ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

Scuola di Scienze
Corso di Laurea Magistrale in Ingegneria e Scienze Informatiche

STUDIO E REALIZZAZIONE DELL'INTERFACCIA GRAFICA PER UN APPLICATIVO HEALTH

Elaborato in
Programmazione Di Applicazioni Data Intensive

Relatore
Prof.sa Antonella Carbonaro

Co-relatori
Prof. Gianluca Moro
Dott. Giacomo Frisoni

Presentata da
Matteo Sertori

Prima Sessione di Laurea
Anno Accademico 2020 – 2021

PAROLE CHIAVE

User Experience Design

Data Science

User Interface Design

Bootstrap

R Shiny

A Genga e Betta

Introduzione

Le malattie rare sono malattie che colpiscono un numero limitato di persone rispetto alla popolazione generale e vengono sollevate questioni specifiche in relazione alla loro rarità. In Europa, una malattia è considerata rara quando colpisce 1 persona su 2000. Una malattia può essere rara in una regione, ma comune in un'altra.

Il campo delle malattie rare soffre di una carenza di conoscenze mediche e scientifiche. Per molto tempo medici e ricercatori non erano a conoscenza delle malattie rare e fino a tempi molto recenti non esisteva una vera ricerca o una politica di salute pubblica sulle questioni legate al settore. Non esiste una cura per la maggior parte delle malattie rare, ma il trattamento appropriato e l'assistenza medica possono migliorare la qualità della vita delle persone colpite e prolungare la loro aspettativa di vita. Sono già stati compiuti notevoli progressi per alcune malattie, il che dimostra che non si deve rinunciare alla lotta, ma, al contrario, continuare e intensificare gli sforzi nel campo della ricerca e della solidarietà sociale.

Coloro che sono affetti da queste malattie affrontano tutti difficoltà simili nella ricerca di una diagnosi, informazioni pertinenti e una corretta direzione verso professionisti qualificati. Vengono altre sì sollevate questioni specifiche relative all'accesso a un'assistenza sanitaria di qualità, al supporto sociale e medico generale, al collegamento efficace tra ospedali e studi medici generali, nonché all'integrazione professionale e sociale e all'indipendenza.

Le persone colpite da malattie rare sono anche più vulnerabili psicologicamente, socialmente, economicamente e culturalmente. Queste difficoltà potrebbero essere superate con politiche appropriate. A causa della mancanza di sufficienti conoscenze scientifiche e mediche, molti pazienti non vengono diagnosticati e la loro malattia rimane non identificata. Queste sono le persone che soffrono maggiormente delle difficoltà nel ricevere un sostegno adeguato.

Queste incertezze portano alle persone coinvolte (malati, famigliari e anche caregivers) a cercare delle risposte e rimedi chiedendo a chi è riuscito a conviverci e a battere queste malattie rare: i social network sono il mezzo più facile ed intuitivo per poter raggruppare un numero elevato di persone attraverso l'uso di gruppi. La tesi effettuata da Giacomo Frisoni verteva su questo: ottenere

informazioni attraverso l'uso si tecniche di *Text Mining* e *Web Semantico*. Usando però questi strumenti (molto tecnici) un utente normale non sarebbe in grado di visionare tali informazioni.Grazie alla creazione di una Graphics User Interface pubblicata su un sito Web, qualsiasi utente potrebbe visionare e riuscire anche a trovare una soluzione alla propria malattia.

In questo elaborato si prenderà in esame una specifica malattia rara: la *Acalasia Esofagea* (ORPHA:930); ma con pochissime modifiche si potrebbe applicare a qualsiasi altro tipo di malattia o qualsialsi altro ambito di studio interessato.

Il lavoro di tesi è stato suddiviso nei seguenti capitoli:

- **Capitolo 1** - Analisi preliminare sullo studio della User Experience Design;
- **Capitolo 2** - Panoramica sulle varie tecnologie esistenti in letteratura utilizzabili per il problema posto;
- **Capitolo 3** - Introduzione alla modellazione del problema con i primi approcci alle difficoltà riscontrate e alle relative soluzioni;
- **Capitolo 4** - Il lavoro realizzato e il codice impiegato.

Indice

Elenco delle figure

Capitolo 1

User Experience Design

1.1 Introduzione

La qualità percepita di un sito Web è fortemente dipendente dagli utenti e, più precisamente, dalla loro esperienza di navigazione. Alla base di una buona progettazione deve esserci, senza dubbio, una buona organizzazione dei contenuti, ma non solo. La User Experience va oltre tutto questo. Quando un utente raggiunge un sito, deve essere invogliato alla navigazione, a consultare i contenuti proposti, a volerci ritornare non solo perché ha raggiunto il proprio obiettivo, ma anche per una serie di sensazioni, emozioni e reazioni che coinvolgono la sua psicologia nel modo più positivo possibile. In tutto ciò, il design è essenziale: ad un qualsiasi utente deve essere permesso di trovare ciò che cerca in modo agevole, navigando con spontaneità all'interno del sito Web e potendo godere di un'esperienza che provochi sensazioni positive, tali da spingerlo a ritornare su quel determinato sito Web. Ad oggi la concorrenza è elevatissima. Come se non bastasse, le operazioni che si possono svolgere online sono aumentate, e continuano ad aumentare, a dismisura, con la possibilità di accesso per tutti e ovunque. Con uno scenario di tale complessità, il successo è garantito solo a quei prodotti che riusciranno a fornire agli utilizzatori una User Experience straordinaria.

1.2 Usabilità e User Experience

Alla nascita dei primi strumenti informatici, progettista ed utente coincidevano e l'attenzione era rivolta, principalmente, al mezzo piuttosto che allo scopo. Con la diffusione dell'informatica negli uffici e nelle case negli anni Ottanta, si comincia a parlare di fattori umani nella progettazione dei prodotti informatici. È in questo momento che nasce la Human Computer Interaction o

HCI , ovvero la scienza che si occupa dell'interazione uomo macchina e che si prefissa l'obiettivo di avvicinare il modello del progettista a quello dell'utente. Ovviamente, gli obiettivi prefissi dall'HCI risultarono difficili da raggiungere all'inizio, ma, negli anni Novanta, si decise di intervenire direttamente sulla progettazione: il processo divenne iterativo e si integrarono i processi di analisi con attività che permettessero di individuare i bisogni degli utenti[1]. Analizzando un grande esponente di siti Web, Jakob Nielsen stilò una serie di principi che permettono la valutazione dell'usabilità; questi hanno a che fare con:

- la **navigabilità del sito Web**. Il sistema di navigazione del sito deve aiutare l'utente a orientarsi e a trovare le informazioni;
- lo **svolgimento di interazioni**. L'interfaccia deve rendere semplice l'operazione;
- l'**utilità attesa**. Il sito Web deve avere contenuti che corrispondono alle attese degli utenti;
- la **completezza dei contenuti**. Il sito Web deve contenere informazioni con il grado di completezza che gli utenti desiderano;
- la **comprensibilità dei contenuti**;
- la **grafica**. Un sito deve avere una grafica piacevole, ma funzionale ai suoi obiettivi.

Garantire una buona usabilità è importante perché riesce a ottimizzare la navigabilità del sito e allo stesso tempo riduce i punti di uscita soddisfacendo le attese degli utenti con una piacevole permanenza sul sito Web. Sempre per usare le parole di Nielsen, i motivi per cui un utente sarà spinto a ritornare in un sito e i fattori che ne determinano l'eccellenza sono racchiusi negli acronimi "H.O.M.E.".

- **High-quality**. Elevata qualità dei contenuti;
- **Often**; Frequenza degli aggiornamenti;
- **Minimal**; Breve durata dei tempi di scaricamento;
- **Ease**; Facilità di utilizzo;

Tenendo presente l'enorme concorrenza, è bene non incorrere in errori legati all'usabilità perché se un utente arrivasse sul sito e lo abbandona, sarebbe davvero un errore "costoso", ma soprattutto evitabile. Spesso la User Experience e l'usabilità sono trattati come sinonimi, ma, in realtà, appartengono

a sfere diverse. La User Experience non è l'usabilità: la prima si focalizza sul come l'utente vive l'interazione con un sistema, mentre la seconda si concentra sull'efficacia dell'interfaccia. L'usabilità è una precondizione, cioè un applicativo Web deve essere usabile per consentire agli utenti di raggiungere i loro scopi; tuttavia, ci sono applicativi usabili che non forniscono una buona User Experience: è necessario andare oltre l'usabilità, creando dei prodotti effettivamente adeguati alle conoscenze e al contesto d'uso degli utenti. Gli aspetti coinvolti sono molteplici: il design, l'usabilità, le performance, l'utilità, il marketing, l'accessibilità, e anche la sfera psicologica ed emozionale, perché la User Experience non si preoccupa solo del completamento del task richiesto dall'utente, ma anche della qualità con cui viene raggiunto.

1.3 Cenni di storia

Nell'era batch (1945–1968), la potenza di calcolo era estremamente scarsa e costosa. Le interfacce utente erano rudimentali. Le interfacce utente erano state considerate un sovraccarico per il computer e il software è stato progettato per mantenere il processore al massimo utilizzo con il minor sovraccarico possibile.

Il lato input delle interfacce utente per le macchine batch era principalmente schede perforate o supporti equivalenti come il nastro di carta. Con l'eccezione limitata della console dell'operatore del sistema, gli esseri umani non interagivano affatto con le macchine batch in tempo reale.

L'invio di un lavoro a una macchina batch era coinvolta, in primo luogo, con la preparazione di un mazzo di schede perforate che descrivono un programma e un set di dati. La perforazione delle schede del programma non era eseguita sul computer stesso, ma su keypunch, macchine specializzate simili a macchine da scrivere che erano notoriamente ingombranti, spietate e soggette a guasti meccanici. L'interfaccia del software era altrettanto spietata, con sintassi molto rigide che dovevano essere analizzate dai compilatori e dagli interpreti più piccoli possibili.

I primi sistemi batch davano al lavoro attualmente in esecuzione l'intero computer; i deck e i nastri del programma dovevano includere quello che ora potremmo pensare come codice del sistema operativo per parlare con i dispositivi I / O e fare qualunque altra operazione di pulizia fosse necessaria. A metà del periodo discontinuo, dopo il 1957, vari gruppi iniziarono a sperimentare i cosiddetti sistemi "load-and-go". Questi utilizzavano un programma di monitoraggio che era sempre residente sul computer. Un'altra funzione del monitor era quella di effettuare una migliore verifica degli errori sui lavori inviati, rilevando gli errori prima e in modo più intelligente e generando un feedback più utile per

gli utenti. Pertanto, i monitor rappresentavano il primo passo verso entrambi i sistemi operativi e le interfacce utente progettate esplicitamente.

Nel 1969 le interfacce da riga di comando (CLI) si sono evolute dai monitor batch collegati alla console di sistema. Il loro modello di interazione era una serie di transazioni richiesta-risposta, con richieste espresse come comandi testuali in un vocabolario specializzato. La latenza era di gran lunga inferiore rispetto ai sistemi batch, scendendo da giorni o ore a secondi. Di conseguenza, i sistemi a riga di comando hanno permesso all'utente di cambiare idea sulle fasi successive della transazione in risposta a feedback in tempo reale o quasi in tempo reale su risultati precedenti. Il software potrebbe essere esplorativo e interattivo in modi prima impossibili. Ma queste interfacce ponevano ancora un carico mnemonico relativamente pesante sull'utente, richiedendo un serio investimento di sforzo e tempo di apprendimento da padroneggiare.

I primi sistemi a riga di comando combinavano teleprinter e computer, adattando una tecnologia matura che si era dimostrata efficace per mediare il trasferimento di informazioni tra esseri umani. Le teleprinter erano state originariamente inventate come dispositivi per la trasmissione e la ricezione automatiche del telegrafo; avevano una storia che risale al 1902 e si erano già affermati nelle redazioni e altrove nel 1920.

La diffusa adozione di terminali video-display (VDT) a metà degli anni '70 ha inaugurato la seconda fase dei sistemi a riga di comando. Riducono ulteriormente la latenza ed aiutarono a reprimere la resistenza conservatrice alla programmazione interattiva tagliando i materiali di consumo come l'inchiostro e carta.

Altrettanto importante, l'esistenza di uno schermo accessibile - una visualizzazione bidimensionale di testo che poteva essere modificata in modo rapido e reversibile - ha reso economico per i progettisti di software implementare interfacce che potrebbero essere descritte come visive piuttosto che testuali. Le applicazioni pionieristiche di questo tipo erano giochi per computer e editor di testo; i primi discendenti di alcuni dei primi esemplari, come canaglia e vi, sono ancora parte viva della tradizione Unix.

Nel 1985, con l'inizio di Microsoft Windows e altre interfacce utente grafiche, IBM ha creato quello che viene chiamato lo standard SAA (Systems Application Architecture) che include il derivato Common User Access (CUA). CUA ha creato con successo ciò che conosciamo e utilizziamo oggi in Windows e la maggior parte delle più recenti applicazioni DOS o Windows Console utilizzerà anche quello standard.

Ciò ha definito che un sistema di menu a discesa dovrebbe essere nella parte superiore dello schermo, la barra di stato in basso, i tasti di scelta rapida dovrebbero rimanere gli stessi per tutte le funzionalità comuni (ad esempio F2 per aprire funzionerebbe in tutte le applicazioni che seguivano lo standard

SAA). Ciò ha notevolmente aiutato la velocità con cui gli utenti potevano apprendere un'applicazione in modo che diventasse rapidamente uno standard del settore.

1.4 User Interface Design

La User Interface o UI è la progettazione di interfacce utente per qualsiasi macchina e software. In parole povere, il design dell'interfaccia utente è incentrato sul miglioramento dell'interattività e del contenuto visivo del prodotto. Prendiamo un sito Web come esempio: ci sono pagine, menu, cursori, pulsanti. Questi elementi visivi devono essere attraenti, funzionare come previsto e aiutare a interagire con il prodotto. Lo specialista dell'interfaccia utente si assicura che gli elementi dell'interfaccia siano organizzati in modo competente, strutturati, correlati, raggruppati, eseguiti esattamente e in uno stile simile.

Per comprendere meglio la progettazione dell'interfaccia utente, ci si può rivolgere ai 6 principi di progettazione dell'interfaccia utente descritti qui sotto.

- Il **Principio della struttura**. Si riferisce al modo in cui è organizzata l'interfaccia utente. La progettazione dell'interfaccia utente deve essere eseguita con uno scopo specifico, essere significativa e utile. Viene eseguito sulla base di modelli chiari e coerenti che sono evidenti e riconoscibili per gli utenti, mettendo insieme le cose correlate e separando le cose non correlate, differenziando le cose dissimili e facendo assomigliare le cose simili tra loro.

- Il **Principio della semplicità**. Si tratta di semplificare le attività per gli utenti. Include anche il processo di chiarimento della comunicazione e di fornire buone scorciatoie significativamente correlate a procedure più lunghe.

- Il **Principio della visibilità**. L'utente dovrebbe avere tutti i contenuti e le opzioni necessari in un luogo facilmente accessibile. Nulla dovrebbe distrarre l'utente con informazioni estranee o non necessarie.

- Il **Principio del feedback**. Gli utenti dovrebbero essere informati di azioni o interpretazioni, cambiamenti di stato o condizione ed errori o eccezioni che sono rilevanti e di interesse per l'utente attraverso un linguaggio chiaro, conciso e inequivocabile familiare agli utenti.

- Il **Principio della tolleranza**. Il design dovrebbe essere flessibile e tollerante, riducendo il costo di errori e uso improprio. Dovrebbe consentire l'annullamento e la ripetizione, prevenendo anche gli errori laddove

possibile tollerando vari input e sequenze e interpretando tutte le azioni ragionevoli ragionevoli.

- Il **Principio del riuso**. Il progetto dovrebbe riutilizzare componenti e comportamenti interni ed esterni, mantenendo la coerenza con lo scopo piuttosto che una coerenza semplicemente arbitraria, riducendo così la necessità per gli utenti di ripensare e ricordare.

1.5 User Experience Design

La user Experience o UX (in italiano "esperienza di utente") è un termine utilizzato per definire la relazione tra una persona e un programma. Il termine presuppone un approccio olistico: cerca di comprendere tutto ciò che ruota attorno all'interazione di un utente con un'azienda, un marchio o un'istituzione. L'esperienza utente coinvolge tutti gli aspetti esperienziali, affettivi, l'attribuzione del senso e del valore collegato ad un prodotto o servizio, all'interazione con esso e quanto ad esso è correlato, ma include anche le percezioni personali su aspetti quali l'utilità, la semplicità d'utilizzo e l'efficienza del sistema. Il processo di progettazione UX è stato utilizzato per secoli, ma mancava di un'etichettatura chiara. Ogni prodotto di successo nella storia umana ha coinvolto la progettazione di UX nel processo di sviluppo. Anche se i loro creatori non hanno usato un termine particolare. Più tardi, sono comparsi i termini come design centrato sull'utente e usabilità. Erano piuttosto diffusi nell'industria del software. Ma non sono stati in grado di comunicare pienamente l'obiettivo del creatore. Pertanto, nel 1988 Donald Norman ha coniato il termine design di User Experience.

Donald Norman spiegò il motivo per cui era apparso il termine[2]:

> "Ho inventato il termine perché pensavo che Human Interface e usabilità fossero troppo stretti: volevo coprire tutti gli aspetti dell'esperienza della persona con un sistema, inclusi design industriale, grafica, interfaccia, interazione fisica e manuale."

Il design di UX si concentra chiaramente sulle esigenze e le aspettative dell'utente su come il prodotto funziona, si presenta e si sente. Va ben oltre il contenuto visivo, tenendo conto di ogni piccolo dettaglio. In senso stretto, la UX è incentrata sull'interazione dell'utente finale con l'interfaccia del sito Web, dell'app mobile o del programma. Pertanto, i progettisti di UX lavorano direttamente con l'architettura del contenuto, la progettazione grafica e il contenuto stesso. Studiano il comportamento dell'utente target e sviluppano i loro prototipi. Il design della UX è una delle parti più importanti di una applicazione, perché aiuta a soddisfare le esigenze degli utenti. Un sondaggio

CEI afferma che l'86 percento degli acquirenti è pronto a pagare di più per una migliore esperienza del cliente. È ovvio ora che un ottimo design UX può aiutare a differenziare e attrarre gli utenti. Una UX ben ponderata può trasformare un visitatore in un utente e un utente in un cliente. Pertanto, i nomi per la pratica di progettazione UX possono essere cambiati, ma sarebbe comunque vitale per il processo di creazione di qualsiasi prodotto digitale.[3]

Lo scopo della UX è quello di realizzare un prodotto che corrisponda alla concezione dell'utente di un'esperienza significativa e sostanziale. Gli obiettivi dell'esperienza utente sono scoprire le esigenze dell'utente e cercare i modi per soddisfarle. E questo processo è, infatti, senza fine. La user experience non è qualcosa che si può fare una volta e dimenticare: anche quando viene lanciato il prodotto finale, il feedback degli utenti deve ancora essere elaborato e bisogna prestare attenzione. Lo scopo di una UX ben ponderata è quello di creare la migliore soluzione per il pubblico di destinazione del prodotto. Il prodotto non sarà mai in grado di dare tutto a tutti. Pertanto, i progettisti di UX si concentrano solo sugli utenti target e sulle loro esigenze. Il design UX non è qualcosa che si può coinvolgere nel processo quando il prodotto è quasi pronto. A volte i proprietari dei prodotti chiedono ai progettisti di UX di migliorare il prodotto quasi finito. Succede più frequentemente di quanto possa sembrare e, ovviamente, non funziona in questo modo. UX dovrebbe essere un elemento integrante dello sviluppo del prodotto sin dalla fase di convalida dell'idea. L'obiettivo principale della progettazione di UX è attrarre e trattenere gli utenti, fidelizzarli e motivarli ad acquistare i tuoi prodotti o servizi.

Lo schema sottostante a nido d'ape è uno strumento che spiega le varie sfaccettature della progettazione dell'esperienza utente e aiuta a trovare un punto debole tra le varie aree di una buona esperienza utente.

- **Utilizzabile**. Il sistema in cui viene fornito il prodotto o il servizio deve essere semplice e facile da usare. I sistemi dovrebbero essere progettati in modo familiare e di facile comprensione. La curva di apprendimento che un utente deve attraversare dovrebbe essere il più breve e indolore possibile.

- **Utile**. Il prodotto o il servizio di un'azienda deve essere utile e soddisfare un'esigenza. Se il prodotto o il servizio non è utile o soddisfa i desideri o le esigenze dell'utente, non esiste un vero scopo per il prodotto stesso.

- **Desiderabile**. L'estetica visiva del prodotto, servizio o sistema deve essere attraente e facile da tradurre. Il design dovrebbe essere minimale e al punto.

- **Ricercabile**. Le informazioni devono essere reperibili e facili da navigare. Se l'utente ha un problema, dovrebbe essere in grado di trovare rapidamente una soluzione. La struttura di navigazione dovrebbe anche essere impostata in modo sensato.
- **Accessibile**. Il prodotto o i servizi devono essere progettati in modo tale che anche gli utenti con disabilità possano avere la stessa esperienza utente di altri.
- **Credibile**. L'azienda e i suoi prodotti o servizi devono essere [4].

Figura 1.1: Le caratteristiche che un'applicazione deve possedere

Ogni applicazione sarà diversa in base all'equilibrio tra contesto, contenuto e utenti. Tuttavia, tenendo presenti tutti questi punti, è più semplice definire le priorità. Ciò è essenziale per aiutare le aziende a scomporre le attività al fine di formulare una strategia verso un obiettivo finale. Ad esempio, una ri-progettazione completa del sito Web è un'impresa enorme e può essere piuttosto costosa. Osservando le parti interessate a nido d'ape in grado di identificare le aree più importanti e iniziare il progetto eliminando le priorità di alto livello, consentendo così agli straordinari di ridefinire completamente l'esperienza dell'utente in meglio. Il nido d'ape aiuta anche a identificare tutte

le aree che sono importanti per una buona esperienza utente e che possono essere ulteriormente suddivise in modo più approfondito. È più importante che il servizio o prodotto sia reperibile piuttosto che desiderabile? Si ha bisogno di migliorare la credibilità nel tuo mercato? In breve, l'esperienza utente a nido d'ape è uno strumento utile sia per i progettisti che per le parti interessate. Descrivendo e definendo tutte le aree che sono importanti per le aziende di progettazione di UX, è possibile comprendere meglio cos'è la UX e perché è essenziale, aiutare a definire le priorità nella progettazione e servire come strumento per migliorare continuamente le aree di prodotti e servizi.

1.5.1 Progettazione della User Experience

Un'interazione richiede che l'utente utilizzi le capacità percettive, motorie e cognitive mentre osserva per manipolare e interpretare una applicazione. É per questo che gli UX Designer devono passare attraverso 7 passi per rendere l'esperienza utente più facile e intuitiva possibile.

- **Formazione dell'obiettivo**. L'obiettivo è ciò che l'utente sta cercando di raggiungere con l'interfaccia e quindi rappresenta la motivazione dell'utente per l'utilizzo dell'interfaccia (necessità, interesse, curiosità, ecc.). Gli obiettivi sono descritti come attività di "alto livello" e può includere esplorazione, analisi, sintesi e presentazione.

- **Creazione dell'intenzione**. L'intenzione è l'obiettivo specifico dell'uso dell'applicazione. Di conseguenza, le intenzioni sono descritte come attività di "basso livello": quello che l'applicazione deve fornire, quello che deve mostrare e gli obiettivi finali.

- **Specifica di un'azione**. L'utente deve quindi tradurre la propria intenzione di trovare una informazione specifica trovandola nell'interfaccia. L'interfaccia ha bisogno di aiutare l'utente segnalando all'utente come interagire con l'interfaccia, affinché l'utente specifichi quale operatore supporta al meglio la sua problematica prima di eseguire l'azione.

- **Esecuzione di un'azione**. L'utente deve eseguire l'azione specificata utilizzando l'interfaccia e i propri sistemi di input, come un dispositivo di puntamento (ad es. mouse, touchscreen), dispositivo di codifica (ad es. tastiera, tastiera) o altra modalità (ad es. riconoscimento gestuale o vocale). Una volta eseguita l'azione, il dispositivo di elaborazione elabora la richiesta e, in caso di successo, restituisce la risposta.

- **Percezione dello stato del sistema**. Una volta restituito, l'utente visualizza quindi la risposta. Qui, un forte feedback — o segnali all'utente

su ciò che è accaduto a seguito dell'interazione — è necessario per chiarire come è cambiata la schermata.

- **Interpretazione dello stato del sistema**. Dopo aver percepito la modifica della rappresentazione della applicazione attraverso feedback, l'utente deve quindi dare un senso all'aggiornamento. Un modo per descrivere questo stadio è completamento dell'intenzione: una volta restituita la risposta, può essere utilizzata per eseguire quella dell'utente attività di basso livello.

- **Valutazione del risultato**. La valutazione confronta l'intuizione con il risultato atteso con determinare se l'obiettivo è stato raggiunto. Ciò include una valutazione critica dell'intuizione (*"sembra giusto il risultato atteso?"*) e meta-valutazione dell'obiettivo generale (*"ho ottenuto la mia risposta?"*). Di seguito questa valutazione, l'utente può rivedere il proprio obiettivo e inizializzare un nuovo scambio di interazioni, riavviando la sequenza dei sette stadi [5, 6].

1.5.2 Testing dell'UX

Il test dell'UX è una tecnica utilizzata nella progettazione dell'interazione centrata sull'utente per valutare un prodotto testandolo sugli utenti. Questo può essere visto come una pratica di usabilità insostituibile, poiché fornisce un input diretto su come gli utenti reali utilizzano il sistema. Si concentra sulla intuitività del design del prodotto con gli utenti che non hanno alcuna esposizione precedente ad esso. Tale test è fondamentale per il successo di un prodotto finale in quanto un'applicazione completamente funzionante che crea confusione tra i suoi utenti non durerà a lungo. Ciò è in contrasto con i metodi di ispezione dell'usabilità in cui gli esperti utilizzano metodi diversi per valutare un'interfaccia utente senza coinvolgere gli utenti.

Il test di usabilità si concentra sulla misurazione della capacità di un prodotto creato dall'uomo di raggiungere lo scopo previsto. Esempi di prodotti che beneficiano comunemente dei test di usabilità sono alimenti, prodotti di consumo, siti Web o applicazioni Web, interfacce di computer, documenti e dispositivi. I test di usabilità misurano l'usabilità, o la facilità d'uso, di uno specifico oggetto o insieme di oggetti, mentre studi generali di interazione uomo-computer tentano di formulare principi universali.[7]

Guerilla Testing

Martin Belam ha definito i test di guerriglia come:

> "the art of pouncing on lone people in cafes and public spaces, and quickly filming them whilst they use a website for a couple of minutes."

Fondamentalmente, eseguire un guerilla test significa andare in un bar o in un altro luogo pubblico per chiedere alle persone di utilizzare il prototipo creato. È un test a basso costo e relativamente semplice che consente un feedback reale dell'utente. Questo tipo di test ha le seguenti caratteristiche:

- i partecipanti non sono assunti ma vengono avvicinati da persone che conducono sessioni di test;
- le sessioni stesse sono brevi (in genere 10-15 minuti) e sono strutturate attorno a particolari obiettivi di ricerca chiave;
- l'output è tipicamente qualitativo piuttosto che quantitativo.

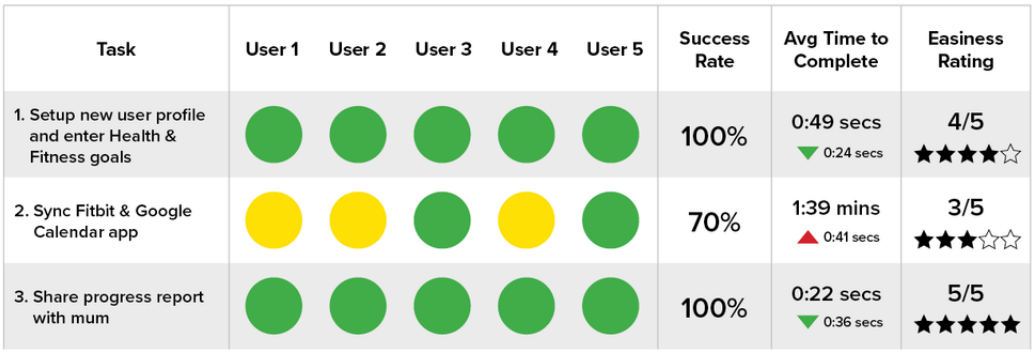

Figura 1.2: Esempio di Scheda di Valutazione del Guerilla Testing

I test aiutano a convalidare rapidamente quanto sia efficiente la progettazione sul pubblico previsto o se funzionalità specifiche funzionano nel modo previsto.I partecipanti al test vengono scelti in modo casuale. Viene richiesto di eseguire un rapido test di usabilità, spesso in cambio di un piccolo regalo. È un test a basso costo e relativamente semplice che consente un feedback reale dell'utente.

Quando usarlo

Il guerilla testing funziona al meglio nelle prime fasi del processo di sviluppo del prodotto. Quando si ha un design tangibile (wireframe o prototipi lo-fi) e quando si ha bisogno di sapere se ci si sta muovendo nella giusta direzione o meno.

I guerrilla test sono anche utili per raccogliere opinioni personali e impressioni emotive su idee e concetti.

È sempre importante capire che i partecipanti al test nei guerilla testing potrebbero non rappresentare il pubblico di destinazione del prodotto. Ecco perché i guerilla testing potrebbero non essere adatti per testare prodotti di nicchia che richiedono competenze speciali.

Cosa ricordare

Le attività selezionate per la sessione di test svolgono un ruolo fondamentale nel determinare se i risultati saranno utili o meno. Poiché è impossibile testare tutto in una volta, è necessario dare la priorità a tutti i possibili scenari di interazione e selezionare quello più probabile (flusso di utenti principali). È anche importante ricordare che si ha un tempo limitato per ogni sessione di test. Di solito, le persone che partecipano ai test di guerriglia daranno un massimo di 5-10 minuti del loro tempo.

Test di usabilità in laboratorio

Il test di usabilità in laboratorio è un test eseguito in ambienti speciali (laboratori) e supervisionato da un moderatore. Un moderatore è un professionista che sta cercando di ottenere feedback dagli utenti live. Durante un test moderato, i moderatori facilitano i partecipanti al test attraverso attività, rispondendo alle loro domande e rispondendo al loro feedback in tempo reale.

Quando usarlo

Il test di usabilità in laboratorio funziona meglio quando è necessario disporre di informazioni approfondite su come gli utenti reali interagiscono con il prodotto e su quali problemi devono affrontare. Aiuterà a indagare sul ragionamento alla base del comportamento dell'utente. Il fatto che questo test sia moderato consente di raccogliere più informazioni qualitative. Allo stesso tempo, i test di laboratorio possono essere costosi da organizzare ed eseguire perché è necessario proteggere un ambiente, assumere partecipanti ai test e un moderatore. Un altro problema con questo test è il numero di partecipanti al test in un singolo round. Di solito, si ha 5-10 partecipanti per ciclo di ricerca in un ambiente controllato. Pertanto, è importante garantire che tutti i partecipanti al test riflettano la tua base di clienti effettiva.

Cosa ricordare

Il test di usabilità in laboratorio richiede di avere un moderatore qualificato e un posto per eseguire un test. Ecco alcune cose da ricordare quando si sceglie un moderatore:

- Un moderatore dovrebbe essere sempre pronto ad aiutare i partecipanti al test a comprendere lo scopo del test (descrivere l'obiettivo) e mantenere il partecipante in pista se hanno qualche tipo di confusione. Tuttavia, ciò non significa che un moderatore dovrebbe dire ai partecipanti al test cosa dovrebbero fare.

- Il moderatore dovrebbe essere bravo a decodificare il linguaggio del corpo. Ricorda che ciò che dicono i partecipanti al test non è sempre lo stesso di quello che pensano. Ecco perché un moderatore dovrebbe essere bravo a osservare e analizzare il linguaggio del corpo e le espressioni facciali.
- L'intervista post-test è una parte essenziale di questo tipo di test. I moderatori raggiungono i partecipanti al test dopo la sessione di test e pongono loro alcune domande importanti.

Con i test di laboratorio, c'è sempre il rischio che l'ambiente controllato sia diverso dall'ambiente reale dell'utente. Posizionando l'utente in un'atmosfera controllata, c'è sempre il rischio di creare un comportamento dell'utente non realistico.

Test di usabilità remota non moderato

Il test di usabilità remota non moderato viene eseguito in remoto senza un moderatore. Offre risultati di test utente rapidi, robusti ed economici da utilizzare per ulteriori analisi. Ai partecipanti al test viene chiesto di completare le attività nel proprio ambiente utilizzando i propri dispositivi e senza un moderatore presente, che porta al naturale utilizzo del prodotto. Il costo dei test non moderati è inferiore; tuttavia, questo tipo di test offre risultati di test meno dettagliati.

Quando usarlo I test di usabilità remota non moderati funzionano al meglio quando è necessario ottenere un campione di grandi dimensioni per dimostrare i risultati critici della ricerca moderata iniziale. In altre parole, hai una particolare ipotesi che desideri convalidare su un ampio segmento dei tuoi utenti. I test di usabilità remota non moderati ti aiuteranno a testare una domanda specifica o osservare modelli di comportamento dell'utente.

Cosa ricordare I test di usabilità a distanza non vanno in profondità nel ragionamento di un partecipante al test. Ecco perché non è consigliabile utilizzare il test remoto non moderato come primo metodo di test di usabilità.

Session Recording

La session recording è un metodo per registrare le azioni che gli utenti reali (ma anonimi) eseguono mentre interagiscono con un sito. I dati della session recording aiutano a capire quali contenuti / funzionalità sono più interessanti per gli utenti (tramite analisi della mappa di calore) e quali problemi di interazione affrontano gli utenti mentre interagiscono con il prodotto.

Quando usarlo La session recording ti aiuterà a comprendere i principali problemi che gli utenti devono affrontare quando interagiscono con il tuo prodotto.

Cosa ricordare Per condurre una registrazione di sessione, è necessario utilizzare uno strumento speciale come HotJar a tale scopo.

La session recording funziona al meglio quando viene utilizzata insieme a un altro tipo di test di usabilità. Analizzando i risultati della session recording, si forma un'ipotesi sui problemi che gli utenti devono affrontare, ma spesso è necessario condurre un altro test per capire perché affrontano questo problema.[8]

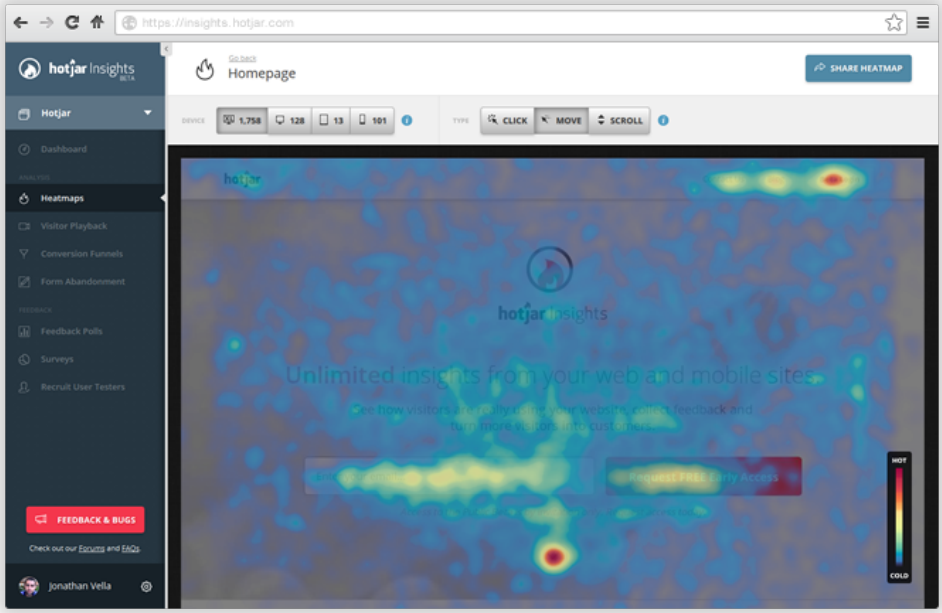

Figura 1.3: Esempio di Scheramta con Session Recording

1.6 Human–Computer Interaction

La Human–Computer IInteraction o HCI è lo studio di come le persone interagiscono con le macchine e in che misura le macchine sono o non sono sviluppate per un'interazione con gli esseri umani. Come suggerisce il nome, HCI è composto da tre parti: l'utente, il computer stesso e il modo in cui lavorano insieme.

Utente

Per "utente", si intende un singolo utente o un gruppo di utenti che lavorano insieme. È fondamentale apprezzare il modo in cui i sistemi sensoriali delle persone (vista, udito, tocco) trasmettono informazioni. Inoltre, diversi utenti

formano concezioni o modelli mentali diversi sulle loro interazioni e hanno modi diversi di apprendere e mantenere la conoscenza e. Inoltre, le differenze culturali e nazionali svolgono un ruolo.

Computer

Quando si parla di computer, ci si riferisce a qualsiasi tecnologia che spazia dai computer desktop ai sistemi di computer su larga scala. Ad esempio, se si discute della progettazione di un sito Web, il sito Web stesso verrebbe indicato come "il computer". Anche dispositivi come telefoni cellulari o videoregistratori possono essere considerati "computer".

Interazione

Ci sono ovvie differenze tra umani e macchine. Nonostante ciò, HCI cerca di assicurarsi che entrambi si mettano d'accordo e interagiscano con successo. Al fine di ottenere un sistema utilizzabile, è necessario applicare ciò che si sa sugli esseri umani e computer e consultare i probabili utenti d'interesse durante l'intero processo di progettazione. Nei sistemi reali, il programma e il budget sono importanti ed è fondamentale trovare un equilibrio tra ciò che sarebbe l'ideale per gli utenti e ciò che è fattibile nella realtà.

Capitolo 2

Le tecnologie disponibili

In questa prima parte verranno descritte più nello specifico tutte le tecnologie che sono state solo accennate nell'introduzione, per rendere più chiari gli obiettivi e le metodologie di lavoro impiegate nei capitoli successivi.

Il **Data Science** è un campo interdisciplinare che utilizza metodi, processi, algoritmi e sistemi scientifici per estrarre conoscenze e approfondimenti da molti dati strutturali e non strutturati. il data Science è legata al data mining, al deep learning e ai big data.

Il data science è un "concetto per unificare le statistiche, l'analisi dei dati, l'apprendimento automatico, la conoscenza del dominio e i relativi metodi" al fine di "comprendere e analizzare i fenomeni reali" con i dati. Utilizza tecniche e teorie tratte da molti campi nel contesto della matematica, della statistica, dell'informatica, della conoscenza del dominio e dell'informatica. Il vincitore del premio Turing Jim Gray ha immaginato la scienza dei dati come un "quarto paradigma" della scienza (empirica, teorica, computazionale e ora basata sui dati) e ha affermato che "tutto ciò che riguarda la scienza sta cambiando a causa dell'impatto della tecnologia dell'informazione" e del diluvio di dati[9]. Per quanto riguarda la programmazione di applicazioni dedicate alla data science i due linguaggi spiù utilizzati sono R e Phyton: entrambi molto potenti ma ognuno con i suoi vantaggi e svantaggi.

Figura 2.1: Python vs R

2.1 R

R è un linguaggio e un ambiente per l'elaborazione statistica e per la grafica. È un progetto GNU che è simile al linguaggio e all'ambiente S che è stato sviluppato nei Bell Laboratories (precedentemente AT&T, ora Lucent Technologies) da John Chambers e colleghi. R può essere considerata una diversa implementazione di S. Ci sono alcune differenze importanti, ma molto codice scritto per S viene eseguito inalterato su R.[10]

R offre un'ampia varietà di tecniche statistiche (modellistica lineare e non lineare, test statistici classici, analisi di serie temporali, classificazione, raggruppamento ecc.) e tecniche grafiche R è disponibile come software libero secondo i termini della GNU General Public License della Free Software Foundation sotto forma di codice sorgente. Si compila e funziona su una vasta gamma di piattaforme UNIX e sistemi simili (inclusi FreeBSD e Linux), Windows e MacOS. Di seguito sono indicati vari vantaggi del linguaggio R, che aiuteranno a comprendere meglio il concetto.

- **Open Source:** R è un linguaggio di programmazione open source. Ciò significa che chiunque può lavorare con R senza bisogno di una licenza. Inoltre, si può contribuire allo sviluppo di R personalizzando i pacchetti, sviluppandone di nuovi e risolvendo problemi;
- **Supporto esemplare per il wrangling dei dati:** R fornisce un supporto esemplare per il wrangling dei dati. I pacchetti come dplyr, readr sono in grado di trasformare i dati disordinati in una forma strutturata;
- **Vasta gamma di librerie:** R ha una vasta gamma di librerie. Con oltre 10.000 librerie nel repository CRAN, il numero è in costante aumento. Queste librerie si rivolgono a tutte le aree del settore.
- **Stampa e rappresentazione grafica dei dati:** R facilita la stampa e la rappresentazione grafica dei dati. Le librerie popolari come ggplot2 difendono con fermezza grafici estetici e visivamente accattivanti che distinguono R dagli altri linguaggi di programmazione.
- **Altamente compatibile:** R è altamente compatibile e può essere abbinato a molti altri linguaggi di programmazione come C, C ++, Java e Python. Può anche essere integrato con tecnologie come Hadoop e vari altri sistemi di gestione dei database.
- **Indipendente dalla piattaforma:** R è un linguaggio indipendente dalla piattaforma. È un linguaggio di programmazione multi piattaforma, il che significa che può essere eseguito abbastanza facilmente su Windows, Linux e Mac.

- **Report accattivanti:** Con pacchetti come Shiny e Markdown, riportare i risultati di un'analisi è estremamente semplice con R. Puoi creare report con i dati, i grafici e gli script R incorporati in essi. Puoi persino creare app Web interattive che consentono all'utente di giocare con i risultati e i dati.
- **Operazioni di apprendimento automatico:** R fornisce varie strutture per eseguire operazioni di apprendimento automatico come classificazione, regressione e fornisce anche funzionalità per lo sviluppo di reti neurali artificiali.
- **Statistiche:** R è ben nota come la lingua franca delle statistiche. Questo è il motivo principale per cui R è dominante tra gli altri linguaggi di programmazione per lo sviluppo di strumenti statistici.
- **In continua crescita:** R è un linguaggio di programmazione in continua evoluzione. È una tecnologia all'avanguardia che fornisce aggiornamenti ogni volta che viene aggiunta una nuova funzionalità.

Gli svantaggi della programmazione R sono invece:

- **Origine debole:** R condivide la sua origine con un linguaggio di programmazione molto più vecchio "S". Ciò significa che il pacchetto di base non supporta la grafica dinamica o 3D. Grazie alle librerie di R come Ggplot2 e Plotly, è possibile creare grafica dinamica, 3D e animata.
- **Gestione dei dati:** In R, la memoria fisica memorizza gli oggetti. Ciò è in contrasto con altre lingue come Python. Inoltre, R utilizza più memoria rispetto a Python e richiede tutti i dati in un unico posto, cioè nella memoria. Pertanto, non è un'opzione ideale quando si tratta di Big Data. Tuttavia, con i pacchetti di gestione dei dati e la possibile integrazione con Hadoop, questo è facilmente coperto.
- **Sicurezza di base:** R manca di una sicurezza di base. Questa funzione è una parte essenziale della maggior parte dei linguaggi di programmazione come Python. Per questo motivo, ci sono diverse restrizioni con R in quanto non può essere incorporato in un'applicazione web.
- **Linguaggio complicato:** R non è una lingua facile da imparare. Ha una ripida curva di apprendimento. Per questo motivo, le persone che non hanno precedenti esperienze di programmazione potrebbero avere difficoltà a imparare R.
- **Minore velocità:** I pacchetti R e il linguaggio di programmazione R sono molto più lenti di altri linguaggi come MATLAB e Python.

- **Diffusione in vari pacchetti:** Gli algoritmi in R sono distribuiti su diversi pacchetti. I programmatori senza una conoscenza preliminare dei pacchetti potrebbero avere difficoltà a implementare algoritmi.

Alcune delle importanti applicazioni di R Programming Language nel dominio di Data Science sono:

- **Finanza:** La scienza dei dati è ampiamente utilizzata nel settore finanziario. R è lo strumento più popolare in questo ambito. Questo perché R fornisce una suite statistica avanzata in grado di svolgere tutte le attività finanziarie necessarie. Con l'aiuto di R, gli istituti finanziari sono in grado di eseguire la misurazione del rischio al ribasso, regolare le prestazioni del rischio e utilizzare visualizzazioni come grafici a candele, grafici a densità, grafici a discesa, ecc. Le industrie finanziarie stanno anche sfruttando i processi statistici delle serie temporali di R, per modellare il movimento del loro mercato azionario e prevedere i prezzi delle azioni. R fornisce anche servizi per il mining di dati finanziari attraverso i suoi pacchetti come quantmod, pdfetch, TFX, pwt, ecc. Con l'aiuto di RShiny, si possono visualizzare i prodotti finanziari attraverso visualizzazioni vivide e coinvolgenti.

- **Attività bancarie:** Proprio come gli istituti finanziari, le industrie bancarie utilizzano la R per la modellizzazione del rischio del credito e altre forme di analisi del rischio. R viene anche utilizzato insieme a Hadoop per facilitare l'analisi della qualità, la segmentazione e la fidelizzazione dei clienti. Bank of America utilizza R per l'informativa finanziaria. Con l'aiuto di R, i data scientist della BOA sono in grado di analizzare le perdite finanziarie e di utilizzare gli strumenti di visualizzazione di R.

- **Assistenza sanitaria:** Genetica, Bioinformatica, Drug Discovery, Epidemiologia sono alcuni dei settori dell'assistenza sanitaria che fanno un uso intensivo di R. Con l'aiuto di R, queste aziende sono in grado di segmentare i dati ed elaborare le informazioni, fornendo uno sfondo essenziale per ulteriori analisi ed elaborazione dei dati. Per un trattamento più avanzato come la scoperta di farmaci, R è ampiamente utilizzato per eseguire studi preclinici e analizzare i dati sulla sicurezza dei farmaci. R è anche popolare per il suo pacchetto Bioconduttore che fornisce varie funzionalità per l'analisi dei dati genomici. R è anche usato per la modellistica statistica nel campo dell'epidemiologia, dove i data scientist analizzano e predicono la diffusione delle malattie.

- **Commercio elettronico:** Il settore dell'e-commerce è uno dei settori più importanti che utilizzano la scienza dei dati. R è uno degli strumenti

standard utilizzati nell'e-commerce. Dal momento che queste società basate su Internet hanno a che fare con varie forme di dati, strutturati e non strutturati, nonché da varie fonti di dati come fogli di calcolo e database (SQL e NoSQL), R si rivela una scelta efficace per questi settori. Le aziende di e-commerce utilizzano R per analizzare i prodotti di cross-selling ai propri clienti. Nel cross-selling, si suggerisce al cliente ulteriori prodotti che completano il loro acquisto originale. Questi tipi di suggerimenti e raccomandazioni vengono analizzati al meglio con l'aiuto di R.[11][12]

2.1.1 R o Python: le motivazioni della scelta

Per un numero crescente di persone, la data science è una parte centrale del loro lavoro. L'aumentata disponibilità dei dati, il calcolo più potente e l'enfasi posta sulle decisioni basate sull'analisi nel mondo degli affari lo hanno reso un periodo d'oro per la scienza dei dati. I due strumenti di programmazione più popolari per il lavoro di data science sono Python e R al momento. È difficile sceglierne uno tra i due linguaggi di analisi dei dati incredibilmente flessibili. Entrambi sono gratuiti e open source e sono stati sviluppati nei primi anni '90 - R per analisi statistiche e Python come linguaggio di programmazione generico. Per chiunque sia interessato all'apprendimento automatico, lavorando con set di dati di grandi dimensioni o creando visualizzazioni di dati complessi, sono assolutamente essenziali.

Una breve panoramica della storia di Python e R

Python è stato rilasciato nel 1989 con una filosofia che enfatizza la leggibilità e l'efficienza del codice. È un linguaggio di programmazione orientato agli oggetti, il che significa che raggruppa dati e codice in oggetti che possono interagire e modificarsi l'un l'altro. Java, C ++ e Scala sono altri esempi. Questo approccio sofisticato consente ai data scientist di eseguire attività con maggiore stabilità, modularità e leggibilità del codice. La scienza dei dati è solo una piccola parte del diverso ecosistema Python. La suite di Python di librerie specializzate di deep learning e altre macchine di apprendimento automatico include strumenti popolari come scikit-learn, Keras e TensorFlow, che consentono ai data scientist di sviluppare sofisticati modelli di dati che si collegano direttamente a un sistema di produzione.

R è stato sviluppato nel 1992 ed è stato il linguaggio di programmazione preferito dalla maggior parte dei data scientist per anni. È un linguaggio procedurale che funziona suddividendo un'attività di programmazione in una serie di passaggi, procedure e subroutine. Questo è un vantaggio quando si

tratta di costruire modelli di dati perché rende relativamente facile capire come vengono eseguite operazioni complesse; tuttavia, è spesso a scapito delle prestazioni e della leggibilità del codice. La comunità orientata all'analisi di R ha sviluppato pacchetti open source per specifici modelli complessi che uno scienziato di dati dovrebbe altrimenti costruire da zero. R enfatizza inoltre i report di qualità con il supporto di visualizzazioni pulite e framework per la creazione di applicazioni web interattive. D'altro canto, prestazioni più lente e mancanza di funzionalità chiave come test unitari e framework Web sono motivi comuni che alcuni data scientist preferiscono cercare altrove.

Principali differenze tra i due linguaggi

R e Python sono due dei linguaggi di programmazione più popolari nel dominio analitico e sono considerati contendenti da molti analisti e scienziati di dati. Hanno punti in comune come quello di essere entrambi gratuiti, sono supportati da community molto attive e offrono potenti strumenti e librerie open source. Per quanto meravigliose siano queste somiglianze, il fatto che sia R sia Python abbiano tutti e tre i punti in comune può spesso rendere difficile la scelta l'una sull'altra. Quindi, non sono solo le capacità di un programma che influenzano la preferenza di R o Python, è anche il contesto in cui viene utilizzato. La forza di R sta nei modelli statistici e grafici e vede una maggiore adozione da accademici, data scientist e statistici . Considerando anche che, Python, si concentra maggiormente sulla produttività e la leggibilità del codice, è popolare tra sviluppatori, ingegneri e programmatori. Come linguaggio generico Python è ampiamente usato in molti campi, incluso lo sviluppo web. Sta inoltre guadagnando popolarità tra gli investimenti bancari e gli hedge fund ed è implementato dalle banche per piattaforme di pricing, gestione del rischio e gestione commerciale. Tuttavia, sorprendentemente, a differenza di R, conoscere Python non è ancora un requisito comune per i talenti tecnologici che lavorano nella maggior parte dei settori dei servizi finanziari. Quindi, nel dibattito tra Python e R, i data scientist con un background di ingegneria del software pesante potrebbero preferire Python, mentre gli statistici possono fare affidamento maggiormente su R. Di seguito vengono mostrati le principali differenze dei due linguaggi di programmazione.

- **Usabilità:** Python ha acquisito una risposta positiva dai data scientist coinvolti nell'apprendimento automatico. Poiché la curva di apprendimento è bassa per i suoi utenti, la vera forza di Python risiede nella sua semplicità, leggibilità senza pari e flessibilità, il tutto alimentato da una sintassi precisa ed efficiente. Essendo un linguaggio di programmazione completo, Python è ideale per l'implementazione di algoritmi per l'uso in produzione e per l'integrazione di app Web in attività di analisi dei

dati. D'altra parte, R è ottimo per il lavoro esplorativo ed è adatto per complesse analisi statistiche, a causa del suo numero crescente di pacchetti. Ma lo svantaggio per i principianti di R è che R ha una ripida curva di apprendimento e spesso rende difficile la ricerca di pacchetti. Ciò può prolungare il processo di analisi dei dati e causare ritardi nell'attuazione. Sebbene R sia un ottimo strumento, è limitato in termini di ciò che può realizzare oltre all'analisi dei dati. Molte delle librerie utente in R sono scritte male e spesso considerate lente, il che può essere un problema per gli utenti.

- **Librerie e Pacchetti:** Python ha ampie librerie che riducono significativamente l'intervallo di tempo tra l'inizio del progetto e risultati significativi. Il repository di software per il linguaggio di programmazione Python è così ricco che Python Package Index (PyPI) comprende attualmente 130.641 pacchetti. La libreria ha una varietà di ambienti per testare e confrontare algoritmi di machine learning. I pacchetti offrono soluzioni non solo intuitive ma anche flessibili. Un buon esempio è PyBrain, una libreria modulare di apprendimento automatico che offre potenti algoritmi per le attività di apprendimento automatico. Considerata una popolare libreria di apprendimento automatico, Scikit-learning offre strumenti di data mining per rafforzare l'usabilità di apprendimento automatico esistente di Python. In confronto, CRAN (Comprehensive R Archive Network) rimane un enorme repository con 10.000 pacchetti che possono essere facilmente installati in R. Gli utenti attivi contribuiscono quotidianamente al crescente repository e molte delle funzionalità di R (come il calcolo statistico, la visualizzazione dei dati) non hanno eguali. Mentre la curva di apprendimento per i principianti è ripida, una volta che un utente conosce le basi, diventa molto più veloce apprendere tecniche avanzate. Per molti statistici, l'implementazione e la documentazione in R sono più accessibili che in Python.

  Ma i pacchetti appena installati sia in Python che in R stanno alleviando i punti deboli di cui ciascuno soffre. Ad esempio, Altair per Python e dplyr per R supportano il flusso tradizionale di visualizzazione e wrangling dei dati.

- **Data Visualization:** La visualizzazione dei dati è parte integrante dell'analisi dei dati e può semplificare le informazioni complesse identificando modelli e correlazioni. I pacchetti di visualizzazione di R includono ggplot2, ggvis, googleVis e rCharts. Le visualizzazioni tramite R possono rendere in modo efficiente ed efficace il set di dati grezzi più complesso dall'aspetto informativo e gradevole alla vista. Rispetto a R, Python

ha un'enorme quantità di opzioni interattive come Geoplotlib e Bokeh e scegliere il migliore e il più rilevante a volte può diventare estenuante e complesso. La visualizzazione dei dati viene fornita meglio tramite R ed anche meno complicata.

Finora, Python è considerato uno sfidante per R e rimane più popolare grazie alla sua ampia usabilità e perché può implementare il codice di produzione. Ma per essere onesti, sia R che Python hanno i loro pro e contro e la decisione di implementare quello giusto dipende principalmente dal tipo di set di dati che si sta osservando e dal problema che bisogna risolvere. Quindi si è deciso di utilizzare R in primo luogo per lo studio dei dati ma anche perché l'applicazione sottostante è stata scritta principalmente in R per cui risulterà molto più facile effettuare il collegamento.

2.1.2 R Shiny

Shiny è un pacchetto R che consente di creare app Web interattive utilizzando sia il potere statistico di R sia l'interattività del web moderno. Un'alternativa eccellente ed efficiente ai fogli di calcolo e alle visualizzazioni stampate, R Shiny consente di risparmiare spazio e tempo nella costruzione, automazione e distribuzione di visualizzazioni di dati e analisi statistiche. Sebbene ciò possa sembrare spaventoso a causa delle parole "pagine Web", è orientato agli utenti R che hanno 0 esperienza con lo sviluppo Web e non è necessario per conoscere qualsiasi HTML / CSS / JavaScript. Shiny è un potentissimo pacchetto: si deve pensare come a un modo semplice per creare una pagina web interattiva e quella pagina web può interagire perfettamente con R e visualizzare oggetti R (grafici, tabelle, di qualsiasi altra cosa tu faccia in R). [13]

Ogni app Shiny è composta da due parti: una pagina Web che mostra l'app all'utente e un server che alimenta l'app. Il server che esegue l'app può essere un pc (ad esempio quando si esegue un'app da RStudio in locale) o un server da qualche altra parte. Nella terminologia Shiny, sono chiamati UI (interfaccia utente) e server. L'interfaccia utente è solo un documento web che l'utente può vedere, è l'HTML che si scrive usando le funzioni di Shiny. L'interfaccia utente è responsabile della creazione del layout dell'app e del comunicare a Shiny esattamente cosa sta succendendo. Il server è responsabile della logica dell'app; è l'insieme di istruzioni che comunicano alla pagina web cosa mostrare quando l'utente interagisce con la pagina.[14]

Grazie a ciò è facile modellare l'app come un MVC. Model-View-Controller (MVC) è un pattern utilizzato in programmazione per dividere il codice in blocchi dalle funzionalità ben distinte. Un client, tipicamente un browser, inoltra la richiesta ad un server per una pagina HTML. Il server ospita un'applicazione

scritta in un linguaggio di programmazione lato server che preleva i dati da un database, li elabora e li restituisce al client in formato HTML.

La parte più "attiva" in questo procedimento è l'applicazione Web che ha il compito di reperire ed inviare le informazioni. Il pattern MVC è quindi composto da:

- **model**. Contiene i metodi di accesso ai dati.
- **view**. Si occupa di visualizzare i dati all'utente e gestisce l'interazione fra quest'ultimo e l'infrastruttura sottostante.
- **controller**. Riceve i comandi dell'utente attraverso il View e reagisce eseguendo delle operazioni che possono interessare il Model e che portano generalmente ad un cambiamento di stato del View.

Attraverso R Shiny si avrà la UI che controllerà la View, mentre il Server attraverso gli observer controllerà la parte del Controller e attraverso le funzioni controllerà il Model.[15]

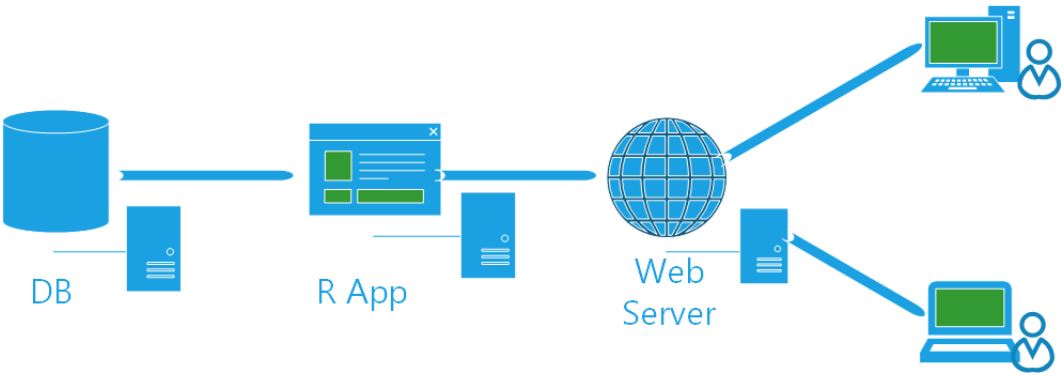

Figura 2.2: MVC in R Shiny

Shiny utilizza per la visualizzazione web Bootstrap. Bootstrap è una raccolta di strumenti liberi per la creazione di siti e applicazioni per il Web. Essa contiene modelli di progettazione basati su HTML e CSS, sia per la tipografia, che per le varie componenti dell'interfaccia, come moduli, pulsanti e navigazione, così come alcune estensioni opzionali di JavaScript.

2.1.3 Package Utilizzati

In R, il pilastro fondamentale del codice condivisibile è il package. Un pacchetto raggruppa codice, dati, documentazione e test ed è facile da condividere con altri. A gennaio 2015, c'erano oltre 6.000 pacchetti disponibili sulla rete

completa R Archive comunemente chiamato **CRAN** (The Comprehensive R Archive Network).[16]

Questa enorme varietà di pacchetti è uno dei motivi per cui R ha così tanto successo: è probabile che qualcuno abbia già risolto un problema su cui ci si sta lavorando da tempo e che si possa trarre vantaggio dal lavoro scaricando semplicemente il pacchetto. Da CRAN è possibile ottenere l'ultima versione ufficiale di R, istantanee giornaliere di R, file tar compressi, una grande quantità di codice aggiuntivo fornito e binari precompilati per vari sistemi operativi (Linux, Mac OS Classic, macOS e MS Windows). CRAN fornisce anche accesso alla documentazione su R, mailing list esistenti e sistema di tracciamento bug R. Il sito principale di CRAN presso WU (Wirtschaftsuniversität Wien) in Austria è disponibile all'indirizzo URL `https://CRAN.R-project.org/` ed è replicato quotidianamente su molti siti in tutto il mondo (`https://CRAN.R-project.org/mirrors.htm`). Per installare un package è veramente facile e questo rende ancora più accessibile a tutti questi determinati sorgenti di codice:

```
> install.packages("nomelibreria")
> library(nomelibreria)
```

La maggior parte di questi pacchetti hanno anche una documentazione descritta nei siti dedicati oppure attraverso questo link `https://rdrr.io/`.

shinydashboard

shinydashboard (`https://cran.r-project.org/web/packages/shinydashboard/index.html`) è un pacchetto R il cui compito è semplificare (come suggerisce il nome) la creazione di dashboard su Shiny. La parte dell'interfaccia utente di un'app Shiny creata con shinydashboard ha 3 elementi di base racchiusi all'interno di una *dashboardPage()*:

```
## app.R ##
library(shiny)
library(shinydashboard)
ui <- dashboardPage(
  dashboardHeader(),
  dashboardSidebar(),
  dashboardBody()
)
server <- function(input, output) { }
shinyApp(ui, server)
```

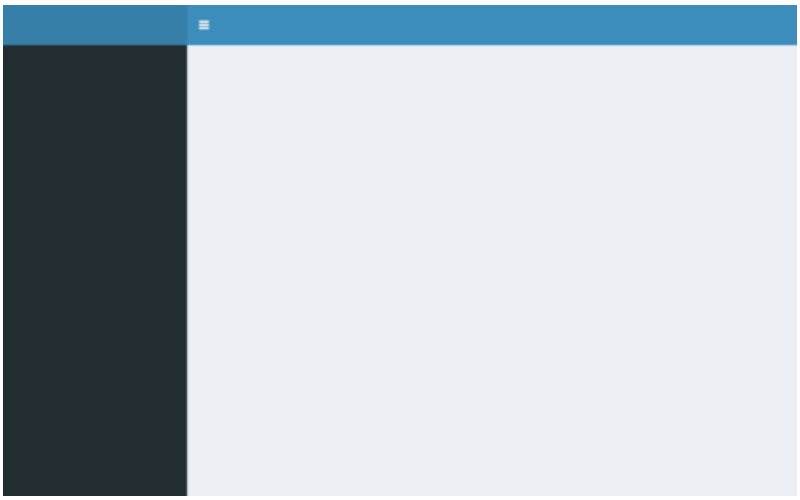

Figura 2.3: Esempio di dashboard creata con shynidashboard

Ogni posizione della dashboard poi è personalizzabile attraverso funzioni già pre impostate nel pacchetto ma anche utilizzando grafici e funzioni racchiuse all'interno di altri pacchetti. [17]

shinyjs

shinyjs è un pacchetto R che consente di eseguire operazioni JavaScript utili in applicazioni Shiny senza dover conoscere in modo specifico JavaScript: questo pacchetto consente di migliorare facilmente l'interazione dell'utente e l'esperienza utente nelle app Shiny. Può anche essere utilizzato per eseguire facilmente le proprie funzioni JavaScript personalizzate da R. Esempi di utilizzo di shinyjs:

```
observe({
  if (is.null(input$name) || input$name == "") {
    shinyjs::disable("submit")
  } else {
    shinyjs::enable("submit")
  }
})

observe({
  shinyjs::toggleState("submit", !is.null(input$name) && input$name
      != "")
})
```

plotly

Plotly è uno dei migliori strumenti di visualizzazione dei dati disponibili basato sulla libreria di visualizzazione D3.js, HTML e CSS. È stato creato usando Python e il framework Django; Attraverso questo pacchetto si può utilizzare per creare visualizzazioni interattive di dati online. È compatibile con una serie di lingue/strumenti: R, Python, MATLAB, Perl, Julia, Arduino.[18]

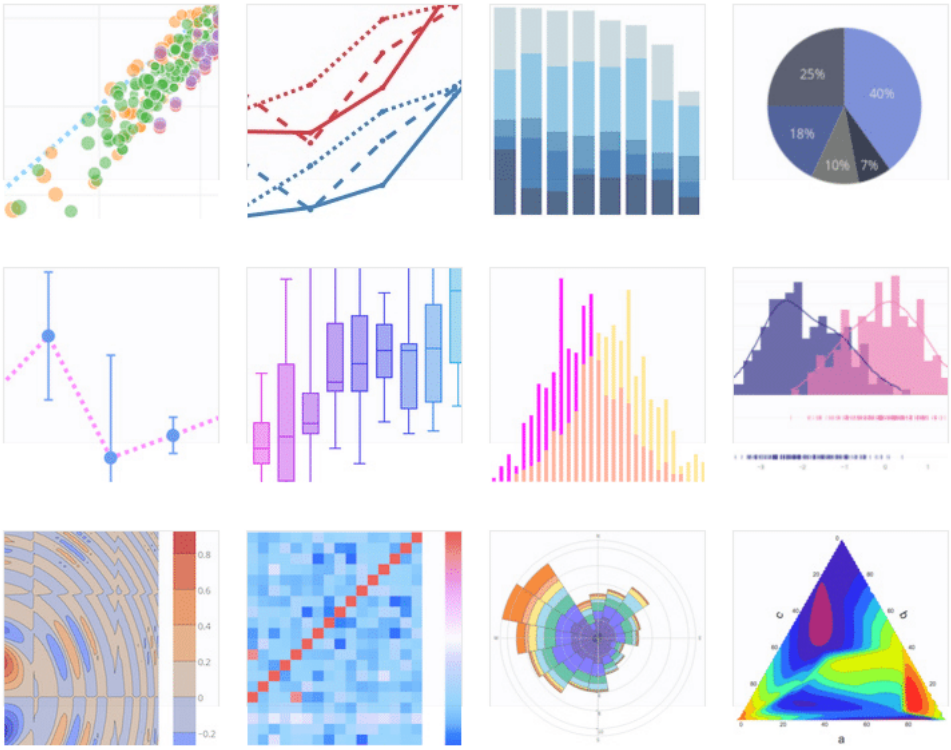

Figura 2.4: Esempi di grafici creati da Plotly

Plotly è uno strumento potente e vantaggioso ma anche con qualche limite:

- i grafici realizzati utilizzando la versione plotly community sono sempre pubbliche e possono essere visualizzate da chiunque.
- per una versione plotly community, esiste un limite massimo per le chiamate API al giorno.
- esiste anche un numero limitato di tavolozze di colori disponibili nella versione community.

Mentre i vantaggi sono:

- consente di creare visualizzazioni interattive create utilizzando D3.js senza nemmeno dover conoscere D3.js. Fornisce compatibilità con un numero di linguaggi/strumenti diversi come R, Python, MATLAB, Perl, Julia, Arduino.

- l'utilizzo di grafici interattivi può essere facilmente condiviso online.

- plotly può anche essere utilizzato da persone senza background tecnico per la creazione di grafici interattivi caricando i dati e utilizzando la GUI plotly.

- plotly è compatibile con ggplots in R e Python.

- permette di incorporare grafici interattivi in progetti o siti Web usando iframe o html.

ggplot2

Il pacchetto ggplot2 offre un linguaggio grafico potente per la creazione di grafici eleganti e complessi. La sua popolarità nella comunità R è esplosa negli ultimi anni. Basato originariamente su *The Grammar of Graphics* di Leland Wilkinson, ggplot2 consente di creare grafici che rappresentano dati numerici e categorici sia univariati che multivariati in modo semplice. Il raggruppamento può essere rappresentato da colore, simbolo, dimensione e trasparenza. La creazione di grafici a traliccio (cioè il condizionamento) è relativamente semplice.[19] Padroneggiare il linguaggio ggplot2 può essere impegnativo: esiste una funzione di supporto chiamata qplot() che può nascondere gran parte di questa complessità quando si creano grafici standard.

Contrariamente alla grafica R di base, ggplot2 consente all'utente di aggiungere, rimuovere o alterare componenti in un grafico ad alto livello di astrazione. Questa astrazione ha un costo, con ggplot2 più lento della grafica reticolare. I grafici possono essere creati tramite la funzione qplot() in cui argomenti e valori predefiniti devono essere simili alla funzione plot() di base R. Una capacità di stampa più complessa è disponibile tramite ggplot() che espone l'utente a elementi più espliciti della grammatica.

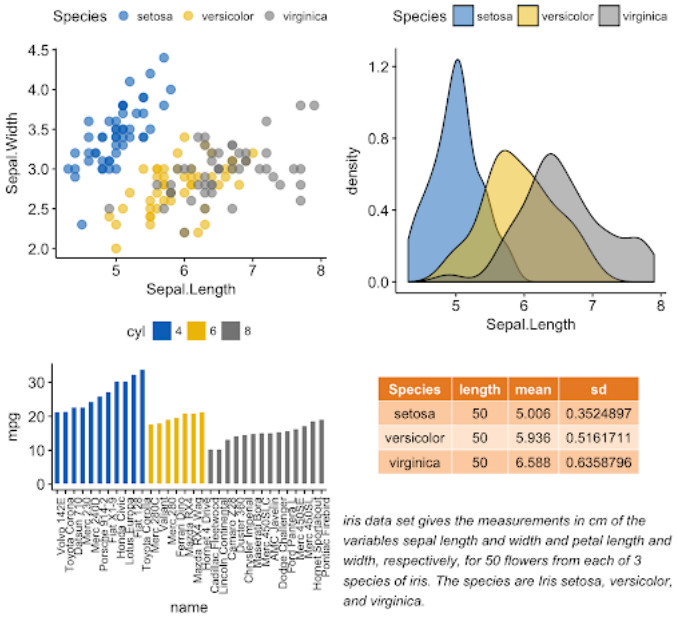

Figura 2.5: Esempi di grafici creati da ggplot2

2.1.4 Pubblicazione dell'applicazione

Grazie ad R Shiny è possibile in modo veloce e sicuro pubblicare le applicazioni su Web. Quando si tratta di condividere app Shiny, si ha due opzioni di base:

- **Condividere l'app Shiny come script R.** Questo è il modo più semplice per condividere un'app, ma funziona solo se gli utenti hanno R sul proprio computer (e sanno come usarlo): gli utenti possono utilizzare questi script per avviare l'app dalla propria sessione R.
- **Condividere l'app Shiny come una pagina web.** Questo è sicuramente il modo più intuitivo per condividere un'app Shiny. Gli utenti possono navigare nell'app tramite Internet con un browser web; Troveranno l'app completamente renderizzata, aggiornata e pronta per l'uso.

Condividere l'app Shiny come script R

Chiunque abbia R può eseguire l'app Shiny: avrà bisogno di una copia del file app.R, oltre a qualsiasi materiale supplementare utilizzato nell'app (ad esempio, cartelle www o file helper.R). L'utente può inserire i file in una directory dell'app nella propria directory di lavoro e si può avviare l'app in R con gli stessi comandi del pc creatore.

Shiny ha tre comandi incorporati che semplificano l'uso dei file ospitati online: runUrl, runGitHub e runGist.

runUrl scaricherà e avvierà un'app Shiny direttamente da un collegamento web. Per utilizzare runURL:

- Salvare la directory dell'app Shiny come file zip
- Ospitare il file zip su un link in una pagina web. Chiunque abbia accesso al collegamento può avviare l'app dall'interno di R eseguendo:

```
library(shiny)
runUrl( "<the weblink>")
```

Se non si dispone di una pagina web in cui ospitare i file, si può ospitare i file gratuitamente su www.github.com. GitHub è un popolare sito di hosting di progetti per sviluppatori R poiché non si limita a ospitare file. GitHub fornisce molte funzionalità per supportare la collaborazione, come tracciatori di problemi, wiki e una stretta integrazione con il sistema di controllo della versione git. Per utilizzare GitHub, dovrai registrarti (è gratuito) e scegliere un nome utente. Per condividere un'app tramite GitHub, crea un repository del progetto su GitHub. Quindi si archivia il file app.R nel repository, insieme a tutti i file supplementari utilizzati dall'app.

```
runGitHub( "<your repository name>", "<your user name>")
```

Se si desidera utilizzare un modo anonimo per pubblicare file online, GitHub offre un servizio di montaggio per la condivisione di file su gist.github.com. Non è necessario registrarsi per un account GitHub per utilizzare questo servizio. Anche se si ha un account GitHub, l'essenziale può essere un modo semplice e veloce per condividere progetti Shiny.

Dopo aver creato un gist, gli utenti possono avviare l'app con *runGist* ("<gist number>") dove "<gist number>" è il numero che appare alla fine dell'indirizzo web del tuo Gist.

```
runGist("eb3470beb1c0252bd0289cbc89bcf36f")
```

Condividere l'app Shiny come una pagina web.

Tutti i metodi precedenti condividono la stessa limitazione: richiedono che l'utente abbia installato R e Shiny sul proprio computer. Tuttavia, Shiny crea l'opportunità perfetta per condividere l'output con persone che non hanno R (e non hanno intenzione di ottenerlo). L'app Shiny sembra essere uno degli

strumenti di comunicazione più utilizzati al mondo: una pagina web. Se si ospita l'app su un proprio indirizzo URL, gli utenti possono visitare l'app (e non devono preoccuparsi del codice che la genera).

Se si ha familiarità con l'hosting web o si ha accesso a un reparto IT, è possibile ospitare le proprie app Shiny su un server; se invece si preferisce un'esperienza più semplice o si ha bisogno di supporto, RStudio offre tre modi per ospitare l'app Shiny come pagina web:

- shinyapps.io. Il modo più semplice per trasformare l'app Shiny in una pagina web è utilizzare shinyapps.io, il servizio di hosting di RStudio per le app Shiny. shinyapps.io consente di caricare l'app direttamente dalla sessione R su un server ospitato da RStudio. Si ha il controllo completo sull'app, inclusi gli strumenti di amministrazione del server.

- Shiny Server. Shiny Server è un programma complementare a Shiny che crea un server web progettato per ospitare app Shiny. È gratuito, open source e disponibile su GitHub. Shiny Server è un programma server che i server Linux possono eseguire per ospitare un'app Shiny come pagina web. Per utilizzare Shiny Server, si ha bisogno di un server Linux con supporto esplicito per Ubuntu 12.04 o versione successiva (64 bit) e CentOS / RHEL 5 (64 bit). È possibile ospitare più applicazioni Shiny su più pagine Web con lo stesso Shiny Server e distribuire le app attraverso un firewall.

- RStudio Connect. Se si utilizza Shiny in un contesto a scopo di lucro, potrebbe tornare utile degli strumenti server forniti con la maggior parte dei programmi server a pagamento, come: Autenticazione con password, Supporto SSL, Strumenti dell'amministratore e Supporto prioritario. In tal caso viene in contro RStudio Connect, una piattaforma di pubblicazione per il lavoro che i team creano nelle applicazioni R. Share Shiny, report R Markdown, dashboard, grafici, notebook Jupyter e altro in un unico comodo posto. Con RStudio Connect, si può pubblicare dall'IDE RStudio con la semplice pressione di un pulsante e pianificare l'esecuzione di report e policy di sicurezza flessibili.

2.2 HTML

Hypertext Markup Language (HTML) è il linguaggio di markup standard per i documenti progettati per essere visualizzati in un browser web. Può essere assistito da tecnologie come Cascading Style Sheets (CSS) e linguaggi di scripting come JavaScript, ASP e PHP.

I browser Web ricevono documenti HTML da un server locale o Web o dalla memoria locale e li trasformano in pagine Web multimediali. L'HTML descrive semanticamente la struttura di una pagina Web e in origine includeva spunti per l'aspetto del documento.

Gli elementi HTML sono i "mattoni" delle pagine HTML. Con i costrutti HTML, le immagini e altri oggetti come i moduli interattivi possono essere incorporati nella pagina renderizzata. L'HTML fornisce un mezzo per creare documenti strutturati indicando la semantica strutturale per testo come titoli, paragrafi, elenchi, collegamenti, citazioni e altri elementi. Gli elementi HTML sono delineati da tag, scritti usando parentesi angolari. Tag come <img /> e <input /> introducono direttamente il contenuto nella pagina. Altri tag come <p> circondano e forniscono informazioni sul testo del documento e possono includere altri tag come elementi secondari. I browser non visualizzano i tag HTML, ma li usano per interpretare il contenuto della pagina.

L'HTML può incorporare programmi scritti in un linguaggio di scripting come JavaScript, che influisce sul comportamento e sul contenuto delle pagine Web. L'inclusione dei CSS definisce l'aspetto e il layout dei contenuti. Il World Wide Web Consortium (W3C), ex manutentore dell'HTML e attuale manutentore degli standard CSS, ha incoraggiato l'uso di CSS su HTML presentazionale esplicito dal 1997.

2.2.1 Elementi dell'HTML

Un elemento HTML è un singolo componente di un documento HTML. Rappresenta la semantica o il significato. Ad esempio, l'elemento title rappresenta il titolo del documento.

La maggior parte degli elementi HTML sono scritti con un tag di inizio (o tag di apertura) e un tag di fine (o tag di chiusura), con contenuto in mezzo. Gli elementi possono anche contenere attributi che ne definiscono le proprietà aggiuntive. Ad esempio, un paragrafo, che è rappresentato dall'elemento p, sarebbe scritto come:

Figura 2.6: Caratteristiche di un tag HTML

In HTML, i nomi dei tag e degli attributi non fanno distinzione tra maiuscole e minuscole (ma la maggior parte dei valori degli attributi fa distinzione tra maiuscole e minuscole). Significa il tag <P> e il tag <p> definisce la stessa cosa in HTML che è un paragrafo.

```
<p>Paragrafo 1</p>
<P>Paragrago 2</P>
```

Gli empty elements (chiamati anche self-closing o void elements) non sono tag contenitore - ciò significa che non è possibile scrivere <hr> alcuni contenuti </hr> o
 alcuni contenuti </br>.

Un tipico esempio di elemento vuoto è l'elemento
, che rappresenta un'interruzione di riga. Alcuni altri elementi vuoti comuni sono <img>, <input>, <link>, <meta>, <hr>, ecc.

```
<p>Questo paragrafo contiene <br> una interruzione di riga.</p>
<img src="../cielo.jpg" alt="Cielo">
<input type="text" name="nome" value="Matteo Sertori">
```

Gli elementi possono essere posizionati in due gruppi distinti: i block level e l'inline level.

I **block level** sono quegli elementi che sono formattati visivamente come blocchi (ovvero occupa l'intera larghezza disponibile), con un'interruzione di riga prima e dopo l'elemento. Ad esempio, elemento di paragrafo (<p>), intestazioni (da <h1> a <h6>), divisioni ( ) ecc. In generale, i block level possono contenere elementi inline e altri block level. I block level all'interno dell'HTML possono essere:

```
<address> <article> <aside> <blockquote> <canvas> <dd> <div>
<dl> <dt> <fieldset> <figcaption> <figure> <footer> <form>
<h1>-<h6> <header> <hr> <li> <main> <nav> <noscript> <ol>
<p> <pre> <section> <table> <tfoot> <ul> <video>
```

I **inline level** sono quei elementi del documento di origine che non formano nuovi blocchi di contenuto; il contenuto è distribuito in righe. Ad esempio, parti di testo enfatizzate all'interno di un paragrafo (), span (), elemento forte () ecc. Gli inline leve in genere possono contenere solo testo e altri elementi incorporati. Gli inline level all'interno dell'HTML possono essere:

```
<a> <abbr> <acronym> <b> <bdo> <big> <br> <button> <cite>
<code> <dfn> <em> <i> <img> <input> <kbd> <label> <map>
<object> <output> <q> <samp> <script> <select> <small> <span>
<strong> <sub> <sup> <textarea> <time>
```

```
<tt>   <var>
```

2.2.2 Attributi HTML

Gli attributi definiscono caratteristiche o proprietà aggiuntive dell'elemento come larghezza e altezza di un'immagine. Gli attributi sono sempre specificati nel tag iniziale (o tag di apertura) e di solito è costituito da coppie nome / valore come nome = "valore". I valori degli attributi devono sempre essere racchiusi tra virgolette.

Inoltre, alcuni attributi sono richiesti per determinati elementi. Ad esempio, un tag <img> deve contenere gli attributi src e alt. Diamo un'occhiata ad alcuni esempi di utilizzo degli attributi:

```
<a href="https://www.google.com/" title="Cerca su Google">Google</a>
<img src="images/sorriso.png" width="45" height="54" alt="Sorriso">
<input type="text" value="Matteo Sertori">
```

General Purpose Attributes

Ci sono degli attributi come id, title, class, style che sono comuni per la maggior parte degli elementi. Di seguito vengono descritti i loro usi:

- **ID:** L'attributo id viene utilizzato per assegnare un nome o identificatore univoco a un elemento all'interno di un documento. Ciò semplifica la selezione dell'elemento mediante CSS o JavaScript.

  ```
  <div id="container">
      <h2 id="titolo">Titolo</h2>
      <p id="paragrafo">Paragrafo</p>
  </div>
  ```

  L'id di un elemento deve essere univoco in un singolo documento. Non è possibile nominare due elementi nello stesso documento con lo stesso ID e ogni elemento può avere un solo ID.

- **class**: Come l'attributo id, l'attributo class viene utilizzato anche per identificare gli elementi. Ma a differenza di id, l'attributo class non deve essere univoco nel documento. Ciò significa che puoi applicare la stessa classe a più elementi in un documento, come mostrato nell'esempio seguente:

```
<div id="container">
    <h2 id="titolo" class="testo-bianco">Titolo</h2>
    <p id="paragrafo" class="testo-bianco">Paragrafo</p>
</div>
```

Poiché una classe può essere applicata a più elementi, qualsiasi regola di stile scritta in quella classe verrà applicata a tutti gli elementi che hanno quella classe

- **style**: L'attributo style consente di specificare regole di stile CSS come colore, font, bordo, ecc. Direttamente all'interno dell'elemento.

```
<div id="container">
    <h2 id="titolo" class="testo-bianco"
        style="background-color:red;">Titolo</h2>
    <p id="paragrafo" class="testo-bianco"
        style="background-color:green;">Paragrafo</p>
</div>
```

2.3 CSS

Cascading Style Sheets (CSS) è un linguaggio di fogli di stile utilizzato per descrivere la presentazione di un documento scritto in un linguaggio di markup come HTML. Il CSS è una tecnologia fondamentale del World Wide Web, insieme a HTML e JavaScript.

CSS è progettato per consentire la separazione della presentazione e del contenuto, inclusi layout, colori e caratteri. Questa separazione può migliorare l'accessibilità del contenuto, fornire maggiore flessibilità e controllo nella specifica delle caratteristiche di presentazione, consentire a più pagine Web di condividere la formattazione specificando il CSS pertinente in un file .css separato e ridurre la complessità e la ripetizione del contenuto strutturale.

La separazione di formattazione e contenuto rende anche possibile presentare la stessa pagina di markup in diversi stili per diversi metodi di rendering, come su schermo, in stampa, con la voce (tramite browser vocale o screen reader) e su Braille dispositivi tattili. CSS ha anche regole per la formattazione alternativa se si accede al contenuto su un dispositivo mobile.

Il nome "a cascata" deriva dallo schema di priorità specificato per determinare quale regola di stile si applica se più di una regola corrisponde a un elemento particolare. Questo schema di priorità a cascata è prevedibile.

Le specifiche CSS sono gestite dal World Wide Web Consortium (W3C). Tipo di supporto Internet (tipo MIME) text/css è registrato per l'uso con CSS da RFC 2318 (marzo 1998). Il W3C gestisce un servizio di convalida CSS gratuito per i documenti CSS.

Oltre all'HTML, altri linguaggi di markup supportano l'uso di CSS tra cui XHTML, XML semplice, SVG e XUL.

2.3.1 CSS interni ed esterni

Gli stili di una pagina possono essere definiti sia all'interno del file nel quale devono operare, sia in un file a parte.

I **fogli di stile esterni** (o collegati) sono quelli che meglio rispondono alle esigenze degli sviluppatori, e soprattutto quelli che meglio interpretano la filosofia dei fogli di stile. Per comprendere pienamente le peculiarita' dei CSS esterni si pensi ad un sito di grandi dimensioni con 1000 pagine HTML e un certo tipo di formattazione del testo. Improvvisamente esigenze estetiche impongono di modificare colore di sfondo, tipo di carattere e allineamento del testo; è quì che si presentano i primi problemi che mettono in evidenza la versatilità dei fogli di stile. La soluzione a questo problema è che gli stili dei singoli marcatori vengano raggruppati in un unico documento indipendente dal resto delle pagine del sito, e da queste semplicemente richiamati con una riga di codice. In questo modo si separano la formattazione dei 1000 fogli del sito dal loro contenuto; una modifica sul file del "foglio di stile esterno" genera automaticamente "a cascatala" la stessa modifica su tutti i documenti di quel sito che richiamano tale file.

Creare un foglio di stile esterno e richiamarlo all'interno delle pagine HTML e' semplice, basta definire in un file esterno e collegarlo attraverso il tag link all'interno dell'head del file da formattare:

```
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="stile.css">
    </head>
```

I vantaggi di usare i fogli di stile esterni sono:

- **pieno controllo della struttura della pagina**. Il CSS consente di visualizzare la pagina Web secondo gli standard HTML del W3C senza scendere a compromessi con l'aspetto reale della pagina. Google è il principale motore di ricerca e la principale fonte di traffico. Google attribuisce pochissimo valore alle pagine Web ben organizzate, poiché il valore è scarso, quindi molti designer lo ignorano. Ma prendendo un piccolo valore può indirizzare molto traffico al sito web.

- **dimensioni ridotte del file**. Includendo lo stile del testo in un file separato, è possibile ridurre drasticamente le dimensioni del file delle pagine. Inoltre, il rapporto contenuto-codice è di gran lunga maggiore rispetto alle semplici pagine HTML, rendendo così la struttura della pagina più facile da leggere sia per il programmatore che per gli spider.

- **meno tempo di caricamento**. Da quando Google ha incluso il tempo di caricamento nel suo algoritmo, diventa più importante esaminare il tempo di caricamento della pagina e un altro vantaggio di avere pagine di dimensioni ridotte si traduce in costi di larghezza di banda ridotti.

- **valutazione della pagina**. Nella parte SEO è molto importante utilizzare CSS esterni. Gli spider dei motori di ricerca saranno in grado di indicizzare le pagine molto più velocemente, poiché le informazioni importanti possono essere collocate più in alto nel documento HTML. Inoltre, la quantità di contenuto pertinente sarà maggiore della quantità di codice in una pagina. Il motore di ricerca non dovrà guardare troppo lontano nel codice per trovare il contenuto reale. I CSS ti aiuteranno a creare pagine altamente leggibili, ricche di contenuti, che si riveleranno estremamente utili nella tua campagna SEO. Un migliore posizionamento del sito significa una migliore visibilità sul Web e ciò può tradursi in un maggior numero di visitatori e, in definitiva, in un aumento delle vendite o del numero di contratti.

La peculiarità fondamentale dei **fogli di stile interni** e' di venire assegnati all'intero documento e non a singole istanze come l'HTML classico. Si tratta, dunque, di un insieme di definizioni di stile inserite tra marcature <STYLE> e posizionate tra i tag <HEAD> del documento e il <BODY> dello stesso.

```
<head>
    <style type="text/css">
        div .colore-bianco { background-color:white }
        div .colore-nero { background-color:black }
    </style>
</head>
```

Impostato il CSS incorporato all'interno del documento gli stili vengono assegnati automaticamente al testo racchiuso tra i marcatori. In questo modo cambiando solo i valori degli attributi di stile impostati nel foglio (aumentando la grandezza o modificando il colore, per esempio), il testo del documento tra i marcatori viene automaticamente aggiornato al nuovo stile.

L'attributo TYPE del tag <STYLE> definisce il linguaggio in formato MIME del foglio di stile. In altre parole questo attributo indica al browser il

tipo di foglio di stile supportato, altrimenti ignorato. Ms Internet Explorer supporta i CSS solo in formato MIME, ma esistono i CSS in formato text/jass, cioè accessibili in javascript (e questa seconda è l'interpretazione di stile data da Netscape). Se l'attributo TYPE viene omesso, il browser lo identifica di default con "text/css". Anche Netscape communicator supporta questo attributo.

Rispetto ai fogli di stile in linea i CSS incorporati consentono il controllo di un intero documento, ma per interi siti non sono sufficienti a fornire il supporto tecnico necessario. Se per esempio per il testo del documento è stato usato un certo font e improvvisamente si decide di modificarlo, sarà necessario agire su tutti i documenti del sito perchè le modifiche si estendano all'intero sito Web. Tuttavia questa modalità di applicare lo stile è particolarmente adeguata quando si vuole usare CSS per creare delle pagine di presentazione (introduzione di grafica) dove interessa che solo la singola pagina abbia quel determinato aspetto.

I vantaggi di usare i fogli di stile interni sono:

- **problema della cache**. Gli stili interni verranno letti da tutti i browser a meno che non vengano nascosti. Ciò rimuove la possibilità di utilizzare media=all o @import per nascondere gli stili da vecchi browser, come IE4 e NN4.

- **nessun download aggiuntivo**. Nessun download aggiuntivo necessario per ricevere informazioni di stile o abbiamo meno richieste HTTP

Mentre gli svantaggi di usare i fogli di stile interni sono:

- **documenti multipli**. Questo metodo non può essere utilizzato se si desidera utilizzarlo su più pagine Web.

- **caricamento lento della pagina:** Poiché ci sono meno richieste HTTP ma utilizzando il CSS interno, il caricamento della pagina è lento rispetto a Inline ed CSS esterno.

- **grande dimensione del file**. Durante l'utilizzo del CSS interno la dimensione della pagina aumenta ma aiuta solo i designer che lavorano offline ma quando il sito Web va online consuma molto tempo rispetto a quello offline.

2.3.2 Selettori

I selettori CSS vengono utilizzati per selezionare il contenuto che si desidera modellare. Esistono diversi tipi di selettori nei CSS:

- **selettore degli Elementi**. Il selettore di elementi seleziona gli elementi HTML in base al nome dell'elemento.

```
div { background-color:white }
```

- **selettore ID**. Il selettore ID utilizza l'attributo id di un elemento HTML per selezionare un elemento specifico. L'id di un elemento è univoco all'interno di una pagina, quindi il selettore id viene utilizzato per selezionare un elemento univoco. Per selezionare un elemento con un ID specifico si utilizza un carattere hash (#), seguito dall'id dell'elemento.

```
#bianco { background-color:white }
```

- **selettore classe**. Il selettore di classe seleziona gli elementi HTML con un attributo di classe specifico. Per selezionare elementi con una classe specifica, basta scrivere un carattere punto (.), seguito dal nome della classe; si può anche specificare che solo gli elementi HTML specifici devono essere interessati da una classe.

```
.bianco { background-color:white }
div.nero { background-color:black }
```

- **Selettore universale:** Il selettore universale (*) seleziona tutti gli elementi HTML nella pagina.

```
* { background-color:white }
```

2.3.3 Pattern matching

Nel CSS, le regole di corrispondenza dei modelli determinano quali regole di stile si applicano agli elementi nella struttura del documento. Questi schemi, chiamati selettori, possono variare da semplici nomi di elementi a ricchi schemi contestuali. Se tutte le condizioni nel modello sono vere per un determinato elemento, il selettore corrisponde all'elemento.

La distinzione tra maiuscole e minuscole dei nomi degli elementi della lingua del documento nei selettori dipende dalla lingua del documento. Ad esempio, in HTML, i nomi degli elementi non fanno distinzione tra maiuscole e minuscole, ma in XML fanno distinzione tra maiuscole e minuscole.

*	Match di ogni elemento
E	Corrisponde a qualsiasi elemento E (ovvero un elemento di tipo E).
E F	Corrisponde a qualsiasi elemento F che discende da un elemento E.
E >F	Corrisponde a qualsiasi elemento F che è figlio di un elemento E.
E:first-child	Corrisponde all'elemento E quando E è il primo figlio del padre.
E:link E:visited	Corrisponde all'elemento E se E è un link di cui la destinazione non è ancora stata visitata (:link) o già visitata (:visited).
E:active E:hover E:focus	Corrisponde a E durante determinate azioni dell'utente.
E + F	Corrisponde a qualsiasi elemento F immediatamente preceduto da un elemento fratello E.

2.4 Boostrap

Bootstrap è un framework CSS gratuito e open source diretto allo sviluppo web front-end reattivo e mobile-first. Contiene modelli di progettazione basati su CSS e (facoltativamente) JavaScript per tipografia, moduli, pulsanti, navigazione e altri componenti dell'interfaccia.

Bootstrap è il sesto progetto più stellato su GitHub, con oltre 135.000 stelle, dietro freeCodeCamp (quasi 307.000 stelle) e marginalmente dietro il framework Vue.js. Secondo Alexa Rank, Bootstrap è tra i primi 2000 negli Stati Uniti, mentre vuejs.org è tra i primi 7000 negli Stati Uniti.

Di recente, Bootstrap ha guadagnato molta popolarità come framework front-end numero uno tra gli sviluppatori web. È stato sviluppato dal team di Twitter ed è una combinazione di codice JavaScript, CSS e HTML pensato per aiutare gli sviluppatori Web a costruire componenti dell'interfaccia utente. È fondamentalmente una raccolta gratuita di strumenti utilizzati per sviluppare siti Web e applicazioni web reattivi.

Bootstrap è un framework Web che si concentra sulla semplificazione dello sviluppo di pagine Web informative (rispetto alle app Web). Lo scopo principale di aggiungerlo a un progetto Web è applicare le scelte di colore, dimensioni, carattere e layout di Bootstrap a quel progetto. Pertanto, il fattore principale è se gli sviluppatori responsabili trovano queste scelte di loro gradimento. Una volta aggiunto a un progetto, Bootstrap fornisce le definizioni di stile di base per tutti gli elementi HTML. Il risultato è un aspetto uniforme per selettori, tabelle ed elementi nei browser Web. Inoltre, gli sviluppatori possono sfruttare le classi CSS definite in Bootstrap per personalizzare ulteriormente l'aspetto dei loro contenuti.

Bootstrap include anche diversi componenti JavaScript sotto forma di plugin

jQuery. Forniscono ulteriori elementi dell'interfaccia utente come finestre di dialogo, descrizioni comandi e caroselli. Ogni componente Bootstrap è costituito da una struttura HTML, dichiarazioni CSS e, in alcuni casi, da un codice JavaScript. Estendono anche la funzionalità di alcuni elementi dell'interfaccia esistenti, inclusa ad esempio una funzione di completamento automatico per i campi di input.

I componenti più importanti di Bootstrap sono i suoi componenti di layout, in quanto influenzano un'intera pagina Web. Il componente di layout di base è chiamato "container", poiché ogni altro elemento della pagina è inserito in esso. Gli sviluppatori possono scegliere tra un container a larghezza fissa e un container a larghezza fluida. Mentre quest'ultimo riempie sempre la larghezza della pagina Web, il primo utilizza una delle quattro larghezze fisse predefinite, a seconda della dimensione dello schermo che mostra la pagina:

- Più piccolo di 576 pixel
- 576–768 pixel
- 768–992 pixel
- 992-1200 pixel
- Più grande di 1200 pixel

Una volta inizializzato un container, altri componenti del layout Bootstrap implementano un layout CSS Flexbox attraverso la definizione di righe e colonne. Una versione precompilata di Bootstrap è disponibile sotto forma di un file CSS e tre file JavaScript che possono essere facilmente aggiunti a qualsiasi progetto. La forma grezza di Bootstrap, tuttavia, consente agli sviluppatori di implementare ulteriori personalizzazioni e ottimizzazioni delle dimensioni. Questo modulo non elaborato è modulare, il che significa che lo sviluppatore può rimuovere componenti non necessari, applicare un tema e modificare i file Sass non compilati.

Tuttavia, come con qualsiasi framework, ci sono alcuni pro e contro chiari nell'utilizzo di Bootstrap.

I vantaggi di Boostrap sono:

- **risparmia tempo**. La scrittura del codice può richiedere molto tempo, soprattutto se si documenta il lavoro e biosgna leggere il codice per capire dove operare. Bootstrap non solo ha un'ottima documentazione su ogni componente, ma utilizzandolo non si dovrà più scrivere codice. Tutto ciò di cui si ha bisogno per iniziare è una conoscenza di base di HTML, CSS o Javascript e si puà iniziare a sviluppare ed è anche estremamente semplice da usare.

- **bootstrap incoraggia la coerenza**. I dipendenti di Twitter hanno iniziato a sviluppare Bootstrap pensando a un framework che sarà coerente sopra ogni altra cosa. Risolvendo i problemi di coerenza tra il fronte dello sviluppo e gli utenti finali Bootstrap è diventato il framework numero uno tra gli sviluppatori web. Il problema principale che ha risolto sono varie incongruenze che esistono tra sviluppatori e designer in un progetto perché i risultati finali di Bootstrap sembrano sempre gli stessi su ogni piattaforma e ogni browser. Riconoscendo la necessità di associare designer e sviluppatori, Bootstrap è diventato framework open source nel 2011.

- **migliore lavoro di squadra**. Una delle feature migliori di Bootstrap è che i programmatori saranno liberi di scrivere codice senza doversi preoccupare di come sarà il loro lavoro alla fine. Se una nuova persona si unisce al team, sarà estremamente facile per lei imparare come utilizzare il framework. Tutto ciò che deve sapere sul framework puo trovarlo nella documentazione fornita da Boostrap.

- **bootstrap offre un eccellente sistema di griglia (Grid System)**. È necessaria una buona grid se si cerca di creare un layout di pagina e Bootstrap ha probabilmente il miglior sistema di griglia reattivo. Il contenuto del sito Web è sempre diviso in 12 colonne fluide e reattive. Rende il lavoro attraverso le colonne un gioco da ragazzi che è molto interessante quando si sta cercando di nascondere contenuti specifici della piattaforma. Alcuni elementi possono essere resi visibili solo su un desktop e viceversa. L'uso delle classi predefinite renderà l'utilizzo della griglia molto più semplice e veloce.

- **reattività**. Se c'è una cosa certa oggi è che i dispositivi mobili continueranno a crescere in popolarità. Oggi la maggior parte delle ricerche iniziali viene eseguita tramite dispositivi mobili, il che dimostra quanto sia importante avere un sito Web reattivo. Questo è il motivo per cui la maggior parte delle aziende come Webwingz Web Development consiglia di fare sempre tutto il possibile per la reattività del tuo sito Web. Bene, Bootstrap si basa sulla creazione di siti Web ottimizzati per i dispositivi mobili. Non si deve letteralmente fare nulla per raggiungere il livello richiesto di reattività, grazie alla griglia fluida già menzionata.

Mentre i contro sono:

- **ogni sito Web Bootstrap è simile**. Bootstrap è stato creato dal team di Twitter allo scopo di lavorare più velocemente su un'interfaccia standardizzata, il che rende Bootstrap uno strumento appositamente

progettato che non è completamente allineato con i concetti standard di sviluppo web. Lo svantaggio principale è che tutto ciò che è stato creato con Bootstrap avrà un aspetto molto simile. Si potrebbe essere in grado di sovrascrivere e modificare manualmente i fogli di stile, ma questa modalità di lavoro inibisce lo scopo di utilizzare inizialmente il framework. E anche se lo si fa, non cambia il fatto che tutti i siti web creati attraverso questo framework saranno estremamente riconoscibili come Bootstrap.

- **curva di apprendimento**. Sebbene sia facile imparare ad utilizzare Bootstrap, si deve comunque investire del tempo. Si dovrà conoscere tutte le classi CSS Bootstrap disponibili, ma anche come i componenti Bootstrap accedono a queste classi. Si avrà anche bisogno di un pò di tempo per sperimentare e abituarsi al sistema a griglia. Come già accennato, la straordinaria documentazione di Bootstrap è molto utile per completare il processo di apprendimento e, una volta abituati, non sarà necessario molto tempo per adattarsi alle nuove versioni del codice.

- **può essere pesante**. Sebbene sia facile creare un sito Web reattivo con Bootstrap, i risultati finali possono essere un pò pesanti per gli utenti in termini di tempi di caricamento e problemi di consumo della batteria. Inoltre, i file generati da Bootstrap possono essere di dimensioni enormi, il che può rallentare le cose abbastanza pesantemente per browser. Si dovrebbe essere in grado di andare avanti ed eliminare le cose manualmente, ma ciò vanifica di nuovo lo scopo di utilizzare il framework.

2.5 GIT

Git è un sistema di controllo della versione distribuito per tenere traccia delle modifiche al codice sorgente durante lo sviluppo del software. È progettato per coordinare il lavoro tra i programmatori, ma può essere utilizzato per tenere traccia delle modifiche in qualsiasi set di file. I suoi obiettivi includono velocità, integrità dei dati e supporto per flussi di lavoro distribuiti e non lineari.

Git è stato creato da Linus Torvalds nel 2005 per lo sviluppo del kernel Linux, con altri sviluppatori del kernel che hanno contribuito al suo sviluppo iniziale. Il suo attuale manutentore dal 2005 è Junio Hamano. Come con la maggior parte degli altri sistemi di controllo della versione distribuiti e, diversamente dalla maggior parte dei sistemi client-server, ogni directory Git su ogni computer è un repository completo con cronologia completa e capacità di tracciamento completo della versione, indipendentemente dall'accesso alla rete

o da un server centrale. Git è un software gratuito e open source distribuito secondo i termini della GNU General Public License versione 2.

2.5.1 Caratteristiche

Sistema Distribuito

I sistemi distribuiti sono quelli che consentono agli utenti di eseguire lavori su un progetto da tutto il mondo. Un sistema distribuito contiene un repository centrale a cui possono accedere molti collaboratori remoti utilizzando un sistema di controllo versione. Git è uno dei sistemi di controllo delle versioni più popolari attualmente in uso. Avere un server centrale comporta un problema di perdita di dati o disconnessione dei dati in caso di guasto del sistema al server centrale. Per affrontare questo tipo di situazione, Git rispecchia l'intero repository su ogni istantanea della versione che viene estratta dall'utente. In questo caso, se il server centrale si arresta in modo anomalo, la copia dei repository può essere recuperata dagli utenti che hanno scaricato l'ultima istantanea del progetto.

Avendo un sistema distribuito, Git consente agli utenti di lavorare contemporaneamente allo stesso progetto, senza interferire con il lavoro degli altri. Quando un determinato utente ha terminato la propria parte del codice, invia le modifiche al repository e queste modifiche vengono aggiornate nella copia locale di ogni altro utente remoto che estrae l'ultima copia del progetto.

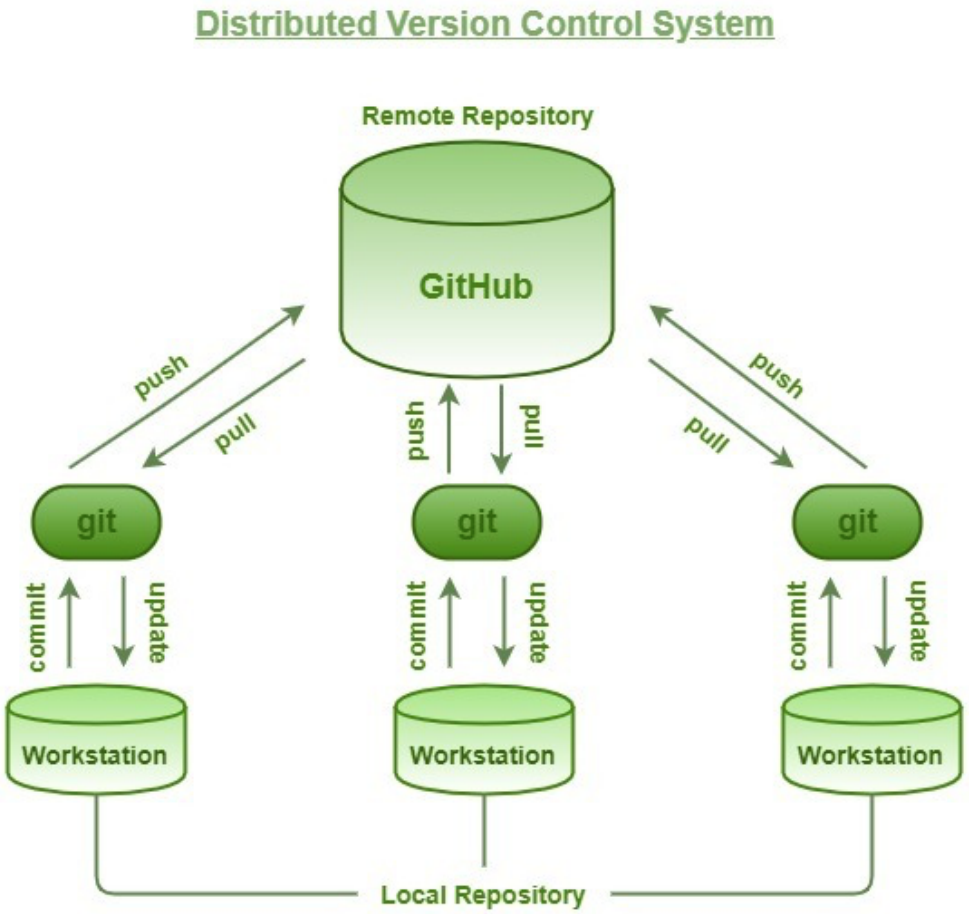

Figura 2.7: Distributed Version Control System in GIT

Compatibilità

È compatibile con tutti i sistemi operativi attualmente in uso. I repository Git possono anche accedere ai repository di altri sistemi di controllo versione come SVN, CVK, ecc. Git può accedere direttamente ai repository remoti creati da questi SVN. Quindi, gli utenti che non stavano usando Git in primo luogo possono anche passare a Git senza passare attraverso il processo di copia dei loro file dai repository di altri VCS in Git-VCS. Git può anche accedere ai repository centrali di altri VCS. Quindi, è possibile eseguire lavori su Git-SVN e utilizzare il repository centrale come lo stesso. Git ha un'emulazione del server CVS, che consente l'uso dei client CVS e dei plugin IDE esistenti per accedere ai repository Git.

Sviluppo non lineare

Git consente agli utenti di tutto il mondo di eseguire operazioni su un progetto in remoto. Un utente può ritirare qualsiasi parte del progetto ed eseguire l'operazione richiesta e quindi aggiornare ulteriormente il progetto. Questo può essere fatto dal comportamento di sviluppo non lineare di Git. Git supporta la rapida ramificazione e fusione e include strumenti specifici per la visualizzazione e la navigazione di una cronologia di sviluppo non lineare. Un presupposto importante in Git è che un cambiamento verrà unito più spesso di quanto sia scritto. Git registra lo stato corrente del progetto in una forma ad albero. Un nuovo ramo può essere aggiunto all'albero in qualsiasi momento e viene ulteriormente unito al progetto finale una volta completato.

Branching

Git consente ai suoi utenti di lavorare su una linea parallela ai file di progetto principali. Queste linee sono chiamate rami. Le filiali in Git forniscono una funzionalità per apportare modifiche al progetto senza influire sulla versione originale. Il ramo principale di una versione conterrà sempre il codice di qualità della produzione. Ogni nuova funzionalità può essere testata e lavorata sui rami e inoltre può essere unita al ramo principale. La diramazione e l'unione possono essere eseguite molto facilmente con l'aiuto di alcuni comandi Git. Una singola versione di un progetto può contenere n numero di filiali in base alle esigenze dell'utente.

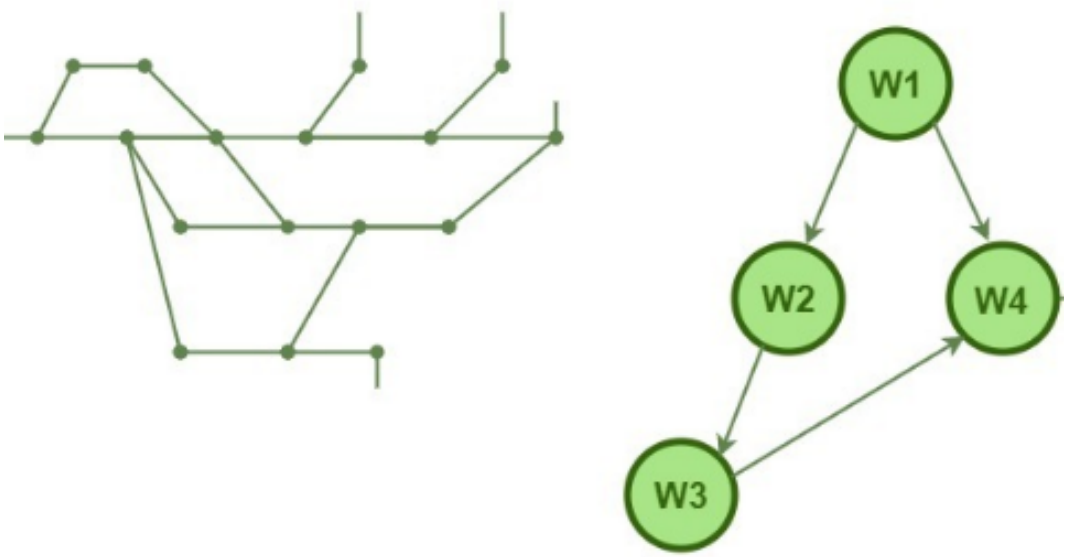

Figura 2.8: Esempio tipico di Branch in GIT

Leggero

Git archivia tutti i dati dal repository centrale al repository locale mentre viene eseguita la clonazione. Potrebbero esserci centinaia di utenti che lavorano allo stesso progetto e quindi i dati nel repository centrale potrebbero essere molto grandi. Uno potrebbe essere preoccupato che la clonazione di molti dati nei computer locali potrebbe causare un errore del sistema, ma Git ha già risolto questo problema. Git segue i criteri di compressione senza perdita di dati che comprime i dati e li archivia nel repository locale occupando uno spazio minimo. Ogni volta che è necessario per questi dati, segue la tecnica inversa e consente di risparmiare molto spazio di memoria.

Velocità

Poiché Git archivia tutti i dati relativi a un progetto nel repository locale mediante il processo di clonazione, è molto efficiente recuperare i dati dal repository locale invece di fare lo stesso dal repository remoto. Git è molto veloce e scalabile rispetto ad altri sistemi di controllo di versione che si traducono in una gestione efficiente di grandi progetti. La potenza di recupero da un repository locale è circa 100 volte più veloce di quanto possibile con il server remoto. Secondo un test condotto da Mozilla, Git è un ordine di grandezza più veloce, che è circa 10 volte più veloce di altri strumenti VCS. Questo perché Git è in realtà scritto in linguaggio C, a differenza di altri linguaggi, molto vicino al linguaggio macchina e quindi rende l'elaborazione molto veloce.

Open Source

Git è un sistema di controllo della versione distribuito gratuito e open source progettato per gestire qualsiasi cosa, dai progetti piccoli a quelli molto grandi, con velocità ed efficienza. Si chiama open-source perché offre la flessibilità di modificare il suo codice sorgente in base alle esigenze dell'utente. A differenza di altri sistemi di controllo della versione che forniscono funzionalità a pagamento come lo spazio del repository, la privacy dei codici, l'accuratezza e la velocità, ecc. Git è tutto un software open source che fornisce queste funzionalità in modo gratuito e persino in un modo migliore di altri. Essere Git open source consente a più persone di lavorare allo stesso progetto contemporaneamente e collaborare tra loro in modo molto semplice ed efficiente. Quindi, Git è considerato il miglior sistema di controllo della versione disponibile al giorno d'oggi.

Affidabile

Fornendo un repository centrale che viene clonato ogni volta che un utente esegue l'operazione Pull, i dati del repository centrale vengono sempre sottoposti a backup nel repository locale di ogni collaboratore. Pertanto, in caso di crash del server centrale, i dati non andranno mai persi in quanto possono essere recuperati facilmente da qualsiasi macchina locale dello sviluppatore. Una volta che il server centrale è stato riparato, i dati possono essere recuperati da uno qualsiasi dei vari collaboratori. C'è una probabilità molto bassa che i dati non siano disponibili con nessuno sviluppatore perché quello che ha lavorato sul progetto per ultimo avrà sicuramente l'ultima versione del progetto sul suo computer locale. Lo stesso vale per il cliente. Se uno sviluppatore perde i suoi dati a causa di un errore tecnico o di uno dei motivi imprevisti, può facilmente estrarre i dati dal repository centrale e ottenere l'ultima versione degli stessi sul proprio computer locale. Quindi, il push dei dati sul repository centrale rende Git più affidabile su cui lavorare.

Sicuro

Git registra tutti gli commit eseguiti da ciascuno dei collaboratori sulla copia locale dello sviluppatore. Un file di registro viene gestito e viene inviato al repository centrale ogni volta che viene eseguita l'operazione di push. Quindi, se si verifica un problema, può essere facilmente monitorato e gestito dallo sviluppatore. Git utilizza SHA1 per archiviare tutti i record sotto forma di oggetti nell'Hash. Ogni oggetto collabora tra loro con l'uso di questi tasti Hash. SHA1 è un algoritmo crittografico che converte l'oggetto di commit in un codice esadecimale di 14 cifre. Aiuta a memorizzare il record di tutti gli commit effettuati da ciascuno degli sviluppatori. Quindi, facilmente diagnosticabile ciò che commette ha portato al fallimento del lavoro.

Trello

Trello è un web-based Kanban-style creatore di liste distrubito da Atlassian. Un Kanban board è uno degli strumenti che possono essere utilizzati per implementare Kanban per gestire il lavoro a livello personale o organizzativo. Le schede kanban rappresentano visivamente il lavoro nelle varie fasi di un processo usando le "card" per rappresentare gli oggetti di lavoro e le colonne per rappresentare ogni fase del processo. Le card vengono spostate da sinistra a destra per mostrare i progressi e aiutare a coordinare i team che svolgono il lavoro. Una Kanban board può essere divisa in "swimlanes" orizzontali che rappresentano diversi tipi di lavoro o squadre diverse che eseguono il lavoro. Le schede Kanban possono essere utilizzate nel lavoro di conoscenza o per i

processi di produzione. Le schede semplici hanno colonne per "attesa", "in corso" e "completato" o "da fare", "fare" e "fatto". È possibile creare schede Kanban complesse che suddividono il lavoro "in progress" in più colonne per visualizzare il flusso di lavoro attraverso un'intera mappa di flussi di valori. I principali vantaggi di utilizzare Trello sono:

- **sistema di schede ben organizzato**. Gli sviluppatori di Trello hanno sicuramente cercato il metodo di organizzazione del flusso di lavoro più semplice e intuitivo e si sono imbattuti nel loro esclusivo sistema di scheda per una visione completa dei progressi. Con una scheda assegnata per ciascun progetto e una scheda per ciascuna attività, esiste un rischio minimo di confusione, poiché tutti gli incarichi sono in ordine e possono essere monitorati con elenchi di prestazioni specifici.

- **elaborazione fluida**. Il compito di Trello è di organizzare la gestione del progetto, motivo per cui si assicura che sia possibile modificare gli elenchi di attività in linea, utilizzando il meccanismo di trascinamento della selezione più semplice possibile. Gli elenchi sono personalizzabili con cura, il che significa che puoi seguire esclusivamente metriche di interesse personale e utilizzare le notifiche automatizzate per rimanere aggiornato su tutte le modifiche e le alterazioni.

- **collaborazione**. Uno dei vantaggi che si nota costantemente nelle recensioni di Trello è che molti esperti sostengono che Trello è soprattutto un sistema di collaborazione, un'affermazione che può essere facilmente giustificata dal numero di funzioni di squadra disponibili nel sistema: usandolo, si può consentire a tutto il team di partecipare a discussioni importanti sia per riunioni di gruppo che per sessioni di chat individuali), inviare diatribes e note, condividere file di tutti i formati e commentare singole attività e compiti. Il sistema ti consente anche di caricare file direttamente dal tuo account Dropbox, Box o Google Drive.

- **tempi ragionevoli**: Si può utilizzare il calendario di accensione di Trello per stabilire le priorità delle attività con scadenze più brevi, assegnare le operazioni dell'ultimo minuto e visualizzarle in base al loro stato.

- **database ricercabile**. Trello si assicura che tutte le discussioni pertinenti e i dati aziendali vengano elegantemente archiviati nel sistema per un ulteriore utilizzo e sottoposti a backup contro eventuali violazioni o guasti. Sono disponibili numerose etichette e filtri di ricerca per aiutare a individuare il file desiderato in pochissimo tempo.

- **sicurezza**. Il sistema è progettato per soddisfare i più elevati standard di sicurezza e utilizza un meccanismo di crittografia per garantire che i

dati non finiscano mai nelle mani sbagliate. Come amministratore, sarà permesso di impostare le autorizzazioni e decidere che le bacheche sono private e accessibili solo ai partecipanti autorizzati.

- **integrazioni**. Trello funziona con le API degli sviluppatori pubblici, il che significa che si può collegarlo letteralmente a qualsiasi app / sistema, estensione o plug-in di terze parti.

- **ottimizzazione mobile**. Trello è un sistema di collaborazione ottimizzato per dispositivi mobili in cui è possibile accedere ai dati da qualsiasi dispositivo, compresi quelli che funzionano con Android e iOS.

Mentre gli svantaggi sono:

- **online software**. Come software online, Trello fa affidamento sulla disponibilità di dati per il suo funzionamento, che è anche lo stesso problema di un software simile. L'accesso a Internet è ciò che lo alimenta e consente agli utenti di svolgere i propri compiti. Quindi, quando non c'è una connessione internet, Trello non può funzionare né può essere aggiornato.

- **commenti di Trello**. Negli altri software simili possono scrivere commenti e modificarli. Trello, d'altra parte, consente solo di commentare, ma non è possibile modificarli. Se un commento è già stato pubblicato e salvato, l'unica cosa che si può fare per correggerlo è creare un commento completamente nuovo.

- **memoria limitata**. Trello consente gli allegati, ma gli utenti della versione gratuita hanno solo un limite massimo di 10 MB per upload, che è piuttosto piccolo. Tuttavia, questo può essere aggiornato a 250 MB per upload se l'utente aggiorna anche il suo abbonamento a Gold.

- **è adatto a piccoli progetti**. Le persone che hanno progetti e team più piccoli possono sicuramente trarre vantaggio da Trello. Tuttavia, potrebbe essere un problema se il team cresce e il progetto diventa più complesso.

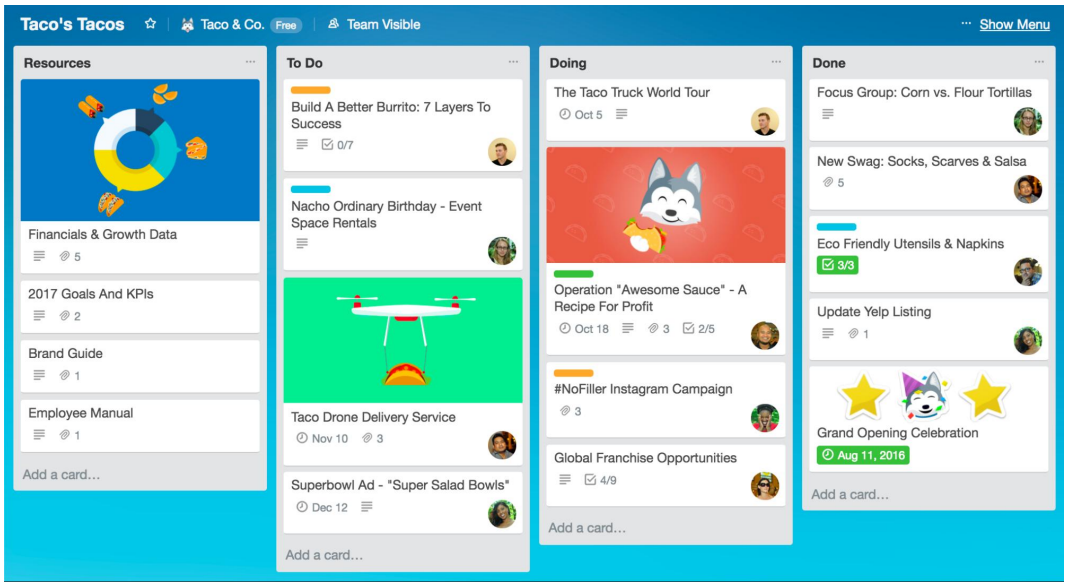

Figura 2.9: Esempio di Board di Trello

Capitolo 3

Sviluppo della Applicazione

In questo capitolo si spiegano tutti i passaggi per arrivare all'applicazione finale.

Lo sviluppo del software è il processo di ideazione, specifica, progettazione, programmazione, documentazione, test e correzione dei bug coinvolti nella creazione e manutenzione di applicazioni, framework o altri componenti software. Lo sviluppo del software è un processo di scrittura e mantenimento del codice sorgente, ma in un senso più ampio, include tutto ciò che è coinvolto tra la concezione del software desiderato e la manifestazione finale del software, a volte in un processo pianificato e strutturato. Pertanto, lo sviluppo del software può includere ricerca, nuovo sviluppo, prototipazione, modifica, riutilizzo, reingegnerizzazione, manutenzione o qualsiasi altra attività che comporti prodotti software.

Esistono molti approcci alla gestione dei progetti software, noti come modelli, metodologie, processi o modelli del ciclo di vita dello sviluppo software. Il modello a cascata è una versione tradizionale, in contrasto con la più recente innovazione dello sviluppo software agile.

Il **waterfall model** o modello a cascata è una suddivisione delle attività del progetto in fasi sequenziali lineari, in cui ogni fase dipende dai risultati del precedente e corrisponde a una specializzazione dei compiti. L'approccio è tipico per alcune aree della progettazione ingegneristica. Nello sviluppo del software, tende ad essere tra gli approcci meno iterativi e flessibili, poiché i progressi fluiscono in gran parte in una direzione ("verso il basso" come una cascata) attraverso le fasi di concezione, avvio, analisi, progettazione, costruzione, test, implementazione e manutenzione.

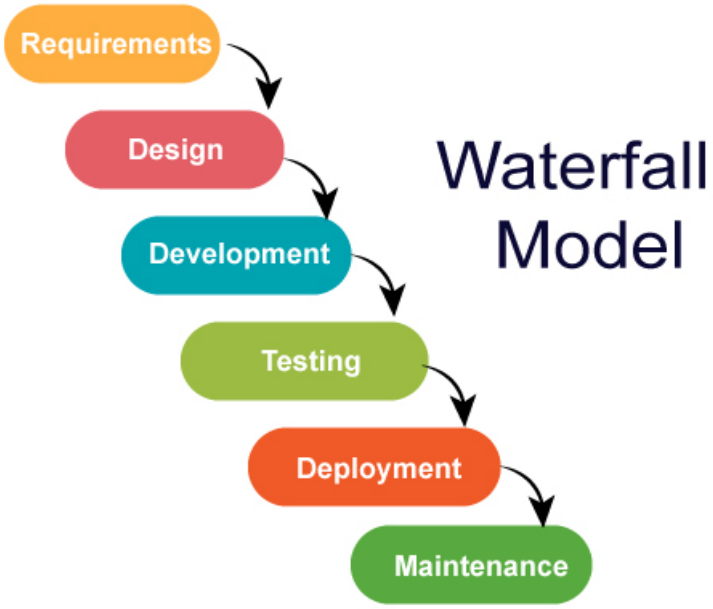

Figura 3.1: Wayerfall Model

Lo **sviluppo software agile** comprende vari approcci allo sviluppo del software in base ai quali i requisiti e le soluzioni evolvono attraverso lo sforzo collaborativo dei team auto-organizzanti e inter funzionali e dei loro clienti / utenti / utenti finali. Promuove la pianificazione adattiva, lo sviluppo evolutivo, la consegna anticipata e il miglioramento continuo e incoraggia risposte flessibili al cambiamento.

Il termine agile (a volte scritto Agile) è stato reso popolare, in questo contesto, dal Manifesto per lo sviluppo di software Agile. I valori e i principi adottati in questo manifesto derivano e sostengono un'ampia gamma di framework di sviluppo software, tra cui Scrum e Kanban.

Nello sviluppo Agile, uno **sprint** è un periodo di tempo prestabilito durante il quale è necessario completare un lavoro specifico e prepararlo per la revisione.

Ogni sprint inizia con una riunione di pianificazione. Durante l'incontro, il proprietario del prodotto (la persona che richiede il lavoro) e il team di sviluppo concordano esattamente quale lavoro verrà realizzato durante lo sprint. Il team di sviluppo ha l'ultima parola quando si tratta di determinare quanto lavoro può realisticamente essere realizzato durante lo sprint, e il proprietario del prodotto ha l'ultima parola su quali criteri devono essere soddisfatti affinché il lavoro sia approvato e accettato.

Il flusso di lavoro dello sprint ha lo scopo di aiutare i membri del team a valutare il proprio lavoro e a comunicare tra loro durante l'intero processo; il

flusso di lavoro è seguito per ogni sprint. A fine dello sprint il team si riunirà e seguirà il seguente schema:

- Discutere e concordare reciprocamente sulla portata del lavoro che si intende svolgere durante tale sprint.
- Selezionare i task arretrati che possono essere completati nello sprint
- Preparare un backlog di sprint che includa il lavoro necessario per completare gli elementi del backlog
- Concordare l'obiettivo dello sprint, una breve descrizione di ciò che si sta prevedendo di consegnare al termine dello sprint.
- Man mano che viene elaborato il lavoro dettagliato, alcuni elementi del backlog del programma possono essere suddivisi o reinseriti nel backlog se il team non crede più di poter completare il lavoro richiesto in un singolo sprint
- Una volta che il team di sviluppo ha preparato il proprio backlog di sprint, prevedono (generalmente votando) quali attività verranno consegnate all'interno dello sprint.

3.1 Approccio Utilizzato

Prima di decidere su quale approccio appoggiarsi si è deciso in primo luogo di capire ed elencare i vantaggi e gli svantaggi dei due metodi per poi scegliere quello che è più vicino alle necessità.

3.1.1 Vantaggi e svantaggi del Waterfall Model

Il metodo a cascata è un approccio lineare allo sviluppo del software. In questa metodologia, la sequenza di eventi è simile a:

- Raccogliere e documentare i requisiti
- Creazione del design dell'applicazione
- Creazione del codice e dei unit test
- Eseguire test di sistema
- Eseguire test di accettazione dell'utente (UAT)
- Risolvere eventuali problemi e bug
- Consegna del prodotto finito

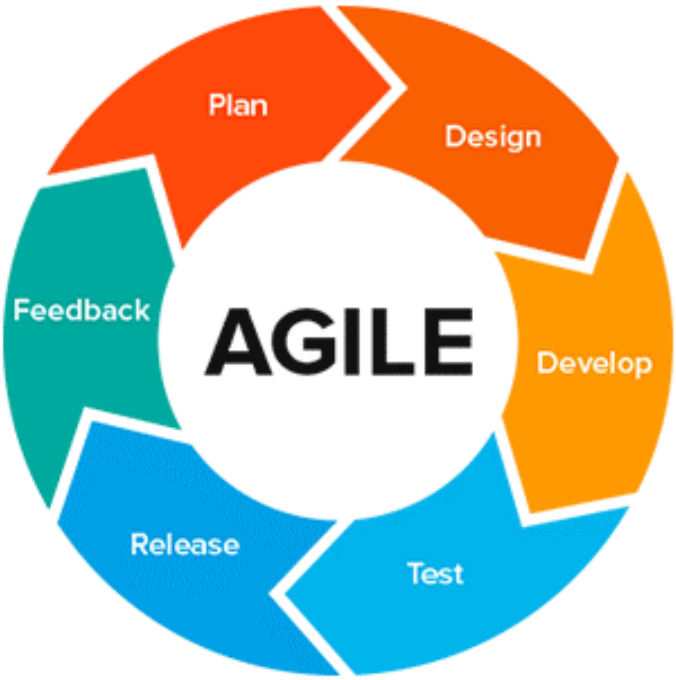

Figura 3.2: Agile Model

Vantaggi del Waterfall Model

- Gli sviluppatori e i clienti concordano su ciò che verrà consegnato all'inizio del ciclo di vita dello sviluppo. Ciò rende la pianificazione e la progettazione più semplici.

- I progressi vengono misurati più facilmente, poiché l'intero ambito di lavoro è noto in anticipo.

- Durante lo sforzo di sviluppo, è possibile che vari membri del team siano coinvolti o continuino con altri lavori, a seconda della fase attiva del progetto. Ad esempio, gli analisti aziendali possono conoscere e documentare ciò che deve essere fatto, mentre gli sviluppatori stanno lavorando su altri progetti. I tester possono preparare script di test dalla documentazione dei requisiti mentre è in corso la codifica.

- Ad eccezione di revisioni, approvazioni, riunioni sullo stato del progetto ecc.. la presenza del cliente non è strettamente necessaria dopo la fase dei requisiti.

- Poiché la progettazione è completata nelle prime fasi del ciclo di vita dello sviluppo, questo approccio si presta a progetti in cui è necessario progettare più componenti software (a volte in parallelo) per l'integrazione con sistemi esterni.

- Infine, il software può essere progettato completamente e con maggiore attenzione, sulla base di una comprensione più completa di tutti i risultati del software. Ciò fornisce una migliore progettazione del software con una minore probabilità di "effetto frammentario", un fenomeno di sviluppo che può verificarsi quando vengono definite parti di codice e successivamente aggiunte a un'applicazione in cui potrebbero o meno adattarsi.

Svantaggi del Waterfall Model

- Un'area che quasi sempre manca è l'efficacia dei requisiti. Raccogliere e documentare i requisiti in modo significativo per un cliente è spesso la parte più difficile dello sviluppo del software. A volte i clienti sono intimiditi da dettagli forniti all'inizio del progetto, che sono richiesti con questo approccio. Inoltre, i clienti non sono sempre in grado di visualizzare un'applicazione da un documento di requisiti. Wireframe e mockup possono aiutare, ma non c'è dubbio che la maggior parte degli utenti finali abbia qualche difficoltà a mettere insieme questi elementi con i requisiti scritti per arrivare a una buona immagine di ciò che otterranno.

- Un altro potenziale svantaggio del puro sviluppo a cascata è la possibilità che il cliente non sia soddisfatto del prodotto software consegnato. Poiché tutti i risultati finali si basano su requisiti documentati, un cliente potrebbe non vedere ciò che verrà consegnato fino a quando non sarà quasi finito. A quel punto, i cambiamenti possono essere difficili (e costosi) da attuare.

3.1.2 Vantaggi e svantaggi del Metodo Agile

Agile è un approccio iterativo allo sviluppo basato su team. Questo approccio enfatizza la rapida consegna di un'applicazione in componenti funzionali completi. Invece di creare attività e pianificazioni, tutto il tempo è "time-boxed" in fasi chiamate "sprint". Ogni sprint ha una durata definita (di solito in settimane) con un elenco continuo di risultati, pianificato all'inizio dello sprint. Ai prodotti finali viene assegnata la priorità in base al valore aziendale determinato dal cliente. Se non è possibile completare tutto il lavoro pianificato per lo sprint, il lavoro viene ridistribuito e le informazioni vengono utilizzate per la futura pianificazione dello sprint.

Una volta completato, il lavoro può essere rivisto e valutato dal team di progetto e dal cliente, attraverso build giornaliere e dimostrazioni di fine sprint. Agile conta su un livello molto elevato di coinvolgimento dei clienti durante tutto il progetto, ma soprattutto durante queste revisioni.

Vantaggi del metodo agile

- Il cliente ha frequenti e precoci opportunità di vedere il lavoro consegnato e di prendere decisioni e cambiamenti durante il progetto di sviluppo.
- Il cliente acquisisce un forte senso di proprietà collaborando ampiamente e direttamente con il team di progetto per tutto il progetto.
- Se il time to market per una specifica applicazione è una preoccupazione maggiore rispetto al rilascio di un set completo di funzionalità al lancio iniziale, Agile può produrre più rapidamente una versione base del software funzionante su cui è possibile basarsi in iterazioni successive.
- Lo sviluppo è spesso più incentrato sull'utente, probabilmente a seguito di una direzione più e frequente da parte del cliente.

Svantaggi del Metodo Agile

- L'alto livello di coinvolgimento dei clienti, benché ottimo per il progetto, può presentare problemi per alcuni clienti che semplicemente non hanno il tempo o l'interesse per questo tipo di partecipazione.
- Agile funziona al meglio quando i membri del team di sviluppo sono completamente dedicati al progetto.
- Poiché Agile si concentra sulla consegna in time-box e sulla frequente ri-prioritizzazione, è possibile che alcuni articoli impostati per la consegna non vengano completati entro il lasso di tempo assegnato. Potrebbero essere necessari sprint aggiuntivi (oltre a quelli inizialmente previsti), aumentando il costo del progetto. Inoltre, il coinvolgimento dei clienti porta spesso a funzionalità aggiuntive richieste durante il progetto. Ancora una volta, ciò può aumentare il tempo e i costi complessivi dell'implementazione.
- Le strette relazioni di lavoro in un progetto Agile sono più facili da gestire quando i membri del team si trovano nello stesso spazio fisico, il che non è sempre possibile. Tuttavia, esistono diversi modi per gestire questo problema, come webcam, strumenti di collaborazione, ecc.

- La natura iterativa dello sviluppo Agile può portare a un frequente refactoring se l'intero ambito del sistema non viene preso in considerazione nell'architettura e nel design iniziali. Senza questo refactoring, il sistema può soffrire di una riduzione della qualità complessiva. Questo diventa più pronunciato in implementazioni su larga scala o con sistemi che includono un alto livello di integrazione.

Dopo un'attenta analisi di ogni vantaggio e svantaggio di queste due metodologie si è deciso di optare sul metodo Agile: Oltre ad essere innovativo per molti aspetti, permette di avere più controllo sulle modifiche che in questo progetto risulta essenziale. Uno degli svantaggi principali del metodo agile è quello di avere un cliente poco esperto in materia: in questo caso invece avendo come "clienti" persone molto qualificate e presenti (professori e tesisti) questo problema si trasforma in un vero e proprio vantaggio da sfruttare in qualsiasi momento del progetto.

3.2 Analisi del Progetto

In primo luogo si è iniziato con l'analisi del progetto: ha lo scopo generale di chiarire, dettagliare e documentare le funzioni, i servizi e le prestazioni che devono essere offerti da un sistema software o programma, al fine di risolvere un dato problema nel contesto in cui esso dovrà operare. Le informazioni raccolte nella fase di analisi rappresentano il punto di partenza per la progettazione di un prodotto software e per l'intero processo della sua realizzazione, validazione e manutenzione.

Si ricorda che si è utilizzato il metodo agile per lo sviluppo dell'applicazione: per cui il progetto è stato diviso in sprint. Ogni sprint dovrebbe tradursi in una bozza, prototipo o versione realizzabile del progetto finale consegnabile. Lo scopo degli sprint è di suddividere un progetto in blocchi di dimensioni ridotte. Ciò consente al team di pianificare un singolo sprint alla volta e di adattare gli sprint futuri in base al risultato degli sprint già completati.Uno sprint in Agile deve essere impostato in timebox e ogni sprint deve avere la stessa lunghezza.

3.2.1 Presentazione della problematica

Il primo contatto col progetto è avvenuto tramite la presentazione del progetto: Giacomo ha iniziato, attraverso l'uso di slide esplicative, a presentare il progetto che ha svolto. Ha iniziato raccontando la sua idea e perché ha voluto intraprendere questo tipo di progetto, passando poi a spiegare le varie fasi di progettazione fino alla visione del suo prodotto. Finita questa fase riassuntiva del progetto si è iniziata la prima fase: la fase dell'intervista conoscitiva. Questa

fase è molto importante in quanto si iniziano a formare i primi pilastri che formano poi tutta la base e la conoscenza necessaria dove cui si basa l'intera applicazione. La gestione delle interviste è molto complessa in quanto i clienti potrebbero:

- Avere solo una vaga idea dei requisiti;
- Non essere in grado di esprimere i requisiti in termini comprensibili;
- Fornire requisiti in conflitto;
- Essere poco disponibili a collaborare.

Si è decisi di usare questa modalità al posto di altre (studio di documenti che esprimono i requisiti in forma, testuale osservazione passiva o attiva del processo da modellare, studio di sistemi software esistenti) in quanto Giacomo è una persona molto competente in materia e soprattutto il creatore della applicazione sottostante a quella che si sta per creare: ha pieno controllo e conosce bene il codice quindi sicuramente la persona che più di tutte può conoscere i problemi e i risultati attesi dalla nuova applicazione.

Di seguito si mostrerà una parte dell'intervista:

> *Qual'è l'obiettivo finale dell'applicazione?*
>
> L'applicazione deve mostrare tutti i risultati in maniera grafica e confidente all'utente. In questo momento l'applicazione già esegue questa operazione ma in maniera troppo "da esperti" attraverso i log di sistema e R Studio. Il sistema dovrà essere user friendly, utilizzabile e modificabile dall'utente attraverso input di sistema.
>
> *A quali utenti è rivolto?*
>
> Ai pazienti, ai familiari e a tutti gli utenti esperti in materia.
>
> *Per quanto riguarda le tecnologie, secondo lei quali sono le più adatte?*
>
> L'applicazione che ho creato è stata eseguita attraverso R e le sue librerie. Io consiglieri di utilizzare R Shiny: un ottimo package il quale crea in modo dinamico e intrigante interfacce grafiche partendo da R.
>
> *Pensa che questo progetto potrebbe essere successivamente ampliato per futuri scopi?*
>
> Si, è un ottimo trampolino di lancio per future applicazioni sempre di questo ambito.

Pensa che questo progetto possa aiutare molte persone?

Si, le persone che purtroppo soffrono di queste malattie così rare sono sempre in balia degli eventi. Non sanno mai a chi rivolgersi o se le cure che stanno eseguendo possano avere gli effetti desiderati. Questa applicazione ha il potere di cercare almeno una soluzione e delle vie alternative a questi tipi di problemi.

Si ricorda che i passi premilinari e anche tutti i successivi passi sono stati monitorati sulla piattaforma Trello: questi dati non sono stati esposti nè mostrati a nessuna persona ma sono utilizzati personalmente per avere un'idea d'insieme del progetto su ogni punto già eseguito e quelli ancora da svolgere.

Dopo questa intervista si è iniziato a ricercare sul web come altri progetti simili hanno rappresentato il problema graficamente. Questo procedimento è utile per non incappare a problemi come:

- difficoltà di trovare quello che serve (strategia, architettura dei contenuti)
- mancanza di contenuti e funzioni comprensibili (proposta di valore, copy, UI design)
- difficoltà di utlizzo (usabilità e UI)

Dopo un'attenta ricerca si è arrivati a questo mockup:

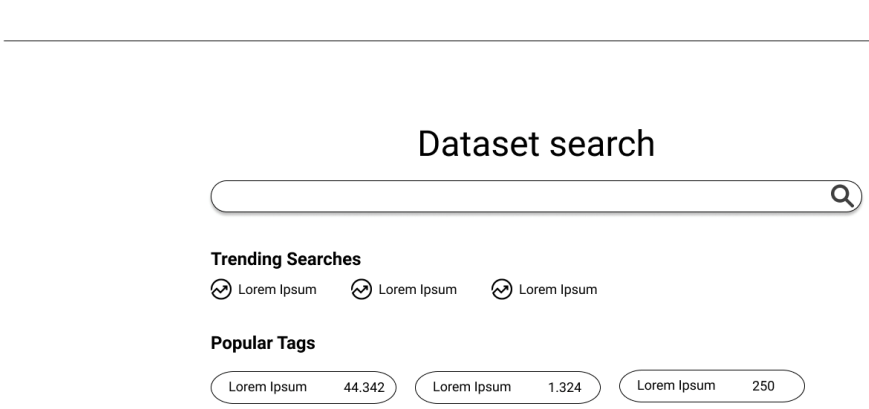

Figura 3.3: Primo Mockup dell'applicazione

I punti fermi per poter realizzare questo tipo di applicazione sono stati:

- **Barra di Ricerca:** La barra di ricerca è l'emblema e il pilastro se si vuole realizzare un'applicazione in grado di rispondere ai quesiti di un utente. Si è pensato a una barra di cerca simime a quella di Google. Questo perché se l'utente ha già confidenza con essa, è più propenso ad utilizzarla e soprattutto si immagina già il proprio utilizzo: fondamentale avere un'interfaccia pulita e semplice per quanto riguarda la GUI;

- **Risultati Grafici:** Dopo la ricerca i risultati grafici sono la base del progetto. Le ricerche e i risultati devono far mostrare all'utente elementi essenziali ed immediati a inizio pagina per poi proseguire con dati, sempre importanti, ma più articolati. Questo perché l'utente ha bisogno subito di un riscontro visivo per appagare la vista per poi concentrarsi meglio su tutta la pagina;

- **Aspetti Grafici:** per quanto riguarda l'aspetto grafico si è deciso di virare su un tema classico, con toni di colore tra il blu e il grigio. Questo perché si vuole tenere un carattere istituzionale e non troppo confidenziale. In ogni modo si utilizzerà il Material Design creato da Google e utilizzato da Boostrap per rendere comunque una visione d'insieme simile alle applicazioni mobile per rendere ancora più confidente l'utente all'applicazione.

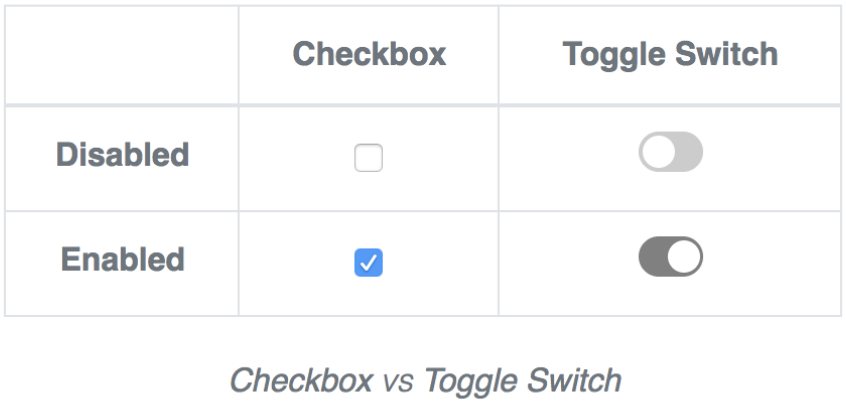

Figura 3.4: Esempi dell'utilizzo del Material Design

3.2.2 Primo Sprint

Essendo stato il primo sprint, questa riunione non avrà i soliti punti fermi che riguardano uno sprint agile: presentazione delle modifiche effettuate, possibili aggiustamenti e consigli, nuove feature da implementare.

Esposizione del Prototipo

Essendo stata la prima riunione non si è parlato dei progressi eseguiti né è stato mostrato alcun prototipo.

Chiarimenti e aggiustamenti da eseguire

Nella prima riunione si sono delineate le prime feature che il progetto doveva avere:

- Un input cerca attraverso il quale l'utente sceglie le parole che gli interessa (una o più)
- Dopo aver cliccato il tasto cerca, l'applicazione doveva mostrare la tabella dei termini più vicini alle parole ricercate e una wordcloud con sempre all'interno le parole ricercate
- Si è concordato di iniziare con tabelle e dati di test: prima di collegarsi alla vera e propria applicazione per comodità si è utilizzata una libreria che permette di ottenere dati senza dover compilarli a mano.

Infine, si è deciso la durata dello sprint scegliendo giorno e ora del prossimo incontro.

3.2.3 Secondo Sprint

Esposizione del Prototipo

Nella seconda riunione si è iniziato innanzi tutto con la presentazione dell'applicazione.

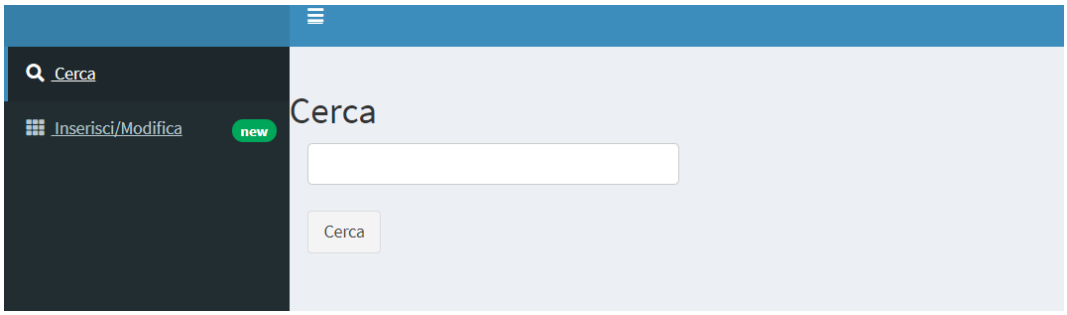

Figura 3.5: Schermata Tab Ricerca (secondo sprint)

Questa è la pagina iniziale: l'utente appena apre la pagina web viene reindirizzato nella pagina Cerca: questo è molto importante in quanto l'utente ha bisogno di capire subito dove andare e cosa fare.

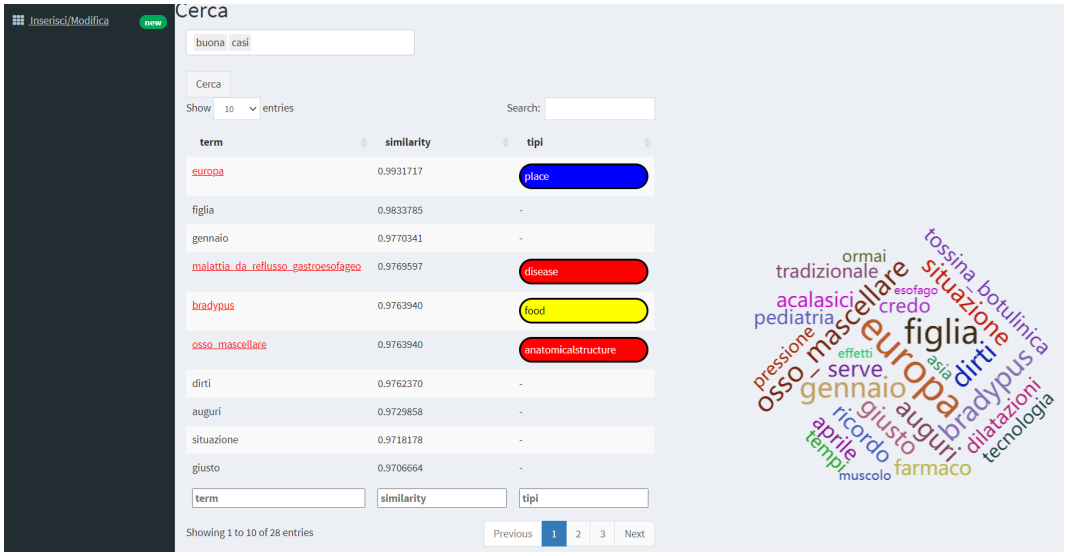

Figura 3.6: Rappresentazione della frase cercata

Quando l'utente ha cercato le parole e ha premuto cerca gli viene mostrato una tabella e la wordcloud. Si è deciso anche di aggiungere delle classi di appartenenza per certe parole e di aggiungere il link a Wikidata.

Chiarimenti e aggiustamenti da eseguire

Ora si è deciso di modificare ancora l'applicazione per rendere ancora più user-friendly e ricca di informazioni:

- La ricerca non deve avvenire attraverso parole ma si deve dare all'utente la possibilità di inserire anche una frase

- Creare un sezione con dei filtri per filtrare al meglio le ricerche
- Creare grafici 2d e 3d per rendere più appetibile agli utenti la ricerca

Anche in questo caso si è deciso la durata dello sprint scegliendo giorno e ora del prossimo incontro.

3.2.4 Terzo Sprint

Esposizione del Prototipo

Nella riunione si è iniziato parlando delle novità:

- La ricerca ora è stata potenziata attraverso dei filtri.
- Sono stati creati nuovi grafici sia 2d che 3d
- Sono state create nuove schede per avere una grafica più pulita e chiara

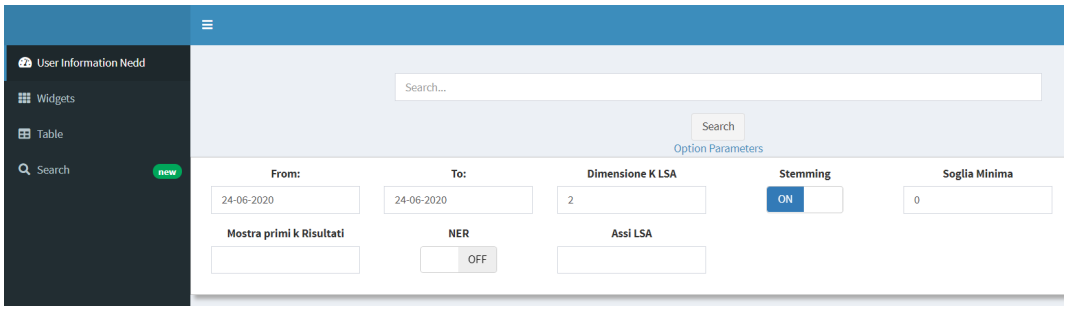

Figura 3.7: Opzioni avanzate di ricerca

Ora il testo da inserire può essere anche una frase (prima si potevano solo scegliere delle parole), inoltre si è aggiunta la possibilità di perfezionare la ricerca attraverso parametri. In questo caso i parametri sono scelti di default da parte del programmatore ma cliccando l'apposito bottone si aprirà un menù di impostazione.

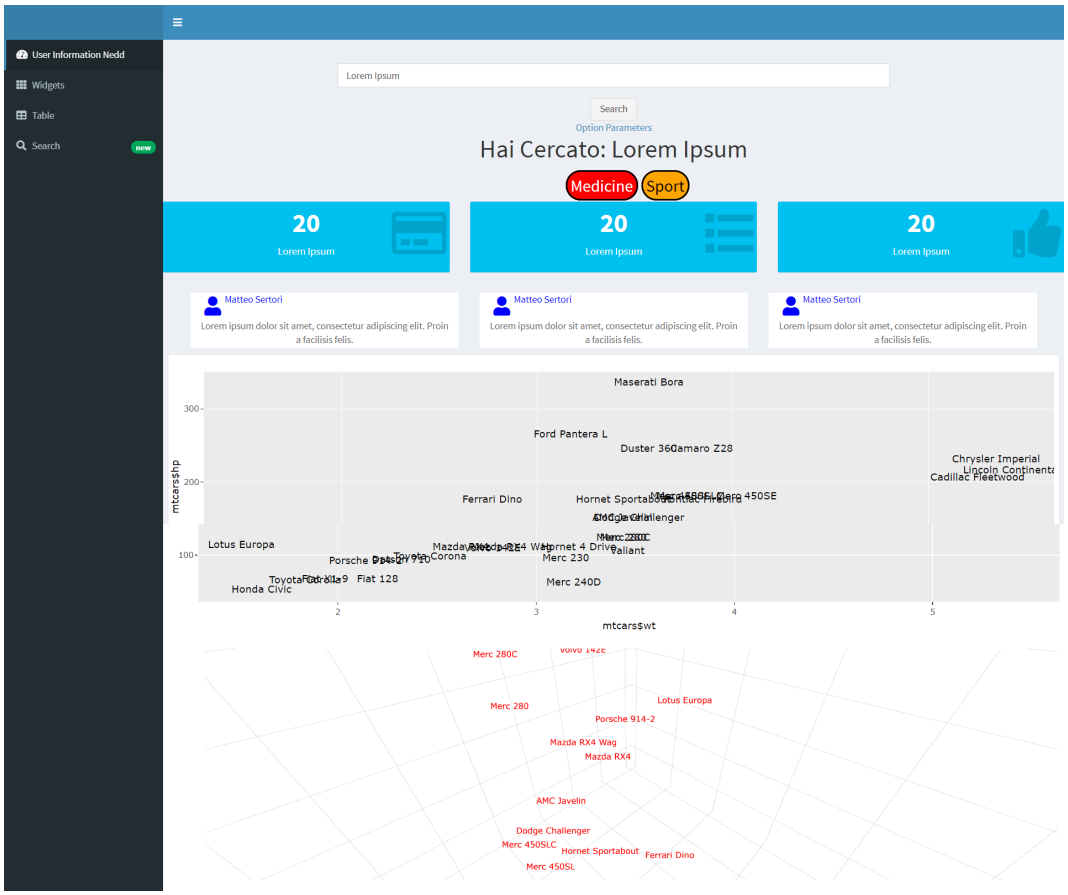

Figura 3.8: Rappresentazione della frase cercata (terzo sprint)

Quando si cercherà un parola di primo impatto comparirà un loading visuale. É stato inserito per avere un feedback visivo su quello che si sta facendo: se non fosse presente l'utente non capirebbe cosa sta succedendo e potrebbe anche cambiare pagina impaziente di non aver avuto nessun riscontro immediato. Nella schermata comparirà:

- La frase o la parola che si è cercata
- Categorie a cui appartiene la frase cercata
- Eventuali box nei quali si possono inserire valori importanti della ricerca
- I post che rappresentano la ricerca effettuata.
- Due grafici che rappresentano i dati cercati.

Si può notare che i grafici sono tutti interattivi: l'utente può zoomare, spostare e scegliere qualsiasi porzione di grafico a suo piacimento. Questo è importante

in quanto in grafici di grandi dimensioni si ha sempre il rischio ti incappare in grafici la cui lettura è di difficile incomprensione.

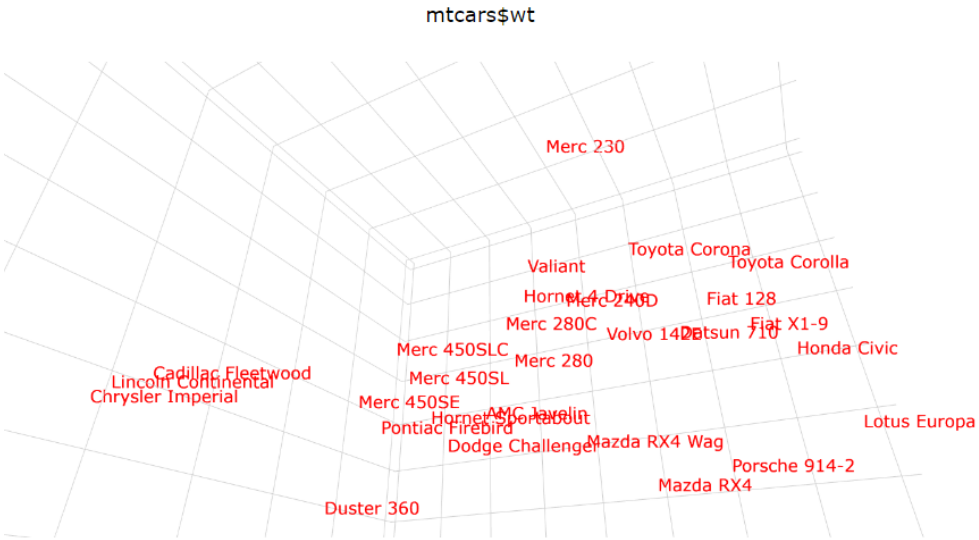

Figura 3.9: Esempio di visualizzazione del grafico 3D

La sezione "widget" contiene altri grafici, anche loro importanti ma si è comunque deciso di metterli in secondo piano sempre per la questione di lasciare solo nella pagina di apertura i dati pilastro.

Figura 3.10: Visualizzazione di WordCloud e Istogrammi

E infine si è creata una sezione dove l'utente può usare per ricercare in modo più veloce le frasi. Ci sono due colonne la colonna di sinistra rappresenta le ricerche più popolari fatti degli utenti mentre nella colonna di destra le ultime ricerche fatte dall'utente stesso.

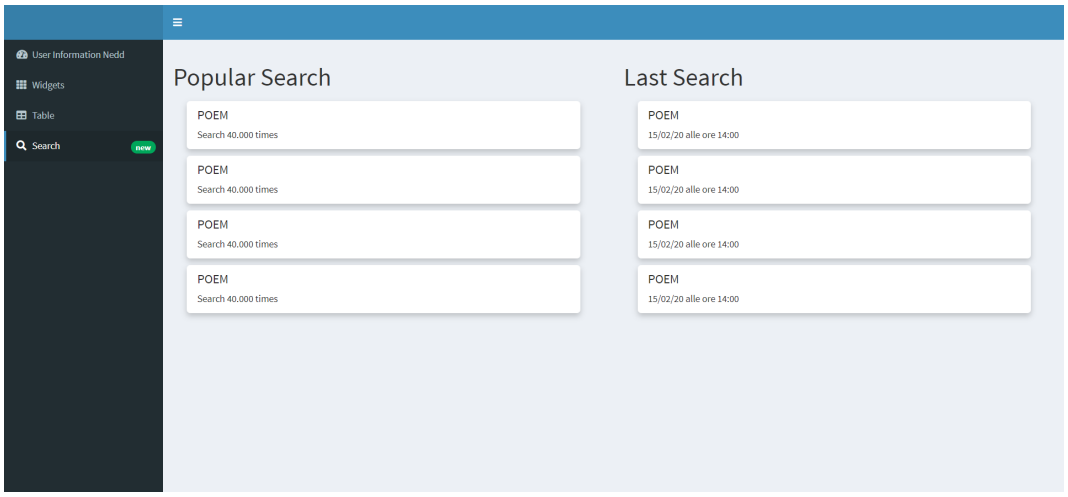

Figura 3.11: Rappresentazione delle ricerche popolari e delle ultime ricerche

Non si è ancora però potuto eseguire la ricerca attraverso suggerimenti in quanto ancora non si è arrivata a una soluzione adeguata per il progetto: si è deciso di tenerla in standby per una futura release del prodotto.

3.2.5 Quarto Sprint

Con il quarto sprint è avvenuta l'integrazione con lápp vera e propria: attraverso l'aiuto di Giacomo (creatore delláp sottostante) si è riusciti a rendere reale tutta lápplicazione che fino ad ora era solo un prototipo. A questo punto si è deciso di iniziare a eseguire tutti i passaggi per la pubblicazione sul Web dell'applicazione.

Capitolo 4

Il codice prodotto

4.1 App.r

```
##########################################
## Restore session with loaded packages
##########################################

# Suppress console warnings printing about packages R version
library(session)
suppressWarnings(
  tryCatch(
    restore.session("sessions/packages_env_loaded.Rda"),
    error=function(e){}))


########################
## Load Packages
########################

source("shinyapp_packages.r")


########################
## Load Analysis Backend
########################

source("analysis_backend.r")
source("analysis_backend_visual.r")
```

```
########################
## Shiny App Definition
########################

source("ui.r", local = TRUE)
source("server.r")

shinyApp(ui = ui, server = server)
```

4.2 Server.r

```
server <- shinyServer(function(input, output) {
  show_modal_spinner(spin = "fading-circle",
                     color = "#112446",
                     text = "Initalizes default latent semantic
                         space...") # show loading on modal window

  # Initalizes default latent semantic space
  ss <- initPrecalculatedSemanticSpace()
  # The application must not be available until this initial
      pre-calculation is complete (around 10 secs)

  remove_modal_spinner() # remove it when done

  ##############################
  #  SEARCH ADVANCED OPTIONS #
  ##############################

  observeEvent(input$option, {
    toggle(
      id = 'advaced_option',
      anim = TRUE,
      animType = "slide",
      time = 0.5
    )
  })


  ####################
  #  SEARCH RESULTS #
  ####################
```

```
observeEvent(input$search, {
  ######################
  # QUERY
  ######################

  show_modal_spinner(spin = "fading-circle",
                     color = "#112446",
                     text = "Loading...") # show loading on modal
                         window
  query_res <- makeUserQuery(ss, input$searchText)
  remove_modal_spinner() # remove it when done

  # Show statistics and semantically near posts
  shinyjs::toggle("boxes")
  shinyjs::toggle("post-container")
  shinyjs::toggle("plot1")
  shinyjs::toggle("plot")

  # Show search input
  output$search_word_label <- renderText({
    paste('Hai Cercato:', query_res$new_doc$d)
  })

  View(query_res)

  ######################
  # NEAREST DOCS
  ######################

  # TODO: show nearest docs
  v <- list()
  v <- lapply(1:length(query_res$nearest_docs), function(i) {

      column(4,
             tags$div(
               class = "post",
               column(12,
                      tags$div(
                        class = "post-content",
                        tags$span(query_res$nearest_docs[[i]])
                      ))
             ))
  })
```

```
  output$post <- renderUI(v)

#####################
# 2D LSA
#####################

# Get colors associated to pre-calculated opinion classes
opinion_classes_cols <-
  getColsForOpinionClasses(opinion_classes)
terms_visual <-
  getTermsRepresentationWithNER(ss$voc$terms_without_types,
      ss$voc$terms_classes)

tlsn <- normRows(ss$lsa_data$tls)

plot(tlsn, pch = 20)

output$scatterPlot <- renderPlotly({

  tlsn <- normRows(ss$lsa_data$tls)

  plot_ly(x = tlsn[,1], y = tlsn[,2], type = 'scatter',
                mode = 'text', text = terms_visual$terms_labels,
                    textposition = 'middle right',
                textfont = list(color = '#000000', size = 16))

})

#####################
# WIDGETS
# ------------------
# MAKE QUERY
#####################

output$histogram <- renderPlotly({
  fig <- plot_ly(
    type = 'histogram',
    marker = list(color = 'green'),
    x = ~ rnorm(100, 5),
    name = "rnorm(100, 5)"
  )

  fig <- fig %>% add_trace(
```

```
      type = 'histogram',
      x = ~ rnorm(40, 5),
      marker = list(color = 'red'),
      name = "rnorm(40, 5)"
    )


    fig <- fig %>% add_trace(
      type = 'histogram',
      x = ~ rnorm(20, 5),
      marker = list(color = 'gray'),
      name = "rnorm(20, 5)"
    )

    fig <- fig %>% layout(barmode = "stack",
                          bargap = 0.1)

    ######################
    # WORD CLOUDS
    ######################
    wordcloud <- getWordCloudDataFromTdm(ss$tdmw)
    renderWordcloud("wordcloud1", data = wordcloud)
    #renderWordcloud("wordcloud2", data = wordcloud)


    fig
  })

  ######################
  # WORD CLOUDS
  ######################

  wordcloud <-
    data.frame(name = mtcars$names, value = mtcars$mpg)
  renderWordcloud("wordcloud1", data = wordcloud)
  renderWordcloud("wordcloud2", data = wordcloud)

})

})
```

4.3 Ui.r

```
ui <- shinyUI(dashboardPage(

  #############
  #  HEADER #
  #############

  dashboardHeader(),

  ##############
  #  SIDEBAR #
  ##############

  dashboardSidebar(sidebarMenu(
    menuItem(
      "User Information Need",
      tabName = "user_information_need",
      icon = icon("dashboard")
    ),
    menuItem("Widgets", tabName = "widgets", icon = icon("th")),
    menuItem("Table", tabName = "table", icon = icon("table")),
    menuItem(
      "Search",
      tabName = "popular_search",
      icon = icon("search"),
      badgeLabel = "new",
      badgeColor = "green"
    )
  )),

  ###########
  #  BODY  #
  ###########

  dashboardBody(
    loadEChartsLibrary(),
    useShinyjs(),
    tags$head(includeCSS(path = "www/style.css")),

    tabItems(

      ################################
```

```
#  B1. USER INFORMATION NEED #
##############################

tabItem(
  tabName = "user_information_need",

  fluidRow(

    # Search Bar

    column(
      12,
      textInput(
        inputId = "searchText",
        label = "",
        value = "",
        width = "800px",
        placeholder = "Search..."
      ),
      actionButton("search", "Search"),
      #uiOutput("var1_select")

    ),

    # Advanced Options

    align = "center",
    column(12,
           actionLink("option", "Option Parameters")),
    column(
      12,
      fluidRow(
        class = "advaced_option",
        id = 'advaced_option',
        column(
          2,
          dateInput(
            'fromDate',
            'From:',
            value = NULL,
            format = "dd-mm-yyyy",
            startview = "month"
          )
        ),
```

```
      column(
        2,
        dateInput(
          'toDate',
          'To:',
          value = NULL,
          format = "dd-mm-yyyy",
          startview = "month"
        )
      ),
      column(2,
             numericInput('dim_k', 'Dimensione K LSA', 2)),
      column(
        2,
        tags$label("Stemming", class = "control-label"),
        switchInput(inputId = "stemming", value = FALSE)
      ),
      column(2,
             numericInput('soglia_min', 'Soglia Minima', 0)),
      column(2,
             textInput('sim_cos', 'Similarit Coseno')),
      column(2,
             textInput('k_ris', 'Mostra primi k Risultati')),
      column(
        2,
        tags$label("NER", class = "control-label"),
        switchInput(inputId = "ner", value = FALSE)
      ),
      column(2,
             textInput('assi_lsa', 'Assi LSA'))

    )
  ),
  fluidRow(id = "search-word",
           h1(textOutput(
             "search_word_label"
           )),
           h3(htmlOutput("tagSearchWord"))),
  ####################
  # POST
  ###################
  column(
    12,
    fluidRow(
```

```
            class = "post-container",
            id = "post-container",
            uiOutput("post")
          ) %>% shinyjs::hidden(),
        ),
        column(
          12,
          fluidRow(class = "plot1", id = "plot1",
                   plotlyOutput("scatterPlot")) %>% shinyjs::hidden()
        ),
        column(
          12,
          fluidRow(class = "plot", id = "plot",
                   plotlyOutput('plot')) %>% shinyjs::hidden(),
        )

      )
    ),
    #################
    #  B3. WIDGETS #
    #################

    tabItem(tabName = "widgets",
           fluidRow(
             column(
               6,
               tags$div(id = "wordcloud1", style =
                   "width:100%;height:700px"),
               deliverChart(div_id = "wordcloud1")
             ),
             column(
               6,
               tags$div(id = "wordcloud2", style =
                   "width:100%;height:700px"),
               deliverChart(div_id = "wordcloud2")
             ),
             column(12,
                    plotlyOutput('histogram'))
           ))
    )
  )
 )
))
```

Conclusioni e sviluppi futuri

Questa tesi ha presentato i principali approcci utilizzabili per rendere un'app web appetibile e soprattutto utilizzabile dagli utenti. La possibilità di utilizzare in modo semplice e intuitivo un'applicazione è diventato fondamentale soprattutto in questo periodo, attraverso l'uso massivo di Internet, dove molti utenti che fino ad ora erano estranei al mondo del web sono stati catapultati dentro anche senza volerlo.

Si è partiti con lo studio dell'User Experience Design, passando dalla usabilità alla Human Computer Interaction: passi fondamentali per rendere un'applicazione appetibile e usabile anche dai non esperti del settore.

Si è passati poi all'analisi del progetto, allo studio delle possibili metodologie di lavoro e alla realizzazione passo per passo dell'applicativo.

Si è dimostrato che attraverso un'attenta analisi delle tecnologie che l'uso del web non è vincolante solo alla creazione di pagine web, ma bensì anche alla creazione di Web App di ogni genere.

La potenza di questo progetto è quella di essere in grado, dopo opportune modifiche, di ospitare qualsiasi ambito di studio: rendere l'applicazione riusabile è forse uno dei punti forti che qualsiasi applicazione dovrebbe avere.

Questa tesi è un trampolino di lancio, seppur ancora acerba, per creare applicativi in grado di rispondere a ogni domanda che un utente potrebbe chiedere in riferimento a una determinata malattia. Uno dei futuri sviluppi sicuramente potrebbe essere quello di progettare un portale attraverso il quale gli utenti siano raggruppati per aree d'interesse e si aiutassero ad arricchire le informazioni.

Ringraziamenti

Questa tesi di laurea coincide per me con il raggiungimento di uno dei più importanti obiettivi della mia vita. Essere qui a scrivere oggi i ringraziamenti è una grande vittoria! Desidero ringraziare, quindi, tutti coloro che hanno permesso e favorito il raggiungimento di tale obiettivo.

Non è facile citare e ringraziare, in poche righe, tutte le persone che hanno contribuito alla nascita e allo sviluppo di questo elaborato: chi con una collaborazione costante, chi con un supporto morale o materiale, chi con consigli e suggerimenti o solo con parole di incoraggiamento.

Spero di non dimenticare nessuno, ma se tu, mio caro lettore, non dovessi trovare il tuo nome nelle prossime righe... dimmelo e ti ringrazierò di persona, anche per il solo fatto che tu abbia interesse a leggere.

Il primo ringraziamento va al relatore di questo lavoro, la Prof.sa Antonella Carbonaro e i correlatori il Prof. Gianluca Moro e l'Ing. Giacomo Frisoni, che hanno reso possibile tutto ciò e ha acceso il mio personale interesse nei confronti di questa splendida disciplina.

Ringrazio infinitamente i miei genitori che mi hanno sempre sostenuto, appoggiando ogni mia decisione, fin dalla scelta del mio percorso di studi. Senza il supporto morale dei miei genitori, non sarei mai potuto arrivare fin qui. Grazie per esserci sempre stati soprattutto nei momenti di sconforto.

Ringrazio Studio Azione: Marcello, Davide, Ramona e Giorgia che in questi 3 anni mi hanno supportato e sopportato al lavoro aiutandomi negli ultimi mesi quando ancora vedevo lontano questo importante obiettivo.

Ringrazio tutti i miei amici in particolare il gruppo #Einstein il quale lo considero la mia seconda famiglia.

Ringrazio la Sole che è sempre presente e so che c'è sempre nei momenti di bisogno.

Ringrazio Sami che lo considero come un fratello.

Ringrazio Francesco che anche se è lontano è come se fosse sempre qua a sostenermi.

Ringrazio la mia fidanzata Giulia per avermi trasmesso la sua immensa forza e il suo coraggio. Grazie per tutto il tempo che mi hai dedicato. Grazie perché ci sei sempre stata.

E infine... un grazie di cuore a Luca Genghini e Masi Elisabetta ai quali dedico anche la tesi, con cui ho condiviso l'intero percorso universitario. È grazie a loro che ho superato i momenti più difficili: sono diventati due delle persone a cui tengo più di tutti. Mi hanno fatto arrabbiare, sorridere, piangere ma comunque siamo sempre rimasti uniti. Abbiamo vissuto 5 anni praticamente insieme tra lezioni, esami e viaggi in treno. Abbiamo fatto di tutto e non nego che mi manca troppo quei giorni. Purtroppo ora le nostre strade si sono divise ma spero veramente con tutto il cuore di ritornare a fare team con loro.

Senza di loro io oggi non sarei qui.

Bibliografia

[1] Luis Torres Melgarejo, Claudia Zapata Del Río, and Eder Quispe Vilchez. Exploring the relationship between web presence and web usability in peruvian universities. In Aaron Marcus and Wentao Wang, editors, *Design, User Experience, and Usability. Practice and Case Studies - 8th International Conference, DUXU 2019, Held as Part of the 21st HCI International Conference, HCII 2019, Orlando, FL, USA, July 26-31, 2019, Proceedings, Part IV*, volume 11586 of *Lecture Notes in Computer Science*, pages 596–615. Springer, 2019.

[2] Aaron Bangor, Philip T Kortum, and James T Miller. An empirical evaluation of the system usability scale. *Intl. Journal of Human–Computer Interaction*, 24(6):574–594, 2008.

[3] Marc Hassenzahl, Kai Eckoldt, and Meinald T. Thielsch. User experience und experience design - konzepte und herausforderungen. In Henning Brau, Sarah Diefenbach, Marc Hassenzahl, Kirstin Kohler, Franz Koller, Matthias Peissner, Kostanija Petrovic, Meinald T. Thielsch, Daniel Ullrich, and Dirk Zimmermann, editors, *Berichtband des siebten Workshops des German Chapters der Usability Professionals Association e.V., Usability Professionals 2009, Berlin, Germany, September 6-9, 2009*, pages 233–237. Fraunhofer Verlag, 2009.

[4] Leah Reeves, Jennifer C. Lai, James A. Larson, Sharon L. Oviatt, T. S. Balaji, Stéphanie Buisine, Penny Collings, Philip R. Cohen, Ben Kraal, Jean-Claude Martin, Michael F. McTear, T. V. Raman, Kay M. Stanney, Hui Su, and Qian Ying Wang. Guidelines for multimodal user interface design. *Commun. ACM*, 47(1):57–59, 2004.

[5] Daniel J. Steinbock and Sridhar Rao. User experience design patterns for pseudo-sentient agents. In Regina Bernhaupt, Florian 'Floyd' Mueller, David Verweij, Josh Andres, Joanna McGrenere, Andy Cockburn, Ignacio Avellino, Alix Goguey, Pernille Bjøn, Shengdong Zhao, Briane Paul Samson, and Rafal Kocielnik, editors, *Extended Abstracts of the 2020*

CHI Conference on Human Factors in Computing Systems, CHI 2020, Honolulu, HI, USA, April 25-30, 2020, pages 1–4. ACM, 2020.

[6] Gabor Aranyi and Paul van Schaik. Testing a model of user-experience with news websites. *J. Assoc. Inf. Sci. Technol.*, 67(7):1555–1575, 2016.

[7] Róbert Móro, Jakub Daráz, and Mária Bieliková. Visualization of gaze tracking data for UX testing on the web. In Federica Cena, Altigran Soares da Silva, and Christoph Trattner, editors, *Hypertext 2014 Extended Proceedings: Late-breaking Results, Doctoral Consortium and Workshop Proceedings of the 25th ACM Hypertext and Social Media Conference (Hypertext 2014), Santiago, Chile, September 1-4, 2014*, volume 1210 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2014.

[8] Sandra Sproll, Matthias Peissner, Christina Sturm, and Michael Burmester. UX concept testing: Integration von user experience in frühen phasen der produktentwicklung. In Henning Brau, Sarah Diefenbach, Katharina Göring, Matthias Peissner, and Kostanija Petrovic, editors, *Jahresband 2010 der Workshops der German Usability Professionals' Association e.V., Usability Professionals 2010, Duisburg, Germany, September 12-15, 2010*, pages 195–200. Fraunhofer Verlag, 2010.

[9] Tony Hey, Stewart Tansley, Kristin Tolle, et al. *The fourth paradigm: data-intensive scientific discovery*, volume 1. Microsoft research Redmond, WA, 2009.

[10] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2019.

[11] RFfSC R Core Team et al. R: A language and environment for statistical computing, 2013.

[12] TEAM RCore. R: a language and environment for statistical computing. r foundation for statistical computing, vienna, austria, 2016.

[13] Chris Beeley. *Web application development with R using Shiny*. Packt Publishing Ltd, 2013.

[14] Gail Potter, Jimmy Wong, Irvin Alcaraz, Peter Chi, et al. Web application teaching tools for statistics using r and shiny. *Technology Innovations in Statistics Education*, 9(1), 2016.

[15] Diana M Selfa, Maya Carrillo, and M Del Rocio Boone. A database and web application based on mvc architecture. In *16th International Conference*

on Electronics, Communications and Computers (CONIELECOMP'06), pages 48–48. IEEE, 2006.

[16] Achim Zeileis. Cran task views. *R News*, 5(1):39–40, 2005.

[17] Winston Chang and Barbara Borges Ribeiro. shinydashboard: Create dashboards with 'shiny'. r package version 0.5. 1, 2015.

[18] Carson Sievert. *Interactive Web-Based Data Visualization with R, plotly, and shiny*. CRC Press, 2020.

[19] Hadley Wickham. ggplot2. *Wiley Interdisciplinary Reviews: Computational Statistics*, 3(2):180–185, 2011.