UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

"Progettazione, sviluppo e test di un'applicazione Android per la configurazione di dispositivi domotici"

Relatore: Prof. Fantozzi Carlo

Laureando: Bernardi Nicolò

ANNO ACCADEMICO 2022 – 2023

16 novembre 2023

Sommario

Il presente documento descrive il lavoro svolto durante il periodo di tirocinio dal 4/09/2023 al 16/10/2023, della durata di circa 255 ore, presso l'azienda VIMAR S.p.A. nella sede di ricerca e sviluppo di Padova. L'attività svolta durante questa esperienza lavorativa riguarda la progettazione, sviluppo e implementazione di una nuova funzionalità della app VIEW di Vimar. Questo sviluppo ha lo scopo di permettere all'utente, che dispone di dispositivi della serie civile domotica Linea XT By-Me, dotati di matrici di LED, di poter adottare e associare un'icona a queste matrici tra un set predefinito di icone, attraverso l'utilizzo di un'apposita sezione dell'app. La funzionalità è stata sviluppata per dispositivi Android utilizzando il linguaggio Kotlin e utilizzando l'IDE Android Studio.

Indice

Capitolo 1

Introduzione

1.1 L'azienda

L'attività di tirocinio si è svolta presso l'azienda VIMAR S.p.A., un'azienda italiana che produce materiale elettrico per impianti, domotica, sistemi intelligenti e interconnessi in ambito residenziale e terziario. Viene fondata il 1º maggio 1945 dagli imprenditori Walter Viaro e Francesco Gusi a Marostica, in provincia di Vicenza, e Il nome della società deriva dall'unione delle lettere iniziali del cognome di Walter Viaro, uno dei suoi fondatori, e dal nome del comune sede dell'azienda: *VI*aro *MAR*ostica. Oggi VIMAR è uno dei principali player italiani nel settore elettronico ed elettrico di bassa tensione. Con oltre 12.000 articoli distribuiti in più di 100 nazioni, 185 milioni di pezzi prodotti all'anno, 1.300 collaboratori, di cui 1.000 in Italia, 4 stabilimenti a Marostica, 9 filiali commerciali nel mondo, l'azienda è un punto di riferimento per coloro che cercano soluzioni performanti, qualità del prodotto, innovazione tecnologica, ma anche un design in linea con le più attuali tendenze del mercato.[1]

Figura 1.1: Logo VIMAR

1.2 Domotica

La domotica è uno dei campi di cui si occupa VIMAR e che è alla base di ciò che si è andato a sviluppare, in quanto i dispositivi con cui il progetto ha interagito sono domotici.

Con domotica intendiamo quella disciplina, sintesi di diverse altre discipline, tra cui l'informatica, l'elettronica e le comunicazioni digitali, che si occupa dello studio delle tecnologie volte a migliorare la qualità della vita nella casa e più in generale negli edifici. Il termine deriva dal neologismo francese domotique, che a sua volta derivante dal latino "domus", che significa casa, accostato al termine informatique. La domotica, quindi, permette di rendere intelligenti apparecchiature e sistemi attraverso gli impianti elettrici e unità di gestione, con lo scopo di migliorare il controllo, comfort, sicurezza, gestione energetica e comunicazione negli edifici.[2] [3]

Le tecnologie sopra descritte, applicate ad un'abitazione, danno vita al concetto di Smart Home. Con Smart Home si indica un ambiente abitativo, opportunamente progettato e tecnologicamente attrezzato, il quale mette a disposizione dell'utente impianti, dotati di sensori e automazioni, dove apparecchiature e sistemi sono in comunicazione tra loro e sono in grado di svolgere funzioni autonome o parzialmente autonome. Queste funzioni autonome o parzialmente autonome possono attivarsi in base a parametri ambientali rilevati o comandi impartiti o programmati dall'utilizzatore, da locale o remoto.[4] [5]

Gli elementi fisici principali su cui si basa la Smart Home sono i seguenti.

- Rete interna: via cavo, wireless
- Unità di controllo intelligente: gateway per gestire le funzionalità dei sistemi
- Domotica: dispositivi adatti presenti nell'abitazione collegati a sistemi o servizi esterni

Con il termine gateway in linea generale si indica il servizio di inoltro dei pacchetti generati da una rete locale verso una esterna. In pratica il gateway ha il compito di tradurre i dati dal protocollo usato dalla rete sorgente e a quello della rete di destinazione.

Dal punto di vista domotico questo ha lo scopo di fare da interfaccia tra il software di gestione e i dispositivi domotici del sistema collegati ad esso, traducendo i messaggi del sistema domotico in quelli ad alto livello della piattaforma e viceversa. Oltre a ciò ha anche l'ulteriore scopo di creare un livello logico di astrazione per i dispositivi connessi, in modo da nascondere al sistema di controllo le caratteristiche hardware e software dei dispostivi collegati.[6]

Vimar si introduce sul mercato della domotica e dello Smart Home alla fine degli anni Novanta con By-me, il sistema di automazione connessa dedicato al controllo completo di luci, temperatura, diffusione sonora, automazione di tende e tapparelle, irrigazione, gestione energetica e termoregolazione multi-zona. Questo sistema permette la gestione e la supervisione dei dispostivi domotici dell'utente, prodotti dalla Vimar e non, basandosi su dei gateway proprietari come interfaccia per i dispositivi di controllo. I gateway di By-Me possono essere di 3 tipi: Due fili, dedicati ai sistemi videocitofonici più datati, Bluetooth e IP, dedicati a sistemi videocitofonici più moderni e avanzati e alla domotica. Ai gateway vengono collegati tutti i dispositivi domotici dell'impianto dell'utente e il loro scopo è quello di permettere il dialogo tra gli elementi collegati e i sistemi di gestione interni locali e remoti, proprietari di VIMAR, tra cui l'app VIEW. [1]

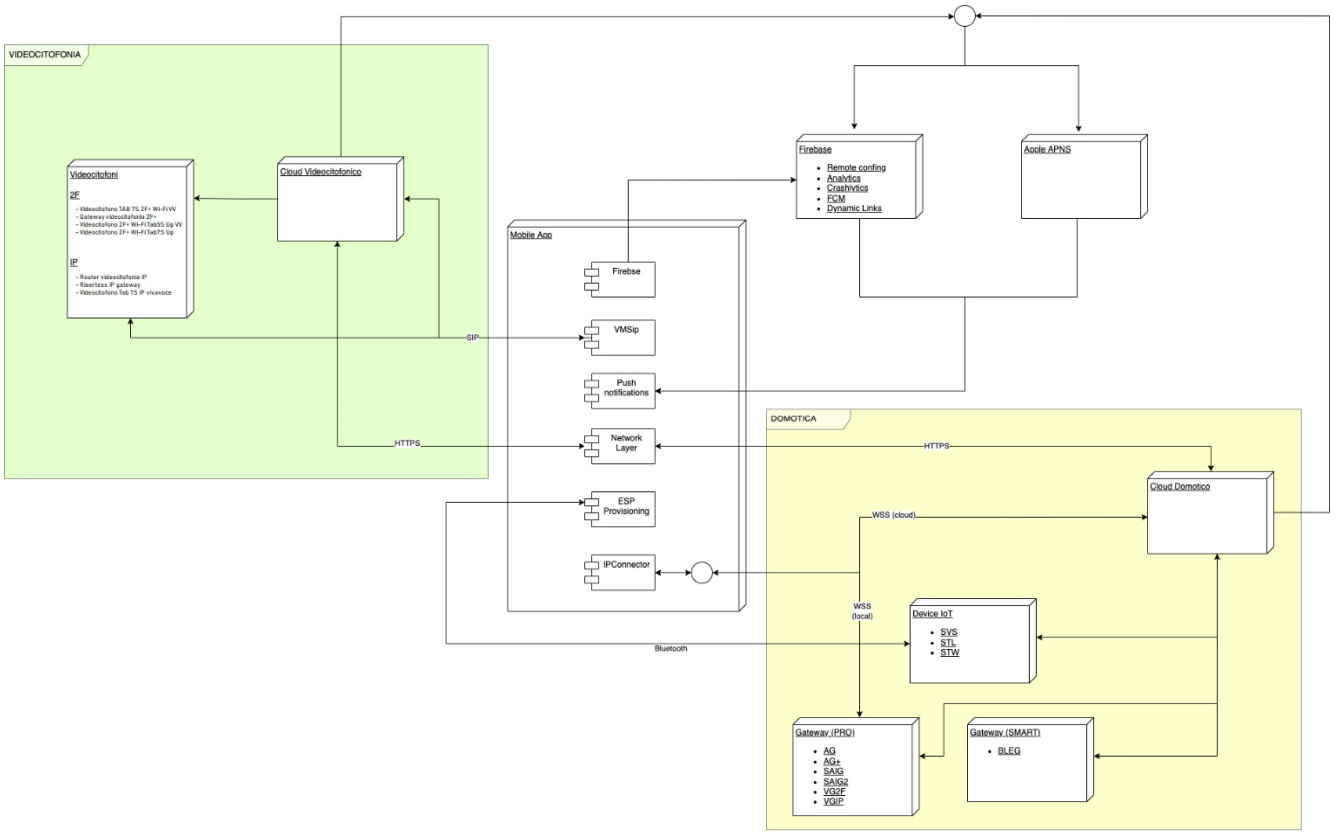

Figura 1.2: Schema ecosistema Vimar

1.3 Linea XT

I dispositivi con cui ha interagito il progetto, che ne hanno resa necessaria la creazione, sono quelli della Linea XT By-Me di VIMAR. Sono dei comandi domotici caratterizzati da una grande scalabilità ed espandibilità senza interventi di cablaggio: ad esempio, danno la possibilità di variare la propria funzione di controllo con un semplice comando da dispositivo remoto o locale. Inoltre l'altra importante caratteristica, la quale interessa al progetto, è la presenza sulla superficie di questi pulsanti di matrici configurabili a LED 5X5, che possono arrivare attualmente fino ad un massimo di 3 matrici per dispositivo a seconda del modello.

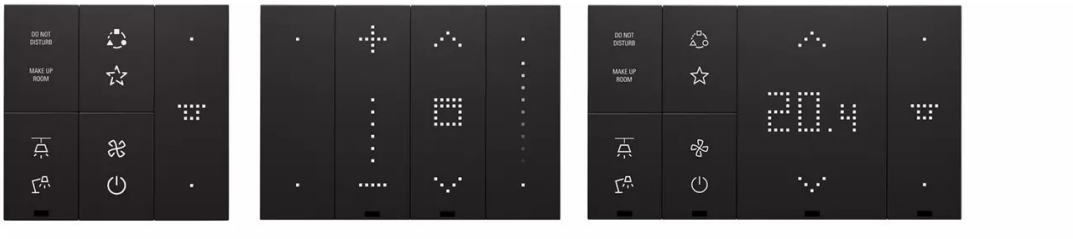

Figura 1.3: Linea XT By-Me di VIMAR versione nera

I dispositivi della linea XT che sono stati interessati dal progetto e che dispongono di matrici configurabili dall'utente, indicate dai puntini neri nella rappresentazione grafica del comando, sono:

- Comando XT By-Me 1M (cod. art. 32021), dotato di sola matrice LED 5X5 centrale

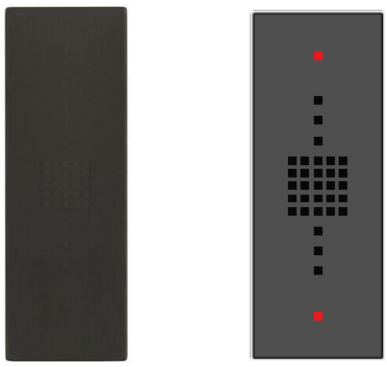

Figura 1.4: confronto tra pulsante reale cod. 32021 e rappresentazione grafica

- Comando XT By-Me advanced 1M (cod. Art. 32023), dotato di 3 matrici LED 5X5: una centrale, una nella parte bassa e una nella parte alta

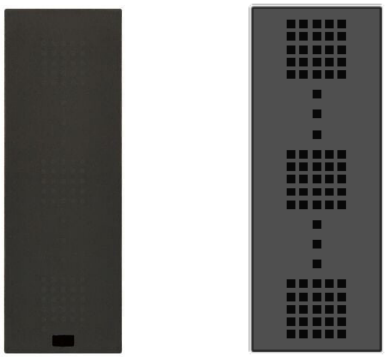

Figura 1.5: confronto tra pulsante reale cod. 32023 e rappresentazione grafica

- Multisensore XT By-me 2M (cod. Art. 32042), dotato di una sola matrice LED 5X5 centrale

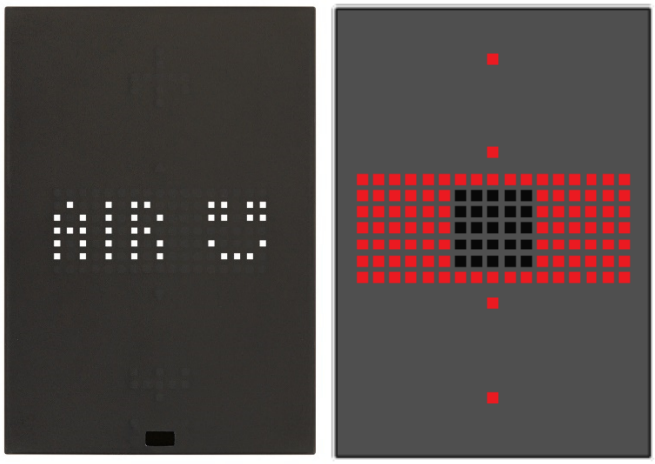

Figura 1.6: confronto tra pulsante reale cod. 32042 e rappresentazione grafica

1.4 Obiettivi

L'obiettivo principale di questo tirocinio dal punto di vista lavorativo è stato quello di riuscire a implementare la nuova funzionalità dell'app che mi è stata assegnata, soddisfacendo tutti i requisiti definiti dall'azienda nel piano di lavoro. Lo scopo finale è stato quindi di rendere questo sviluppo pronto per l'eventuale rilascio in produzione da parte dell'azienda.

Dal punto di vista personale invece l'obiettivo di questo tirocinio è stato quello, interfacciandosi con il mondo del lavoro, di apprendere tecniche, metodologie e strumenti usati in ambito lavorativo e professionale dello sviluppo Android. Inoltre, un ulteriore scopo di questa esperienza è stato anche quello di approfondire ciò che ho imparato negli anni di Università ed ampliare le mie conoscenze nel campo dello sviluppo Android, grazie anche all'affiancamento con un team di sviluppatori esperti.

1.5 Descrizione progetto

Il progetto di tirocinio si è basato sulla creazione di una nuova sezione e funzionalità alla già esiste applicazione Vimar VIEW, nella sua versione per dispositivi Android. Questa nuova funzionalità ha lo scopo di permettere all'utente, che dispone di dispositivi della serie civile domotica Linea XT By-Me, dotati di matrici LED, di poter scegliere e associare un'icona a queste matrici da un set predefinito di icone, attraverso l'utilizzo di un'apposita sezione dell'app. Il progetto ha avuto una durata di 255 ore ed è stato svolto presso la sede del dipartimento di ricerca e sviluppo della Vimar di Padova, con l'inserimento nel team di sviluppo Android.

Il linguaggio utilizzato dall'azienda per lo sviluppo delle app Android, e che è stato usato anche per questo progetto è Kotlin.

1.6 Piano di Lavoro

Le 255 ore del progetto di tirocinio sono state suddivise tra:

1) Introduzione all'azienda
2) Introduzione ai prodotti aziendali ed in particolare ai gateway di automazione domotica
3) Introduzione alla metodologia di lavoro adottata e agli strumenti di lavoro
4) Introduzione alle procedure di versionamento e alle norme di codifica adottate
5) Introduzione e studio del protocollo IpConnector
6) Introduzione all'app VIEW, alla relativa architettura ed alle librerie proprietarie adottate
7) Analisi e studio dei requisiti delle nuove funzionalità
8) Analisi e studio delle tavole grafiche
9) Progettazione architetturale
10) Codifica delle nuove funzioni di sistema nella libreria IpConnector
11) Sviluppo della nuova funzionalità nell'app VIEW
12) Progettazione e codifica della UI
13) Testing automatico

Capitolo 2
Metodologie

2.1 Metodo di lavoro

Il team di sviluppo Android nel quale sono stato inserito adotta un ciclo di sviluppo del software di tipo agile: in particolare, utilizza il metodo Scrum.[7] Questo prevede un modello iterativo nel quale ogni ciclo di sviluppo viene definito sprint: nel nostro team ogni sprint ha una durata di due settimane. Il primo lunedì della prima settimana dello sprint viene effettuata la retrospective delle due settimane precedenti e lo sprint planning. Nella retrospective il team analizza lo sprint precedente appena concluso con i risultati raggiunti e gli eventuali problemi riscontrati. Nello sprint planning vengono definiti tutti i vari ticket delle due settimane successive, i quali consistono in due tipologie di lavoro: i task, dove si implementano nuove funzionalità, e il bug fixing dove si risolvono problemi segnalati a componenti precedentemente create. Quotidianamente alle 9.15 viene fatto uno stand-up meeting di 15 minuti con i membri del team, nel quale si descrive ciò che si è fatto nel giorno precedente, eventuali problemi riscontrati e ciò che verrà eseguito nel giorno stesso. Infine, occasionalmente, viene fatto un poker planning dove si stima quante ore uomo potrebbero occupare i vari ticket. Con poker planning si intende una pratica per la stima dei tempi di realizzazione in cui si prendono i vari ticket e, per ognuno, i membri del team mostrano una carta che indica il numero di ore per esso stimate necessarie; fatto ciò si definisce la durata del task come la media dei valori delle carte mostrate dai membri.

Per quanto riguarda invece la politica aziendale sugli sviluppi delle nuove funzionalità o progetti, questi vengono programmati a fine anno per i successivi due anni, mentre, per quanto riguarda i rilasci di nuove versioni, nel Play store per Android e nell'App store per iOS, queste vengono fatte quattro volte l'anno.

2.2 Procedure di versionamento del codice

Secondo regole aziendali e del team di sviluppo, nel quale sono stato inserito, il codice completo dell'app VIEW, nella quale si è andati ad inserire gli sviluppi del progetto, è presente in due versioni principali: la prima è quella che presenta il codice attualmente in produzione e cioè l'ultima versione dell'app rilasciata agli utenti, mentre la seconda consiste nell'app con tutte le modifiche, nuovi sviluppi e rimozioni di bug fatte dall'ultimo rilascio; questa seconda versione verrà fornita agli utenti al prossimo aggiornamento.

Per quanto riguarda invece il versionamento per lo sviluppo di nuove funzionalità, interessato dal progetto, questo viene fatto in un nuovo branch dell'app, creato dalla versione del codice che dovrà essere rilasciata. Successivamente, per lo sviluppo vengono assegnati una serie di task, i quali rappresentano delle singole funzioni da implementare fino alla completa creazione della nuova funzionalità. Per ogni task quindi, man mano che si avanza con il loro svolgimento, si crea un ulteriore branch, dal precedente, dove si sviluppa l'incremento descritto dal task. Una volta terminato il task si apre un pull request (pr) su Jira Software, dove ogni membro del team controlla il codice prodotto durante lo sviluppo del task. Al termine dell'approvazione del codice da parte di tutti i membri, la pr viene chiusa assieme al task e viene fatto il merge del codice prodotto con il branch precedente. Al termine di tutti i task, dopo l'approvazione da parte del team, viene fatto il merge del codice prodotto nel branch di sviluppo con la versione di preproduzione dell'app.

Infine, durante lo sviluppo della funzionalità del task, viene spesso eseguito, dopo una serie di modifiche importanti al codice, un commit e push del codice al branch in cui si sta lavorando, il quale viene accompagnato da una breve descrizione di ciò che si è fatto in quelle modifiche. Ciò viene fatto sia per tenere traccia di tutti i cambiamenti che vengono fatti al codice, sia per avere un punto stabile in cui fare un roolback delle modifiche in caso di problemi successivi.

2.3 Strumenti

2.3.1 Android Studio

Figura 2.1: logo Android Studio

Android Studio è l'ambiente di sviluppo integrato (IDE) ufficiale per lo sviluppo di app Android. [8] Si basa sul software IntelliJ IDEA e fornisce una serie di strumenti a supporto dello sviluppo come:

- Un sistema di compilazione basato su Gradle
- Un debugger
- Una serie di modelli di base a seconda della tipologia di app che si vuole creare
- Un emulatore virtuale
- Integrazione con Git
- Strumenti di test e log
- Supporto per C++ e NDK

2.3.2 Bitbucket

Figura 2.2: logo Bitbucket

Bitbucket è uno strumento web based, fornito da Atlassian, utilizzato in Vimar come repository per il codice e per il controllo di versione, vista l'integrazione con Git. Questo strumento oltre che a tenere traccia del controllo versione permette anche al team di lavorare in collaborazione sul codice e l'eventuale creazione di branch di lavoro. [9] Può lavorare in abbinamento con altri due prodotti Atlassian che sono Confluence e Jira Software.

2.3.3 Confluence

Figura 2.3: logo Confluence

Confluence è una piattaforma collaborativa fornita da Atlassian [10]. È stata usata nel nostro caso come strumento di collaborazione per la creazione ed organizzazione di tutta la documentazione legata ai prodotti software, la quale viene sempre resa disponibile nella sua versione più aggiornata ai membri del team. Inoltre questo strumento fornisce anche un sistema di controllo sui permessi di accesso ai documenti.

2.3.4 Jira Software

Figura 2.4: logo Jira Software

Jira Software è uno strumento, fornito da Atlassian, per il tracciamento, monitoraggio, bug tracking e rilascio dei progetti sviluppati con metodologie agili. [11] Utile nel nostro caso per permettere al team leader di assegnare i task ai vari sviluppatori, indicando anche il branch di lavoro.

2.3.5 Git

Figura 2.5: logo Git

Git è un sistema di controllo delle versioni distribuito, gratuito e open source, progettato per la gestione dei progetti. [12] Nel nostro caso è stato utilizzato in associazione con Android Studio e Bitbucket per il controllo di versione del codice del progetto.

2.3.6 Ktlint

Figura 2.6: logo ktlint

Ktlint è uno strumento di linting, il quale consiste nel processo di analisi del codice sorgente, utilizzato per controllare la formattazione del codice scritto in Kotlin: questo permette di avere un codice pulito in linea con uno stile standard. Ktlint quindi permette di segnalare gli errori di formattazione, che individua rispetto alle linee guida, e anche in alcuni casi correggerli automaticamente. [13] Nell'azienda viene utilizzato per allineare in codice prodotto in collaborazione con i membri del team che lavorano allo stesso progetto, secondo il coding style ufficiale consigliato da Google: Kotlin Code Style.[14]

2.3.7 Microsoft Teams

Figura 2.7: logo Microsoft Teams

Microsoft Teams è una piattaforma di comunicazione che fornisce chat di lavoro persistenti, funzioni per videochiamate tra più utenti e la possibilità di condividere contenuti. [15] Nel nostro caso è stato utilizzato per le comunicazioni con tutti i membri del team e della azienda, e per i vari meeting. Questo sistema è molto utile in quanto, visto la presenza di più sedi e di dipendenti in smart working, permette di rimanere sempre in contatto con tutti i colleghi e membri del team.

2.3.8 Adobe XD

Figura 2.8: logo Adobe XD

Adobe XD è una piattaforma sviluppata da Adobe che consente ai progettisti di UI di progettare esperienze utente interattive per Web e app per dispositivi mobili, disegnandole su apposite tavole da disegno.[16] Nel nostro caso è stato utilizzato solo per la consultazione delle tavole grafiche, le quali ci sono state fornite dai grafici, contenenti l'aspetto e i dettagli di colori, testo e dimensioni degli elementi dell'interfaccia.

Capitolo 3

APP Vimar VIEW

3.1 Descrizione

L'applicazione a cui sono stati aggiunti gli elementi di questo progetto è Vimar VIEW. App della VIMAR che permette all'utente di interfacciarsi e controllare gli impianti domotici VIMAR da remoto e non, partendo dal controllo delle semplici luci di casa, alla videocitofonia, fino ad arrivare all'antintrusione.[1]

L'applicazione è disponibile sia per sistemi Android, a partire dalla versione 8.0 (API level 26), sia per iOS dalla versione 14.

Figura 3.1: Immagini dalla pagina del Play Store dell'app Vimar VIEW

3.2 Struttura

In Android l'app VIEW è strutturata in più moduli i quali, a loro volta, raggruppano al loro interno gli elementi del codice per aree di utilizzo e interesse. Altra caratteristica di VIEW è il fatto che presenta un'unica activity con all'interno un FragmentContainer, permettendo così di gestire tutte le varie schermate attraverso vari Fragment e Navgraph.

All'interno dell'app sono presenti anche varie cartelle con qualificatore le quali permettono di gestire tutte le varie risorse, come file drawable, stringhe e colori, in base allo schermo, lingua e modalità dark o light del dispositivo.

Per scelta aziendale, inoltre, l'applicazione funziona solo in modalità portrait, restando quindi in quest'unica orientazione anche in caso di rotazione del dispositivo .

3.3 Librerie

Le principali librerie proprietarie su cui si basa l'app VIEW sono IpConnector e VMSip. Queste librerie, a seconda del tipo di impianto, possono lavorare in coppia o singolarmente.

La libreria IP Connector è una libreria che implementa l'omonimo protocollo VIMAR. Il protocollo IP Connector è finalizzato alla comunicazione bidirezionale tra un "server" e uno o più "client", attraverso un collegamento di rete (sia locale sia remoto) per svolgere le seguenti funzioni:

- Identificare automaticamente la presenza di server in rete
- Registrare il client come interlocutore autenticato nel server
- Ricevere informazioni sulla struttura e le risorse esportate dal server
- Inviare comandi e richieste di stato in modo sincrono
- Ricevere informazioni di cambio stato in modo asincrono

Il protocollo prevede inoltre meccanismi di protezione della comunicazione, gestione di particolari situazioni e ripristino del canale di comunicazione. IpConnector quindi permette la comunicazione da parte dei dispositivi esterni con i gateway domotici VIMAR e, di conseguenza, con gli elementi domotici collegati ad esso. Il protocollo di comunicazione prevede l'incapsulamento delle informazioni in formato JSON, che prevede una sintassi specifica degli elementi inseriti. Le informazioni relative ai sottosistemi esportati dai dispositivi di tipo gateway sono strutturate secondo la specifica delle System Functions Vimar (SF). Ognuna di queste consiste in un oggetto comandabile dal sistema. Per ogni SF inoltre sono previsti una serie di System Functions Elements (SFE), ciascuno dei quali determina una singola funzione atomica del dispositivo (ad esempio lo stato on/off del dispositivo). La libreria IpConnector incapsula tutto ciò permettendo all'app di inviare e ricevere dati dai vari gateway attraverso delle semplici funzioni e inoltre organizza tutti i dati ricavati o inviati in comode DataClass. Quest'ultime sono delle classi che hanno lo scopo principale di contenere i dati, fornendo anche una serie di funzioni che permettono di maneggiarli agevolmente.

VMSip è invece una libreria proprietaria di Vimar che ha lo scopo di comunicare con gli elementi citofonici che siano dotati o no di video. Attraverso questa libreria è quindi possibile registrare gli smartphone ai vari dispositivi citofonici, permettendo così di inviare e ricevere comandi, messaggi e chiamate SIP dall'impianto. Questa libreria consiste in un wrapper ad un'ulteriore libreria, chiamata Linphone [17], disponibile open-source da un'azienda specializzata esterna; ciò è necessario perché Vimar necessita di ulteriori dati e funzioni durante la comunicazione, rispetto a quelle fornite di base. VMSip inoltre può lavorare in comunicazione mista con IpConnector quando si devono gestire determinati tipi di impianti citofonici che necessitano di entrambe le librerie: in questo tipo di sistemi IpConnector fornisce i contatti dei dispositivi sui quali VMSip esegue il comando.

3.4 Pattern

L'app VIEW di VIMAR, e quindi anche il progetto che si ha sviluppato, si basa sul pattern architetturale Model-View-ViewModel (MVVM) con Clean Architecture.

L' MVVM è un pattern architetturale che permette di avere una netta separazione tra l'interfaccia utente, i dati e la business logic di un'applicazione.

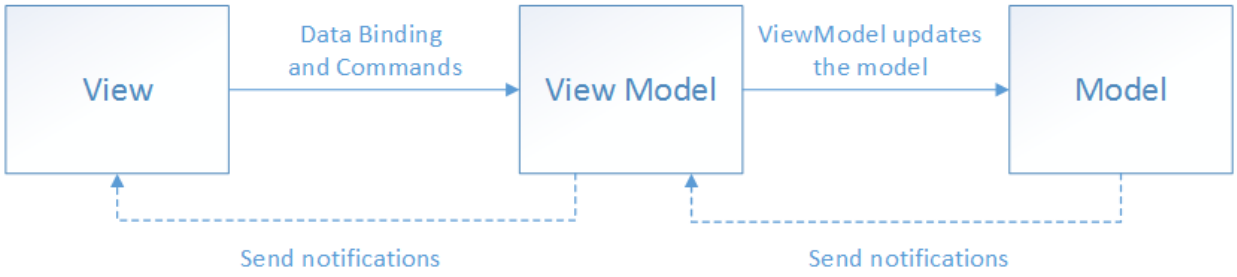

Figura 3.2: Rappresentazione schematica Model-View-ViewModel (MVVM)

MVVM si basa principalmente su tre elementi:

- **View**: Consiste nell'interfaccia grafica che mostra le informazioni all'utente e ne gestisce l'aspetto e la struttura, ad esempio può essere un Fragment
- **ViewModel**: Classe che ha la responsabilità di fornire i dati necessari alla View e fa da ponte tra essa e il Model
- **Model**: Sono classi che contengono i dati dell'app, repository o business logic

In particolare la View osserva e notifica il ViewModel in caso di interazioni con l'utente. Il ViewModel invece non ha alcun riferimento alla View, ma in base alle notifiche che riceve è in grado di osservare e modificare il Model. Il Model in caso di cambiamenti notifica il ViewModel il quale, a sua volta, li rende disponibili alla View. [18]

Con Clean Architecture si intende invece una struttura del codice dell'applicazione in cui questo viene suddiviso in livelli che aderiscono ad una regola: il livello interno non conosce nulla del livello esterno e inoltre il livello esterno dipende da quello interno per il suo funzionamento.

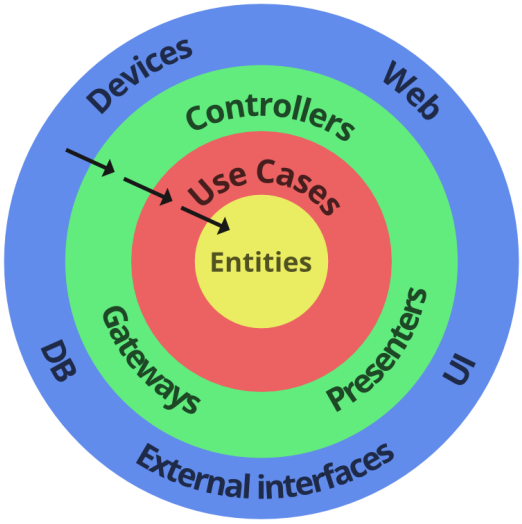

Figura 3.3: Rappresentazione struttura a livelli della Clean Architecture

Più precisamente, come possiamo vedere dalla Figura 4.3, questa architettura colloca nell'anello più esterno l'interfaccia utente, la quale recupera i dati dal livello successivo che è il livello di presentazione, dove nel nostro caso si collocano i ViewModel. Questi quindi possono utilizzare gli elementi del livello inferiore chiamati UseCase per accedere ai dati veri e proprio contenuti nel nucleo del modello. Lo scopo della Clean Architecture è quello di rendere il codice scalabile e facile da mantenere: infatti questa separazione in livelli permette di eseguire agevolmente modifiche ai vari elementi. [19]

Capitolo 4
Analisi dei requisiti

4.1 Esigenze

L'azienda per lo sviluppo di questo progetto mi ha fornito una lista di otto esigenze tramite le quali ricavare i requisiti necessari per lo sviluppo. Queste otto esigenze raccolgono tutte le caratteristiche necessarie che doveva avere il progetto per permettere all'utente di personalizzare secondo specifiche le matrici LED dei dispositivi tramite l'app. Inoltre all'interno di queste esigenze ci sono anche degli elementi che interessano una predisposizione per implementazioni future, come l'inserimento delle icone personalizzate e dinamiche. Per il testo completo delle varie esigenze consultare l'appendice A al punto specificato per ogni esigenza.

Le otto esigenze fornite sono:

- **ESG01-001**: Si richiede che i comandi digitali della Linea XT By-me siano personalizzabili con una matrice di LED dinamica dall'app VIEW. (Appendice A.1)

- **ESG01-002**: Personalizzazione comando XT By-me 1M (32021.x). (Appendice A.2)

- **ESG01-003**: Personalizzazione comando XT By-me advanced 1M (32023.x). (Appendice A.3)

- **ESG01-004**: Personalizzazione Comando XT By-me multisensore 2M (32042.x). (Appendice A.4)

- **ESG01-005**: La matrice LED deve essere personalizzabile tramite un set di icone predefinite, personalizzabili o dinamiche. (Appendice A.5)

- **ESG01-006**: Le icone devono essere mostrare in una lista di anteprime selezionabili (Appendice A.6)

- **ESG01-007**: Un'icona potrà essere applicabile in "bulk action" a tutte le matrici dello stesso tipo o solo alla matrice scelta per la modifica. (Appendice A.7)

- **ESG01-008**: Si richiede la possibilità di personalizzare le matrici LED con le personalizzazioni disponibili a seconda della funzione a cui è associata la matrice (Appendice A.8)

4.2 Requisiti

Dalle esigenze specificate dall'azienda sono stati ricavati, a seguito dell'analisi, i seguenti requisiti che dovranno essere implementati per la realizzazione del progetto. Per il testo completo dei vari requisiti si può consultare l'appendice B al punto specificato per ogni esigenza.

REQ01-10 riferito all'esigenza ESG01-001: L'utente deve avere a disposizione all'interno dell'app VIEW un'interfaccia grafica specifica di personalizzazione. Precisamente l'utente dovrà navigare fino alla lista dei dispositivi presenti dove, una volta selezionato il dispositivo, verrà visualizzata una schermata in cui sarà possibile scegliere la posizione della matrice che si vuole personalizzare e infine, premendo su uno dei massimo tre slot, verrà visualizzata la schermata di scelta dell'icona. (Appendice B.1)

REQ01-020 riferito all'esigenza ESG01-002: All'interno dell'app VIEW verrà collegata la personalizzazione di questo componente con una schermata che permetterà di scegliere attraverso un riquadro cliccabile solo la matrice centrale, l'unica disponibile nel dispositivo. Premendo questo riquadro si arriverà alla pagina di selezione dell'icona. (Appendice B.2)

REQ01-030 riferito all'esigenza ESG01-003: All'interno dell'app VIEW verrà collegata la personalizzazione di questo modello di componente con una schermata che fornisce, attraverso dei riquadri cliccabili, la possibilità di personalizzare tutte e tre le matrici presenti. L'utente premendo su un riquadro andrà alla pagina di selezione dell'icona per quella determinata matrice. (Appendice B.3)

REQ01-040 riferito all'esigenza ESG01-004: All'interno dell'app VIEW verrà collegata la personalizzazione di questo componente con una schermata che permetterà di scegliere attraverso un riquadro cliccabile solo la matrice centrale, unica disponibile nel dispositivo. Premendo questo riquadro si arriverà alla pagina di selezione dell'icona. Questo dispositivo è personalizzabile solo se utilizzato in modalità "comando" (Appendice B.4)

REQ01-050 riferito all'esigenza ESG01-005: L'utente, dopo aver premuto sul punto desiderato nella schermata di scelta della posizione dell'icona da inserire, vedrà una nuova schermata dove sarà presente inizialmente la lista, suddivisa in pagine, delle icone predefinite tra cui può scegliere. In questa schermata saranno inoltre presenti in alto tre tasti, denominati rispettivamente "Predefinite", "Personalizzate" e "Dinamiche". Questi tasti, se premuti, portano alle rispettive pagine di scelta delle icone predefinite, personalizzate e dinamiche. (Appendice B.5)

REQ01-060 riferito all'esigenza ESG01-006: Quando l'utente, che sia nella schermata di selezione delle icone predefinite, dinamiche o personalizzate, ha scelto e cliccato l'icona che vorrà applicare a quella determinata matrice, potrà confermare e applicare la sua scelta premendo il tasto "Fatto". Una volta premuto verrà visualizzato un pop-up che indica la conferma dei cambiamenti. Inoltre, se un utente esegue la selezione, cioè il clic su un'icona su una determinata pagina, questa verrà evidenziata da un contorno di colore diverso. La scelta verrà evidenziata anche nei bottoni di selezione della categoria attraverso la comparsa di un puntino grigio. (Appendice B.6)

REQ01-070 riferito all'esigenza ESG01-007: All'utente, una volta selezionata e confermata una determinata icona, qualora alla funzione selezionata siano associati più matrici o tasti dello stesso tipo, viene data la possibilità di applicare la stessa personalizzazione contemporaneamente a tutti, oppure solo al tasto o alla matrice che si sta personalizzando. Questo viene fatto attraverso un pop-up di scelta. (Appendice B.7)

REQ01-080 riferito all'esigenza ESG01-008: Per permettere all'utente di personalizzare le matrici Led per funzione nella schermata di personalizzazione del dispositivo vengono disposti i quadratini, cliccabili per l'assegnamento dell'icona alla posizione, solo nelle posizioni che posso essere personalizzabili in quel dispositivo usato per quella determinata funzione, oscurando quelli non disponibili. Inoltre una volta che l'utente preme in uno dei quadratini viene mandato alla schermata di selezione dell'icona dove visualizzerà e potrà selezionare solamente le icone e le animazioni che può utilizzare in quel determinato dispositivo usato per quella determinata funzione; per la lista delle varie casistiche consultare l'appendice nella sezione B.8.

4.3 Tabella riassuntiva

Le varie relazioni tra esigenze fornite e requisiti definiti sono riassunte nella tabella sottostante:

ID Requisito	Descrizione	ID Esigenza	Data Requisito
REQ01-010	Personalizzazione comandi digitali della Linea XT By-me da app VIEW, selezionando le icone da una libreria predefinita	ESG01-001	6/09/2023
REQ01-020	Metodo di selezione icona per dispositivo con singola matrice personalizzabile centrale [Comando XT By-me 1M (32021.x)]	ESG01-002	6/09/2023
REQ01-030	Metodo di selezione icona per dispositivo con tre matrici personalizzabile [Comando XT By-me advanced 1M (32023.x)]	ESG01-003	6/09/2023
REQ01-040	Metodo di selezione icona per dispositivo con singola matrice personalizzabile centrale [Comando XT By-me multisensore 2M (32042.x)]	ESG01-004	6/09/2023
REQ01-050	Scelta dell'icona tra i tre tipi di set disponibili	ESG01-005	6/09/2023
REQ01-060	Selezionabilità e anteprima dell'icona	ESG01-006	6/09/2023
REQ01-070	Possibilità di applicare il cambiamento dell'icona a tutte le matrici dello stesso tipo	ESG01-007	6/09/2023
REQ01-080	Metodo di selezione delle icone personalizzate a seconda della funzione di utilizzo del dispositivo	ESG01-008	6/09/2023

Capitolo 5

Analisi delle tavole grafiche

5.1 Descrizione

Per la realizzazione delle schermate dell'app che è stata sviluppata mi sono state fornite dal reparto di grafica una serie di tavole grafiche. Queste tavole grafiche rappresentano nei minimi dettagli l'aspetto che doveva avere ogni singola schermata secondo le scelte del reparto incaricato dell'azienda, partendo dagli elementi che dovranno essere presenti, e arrivando ai colori, sia in modalità light che dark, al testo, ai font e tutte le varie spaziature. Inoltre, mi è stato fornito anche un documento riassuntivo con tutte le tavole che, oltre all'aspetto, descrive anche tutte le navigazioni che dovevano essere presenti tra gli elementi creati. Queste schermate sono state realizzate tramite programmi di editing grafico e dovevano essere quindi ricreate tramite codice all'interno dell'app, nel modo più simile possibile, aiutandosi con tutti i dati che forniscono come i dp per le spaziature, i codici esadecimali dei colori presenti o i font e dimensioni del testo che troviamo all'interno. Infine mi è stata inoltre fornita una tavola che contiene tutte le icone o immagini che sono state necessarie alla realizzazione del progetto.

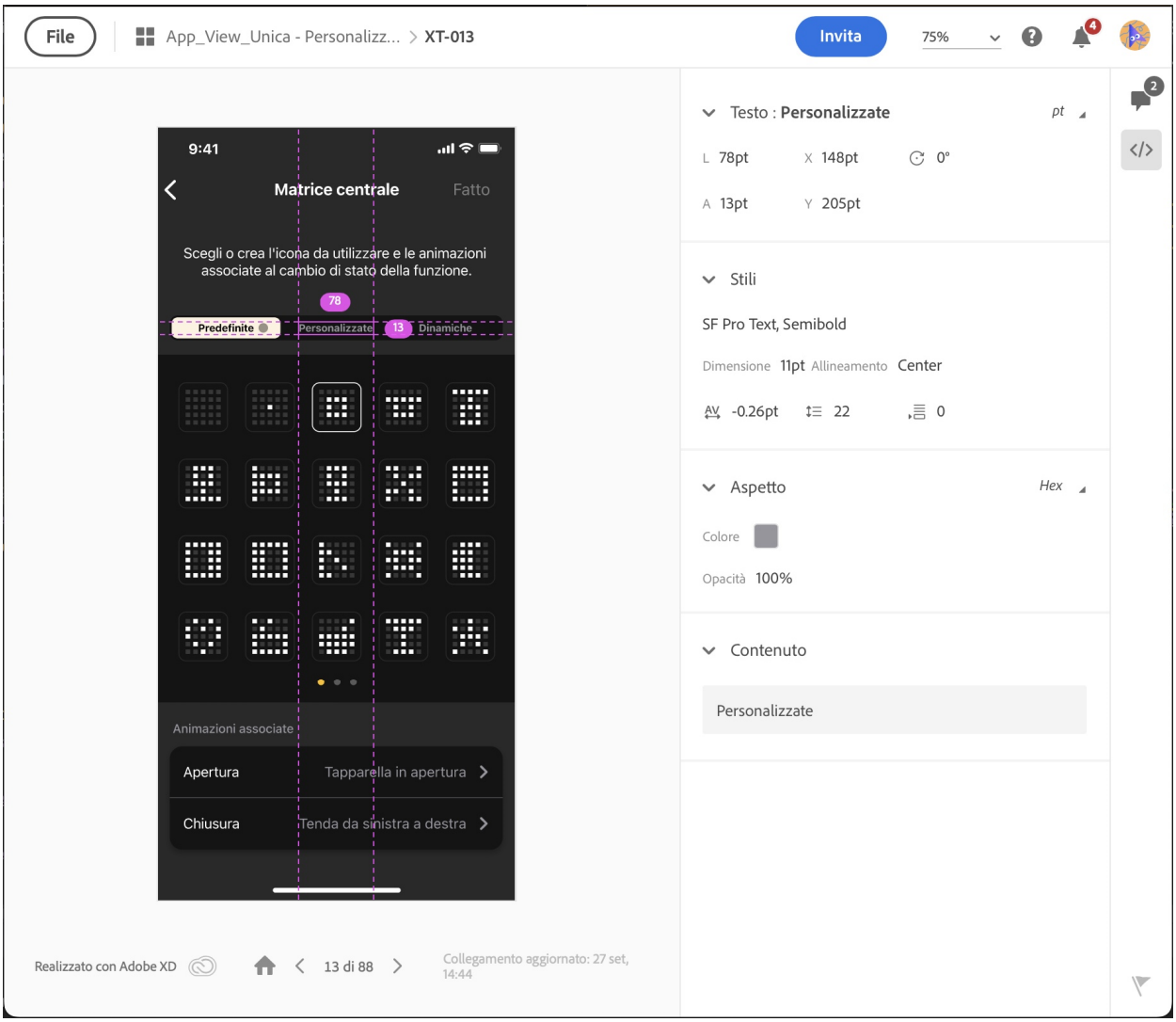

Figura 5.1: Tavola grafica XT-013

5.2 Documento delle tavole grafiche

Le seguenti immagini consistono negli elementi più rilevanti nel documento riassuntivo delle tavole grafiche che mi è stato fornito, il quale per ogni schermata fornisce anche un breve spiegazione della rappresentazione e delle reazioni alle interazioni con essa.

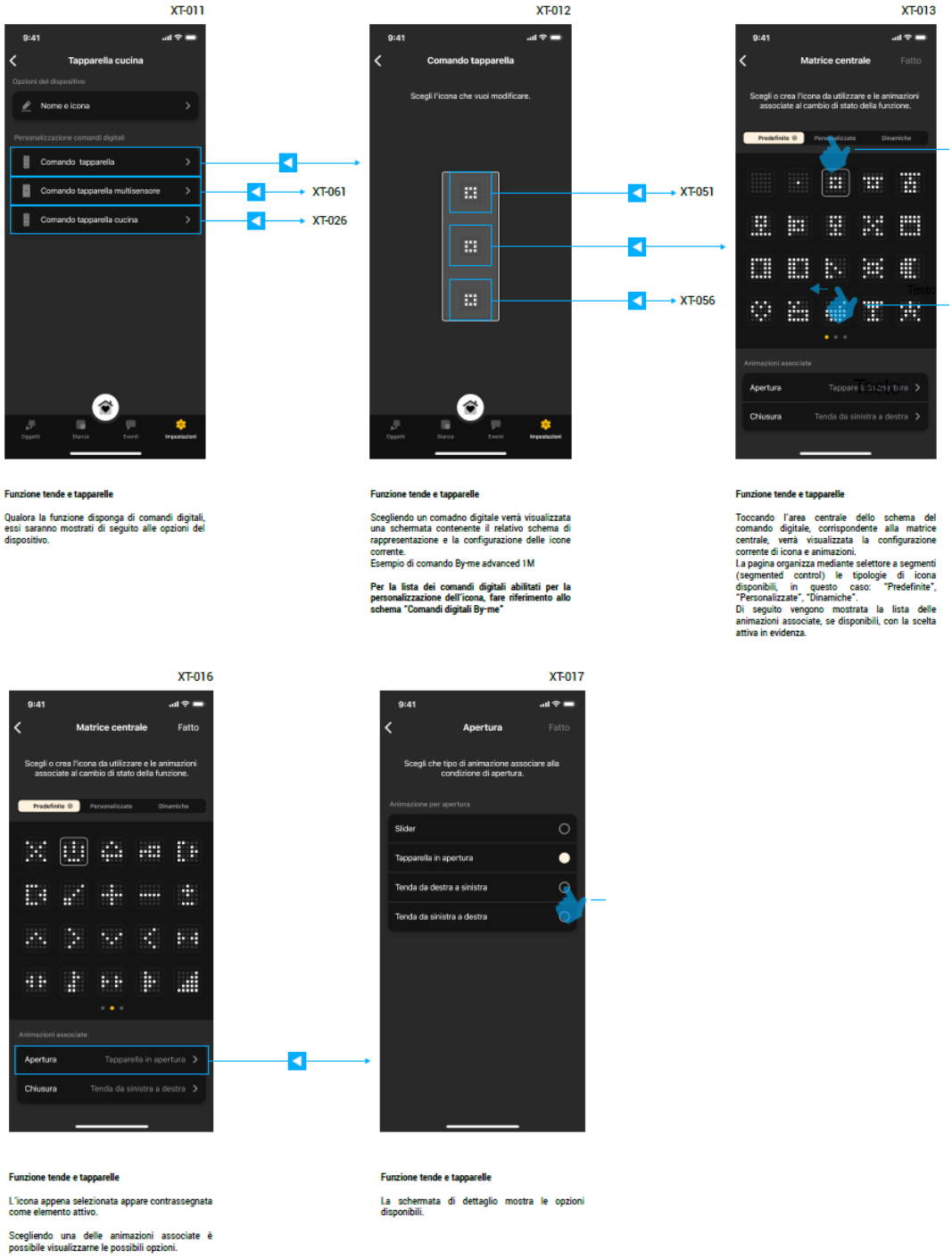

Figura 5.2: Parte del documento delle tavole grafiche

Capitolo 6

Schermate grafiche

6.1 Realizzazione

L'implementazione del progetto è iniziata con la realizzazione delle varie schermate grafiche che dovranno rispettare il più fedelmente possibile le tavole grafiche fornite, introdotte precedentemente.

Prima si è partiti con una fase di test preliminare delle varie idee per l'implementazione e per quali elementi grafici utilizzare: ciò è stato fatto con la creazione in un piccolo progetto separato in cui vengono testati e creati i vari elementi. Ad esempio sono stati fatti dei test grafici per la rappresentazione visiva delle varie matrici e della lista di matrici, dei quali se ne parlerà nelle sezioni 6.2 e 6.3, o per la realizzazione del seguente elemento presente nella schermata XT-013:

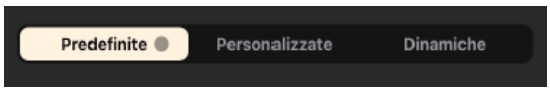

Figura 6.1: Bottoni presenti nella tavola grafica XT-013

Qui ad esempio il dubbio consisteva se usare un elemento grafico già esistente chiamato ChipGroup, contenente le varie Chip, che in questo caso non sono altro che una specie di bottoni cliccabili con scritto "Predefinite", "Personalizzate" e "Dinamiche ", oppure creare da sè l'elemento con degli elementi di base, attraverso la composizione di vari layout con delle TextView cliccabili, opportunamente gestite, che rappresentano i bottoni.

Alla fine, nonostante la maggiore facilità di utilizzo e implementazione delle Chip, si è preferito optare per la seconda scelta perché permette una maggiore possibilità di personalizzazione che rende il componente più simile alle tavole grafiche.

Successivamente alla fase di test si è passati alla realizzazione vera e propria dei vari Fragment che conterranno le schermate previste per la realizzazione del progetto. Questi vengono creati attraverso la codifica di file di tipo XML, uno per Fragment, associato ad una classe che permette il suo controllo e modifica.

I Fragment e quindi le schermate più importanti realizzate sono le seguenti.

6.1.1 Schermata XT-012

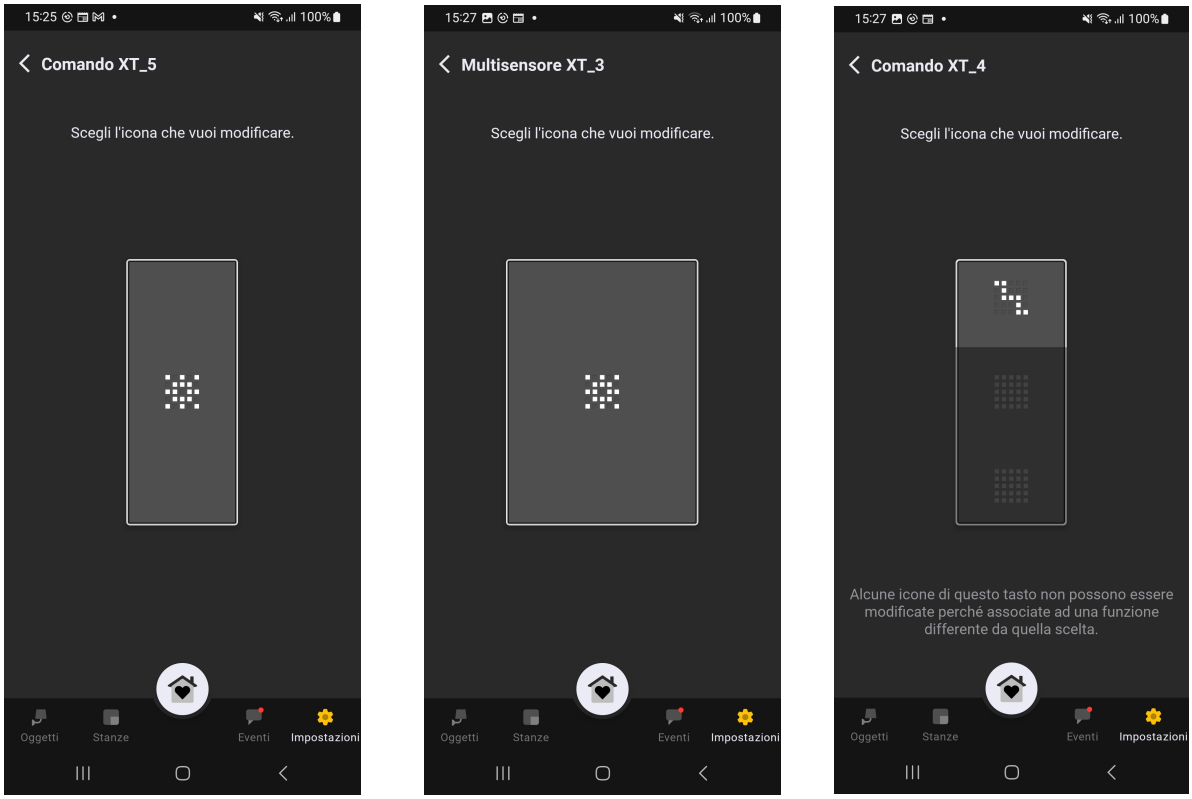

Figura 6.2: Schermata XT-012 configurata in base ai dispositivi e alle loro caratteristiche

Questa schermata permette una visione delle icone caricate nel dispositivo e del dispositivo stesso, una rappresentazione di ciò che l'utente può configurare e la possibilità di scelta della posizione dell'icona che vorrà modificare.

La schermata si basa su un ConstraintLayout, container che permette la disposizione di vari elementi grafici al suo interno, posizionati secondo dei vincoli posti dal programmatore. All'interno vengono quindi disposti i vari elementi che sono:

- Un' AppBar, cioè una barra superiore custom di Vimar, opportunamente configurata.
- Due TextView per il testo presente nella schermata. Una è visibile solo in caso di parti del comando non selezionabili.
- Un'ImageView, che rappresenta in modo stilizzato il dispositivo scelto per le modifiche.
- Le tre matrici, opportunamente visibili a seconda della presenza nel dispositivo scelto, rappresentate tramite l'elemento custom creato appositamente chiamato IconMatrix, il quale verrà descritto successivamente.

L'ImageView può visualizzare come risorsa una serie di sfondi creati attraverso l'utilizzo di shape in file di tipo Drawable, i quali rappresentano tutti i vari casi i cui può trovarsi il dispositivo. Questi casi consistono in Comando XT (1 o 3 matrici, sfondo stretto) o Multisensore XT (1 matrice, sfondo largo), ed inoltre rappresenta, annerendo rispettivamente le varie parti dello sfondo, se una matrice presente nel dispositivo è selezionabile o no.

6.1.2 Schermata XT-013

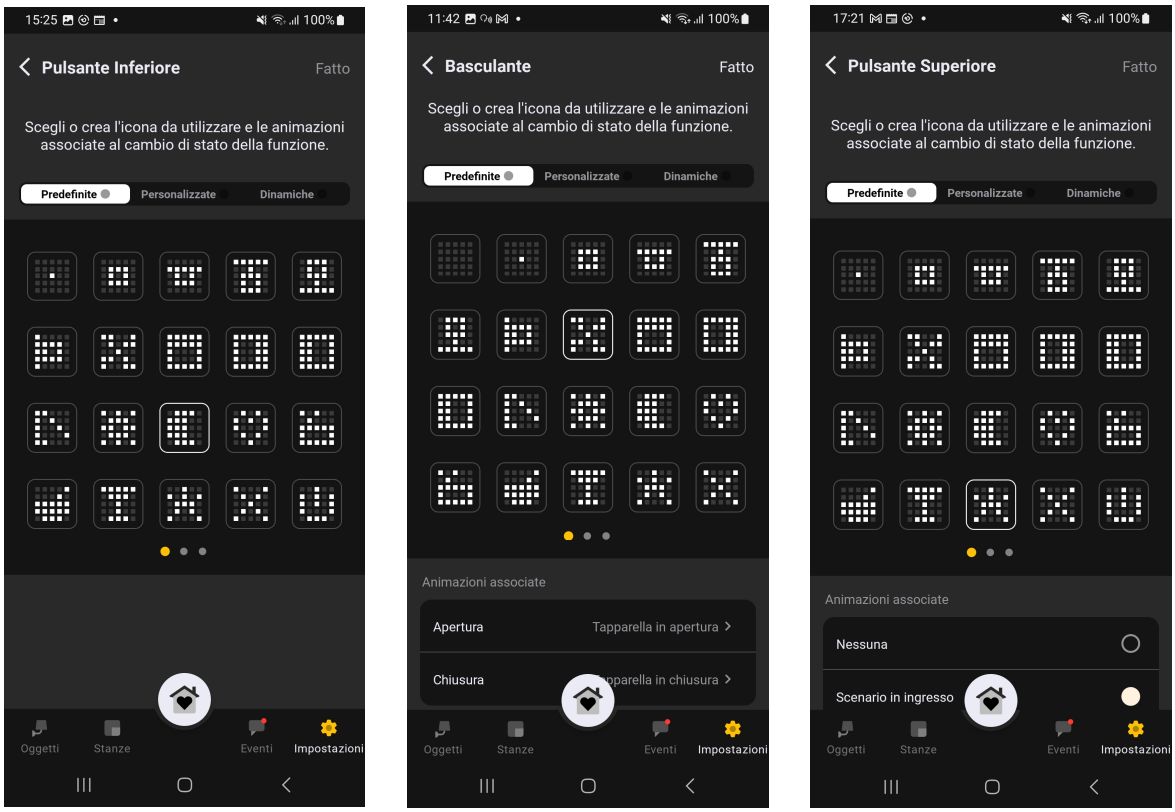

Figura 6.3: Schermata XT-013 configurata in base alle icone e animazioni disponibili per la matrice che si sta modificando

Questa schermata permette la scelta dell'icona nella lista di quelle predefinite, e, inoltre, ne permette il caricamento nel dispositivo alla pressione del pulsante "fatto". In questa pagina è inoltre presente una predisposizione per la futura implementazione, non interessata dal mio progetto, che permette anche la scelta di icone dinamiche e personalizzate dall'utente. Le pagine di scelte delle icone predefinite, dinamiche e personalizzate vengono opportunamente rese visibili con i rispettivi bottoni a seconda della disponibilità o no per quel dispositivo in quel determinato contesto.

La schermata si basa su uno ScrollView, container che, grazie alla capacità di scroll, permette la visione completa della pagina su qualsiasi dispositivo indipendentemente dalla lunghezza dello schermo e dagli elementi contenuti

verticalmente nella pagina. Nello ScrollView è stato quindi inserito un ConstraintLayout per la disposizione dei vari elementi grafici. All'interno vengono disposti i vari elementi, che sono :

- Un' AppBar, cioè una barra superiore custom di Vimar, opportunamente configurata.
- La composizione di TextView e Layout per la creazione dei tre tasti "Predefinite", "Personalizzate", "Dinamiche", precedentemente introdotta.
- Le TextView per i testi.
- Una RecyclerView, con un layout degli elementi caricati opportunamente configurato, per l'eventuale gestione della lista delle animazioni associate all'icona.
- Un FragmentContainer.

Il FragmentContainer è stato inserito nell'ottica di predisposizione anche alle funzionalità, non attinenti al mio progetto, ma che verranno in seguito implementate, di selezione di icone dinamiche e personalizzate. Questo perchè al suo interno, oltre al Fragment contenente la lista di icone predefinite, verranno caricati anche quelli per le altre tipologie di icone al premere dei rispettivi tasti.

Per quanto riguarda la lista di animazioni da associare all'icona, se disponibili per la matrice selezionata per le modifiche, potrà avere due aspetti diversi a seconda se sono animazioni associate un dispositivo di controllo di scenario, oppure di apertura e chiusura. Infatti in caso fossero animazioni di apertura e chiusura queste verrebbero visualizzate con il nome della categoria a cui appartengono (Apertura o Chiusura) e il nome dell'animazione associata a quella categoria, permettendo inoltre tramite click di accedere alla schermata di modifica dell'animazione associata. Invece, se fossero animazioni di scenario, queste verrebbero visualizzate come una lista di animazioni associate allo scenario direttamente selezionabili tramite click.

Infine per questa schermata è stata predisposta anche una serie di Dialog per informare l'utente in caso di uscita dalla schermata senza salvataggio delle modifiche, possibilità di eseguire la bulk action e infine la corretta applicazione delle modifiche.

6.1.3 Fragment lista di icone predefinite

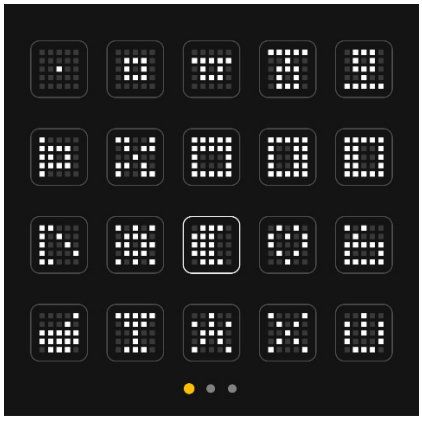

Figura 6.4: Fragment lista di icone predefinite

Questa schermata permette la selezione di una determinata icona dalla lista di icone predefinite disponibili per quel determinato dispositivo.

La schermata si basa su un ConstraintLayout nella quale è inserito l'elemento grafico custom, creato appositamente per il progetto, IconGridList, il quale verrà esposto nella sezione 6.3.

6.1.4 Schermata XT-017

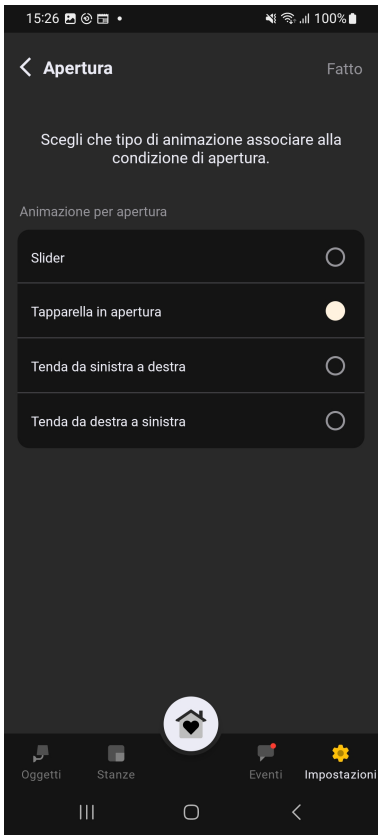

Figura 6.5: Schermata XT-017

Questa schermata permette la selezione dell'animazione da associare all'azione di apertura o chiusura, se disponibili per la matrice che si sta modificando. L'animazione viene scelta da una lista di animazioni predefinite per quella determinata matrice in quel contesto. La schermata si basa su un ConstraintLayout, nel quale sono stati inseriti i seguenti elementi:

- Un' AppBar, cioè una barra superiore custom di Vimar.
- Una TextView per la descrizione.
- Una RecyclerView con un layout degli elementi caricati opportunamente configurato. Permettendo di avere una lista nella quale si può selezionare un unico elemento tra i presenti, che verrà evidenziato dal pallino a sinistra del nome.

6.2 IconMatrix

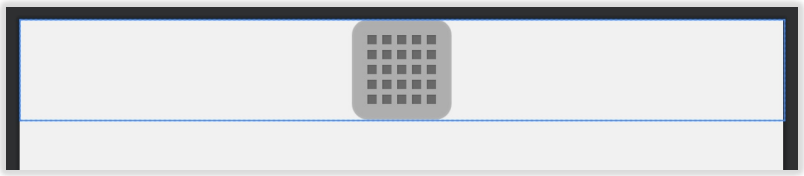

Figura 6.6: Layout dell'elemento custom IconMatrix

La rappresentazione delle matrici Led all'interno dell'app è stata realizzata attraverso un elemento grafico custom, creato appositamente, chiamato IconMatrix. Questo elemento si basa su una classe chiamata IconMatrix e il suo layout, il quale è composto da una un GridLayout 5X5, che rappresenta la matrice LED, e all'interno 25 ImageView le quali rappresentano, rispettivamente alla posizione, il singolo LED. All'interno di queste ImageView vengono caricate, a seconda se il rispettivo led è acceso o spento, un'immagine bianca o grigia. IconMatrix inoltre presenta due parametri appositamente creati, uno per l'inserimento o no dello sfondo dietro ai vari riquadrini, e l'altro per l'inserimento dello sfondo in caso di selezione dell'icona. Questi sfondi vengono caricati nel background del GridLayout e sono stati creati attraverso l'utilizzo di shape in file xml di tipo Drawable.

Per la realizzazione di questo elemento in fase di test si era pensato di ricorrere ad una RecyclerView con GridLayout, in modo da non aver problemi per un eventuale modifica futura del numero dei led nelle matrici dei dispositivi. Questa idea, però, è stata successivamente scartata in quanto le prestazioni, in fatto di caricamento e lag, erano molto peggiori rispetto al semplice GridLayout descritto precedentemente.

6.3 IconGridList

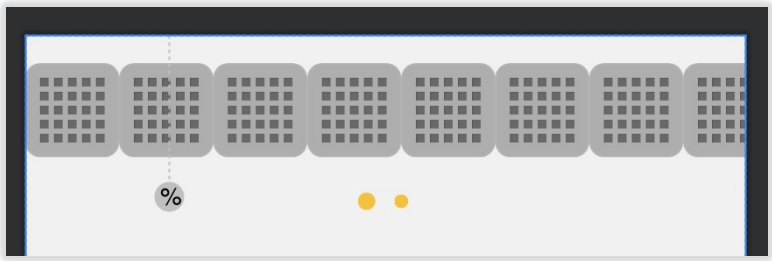

Figura 6.7: Layout dell'elemento custom IconGridList

La rappresentazione della lista di icone predefinite è stata creata attraverso la creazione di un elemento grafico custom, creato appositamente, chiamato IconGridList. Questo elemento si basa su una classe chiamata IconGridList e il suo layout. Quest'ultimo è composto da una RecyclerView orizzontale con GridLayout 5X4, nella quale vengono caricati i vari elementi di tipo IconMatrix e cioè una lista di matrici selezionabili. Infatti per il popolamento di questo elemento viene passata alla RecyclerView una lista di elementi di tipo Icon, opportunamente ricavati da IpConnector, i quali vengono rielaborati per popolare le varie IconMatrix. Inoltre la RecyclerView viene accoppiata ad uno SnapHelper, strumento che permette di scorrere un certo numero di elementi alla volta creando l'effetto di cambio pagina, e da un elemento custom Vimar preesistente, che consiste nei pallini che indicano il numero di pagine.

6.4 Traduzioni

L'applicazione secondo specifiche dove supportare 15 lingue: italiano, inglese, francese, tedesco, spagnolo, greco, ebraico, polacco, portoghese, russo, turco, cinese, arabo, olandese e svedese. Per permettere il caricamento automatico della corretta lingua delle stringhe presenti nelle schermate, a seconda del settaggio del dispositivo, viene utilizzato il sistema di cartelle con qualificatori. Questo sistema consiste nel creare la cartella di default, che contiene la lingua di default dell'applicazione, nel nostro caso inglese, e altre cartelle con lo stesso nome seguite da un qualificatore, nomeCartella – CodiceLingua (Es. per l'italiano: value-it). Tutte queste cartelle contengono il file strings.xml, uguale per tutte con gli stessi identici tag delle stringhe all'interno, con l'unica differenza nel contenuto della stringa tradotto nella lingua specificata dal qualificatore. Il sistema Android è in grado di rilevare la lingua usata nel dispositivo, caricando automaticamente le stringhe delle varie interfacce dal file strings.xml rispettivo.

Secondo procedura aziendale quindi, per la gestione delle traduzioni nelle lingue supportate, prima si inseriscono in un determinato documento tutte le stringhe necessarie per le schermate da creare in italiano, che poi vengono inviata al personale dedicato alle traduzioni. Successivamente viene ricevuto, di ritorno dall'ufficio incaricato, un documento contenente tutte le stringhe tradotte nelle varie lingue, le quali dovranno essere poi inserite nei rispettivi file strings.xml della cartella con il corretto qualificatore.

6.5 Gestione temi

L'applicazione secondo specifiche supporta due temi, uno chiaro e uno scuro. Per fare ciò vengono sfruttati i colori i quali vengono caricati nei vari elementi grafici, testi e file di tipo drawable da un file color.xml contenete tutti i loro codici esadecimali. Infatti questi utilizzano un metodo come quello precedentemente descritto per la lingua, avendo a disposizione le cartelle Color-light e Color-dark, contenenti entrambe il file color.xml, con all'interno i tag dei colori utilizzati collegati al rispettivo codice per il tema chiaro o scuro.

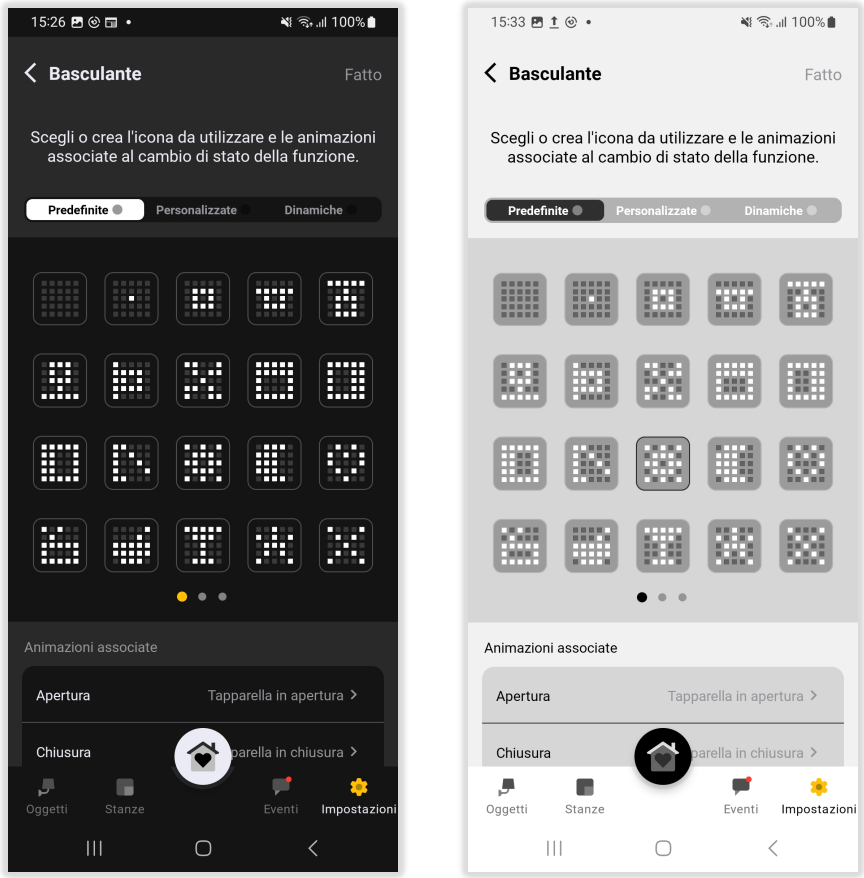

Figura 6.8: Confronto tra tema dark e light

Capitolo 7

Logica di funzionamento

7.1 Navigazione

Successivamente alla realizzazione della grafica si è passati alla codifica delle logiche di funzionamento iniziando con la navigazione tra le varie schermate. La navigazione tra i Fragment all'interno dell'app si basa sull'utilizzo del Navigation Component, utilizzando i file XML dei grafi di navigazione contenuti nella cartella navigation. All'interno dell'app, per evitare di avere un singolo grafo di grandi dimensioni, sono presenti più grafi che organizzano i vari Fragment a seconda di aree di interesse o funzionalità. A loro volta quindi questi vari NavGraph presenti si richiamano tra loro per permettere il flusso di navigazione completo. Pertanto per l'implementazione della funzionalità interessata è stato necessario implementare un nuovo grafo, il quale contiene tutti i vari Fragment creati e le varie rotte di navigazione con il loro rispettivi parametri, i quali vengono inviati al Fragment successivo nello spostamento.

7.2 Modifiche a IpConnector

Per il controllo di questi nuovi dispositivi sono state fatte delle modifiche nel protocollo IpConnector, perciò per la sua gestione è stato necessario anche inserire delle nuove funzionalità alla libreria che lo implementa. A questo scopo quindi sono state aggiunte una System Function, che rappresenta i dispositivi con matrice configurabile, 4 System Function Elements, per la gestione dei nuovi stati dei dispositivi, e le relative Data Class per l'organizzazione dei nuovi dati trasmessi.

7.3 Codifica delle matrici a Led

Nella trasmissione dei dati tramite IpConnector, e quindi in tutto il progetto, le varie icone, cioè l'informazione dei vari LED accessi o spenti delle matrici 5X5, sono codificate in un bitmap. Con bitmap si intende una stringa di 10 caratteri esadecimali, i quali coverti in binario creano una sequenza composta da 5 byte che codificano le informazioni di rappresentazione di una matrice LED 5X5 così composta:

```
  a b c d e
1 * * * * *
2 * * * * *
3 * * * * *
4 * * * * *
5 * * * * *
```

Ogni byte del bitmap rappresenta una riga della matrice e ogni bit un LED (1 = acceso, 0 = spento), il dato complessivo è quindi così organizzato:

```
Byte  |_________ 1 ________|_________ 2 _________|_________ 3 _________|

Bit   | 7| 6| 5| 4| 3| 2| 1| 0| 7| 6| 5| 4| 3| 2| 1| 0| 7| 6| 5| 4| 3| 2| 1| 0|

      | 0| 0| 0|e1|d1|c1|b1|a1| 0| 0| 0|e2|d2|c2|b2|a2| 0| 0| 0|e3|d3|c3|b3|a3|

Byte  |_________ 4 _________|_________ 5 _________|

Bit   | 7| 6| 5| 4| 3| 2| 1| 0| 7| 6| 5| 4| 3| 2| 1| 0|

      | 0| 0| 0|e4|d4|c4|b4|a4| 0| 0| 0|e5|d5|c5|b5|a5|
```

Pertanto è stato necessario creare un algoritmo che converta la sequenza di caratteri esadecimali, ricevuti per ogni icona, in una lista di elementi di tipo Boolean, la quale viene fornita all'elemento IconMatrix per la visualizzazione dell'icona all'interno dell'app. Successivamente quindi per IconMatrix ogni elemento booleano della lista passata corrisponde ad un LED della matrice che andrà a visualizzare, perciò, scorrendo gli elementi del GridLayout, partendo da in alto a sinistra, colorerà di bianco l'ImageView se il valore

corrispondente è true , mentre caricherà il grigio in caso di valore false (Grigio = LED acceso, Bianco = LED spento). Inoltre è stato creato anche l'algoritmo contrario il quale dalla lista di Boolean, ricavata da IconMatrix, genera il rispettivo bitmap dell'icona

7.4 Interazione con la domotica

Per la realizzazione delle logiche di funzionamento, per il recupero, gestione e invio dei dati con i dispositivi domotici, si è strutturato il codice in modo da rispettare il pattern dell'app MVVM con Clean Architecture. Pertanto sono stati creati, oltre alle classi necessarie per la gestione dei vari Fragment, un ViewModel e una serie di classi UseCase per il recupero e invio delle informazioni necessarie da IpConnector, il quale comunica con i dispositivi domotici collegati all'impianto. Con classe UseCase si intende una classe che svolge una singola operazione utile per il funzionamento dell'app, la quale può quindi essere richiamata in più punti del codice. Per quanto riguarda il ViewModel questo viene condiviso tra tutti gli elementi del grafo di navigazione a cui è associato, in modo da mantenere a disposizione per tutte le schermate, fino alla conferma, i dati del dispositivo e delle modifiche effettuate all'icona. Questo perché nel ViewModel, oltre alle funzioni che richiamano le varie classi UseCase, sono stati inseriti anche una serie di variabili e LiveData per contenere sia tutti i dati necessari per le schermate di visualizzazione, sia quelli per tenere traccia dei cambiamenti apportati ai settaggi del dispositivo, fatti nelle pagine di scelta dell'icona e animazioni.

Entrando nel dettaglio delle UseCase create, queste vengono utilizzate, passando attraverso il ViewModel dalle varie classi dei Fragment, per il caricamento dei dati necessari alla visualizzazione o per l'invio di informazioni a IpConnector, inoltre ognuno di essi rappresenta una singola azione da svolgere e ne contiene il codice per l'implementazione.

In particolare per il funzionamento dell'app :

- La schermata, identificata sulle tavole come XT-012, che permette la scelta della posizione dell'icona da modificare nel dispositivo, invoca il metodo del ViewModel che richiama la UseCase, la quale, partendo dall'oggetto che identifica il dispositivo nell'impianto, permette di ottenere gli oggetti che rappresentano i dati delle massimo tre matrici cliccabili nel dispositivo utilizzato per quella determinata funzione. L'identificativo del dispositivo è fornito come parametro di navigazione direttamente dalla schermata precedente di scelta del dispositivo, preesistente nell'app. Successivamente quindi questa UseCase carica i dati in un LiveData sul ViewModel e quindi la schermata resta in ascolto del dato. Una volta che riceve il dato quindi carica gli elementi interessati nella schermata, con una opportuna configurazione sia a seconda del dispositivo, sia se nel dispositivo sono personalizzabili o no tutte le matrici presenti. Infine quando si clicca su una delle matrici per effettuare la modifica, viene caricato nel ViewModel l'oggetto specifico che rappresenta quella determinata sezione personalizzabile del comando.

- La schermata, identificata sulle tavole come, XT-013, che contiene la pagina di scelta delle icone, animazioni e conferma, invoca invece tramite il ViewModel due classi UseCase. Una viene invocata per l'invio a IpConnector delle modifiche, e quindi il salvataggio delle modifiche all'interno del dispositivo, dopo la conferma tramite il pulsante fatto. Qui il metodo della UseCase riceve come parametro un'oggetto di una DataClass delle modifiche fatte, opportunamente creata e popolata secondo le regole del protocollo, la quale viene quindi inviata a IpConnector tramite un metodo apposito della libreria. L'altra UseCase invece viene invocata per l'ottenimento della lista dei dispositivi simili a quello che si sta modificando in quella determinata funzione. Quest'ultimi possono essere interessati dall'applicazione della modifica in "bulk action", cioè l'applicazione di un'icona e/o animazione a tutte le matrici dello stesso tipo presenti nell'impianto, a patto che a loro volta abbiano la disponibilità

dell'icona e/o animazioni selezionate nella loro lista di icone e animazioni possibili. Questa classe UseCase prende in input l'oggetto che identifica la matrice scelta per la modifica, tra quelle disponibili nel dispositivo, l'icona e le animazioni scelte, fa le opportune verifiche tra tutti gli elementi collegati, e ne ritorna la lista in un dato del ViewModel. Il dato salvato nel ViewModel viene quindi osservato dal fragment che, in base alla disponibilità di elementi, fornisce all'utente anche la possibilità di applicazione della modifica in "bulk action", oltre alla singola matrice scelta. Infine, per quanto riguarda questa schermata, la lista delle animazioni associate al dispositivo, mostrate nella RecyclerView, vengono ricavate dagli attributi dell'oggetto che identifica la matrice, caricato nel ViewModel nella schermata precedente in fase di scelta dell'icona da modificare.

- Per quanto riguarda la schermata del Fragment di scelta dell'icona dalla lista delle predefinite, qui si resta in ascolto del dato del ViewModel che indica la matrice selezionata per la modifica. Infatti dagli attributi di questo si ricava la lista delle icone predefinite disponibili per quell'elemento, la quale viene passata all'elemento IconGridList per il popolamento.

- Nella schermata di dettaglio di scelta delle animazioni associate alle funzioni di apertura e chiusura, XT-017, per il popolamento degli elementi selezionabili della lista viene osservato un dato contenuto nel ViewModel. Questo dato rappresenta la lista di animazioni associate a quella funzione, il quale viene salvato, in fase di selezione della funzione da modificare, nella schermata precedente. La schermata precedente a sua volta lo ricava dall'elemento che identifica la matrice scelta per la modifica presente all'interno del ViewModel. Inoltre in questa schermata viene anche implementato un meccanismo di selezione univoca dell'animazione dalla lista cliccabile di animazioni, la quale viene confermata e caricata nella funzione al premere del pulsante "fatto".

Capitolo 8

Test

8.1 Test di unità

Durante il progetto sono stati implementati una serie test automatici di unità usando la libreria JUnit. Questi test permettono di verificare il corretto funzionamento di una singola porzione di codice o di una singola componente e sono a scatola nera, cioè viene inviato un input e si verifica che l'output della funzione sia quello corretto. Questi sono stati implementati per avere sempre un controllo sul corretto funzionamento degli algoritmi in caso di eventuali modifiche al codice. Nel nostro caso hanno interessato soprattutto gli algoritmi di trasformazione dei dati per IconMatrix precedentemente descritti, cioè la funzione che trasforma la stringa esadecimale del bitmap dell'icona nella corretta lista di true e false da mandare alla matrice, e il suo algoritmo contrario. Questi infatti sono stati entrambi testati rispettivamente con vari bitmap e liste di boolean, comprendendo vari casi limite come una matrice tutta vuota, una tutta piena o valori non accettabili.

8.2 Test manuali sul dispositivo

Durante il tirocinio, per verificare manualmente se ciò che stavo facendo funzionasse e apparisse a schermo correttamente, l'app veniva eseguita nel dispositivo Samsung Galaxy A33 che mi è stato fornito dall'azienda, in quanto l'app è troppo complessa e pesante per essere eseguita in un emulatore fornito da Android Studio. Mi è stato anche affidato un impianto domotico di test con installati i vari dispositivi della Linea XT interessati, in varie configurazioni, in modo da poter testare tutte le varie funzionalità e condizioni.

Infine, per verificare che l'app funzionasse correttamente, sia dal punto di vista grafico che di funzionalità, sono stati fatti dei test manuali su più dispositivi Android diversi per marca, dimensioni e versione Android.

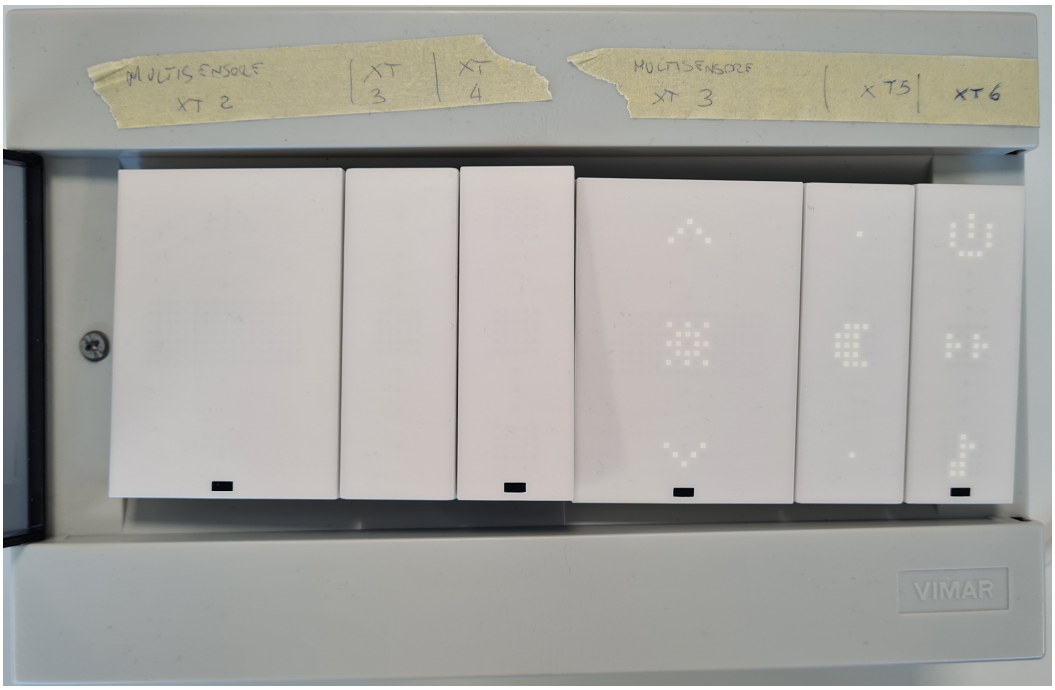

Figura 8.1: Quadro di prova fornito con dispositivi della Linea XT

Capitolo 9

Conclusioni

9.1 Conclusioni sul progetto

Al termine del periodo di tirocinio si può affermare che la nuova funzionalità dell'app VIEW, interessata dal progetto, è stata completamente implementata, soddisfacendo tutti i requisiti che erano stati definiti dall'azienda e arrivando allo scopo finale di essere rilasciata in produzione.

9.2 Conclusioni dal punto di vista personale

Dal punto di vista personale posso dire che l'esperienza in VIMAR è stata molto utile e costruttiva in quanto mi ha permesso, grazie all'affiancamento con un team di sviluppatori esperti, di conoscere e apprendere nuovi concetti, strumenti e metodologie di lavoro adottate nell'ambito dello sviluppo Android e della programmazione professionale. Il tirocinio mi ha permesso inoltre di conoscere e sperimentare come funziona il mondo lavorativo, con la possibilità di mettere in pratica e approfondire i vari concetti appresi durante il mio percorso universitario.

APPENDICE

A. Esigenze

Le seguenti esigenze contengono il testo della documentazione sulle esigenze, fornitami dall'azienda durante il tirocinio, riportato fedelmente.

A.1 ESG01-001:

Si richiede che i comandi digitali della Linea XT By-me siano personalizzabili con una matrice di LED dinamica da APP VIEW, selezionando le possibili icone da una libreria predefinita.

A.2 ESG01-002:

Comando XT By-me 1M (32021.x).
Si richiede per questo modello di comando la personalizzazione della matrice LED centrale disponibile.

A.3 ESG01-003:

Comando XT By-me advanced 1M (32023.x).
Si richiede per questo modello di comando la personalizzazione della matrice LED centrale e dei tasti superiore e inferiore.

A.4 ESG01-004:

Comando XT By-me multisensore 2M (32042.x). Si richiede per questo modello di comando la sola personalizzazione della matrice centrale disponibile.

A.5 ESG01-005:

La matrice LED deve essere personalizzabile tramite un set di icone predefinite, personalizzabili a mano dall'utente amministratore o da un set di icone dinamiche. Quest'ultime mostrano lo stato effettivo della funzione del dispositivo.

A.6 ESG01-006:

Le icone devono essere mostrare in una lista di anteprime e ogni anteprima può essere selezionata.

A.7 ESG01-007:

Un'icona potrà essere applicabile (dando la possibilità di scelta all'utente) tramite "bulk action" a tutte le matrici dello stesso tipo. Oppure altrimenti solo alla matrice scelta.

A.8 ESG01-008:

Si richiede la possibilità di personalizzare le matrici led per funzione, dalle singole impostazioni di una funzione (ad esempio di una luce o una tapparella) deve esserci la possibilità di accedere al menù di personalizzazione. Di seguito una tabella che contiene le personalizzazioni possibili per funzione:

Funzione	Elemento	Icone predefinite	Icone personalizzate	Icone dinamiche	Slider	Animazione 1	Animazione 2
Tende e tapparelle Disponibile per tutto tranne: • Widget personalizzato	Tasto superiore (solo 1M advanced)	✓	✓				
	Matrice centrale	✓	✓	✓ (solo se con posizione)	✓ (solo se con posizione)	Apertura (XT-017)	Chiusura (XT-020)
	Tasto inferiore (solo 1M advanced)	✓	✓				
Luci Disponibile per tutto tranne: • Widget personalizzato	Tasto superiore (solo 1M advanced)	✓	✓				
	Matrice centrale	✓	✓	✓ (solo se dimmerabili)	✓ (solo se dimmerabili)		
	Tasto inferiore (solo 1M advanced)	✓	✓				
Scenario Disponibile come pulsante attivazione scenario	Tasto superiore (solo 1M advanced)	✓	✓			XT-073	
	Matrice centrale	✓	✓				
	Tasto inferiore (solo 1M advanced)	✓	✓			XT-073 (come superiore)	
Accessi e presenze Disponibile per tutto tranne: • Contatto • Widget personalizzato	Tasto superiore (solo 1M advanced)	✓	✓				
	Matrice centrale	✓	✓				
	Tasto inferiore (solo 1M advanced)	✓	✓				
Audio Disponibile per tutto tranne: • Sorgente audio • Widget personalizzato	Tasto superiore (solo 1M advanced)	✓	✓				
	Matrice centrale	✓	✓	✓ (solo per zona audio)	✓ (solo per zona audio)		
	Tasto inferiore (solo 1M advanced)	✓	✓				
Clima Disponibile per tutto tranne: • Sensori solo visualizzazione • Integrazione KNX • Widget personalizzato	Tasto superiore (solo 1M advanced)	✓	✓				
	Matrice centrale	✓	✓				
	Tasto inferiore (solo 1M advanced)	✓	✓				
Sensori Disponibile per tutto tranne: • Sensori solo visualizzazione • Widget personalizzato	Tasto superiore (solo 1M advanced)	✓	✓				
	Matrice centrale	✓	✓				
	Tasto inferiore (solo 1M advanced)	✓	✓				
Energia Disponibile solo nei carichi, per forzatura	Tasto superiore (solo 1M advanced)	✓	✓				
	Matrice centrale	✓	✓				
	Tasto inferiore (solo 1M advanced)	✓	✓				
Varie Disponibile per tutto tranne: • Unità logica • Temporizzzioni elementari • Allarme tecnico personalizzato • Widget personalizzato	Tasto superiore (solo 1M advanced)	✓	✓				
	Matrice centrale	✓	✓	✓ (solo se regolatore)	✓ (solo se regolatore)		
	Tasto inferiore (solo 1M advanced)	✓	✓				

Figura A.1: Tabella personalizzazioni possibili per funzione

B. Requisiti

I seguenti requisiti contengono il testo della documentazione sui requisiti, prodotta durante il tirocinio, riportato fedelmente.

B.1 REQ01-10 riferito all'esigenza ESG01-001:

L'utente deve avere a disposizione all'interno dell'app VIEW, collegata rispettivamente ai dispositivi che supportano questa caratteristica, una interfaccia grafica specifica di personalizzazione. Con dispositivi che supportano questa caratteristica si indicano i dispositivi digitali della Linea XT by me:

- Comando XT By-me 1M (cod. art. 32021) [REQ01-020]
- Comando XT By-me advanced 1M (cod. art. 32023) [REQ01-030]
- Multisensore XT By-me 2M (cod. art. 32042) [REQ01-0400]

Usati nelle funzioni di:

- Controllo luci
- Controllo tende e tapparelle
- Controllo scenario
- Controllo sistemi di accesso e presenze
- Controllo di sistemi audio
- Controllo del clima
- Controllo di sensori
- Controllo di sistemi di energia
- Controllo di sistemi della categoria varie

L'utente per personalizzare la matrice led di un comando, tra quelli precedentemente descritti, di un dispositivo: deve partire innanzitutto dalla schermata delle impostazioni, premere sulla voce stanze e dispositivi, selezionare la stanza in cui è presente il dispositivo;

da lì verrà visualizzata la lista dei dispositivi presenti e da lì potrà selezionare il dispositivo per cui vuole personalizzare il comando. Una volta selezionato il dispositivo verrà visualizzata una schermata in cui sarà possibile scegliere la posizione della matrice che si vuole personalizzare. Infine, premendo su uno dei massimo tre slot, verrà visualizzata la schermata di scelta dell'icona tra la lista di quelle predefinite, personalizzate ed eventualmente dinamiche.

Un'ulteriore funzione consiste nel fatto che l'utente può accedere alla schermata di personalizzazione partendo dalla schermata oggetti in due modi. Il primo modo consiste nella pressione prolungata dell'icona dell'elemento scelto, da lì poi si apre un dialog che, premendo su impostazioni, porterà alla schermata di opzioni del dispositivo dove si potrà selezionare l'elemento che si vuole personalizzare, andando alla schermata di selezione della posizione delle icone. Il secondo metodo consiste nel cliccare sull'elemento di cui si vuole selezionare il comando; lì si verrà mandati ad una finestra di dettaglio, dove, premendo i tre puntini in alto a sinistra e successivamente su impostazioni, nel menù che si apre, si verrà indirizzati alla schermata, precedentemente descritta, di opzioni del dispositivo e da lì si può procedere come sopra indicato.

B.2 REQ01-020 riferito all'esigenza ESG01-002:

All'interno dell'app VIEW verrà collegata la personalizzazione di questo modello di componente con una schermata, raffigurante in modo stilizzato il dispositivo, la quale permetterà di scegliere attraverso un riquadro cliccabile solo la matrice centrale, unica disponibile nel dispositivo. Quindi l'utente può premere solamente al centro del dispositivo stilizzato, andando alla pagina di selezione dell'icona. Con questo dispositivo, se associato all'utilizzo con tende e tapparelle con posizione, luci dimmerabili o audio zona, l'utente avrà la possibilità di assegnagli, dalla schermata di scelta delle icone dinamiche, l'icona slider, la quale, oltre ad utilizzare la matrice LED centrale, utilizza anche i due LED sopra e sotto la matrice per rappresentare una striscia bianca.

B.3 REQ01-030 riferito all'esigenza ESG01-003:

All'interno dell'app VIEW verrà collegata la personalizzazione di questo modello di componente con una schermata, raffigurante in modo stilizzato il dispositivo, la quale fornisce, attraverso dei riquadri cliccabili, la possibilità di selezionare e quindi di personalizzare tutte e tre le matrici presenti. Quindi l'utente può premere nella parte alta, centro e in basso del dispositivo stilizzato, andando alla pagina di selezione dell'icona per quel determinato riquadro.

In questi dispostivi, aventi più matrici, se non è stata ancora associata un'immagine a una delle tre posizioni questa verrà raffigurata con l'icona di default con tutti i LED spenti.

Infine, in questi dispositivi aventi più matrici, c'è la possibilità che, se la funzione selezionata è associata solo parzialmente al dispositivo, lo schema stilizzato di rappresentazione mostrerà le aree, associate ad altre funzioni, come non disponibili per la selezione, annerendo rispetto all'area selezionabile la loro posizione e facendo comparire una scritta che ne descrive il motivo.

B.4 REQ01-040 riferito all'esigenza ESG01-004:

All'interno dell'app VIEW verrà collegata la personalizzazione di questo modello di componente con una schermata, raffigurante in modo stilizzato il dispositivo, la quale permetterà di scegliere attraverso un riquadro cliccabile solo la matrice centrale, unica disponibile nel dispositivo. Quindi l'utente può premere solamente al centro del dispositivo stilizzato, andando alla pagina di selezione dell'icona.

Il dispositivo multisensore è solo parzialmente oggetto della funzionalità (solo quando viene utilizzato come "comando", non quando è usato come termostato): infatti l'utente potrà personalizzare mediante l'applicazione VIEW la sola matrice centrale, mentre resterà compito dell'installatore la personalizzazione delle matrici superiore e inferiore.

La matrice centrale ha a disposizione tutti i 25 led previsti, quindi fornisce all'utente la possibilità di fare delle personalizzazioni complete (ed eventualmente un disegno pixel-by-pixel). Le matrici superiore e inferiore hanno invece un set minimale di LED (solo quelli necessari a disegnare i simboli relativi alle funzioni specifiche per cui tali dispositivi sono progettati), quindi l'utente avrebbe solo una lista limitatissima di opzioni. Per tale motivo, vista la ridotta scelta di personalizzazioni disponibili per l'utente, le matrici superiore e inferiore vengono escluse dalla funzione.

B.5 REQ01-050 riferito all'esigenza ESG01-005:

L'utente, dopo aver premuto sul punto desiderato nella schermata di scelta della posizione dell'icona da inserire, accede a una nuova schermata dove sarà presente inizialmente la lista, suddivisa in pagine, delle icone predefinite tra cui può scegliere. In questa schermata saranno inoltre presenti in alto tre tasti, denominati rispettivamente "Predefinite", "Personalizzate" e "Dinamiche". Questi tasti, se premuti, portano alle rispettive pagine di scelta delle icone predefinite, personalizzate e dinamiche.

Se l'utente dovesse premere sul bottone "Personalizzate" verrà inviato alla pagina di selezione delle icone personalizzate, sempre se non sia già in quella schermata; in quest'ultimo caso il bottone non scatenerà nulla

Se l'utente dovesse premere sul bottone "Dinamiche" verrà inviato alla pagina di selezione delle icone dinamiche, sempre se non sia già in quella schermata; in quest'ultimo caso il bottone non scatenerà nulla.

Se l'utente dovesse premere sul bottone "Predefinite" verrà inviato alla pagine di selezione delle icone predefinite descritta precedentemente, sempre se non sia già in quella schermata; in quest'ultimo caso il bottone non scatenerà nulla.

B.6 REQ01-060 riferito all'esigenza ESG01-006:

Quando l'utente, che sia nella schermata di selezione delle icone predefinite, dinamiche o personalizzate, ha scelto e cliccato l'icona che vorrà applicare a quella determinata matrice, potrà confermare e applicare la sua scelta premendo il tasto "Fatto", presente in alto a destra. Il tasto si abilita dopo che l'utente ha effettivamente cliccato su un'icona. Una volta premuto "Fatto" verrà visualizzato un pop-up che indica la conferma dei cambiamenti e l'icona scelta verrà visualizzata, nella sua posizione, nella pagina con la rappresentazione stilizzata del comando. Inoltre, se un utente esegue la selezione, cioè il clic su un'icona su una determinata pagina, questa verrà evidenziata da un contorno di colore diverso. La scelta verrà evidenziata anche nei bottoni di selezione della categoria attraverso la comparsa di un puntino grigio. Questo puntino ha lo scopo di identificare la categoria di icone in cui si è fatta la scelta, in modo da renderla sempre presente e facilmente individuabile, anche a seguito di un cambio di pagina tra le tre categorie disponibili dopo la selezione.

B.7 REQ01-070 riferito all'esigenza ESG01-007:

All'utente, una volta selezionata e confermata una determinata icona o animazione, qualora alla funzione selezionata siano associati più matrici o tasti dello stesso tipo, viene data la possibilità di applicare la stessa personalizzazione contemporaneamente a tutti, oppure solo al tasto o alla matrice che si sta personalizzando. Questo viene fatto attraverso un pop-up di richiesta, subito dopo la conferma, nel quale l'utente può scegliere se assegnare questa icona alla matrice selezionata o a tutte le matrici dello stesso tipo. In caso l'utente decida di applicarlo solo alla matrice selezionata, questo vedrà solamente la propria icona scelta applicata al comando stilizzato, mentre, se scegliesse di applicarlo a tutte, vedrà anche un pop-up che avvisa del cambiamento avvenuto.

B.8 REQ01-080 riferito all'esigenza ESG01-008:

Per permettere all'utente di personalizzare le matrici Led per funzione nella schermata di personalizzazione del dispositivo vengono disposti i quadratini, cliccabili per l'assegnamento dell'icona alla posizione, solo nelle posizioni che posso essere personalizzabili in quel dispositivo usato per quella determinata funzione, oscurando quelli non disponibili. Inoltre una volta che l'utente preme in uno dei quadratini viene mandato alla schermata di selezione dell'icona dove visualizzerà e potrà selezionare solamente le icone che può utilizzare in quel determinato dispositivo usato per quella determinata funzione.

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di tende e tapparelle, indipendentemente dalla presenza della posizione o no, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per le matrici sopra e sotto vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di tende e tapparelle senza posizione, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate". Inoltre potrà anche scegliere se assegnare o no una possibile animazione associata all'icona, per l'apertura o chiusura, da una lista predefinita.

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di tende e tapparelle con posizione, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare tutti e tre i tipi di icone: predefinite, personalizzate o dinamiche (tra queste sarà presente anche l'icona speciale slider) e di conseguenza vedrà tutti e tre i bottoni "Predefinite", "Personalizzate" e "Dinamiche". Inoltre potrà anche scegliere se assegnare o no una possibile animazione associata all'icona, per l'apertura o chiusura, da una lista predefinita.

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di tende e tapparelle senza posizione, dotato di una matrice (centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate". Inoltre potrà anche scegliere se assegnare o no una possibile animazione associata all'icona, per l'apertura o chiusura, da una lista predefinita.

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di tende e tapparelle con posizione, dotato di una matrice (centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare tutti e tre i tipi di icone: predefinite, personalizzate o dinamiche (tra queste sarà presente anche l'icona speciale slider). Di conseguenza, vedrà tutti e tre i bottoni "Predefinite", "Personalizzate" e "Dinamiche". Inoltre potrà anche scegliere se assegnare o no una possibile animazione associata all'icona, per l'apertura o chiusura, da una lista predefinita.

Se l'utente volesse personalizzare un dispositivo assegnato al controllo delle luci dimmerabili o no, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per le matrici sopra e sotto vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo delle luci non dimmerabili, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo delle luci dimmerabili, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare tutti e tre i tipi di icone: predefinite, personalizzate o dinamiche (tra queste sarà presente anche l'icona speciale slider). Di conseguenza, vedrà tutti e tre i bottoni "Predefinite", "Personalizzate" e "Dinamiche".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo delle luci non dimmerabili, dotato di una matrice (centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo delle luci dimmerabili, dotato di una matrice(centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare tutti e tre i tipi di icone: predefinite, personalizzate o dinamiche (tra queste sarà presente anche l'icona speciale slider). Di conseguenza, vedrà tutti e tre i bottoni "Predefinite", "Personalizzate" e "Dinamiche".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di scenario, dotato di 3 matrici (sopra, centro, sotto) nella schermata di scelta dell'icona per le matrici sopra e sotto vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate". Inoltre potrà anche scegliere se assegnare o no una possibile animazione associata all'icona tra quelle proposte.

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di scenario, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di scenario, dotato di una matrice(centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sistemi di accessi e presenze, dotato di 3 matrici (sopra, centro, sotto) nella schermata di scelta dell'icona per le matrici sopra e sotto vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate". Se l'utente volesse

personalizzare un dispositivo assegnato al controllo di sistemi di accessi e presenze, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sistemi di accessi e presenze, dotato di una matrice(centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sistemi audio, con o senza zona audio, dotato di 3 matrici (sopra, centro, sotto) nella schermata di scelta dell'icona per le matrici sopra e sotto vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sistemi audio, senza zona audio, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sistemi audio, con zona audio, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare tutti e tre i tipi di icone: predefinite, personalizzate o dinamiche (tra queste sarà presente anche l'icona speciale slider). Di conseguenza, vedrà tutti e tre i bottoni "Predefinite", "Personalizzate" e "Dinamiche".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sistemi audio, senza zona audio, dotato di una matrice (centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sistemi audio, con zona audio, dotato di una matrice(centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare tutti e tre i tipi di icone: predefinite, personalizzate o dinamiche (tra queste sarà presente anche l'icona speciale slider). Di conseguenza, vedrà tutti e tre i bottoni "Predefinite", "Personalizzate" e "Dinamiche".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sistemi del clima, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per le matrici sopra e sotto vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sistemi del clima, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sistemi del clima, dotato di una matrice(centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sensori, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per le matrici sopra e sotto vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sensori, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di sensori, dotato di una matrice(centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di energia, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per le matrici sopra e sotto vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di energia, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato al controllo di energia, dotato di una matrice(centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato a dispositivi di tipo varie, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per le matrici sopra e sotto vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato a dispositivi di tipo varie, nello specifico non regolatore, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite" e "Personalizzate".

Se l'utente volesse personalizzare un dispositivo assegnato a dispositivi di tipo varie, nello specifico regolatore, dotato di 3 matrici (sopra, centro, sotto), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare tutti e tre i tipi di icone: predefinite, personalizzate o dinamiche (tra queste sarà presente anche l'icona speciale slider). Di conseguenza, vedrà tutti e tre i bottoni "Predefinite", "Personalizzate" e "Dinamiche".

Se l'utente volesse personalizzare un dispositivo assegnato a dispositivi di tipo varie, nello specifico non regolatore, dotato di una matrice (centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare solo icone predefinite o personalizzate e di conseguenza vedrà solo i due bottoni "Predefinite "e "Personalizzate ".

Se l'utente volesse personalizzare un dispositivo assegnato a dispositivi di tipo varie, nello specifico regolatore, dotato di una matrice(centro), nella schermata di scelta dell'icona per la matrice centrale vedrà che potrà selezionare tutti e tre i tipi di icone: predefinite, personalizzate o dinamiche (tra queste sarà presente anche l'icona speciale slider). Di conseguenza, vedrà tutti e tre i bottoni "Predefinite ", "Personalizzate "e "Dinamiche ".

Bibliografia

[1] Vimar: https://www.vimar.com/it/it

[2] Domotica:

Renato Zaccaria, 2007, *Mondo Digitale n.4*,

[3] Domotica:

Domenico Triscuglio, 2009, *Introduzione alla domotica*, terza edizione, Tecniche nuove.

[4] Smart home:

Alam, Muhammad Raisul, Reaz, Mamun Bin Ibne, Ali, Mohd Alauddin Mohd, 2012, *A Review of Smart Homes - Past, Present, and Future*, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)

[5] Smart home:

Li Jiang, Da-You Liu, Bo Yang, 2004, *Smart home research*, IEEE 2004 International Conference on Machine Learning and Cybernetics, Shanghai, China

[6] Gateway:

Vittorio Miori, Dario Russo, Luca Ferrucci, Loredana Pillitteri,2019, *L'adattamento dei gateway al framework di interoperabilità domotica,* ISTI Technical reports

[7] Scrum:

Kenneth S. Rubin, 2012, *Essential Scrum: A Practical Guide to the Most Popular Agile Process,* Addison-Wesley Professional

[8] Android Studio: https://developer.android.com/studio/intro?hl=it

[9] Bitbucket: https://www.atlassian.com/it/software/bitbucket

[10] Confluence: https://www.atlassian.com/it/software/confluence

[11] Jira Software: https://www.atlassian.com/it/software/jira

[12] Git: https://git-scm.com/

[13] Ktlint: https://pinterest.github.io/ktlint/1.0.0/

[14] Kotlin Code Style: https://developer.android.com/kotlin/style-guide?hl=en

[15] Microsoft Teams: https://www.microsoft.com/it-it/microsoft-teams/group-chat-software

[16] Adobe XD: https://helpx.adobe.com/it/xd/help/adobe-xd-overview.html

[17] Linphone: https://www.linphone.org/

[18] Model-View-ViewModel:

https://learn.microsoft.com/en-us/dotnet/architecture/maui/mvvm#the-mvvm-pattern

[19] Clean Architecture:

Eran Boudjnah, 2022, *Clean Architecture for Android: Implement Expert-led Design Patterns to Build Scalable, Maintainable, and Testable Android Apps*, BPB Publications