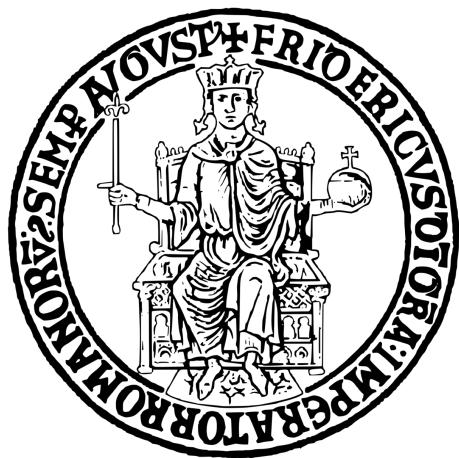


UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO
II

Scuola politecnica e delle scienze di base
Dipartimento di ingegneria elettrica e tecnologie
dell'informazione



Corso di Laurea in Informatica
Tesi di Laurea
Anno Accademico 2020/21

Metodologie CAA, PECS e Realtà Aumentata per
l'Autismo

Candidato:
Quirile Alessandro N86002559

Relatore:
Prof. Cutugno Francesco

Ottobre 2021

*Di cicale scoppiate imagine hanno
versi ch'in laude dei signor si fanno*
- Ludovico Ariosto

Indice

1 Presentazione dell'idea progettuale	8
1.1 Le metodologie impiegate	8
1.1.1 Realtà aumentata	8
1.1.2 Comunicazione aumentativa-alternativa	8
1.1.3 PECS	9
2 Documento dei requisiti Software	10
2.1 Modello funzionale	10
2.1.1 Requisiti funzionali	10
2.1.2 Requisiti non-funzionali	10
2.1.3 Requisiti di dominio	11
2.1.4 Modellazione dei casi d'uso	12
2.1.5 Prototipazione visuale per le principali schermate utente	13
2.1.6 Tabelle di Cockburn	18
2.2 Modelli di dominio	21
2.2.1 Classi, oggetti e relazioni di analisi	21
2.2.2 Sequence Diagram di Analisi	24
2.2.3 Statechart di Analisi	27
2.2.4 Activity Diagram	30
2.3 Impiego delle risorse e pianificazione del lavoro	33
3 Individuazione del target degli utenti	37
3.1 Individuazione di pattern e variabili comportamentali	39
3.2 Personas-type	40
3.2.1 Marco Tulli	41
3.2.2 Giulia Improta	42
3.2.3 Rebecca Caputo	43
3.3 Personas-goal e business goal	44
3.3.1 Primary-personas type goal e business goal	44
3.3.2 Secondary-personas type goal e business goal	45
3.3.3 Negative-personas type e business goal	46
4 Studio della valutazione a priori dell'usabilità	47
5 Documento di Design del Sistema	50
5.1 Analisi dell'architettura	50
5.1.1 Architettura esterna	50
5.2 Class Diagram di Design	55
5.3 CRC Card	56
5.4 Sequence Diagram di Design	63

6 Documento di Testing del sistema	66
6.1 Test Plan per System Testing	66
6.2 Valutazione sul campo dell'usabilità soggettiva e oggettiva	69
6.2.1 Valutazioni euristiche	71
6.2.2 Test di usabilità	72
6.2.3 5-seconds test	74
6.2.4 Analisi di coerenza	75
7 Confronto coi competitor	79
7.1 Chi sono i nostri competitor?	79
7.2 Analisi esterna	79
7.3 Analisi interna	80
7.4 SWOT	80
8 Glossario	82
9 Bibliografia	83

Capitolo 1

Presentazione dell'idea progettuale

Augmentus è un'applicazione per smartphone Android che vuole fornire un supporto alle persone autistiche sfruttando la Realtà Aumentata. Configurandosi come strumento per la Comunicazione Aumentativa Alternativa (CAA), Augmentus è in grado di semplificare ed incrementare la comunicazione e l'interazione per mezzo di espedienti moderni ed accattivanti.

L'applicazione funziona per mezzo di carte fisiche che implementano la metodologia PECS: se scannerizzate da Augmentus, le carte evocano un'animazione in realtà aumentata per incentivare un'azione o per fornire una guida, ad esempio su come apparecchiare la tavola.

Le figure minimali disegnate sulle carte PECS sono complementate dalle animazioni in realtà aumentata. Le carte non solo si configurano come sussidio per la CAA, ma possono essere utilizzate come strumento per la fidelizzazione ed il marketing della Fondazione Istituto Antoniano, essendo brandizzate.

1.1 Le metodologie impiegate

Le metodologie impiegate sono quelle di realtà aumentata e quelle di comunicazione aumentativa-alternativa mediante PECS.

1.1.1 Realtà aumentata

Con la Realtà Aumentata è possibile aggiungere contenuti virtuali in un ambiente fisico reale. Secondo il rapporto *La realtà aumentata e nuove prospettive educative* [1] la Realtà Aumentata è in grado di fornire diversi vantaggi sull'apprendimento grazie alla possibilità di sovrapporre dati digitali (che potremmo definire *irreali*) nel mondo *reale* che ci circonda. In contesti delicati, ad esempio quando l'utente è autistico, è possibile veicolare la sua attenzione soltanto sugli aspetti realmente significativi, ad esempio evidenziando una parte dello scenario e oscurarne un'altra.

1.1.2 Comunicazione aumentativa-alternativa

Per comunicazione aumentativa-alternativa (CAA) si intende una serie di metodologie atte ad *aumentare* oppure *proporre* nuove modalità di comunicazione in base alle caratteristiche del singolo utente avvalendosi di ausili e tecnologie avanzate che fanno da supporto, come spiegato in *Guida pratica per le buone prassi di comunicazione aumentativa-alternativa* [2].

1.1.3 PECS

PECS è l'acronimo di *Picture Exchange Communication System* ovvero *Sistema di comunicazione tramite lo scambio di immagini*, come spiegato nell'articolo *Guida pratica per le buone prassi di comunicazione aumentativa-alternativa* [2]. È una tecnica sviluppata *ad hoc* per il disturbo dello spettro autistico da Lori A. Frost e Andrew S. Bondy. Oltre ad essere poco costosa in termini di risorse, è una soluzione "portatile" nel senso che può essere applicata facilmente in diversi contesti, come per esempio a casa, a scuola, al parco giochi, in spiaggia e così via. Il sistema di calibrazione delle immagini PECS deve basarsi sul grado di riconoscimento della specifica persona: occorre, quindi, verificare ad esempio se il ragazzo riconosce le figure geometriche elementari, se riesce a distinguere chiaramente i colori, oppure ha un grado di riconoscimento ancora maggiore che gli consente di distinguere e identificare immagini geometriche più complesse o addirittura tridimensionali, come cubi o parallelepipedi. La metodologia PECS può essere implementata attraverso vari strumenti: carte, oggetti di varia natura, pupazzi e così via. Quella più usata è l'implementazione tramite carte perché ritenute più economiche e in certi casi più resistente e meno pericolose. In Figura 1.1 vengono mostrate alcune rappresentazioni PECS per l'autismo.



Figura 1.1: Esempi di PECS

Capitolo 2

Documento dei requisiti Software

In questo capitolo verranno analizzate e formalizzate le funzionalità principali, costruendo le gerarchie funzionali del prodotto e dettagliando la pianificazione del lavoro nella sua totalità.

2.1 Modello funzionale

Di seguito è riportata la classificazione dei requisiti ricavati mediante interviste, scenari e ricerche combinati tra loro. I requisiti utenti così raccolti sono stati successivamente analizzati e studiati per trasformarli nei requisiti di sistema.

2.1.1 Requisiti funzionali

- **Scannerizzare una carta** - il Software deve poter scannerizzare una carta a cui è stato associato un file multimediale generico in realtà aumentata;
- **Mostrare il sito web dell'istituto** - il Software deve poter mostrare in maniera semplice il sito web dell'Istituto Antoniano, già sviluppato da terzi;
- **Visualizzare il percorso per raggiungere l'Antoniano** - il Software deve poter mostrare all'utente l'itinerario per raggiungere il semiresidenziale dalla sua posizione.

2.1.2 Requisiti non-funzionali

- **User-centered** - il sistema deve essere volta ad una progettazione user-centered, adatta per l'autismo;
- **Interfaccia mobile** - il Software deve essere disponibile per le interfacce mobili Android;
- **Usabilità** - il Software deve essere facile da usare per l'utente;
- **Compatibilità** - il Software dovrebbe essere compatibile col maggior numero di dispositivi possibile, per abbracciare un pubblico ampio;
- **Velocità di risposta** - il Software dovrebbe rispondere in maniera molto rapida agli input dell'utente; la risposta dovrebbe giungere entro 5 secondi nel 90% dei casi e fornendo feedback adeguati e coerenti;
- **Impiego di un API per le mappe** - per implementare il requisito di calcolo del percorso dalla posizione attuale, il Software dovrà impiegare un'API per le mappe;
- **Impiego di un modulo per l'AR** - il Software deve utilizzare un modulo per gestire la realtà aumentata;

- **Affidabilità su diversa grandezza di dispositivi** - il Software dovrebbe funzionare sia su uno schermo molto piccolo che su uno molto grande, senza perdere in termini di qualità delle interfacce né tantomeno in termini di funzionalità offerte;
- **Coerenza dei colori** - per aumentare il livello di usabilità, il Software dovrebbe utilizzare colori nitidi che non affaticino la vista e che facciano focalizzare l'attenzione dell'utente sulle parti significative dell'applicazione, come determinati pulsanti o scritte;
- **Storage** - il Software deve utilizzare una tecnologia di storage per conservare i file multimediali associati alle carte da scannerizzare;
- **Semplificazione** - è importante semplificare l'animazione 3D in realtà aumentata e, ai fini comunicativi, occorre minimizzare quanto più possibile le immagini sulle carte fisiche PECS;
- **Sequenzialità** - è interessante evitare di rappresentare sulla carta fisica un numero che rappresenti l'ordine di quella azione rispetto alla sequenzialità, in modo tale che sia l'utente a determinarla.

2.1.3 Requisiti di dominio

- **TC 159/SC 4:** il Software deve essere standardizzato rispetto all'ergonomia dell'interazione con l'utente e tutte le persone che lo hanno progettato e mantenuto. Le aree di standardizzazione comprendono l'ergonomia dell'hardware, del software, inclusa la progettazione dello human-computer interaction e i metodi e processi dello human-centered design, inclusa l'ingegneria dell'usabilità e i metodi di progettazione partecipativa;
- **ISO 13407 human-centered design:** il Software va progettato con un'accezione "umano-centrica";
- **Privacy-by-design:** il Software è stato sviluppato in base al Regolamento Europeo sulla Privacy che impone alle organizzazioni e ai sistemi che trattano dati personali in modo che sia conforme da subito alle regole della privacy, spostando la responsabilità del corretto trattamento tramite strumenti informatici idonei sul titolare e sul responsabile del trattamento, quando identificato;
- **UE 679/2016:** il Software dovrà essere conforme ai requisiti imposti dal regolamento UE 679/2016 il quale prevede che tutti i dati personali siano protetti, compresa l'eventuale pseudonominizzazione dei dati personali - quando necessaria - e la cifratura dei dati stessi;
- **ISDP 10003:2015:** il Software dovrà aderire alle norme vigenti EU in tema di trattamenti di dati personali, qualora previsti.

2.1.4 Modellazione dei casi d'uso

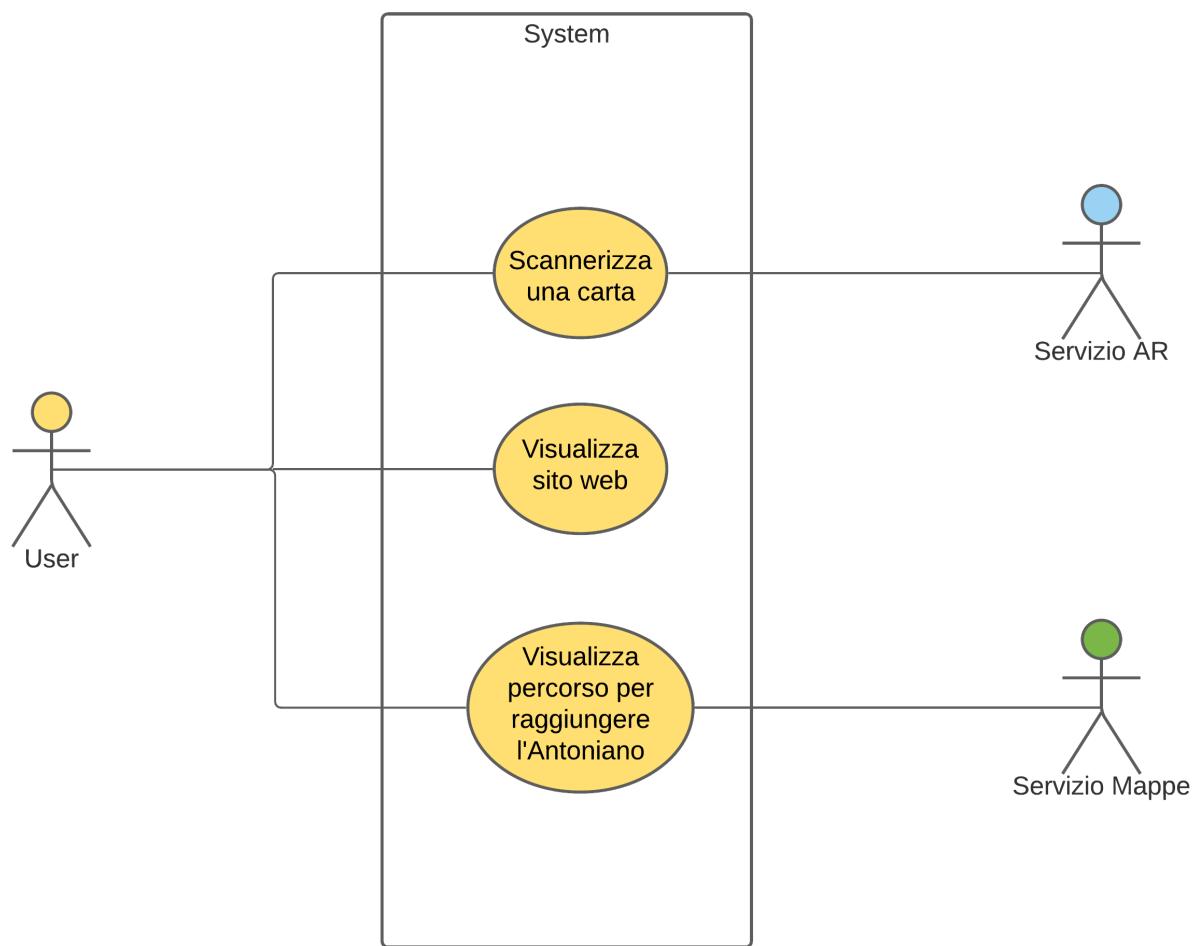


Figura 2.1: Use Case Diagram

2.1.5 Prototipazione visuale per le principali schermate utente

Bassa fedeltà

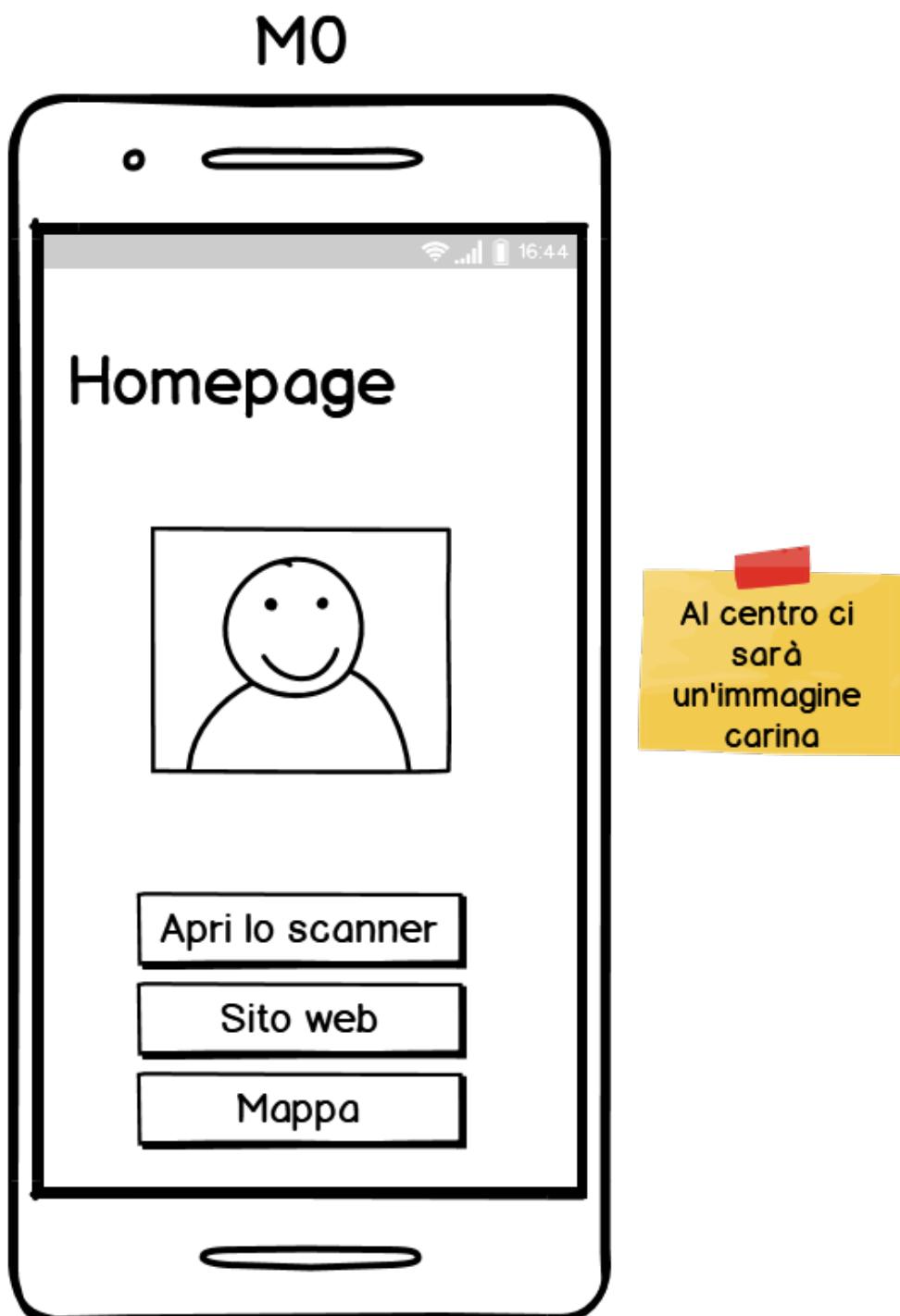


Figura 2.2: Homepage

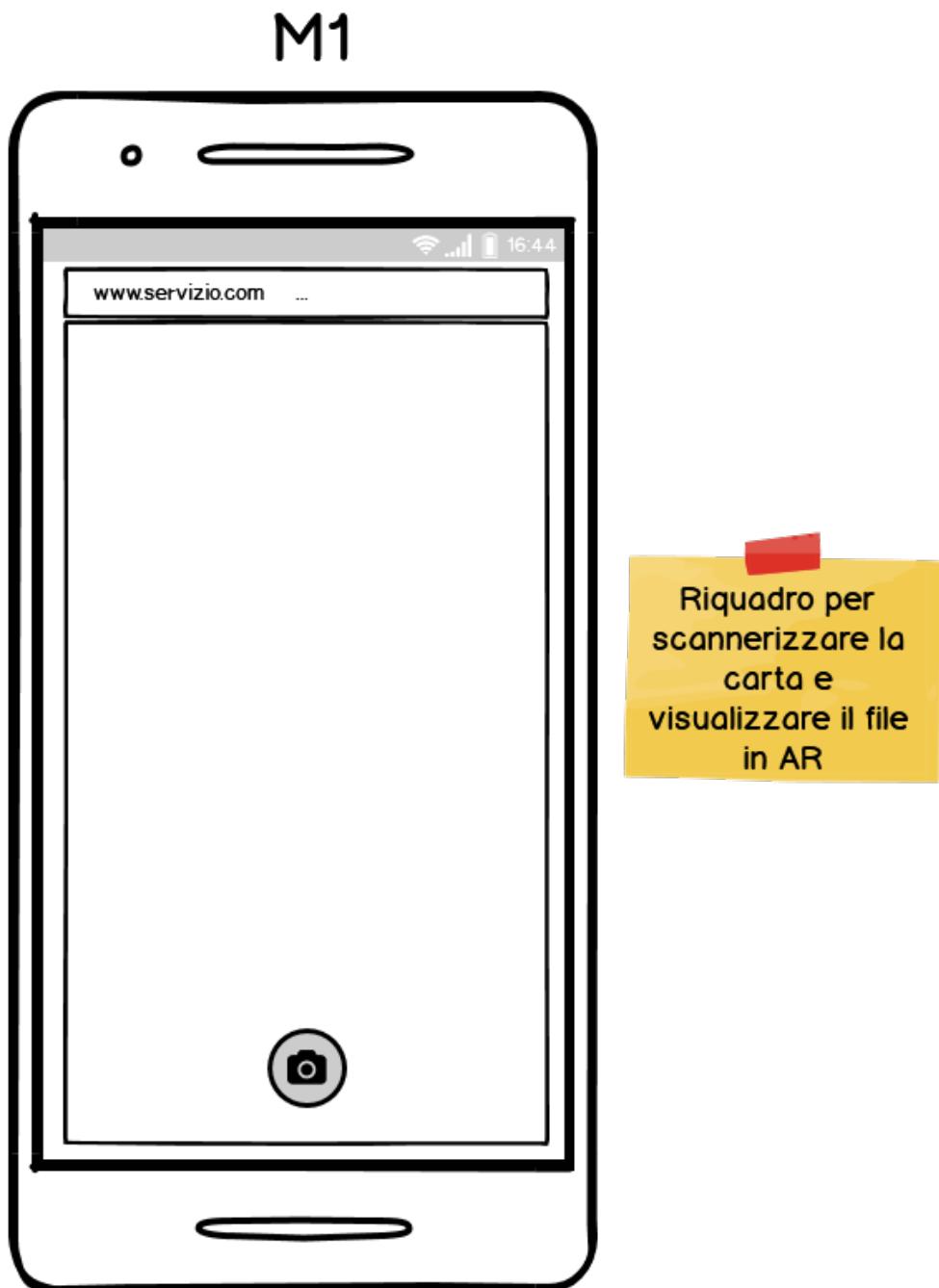


Figura 2.3: Servizio AR

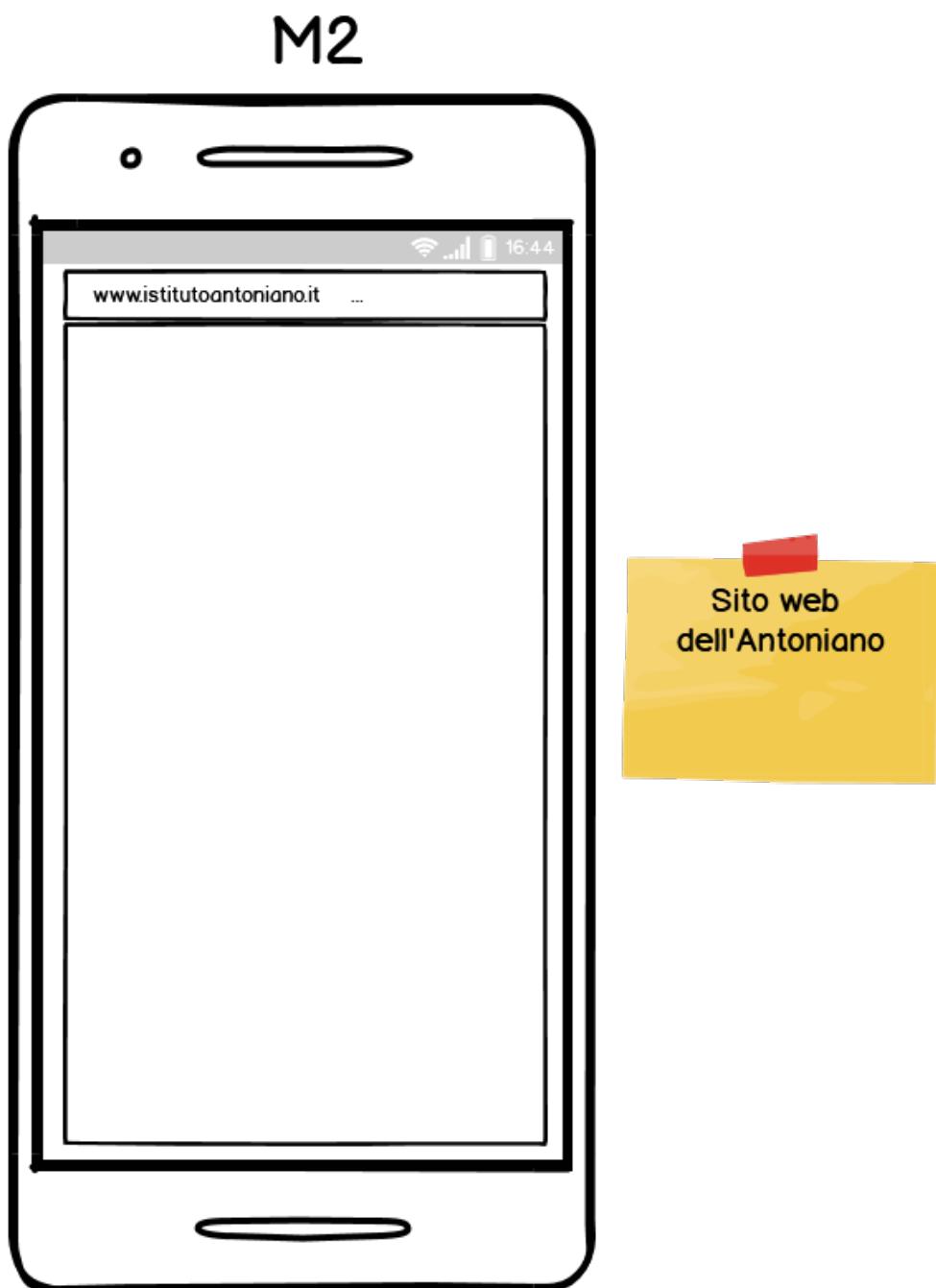


Figura 2.4: Schermata sito web



Figura 2.5: Schermata mappa

Media fedeltà

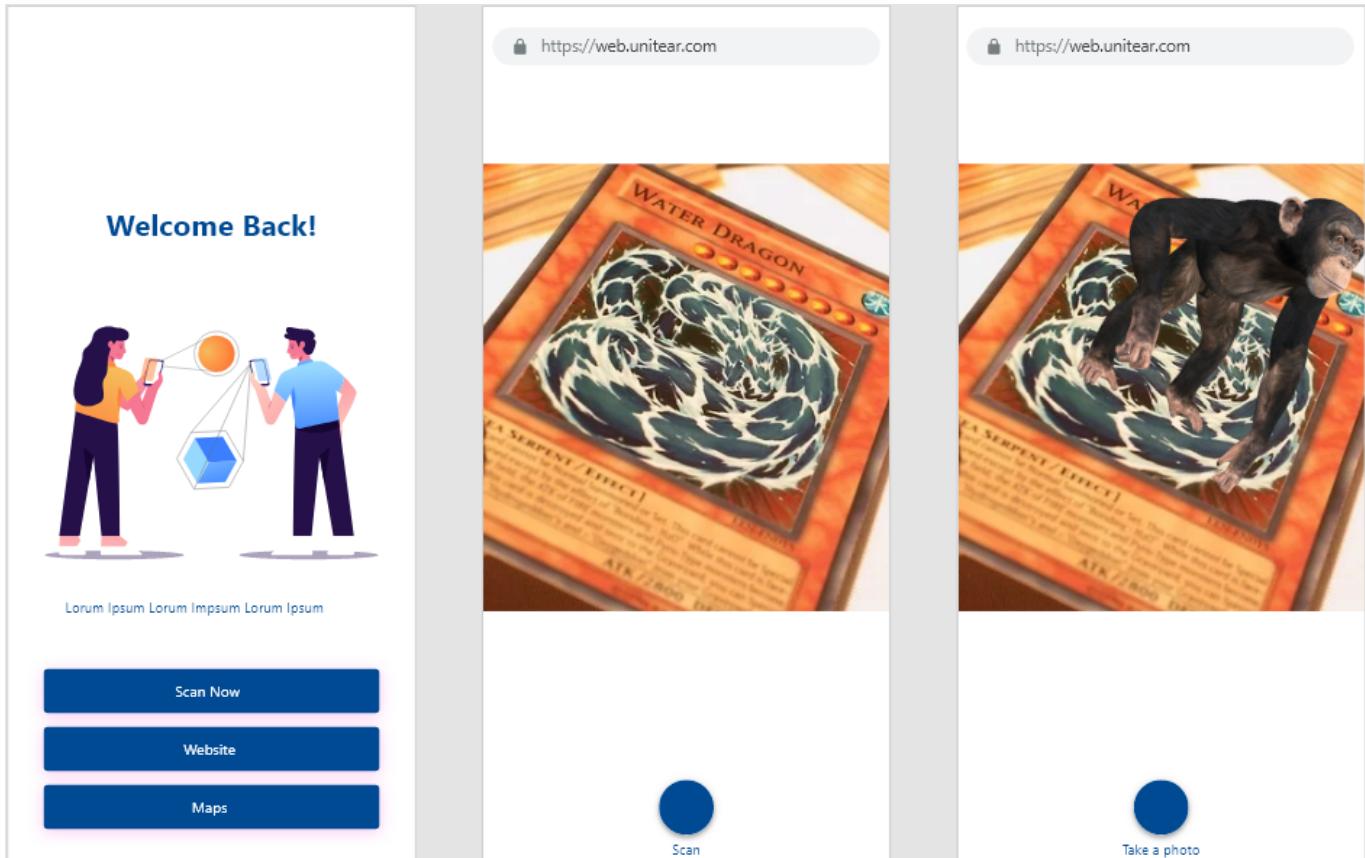


Figura 2.6: Protipazione visuale di media fedeltà delle principali schermate

2.1.6 Tabelle di Cockburn

Di seguito ci sono le tabelle di Cockburn relative alla prototipazione visuale a bassa fedeltà mostrata nel paragrafo precedente.

Tabella 2.1: Scannerizza carta

Use Case	Scannerizza carta			
Goal in Context	Consente ad un utente di scannerizzare una carta			
Preconditions				
Success End Conditions	L'utente scannerizza la carta e visualizza la sua animazione			
Failed End Conditions	L'utente annulla l'operazione			
Primary Actor	Utente			
Secondary Actor	Servizio AR			
Trigger	L'utente apre l'app e clicca sul pulsante "Scan"			
Description	Step	User Action	Secondary actor	System
	1	L'utente apre l'app e clicca sul pulsante "Scan"		
	2			Mostra la pagina per lo scan (M1)
	3	Inquadra la carta e preme "Scan"		
	4		Fornisce il file associato alla carta	
	5			Mostra l'animazione
Extension	Step	User Action	Secondary actor	System
	3.1	Torna indietro		
	4.1			Ritorna alla pagina precedente
Notes	La carta deve avere associato un file in AR			

Tabella 2.3: Visualizza sito web

Use Case	Visualizza sito web		
Goal in Context	Consente ad un utente di visualizzare il sito web dell'Antoniano		
Preconditions			
Success End Conditions	L'utente ha visualizzato il sito web		
Failed End Conditions	L'utente annulla l'operazione tornando indietro		
Primary Actor	Utente		
Trigger	L'utente desidera visualizzare il sito web dell'Antoniano ed apre l'app		
Description	Step	User Action	System
	1	L'utente desidera visualizzare il sito web dell'Antoniano ed apre l'app	
	2		Mostra homepage (M0)
	3	Clicca su "Website"	
	4		Carica la pagina web (M2)
Extension	Step	User Action	System
	3.1	Torna indietro	
	4.1		Ritorna alla pagina precedente
Notes	Il sito web è già precedentemente sviluppato da terze parti		

Tabella 2.5: Visualizza percorso

Use Case	Visualizza percorso per raggiungere l'Antoniano			
Goal in Context	Consente ad un utente di visualizzare il percorso per raggiungere l'Antoniano			
Preconditions	-			
Success End Conditions	L'utente visualizza il percorso per raggiungere l'Antoniano			
Failed End Conditions	L'utente annulla l'operazione			
Primary Actor	Utente			
Secondary Actor	Servizio Mappe			
Trigger	L'utente apre l'app e clicca sul pulsante "Maps"			
Description	Step	User Action	Secondary actor	System
	1	L'utente apre l'app e clicca sul pulsante "Maps"		
	2		Fornisce la mappa	
	3			Mostra la schermata M3
Extension	Step	User Action	Secondary actor	System
	3.1	Torna indietro		
	4.1			Ritorna alla pagina precedente
Notes				

2.2 Modelli di dominio

Questa fase consiste nella rappresentazione del problema in classi UML d'analisi, ovvero indipendenti da scelte tecnologiche ma che hanno l'obiettivo di descrivere il dominio del problema nel modo più minimale possibile. È stata utilizzata l'euristica *3-Objects Type* che consiste nel dividere le classi in tre macrocomponenti ognuna con *una* responsabilità differente: le classi <<Boundary>> rappresentano l'interfaccia o parte di essa, le classi <<Control>> rappresentano la logica di business e le classi <<Model>> rappresentano un'informazione (classi di persistenza). I colori sono stati utilizzati per facilitare la lettura di queste distinzioni.

2.2.1 Classi, oggetti e relazioni di analisi

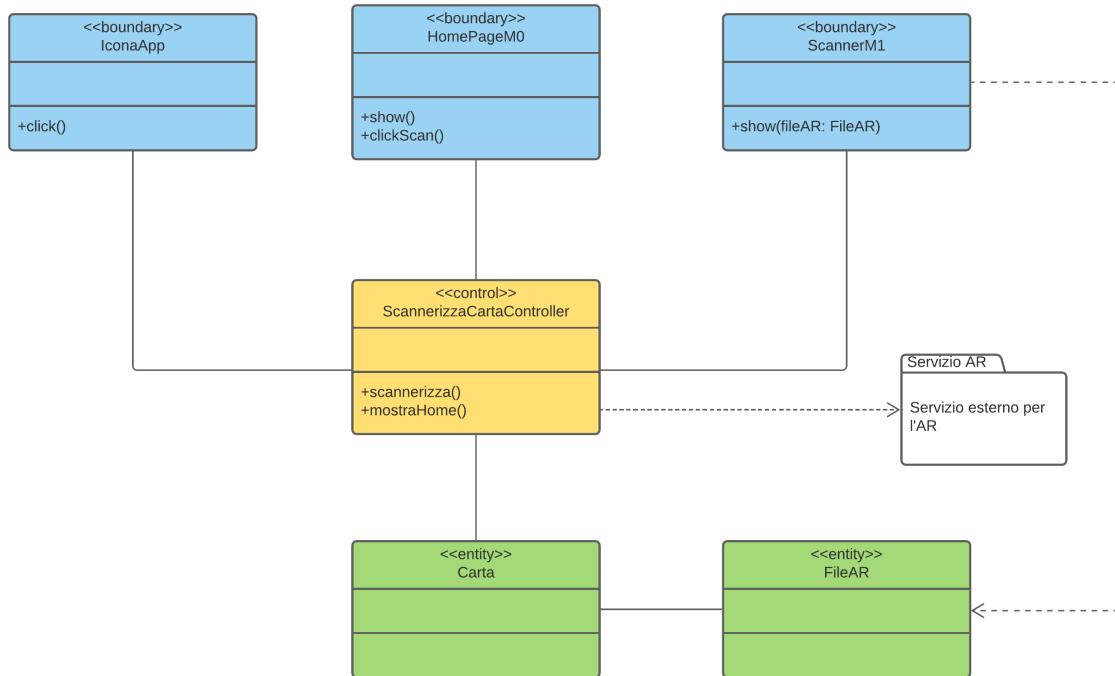


Figura 2.7: Class Diagram di Analisi - Scannerizza carta

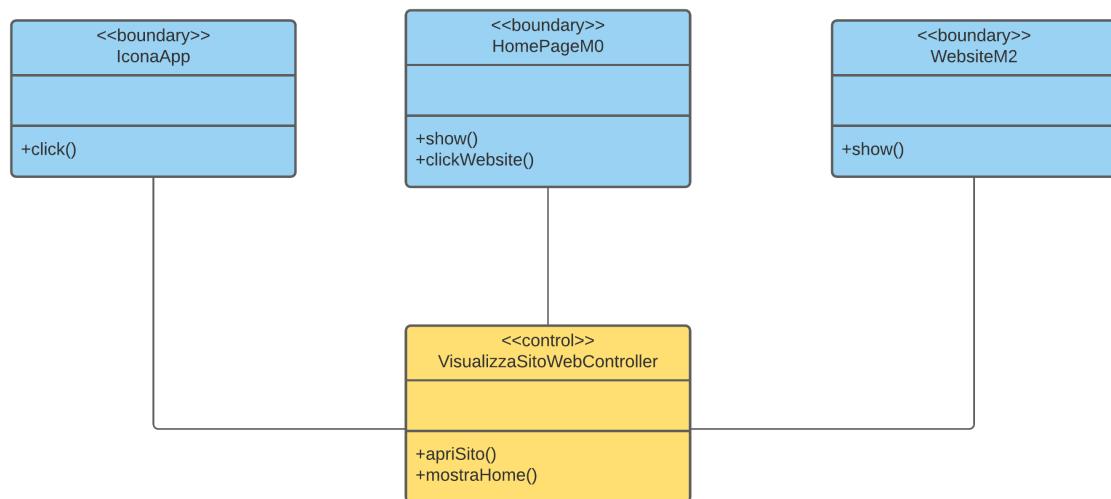


Figura 2.8: Class Diagram di Analisi - Visualizza sito web

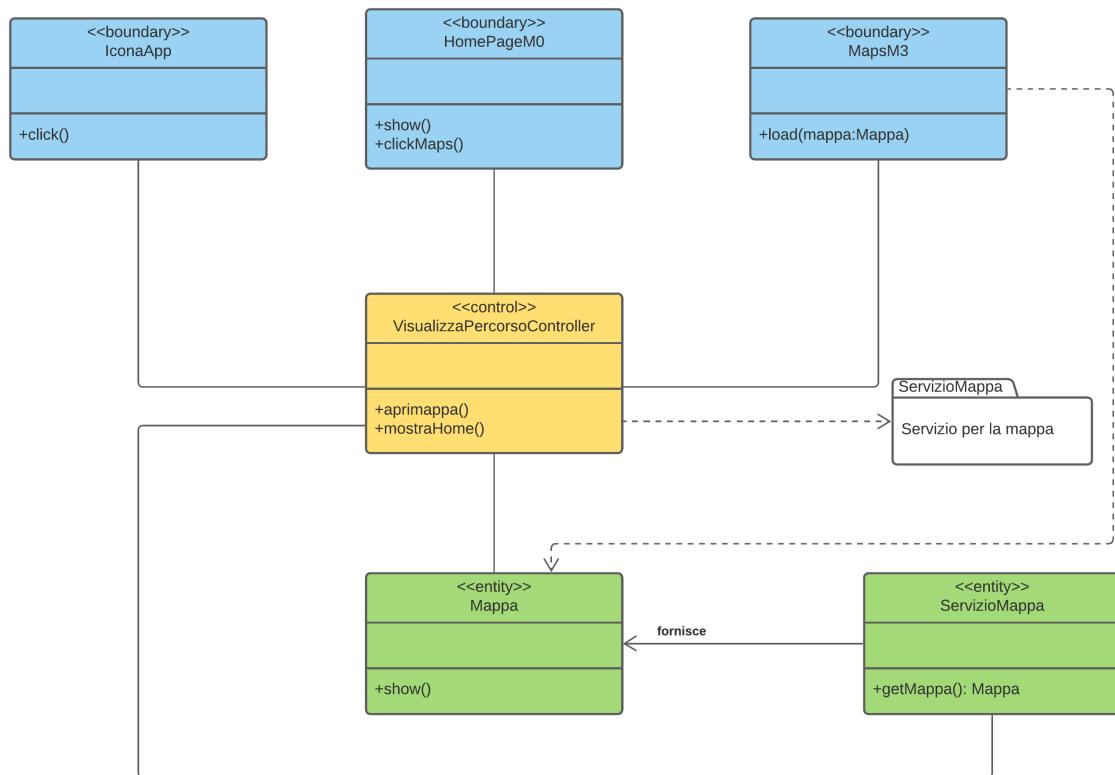


Figura 2.9: Class Diagram di Analisi - Visualizza percorso per raggiungere l'Antoniano

2.2.2 Sequence Diagram di Analisi

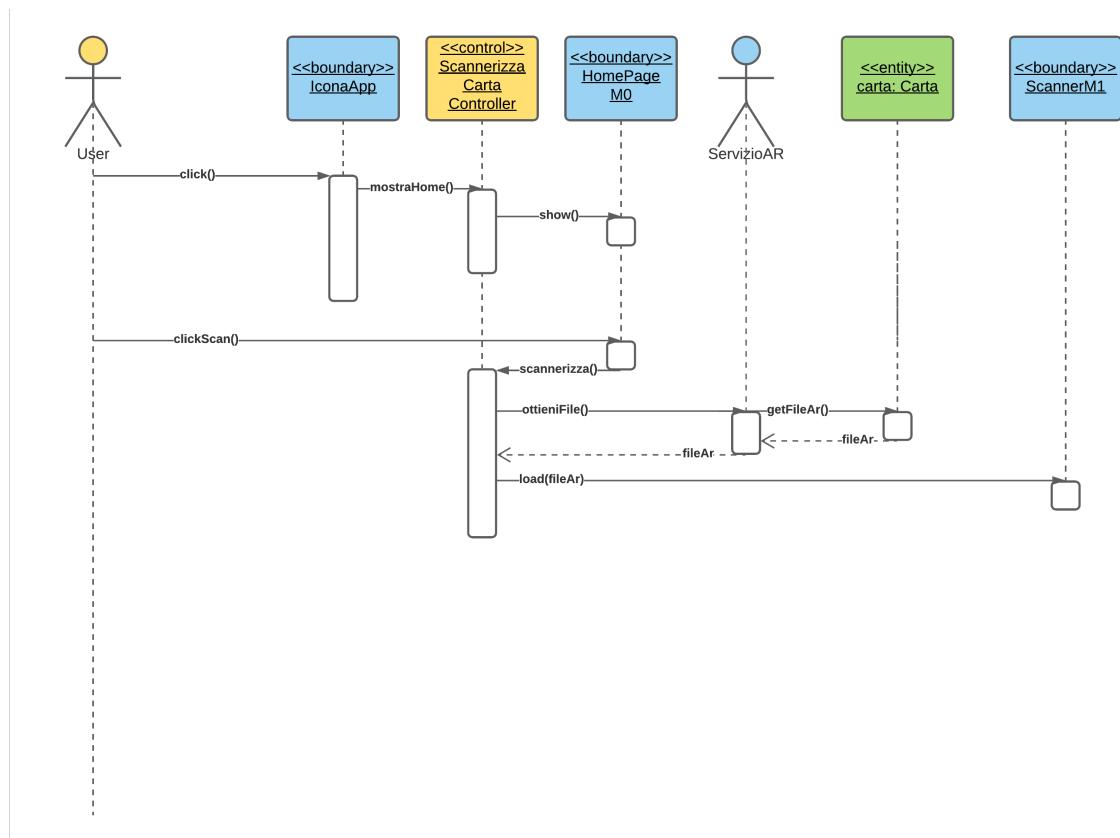


Figura 2.10: Sequence Diagram di Analisi - Scannerizza carta

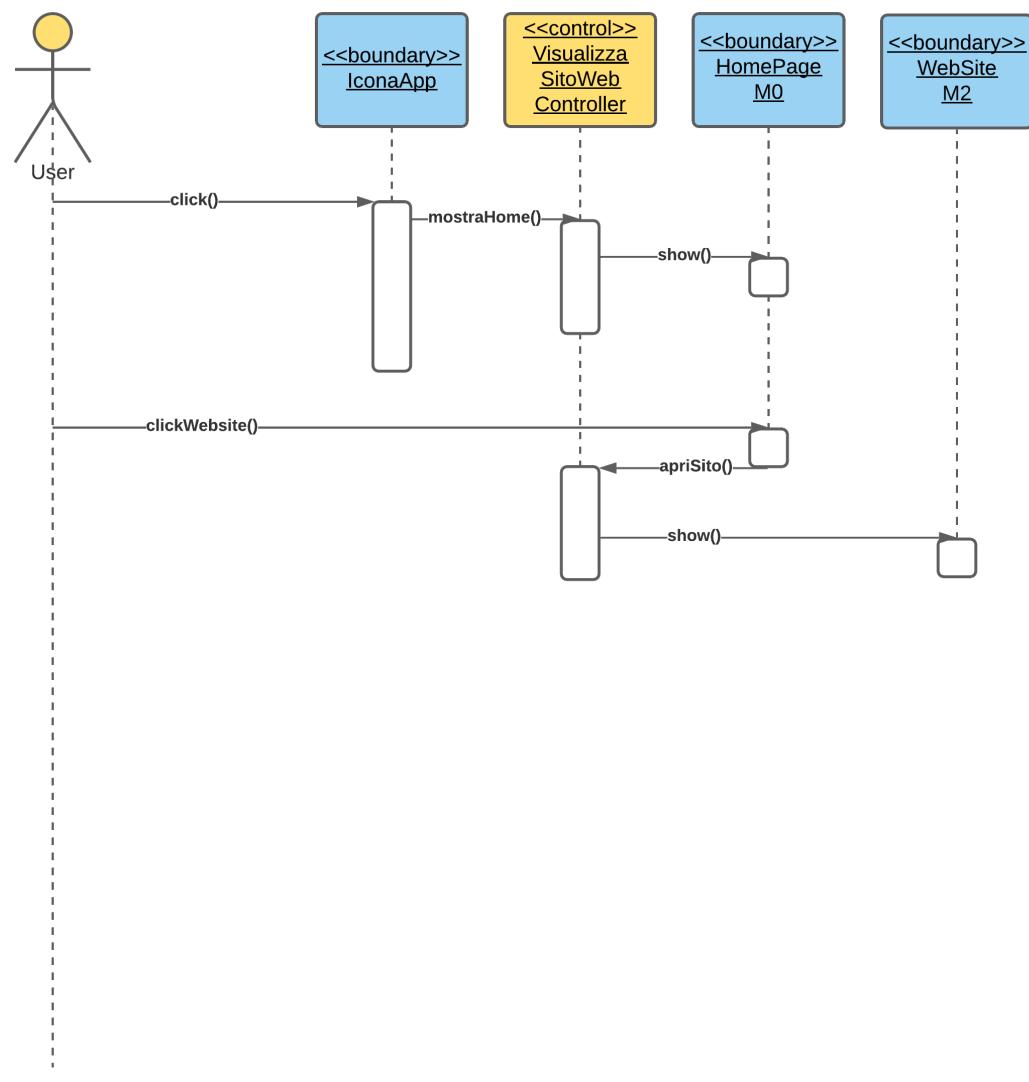


Figura 2.11: Sequence Diagram di Analisi - Visualizza sito web

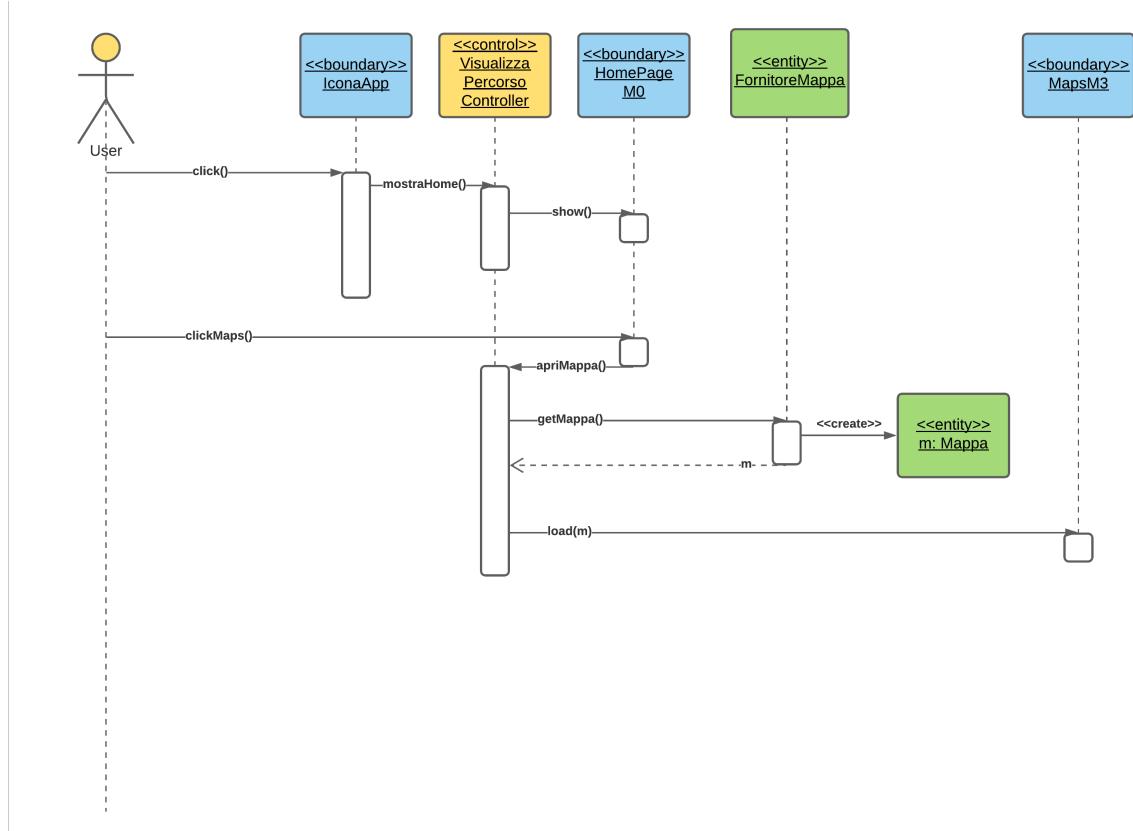


Figura 2.12: Sequence Diagram di Analisi - Visualizza percorso per raggiungere l'Antoniano

2.2.3 Statechart di Analisi

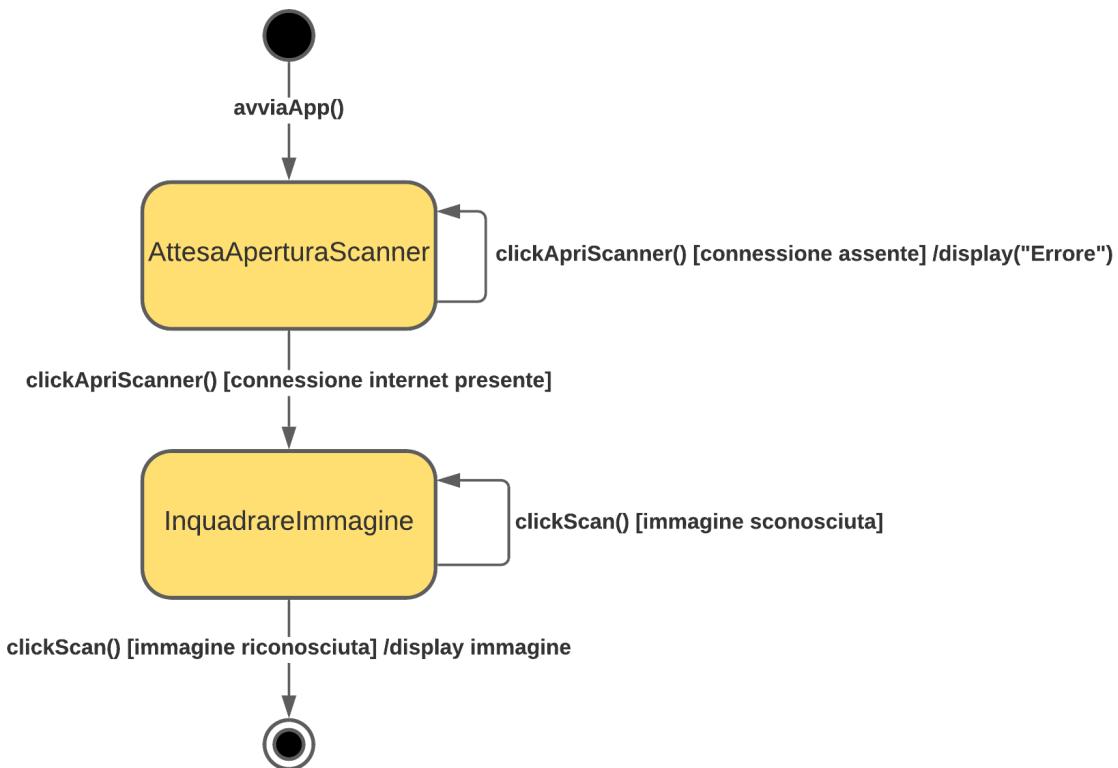


Figura 2.13: Statechart - Scannerizza carta

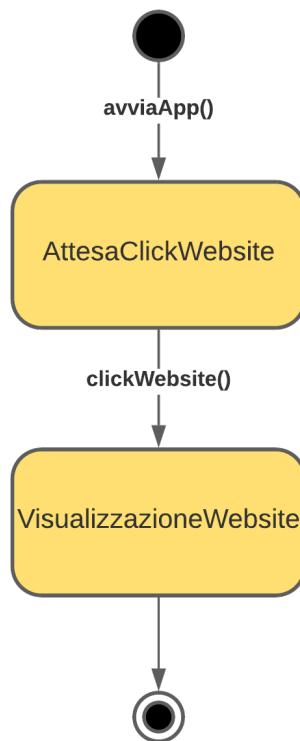


Figura 2.14: Statechart - Visualizza sito web

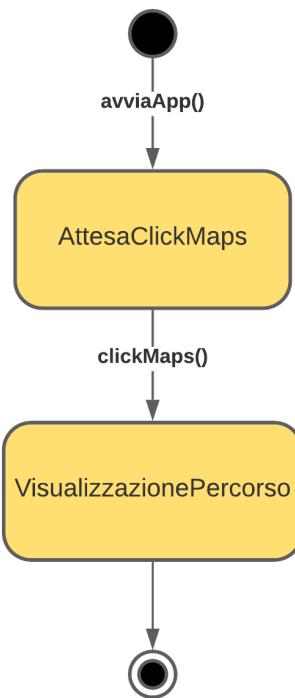


Figura 2.15: Statechart - Visualizza percorso per raggiungere l'Antoniano

2.2.4 Activity Diagram

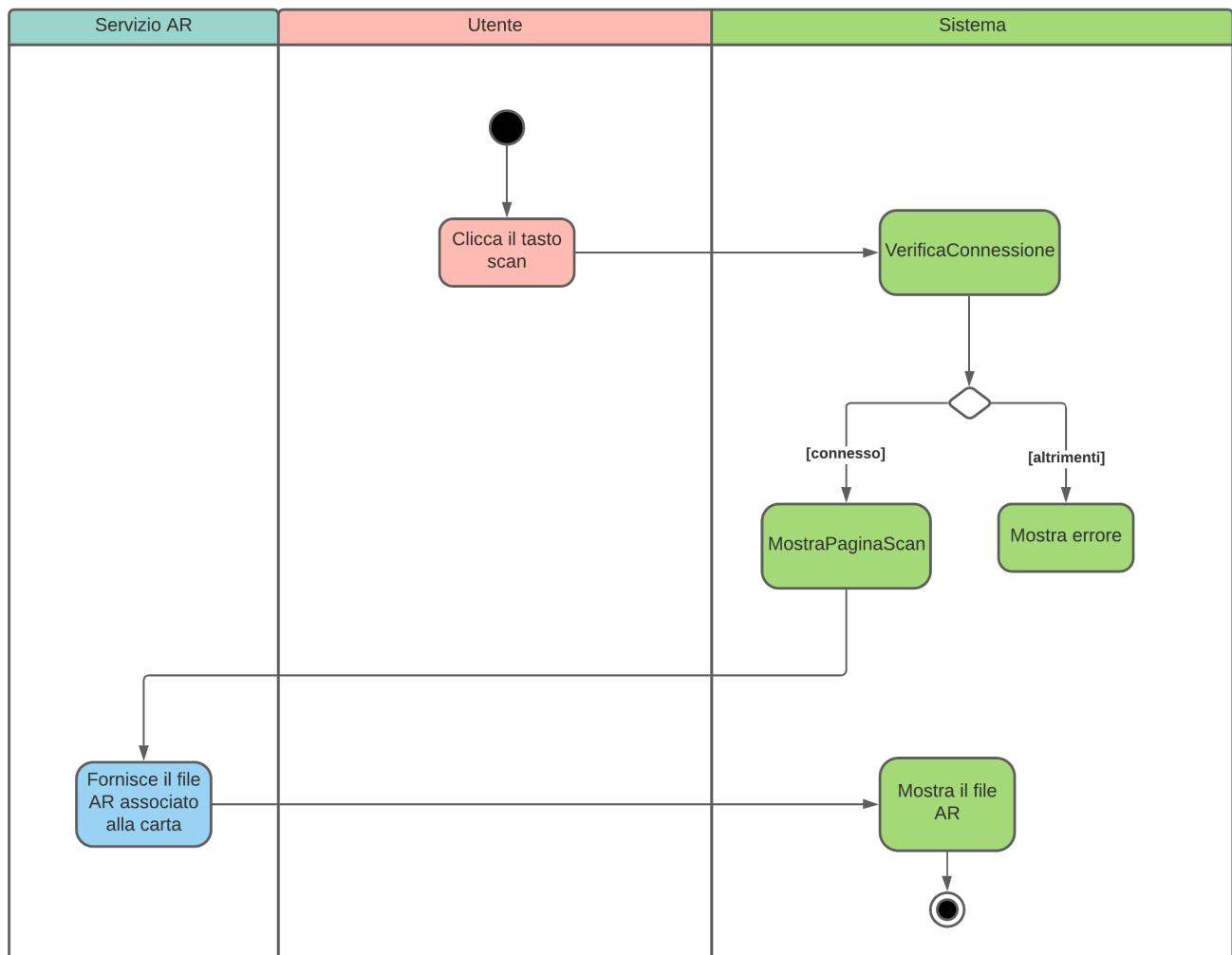


Figura 2.16: Activity Diagram - Scannerizza carta

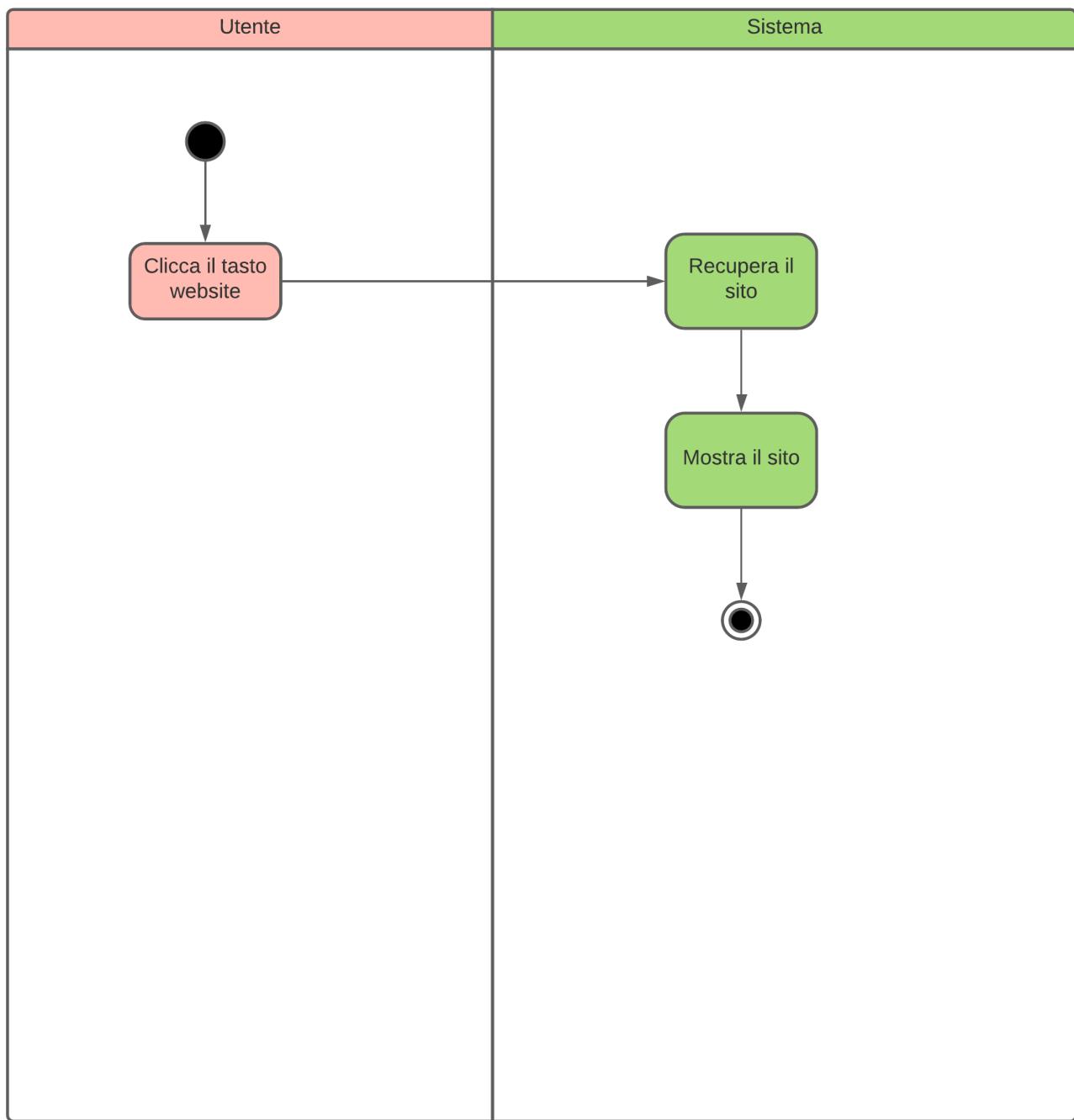


Figura 2.17: Activity Diagram - Visualizza sito web

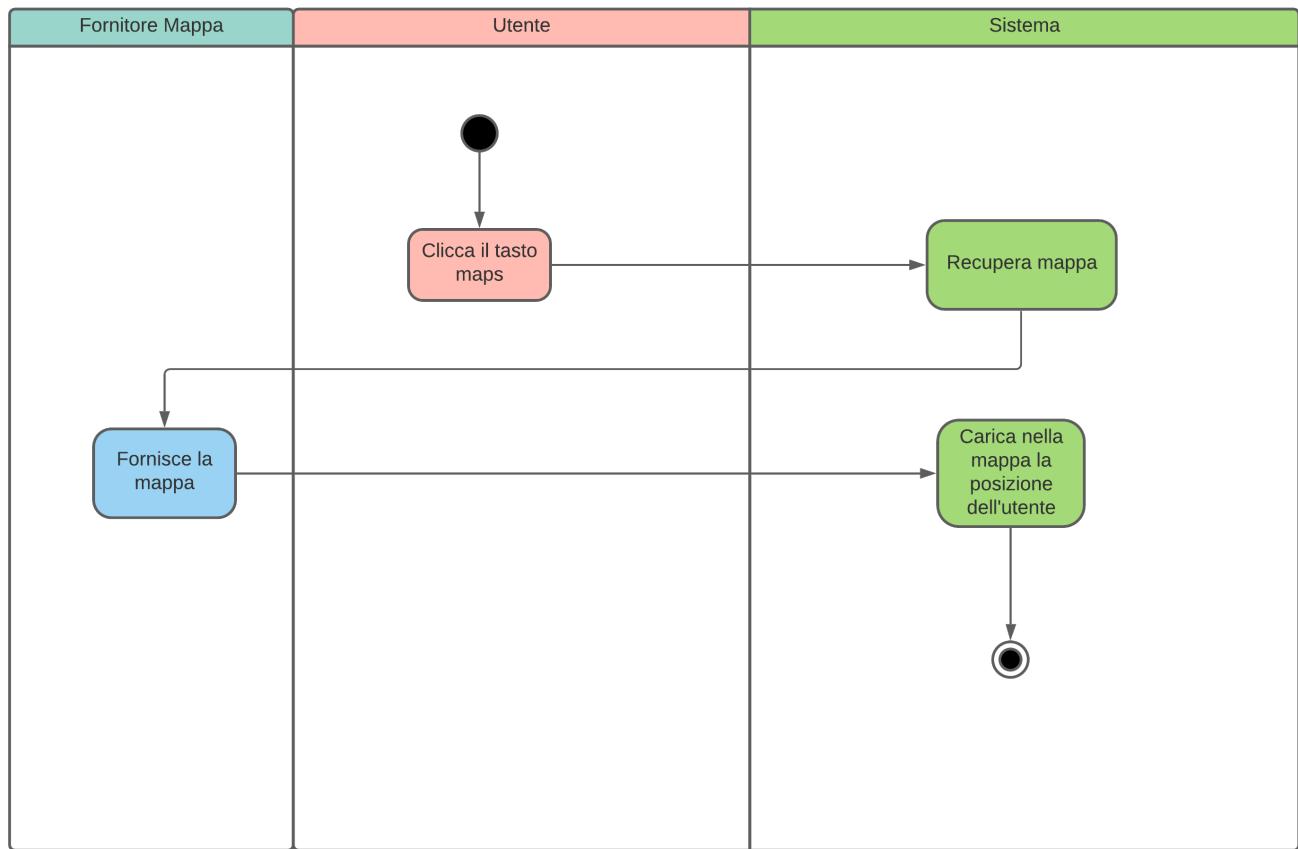


Figura 2.18: Activity Diagram - Visualizza percorso per raggiungere l'Antoniano

2.3 Impiego delle risorse e pianificazione del lavoro

Il Project Management è consistito nella gestione del progetto sia in termini di risorse che in termini di benessere psicologico del team operativo. La gestione verteva sul:

- Definire un obiettivo da raggiungere: concludere il progetto
- Pianificare un modo per conseguirlo: modello a cascata (*waterfall model*)
- Definire risorse: il team ed i costi
- Effettuare checkpoint ovvero date significative
- Effettuare valutazioni periodiche

Le *key-question* a cui il project management ha risposto sono le seguenti:

- Cosa e quando realizzare - bisogna realizzare il progetto entro Ottobre 2021
- Quanto costerà - il minimo possibile
- Come è stato misurato l'andamento del progetto - attraverso apposite metodologie e diagrammi come PERT

Il primissimo passo effettuato è stato scegliere il modello che al meglio potesse rispondere al nostro working plan. È stato scelto il modello a cascata per numerosi motivi:

- Supera l'approccio "code and fix"
- Supera le difficoltà dei modelli agili - ad esempio un modello come SCRUM avrebbe necessitato di un team più grande, di ceremonie e artefatti: troppo complicato
- Gli attributi intrisechi del modello sono ottimali - comprensibilità¹ alta, visibilità² eccellente, supportabilità³ alta, accettabilità⁴ discreta, affidabilità⁵ massima, robustezza⁶ non riesce a gestire modifiche alla traccia, manutenibilità⁷ ottima e rapidità⁸ scarsa

Tuttavia, gli svantaggi di questo modello sono:

- Il modello è sequenziale - ogni fase genera un documento per quella successiva e quindi è molto importante che non si commettano errori fase per fase
- Delicatezza della fase di Requirement Collection - la fase di raccolta dei requisiti col cliente è, nel modello a cascata, estremamente delicata ed andrebbe chiesto al cliente qualsiasi cosa non sia completamente chiara
- Fragile ai cambiamenti di traccia in corso d'opera - nel caso i requisiti cambiassero di molto, si andrebbe in contro a dei rallentamenti rispetto alla data prefissata per il deploy

¹Quanto è facile interpretare il tutto

²A che punto si è con lo sviluppo?

³Quanti strumenti di supporto ci sono?

⁴Le persone del team sono contente nel ruolo assegnato?

⁵Quale sarà la qualità del software prodotto?

⁶Quanto è robusto il software a cambiamenti in corso d'opera?

⁷Quanto è robusto il software a cambiamenti futuri?

⁸Quanto tempo ci vuole per il deploy?

Dopo aver scelto il modello che al meglio si presti alla causa, occorre definire il *Work Breakdown Structure*, ovvero una struttura ad albero che descrive le fasi del progetto e la loro sequenzialità per mezzo di numeri. Esso è rappresentato in Figura 2.19

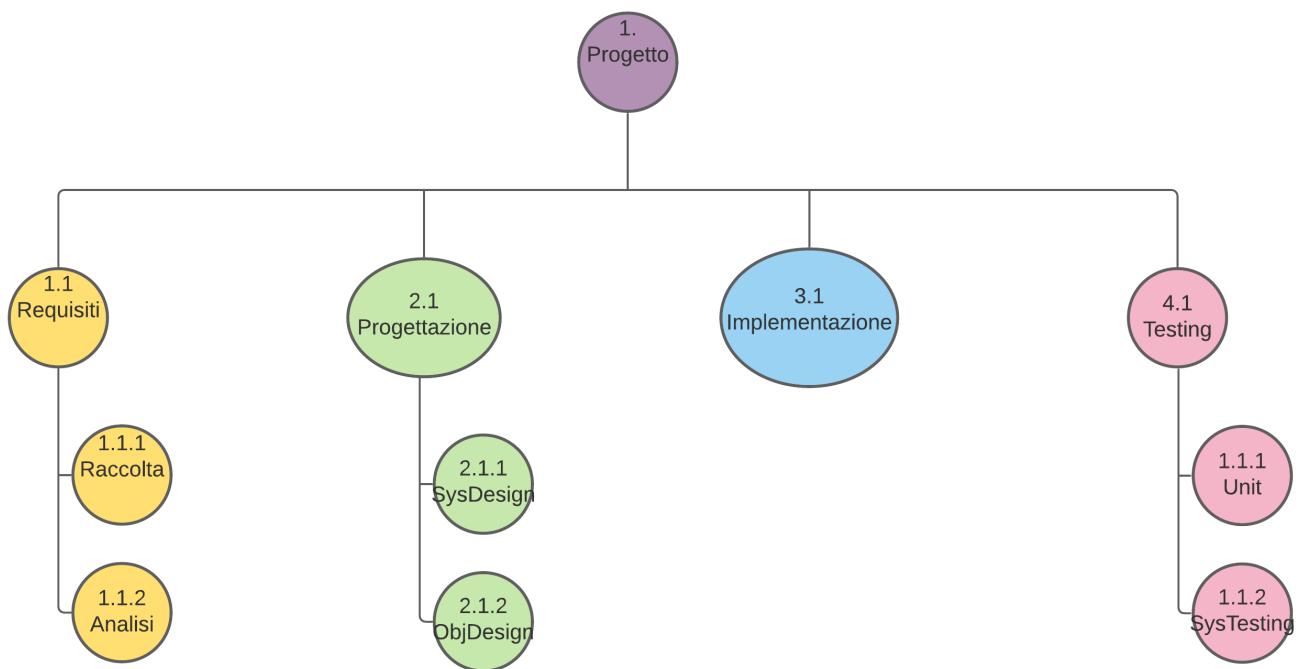


Figura 2.19: Project Management - Work Breakpoint Structure

Di seguito è definito il diagramma di PERT che consente di specificare dipendenze fra *task* e calcolare un percorso critico che non dovrebbe mai subire rallentamenti per non ritardare la data del deploy. Il diagramma di PERT è essenzialmente un grafo orientato che descrive le attività del progetto ed è mostrato in Figura 2.20

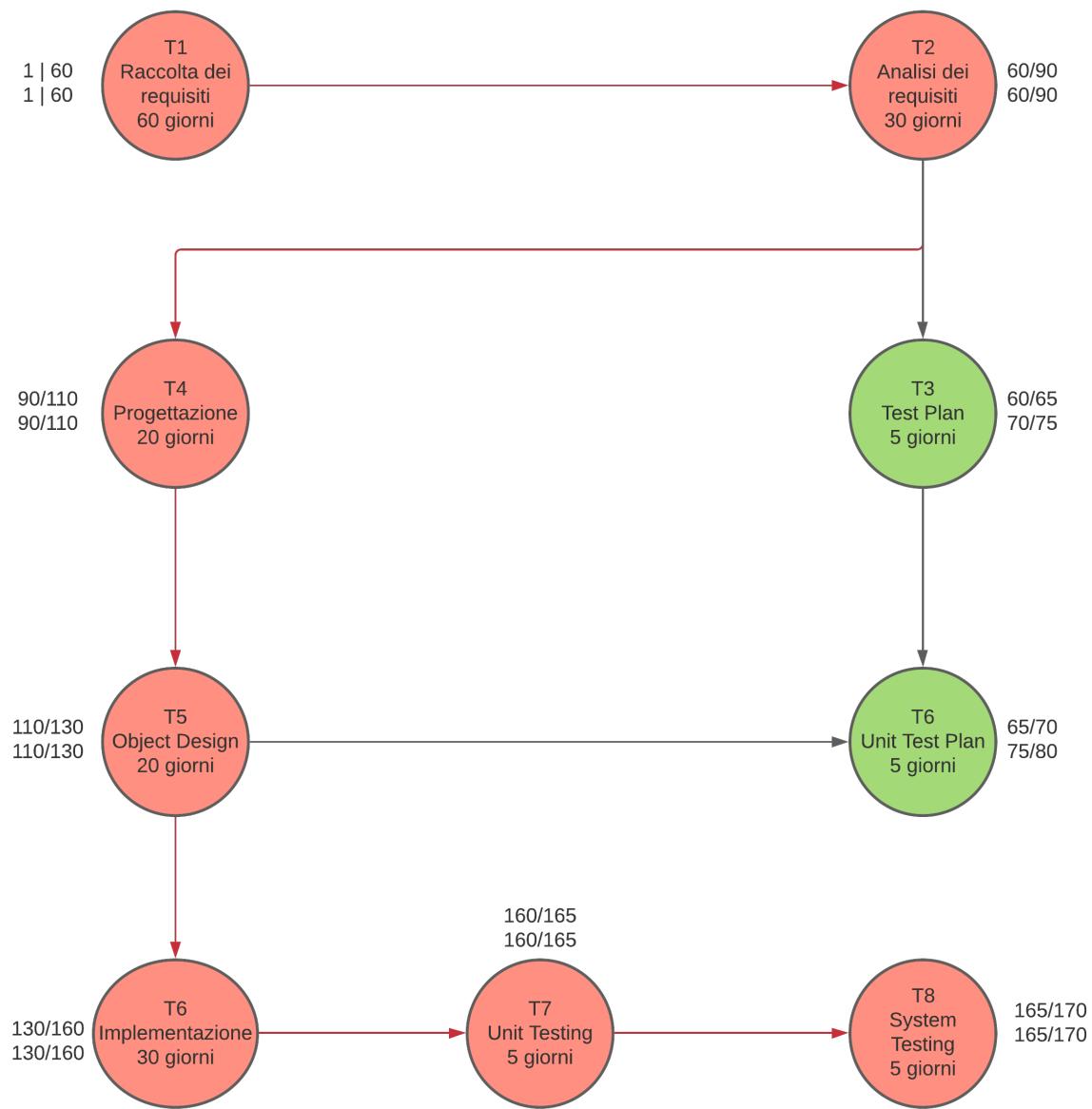


Figura 2.20: Project Management - Diagramma di PERT

I nodi e gli archi marcati in rosso sono quelli che costituiscono il cammino critico, ovvero quel percorso le cui attività non dovrebbero mai essere ritardate e per cui *best case* e *worst case* coincidono.

Come mostrato in Figura 2.21. È importante parallelizzare le fasi costruttive, rappresentate a sinistra, con quelle distruttive, rappresentate a destra, in primis per ragioni di tempo (è impensabile spendere sei mesi, ad esempio, in costruzione ed altri sei in distruzione), ma anche perché - avendo scelto il modello a cascata - era essenziale assicurarsi che non vi fossero errori generati in una fase e immessi in input alla fase successiva.

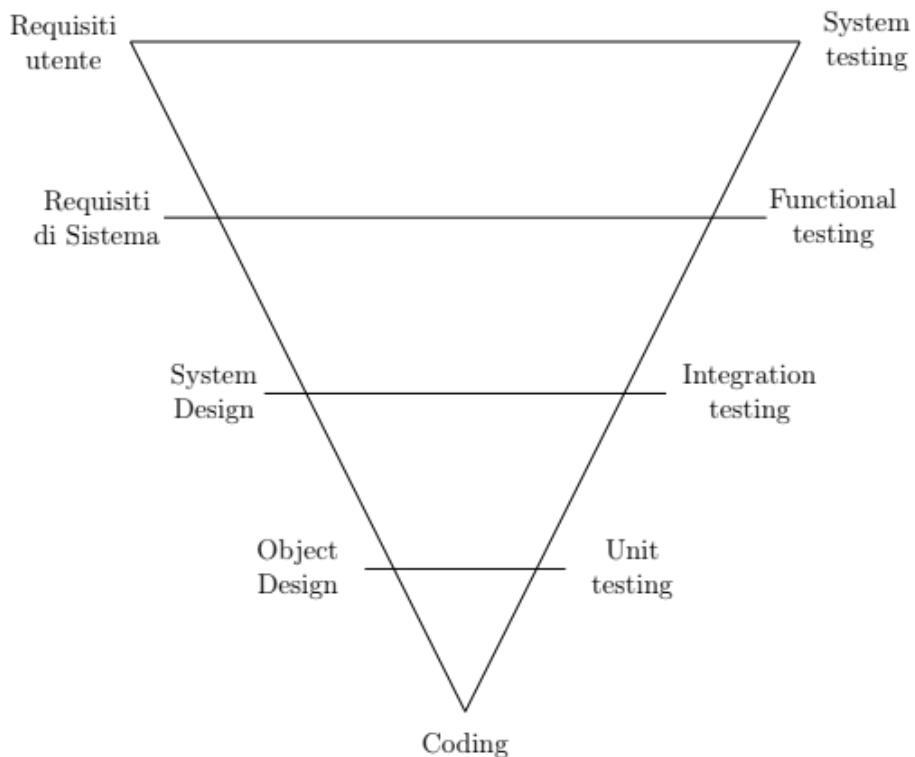


Figura 2.21: Project Management - Costruzione e Distruzione

Capitolo 3

Individuazione del target degli utenti

È importante che il sistema venga progettato in modo tale che sia incentrato sull'utente, come mostrato in Figura 3.1. Esso, inoltre, dovrebbe essere di facile utilizzo e non dovrebbe richiedere all'utente grossi sforzi in termini di memoria a breve termine, ma dovrebbe aiutarlo a porre attenzione soltanto sulle parti realmente essenziali dell'interfaccia attraverso una scelta strategica sui colori, sulle forme dei pulsanti e sulla loro posizione relativa.

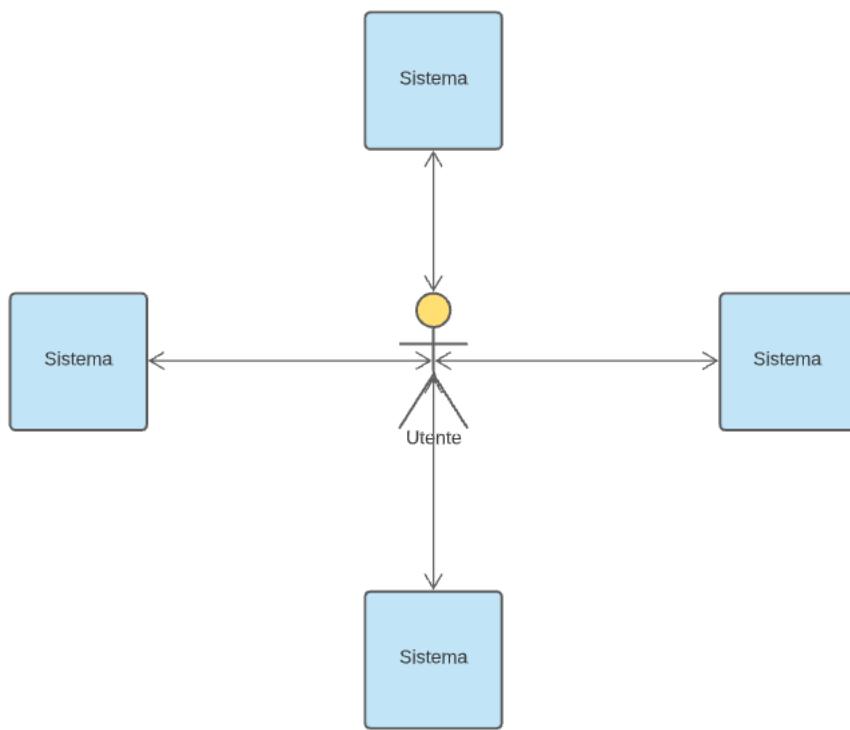


Figura 3.1: Sistemi user-centered

Per far ciò è necessario definire chi sarà il target di utenti a cui è destinato il prodotto. L'utente andrà studiato sia da un punto di vista cognitivo, inteso come animale sociale con percezione ed appercezione, ma anche da un punto di vista fisico o motorio. La letteratura medica studia una varietà di profili psicologici piuttosto interessanti; la diversità degli esseri umani sono naturali e determinanti e lo saranno ancora di più quando, nelle sezioni successive, verranno definite le personas: in quel caso una personas non solo sarà determinata dalle sue condizioni cognitive, fisiche e motorie, ma anche da caratteristiche proprie che lo identificano in quanto uomo: nome, cognome, età, una lingua, una formazione ed, in generale, la sua storia personale. Lo studio, inoltre, sarà puntualizzato tenendo in considerazione anche i rapporti che hanno le personas le une con le altre, analizzando quindi il profilo

inter-personale e non solo limitandosi a quello intra-personale, valutandone quindi i comportamenti sociali. Infine occorre definire quale sarà il ruolo dell'utente all'interno del contesto in cui lo si pensa.

L'istituto Antoniano vanta decine di persone autistiche divise in base al loro livello di funzionamento, al loro livello di collaborazione ed in base alla loro età anagrafica. Augmentus è orientato verso i gruppi ad alto funzionamento, di età anagrafica compresa tra i 18 e i 25 anni (giovani-adulti) e con un buon livello di collaborazione. I ragazzi di questo gruppo, inoltre, sono sottoposti ad un training di social skill, per verificare le loro abilità sociali ma anche le loro abilità con strumenti tecnologici come smartphone. Più formalmente, il pool di utenti target è rappresentato dallo spicchio di intersezione dato dai quattro insiemi del diagramma di Eulero-Venn in Figura 3.2. L'esistenza di questi quattro insiemi e di ciascuna intersezione generata a partire da essi è giustificata dalle indagini compiute dall'Antoniano.

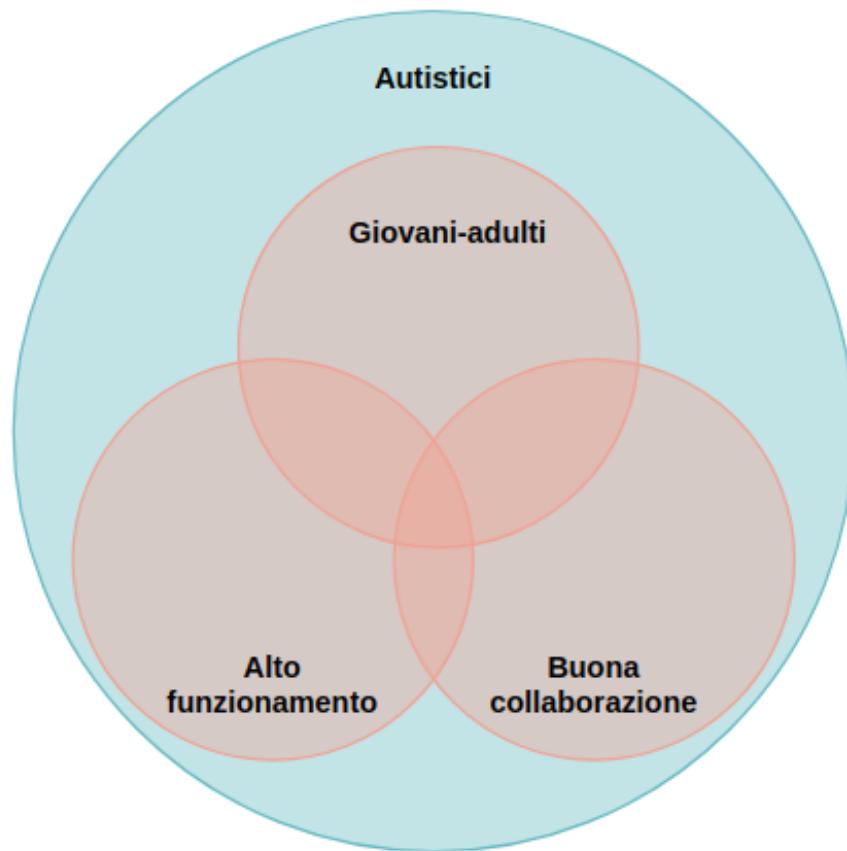


Figura 3.2: User pool target

3.1 Individuazione di pattern e variabili comportamentali

Sulla base di quanto detto finora, è possibile formalizzare le variabili comportamentali degli utenti target. In particolare:

- Attività (activity) - l'utente usa lo smartphone e conosce almeno le sue funzionalità di base;
- Atteggiamenti (attitudes) - l'utente ha una sufficienza conoscenza del dominio tecnologico;
- Attitudini (aptitudes) - l'utente è ad alto funzionamento ed ha un buon grado di collaborazione ma potrebbe necessitare di un sussidio su alcune azioni, come ad esempio apparecchiare la tavola o sbucciare una mela;
- Motivazioni (motivations) - l'utente è motivato all'uso di Augmentus poiché è già abituato alla CAA e alle rappresentazioni PECS: in altre parole è un contesto a cui già è abituato;
- Abilità (skills) - l'utente ha sufficienti abilità di social skill e conosce, seppur in linea di massima, le funzionalità principali che il suo smartphone gli concede, come ad esempio accendere, spegnere, cliccare sui pulsanti, aprire un'app e così via;

La Figura 3.3 sintetizza la distribuzione delle variabili comportamentali per mezzo di un formalismo pseudo-statistico.

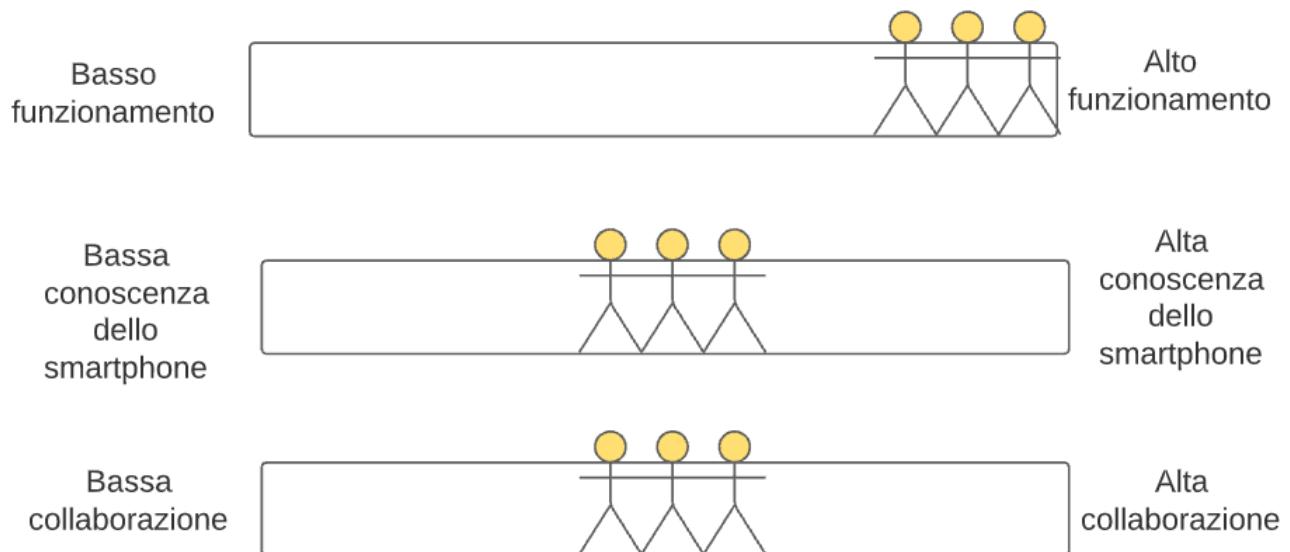


Figura 3.3: Distribuzione delle variabili comportamentali

Uno studio a raggruppamenti e statistico di questo tipo consente di generalizzare la soluzione al problema, rendendola presumibilmente adatta ad ogni contesto più o meno simile a questo descritto.

3.2 Personas-type

A partire dalle constatazioni e dalle analisi compiute finora, possiamo costruire le personas-type in modo tale da priorizzare gli utenti a cui è risolta l'app e definire quale tra i suoi elementi deve essere il target primario del design. Ogni personas individuata in questa fase sarà sintetizzata attraverso una scheda che non ha solo la funzione di fornire una schematizzazione delle caratteristiche principali, ma anche una funzione per comunicare la ricerca con terzi o col team di sviluppo.

3.2.1 Marco Tulli

La prima personas che individuiamo è Marco Tulli, di 22 anni, residente a Portici. Rappresenta la primary personas-type, cioè quella classe di utenti target i cui bisogni devono essere completamente e felicemente soddisfatti e con cui il sistema deve empatizzare nel migliore dei modi. Marco è giovane, si reca presso l'Antoniano di cui già ha fatto conoscenza delle metodologie CAA e PECS. Non ha mai provato tecnologie di realtà aumentata ma vi si mostra curioso. È collaborativo e tranquillo ed è motivato all'uso dell'applicazione perché sa che gli può conferire maggior autonomia, superando le sue frustrazioni. Le sue caratteristiche sono sintetizzate in Figura 3.4.

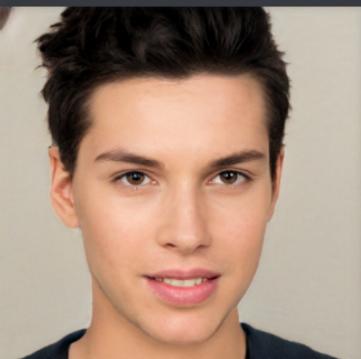
NAME	EMPHATIZATION	TYPE
Marco Tulli	 100 %	Primary Personas Type
	Background <ul style="list-style-type: none"> Giovane-adulto che vive in casa dei genitori che gli prestano aiuto in determinate occasioni Non ha mai avuto esperienze con la realtà aumentata Già ha esperienze con la CAA e le metodologie PECS 	
	Goals <ul style="list-style-type: none"> Assumere maggiore autonomia ed incrementare le proprie abilità per mezzo dell'applicazione Imparare a fare cose nuove arricchendo il suo vocabolario di azioni quotidiane 	
Demographics <ul style="list-style-type: none"> Residente in Portici (NA) Conosce l'istituto Antoniano e spesso si reca presso il semiresidenziale 	Motivations <ul style="list-style-type: none"> Il modulo di realtà aumentata motiva Marco all'uso dell'applicazione, essendo una tecnologia moderna e di cui probabilmente non era a conoscenza Può voler collezionare ogni carta PECS come forma di rinforzo positivo 	Frustrations <ul style="list-style-type: none"> L'autismo lo ostacola e potrebbe richiedergli uno sforzo in più per imparare ad usare correttamente l'applicazione nella sua totalità
Skills Tips <p>Buone social skill, buona conoscenza tecnologica e sufficiente conoscenza delle funzionalità principali del suo smartphone</p>		
Expectations <ul style="list-style-type: none"> Vuole acquisire una maggiore autonomia e talvolta essere assistito nelle attività quotidiane Vuole rifarsi alle metodologie di CAA e PECS che ha già incontrato all'Antoniano 		
UXPRESSIA <small>This persona was built in uxpressia.com</small>		

Figura 3.4: Marco Tulli - Primary personas type

3.2.2 Giulia Improta

Adesso occorre definire la secondary personas-type, ovvero un'altra personas che abbia qualche necessità in più rispetto a Marco ma che possano comunque essere soddisfatte in maniera piuttosto agile senza sconvolgere l'abilità che ha il sistema di soddisfare le necessità di Marco. Giulia è una ventenne residente a San Giorgio a Cremano e studia alle scuole superiori. Si reca molto spesso al semiresidenziale dell'Antoniano accompagnata dai suoi genitori che, non conoscendo bene la strada da percorrere per arrivare, spesso fanno più chilometri del previsto. Le sue caratteristiche sono sintetizzate in Figura 3.5.

NAME	EMPHATIZATION	TYPE
Giulia Improta	 80 %	Secondary Personas Type
	Background <ul style="list-style-type: none"> Studentessa presso le scuole superiori Non ha mai avuto esperienze con la realtà aumentata Già ha esperienze con la CAA e le metodologie PECS 	
Demographics <ul style="list-style-type: none"> Residente in San Giorgio a Cremano (NA) Conosce l'istituto Antoniano e spesso si reca presso il semiresidenziale 	Goals <ul style="list-style-type: none"> Assumere maggiore autonomia ed incrementare le proprie abilità per mezzo dell'applicazione Imparare a fare cose nuove arricchendo il suo vocabolario di azioni quotidiane Visitare più facilmente il sito dell'Antoniano ed aiutare i suoi genitori a trovare il percorso più veloce per raggiungere il semiresidenziale da casa sua 	Frustrations <ul style="list-style-type: none"> L'autismo lo ostacola e potrebbe richiedergli uno sforzo in più per imparare ad usare correttamente l'applicazione nella sua totalità È lenta a scrivere con lo smartphone, per cui spesso evita di utilizzare Google Maps per compiere le sue query
Skills Tips Buone social skill, buona conoscenza tecnologica e sufficiente conoscenza delle funzionalità principali del suo smartphone	Motivations <ul style="list-style-type: none"> Il modulo di realtà aumentata motiva Giulia all'uso dell'applicazione, essendo una tecnologia moderna e di cui probabilmente non era a conoscenza Può voler collezionare ogni carta PECS come forma di rinfoco positivo Mostra le animazioni di realtà aumentata ai suoi amici con cui le commenta 	
Expectations <ul style="list-style-type: none"> Vuole acquisire una maggiore autonomia e talvolta essere assistita nelle attività quotidiane Vuole rifarsi alle metodologie di CAA e PECS che ha già incontrato all'Antoniano 		
UXPRESSIA <small>This persona was built in upressoia.com</small>		

Figura 3.5: Giulia Improta - Secondary personas type

3.2.3 Rebecca Caputo

Rebecca rappresenta l'utente le cui aspettative e obiettivi non possono essere soddisfatti dall'applicazione perché completamente estranee ai business goal di sistema. Le sue caratteristiche sono sintetizzate in Figura 3.6.

NAME	TYPE
Rebecca Caputo	Negative Personas Type
	Background <ul style="list-style-type: none">Ex insegnante in pensioneConosce la realtà aumentata ma di cui non è particolarmente una fanNon conosce le metodologie CAA e PECS
	Goals <ul style="list-style-type: none">Visitare almeno 5 paesi nel 2021Comprare una nuova casaAcquistare uno smartphone e imparare ad usare applicazioni di messaggistica per contattare sua sorella in Belgio
Demographics <ul style="list-style-type: none">Residente in San Benedetto del Tronto (AP)Non conosce l'istituto Antoniano	
Skills Tips <p>Pessime social skill, scarsa conoscenza del dominio tecnologico e non possiede uno smartphone</p>	
Expectations <ul style="list-style-type: none">Vuole fare più viaggiVuole comprare un cane	

UXPRESSIA
This persona was built in uxpressia.com

Figura 3.6: Rebecca Caputo - Negative personas type

3.3 Personas-goal e business goal

In questo paragrafo verrà mostrata la relazione che sussiste tra le necessità di ciascuna personas-type definite al paragrafo precedente e quelle del sistema.

3.3.1 Primary-personas type goal e business goal

Per definizione i bisogni e le necessità del primary-personas type, ovvero Marco, devono essere completamente soddisfatte. Come si evince dalla Figura 3.7, gli obiettivi di Marco sono tutti pienamente soddisfatti dal momento che, formalmente, sono un sottoinsieme dei goal del sistema.

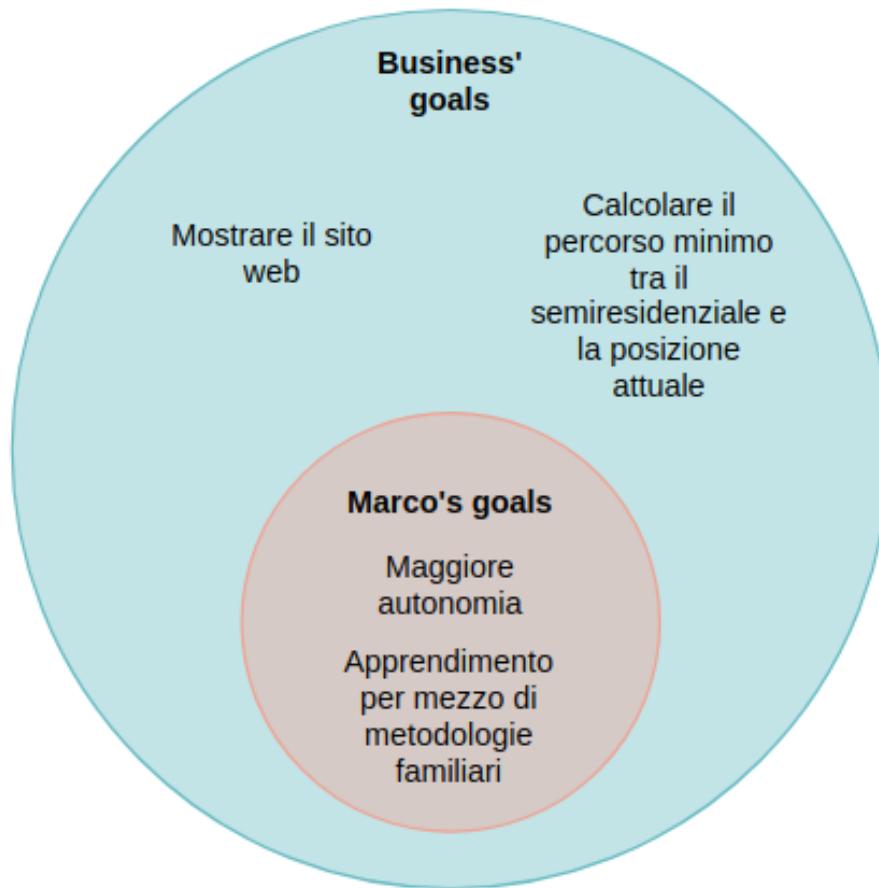


Figura 3.7: Primary-personas goal e business goal

3.3.2 Secondary-personas type goal e business goal

Anche gli obiettivi del secondary personas-type, cioè Giulia, devono essere soddisfatti in maniera agile dal sistema senza perdere di efficienza e completezza rispetto alle necessità di Marco. In Figura 3.8 è mostrata la relazione di contenimento, verificando che le necessità di Giulia sono un'estensione di quelle di Marco e sono agilmente soddisfacibili confrontando queste necessità con quelle rappresentate dal business goal in Figura 3.7.

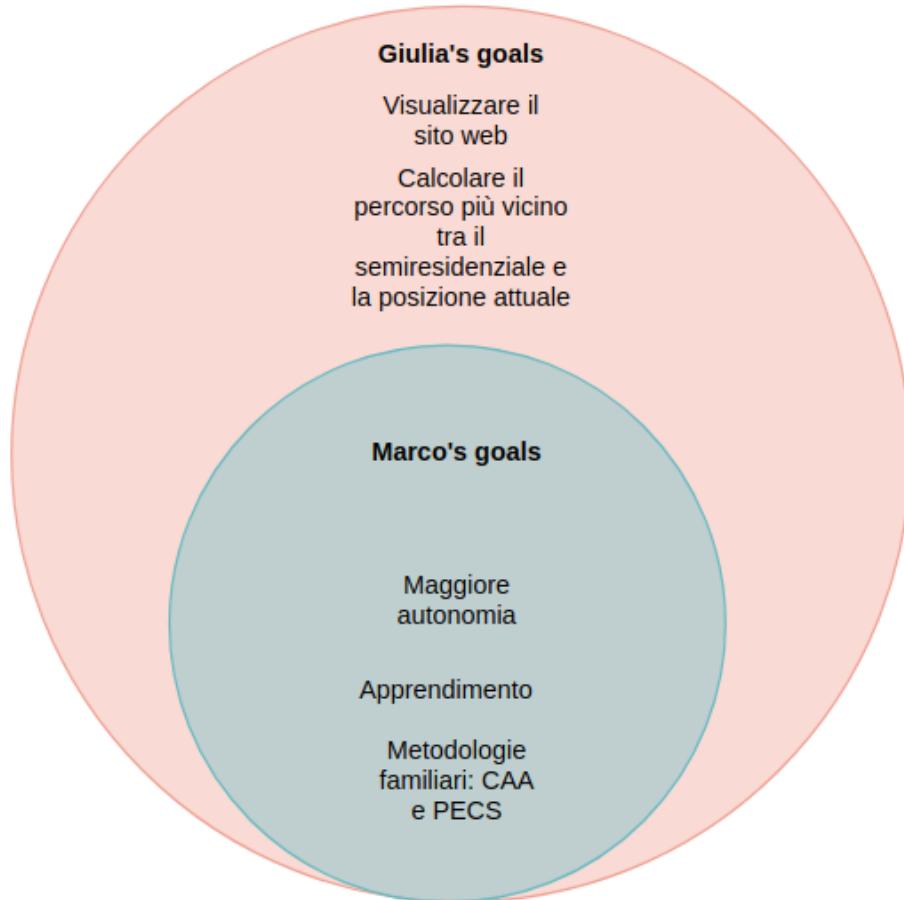


Figura 3.8: Primary-personas goal e secondary-personas goal

3.3.3 Negative-personas type e business goal

Le aspettative e gli obiettivi delle negative-personas type, rappresentate da Rebecca, non sono soddisfacibili dal sistema poiché l'intersezione dei due insiemi è vuota, come mostrato in Figura 3.9.

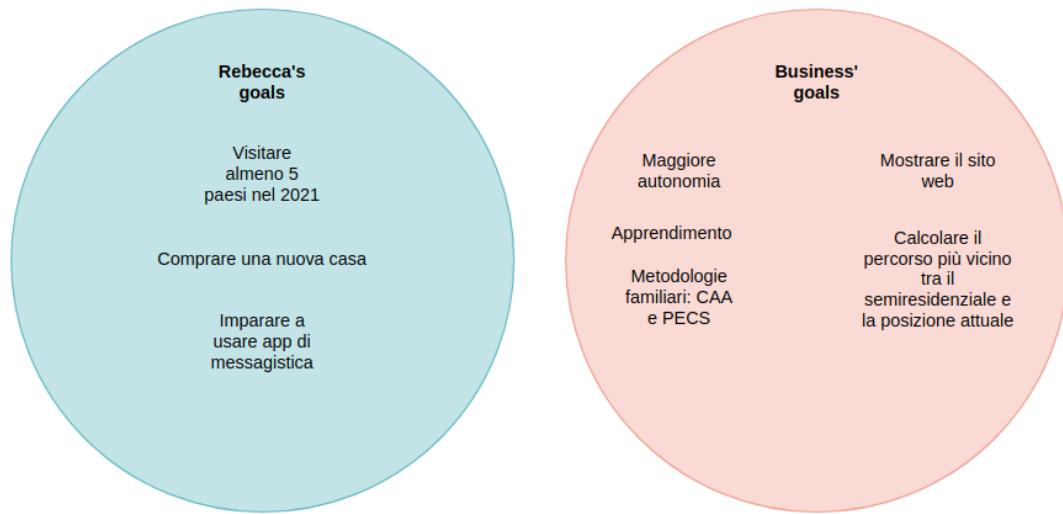


Figura 3.9: Negative-personas goal e business goal

Capitolo 4

Studio della valutazione a priori dell'usabilità

L'uomo interagisce con oggetti, non necessariamente tecnologici, che spesso interpreta come di difficile utilizzo. L'analisi che consiste di individuare dove l'utente riscontra una difficoltà è necessaria per eliminarla e quindi ricercare l'usabilità del sistema. In questo capitolo affronteremo la questione di valutare l'usabilità del prodotto a priori, ovvero verificarne mediante metodi scientifici - e a partire dal prodotto non ancora esistente ma in stato primordiale - l'usabilità.

Poiché in questa fase il prodotto finale non esiste ancora, ma si è ancora in una fase di prototipazione, la valutazione dell'usabilità sarà informale, utilizzando sketch e prototipi. Principalmente sono stati impiegati prototipi di ruolo (role prototype) e quelli che consentano di valutare l'interazione uomo-macchina (look&feel prototype). Quelli relativi agli aspetti implementativi e algoritmici (implementation prototype) sono stati ignorati in questa prima fase, ma sono stati presi in considerazione dal team di sviluppo per scegliere le migliori strutture dati che potessero garantire un tempo di esecuzione minimo. In Figura 4.1 viene mostrata la natura dei prototipi impiegati in questa prima fase di valutazione a priori. Il cerchio giallo rappresenta l'orientamento dello scopo dei prototipi impiegati in questa fase: essi, come detto, sono stati costruiti in modo tale da valutare in maniera equa il ruolo (del prodotto nella vita del suo utente) e l'interfaccia, per massimizzare l'usabilità del prodotto e ricercare le pratiche di buon design sull'interazione uomo-macchina.

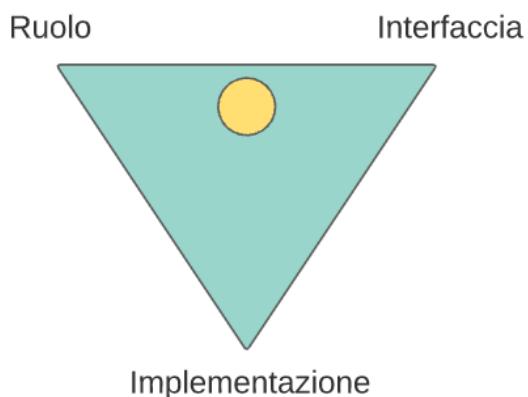


Figura 4.1: Spazio dei prototipi in relazione al loro scopo

I prototipi che sono stati impiegati in questa fase, inoltre, sono stati costruiti in modo tale da essere dinamici ma anche parzialmente interattivi, per venire in contro all'esigenza di costruire un sistema user-centered: durante le primissime fasi è stata compiuta una prototipazione a schizzo

come mostrato in Figura 4.2. Come si nota, inizialmente erano state pensate due alternative per la homepage: l'alternativa α oppure l'alternativa β . Si è evinto che l'alternativa α fosse troppo caotica, specialmente per uno user pool così delicato: troppe informazioni sullo schermo, troppi pulsanti e troppo dispersivo. L'alternativa β , invece, è stata ripulita di parecchio, mostrando solo le informazioni strettamente necessarie all'uso di Augmentus.

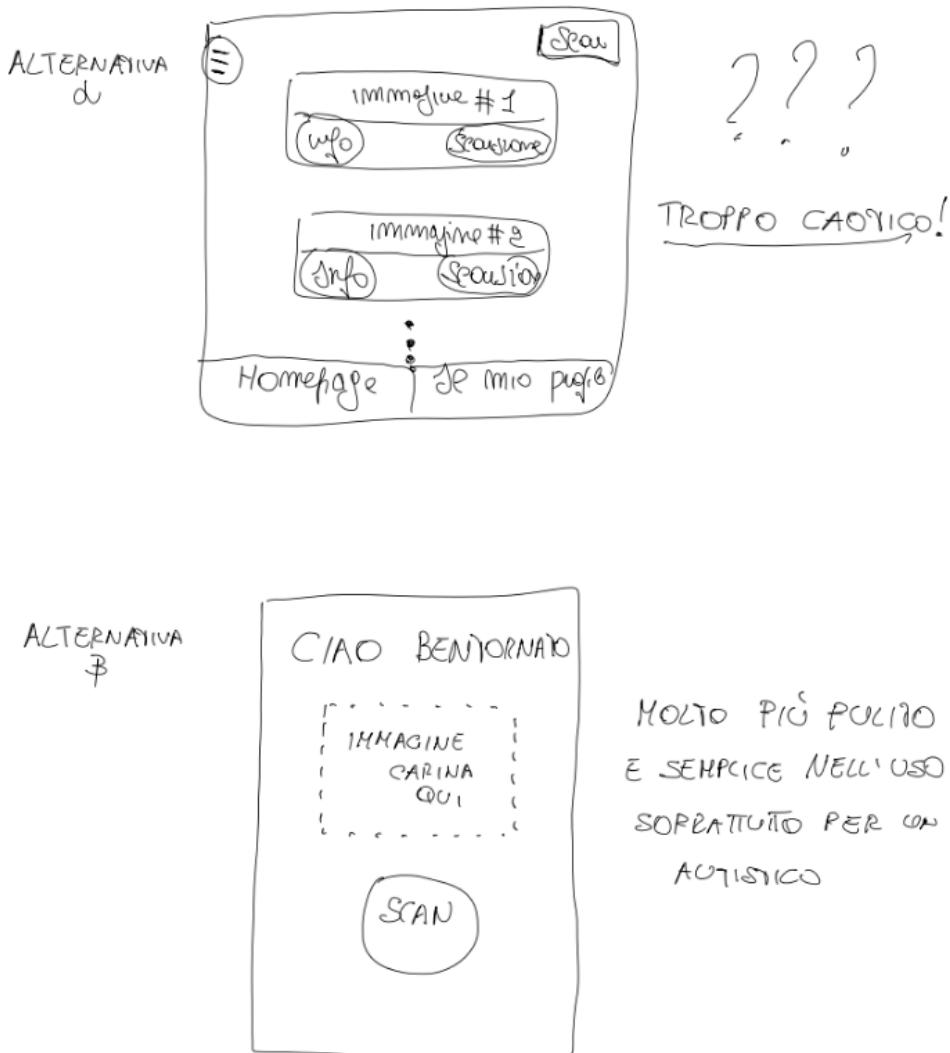


Figura 4.2: Prototipazione iniziale "a schizzo" della homepage

La Figura 4.3, invece, mostra una rappresentazione schematica per constatare l'aspetto macroscopico del sistema e avere una vista generale delle sue funzionalità e interfacce principali.

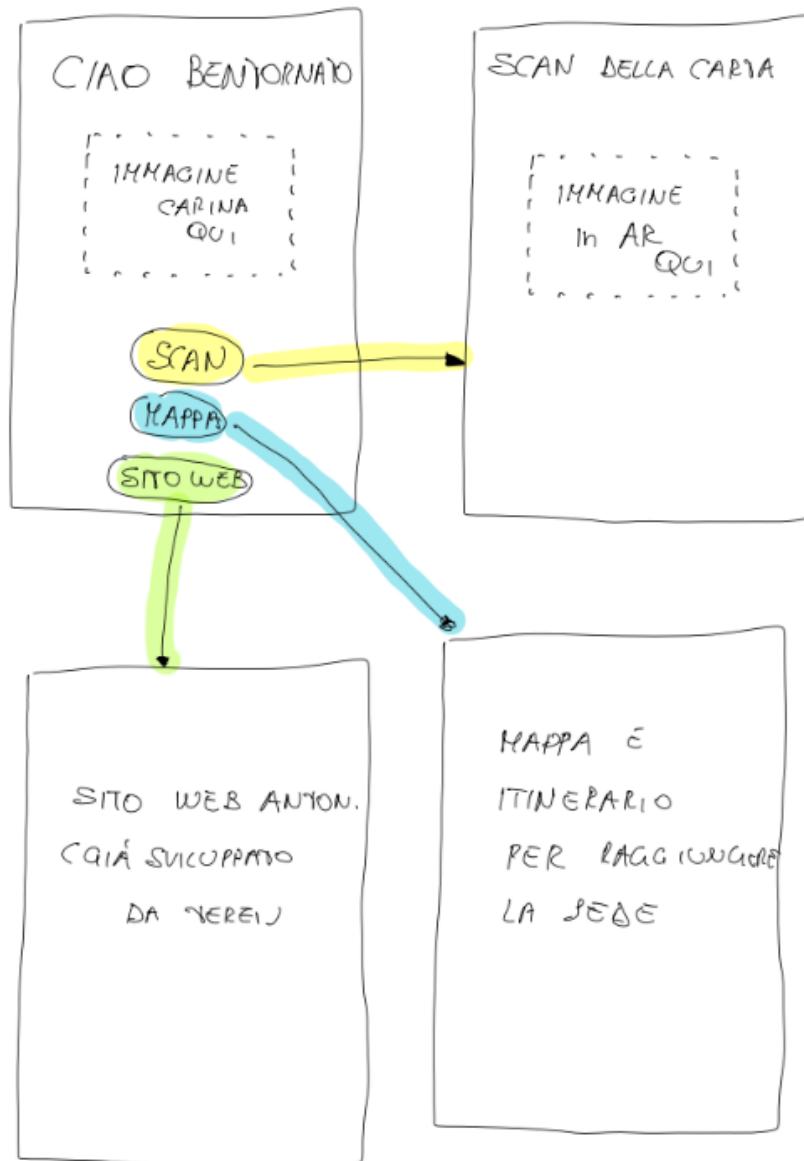


Figura 4.3: Prototipazione iniziale "a schizzo" del prodotto

Una volta che il prototipo a schizzo è stato validato, rimuovendo aspetti deboli o poco convincenti, sono stati costruiti prototipi dinamici e interattivi attraverso animazioni su Adobe Xd, come si vedrà nei capitoli successivi: l'utente valutatore, infatti, poteva guardare ciascuna delle schermate principali del prodotto e, cliccando opportunamente sui bottoni che vi comparivano, poteva assistere alla transizioni oppure animazioni di vario genere.

Capitolo 5

Documento di Design del Sistema

5.1 Analisi dell'architettura

L'obiettivo principale della scelta di un'architettura è quello di impostare la visione complessiva del sistema in termini di visibilità dall'esterno. Di conseguenza, definire un'architettura consiste nell'identificare i moduli e le politiche di comunicazione tra essi.

5.1.1 Architettura esterna

Per una decomposizione ottimale ed un livello di manutenibilità¹ elevato è stata scelta un'architettura bassata su *layer* (strati). Ogni layer rappresenta una decomposizione gerarchica in macro blocchi e ogni macro blocco rappresenta un sottosistema che offre dei servizi. L'architettura a tre strati adottata è chiusa: ciò significa che ogni layer potrà accedere solo al layer immediatamente al di sotto di esso; in altre parole, la modifica a un livello non influenzerà quello superiore (ad esempio sarà possibile modificare la tecnologia del database senza che i client vi siano influenzati), portando quindi a un livello di manutenibilità, modularità², riusabilità³ e portabilità⁴ più alto. Bisognerebbe sempre progettare per anticipare il cambiamento. In Figura 5.1 è rappresentata l'architettura a tre livelli adottata: l'utente mobile a cui è destinato Augmentus utilizza il suo smartphone per scannerizzare una carta. Il client comunica quindi con un server di terze parti il quale verifica se esiste una corrispondenza tra la carta ed un file multimediale in AR. Se c'è vuol dire che era stato precedentemente associato dall'amministratore backoffice e verrà quindi restituito al client.

¹Qualità interna di un Software che stabilisce quanto sia facile da parte degli sviluppatori la modifica di qualche funzionalità o l'aggiunta di nuove, sia in termini di requisiti del cliente che in termini di cambiamenti legislativi

²Scissione del Software in micro-componenti

³Qualità interna del Software che stabilisce la possibilità di riciclare classi o, in generale, brani di codice per altri progetti

⁴Qualità interna del Software che stabilisce su quante piattaforme è possibile eseguirlo

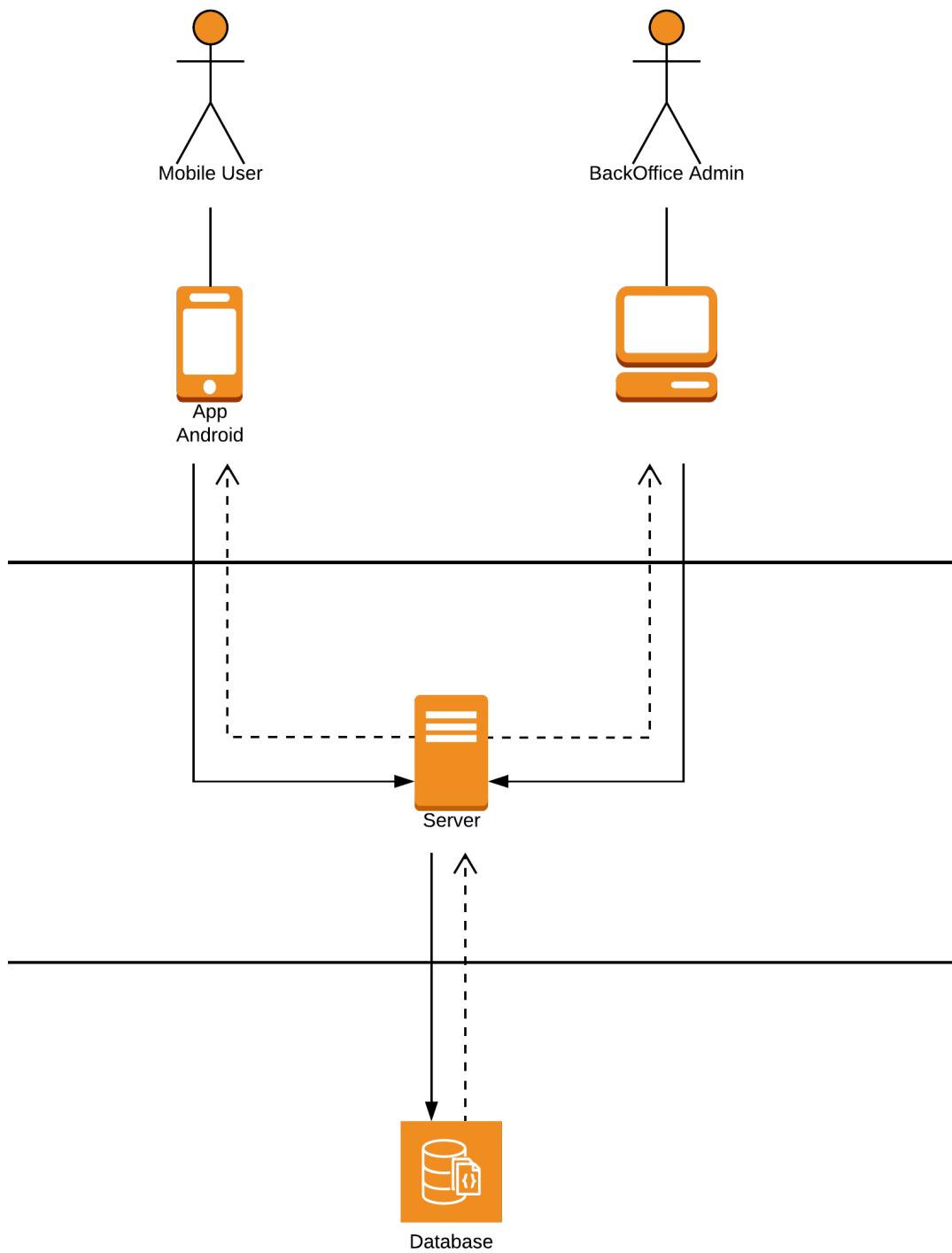


Figura 5.1: Architettura del Sistema

Il Client

Il client Android è stato progettato usufruendo dei principali Design Pattern dell'ingegneria del software. In particolare:

- Factory Method - design pattern creazionale che permette di creare istanze di classi in un unico punto del codice in base ad una configurazione o logica a runtime. In particolare, nel nostro progetto il Factory Method è stato usato per istanziare la giusta classe in base alle tecnologie scelte. Questo naturalmente porta a un livello di manutenibilità molto alto, pur aumentando leggermente la dipendenza tra la classe Factory e quelle che implementano l'interfaccia;
- Singleton - design pattern creazionale che consente di avere un'unica istanza per una certa classe;
- MVC - scissione delle responsabilità in tre macrocomponenti quali model, view e controller.

Le tecnologie impiegate

Durante la fase di System Design sono state scelte le tecnologie e i dettagli implementativi che potevano al meglio realizzare il software.

- Java - uno dei linguaggi di programmazione Object-Oriented più utilizzati, molte API disponibili e documentazione ufficiale molto ricca;
- UniteAR - un servizio molto scalabile e semplice da utilizzare per garantire le funzionalità in AR;
- Google Maps - un servizio per poter visualizzare il percorso dalla posizione corrente per raggiungere l'Antoniano.

Perché Java?

Java è uno dei linguaggi Object-Oriented più utilizzati inoltre è oggetto di studio in numerosi corsi del corso di laurea. Da un punto di vista pratico, l'impiego di questo linguaggio di programmazione ha implicato numerosi vantaggi. In primis, Java è un metà compilato e metà interpretato: questo consente di attingere a caratteristiche e peculiarità sia dei compilatori che degli interpreti, come ad esempio il numero di controlli elevato - caratteristica di un compilato - oppure un'interezione può snella - caratteristica di un interpretato. Java, inoltre, consente di gestire in maniera più sicura la memoria allocata: attraverso un sistema di Garbage Collector, esso riesce in automatico a individuare aree di memoria dereferenziate e disalocarle, senza tuttavia richiedere un'operazione esplicita da parte del programmatore (ad esempio, il linguaggio C presupponeva l'uso di `free` per liberare memoria precedentemente allocata in maniera dinamica - e questo poteva essere pericoloso). Inoltre si tratta di un linguaggio (quasi) fortemente tipizzato, perché i controlli di tipo sono (quasi) tutti sempre effettuati. Java è detto anche indipendente dalla macchina e questo permette di elevare il livello di portabilità del sistema. Il motivo di ciò è dovuto al fatto che il bytecode viene eseguito sulla Java Virtual Machine che funge anche da interprete e da sandbox, ovvero un ambiente protetto in cui ciò che accade al suo interno non influenzerà l'ambiente esterno che lo ospita e quindi senza danneggiare il sistema (al peggio avviene un Denial of Service, consumando tutte le risorse del processore). La creazione del bytecode (`.class`) da parte del compilatore di Java consente di definire il fulcro per cui questo linguaggio risulta essere un ottimo compromesso tra i linguaggi compilati e quelli interpretati: è più portabile di un linguaggio compilato ma più efficiente di uno interpretato. È un linguaggio molto consigliato per le applicazioni web non solo per le caratteristiche appena descritte, ma anche perché risulta molto sicuro: come già anticipato, non ammette aritmetica dei puntatori (potrebbe essere pericoloso gestire esplicitamente le aree di memoria, oppure potrebbe portare ad uno stato indicibile e accessi non protetti alle aree di memoria portano ai più comuni problemi di sicurezza di un'applicazione⁵); quando un programma Java entra in esecuzione, la JVM costruisce per essa una sandbox in cui, come detto, ciò che accade al suo interno non influenza in modo critico il sistema che lo ospita. Oltre a ciò, all'interno della sandbox vengono fatti anche diversi controlli sulle classi che vi transitano, oppure controlli che mirano a gestire accessi all'esterno della sandbox. È importante notare che non è sufficiente implementare costrutti sicuri solo nel linguaggio utilizzato, ma anche nell'ambiente di esecuzione: il bytecode può essere generato a partire da codice `.java` ma anche da codice che deriva da altri linguaggi di programmazione meno sicuri, oppure addirittura potrebbe essere stato scritto a mano con intenzioni ostili; per fare ciò l'esecuzione di una classe Java prevede il caricamento da parte del ClassLoader del bytecode `.class`, poi l'esecuzione del metodo `main` ed infine il caricamento delle classi accessorie. Così facendo il bytecode viene verificato ogni volta che una classe viene caricata. Il controllo comunque può essere più o meno flessibile quando le classi sono create localmente (poichè ritenute affidabili di natura), mentre è più rigido se le classi provengono da altre fonti. Il Class Loader, dopo aver compiuto queste verifiche e solo quando è tutto ok, crea le classi vere e proprie, preparando un namespace diverso per ogni fonte da cui provengono le classi, evitando quindi problemi di collisione. Il Class Loader, inoltre, evita che classi provenienti da host diversi possano comunicare tra di loro, questo per escludere la possibilità che un programma poco affidabile possa attingere e ricavare informazioni da uno affidabile. Per aumentare maggiormente il livello di sicurezza del linguaggio, Java prevede che ogni JVM sia dotata (al più) di un Security Manager responsabile delle politiche di sicurezza. È possibile restringere oppure allentare i vincoli e i controlli sulle operazioni settando opportunamente le politiche del Security Manager e ogni qual volta venga tentato di fare un'azione vietata, verrà sollevata una `SecurityException`.

⁵Molti virus traggono vantaggio dalla possibilità di dirottare l'aritmetica dei puntatori e quindi fare accesso e modificare aree di memoria in modo incontrollato

Perché un servizio dedicato esclusivamente all'AR?

L'impiego del servizio UniteAR come modulo non integrato nel client è motivato da diversi fattori. In particolare:

- Divisione delle responsabilità - il client farà *solo* da client ed il servizio di AR farà *solo* da modulo per l'AR, ricercando l'alta coesione dei moduli del sistema;
- Basso accoppiamento - il client comunicherà col server Unite AR in maniera "libera" e senza linee di codice superflue indipendentemente dalla sua natura (client Android, iOS ecc);
- Indipendenza - utilizzare un servizio disaccoppiato dal client permette di provarlo anche se il client non è stato ancora sviluppato, e viceversa.
- Do not reinvent the wheel - il servizio di AR fornisce delle funzionalità e tool già pronti per aumentare la velocità di sviluppo dell'app;
- Leggerezza - i file multimediali in AR non vengono salvati nel client ma richiesti *ad hoc* ogni volta ve ne sia la necessità per non appesantire il sistema, così come la logica di computer vision per il riconoscimento dell'immagine.

5.2 Class Diagram di Design

Di seguito viene riportato il Class Diagram di Design che tiene conto delle scelte implementative giustificate finora.

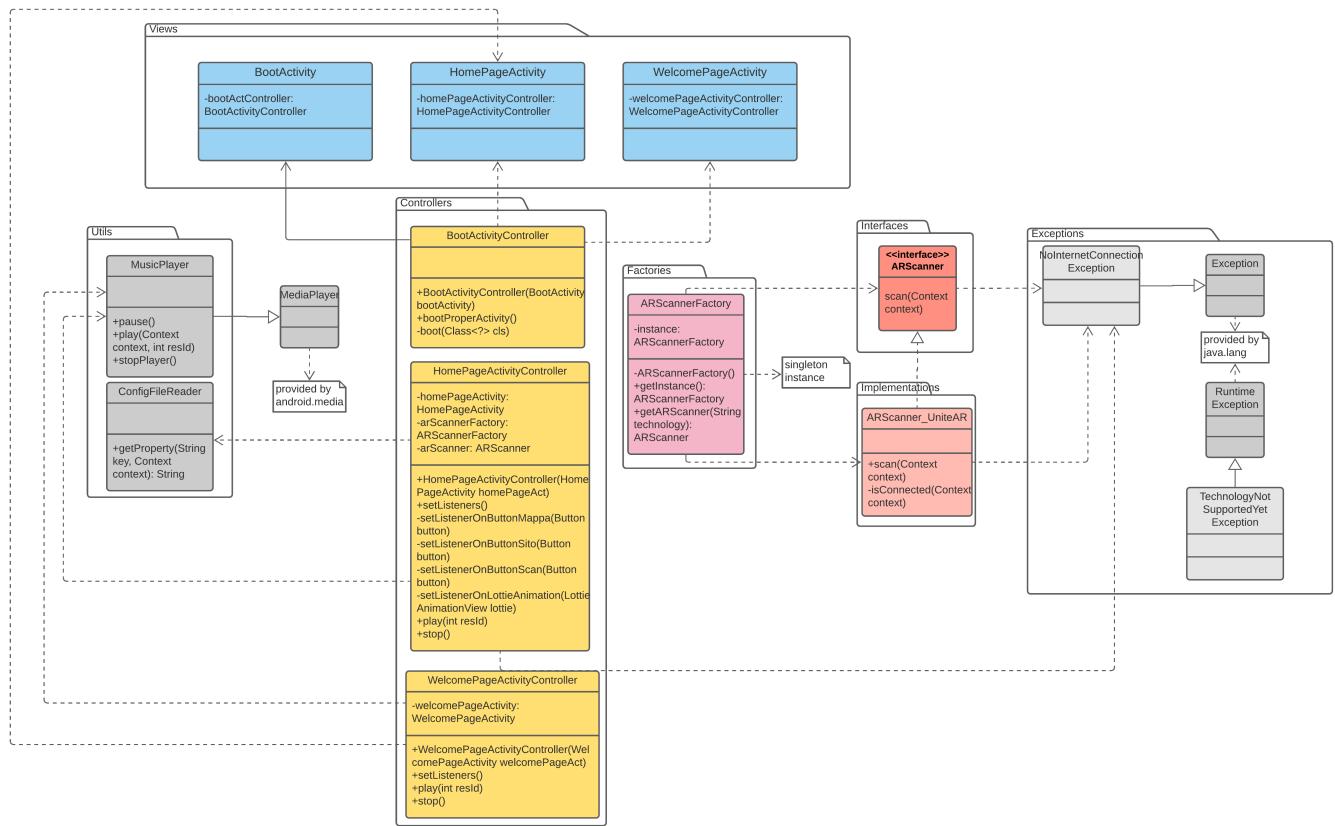


Figura 5.2: Class Diagram di Design

5.3 CRC Card

Di seguito vengono riportate le carte *Class-Responsibility-Collaborators* che descrivono la responsabilità di una classe e le classi ausiliare per il suo funzionamento. Questa schematizzazione permette di verificare se ogni classe è sufficientemente coesa e poco dipendente dalle altre e descrive il motivo per cui è stata costruita.

Views:

Class Name	BootActivity
Superclass	AppCompatActivity
Subclasses	-
Implements	-
Responsability	Collaborators
Rappresenta la schermata di boot per Android	BootActivityController

Class Name	HomePageActivity
Superclass	AppCompatActivity
Subclasses	-
Implements	-
Responsability	Collaborators
Rappresenta l'homepagina dell'app	HomePageActivityController

Class Name	WelcomePageActivity
Superclass	AppCompatActivity
Subclasses	-
Implements	-
Responsability	Collaborators
Rappresenta la schermata di benvenuto quando si installa l'app per la prima volta	WelcomePageActivityController

Utils:

Class Name	ConfigFileReader
Superclass	-
Subclasses	-
Implements	-
Responsability	Collaborators
Fornisce un metodo per leggere coppie key=value all'interno del file config.properties per aumentare la manutenibilità del software	-

Class Name	MusicPlayer
Superclass	MediaPlayer
Subclasses	-
Implements	-
Responsability	Collaborators
Fornisce dei metodi per riprodurre, sospendere o interrompere una traccia audio	-

Interfaces:

Class Name	ARScanner
Superclass	-
Subclasses	-
Implements	-
Responsability	Collaborators
Fornisce un'interfaccia generica ed implementabile per scannerizzare una carta	NoInternetConnection-Exception

Implementations:

Class Name	ARScanner_UniteAR
Superclass	-
Subclasses	-
Implements	ARScanner
Responsability	Collaborators
Implementa il metodo dell'interfaccia ARScanner per la tecnologia UniteAR	NoInternetConnection-Exception

Factories:

Class Name	ARScannerFactory
Superclass	-
Subclasses	-
Implements	-
Responsability	Collaborators
Classe singleton per fornire la giusta classe che implementa l'interfaccia ARScanner in base alla tecnologia scelta	ARScanner, ARScanner_UniteAR

Exceptions:

Class Name	NoInternetConnection-Exception
Superclass	Exception
Subclasses	-
Implements	-
Responsability	Collaborators
Rappresenta l'exception sollevata quando non si è connessi alla rete	-

Class Name	TechnologyNotSupportedYetException
Superclass	RuntimeException
Subclasses	-
Implements	-
Responsability	Collaborators
Rappresenta l'exception sollevata quando si sceglie una tecnologia non ancora supportata	-

Controllers:

Class Name	BootActivityController
Superclass	-
Subclasses	-
Implements	-
Responsability	Collaborators
Logica di business per la view BootActivity	BootActivity, HomePageActivity, WelcomePageActivity

Class Name	HomePageActivity-Controller
Superclass	-
Subclasses	-
Implements	-
Responsability	Collaborators
Logica di business per la view HomePageActivity	HomePageActivity

Class Name	WelcomePageActivity-Controller
Superclass	-
Subclasses	-
Implements	-
Responsability	Collaborators
Logica di business per la view WelcomePageActivity	WelcomePageActivity

5.4 Sequence Diagram di Design

Di seguito vengono riportati i Sequence Diagram di Design di ciascuna funzionalità offerta dal sistema, tenendo conto delle scelte tecnologiche in base al Class Diagram di Design presentato nel paragrafo precedente.

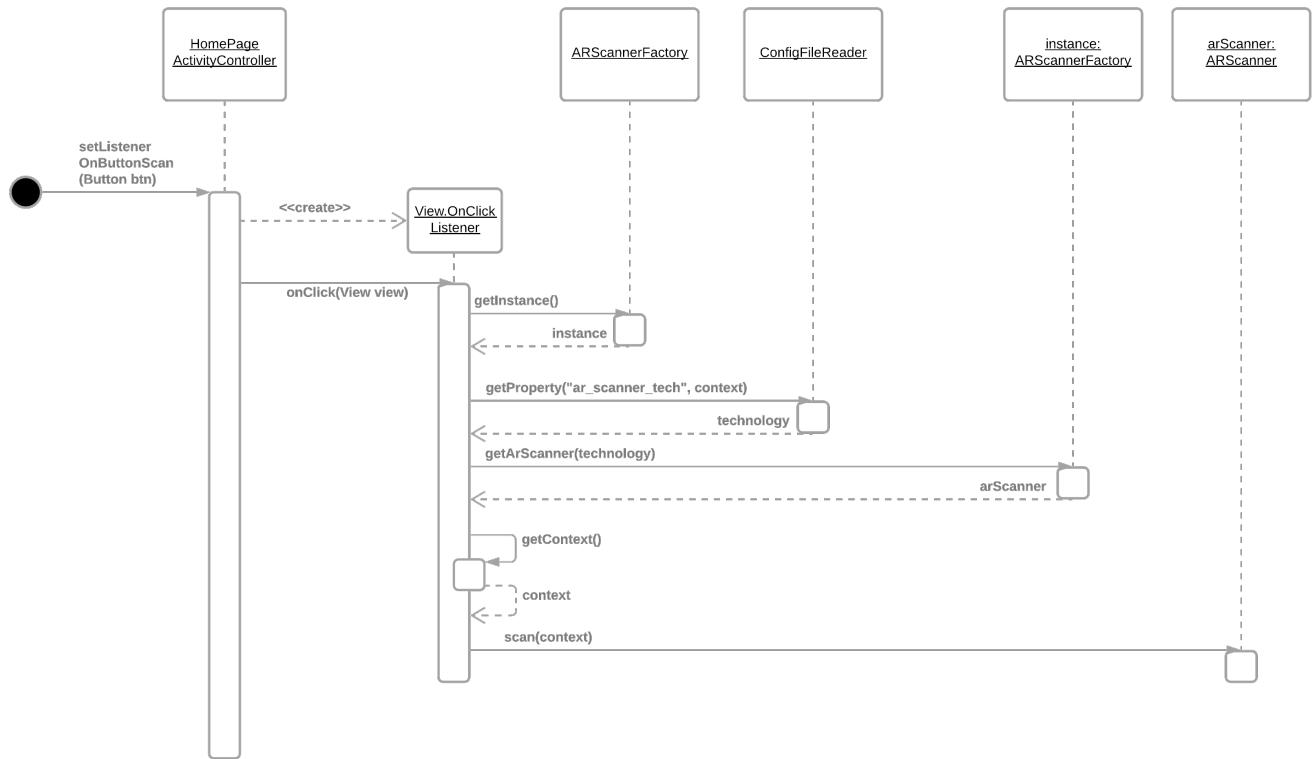


Figura 5.3: Sequence Diagram di Design - Scannerizza carta

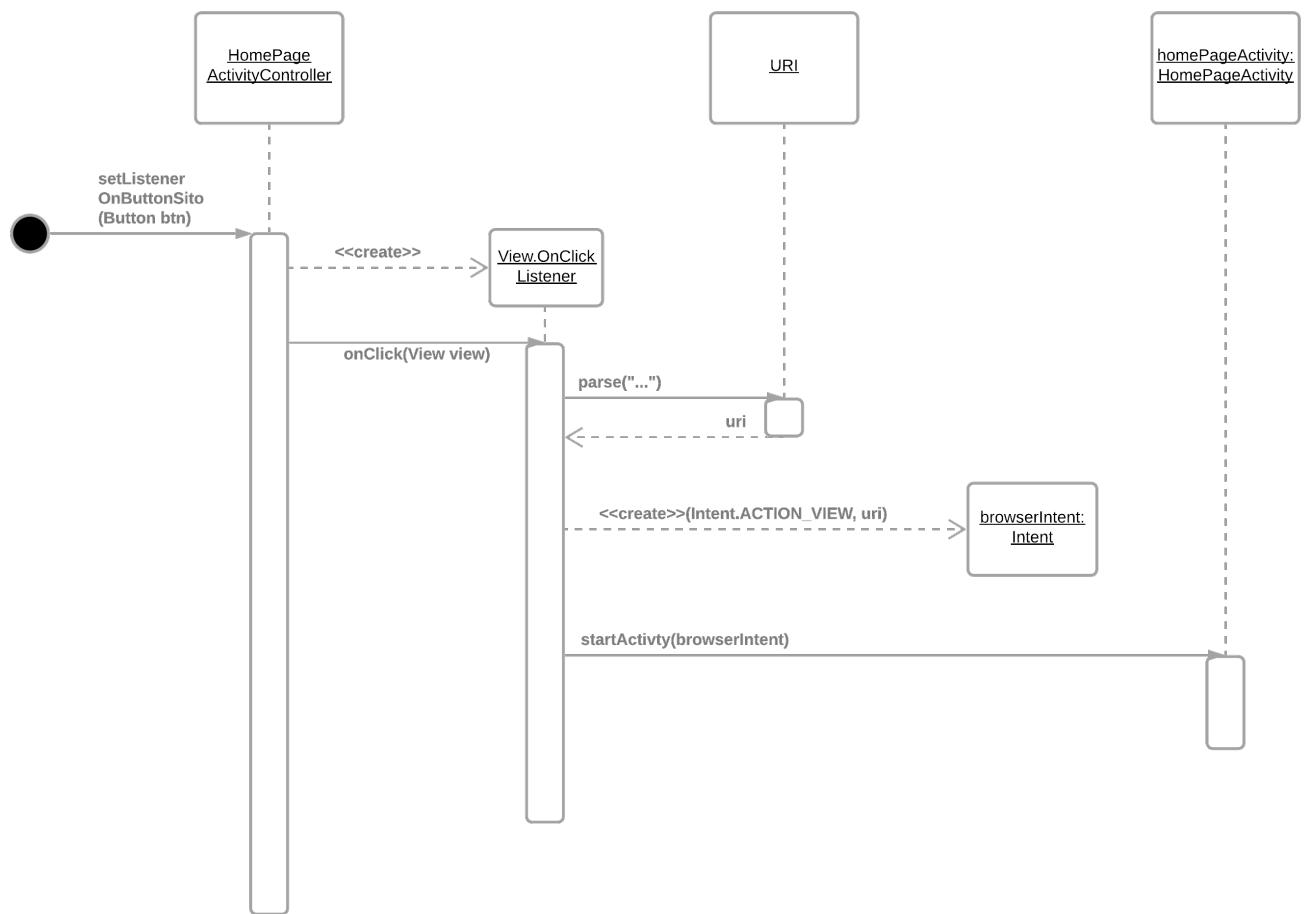


Figura 5.4: Sequence Diagram di Design - Visualizza sito web

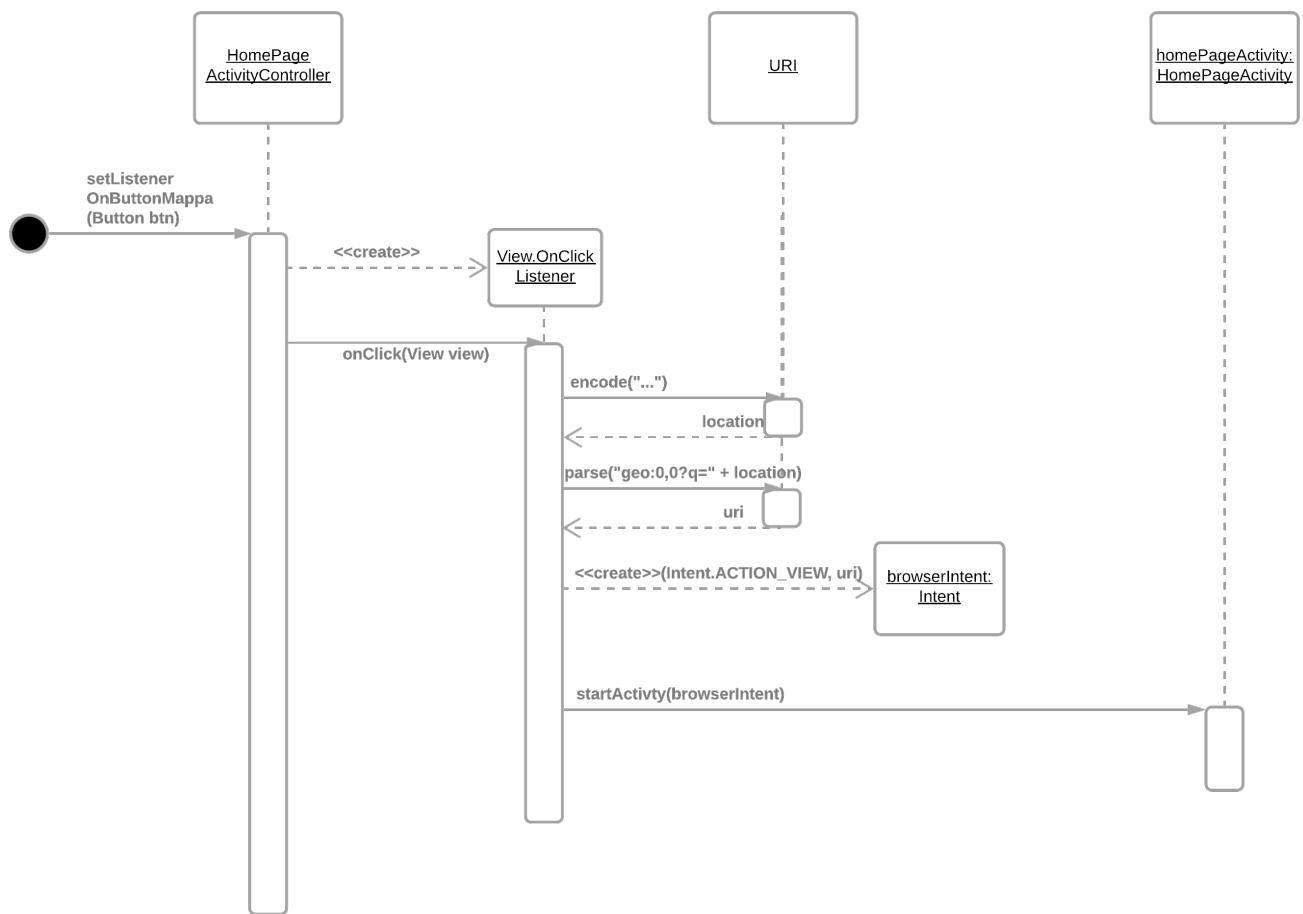


Figura 5.5: Sequence Diagram di Design - Visualizza percorso per l'Antoniano

Capitolo 6

Documento di Testing del sistema

6.1 Test Plan per System Testing

Di seguito sono riportati i test plan atti all'organizzazione della fase di system testing per verificare che tutti i requisiti funzionali descritti nel paragrafo 2.1.1 siano stati correttamente ed effettivamente implementati. Il simbolo ✓ verifica questo risultato.

Tabella 6.1: Test - Scannerizza carta

Test ID	1		
Test Name	Scannerizza carta		
Test Description	Verifica la funzionalità Scannerizza carta		
	Input	Risultato atteso	Risultato ottenuto?
	Apre la pagina per scannerizzare la carta, la inquadra e preme scan	Carta scannerizzata e visualizzazione del file multimediale in AR	✓
	Apre la pagina per scannerizzare la carta ma senza connessione ad internet	Operazione fallita e visualizzazione di un messaggio di errore	✓
	Apre la pagina per scannerizzare la carta, la inquadra ma preme indietro	Operazione annullata	✓
Notes			

Tabella 6.3: Test - Visualizza sito web

Test ID	2		
Test Name	Visualizza sito web		
Test Description	Verifica la funzionalità Visualizza sito web		
	Input	Risultato atteso	Risultato ottenuto?
	Apre l'app e clicca sul pulsante "Sito web"	Visualizzazione del sito web	✓
	Apre l'app ma preme tasto indietro	Operazione annullata	✓
Notes	Il sito web è precedentemente sviluppato da terzi e raggiungibile		

Tabella 6.5: Test - Visualizza percorso per raggiungere l'Antoniano

Test ID	3		
Test Name	Visualizza percorso per raggiungere l'Antoniano		
Test Description	Verifica la funzionalità Visualizza percorso per raggiungere l'Antoniano		
	Input	Risultato atteso	Risultato ottenuto?
	Apre l'app e clicca sul pulsante "Mappa"	Visualizzazione del percorso per raggiungere l'Antoniano	✓
	Apre l'app ma preme tasto indietro	Operazione annullata	✓
Notes	La posizione dell'Antoniano deve essere stata salvata su Google Maps		

6.2 Valutazione sul campo dell'usabilità soggettiva e oggettiva

In questo capitolo viene valutata l'usabilità del software in maniera soggettiva ed oggettiva attraverso specifiche tecniche scientifiche riconosciute nell'ambito dell'ingegneria del software e dell'interazione uomo-macchina. Sono state seguite le best-practise introdotte da Shneiderman, conosciute con il nome delle "otto regole d'oro". Una schematizzazione di quanto compiuto è rappresentato nella Tabella 6.7 che verifica la coerenza del sistema rispetto a tali regole.

Tabella 6.7: Coerenza del sistema rispetto alle otto regole d'oro di Schneiderman

Obiettivo	Valutare se il sistema rispetta le otto regole d'oro di Schneiderman		
	Regola	Risultato atteso	Risultato ottenuto?
	Battersi per la coerenza	I termini dovrebbero essere coerenti in situazioni simili e familiari per l'utente	✓
	Consentire agli utenti abituali l'uso di scorciatoie	Talvolta alcune operazioni ricorrenti potrebbero diventare tediose se si è costretti a fare sempre le stesse operazioni	✓
	Offrire feedback informativi	Occorrerebbe fornire un feedback immediato da parte del sistema per ogni azione dell'utente, coerentemente con l'azione intrapresa	✓
	Progettare i dialoghi dando la precedenza alle forme di chiusura	Per una maggiore semplicità di lettura	✓
	Offrire gestioni semplici di errori	Gli errori dovrebbero essere gestiti in maniera semplice e tale da non spaventare l'utente e il sistema dovrebbe fornire un meccanismo semplice per informare l'utente e gestire l'errore stesso	✓
	Permettere azioni reversibili	In questo modo l'utente capisce che una qualsiasi azione può essere reversibile e sarà quindi spinto a provare funzioni ad egli ancora sconosciute	✓

Continua alla pagina successiva

Tabella 6.7: Coerenza del sistema rispetto alle 8 regole d'oro di Schneiderman

	Supportare il locus of control	L'utente dovrebbe essere sempre in grado di conoscere cosa sta succedendo al sistema in ogni istante di tempo	✓
	Riduci il carico di memoria a breve termine e i movimenti del corpo	Non dovrebbe essere richiesto all'utente di impiegare la propria memoria a breve termine né di compiere grossi spostamenti col dito sul display per compiere azioni quotidiane	✓
Notes	Relative al design delle interfacce		

6.2.1 Valutazioni euristiche

Oltre a verificare la coerenza del sistema rispetto alle regole introdotte da Shneiderman, occorre valutare l'usabilità anche per mezzo delle euristiche di Nielsen e per mezzo di test di usabilità. Attraverso le euristiche di Nielsen è stato possibile valutare l'usabilità soggettiva del sistema. Esse infatti sono regole generali e dipendenti dall'esperienza e dall'abilità del singolo utente valutatore.

Tabella 6.9: Valutazione mediante euristica di Nielsen

Obiettivo	Valutare l'usabilità tramite euristiche di Nielsen		
	Euristica	Risultato atteso	Risultato ottenuto?
	Visibilità dello stato del sistema	Impiego di feedback in tempi ragionevoli per informare l'utente	✓
	Corrispondenza tra il mondo reale e il sistema	Il sistema dovrebbe parlare la lingua dell'utente, in termini di simboli, frasi e concetti familiari	✓
	Libertà e controllo da parte dell'utente	Fornire all'utente dei pulsanti di Undo e Redo	✓
	Consistenza e standard	Seguire lo standard	✓
	Prevenzione degli errori	Impiegare metodologie di prevenzione di errore come white-list e chiedere conferma prima che un'azione critica venga eseguita	✓
	Riconoscere e non ricordare	Minimizzare l'uso della memoria dell'utente	✓
	Flessibilità ed efficienza	Metodologie di accelerazione per procedure ricorrenti	✓
	Design minimalista	Messaggi brevi e interfacce pulite	✓
	Gestione degli errori in maniera semplice	I messaggi di errore dovrebbero essere espressi in un linguaggio naturale	✓
	Supporto	Possibilità dell'utente di avere supporto	✓
Notes	Valutazioni soggettive e da complementare con i test di usabilità		

Con questo tipo di valutazione è possibile ottenere buoni risultati a costo zero impiegando molti valutatori per lo stesso sistema e che non comunichino tra loro, cercando quindi di avere valutazioni indipendenti le une dalle altre. Per questo motivo la valutazione mediante euristica di Nielsen deve necessariamente essere accompagnata da una valutazione più oggettiva attraverso i test di usabilità.

6.2.2 Test di usabilità

I test di usabilità in questa fase sono classificabili in test di compito e test di scenario. Nel primo caso l'utente svolge un singolo task che permette di specificare una funzione specifica del sistema. Nel secondo caso, invece, all'utente viene indicato un obiettivo da raggiungere attraverso una serie di compiti elementari, senza però indicarli in maniera esplicita. Per test di scenario più complessi, è possibile costruire una "storia" al contorno.

Test di compito

Ciascuno dei compiti riportati nelle seguenti tabelle, per ogni utente valutatore dell'usabilità, ha riportato un tasso di successo pari a 1. In altre parole, tutti gli utenti sono stati in grado in maniera completa e non parziale a svolgere il compito assegnato. Formalmente, su una taglia di 10 persone

$$sr = \frac{\#\text{successi}}{\#\text{valutatori}} = \frac{10}{10} = 1$$

portando quindi la probabilità di successo, in percentuale, al 100%. Questo è sintetizzato dalla seguente Tabella:

	Compito 1	Compito 2	Compito 3	Compito 4	Compito 5	Compito 6
Valutatore 1	S	S	S	S	S	S
Valutatore 2	S	S	S	S	S	S
Valutatore 3	S	S	S	S	S	S
Valutatore 4	S	S	S	S	S	S
Valutatore 5	S	S	S	S	S	S
Valutatore 6	S	S	S	S	S	S
Valutatore 7	S	S	S	S	S	S
Valutatore 8	S	S	S	S	S	S
Valutatore 9	S	S	S	S	S	S
Valutatore 10	S	S	S	S	S	S

Test di scenario

Durante la valutazione dell'usabilità per mezzo di test di scenario sono stati ipotizzati i seguenti scenari:

1. Apri per la prima volta l'applicazione e seguì il tutorial;
2. Apri lo scanner per scannerizzare una tua carta PECS;
3. Immagina di voler raggiungere in questo istante l'Antoniano. Quanti minuti di macchina ti ci vorrebbero?
4. Cerca informazioni sull'Antoniano visitando il suo sito web;
5. Esci dall'applicazione poi ritorna e verifica quanti chilometri intercorrono tra te e l'Antoniano.

Anche in questo caso tutti gli utenti scelti come valutatori hanno mostrato un tasso di successo pari a 1, come sintetizzato dalla seguente tabella:

	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5
Valutatore 1	S	S	S	S	S
Valutatore 2	S	S	S	S	S
Valutatore 3	S	S	S	S	S
Valutatore 4	S	S	S	S	S
Valutatore 5	S	S	S	S	S
Valutatore 6	S	S	S	S	S
Valutatore 7	S	S	S	S	S
Valutatore 8	S	S	S	S	S
Valutatore 9	S	S	S	S	S
Valutatore 10	S	S	S	S	S

6.2.3 5-seconds test

Un altro test che è stato compiuto a prodotto finito è il cosiddetto "5-seconds test". Il 5-seconds test è una tecnica utilizzata nel testing di usabilità ed ha come obiettivo quello di catturare la prima impressione che un gruppo più o meno ampio di valutatori ha vedendo ciascuna interfaccia per soli cinque secondi. Allo scadere dei cinque secondi, al valutatore viene sottratta l'interfaccia e gli viene chiesto se ricorda, in maniera anche vaga, cosa ci fosse su quell'interfaccia o, per meglio dire, a cosa servisse quell'interfaccia e cosa rappresentasse.

L'applicazione ha superato con successo il 5-seconds testing per ciascuna interfaccia principale e per ogni valutatore, come sintetizzato dalla seguente Tabella, dove 1 rappresenta la schermata di benvenuto, 2 l'homepage, 3 la schermata per lo scanner, 4 la schermata per il sito, 5 la schermata per la mappa.

	Int. 1	Int. 2	Int. 3	Int. 4	Int. 5
Valutatore 1	S	S	S	S	S
Valutatore 2	S	S	S	S	S
Valutatore 3	S	S	S	S	S
Valutatore 4	S	S	S	S	S
Valutatore 5	S	S	S	S	S
Valutatore 6	S	S	S	S	S
Valutatore 7	S	S	S	S	S
Valutatore 8	S	S	S	S	S
Valutatore 9	S	S	S	S	S
Valutatore 10	S	S	S	S	S

6.2.4 Analisi di coerenza

In questo paragrafo verrà fatta un'analisi di coerenza delle icone, dei colori, della legibility e della readability verificando, ad esempio, che le icone siano utilizzate in modo appropriato.

Delle icone

È importante scegliere le icone che siano più evocative per l'utente, specialmente quando egli è autistico. Ad esempio se si dovesse progettare un'icona per la funzione "Cerca" non avrebbe senso utilizzare un'icona a forma di Wi-Fi, ma occorrerebbe progettare un'icona che abbia come semantica quella di ricercare/trovare. La Tabella 6.11 mostra una valutazione della coerenza delle icone effettivamente presenti nell'applicazione rispetto al loro significato usuale o standard.

Tabella 6.11: Analisi di coerenza delle icone

Obiettivo	Valutare la coerenza delle icone presenti nell'applicazione rispetto al loro significato usuale		
	Icona	Significato usuale/standard	Significato effettivo nell'app
	Rettangolo cliccabile	Pulsante	Come da standard
	Punto interrogativo	Aiuto	Come da standard
	Impronta digitale	Scannerizzare	Come da standard

Dei colori

Lo studio dei colori non è banale: occorrerebbe trovare la miglior combinazioni di colori in base ad alcuni criteri più o meno complessi come l'obiettivo del prodotto, la finalità ed i suoi mezzi. In Tabella 6.13 è valutata la coerenza dei colori utilizzati nell'app. Da notare che i colori utilizzati nell'app sono coerenti anche con quelli utilizzati nel sito web, sviluppato precedentemente da terzi.

Tabella 6.13: Analisi di coerenza dei colori

Obiettivo	Valutare la coerenza dei colori utilizzati nell'app		
	Colore	Significato psicologico	Impiego effettivo nell'app
	Bianco	Pulizia, lumonosità	Colore principale
	Blu	Calmante, compromesso	Colore accessorio

Della legibility

Per aumentare il grado che un sistema ha di essere user-friendly, è importante ricercare gli espedienti che aumentino il più possibile il suo livello di legibility, in modo tale da consentire all'utente di distinguere più facilmente le parole. La seguente Tabella analizza il grado di legibility del sistema.

Tabella 6.15: Analisi della legibility

Obiettivo	Valutare la legibility del sistema		
	Regola	Risultato atteso	Risultato ottenuto?
	Utilizzare font standard e senza grazie, come il roboto per Android	Bisognerebbe utilizzare font standard per non affaticare la vista del lettore	✓
	Utilizzare una dimensione per il font tra il 10 e il 12	Bisognerebbe utilizzare una dimensione per il font che sia consona alla lettura	✓
	Utilizzare in maniera coerente le maiuscole e le minuscole	La leggibilità di un testo dipende da vari fattori, tra cui l'uso coerente di maiuscole e minuscole	✓
	Evitare il corsivo	Evitare il corsivo perché potrebbe portare ad un effetto di aliasing su alcuni schermi	✓
	Evitare testi lunghi	Bisognerebbe evitare testi lunghi	✓
	Preferire caratteri scuri su sfondi bianchi e viceversa	Bisognerebbe utilizzare in maniera coerente le tinte delle scritte	✓
	Non affiancare scritte di tinte spettralmente lontane	Bisognerebbe evitare di affiancare scritte che siano lontane nello spettro visivo, come ad esempio verde e blu, per evitare problemi di messa a fuoco visiva	✓
	Non veicolare tutta l'informazione per mezzo di colori	Per gli utenti daltonici	✓

Della readability

Occorre analizzare la readability del sistema per verificare la semplicità con cui l'utente acquisisce le informazioni. In questa fase viene utilizzata una funzione $f : (x, y, z) \in \mathbb{N}^3 \mapsto r \in [0; 100]$ che implementa l'indice di leggibilità (index of readability) di un sistema definita come

$$f(x, y, z) = 89 + \frac{300x - 10y}{z}$$

dove x rappresenta il numero di frasi, y rappresenta il numero di lettere e z rappresenta il numero di parole.

La seguente Tabella analizza il grado di readability del sistema.

Tabella 6.17: Analisi della readability

Obiettivo	Valutare la readability del sistema		
	Regola	Risultato atteso	Risultato ottenuto?
	Usare parole precise ma semplici	Bisognerebbe usare parole precisi ma semplici che forniscano in maniera ottimale l'informazione	✓
	Omettere parole inutili	Soprattutto preposizioni articolare	✓
	Costruire periodi/frasi brevi	Di solito sono più facilmente interpretabili	✓
	Esprimere idee simili in forma simile	Ricercare quindi la coerenza	✓
	Preferire la costruzione positiva a quella negativa	Usualmente è più chiara e diretta	✓

Capitolo 7

Confronto coi competitor

L'analisi dei competitor risulta necessaria ed utile per diversi aspetti: in primis fornisce informazioni sulle altre applicazioni simili ad Augmentus le quali possono essere utilizzate non solo per indebolire i concorrenti ma per portare numerosi vantaggi e migliorie al proprio prodotto. Un'analisi dei punti di forza dei concorrenti, infatti, può fornire informazioni utili su sistemi o meccaniche di funzionamento che possono essere implementate anche in Augmentus mentre un'analisi dei punti deboli può fornire informazioni su quali caratteristiche enfatizzare per sbaragliare la concorrenza.

7.1 Chi sono i nostri competitor?

I competitor di Augmentus sono tutte quelle applicazioni che implementano la realtà aumentata per la metodologia di CAA e PECS.

7.2 Analisi esterna

L'analisi esterna è l'analisi compiuta a partire da fonti esterne come brochure o cataloghi, siti internet pubblici, news e media, fiere, eventi, convegni, dati pubblici come bilanci e report e clienti comuni. L'analisi della concorrenza esamina l'offerta sul territorio rivolgendo l'attenzione alle caratteristiche delle imprese direttamente concorrenti (competitor), cioè tutte quelle che si rivolgono allo stesso target di persone o simile.

Un'analisi di questo tipo occorre per comprendere le diversità e le analogie più rilevanti dei prodotti o dei servizi della concorrenza. L'obiettivo che ci si pone è quello di evidenziare i nostri punti di forza (strength) e quelli di debolezza (weakness) rispetto ai concorrenti. Le informazioni raccolte dall'analisi esterna consentono di individuare gli elementi su cui poter fondare il proprio vantaggio competitivo, cioè la combinazione di risorse che assicuri al prodotto un successo imprenditoriale nel medio/lungo periodo.

7.3 Analisi interna

L'analisi interna è l'analisi compiuta a partire da fonti interne come esperienza personale, database, rappresentanti del prodotto, contatti di business, report privati di vendita e dipendenti commerciali.

È stata utilizzata l'analisi SWOT per valutare i punti di forza e debolezza, in maniera più precisa, del prodotto rintracciando le minacce e le opportunità che derivano dall'ambiente esterno oppure punti di forza e punti di debolezza che derivano dall'ambiente interno, come mostrato nel paragrafo successivo e ampiamente discusso nell'articolo *SWOT Analysis* [3].

7.4 SWOT

Questo paragrafo mostra i punti evinti dalla precedente analisi esterna ed interna disponendoli nella cosiddetta matrice SWOT.

Focus interno - punti di forza (del prodotto):

1. Conforme alle best practise della human-computer interaction e dell'ingegneria del software;
2. Combinazione innovativa tra l'AR e le metodologie CAA e PECS;
3. Semplicissima da utilizzare;
4. Modulare, portatile, efficiente;
5. L'utente già conosce la CAA e già è abituato alle rappresentazioni PECS.

Focus interno - punti di debolezza (del prodotto):

1. Finanza economica bassa;
2. Server e Database poco potenti (ma comunque adatti per il bacino di pubblico che Augmentus ospiterà e soprattutto facilmente scalabili);
3. Applicazione client esclusiva per Android.

Focus esterno - opportunità (dall'esterno che favoriscono la crescita del prodotto):

1. Inserimento di nuovi membri nel team di sviluppo, come ad esempio grafici o animatori per le carte PECS e i contenuti in AR;
2. Aumento del budget disponibile e conseguente miglioramento delle risorse e dei servizi utilizzati dall'app;
3. Fidelizzazione, pubblicità mediante le carte PECS brandizzate Istituto Antoniano.

Focus esterno - minacce (dall'esterno che favoriscono la decrescita del prodotto):

1. Mercato competitivo;
2. Potenzialmente diversi concorrenti.

Il diagramma in Figura 7.1 mostra la matrice SWOT utilizzata per questa analisi.



Figura 7.1: Matrice SWOT

Capitolo 8

Glossario

Dato personale - qualsiasi informazione riguardante una persona fisica identificata o identificabile («interessato»); si considera identificabile la persona fisica che può essere identificata, direttamente o indirettamente, con particolare riferimento a un identificativo come il nome, un numero di identificazione, dati relativi all'ubicazione, un identificativo online o a uno o più elementi caratteristici della sua identità fisica, fisiologica, genetica, psichica, economica, culturale o sociale

CRUD - tipiche operazioni di Create-Retrieve-Update-Delete effettuate su un database

Flood - abuso di messaggi o testo in generale

Paginazione - proiezione parziale dei dati al fine di preservare performance ed evitare carichi alla rete

Comprendibilità - caratteristica di un software che quantifica quanto è facile interpretare il tutto

Visibilità - caratteristica di un software che quantifica a che punto si è con lo sviluppo dello stesso

Supportabilità - caratteristica di un software che quantifica quanti strumenti di supporto ci sono

Accettabilità - caratteristica di un software che quantifica quanto sono contente le persone nel team nel ruolo assegnatogli

Affidabilità - caratteristica di un software che quantifica la qualità dello stesso

Robustezza - caratteristica di un software che quantifica quanto è robusto il software a cambiamenti in corso d'opera

Manutenibilità - caratteristica di un software che quantifica quanto è robusto e adattabile il software a cambiamenti futuri

Rapidità - caratteristica di un software che quantifica il tempo necessario al deploy

Modularità - scissione del software in micro-componenti

Riusabilità - qualità interna del software che stabilisce la possibilità di riciclare classi o, in generale, brani di codice per altri progetti

Portabilità - qualità interna del software che stabilisce su quante piattaforma possa essere eseguito

Scalabilità - qualità esterna del software che stabilisce quanto è adattabile lo stesso ad un cambiamento rispetto alla mole (crescente o decrescente) verso cui è orientata

Stateless - caratteristica del protocollo http che stabilisce che non riesce a tenere traccia delle informazioni sulla sessione di comunicazione tra client e server

Approccio "Code-and-fix" - metodologia per cui, ciclicamente, si scrive del codice, lo compila e poi lo si riprova

Capitolo 9

Bibliografia

- [1] G. Arduini, "La realtà aumentata e nuove prospettive educative," *Education sciences & society*, vol. 3, no. 2, 2012.
- [2] I. A. R. ALFIERI, G. PRATICA, and P. L. B. PRASSI, "Comunicazione aumentativa alternativa," 2009.
- [3] Wikipedia contributors, "Swot analysis — Wikipedia, the free encyclopedia." https://en.wikipedia.org/w/index.php?title=SWOT_analysis&oldid=1035678737, 2021. [Online; accessed 28-July-2021].

Elenco delle figure

1.1 Esempi di PECS	9
2.1 Use Case Diagram	12
2.2 Homepage	13
2.3 Servizio AR	14
2.4 Schermata sito web	15
2.5 Schermata mappa	16
2.6 Protipazione visuale di media fedeltà delle principali schermate	17
2.7 Class Diagram di Analisi - Scannerizza carta	21
2.8 Class Diagram di Analisi - Visualizza sito web	22
2.9 Class Diagram di Analisi - Visualizza percorso per raggiungere l'Antoniano	23
2.10 Sequence Diagram di Analisi - Scannerizza carta	24
2.11 Sequence Diagram di Analisi - Visualizza sito web	25
2.12 Sequence Diagram di Analisi - Visualizza percorso per raggiungere l'Antoniano	26
2.13 Statechart - Scannerizza carta	27
2.14 Statechart - Visualizza sito web	28
2.15 Statechart - Visualizza percorso per raggiungere l'Antoniano	29
2.16 Activity Diagram - Scannerizza carta	30
2.17 Activity Diagram - Visualizza sito web	31
2.18 Activity Diagram - Visualizza percorso per raggiungere l'Antoniano	32
2.19 Project Management - Work Breakpoint Structure	34
2.20 Project Management - Diagramma di PERT	35
2.21 Project Management - Costruzione e Distruzione	36
3.1 Sistemi user-centered	37
3.2 User pool target	38
3.3 Distribuzione delle variabili comportamentali	39
3.4 Marco Tulli - Primary personas type	41
3.5 Giulia Improta - Secondary personas type	42
3.6 Rebecca Caputo - Negative personas type	43
3.7 Primary-personas goal e business goal	44
3.8 Primary-personas goal e secondary-personas goal	45
3.9 Negative-personas goal e business goal	46
4.1 Spazio dei prototipi in relazione al loro scopo	47
4.2 Prototipazione iniziale "a schizzo" della homepage	48
4.3 Prototipazione iniziale "a schizzo" del prodotto	49
5.1 Architettura del Sistema	51
5.2 Class Diagram di Design	55
5.3 Sequence Diagram di Design - Scannerizza carta	63

ELENCO DELLE FIGURE

85

5.4 Sequence Diagram di Design - Visualizza sito web	64
5.5 Sequence Diagram di Design - Visualizza percorso per l'Antoniano	65
7.1 Matrice SWOT	81

Elenco delle tabelle

2.1	Scannerizza carta	18
2.3	Visualizza sito web	19
2.5	Visualizza percorso	20
6.1	Test - Scannerizza carta	66
6.3	Test - Visualizza sito web	67
6.5	Test - Visualizza percorso per raggiungere l'Antoniano	68
6.7	Coerenza del sistema rispetto alle otto regole d'oro di Schneiderman	69
6.9	Valutazione mediante euristica di Nielsen	71
6.11	Analisi di coerenza delle icone	75
6.13	Analisi di coerenza dei colori	76
6.15	Analisi della legibility	77
6.17	Analisi della readability	78