

Specifica Tecnica

Gruppo LaTeXBiscotti — Progetto UMAP

Informazioni sul documento	
Versione	2.0.0
Redazione	Andrea Barcaro
	Giovanni Rodighiero
	Pietro Marchetto
	Marco Baggio
	Federica Speggiorin
Verifica	Filippo Todescato
	Simone Garbin
Approvazione	Filippo Todescato
Uso	Esterno
Distribuzione	Prof. Tullio Vardanega
	Prof. Riccardo Cardin
	Gruppo LaTeXBiscotti

Descrizione

Questo documento descrive la specifica tecnica e l'architettura dell'applicazione UMAP.



Diario delle Modifiche

Versione	Data	Persone coinvolte	Descrizione
1.0.0	2016-04-11	Filippo Todescato (Responsabile)	Approvazione Documento.
0.1.0	2016-04-03	Simone Garbin (Verificatore)	Verifica Documento.
0.0.19	2016-03-26	Federica Speggiorin, Andrea Barcaro (Progettisti)	Stesura sezione §8.2.
0.0.18	2016-03-26	Federica Speggiorin, Pietro Marchetto (Progettisti)	Stesura sezione §8.1.
0.0.17	2016-03-26	Pietro Marchetto (Progettista)	Stesura sezione §4.4.
0.0.16	2016-03-24	Marco Baggio (Progettista)	Stesura sezione §4.3.
0.0.15	2016-03-22	Filippo Todescato (Responsabile)	Stesura sezione §7.
0.0.14	2016-03-22	Pietro Marchetto (Progettista)	Stesura sezione §4.2.
0.0.13	2016-03-20	Filippo Todescato (Progettista)	Stesura sezione §4.1.
0.0.12	2016-03-17	Pietro Marchetto (Progettista)	Correzione errori vari sezione §5 e relative sottosezioni.
0.0.11	2016-03-16	Pietro Marchetto (Progettista)	Inserimento sezioni §5.5 e §5.6 con relative sottosezioni.
0.0.10	2016-03-15	Andrea Barcaro (Progettista)	Stesura Appendice A, sezione §A.1.
0.0.9	2016-03-15	Andrea Barcaro (Progettista)	Inizio stesura sezione §6, stesura sezione §6.1.
0.0.8	2016-03-15	Federica Speggiorin (Progettista)	Stesura completa indice.
0.0.7	2016-03-14	Giovanni Rodighiero, Federica Speggiorin (Progettisti)	Stesura sezione §3.2.
0.0.6	2016-03-13	Federica Speggiorin (Progettista)	Stesura sezioni dalla §5.1 alla §5.4.
0.0.5	2016-03-12	Pietro Marchetto (Progettista)	Stesura sezione §3.1.
0.0.4	2016-03-11	Giovanni Rodighiero (Progettista)	Modifica sezioni §2.2, §2.8; Stesura sezioni §2.3, §2.5, §2.6, §2.7, §2.9.
0.0.3	2016-03-11	Andrea Barcaro (Progettista)	Stesura sezioni §2.2, §2.4, §2.8.



0.0.2	2016-03-10	Andrea Barcaro (Progettista)	Stesura indice sezione §2, con scrittura sezione §2.1.
0.0.1	2016-03-10	Andrea Barcaro (Progettista)	Creazione documento e prima stesura sezione §1.



Indice

1	Introduzione	11
1.1	Scopo del Documento	11
1.2	Scopo del Prodotto	11
1.3	Glossario	11
1.4	Riferimenti	11
1.4.1	Normativi	11
1.4.2	Informativi	11
2	Tecnologie Utilizzate	13
2.1	Scala	13
2.2	Play Framework	13
2.3	Amazon Web Services	14
2.4	Node.js	14
2.5	MongoDB	15
2.6	HTML5	15
2.7	SASS	15
2.8	Twitter Bootstrap	15
2.9	Javascript	16
3	Definizione del Prodotto	17
3.1	Metodo e formalismo di specifica	17
3.2	Architettura generale	17
4	Descrizione Componenti e Classi	21
4.1	Moduli Esterni	21
4.1.1	Amazon Machine Learning (AWS SDK)	21
4.1.1.1	Descrizione	21
4.1.1.2	Utilizzo	21
4.1.2	Silhouette	21
4.1.2.1	Descrizione	21
4.1.2.2	Utilizzo	22
4.1.3	MongoDB Module	22
4.1.3.1	Descrizione	22
4.1.3.2	Utilizzo	22
4.1.4	BrokerMQTT Module	22
4.1.4.1	Descrizione	22
4.1.4.2	Utilizzo	22
4.2	Models	23
4.2.1	app :: models :: User	23
4.2.1.1	Descrizione	23
4.2.1.2	Utilizzo	23
4.2.1.3	Relazioni con altre classi	23
4.2.2	app :: models :: Company	24
4.2.2.1	Descrizione	24
4.2.2.2	Utilizzo	24
4.2.2.3	Relazioni con altre classi	24
4.2.3	app :: models :: Thing	24
4.2.3.1	Descrizione	24
4.2.3.2	Utilizzo	24
4.2.3.3	Relazioni con altre classi	24
4.2.4	app :: models :: ThingType	24
4.2.4.1	Descrizione	24



4.2.4.2	Utilizzo	25
4.2.4.3	Relazioni con altre classi	25
4.2.5	app :: models :: Notification	25
4.2.5.1	Descrizione	25
4.2.5.2	Utilizzo	25
4.2.5.3	Relazioni con altre classi	25
4.2.6	app :: models :: Charts	25
4.2.6.1	Descrizione	25
4.2.6.2	Utilizzo	26
4.2.6.3	Relazioni con altre classi	26
4.2.7	app :: models :: Roles	26
4.2.7.1	Descrizione	26
4.2.7.2	Utilizzo	26
4.2.7.3	Relazioni con altre classi	26
4.2.8	app :: models :: daos :: chart :: ChartDAO	26
4.2.8.1	Descrizione	26
4.2.8.2	Utilizzo	26
4.2.8.3	Relazioni con altre classi	27
4.2.9	app :: models :: daos :: chart :: ChartDAOImpl	27
4.2.9.1	Descrizione	27
4.2.9.2	Utilizzo	27
4.2.9.3	Relazioni con altre classi	27
4.2.10	app :: models :: daos :: company :: CompanyDAO	27
4.2.10.1	Descrizione	27
4.2.10.2	Utilizzo	27
4.2.10.3	Relazioni con altre classi	27
4.2.11	app :: models :: daos :: company :: CompanyDAOImpl	28
4.2.11.1	Descrizione	28
4.2.11.2	Utilizzo	28
4.2.11.3	Relazioni con altre classi	28
4.2.12	app :: models :: daos :: thing :: ThingDAO	28
4.2.12.1	Descrizione	28
4.2.12.2	Utilizzo	28
4.2.12.3	Relazioni con altre classi	28
4.2.13	app :: models :: daos :: thing :: ThingDAOImpl	28
4.2.13.1	Descrizione	28
4.2.13.2	Utilizzo	28
4.2.13.3	Relazioni con altre classi	29
4.2.14	app :: models :: daos :: thingType :: ThingTypeDAO	29
4.2.14.1	Descrizione	29
4.2.14.2	Utilizzo	29
4.2.14.3	Relazioni con altre classi	29
4.2.15	app :: models :: daos :: thingtype :: ThingTypeDAOImpl	29
4.2.15.1	Descrizione	29
4.2.15.2	Utilizzo	29
4.2.15.3	Relazioni con altre classi	29
4.2.16	app :: models :: daos :: user :: UserDAO	30
4.2.16.1	Descrizione	30
4.2.16.2	Utilizzo	30
4.2.16.3	Relazioni con altre classi	30
4.2.17	app :: models :: daos :: user :: UserDAOImpl	30
4.2.17.1	Descrizione	30
4.2.17.2	Utilizzo	30
4.2.17.3	Relazioni con altre classi	30



4.2.18	app :: models :: daos :: services :: user :: UserService	30
4.2.18.1	Descrizione	30
4.2.18.2	Utilizzo	30
4.2.18.3	Relazioni con altre classi	31
4.2.19	app :: models :: daos :: services :: user :: UserServiceImpl	31
4.2.19.1	Descrizione	31
4.2.19.2	Utilizzo	31
4.2.19.3	Relazioni con altre classi	31
4.3	Controllers	31
4.3.1	app :: controllers :: superAdmin :: SuperAdminController	31
4.3.1.1	Descrizione	31
4.3.1.2	Utilizzo	32
4.3.1.3	Relazioni con altre classi	32
4.3.2	app :: controllers :: superAdmin :: UserController	32
4.3.2.1	Descrizione	32
4.3.2.2	Utilizzo	32
4.3.2.3	Relazioni con altre classi	32
4.3.3	app :: controllers :: superAdmin :: CompanyController	32
4.3.3.1	Descrizione	32
4.3.3.2	Utilizzo	32
4.3.3.3	Relazioni con altre classi	33
4.3.4	app :: controllers :: superAdmin :: ThingController	33
4.3.4.1	Descrizione	33
4.3.4.2	Utilizzo	33
4.3.4.3	Relazioni con altre classi	33
4.3.5	app :: controllers :: admin :: AdminController	34
4.3.5.1	Descrizione	34
4.3.5.2	Utilizzo	34
4.3.5.3	Relazioni con altre classi	34
4.3.6	app :: controllers :: admin :: UserController	34
4.3.6.1	Descrizione	34
4.3.6.2	Utilizzo	34
4.3.6.3	Relazioni con altre classi	34
4.3.7	app :: controllers :: admin :: ThingController	35
4.3.7.1	Descrizione	35
4.3.7.2	Utilizzo	35
4.3.7.3	Relazioni con altre classi	35
4.3.8	app :: controllers :: admin :: ThingTypeController	35
4.3.8.1	Descrizione	35
4.3.8.2	Utilizzo	35
4.3.8.3	Relazioni con altre classi	35
4.3.9	app :: controllers :: admin :: ChartController	36
4.3.9.1	Descrizione	36
4.3.9.2	Utilizzo	36
4.3.9.3	Relazioni con altre classi	36
4.3.10	app :: controllers :: user :: UserController	36
4.3.10.1	Descrizione	36
4.3.10.2	Utilizzo	36
4.3.10.3	Relazioni con altre classi	36
4.3.11	app :: controllers :: user :: ThingController	37
4.3.11.1	Descrizione	37
4.3.11.2	Utilizzo	37
4.3.11.3	Relazioni con altre classi	37
4.3.12	app :: controllers :: user :: ThingTypeController	37



4.3.12.1	Descrizione	37
4.3.12.2	Utilizzo	37
4.3.12.3	Relazioni con altre classi	37
4.3.13	app :: controllers :: shared :: adminUser :: NotificationController	38
4.3.13.1	Descrizione	38
4.3.13.2	Utilizzo	38
4.3.13.3	Relazioni con altre classi	38
4.3.14	app :: controllers :: shared :: authentication :: AuthenticationController	38
4.3.14.1	Descrizione	38
4.3.14.2	Utilizzo	38
4.3.14.3	Relazioni con altre classi	39
4.3.15	app :: controllers :: shared :: account :: AccountController	39
4.3.15.1	Descrizione	39
4.3.15.2	Utilizzo	39
4.3.15.3	Relazioni con altre classi	39
4.3.16	app :: controllers :: shared :: account :: PasswordRecoverController	39
4.3.16.1	Descrizione	39
4.3.16.2	Utilizzo	39
4.3.16.3	Relazioni con altre classi	39
4.3.17	app :: controllers :: shared :: account :: PasswordResetController	40
4.3.17.1	Descrizione	40
4.3.17.2	Utilizzo	40
4.3.17.3	Relazioni con altre classi	40
4.3.18	app :: controllers :: shared :: administrator :: EngineController	40
4.3.18.1	Descrizione	40
4.3.18.2	Utilizzo	40
4.3.18.3	Relazioni con altre classi	40
4.4	Views	42
4.4.1	app :: views :: superAdmin :: home	42
4.4.1.1	Descrizione	42
4.4.1.2	Utilizzo	42
4.4.2	app :: views :: superAdmin :: user :: users	43
4.4.2.1	Descrizione	43
4.4.2.2	Utilizzo	43
4.4.3	app :: views :: superAdmin :: user :: addUser	43
4.4.3.1	Descrizione	43
4.4.3.2	Utilizzo	43
4.4.4	app :: views :: superAdmin :: user :: editUser	43
4.4.4.1	Descrizione	43
4.4.4.2	Utilizzo	43
4.4.5	app :: views :: superAdmin :: company :: companies	43
4.4.5.1	Descrizione	43
4.4.5.2	Utilizzo	43
4.4.6	app :: views :: superAdmin :: company :: addCompany	43
4.4.6.1	Descrizione	43
4.4.6.2	Utilizzo	43
4.4.7	app :: views :: superAdmin :: company :: editCompany	44
4.4.7.1	Descrizione	44
4.4.7.2	Utilizzo	44
4.4.8	app :: views :: superAdmin :: thing :: things	44
4.4.8.1	Descrizione	44
4.4.8.2	Utilizzo	44
4.4.9	app :: views :: superAdmin :: thing :: addThing	44
4.4.9.1	Descrizione	44



4.4.9.2	Utilizzo	44
4.4.10	app :: views :: superAdmin :: thing :: editThing	44
4.4.10.1	Descrizione	44
4.4.10.2	Utilizzo	44
4.4.11	app :: views :: superAdmin :: thingType :: thinTypes	44
4.4.11.1	Descrizione	44
4.4.11.2	Utilizzo	44
4.4.12	app :: views :: superAdmin :: thingType :: addThingType	45
4.4.12.1	Descrizione	45
4.4.12.2	Utilizzo	45
4.4.13	app :: views :: superAdmin :: thingType :: editThingType	45
4.4.13.1	Descrizione	45
4.4.13.2	Utilizzo	45
4.4.14	app :: views :: superAdmin :: engine :: engine	45
4.4.14.1	Descrizione	45
4.4.14.2	Utilizzo	45
4.4.15	app :: views :: admin :: thing :: thingDetails	45
4.4.15.1	Descrizione	45
4.4.15.2	Utilizzo	45
4.4.16	app :: views :: admin :: thingType :: thingTypeDetails	45
4.4.16.1	Descrizione	45
4.4.16.2	Utilizzo	45
4.4.17	app :: views :: admin :: chart :: newChart	46
4.4.17.1	Descrizione	46
4.4.17.2	Utilizzo	46
4.4.18	app :: views :: admin :: engine :: editFunctions	46
4.4.18.1	Descrizione	46
4.4.18.2	Utilizzo	46
4.4.19	app :: views :: admin :: engine :: editData	46
4.4.19.1	Descrizione	46
4.4.19.2	Utilizzo	46
4.4.20	app :: views :: admin :: user :: users	46
4.4.20.1	Descrizione	46
4.4.20.2	Utilizzo	46
4.4.21	app :: views :: admin :: user :: addUser	46
4.4.21.1	Descrizione	46
4.4.21.2	Utilizzo	46
4.4.22	app :: views :: admin :: user :: editUser	47
4.4.22.1	Descrizione	47
4.4.22.2	Utilizzo	47
4.4.23	app :: views :: admin :: home	47
4.4.23.1	Descrizione	47
4.4.23.2	Utilizzo	47
4.4.24	app :: views :: user :: home	47
4.4.24.1	Descrizione	47
4.4.24.2	Utilizzo	47
4.4.25	app :: views :: user :: thing :: thingDetails	47
4.4.25.1	Descrizione	47
4.4.25.2	Utilizzo	47
4.4.26	app :: views :: user :: thingType :: thingTypeDetails	47
4.4.26.1	Descrizione	47
4.4.26.2	Utilizzo	47
4.4.27	app :: views :: shared :: adminUser :: things	48
4.4.27.1	Descrizione	48



4.4.27.2	Utilizzo	48
4.4.28	app :: views :: shared :: adminUser :: thingTypes	48
4.4.28.1	Descrizione	48
4.4.28.2	Utilizzo	48
4.4.29	app :: views :: shared :: adminUser :: notifications	48
4.4.29.1	Descrizione	48
4.4.29.2	Utilizzo	48
4.4.30	app :: views :: shared :: adminUser :: addNotification	48
4.4.30.1	Descrizione	48
4.4.30.2	Utilizzo	48
4.4.31	app :: views :: shared :: adminUser :: editNotification	48
4.4.31.1	Descrizione	48
4.4.31.2	Utilizzo	48
4.4.32	app :: views :: shared :: authentication :: singIn	49
4.4.32.1	Descrizione	49
4.4.32.2	Utilizzo	49
4.4.33	app :: views :: shared :: authentication :: singOut	49
4.4.33.1	Descrizione	49
4.4.33.2	Utilizzo	49
4.4.34	app :: views :: shared :: authentication :: passwordRecover	49
4.4.34.1	Descrizione	49
4.4.34.2	Utilizzo	49
4.4.35	app :: views :: shared :: authentication :: resetPassword	49
4.4.35.1	Descrizione	49
4.4.35.2	Utilizzo	49
4.4.36	app :: views :: shared :: account :: editAccount	49
4.4.36.1	Descrizione	49
4.4.36.2	Utilizzo	49
4.4.37	app :: views :: shared :: layout :: header	50
4.4.37.1	Descrizione	50
4.4.37.2	Utilizzo	50
4.4.38	app :: views :: shared :: layout :: footer	50
4.4.38.1	Descrizione	50
4.4.38.2	Utilizzo	50
4.4.39	app :: views :: shared :: layout :: layout	50
4.4.39.1	Descrizione	50
4.4.39.2	Utilizzo	50
4.4.40	app :: views :: shared :: layout :: nav	50
4.4.40.1	Descrizione	50
4.4.40.2	Utilizzo	50
4.4.41	app :: views :: shared :: errors :: 404	50
4.4.41.1	Descrizione	50
4.4.41.2	Utilizzo	50
4.4.42	app :: views :: shared :: errors :: 502	50
4.4.42.1	Descrizione	50
4.4.42.2	Utilizzo	51
5	Diagrammi di attività	52
5.1	Attività Principali	53
5.2	Utente non Autenticato	55
5.2.1	Recupera password	55
5.2.2	Esegui reset password	56
5.2.3	Effettua login	57
5.3	Utente Autenticato	58



5.3.1	Modifica profilo	58
5.3.1.1	Modifica informazioni personali	59
5.3.1.2	Modifica email	60
5.3.1.3	Modifica password	61
5.4	User	62
5.4.1	Visualizza Oggetti	62
5.4.2	Visualizza modelli	63
5.4.3	Definizione sistemi di allerting	64
5.5	Admin	65
5.5.1	Visualizza Oggetti	65
5.5.1.1	Seleziona Oggetto	66
5.5.2	Visualizza Modelli	67
5.5.2.1	Seleziona Modello	68
5.5.3	Gestione Utenti	69
5.5.3.1	Inserisci Utente	70
5.5.3.2	Modifica Utente	71
5.5.4	Gestione Engine	72
5.5.5	Creazione Grafici	73
5.6	Super Admin	74
5.6.1	Gestione Company	74
5.6.1.1	Inserisci Company	75
5.6.1.2	Modifica Company	76
5.6.2	Gestione Utenti	77
5.6.2.1	Inserisci Utente	78
5.6.2.2	Modifica Utente	79
5.6.3	Gestione Oggetti	80
5.6.3.1	Gestisci Thing	81
5.6.3.2	Gestisci Modelli	84
5.6.4	Gestione Engine	87
6	Design Pattern	88
6.1	Design Pattern Architettureali	88
6.1.1	MVC	88
7	Stime di fattibilità e bisogno di risorse	89
8	Tracciamento	90
8.1	Tracciamento requisiti-componenti	90
8.2	Tracciamento componenti-requisiti	104
A	Descrizione Design Pattern	112
A.1	Design Pattern Architettureali	112
A.1.1	MVC	112
A.1.1.1	Model	112
A.1.1.2	View	113
A.1.1.3	Controller	113

Elenco delle tabelle

2	Tracciamento requisiti-componenti.	103
3	Tracciamento componenti-requisiti.	111



Elenco delle figure

1	Play! framework MVC model.	17
2	Ciclo di vita Play! framework.	18
3	Architettura Generale.	19
4	Diagramma package - Visione generale package Models	23
5	Diagramma package - Visione generale package Controllers	31
6	Diagramma package - Visione generale package Views	42
7	Diagramma di attività - Attività Principali.	53
8	Diagramma di attività - Recupera password.	55
9	Diagramma di attività - Esegui reset password.	56
10	Diagramma di attività - Effettua login.	57
11	Diagramma di attività - Modifica profilo.	58
12	Diagramma di attività - Modifica informazioni personali.	59
13	Diagramma di attività - Modifica email.	60
14	Diagramma di attività - Modifica password.	61
15	Diagramma di attività - Visualizza oggetti.	62
16	Diagramma di attività - Visualizza modelli.	63
17	Diagramma di attività - Definizione sistemi di alerting.	64
18	Diagramma di attività - Visualizza Oggetti.	65
19	Diagramma di attività - Seleziona Oggetto.	66
20	Diagramma di attività - Visualizza Modelli.	67
21	Diagramma di attività - Seleziona Modello.	68
22	Diagramma di attività - Gestione Utenti.	69
23	Diagramma di attività - Inserisci Utente.	70
24	Diagramma di attività - Modifica Utente.	71
25	Diagramma di attività - Gestione Engine.	72
26	Diagramma di attività - Creazione Grafici.	73
27	Diagramma di attività - Gestione Company.	74
28	Diagramma di attività - Inserisci Company.	75
29	Diagramma di attività - Modifica Company.	76
30	Diagramma di attività - Gestione Utenti.	77
31	Diagramma di attività - Inserisci Utente.	78
32	Diagramma di attività - Modifica Utente.	79
33	Diagramma di attività - Gestione Oggetti.	80
34	Diagramma di attività - Gestisci Thing.	81
35	Diagramma di attività - Inserisci Thing.	82
36	Diagramma di attività - Modifica Thing.	83
37	Diagramma di attività - Gestisci Modelli.	84
38	Diagramma di attività - Inserisci Modello.	85
39	Diagramma di attività - Modifica Modello.	86
40	Diagramma di attività - Gestione Engine.	87
41	MVC - Diagramma di interazione	112



Introduzione

Scopo del Documento

Il documento in esame si pone come obiettivo quello di definire la progettazione ad alto livello del prodotto software UMAP. In seguito, verrà presentata l'architettura generale ideata per il progetto in esame, illustrandone le varie caratteristiche, i componenti software che la compongono e una descrizione approfondita dei *Design Pattern_G* che si andranno ad utilizzare.

Scopo del Prodotto

Lo scopo del progetto è la realizzazione di un *algoritmo predittivo_G* in ambiente *Internet of Things_G*, in grado di analizzare i dati provenienti da “oggetti” inseriti in diversi contesti e fornire delle previsioni su possibili guasti o interazioni con nuovi utenti ed identificare dei pattern di comportamento di questi ultimi, al fine di prevedere le azioni degli stessi rispetto ad altri oggetti o ad altri contesti.

Glossario

Per evitare tutte le possibili incomprensioni e ambiguità sul linguaggio utilizzato e per massimizzare la comprensione da parte di tutti del documento, della terminologia specifica e di quella di dominio, degli acronimi e di tutte quelle parole che necessitano chiarimento, è stato redatto un Glossario, consultabile nel documento *Glossario v3.0.0*. Tutti i termini, la cui spiegazione è presente in *Glossario v3.0.0*, sono evidenziati, a tale scopo, con una G pedice.

Riferimenti

Normativi

- **Analisi dei Requisiti:** *Analisi dei Requisiti v3.0.0*;
- **Norme di Progetto:** *Norme di Progetto v3.0.0*;

Informativi

- **Informazioni Scala**
<http://jim-mcbeath.blogspot.it/2010/12/scala-pros-and-cons.html>
- **Documentazione Play Framework**
<https://www.playframework.com/documentation/2.5.x/Home>
- **Documentazione AWS Machine Learning**
<https://aws.amazon.com/it/machine-learning/>
- **Informazioni Node.js**
<http://voidcanvas.com/describing-node-js/>
- **Documentazione MongoDB**
<https://docs.mongodb.org/manual/>
- **Documentazione Twitter Bootstrap**
<https://bootstrapdocs.com/>



- **Design Pattern**

- http://www.math.unipd.it/~rcardin/sweb/Design%20Pattern%20Architetturali%20-%20Model%20View%20Controller_4x4.pdf
- <http://www.claudiodesio.com/ooa&d/mvc.htm>



Tecnologie Utilizzate

In tale sezione vengono descritte e discusse tutte le tecnologie che il gruppo di lavoro LaTeXBiscotti intende utilizzare per lo sviluppo del progetto UMAP. Verrà inoltre fornita una motivazione all'utilizzo di ogni singola tecnologia scelta.

Scala

Scala è un linguaggio di programmazione studiato ed ideato per integrare le caratteristiche e le funzionalità dei linguaggi orientati agli oggetti e dei linguaggi funzionali. Il codice prodotto da questo linguaggio, una volta compilato, produce del *bytecode_G* che può essere eseguito su una Java *JVM_G*.

Aspetti **positivi** considerati:

- **Alta Produttività:** da studi effettuati, si è riscontrato che, dato un programma scritto in Scala, le linee di codice di uno equivalente scritto in Java stanno in un rapporto che può variare da 1:10 a 1:2. Assunte tali considerazioni, si capisce come si possa essere molto più produttivi in un periodo di tempo breve, paragonabile alla fase di Codifica dell'applicativo software;
- **Integrazione con Java:** Scala si integra perfettamente con il linguaggio Java, permettendo di usufruire a pieno della vasta mole di librerie ed *API_G* fornite dal suddetto linguaggio;
- **Interesse del Gruppo:** il gruppo di lavoro, sin da subito, ha espresso la forte volontà di esplorare nuove tecnologie, non strettamente legate ai corsi universitari frequentati; tale scelta è dettata quindi anche per poter arricchire il numero delle competenze di ogni membro del gruppo.

Aspetti **negativi** considerati:

- **Sintassi:** proprio per consentire un minor numero di linee di codice a parità di risultato ottenuto con altri linguaggi, la sintassi può risultare spesso complessa e controintuitiva, soprattutto a chi è alle prime armi.

Play Framework

Play Framework è un framework *open-source_G*, adotta il pattern MVC ed è ottimizzato per la costruzione di applicazioni web sviluppate in Java o Scala.

Aspetti **positivi** considerati:

- **Incremento Produttività:** utilizzando nativamente il paradigma *Convention Over Configuration_G* permette di velocizzare di molto lo sviluppo rendendo automatica l'implementazione di varie componenti a partire da altre definite dal programmatore, così facendo le varie configurazioni che si creano diventano molto più manutenibili, essendo generate dal framework. Rimane comunque possibile sovrascrivere e personalizzare tali configurazioni qualora lo si voglia, senza perdere quindi flessibilità e libertà di sviluppo;
- **Hit Refresh Workflow:** il framework implementa una funzionalità che consente di visualizzare i cambiamenti e le modifiche apportate semplicemente aggiornano la pagina del browser, gestendo quindi automaticamente una eventuale ricompilazione e riavvio dell'applicazione, velocizzando lo sviluppo di quest'ultima;
- **Templating html:** viene offerta la possibilità di scrivere codice html tramite un sistema di templating che utilizza il motore *Groovy_G*, così facendo è possibile gestire vari pezzi di codice con elementi della programmazione classica come espressioni condizionali ed ereditarietà tra più file;
- **Assets compiling:** è presente un compilatore per codice come *sass_G* o *less_G* che genera automaticamente il corrispettivo css;



- **Segnalazione degli Errori:** ogni errore rilevato nel codice viene riportato nella finestra del browser, specificando in maniera ottimale il tipo d'errore, la sua posizione nel codice, il percorso del file in cui risiede e varie informazioni utili in fase di sviluppo. L'insieme di questi elementi consentirà al team di mantenere un livello elevato di efficienza, in quanto tutti gli errori verranno individuati e corretti velocemente, senza perdere troppo tempo per identificarli;
- **Licenza Apache 2.0:** tale strumento viene rilasciato sotto la licenza *Apache 2.0_G*. Pertanto, a prodotto finito e qualora questo si inserisse nel mercato, il gruppo non sarebbe costretto a rilasciare anche il codice sorgente: è richiesto solamente il rilascio di un richiamo al fatto che l'applicazione sviluppata ha come punto di partenza uno strumento rilasciato sotto licenza *Apache 2.0_G*.

Aspetti **negativi** considerati:

- **Diffusione:** l'utilizzo di Play per la realizzazione di applicazioni web è molto inferiore ad altre alternative equivalenti, come ad esempio Ruby on Rails o Sails.js, ciò si riflette sulla quantità di guide, tutorial ed esempi di progetti da utilizzare per fare pratica con il framework e l'integrazione dello stesso con i moduli esterni scelti. Tali moduli esterni, sviluppati spesso da terzi, sono disponibili in un numero piuttosto ridotto e ciò limita la libertà di scelta anche in relazione alla documentazione spesso non chiara o con pochi esempi da seguire.
Un ulteriore svantaggio è il passaggio dalla versione 2.3 alla 2.4, che influenza pesantemente la struttura del codice introducendo la Dependency Injection e costringendo a un refactoring delle classi. Ciò si riflette nel limitare ancora di più la possibilità di apprendimento attraverso esempi e moduli disponibili in rete, poichè molti di essi utilizzano la versione 2.3, la cui conversione a 2.4, necessita di varie modifiche.
Il sistema di gestione delle dipendenze verso plugin e moduli esterni è gestito tramite SBT risulta comodo e pratico, tuttavia per quanto è stato possibile riscontrare fin'ora, non raggiunge i livelli di efficienza offerti da package managers equivalenti come NPM o RubyGems.

Amazon Web Services

Sono una collezione di servizi web che insieme creano una piattaforma di cloud computing. Le ragioni che ci hanno spinto a orientarci su tale servizio piuttosto che un competitor, ad esempio Heroku, sono le seguenti:

- **Flessibilità della piattaforma:** il gran numero di servizi offerti e la possibilità di “combinarli” liberamente fa di questa piattaforma uno strumento molto versatile che potremo sfruttare al massimo, tra cui Amazon Machine Learning che utilizzeremo per elaborare i dati e fornire previsione sugli stessi;
- **Esperienza del proponente:** Zero12 è partner di AWS e ci ha consigliato tale piattaforma vista la loro esperienza positiva nell'utilizzo in altri progetti;

Non sono stati riscontrati aspetti negativi degni di nota verso tale suite di servizi, se non l'inesperienza del gruppo verso gli stessi, ma tale situazione è estesa anche verso altri competitor con un offerta equivalente.

Node.js

Node.js è un *framework_G* per lo sviluppo di applicazioni web *server-side_G* che utilizza il motore javascript v8 di Google. Nel prodotto che si andrà a definire, Node.js verrà utilizzato per la realizzazione del *broker MQTT_G*, ovvero il sistema che si occuperà di gestire la ricezione dei dati da parte degli oggetti e di registrarli nella base di dati. La scelta di utilizzare la seguente tecnologia è dettata dalle seguenti osservazioni:

- **Efficienza:** in una situazione ideale, il sistema che verrà prodotto dovrà essere in grado di gestire un elevato traffico di rete (invio di dati da parte di moltissimi oggetti); pertanto l'efficienza deve essere sin da subito un punto di fondamentale importanza. Tale caratteristica è garantita dall'ambiente Node.js, il quale, basandosi essenzialmente su un paradigma di elaborazione dati in maniera asincrona, fa dell'efficienza il suo punto di forza maggiore;



- **Interesse del Gruppo:** medesime motivazioni che hanno spinto alla scelta di Scala.

Non sono stati riscontrati aspetti negativi degni di nota.

MongoDB

Un database orientato ai documenti, classificato come NoSQL, infatti al posto delle tradizionali tabelle MongoDB usa una struttura simile al formato JSON con una schemi dinamici, detti BSON. Abbiamo optato per tale database per i seguenti motivi:

- **NoSQL:** una struttura di questo tipo è molto più flessibile e semplice da modellare e progettare e risulta più semplice anche da scalare orizzontalmente, scenario più che plausibile vista la potenziale mole di dati che verranno salvati;
- **Ad hoc queries:** le queries sui dati memorizzati possono ritornare specifici campi dei documenti salvati e possono includere delle funzioni javascript che verranno mandate al database per essere eseguite, permettendo agli sviluppatori la scrittura di queries più “eleganti”;
- **Esperienza del proponente:** Zero12 è partner di MongoDB è ha pertanto consigliato al gruppo tale tecnologia che utilizza quotidianamente e con la quale ha avuto ottimi riscontri.

Aspetti **negativi** considerati:

- **NoSQL:** se come detto una struttura noSQL risulta più flessibile, è altrettanto vero che richiede una progettazione più attenta mirata al capire quando utilizzare relazioni verso altre collections, piuttosto che preferire una struttura embedded.

HTML5

Quinta versione del linguaggio di markup per il web, introduce nuovi elementi tag e semplifica la sintassi rispetto a xhtml, introduce nuove *api*_G come l’interfaccia drag and drop. Nonostante non sia ancora standard W3C abbiamo scelto di utilizzarlo poichè i vincoli rispetto ai browser lo consentono e questa ultima versione del markup permette l’implementazione di varie funzionalità e l’interazione con numerose librerie ed API.

SASS

Syntactically Awesome Stylesheets è un linguaggio di scripting interpretato nel comune CSS. La versione più recente di tale sintassi ricalca le principali regole sintattiche dei file css classici, ma porta dei vantaggi quali la possibilità di utilizzo di variabili, mixins, ereditarietà, annidamento, che rendono lo sviluppo del codice molto più modulabile e riutilizzabile. Va comunque detto che tale sintassi si fa apprezzare per lo più in progetti corposi, ma può risultare poco intuitiva per chi è alle prime armi.

Twitter Bootstrap

Twitter Bootstrap è uno dei framework più famosi per lo sviluppo di progetti Web. Esso contiene modelli di progettazione basati su HTML e CSS (con alcune estensioni opzionali che prendono come riferimento JavaScript), sia per la tipografia, che per le varie componenti dell’interfaccia, come moduli, bottoni e navigazione. Le motivazioni che hanno portato alla scelta dei suddetti strumenti sono le seguenti:

- **Supporto nativo al *responsive web design*_G:** tale caratteristica garantisce che il layout delle pagine web si regoli dinamicamente, tenendo conto delle caratteristiche del dispositivo utilizzato, sia esso desktop, tablet o telefono cellulare. Grazie a tale caratteristica si va a sgravare il lavoro del gruppo, evitando di concentrare troppo tempo per la realizzazione di un’interfaccia grafica completamente *responsive*, fornita direttamente dalla tecnologia in esame;



- **Documentazione e supporto della comunità.**

Non sono stati riscontrati aspetti negativi degni di nota per tale framework.

Javascript

Linguaggio di scripting più diffuso in ambito web, si integra con tutti i framework che verranno utilizzati ed è per tanto indispensabile.

Definizione del Prodotto

Metodo e formalismo di specifica

La piattaforma che verrà realizzata è composta da più moduli indipendenti tra loro che uniti insieme danno vita al progetto vero e proprio. A valle di questa considerazione, è possibile scomporre l'architettura del sistema nei seguenti moduli:

- Business Logic (core della piattaforma);
- Engine Predittivo API;
- Frond End applicativo;
- Authentication API;
- Broker MQTT;

Il progetto verrà sviluppato attraverso il framework Play!, il quale impone l'uso del pattern architetturale MVC, che per tanto influenza fortemente l'organizzazione e la progettazione architetturale dell'applicazione. Play! propone due strati principali, il *Presentation layer*, che si suddivide a sua volta in *Controller layer* e *View layer*, e il *Model layer*; la progettazione seguirà quindi tale schema integrando, dove possibile, moduli già disponibili e consigliati per interagire al meglio con Play! framework. La progettazione è stata sia del tipo top-down, per quanto riguarda l'organizzazione secondo modello MVC, in particolare riguardo a come adattare l'applicazione all'anatomia di Play!, ma anche del tipo bottom-up, poichè da subito abbiamo cercato moduli e API da riutilizzare e integrare all'interno del progetto.

Architettura generale

Come detto l'architettura segue il modello imposto da Play! rappresentato dal seguente diagramma:

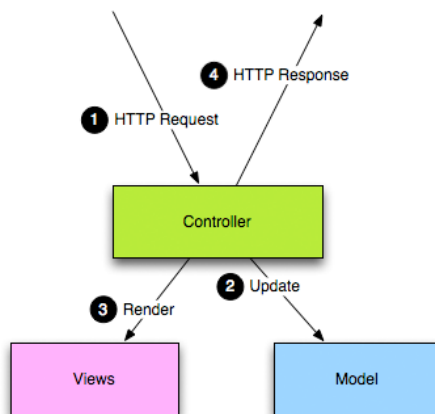


Figura 1: Play! framework MVC model.

Play! è inoltre completamente stateless e orientato unicamente a un approccio del tipo richiesta-risposta HTTP, il ciclo di vita tipico è quindi composto da una serie di richieste HTTP che seguono questo schema:

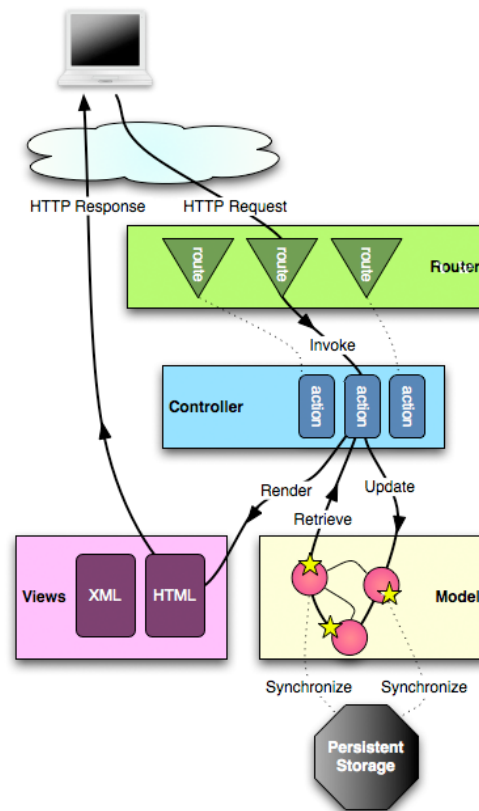


Figura 2: Ciclo di vita Play! framework.

Alla luce di tali premesse, l'architettura che ne deriva è la seguente:

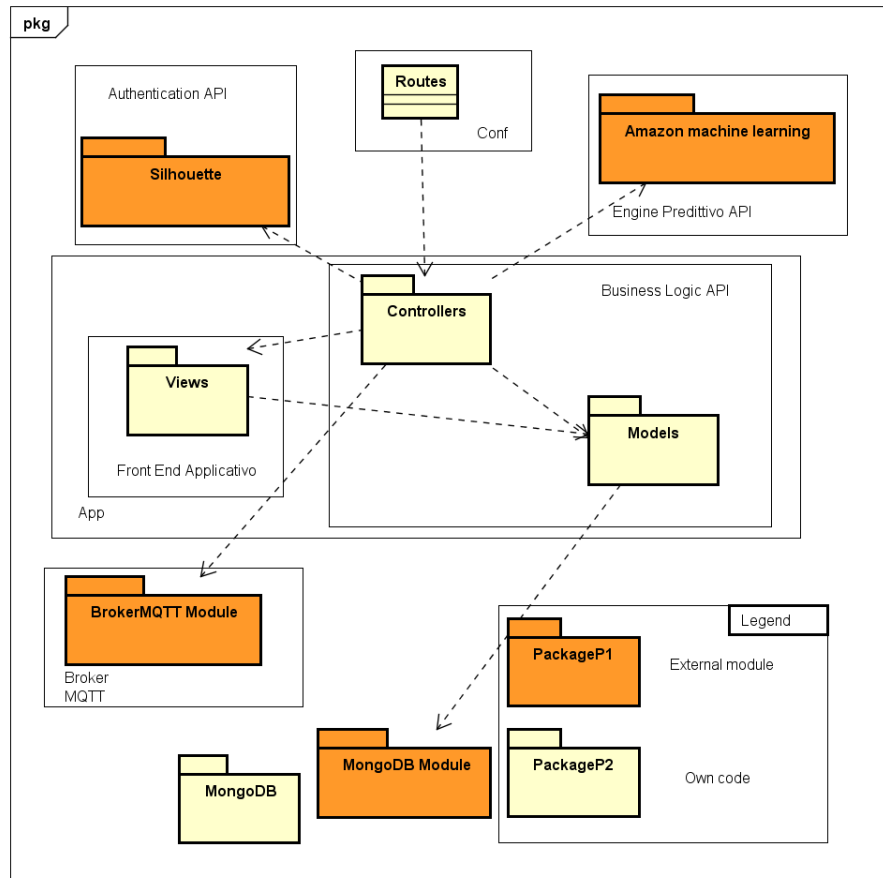


Figura 3: Architettura Generale.

Di seguito vi è una breve spiegazione dei vari package di cui ci compone il diagramma precedente.

- **Business Logic**

- *Controllers*: un package che contiene tutte le azioni da intraprendere al verificarsi di un determinato evento, o meglio in risposta a una richiesta HTTP, faranno uso dell' **Authentication API** e dell'**Engine predittivo** e ovviamente si occuperanno dell'interazione tra *Views* e *Model*, sfruttando il modulo *Reactive Mongo* per l'interazione con la base di dati;
- *Models*: contiene le classi che modellano le varie strutture dati, le operazioni su di essi e le funzionalità relative alla base di dati, che si appoggiano anch'esse al modulo *Reactive Mongo*.

- **Engine Predittivo**: è il modulo che si occupa di fare predizioni e elaborazione dei dati raccolti, utilizza il servizio di AWS *Amazon Machine Learning*;
- **Front End applicativo**: rappresentato di fatto dalla componente *Views* del pattern MVC di Play!, tale package contiene infatti tutti i template html utili a creare un'interfaccia grafica che faciliti l'interazione con la piattaforma;
- **Authentication API**: il modulo che gestisce il meccanismo dell'autenticazione, creazione e distruzione delle sessioni, si appoggia al modulo *play2-auth*;
- **Broker MQTT**: rappresenta il mediatore che gestisce lo scambio di informazione tra gli oggetti fisici in uso e la piattaforma stessa, è implementato tramite il servizio di AWS;



- **Routes:** è il file che si occupa di mappare l'esecuzione di un determinato metodo di un dato controller a una corrispondente richiesta HTTP.



Descrizione Componenti e Classi

In tale sezione verranno illustrate le principali caratteristiche di tutte le classi e di tutti i moduli esterni presenti nel progetto UMAP. Per convenzione, le classi verranno elencate precedute dalla locazione fisica (package) in cui sono contenute, in modo da fornire un rapido strumento per reperirle.

Moduli Esterni

Il progetto UMAP prevede di utilizzare i seguenti moduli/librerie esterni:

- **Amazon Machine Learning (AWS SDK);**
- **Silhouette;**
- **MongoDB Module;**
- **BrokerMQTT Module.**

Tale scelta è stata effettuata per alleggerire il lavoro nelle attività di Progettazione e di Codifica, in modo da poter rilasciare il prodotto finale il prima possibile. Inoltre, si è rilevato che i suddetti moduli svolgano i propri compiti nella maniera ideale per gli obiettivi prefissati nel progetto didattico del gruppo di lavoro LaTeXeBiscotti.

Amazon Machine Learning (AWS SDK)

4.1.1.1 Descrizione

Tale tecnologia fa parte della collezione di servizi di *cloud computing_G* che compongono la piattaforma “on demand” *Amazon Web Services_G* offerta dall’azienda Amazon. Tale modulo permette di creare modelli di *machine learning* senza aver bisogno di conoscere la complessità e le tecnologie che stanno alla base di tale modulo. Una volta creati i suddetti modelli, Amazon Machine Learning permette di ottenere facilmente delle predizioni usando delle semplici *API_G*, con il vantaggio notevole di non dover implementare codice e di non dover gestire l’infrastruttura di base per concepirle.

Per una spiegazione più dettagliata del modulo si rimanda a:

<https://aws.amazon.com/it/machine-learning/>

4.1.1.2 Utilizzo

Tale modulo è utilizzato dal package **controllers**, in particolare dalle classi che hanno il compito della gestione dei dati e delle regole predittive; una volta inviati i dati al modulo, questo ritornerà un pacchetto di previsioni, correlato da grafici significativi che potranno essere utilizzati dagli utenti autorizzati. I suddetti controller avranno il compito di importare al loro interno, e quindi utilizzare i seguenti package:

- `com.amazonaws.services.machinelearning.AmazonMachineLearningClient;`
- `com.amazonaws.services.machinelearning.model._;`
- `com.amazonaws.services.machinelearning.AmazonMachineLearningClient;`
- `com.amazonaws.services.machinelearning.model.{GetMLModelRequest, PredictRequest}.`

Silhouette

4.1.2.1 Descrizione

In tale modulo viene gestita tutta la logica specifica per le operazioni di autenticazione e di riconoscimento del ruolo dell’utente autenticato. In generale tale modulo fornisce all’utente un *token* per tutta la durata



della sessione all'interno della piattaforma, tale *token* sarà utile anche per il riconoscimento del suo ruolo all'interno della stessa.

4.1.2.2 Utilizzo

Tale modulo è utilizzato dal package **controllers**, in particolare dalle classi che hanno il compito di gestire le operazioni di, ad esempio, login o di logout, e dalle classi che devono indirizzare l'utente a views differenti in base al suo ruolo all'interno della piattaforma.

MongoDB Module

4.1.3.1 Descrizione

Tale modulo si occupa di interfacciarsi nella maniera più efficace ed efficiente possibile con la base di dati.

4.1.3.2 Utilizzo

Tale modulo è utilizzato dai packages **controllers** e **models**, in particolare nelle operazioni di lettura e scrittura effettuate dalla varie classi sulla base di dati.

BrokerMQTT Module

4.1.4.1 Descrizione

Il suddetto modulo si fare da mediatore tra i vari messaggi provenienti dai vari *things* collegati con la piattaforma e viceversa.

4.1.4.2 Utilizzo

Il modulo viene utilizzato dal package **contollers**, il quale si occupa poi, nelle classi abilitate al compito, di inserire correttamente nella base di dati le informazioni ricevute dai *things* o di inviare update o comandi agli stessi.

Models

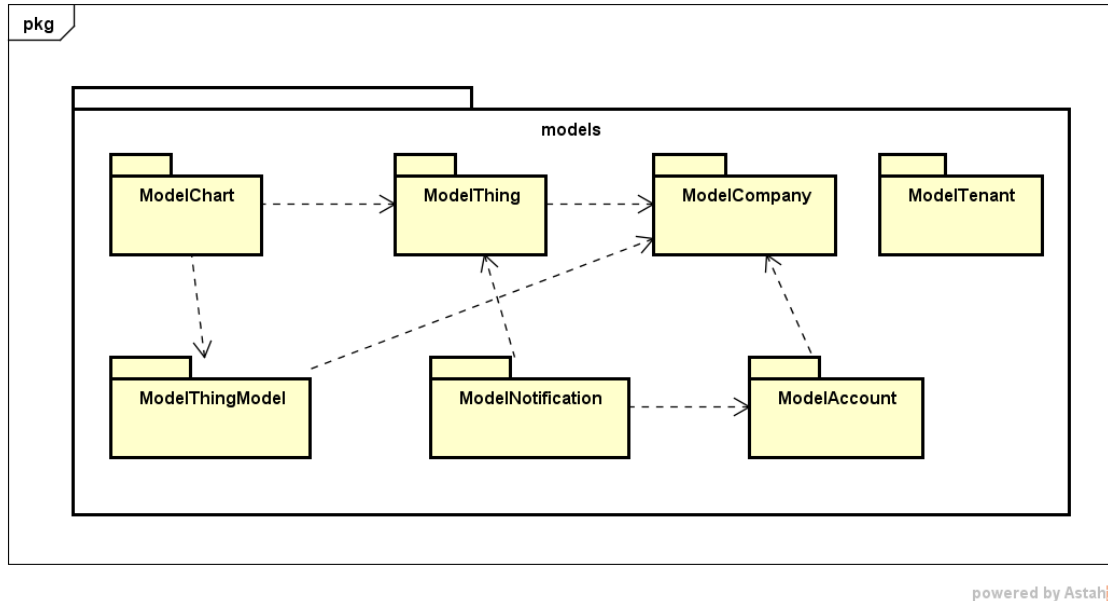


Figura 4: Diagramma package - Visione generale package Models

app :: models :: User

4.2.1.1 Descrizione

Classe contenente tutte le informazioni inerenti ad un qualsiasi utente inserito nella piattaforma. Tali informazioni riguardano essenzialmente, il codice univoco, l'anagrafica dell'utente con la sua email di riferimento e la company alla quale egli appartiene.

4.2.1.2 Utilizzo

Tale classe viene utilizzata per modellare nel database tutti i campi dati riguardanti un account. Essa riceve richieste di modifica, aggiunta, eliminazione e restituzione dei dati da parte del controller.

4.2.1.3 Relazioni con altre classi

- **app :: models :: Company:** relazione entrante, riferimento alla company a cui appartiene un particolare account;
- **app :: controllers :: superAdmin :: SuperAdminController;**
- **app :: controllers :: superAdmin :: UserController;**
- **app :: controllers :: admin :: AdminController;**
- **app :: controllers :: admin :: UserController;**
- **app :: controllers :: user :: UserController;**
- **app :: controllers :: shared :: authentication :: AuthenticationController;**
- **app :: controllers :: shared :: account :: AccountController;**
- **app :: controllers :: shared :: account :: PasswordRecoverController;**



- `app :: controllers :: shared :: account :: PasswordResetController`.

`app :: models :: Company`

4.2.2.1 Descrizione

Classe contenente tutte le informazioni inerenti ad una company inserita nella piattaforma. Per ogni company sono di interesse, ad esempio, il nome, la ragione sociale e l'indirizzo di ubicazione.

4.2.2.2 Utilizzo

Tale classe viene utilizzata per modellare nel database tutti i campi dati riguardanti una company. Essa riceve richieste di modifica, aggiunta, eliminazione e restituzione dei dati da parte del controller.

4.2.2.3 Relazioni con altre classi

- `app :: controllers :: superAdmin :: CompanyController`;

`app :: models :: Thing`

4.2.3.1 Descrizione

Classe contenente tutte le informazioni inerenti ad un oggetto (o *thing*) inserito nella piattaforma. Per ogni *thing* importano, ad esempio, il codice univoco, il serial number, la data dell'ultimo aggiornamento, la password utile per inviare dati al database, il suo stato (se in funzione o meno) e la company a cui esso appartiene.

4.2.3.2 Utilizzo

Tale classe viene utilizzata per modellare nel database tutti i campi dati riguardanti un *thing*. Essa riceve richieste di modifica, aggiunta, eliminazione (da un utente *Super Admin_G*) e restituzione (da un utente *Super Admin_G* o *Admin_G*) dei dati da parte del controller.

4.2.3.3 Relazioni con altre classi

- `app :: models :: Company`: relazione entrante, riferimento alla company a cui appartiene un particolare oggetto;
- `app :: controllers :: superAdmin :: ThingController`;
- `app :: controllers :: admin :: ThingController`;
- `app :: controllers :: admin :: ChartController`;
- `app :: controllers :: admin :: ThingController`;
- `app :: controllers :: user :: ThingController`;
- `app :: controllers :: shared :: administrator :: EngineController`.

`app :: models :: ThingType`

4.2.4.1 Descrizione

Classe contenente tutte le informazioni inerenti ai modelli ovvero le tipologie di *things* inseriti nella piattaforma. Per ogni modello importano, ad esempio, il nome, la marca, il codice univoco del modello, la company da cui il modello in questione è stato creato e la lista di Company che lo utilizzano.



4.2.4.2 Utilizzo

Tale classe viene utilizzata per modellare nel database tutti i campi dati riguardanti un modello di *things*. Essa riceve richieste di modifica, aggiunta, eliminazione e restituzione dei dati da parte del controller.

4.2.4.3 Relazioni con altre classi

- **app :: models :: Company:** relazione entrante, riferimento alla company a cui appartiene un particolare modello;
- **app :: controllers :: superAdmin :: ThingController;**
- **app :: controllers :: admin :: ThingTypeController;**
- **app :: controllers :: admin :: ChartController;**
- **app :: controllers :: user :: ThingTypeController;**
- **app :: controllers :: shared :: administrator :: EngineController.**

app :: models :: Notification

4.2.5.1 Descrizione

Classe contenente tutte le informazioni inerenti ad una notifica inserita da un utente $user_G$ all'interno della piattaforma. Tali informazioni riguardano essenzialmente il suo codice univoco, una descrizione della notifica, l'oggetto o modello scatenante, l'indirizzo email dell'utente a cui inviare la notifica e il tipo di valore su cui inserire valori minimi e massimi, che verranno controllati e utilizzati come soglia per un possibile invio di notifiche.

4.2.5.2 Utilizzo

Tale classe viene utilizzata per modellare nel database tutti i campi dati riguardanti una notifica. Essa riceve richieste di modifica, aggiunta, eliminazione e restituzione dei dati da parte del controller, e notifica alla vista quando gli stessi sono disponibili o aggiornati.

4.2.5.3 Relazioni con altre classi

- **app :: models :: User:** relazione entrante, recupero dell'indirizzo email a cui inviare la notifica;
- **app :: models :: Thing:** relazione entrante, recupero dei campi dati di un *thing* da cui far scaturire la notifica (se ad esempio tali dati superano un particolare limite);
- **app :: models :: ThingType:** relazione entrante, recupero dei campi dati di un modello da cui far scaturire la notifica (se ad esempio tali dati superano un particolare limite);
- **app :: controllers :: shared :: adminUser :: NotificationController;**

app :: models :: Charts

4.2.6.1 Descrizione

Tale classe viene utilizzata per modellare i grafici utilizzati poi per la rappresentazione delle analisi predittive. Tali informazioni riguardano essenzialmente il codice univoco del grafico, il suo nome e la lista di oggetti o modelli che lo utilizzano.



4.2.6.2 Utilizzo

Verrà utilizzato per fornire le informazioni necessarie alle classi che si occupano di visualizzare i modelli di analisi e relativi grafici attualmente salvati nella base di dati.

4.2.6.3 Relazioni con altre classi

- **app :: models :: Thing**: relazione entrante, riferimento alla classe che rappresenta un oggetto;
- **app :: models :: ThingType**: relazione entrante, riferimento alla classe che rappresenta un modello d'oggetto;
- **app :: controllers :: admin :: ThingController**;
- **app :: controllers :: admin :: ThingTypeController**;
- **app :: controllers :: admin :: ChartController**;
- **app :: controllers :: user :: ThingController**;
- **app :: controllers :: user :: ThingTypeController**;
- **app :: controllers :: shared :: administrator :: EngineController**.

app :: models :: Roles

4.2.7.1 Descrizione

Tale classe viene utilizzata per gestire l'autorizzazione dei vari tipi d'utente durante la navigazione nel sito, permettendo loro la visualizzazione solo delle zone di loro competenza.

4.2.7.2 Utilizzo

Verrà utilizzato per verificare il tipo d'utente dopo una richiesta di visualizzazione di una pagina dell'applicativo e in base ad esso sarà possibile visualizzarla o altrimenti verrà indirizzato alla pagina di competenza.

4.2.7.3 Relazioni con altre classi

- **app :: models :: User**: relazione entrante, riferimento alla classe che rappresenta i vari utenti all'interno della piattaforma;

app :: models :: daos :: chart :: ChartDAO

4.2.8.1 Descrizione

Tale classe viene utilizzata per descrizione delle varie operazioni effettuabili nel database per quanto riguarda la classe Chart, come le ricerche o il salvataggio di un nuovo grafico.

4.2.8.2 Utilizzo

Verrà utilizzato per la gestione del database per quanto riguarda le operazioni sui grafici.



4.2.8.3 Relazioni con altre classi

- **app :: models :: Chart**: relazione entrante, riferimento alla classe che rappresenta il grafico;
- **app :: controllers :: admin :: ThingController**;
- **app :: controllers :: admin :: ThingTypeController**;
- **app :: controllers :: admin :: ChartController**;
- **app :: controllers :: user :: ThingController**;
- **app :: controllers :: user :: ThingTypeController**;
- **app :: controllers :: shared :: administrator :: EngineController**.

app :: models :: daos :: chart :: ChartDAOImpl

4.2.9.1 Descrizione

Tale classe viene utilizzata per realizzare l'implementazione dei metodi descritti nella classe ChartDAO riguardanti le varie operazioni effettuabili nel database per quanto riguarda la classe Chart, come le ricerche o il salvataggio di un nuovo grafico.

4.2.9.2 Utilizzo

Verrà utilizzato per l'implementazione dei metodi utilizzati dal controller per la gestione del database per quanto riguarda le operazioni sui grafici.

4.2.9.3 Relazioni con altre classi

- **app :: models :: Chart**: relazione entrante, riferimento alla classe che rappresenta il grafico;
- **app :: models :: daos :: ChartDAO**.

app :: models :: daos :: company :: CompanyDAO

4.2.10.1 Descrizione

Tale classe viene utilizzata per descrizione delle varie operazioni effettuabili nel database per quanto riguarda la classe Company, come le ricerche o il salvataggio di una nuova company.

4.2.10.2 Utilizzo

Verrà utilizzato per la gestione del database per quanto riguarda le operazioni sulle company.

4.2.10.3 Relazioni con altre classi

- **app :: models :: Company**: relazione entrante, riferimento alla classe che rappresenta la company;
- **app :: controllers :: superAdmin :: CompanyController**.



app :: models :: daos :: company :: CompanyDAOImpl

4.2.11.1 Descrizione

Tale classe viene utilizzata per realizzare l'implementazione dei metodi descritti nella classe CompanyDAO riguardanti le varie operazioni effettuabili nel database per quanto riguarda la classe Company, come le ricerche o il salvataggio di una nuova company.

4.2.11.2 Utilizzo

Verrà utilizzato per l'implementazione dei metodi utilizzati dal controller per la gestione del database per quanto riguarda le operazioni sulle company.

4.2.11.3 Relazioni con altre classi

- **app :: models :: Company**: relazione entrante, riferimento alla classe che rappresenta la company;
- **app :: models :: daos :: CompanyDAO**.

app :: models :: daos :: thing :: ThingDAO

4.2.12.1 Descrizione

Tale classe viene utilizzata per descrizione delle varie operazioni effettuabili nel database per quanto riguarda la classe Thing, come le ricerche o il salvataggio di un nuovo oggetto.

4.2.12.2 Utilizzo

Verrà utilizzato per la gestione del database per quanto riguarda le operazioni sugli oggetti.

4.2.12.3 Relazioni con altre classi

- **app :: models :: Thing**: relazione entrante, riferimento alla classe che rappresenta l'oggetto;
- **app :: controllers :: superAdmin :: ThingController**;
- **app :: controllers :: admin :: ThingController**;
- **app :: controllers :: admin :: ChartController**;
- **app :: controllers :: user :: ThingController**;
- **app :: controllers :: shared :: administrator :: EngineController**.

app :: models :: daos :: thing :: ThingDAOImpl

4.2.13.1 Descrizione

Tale classe viene utilizzata per realizzare l'implementazione dei metodi descritti nella classe ThingDAO riguardanti le varie operazioni effettuabili nel database per quanto riguarda la classe thing, come le ricerche o il salvataggio di un nuovo oggetto.

4.2.13.2 Utilizzo

Verrà utilizzato per l'implementazione dei metodi utilizzati dal controller per la gestione del database per quanto riguarda le operazioni sugli oggetti.



4.2.13.3 Relazioni con altre classi

- **app :: models :: Thing**: relazione entrante, riferimento alla classe che rappresenta l'oggetto;
- **app :: models :: daos :: ThingDAO**.

app :: models :: daos :: thingType :: ThingTypeDAO

4.2.14.1 Descrizione

Tale classe viene utilizzata per descrizione delle varie operazioni effettuabili nel database per quanto riguarda la classe ThingType, come le ricerche o il salvataggio di un nuovo modello.

4.2.14.2 Utilizzo

Verrà utilizzato per la gestione del database per quanto riguarda le operazioni sui modelli.

4.2.14.3 Relazioni con altre classi

- **app :: models :: ThingType**: relazione entrante, riferimento alla classe che rappresenta il modello d'oggetto;
- **app :: controllers :: superAdmin :: ThingController**;
- **app :: controllers :: admin :: ThingTypeController**;
- **app :: controllers :: admin :: ChartController**;
- **app :: controllers :: user :: ThingTypeController**;
- **app :: controllers :: shared :: administrator :: EngineController**.

app :: models :: daos :: thingtype :: ThingTypeDAOImpl

4.2.15.1 Descrizione

Tale classe viene utilizzata per realizzare l'implementazione dei metodi descritti nella classe ThingTypeDAO riguardanti le varie operazioni effettuabili nel database per quanto riguarda la classe thingType, come le ricerche o il salvataggio di un nuovo modello.

4.2.15.2 Utilizzo

Verrà utilizzato per l'implementazione dei metodi utilizzati dal controller per la gestione del database per quanto riguarda le operazioni sui modelli.

4.2.15.3 Relazioni con altre classi

- **app :: models :: ThingType**: relazione entrante, riferimento alla classe che rappresenta il modello d'oggetto;
- **app :: models :: daos :: ThingTypeDAO**.



app :: models :: daos :: user :: UserDao

4.2.16.1 Descrizione

Tale classe viene utilizzata per descrizione delle varie operazioni effettuabili nel database per quanto riguarda la classe User, come le ricerche o il salvataggio di un nuovo user.

4.2.16.2 Utilizzo

Verrà utilizzato per la gestione del database per quanto riguarda le operazioni sugli user.

4.2.16.3 Relazioni con altre classi

- **app :: models :: User:** relazione entrante, riferimento alla classe che rappresenta lo user;
- **app :: controllers :: superAdmin :: UserController;**
- **app :: controllers :: admin :: UserController;**
- **app :: controllers :: shared :: authentication :: AuthenticationController;**
- **app :: controllers :: shared :: account :: AccountController;**
- **app :: controllers :: shared :: account :: PasswordRecoverdController;**
- **app :: controllers :: shared :: account :: PasswordResetController.**

app :: models :: daos :: user :: UserDaoImpl

4.2.17.1 Descrizione

Tale classe viene utilizzata per realizzare l'implementazione dei metodi descritti nella classe UserDao riguardanti le varie operazioni effettuabili nel database per quanto riguarda la classe user, come le ricerche o il salvataggio di un nuovo user.

4.2.17.2 Utilizzo

Verrà utilizzato per l'implementazione dei metodi utilizzati dal controller per la gestione del database per quanto riguarda le operazioni sugli user.

4.2.17.3 Relazioni con altre classi

- **app :: models :: User:** relazione entrante, riferimento alla classe che rappresenta lo user;
- **app :: models :: daos :: UserDao.**

app :: models :: daos :: services :: user :: UserService

4.2.18.1 Descrizione

Tale classe viene utilizzata per la definizioni di metodi che servono alla libreria Silhouette per l'identificazione degli oggetti di tipo user e per il suo salvataggio.

4.2.18.2 Utilizzo

Verrà utilizzato per la gestione del database per quanto riguarda le operazioni sugli oggetti.

4.2.18.3 Relazioni con altre classi

- `app :: models :: User`: relazione entrante, riferimento alla classe che rappresenta lo user;

`app :: models :: daos :: services :: user :: UserServiceImpl`

4.2.19.1 Descrizione

Tale classe viene utilizzata per l'implementazione dei metodi descritti nella classe `UserService` utili all'identificazione degli oggetti di tipo `user` e per il suo salvataggio.

4.2.19.2 Utilizzo

Verrà utilizzato per la gestione del database per quanto riguarda le operazioni sugli oggetti.

4.2.19.3 Relazioni con altre classi

- `app :: models :: User`: relazione entrante, riferimento alla classe che rappresenta lo user;
- `app :: models :: daos :: services :: user :: UserService`;

Controllers

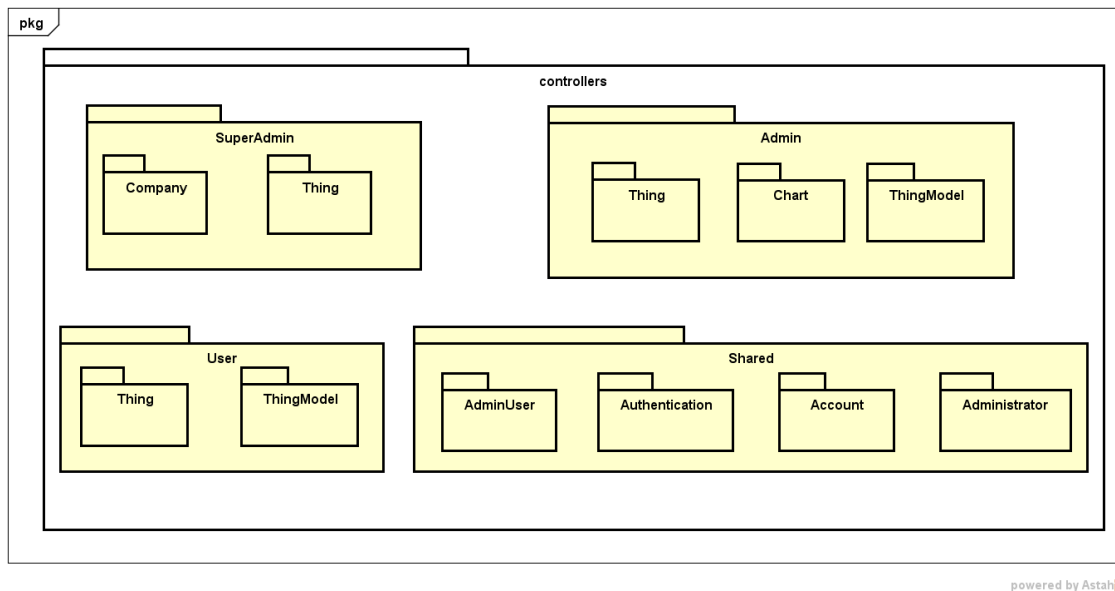


Figura 5: Diagramma package - Visione generale package Controllers

`app :: controllers :: superAdmin :: SuperAdminController`

4.3.1.1 Descrizione

Classe contenente un metodo che renderizza la pagina di home di un `SuperAdmin`.



4.3.1.2 Utilizzo

Tale classe viene utilizzata come classe di partenza per gestire le varie funzionalità offerte al SuperAdmin.

4.3.1.3 Relazioni con altre classi

- **app :: models :: User**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: views :: superAdmin :: home**: relazione entrante, tale classe contiene l'interfaccia grafica relativa alla home di un SuperAdmin.

app :: controllers :: superAdmin :: UserController

4.3.2.1 Descrizione

Classe contenente tutti i comandi atti a cambiare lo stato del model rappresentante gli utenti iscritti alla piattaforma.

4.3.2.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a cambiare lo stato del model rappresentante gli utenti iscritti alla piattaforma. Tali comandi sono, ad esempio, l'aggiunta e la modifica dell'account di un utente.

4.3.2.3 Relazioni con altre classi

- **app :: models :: User**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: models :: daos :: user :: UserDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe User;
- **app :: views :: superAdmin :: user :: editUser**: relazione entrante, tale classe viene usata per mostrare la form di modifica e lanciare la richiesta di edit di una dato User;
- **app :: views :: superAdmin :: user :: users**: relazione entrante, tale classe viene usata per mostrare la lista di tutti gli Users con relativa interfaccia per lanciare le classiche operazioni CRUD;
- **app :: views :: superAdmin :: user :: addUser**: relazione entrante, tale classe viene usata per mostrare la form di aggiunta di un nuovo User.

app :: controllers :: superAdmin :: CompanyController

4.3.3.1 Descrizione

Classe contenente tutti i comandi atti a cambiare lo stato del model rappresentante le company.

4.3.3.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a cambiare lo stato del model rappresentante le company. Tali comandi sono, ad esempio, l'aggiunta e la modifica di una company.



4.3.3.3 Relazioni con altre classi

- **app :: models :: Company:** relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: models :: daos :: company :: CompanyDAO:** relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Company;
- **app :: views :: superAdmin :: company :: editCompany:** relazione entrante, tale classe viene usata per mostrare la form di modifica e lanciare la richiesta di edit di una data company;
- **app :: views :: superAdmin :: company :: companies:** relazione entrante, tale classe viene usata per mostrare la lista di tutte le company con relativa interfaccia per lanciare le classiche operazioni CRUD;
- **app :: views :: superAdmin :: company :: addCompany:** relazione entrante, tale classe viene usata per mostrare la form di aggiunta di una nuova company.

app :: controllers :: superAdmin :: ThingController

4.3.4.1 Descrizione

Classe contenente tutti i comandi atti a cambiare lo stato del model rappresentante i *things*.

4.3.4.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a cambiare lo stato del model rappresentante i *things*. Tali comandi sono, ad esempio, l'aggiunta e la modifica di un *things*.

4.3.4.3 Relazioni con altre classi

- **app :: models :: Thing:** relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: models :: ThingType:** relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: models :: daos :: thing :: ThingDAO:** relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Thing;
- **app :: models :: daos :: thingType :: ThingTypeDAO:** relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe ThingType;
- **app :: views :: superAdmin :: thing :: editThing:** relazione entrante, tale classe viene usata per mostrare la form di modifica e lanciare la richiesta di edit di un thing;
- **app :: views :: superAdmin :: thing :: things:** relazione entrante, tale classe viene usata per mostrare la lista di tutte i things con relativa interfaccia per lanciare le classiche operazioni CRUD;
- **app :: views :: superAdmin :: thing :: addThing:** relazione entrante, tale classe viene usata per mostrare la form di aggiunta di un nuovo thing.
- **app :: views :: superAdmin :: thingType :: editThingType:** relazione entrante, tale classe viene usata per mostrare la form di modifica e lanciare la richiesta di edit di un thingType;
- **app :: views :: superAdmin :: thingType :: thingTypes:** relazione entrante, tale classe viene usata per mostrare la lista di tutte i thingsType con relativa interfaccia per lanciare le classiche operazioni CRUD;



- **app :: views :: superAdmin :: thingType :: addThingType:** relazione entrante, tale classe viene usata per mostrare la form di aggiunta di un nuovo thingType.

app :: controllers :: admin :: AdminController

4.3.5.1 Descrizione

Classe contenente un metodo che renderizza la pagina di home di un Admin.

4.3.5.2 Utilizzo

Tale classe viene utilizzata come classe di partenza per gestire le varie funzionalità offerte all'Admin.

4.3.5.3 Relazioni con altre classi

- **app :: models :: User:** relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: views :: admin :: home:** relazione entrante, tale classe contiene l'interfaccia grafica relativa alla home di un Admin.

app :: controllers :: admin :: UserController

4.3.6.1 Descrizione

Classe contenente tutti i comandi atti a cambiare lo stato del model rappresentante gli utenti iscritti alla piattaforma.

4.3.6.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a cambiare lo stato del model rappresentante gli utenti iscritti alla piattaforma per la propria company. Tali comandi sono, ad esempio, l'aggiunta e la modifica dell'account di un utente.

4.3.6.3 Relazioni con altre classi

- **app :: models :: User:** relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: models :: daos :: user :: UserDAO:** relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe User;
- **app :: views :: admin :: user :: editUser:** relazione entrante, tale classe viene usata per mostrare la form di modifica e lanciare la richiesta di edit di una dato User;
- **app :: views :: admin :: user :: users:** relazione entrante, tale classe viene usata per mostrare la lista di tutti gli Users con relativa interfaccia per lanciare le classiche operazioni CRUD;
- **app :: views :: admin :: user :: addUser:** relazione entrante, tale classe viene usata per mostrare la form di aggiunta di un nuovo User.



app :: controllers :: admin :: ThingController

4.3.7.1 Descrizione

Classe contenente tutti i comandi atti a cambiare lo stato del model rappresentante i *things*, in particolare per quanto riguarda l'aggiornamento del *firmware_G* del *thing* stesso, o l'invio di comandi.

4.3.7.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a cambiare lo stato del model rappresentante i *things*, in particolare per quanto riguarda l'aggiornamento del *firmware_G* del *thing* stesso, o l'invio di comandi.

4.3.7.3 Relazioni con altre classi

- **app :: models :: Thing**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne o leggere lo stato;
- **app :: models :: daos :: thing :: ThingDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Thing;
- **app :: models :: Chart**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne o leggere lo stato;
- **app :: models :: daos :: chart :: ChartDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Chart;
- **app :: views :: shared :: adminUser :: things**: relazione entrante, tale classe visualizza la lista di Things in uso dalla propria company;
- **app :: views :: admin :: thing :: thingDetails**: relazione entrante, verso tale classe vengono notificate le informazioni relative allo stato del Thing, ai grafici che lo interessano e all'invio di comandi e aggiornamento firmware degli oggetti dello stesso tipo.

app :: controllers :: admin :: ThingTypeController

4.3.8.1 Descrizione Classe contenente tutti i comandi atti a cambiare lo stato del model rappresentante il tipo dei *things*, in particolare per quanto riguarda l'aggiornamento del *firmware_G* del *thing* stesso.

4.3.8.2 Utilizzo Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a cambiare lo stato del model rappresentante le tipologie dei *things*, in particolare per quanto riguarda l'aggiornamento del *firmware_G* del tipo stesso.

4.3.8.3 Relazioni con altre classi

- **app :: models :: ThingType**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a leggerne lo stato;
- **app :: models :: daos :: thingType :: ThingTypeDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe ThingType;
- **app :: models :: Chart**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne o leggere lo stato;
- **app :: models :: daos :: chart :: ChartDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Chart;



- **app :: views :: shared :: adminUser :: thingTypes**: relazione entrante, verso tale classe vengono notificate le informazioni sulle tipologie dei things in uso;
- **app :: views :: admin :: thingType :: thingTypeDetails**: relazione entrante, verso tale classe vengono notificate le informazioni relative allo stato del model, ai grafici che lo interessano e all'invio di comandi e aggiornamento firmware della tipologia di oggetto selezionato.

app :: controllers :: admin :: ChartController

4.3.9.1 Descrizione

Classe contenente tutti i comandi atti a creare un nuovo modello di analisi, predittiva o meno, combinando tipi di dati input, funzioni e grafici da visualizzare.

4.3.9.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller per costruire un modello di analisi, predittiva o meno, sulla base della configurazione di parametri scelti dall'utente tramite interfaccia grafica.

4.3.9.3 Relazioni con altre classi

- **app :: models :: Thing**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a leggerne lo stato;
- **app :: models :: daos :: thing :: ThingDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Thing;
- **app :: models :: ThingType**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a leggerne lo stato;
- **app :: models :: daos :: thingType :: ThingTypeDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe ThingType;
- **app :: models :: Chart**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a leggerne lo stato;
- **app :: models :: daos :: chart :: ChartDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Chart;
- **app :: views :: admin :: chart :: newChart**: relazione entrante, verso tale classe vengono notificate le informazioni per permettere la scelta dei parametri con cui configurare il modello di analisi.

app :: controllers :: user :: UserController

4.3.10.1 Descrizione

Classe contenente un metodo che renderizza la pagina di home di uno User.

4.3.10.2 Utilizzo

Tale classe viene utilizzata come classe di partenza per gestire le varie funzionalità offerte dello User.

4.3.10.3 Relazioni con altre classi

- **app :: models :: User**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;



- **app :: views :: user :: home:** relazione entrante, tale classe contiene l'interfaccia grafica relativa alla home di uno User.

app :: controllers :: user :: ThingController

4.3.11.1 Descrizione

Classe contenente tutti i comandi atti a leggere lo stato del model rappresentante i *things* appartenenti ad una particolare *Company_G*.

4.3.11.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a leggere lo stato del model rappresentante i *things* appartenenti ad una particolare *Company_G*.

4.3.11.3 Relazioni con altre classi

- **app :: models :: Thing:** relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a leggerne lo stato;
- **app :: models :: daos :: thing :: ThingDAO:** relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Thing;
- **app :: models :: Chart:** relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne o leggere lo stato;
- **app :: models :: daos :: chart :: ChartDAO:** relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Chart;
- **app :: views :: shared :: adminUser :: things:** relazione entrante, verso tale classe saranno notificate le informazioni sui vari things della company in questione;
- **app :: views :: user :: thing :: thingDetails:** relazione entrante, verso tale classe saranno notificate le informazioni di uno specifico oggetto appartenente della company in questione.

app :: controllers :: user :: ThingTypeController

4.3.12.1 Descrizione

Classe contenente tutti i comandi atti a leggere lo stato del model rappresentante le tipologie dei *things* appartenenti ad una particolare *company_G*.

4.3.12.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a leggere lo stato del model rappresentante le tipologie dei *things* appartenenti ad una particolare *company_G*.

4.3.12.3 Relazioni con altre classi

- **app :: models :: ThingType:** relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a leggerne lo stato;
- **app :: models :: daos :: thingType :: ThingTypeDAO:** relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe ThingType;



- **app :: models :: Chart:** relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne o leggere lo stato;
- **app :: models :: daos :: chart :: ChartDAO:** relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Chart;
- **app :: views :: shared :: adminUser :: thingTypes:** relazione entrante, verso tale classe saranno notificate le informazioni sulle varie tipologie di oggetti appartenenti alla company in questione;
- **app :: views :: user :: thingType :: thingTypesDetails:** relazione entrante, verso tale classe saranno notificate le informazioni su uno specifico modello di oggetto, così come i grafici che lo interessano, appartenente alla company in questione.

app :: controllers :: shared :: adminUser :: NotificationController

4.3.13.1 Descrizione

Classe contenente tutti i comandi atti a modificare lo stato del model rappresentante le notifiche riguardanti un particolare *thing* appartenente alla company di riferimento.

4.3.13.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a leggere lo stato del model rappresentante le notifiche riguardanti un particolare *thing* appartenente alla company di riferimento.

4.3.13.3 Relazioni con altre classi

- **app :: models :: Notification:** relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: models :: daos :: notification :: NotificationDAO:** relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Notification;
- **app :: views :: shared :: adminUser :: notification :: notification:** relazione entrante, verso tale classe vengono notificate le informazioni relative alle impostazioni di notifica salvate e in uso.
- **app :: views :: shared :: adminUser :: notification :: addNotification:** relazione entrante, tale classe renderizza l'interfaccia grafica per gestire l'aggiunta di una nuova notifica;
- **app :: views :: shared :: adminUser :: notification :: editNotification:** relazione entrante, tale classe renderizza l'interfaccia grafica per gestire la modifica di una nuova notifica esistente.

app :: controllers :: shared :: authentication :: AuthenticationController

4.3.14.1 Descrizione

Classe contenente tutti i comandi atti a gestire le operazioni di autenticazione di un particolare utente alla piattaforma.

4.3.14.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a gestire le operazioni di autenticazione di un particolare utente alla piattaforma.



4.3.14.3 Relazioni con altre classi

- **app :: models :: User**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a verificarne lo stato (utente presente nella piattaforma e con le credenziali corrette per l'accesso o meno);
- **app :: models :: daos :: user :: UserDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe User;
- **app :: views :: shared :: authentication :: signIn**: relazione entrante, verso tale classe sono notificate le informazioni da visualizzare per procedere al login nella piattaforma;
- **app :: views :: shared :: authentication :: signOut**: relazione entrante, verso tale classe si riverbereranno gli esiti di un'operazione di logout eseguita da un qualsiasi punto della piattaforma.

app :: controllers :: shared :: account :: AccountController

4.3.15.1 Descrizione

Classe contenente tutti i comandi atti a modificare lo stato del model rappresentante gli account che hanno già accesso alla piattaforma.

4.3.15.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a modificare lo stato del model rappresentante gli account che hanno già acceduto alla piattaforma (operazioni di modifica profilo).

4.3.15.3 Relazioni con altre classi

- **app :: models :: User**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: models :: daos :: user :: UserDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe User;
- **app :: views :: shared :: account :: editAccount**: relazione entrante, verso tale classe sono notificate le informazioni relative all'account in uso e per procedere alla modifica di questo.

app :: controllers :: shared :: account :: PasswordRecoverController

4.3.16.1 Descrizione

Classe contenente tutti i comandi atti a interagire con il model rappresentante gli account per un particolare utente che ha dimenticato o smarrito la password di accesso alla piattaforma.

4.3.16.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a leggere lo stato del model rappresentante gli account, in modo tale che l'utente abbia la possibilità di recuperare le credenziali di accesso alla piattaforma.

4.3.16.3 Relazioni con altre classi

- **app :: models :: User**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a leggerne lo stato;



- **app :: models :: daos :: user :: UserDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe User;
- **app :: views :: shared :: authentication :: passwordRecover**: relazione entrante, verso tale classe sono notificate le informazioni utili per generare l'interfaccia grafica relativa al recupero password.

app :: controllers :: shared :: account :: PasswordResetController

4.3.17.1 Descrizione

Classe contenente tutti i comandi atti a modificare lo stato del model rappresentante gli account per un particolare utente che ha dimenticato o smarrito la password di accesso alla piattaforma.

4.3.17.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a modificare lo stato del model rappresentante gli account, in modo tale che l'utente abbia la possibilità di resettare le credenziali di accesso alla piattaforma.

4.3.17.3 Relazioni con altre classi

- **app :: models :: User**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: models :: daos :: user :: UserDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe User;
- **app :: views :: Shared :: Authentication :: ResetPassword**: relazione entrante, verso tale classe sono notificate le informazioni per generare l'interfaccia grafica che fornisce il mezzo per effettuare il reset password.

app :: controllers :: shared :: administrator :: EngineController

4.3.18.1 Descrizione

Classe contenente tutti i comandi atti a cambiare lo stato del model rappresentante la configurazione del motore di analisi predittiva: il *Super Admin_G* o Admin, potrà da qui filtrare, o meno, i dati provenienti dal *thing* di riferimento e le regole predittive che si potranno applicare a tali dati. La selezione di questi componenti sarà visibile all'utente *Admin_G*, che andrà quindi a combinarli secondo le sue necessità.

4.3.18.2 Utilizzo

Tale classe viene utilizzata per reagire alle richieste HTTP generate dalla view corrispondente a tale controller e per la gestione dei comandi atti a cambiare lo stato del model rappresentante la configurazione delle funzioni disponibili durante la creazione di grafici e del model relativo alla tipologia di oggetti (modello).

4.3.18.3 Relazioni con altre classi

- **app :: models :: Thing**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: models :: daos :: thing :: ThingDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Thing;
- **app :: models :: ThingType**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;



- **app :: models :: daos :: thingType :: ThingTypeDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe ThingType;
- **app :: models :: Charts**: relazione entrante, verso tale classe sono rivolte le operazioni che ne vanno a modificarne lo stato;
- **app :: models :: daos :: charts :: ChartsDAO**: relazione entrante, viene utilizzata tale classe per le operazioni che vanno a interessare la base di dati e la classe Chart;
- **app :: views :: superAdmin :: engine :: engine**: relazione entrante, verso tale classe vengono notificate le informazioni relative ai tipi di dati degli oggetti e funzioni disponibili durante la creazione di grafici per ogni Company;
- **app :: views :: admin :: engine :: editFunctions**: relazione entrante, verso tale classe vengono notificate le informazioni relative alle funzioni disponibili durante la creazione di grafici per la Company dell'Admin loggato;
- **app :: views :: admin :: engine :: editData**: relazione entrante, verso tale classe vengono notificate le informazioni relative ai tipi di dato disponibili durante la creazione di grafici per la Company dell'Admin loggato.

Views

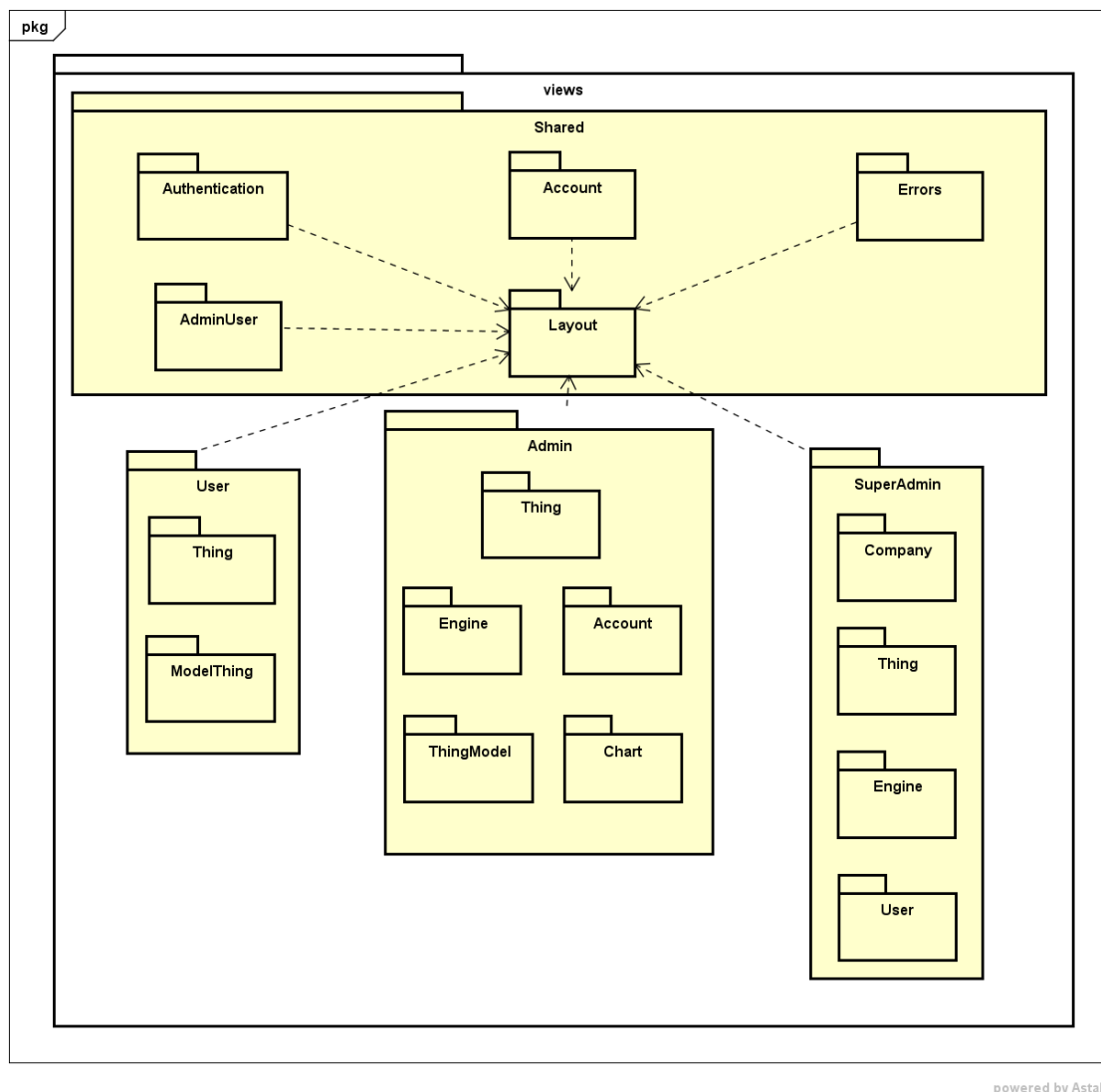


Figura 6: Diagramma package - Visione generale package Views

app :: views :: superAdmin :: home

4.4.1.1 Descrizione

Classe che visualizza la homepage per un utente di tipo SuperAdmin.

4.4.1.2 Utilizzo

Viene utilizzata dalla classe `app :: controllers :: superAdmin :: SuperAdminController` per visualizzare la pagina di home della propria tipologia di account.



app :: views :: superAdmin :: user :: users

4.4.2.1 Descrizione

Classe che visualizza la lista con tutti gli utenti iscritti alla piattaforma e la relativa interfaccia per lanciare le classiche operazioni CRUD.

4.4.2.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: UserController per gestire le richieste di operazione relative agli utenti iscritti alla piattaforma.

app :: views :: superAdmin :: user :: addUser

4.4.3.1 Descrizione

Classe che visualizza la form necessaria ad aggiungere un nuovo utente alla piattaforma.

4.4.3.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: UserController per gestire le richieste di operazione relative agli utenti iscritti alla piattaforma.

app :: views :: superAdmin :: user :: editUser

4.4.4.1 Descrizione

Classe che visualizza la form necessaria a modificare un utente presente nella piattaforma.

4.4.4.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: UserController per gestire le richieste di operazione relative agli utenti iscritti alla piattaforma.

app :: views :: superAdmin :: company :: companies

4.4.5.1 Descrizione

Classe che visualizza la lista con tutte le company presenti nella piattaforma e le relative operazioni di CRUD disponibili.

4.4.5.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: CompanyController per visualizzare la pagina di gestione delle company.

app :: views :: superAdmin :: company :: addCompany

4.4.6.1 Descrizione

Classe che visualizza la form per aggiungere una nuova company alla piattaforma.

4.4.6.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: CompanyController per visualizzare la pagina di aggiunta di una nuova company.



app :: views :: superAdmin :: company :: editCompany

4.4.7.1 Descrizione

Classe che visualizza la form per modificare i dati di una company già presente nella piattaforma.

4.4.7.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: CompanyController per visualizzare la pagina di modifica di una company presente nella piattaforma.

app :: views :: superAdmin :: thing :: things

4.4.8.1 Descrizione

Classe che visualizza la lista con tutte i thing presenti nella piattaforma e le relative operazioni di CRUD disponibili.

4.4.8.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: ThingController per visualizzare la pagina di gestione dei thing.

app :: views :: superAdmin :: thing :: addThing

4.4.9.1 Descrizione

Classe che visualizza la form per aggiungere una nuova company alla piattaforma.

4.4.9.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: ThingController per visualizzare la pagina di aggiunta di un nuovo thing per una data company.

app :: views :: superAdmin :: thing :: editThing

4.4.10.1 Descrizione

Classe che visualizza la form per modificare i dati di un thing già presente nella piattaforma.

4.4.10.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: ThingController per visualizzare la pagina di modifica di un thing presente nella piattaforma.

app :: views :: superAdmin :: thingType :: thinTypes

4.4.11.1 Descrizione

Classe che visualizza la lista con tutte i tipi di thing presenti nella piattaforma e le relative operazioni di CRUD disponibili.

4.4.11.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: ThingTypeController per visualizzare la pagina di gestione dei tipi dei thing.



app :: views :: superAdmin :: thingType :: addThingType

4.4.12.1 Descrizione

Classe che visualizza la form per aggiungere un nuovo tipo di thing alla piattaforma.

4.4.12.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: ThingTypeController per visualizzare la pagina di aggiunta di un nuovo tipo di thing per una data company.

app :: views :: superAdmin :: thingType :: editThingType

4.4.13.1 Descrizione

Classe che visualizza la form per modificare i dati di un tipo di thing già presente nella piattaforma.

4.4.13.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: superAdmin :: ThingController per visualizzare la pagina di modifica di un tipo di thing presente nella piattaforma.

app :: views :: superAdmin :: engine :: engine

4.4.14.1 Descrizione

Classe che visualizza delle tabelle relative alle funzioni utilizzabili in fase di definizione di un modello di analisi e un interfaccia relativa a quali dati per ogni modello di oggetto sono effettivamente inoltrati alla company.

4.4.14.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: shared :: administrator :: EngineController per filtrare le funzioni che rendere disponibili a una data company, così come per i dati relativi ad ogni modello di oggetto che vengono effettivamente inoltrati alla company.

app :: views :: admin :: thing :: thingDetails

4.4.15.1 Descrizione

Classe che visualizza una tabella contenente informazioni relative allo stato del Thing, ai grafici che lo interessano e all'invio di comandi e aggiornamento firmware degli oggetti dello stesso tipo.

4.4.15.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: admin :: ThingController per visualizzare la dei dettagli del things, dei grafici che lo interessano e della form per l'invio di comandi e update firmware.

app :: views :: admin :: thingType :: thingTypeDetails

4.4.16.1 Descrizione Classe che visualizza una tabella contenente informazioni relative allo stato del Thing, ai grafici che lo interessano e aggiornamento firmware per quella tipologia di oggetto.

4.4.16.2 Utilizzo Viene utilizzata dalla classe app :: controllers :: sdmin :: thingType :: ThingTypeController per visualizzare la pagina dei dettagli di un modello di thing, i grafici che lo interessano e la form per l'invio di update del firmware.



app :: views :: admin :: chart :: newChart

4.4.17.1 Descrizione

Classe che visualizza una pagina con la procedura a passaggi che permette di creare dei grafici personalizzati per un particolare modello di oggetti selezionando le funzioni, il tipo di grafico, i dati e come relazionarli tra loro.

4.4.17.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: admin :: chart :: ChartController per visualizzare la pagina con la procedura di creazione grafici.

app :: views :: admin :: engine :: editFunctions

4.4.18.1 Descrizione Classe che visualizza delle tabelle relative alle funzioni utilizzabili in fase di definizione di un modello di analisi.

4.4.18.2 Utilizzo Viene utilizzata dalla classe app :: controllers :: shared :: administrator :: EngineController per filtrare le funzioni che rende disponibili alla Company dell'Admin autenticato.

app :: views :: admin :: engine :: editData

4.4.19.1 Descrizione Classe che visualizza delle tabelle relative ai dati visualizzabili e utilizzabili poi in fase di definizione di un modello di analisi.

4.4.19.2 Utilizzo Viene utilizzata dalla classe app :: controllers :: shared :: administrator :: EngineController per filtrare i dati da rendere disponibili alla Company dell'Admin autenticato.

app :: views :: admin :: user :: users

4.4.20.1 Descrizione

Classe che visualizza la lista con tutti gli utenti iscritti alla piattaforma e la relativa interfaccia per lanciare le classiche operazioni CRUD.

4.4.20.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: admin :: UserController per gestire le richieste di operazione relative agli utenti iscritti alla piattaforma.

app :: views :: admin :: user :: addUser

4.4.21.1 Descrizione

Classe che visualizza la form necessaria ad aggiungere un nuovo utente alla piattaforma.

4.4.21.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: admin :: UserController per gestire le richieste di operazione relative agli utenti iscritti alla piattaforma.



app :: views :: admin :: user :: editUser

4.4.22.1 Descrizione

Classe che visualizza la form necessaria a modificare un utente presente nella piattaforma.

4.4.22.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: admin :: UserController per gestire le richieste di operazione relative agli utenti iscritti alla piattaforma.

app :: views :: admin :: home

4.4.23.1 Descrizione

Classe che visualizza la homepage per un utente di tipo Admin.

4.4.23.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: admin :: AdminController per visualizzare la pagina di home della proprio tipologia di account.

app :: views :: user :: home

4.4.24.1 Descrizione

Classe che visualizza la homepage per un utente di tipo User.

4.4.24.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: user :: UserController per visualizzare la pagina di home della proprio tipologia di account.

app :: views :: user :: thing :: thingDetails

4.4.25.1 Descrizione

Classe che visualizza una tabella contenente i dati di un particolare $thing_G$ e i grafici relativi allo stesso.

4.4.25.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: user :: ThingController per visualizzare le informazioni dei singoli oggetti.

app :: views :: user :: thingType :: thingTypeDetails

4.4.26.1 Descrizione

Classe che visualizza una tabella contenente il riepilogo dei dati degli oggetti di un particolare modello e i grafici generici relativi al modello stesso.

4.4.26.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: user :: ThingTypeController per visualizzare le informazioni dei singoli modelli.



app :: views :: shared :: adminUser :: things

4.4.27.1 Descrizione

Classe che visualizza la pagina con la lista dei things in uso dalla propria company.

4.4.27.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: admin :: ThingController e dalla classe app :: controllers :: user :: ThingController per la visualizzazione della pagina con la lista dei things.

app :: views :: shared :: adminUser :: thingTypes

4.4.28.1 Descrizione

Classe che visualizza la pagina con la lista delle tipologie di things in uso dalla propria company.

4.4.28.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: admin :: ThingTypeController e dalla classe app :: controllers :: user :: ThingTypeController per la visualizzazione della pagina con la lista dei modelli dei things in uso dalla propria company.

app :: views :: shared :: adminUser :: notifications

4.4.29.1 Descrizione

Classe che visualizza la lista con tutte le notifiche impostate per i thing o un loro tipo e la relativa interfaccia per lanciare le classiche operazioni CRUD.

4.4.29.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: shared :: adminUser :: NotificationController per gestire le richieste di operazione relative alle notifiche su thing e loro tipi.

app :: views :: shared :: adminUser :: addNotification

4.4.30.1 Descrizione

Classe che visualizza la form necessaria ad aggiungere una nuova notifica per un thing o una loro tipologia.

4.4.30.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: shared :: adminUser :: NotificationController per gestire le richieste di operazione relative alle notifiche su thing e loro tipi.

app :: views :: shared :: adminUser :: editNotification

4.4.31.1 Descrizione

Classe che visualizza la form necessaria a modificare una notifica impostata per un dato thing o una tipologia di esso.

4.4.31.2 Utilizzo

Viene utilizzata dalla classe app :: controllers :: shared :: adminUser :: NotificationController per gestire le richieste di operazione relative alle notifiche su thing e loro tipi.



app :: views :: shared :: authentication :: signIn

4.4.32.1 Descrizione

Classe che visualizza la pagina di $login_G$ con il $form_G$ necessario ad autenticarsi.

4.4.32.2 Utilizzo

Viene utilizzata dalla classe `app :: controllers :: shared :: authentication :: AuthenticationController` per la visualizzazione della pagina di $login_G$.

app :: views :: shared :: authentication :: singOut

4.4.33.1 Descrizione

Classe che visualizza gli esiti di un'operazione di logout.

4.4.33.2 Utilizzo

Viene utilizzata dalla classe `app :: controllers :: shared :: authentication :: AuthenticationController` per la visualizzazione della pagina necessaria dopo un'operazione di logout.

app :: views :: shared :: authentication :: passwordRecover

4.4.34.1 Descrizione

Classe che visualizza la pagina di recupero password e il form associato atto ad assolvere tale compito.

4.4.34.2 Utilizzo

Viene utilizzata dalla classe `app :: controllers :: shared :: account :: PasswordRecoverController` per la visualizzazione della pagina di recupero password.

app :: views :: shared :: authentication :: resetPassword

4.4.35.1 Descrizione

Classe che visualizza la pagina di reset password e il form associato atto ad assolvere tale compito.

4.4.35.2 Utilizzo

Viene utilizzata dalla classe `app :: controllers :: shared :: account :: PasswordResetController` per la visualizzazione della pagina di reset password.

app :: views :: shared :: account :: editAccount

4.4.36.1 Descrizione

Classe che visualizza una pagina per la modifica del profilo contenente i $form_G$ per la modifica delle varie informazioni, sia anagrafiche che di autenticazione.

4.4.36.2 Utilizzo

Viene utilizzata dalla classe `app :: controllers :: shared :: account :: AccountController` per la visualizzazione della pagina di modifica del profilo.



app :: views :: shared :: layout :: header

4.4.37.1 Descrizione

Classe che permette la creazione dell' $header_G$ delle varie $view_G$.

4.4.37.2 Utilizzo

Viene utilizzata da tutte le $view_G$ per creare la parte di $header_G$.

app :: views :: shared :: layout :: footer

4.4.38.1 Descrizione

Classe che permette la creazione dell' $footer_G$ delle varie $view_G$.

4.4.38.2 Utilizzo

Viene utilizzata da tutte le $view_G$ per creare la parte di $footer_G$.

app :: views :: shared :: layout :: layout

4.4.39.1 Descrizione

Classe che permette la creazione del $layout_G$ delle varie $view_G$.

4.4.39.2 Utilizzo

Viene utilizzata da tutte le $view_G$ per creare la parte di $layout_G$.

app :: views :: shared :: layout :: nav

4.4.40.1 Descrizione

Classe che permette la creazione del menu di navigazione delle varie $view_G$. Ogni view riempirà il proprio menu a seconda dell'esigenza.

4.4.40.2 Utilizzo

Viene utilizzata da tutte le $view_G$ per creare il menu di navigazione.

app :: views :: shared :: errors :: 404

4.4.41.1 Descrizione

Classe che visualizza la pagina di errore 404_G .

4.4.41.2 Utilizzo

Tale classe entra in gioco ogni qual volta un utente inserisca una URL_G non corretta all'interno della barra degli indirizzi del browser per raggiungere una parte della piattaforma.

app :: views :: shared :: errors :: 502

4.4.42.1 Descrizione

Classe che visualizza la pagina di errore 502_G .



4.4.42.2 Utilizzo

Tale classe entra in gioco quando un qualsiasi utente, inserendo una URL_G all'interno della barra degli indirizzi del browser, cerchi di entrare in una pagina per la quale non ha i permessi di accesso.



Diagrammi di attività

In questa sezione vengono riportati i diagrammi di attività prodotti durante la progettazione architeturale. Essi hanno l'obiettivo di descrivere le interazioni degli utenti con il sistema.

Verrà inizialmente fornito uno schema generale, di alto livello, indicante le attività principali eseguibili dagli utenti all'interno della piattaforma. Successivamente, saranno presenti i diagrammi di dettaglio di tali attività.

Attività Principali

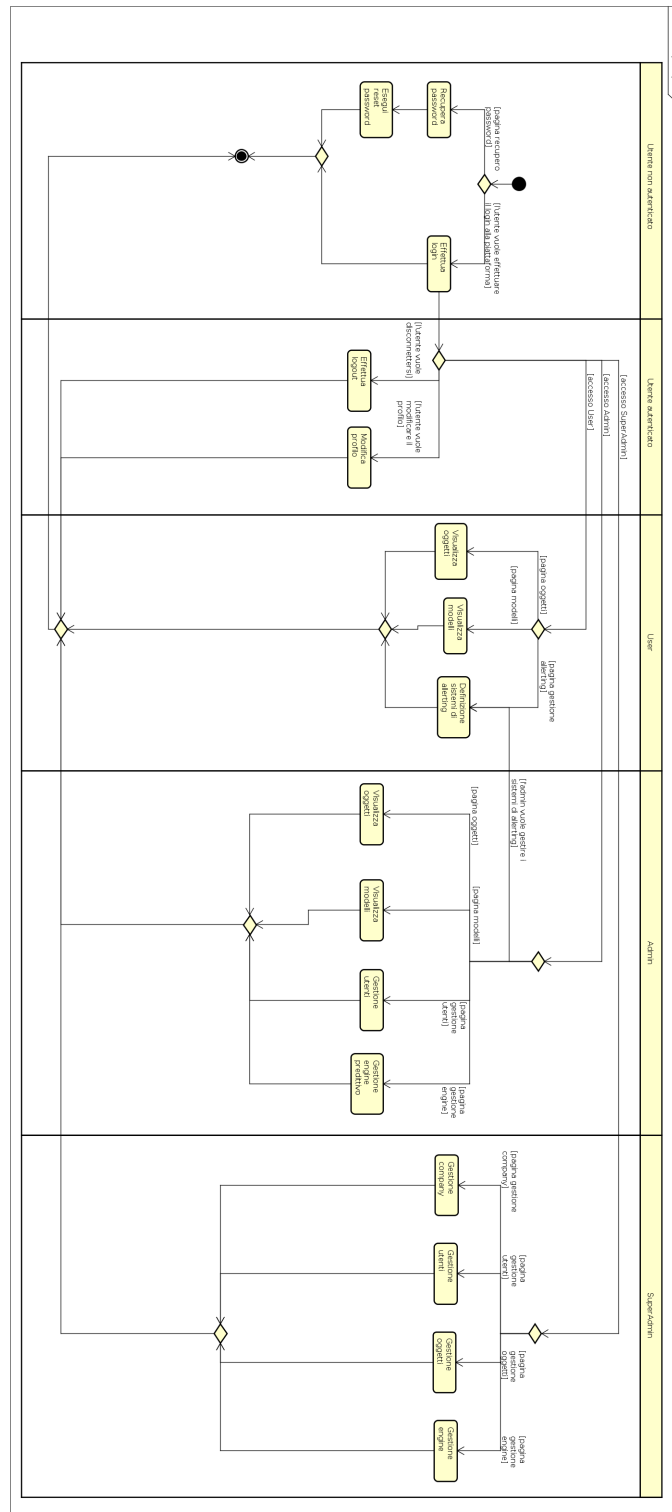


Figura 7: Diagramma di attività - Attività Principali.



La piattaforma $UMAP_G$ è costituita da una serie di pagine web che permettono di utilizzare il sistema. Un utente, accedendo inizialmente all'applicazione, ha la possibilità di effettuare il login o di recuperare la password dimenticata. Una volta entrato, a seconda della categoria alla quale appartiene, viene indirizzato alla pagina ad esso dedicata. Seguono le funzioni predisposte per ogni categoria di utente.

Uno User può:

- Visualizzare l'elenco degli oggetti della company a cui è associato;
- Visualizzare l'elenco dei modelli degli oggetti della company;
- Definire sistemi di alerting personalizzati.

Un Admin può:

- Visualizzare gli oggetti della propria $company_G$ e inviargli comandi da terminale e aggiornamenti;
- visualizzare i modelli della propria $company_G$ e inviargli aggiornamenti;
- gestire gli utenti della propria $company_G$;
- gestire l'algoritmo predittivo per la propria $company_G$;
- creare grafici per gli utenti della propria $company_G$.

Un SuperAdmin può:

- Gestire le $company_G$;
- gestire gli utenti della piattaforma;
- gestire gli oggetti della piattaforma (Thing e modelli);
- gestire l'algoritmo predittivo per tutte le $company_G$ della piattaforma.

Sono previste inoltre, per ogni categoria di utente, funzionalità per:

- Modificare il proprio profilo;
- Effettuare il logout.

Utente non Autenticato

Recupera password

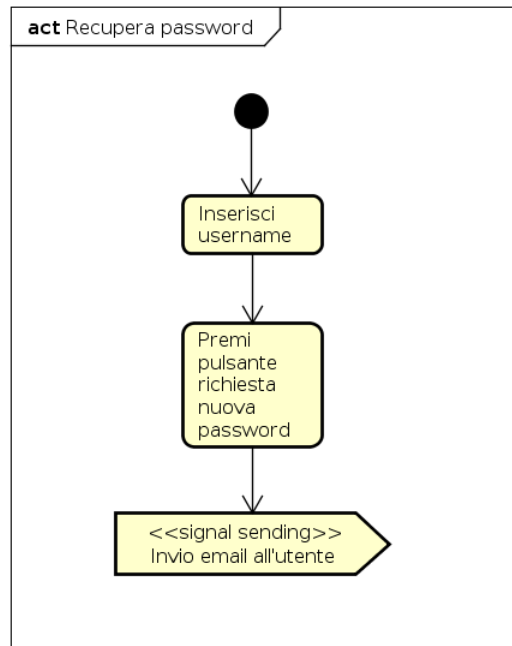


Figura 8: Diagramma di attività - Recupera password.

Nella pagina predisposta per il recupero della password, l'utente dovrà inserire il proprio username e premere il pulsante di invio. Il sistema procederà ad inviare una email contenente un link che rimanda alla pagina di inserimento di una nuova password.

Esegui reset password

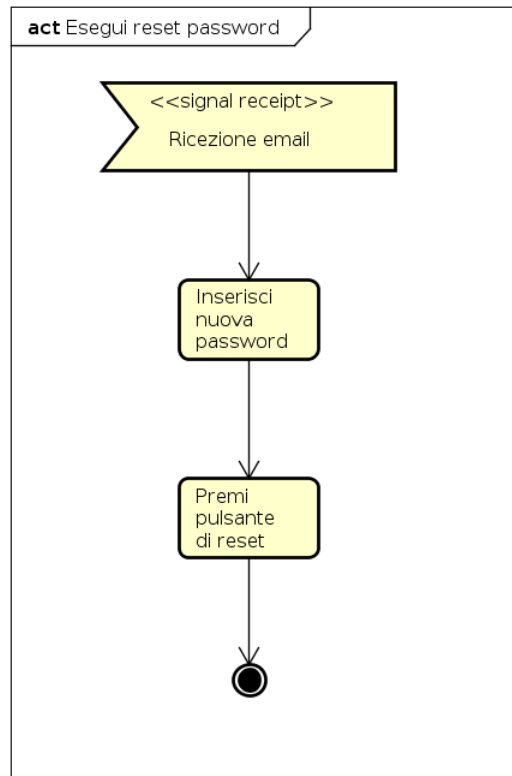


Figura 9: Diagramma di attività - Esegui reset password.

Una volta ricevuta l'email, basterà per l'utente cliccare sul link in essa contenuto per essere indirizzato alla pagina di inserimento di una nuova password. Tale pagina contiene un campo dati per il suo inserimento. Il pulsante di invio causerà il salvataggio della nuova password associata all'utente permettendogli, quindi, di accedere nuovamente al sistema.

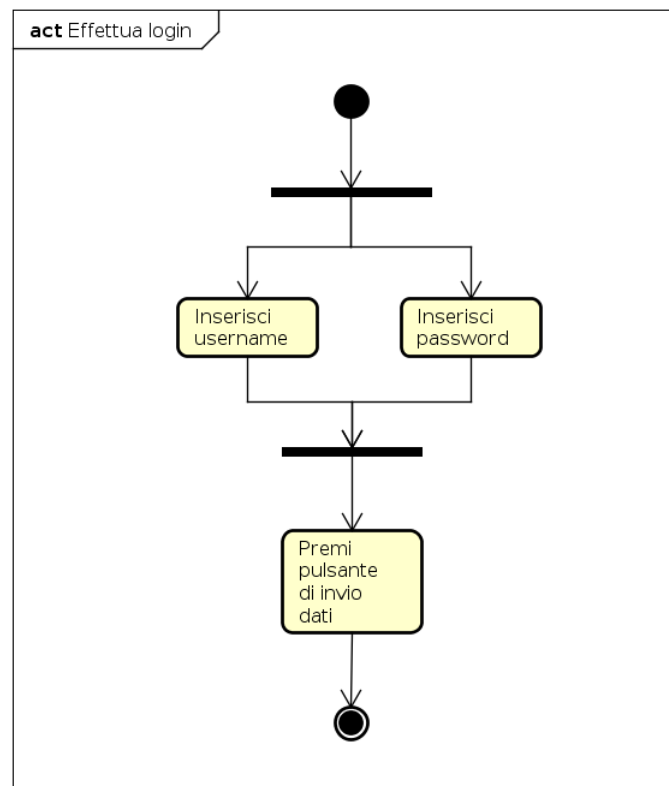
Effettua login

Figura 10: Diagramma di attività - Effettua login.

Per effettuare il login l'utente deve inserire i propri username e password e cliccare sul pulsante di invio. Il sistema, dopo aver verificato la correttezza delle credenziali, indirizzerà l'utente alla pagina ad esso dedicata, a seconda della categoria a cui appartiene.

Utente Autenticato

Modifica profilo

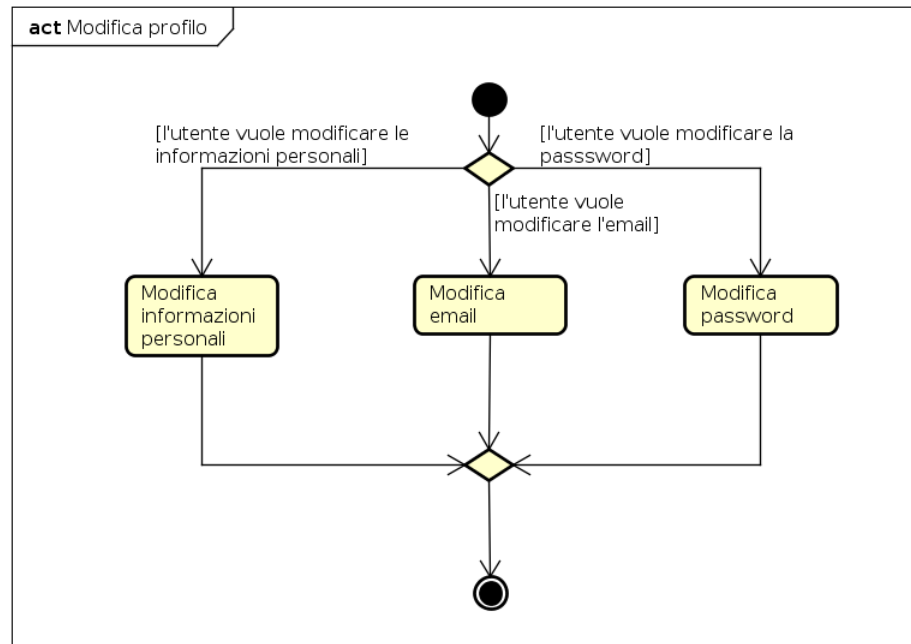


Figura 11: Diagramma di attività - Modifica profilo.

Nella pagina dedicata alla modifica del profilo, l'utente può scegliere se:

- Modificare le informazioni personali (nome e cognome);
- Modificare l'email;
- Modificare la password.

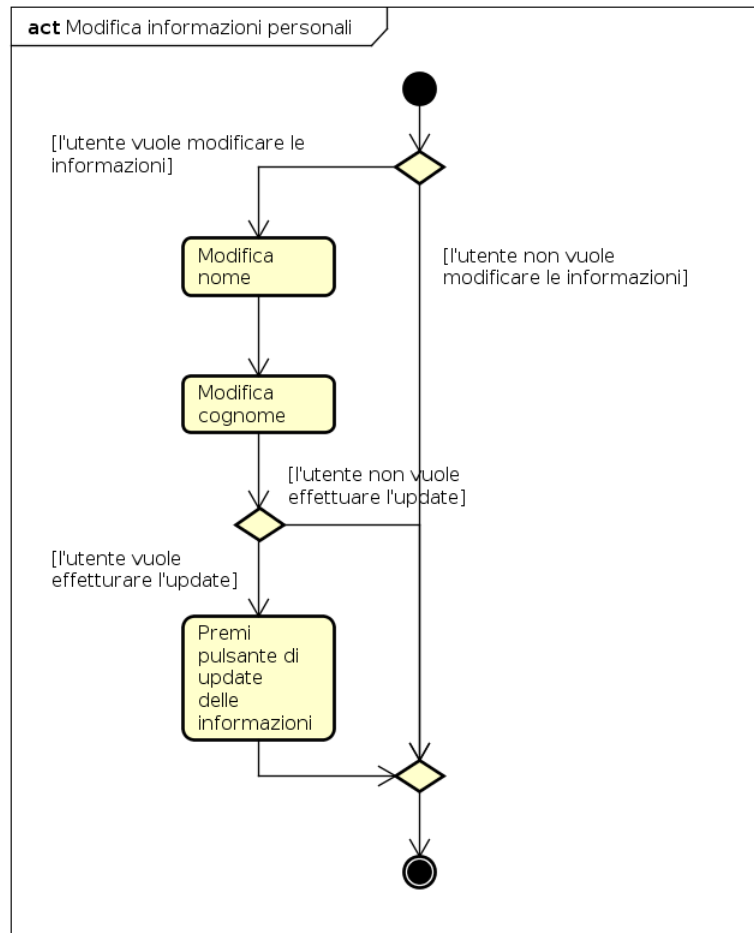


Figura 12: Diagramma di attività - Modifica informazioni personali.

5.3.1.1 Modifica informazioni personali Per quanto riguarda la modifica della informazioni personali, l'utente avrà a disposizione due campi dati in cui inserire i nuovi nome e cognome. Esso può anche annullare in ogni momento la procedura attraverso un apposito pulsante. Il pulsante di invio, invece, causa il salvataggio delle nuove informazioni da parte del sistema.

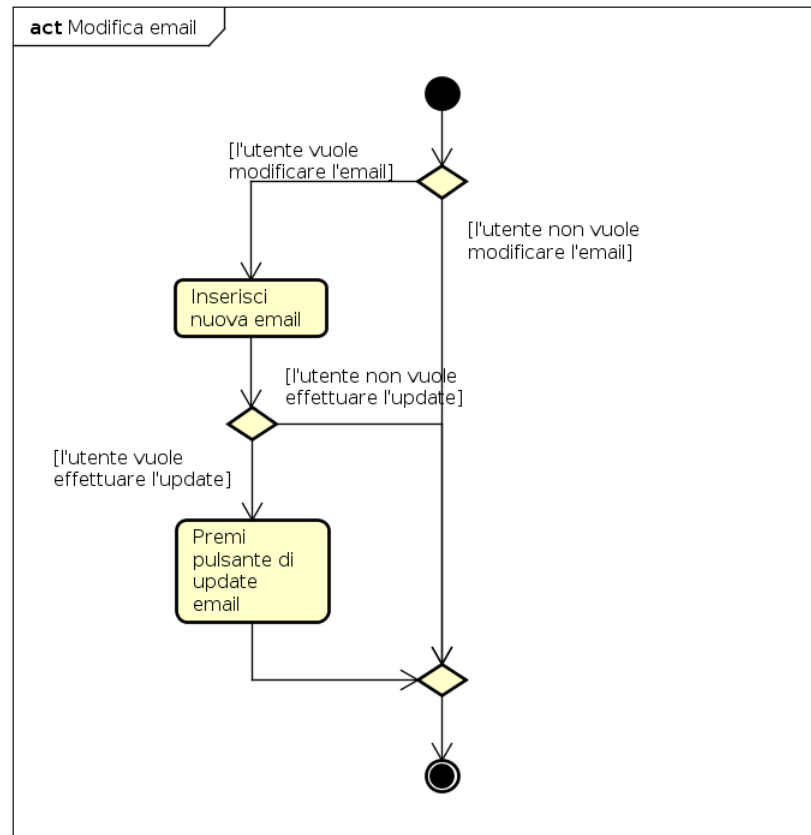


Figura 13: Diagramma di attività - Modifica email.

5.3.1.2 Modifica email Nella pagina dedicata alla modifica dell'email un campo dati permette all'utente di inserire il nuovo indirizzo. Il pulsante di invio causa il salvataggio delle modifiche nel sistema. Anche qui l'operazione può essere annullata in qualunque momento attraverso il pulsante apposito.

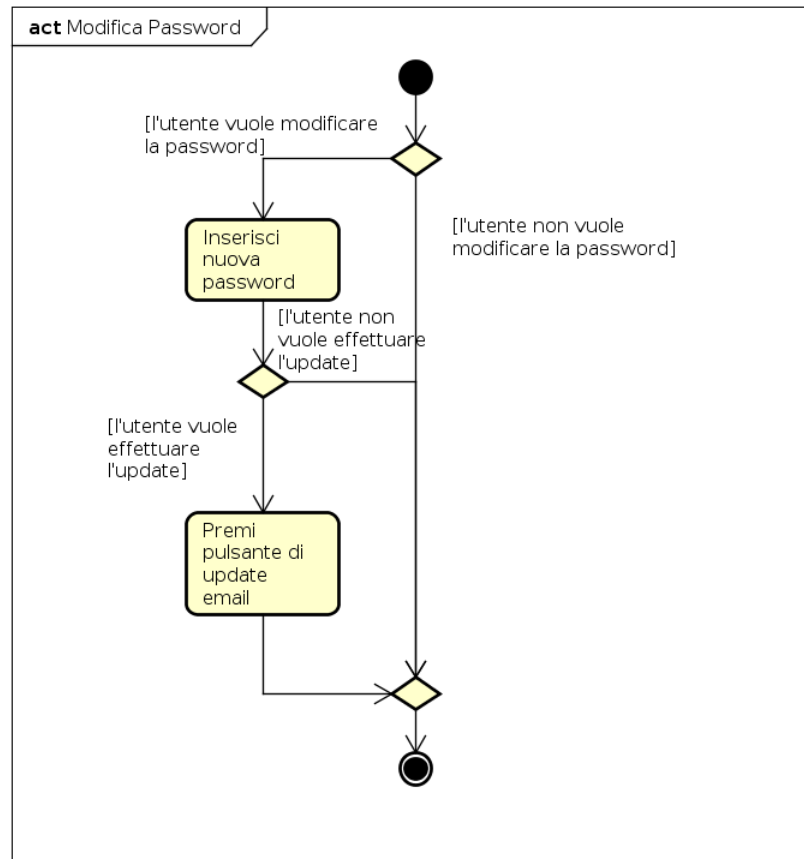


Figura 14: Diagramma di attività - Modifica password.

5.3.1.3 Modifica password Nella pagina dedicata alla modifica della password, un campo dati permette all'utente di inserire quella nuova. Il pulsante di invio causa il salvataggio delle modifiche nel sistema. Anche qui l'operazione può essere annullata in qualunque momento attraverso il pulsante apposito.

User

Visualizza Oggetti

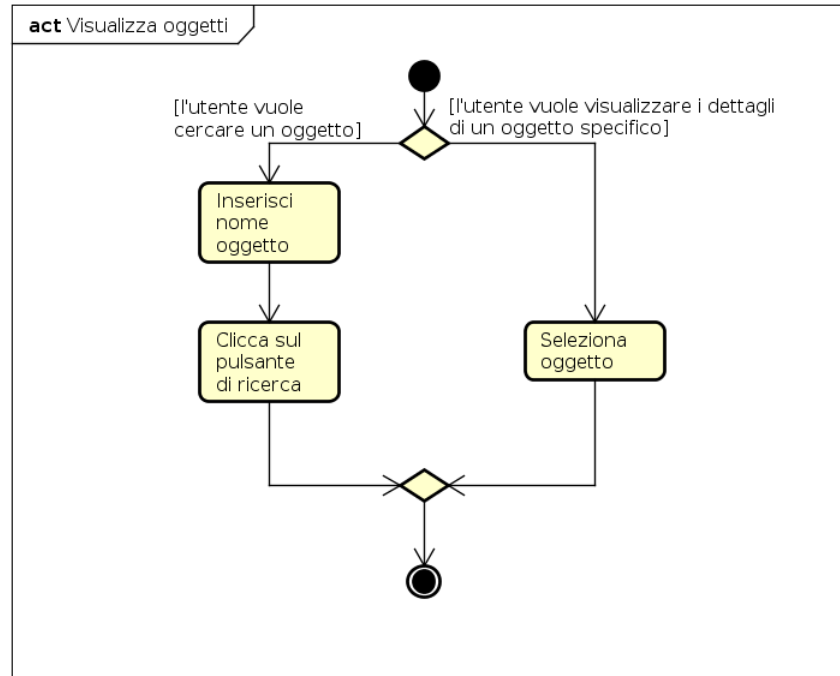


Figura 15: Diagramma di attività - Visualizza oggetti.

Una volta che l'utente entra nell'area dedicata alla visualizzazione degli oggetti, gli viene presentata una tabella contenente la lista degli oggetti appartenente alla company a cui è associato. Da qui può decidere se:

- Cercare un oggetto per nome: per eseguire questa attività l'utente deve inserire il nome dell'oggetto e successivamente, cliccando il pulsante di ricerca, la tabella si aggiorna con i risultati;
- Selezionare un oggetto per vedere i dettagli ad esso associati.

Visualizza modelli

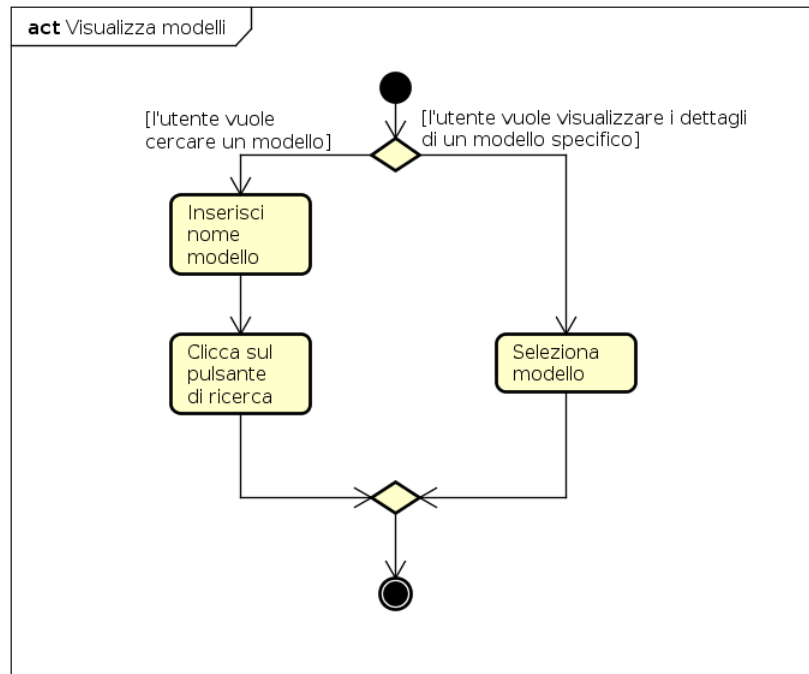


Figura 16: Diagramma di attività - Visualizza modelli.

Nella pagina dedicata alla visualizzazione dei modelli, l'utente visualizza una tabella contenente l'elenco dei modelli degli oggetti appartenenti alla company a cui è associato. Qui ha la possibilità di:

- Cercare un modello per nome: inserendo il nome di un modello e cliccando sul pulsante di ricerca, la tabella si aggiorna con i risultati trovati.
- Selezionare un modello per visualizzarne i dettagli.

Definizione sistemi di alerting

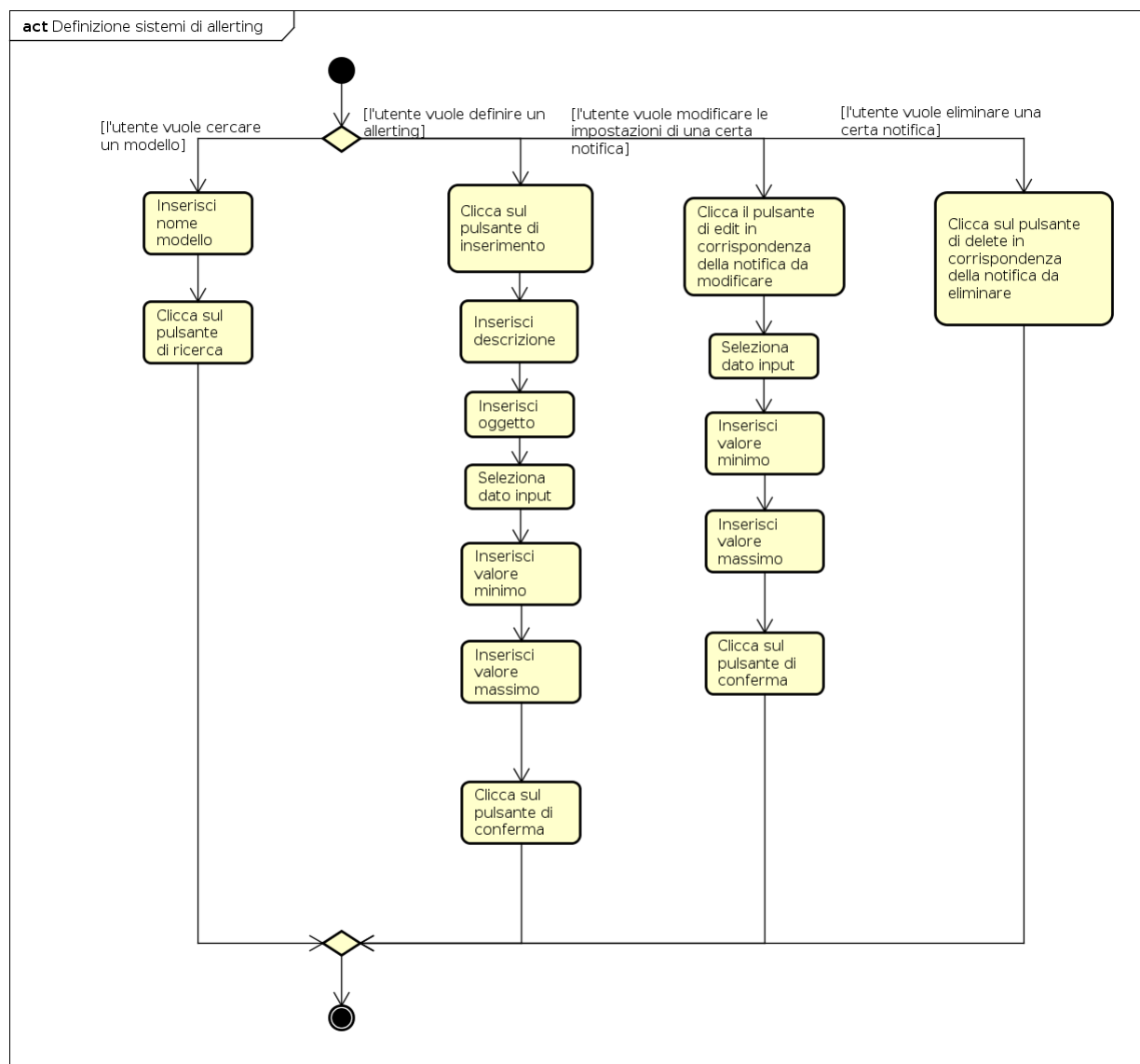


Figura 17: Diagramma di attività - Definizione sistemi di alerting.

Nella piattaforma è prevista la possibilità, per l'utente, di definire sistemi di alerting personalizzati. La pagina presenta, innanzitutto, una tabella contenente le notifiche già inserite e i relativi dettagli. Le operazioni possibili sono:

- Ricerca delle notifiche relative a oggetti di un certo modello;
- Inserimento di nuove notifiche;
- Modifica di notifiche già esistenti;
- Eliminazione di una notifica.

Per cercare un modello è sufficiente, per l'utente, inserire il nome nella barra di ricerca e cliccare sul pulsante apposito. La tabella si aggiornerà con tutte le notifiche predisposte per oggetti di quel modello.

Cliccando sul pulsante di creazione di un nuovo alerting, il sistema presenta all'utente un pop-up in cui inserire tutte le informazioni necessarie alla creazione di una nuova notifica, associata ad uno specifico oggetto.

Cliccando sul pulsante di modifica in corrispondenza di una specifica notifica è possibile, nel pop-up che si apre, modificare le informazioni precedentemente inserite per definire la notifica.

Infine viene data la possibilità all'utente di eliminare un alerting precedentemente definito, cliccando sul pulsante di eliminazione in corrispondenza della notifica desiderata.

Le operazioni di creazione e modifica di una notifica possono essere annullate in qualunque momento attraverso un pulsante apposito nella finestra del pop-up.

Admin

Visualizza Oggetti

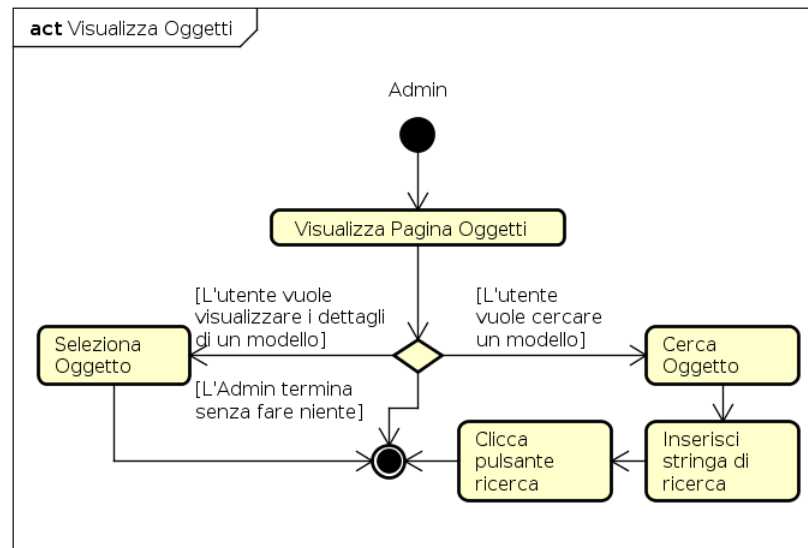


Figura 18: Diagramma di attività - Visualizza Oggetti.

Nella pagina dedicata alla visualizzazione degli oggetti, l' $admin_G$ visualizza una tabella contenente l'elenco degli oggetti appartenenti alla propria $company_G$. Qui ha la possibilità di:

- Cercare un oggetto per nome: inserendo una stringa e cliccando sul pulsante di ricerca, la tabella si aggiorna con i risultati trovati.
- Selezionare un oggetto per visualizzarne i dettagli.

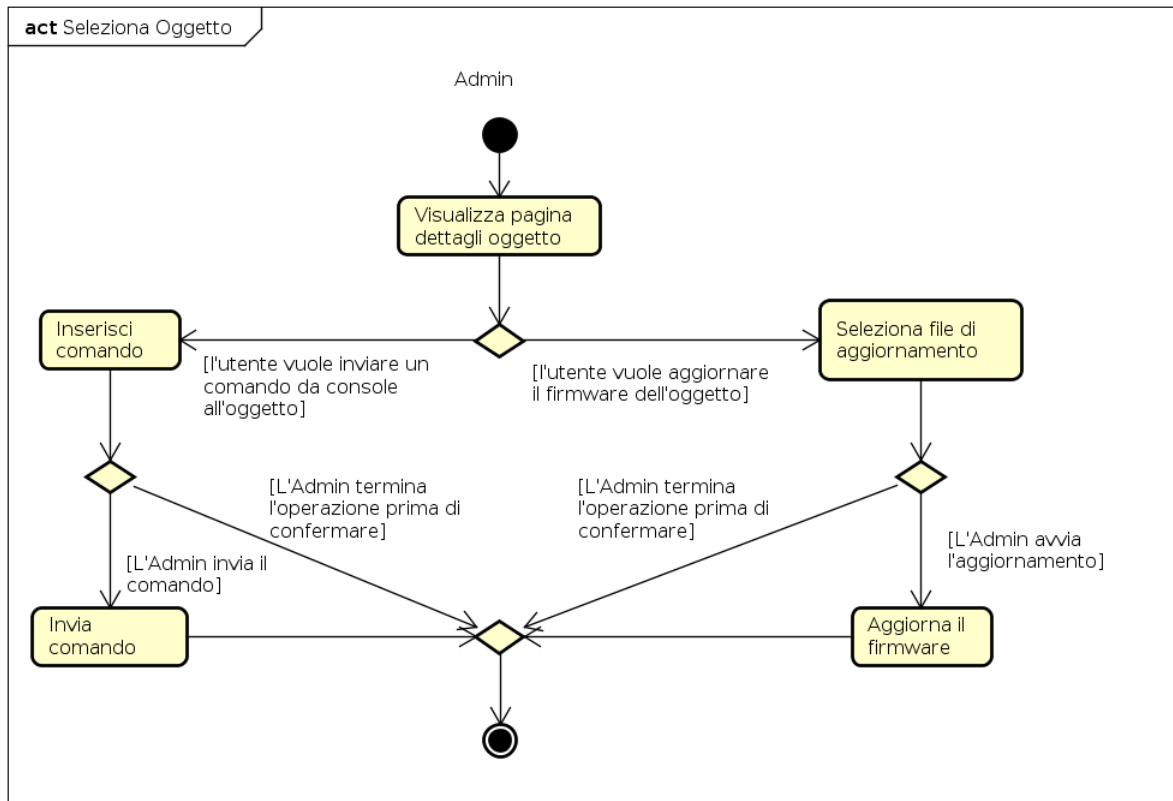


Figura 19: Diagramma di attività - Seleziona Oggetto.

5.5.1.1 Seleziona Oggetto Selezionando un oggetto $l'admin_c$ aprirà la pagina con i dettagli relativi a quel particolare oggetto. Oltre a visualizzare una tabella e dei grafici relativi ad esso, sarà possibile:

- Inviare dei comandi da riga di comando all'oggetto
- Caricare un file per l'aggiornamento e di conseguenza aggiornare il firmware dell'oggetto.

Visualizza Modelli

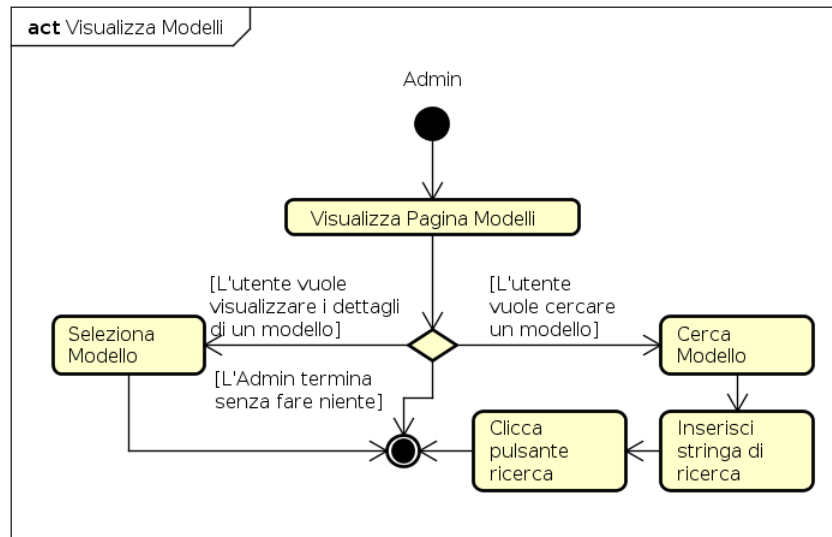


Figura 20: Diagramma di attività - Visualizza Modelli.

Nella pagina dedicata alla visualizzazione dei modelli, l' $admin_G$ visualizza una tabella contenente l'elenco dei modelli di oggetti appartenenti alla propria $company_G$. Qui ha la possibilità di:

- Cercare un modello per nome: inserendo una stringa e cliccando sul pulsante di ricerca, la tabella si aggiorna con i risultati trovati.
- Selezionare un modello per visualizzarne i dettagli.

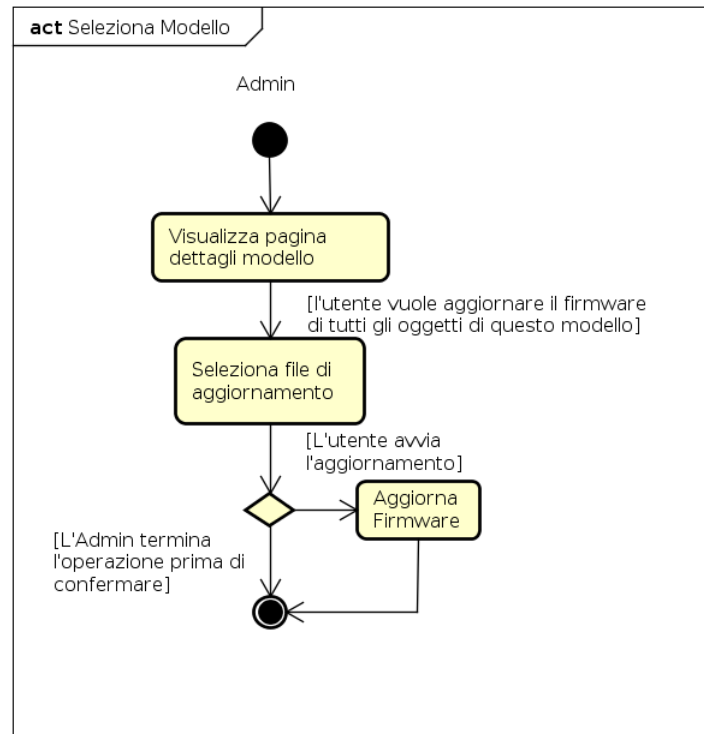


Figura 21: Diagramma di attività - Seleziona Modello.

5.5.2.1 Seleziona Modello Selezionando un modello l' $admin_G$ aprirà la pagina con i dettagli relativi a quel particolare modello. Oltre a visualizzare una tabella e dei grafici relativi ad esso, sarà possibile caricare un file per l'aggiornamento e di conseguenza aggiornare il firmware di tutti gli oggetti di quel particolare modello facenti parte della propria $company_G$.

Gestione Utenti

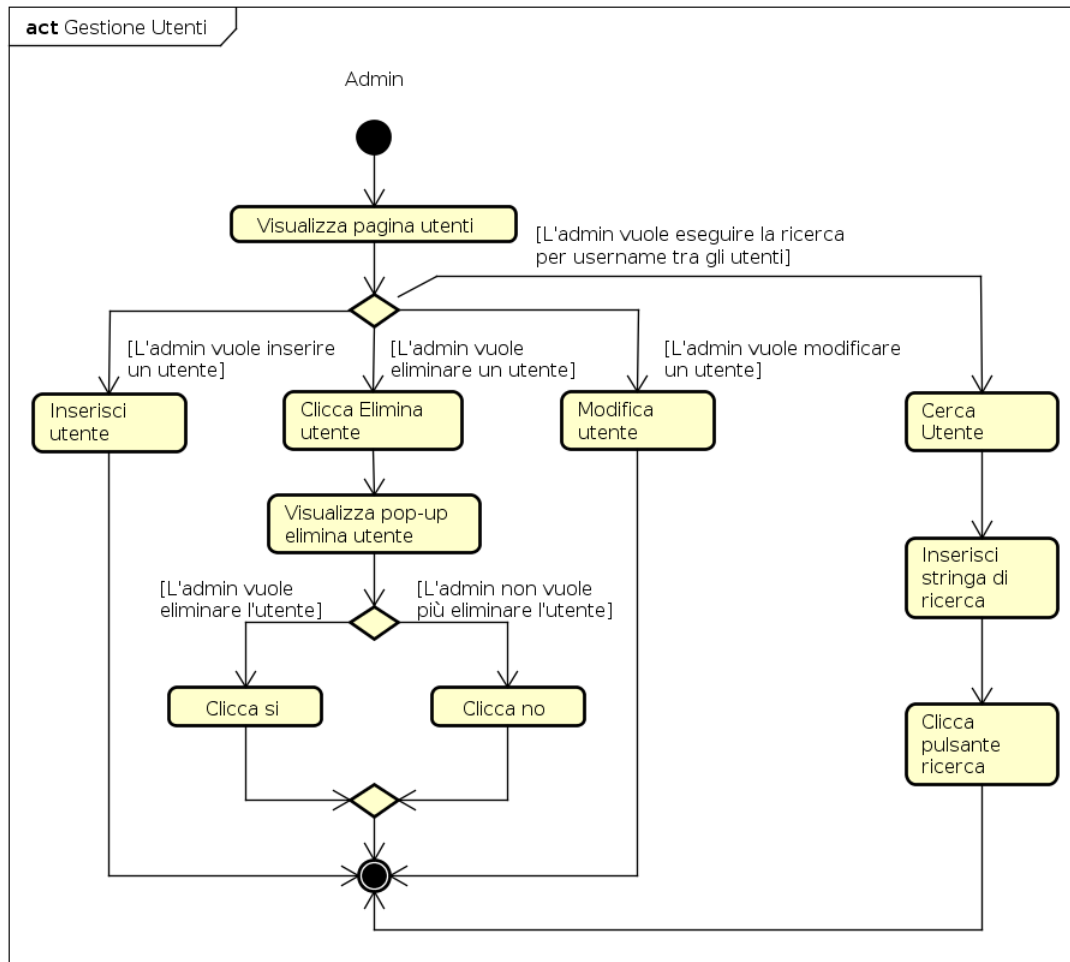


Figura 22: Diagramma di attività - Gestione Utenti.

Nella pagina dedicata alla visualizzazione degli utenti, l' $admin_G$ visualizza una tabella contenente l'elenco degli utenti appartenenti alla propria $company_G$. Qui ha la possibilità di:

- Inserire un nuovo utente;
- Eliminare un utente esistente premendo l'apposito pulsante;
- Modificare un utente già esistente;
- Cercare un utente per username: inserendo una stringa e cliccando sul pulsante di ricerca, la tabella si aggiorna con i risultati trovati.

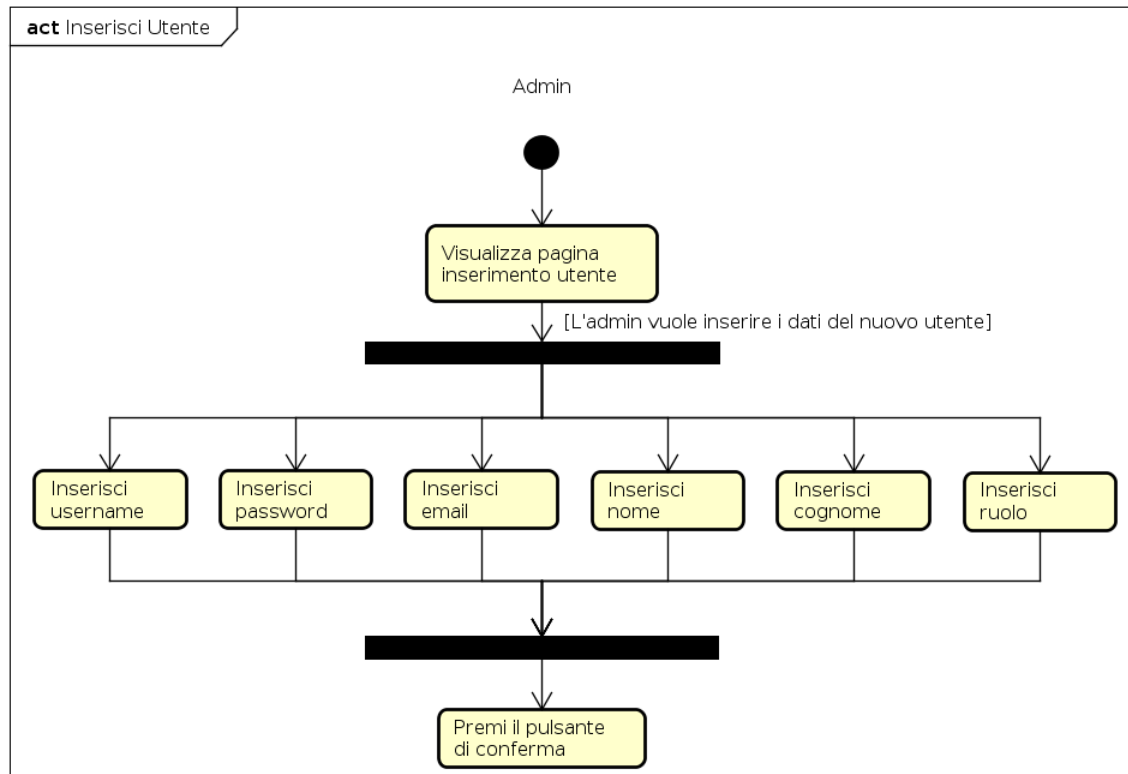


Figura 23: Diagramma di attività - Inserisci Utente.

5.5.3.1 Inserisci Utente Visualizzando la pagina per l'inserimento dell'utente l' $admin_G$ potrà inserire tutti i campi obbligatori per aggiungere un nuovo utente alla propria $company_G$ e inserirlo. I campi sono:

- $Username_G$;
- Password;
- E-mail;
- Nome;
- Cognome;
- Ruolo.

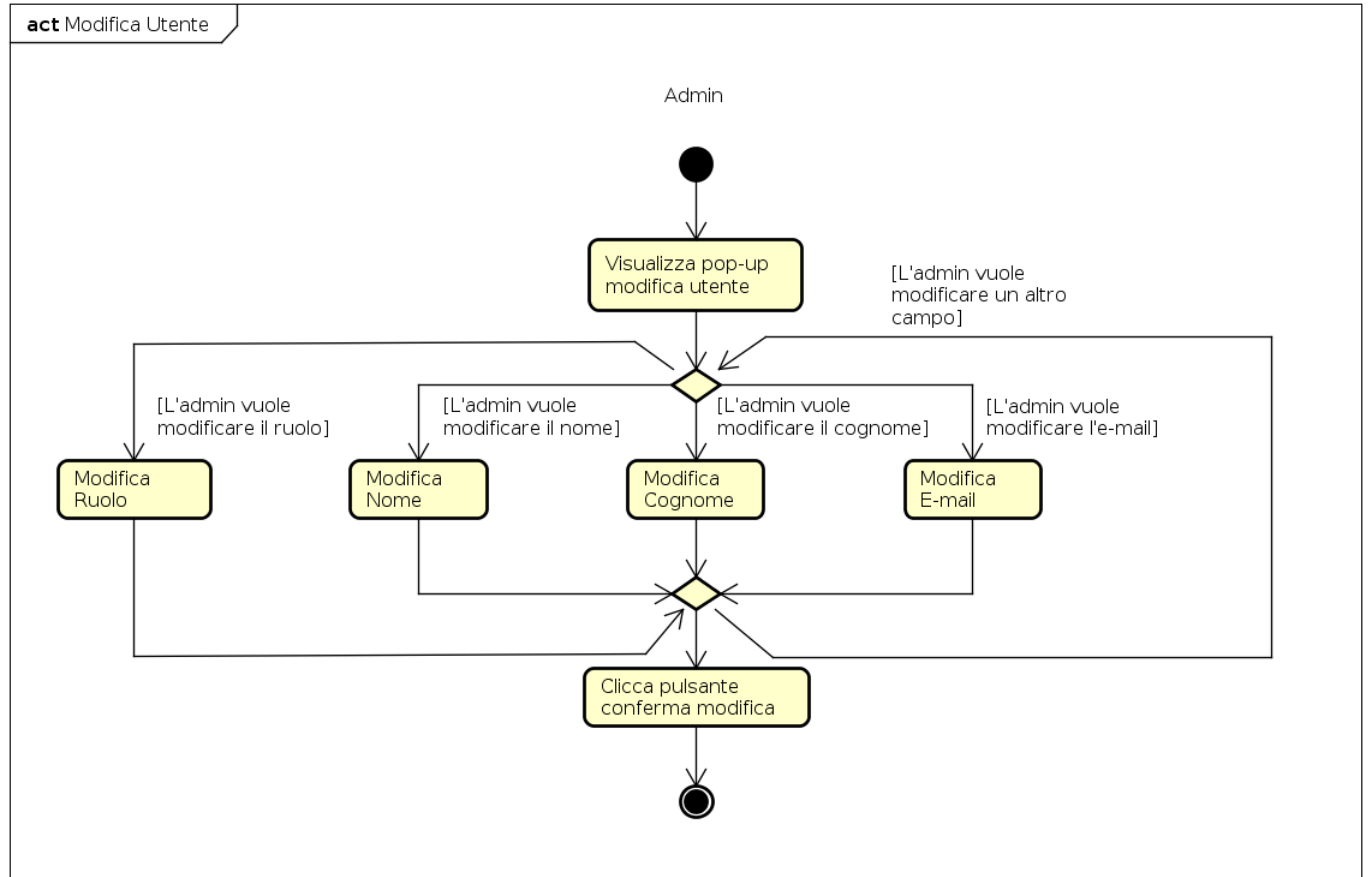


Figura 24: Diagramma di attività - Modifica Utente.

5.5.3.2 Modifica Utente Visualizzando la pagina per modificare un utente l' $admin_G$ potrà cambiare tutti i campi tranne lo $username_G$, che identifica univocamente l'utente, e la password. I campi sono:

- E-mail;
- Nome;
- Cognome;
- Ruolo.

Gestione Engine

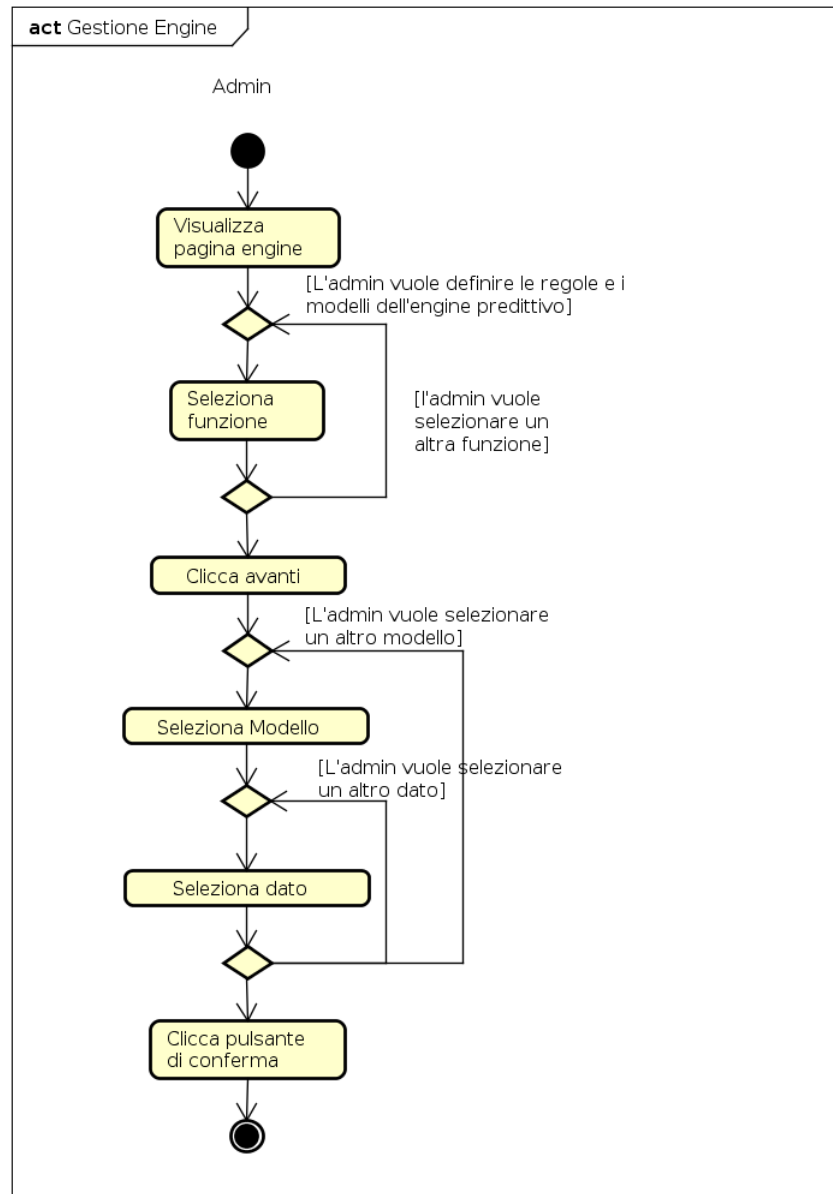


Figura 25: Diagramma di attività - Gestione Engine.

Nella pagina dedicata alla gestione dell'engine è presente un procedimento a due fasi che permetterà all' $admin_G$ di decidere prima le funzioni e poi i dati che potranno essere utilizzati dall'algoritmo e dagli $admin_G$ per la creazione di grafici e tabelle relative agli oggetti e ai modelli relativi ad essi.

Creazione Grafici

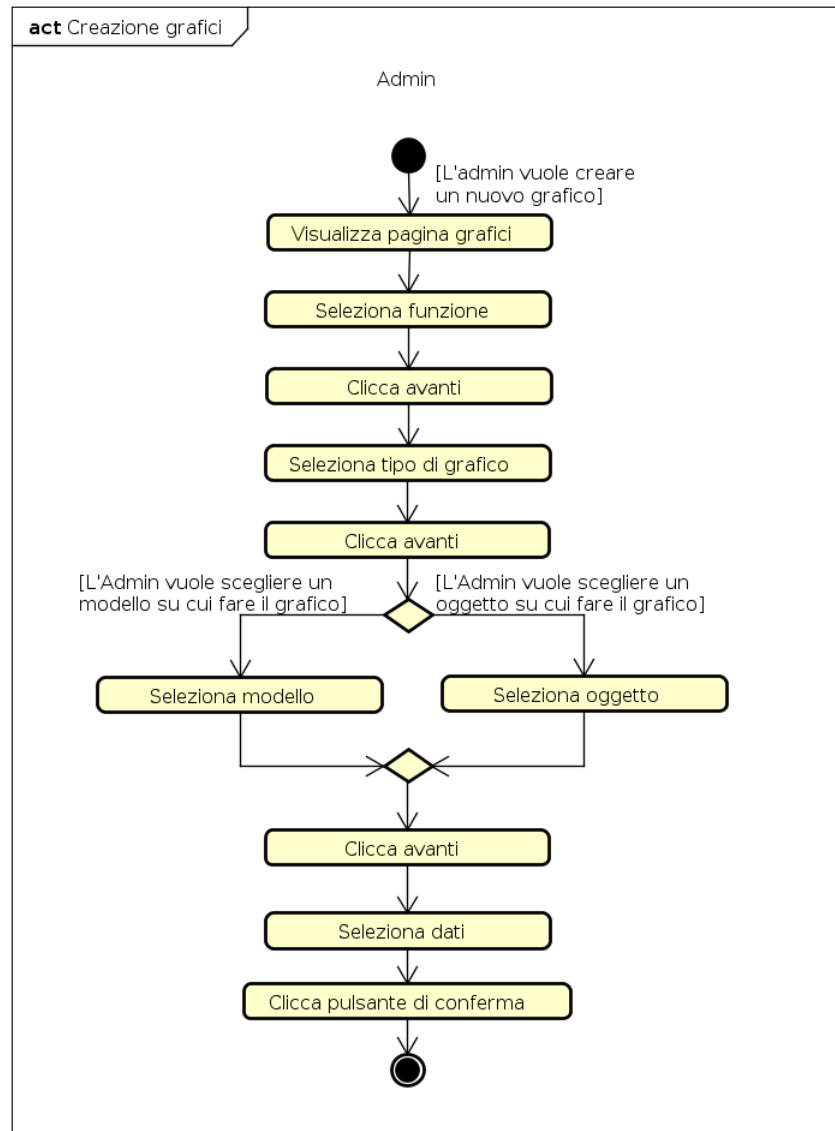


Figura 26: Diagramma di attività - Creazione Grafici.

Nella pagina dedicata alla creazione di grafici sarà possibile, attraverso un procedimento a quattro fasi, decidere in ordine la funzione da utilizzare, il tipo di grafico da creare, l'oggetto o il modello e quali dei suoi campi dati analizzare.

La pagina presenterà quindi 2 sezioni distinte:

- Una prima sezione più in alto che cambierà dinamicamente a seconda della fase in cui si trova il procedimento. Conterrà gli elementi da poter scegliere, tramite un meccanismo di *drag & drop*;
- La seconda sezione fa da contenitore per gli elementi già selezionati, in ogni fase del procedimento sarà possibile vedere tutte le scelte prese durante le fasi precedenti.

Super Admin

Gestione Company

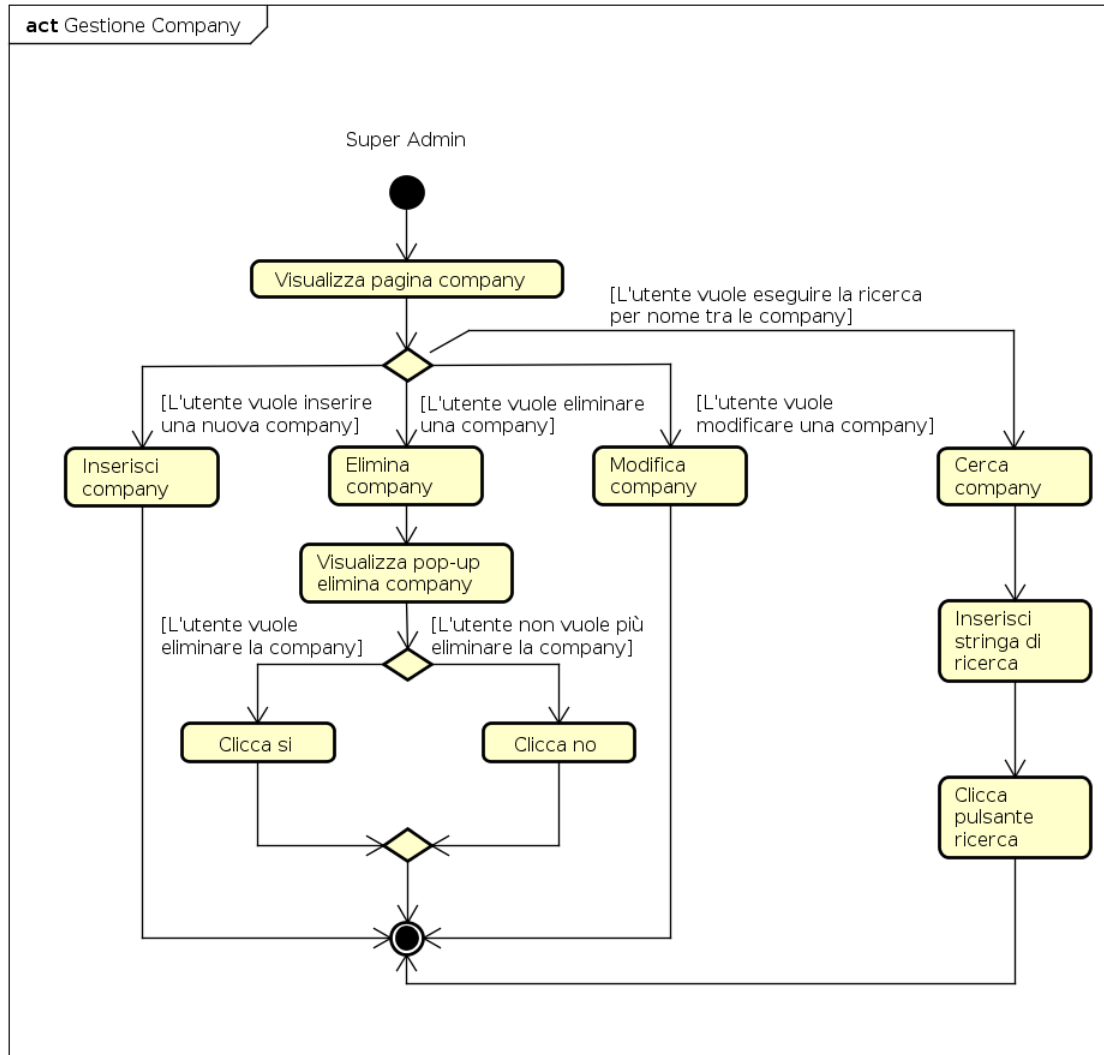


Figura 27: Diagramma di attività - Gestione Company.

Nella pagina dedicata alla gestione delle $company_G$ sarà presente una tabella con la lista di tutte le $company_G$ presenti nella piattaforma. Sarà inoltre possibile eseguire le seguenti operazioni:

- Inserire una nuova $company_G$;
- Eliminare una $company_G$ già esistente tramite l'apposito pulsante;
- Modificare una $company_G$ già esistente tramite l'apposito pulsante;
- Cercare una $company_G$ per nome: inserendo una stringa e premendo il pulsante di ricerca la tabella si aggiornerà con i risultati trovati.

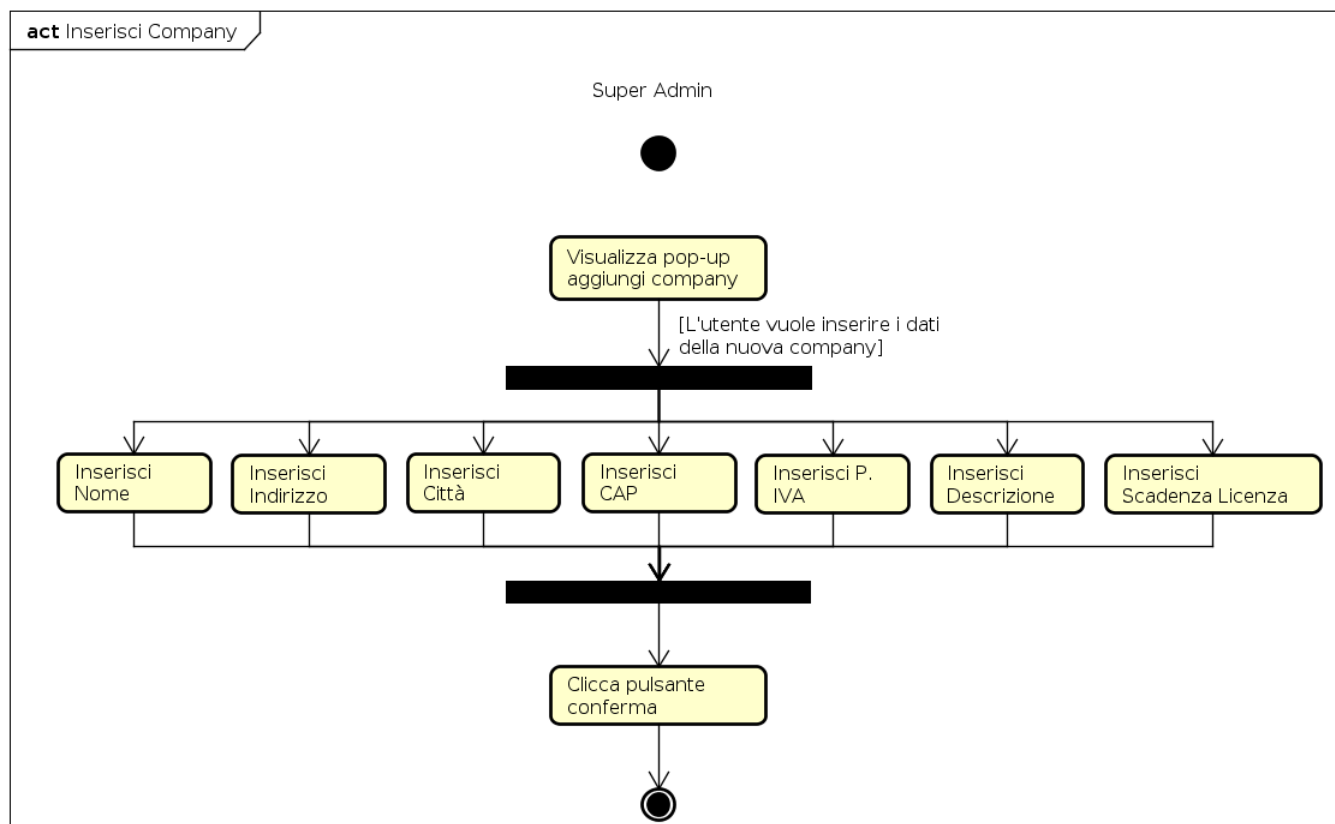


Figura 28: Diagramma di attività - Inserisci Company.

5.6.1.1 Inserisci Company Cliccando sul pulsante per l'inserimento di una $company_G$ verrà visualizzato il pop-up relativo che permetterà di inserire i campi obbligatori per l'aggiunta di una nuova $company_G$. I campi sono:

- Nome;
- Indirizzo;
- Città;
- CAP;
- P.IVA;
- Descrizione;
- Scadenza Licenza.

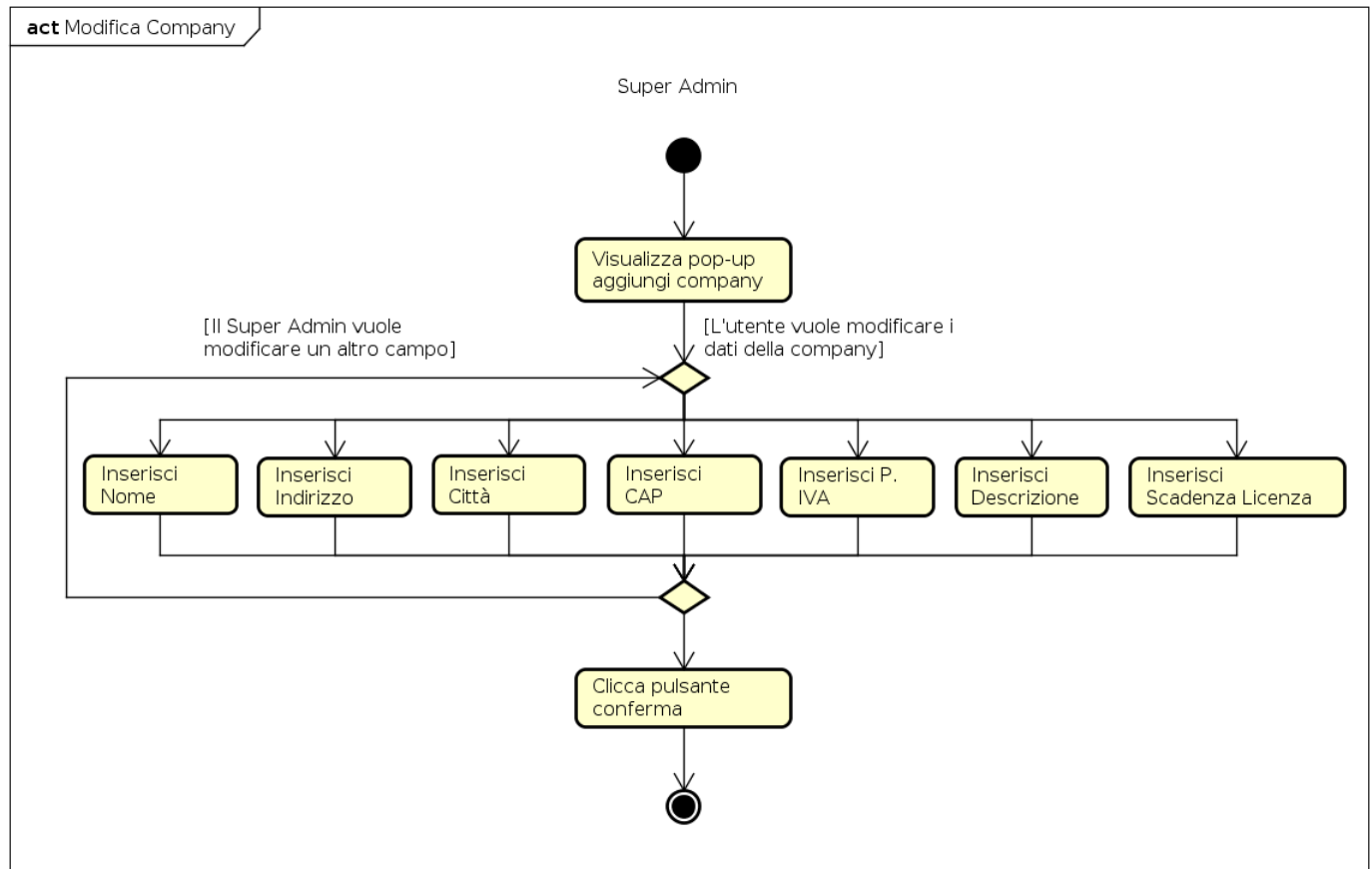


Figura 29: Diagramma di attività - Modifica Company.

5.6.1.2 Modifica Company Cliccando sul pulsante per la modifica della $company_G$ verrà visualizzato il pop-up relativo che permetterà di modificare i campi relativi alla $company_G$ selezionata. I campi sono:

- Nome;
- Indirizzo;
- Città;
- CAP;
- P.IVA;
- Descrizione;
- Scadenza Licenza.

Gestione Utenti

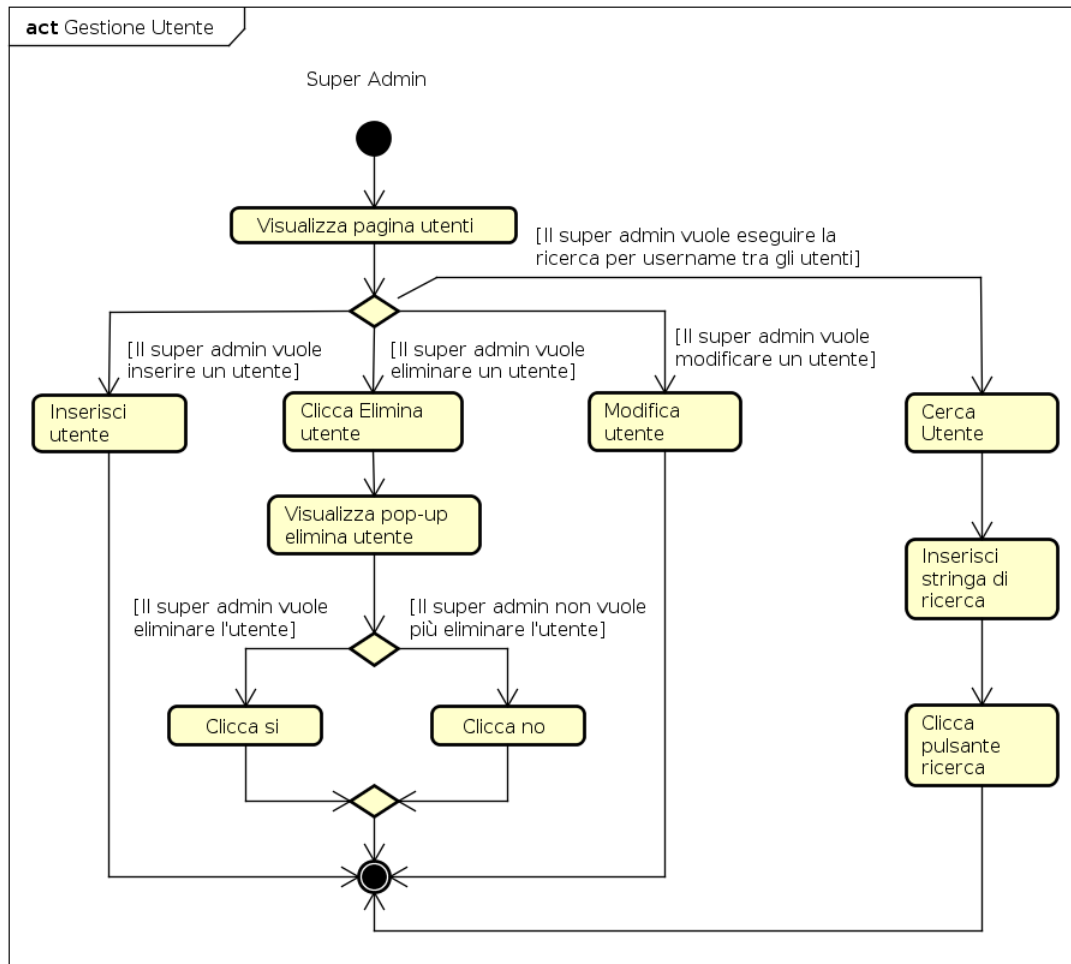


Figura 30: Diagramma di attività - Gestione Utenti.

Nella pagina dedicata alla visualizzazione degli utenti, il *super admin_G* visualizza una tabella contenente l'elenco degli utenti presenti nella piattaforma. Qui ha la possibilità di:

- Inserire un nuovo utente;
- Eliminare un utente esistente premendo l'apposito pulsante;
- Modificare un utente già esistente;
- Cercare un utente per username: inserendo una stringa e cliccando sul pulsante di ricerca, la tabella si aggiorna con i risultati trovati.

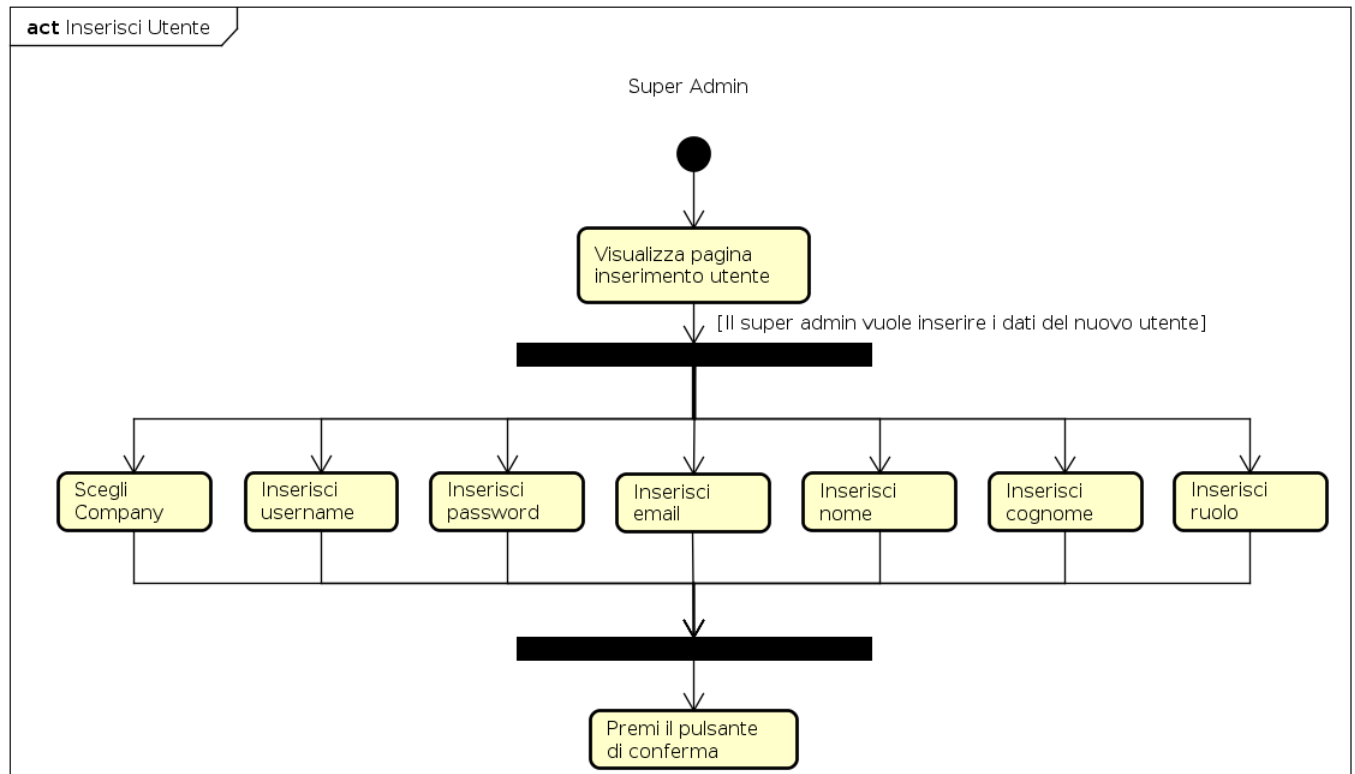


Figura 31: Diagramma di attività - Inserisci Utente.

5.6.2.1 Inserisci Utente Visualizzando la pagina per l'inserimento dell'utente il *super admin_G* potrà inserire tutti i campi obbligatori per aggiungere un nuovo utente alla piattaforma e inserirlo. I campi sono:

- *Company_G*;
- *Username_G*;
- Password;
- E-mail;
- Nome;
- Cognome;
- Ruolo.

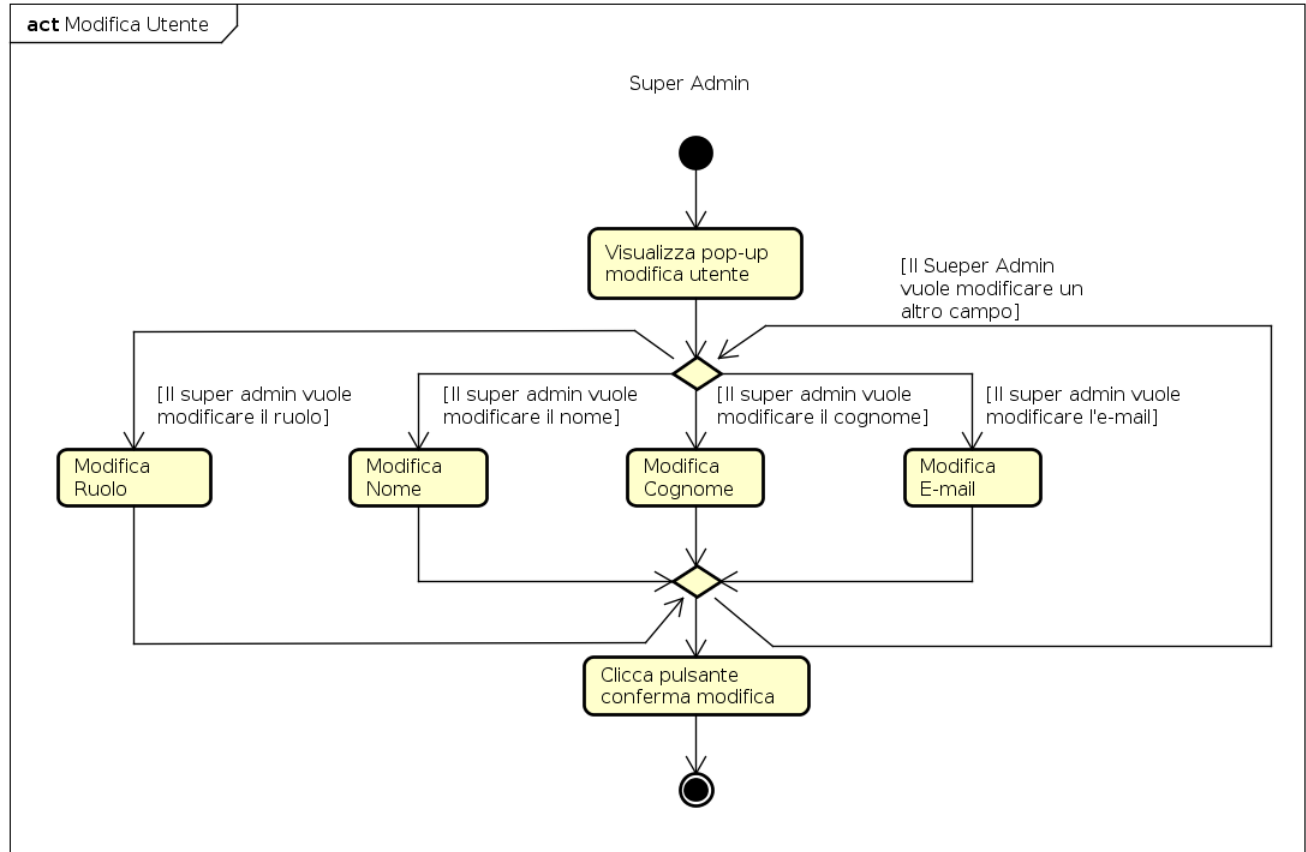


Figura 32: Diagramma di attività - Modifica Utente.

5.6.2.2 Modifica Utente Premendo il pulsante per la modifica apparirà l'apposito pop-up per modificare un utente. Il *super admin_G* potrà cambiare tutti i campi tranne lo *username_G*, che identifica univocamente l'utente, e la password. I campi sono:

- E-mail;
- Nome;
- Cognome;
- Ruolo.

Gestione Oggetti

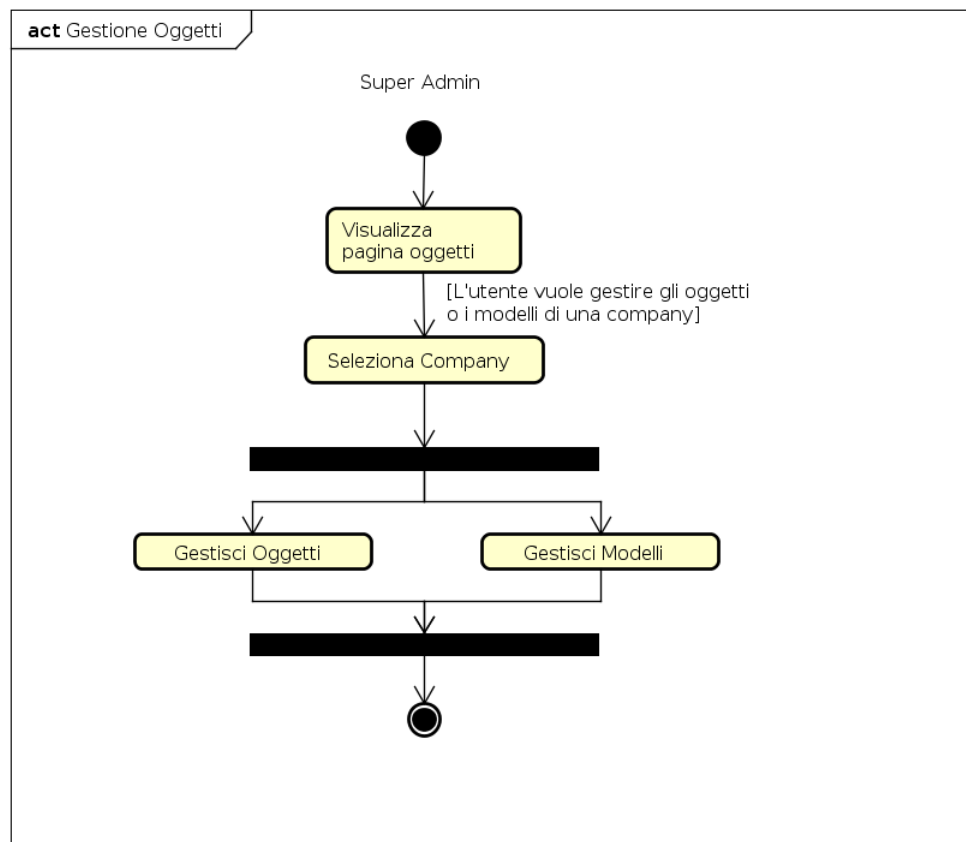


Figura 33: Diagramma di attività - Gestione Oggetti.

Nella pagina dedicata alla gestione degli oggetti saranno presenti due tabelle contenenti rispettivamente la lista di tutti gli oggetti presenti nella piattaforma e la lista di tutti i modelli di oggetti presenti nella piattaforma. Sarà possibile, attraverso l'apposita tabella, gestire gli oggetti o i modelli.

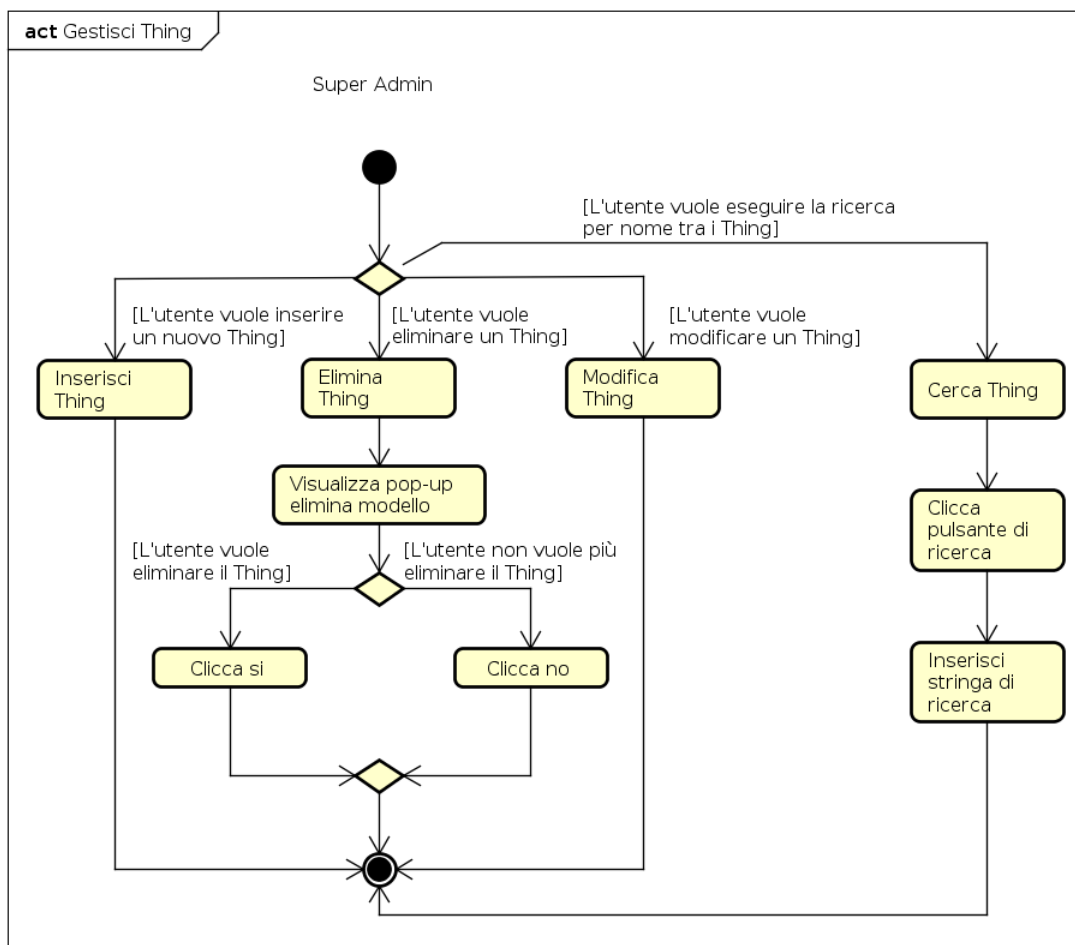


Figura 34: Diagramma di attività - Gestisci Thing.

5.6.3.1 Gestisci Thing Dalla tabella degli oggetti sarà possibile eseguire le seguenti operazioni:

- Inserire un nuovo $Thing_G$ all'interno della piattaforma;
- Eliminare un $Thing_G$ già presente nella piattaforma;
- Modificare un $Thing_G$ già presente nella piattaforma;
- Cercare un $Thing_G$ per nome: inserendo una stringa e cliccando sul pulsante di ricerca, la tabella si aggiorna con i risultati trovati.

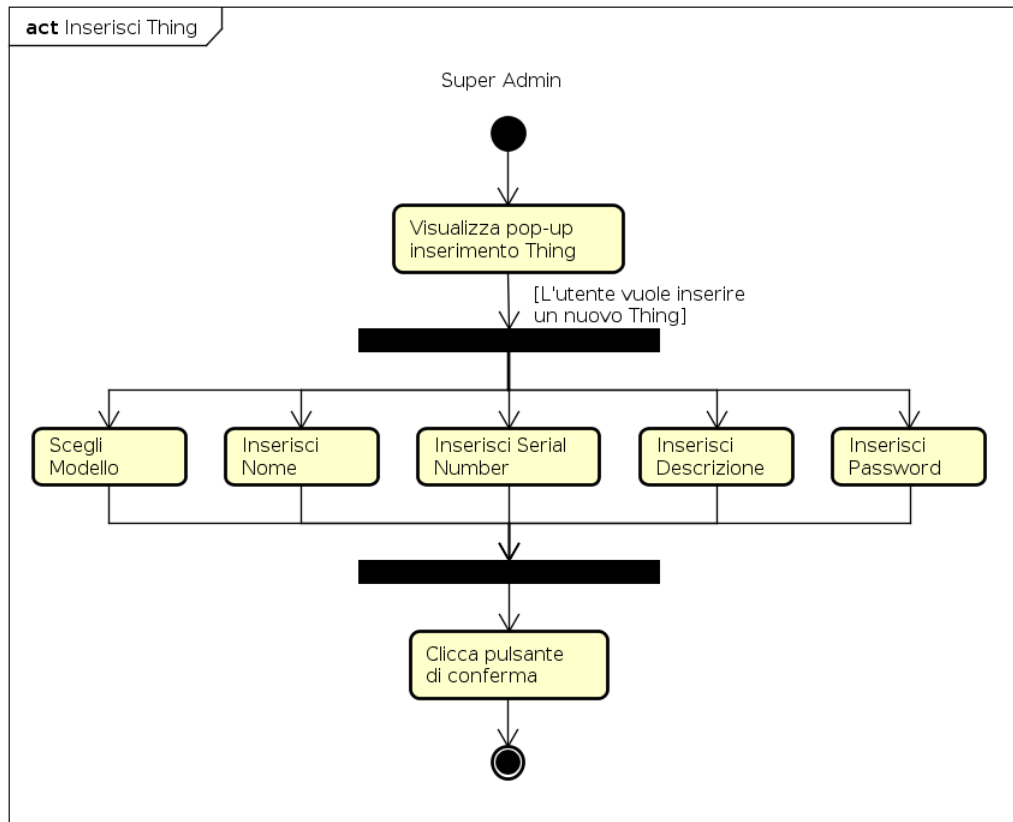


Figura 35: Diagramma di attività - Inserisci Thing.

5.6.3.1.1 Inserisci Thing Premendo l'apposito pulsante si aprirà un pop-up per l'inserimento di un nuovo *Thing_G* dove sarà possibile inserire i campi dati obbligatori per la sua creazione. I campi sono i seguenti:

- Modello;
- Nome;
- Password;
- Serial Number;
- Descrizione;

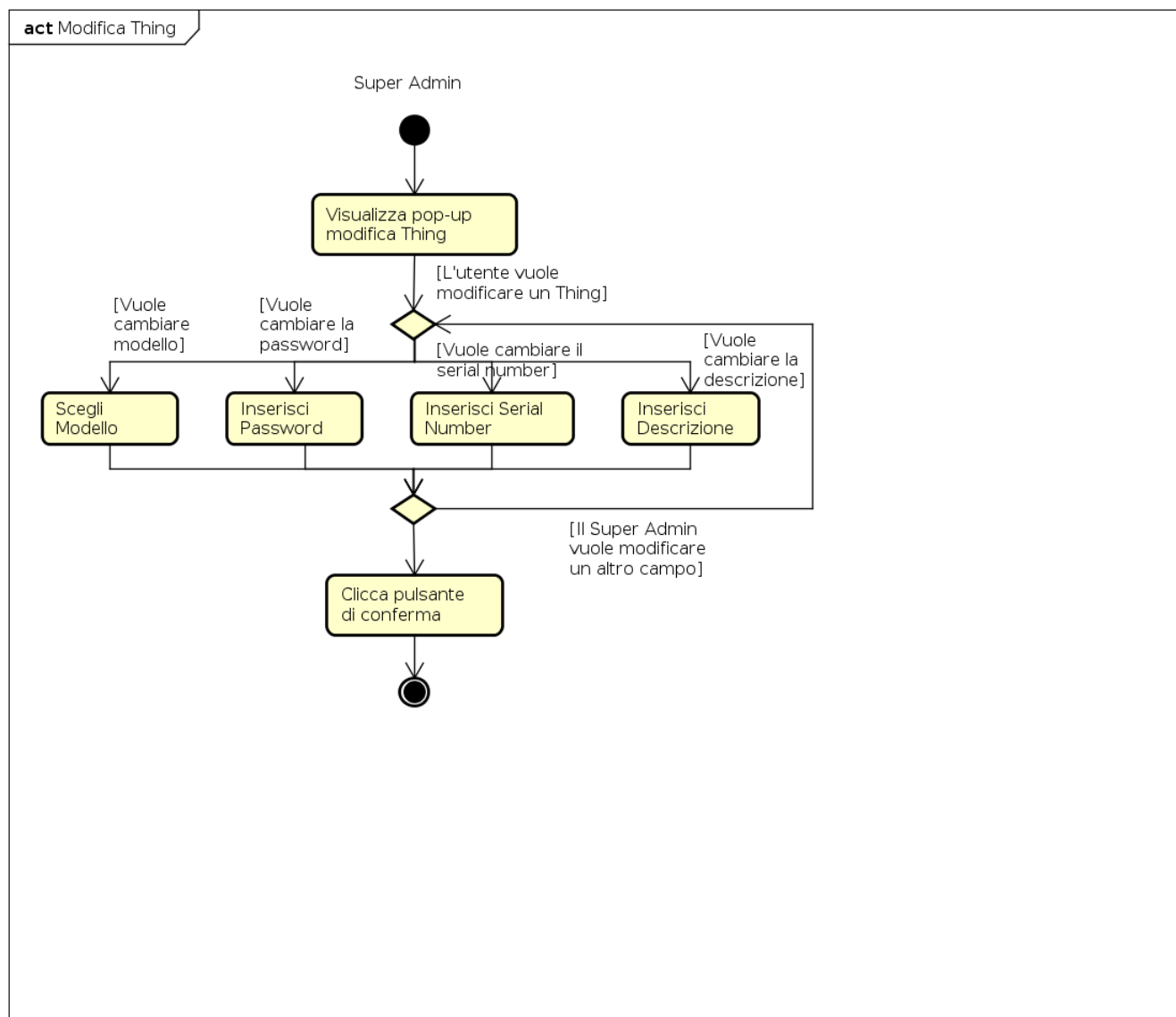


Figura 36: Diagramma di attività - Modifica Thing.

5.6.3.1.2 Modifica Thing Premendo l'apposito pulsante si aprirà un pop-up per la modifica del *Thing_G* selezionato dove sarà possibile modificare tutti i campi dati tranne il nome, che lo identifica univocamente. I campi sono i seguenti:

- Modello;
- Password;
- Serial Number;
- Descrizione;

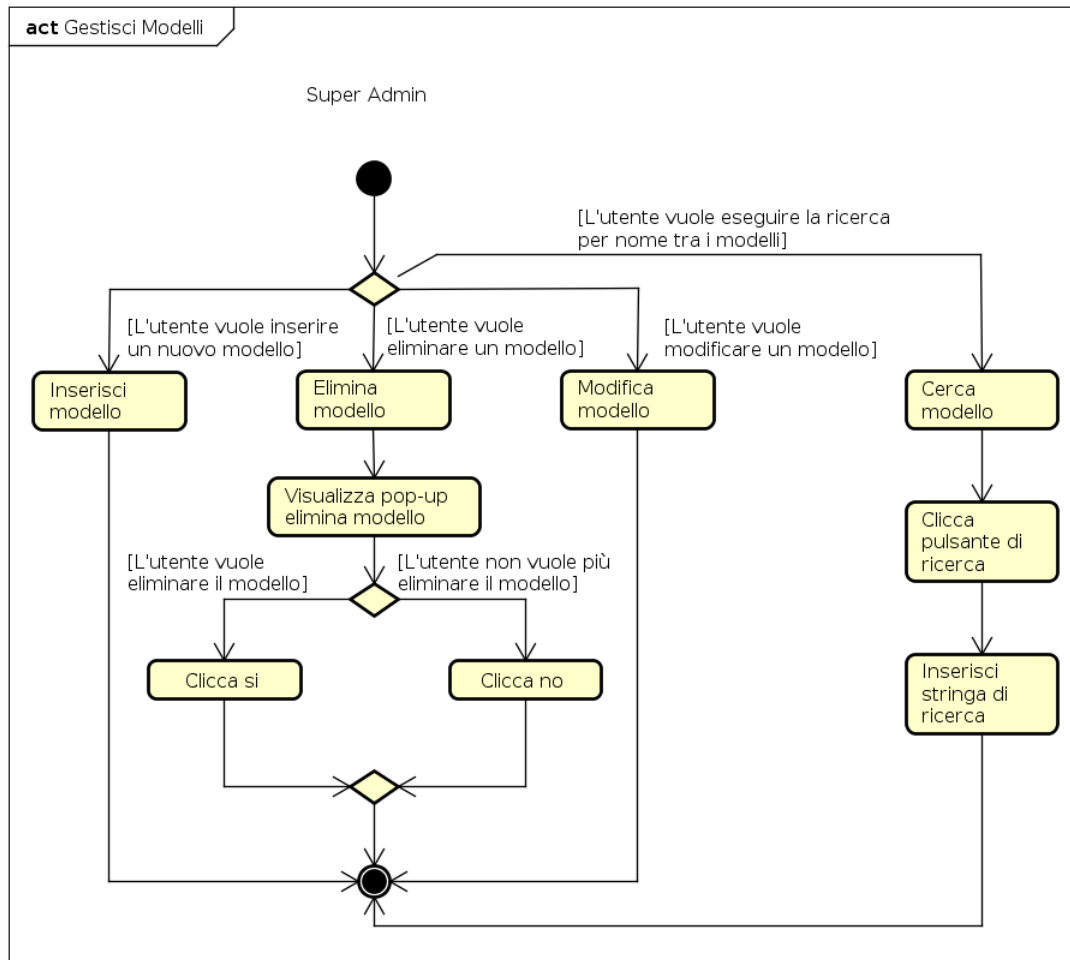


Figura 37: Diagramma di attività - Gestisci Modelli.

5.6.3.2 Gestisci Modelli Dalla tabella dei modelli sarà possibile eseguire le seguenti operazioni:

- Inserire un nuovo modello all'interno della piattaforma;
- Eliminare un modello già presente nella piattaforma;
- Modificare un modello già presente nella piattaforma;
- Cercare un modello per nome: inserendo una stringa e cliccando sul pulsante di ricerca, la tabella si aggiorna con i risultati trovati.

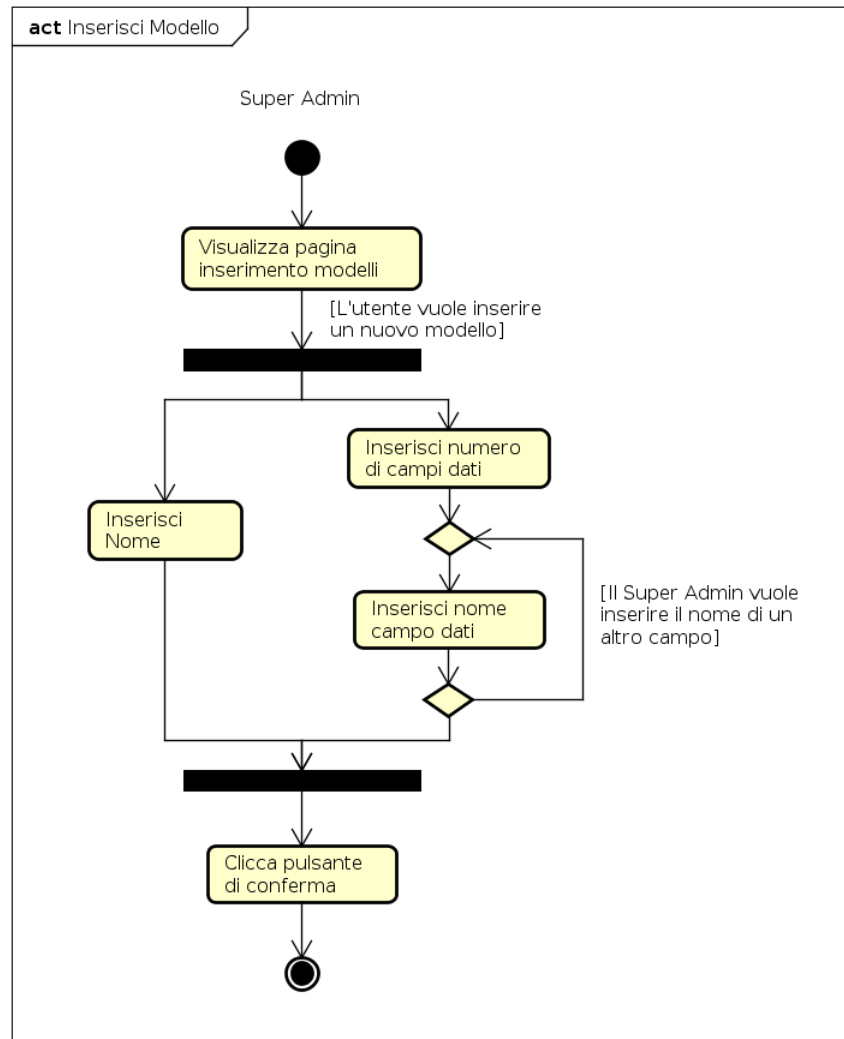


Figura 38: Diagramma di attività - Inserisci Modello.

5.6.3.2.1 Inserisci Modello Premendo l'apposito pulsante si aprirà la pagina per l'inserimento di un nuovo modello dove sarà possibile decidere un nome identificativo per il modello, il numero di dati che gli oggetti di quel determinato modello invieranno alla piattaforma e un nome per ognuno di questi campi dati.

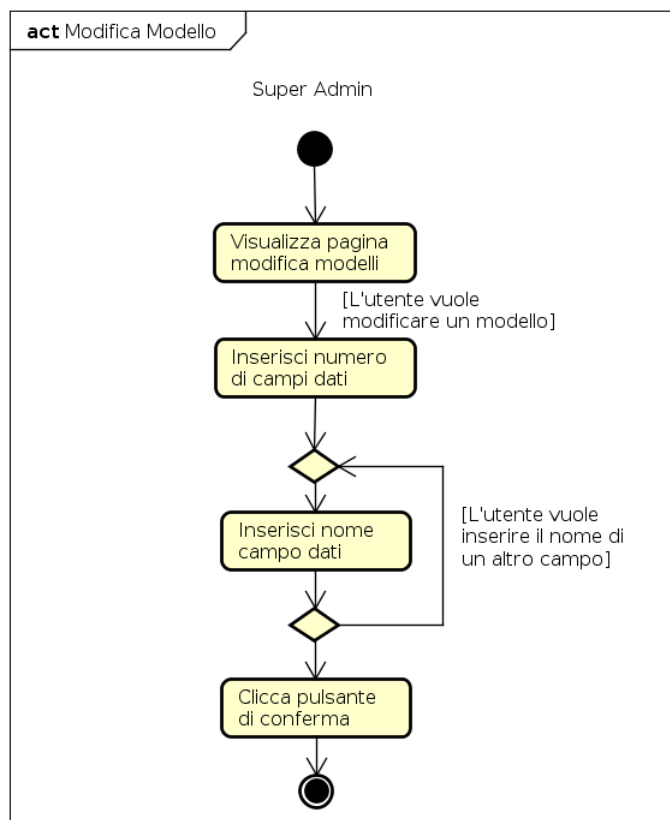


Figura 39: Diagramma di attività - Modifica Modello.

5.6.3.2.2 Modifica Modello Premendo l'apposito pulsante si aprirà la pagina per la modifica del modello selezionato dove sarà possibile modificare numero e nome dei dati che gli oggetti di quel determinato modello invieranno alla piattaforma. Non sarà invece possibile modificare il nome, che lo identifica univocamente.

Gestione Engine

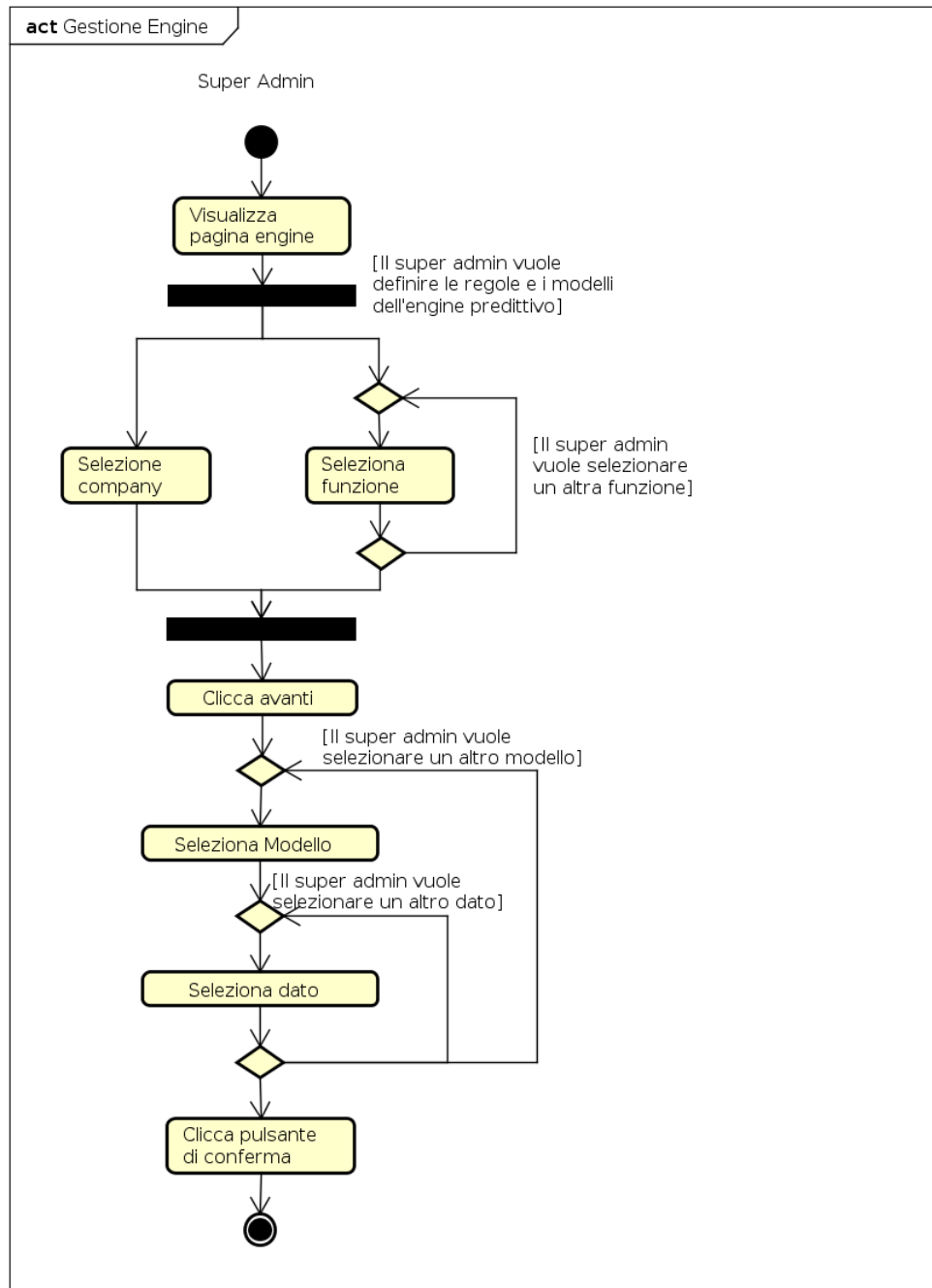


Figura 40: Diagramma di attività - Gestione Engine.

Nella pagina dedicata alla gestione dell'engine è presente un procedimento a due fasi che permetterà al *Super Admin_G* di decidere prima le funzioni e poi i dati che potranno essere utilizzati dall'algoritmo e dagli *Admin_G* per la creazione di grafici e tabelle relative agli oggetti e ai modelli relativi ad essi.



Design Pattern

Quando si parla di $Design\ Pattern_G$, si intende un insieme di soluzioni a problemi ricorrenti. Tali *best practice*_G possono essere applicata nell'attività di progettazione, in modo tale da renderne più agevole lo svolgimento. Per convenzione, i $Design\ Pattern_G$ si dividono in:

- **$Design\ Pattern_G$ architetturali:** pensati per definire e strutturare l'architettura del software ad un livello più elevato di astrazione;
- **$Design\ Pattern_G$ creazionali:** definiti per costruire un oggetto senza conoscere la loro implementazione (vanno quindi a nascondere i costruttori delle classi in cui si applicano);
- **$Design\ Pattern_G$ strutturali:** ideati per il riutilizzo di classi già esistenti, fornendone un'interfaccia specifica allo scopo;
- **$Design\ Pattern_G$ comportamentali:** sviluppati per consentire legami tra oggetti.

Si andrà a dettagliare in profondità ogni $Design\ Pattern_G$ citato in seguito in Appendice A.

Design Pattern Architetturali

MVC

- **Scopo dell'utilizzo:** si utilizza il pattern MVC per separare la *business logic*_G dalla rappresentazione grafica;
- **Contesto di utilizzo:** il pattern MVC_G viene utilizzato per poter sviluppare correttamente con Play Framework. Tale strumento, infatti, consente di costruire un'applicazione software solamente seguendo in pattern in esame.



Stime di fattibilità e bisogno di risorse

L'architettura sopra definita ha raggiunto un livello di dettaglio tale da poter fornire una stima sulla fattibilità e di bisogno di risorse.

Per quanto riguarda la scelta delle tecnologie da adottare per lo sviluppo del prodotto, si è visto che risultano adeguate a ricoprire le esigenze progettuali.

Gli strumenti scelti sono conosciuti dalla maggioranza dei membri del gruppo, che si impegneranno comunque ad approfondire le loro conoscenze a riguardo. Per riassumere, gli strumenti adottati sono i seguenti:

- **Play Framework**, per la stesura del codice;
- **Amazon Web Services**, che fornisce un pacchetto di *machine learning*_G molto ben strutturato e ideale per il prodotto che si andrà a sviluppare.

Tali strumenti potranno garantire, in unione con le tecnologie scelte, una realizzazione efficace di tutti gli aspetti architetturali.



Tracciamento

Tracciamento requisiti-componenti

Requisiti	Componenti
R0F1	app :: models, app :: controllers :: shared :: account, app :: controllers :: shared :: authentication, app :: view :: shared :: authentication
R0F1.1	app :: models, app :: controllers :: shared :: account, app :: controllers :: shared :: authentication, app :: view :: shared :: authentication
R0F1.1.1	app :: models, app :: controllers :: shared :: account, app :: controllers :: shared :: authentication, app :: view :: shared :: authentication
R0F1.1.2	app :: models, app :: controllers :: shared :: account, app :: controllers :: shared :: authentication, app :: view :: shared :: authentication
R0F1.1.3	app :: models, app :: controllers :: shared :: account, app :: controllers :: shared :: authentication, app :: view :: shared :: authentication
R0F2	app :: models, app :: controllers :: shared :: authentication, app :: controllers :: shared :: account, app :: view :: shared :: authentication
R1F2.1	app :: models, app :: controllers :: shared :: account, app :: controllers :: shared :: authentication, app :: view :: shared :: authentication
R0F2.2	app :: controllers :: shared :: account, app :: controllers :: shared :: authentication, app :: view :: shared :: authentication
R0F2.2.1	app :: controllers :: shared :: account, app :: controllers :: shared :: authentication, app :: view :: shared :: authentication
R1F3	app :: models, app :: controllers :: shared :: account, app :: view :: shared :: authentication
R1F3.1	app :: models, app :: controllers :: shared :: account, app :: controllers :: shared :: authentication, app :: view :: shared :: authentication
R1F3.1.1	app :: models, app :: controllers :: shared :: account, app :: view :: shared :: authentication
R1F3.2	app :: controllers :: shared :: account, app :: view :: shared :: authentication



R1F3.3	app :: controllers :: shared :: account, app :: view :: shared :: authentication,
R1F3.3.1	app :: controllers :: account, app :: views :: shared :: authentication
R1F4	app :: models, app :: controllers :: shared :: authentication, app :: views :: shared :: authentication
R1F4.1	app :: controllers :: shared :: authentication
R0F5	app :: models, app :: models :: daos :: user, app :: controllers :: shared :: account, app :: views :: shared :: account
R2F5.1	app :: models, app :: models :: daos :: user, app :: controllers :: shared :: account, app :: views :: shared :: account
R2F5.2	app :: models, app :: models :: daos :: user, app :: controllers :: shared :: account, app :: views :: shared :: account
R0F5.3	app :: models, app :: models :: daos :: user, app :: controllers :: shared :: account , app :: views :: shared :: account
R0F5.4	app :: models, app :: models :: daos :: user, app :: controllers :: shared :: account, app :: views :: shared :: account
R1F6	app :: models, app :: models :: daos :: user, app :: controllers :: shared :: account, app :: views :: shared :: account
R0F7	app :: models, app :: model :: daos :: company, app :: controllers :: superAdmin app :: views :: superAdmin
R0F8	app :: models, app :: model :: daos :: company, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.1	app :: models, app :: model :: daos :: company, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.1.1	app :: models app :: model :: daos :: company, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.1.2	app :: models app :: model :: daos :: company, app :: controllers :: superAdmin, app :: views :: superAdmin



R0F8.1.3	app :: models app :: model :: daos :: company, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.1.4	app :: models app :: model :: daos :: company, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.1.5	app :: models app :: model :: daos :: company, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.1.6	app :: models app :: model :: daos :: company, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.1.7	app :: models app :: model :: daos :: company, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.1.8	app :: models app :: model :: daos :: company, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.1	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.1.1	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.1.2	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.1.3	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.1.4	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.1.5	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin



R0F8.2.2	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.3	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.3.1	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.3.2	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.3.3	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.4	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.4.1	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.5	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R0F8.2.6	app :: models, app :: models :: daos :: thing, app :: controllers :: superAdmin, app :: views :: superAdmin
R1F8.3	app :: models, app :: models :: daos :: thing, app :: controllers :: admin, app :: views :: admin
R1F8.3.1	app :: models, app :: models :: daos :: thing, app :: controllers :: admin, app :: views :: admin
R1F8.3.1.1	app :: models, app :: models :: daos :: thing, app :: controllers :: admin, app :: views :: admin
R1F8.3.2	app :: models, app :: models :: daos :: thing, app :: controllers :: admin, app :: views :: admin
R0F8.4	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin



R0F8.4.1	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.1.1	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.1.2	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.1.3	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.1.4	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.1.5	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.1.6	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.2	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.2.1	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.2.2	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.2.3	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin



R0F8.4.2.4	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.3	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.3.1	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.3.2	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F8.4.4	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R1F8.4.5	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R1F8.4.6	app :: models, app :: models :: daos :: company, app :: models :: daos :: user, app :: controllers :: superAdmin, app :: controllers :: admin, app :: views :: superAdmin, app :: views :: admin
R0F9	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.1	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.1.1	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.1.2	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.1.2.1	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType



R0F9.1.2.2	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.2	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.2.1	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.2.2	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.2.2.1	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.2.2.2	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.3	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.4	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.5	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.6	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.7	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F9.7.1	app :: models, app :: models :: daos :: thingType app :: controllers :: superAdmin, app :: views :: SuperAdmin :: thingType
R0F10	app :: controllers :: shared :: administrator, app :: views :: superAdmin :: engine, app :: views :: admin :: engine



R0F10.1	app :: controllers :: shared :: administrator, app :: views :: superAdmin :: engine, app :: views :: admin :: engine
R0F10.2	app :: controllers :: shared :: administrator, app :: views :: superAdmin :: engine, app :: views :: admin :: engine
R0F10.3	app :: controllers :: shared :: administrator, app :: views :: superAdmin :: engine, app :: views :: admin :: engine
R0F11	app :: models, app :: controllers :: shared :: administrator, app :: controllers :: admin, app :: views :: admin :: chart
R0F11.1	app :: models, app :: controllers :: shared :: administrator, app :: controllers :: admin, app :: views :: admin :: chart
R0F11.2	app :: models, app :: controllers :: shared :: administrator, app :: controllers :: admin, app :: views :: admin :: chart
R0F11.3	app :: models, app :: controllers :: shared :: administrator, app :: controllers :: admin, app :: views :: admin :: chart
R0F11.3.1	app :: models, app :: controllers :: shared :: administrator, app :: controllers :: admin, app :: views :: admin :: chart
R0F11.3.2	app :: models, app :: controllers :: shared :: administrator, app :: controllers :: admin, app :: views :: admin :: chart
R0F11.3.3	app :: models, app :: controllers :: shared :: administrator, app :: controllers :: admin, app :: views :: admin :: chart
R0F12	app :: models, app :: models :: daos :: thing, app :: controllers :: admin, app :: controllers :: user, app :: views :: user :: thing
R0F13	app :: models, app :: models :: daos :: thing, app :: controllers :: admin, app :: controllers :: user, app :: views :: user :: thing
R0F13.1	app :: models, app :: models :: daos :: thing, app :: controllers :: admin, app :: controllers :: user, app :: views :: user :: thing



R0F14	app :: models, app :: models :: daos :: thing, app :: controllers :: admin, app :: controllers :: user, app :: views :: admin :: thing,
R0F14.1	app :: models, app :: models :: daos :: thing, app :: controllers :: admin, app :: controllers :: user, app :: views :: admin :: thing,
R0F15	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications
R0F15.1 app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications	
R0F15.1.1	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications
R0F15.1.2	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications
R0F15.1.3	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications
R0F15.1.4	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications
R0F15.1.5	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications
R0F15.1.6	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications
R0F15.2	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications
R0F15.2.1	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications
R0F15.2.2	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications



R0F15.2.3	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications
R0F15.2.4	app :: models, app :: models :: daos :: notification app :: controllers :: shared :: adminUser, app :: views :: shared :: adminUser :: notifications
R0F16	app :: models, app :: models :: daos :: thing, app :: controllers :: shared :: authentication
R0F16.1	app :: models, app :: models :: daos :: thing, app :: controllers :: shared :: authentication
R0F16.2	app :: models, app :: models :: daos :: thing, app :: controllers :: shared :: authentication
R0F17	app :: models, app :: models :: daos :: thing, app :: controllers :: shared :: authentication
R0F18	app :: models, app :: models :: daos :: thing, app :: controllers :: shared :: authentication
R1F19	app :: models, app :: models :: daos :: thing, app :: controllers :: admin :: thing, app :: controllers :: admin :: thingType, app :: views :: admin :: thing :: thingthingTypeDetails, app :: views :: admin :: thingType

R0F20	app :: views :: SuperAdmin :: Company :: CompanyManagement, app :: views :: SuperAdmin :: Thing :: ThingsSuperAdminManagement, app :: views :: SuperAdmin :: Thing :: NewModelSuperAdminManagement, app :: views :: SuperAdmin :: Thing :: EditModelSuperAdminManagement, app :: views :: SuperAdmin :: Engine :: EngineSuperAdminManagement, app :: views :: SuperAdmin :: User :: UserSuperAdminManagement, app :: views :: Admin :: Thing :: ThingsManagementDetails, app :: views :: Admin :: ThingModel :: ThingsModelManagementDetails, app :: views :: Admin :: Chart :: ChartsModelsManagement, app :: views :: Admin :: Engine :: EngineManagement, app :: views :: Admin :: Account :: UserManagement, app :: views :: User :: Thing :: ThingDetails, app :: views :: User :: ModelThing :: ThingsModelDetails, app :: views :: Shared :: AdminUser :: Things, app :: views :: Shared :: AdminUser :: ThingsModels, app :: views :: Shared :: AdminUser :: Notifications, app :: views :: Shared :: Authentication :: index, app :: views :: Shared :: Authentication :: logout, app :: views :: Shared :: Authentication :: PasswordRecover, app :: views :: Shared :: Authentication :: ResetPassword, app :: views :: Shared :: Account :: account, app :: views :: Shared :: Layout :: header, app :: views :: Shared :: Layout :: footer, app :: views :: Shared :: Layout :: layout, app :: views :: Shared :: Layout :: nav, 4.4.26 app :: views :: Shared :: Errors :: 404, app :: views :: Shared :: Errors :: 502
R0F21	app :: views :: admin :: chart
R0F22	app :: controllers :: shared :administrator
R0F23	app :: models :: daos
R0F23.1	app :: models :: daos
R0F23.2	app :: models :: daos
R0F24	app :: models, app :: controllers :: shared ::authentication, app :: controllers :: shared :administrator, app :: controllers :: shared :: account
R0F25	app :: models, app :: models :: daos :: thingType, app :: controllers :: admin, app :: controllers :: user, app :: views :: admin :: thingType
R0F26	app :: models, app :: models :: daos :: thingType, app :: models :: daos :: chart, app :: controllers :: admin, app :: controllers :: user, app :: views :: admin :: thingType



R0F27	app :: models, app :: models :: daos :: thing app :: controllers :: admin app :: views :: admin
R0F27.1	app :: models, app :: models :: daos :: thing app :: controllers :: admin app :: views :: admin
R0F28	app :: models, app :: controllers :: dao, app :: controllers :: admin, app :: views :: admin :: thingType,
R0F28.1	app :: models, app :: controllers :: dao, app :: controllers :: admin, app :: views :: admin :: thingType,
R0F28.1.1	app :: models, app :: controllers :: dao, app :: controllers :: admin, app :: views :: admin :: thingType,
R0F29	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F29.1	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F29.2	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F29.3	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F29.4	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F29.5	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F29.6	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F29.7	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F30	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company



R0F31	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F31.1	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F32	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F33	app :: models, app :: controllers :: dao :: company, app :: controllers :: superAdmin, app :: views :: superAdmin :: company
R0F34	app :: models, app: models :: dao, app :: controllers :: admin, app :: controllers :: user, app :: views :: shared :: adminUser :: things
R0F34.1	app :: models, app: models :: dao, app :: controllers :: admin, app :: controllers :: user, app :: views :: shared :: adminUser :: things
R0F35	app :: models, app: models :: dao, app :: controllers :: admin, app :: controllers :: user, app :: views :: shared :: adminUser :: thingsType
R0F35.1	app :: models, app: models :: dao, app :: controllers :: admin, app :: controllers :: user, app :: views :: shared :: adminUser :: thingsType
R0F36	app :: models, app :: controllers :: shared :: ad- minUser, app :: views :: shared :: adminUser :: Notifications
R0F36.1	app :: models, app :: controllers :: shared :: ad- minUser, app :: views :: shared :: adminUser :: Notifications
R0F37	app :: models, app :: models :: daos, app :: con- trollers :: shared :: administrator, app :: views :: superAdmin :: company
R0F38	app :: models, app :: models :: daos, app :: control- lers :: shared :: administrator, app :: views :: user :: thing
R0F38.1	app :: models, app :: models :: daos, app :: control- lers :: shared :: administrator, app :: views :: user :: chart



R0F39	app :: models, app :: models :: daos, app :: controllers :: shared :: administrator, app :: views :: admin :: engine
R0F39.1	app :: models, app :: models :: daos, app :: controllers :: shared :: administrator, app :: views :: admin :: engine
R0F39.2	app :: models, app :: models :: daos, app :: controllers :: shared :: administrator, app :: views :: admin :: engine
R0F39.3	app :: models, app :: models :: daos, app :: controllers :: shared :: administrator, app :: views :: admin :: engine
R0F40	app :: models, app :: models :: daos, app :: controllers :: shared :: administrator, app :: views :: admin :: engine

Tabella 2: Tracciamento requisiti-componenti.



Tracciamento componenti-requisiti

Componenti	Requisiti
Requisiti	
app :: models	R0F1, R0F1.1, R0F1.1.1, R0F1.1.2, R0F1.1.3, R0F2, R0F2.1, R1F3, R1F3.1, R1F3.1.1, R1F4, R0F5, R2F5.1, R2F5.2, R0F5.3, R0F5.4, R1F6, R0F7, R0F8, R0F8.1, R0F8.1.1, R0F8.1.2, R0F8.1.3, R0F8.1.4, R0F8.1.5, R0F8.1.6, R0F8.1.7, R0F8.1.8, R0F8.2, R0F8.2.1, R0F8.2.1.1, R0F8.2.1.2, R0F8.2.1.3, R0F8.2.1.4, R0F8.2.1.5, R0F8.2.2, R0F8.2.3, R0F8.2.3.1, R0F8.2.3.2, R0F8.2.3.3, R0F8.2.3.4, R0F8.2.4.1, R0F8.2.5, R0F8.2.6, R1F8.3, R1F8.3.1, R1F8.3.1.1, R1F8.3.2, R0F8.4, R0F8.4.1, R0F8.4.1.1, R0F8.4.1.2, R0F8.4.1.3, R0F8.4.1.4, R0F8.4.1.5, R0F8.4.1.6, R0F8.4.2, R0F8.4.2.1, R0F8.4.2.2, R0F8.4.2.3, R0F8.4.2.4, R0F8.4.3, R0F8.4.3.1, R0F8.4.3.2, R0F8.4.4, R1F8.4.5, R1F8.4.6, R0F9, R0F9.1.1, R0F9.1.2, R0F9.1.2.1, R0F9.1.2.2, R0F9.2, R0F9.2.1, R0F9.2.2, R0F9.2.2.1, R0F9.2.2.2, R0F9.3, R0F9.4, R0F9.5, R0F9.6, R0F9.7, R0F9.7.1, R0F11, R0F11.1, R0F11.2, R0F11.3, R0F11.3.1, R0F11.3.2, R0F11.3.3, R0F12, R0F13, R0F13.1, R0F14, R0F14.1, R0F15, R0F15.1, R0F15.1.1, R0F15.1.2, R0F15.1.3, R0F15.1.1.4, R0F15.1.1.5, R0F15.1.1.6, R0F15.2, R0F15.2.1, R0F15.2.2, R0F15.2.3, R0F15.2.4, R0F16, R0F16.1, R0F16.2, R0F17, R0F18, R1F19, R0F24, R0F25, R0F26, R0F27, R0F27.1, R0F28, R0F28.1, R0F28.1.1, R0F29, R0F29.1, R0F29.2, R0F29.3, R0F29.4, R0F29.5, R0F29.6, R0F29.7, R0F30, R0F31, R0F31.1, R0F32, R0F33
app :: models :: daos :: user	R0F5, R2F5.1, R2F5.2, R0F5.3, R0F5.4, R2F6, R0F8.4, R0F8.4.1, R0F8.4.1.1, R0F8.4.1.2, R0F8.4.1.3, R0F8.4.1.4, R0F8.4.1.5, R0F8.4.1.6, R0F8.4.2, R0F8.4.2.1, R0F8.4.2.2, R0F8.4.2.3, R0F8.4.2.4, R0F8.4.3, R0F8.4.3.1, R0F8.4.3.2, R0F8.4.4, R1F8.4.5, R1F8.4.6
app :: model :: daos :: company	R0F7, R0F8, R0F8.1, R0F8.1.1, R0F8.1.2, R0F8.1.3, R0F8.1.4, R0F8.1.5, R0F8.1.6, R0F8.1.7, R0F8.1.8, R0F8.4, R0F8.4.1, R0F8.4.1.1, R0F8.4.1.2, R0F8.4.1.3, R0F8.4.1.4, R0F8.4.1.5, R0F8.4.1.6, R0F8.4.2, R0F8.4.2.1, R0F8.4.2.2, R0F8.4.2.3, R0F8.4.2.4, R0F8.4.3, R0F8.4.3.1, R0F8.4.3.2, R0F8.4.4, R1F8.4.5, R1F8.4.6, R0F29, R0F29.1, R0F29.2, R0F29.3, R0F29.4, R0F29.5, R0F29.6, R0F29.7, R0F30, R0F31, R0F31.1, R0F32, R0F33



app :: models :: daos :: thing	R0F8.2, R0F8.2.1, R0F8.2.1.1, R0F8.2.1.2, R0F8.2.1.3, R0F8.2.1.4, R0F8.2.1.5, R0F8.2.2, R0F8.2.3, R0F8.2.3.1, R0F8.2.3.2, R0F8.2.3.3, R0F8.2.3.4, R0F8.2.4.1, R0F8.2.5, R0F8.2.6, R1F8.3, R1F8.3.1, R1F8.3.1.1, R1F8.3.2, R0F9, R0F9.1.1, R0F9.1.2, R0F9.1.2.1, R0F9.1.2.2, R0F9.2, R0F9.2.1, R0F9.2.2, R0F9.2.2.1, R0F9.2.2.2, R0F9.3, R0F9.4, R0F9.5, R0F9.6, R0F9.7, R0F9.7.1, R0F12, R0F13, R0F13.1, R0F14, R0F14.1, R0F16, R0F16.1, R0F16.2, R0F17, R0F18, R1F19, R0F27, R0F27.1
app :: models :: daos	R0F23, R0F23.1, R0F23.2, R0F28, R0F28.1, R0F28.1.1
app :: models :: daos :: chart	R0F26
app :: models :: daos :: thingType	R0F25, R0F26
app :: models :: daos :: notification	R0F15, R0F15.1, R0F15.1.1, R0F15.1.2, R0F15.1.3, R0F15.1.1.4, R0F15.1.1.5, R0F15.1.1.6, R0F15.2, R0F15.2.1, R0F15.2.2, R0F15.2.3, R0F15.2.4
app :: controllers :: shared :: account	R0F1, R0F1.1, R0F1.1.1, R0F1.1.2, R0F1.1.3, R0F2, R0F2.1, R0F2.2, R0F2.2.1, R1F3, R1F3.1, R1F3.1.1, R1F3.2, R1F3.3, R0F5, R2F5.1, R2F5.2, R0F5.3, R0F5.4, R1F6, R0F24
app :: controllers :: shared :: adminUser	R0F15, R0F15.1, R0F15.1.1, R0F15.1.2, R0F15.1.3, R0F15.1.1.4, R0F15.1.1.5, R0F15.1.1.6, R0F15.2, R0F15.2.1, R0F15.2.2, R0F15.2.3, R0F15.2.4
app :: controllers :: shared :: authentication	R0F1, R0F1.1, R0F1.1.1, R0F1.1.2, R0F1.1.3, R0F2, R0F2.1, R0F2.2, R0F2.2.1, R1F3, R1F3.1, R1F4, R1F4.1, R0F16, R0F16.1, R0F16.2, R0F17, R0F18, R0F24
app :: controllers :: shared :: administrator	R0F10, R0F10.1, R0F10.2, R0F10.3, R0F11, R0F11.1, R0F11.2, R0F11.3, R0F11.3.1, R0F11.3.2, R0F11.3.3, R0F22, R0F24



app :: controllers :: superAdmin	R0F7, R0F8, R0F8.1, R0F8.1.1, R0F8.1.2, R0F8.1.3, R0F8.1.4, R0F8.1.5, R0F8.1.6, R0F8.1.7, R0F8.1.8, R0F8.2, R0F8.2.1, R0F8.2.1.1, R0F8.2.1.2, R0F8.2.1.3, R0F8.2.1.4, R0F8.2.1.5, R0F8.2.2, R0F8.2.3, R0F8.2.3.1, R0F8.2.3.2, R0F8.2.3.3, R0F8.2.3.4, R0F8.2.4.1, R0F8.2.5, R0F8.2.6, R0F8.4, R0F8.4.1, R0F8.4.1.1, R0F8.4.1.2, R0F8.4.1.3, R0F8.4.1.4, R0F8.4.1.5, R0F8.4.1.6, R0F8.4.2, R0F8.4.2.1, R0F8.4.2.2, R0F8.4.2.3, R0F8.4.2.4, R0F8.4.3, R0F8.4.3.1, R0F8.4.3.2, R0F8.4.4, R1F8.4.5, R1F8.4.6, R0F9, R0F9.1.1, R0F9.1.2, R0F9.1.2.1, R0F9.1.2.2, R0F9.2, R0F9.2.1, R0F9.2.2, R0F9.2.2.1, R0F9.2.2.2, R0F9.3, R0F9.4, R0F9.5, R0F9.6, R0F9.7, R0F9.7.1, R0F29, R0F29.1, R0F29.2, R0F29.3, R0F29.4, R0F29.5, R0F29.6, R0F29.7, R0F30, R0F31, R0F31.1, R0F32, R0F33
app :: controllers :: user	R0F12, R0F13, R0F13.1, R0F14, R0F14.1, R0F25, R0F26
app :: controllers :: admin	R1F8.3, R1F8.3.1, R1F8.3.1.1, R1F8.3.2, R0F8.4, R0F8.4.1, R0F8.4.1.1, R0F8.4.1.2, R0F8.4.1.3, R0F8.4.1.4, R0F8.4.1.5, R0F8.4.1.6, R0F8.4.2, R0F8.4.2.1, R0F8.4.2.2, R0F8.4.2.3, R0F8.4.2.4, R0F8.4.3, R0F8.4.3.1, R0F8.4.3.2, R0F8.4.4, R1F8.4.5, R1F8.4.6, R0F11, R0F11.1, R0F11.2, R0F11.3, R0F11.3.1, R0F11.3.2, R0F11.3.3, R0F12, R0F13, R0F13.1, R0F14, R0F14.1, R0F25, R0F26, R0F27, R0F27.1, R0F28, R0F28.1, R0F28.1.1
app :: controllers :: admin :: thing	R1F19
app :: controllers :: admin :: thingType	R1F19, R0F25, R0F26
app :: view :: shared :: authentication	R0F1, R0F1.1, R0F1.1.1, R0F1.1.2, R0F1.1.3, R0F2, R0F2.1, R0F2.2, R0F2.2.1, R1F3, R1F3.1, R1F3.1.1, R1F3.2, R1F3.3, R1F4
app :: view :: shared :: notification	R0F15, R0F15.1, R0F15.1.1, R0F15.1.2, R0F15.1.3, R0F15.1.1.4, R0F15.1.1.5, R0F15.1.1.6, R0F15.2, R0F15.2.1, R0F15.2.2, R0F15.2.3, R0F15.2.4
app :: view :: shared :: account	R0F5, R2F5.1, R2F5.2, R0F5.3, R0F5.4, R1F6



app :: view :: superAdmin	R0F7, R0F8, R0F8.1, R0F8.1.1, R0F8.1.2, R0F8.1.3, R0F8.1.4, R0F8.1.5, R0F8.1.6, R0F8.1.7, R0F8.1.8, R0F8.2, R0F8.2.1, R0F8.2.1.1, R0F8.2.1.2, R0F8.2.1.3, R0F8.2.1.4, R0F8.2.1.5, R0F8.2.2, R0F8.2.3, R0F8.2.3.1, R0F8.2.3.2, R0F8.2.3.3, R0F8.2.3.4, R0F8.2.4.1, R0F8.2.5, R0F8.2.6, R0F8.4, R0F8.4.1, R0F8.4.1.1, R0F8.4.1.2, R0F8.4.1.3, R0F8.4.1.4, R0F8.4.1.5, R0F8.4.1.6, R0F8.4.2, R0F8.4.2.1, R0F8.4.2.2, R0F8.4.2.3, R0F8.4.2.4, R0F8.4.3, R0F8.4.3.1, R0F8.4.3.2, R0F8.4.4, R1F8.4.5, R1F8.4.6
app :: views :: superAdmin :: thingType	R0F9, R0F9.1.1, R0F9.1.2, R0F9.1.2.1, R0F9.1.2.2, R0F9.2, R0F9.2.1, R0F9.2.2, R0F9.2.2.1, R0F9.2.2.2, R0F9.3, R0F9.4, R0F9.5, R0F9.6, R0F9.7, R0F9.7.1
app :: views :: superAdmin :: company	R0F29, R0F29.1, R0F29.2, R0F29.3, R0F29.4, R0F29.5, R0F29.6, R0F29.7, R0F30, R0F31, R0F31.1, R0F32, R0F33
app :: view :: admin	R1F8.3, R1F8.3.1, R1F8.3.1.1, R1F8.3.2, R0F8.4, R0F8.4.1, R0F8.4.1.1, R0F8.4.1.2, R0F8.4.1.3, R0F8.4.1.4, R0F8.4.1.5, R0F8.4.1.6, R0F8.4.2, R0F8.4.2.1, R0F8.4.2.2, R0F8.4.2.3, R0F8.4.2.4, R0F8.4.3, R0F8.4.3.1, R0F8.4.3.2, R0F8.4.4, R1F8.4.5, R1F8.4.6, R0F27, R0F27.1
app :: view :: superAdmin :: engine	R0F10, R0F10.1, R0F10.2, R0F10.3
app :: view :: admin :: engine	R0F10, R0F10.1, R0F10.2, R0F10.3
app :: view :: admin :: chart	R0F11, R0F11.1, R0F11.2, R0F11.3, R0F11.3.1, R0F11.3.2, R0F11.3.3, R0F21
app :: view :: admin :: thing :: thingTypeDetails	R1F19
app :: view :: admin :: thingType	R1F19, R0F28, R0F28.1, R0F28.1.1
app :: view :: user :: thing	R0F12, R0F13, R0F13.1, R0F14, R0F14.1
app :: controllers :: Shared :: Account :: ControllerAccount	R0F24, R0F5, R2F5.1, R2F5.2, R0F5.3, R0F5.4, R1F6
app :: controllers :: Shared :: Account :: ControllerPasswordRecover	R1F3.2, R1F3, R1F3.1, R1F3.1.1
app :: controllers :: Shared :: Account :: ControllerResetPassword	R1F3.3, R1F3.3.1
app :: controllers :: Shared :: AdminUser :: ControllerNotification	R0F15, R0F15.1, R0F15.1.1, R0F15.1.2, R0F15.1.3, R0F15.1.4, R0F15.1.5, R0F15.1.6, R0F15.2, R0F15.2.1, R0F15.2.2, R0F15.2.3, R0F15.2.4, R0F36, R0F36.1



app :: controllers :: Shared :: Administrator :: ControllerEngineManagement	R0F10.1, R0F10.2, R0F10.3, R0F10, R0F37, R0F39
app :: controllers :: Shared :: Administrator :: ControllerUserManagement	R0F8.4, R0F8.4.1, R0F8.4.2, R1F8.4.3, R1F8.4.4, R0F38, R0F38.1, R0F38.2, R0F38.3, R0F38.4, R0F38.4.1, R0F38.5, R0F8.4.1.1, R0F8.4.1.2, R0F8.4.1.3, R0F8.4.1.4, R0F8.4.1.5, R0F8.4.1.6, R0F8.4.2.1, R0F8.4.2.2, R0F8.4.2.3, R0F8.4.2.4, R0F8.4.2.5
app :: controllers :: Shared :: Authentication :: ControllerAuthentication	R0F1, R0F1.1, R0F1.1.1, R0F1.1.2, R0F2, R1F2.1, R1F4
app :: views :: Admin :: Account :: UserManagement	R0F8.4, R0F8.4.1, R0F8.4.2, R1F8.4.3, R1F8.4.4, R0F20, R0F27, R0F27.1, R0F27.2, R0F27.3, R0F27.3.1, R0F27.4, R0F38, R0F38.1, R0F38.2, R0F38.3, R0F38.4, R0F38.4.1, R0F38.5, R0F8.4.1.1, R0F8.4.1.2, R0F8.4.2.2, R0F8.4.1.3, R0F8.4.1.4, R0F8.4.1.5, R0F8.4.1.6, R0F8.4.2.1, R0F8.4.2.3, R0F8.4.2.4, R0F8.4.2.5
app :: views :: Admin :: Chart :: ChartsModelsManagement	R0F11, R0F11.1, R0F11.2, R0F11.3, R0F11.3.1, R0F11.3.2, R0F11.3.3, R0F20
app :: views :: Admin :: Engine :: EngineManagement	R0F10, R0F10.1, R0F10.2, R0F10.3, R0F20
app :: views :: Admin :: Thing :: ThingsManagementDetails	R1F8.3, R1F8.3.1, R1F8.3.1.1, R1F8.3.2, R0F12, R0F13, R0F20
app :: views :: Admin :: ThingModel :: ThingsModelManagementDetails	R1F19, R0F20, R0F25, R0F26, R0F28.1
app :: views :: Shared :: Account :: account	R0F5, R0F5.3, R0F5.4, R2F5.1, R2F5.2, R1F6, R0F20
app :: views :: Shared :: AdminUser :: Notifications	R0F15, R0F15.1, R0F15.1.1, R0F15.1.2, R0F15.1.3, R0F15.1.4, R0F15.1.5, R0F15.1.6, R0F15.2, R0F15.2.1, R0F15.2.2, R0F15.2.3, R0F15.2.4, R0F20, R0F36, R0F36.1
app :: views :: Shared :: AdminUser :: Things	R0F20, R0F12, R0F34, R0F34.1,
app :: views :: Shared :: AdminUser :: ThingsModels	R0F13, R0F14, R0F20, R0F25, R0F28, R0F28.1, R0F28.1.1, R0F35, R0F35.1
app :: views :: Shared :: Authentication :: PasswordRecover	R1F3, R1F3.1, R1F3.1.1, R0F20
app :: views :: Shared :: Authentication :: ResetPassword	R1F3.3, R1F3.3.1, R0F20
app :: views :: Shared :: Authentication :: index	R0F1, R0F1.1, R0F1.1.1, R0F1.1.2, R0F2, R1F2.1, R0F20



app :: views :: Shared :: Authentication :: logout	R1F4,R0F20,
app :: views :: Shared :: Errors :: 404	R0F20
app :: views :: Shared :: Errors :: 502	R0F20
app :: views :: Shared :: Layout :: footer	R0F20
app :: views :: Shared :: Layout :: header	R0F20
app :: views :: Shared :: Layout :: layout	R0F20
app :: views :: Shared :: Layout :: nav	R0F20
app :: views :: SuperAdmin :: Company :: CompanyManagement	R0F7, R0F8, R0F8.1, R0F8.1.1, R0F8.1.2, R0F8.1.3, R0F8.1.4, R0F8.1.5, R0F8.1.6, R0F8.1.7, R0F20, R0F29, R0F29.1, R0F29.2, R0F29.3, R0F29.4, R0F29.5, R0F29.6, R0F29.7, R0F30, R0F31, R0F31.1, R0F32, R0F33
app :: views :: SuperAdmin :: Engine :: EngineSuperAdminManagement	R0F10, R0F10.1, R0F10.2, R0F10.3, R0F20, R0F37, R0F39
app :: views :: SuperAdmin :: Thing :: EditModelSuperAdminManagement	R0F9.2, R0F9.2.1, R0F9.2.2, R0F9.2.2.1, R0F9.2.2.2, R0F9.4, R0F20
app :: views :: SuperAdmin :: Thing :: NewModelSuperAdminManagement	R0F9, R0F9.1, R0F9.1.1, R0F9.1.2, R0F9.1.2.1, R0F9.1.2.2, R0F9.3, R0F20
app :: views :: SuperAdmin :: Thing :: ThingsSuperAdminManagement	R0F8.2, R0F8.2.1, R0F8.2.1.1, R0F8.2.1.2, R0F8.2.1.3, R0F8.2.1.4, R0F8.2.1.5, R0F8.2.2, R0F8.2.3, R0F8.2.3.1, R0F8.2.3.2, R0F8.2.3.3, R0F8.2.3.4, R0F8.2.3.5, R0F8.2.4, R0F8.2.4.1, R0F8.2.5, R0F8.2.6, R0F9.5, R0F9.6, R0F9.7, R0F9.7.1, R0F20
app :: views :: SuperAdmin :: User :: UserSuperAdminManagement	R0F8.4, R0F8.4.1, R0F8.4.2, R1F8.4.3, R1F8.4.4, R0F20, R0F8.4.1.1, R0F8.4.1.2, R0F8.4.1.3, R0F8.4.1.4, R0F8.4.1.5, R0F8.4.1.6, R0F8.4.2.1, R0F8.4.2.2, R0F8.4.2.3, R0F8.4.2.4, R0F8.4.2.5
app :: views :: User :: ModelThing :: ThingsModelDetails	R0F20, R0F26
app :: views :: User :: Thing :: ThingDetails	R0F20
app::controllers::Admin::Chart::ControllerChart	R0F11, R0F11.1, R0F11.2, R0F11.3.1, R0F11.3.2, R0F11.3.3, R0F11.3
app::controllers::Admin::Thing::ControllerThingManagement	R0F12, R0F13, R1F8.3, R1F8.3.1, R1F8.3.1.1, R1F8.3.2, R0F14, R1F19, R0F34, R0F34.1
app::controllers::Admin::ThingModel::ControllerThingsModelManagement	R1F19, R0F35, R0F26, R0F28, R0F28.1.1



app::controllers::SuperAdmin::Company::ControllerCompanyManagement	R0F7, R0F8, R0F8.1, R0F8.1.1, R0F8.1.2, R0F8.1.3, R0F8.1.4, R0F8.1.5, R0F8.1.6, R0F8.1.7, R0F29, R0F29.1, R0F29.2, R0F29.3, R0F29.4, R0F29.5, R0F29.6, R0F29.7, R0F30, R0F31, R0F31.1, R0F32, R0F33
app::controllers::SuperAdmin::Thing::ControllerSuperAdminThingManagement	R0F9, R0F9.1, R0F9.1.1, R0F9.1.2, R0F9.1.2.1, R0F9.1.2.2, R0F9.2, R0F9.2.1, R0F9.2.2, R0F9.2.2.1, R0F9.2.2.2, R0F9.3, R0F9.4, R0F8.2, R0F8.2.1, R0F8.2.1.1, R0F8.2.1.2, R0F8.2.1.3, R0F8.2.1.4, R0F8.2.1.5, R0F8.2.2, R0F8.2.3, R0F8.2.3.1, R0F8.2.3.2, R0F8.2.3.3, R0F8.2.4, R0F8.2.4.1, R0F8.2.5, R0F8.2.6, R0F9.5, R0F9.6, R0F9.7, R0F9.7.1
app::controllers::User::Thing::ControllerThing	R0F12, R0F13, R0F14, R0F34, R0F34.1
app::controllers::User::ThingModel::ControllerThingsModel	R0F35, R0F35.1, R0F25, R0F26
app::models::ModelAccount::Account	R0F1, R0F1.1, R0F1.1.1, R0F1.1.2, R0F2, R1F2.1, R1F3, R1F3.1, R1F3.1.1, R1F3.2, R1F4, R0F5, R2F5.1, R2F5.2, R0F5.3, R0F5.4, R1F6, R0F8.4, R0F8.4.1, R0F8.4.2, R1F8.4.3, R1F8.4.4, R0F24, R0F27, R0F27.1, R0F27.2, R0F27.3, R0F27.3.1, R0F27.4, R0F38, R0F38.1, R0F38.2, R0F38.3, R0F38.4, R0F38.4.1, R0F38.5
app::models::ModelChart::Charts	R0F11, R0F11.1, R0F11.2, R0F11.3, R0F11.3.1, R0F11.3.2, R0F11.3.3
app::models::ModelCompany::Company	R0F7, R0F8, R0F8.1, R0F8.1.1, R0F8.1.2, R0F8.1.3, R0F8.1.4, R0F8.1.5, R0F8.1.6, R0F8.1.7, R0F8.4, R0F8.4.1, R0F8.4.2, R1F8.4.3, R1F8.4.4, R0F29, R0F29.1, R0F29.2, R0F29.3, R0F29.4, R0F29.5, R0F29.6, R0F29.7, R0F30, R0F31, R0F31.1, R0F32, R0F33, R0F34, R0F34.1, R0F37, R0F38, R0F38.1, R0F38.2, R0F38.3, R0F38.4, R0F38.4.1, R0F38.5, R0F39
app::models::ModelNotification::Notification	R0F15, R0F15.1, R0F15.1.1, R0F15.1.2, R0F15.1.3, R0F15.1.4, R0F15.1.5, R0F15.1.6, R0F15.2, R0F15.2.1, R0F15.2.2, R0F15.2.3, R0F15.2.4, R0F36, R0F36.1
app::models::ModelTenant::Tenant	R0F1, R0F1.1, R0F1.1.1, R0F1.1.2, R0F2, R1F2.1



app::models::ModelThing::Thing	R0F8.2, R0F8.2.1, R0F8.2.1.1, R0F8.2.1.2, R0F8.2.1.3, R0F8.2.1.4, R0F8.2.1.5, R0F8.2.2, R0F8.2.3, R0F8.2.3.1, R0F8.2.3.2, R0F8.2.3.3, R0F8.2.3.4, R0F8.2.3.5, R0F8.2.4, R0F8.2.4.1, R0F8.2.5, R0F8.2.6, R1F8.3, R1F8.3.1, R1F8.3.1.1, R1F8.3.2, R0F12, R0F13, R0F14, R0F15, R0F15.1, R0F15.1.1, R0F15.1.2, R0F15.1.3, R0F15.1.4, R0F15.1.5, R0F15.1.6, R0F15.2, R0F15.2.1, R0F15.2.2, R0F15.2.3, R0F15.2.4, R0F16, R0F16.1, R0F16.2, R0F17, R0F18, R1F19
app::models::ModelThingModel::Model	R0F8.2, R0F8.2.1, R0F8.2.1.1, R0F8.2.1.2, R0F8.2.1.3, R0F8.2.1.4, R0F8.2.1.5, R0F8.2.2, R0F8.2.3, R0F8.2.3.1, R0F8.2.3.2, R0F8.2.3.3, R0F8.2.3.4, R0F8.2.3.5, R0F8.2.4, R0F8.2.4.1, R0F8.2.5, R0F8.2.6, R1F19, R0F25, R0F26, R0F28, R0F28.1, R0F28.1.1, R0F35, R0F35.1, R0F36.1

Tabella 3: Tracciamento componenti-requisiti.

Descrizione Design Pattern

Design Pattern Architetture

MVC

Model-View-Controller (MVC) è un pattern architetturale atto all'implementazione di interfacce utente. L'applicazione, per poter essere coerente con la struttura proposta da MVC, deve separare i componenti software che implementano il modello delle funzionalità di business dai componenti che implementano la logica di presentazione e di controllo che utilizzano tali funzionalità. Vengono quindi definite tre tipologie di componenti che soddisfano tali requisiti:

- **Model:** implementa le funzionalità di business;
- **View:** implementa la logica di presentazione;
- **Controller:** implementa la logica di controllo.

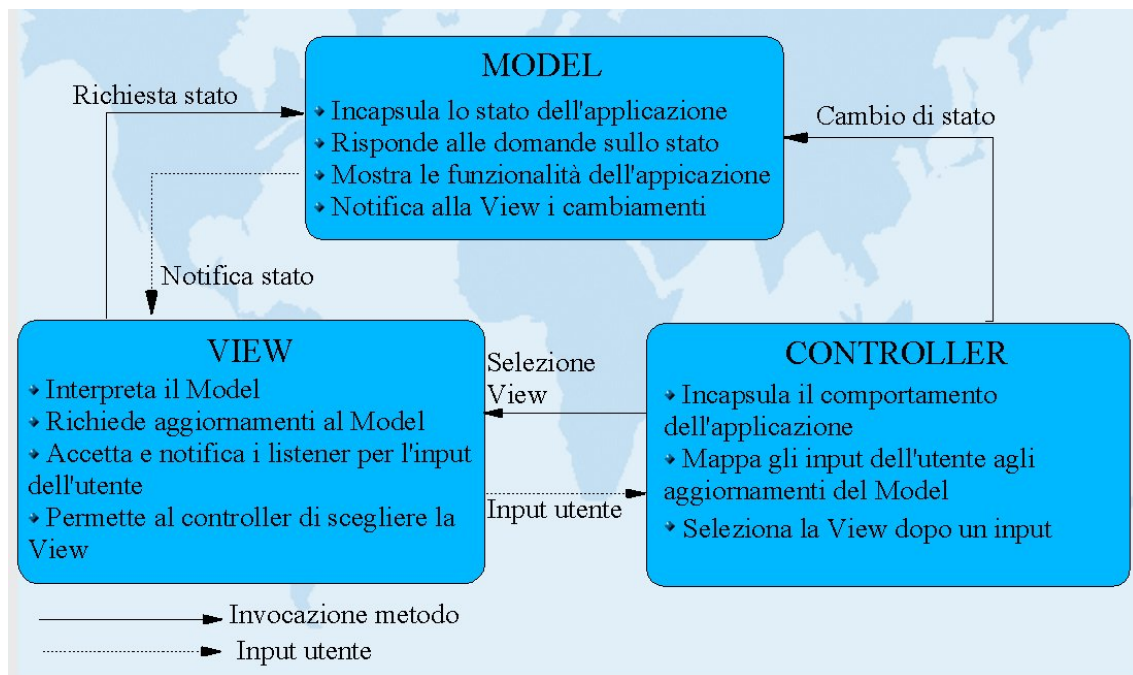


Figura 41: MVC - Diagramma di interazione

A.1.1.1 Model

Il *core* dell'applicazione viene implementato dal Model, che incapsulando lo stato della stessa definisce i dati e le operazioni che possono essere eseguite su questi. Quindi definisce le regole di business per l'interazione con i dati, esponendo alla View ed al Controller rispettivamente le funzionalità per l'accesso e l'aggiornamento. Per lo sviluppo del Model quindi è solito utilizzare le tipiche tecniche di progettazione *object oriented*, al fine di ottenere un componente software che astragga al meglio concetti importati dal mondo reale. Il Model può inoltre avere la responsabilità di notificare ai componenti della View eventuali aggiornamenti verificatisi in seguito a richieste del Controller, al fine di permettere alle View di presentare agli occhi degli utenti dati sempre aggiornati.



A.1.1.2 View

La logica di presentazione dei dati viene gestita solo e solamente dalla View. Ciò implica che questa deve fondamentalmente gestire la costruzione dell'interfaccia grafica che rappresenta il mezzo mediante il quale gli utenti interagiranno con il sistema. Ogni GUI_G può essere costituita da schermate diverse che presentano più modi di interagire con i dati dell'applicazione. Per far sì che i dati presentati siano sempre aggiornati è possibile adottare due strategie note come “push model” e “pull model”. Il push model adotta il pattern **Observer**, registrando le View come osservatori del Model. Le View possono quindi richiedere gli aggiornamenti al Model in tempo reale grazie alla notifica di quest'ultimo. Benché questa rappresenti la strategia ideale, non è sempre applicabile, in quanto richiede che tutta l'applicazione sia sviluppata nella medesima tecnologia. In tali casi è possibile utilizzare il pull model, dove la View richiede gli aggiornamenti quando “lo ritiene opportuno”. Inoltre la View delega al Controller l'esecuzione dei processi richiesti dall'utente dopo averne catturato gli input e la scelta delle eventuali schermate da presentare.

A.1.1.3 Controller

Questo componente ha la responsabilità di trasformare le interazioni dell'utente della View in azioni eseguite dal Model. Il Controller non rappresenta un semplice ponte tra View e Model. Realizzando la mappatura tra input dell'utente e processi eseguiti dal Model e selezionando la schermate della View richieste, il Controller implementa la logica di controllo dell'applicazione.