Metriche Twitter

1. Introduzione

Si vuole sviluppare un app che permette il calcolo di alcune metriche su uno o più account Twitter.

Le metriche dovranno identificare sia le proprietà del post, classificandolo sulla base del tipo di contenuto (testo, link, immagine, video), sia il riscontro ottenuto dal post (retweet, like, reply).

Metriche da sviluppare:

- Media degli intervalli temporali tra i tweet postati.
- Media dei retweet tra i tweet postati.
- Media delle mention tra i tweet postati.
- Data una lista di utenti, individuare i tweet nei quali uno dei membri della lista menziona un'altro membro.

Le metriche dovranno essere visualizzate attraverso diagrammi o infografiche.

Il progetto deve presentare le seguenti caratteristiche:

HTML5/CSS

- Le pagine devono essere sviluppate in formato HTML5.
- Tutte le pagine devono essere validate.
- Il layout delle pagine deve essere sviluppato con CSS.
- L'applicazione dovrà servirsi di almeno una API HTML5.

AJAX

- Il progetto deve implementare una o più chiamate XMLHttpRequest.
- Le chiamate possono interrogare dati in JSON, XML, XHTML, TXT.

NodeJS

- Il progetto deve implementare una o più chiamate a un servizio NodeJS.
- Le chiamate devono interrogare o caricare dati in JSON o XML.

1.1. Breve analisi dei requisiti

1.1.1. Destinatari

Capacità e possibilità tecniche.

La web-app richiede un basso livello di esperienza per essere utilizzata, gli utenti non hanno bisogno di essere guidati durante l'utilizzo.

Non è richiesta un ampia disponibilità di banda, infatti le risorse scambiate si limitano a brevi informazioni testuali (in formato JSON), l'immagine del profilo degli utenti (in jpg a bassa risoluzione) e grafici delle metriche (in SVG).

La web-app è stata progetta per essere utilizzata su qualunque device, tramite un web browser. Infatti sfruttando il Bootstrap di Twitter, sono state inserite delle regole per gestire display di diverse dimensioni.

Linguaggio.

Non è richiesta la conoscenza di alcun linguaggio specifico per l'utente finale, le uniche conoscenze necessarie sono le terminologie utilizzate da Twitter.

Motivazione.

L'utente andrà ad utilizzare la web-app per scopi di ricerca/statistica.

Il livello di consapevolezza dell'utente sarà di predisposizione alla ricerca diretta.

Il livello di motivazione dell'utente sarà di ricerca attiva.

	Active	Passive
Directed	**Searching**	**Monitoring**
Undirected	Browsing	Being Aware

Osservando il modello di Bates l'utente si posizione nel quadrante in alto a destra.
Una strategia utilizzata per fornire una buona esperienza di navigazione all'utente è stata quella di predisposta una barre di navigazione (presente in tutte le pagine).

1.1.2. Modello di valore

La semplicità nell'ottenere informazioni riguardo uno o più account di Twitter fornisce valore alla web-app.

Non è possibile identificare le transazioni che generano direttamente un valore in quanto la web-app è pensata per essere utilizzata gratuitamente. Una stima sarebbe fattibile solo con l'introduzione della pubblicità nelle varie pagine.

1.1.3. Flusso dei dati

I contenuti vengono reperiti dall'API di Twitter in forma gratuita (rispettando però dei limiti nel numero di richieste).

Utilizzando l'API gratuita i costi si limitano solamente a quelli di gestione del server. (Twitter mette anche a disposizione un API commerciale a pagamento)

Le trasformazioni applicate si limitano al calcolo di alcune metriche sui contenuti stessi, non sono stati previsti dei metodi di archiviazione in quanto si stratta di dati già presenti nella rete.

Inoltre non c'è una frequenza specifica di aggiornamento dei contenuti, in quanto l'aggiornamento dipende direttamente da come gli utenti utilizzano il social network.

1.1.4. Aspetti tecnologici

Lo standard utilizzato per la gestione e condivisione dei contenuti è il JSON. E' stato preferito rispetto ad altre tecnologie per la sua semplicità nel rappresentare degli oggetti e scambiare informazioni.

Per la realizzazione della web app sono state utilizzate le seguenti tecnologie (indicate dai requisiti del progetto):

- **HTML5/CSS**:
  - Tutte le pagine sono state sviluppate in formato HTML5 e validate.
  - Per la realizzazione del layout è stata utilizzata la libreria del Bootstrap di Twitter in combinazione a delle regole di CSS personalizzare. Si è scelto di utilizzare il Bootstrap in quanto permette di creare rapidamente un design responsivo con elementi che sono già familiari agli utenti.
  - E' stata a utilizzata l'API HTML5 webstorage per salvare la cronologia, si è scelta questa tecnologia per permette all'utente di visualizzare rapidamente le ricerche precedenti (anche offline).

- **AJAX**:
  - Sono state implementate più chiamate di tipo GET e POST, per scambiare dati in formato JSON. E' stata preferita questa tecnologia per la sua facile implementazione e perché già presente nel Bootrap di Twitter.

- **NodeJS**:
  - E' stata utilizzata questa tecnologia per creare il back-end della web-app.
  - Sono stati utilizzati inoltre moduli "Express" e "Twit", il primo per avere un deploy rapido del back-end e il secondo per avere un'interfaccia con l'API di Twitter evitando di dover gestire manualmente ogni singola richiesta e l'autenticazione.

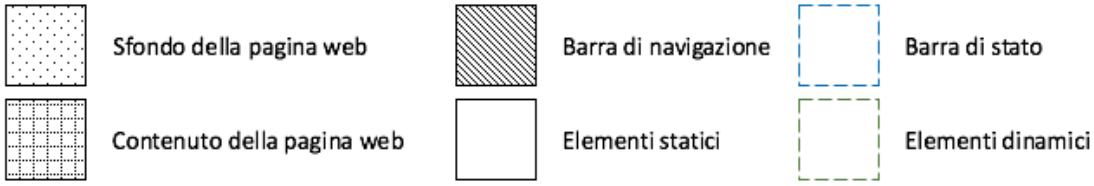
Sono state utilizzate inoltre le seguenti tecnologie (non richieste nelle specifiche del progetto):

- **SVG**:
  - E’ stata utilizzata questa tecnologia per realizzare dei grafici di tipo “donutchart”. Si è preferita rispetto ad altre in quanto manipolando esclusivamente due valori (percentuale e colore) è stato possibile realizzare numerosi grafici a fronte di un basso costo computazionale.

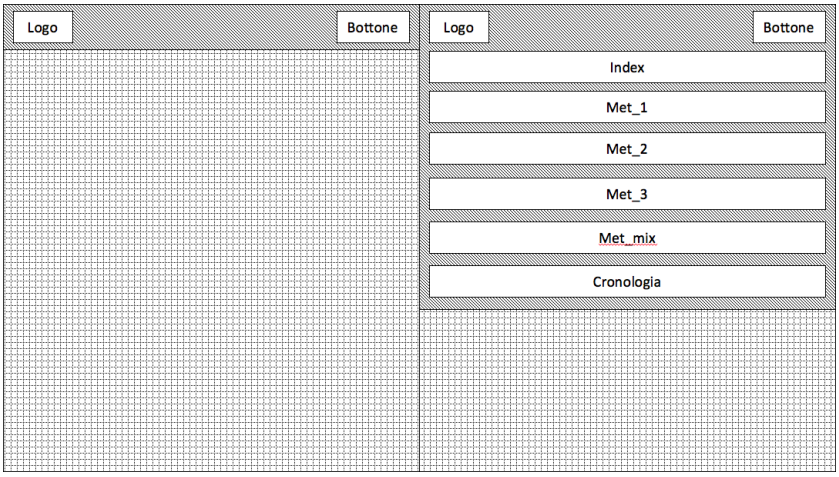
- **API Google Chart**:
  - Questa tecnologia è stata utilizzata per graficare i dati raccolti con le varie metriche e classificazioni. Si è preferita rispetto ad altre soluzioni per il suo basso costo computazionale, in quanto l’elaborazione del grafico viene fatta dai server di Google, i quali restituiscono un immagine in formato SVG.

2. Interfacce

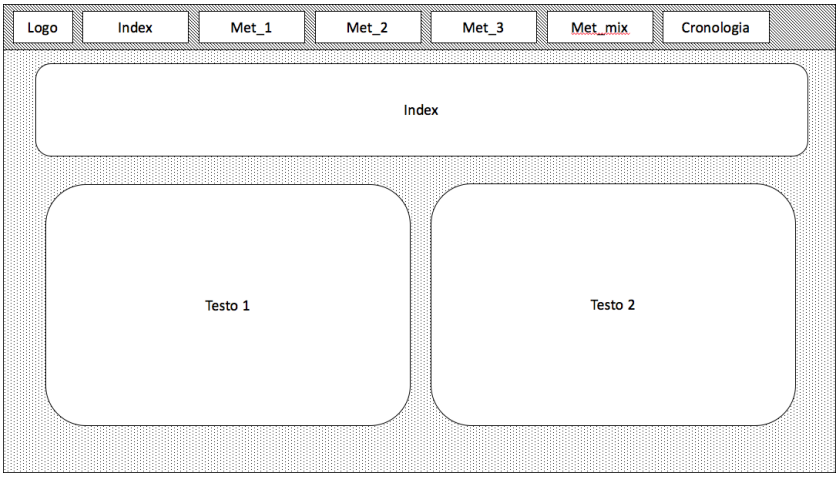
Nello sviluppo della web-app state predisposte 4 interfacce principali rappresentati con la seguente legenda:



Tutte le interfacce sono state realizzate utilizzando il Bootstrap di Twitter, implementando un foglio di stile personalizzato che va a personalizzare alcuni elementi del bootstrap.

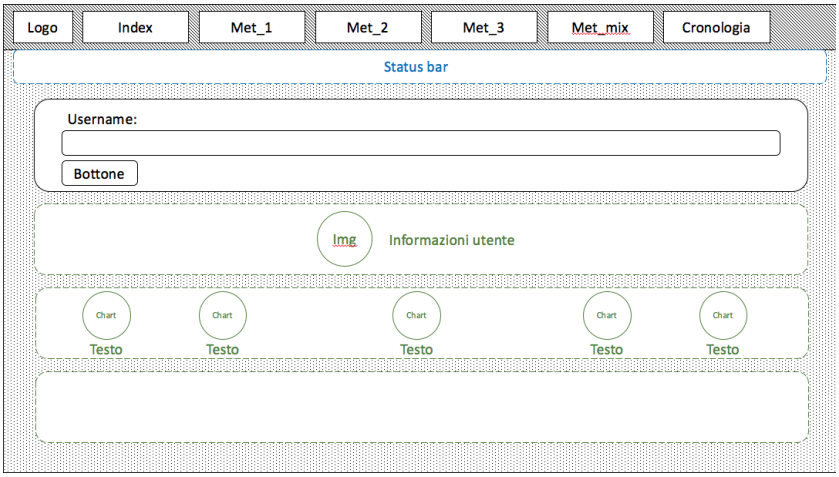


Una caratteristica comune a tutte le interfacce è la presenza di una barra di navigazione (navbar) nella porzione superiore della pagina, che permette all'utente di spostarsi fra le varie sezioni della web-app. La navbar è responsiva, e cambia disposizione in base alla dimensione del display del dispositivo.



Pagina di Index

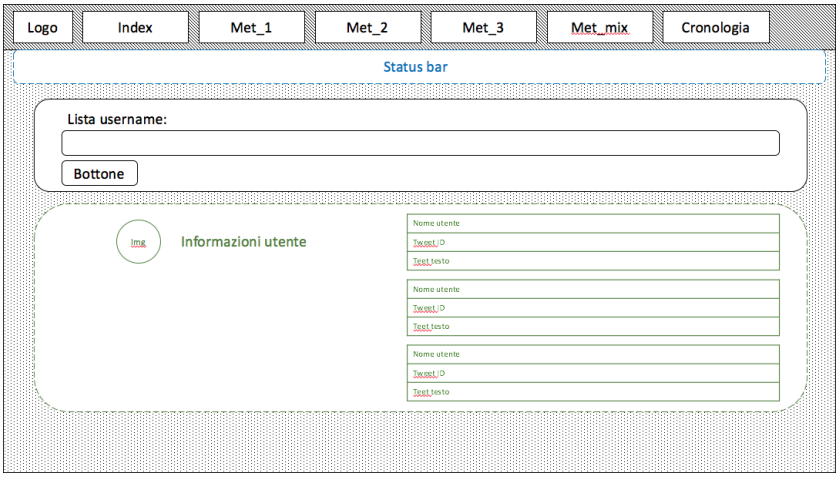
La pagina di index è composta principalmente da due sezioni che illustrano il funzionamento della web-app.



Interfaccia Metrica (utente singolo)

Questo tipo di interfaccia è presente in più pagine, è composte da:

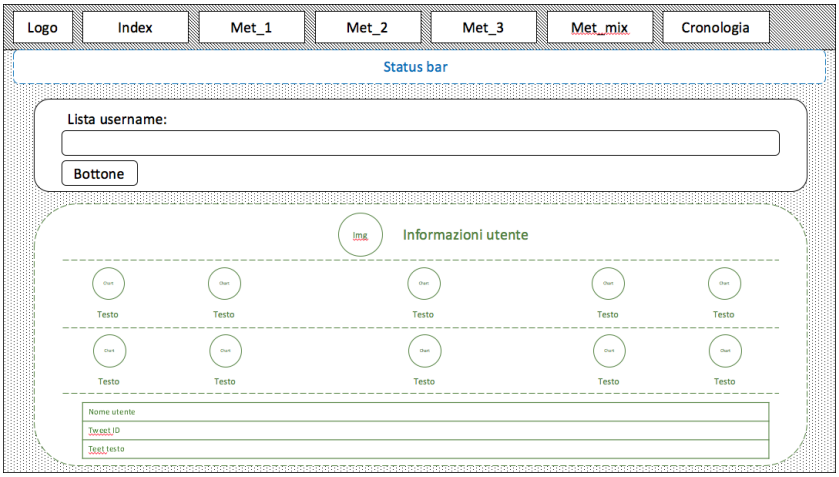
- Un form che permette l'inserimento di un username.
- Una barra di stato che viene resa quando è necessario mostrare un messaggio all'utente.
- Tre elementi dinamici che descrivo le informazioni dell'utente, la classificazione dei post e la metrica calcolata.



Interfaccia Lista Mentions (utenti multipli)

Questo tipo di interfaccia è presente solo in una pagina ed è composte da:

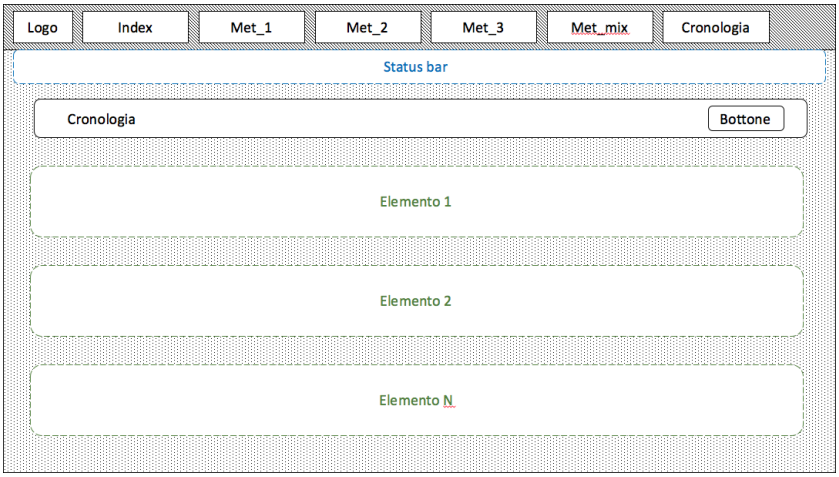
- Un form che permette l'inserimento di più username.
- Una barra di stato che viene resa quando è necessario mostrare un messaggio all'utente.
- N elementi dinamici per N username inseriti, ogni elemento dinamico mostra le informazioni sull'utente e le mantions (in relazione agli altri utenti).



Interfaccia Metriche Mix (utenti multipli)

Questo tipo di interfaccia è presente solo in una pagina ed è composte da:

- Un form che permette l'inserimento di più username.
- Una barra di stato che viene resa quando è necessario mostrare un messaggio all'utente.
- N elementi dinamici per N username inseriti, ogni elemento dinamico mostra le informazioni sull'utente, la classificazione dei post, il calcolo di metriche multiple e le mantions (in relazione agli altri utenti).



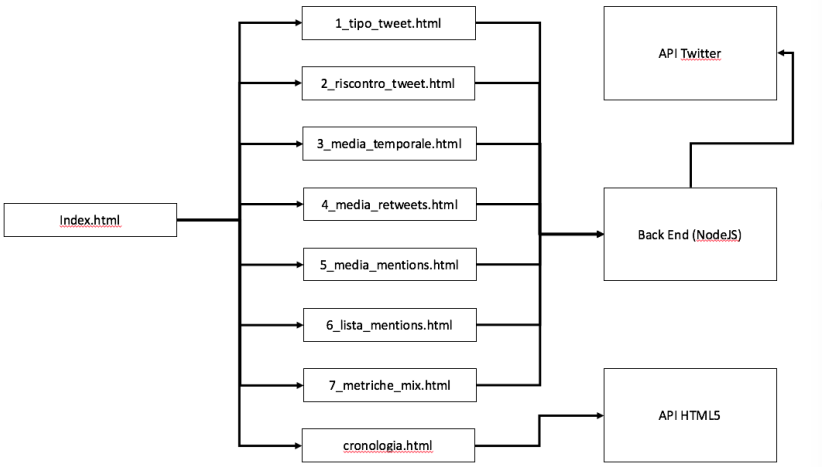
Interfaccia Cronologia

Questo tipo di interfaccia è presente solo in una pagina ed è composte da:

- Un form che permette l'inserimento di più username. Una barra di stato che viene resa quando è necessario mostrare un messaggio all'utente.
- N elementi dinamici per N username inseriti, ogni elemento dinamico mostra le informazioni sull'utente, la classificazione dei post, il calcolo di metriche multiple e le mantions (in relazione agli altri utenti).

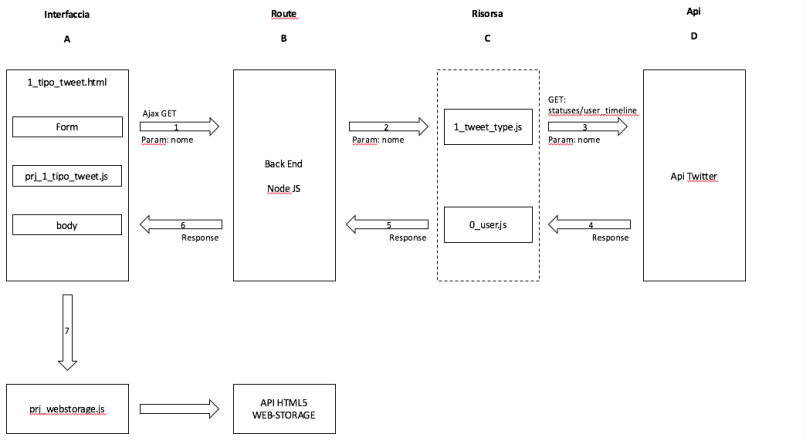
3. Architettura

3.1. Diagramma dell'ordine gerarchico delle risorse



La web-app si compone di una pagina di index, una pagina per la cronologia e 7 pagine il calcolo delle metriche. Quest'ultime rappresentano un interfaccia del back-end in NodeJs, il quale viene utilizzato per interrogare l'API di twitter e calcolare le metriche.

3.2. Descrizione delle risorse

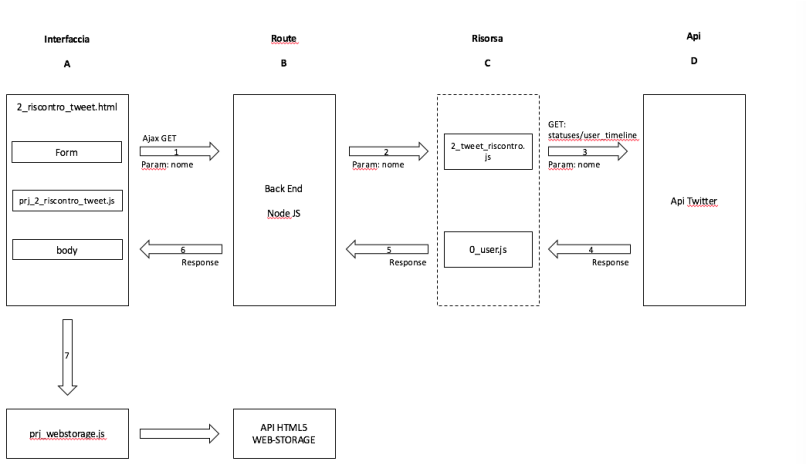


Risorsa: /tweet_type

Metrica: Classificazione dei tweets per tipo di contenuto (Singolo Utente)

1. Alla pressione del bottone "Submit", lo script "prj_1_tipo_tweet.js" si otterrà il valore del campo "inp_name" e lo utilizzerà come parametro nella richiesta di tipo XMLHttpRequest con metodo GET al back-end.
2. Il back-end provvederà a ricevere la richiesta ed identificare la route corretta.
3. Verrà quindi eseguita una richiesta all'API di Twitter tramite lo script "1_tweet_type.js".

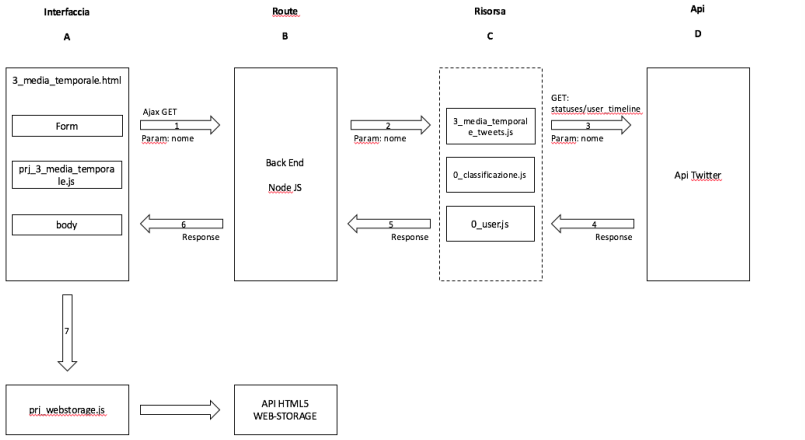
4. L'API di twitter risponde con dei risultati in formato JSON.
5. I risultati forniti dall'API vengono vengono utilizzati per:
   - Ottenere delle informazioni sull'utente (sfruttando lo script 0_user.js).
   - Calcolare la classificazione dei tweets.
   - Generare un array contenente i tipi di dato per ciascun post (da graficare).
6. Il back-end fornisce la risposta allo script "prj_1_tipo_tweet.js " il quale si occuperà di inserire le informazioni nel body della pagina.
7. Viene aggiunta alla cronologia una voce relativa alla metrica calcolata, l'utente richiesto e l'orario di esecuzione.



Risorsa: /tweet_riscontro

Metrica: Classificazione dei tweets in base al riscontro (Singolo Utente)

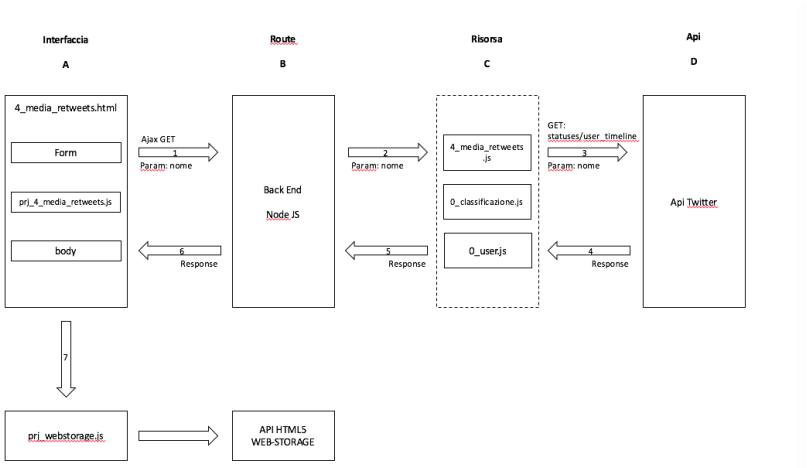
1. Alla pressione del bottone "Submit", lo script "prj_2_riscontro_tweet.js" si otterrà il valore del campo "inp_name" e lo utilizzerà come parametro nella richiesta di tipo XMLHttpRequest con metodo GET al back-end.
2. Il back-end provvederà a ricevere la richiesta ed identificare la route corretta.
3. Verrà quindi eseguita una richiesta all'API di Twitter tramite lo script "2_tweet_riscontro.js ".
4. L'API di twitter risponde con dei risultati in formato JSON.
5. I risultati forniti dall'API vengono vengono utilizzati per:
   - Ottenere delle informazioni sull'utente (sfruttando lo script 0_user.js).
   - Calcolare la classificazione dei tweets.
   - Generare un array contenente i riscontri per ciascun post (da graficare).
6. Il back-end fornisce la risposta allo script "prj_2_riscontro_tweet.js" il quale si occuperà di inserire le informazioni nel body della pagina.
7. Viene aggiunta alla cronologia una voce relativa alla metrica calcolata, l'utente richiesto e l'orario di esecuzione.



Risorsa: /media_temporale

Metrica: Calcolo della media degli intervalli temporali tra i tweet postati (Singolo Utente)

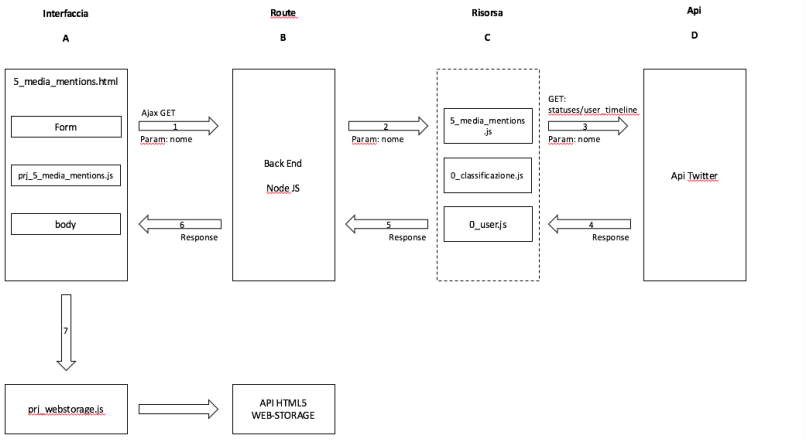
1. Alla pressione del bottone “Submit”, lo script “prj_3_media_temporale.js ” si otterrà il valore del campo “inp_name” e lo utilizzerà come parametro nella richiesta di tipo XMLHttpRequest con metodo GET al back-end.
2. Il back-end provvederà a ricevere la richiesta ed identificare la route corretta.
3. Verrà quindi eseguita una richiesta all’API di Twitter tramite lo script “3_media_temporale_tweets.js ”.
4. L’API di twitter risponde con dei risultati in formato JSON.
5. I risultati forniti dall’API vengono vengono utilizzati per:
   - Ottenere delle informazioni sull’utente (sfruttando lo script 0_user.js).
   - Calcolare la classificazione dei post (sfruttando lo script 0_classificazione.js).
   - Calcolare la media degli intervalli temporali tra i tweet postati.
   - Generare un array contenente le medie temporali parziali (da graficare).
6. Il back-end fornisce la risposta allo script “prj_3_media_temporale.js“ il quale si occuperà di inserire le informazioni nel body della pagina.
7. Viene aggiunta alla cronologia una voce relativa alla metrica calcolata, l’utente richiesto e l’orario di esecuzione.



Risorsa: /media_retweet

Metrica: Calcolo della media dei retweet tra i tweet postati (Singolo Utente)

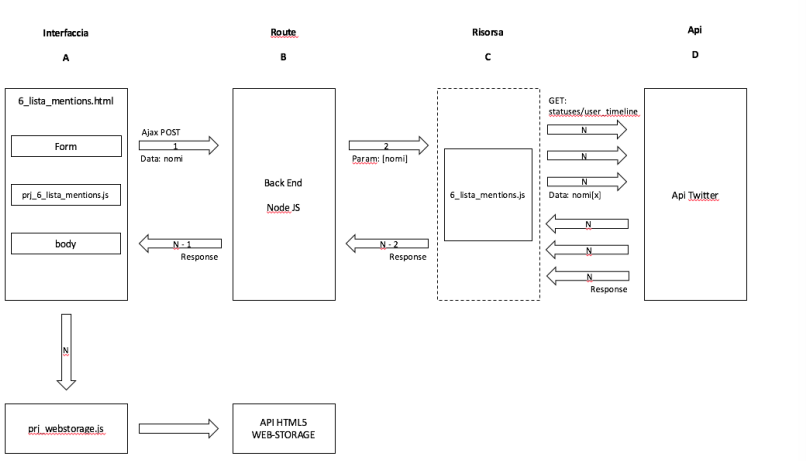
1. Alla pressione del bottone “Submit”, lo script “prj_4_media_retweets.js ” si otterrà il valore del campo “inp_name” e lo utilizzerà come parametro nella richiesta di tipo XMLHttpRequest con metodo GET al back-end.
2. Il back-end provvederà a ricevere la richiesta ed identificare la route corretta.
3. Verrà quindi eseguita una richiesta all’API di Twitter tramite lo script “4_media_retweets.js ”.
4. L’API di twitter risponde con dei risultati in formato JSON.
5. I risultati forniti dall’API vengono vengono utilizzati per:
   - Ottenere delle informazioni sull’utente (sfruttando lo script 0_user.js).
   - Calcolare la classificazione dei post (sfruttando lo script 0_classificazione.js).
   - Calcolare la media media dei retweet tra i tweet postati.
   - Generare un array contenente il numero dei retweet per ciascun post (da graficare).
6. Il back-end fornisce la risposta allo script “prj_4_media_retweets.js “ il quale si occuperà di inserire le informazioni nel body della pagina.
7. Viene aggiunta alla cronologia una voce relativa alla metrica calcolata, l’utente richiesto e l’orario di esecuzione.



Risorsa: /media_mentions

Metrica: Calcolo della media delle mantions tra i tweet postati (Singolo Utente)

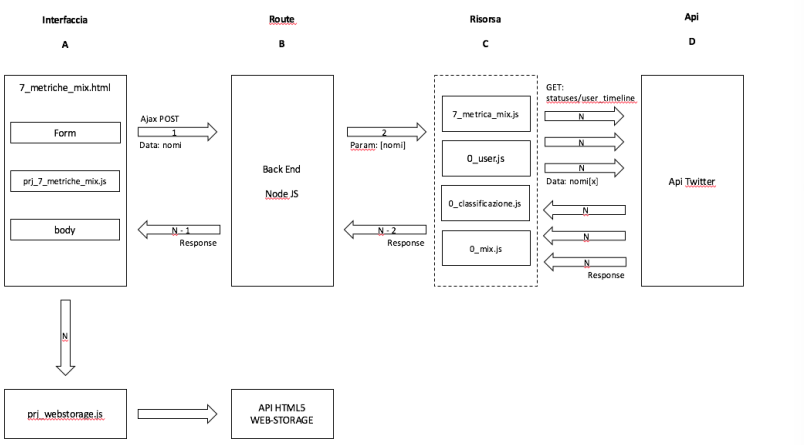
1. Alla pressione del bottone “Submit”, lo script “prj_5_media_mentions.js” si otterrà il valore del campo “inp_name” e lo utilizzerà come parametro nella richiesta di tipo XMLHttpRequest con metodo GET al back-end.
2. Il back-end provvederà a ricevere la richiesta ed identificare la route corretta.
3. Verrà quindi eseguita una richiesta all’API di Twitter tramite lo script “5_media_mentions.js ”.
4. L’API di twitter risponde con dei risultati in formato JSON.
5. I risultati forniti dall’API vengono vengono utilizzati per:
   - Ottenere delle informazioni sull’utente (sfruttando lo script 0_user.js).
   - Calcolare la classificazione dei post (sfruttando lo script 0_classificazione.js).
   - Calcolare la media media delle mantions tra i tweet postati.
   - Generare un array contenente il numero delle mantions per ciascun post (da graficare).
6. Il back-end fornisce la risposta allo script “ prj_5_media_mentions.js “ il quale si occuperà di inserire le informazioni nel body della pagina.
7. Viene aggiunta alla cronologia una voce relativa alla metrica calcolata, l’utente richiesto e l’orario di esecuzione.



Risorsa: /lista_mentions

Metrica: Identificazione dei tweet nei quali uno dei membri della lista menziona un'altro membro (Multi Utente)

1. Alla pressione del bottone "Submit", lo script "prj_6_lista_mentions.js" si otterrà il valore del campo "inp_names" e lo utilizzerà come dato nella richiesta di tipo XMLHttpRequest con metodo POST al back-end.
   (Si preferisce il metodo in modo che la lista degli username non sia limitata alla lunghezza massima di un URL)
2. Il back-end provvederà a ricevere la richiesta ed identificare la route corretta.
3. Verranno eseguite N richieste all'API di Twitter per N utenti, tramite lo script "6_lista_mentions.js".
4. L'API di twitter risponde con dei risultati in formato JSON.
5. Lo script 6_lista_mentions.js, una volta ricevute tutte le risposte dall'API, le utilizza per:
   - Ottenere delle informazioni sull'utente.
   - Identificare i tweet dove un utente menzione uno dei membri della lista fornita.
6. Il back-end fornisce la risposta allo script "prj_6_lista_mentions.js " il quale si occuperà di inserire le informazioni nel body della pagina.
7. Viene aggiunta alla cronologia una voce relativa alla metrica calcolata, l'utente richiesto e l'orario di esecuzione.



Risorsa: /metrica_mix

Metrica: Calcolo di tutte le classificazioni e metriche (Multi Utente)

1. Alla pressione del bottone “Submit”, lo script “prj_7_metriche_mix.js” si otterrà il valore del campo “inp_names” e lo utilizzerà come dato nella richiesta di tipo XMLHttpRequest con metodo POST al back-end.
   (Si preferisce il metodo in modo che la lista degli username non sia limitata alla lunghezza massima di un URL)
2. Il back-end provvederà a ricevere la richiesta ed identificare la route corretta.
3. Verranno eseguite N richieste all’API di Twitter per N utenti, tramite lo script “7_metrica_mix.js ”.
4. L’API di twitter risponde con dei risultati in formato JSON.
5. Lo script “7_metrica_mix.js ”, una volta ricevute tutte le risposte dall’API, le utilizza per:
   - Ottenere delle informazioni sull’utente (sfruttando lo script 0_user.js).
   - Calcolare la classificazione dei post (sfruttando lo script 0_classificazione.js).
   - Calcolare le varie metriche sui post (sfruttando lo script 0_mix.js).
6. Il back-end fornisce la risposta allo script “prj_7_metriche_mix.js“ il quale si occuperà di inserire le informazioni nel body della pagina.

   Viene aggiunta alla cronologia una voce relativa alla metrica calcolata, l’utente richiesto e l’orario di esecuzione.

4. Codice

Frammenti del codice più significativo

4.1. HTML

```
<!-- NAVBAR inizio -->
<nav class="navbar-default">
    <div class="container-fluid">
        <div class="navbar-header">
            <a class="navbar-brand" href="index.html">PWM</a>
            <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#navbarCollapse">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            </button>
        </div>
        <div class="collapse navbar-collapse" id="navbarCollapse">
            <ul class="nav navbar-nav">
                <li class="active"><a href="index.html">Home</a></li>
                <li><a href="1_tipo_tweet.html">Tipo Tweet</a></li>
                <li><a href="2_riscontro_tweet.html">Riscontro Tweet</a></li>
                <li><a href="3_media_temporale.html">Media Temporale</a></li>
                <li><a href="4_media_retweets.html">Media Retweets</a></li>
                <li><a href="5_media_mentions.html">Media Mentions</a></li>
                <li><a href="6_lista_mentions.html">Lista Mention</a></li>
                <li><a href="7_metriche_mix.html">Metriche Mix</a></li>
                <li><a href="cronologia.html">Cronologia</a></li>
            </ul>
        </div>
    </div>
</nav>
<!-- NAVBAR Fine -->
```

Navbar (comune a tutte le pagine):
La navbar è composta da un logo, un bottone per aprire/collasare il menù (quando il display del dispositivo è troppo piccolo per visualizzare l'intera barra) e dei collegamenti alle altre pagine.

```
<!-- Bootstrap -->
<link href="css/bootstrap.min.css" rel="stylesheet">
<link href="css/custom.css" rel="stylesheet">
```

Collegamenti ai fogli di stile (comune a tutte le pagine):
Il primo collegamento fa riferimento al principale foglio di stile del bootstrap, il secondo collegamento fa riferimento a un foglio di stile usato per specificare delle regole specifiche o delle regole di ovverride.

```
<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>
<!-- Include all compiled plugins (below), or include individual files as needed -->
<script src="js/bootstrap.min.js"></script>
<!-- JS per Google Charts -->
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>

<script src="./js/prj_webstorage.js"></script>
```

Collegamenti agli script (comune a tutte le pagine):
Il primo collegamento fa riferimento alla libreria jquery (utilizzata da bootstrap e da gli script che effettuano richieste AJAX), il secondo collegamento si riferisce alla libreria del bootstrap.
Il terzo collegamento fa riferimento all'api di google per la generazione dei grafici.
Il quarto collegamento fa riferimento allo script che gestisce la cronologia della web-app.
Ogni pagina ha un ulteriore collegamento ad uno script specifico per le sue funzionalità.

```
<div id="message-container" class="hidden">
    <div id="message" style="padding: 5px;">
    </div>
</div>
```

Container per la visualizzazione di messaggi di stato (comune a tutte le pagine):
Questo container viene visualizzato solo in presenza di un messaggio per l'utente.

```
<div class="container">
    <div class="form-group">
        <label for="inp_name">Twitter Username:</label>
        <input id="inp_name" type="text" class="form-control">
    </div>

    <div class="form-group">
        <button id="btn_submit" type="submit" class="btn btn-default" onClick="fx_run()">Submit</button>
    </div>
</div>



<div id="conn_user" class="container hidden">
    <div class="row">
        <div class="col-md-12">
            <div class="horizontal-center-div">
                <img id="user_img" src="#" class="img-circle img_profilo" alt="Immagine del profilo">
                <span class="align-middle" id="user_info"></span>
            </div>
        </div>
    </div>
</div>


<div id="con_riscontro" class="container hidden">
    <div class="row">
        <div class="col-md-4 col-sm-4 col-xs-4">
            <div id="risc_like_chart_div" class="counter"></div>
            <div id="risc_like_text_div" class="text-center"></div>
        </div>

        <div class="col-md-4 col-sm-4 col-xs-4">
            <div id="risc_retweetted_chart_div" class="counter"></div>
            <div id="risc_retweetted_text_div" class="text-center"></div>
        </div>

        <div class="col-md-4 col-sm-4 col-xs-4">
            <div id="risc_reply_chart_div" class="counter"></div>
            <div id="risc_reply_text_div" class="text-center"></div>
        </div>

    </div>
</div>


<div id="con_chart" class="container hidden">
    <div class="row">
        <div class="col-md-12">
            <div id="chart_div_1" class="chart"></div>
        </div>
    </div>
</div>
```

Frammento (pagina 2_riscontro_tweet.html):
In questo frammento viene mostrata la composizione tipica del corpo di una pagina.
In tutte le pagine (ad eccezione dell'index e della cronologia) viene presentato all'utente un container contenente un form, il quale verrà utilizzato per specificare gli username degli account per i quali verrano calcolate le metriche.

4.2. CSS

```css
body {
    background: #ececec;
}

h3 {
    margin-top: 10px;
    margin-bottom: 10px;
}

p {
    margin: 0 0 0px;
}

ol, ul{
    margin-bottom: 0px;
}
```

Definizione di regole specifiche per i tag delle pagine.

```css
.container {
    background: #ffffff;
    border-radius: 5px;
    margin-top: 20px;

    padding-top: 10px;
    padding-bottom: 10px;
}
```

Classe applicata ai container.

```css
.horizontal-center-div {
    display: flex;
    justify-content: center;
    align-items: center;
}

.horizontal-center{
    display: flex;
    align-items: center;
}
```

Classi utilizzate per centrare gli elementi.

```css
.hidden{
    visibility:hidden;
}

.visible{
    visibility:visible;
}
```

Classi per la visibilità dei vari elementi.

```css
.chart {
    width: 100%;
}

.counter {
    transform: rotate(-90deg);
    height: 100px;
}

.img_profilo{
    max-height: 100px;
    float:left;
    padding-right: 10px;
}

.img_profilo_small{
    max-height: 50px;
    float:left;
    padding-right: 10px;
}
```

```css
.btn_cancella {
    float: right;
    margin-top: 5px;
}

.card {                          /* INDEX */
    background: #ffffff;
    padding-top: 10px;
    padding-bottom: 10px;
    padding-left: 10px;
    padding-right: 10px;
    border-radius: 5px;
    margin-top: 20px;
}

.metrics-row{                    /* Metrica 7 */
    background: #ffffff;
    padding-top: 10px;
    padding-bottom: 10px;
}
```

Classi specifiche per i vari elementi delle pagine.

4.3. JAVASCRIPT

```
function handle_data(data){

    document.getElementById("conn_user").classList.remove("hidden");
    document.getElementById("con_classificazione").classList.remove("hidden");
    document.getElementById("con_chart").classList.remove("hidden");

    user_data (data);
    classificazione (data);
    add_crono ("Tipo Tweet", data.data.user.screen_name);

    var raw = data.data.raw_array;
    var arr = [["Post", "Foto", "Video", "Link", "Testo"]];
    for (i=0; i<raw.length; i++){
        arr.push ([i + 1, raw[i].photo, raw[i].video, raw[i].link, raw[i].text]);
    }

    var options = {
        title: "Tipi di dati contenuti nei Tweets",
        hAxis: {title: "Singoli Tweets",  titleTextStyle: {color: '#333'}},
        vAxis: {minValue: 0}
    };

    var element = "chart_div_1";

    $(window).resize(function(){
        draw_area_chart();
    });

    google.charts.setOnLoadCallback(draw_area_chart);


    function draw_area_chart() {
        var data = google.visualization.arrayToDataTable(arr);

        var chart = new google.visualization.SteppedAreaChart(document.getElementById(element));
        chart.draw(data, options);
    }

}
```

Generazione grafici con API Google Charts (prj_1_tipo_tweet.js)

```
function add_crono (data, utente){
    var cronologia = [];

    if (localStorage.length != 0) {
        cronologia = JSON.parse(localStorage.getItem("cronologia"));
    }

    var today = new Date();
    var date = str_pad(today.getDate())+"/"+str_pad((today.getMonth()+1))+"/"+today.getFullYear();
    var time = str_pad(today.getHours()) + ":" + str_pad(today.getMinutes())+":"+str_pad(today.getSeconds());
    var dateTime = date + " " + time;

    cronologia.push ({'time':dateTime, 'metrica':data, 'utente':utente});
    localStorage.setItem("cronologia", JSON.stringify(cronologia));
}

function str_pad(n) {
    return String("00" + n).slice(-2);
}


function get_crono (){
    var cronologia = [];

    if (localStorage.length != 0) {
        cronologia = JSON.parse(localStorage.getItem("cronologia"));
    }

    return cronologia;
}



function reset_crono (){
    localStorage.removeItem("cronologia");
}
```

Cronologia con API Webstorage (prj_webstorage.js)

```
function fx_run (){

    var name = document.getElementById("inp_name").value;

    if (name == "" || name == null){
        error_man ("Il campo è vuoto");

    } else {
        status_man ("Richiesta in corso...");

        $.ajax({
            url: 'http://127.0.0.1:8000/tweet_type?nome=' + name,
        })
        .done(function(data, err) {
            var jdata = JSON.parse(data)

            if (jdata.error == 0){
                status_man ("");
                handle_data(jdata);

            } else {
                error_man ("Errore: " + JSON.stringify(jdata.data));
            }

        })
        .fail(function(err) {
            error_man ("Errore nella richiesta AJAX");
        })
    }

}
```

Richiesta GET con AJAX (prj_1_tipo_tweet.js)

```
function fx_run (){

    var nomi = document.getElementById("inp_names").value;

    if (nomi == "" || nomi == null){
        error_man ("Il campo è vuoto");

    } else {
        status_man ("Richiesta in corso...");

        $.ajax({
            type: 'POST',
            data: 'nomi=' + nomi,
            url: 'http://127.0.0.1:8000/metrica_mix'
        })
        .done(function(data, err) {

            var jdata = JSON.parse(data)

            if (jdata.error == 0){
                status_man ("");
                handle_data(jdata);

            } else {
                error_man ("Errore: " + JSON.stringify(jdata.data));
            }

        })
        .fail(function(err) {
            error_man ("Errore nella richiesta AJAX");
        })
    }

}
```

Richiesta POST con AJAX (prj_webstorage.js)

4.4. API

```
DellXS3:~ user$ curl http://127.0.0.1:8000/tweet_type?nome=SNitopi
{"error":0,"data":{"raw_array":[{"photo":0,"video":0,"link":0,"text":1},{"photo"
:0,"video":0,"link":0,"text":1},{"photo":0,"video":0,"link":1,"text":1},{"photo"
:0,"video":0,"link":1,"text":1},{"photo":0,"video":0,"link":0,"text":1},{"photo"
:2,"video":0,"link":0,"text":0},{"photo":0,"video":1,"link":0,"text":1},{"photo"
:0,"video":0,"link":1,"text":0},{"photo":0,"video":0,"link":0,"text":1},{"photo"
:1,"video":0,"link":0,"text":1},{"photo":0,"video":0,"link":0,"text":1}],"photo"
:2,"video":1,"mixed":0,"link":3,"text":5,"tot_posts":11,"user":{"user_img":"http
s://abs.twimg.com/sticky/default_profile_images/default_profile_400x400.png","sc
reen_name":"SNitopi","name":"Salvatore Nitopi","statuses_count":11,"followers_co
unt":0}}}
DellXS3:~ user$
```

Richiesta GET all'endpoint /tweet_type

```
DellXS3:~ user$ curl http://127.0.0.1:8000/media_temporale?nome=SNitopi
{"error":0,"data":{"raw_array":[7274.683333333333,2.183333333333333,41.683333333
33333,699.8,3821.2833333333333,5.466666666666667,1.4166666666666667,638.41666666
66666,31.25,2.6666666666666665],"media":1138.0772727272727,"max":7274.6833333333
33,"class":{"favorited":1,"retweeted":0,"replyed":1,"photo":2,"video":1,"mixed":
0,"link":3,"text":5,"tot_posts":11},"user":{"user_img":"https://abs.twimg.com/st
icky/default_profile_images/default_profile_400x400.png","screen_name":"SNitopi"
,"name":"Salvatore Nitopi","statuses_count":11,"followers_count":0}}}
DellXS3:~ user$
```

Richiesta GET all'endpoint /media_temporale

```
DellXS3:~ user$ curl --data 'nomi=realDonaldTrump, WhiteHouse' http://127.0.0.1:8000/metrica_mix
{"error":0,"data":[{"user":{"user_img":"https://pbs.twimg.com/profile_images/874276197357596672/kUuht00m_400x4
00.jpg","screen_name":"realDonaldTrump","name":"Donald J. Trump","statuses_count":36639,"followers_count":4529
8026},"class":{"favorited":180,"retweeted":197,"replyed":2,"photo":3,"video":13,"mixed":0,"link":145,"text":36
,"tot_posts":197},"m_temporale":{"media":217.05769881556682,"max":1751.1},"m_retweets":{"media":20112.89847715
736,"max":55523},"m_mantions":{"media":0.27918781725888325,"max":4},"l_mantions":[{"muser":"WhiteHouse","tweet
_id":"943908041803657216","tweet_text":"THANK YOU to everyone who joined me at the @WhiteHouse yesterday. Toge
ther, we are MAKING AMERICA GREAT AGAIN!… https://t.co/aykO3m2rVg"},{"muser":"WhiteHouse","tweet_id":"93900565
9982348291","tweet_text":"Tonight, @FLOTUS Melania and I were thrilled to welcome so many wonderful friends to
 the @WhiteHouse – and wish the… https://t.co/NlHEGuiWCF"},{"muser":"WhiteHouse","tweet_id":"93886175803857715
2","tweet_text":"Today, the U.S. flag flies at half-staff at the @WhiteHouse, in honor of National Pearl Harbo
r Remembrance Day.… https://t.co/561NBdqSsb"},{"muser":"WhiteHouse","tweet_id":"938469354089385985","tweet_tex
t":"Join me live from the @WhiteHouse via #Periscope\nhttps://t.co/wLXKoISSTG"},{"muser":"WhiteHouse","tweet_i
d":"936646898408218626","tweet_text":"RT @WhiteHouse: President Trump proclaims today as #WorldAIDSDay: https:
//t.co/AMqW8mWiak https://t.co/XGp8EelNzz"},{"muser":"WhiteHouse","tweet_id":"936339260218724353","tweet_text"
:"Today, it was my great honor to meet with the Crown Prince of Bahrain at the @WhiteHouse. Bahrain and the Un
ited St… https://t.co/xShHfc3Yxo"}]},{"user":{"user_img":"https://pbs.twimg.com/profile_images/822251886988267
520/28GpoRsy_400x400.jpg","screen_name":"WhiteHouse","name":"The White House","statuses_count":2668,"followers
_count":16205294},"class":{"favorited":108,"retweeted":200,"replyed":0,"photo":36,"video":10,"mixed":0,"link":
73,"text":81,"tot_posts":200},"m_temporale":{"media":148.01616666666666,"max":1398.7333333333333},"m_retweets"
:{"media":4894.575,"max":48859},"m_mantions":{"media":0.735,"max":5},"l_mantions":[{"muser":"realDonaldTrump",
"tweet_id":"946256039766261760","tweet_text":"RT @realDonaldTrump: “Arrests of MS-13 Members, Associates Up 83
% Under Trump” \nhttps://t.co/70iPHy2Yqn"},{"muser":"realDonaldTrump","tweet_id":"946256031868444672","tweet_t
ext":"RT @realDonaldTrump: “On 1/20 - the day Trump was inaugurated - an estimated 35,000 ISIS fighters held a
pprox 17,500 square miles of territ…"},{"muser":"realDonaldTrump","tweet_id":"945287718313103363","tweet_text"
:"RT @realDonaldTrump: MERRY CHRISTMAS!!! https://t.co/mYtV5GNdLl"},{"muser":"realDonaldTrump","tweet_id":"945
140987931910145","tweet_text":"RT @realDonaldTrump: MERRY CHRISTMAS!! https://t.co/xa2qxcisVV"},{"muser":"real
DonaldTrump","tweet_id":"944308921413533696","tweet_text":"RT @realDonaldTrump: Today, it was my great honor t
o sign the largest TAX CUTS and reform in the history of our country. Full remarks: http…"},{"muser":"realDona
ldTrump","tweet_id":"943910184988086272","tweet_text":"RT @realDonaldTrump: THANK YOU to everyone who joined m
e at the @WhiteHouse yesterday. Together, we are MAKING AMERICA GREAT AGAIN! https:/…"},{"muser":"realDonaldTr
ump","tweet_id":"943598233011580929","tweet_text":"RT @realDonaldTrump: WE ARE MAKING AMERICA GREAT AGAIN! htt
ps://t.co/HY353gXV0R"},{"muser":"realDonaldTrump","tweet_id":"943544120664121344","tweet_text":"RT @realDonald
Trump: We are delivering HISTORIC TAX RELIEF for the American people!\n#TaxCutsandJobsAct https://t.co/lLgATrC
h5o"},{"muser":"realDonaldTrump","tweet_id":"943534711669972994","tweet_text":"RT @realDonaldTrump: Together,
we are MAKING AMERICA GREAT AGAIN! https://t.co/47k9i4p3J2"},{"muser":"realDonaldTrump","tweet_id":"9432139900
16159744","tweet_text":"RT @realDonaldTrump: Congratulations to Paul Ryan, Kevin McCarthy, Kevin Brady, Steve
Scalise, Cathy McMorris Rodgers and all great House R…"},{"muser":"realDonaldTrump","tweet_id":"94292183254979
```

Richiesta POST all'endpoint /metrica_mix (frammento)

4.5. Node.js

```
var express = require('express');
var router = express.Router();

var config = require ('../config');

var Twit = require ('twit');
var T = new Twit (config.api_twitter);

// https://apigee.com/console

////////////////////////////////////////////////////////////////////////////////
// METRICA 1

router.get('/tweet_type', function(req, res, next) {

    var nome = req.query.nome;

    if (typeof nome == 'undefined' || nome == null){
        res.send( JSON.stringify({error:1, 'data':'nome non fornito'}) + '\n' );

    } else {

        var m = require ('../metriche/1_tweet_type');

        m.calcola (T, nome, function(err, data) {
            if (err) {                                  // In caso di errore
                var msg = ""
                if ( "message" in err )
                    msg = err.message;

                console.error(err);
                res.send( JSON.stringify({'error':2, 'data':'Errore nel callback della metrica: ' + msg}) + '\n' )
            } else if  (data == null) {
                res.send( JSON.stringify({'error':3, 'data':'data non contiene informazioni'}) + '\n' )
            } else {
                res.send( JSON.stringify({'error':0, 'data':data}) + '\n' );
            }
        });
    }

});

// curl http://127.0.0.1:8000/tweet_type
// curl http://127.0.0.1:8000/tweet_type?nome=SNitopi
```

Frammento di: routes/api.js

```
module.exports = {

    calcola: function (T, nome, callback){

        T.get('statuses/user_timeline', { screen_name: nome, count: 200 }, function(err, data, response) {

            if( !err && data != "undefined" && data != null && data.length > 0) {

                var u = require ('../metriche/0_user');
                var user_info = u.get_info (data);

                var c = require ('../metriche/0_classificazione');
                var classificazione = c.classifica (data);

                var totale = 0;
                var media = 0;
                var max = 0;

                var tempi = [];
                for (var i = 0; i < data.length; i++) {
                    var d = new Date(data[i].created_at);                   // Timestamp in millisecondi
                    tempi.push(d.getTime());
                };


                var medie = [];
                for (var i = 0; i < tempi.length - 1; i++) {

                    var actual = (tempi[i] - tempi[i + 1]) / 1000 / 60      // Ottengo max per calcolare la percentuale
                    if (actual > max){
                        max = actual;
                    }

                    totale += (tempi[i] - tempi[i + 1]);                    // Per il calcolo della media totale
                    medie.push(( (tempi[i] - tempi[i + 1]) / 1000 / 60));   // Per il calcolo delle medie (in minuti)
                };

                media = (totale / data.length) / 1000 / 60;                 // Media in minuti

                var out = { 'raw_array':medie, 'media':media, 'max':max,
                            'class':classificazione, 'user':user_info
                          };
                return callback(null, out);

            }

            else {
                return callback(err);
            }

        });
    }
```

Frammento di: metriche/3_media_temporale_tweets.js

5. Conclusioni

Lo sviluppo di questo progetto è stato un'ottima occasione per mettere in pratica le nozioni acquisite durante il corso di programmazione web e mobile.

Grazie al lavoro svolto ho potuto comprendere in maniera più dettagliata il funzionamento di numerose tecnologie come HTML5, CSS (che mi hanno permesso sviluppare pagine web efficaci e responsive), JAVASCRIPT, NODEJS (per la realizzazione della parte computazionale) e il ruolo fondamentale che giocano le API nel web moderno.

6. Nota bibliografica e sitografica

Vengono elencate di seguito le principali documentazioni utilizzate per la realizzazione del progetto:

1. Bootstrap di Twitter:
   https://getbootstrap.com/docs/3.3/css/

2. API Google Chart:
   https://developers.google.com/chart/

3. Esempi sull'implementazione di Twit:
   https://github.com/ttezel/twit

4.