UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

DIPARTIMENTO DI SCIENZE FISICHE,
INFORMATICHE E MATEMATICHE
CORSO DI LAUREA IN INFORMATICA

PROGETTAZIONE E SVILUPPO DI CUSTOM FORM MEDIANTE GOOGLE APPS SCRIPT

Maxwell Oppong Asomani Manu

Tesi di Laurea

Relatori:
prof. Riccardo Martoglia
prof. Giacomo Cabri

Anno Accademico 2016/2017

Ringraziamenti

I più sentiti ringraziamenti vanno ai miei relatori, l'Ing. Riccardo Martoglia e l'Ing. Giacomo Cabri, per il supporto che mi è stato offerto durante tutto il percorso. Un ringraziamento speciale, inoltre va a famigliari ed amici, perché come un albero che senza radici non potrebbe far altro che cascare, mi sono stati vicini da sempre senza farmi mai mancare l'appoggio nelle decisioni importati della vita.

Parole chiave

Google
AppsScript
JavaScript
Form
Stylesheet

Indice

Indice delle figure

Introduzione

La parola Google ha un significato ben preciso e non è stata scelta a caso, da Larry Page e Sergey Brin, i fondatori del motore di ricerca. Nel 1997 scelsero il nome del loro prodotto prendendo spunto dalla parola Googol. Tale termine fu coniato da un matematico americano, Edward Kasner nel 1930, per indicare un numero ben preciso;
1 Googol = 10^{100}.

Kasner, impegnato nella stesura di un libro di divulgazione, "*Matematica e Immaginazione*", intendeva mostrare la differenza che sussiste tra l'infinito e un numero molto grande. Il nome Google nacque per un errore di battitura nel momento della registrazione del dominio web, i fondatori ritennero simpatico il nuovo nome e decisero di preservarlo.

Oggi Google è sinonimo di ricerca sul web, ma non è solo questo. Tra i servizi che offre, mette a disposizione una suite di applicazioni, chiamata GSuite. Sono strumenti di produttività per il cloud computing, offerti in abbonamento per le aziende. Questi prodotti sono disponibili gratuitamente agli utenti privati, ed è su questi ultimi che sposteremo la nostra attenzione. Durante la trattazione degli argomenti verrà usato il nome Google Apps per riferirsi ad una specifica categoria di applicazioni, cioè le applicazioni offerte gratuitamente agli utenti privati e facenti parte di GSuite.

Google Apps è una soluzione SaaS (Software as a Service) collaborativa e basata sul web, che personalizza la piattaforma ed il marchio proprietari di Google.

Google Apps personalizza le applicazioni per diversi settori, quali;

- Google Apps Free: Versione gratuita.
- Google Apps for Education: Applicazioni gratuite per gli istituiti scolastici.
- Google Apps for Business: Versione a pagamento.
- Google Apps for Government: Strumenti governativi certificati di collaborazione basati sul web.
- Google Apps for Nonprofit: Strumenti di collaborazione e comunicazione per le organizzazioni non profit statunitensi.

Google Apps Script è un linguaggio di scripting cloud JavaScript, che fornisce semplici modi per automatizzare le attività tra i prodotti Google e servizi di terzi. Verrà utilizzato per implementare funzionalità tra gli applicativi che saranno presentati.
Il progetto consiste nel mostrare le capacità del linguaggio e dell'ambiente di lavoro, messe in pratica su un caso specifico, quello della modifica di un form preliminare. L'intento principale è di riuscire a palesare i vantaggi che si possono trarre da un approccio a questo tipo di programmazione, senza tuttavia celarne i limiti della stessa.

Considerando gli obiettivi che ci siamo posti, troviamo senz'altro la volontà di far conoscere in maniera più approfondita particolari applicazioni, le stesse, che molti già conoscono, ed è proprio per questo che ci siamo focalizzati sulle differenze sostanziali che intercorrono tra un uso tradizionale e ciò che propone Google come alternativa a determinati strumenti. Per condurre il lettore attraverso gli obiettivi, è stato utile mostrare la flessibilità di cui sono caratterizzati i software proposti.

Il caso di studio considerato per realizzare gli obiettivi è stato un form, creato per Democenter-Sipe.
La Fondazione riunisce istituzioni, associazioni di categoria, fondazioni bancarie e oltre sessanta imprese del territorio e fa parte della rete regionale dell'alta Tecnologia

dell'Emilia-Romagna. Ha sede all'interno del campus della Facoltà di Ingegneria 'Enzo Ferrari' di Modena.

Vi era la necessità di mettere a disposizione una piattaforma per la condivisione dei contatti durante le attività dei partner di progetto, dove con contatto si considera una interazione o uno scambio di recapiti fra le parti.

Struttura della tesi

La tesi si compone di due parti, la prima è inerente alla descrizione della piattaforma e degli strumenti che sono stati utilizzati come approccio al progetto, mentre la seconda parte è maggiormente a scopo implementativo.

Parte Prima: Stato dell'arte

- **Capitolo 1 - Piattaforma e Strumenti Utilizzati.** Il capitolo in questione mira ad una presentazione più approfondita degli applicativi Google, spiegando le motivazioni che hanno portato al loro utilizzo all'interno dell'elaborato. Versatilità è la caratteristica principale di tali strumenti, e riuscire a comprenderli in maniera approfondita è necessario per poterne sfruttare appieno le potenzialità.

- **Capitolo 2 - Google Apps Script.** Il capitolo secondo è stato riservato per approfondire ciò che rappresenta l'ambiente di esecuzione ed il linguaggio che viene utilizzato. Il connubio tra le due, permette di far interagire simultaneamente, ipoteticamente tutte le applicazioni messe a disposizione da Google, e di farle funzionare in maniera assolutamente precisa, per di più permette di farlo con sufficiente semplicità, celando la maggior parte delle complicazioni.

Parte seconda: L'applicazione

- **Capitolo 3 - Progettazione.** In questo capitolo vengono spiegati quali sono i requisiti dell'applicazione che intendiamo sviluppare.

- **Capitolo 4 - Implementazione.** Il capitolo quarto, è improntato alla comprensione delle metodologie utilizzate per realizzare il progetto. Ove con metodologie si intendono quei processi realizzativi che hanno consentito di affinare progressivamente un modulo, da uno stadio pressoché larvale, e di decorarlo con funzioni e layout per raggiungere quello che è il nostro obiettivo finale.

Parte I - Stato dell'arte

Capitolo 1

1 Piattaforma e strumenti utilizzati

Nel seguente capitolo vengono descritti gli strumenti utilizzati per realizzare il progetto. Si considera una visione globale degli strumenti che offre Google, e si spiega come sono stati utilizzati all'interno del progetto, ogni strumento viene approfondito in proporzione all'utilizzo che se n'è fatto.

Alcuni strumenti tuttavia non sono stati approfonditi o addirittura citati in quanto, è stato valutato il loro impatto poco sufficiente per una trattazione, seppur minima.

1.1 Piattaforma Google

La piattaforma di riferimento è Google, per la diffusione anche in strumenti mobile, ed essendo già la piattaforma di riferimento di Democenter è stata preferita ad altre.

Quella di Google è una piattaforma vasta, fatta di applicazioni e servizi di cui il motore di ricerca è solo la punta dell'iceberg.

Con il termine Google Apps si intende una serie di applicazioni gratuite ospitate sui server Google.

La piattaforma Google include molte applicazioni web disponibili gratuitamente agli utenti, tra la moltitudine di applicazioni quelle che hanno interessato maggiormente il progetto e che quindi meritano una particolare trattazione sono:

- Google Fogli
- Google Documenti
- Google Drive
- Google Moduli
- Google Contacts
- Google Apps Script

È possibile utilizzare Google Documenti, Fogli e Moduli in qualsiasi web browser o su qualsiasi dispositivo portatile abilitato per il web. È possibile condividere, commentare e revisionare in collaborazione in tempo reale documenti, fogli elettronici e sondaggi.

Google sostiene che più di 5 milioni di aziende stiano utilizzando gli strumenti di Google Apps, nella versione gratuita o a pagamento. La gamma di clienti spazia tra i settori a livello globale e include Uber, AllSaints, BuzzFeed, Design Within Reach, Virgin, PwC e altri.

Google assicura di non trattenere i dati dei clienti che sono memorizzati nei data center e l'accesso è limitato a dipendenti e personale selezionati senza condividere i dati con altri enti.

1.2 Google Fogli

Google fogli è probabilmente il miglior punto di partenza per approcciarsi a Google Apps Script, dotato di tutte le funzionalità base di un foglio elettronico, utilizza la classica griglia numerata sulle righe, e per le colonne utilizza le lettere per l'organizzazione dei dati al suo interno, vedere figura 1.1.

Ciò che lo contraddistingue dagli altri fogli di lavoro è il fatto di poterlo aprire in un generico browser senza bisogno di alcuno specifico programma da installare né di licenze particolari, i dati vengono elaborati e processati direttamente sui server Google e quindi non in locale sul computer dell'utilizzatore. Inoltre è presente la condivisione tra utenti diversi permettendo di lavorarci in realtime simultaneamente.

Non necessita di salvataggio, in quanto ciò avviene in automatico ogniqualvolta venga modificato una cella del foglio, inoltre ogni cambiamento viene memorizzato all'interno di uno strumento chiamato "controllo di revisione", accessibile da menu File, che permette di ripristinare qualsiasi versione precedente e di visualizzare l'utente che ha fatto tali modifiche.

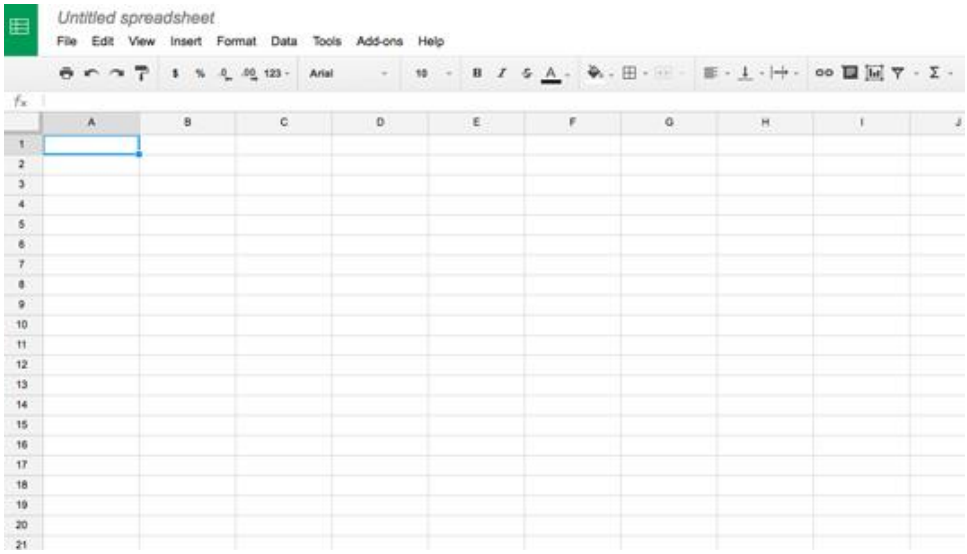

Figura 1.1- Google Fogli

1.3 Google Documenti

Google Documenti è un programma gratuito per scrivere e formattare testi, ma soprattutto un potente strumento di lavoro collaborativo. Permette di condividere un file di testo con collaboratori e di, eventualmente, chiedere suggerimenti, fare osservazioni e di correggere in tempo reale il documento. Utilizzando l'apposito bottone condividi, Google dà la possibilità di scegliere le persone da invitare e di decidere cosa tali persone potranno fare con il documento; possono modificare il documento, commentarlo, visualizzarlo oppure una combinazione delle precedenti, vedere figura 1.2.

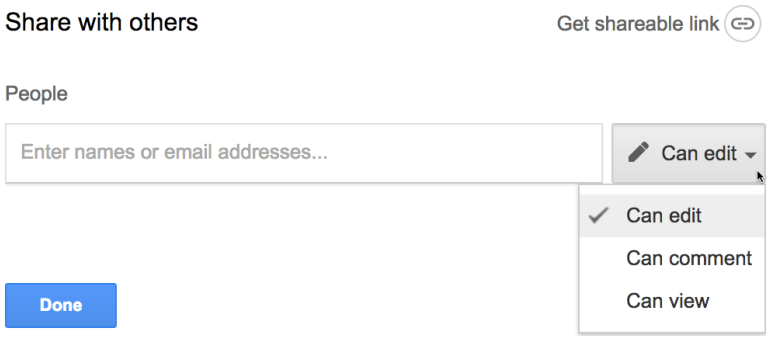

Figura 1.2 - Google Documenti, condivisione

Documenti consente di visualizzare in tempo reale gli utenti che stanno lavorando su un determinato file e aggiungere commenti. Nei commenti ci si può rivolgere a una o più persone specifiche, il commento verrà notificato agli interessati che potranno rispondere seguendo il thread dei commenti.

L'applicazione è discreta nell'invio di notifiche, avvisa l'utente solo nelle misure che egli decide: l'utente può ricevere tutte le notifiche, solo quelle indirizzate a lui oppure nessuna. È presente una chat tra utenti che condividono il documento che permette di scambiarsi opinioni, sempre in tempo reale e senza dover uscire dal programma stesso.

1.3.1 Suggerimenti di modifica documento

Con Google Documenti è possibile mettersi in modalità "Suggerimenti", cioè la possibilità di suggerire modifiche al testo senza effettivamente andare a modificarlo, il testo suggerito viene mostrato in un colore differente ed il testo da cambiare viene barrato in automatico. Il proprietario deciderà successivamente se accettare tali modifiche.

1.3.2 Altre funzionalità

Google Documenti ha anche altre funzionalità, tra cui:

- **Sommario e segnalibri**; La possibilità di creare un sommario può rendere la navigazione del testo molto più agevole. Per crearne uno, bisogna organizzare il testo con delle gerarchie di intestazioni diverse, selezionare poi la parte che interessa ed infine dal menu Formato, scegliere lo stile di cui si necessita. Documenti ha anche la funzionalità di creazione di segnalibri, basta selezionare una parola e dal menu inserisci cliccare "Segnalibro".
- **Strumento di ricerca.**
- **Cronologia delle versioni**; mentre si lavora, Google salva di continuo il file su Drive, collegato con il proprio account personale e mette a diposizione uno storico di tutti i cambiamenti, e tali cambiamenti si possono ripristinare.
- **Lavoro offline**; esiste la possibilità di lavorare offline, a patto di usare Chrome come browser e di modificare in Drive le impostazioni generali.
- **Componenti aggiuntivi**; non sono altro che funzionalità extra da aggiungere alla barra degli strumenti.

1.4 Google Drive

Drive è un servizio gratuito offerto da Google per il salvataggio dei file online e per l'accesso dei medesimi utilizzando cloud. Inoltre garantisce l'accesso a tutta una suite di applicazioni anch'esse gratuite per la creazione di documenti, fogli, presentazioni e molto altro.

Entrando nello specifico, il servizio Google Drive, di cui fa parte la suite per ufficio offerto da Google, comprendente i programmi gratuiti e basati su web di elaborazione testi, fogli elettronici e sondaggi, le applicazioni sono compatibili con i formati dei file di Microsoft Office, la suite è indissolubilmente legata a Google Drive, tutti i file creati con le applicazioni sono salvati per impostazione predefinita su Google Drive.

Per accedere all'applicativo è sufficiente un account Google ma vi si può accedere anche utilizzando una mail di un dominio differente. Lo spazio di archiviazione è di 15 GB gratuiti, oltre la quale Google offre dei piani di ampliamento spazio, fino a 30TB per gli utenti che optano per il servizio Google Apps for Work, tuttavia esiste la possibilità di ottenere uno spazio di archiviazione illimitato, a patto che gli utenti collegati a tale piano siano almeno 5, vedere figura 1.3.

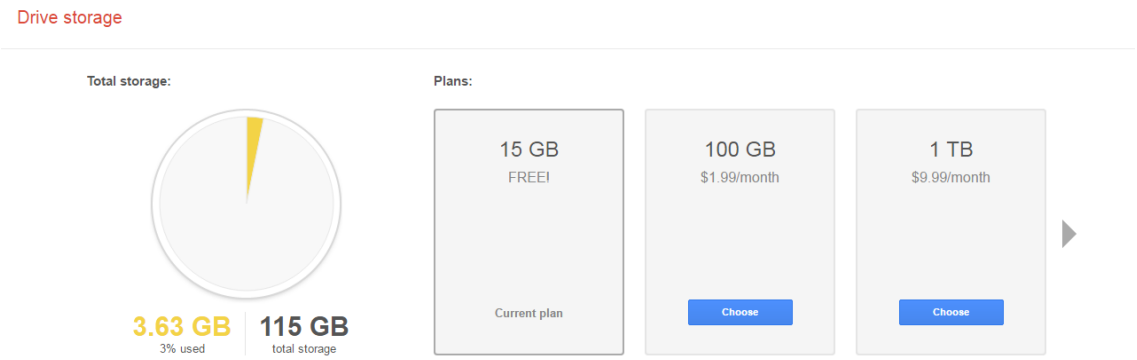

Figura 1.3 - Spazio di archiviazione Drive

Drive offre la possibilità di:

- Sincronizzare file
- Condividere dati
- Collaborazione realtime
- Modifica di file online

Google Drive è disponibile sia nella versione desktop che in quella web.
Prendendo in considerazione la versione web, la sincronizzazione dei file può avvenire in tre modi differenti;

- Selezionando e trascinando i file direttamente all'interno dell'applicazione web.
- Cliccando con il tasto destro sulla schermata principale e selezionando la voce "carica file/cartella".
- Utilizzando il bottone "Nuovo" e selezionando il caricamento del file o della cartella, vedi figura 1.4.

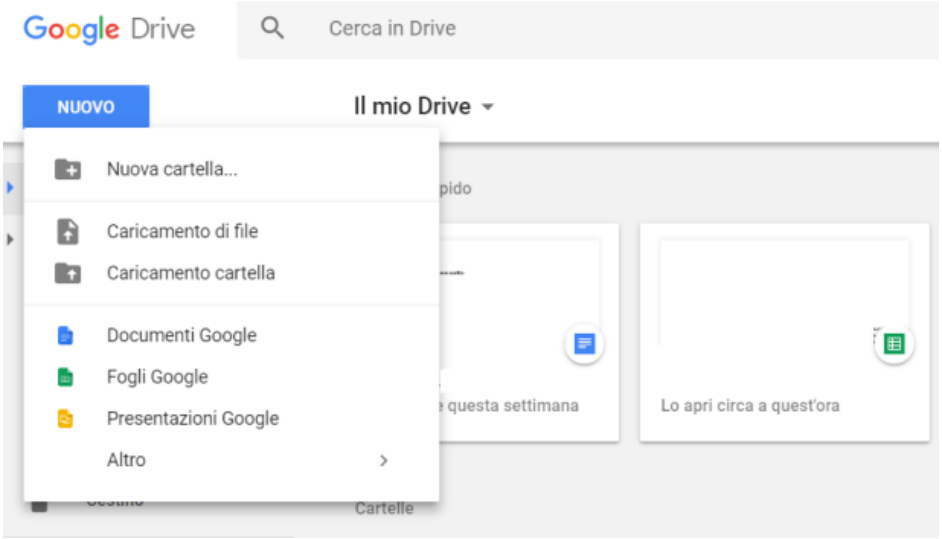

Figura 1.4 - Drive, upload file o cartella

La condivisione di dati avviene utilizzando l'apposito bottone nel menu principale, selezionando un file si avrà la possibilità di ottenere un link condivisibile oppure di inserire l'indirizzo e-mail dei collaboratori che si vogliono aggiungere.

Drive dispone della collaborazione realtime, ciò significa che è possibile far lavorare diversi utenti allo stesso file contemporaneamente, modificare tali file e vederne i risultati in tempo reale. Le opzioni di condivisione sono la modifica, il commento e la visualizzazione.

1.5 Google Moduli

Google Form è un'applicazione gratuita molto semplice da utilizzare mediante la quale è possibile realizzare questionari, quiz, sondaggi e molto altro.

Vi si può accedere tramite Google Drive oppure all'interno del proprio account Google, l'interfaccia è semplice e intuitiva da utilizzare, come in figura 1.5.

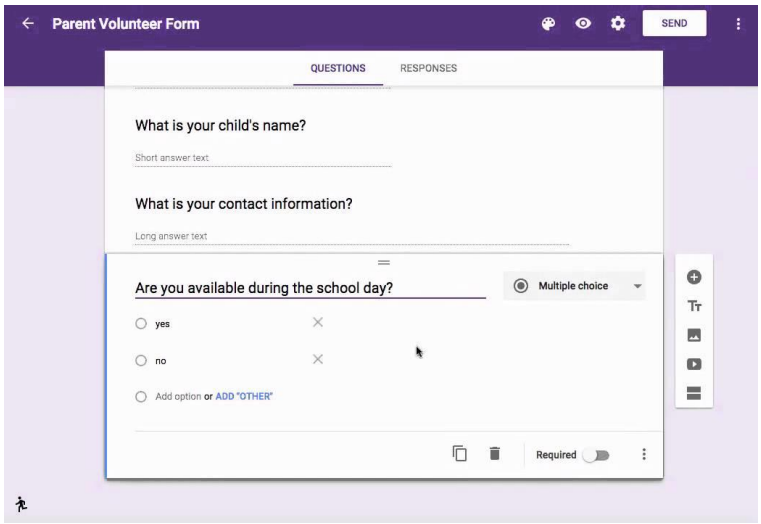

Figura 1.5 - Interfaccia Form

Nella parte alta del Google Form si possono notare tre icone. La prima fa riferimento all'aspetto del modulo. Cliccandoci sopra è possibile personalizzare il tema grafico, per esempio scegliendo il colore ed inserendo un logo.

1.5.1 Impostazioni Moduli

L'icona a forma di ingranaggio è quella delle impostazioni.
La personalizzazione attraverso l'icona dell'ingranaggio permette ad esempio di far compilare una sola volta il questionario a ciascuna persona, che dovrà essere in possesso di un account Google.

Dalle impostazioni del modulo è possibile anche ad esempio cambiare l'ordine delle domande, trasformarlo in un quiz assegnando valori in punti a ciascuna domanda e consentendo la valutazione in automatico.

1.5.2 Anteprima del modulo

L'icona a forma di occhio permette di visualizzare un'anteprima del modulo che si crea.
Qualsiasi tipo di modifica effettuata viene immediatamente salvata in automatico.

1.5.3 Altre impostazioni

L'icona che riporta tre pallini verticali sul lato destro, apre un menu contenente diverse opzioni, tra tutte troviamo anche qui come in tutti i Google Documents, la possibilità di aggiungere collaboratori.

Una volta terminato il nostro form possiamo premere il tasto invia che ci dà la possibilità di scegliere in quale modo condividerlo. Ovvero, si può inviare per e-mail, può essere incorporato (Embedded) all'interno di un sito internet, per esempio, oppure di un blog e si può ottenere un link condivisibile al form.

1.5.4 Le risposte al Form

Nella scheda risposte della schermata principale, Google offre la possibilità di avere un riepilogo dettagliato delle risposte ricevute, alcune di esse vengono anche raggruppare in grafici fornendo una rappresentazione chiara della situazione, figura 1.6.
Le risposte al form vengono riportate all'interno di un foglio di lavoro elettronico al quale si può accedere cliccando sull'icona verde.
All'interno dello spazio di lavoro Drive, nella sezione "i miei file" verrà conservato un documento con il nome del form e tra parentesi "risposte", che conterrà tutte le risposte in formato testuale.

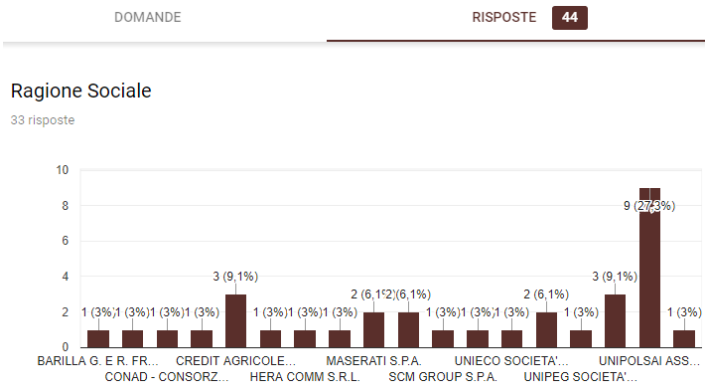

Figura 1.6 - Grafico risposte Ragioni Sociali

1.5.5 Cosa permette di fare Form?

- Creare infiniti form in maniera del tutto gratuita.
- Le risposte vengono salvate in un foglio di lavoro condivisibile.
- Segue un flusso logico, per esempio, un form che alla prima domanda chieda al possessore quale computer usi, MAC o Windows, e le domande successive siano in riferimento alla riposta data.
- Possibilità di ricevere notifiche e-mail per le risposte, vedi figura 1.7.
- I form creati sono mobile-friendly.

Google Form offre un modo veloce per la creazione di sondaggi online, le risposte vengono raccolte in un file Spreadsheet mantenuto online su Google Drive. Creare sondaggi ed invitare persone per email.

Gli invitati possono rispondere da praticamente tutti i browser incluso gli Smartphone ed i Tablet.

Ogni risposta viene visualizzata su una riga singola ed ogni domanda rappresenta una colonna

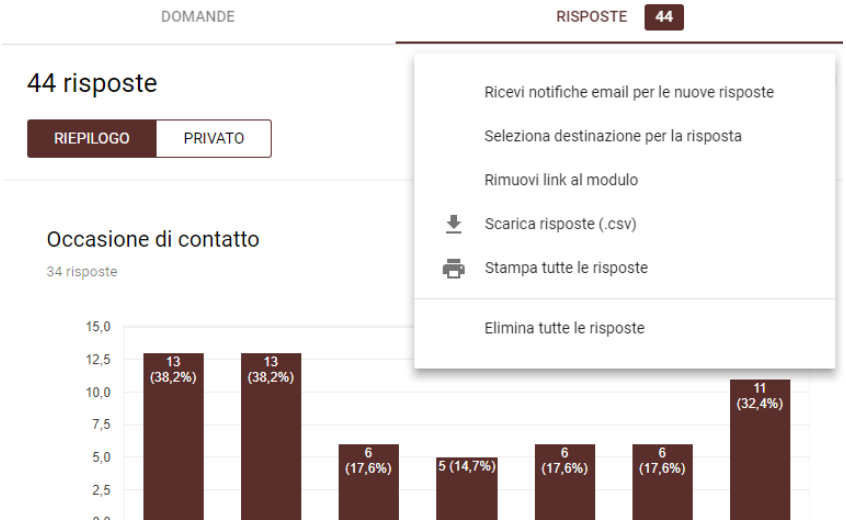

Figura 1.7 - Google Form, menu delle risposte

1.6 Google Contacts

Definire Google Gruppi come un semplice gestore di mailing list online è semplice ed anche riduttivo, Gruppi è principalmente questo ma anche altro.
È possibile scambiare e condividere informazione tra i membri di un gruppo, sia mediante l'invio di email che direttamente online dal sito di Google Gruppi.

La cosa che differenzia questo sistema da un normale servizio newsletter è che, con Gruppi ogni utente può interagire con gli altri partecipanti e quindi ciascuno può partecipare alle varie discussioni che vengono di volta in volta create. Inoltre per far parte di Gruppi è indispensabile un indirizzo email, non necessariamente gli utenti devono essere iscritti a Google.
Gruppi si può usare con qualsiasi indirizzo email, ovviamente con alcune limitazioni.

Per chi è registrato a Gmail è possibile, otre ad interagire con il gruppo mediante il sito stesso, anche utilizzare alcune funzionalità della suite Google.

Gli utenti che non sono iscritti, invece hanno la possibilità di interazione con il gruppo, limitato però, dal fatto che le comunicazioni debbano essere fatte tramite messaggi email.

Ogni account Google può gestire infiniti gruppi e partecipare ad altrettanti. Esistono una moltitudine di funzionalità per poter personalizzare Gruppi in base alle proprie esigenze.

1.6.1 La rubrica Google Contacts

Tuttavia Google Gruppi non va confuso con Google Contatti, la rubrica è personalizzabile a 360 gradi.

I campi sono illimitati e per ogni contatto si possono inserire infiniti numeri di telefono, email, indirizzi, ed altre informazioni mediante campo personalizzato. È possibile inoltre importare con semplicità i contatti da altri account e da altri programmi di posta, nonché da CSV.

Implementa l'auto-completamento mediante la ricerca all'interno delle altre applicazioni di Google, tra le tante troviamo Gmail, Google Drive, Google Calendario.

1.6.2 MyContacts e Other Contacts

Differenzia i gruppi "MyContacts" ed "Other Contacts".
Il primo è un gruppo che viene creato in maniera predefinita e che colleziona ogni nuovo contatto, comprende anche quelli appartenenti ad ogni nuovo gruppo.

Il secondo è a sua volta un gruppo che include gli account soggetti ad invio e ricezione di email. Meccanismo che va ad alimentare il funzionamento dell'auto-completamento.

È da tenere in considerazione che ogniqualvolta avvenga una sincronizzazione, tra un dispositivo mobile o in generale qualsiasi strumento, i contatti che verranno sincronizzati, sono quelli che rientrano nel gruppo "My Contacts".

Esiste la possibilità di convertire contatti dal gruppo di "MyContacts" a "Others Contacts" agevolmente.

Capitolo 2

2 Google Apps Script

Google Apps Script è un ambiente di esecuzione su server di codice in linguaggio JavaScript. Significative differenze, rispetto all'uso tradizionale di JavaScript, come linguaggio client side, che gira all'interno di un browser, sono da mettere in risalto:

- Manca la possibilità di manipolare la pagina, inteso come meccanismo che agisce lato client, dal browser.
- Nell'ambiente di sviluppo sono presenti API che permettono di interagire con gli oggetti all'interno dell'ambiente Google.

Un linguaggio che può cambiare l'approccio ai servizi di Google.
L'universo Google include anche le funzioni di scripting con le quali si può far comunicare tra loro gli applicativi, personalizzarli o automatizzarli in alcune delle loro parti.

2.1 Cos'è uno script?

Uno script è una lista di comandi che vengono eseguiti da un dato programma oppure da un motore di scripting, possono essere usati per automatizzare determinati processi sulla macchina o anche per generare pagine web.

Sono file di testo scritti in un determinato linguaggio di programmazione.
La distinzione tra un programma tradizionale ed uno script, può essere determinata dalle seguenti caratteristiche appartenenti ai linguaggi di scripting:

- Utilizzo di un linguaggio interpretato
- Mancanza di una propria interfaccia grafica
- Richiamo di altri programmi per svolgere operazioni più sofisticate
- Complessità relativamente bassa
- Linearità, uno script può anche accettare input da utente, ma solitamente input diversi non modificano sostanzialmente la struttura del DOM, diagramma a blocchi che descrive il comportamento dello script.

Disporre di un motore di scripting integrato significa poter far interagire un linguaggio di programmazione con il programma stesso e di attivarne le funzioni.

Google offre agli utenti la possibilità di creare funzioni all'interno dei documenti o delle applicazioni stand-alone, e di collegare numerose applicazioni della piattaforma e farle comunicare tra loro. Grazie al linguaggio di scripting basato su JavaScript ed all'ambiente di sviluppo semplice e minimale.

2.2 Script di Google Apps

Gli script delle Google Apps possono essere divisi in tre grandi categorie:

- **Script container bound** (embedded), inseriti all'interno di un documento creato con un'altra applicazione Google. Possono essere usati per creare pagine web autonome, cioè che vengono visualizzate senza alcuna cornice, oppure come contenuto html da inserire all'interno di una pagina realizzata con Google Sites.
- **Script stand-alone**, che possono essere eseguiti in modo autonomo.
- **Script add-on** eseguiti all'interno di applicazioni Google, come Documenti e Fogli. Quando si è in possesso di uno script bounded oppure stand-alone è possibile pubblicare il software come componente aggiuntivo in modo che sia installabile in Chrome Web Store.

2.2.1 Script container bound

Gli script container bound permettono di aggiungere funzioni a uno specifico documento di Google Docs, Sheets o Forms, per esempio, si possono programmare istruzioni da eseguire in serie all'interno di Google Spreadsheet, ottenere una lista di contatti tramite Google Contacts oppure inserire una barra di ricerca all'interno di un form.

All'interno degli scipt container bound sono disponibili funzioni per interagire direttamente con l'applicazione di cui fanno riferimento, manipolare e riutilizzare le informazioni contenute nel documento.

2.2.2 Script stand-alone

Uno script stand-alone è invece una porzione di codice eseguibile indipendente da uno specifico documento, può essere lanciato sia manualmente che in maniera automatica mediante uno scheduler, può automatizzare operazioni comportandosi quasi come un servizio eseguito in background su un sistema operativo.

2.3 Metodo doGet()

Una applicazione fatta con Googe Apps Script è un progetto contenente diversi file Google script, tra i quali file di codice JavaScript e html.
Per trasformare il progetto in un'applicazione web necessita di essere pubblicata con il tasto pubblica dal menu a tendina (pubblica -> distribuisci come applicazione web). In questo modo andremo ad associare una url allo script e a rendere tale url accessibile tramite web.
Il requisito indispensabile per lo script, è di contenere una funzione `doGet()` per quando tale script verrà richiamato via web eseguendo il codice della funzione.
La funzione deve restituire in uscita un oggetto del tipo `HtmlOutput` nel caso di una pagina web, oppure `TextOutput` per un output testuale normale (ad esempio per restituire dei dati codificati in JSON, figura 2.1).

```
1  function doGet(e) {
2    var params = JSON.stringify(e);
3    return HtmlService.createHtmlOutput(params);
4  }
```

Figura 2.1- Esempio doGet()

La funzione `doGet()` ha pieno accesso alle funzionalità di Google Apps Script, in particolare può utilizzare le API standard disponibili in ambiente server per interagire con le applicazioni Google, come Forms e Fogli.

2.4 Templates

I templates sono dei normali file html che permettono di mantenere il codice in maniera lineare e semplice, possono incorporare frammenti di codice e stampare variabili, i risultati di queste porzioni di codice vengono inclusi nello stesso, permettendo la costruzione dinamica di pagine web in base ai dati a disposizione.

In figura 2.2, un riepilogo del concetto alla base di Google Apps Script per generare e servire pagine web dal server Google.

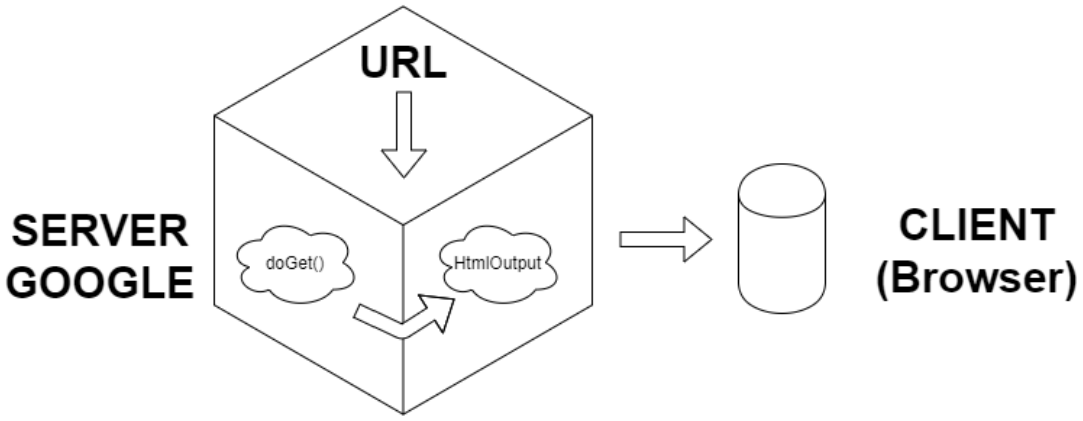

Figura 2.2 - Come Google genera e serve le pagine web

2.5 Servizio di pubblicazione Google

Il servizio di pubblicazione di Google permette di richiamare lo script dall'esterno tramite una sua url, lo stesso servizio richiama il metodo `doGet()`.
L'unico risultato ammissibile è un `HtmlOutput` che costituisce la risposta del server, l'oggetto in questione, non è un semplice testo html, ma serve come intermediario fra il codice ed il server web di Google.

La pagina web che viene servita all'utente viene composta dal server web inserendo un frammento iFrame, generato a partire dall'`HtmlOutput`.
Dal momento che la pagina è esterna si rende necessario specificare con metodi appositi, elementi come il titolo, in modo che il server possa poi leggerli ed impostarli come attributi della pagina nel punto corretto.

La soluzione più semplice è generare codice statico, ma per i nostri scopi si è reso necessario l'introduzione di contenuti dinamici utilizzando codice JavaScript, con strumenti quali jQuery ed AJAX, che facilitano in modo significativo la costruzione di pagine dinamiche.

Il codice che viene servito da Google impone però, qualche limite legato a precauzioni di sicurezza, per evitare il rischio di possibili attacchi trasversali sia sui suoi server che sulla macchina client.

Una applicazione Apps Script Google per poter funzionare come applicazione web deve avere una funzione `doGet()` oppure doPost() al suo interno. Il motivo nasce dal momento in cui si pubblica un progetto come applicazione web, il sistema associa una url all'intero progetto, tale url non prevede alcun parametro che permetta di specificare un nome di funzione quindi è stata adottata questa convenzione, cioè di eseguire una delle due funzioni.

Ricalcando i passi presentati fino ad adesso abbiamo detto che:

- Una volta ultimato un progetto e pubblicato come applicazione web ad esso viene associato una url.
- Chiamando tale url da un browser verrà eseguita una delle due funzioni, `doGet()` oppure `doPost()` presenti nel codice.
- La funzione dovrà restituire un oggetto del tipo `HtmlOutput` oppure TextOutput.
- Il contenuto dell'output verrà restituito al browser o all'applicazione che ha chiamato il metodo.
- Se il contenuto restituito è del tipo html allora l'ambiente JavaScript nel browser, ovvero client-side, manterrà un collegamento con il progetto che lo ha generato.
- Il collegamento si utilizza con la funzione Google.script.run, che permette di eseguire in modo asincrono una funzione del progetto che ha generato la pagina. Questa funzionalità è fondamentale dal momento che permette alla pagina web di interagire con il server. La pagina non è quindi un contenuto a senso unico, cioè non viene generato una tantum per poi stazionare dove si trova, ma è possibile realizzare applicazioni interattive, anche complesse che continuato ad inviare e ricevere dal Google server.

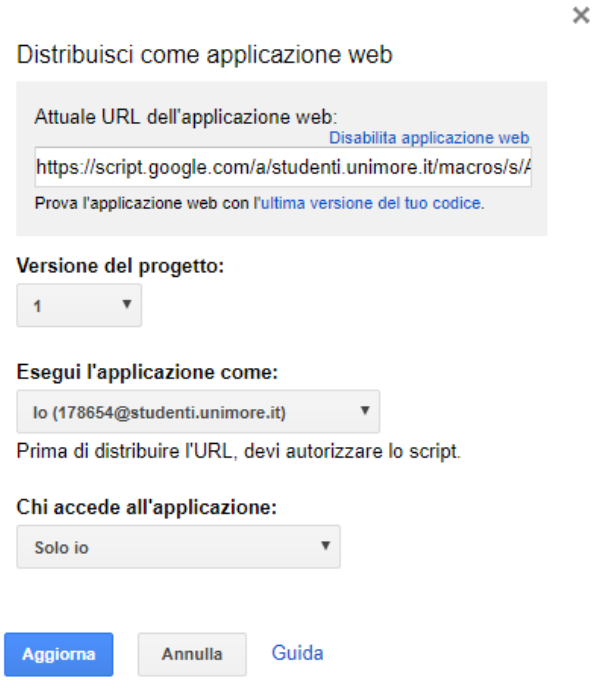

Figura 2.3 - Pubblicazione progetto Apps Script

2.5.1 Che cosa deve fare doGet?

La funzione `doGet()` deve restituire in uscita un oggetto appartenente alla classe `HtmlOutput`, per fare ciò si può:

- Ottenere direttamente tale oggetto da un contenuto html presente in un file del progetto, oppure generato dal codice.
- Ottenere l'oggetto a partire da un template html sulla base dei dati passati all'applicazione. Questo metodo è molto più potente, in quanto ci permette di costruire dinamicamente del codice e soprattutto di passare dei dati alla pagina web.

2.6 Interfaccia di sviluppo

Per quanto riguarda l'interfaccia di sviluppo, quando si creano script realizzati come applicazione web, per esempio, all'interno del progetto devono essere inseriti anche i file Html che andranno a costruire l'interfaccia da mostrare all'interno del browser.

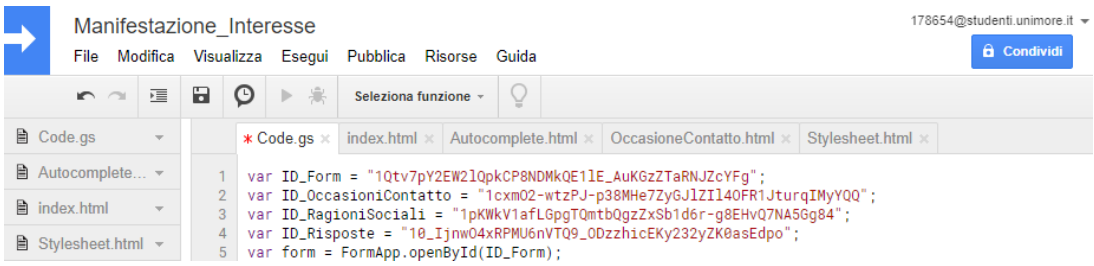

Figura 2.4 - Barra degli strumenti Apps Script

La barra degli strumenti, appena sotto il menu, contiene i pochi pulsanti con cui attivare le funzionalità principali.

Nell’ordine, guardando la figura 2.4:

- I pulsanti per annullare e rispristinare le ultime modifiche.
- Attivazione dell’indentazione automatica del codice.
- Il comando di salvataggio.
- Il comando per l'esecuzione di una funzione selezionata.
- Impostazione dei trigger, ovvero delle condizioni, eventi o timer, che causano l’esecuzione dello script.
- Avvio dell’esecuzione rispettivamente in modalità normale o debug.
- Menu a tendina contenente l’elenco delle funzioni inserite nello script.

2.7 Esecuzione in modalità normale e debug

La differenza tra le due modalità di esecuzione, normale oppure debug, è molto importante.
In modalità debug in sede di sviluppo, quando l’esecuzione del programma si interrompe, per esempio per un errore rilevato, l’ambiente di sviluppo non si limita a segnalare tale errore ma consente al programmatore di verificare lo stato del sistema e di eventualmente correggere l’errore.

Si possono impostare punti di interruzione, cioè punti su cui l’esecuzione del programma dovrà fermarsi forzatamente, vedere figura 2.5.
Quando il programma è in pausa, nei riquadri della zona inferiore della finestra vengono visualizzate tutte le variabili del programma, le loro definizioni e ciò che contengono.
Durante la pausa del programma si può riprendere l’esecuzione o interromperla definitivamente, si può inoltre effettuare il debug passo per passo, eseguire righe di codice una alla volta per verificare il flusso di controllo.

In modalità stand-alone l’ambiente di sviluppo si presenta in modo identico, ma deve essere lanciato direttamente dall’interfaccia di Google Scripts o da Google Drive.

Code.gs

```
1  function emailDataRow(row, email) {
2    Logger.log('Emailing data row ' + row + ' to ' + email);
3    var sheet = SpreadsheetApp.getActiveSheet();
4    var data = sheet.getDataRange().getValues();
5    var rowData = data[row-1].join(" ");
6    Logger.log('Row ' + row + ' data: ' + rowData);
7    MailApp.sendEmail(email,
8                      'Data in row ' + row,
9                      rowData);
10 }
```

⊞ this	Object (851436070)	({myFunction:function myFunction(
row	Number	2.0
email	String	"john@example.com"
⊞ arguments	Arguments (851436108)	
sheet	Object (851436133)	
⊟ data[4]	Array (851437500)	[["Model Number", "10231-RC02"],
⊞ 0[2]	Array (851437501)	["Model Number", "10231-RC02"]
⊞ 1[2]	Array (851437502)	["Cost", 103.24]
⊞ 2[2]	Array (851437503)	["Quantity", 341]
⊞ 3[2]	Array (851437504)	["Total", 35204.84]
rowData	undefined	undefined

Figura 2.5 - Debugger

2.8 Le basi del linguaggio di scripting

I Google Apps Script, aggiungono alla sintassi ed ai costrutti di JavaScript strumenti del Google Web Toolkit, per realizzare gli elementi dell'interfaccia utente, ed una serie di funzioni di libreria necessarie per comunicare con gli strumenti e le applicazioni di Google.

Gli script e le funzioni si possono eseguire nell'ambiente di sviluppo, selezionando il nome della funzione nel menu a tendina dalla barra degli strumenti e premendo il pulsante avvio. Esistono alternative a questo metodo, per esempio aggiungendo una voce di menu a un documento o a un'applicazione, oppure automatizzare il lancio di una funzione tramite un trigger.

2.9 Trigger

Utilizzare un trigger significa stabilire una condizione che faccia scattare l'esecuzione della funzione selezionata. Gli script possono essere eseguiti a intervalli regolari.

Per gli script legati a un documento, la medesima finestra offre un'alternativa ai trigger basati sul tempo, cioè legati ad eventi specifici. Si può quindi, eseguire una funzione all'apertura del file o nel momento in cui il documento viene modificato.

2.10 Script e sicurezza

La prima volta che si lancia uno script, Google chiede all'utente diverse conferme attraverso finestre di autorizzazioni, ciò è conseguenza del fatto che gli script delle Google Apps hanno potenzialmente accesso a tutti i dati che l'utente memorizza nel proprio account.

Il sistema di autorizzazione di Google prima di eseguire uno script elenca i servizi che gli saranno necessari, per informare l'utente di quali siano le applicazioni e i dati a cui si sta per accedere, in lettura e scrittura.

Un esempio viene mostrato nella figura 2.6.

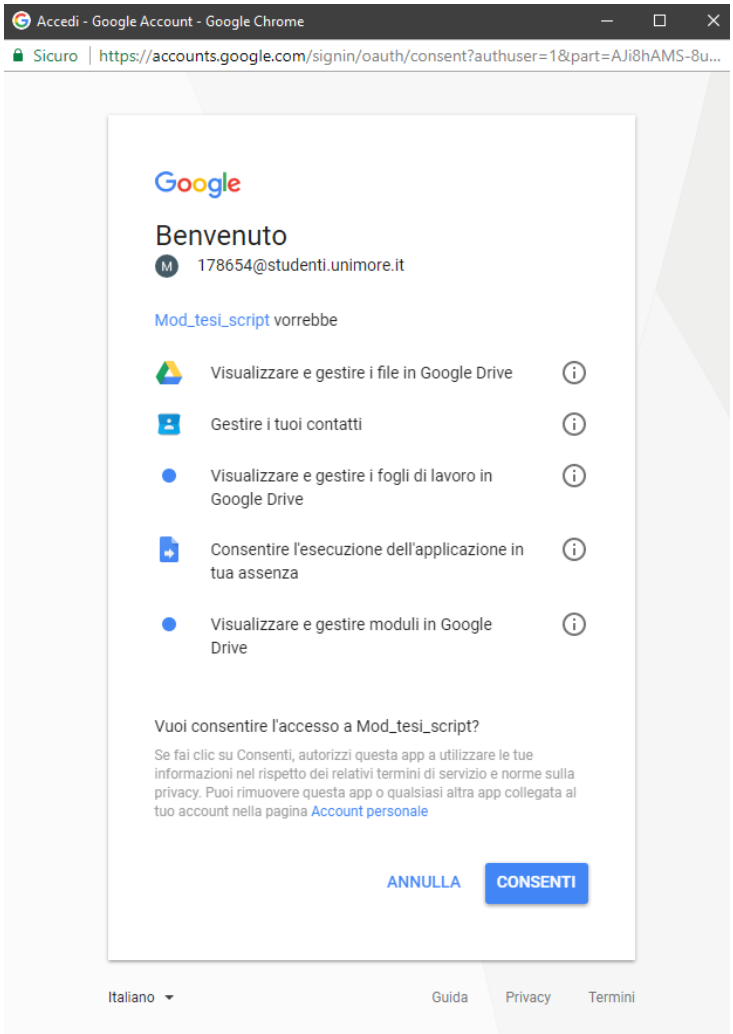

Figura 2.6 - Autorizzazioni Google Apps Script

2.11 Auto-completamento di Apps Script

Una funzionalità molto importante di cui dispone Google Apps Script è l'auto-completamento.

I vantaggi dell'auto-completamento, che può essere attivato dal menu "Modifica" dell'editor oppure con il shortcut Ctrl + Spazio, sono molteplici, tra cui individuare errori del tipo "typo" o errori di sintassi, in quanto suggerisce ogni possibile metodo disponibile quando si inizia a digitare una determinata porzione di codice, vedere figura 2.7.

Per esempio digitando una lettera sull'editor e premendo la coppia di tasti, Ctrl + Spazio, viene mostrato un riquadro contenente tutti i metodi disponibili con tale lettera e a lettere successive vengono mostrate metodi corrispondenti al match di tali sottostringhe.

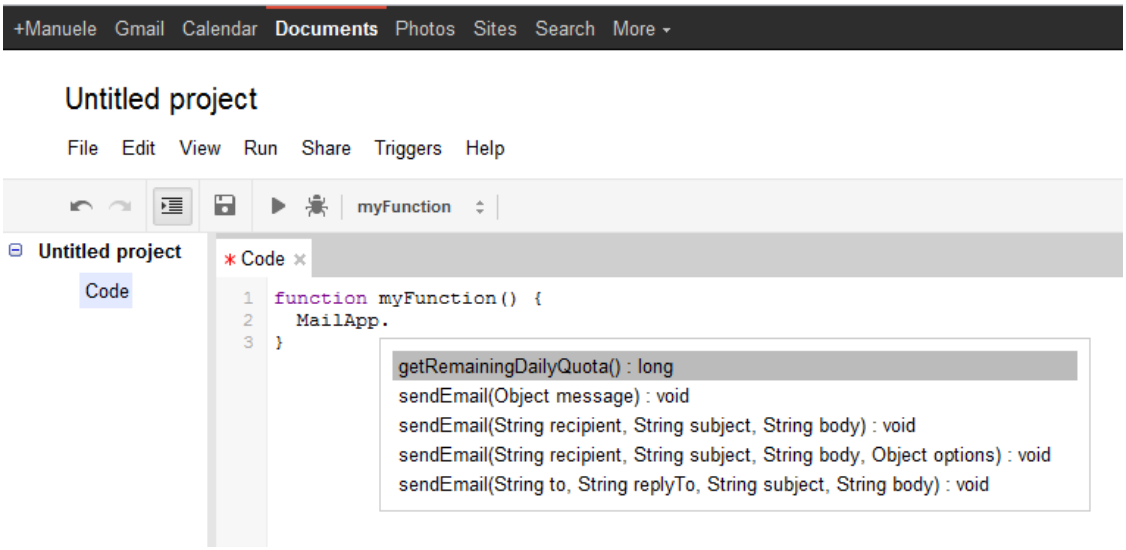

Figura 2.7 - Esempio di auto-completamento per metodi, Apps Script

2.12 Collaborazione su un file

È possibile la collaborazione al file tra diversi utenti in tempo reale, non c'è bisogno di salvare il documento perché, per esempio in un foglio di lavoro Spreadsheet, ogni cella che viene modificata provoca il salvataggio, ed i cambiamenti sono registrati in "cronologia versioni" accessibile dal menu File, permette di ripristinare le versioni precedenti e di sapere quando un utente ha fatto una determinata modifica.

2.13 Documentazione

La documentazione su Google Apps Script è disponibile alla pagina "Google Devolopers", sono disponibili una serie di guide, tutorial e pagine di reference con link a risorse utili, come il forum Stack Overflow con tag "google-apps-script".

Nella pagina principale del sito per sviluppatori di Google troviamo una barra di ricerca che elencherà i risultati limitati a Google Apps Script, i risultati sono composti da pagine di documentazione, ma anche esempi di codice e molto altro.

2.14 Librerie

In fine si possono trovare le librerie nel menu Risorse, una libreria altro non è che una collezione di metodi che estendono il funzionamento della piattaforma e ne colmano le eventuali lacune.
Google mette a disposizione di chiunque la possibilità di creare la propria libreria in Google Apps Script.

Parte II - L’applicazione

Capitolo 3

3 Progettazione

I requisiti descrivono ciò che l'utente si aspetta dal sistema, e specificarli significa esprimerli in modo chiaro, univoco, consistente e completo.
SRS, acronimo di *"Software Requirement Specification"*, riporta in modo corretto e non ambiguo le specifiche dei requisiti che dovrà offrire il software, non descrive alcun dettaglio progettuale o implementativo e non impone vincoli addizionali al prodotto, quali ad esempio, qualità, oggetto di documenti specifici.

I requisiti si distinguono in:

- Funzionali: descrivono cosa deve fare il sistema, generalmente in termini di relazioni fra dati di ingresso e dati di uscita, oppure fra segnali, dall'ambiente al sistema e risposte del sistema. Questi vincoli sono generalmente esprimibili in modo formale
- Non funzionali: esprimono dei vincoli o delle caratteristiche di qualità.

3.1 Requisiti funzionali

A seguire una descrizione del comportamento del software che sarà sviluppato, i casi trattati contengono un diverso livello di astrazione a seconda della necessità ed è stato prioritario focalizzarsi su requisiti che non fossero definiti in maniera troppo grossolana ed ambigua.

3.2 Creazione occasione di contatto

Un'occasione di contatto è definita come una tupla all'interno del foglio di lavoro di Google Sheets.

L'amministratore dei contatti, che ha le autorizzazioni per accedere al file, può aggiungere un campo in fondo all'anagrafica, inserendo il nome di una nuova occasione di contatto, Fogli provvederà a salvare il documento, e a notificare all'amministratore in questione che il salvataggio su Drive è avvenuto con successo.

RF01	**Anagrafica occasioni di contatto**	**Creazione nuova occasione di contatto**
Input	Nome dell'occasione di contatto da creare	
Descrizione	*Sequenza di operazioni* L'amministratore dei contatti accede all'anagrafica delle occasioni di contatto e vi inserisce il nuovo campo, salva e controlla che l'inserimento sia avvenuto correttamente.	
Output	Conferma dell'avvenuta creazione di una nuova occasione di contatto	

Tabella I(Requisito funzionale 1)

3.2.1 Aggiunta di utente a gruppo Google Contacts

Ricerca dell'identificativo dell'ultimo contatto che ha risposto al form e creazione di gruppi che sono stati flaggati durante la compilazione dello stesso form.
Un controllo, verifica che in nessuno dei gruppi preesistenti sia già presente l'identificativo dell'utente e che non si stia creando il duplicato di un gruppo.

Qualora il controllo avesse buon esito, il sistema aggiunge l'utente a tutti i gruppi selezionati di cui non facesse già parte.
Si valuta, eventualmente, di automatizzare il processo, attivandolo ogniqualvolta un form venga sottomesso al server.

RF02	**Anagrafica contatti**	**Aggiunta utente a gruppo**
Input	Identificatori gruppo e utente	
Descrizione	*Sequenza di operazioni* L'amministratore dei contatti seleziona l'utente che dovrà essere aggiunto ad un determinato gruppo, il sistema controlla che nel gruppo in questione non sia già presente l'utente che si vuole aggiungere. Messaggio di errore nel caso che l'utente sia già presente, oppure che il gruppo non esista.	
Output	Conferma avvenuta aggiunta	

Tabella II(Requisito funzionale 2)

3.2.2 Documenti allegati

Una funzione di ricerca file permette di navigare le cartelle di Google Drive, selezionare un file ed automaticamente compilare il form con il link al file selezionato.
Una volta archiviato come informazione della istanza del contatto, tale link permetterà di visualizzare e scaricare il file a tutti gli utenti purché abbiano i diritti di accesso.

RF03	**Documenti allegati**	**Documenti allegati**
Input	File da caricare	
Descrizione	*Sequenza di operazioni* L'amministratore seleziona un file da caricare, il sistema invierà una risposta per l'avvenuto caricamento, altrimenti sarà notificato un messaggio d'errore.	
Output	Conferma dell'avvenuto caricamento del file	

Tabella III(Requisito funzionale 3)

3.2.3 Creazione nuovo form

La classe FormApp permette di aprire o creare un nuovo form. In fase di apertura è necessario un identificativo per il form, se non corretto l'apertura verrà annullata e notificato un errore da parte del sistema.

RF04	**Form**	**Apertura form**
Input	Identificativo del form	
Descrizione	*Sequenza di operazioni* Il sistema, utilizzando l'identificativo fornito dall'amministratore, apre il form. Nel caso che il form non esista oppure sia sbagliato l'identificatore, un messaggio di errore verrà inviato. L'operazione è permessa solo agli utenti autorizzati.	
Output	Conferma corretta apertura form.	

Tabella IV(Requisito funzionale 4)

3.2.4 Accesso anagrafica ragioni sociali

L'amministratore contatti ha accesso alla anagrafica delle "occasioni di contatto" in visualizzazione e inserimento, ma anche modifica e cancellazione. Per accedere, quando autorizzato, necessita dell'apertura del file "export_ragioni_sociali_account_studenti", salvato su Drive.

RF05	**Anagrafica ragioni sociali**	**Accesso ragioni sociali**
Input	Identificativo anagrafica ragioni sociali	
Descrizione	*Sequenza di operazioni* L'amministratore che accede all'anagrafica, ha il privilegio di visualizzazione, inserimento, modifica e cancellazione. Qualora non fossero presenti le autorizzazioni il sistema provvederà a notificarlo.	
Output	Conferma delle operazioni svolte	

Tabella V(Requisito funzionale 5)

3.2.5 Ricerca ragione sociale

All'interno del form è stato inserito un campo ragione sociale. Per permettere agli utenti che compilano il form di indicare una ragione sociale, è stato messo a punto una modalità di selezione in Google Form mediante jQuery. Il meccanismo di auto-completamento viene implementato sul documento contenente l'anagrafica delle ragioni sociali, permette di digitare alcune lettere e fornisce all'interno di un menu a tendina i risultati dei riscontri.

RF06	**Anagrafica ragioni sociali**	**Ricerca ragione sociale**
Input	Identificatore ragione sociale	
Descrizione	*Sequenza di operazioni* Viene chiesto all'utente di digitare almeno le prime lettere della propria ragione sociale, il sistema analizzerà l'anagrafica delle ragioni sociali per poi sottomettere il risultato all'utente.	
Output	Il campo della ragione sociale	

Tabella VI(Requisito funzionale 6)

3.3 Scenario: Inserimento nuovo contatto

Il Form, per la manifestazione di interesse ad un determinato evento, viene sottomesso ai collaboratori, ciascuno di loro dovrà compilarlo secondo le sue esigenze.

L'inserimento dei dati anagrafici, della ragione sociale, di un indirizzo email valido, di almeno un recapito telefonico, dell'indirizzo dell'ufficio, del sito web, del ruolo o posizione occupata e delle occasioni di contatto di interesse sono necessari prima dell'invio.

Per ogni utente che compila il Form e lo invia, il sistema attiverà il trigger.
Il trigger elaborerà le informazioni contenute nel Form e per ogni risposta creerà un nuovo contatto basandosi sui dati in input.

Gli attori presi in considerazione sono:

- Partner di progetto di Democenter.
- Il sistema, volto a rappresentare il programma.

Lo scenario preso in considerazione, dove al partner viene proposto un form da compilare, per manifestare il proprio interesse ad un determinato evento, vede l'attore, che fa le veci di un partner di progetto, compilare il modulo, almeno nelle sue parti essenziali.

Possono essere omessi alcuni campi, ad esempio, uno dei campi adibiti per i recapiti telefonici, infatti il sistema, quanto gli si presenta un campo vuoto non segnala alcun errore. Il campo vuoto viene gestito da Contacts lasciandolo inalterato, tuttavia è consentita una successiva modifica del contatto da parte di un amministratore.

La creazione effettiva del contatto avverrà alla pressione del bottone per inviare il form compilato. Il trigger, verrà poi avviato dall'attore Sistema e saranno processati tutti i campi considerati.
Nella figura 3.1 la rappresentazione del diagramma dei casi d'uso inerente allo scenario.

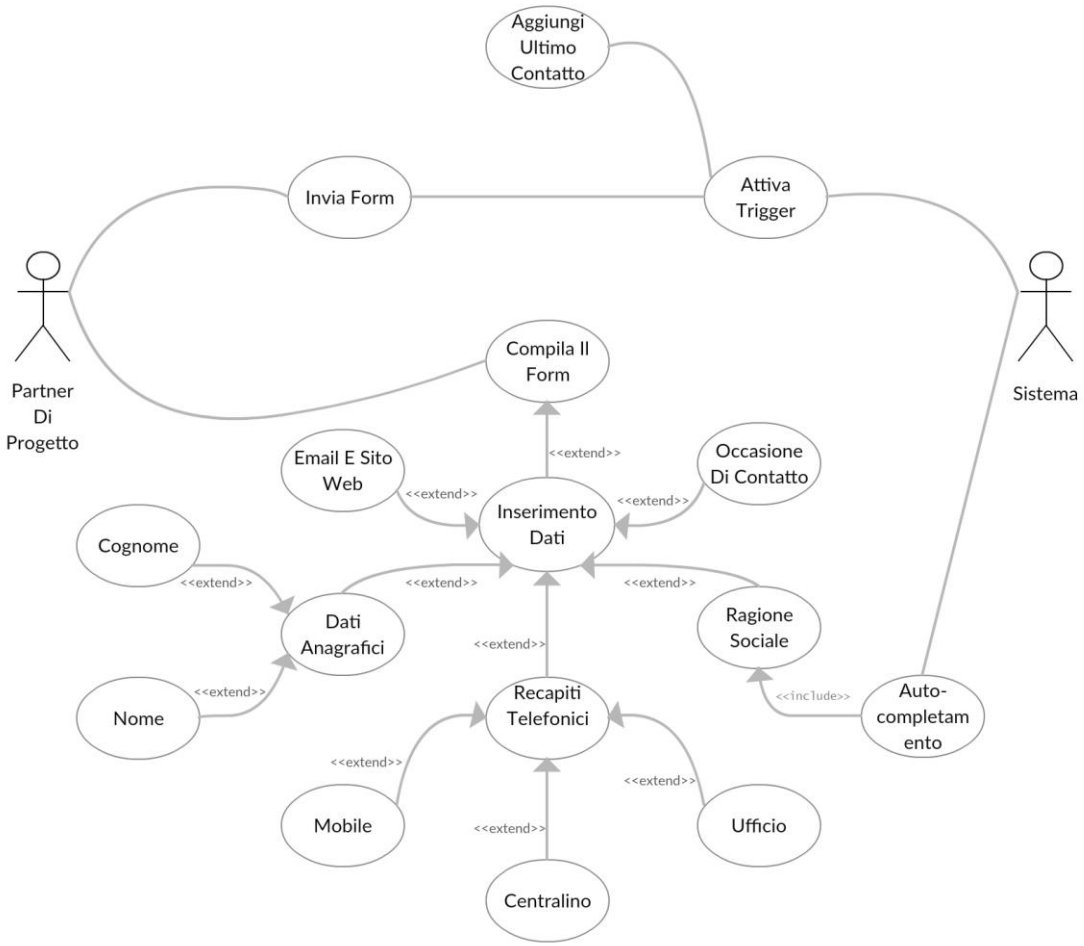

Figura 3.1 - Diagramma dei casi d'uso, inserimento nuovo contatto

3.4 Activity Diagram

L'inserimento di nuovi contatti può avvenire in due modi differenti.

- Per i partner di progetto: viene presentato un form con selezione delle ragioni sociali, da apposita anagrafica, e con un campo che definisce le modalità di contatto, come potrebbe essere uno degli eventi, una visita in azienda o a uno dei laboratori, un contatto telefonico, un incontro ed altro ancora.
- Per gli utenti generici: un form per ciascuna occasione di contatto, tipicamente uno per il sito web e gli altri personalizzati per ciascun evento. Realizzabile tramite link precompilato.

Di seguito il diagramma delle attività che riguarda l'inserimento di un nuovo contatto.

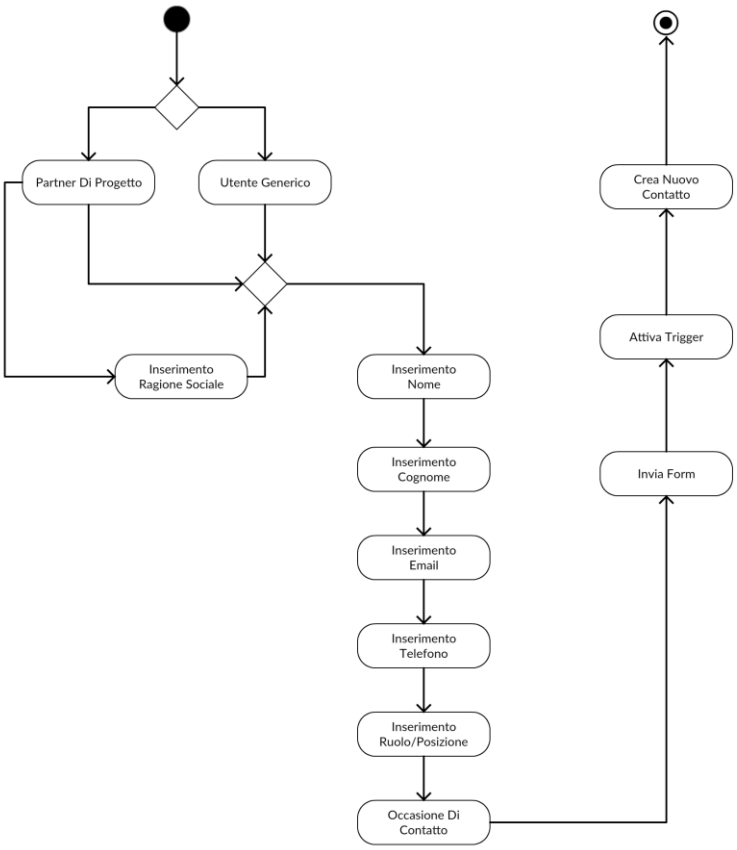

Figura 3.2 - Activity Diagram Inserimento Nuovo Contatto

Capitolo 4

4 Implementazione

Questo capitolo tratta dell'implementazione del progetto, a partire dalla fase preliminare, dove si crea un semplice form contenente i campi che saranno necessari successivamente. Vengono descritte le metodologie e come sono stati utilizzati gli strumenti già ampiamente presentati.

Verrà spiegato come è stato modificato il form iniziale, creato con Google Form, come sono stati ottenuti particolari campi e sarà approfondito l'uso fatto di Chrome DevTools, che è stato fondamentale per avere una corrispondenza tra un semplice form ed il codice implementato utilizzando Apps Script.

Continuando la trattazione, ci soffermeremo su due campi importanti; occasioni di contatto e auto-completamento.
I campi sopracitati spiccano per importanza, non solo perché fanno parte dei requisiti funzionali già discussi, ma anche per come sono stati messi in pratica.

Approfondiremo la funzione `doGet()` nel suo aspetto più pratico, spiegando a cosa è servito effettivamente. Saranno trattati le altre funzioni ed i templates che sono stati usati.

Nel finale affronteremo ciò che riguarda puramente l'aspetto estetico del form e di come si è arrivato da una fase minimale a qualcosa di leggermente più complesso, come si presenta il nostro modulo all'epilogo.

4.1 Creazione di form tramite Google Moduli

In questa fase preliminare, è indispensabile accedere con il proprio account all'indirizzo email "docs.google.com/forms".

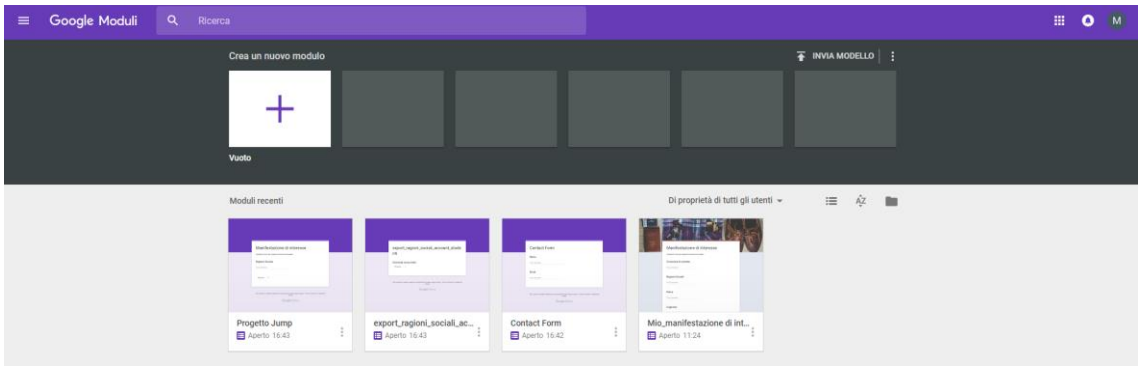

Figura 4.1 - Home Form

Nella schermata che si presenta, come in figura 4.1, scegliere l'icona di creazione di un nuovo modulo. La schermata successiva è la classica presentazione dei moduli, con campi di default e mancanza di un titolo, vedere figura 4.2.

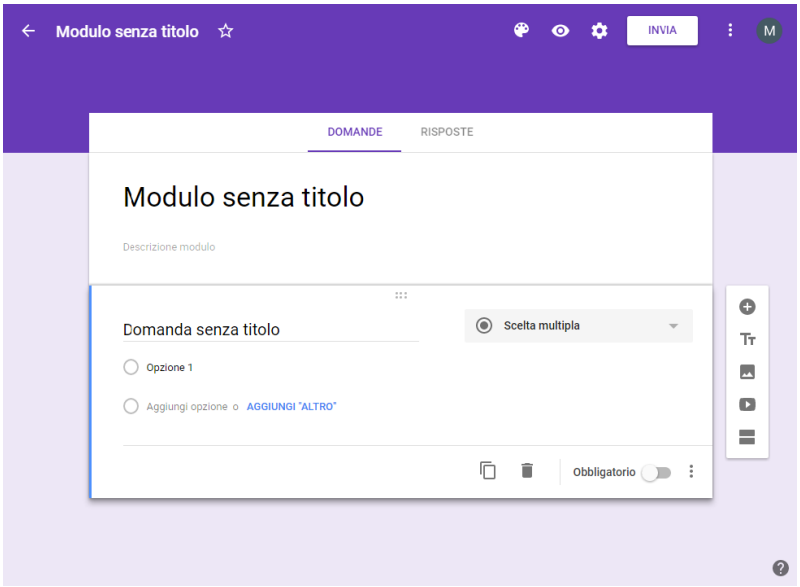

Figura 4.2 - Modulo vuoto

Il form è situato al centro della pagina, con il primo campo riservato per il titolo e per una eventuale descrizione, seguito da un campo generico.

Selezionando il menu a tendina, con scritto "scelta multipla", vengono mostrati numerosi campi tra cui scegliere.
Moduli mette a disposizione 14 campi diversi, tra cui il campo per allegare immagini e video.
Per allegare un'immagine è possibile caricarla da locale, scattare una foto, tramite URL oppure selezionando direttamente dalla cartella Drive l'elemento desiderato.
Per quel che riguarda il caricamento di video, una barra di ricerca permette di aggiungere un video direttamente da YouTube.

Ogni campo possiede un bottone per la duplicazione della stessa, serve a semplificare il lavoro qualora dovessero essere presenti domande molto simili tra loro, e un bottone per eliminare, eventualmente alcuni oggetti.
I campi che sono stati inseriti sono:

- Risposta breve; nome, cognome, ragione sociale, email, telefono mobile telefono ufficio, telefono centralino, indirizzo ufficio, sito web, ruolo/posizione.
- Caselle di controllo; occasione di contatto.

Il modulo attualmente si presenta come in figura 4.3. Mancano da implementare le funzionalità dei campi "Occasione Di Contatto" ed il meccanismo di auto-completamento inerente a "Ragione Sociale".

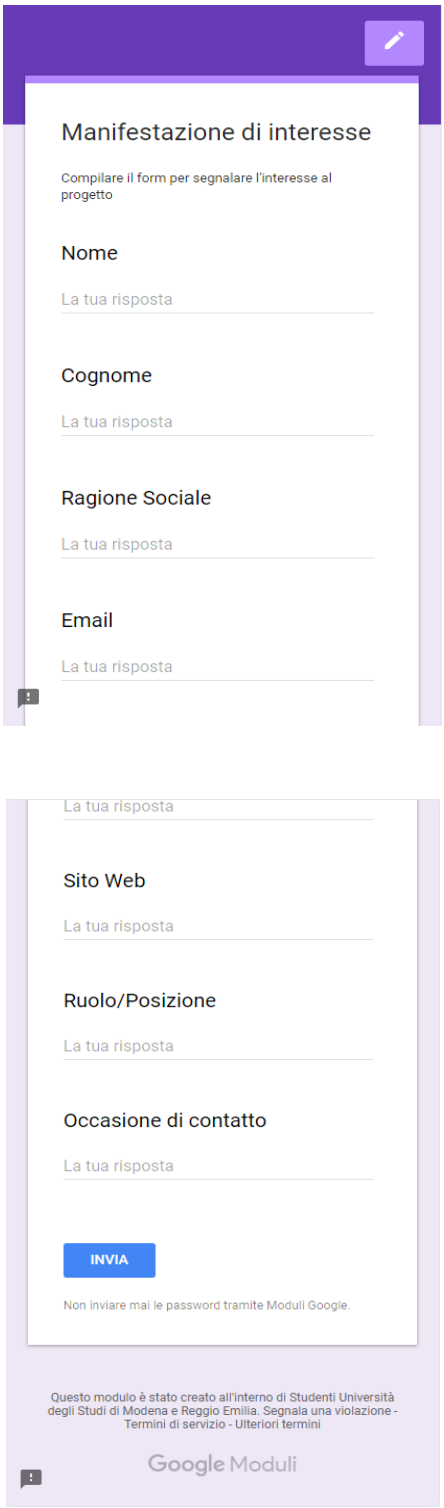

Figura 4.3 - Form preliminare

4.2 Inspector Google

I Google Dev tool sono, in pratica, uno strumento all-in-one dove è possibile testare progetti e capire perché un determinato layout non si stia comportando a dovere. Consentono di effettuare il debug avanzato di applicazioni front-end in maniera semplice e veloce.

4.2.1 Come accedervi

- Dal menu di Chrome, selezionando altri strumenti -> strumenti per sviluppatori.
- Tasto destro mouse su un oggetto di una qualsiasi pagina web e selezionando, Ispeziona.
- Utilizzando le scorciatoie, Ctrl + Shift + I oppure F12 direttamente da Chrome.

4.2.2 Come si presenta la finestra DevTools

DevTools è organizzato in pannelli, raggruppati nella barra superiore della finestra, come in figura 4.4.

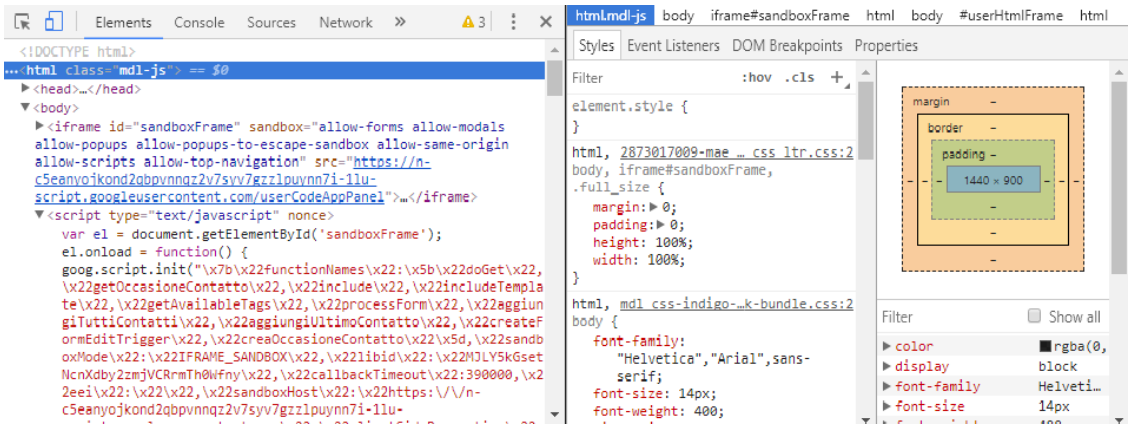

Figura 4.4 - Finestra DevTool

Nel complesso sono presenti 8 pannelli:

- **Elements**.
- **Resources**: è possibile esaminare i dati contenuti nei vari file, come i dati contenuti in sessione e nei cookie.
- **Network**: è possibile esaminare le richieste inviate al server, nelle URL individuate troveremo tutte le informazioni, come i parametri passati nell'Header.
- **Sources**.
- **Timeline**: è possibile esaminare cosa succede quando si carica la pagina ed esaminare il tempo di caricamento degli elementi.
- **Profiles**.
- **Audits**.
- **Console**: è possibile visualizzare i vari messaggi della console, come eventuali errori.

4.2.3 Il pannello Elements

Muovendo il cursore tra gli elementi HTML, questi ultimi verranno evidenziati con un rettangolo blu contenente le misure dell'elemento stesso. L'elemento selezionato è così disponibile per varie modifiche.

- Ispeziona e modifica la struttura DOM di qualsiasi elemento.
- Mostra e cambia le regole CSS degli elementi.
- Visualizza ogni modifica effettuata in locale sul sorgente.

4.2.4 Personalizzazione Modulo Google

In questa fase del progetto andremo ad utilizzare i Chrome Developer Tools per creare un collegamento tra i campi del form originale di Google ed il codice Html.

Aprendo il form creato precedentemente in modalità visualizzazione, e quindi utilizzando il bottone a forma di occhio, che si presenta nella barra degli strumenti, in alto a destra della finestra di Moduli.
Attivando DevTools, utilizzo l'opzione di selezione di un elemento, per selezionare uno dei campi. Il primo campo selezionato, per comparsa, sarà Nome.

Andremo a cercare nel codice Html, un campo nominato "*entry*", seguito da una serie di numeri, come in figura 4.5.

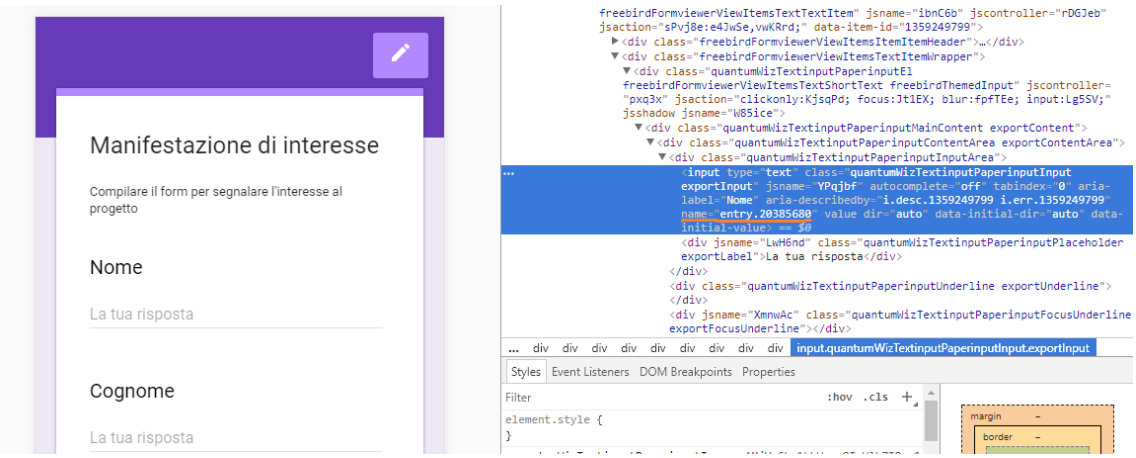

Figura 4.5 - DevTools, campo Entry

Il campo *entry* appena identificato andrà sostituito a tutte le occorrenze dei campi del form all'interno del sorgente Html, all'interno di Apps Script. Questo permetterà di avere una corrispondenza fra i dati inseriti da un utente, e ciò che verrà poi elaborato all'interno del codice.

Nella figura 4.6, vengono ricapitolati i campi standard appena modificati inserendo i campi *entry*.

```html
<form class="form" action="">

      <label>Nome</label>
      <input name="entry.2061900206" type="text" />

      <label>Cognome</label>
      <input name="entry.1997567091" type="text" />

      <label>Ragione Sociale</label>
      <input name="entry.1450737721" type="text" />

      <label>Email</label>
      <input name="entry.765763405" type="text" />

      <label>Telefono Mobile</label>
      <input name="entry.148792327" type="text" />

      <label>Telefono Centralino</label>
      <input name="entry.1310018025" type="text" />

      <label>Telefono Ufficio</label>
      <input name="entry.614283919" type="text" />

      <label>Indirizzo Ufficio</label>
      <input name="entry.48711455" type="text" />

      <label>Sito Web</label>
      <input name="entry.1580136677" type="text" />

      <label>Ruolo/Posizione</label>
      <input name="entry.962344154" type="text" />

      <label>Occasione Di Contatto</label>
      <input name="entry.793889370" type="text" />

</form>
```

Figura 4.6 - Html con inserimento Entry

4.2.5 Attributo Action

Per completare la modifica manca solo da inserire l'attributo "*action*" del form. Sempre tenendo presente l'Inspector, andremo a cercare il tag "form". Dentro tale tag copieremo il campo "action", contenente un link al form stesso, che poi incolleremo nel campo action del file sorgente, come viene mostrato nella figura 4.7.

```
<form class="form" action="https://docs.google.com/a/studenti.unimore.it/forms/d/e/1FAIpQLSc_wbJEWmk1B1in_d5HuirkTPoe
```

Figura 4.7 - Action, Link al form

4.3 Occasioni di contatto

Attualmente il campo occasioni di contatto è una semplice casella di testo, tuttavia, il comportamento desiderato è di gestirla in maniera differente.
Vogliamo che ogni utente che si trovi a dover compilare il modulo possa scegliere tra una serie di occasioni di contatto, cliccando sopra una casella di controllo per ognuna delle opzioni desiderate.

Le scelte dovranno essere mantenute da un amministratore mediante l'utilizzo di un foglio di lavoro Google, adeguatamente aggiornato. Creiamo tale foglio di lavoro e lo chiamiamo *Occasioni Di Contatto*.
Adesso possiamo procedere nel seguente modo; ci spostiamo nell'ambiente di lavoro, Apps Script, e otteniamo l'identificativo del foglio di lavoro, iteriamo mediante un ciclo sui dati ottenuti dal foglio e per ogni tupla andremo ad aggiungere un'opzione per il campo "Occasioni di contatto", rappresentato da una nuova checkbox all'interno del form.

4.3.1 Codice sorgente getOccasioneContatto()

```
1   /*
2    * Funzione che apre uno Spreadsheet, utilizzando
3    * l'identificativo, seleziona il foglio sulla
4    * quale si sta lavorando e ne prende i dati per
5    * ciclarli iterativamente.
6    * Restituisce una lista contenente tutte le
7    * occasioni di contatto.
8    */
9   function getOccasioneContatto() {
10      /* Apertura Spreadsheet con identificatore in ingresso */
11      var ss2 = SpreadsheetApp.openById(ID_OccasioniContatto);
12      /* Focus sul foglio di lavoro che gli si passa */
13      var s2 = ss2.getSheetByName("Database");
14      /* Restituisce un range corrispondente ai dati presenti nello Spreadsheet */
15      var data2 = s2.getDataRange().getValues();
16      /* Numero di  righe da saltare partendo dall'alto */
17      var headers2 = 1;
18      /* L'indice delle colonne inzia da zero */
19      var tagColumn2 = 0;
20      /* Variabile per contenere le occasioni di contatto */
21      var occasioniContatto = [];
22      /* Ad ogni iterazione viene aggiunta una tupla alla variabile */
23      for (var row=headers2; row < data2.length; row++) {
24        occasioniContatto.push(data2[row][tagColumn2]);
25      }
26      return( occasioniContatto );
27  }
```

Figura 4.8 - Sorgente getOccasioniContatto()

La funzione in figura 4.8 si rende molto utile, soprattutto perché è possibile aggiungere occasioni di contatto al foglio di lavoro senza dover modificare il sorgente, direttamente da Spreadsheet aggiornando il documento.

Per quanto riguarda la rappresentazione delle occasioni di contatto all'interno del form, è stato creato un file Html.

4.3.2 Il file OccasioniContatto.html

Il file "OccasioniContatto.html" è un documento html, che utilizza la funzione `getOccasioneContatto()` per ottenere la lista delle occasioni di contatto, e per ognuna di esse, effettua un ciclo e crea l'adeguato codice html, incluso il campo *entry*, precedentemente menzionato.

Il tag `<input>` specifica un campo di inserimento dati nel form, più precisamente un campo checkbox con un identificativo assegnato in base all'indice dell'iterazione ed un valore, in base alla tupla estratta dallo Spreadsheet.

Nella figura 4.9 viene rappresentato il codice sorgente del file.

```
1  <? for (var i = 0; i < getOccasioneContatto().length; ++i) { ?>
2      <span style="display:inline-block">
3      <input type="checkbox" name="entry.793889370" id="i" value="<?!=
       getOccasioneContatto()[i] ?>"/>
4      <?!= getOccasioneContatto()[i] ?>
5      </span>
6  <? } ?>
7
8
```

Figura 4.9 - Sorgente OccasioneContatto()

4.4 Auto-completamento

Con auto-completamento, si intende l'implementazione di un meccanismo che permetta di effettuare ricerche all'interno dell'anagrafica delle ragioni sociali.
La tabella delle ragioni sociali è ottenuta da un'estrazione da appositi database di circa 100K ragioni sociali ed aggiornata periodicamente.

Per permettere agli utenti che compilano il form di indicare una ragione sociale occorre mettere a punto una modalità di selezione adeguata. Ad esempio una funzione di ricerca che durante la digitazione delle prime lettere della ragione sociale, proponga dinamicamente le prime 50 occorrenze.
Il meccanismo di auto-completamento che abbiamo implementato sfrutta jQuery.

4.4.1 jQuery

È una libreria JavaScript per applicazioni web. Nasce con l'obiettivo di semplificare la selezione, la manipolazione, la gestione degli eventi e l'animazione di elementi DOM in pagine HTML, nonché implementare funzionalità AJAX.

Il framework jQuery, può essere visto come un livello di astrazione, include una moltitudine di funzionalità che per essere implementate in codice JavaScript, necessiterebbero di numerose linee di codice. Invece utilizzando jQuery è possibile eseguire determinati compiti, più o meno complessi, scrivendo effettivamente molte meno righe di codice, ciò comporta non solo un guadagno significativo in termini di tempo ma anche una significativa diminuzione delle possibilità di commettere errori.

4.4.2 Come includere jQuery nella pagina html

Esistono molti modi per includere jQuery all'interno della propria pagina web e poter utilizzare tutte le sue funzionalità.

Il metodo che abbiamo adottato è quello di collegarlo direttamente al repository di jQuery, quindi includere direttamente l'URL all'interno della pagina.

4.4.3 Il file Autocomplete.html

Il file Autocomplete.html contiene l'implementazione del meccanismo di auto-completamento sopracitato, nel paragrafo 4.4.
Al suo interno si trovano una parte dedicata all'inclusione di jQuery, seguita da codice html per la rappresentazione del campo Ragione Sociale all'interno del form. L'ultima parte di cui si compone il file, è inerente al codice JavaScript, inserito all'interno del tag `<script> ... </script>`, dove troviamo il codice di due funzioni; la prima è una funzione anonima mentre la seconda è `buildTagList()`.

4.4.3.1 Classe google.script.run(Client-side API)

La prima funzione fa uso di "google.script.run, è un'applicazione JavaScript asincrona client-side disponibile nelle pagine di servizio Html, che può chiamare le funzioni server-side di Apps Script.

Il metodo `withSuccessHandler()` imposta una funzione di ritorno da eseguire se una determinata funzione lato server ritorna con codice di successo. Il valore di ritorno del server viene passato come primo argomento.
Nel nostro caso, la funzione `getAvailableTags()`, se eseguita con successo restituirà una lista di ragioni sociali estratte direttamente dall'anagrafica e le passerà come argomento a `buildTagList()`.

All'interno di `buildTagList()` viene implementato il meccanismo di auto-completamento sui dati della variabile `availableTags`. Come mostrato nella figura 4.10.

```
1   <script src="https://code.jquery.com/jquery-1.9.1.js"></script>
2   <script src="https://code.jquery.com/ui/1.10.3/jquery-ui.js"></script>
3
4
5   <div class="ui-widget">
6       <label>
7           <span class="label-text">Ragione Sociale</span>
8           <input name="entry.1450737721" type="text" id="tags" />
9       </label>
10  </div>
11
12
13  <script>
14  /* La variabile window viene collegato ad un gestore di eventi */
15  window.addEventListener('carica', function() {
16    console.log('Pagina caricata');
17  });
18
19  /* Il codice viene eseguito quando la pagina viene caricata,
20   * esegue buildTagsList se la funzione lato server termina
21   * correttamente.
22   */
23    $(function() {
24    google.script.run.withSuccessHandler(buildTagList)
25        .getAvailableTags();
26  });
27
28  function buildTagList(availableTags) {
29    $( "#tags" ).autocomplete({
30      source: availableTags
31    });
32  }
33  </script>
```

Figura 4.10 - Sorgente Autocomplete.html

4.5 Funzione doGet()

Il servizio Html, consente di pubblicare pagine web in grado di interagire con le funzioni di script di Apps Server. È particolarmente utile per la creazione di applicazioni web o l'aggiunta di interfacce utente personalizzate in Documenti, Fogli e Moduli di Google. È anche possibile utilizzarlo per generare il corpo di email.

A causa di motivazioni dovuti alla sicurezza, gli script non possono inviare direttamente il loro contenuto ad un browser. Devono invece ripulire l'Html in modo che non possa eseguire azioni dannose.

4.5.1 Sandbox

Si tratta di un ambiente isolato e sicuro, che imita un intero sistema di computer. Nella sandbox, i programmi sospetti possono essere eseguiti per monitorare il loro comportamento e capire il loro scopo senza mettere in pericolo la rete di un'organizzazione.

Il metodo `setSandboxMode()`, è utilizzato per impostare la tipologia di sandbox.
Utile per proteggere gli utenti dall'essere serviti con Html o JavaScript dannosi. Il codice client-side viene servito dal servizio Html che lo esegue in un ambiente che impone restrizioni al codice stesso.

4.5.1.1 Sandbox IFRAME

Una modalità sandbox che utilizza `IFRAME` invece della tecnologia Caja, utilizzata dalle modalità `EMULATED` e `NATIVE`.
La modalità `IFRAME` impone molte più restrizioni rispetto alle altre modalità di sandbox, ma funziona più velocemente.
Può non funzionare in alcuni browser meno recenti, come Internet Explorer 9. La modalità di sandbox può essere letta anche in uno script lato client, controllando google.script.sandbox.mode.

4.5.1.2 Sandbox Emulated

Una modalità sandbox legacy che emula la modalità rigorosa di `ECMAScript 5`, utilizzando solo funzionalità disponibili in `ECMAScript 3`

4.5.1.3 Sandbox Native

Una modalità sandbox costruita in cima alla modalità rigorosa di `ECMAScript 5`
Le modalità `NATIVE` e `EMULATE` sono state deprecate il 13 ottobre 2015. Ora è supportata solo la modalità `IFRAME`.

In figura 4.11 il codice `doGet()`.

```
1   /*
2    * Per creare un app web, con il servizio HTML,
3    * il codice deve includere una funzione doGet(),
4    * che indica allo script come servire la pagina.
5    * Deve restituire un oggetto HtmlOutput.
6    */
7   function doGet() {
8     /* Creazione di un oggetto template con relativo nome */
9     var template = HtmlService
10    .createTemplateFromFile('index');
11
12    /* Creazione di un oggetto htmlOutput da un template,
13     * e impostazione del metodo di sandbox.
14     */
15    var htmlOutput = template.evaluate()
16    .setSandboxMode(HtmlService.SandboxMode.IFRAME)
17    .setTitle('Form');
18
19    /* Ottiene titolo ed elementi che fanno parte del form */
20    var title = form.getTitle();
21    var items = form.getItems();
22    return htmlOutput;
23  }
```

Figura 4.11 - Funzione doGet()

4.6 Funzione include()

È una funzione che crea un nuovo oggetto `HtmlOutput` da un file che gli viene dato in ingresso.

Il codice nell'`HtmlOutput` può includere JavaScript incorporato(embedded) e CSS.
È un codice JavaScript standard che manipola in DOM ma non Apps Script.

In figura 4.12 il codice della funzione `include()`.

```
1  /*
2   * Crea un nuovo oggetto HtmlOutput da un file
3   * in ingresso.
4   */
5  function include(filename) {
6    return HtmlService
7    .createHtmlOutputFromFile(filename)
8    .getContent();
9  }
```

Figura 4.12 - Funzione include()

4.7 Funzione includeTemplate()

Gli scipt di Google possono contenere tre tag speciali, chiamati scriptlet.

All'interno di ognuno di essi è possibile scrivere codice eseguibile come in un normale progetto Apps Script.

Possono richiamare funzioni definite in altri file, variabili globali o utilizzare qualsiasi API di Apps Script.

- **Scriptlet standard**: utilizzano la sintassi <? ... ?>, eseguono codice senza esplicitamente pubblicare contenuti sulla pagina.
- **Scriptlet di stampa**: utilizzano la sintassi <?= ... ?>, restituiscono il risultato utilizzando la fuga contestuale. Con fuga contestuale si intende, il meccanismo che permette ad Apps Script di mantenere il contesto dell'output della pagina, all'interno di un attributo Html, di un tag di script lato client o altrove, ed aggiunge automaticamente i caratteri di escape per proteggere dagli attacchi di scripting cross-site (XSS).
- **Scriptlet di stampa forzata**: utilizzano la sintassi <?!= ... ?>, sono come gli script di stampa, eccetto per il fatto che non utilizzano la fuga contestuale. La fuga contestuale è utile se lo script può esser soggetto ad input non autorizzati degli utenti.

Come regola generale, utilizzare script di stampa piuttosto che script di stampa forzata, a meno di casi specifici.

Il codice viene eseguito sul server prima che la pagina venga inviata all'utente. Poiché il codice script viene eseguito prima della pubblicazione della pagina, può essere eseguito solo una volta per pagina, a differenza delle funzioni JavaScript client-side o Apps Script che possono essere richiamate tramite google.script.run.

In figura 4.13 il codice della funzione `includeTemplate()`.

```
1   /*
2    * Crea un nuovo oggetto HtmlTemplate da un file
3    * in ingresso.
4    * Il metodo evaluate() deve precedere la selezione
5    * della sandbox.
6    */
7   function includeTemplate(filename) {
8     return HtmlService
9     .createTemplateFromFile(filename)
10    .evaluate()
11    .setSandboxMode(HtmlService.SandboxMode.IFRAME)
12    .getContent();
13  }
```

Figura 4.13 - Sorgente includeTemplate()

4.8 Funzione aggiungiUltimoContatto()

La funzione per prima cosa, apre il file Spreadsheet delle riposte al form. Si focalizza sull'ultimo contatto che ha risposto al modulo e ne crea i campi utili per Google Contacts.

La scelta fatta, è stata di creare un gruppo per ogni occasione di contatto e aggiungere le istanze di contatto ai gruppi che ogni utente ha selezionato nel form, così da avere per ogni gruppo i contatti accomunati dalla stessa occasione di contatto.

Per fare ciò la funzione controlla, prima della creazione di un nuovo gruppo, che non vi siano già duplicati, e in tal caso la creazione avviene senza problemi. In caso contrario si interrompe la procedura.

Nella figura 4.14, viene mostrata la funzione `aggiungiUltimoContatto()`, che crea una nuova istanza di contatto da aggiungere alla rubrica di Google Contacts.

```
1   /*
2    * Creazione di un nuovo contatto ed aggiunta ai gruppi selezionati
3    * nel form. Controllo di duplicati prima della creazione di nuovi
4    * gruppi su Contacts.
5    */
6   function aggiungiUltimoContatto(){
7     var ss = SpreadsheetApp.openById(ID_Risposte);
8     var s = ss.getActiveSheet();
9     /* Data è la matrice dei dati contenuti sul foglio di lavoro delle risposte */
10    var data = s.getDataRange().getValues();
11    var headers = 1;                    // Numero di righe in cima da saltare
12    data.splice(0,headers);             // Rimozione righe in cima
13    /* Posiziona row sull'ultimo contatto inserito nello Spreadsheet */
14    var row = data[data.length - 1];
15    /*
16     * Le colonne sono indicizzate a partire da zero.
17     * contattoOccasioneContatto, contiene tutte le occasioni del contatto in questione,
18     * stesso discorso per le altre variabili.
19     */
20    var occasioneContatto = row[1];
21    var ragioneSociale = row[2];
22    var firstName = row[3];
23    var lastName = row[4];
24    var emailAdd = row[5];
25    var telefonoMobile = row[6];
26    var telefonoUfficio = row[7];
27    var telefonoCentralino = row[8];
28    var indirizzoUfficio = row[9];
29    var sitoWeb = row[10];
30    var ruoloPosizione = row[11];
31    occasioneContatto = occasioneContatto.split(", ");
32    /* Creazione contatto con nome, cognome ed email */
33    var contact = ContactsApp.createContact(firstName, lastName, emailAdd);
34    /* Aggiunta dei campi mancanti al contatto appena creato */
35    contact.addCustomField("Ragione Sociale", ragioneSociale);
36    contact.addPhone("Telefono Mobile", telefonoMobile);
37    contact.addPhone("Telefono Ufficio", telefonoUfficio);
38    contact.addPhone("Telefono Centralino", telefonoCentralino);
39    contact.addAddress("Indirizzo Ufficio", indirizzoUfficio);
40    contact.addUrl("Sito Web", sitoWeb);
41    contact.addCustomField("Ruolo/Posizione", ruoloPosizione);
42    /*
43     * Iterazione su tutte le occasioni di contatto per
44     * controllare che vi siano duplicati nei gruppi
45     * di Google Contacts.
46     */
47    for(var i=0; i<occasioneContatto.length; i++){
48      if(creaOccasioneContatto(occasioneContatto[i])){
49        var group = ContactsApp.createContactGroup(occasioneContatto[i]);
50        group.addContact(contact);
51      }else{
52        var group = ContactsApp.getContactGroup(occasioneContatto[i]);
53        group.addContact(contact);
54      }
55    }
56  }
```

Figura 4.14 - Sorgente aggiungiUltimoContatto()

4.9 Funzione createFormEditTrigger()

La classe ScriptApp consente agli utenti di creare trigger e di controllare il servizio di pubblicazione degli script. Concede l'accesso e la manipolazione di trigger e script per la pubblicazione.

Il processo di creazione di un trigger avviene con il metodo `newTrigger()`, quando attivato verrà chiamata una determinata funzione, presa come variabile in ingresso.

La funzione `createFormEditTrigger()`, utilizza l'identificativo del modulo per aprirlo in lettura, dopodiché viene creato un trigger apposito, che richiamerà la funzione `aggiungiUltimocontatto()`, in modo da automatizzarla.

La direttiva, `onFormSubmit()`, specifica il comportamento del trigger. Infatti verrà attivato solamente dopo una sottomissione dello stesso form, in modo da rispettare il comportamento desiderato.

Il codice viene presentato nella figura 4.15.

```
1   /*
2    * Funzione che crea un trigger, attivato quando
3    * un utente compila e sottomette il form.
4    */
5   function createFormEditTrigger() {
6     /* Apertura del Form con il suo idetificativo */
7     var form = FormApp.openById(ID_Form);
8     /*
9      * Creazione di un trigger, utilizzando la
10     * la funzione passata in ingresso.
11     */
12    ScriptApp.newTrigger("aggiungiUltimoContatto")
13    .forForm(form)
14    .onFormSubmit()
15    .create();
16  }
```

Figura 4.15 - Sorgente createFormEditTrigger()

4.10 Funzione creaOccasioneContatto()

La classe ContactsApp permette di accedere ai propri contatti presenti in Google Contacts, crearne di nuovi, modificare quelli presenti ed anche eliminare contatti obsoleti.

La funzione, ottiene l'elenco completo dei gruppi di contatti dell'utente.

Un utente può disporre di un elenco di contatti e potenzialmente anche di un elenco di gruppi di contatti.
Ogni gruppo di contatti può contenere altri contatti.

La figura 4.16, mostra il sorgente della funzione in questione.

```
1   /*
2    * Funzione che crea una nuova occasione di contatto,
3    * in Google Contacts, sotto forma di gruppo,
4    *  controllando che non sia un duplicato.
5    */
6   function creaOccasioneContatto(occasioneContatto){
7     /* Lista delle occasioni di contatto */
8     var occasioneDiContatto = getOccasioneContatto();
9     var occasioneDiContatto = occasioneDiContatto.splice(",");
10    /* Lista dei gruppi esistente in Google Contacts. */
11    var groups  = ContactsApp.getContactGroups();
12    /* Ciclo di controllo per evitare gruppi duplicati. */
13    for(var i=0; i<groups.length; i++){
14      if(occasioneContatto === groups[i].getName()){
15        return false;
16      }
17    }
18    return true;
19  }
```

Figura 4.16 - Funzione creaOccasioneContatto()

4.11 Aspetto del Form

L'aspetto del Form si è resa una parte importante del progetto, dal momento che consiste nel presentare un form ad alcuni partner di progetto. È necessario un aspetto che richiami la professionalità intrinseca che la caratterizza, e allo stesso tempo immediatezza del compito che è chiamato a svolgere.

4.11.1 CSS

Il CSS, Cascading Style Sheet, è un linguaggio usato per definire la formattazione di documenti Html, XHtml e XML. Le regole per comporre il CSS sono contenute in un insieme di direttive emanata a partire dal 1996 dal W3C (World Wide Web Corsortium).

Istruiscono un browser o un altro programma su come il documento debba essere presentato all'utente, per esempio definendone i font, i colori, e immagini di sfondo, il layout, il posizionamento delle colonne o di altri elementi sulla pagina.

4.11.1.1 W3C

È un'organizzazione non governativa internazionale che ha come scopo quello di sviluppare tutte le potenzialità del Word Wide Web.
Al fine di riuscire nel proprio intento la principale attività svolta, consiste nello stabilire standard tecnici per il WWW inerenti sia i linguaggi di markup che protocolli di comunicazione.

In generale un linguaggio di markup è un insieme di regole che descrivono i meccanismi di rappresentazione strutturali, semantici o prestazionali di un testo che, utilizzando appunto convenzioni standardizzate, sono utilizzabili su più supporti.

4.11.2 Com'è fatta una regola CSS

È composta i due blocchi principali:

- Il selettore
- Il blocco delle dichiarazioni

Il selettore serve a definire la parte del documento a cui verrà applicata la regola. I selettori possono essere di diverso tipo.
Il blocco delle dichiarazioni è delimitato da parentesi graffe, al suo interno possono trovare posto più dichiarazioni. Ogni dichiarazione è composta dalla coppia:

- Proprietà
- Valore

La proprietà definisce un aspetto dell'elemento da modificare secondo il valore espresso. Proprietà e valore devono essere separati da due punti. Le dichiarazioni vanno invece separate con un punto e virgola.

4.11.2.1 @-rules

Le @-rules sono tipi particolari di costrutti che hanno una caratteristica comune; sono tutti introdotti dal simbolo della chiocciola.
Rappresentano vie alternative, ma spesso più flessibili e potenti. Viene usata per collegare un foglio di stile esterno al documento.

4.12 Stylesheet

Mantenere tutti i codici Html, CSS, e JavaScript in un unico file può rendere il progetto difficile da leggere e manutenere, si è deciso di separare il CSS in un file apposito, chiamato Stylesheet.

In questo file sono state espresse le regole che hanno portato all'aspetto essenziale ma necessario che ci eravamo imposti sin dall'inizio, nella figura 4.17 vengono mostrare alcune delle regole più significative inerenti al progetto.

```
1   @import url(https://fonts.googleapis.com/css?family=Cookie|Raleway:300,700,400);
2   body{
3       background: url('http://tfgms.com/sandbox/dailyui/bg-1.jpg') center no-repeat;
4       background-size: cover;
5       color: #333;
6       font-size: 15px;
7       font-family: 'Raleway', sans-serif;
8   }
9   form button.submit{
10      background: rgba(255,255,255,0.25);
11      border: 1px solid #333;
12      line-height: 1em;
13      padding: 0.5em 0.5em;
14      -webkit-transition: all 0.25s;
15      transition: all 0.25s;
16  }
17  form label.checkbox span:before{
18      content: '\e157';
19      display: inline-block;
20      font-family: 'Glyphicons Halflings';
21      font-size: 1.125em;
22      padding-right: 0.25em;
23      position: relative;
24          top: 1px;
25  }
26  h1{
27      font-size: 3em;
28      margin: 0 0 0.5em 0;
29      text-align: center;
30      font-family: 'Cookie', cursive;
31  }
32  html{
33      font-size: 18px;
34      height: 100%;
35  }
36  .text-center{
37      text-align: center;
38  }
```

Figura 4.17 - Sorgente Stylesheet

4.13 Il Form

Il form finale che viene presentato è inerente ad un determinato progetto, chiamato "Jump", da cui il titolo. I campi sono quelli standard, con l'aggiunta di quelli già discussi ampiamente in precedenza, quali "Ragioni Sociali" ed "Occasioni di Contatto".
Nella figura 4.18 viene mostrato l'aspetto del form finale da presentare ad un partner di uno specifico progetto, nella fattispecie, il progetto Jump.

Figura 4.18 - Form finale

Acronimi

XSS	*Cross-Site Scripting*
CSV	*Comma Separated Variable*
DOM	*Document Object Model*
CSS	*Cascading Stylesheet*
AJAX	*Asynchronous JavaScrit and XML*
XML	*Extensible Markup Language*
API	*Application Program Interfaces*
WWW	*World Wide Web*
SRS	*Software Requirement Specification*
HTML	*HyperText Markup Language*
XHTML	*eXtensible HyperText Markup Language*

Conclusioni

Nel corso dell'elaborato è stato presentato Google Apps Script in tutti i sui aspetti fondamentali, ed è stata introdotta la potenzialità di un tale ambiente di sviluppo.
Per poter comprendere l'approccio da attuarsi ad una tale piattaforma è stato indispensabile approfondire ogni singolo applicativo che ne facesse parte, per poi successivamente avere una visione più nitida del sistema nel complesso. Avendo guadagnato dimestichezza con l'ambiente ed il linguaggio, trovare una metodologia per creare una simbiosi tra i diversi elementi, considerando le loro intrinseche differenze ma allo stesso tempo tutte improntate ad un uso collaborativo vicendevole, è stato un obiettivo che possiamo considerare pienamente raggiunto.
Indispensabile è stato fornire un aspetto che fosse gradevole, vista la natura del progetto, per ottenere tale risultato sono stati utilizzati linguaggi appositi, tra cui il CSS, che ha permesso di strutturare il modulo nelle metodologie e nelle fattezze che sono state presentate.
Giungiamo al traguardo della trattazione auspicando che il lettore al termine dell'elaborato abbia maturato le infinite possibilità che può offrire tale strumento e ci rammarichiamo di non aver potuto disaminare nella sua totalità quello che Google stesso ha definito, in senso metaforico, *"una piattaforma che vive e che respira"*.

Bibliografia

[DanM16] "*Google Apps Script: Una panoramica sul linguaggio che può cambiare l'approccio ai servizi di Google*" di Daniele Marino, PC Professionale, Luglio 2016.

[SerG14] "*Google Apps Script for Beginners: Customize google Apps using Apps Script and explore its powerful features*" di Serge Gabet, Editore PACKT Publishing, febbraio 2014.

[JamF14] "*Google Apps Sc*ript: Web Application Devolepment Essentials, 2nd Edition" di James Ferreira, Editore O'Reilly, Marzo 2014.

[RamG16] "*Learning Google Apps Script: Customize and automate Google Application using Script*" di Ramalingam Ganapathy, Editore PACKT Publishing, Marzo 2016.

[CabG09] "*Ajax*" slides Dipartimento dell'informazione, Giacomo Cabri, Febbraio 2009.