire le POST e uno Store collegato per salvarle.

Vi è inoltre la possibilità di gestire l’utente con cui ci siamo loggati, i ruoli e le API esistenti.

## 2.5 Ambienti di sviluppo e linguaggi

Dal momento che tra gli obiettivi figuravano lo sviluppo di API, vi era la necessità di utilizzare un linguaggio flessibile che potesse gestire diversi tipi di dati in modo semplice e veloce, interagendo inoltre con endpoint e database.

La scelta di un linguaggio che rispettasse i requisiti sopra citati ha orientato l’attenzione verso Python.

### 2.5.1 Anaconda e il linguaggio Python

Anaconda ha svolto un compito molto importante per quanto riguarda la programmazione in Python, infatti, dal momento che per lo sviluppo delle API vi è l’esigenza di poter effettuare operazioni di mapping, andando a sostituire dei placeholder da un file di sorgente con dei dati custom.

Ad esempio, ci sono API che richiedono di effettuare delle query personalizzabili dagli utenti verso un database, richiedendo quindi di inserire dei dati in un file

Uno strumento essenziale è stato il Notebook, che ha permesso di gestire in modo ottimale i programmi, garantendo una buona gestione del codice.

### 2.5.2 MariaDB

MariaDB è un DBMS, non è altro che una fork di MySQL ed è stato largamente utilizzato per poter accedere ai dati anonimizzati caricati sulla piattaforma Moodle di prova.

L’esigenza dell’utilizzo di tale database è data dal momento che è stato utilizzato un database di Moodle di prova, basato su MariaDB, che conteneva dati anonimizzati di prova che andavano importati sul LRS implementato nelle fasi precedenti.  
Il database è stato esplorato per poterne capire la struttura e tipi di dati contenuti, inoltre abbiamo potuto effettuare delle query per poter testare l’efficacia dei programmi realizzati. Per l’accesso a MariaDB stato effettuato uno script in modo da poter entrare senza dover esplicitamente scrivere differenti linee di codice ogni volta.

### 2.5.3 Bash

Come anticipato sono stati effettuati degli script in Bash per funzionalità legate al progetto: il primo con il compito di automatizzare l’accesso a MariaDB e il secondo con l’obiettivo di effettuare delle query ogni delta di tempo impostato.

### 2.5.4 Postman

Per l’inserimento o leggere dei dati si è utilizzato un semplicissimo software che permette di effettuare richieste http, il suo utilizzo si è poi rilevato molto importante anche per l’implementazione di API via Python.

## 2.6 Strumenti utilizzati

Per lavorare al LRS è stato necessario organizzare il lavoro in modo da poter accedere a una singola macchina in remoto accessibile da entrambi gli studenti.

Prima di accedere alla macchina in remoto, per evitare qualsiasi rischio è stato svolto un lavoro di implementazione in locale usando delle macchine virtuali.

### 2.6.1 VMware

Come anticipato si è adottata una strategia di contenimento dei danni, sono state sfruttate al meglio le caratteristiche delle macchine virtuali: garantiscono un ambiente completamente isolato e ripristinabile, quindi si è testata l’installazione del LRS prima in un ambiente virtuale, scelta che si è rivelata molto importante.

### 2.6.2 PuTTY

Dovendo operare su una macchina in remoto si è usato un client con la funzione di emulatore di terminale per accedere tramite della macchina.

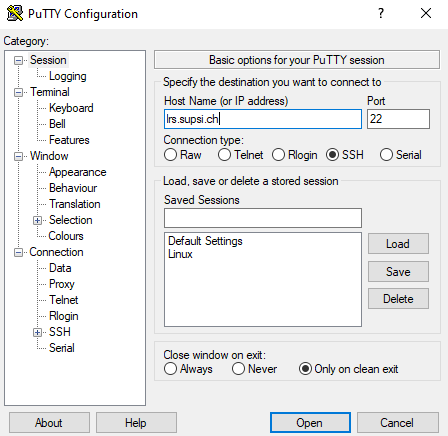


Figura 5: Configurazione PuTTY

L’accesso viene fatto tramite SSH all’indirizzo lrs.supsi.ch, con porta 22.

# 3 Installazione

## 3.1 Installazione custom

Come stabilito in accordo con il Docente il primo tentativo di installazione di Learning Locker è stato fatto attraverso “l’installazione custom”, ovvero l’installazione manuale di ogni singolo pacchetto da linea di comando della macchina messa a disposizione.

Sono stati annotati tutti i passaggi effettuati per poter ricostruire uno storico delle operazioni e, nel caso di errori, poterne comprendere il motivo e risolverli.

L’installazione è stata effettuata seguendo quanto riportato dalla documentazione ufficiale di Learning Locker (<https://docs.learninglocker.net/guides-custom-installation/>), la conoscenza dell’ambiente Unix ci ha permesso di poter ovviare a diversi problemi relativi ai permessi. Di seguito la lista di comandi utilizzati per l’installazione:

* sudo apt-get update && apt-get upgrade
* sudo apt-get -y install curl git python build-essential xvfb apt-transport-https
* sudo curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.37.2/install.sh | bash
* sudo apt install npm
* sudo npm install -g yarn
* sudo npm install -g pm2
* pm2 install pm2-logrotate
* pm2 set pm2-logrotate:compress true
* sudo git clone https://github.com/LearningLocker/learninglocker.git
* sudo npm\_config\_build\_from\_source=true yarn install --ignore-engines
* yarn build all
* Copia di .env.example in .env
* yarn migrate
* sudo git clone https://github.com/LearningLocker/xapi-service.git
* Dentro la directory > yarn install --ignore-engines
* yarn build
* Copia di .env.example in .env
* Dentro la dir di learning locker: pm2 start pm2/all.json
* Dentro la dir di xapi: pm2 start pm2/xapi.json

### 3.1.1 Problemi di versione

Purtroppo, l’installazione custom non è andata a buon fine, mostrando tutti i limiti di Learning Locker, dal momento che questa attività ha generato differenti errori generici senza poter individuare la vera fonte del problema.

Dopo una lunga ricerca online è stata trovata l’origine dei problemi: la versione!

Pare infatti che l’installazione di Learning Locker sia supportata solo da Ubuntu 18.04 e Ubuntu 16.04, mentre la macchina in uso montava una versione di Ubuntu 20.04.

## 3.2 installazione via script

Una volta riscontrato il problema di versioning del sistema operativo si è deciso di optare per un’installazione più semplice: sulla documentazione di Learning Locker è possibile trovare l’installazione via script (<https://docs.learninglocker.net/guides-installing/>), che semplifica nettamente la vita di chi se ne occupa.

È bastato infatti lanciare uno script bash che in modo automatizzato si è occupato di installare tutte le dipendenze e pacchetti necessari, dando la possibilità di poter configurare l’ambiente.

La riga è la seguente:

curl -o- -L https://raw.githubusercontent.com/LearningLocker/deploy/master/deployll.sh > deployll.sh && bash deployll.sh

L’installazione è andata a buon fine in questo caso.

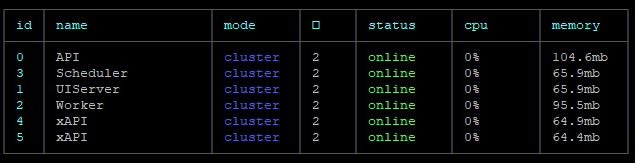


Figura 6: Stato dei servizi di LL dopo l'installazione via script

### 3.2.1 problemi MongoDB

Una volta installato correttamente LL la prima attività da svolgere era quella relativa all’inserimento dei dati.

Tramite l’utilissima Platform Postman si è cercato di effettuare delle post per inserire i dati, tuttavia non è stato possibile.

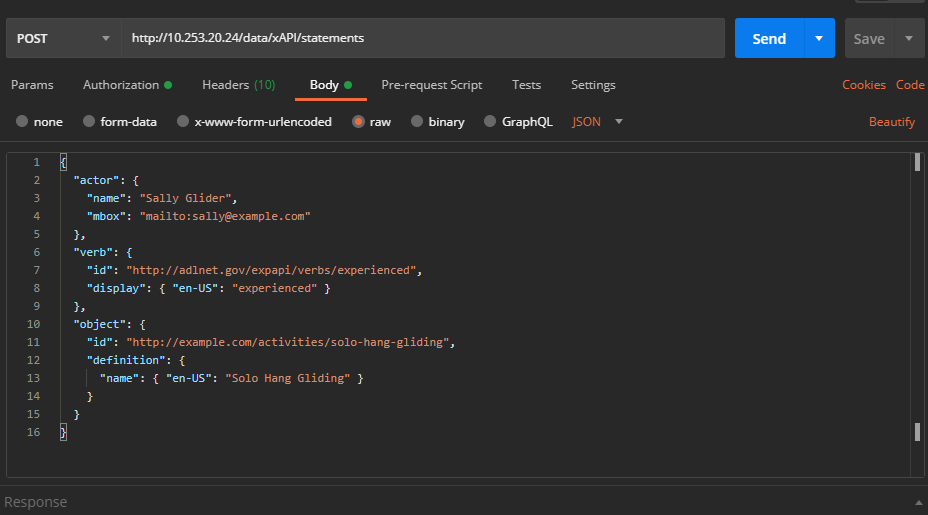
****

Figura 7: Esempio di una post tramite Postman

A seguito di un’analisi dei log è stato possibile trovare un errore nel file di configurazione /learninglocker/current/xapi/.env.

Un’errata configurazione dello script ha fatto in modo che MongoDB desse problemi: MONGO\_URL=mongodb://localhost:27017/learninglocker\_v2?replicaSet=rs0 è stato settato in modo errato e MongoDB non aveva replica, è bastato modificare il parametro nel seguente modo:

MONGO\_URL=mongodb://localhost:27017/learninglocker\_v2.

# 4 Operazioni sul LRS ed API

## 4.1 customQuery.py

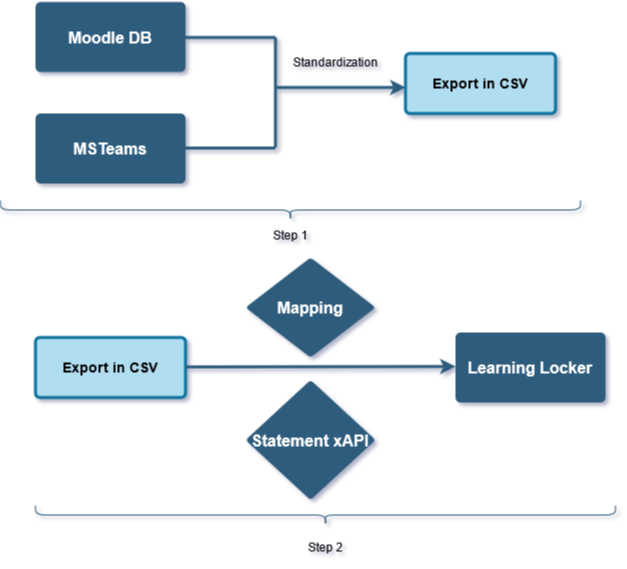
****

Figura 8: Schema della raccolta e normalizzazione dei dati

Dal momento che vi è la necessità di accedere al database contenente i dati di prova di Moodle e di doverli caricare sul nostro LRS, la soluzione migliore è stata quella di automatizzare il più possibile l’acquisizione dei dati.

Per poter effettuare una query customizzata è stata realizzare un’API in Python il cui utilizzo ha diverse configurazioni, come da esempio:

./customQuery.py --query <fileNameQuery> --csv <outputName> --queryParams <params.csv> --mapping <mapping.json>

I parametri hanno le seguenti funzioni:

* fileNameQuery è il nome del file che contiene la query da effettuare, contenente i placeholder che verranno sostituiti in funzione a quello che cerchiamo
* outputName è il nome del file che verrà generato, che conterrà i risultati delle interrogazioni al database
* params.csv è il nome del file che contiene i parametri che sostituiranno i placeholder nella query, la lunghezza di questo file determina il numero di query eseguite
* mapping.json è il nome del file che associa i placeholder (contenuti nel file fileNameQuery) ai parametri personalizzabili (contenuti nel file params.csv)

CustomQuery si occuperà quindi di interrogare il database molteplici volte secondo le regole contenute nel file <fileNameQuery>, esportando il risultato nella directory output nel file <outputName>, con formato csv.

Al fine di poter rieseguire la stessa query con differenti parametri nasce la necessità di passare un file (<params.csv>) e il corrispettivo mapping (<mapping.json>),dando la possibilità di specificare quale valore del csv corrisponde al parametro della query.

Un esempio di utilizzo potrebbe essere, data la seguente query:

L’obiettivo è quello di estrapolare l’identificativo di un utente per un determinato quiz con la relativa valutazione.

Per fare ciò dobbiamo effettuare una JOIN sulle due tabelle del database di Moodle “mdl\_quiz\_grades” e “mdl\_user”.

I placeholder che verranno sostituiti con i parametri che ci interessano sono racchiusi da parentesi graffe e scritti in maiuscolo (in questo caso sono {QUIZ\_ID} e {USER\_ID}).

I valori che verranno messi al posto dei placeholder andaranno specificati in un file aggiuntivo (<params.csv>), la cui struttura è la seugente:



Per associare i placeholder ai dati sopracitati, specifichiamo quanto vogliamo sostituire all’interno del file di mapping (<mapping.json>), andando ad assegnare l’header del file contenete i parametri con il replativo placeholder contenuto all’interno del file che contiene la query <fileNameQuery>). La struttura del file di questo esempio è le seugente:

L’output del comando ./customQuery.py --query query.sql --csv queryResults --queryParams queryParams.csv --mapping mappingQury.json è il seguente:



Per mantenere orgnizzato il repository durante lo svolgimento del progetto è stata creata una gerarichia di directory così strutturata:

* **mappings** 🡪 contiene i file che servono per effettuare il mapping
* **outputs** 🡪 contiene i file generati dagli script
* **params** 🡪 contiene i file in cui specifichiamo i parametri che sostituiranno i **placeholder**
* **queries** 🡪 contiene i file .sql che verranno richiamati per eseguire le query

In base alla tipologia di file che andiamo ad utilizzare dobbiamo porlo correttamente nella directory destinata a quel determinato utilizzo, in modo da non creare un ambiente poco chiaro e confuso.

## 4.2 LLSender.py

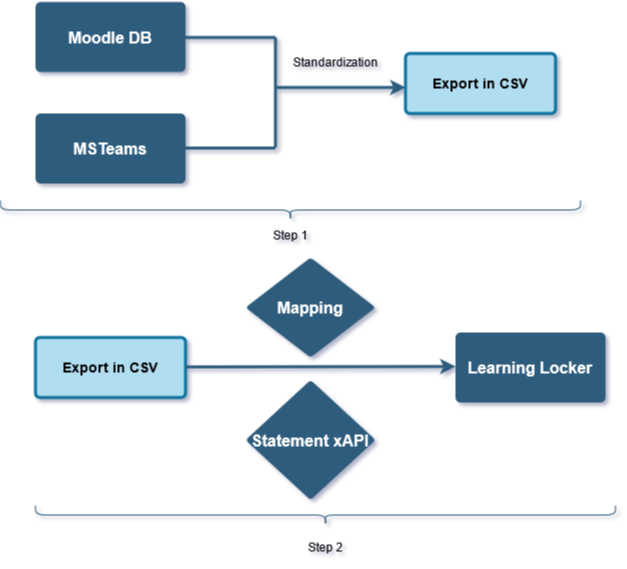
****

Figura 9: Schema del fetiching e inserimento in Learning Locker

Per poter inserire i dati all’interno del nostro LRS, è stato sviluppato uno script in Python in grado di effettuare una POST con dei dati parametrizzabili.

L’esecuzione dello script vuole diversi parametri, come da esempio:

./LLSender.py --source <sourceFile.csv> --mapping <mappingFile.json> --xapi <parameterizedStatement.json>"

La logica dello script è quella di prendere i dati da inserire dal file <sourceFile.csv>, tramite i parametri che sono all’interno del file <parameterizedStatement.json> si effettua un mapping dal file <mappingFile.json>.

Ad esempio, dato il seguente file dei parametri con il relativo mapping:





E uno statement configurabile:



Consente di inserire un qualsiasi utente specificato nei parametri all’interno di un qualsiasi corso anch’esso specificato.

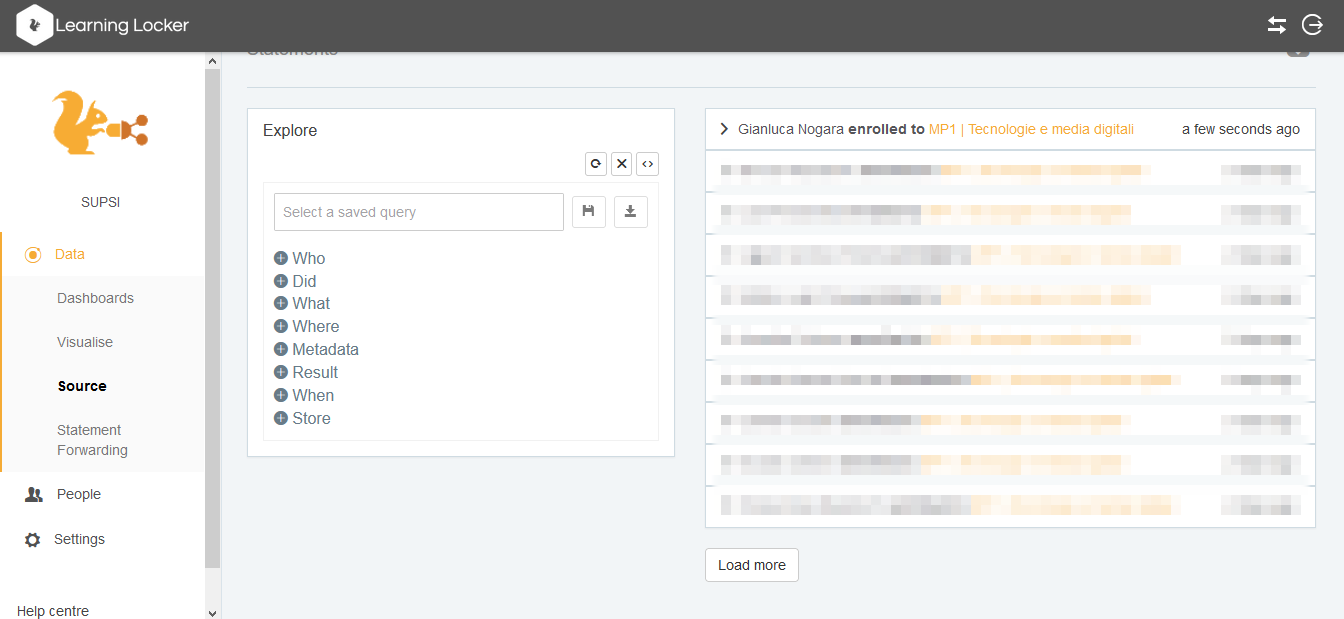


Figura 10: Risultato dell'inserimento visto all'interno di Learning Locker

Il cuore pulsante del programma riguarda il replace dei parametri, che avviene modificando il contenuto dello statement con i dati del file di source:



## 4.3 MSTeams

Dal momento che l’obiettivo del progetto riguardava l’utilizzo di dati da piattaforme di E-Leanring, come ad esempio MSTeams.

Sono stati forniti dei report di alcuni corsi svolti su Teams, la cui struttra è la seguente:

* Digital activity 🡪 che tiene traccia delle presenze
* Summary 🡪 che contiene assignment e valutazioni
* Assignment 🡪 che tiene traccia dei dati riguardanti i compiti assegnati
* Communication 🡪 che contiene le interazioni degli utenti

Il lavoro svolto è stato l’inserimento all’interno del LRS i dati contenuti in “Digital activity” e “Communication” dal momento che i restati report erano vuoti.

Per lo svolgimento di questo lavoro è stato effettuato un parsing dei file in quanto la struttura di questi dati era piuttosto caotica.

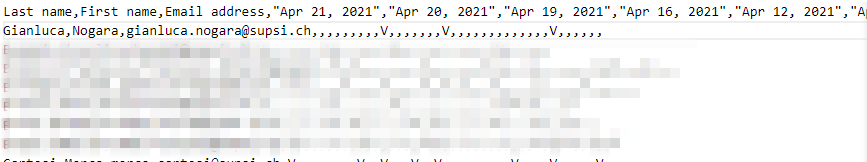


Figura 11: Esempio di dati prima del parsing

Il risultato è stato raggiunto ottenendo dati con una struttura molto più chiara e utilizzabile per le API che sono state sviluppate.

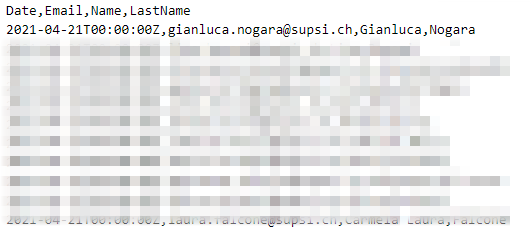


Figura 12: Esempio di dati dopo il parsing

# 5 Piani di lavoro

## 5.1 Piattaforma SCM

Nei requisiti iniziali del relatore, era stato richiesto l’utilizzo di SCRUM (SCM) per lo sviluppo del progetto. Abbiamo organizzato la pianificazione del lavoro in diversi Sprint, in modo tale da tenere traccia delle attività svolte e mantenere l’organizzazione nelle operazioni

Sono stati effettuati sette Sprint, di seguito riportiamo le Issues con i relativi iterazioni.



Figura 13: Organizzazione del lavoro

## 5.2 Repository

Il repository è stato uno strumento cruciale per quanto riguarda la sincronizzazione del gruppo e il mantenimento del codice aggiornato e ordinato.

Il lavoro effettuato tramite Git è iniziato il 06/04/2021, con l’iterazione-05, questo perché le prime cinque iterazioni riguardavano la ricerca di informazioni, l’installazione di LL e la familiarizzazione con il software in questione.

Le attività hanno avuto una cadenza settimanale e bisettimanale in funzione del lavoro svolto e delle Issues aperte non ancora svolte.

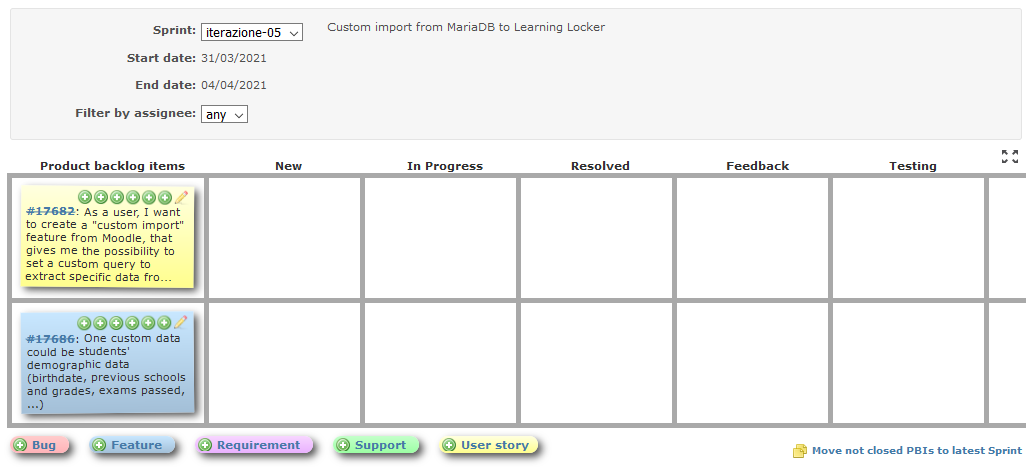


Figura 14: Sprint Board iterazione-05

# 6 Conclusioni

## 6.1 Problemi riscontrati

Purtroppo, durante lo sviluppo del progetto sono stati innumerevoli i problemi che sono stati riscontrati, tutti princiaplente sull’LRS:

* **Poco supporto** da parte dell’azienda che ha prodotto il software, nonostante la documentazione fosse buona, sono diverse le issues aperte e le domande che molti utenti si pongono.
* **I video dimostrativi sono ormai datati** e le tecnologie sono in continuo miglioramento, sarebbe opportuno aggiornare le informazioni e dare maggiore supporto agli sviluppatori.
* **Errori nell’installazione**, infatti durante la fase di installazione, anche se l’esecuzione dello script è stata corretta, il software contiene degli errori in fase di installazione che ha dovuto correggere il Docente.
* **API praticamente inesistenti**, infatti accedendo al sito di LearningPool le API esistono ma non possono essere utilizzate in alcun modo, oltre ai contatti per problemi possiamo cliccare su un bottone con scritto “Request a demonstration” che però non fa nulla.

## 6.2 Implementazioni future

Il progetto è destinato ad essere sviluppato, sarà parte infatti di un progetto di diploma.

Il tool attualmente è in grado di raccogliere i dati di log da iCorsi, inserire delle query personalizzate da iCorsi, e integrare i dati da Teams.

Sarà opportuno integrare dati da social network, come ad esempio Twitter, per ottenere e incrociare altri dati.

Si potrà implementare la possibilità di definire delle nuove metriche, partendo dai dati esistenti, per l’analisi dei dati.

Si potrebbero implementare altre API per estrarre i dati tramite tool di analisi visuale, come Tableau.

Sono innumerevoli inoltre le API che, in funzione delle esigenze, potrebbero essere implementate.

Oltre all’estensione come progetto di diploma è possibile estenderlo a chiunque ne abbia il bisogno, si potrebbe personalizzare per un determinato istituto o per un singolo docente che ha l’esigenza di processare dati di apprendimento.

# 7 Fonti

**LRS:**

<http://www.imsglobal.org/initial-xapicaliper-comparison>

<http://www.imsglobal.org/caliper-11-metric-profiles>

https://www.imsglobal.org/development-resources

<https://xapi.com/statements-101/>

<https://xapi.com/learning-record-store/>

<https://xapi.com/get-lrs/>

<https://adlnet.gov/news/2016/07/25/adl-experience-api-and-ims-caliper-discovery-review/>

<https://docs.learninglocker.net/guides-custom-installation/>

<https://github.com/LearningLocker/deploy/>

<https://docs.learninglocker.net/http-rest/>

<https://registry.tincanapi.com/#home/activityTypes>

<https://registry.tincanapi.com/#uri/>

**API:**

<https://docs.microsoft.com/it-it/microsoftteams/export-teams-content>

<https://docs.microsoft.com/en-us/graph/api/resources/teams-api-overview>

<https://github.com/xAPI-vle/moodle-logstore_xapi>

<https://pypi.org/project/moodlepy/>

**Python:**

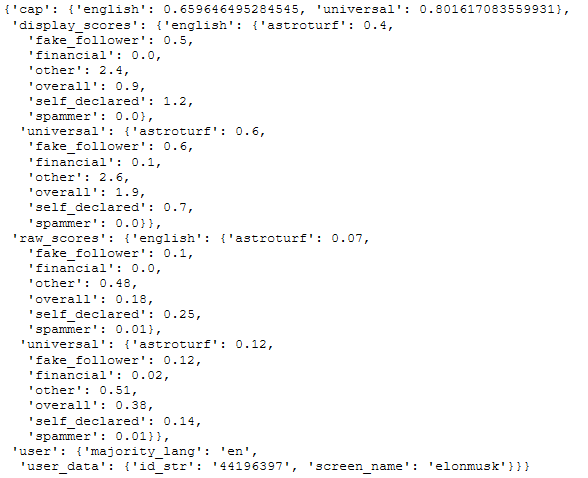
<https://botometer.osome.iu.edu/api>

<https://pypi.org/project/botometer/#description>

<https://cnets.indiana.edu/blog/2020/09/01/botometer-v4/>

<https://blog.quantinsti.com/detecting-bots-twitter-botometer/>

https://rapidapi.com/developer/dashboard



Robert F. Kennedy Jr.

Dr Jane M. Orient

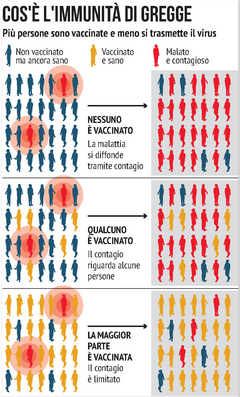
Dottor Massimo Citro

Riccardo Szumski

Paolo Becchi 🡪 libero

* Codice di norimberga (Il Codice traccia una linea di divisione tra sperimentazione lecita e [tortura](https://it.wikipedia.org/wiki/Tortura), e su sperimentazioni non regolate, prive di fondamenti etici)
* Idrossiclorochina
* continuano a dire che grazie ai "vaccini" calano contagi e morti , nel 2020 erano più bassi, capite la propaganda ? 🡪 lockdown
* Vogliamo una terapia domiciliare tempestiva per tutti i cittadini malati di covid19 🡪 si vabbeh con quali risorse

come è possibile che se mi vaccino posso infettarmi?



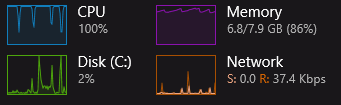
Problemi:

Memoria (parallelizzazione)

bottleneck (lettura in chunk)

Tipi di dati (parsing in date)

ParserError: Error tokenizing data. C error: out of memory



Confronto dei tempi

Da 1000+ secondi a 390 + secondi (users)

