

TEAM 15 REPORT

Corso: Ingegneria del Software A. A. 2020-2021

Partecipanti:

Andrea Mancini, 0000831131

Han Chu, 0000882776

Marco Conti, 0000841421

Pierpaolo Faustini, 0000832011

Shanshan Feng, n° matricola

Indice

1 - Introduzione	3
1.1 - Diagramma dei Casi d'Uso	4
1.2 - Diagramma delle Classi	5
2 - Descrizione degli Sprint	6
2.1 - Sprint #1	6
2.1.1 - Sprint Planning	6
2.1.2 - Sprint Review	6
2.1.3 - Sprint Retrospective	7
2.2 - Sprint #2	7
2.2.1 - Sprint Planning	7
2.2.2 - Sprint Review	7
2.2.3 - Sprint Retrospective	8
2.3 - Sprint #3	8
2.3.1 - Sprint Planning	8
2.3.2 - Sprint Review	9
2.3.3 - Sprint Retrospective	9
2.4 - Sprint #4	10
2.4.1 - Sprint Planning	10
2.4.2 - Sprint Review	10
2.4.3 - Sprint Retrospective	10
3 - Descrizione del Processo Seguito	12
3.1 - Burndown	12
3.2 - Logger	13
3.2.1 - Sprint #1	14
3.2.2 - Sprint #2	15
3.2.3 - Sprint #3	15
3.2.4 - Sprint #4	15
3.2.5 - Andrea Mancini	15
3.2.6 - Han Chu	16
3.2.7 - Marco Conti	17
3.2.8 - Pierpaolo Faustini	18
3.3 - Retrospective finale	19
4 - Struttura e strumenti del progetto	21
4.1 - Gerarchia del progetto	21
4.2 - Strumenti CAS utilizzati	22

1 - Introduzione

Il progetto che andremo a presentare è uno strumento di raccolta e di analisi di informazioni distribuite sul territorio e geolocalizzabili, grazie all'utilizzo di appositi tools (nel nostro caso specifico, i tweets di Twitter).

L'assistere ad una emergenza o un evento non comune, spinge le persone del XXI secolo a condividere tale avvenimento su una piattaforma social (Twitter, Facebook, Instagram,...) in tempo reale. Sfruttando l'immediatezza del gesto e la sua condivisione in tempo reale è possibile utilizzare tale mole di dati per far sì che generi una serie di informazioni utili: se ad esempio siamo in ambito di protezione civile ed avviene un terremoto, la condivisione di stati sui socials è la prima fonte, immediata ed economica, di informazioni dal luogo dell'accaduto da poter sfruttare per ricostruire una situazione operativa. Continuando con altre possibili applicazioni, è possibile immaginare di voler raccogliere informazioni in tempo reale anche per quanto concerne eventi sportivi o musicali, con possibili foto annesse.

Lo **scope** del progetto è quello di creare un'applicazione in grado di sfruttare i dati che le vengono forniti, per rilevare eventi macroscopici rintracciabili grazie a parole chiavi (hashtag dei tweets, o parole all'interno dei tweets stessi) in tempo reale o all'interno di uno storico (repository). Il software dovrà essere in grado di:

- raccogliere i tweets e visualizzarli a schermo;
- filtrare i tweets in base alla posizione, al campo d'interesse e alla data di pubblicazione;
- aggregare i tweets e interagirvi attraverso una dashboard: posizioni sulla mappa, una word cloud con i termini più frequenti, ed alcuni diagrammi che riportano il numero di tweets nell'unità di tempo e nel tipo di dispositivo utilizzato.

Il **product backlog** è composto dalle seguenti *User Stories* (US): [da ordinare]

- Come utente mi interessa raccogliere i tweets, così da poterli analizzare.
- Come utente mi interessa avere uno spazio per interagire con le funzionalità dell'applicazione.
- Come utente mi interessa geolocalizzare i tweets su una mappa.
- Come utente mi interessa visualizzare le parole più utilizzate a schermo.
- Come utente mi interessa geolocalizzare i tweets di una persona specifica, così da poterne seguire gli spostamenti.
- Come utente vorrei sapere l'ora di pubblicazione dei tweets, così da poterli classificare.
- Come utente voglio avere la possibilità di salvare le mie ricerche (cioè i tweets ottenuti) in modo da poterle visualizzare successivamente.
- Come utente mi interessa seguire una certa parola chiave, utente o luogo.
- Come utente mi interessa condividere sui socials i risultati delle mie ricerche.

Nei paragrafi successivi andremo a mostrare il diagramma dei casi d'uso e delle classi per una migliore comprensione della struttura complessiva del progetto.

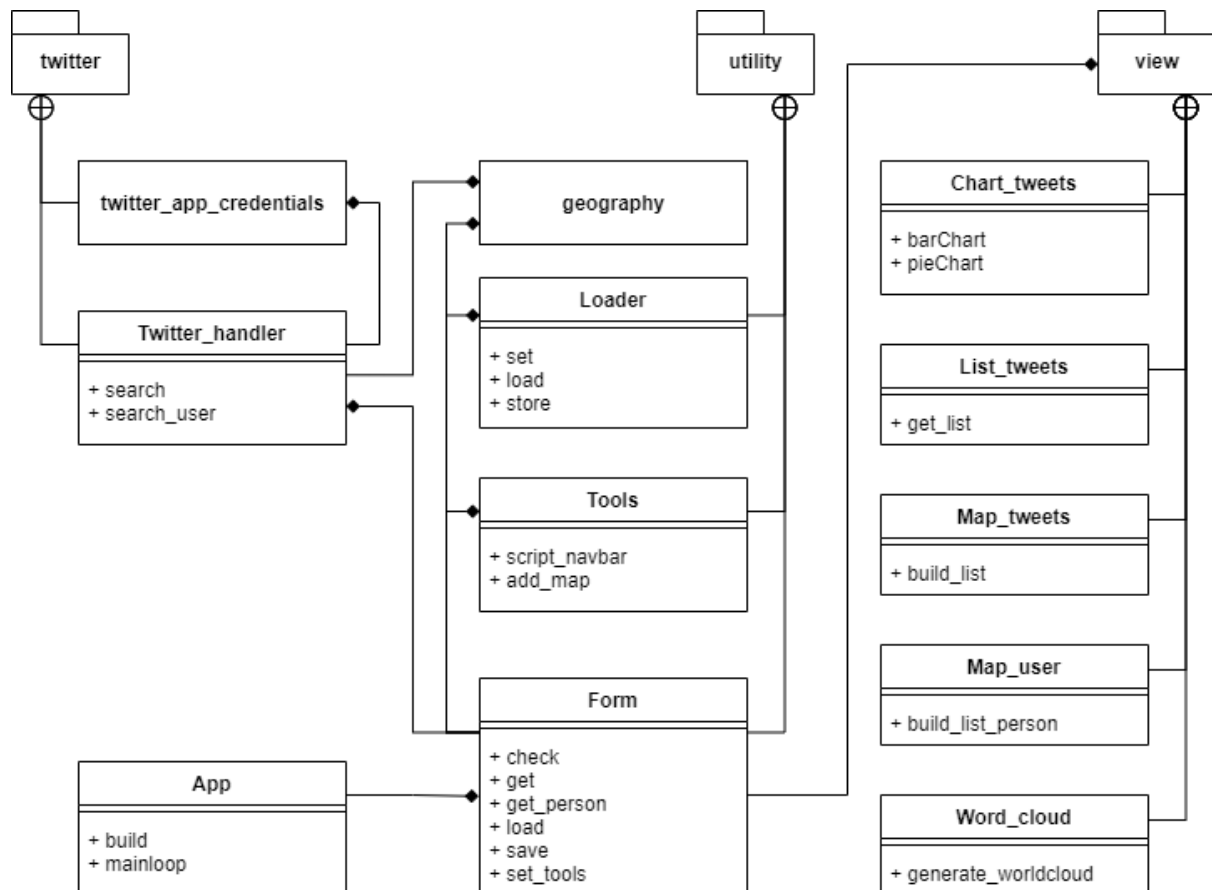
1.1 - Diagramma dei Casi d'Uso

Date le precedenti *User Stories*, il diagramma dei *Casi d'Uso* è:



1.2 - Diagramma delle Classi

Il diagramma delle *Classi*, come viene descritto nel punto [4.1](#), è dato dalla seguente struttura:



Per semplicità abbiamo ommesso i campi e definito solo i metodi principali.

2 - Descrizione degli Sprint

Questa sezione è dedicata alla descrizione dei vari Sprint, con annessi Sprint Planning, Review e Retrospective del progetto Tweeter Tracker. Per comodità queste tre parti resteranno unite e, assieme a tutti i componenti del gruppo abbiamo deciso di non utilizzare le carte Essence per lo svolgimento di queste attività. Data la quantità di lavoro e la facilità con cui abbiamo affrontato gli incontri, non abbiamo ritenuto necessario l'utilizzo di tale strumento all'interno dei nostri meeting. La durata della Sprint Review e Retrospective è stata di circa 3:00, data la lunghezza degli Sprint.

La *definizione di fatto* e le *condizioni di accettazione*, per chiarezza e per evitare elementi ridondanti, abbiamo deciso di inserirle all'interno del product backlog di Taiga, così come il burndown di ogni singolo Sprint.

2.1 - Sprint #1

2.1.1 - Sprint Planning

Questo sprint preparatorio è stato affrontato con l'idea di prendere consapevolezza delle metodologie e degli strumenti agili, messi a disposizione dai Product Owner. Lo **Sprint Goal** si è focalizzato sullo svolgimento di Scrumble, per appunto apprendere le metodologie agili, insieme alla costruzione rudimentale del product backlog.

Non avendo svolto nessuna attività correlata allo sviluppo del codice, lo **sprint backlog** non è stato necessario definirlo, ci siamo concentrati sulla unione del team.

2.1.2 - Sprint Review

La data di inizio dello sprint è stato l'11 ottobre 2020 con termine il 31 ottobre 2020. In questo print iniziale e introduttivo abbiamo dato spazio all'introduzione dei vari componenti del team, così da creare un ambiente rilassato e familiare per tutti.

La prima parte dello sprint ha riguardato, oltre all'introduzione, anche lo scoprire le funzionalità di Scrum e, di conseguenza, Scrumble. Per quanto riguarda la partita di Scrumble abbiamo riscontrato diverse difficoltà nel riuscire a comprendere lo svolgimento del gioco in sé, comportando un allungamento dei tempi di completamento; nonostante questi imprevisti, la partita è stata completata con successo con un livello medio di conoscenza generale di Scrum, permettendoci così di sfruttarlo all'interno della nostra metodologia di lavoro. Successivamente, ci siamo concentrati sulle US, andando ad analizzare le richieste stakeholders e cercando dei modi plausibili per poter descrivere al meglio tali richieste; anche in questo caso vi sono state alcune difficoltà date dall'inesperienza generale dei componenti su tale argomento, che però sono state superate e portate a compimento.

Data l'inesperienza iniziale del team, vi sono stati degli giustificati ritardi durante questa prima fase di sviluppo del progetto, che ci ha portato a non avere la

possibilità di implementare del codice utile da presentare alla fine di questo primo sprint. Dopo un paio di riunioni, siamo giunti alla conclusione di utilizzare il linguaggio Python, guardandola come una sfida da portare a termine, data la flessibilità di questo progetto.

2.1.3 - Sprint Retrospective

Come prima esperienza di certe dinamiche lavorative è stato necessario un certo lavoro di assestamento tra i membri del team, compito non facile lasciato quasi totalmente allo Scrum Master, nel mio caso. Per quanto riguarda gli elementi del team, posso affermare che le personalità sono collimate facilmente, con la totale mancanza di elementi di disturbo o non collaborativi.

In accordo col team, data la scarsa mole di lavoro che siamo riusciti a processare durante questa fase, ci aspettiamo un incremento del lavoro, così come una migliore comunicazione all'interno del team; a tal proposito abbiamo deciso, assieme ai PO, di sfruttare alcuni strumenti open source come Mattermost per la messaggistica mentre Taiga e GitLab per quanto concerne l'organizzazione e lo sviluppo di codice.

2.2 - Sprint #2

2.2.1 - Sprint Planning

Lo **sprint goal** di questo sprint è stato quello di raccogliere i tweets e di definire un'interfaccia grafica basilare per l'applicazione.

Lo **sprint backlog** si è concentrato nel creare una base sulla quale poi poter lavorare separatamente, così da non dover dipendere gli uni dagli altri per le implementazioni successive. Il team si è focalizzato sulla costruzione di un'interfaccia grafica minimale e dell'elemento fondante del progetto, ossia la raccolta dei tweets: la nostra idea comune è stata quella di costruire qualcosa di visivo, così che l'utente potesse avere qualcosa di interagibile, sviluppando al contempo anche la parte fondante, ossia appunto la raccolta dei tweets.

2.2.2 - Sprint Review

Lo sprint è iniziato in data 01 novembre 2020 ed è terminato in data 22 novembre 2020 come da programmato. Le US che ci eravamo prefissati di portare a termine sono state concluse con successo, senza troppi imprevisti nel percorso.

Nonostante il lavoro si stato svolto con successo, vi sono stati dei piccoli problemi durante alcuni momenti dello sviluppo: per lo studio di Essence è stato necessario più tempo del previsto, in quanto il sottoscritto (Scrum Master) ho avuto alcune difficoltà nel capire come funzionavano alcuni "giochi" di Essence; dopo aver appreso le informazioni necessarie alla comprensione di Essence, ho ritenuto opportuno il non utilizzo di quest'ultimo strumento per lo svolgimento delle Sprint Review e Retrospective). Passando alla parte di progetto, vi è stata una difficoltà nella costruzione della GUI: riuscire a riprodurre in python ciò che avevamo

immaginato potesse essere l'interfaccia dell'applicazione, è stata portata a termine grazie ad alcuni video informativi sull'utilizzo di alcune librerie specifiche; è rimasto in sospeso il miglioramento dell'interfaccia grafica per una migliore esperienza utente.

Nella parte del codice vi sono state, invece, molteplici difficoltà: inizialmente un inaspettato difetto nell'IDE, supponiamo, non ha permesso di far funzionare lo stesso codice su due calcolatori differenti, ciò ha portato anche a dover ricreare da zero un'intera porzione, che dopo questa operazione, è tornata a funzionare, comportando però una grossa perdita di tempo. Sempre nella parte codice, la complessità delle API per la raccolta dei tweets ha portato un'ulteriore perdita di tempo. Infine, è stato impiegato del tempo per l'installazione di SonarQube su alcuni dispositivi, ma persistono dei problemi da risolvere.

Infine si è presentato un problema per quanto riguarda la compatibilità di una libreria utilizzata, con il sistema operativo Windows: tale incompatibilità, essendo causata da un errore interno del sistema operativo, non è stata risolta, ma abbiamo comunque deciso di sfruttare tale libreria dato che la quasi totalità dei membri è provvista di un sistema operativo UNIX.

2.2.3 - Sprint Retrospective

Come secondo sprint, ma prima volta dove è stato prodotto del codice, è possibile notare che i membri del team non sono ancora ben coesi; evento prevedibile data la mancanza di precedenti collaborazioni. Ciò ha comportato l'assenza di una buona comunicazione che di conseguenza ha portato sì al compimento del lavoro, ma non a pieno potenziale. Data tale mancanza, prevediamo di incrementare gli incontri tra i membri così da migliorare la coordinazione e la comunicazione, con l'obiettivo di incrementare anche la quantità di lavoro; puntiamo molto sulla comunicazione verbale, nonostante i problemi linguistici interni causati dalle origini di alcuni membri del team, non affidandoci troppo ai sistemi di messaggistica suggeriti dagli stessi PO, in particolare ci avvaliamo della chat vocale del programma Microsoft Teams.

2.3 - Sprint #3

2.3.1 - Sprint Planning

In questo sprint abbiamo perseguito, secondo lo **sprint goal**, ad implementare le US: mostrare i tweets sulla mappa, creare la term cloud, e il tracciamento dei tweets di una persona. L'obiettivo in questo caso era quello di sviluppare le singole parti che ritenevamo necessarie per far sì di avere un'applicazione "decente" così da poterle poi unire, dando a Twitter Tracker delle funzionalità di base solide.

Lo **sprint backlog** ha definito l'implementazione separata dei vari goal, così da poter lavorare più efficacemente sulle singole parti senza creare conflitti fra i vari membri e, all'occorrenza, aiutare chi riscontrava difficoltà nello sviluppo.

2.3.2 - Sprint Review

Come da programma, lo sprint è iniziato il 22 novembre 2020 ed è terminato il 13 dicembre 2020, durando le solite tre settimane. Il lavoro è stato portato a termine con successo, le US che avevamo programmato di svolgere sono state portate a compimento. Ciò non toglie che non vi siano stati dei problemi: gli impegni dei singoli componenti hanno portato ad una conseguente poca frequenza degli incontri, portando ad un rallentamento del lavoro complessivo.

Oltre ai problemi di natura organizzativa, vi sono stati anche dei problemi tecnici, anche se tutti sono stati risolti: durante lo sviluppo della Word Cloud abbiamo avuto dei problemi per quanto concerne l'implementazione delle espressioni regolari all'interno del codice, necessarie per eliminare i link ipertestuali dalla Word Cloud stessa (infine risolti). Inoltre vi sono stati anche dei problemi con la mappa: poiché è stata implementata in HTML abbiamo scelto di utilizzare la libreria webview, ciò ha portato all'impossibilità di poterla integrare con la finestra principale di Twitter Tracker, implementata utilizzando tkinter; questo problema è stato ovviato con l'inserimento di una finestra secondaria per la visualizzazione di mappa e wordcloud, scritta interamente in HTML.

Da evidenziare anche la mancanza di coesione tra alcune parti del codice, che prevediamo condurrà ad un decremento dello sviluppo di alcune parti del progetto, favorendo una maggiore cura nella raffinazione del codice.

I rallentamenti dei lavori e gli impegni dei singoli membri hanno portato ad un incremento del debito tecnico complessivo, fattore che non potrà essere ignorato nei successivi sprint e che di conseguenza influirà sul risultato finale del progetto.

2.3.3 - Sprint Retrospective

La volontà del team a lavorare in modo separato sulle varie parti del progetto ha portato a un maggiore rendimento dal punto di vista del codice scritto, ma anche ad una minore organizzazione delle parti; la mancanza maggiore si è avvertita nella poca organizzazione tra membri (dovuta anche a cause esterne al progetto).

In generale il team ha riscontrato una scarsa presenza da parte della figura del PO, la quale non è riuscita a dare la necessaria chiarezza alla varie esigenze dell'app, portando ad un inevitabile ritardo nello sviluppo di alcune parti e allo slittamento di alcune funzionalità che saranno necessariamente rimandate al quarto sprint.

Prevediamo che sarà necessario del tempo per poter garantire un'esperienza utente ottimale e per far coesistere le varie parti messe in gioco. A tale scopo nel prossimo sprint prevediamo di concentrarci esclusivamente sul far integrare le varie parti di codice, così da rendere il codice omogeneo nelle sue varie componenti, mettendo in secondo piano l'implementazione di nuove eventuali features.

2.4 - Sprint #4

2.4.1 - Sprint Planning

Lo **sprint goal** in questa fase finale è stato quello di far coesistere le parti create in precedenza, cosicché unite, potessero funzionare efficacemente assieme, andando così a creare la versione finale di Twitter Tracker;

Lo **sprint backlog** ha avuto come focus quello di riuscire a concludere il progetto in modo definitivo: aggiungendo due nuove funzionalità, quali il salvataggio della ricerca e il grafico per la visualizzazione della frequenza dei tweet; infine abbiamo accorpato le parti non ancora integrate nel codice finale, facendo attenzione che il software superasse le casistiche da noi ipotizzate.

2.4.2 - Sprint Review

L'ultimo sprint del progetto ha avuto inizio il 14 dicembre 2020 ed è terminato il 3 gennaio 2021 come da previsione. Il lavoro di inclusione e affinamento del codice è stato portato a termine con successo, lasciando spazio anche all'introduzione di nuove features, le ultime incluse all'interno dell'applicazione, ciò è dovuto alla mole di impegni dei singoli componenti, che non ha permesso ulteriori aggiunte al software. Nonostante ciò, non tutte le US richieste sono state aggiunte e anche l'integrazione delle varie parti ha subito svariate battute di arresto, per motivi che discuteremo nella parte di Retrospective.

I problemi principali li abbiamo riscontrati con l'integrazione, data la mancanza di DoD puntali fornite dagli tests e/o programmi di analisi della qualità del codice come SonarQube (la motivazione è spiegata nella sezione [3.3](#)), ciò ha portato un progressivo rallentamento dei lavori i quali, dopo il completamento delle ultime US, sono sfociati anche al di fuori del quarto Sprint, per dei piccoli interventi di rifinitura. Altra causa dei rallentamenti sono stati gli impegni dei singoli membri: provenendo da anni accademici differenti, ogni elemento del team aveva ed ha priorità diverse rispetto agli altri, il che ha portato ognuno di noi a doversi riorganizzare per quanto riguarda la scadenza inizialmente prevista.

Infine, lo studente Han Chu, il quale ha provveduto a completare l'integrazione durante il mese di Gennaio, è stato impossibilitato a partecipare allo sviluppo per buona parte del quarto sprint, dato l'imprevisto malfunzionamento del suo computer principale, per tale motivo è riuscito, dopo le riparazioni, a completare l'integrazione soltanto dopo la fine del quarto Sprint.

2.4.3 - Sprint Retrospective

In questa parte conclusa del progetto Twitter Tracker ha avuto diversi aspetti non positivi, per quanto riguarda l'applicazione dei metodi agili: gli incontri sono diminuiti drasticamente per diversi fattori, da non trascurare l'inizio della sessione di esame, portando ad una organizzazione generale ridotta al minimo, che di conseguenza ha fatto sì che i lavori per il completamento del progetto si protrassero più del

necessario; in generale, le necessità dei singoli membri del team hanno preso il sopravvento sulla coesione precedente, portando i ritmi di lavoro al minimo, per potersi concentrare, ognuno, sui propri impegni universitari.

Nonostante gli sforzi, non siamo riusciti ad includere tutte le US previste, dati problemi citati sopra, potendo così consegnare un software sì testato e funzionante, ma che non corrisponde appieno alle richieste. Le US non realizzate sono: *come utente mi interessa seguire una certa parola chiave, utente o luogo, e come utente mi interessa condividere sui socials i risultati delle mie ricerche.*

3 - Descrizione del Processo Seguito

Il progetto Twitter Tracker è stato pensato inizialmente per essere svolto in tre Sprint per un totale di nove settimane, ma abbiamo deciso di allungarlo a quattro sprint con dodici settimane di lavoro e qualche piccola rifinizione al codice dopo che sono stati conclusi gli sprint.

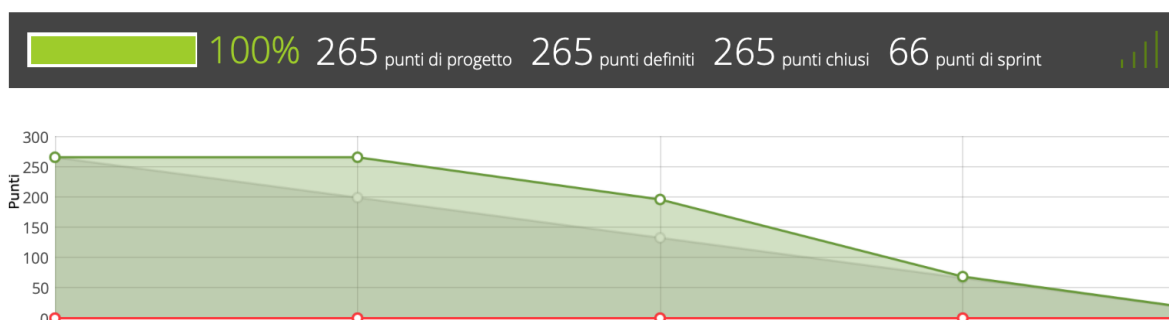
Il primo incontro tra i membri è avvenuto tramite una chiamata svoltasi sullo strumento di condivisione Teams, mentre le prime interazioni tra i membri sono avvenute grazie al gioco Scrumble, utilizzato come mezzo per rompere il ghiaccio e far sì che le personalità presenti cominciassero ad amalgamarsi. Dopo una partita iniziale di prova il team ha svolto una partita completa, riuscendo a concluderla positivamente, ossia terminando il numero di task prefissate ed esauendo il debito tecnico accumulato. Sempre nel primo Sprint, abbiamo deciso di mettere a punto i vari framework a nostra disposizione, sia per capirne le funzionalità, sia per avere una migliore idea di come poter gestire con essi il progetto assegnatoci.

In seguito allo “Sprint introduttivo” il team ha portato avanti il progetto per i tre Sprint successivi, andando a strutturare come segue il processo seguito: prima, lo sviluppo di una interfaccia grafica basilare, assieme alla ricerca dei tweets in base alla parola chiave, in modo tale da avere qualcosa di concreto da consegnare entro il secondo Sprint; successivamente, il team si è concentrato sul sviluppare le varie parti aggiuntive dell'applicazione, concentrandosi sull'aumento di features possibili da implementare; infine nel quarto ed ultimo Sprint il team ha provveduto a far coesistere gli elementi di codice creati singolarmente, così da poter fornire un'applicazione finale stabile, riuscendo anche ad includere altre due funzionalità all'applicazione.

3.1 - Burndown

Di seguito è riportato il grafico Burndown complessivo del progetto per chiarire la distribuzione complessiva del lavoro (per un approfondimento, visitare la repository di Taiga del gruppo 15):

PROGETTO SWE 2020 (TEAM15) BACKLOG



3.2 - Logger

I seguenti screenshots rappresentano i dati del logger dei membri del team:
Logger di Andrea:

My performance

 01/11/2020 - 17/02/2021

Code changes

8206	1513	730	5963
TOTAL CHANGES	LINES ADDED	LINES DELETED	LINES CHANGED

Comments changes


1201	689	512
TOTAL CHANGES	COMMENTS ADDED	COMMENTS DELETED

Tests changes

0	0	0
TOTAL CHANGES	TESTS ADDED	TESTS DELETED

Logger di Han:

My performance

 01/11/2020 - 17/02/2021

Code changes

21838	6322	3801	11715
TOTAL CHANGES	LINES ADDED	LINES DELETED	LINES CHANGED

Comments changes


2273	1110	1163
TOTAL CHANGES	COMMENTS ADDED	COMMENTS DELETED

Tests changes

0	0	0
TOTAL CHANGES	TESTS ADDED	TESTS DELETED

Logger di Pierpaolo:

My performance

 01/11/2020 - 17/02/2021

Code changes

6203	1534	1306	3363
TOTAL CHANGES	LINES ADDED	LINES DELETED	LINES CHANGED

Comments changes

1564	815	749
TOTAL CHANGES	COMMENTS ADDED	COMMENTS DELETED

Tests changes

4	2	2
TOTAL CHANGES	TESTS ADDED	TESTS DELETED

Inoltre, dato che all'inizio non tutti hanno installato il logger proposto dai professori, per tracciare le nostre attività abbiamo tenuto sia un diario di team che un diario individuale dove è descritto la data, la durata e l'argomento delle varie attività.

3.2.1 - Sprint #1

- 08/10/20: (15:00 - 16.15) 1:15
Introduzione: presentazione dei membri
- 11/10/20: (10:00 - 12:00) 2:00
Introduzione: studio di Scrumble
- 12/10/20: (11.30 - 13:45) 2:15
Introduzione: partita introduttiva di Scrumble parte 1
- 13/10/20: (18:15 - 19:15) 1:00
Introduzione: partita introduttiva di Scrumble parte 2
- 18/10/20: (10:30 - 13:15, 14:30 - 15:30) 3:45
Sprint Planning Meeting: scelte delle tecnologie per il progetto (linguaggio, librerie,...) e definizioni delle User Stories
- 22/10/20: (15:30 - 17:30) 2:00
Introduzione: partita completa di Scrumble e valutazione dello Scrumble
- 24/10/20: (10:00 - 13:00) 3:00
Sprint Planning Meeting: accesso (login) dei vari strumenti (Taiga, Mattermost, GitLab,...)
- 26/10/20: (10:00 - 12:00) 2:00
Sprint Planning Meeting: definizioni del backlog di prodotto

- 01/11/20: (15:00 - 17:45) 2:45
Introduzione: Essence burndown e altro

3.2.2 - Sprint #2

- 20/11/20 (15:00 - 17:00) 2:00
Daily Scrum: presentazione al team delle task sviluppate
- 22/11/20 (14:00 - 17:15) 3:15
Sprint Review, Sprint Retrospective e Sprint Planning Meeting #2

3.2.3 - Sprint #3

- 12/12/20 (10:00 - 11:45) 1:45
Daily Scrum
- 14/12/20 (15:00 - 17:00) 2:00
Daily Scrum
- 17/12/20 (15:00 - 17:30) 2:30
Sprint Review e Sprint Retrospective

3.2.4 - Sprint #4

- 24/12/20 (16:00 - 18:00) 2:00
Sprint Planning Meeting #3

3.2.5 - Andrea Mancini

- 11/10/20 (16:00 - 18:00) 2:00
Approfondimento Scrumble
- 17/10/20 (15:00 - 17:00) 2:00
Analisi ambiente CAS
- 19/10/20 (10:00 - 11:00) 1:00
Installazione Eclipse + plugin PyDev
- 20/10/2020 (15:00 - 18:00) 3:00
Lettura e comprensione di Tkinter
- 22/10/20 (16:00 - 18:00) 2:00
Primi sviluppi con Tkinter
- 02/11/20 (15:00 - 16:30) 1:30
Installazione del logger per Eclipse, di sonarqube e setup di Git per Eclipse
- 10/11/20 (15:00 - 18:00) 3:00
Studio di googlemaps
- 14/11/20 (14:00 - 17:00) 3:00
Sviluppo del form di ricerca
- 18/11/20 (18:00 - 20:00) 2:00
Migliorie al form

- 19/11/20 (16:00 - 17:00) 1:30
Analisi con Han del lavoro svolto e refactoring
- 20/11/20 (10:00 - 11:30) 1:30
Refactoring
- 09/12/20 - 15/12/20
Discussione e sviluppo con shanshan della mappa e sulle relative librerie
- 07/12/20 (14:00 - 17:00) 3:00
Integrazione parte grafica con Han
- 12/12/20 (14:00 - 17:00) 3:00
Sviluppo della lista dei Tweet via HTML
- 17/12/20 (9:00 - 11:30) 2:30
Discussione salvataggio dei tweets con Han e Pair Programming con Han
- 18/12/20 (11:00 - 12:00) 1:00
Integrazione della ricerca e parte grafica con Han
- 19/12/20 (11:00 - 12:00) 1:00
Integrazione della ricerca e parte grafica con Han
- 21/12/20 (15:30-18:00)] 2:30
Integrazione con Pierpaolo (WordCloud)
- 23/12/20 (10:00 - 12:30) 2:30
Organizzazione complessiva dei documenti del progetto
- 24/12/20 (16:00-19:00) 3:00
Riunione scelta integrazione multi-thread
- 28/12/20 - 3/01/21
Completamento integrazione con tutte le parti
- 02/01/21 - 03/01/21
Riunioni con Han e Pierpaolo per integrazione

3.2.6 - Han Chu

- 11/10/20 (16:00 - 18:00) 2:00
Approfondimento Scrumble
- 17/10/20 (09:00 - 11:00) 2:00
Analisi ambiente CAS
- 20/10/20 (09:00 - 13:00) 4:00
Installazione Eclipse + plugin PyDev
- 22/10/20 (15:00 - 17:30) 3:30
Lettura e comprensione di TweetHandler
- 23/10/20 (16:00 - 18:00) 2:00
Analisi di alcune API per Twitter
- 05/11/20 (12:00 - 14:30) 2:30
Installazione del logger per Eclipse (fail), di Sonarqube (in progress) e setup di Git per Eclipse (done)
- 08/11/20 (16:00 - 18:00) 2:00
Analisi e test della libreria python-twitter

- 09/11/20 (16:00 - 18:00) 2:00
Analisi della libreria tweepy, in particolare la funzione search, e dell'oggetto tweepy.model.status
- 14/11/20 (14:00 - 17:00) 3:00
Implementazione delle funzioni di ricerca
- 18/11/20 (18:00 - 20:00) 2:00
Realizzazione delle funzioni di importazione ed esportazione dei dati
- 19/11/20 (16:00 - 17:00) 1:30
Analisi con Andrea del lavoro svolto e refactoring
- 20/11/20 (10:00 - 11:30) 1:30
Refactoring
- 22/11/20 (18:00 - 18:30) 0:30
Riunione backend con Pierpaolo
- 29/11/20 (16:00 - 18:00) 2:00
Refactoring
- 02/12/20 (9:00 - 12:00) 3:00
Implementato la ricerca geolocalizzata
- 05/12/20 (15:00 - 17:00) 2:00
Implementato la ricerca utente
- 07/12/20 (14:00 - 17:00) 3:00
Integrazione parte grafica
- 12/12/20 (14:00 - 17:00) 3:00
Refactoring
- 17/12/20 (9:00 - 11:30) 2:30
Discussione salvataggio dei tweets con Andrea e Pair Programming con Andrea
- 18/12/20 (11:00 - 12:00) 1:00
Integrazione della ricerca e parte grafica con Andrea
- 19/12/20 (11:00 - 12:00) 1:00
Integrazione della ricerca e parte grafica con Andrea
- 23/12/20 (10:00 - 12:30) 2:30
Organizzazione complessiva dei documenti del progetto
- 26/12/20 (16:00 - 18:00) 2:00
Implementazione del search bar
- 28/12/20 (15:30 - 18:30) 3:00
Wordcloud con Marco e Pierpaolo

3.2.7 - Marco Conti

- 9/10/20 (16:00 - 18:00) 2:00
Studio elementi Agili
- 10/10/20 (10:00 - 12:00) 2:00
Studio Gioco Scrumble
- 16/10/20 (15:00 - 18:00) 3:00

- Studio elementi di progetto
- 17/10/20 (10:00 - 12:00) 2:00
Installazione Eclipse + PyDev e tentativo risoluzione problemi
- 23/10/20 (16:00 - 18:00) 2:00
Studio di Essence
- 27/10/20 (10:00 - 11:00) 1:00
Controllo e organizzazione di Taiga
- 05/11/20 (17:00 - 18:00) 1:00
Stesura report primo Sprint
- 06/11/20 (15:00 - 16:00) 1:00
Stesura report primo Sprint
- 09/11/20 (9:00 - 10:00) 1:00
Studio e analisi delle librerie Tweepy
- 10/11/20 (9:00 - 10:00) 2:00
Studio e analisi delle librerie Tweepy
- 21/11/20 (10:00 - 12:00) 2:00
Controllo e organizzazione di Taiga
- 25/11/20 (15:00 - 18:00) 3:00
Stesura report secondo sprint
- 27/11/20 (17:00 - 18:00) 1:00
Controllo e organizzazione di Taiga
- 05/12/20 (15:00 - 18:00) 3:00
Studio libreria WordCloud
- 06/12/20 (10:30 - 13:30) 3:00
Implementazione wordcloud con Pierpaolo
- 08/12/20 (10:30 - 12:30) 2:00
Implementazione wordcloud con Pierpaolo
- 17/12/20 (10:00 - 12:00) 2:00
Stesura report terzo sprint
- 21/11/20 (10:00 - 11:00) 1:00
Controllo e organizzazione di Taiga
- 28/12/20 (15:30 - 18:30) 3:00
Wordcloud con Han e Pierpaolo

3.2.8 - Pierpaolo Faustini

- 12/10/20 (09:30 - 12:30) 3:00
Studio di alcuni componenti CAS prima della release dipartimentale del professore (Taiga, Gitlab, Sonarqube)
- 18/10/20 (15:00 - 17:00) 2:00
Installazione plugin PyDev su Eclipse
- 20/10/20 (15:00 - 17:00) 2:00
Lettura e comprensione di 'TweetHandler' del tesista

- 27/10/20 (15:00 - 18:00) 3:00
Studio e setup di git
- 29/10/20 (9:00 - 13:00) 4:00
Studio generale delle API di Twitter
- 30/11/20 (17:00 - 20:00) 3:00
Studio generale della libreria tweepy per Twitter
- 03/11/20 (17:00 - 20:00) 3:00
Studio e setup di Sonarqube ed il Logger con Andrea
- 04/11/20 (15:00 - 17:00) 2:00
Implementazione ricerca dei tweets in base a parola chiave
- 06/11/20 (16:00 - 20:00) 4:00
Implementazione ricerca dei tweets in base a parola chiave
- 12/11/20 (10:00 - 13:00) 3:00
Implementazione ricerca dei tweets in base a parola chiave - Terminata
- 08/11/20 (14:30 - 18:30) 4:00
Implementazione ricerca dei tweets in base a data/ora
- 15/11/20 (16:30 - 20:00) 3:30
Refactoring e consultazione con Han per ricerca tweets in base a data/ora (done)
- 21/11/20 (11:30 - 13:00) 1:30
Refactoring
- 22/11/20 (21:00 - 00:00) 3:00
Studio libreria termcloud
- 25/11/20 (22:00 - 23:30) 1:30
Ultimo refactoring e messo il codice su Gitlab per secondo sprint
- 30/11/20 (21:00 - 23:30) 2:30
Implementazione wordcloud
- 06/12/20 (10:30 - 13:30) 3:00
Implementazione wordcloud con Marco
- 08/12/20 (10:30 - 12:30) 2:00
Implementazione wordcloud con Marco
- 10/12/20 (10:00 - 13:30) 3:30
Implementazione wordcloud
- 11/12/20 (18:30 - 19) 00:30
Messo codice su Gitlab per terzo sprint
- 15/12/20 (21:30 - 23:30) 2:30
Studio tecniche di implementazione di wordcloud interattiva
- 19/12/20 (17:30 - 20:00) 2:30
Implementazione wordcloud interattiva
- 23/12/20 (10:30 - 13:30) 3:00
Implementazione wordcloud interattiva

3.3 - Retrospective finale

Per quanto concerne la retrospettiva finale posso affermare che, in quanto Scrum Master, il team è stato coeso dall'inizio alla fine del processo produttivo, con l'assenza di elementi di disturbo o non collaborativi, facendo sì che si venisse a creare un ambiente confortevole e di collaborazione, senza che nessuno si sentisse escluso durante il processo di sviluppo del codice.

Durante gli sprint, sebbene abbiamo creato un'applicazione funzionante seguendo un procedimento agile, non siamo riusciti ad implementare tutte le US e non abbiamo utilizzato strumenti per la misurazione della qualità del codice come SonarQube e/o test, dovuto all'inesperienza di tutti i membri del team in tale ambito. Tutto ciò ha portato a una definizione di "Fatto" o *Definition of Done* molto astratta e poco chiara, basata molto sulle possibili casistiche, andando a testare manualmente; ciò ha portato una ovvia lentezza nel riuscire a testare adeguatamente il codice.

Inoltre abbiamo avuto problemi a relazionarsi agevolmente con la studentessa Shanshan Feng data la sua scarsa conoscenza della lingua italiana, essendo cinese; fortunatamente anche lo studente Han Chu ha origini cinesi e ha provato (con scarsi risultati) a comunicare con Shanshan Feng.

In merito al processo adottato durante tutto lo sviluppo lo si può considerare Agile nelle sue generalità: vi sono stati gli incontri per la discussione dei vari Sprint, è stato costruito sia un Backlog di Processo che di Sprint e vi è stata la collaborazione delle varie parti, ma forse quello che è più mancato è l'esperienza, per ovvie ragioni, nel riuscire a gestire questa tipologia di processo produttivo. Più volte ci siamo ritrovati a non saper come gestire determinate situazioni: come strutturare adeguatamente il workflow, capire se la distribuzione dei compiti e delle task assegnate fosse giusto o come evidenziare i problemi dei singoli membri così che potessero essere risolti in modo puntuale ed efficace.

Le migliorie possibili da adottare più immediate sono l'approfondimento del metodo Agile e del framework di Scrum, così da poter migliorare l'organizzazione tra i singoli membri, portando ad un inevitabile incremento della produttività e dell'efficienza del team nel risolvere i possibili imprevisti; una conoscenza più capillare del metodo Agile permettere sicuramente ad un team una migliore coesione ed efficienza, assieme anche una maggiore consapevolezza delle proprie capacità, così che la quantità di lavoro necessaria al completamento del progetto, sia ben distribuita durante tutti gli sprints.

In definitiva gli elementi che ci hanno portato a non ritenerci soddisfatti del lavoro intrapreso sono la mancanza di esperienza nell'utilizzo delle metodologie Agili, l'erronea scelta del linguaggio che ci ha portato ad avere diversi problemi e ritardi all'interno dei vari Sprints ed infine la mancanza di una figura di riferimento, che il sottoscritto Scrum Master, non è stato capace di rimpiazzare adeguatamente.

4 - Struttura e strumenti del progetto

Il programma è stato realizzato in Python versione 3.9.0 utilizzando vari package installabili tramite *pip*. I package in questione sono:

- bs4
- datetime
- folium
- html5lib
- json
- math
- matplotlib
- nltk
- numpy
- os
- re
- requests
- sys
- tkcalendar
- tkinter
- tweepy
- urllib
- webview
- wordcloud

Inoltre bisogna scaricare la seguente componente:

```
python -m nltk.downloader stopwords
```

Per eseguire il programma bisogna scaricare il programma da [GitLab](#), aprire il terminale ed andare nella cartella `src` del progetto scaricato e digitare:

```
python3.9 App.py
```

4.1 - Gerarchia del progetto

La repository del programma è strutturato nel seguente elenco:

- **App**, il programma eseguibile.
- **Tweets**, il file JSON che viene automaticamente creato all'avvio del programma per il salvataggio dei tweets analizzati; un file JSON per poter fungere da archivio deve avere la struttura:

```
{  
    "Tweets": []  
}
```

- **twitter**, il modulo per l'interagire con Twitter e contiene:
 - **twitter_app_credentials**, contenente le chiavi d'accesso alle API di Twitter. Il programma non utilizza direttamente le API di Twitter ma utilizza il package *tweepy*.
 - **twitter_handler**, la classe che gestisce l'interazione con Twitter.
- **utility**, modulo contenente le classi e funzioni di supporto al programma, ed è composto dai seguenti file:
 - **form**, classe responsabile della finestra per la ricerca o caricamento dei tweets.
 - **geography**, modulo contenente le funzioni per la gestione dei tweets geolocalizzati.
 - **loader**, classe che gestisce il caricamento e salvataggio dei tweet da analizzare.
 - **tools**, classe responsabile della generazione dell'HTML per l'analisi dei tweets
- **view**, modulo che genera i diversi metodi per l'analisi dei tweets; tali metodi sono rappresentati dai file:
 - **chart_tweets**, classe che si occupa della creazione dei grafici dei tweets.
 - **list_tweets**, classe che si occupa della creazione del HTML per la lista dei tweets.
 - **map_tweets**, classe che si occupa della creazione della mappa dei tweets nella ricerca per parola chiave.
 - **map_user**, classe che si occupa della creazione della mappa dei tweets nella ricerca per utente.
 - **word_cloud**, classe che si occupa della creazione della wordcloud dei tweets.

4.2 - Strumenti CAS utilizzati

Il software per la gestione di repository **GitLab**, è stato utilizzato da tutto il team durante lo svolgimento del progetto. Ogni componente del team ha lavorato nel proprio *branch* per poi eseguire il *merge* nel *branch master*. Nonostante la scarsa conoscenza dello strumento di versionamento Git da parte di alcuni membri del team, terminato il progetto tutti hanno acquisito i principali comandi di Git.

Lo strumento per la comunicazione **Mattermost** è stato utilizzato durante la fase iniziale, dove tutti i componenti del team hanno scaricato l'omonima applicazione mobile e comunicato utilizzando i thread appositi, ma dopo alcuni problemi tecnici "richiede spesso il login" e "non arrivano le notifiche" e dato il frequente uso di Telegram di tutti, abbiamo deciso di utilizzare quest'ultimo servizio di messaggistica.

Il software di gestione controllo per i team agili, **Taiga**, è stato scelto e utilizzato dal team durante tutto lo svolgimento del progetto, utilizzandone le sue caratteristiche chiave, quali la rappresentazione grafica del Backlog di prodotto e di Sprint, la visualizzazione degli Epici e del Kanban e in più la wiki per possibili documenti utili

e/o info per il progetto. L'unica features non utilizzata è stata la parte dei Problemi i quali, non essendo un numero gestibile ogni volta, sono stati risolti senza l'utilizzo di tale funzionalità.

Come detto precedentemente, abbiamo provato ad usare **Sonarqube** all'inizio ma dato l'inesperienza di tutti i membri non siamo riusciti ad utilizzarlo correttamente.

Riguardo alla tracciabilità delle attività del team, abbiamo utilizzato il logger proposto dal docente **InnoMetrics** per Eclipse, sebbene alcuni dei nostri colleghi hanno segnalato la problematica sulla privacy di utilizzare un logger nei propri calcolatori. Alcuni hanno avuto problemi nell'installazione e quindi solo in seguito il terzo sprint è stato utilizzato da tutto il team. I membri che non hanno utilizzato come IDE Eclipse hanno utilizzato **WakaTime**. Durante lo svolgimento del progetto lo studente Han Chu ha avuto dei problemi col proprio computer, e quindi ha continuato a programmare con meno regolarità senza utilizzare nessun logger.

Prima dell'utilizzo del logger i membri hanno tenuto un proprio diario che è stato utilizzato per tutto il progetto, ed è descritto nella sezione [3.2](#).

Link strumenti CAS utilizzati:

- GitLab: <http://aminsep.disi.unibo.it/gitlab/team-15/ignsw2020/>
- Logger: <http://aminsep.disi.unibo.it/innometrics/>
- Mattermost: <http://aminsep.disi.unibo.it:8065/gruppo15/channels/town-square/>
- SonarQube: http://aminsep.disi.unibo.it:9000/dashboard?id=SWE_Team15/
- Taiga: http://aminsep.disi.unibo.it/project/pierre_-progetto-ignsw-2020/