NetBreak

Progetto API Market

Specifica Tecnica

Informazioni sul documento

Nome del documento	SpecificaTecnica 3_0_0.pdf
Data di creazione	25 Febbraio 2017
Ultima modifica	06 Giugno 2017
Versione	3.0.0
Stato	Approvato
Redatto da	Marco Casagrande Dan Serbanoiu Alberto Nicolè Andrea Scalabrin Nicolò Scapin
Verificato da	Andrea Scalabrin
Approvato da	Davide Scarparo
Uso	Esterno
Distribuzione	NetBreak
Destinato a	Prof. Tullio Vardanega, Prof. Riccardo Cardin, ItalianaSoftware S.r.l.
Email di riferimento	netbreakswe@gmail.com

Abstract

Questo documento contiene la *Specifica Tecnica* che il gruppo NetBreak ha prodotto durante l'attività di progettazione ad alto livello del prodotto API Market.

Changelog

Versione	Data	Autore	Ruolo	Descrizione
3.0.0	2017-06-06	Davide Scarparo	Responsabile	Approvazione documento
2.1.0	2017-06-03	Andrea Scalabrin	Verificatore	Verifica documento
2.0.2	2017-05-29	Marco Casagrande	Amministratore	Contestualizzazione Desing pattern sezione #4 e correzione tracciamento sottosezione #7.2
2.0.1	2017-05-26	Nicolò Scapin	Verificatore	precisata versione Angular in sezione #2.1.1.2
2.0.0	2017-03-26	Davide Scarparo	Responsabile	Approvazione documento
1.1.0	2017-03-23	Andrea Scalabrin	Verificatore	Verifica correzioni
1.0.5	2017-03-22	Nicolò Scapin	Progettista	Sostituizione immagine errata del design pattern Singleton sezione #4.2
1.0.4	2017-03-21	Alberto Nicolè	Progettista	Rimozione sezione #2.2: "SOA vs. Microservices Architecture"
1.0.3	2017-03-21	Alberto Nicolè	Progettista	Contestualizzazione Design Pattern nella sezione #4 e aggiunta Facade in sottosezione #4.3
1.0.2	2017-03-20	Nicolò Scapin	Progettista	Rimozione svantaggi tecnologie Java dalla sottosezione #3.5
1.0.1	2017-03-16	Alberto Nicolè	Progettista	Inizio correzione sezione #2 "Specifica di Prodotto" in particolare pagine 3-5
1.0.0	2017-03-05	Davide Scarparo	Responsabile	Approvazione del documento
0.3.0	2017-03-04	Andrea Scalabrin	Verificatore	Verifica del documento
0.2.3	2017-03-04	Nicolò Scapin	Progettista	Stesura della sezione #6: "Diagrammi di sequenza"
0.2.2	2017-03-04	Dan Serbanoiu	Progettista	Stesura della sezione #5: "Diagrammi di attività"
0.2.1	2017-03-04	Alberto Nicolè	Progettista	Correzioni secondo la verifica
0.2.0	2017-03-03	Nicolò Scapin	Verificatore	Verifica del documento
0.1.8	2017-03-03	Davide Scarparo	Progettista	Inserimento diagrammi delle classi per sottosezione #5.3: "Back-end"
0.1.7	2017-03-03	Marco Casagrande	Progettista	Stesura della sottosezione #5.3: "Back-end"

Versione	Data	Autore	Ruolo	Descrizione
0.1.6	2017-03-02	Davide Scarparo	Progettista	Inserimento diagrammi delle classi per sottosezione #5.2: "Front-end"
0.1.5	2017-03-02	Dan Serbanoiu	Progettista	Stesura della sottosezione #5.2: "Front-end"
0.1.4	2017-03-02	Andrea Scalabrin	Progettista	Stesura dei diagrammi dei packages e classi
0.1.3	2017-03-01	Davide Scarparo	Progettista	Introduzione sezione #5: "Diagrammi dei packages e delle classi"
0.1.2	2017-03-01	Alberto Nicolè	Progettista	Stesura della sezione #3: "Tecnologie utilizzate"
0.1.1	2017-03-01	Marco Casagrande	Progettista	Correzioni secondo la verifica
0.1.0	2017-02-28	Andrea Scalabrin	Verificatore	Verifica del documento
0.0.5	2017-02-27	Nicolò Scapin	Progettista	Stesura della sezione #4: "Design pattern utilizzati"
0.0.4	2017-02-27	Dan Serbanoiu	Progettista	Aggiunta sezione #2.2: "SOA vs. Microservices Architecture"
0.0.3	2017-02-26	Andrea Scalabrin	Progettista	Stesura della sezione #2: "Specifica del prodotto"
0.0.2	2017-02-25	Marco Casagrande	Progettista	Stesura della sezione Introduzione
0.0.1	2017-02-25	Davide Scarparo	Responsabile	Creazione template

Indice

Elenco delle figure

1 Introduzione

1.1 Scopo del documento

Lo scopo del presente documento è la definizione ad alto livello della struttura che dovrà avere il prodotto *API Market*$_G$. Verranno presentati la gerarchia dei packages, le principali classi contenute in essi, le interazioni fra esse, e tutti i design pattern utilizzati. Il documento servirà come guida per i *Programmatori* del team *NetBreak* durante le successive attività di Codifica.

1.2 Scopo del prodotto

Lo scopo del prodotto è la realizzazione di un API Market per l'acquisto e la vendita di *microservizi*$_G$. Il sistema offrirà la possibilità di registrare nuove *API*$_G$ per la vendita, permetterà la consultazione e la ricerca di API ai potenziali acquirenti, gestendo i permessi di accesso ed utilizzo tramite creazione e controllo di relative *API key*$_G$. Il sistema, oltre alla web app stessa, sarà corredato di un *API Gateway*$_G$ per la gestione delle richieste e il controllo delle chiavi, e fornirà funzionalità avanzate di statistiche per il gestore della piattaforma e per i fornitori dei microservizi.

1.3 Riferimenti normativi

- NormeDiProgetto 4_0_0.pdf
- AnalisiDeiRequisiti 3_0_0.pdf

1.4 Riferimenti informativi

- **Ingegneria del software - Ian Sommerville - 8a edizione:**
  Porzione dedicata alla Progettazione Architetturale (Capitolo 11)
- **Slides del corso - Design patterns strutturali**
  `http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E04.pdf`
- **Slides del corso - Design patterns creazionali**
  `http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E05.pdf`
- **Slides del corso - Design patterns architetturali**
  - `http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E08.pdf`
  - `http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E09.pdf`
- **Slides del corso - Diagrammi dei packages**
  `http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E02b.pdf`
- **Slides del corso - Diagrammi delle classi**
  `http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E02a.pdf`
- **Slides del corso - Diagrammi di sequenza**
  `http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E03a.pdf`
- **Slides del corso - Diagrammi di attività**
  `http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E03b.pdf`
- **HTML5**
  `https://www.w3.org/TR/html5/`
- **Bootstrap CSS Framework**
  `http://getbootstrap.com/`

- **AngularJS**
  `https://www.angularjs.org/`
- **JQuery JavaScript Library**
  `https://jquery.com/`
- **Jolie Programming Language**
  `http://www.jolie-lang.org/`
- **Java Programming Language**
  `https://www.oracle.com/it/java/index.html`
- **MySQL Database**
  `https://www.mysql.it/`
- **Chris Richardson**
  - `http://microservices.io`
  - `http://microservices.io/patterns/microservices.html`
- **Martin Fowler**
  - `https://martinfowler.com/articles/microservices.html`
  - `https://martinfowler.com/microservices/`
- **IBM**
  `http://www.ibm.com/developerworks/websphere/library/techarticles/1601_clark-trs/1601_clark.html`
- **NGINX**
  - `https://www.nginx.com/blog/introduction-to-microservices/`
  - `https://www.nginx.com/blog/building-microservices-using-an-api-gateway/`

1.5 Glossario

Per semplificare la consultazione e disambiguare alcune terminologie tecniche, le voci indicate con la lettera *G* a pedice sono descritte approfonditamente nel documento Glossario 3_0_0.pdf e specificate solo alla prima occorrenza all'interno del suddetto documento.

2 Specifica del prodotto

In questa sezione verrà descritta in maniera macroscopica l'architettura che verrà adottata durante l'attività di Progettazione. Per semplificare la comprensione, verrà utilizzato un approccio top-down, partendo dalle linee generali, fino a giungere ad una descrizione in dettaglio delle componenti.

2.1 Architettura generale

Il software API Market si caratterizza dalle comuni applicazioni per l'approccio a microservizi adottato nell'implementazione. In maniera differente da un applicazione monolitica, ogni componente viene realizzato come microservizio a sè stante. L'aggregazione di questi microservizi definisce l'applicazione finale e ne realizza il comportamento definitivo. Esulando dall'approccio a microservizi, il sistema sarà realizzato al pari di una classica applicazione *Client-Server*, dove il lato *front-end*$_G$ (Client) si occuperà di fornire all'utente finale l'interfaccia web su cui poter interagire, mentre il lato *back-end*$_G$ (Server) gestirà e salverà i dati, e si occuperà della gestione delle chiamate API (tramite l'opportuna componente *API Gateway*). La base di dati utilizzata, si occupa della raccolta di dati sensibili dell'utenza, della gestione delle applicazioni e delle chiavi e, infine, della gestione dei dati statistici.
Il sistema verrà realizzato, per il lato back-end, tramite microservizi con interfaccia Jolie, che verranno implementati con codice *Java*$_G$ o *Jolie*. Questo approccio, sebbene la tematica sia di recente sviluppo, risulta molto gettonato dalle grandi realtà che recentemente hanno deciso di affacciarsi a questa realtà (come ad esempio l'*AWS*$_G$ Service Registry *Eureka*, usato dalla piattaforma Netflix).
Il lato front-end, invece, verrà implementato tramite le consuete tecnologie *HTML5*$_G$, *CSS3*$_G$ e *JavaScript*$_G$, in particolare, l'utilizzo del framework *AngularJS*$_G$ consentirà un binding in real-time per la rappresentazione delle informazioni, di fatto fornendo una single page application. Nella realizzazione dell'API Market richiesto, verrà utilizzato il desing pattern architetturale ***Model-View-Controller***.

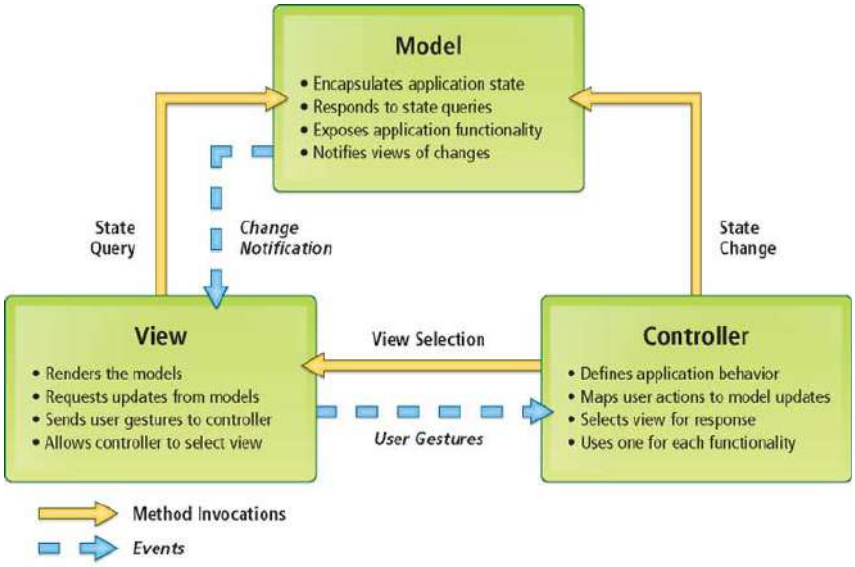

Figura 1: Pattern architetturale Model View Controller

2.1.1 Pattern architetturale Front-End

Il pattern architetturale *Model-View-Controller* (MVC) è composto di tre parti fondamentali, ed in questa sezione verranno presentate brevemente le funzionalità di ognuna. Verrà motivata,

inoltre, la scelta di tale pattern da parte del team.

- **Model**: rappresenta la business logic dell'applicazione, il punto di accesso ai dati, si occupa dell'interazione con essi e ne specifica le modalità di accesso. Inoltre, ha il compito di notificare le Views in caso di cambiamenti;
- **View**: componente che ha il compito di mostrare il contenuto dei *Models* e di comunicare con i *Controllers* riguardo le azioni utente sulla specifica *View*;
- **Controller**: ha il compito di definire il comportamento dell'applicazione, selezionando le corrette *Views* e aggiornando i *Models* in base alle azioni utente.

In fase di progettazione, il gruppo ha preso in esame diversi design pattern architetturali, ma la volontà di realizzare pagine web dinamiche e reattive, unitamente all'uso di tecnologie moderne, su tutte l'utilizzo del framework *AngularJS (Angular 1)*, hanno indirizzato la scelta verso il pattern MVC. AngularJS infatti implementa tutte le componenti del pattern MVC.

2.1.1.1 Componente Model

Il **Model** svolge le seguenti funzioni:

- Incapsula la business logic dell'applicazione;
- Interagisce con i database, nei quali vengono salvati i dati del sistema, come API/microservizi degli utenti sviluppatori della piattaforma, transazioni e profili utenti. Le funzionalità incluse comprendono microservizi in grado di caricare, salvare, modificare e cancellare dati sui rispettivi database;
- Espone delle interfacce contenenti le funzionalità dell'applicazione;
- Notifica le rispettive *Views* nel caso di cambiamenti.

2.1.1.2 Componente View

La componente **View** è un'interfaccia web, ed il gruppo ha scelto di utilizzare le classiche tecnologie *CSS3* e *HTML5* per la struttura di base. All'utilizzo di AngularJS, per la parte View viene affiancato anche il linguaggio *Javascript*. La componente View ha, quindi, lo scopo di svolgere le seguenti funzioni:

- Rappresentare i dati del *Model*;
- Richiedere degli aggiornamenti ai rispettivi *Models*;
- Comunicare e inoltrare tutte le azioni utente ai relativi *Controllers*;
- Permettere ai *Controllers* di selezionare le corrette *Views*.

2.1.1.3 Componente Controller

La componente **Controller** comunica sia con la componente *View* che con la componente *Model*. Essa è l'unica componente a conoscere gli altri, realizzando un totale disaccopiamento tra *View* e *Model*. Il Controller ha il compito di svolgere le seguenti funzioni:

- Definire il comportamento dell'applicazione, in particolare definendo le risposte alle azioni utente;
- Realizzare il data-binding tra le azioni utente sulle *Views* e gli aggiornamenti di stato dei relativi *Models*;

- Selezionare le corrette Views da visualizzare all'utente;
- Ogni *Controller* è associato ad una singola funzionalità.

2.1.2 Pattern architetturale Back-End

Molti dei microservizi pensati per la parte di comunicazione con i database sono stati progettati per essere relizzati seguendo il design pattern ***Aggregator Microservices***.

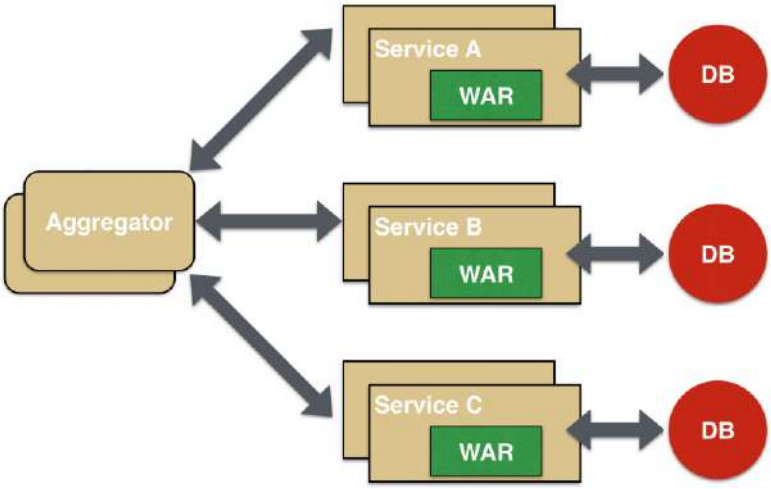

Figura 2: Aggregator Microservices design pattern

Questo design pattern si compone di un *aggregatore*, ovvero un microservizio in grado di collezionare i dati che gli vengono forniti da diversi microservizi, per poi renderli fruibili all'utente, tramite la componente *View*. Utilizzando un design pattern a microservizi è necessario che le varie componenti siano indipendenti tra loro e, quindi, che lo siano anche i relativi database. Tutti i microservizi contenuti in un package, sono indipendenti da microservizi contenuti in altri package, ed ogni microservizio opera sul proprio database associato.

2.1.3 Pattern architetturale API Gateway

Il design pattern architetturale scelto per parte dell'implementazione dell'API Gateway è il design pattern ***Chain Microservices***. Questo tipo di pattern consiste di una sequenza di azioni prestabilite, innescate da una richiesta $HTTP_G$.

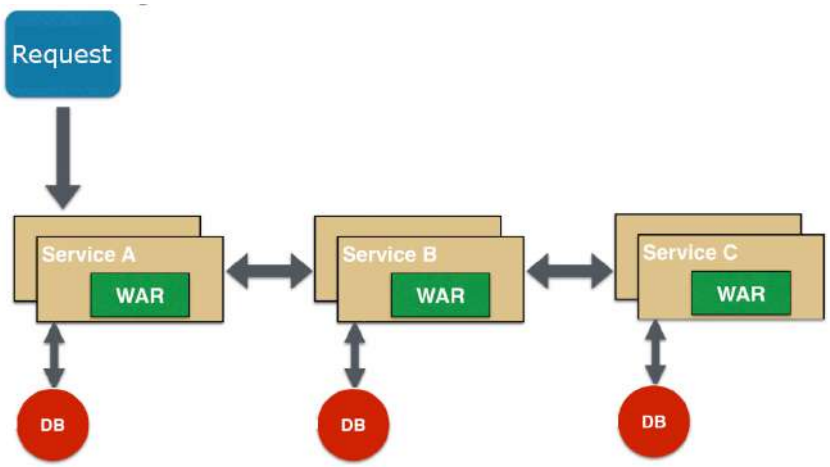

Figura 3: Chain Microservices design pattern

Ogni servizio non può intraprendere la propria esecuzione se non è terminato il servizio precedente. Un punto critico di questa scelta è la lunghezza della catena. L'API Gateway è progettato per elaborare una richiesta e ritornare una risposta al chiamante, trattandosi, dunque, di richieste sincrone. Una catena troppo lunga crea una tempo di attesa non indifferente per il client, considerando che bisogna valutare anche il tempo impiegato dal microservizio target della chiamata all'API Gateway.

3 Tecnologie utilizzate

In questa sezione, vengono descritte le tecnologie utilizzate per la realizzazione della piattaforma API Market. Verranno elencati i pregi e i difetti individuati durante l'analisi, e le motivazioni che hanno spinto il gruppo *NetBreak* a intraprendere tali scelte progettuali e tecnologiche.

3.1 AngularJS

- **Descrizione**: AngularJS è un framework JavaScript per lo sviluppo di applicazioni web Client-side. Può essere aggiunto alle pagine HTML tramite opportune inclusioni. È distribuito come un file JavaScript.
- **Utilizzo**: questa tecnologia verrà utilizzata per rendere le pagine HTML reattive e responsive alle richieste in real-time.
- **Vantaggi**:
  - Interazioni in tempo reale con l'utente per pagine dinamiche single-page;
  - Facilita e snellisce lo sviluppo;
  - Risulta facile da testare;
  - Estende HTML;
  - Offre la possibilità di un data-binding bidirezionale.
- **Svantaggi**:
  - Semplice da imparare, ma con una curva di apprendimento essenzialmente logaritmica;
  - Potrebbe essere complicato un debug degli errori;
  - Documentazione non particolarmente sviluppata rispetto a tecnologie più consolidate.

3.2 Bootstrap 3

- **Descrizione**: Bootstrap 3 è un framework CSS3, utilizzato per semplificare la definizione di un interfaccia grafica sulla struttura HTML.
- **Utilizzo**: questa tecnologia verrà utilizzata per la realizzazione dell'interfaccia grafica per il prodotto API Market.
- **Vantaggi**:
  - Supporta la maggior parte dei browser;
  - Permette un ampio riuso del codice;
  - È basato su SASS, un diffuso pre-processore per il linguaggio CSS;
  - La documentazione è vasta e completa, ed è ben supportato dalla community;
  - Leggero e facilmente personalizzabile;
  - Pensato per un modello responsive, con griglie e breakpoint predisposti.
- **Svantaggi**:
  - La scrittura dello stile può risultare verbosa e può generare HTML poco elegante;
  - I siti realizzati con questo framework, se non personalizzati, risultano molto similari tra loro.

3.3 CSS3

- **Descrizione**: CSS3 è la principale e più aggiornata tecnologia, e viene utilizzata per la descrizione della componente grafica di un sito.
- **Utilizzo**: questa tecnologia viene utilizzata per la personalizzazione dell'aspetto grafico precedentemente definito tramite Bootstrap 3.
- **Vantaggi**:
  - Rappresenta l'ultima versione stabile di questo linguaggio;
  - Aiuta nella creazione di pagine con lo stesso stile e formato;
  - È supportato ampiamente dalla maggior parte dei browser;
  - Risulta facile da apprendere.
- **Svantaggi**:
  - L'uso di questa tecnologia può rendere caotica la creazione di siti utilizzando software di terze parti.

3.4 HTML5

- **Descrizione**: HTML è il linguaggio di markup principale per la definizione di una struttura per un sito.
- **Utilizzo**: questa tecnologia verrà utilizzata per descrivere la struttura delle pagine front-end.
- **Vantaggi**:
  - L'ultima versione adottata ha introdotto significativi miglioramenti riguardo la pulizia del codice;
  - Sono state introdotte funzionalità avanzate, assenti nelle versioni precedenti;
  - La semantica è stata standardizzata in maniera più rigorosa.
- **Svantaggi**:
  - Il supporto all'ultima versione di questo standard potrebbe non essere completamente definita in browser non aggiornati.

3.5 Java

- **Descrizione**: Java è un linguaggio di programmazione orientato agli oggetti, che presenta il grande vantaggio di essere indipendente dalla piattaforma in cui viene eseguito. Java è, ad oggi, il linguaggio di programmazione più diffuso ed utilizzato al mondo.
- **Utilizzo**: questa tecnologia verrà utilizzata nell'implementazione delle interfacce definite in linguaggio Jolie.
- **Vantaggi**:
  - Può essere eseguito ovunque sia disponibile una versione della *Java Virtual Machine*$_G$;
  - Ha una gestione automatica della memoria, tramite *Garbage Collector*$_G$;
  - È relativamente facile da imparare;
  - Supporta nativamente la concorrenza e distribuzione.

3.6 JavaScript ES6

- **Descrizione**: JavaScript è il principale linguaggio per estendere le funzionalità front-end di un sito.
- **Utilizzo**: questa tecnologia sarà utilizzata per l'implementazione di funzionalità aggiuntive, assieme al framework AngularJS.
- **Vantaggi**:
  - È un linguaggio relativamente semplice da imparare;
  - Estende le funzionalità delle pagine web;
  - JavaScript è relativamente veloce per l'utente finale.
- **Svantaggi**:
  - Può presentare facilmente falle relative alla sicurezza;
  - Rallenta il caricamento delle pagine in dispositivi non particolarmente performanti;
  - Deve essere esplicitamente abilitato dall'end-user per poter usufruire delle funzionalità.

3.7 Jolie

- **Descrizione**: Jolie è il primo linguaggio esplicitamente orientato ai microservizi.
- **Utilizzo**: Jolie verrà utilizzato per la definizione delle interfacce nei microservizi che saranno realizzati per il lato back-end.
- **Vantaggi**:
  - Non necessita di framework, ma solamente di un interprete;
  - Funziona tramite una Java Virtual Machine;
  - È nativamente compatibile con Java per l'estensione delle funzionalità;
  - La distribuzione di un microservizio può avvenire su più macchine, senza modificarne l'esecuzione.
- **Svantaggi**:
  - Linguaggio molto giovane e di scarsa diffusione;
  - Documentazione scarna;
  - Limitate tecnologie in grado di interfacciarsi con applicazioni Jolie.

3.8 JQuery

- **Descrizione**: JQuery è una libreria per il linguaggio JavaScript. Viene utilizzata per estendere le funzionalità di tale linguaggio, aggiungendo operazioni implementabili con semplicità.
- **Utilizzo**: questa tecnologia verrà utilizzata per aggiungere funzionalità nel lato front-end.
- **Vantaggi**:
  - È ben supportato dalla community;
  - Rende la manipolazione DOM molto più agevole;

  - Possiede diversi plug-in che estendono in modo ancor maggiore le funzionalità.

- **Svantaggi**:
  - Presenta un ulteriore livello di appesantimento della struttura front-end;
  - La curva di apprendimento è logaritmica e può risultare difficile.

3.9 MySQL

- **Descrizione**: MySQL è un database di tipo relazionale, open-source e sviluppato da Oracle. Tale database è tuttora il più diffuso e utilizzato nella sua categoria.
- **Utilizzo**: questa tecnologia verrà utilizzata per la gestione e il salvataggio dei dati back-end per l'applicazione.
- **Vantaggi**:
  - Valida e consolidata esperienza del team per tale piattaforma;
  - La tecnologia si integra perfettamente con i framework utilizzati;
  - Documentazione estesa e valido supporto da parte della community;
  - Sono disponibili numerosi strumenti per la progettazione della base di dati.
- **Svantaggi**:
  - Alcune alternative offrono un bacino di funzionalità maggiore, sebbene non risultino adatti per interoperare con le altre tecnologie utilizzate.

4 Design pattern utilizzati

4.1 Abstract Factory

L'***Abstract Factory*** è un design pattern creazionale e fornisce un'interfaccia per creare famiglie di prodotti, senza dover esplicitare il nome concreto delle classi a cui si riferisce. In questo modo, si permette che un sistema sia indipendente dall'implementazione degli oggetti concreti e che il client, attraverso l'interfaccia, utilizzi diverse famiglie di prodotti. Quindi, il client conosce solo l'interfaccia per creare le famiglie di prodotti, ma non la sua implementazione concreta. L'*Abstract Factory* è costituito da 5 elementi:

- **AbstractFactory**: dichiara un'interfaccia per le operazioni che crea oggetti product astratti;
- **ConcreteFactory**: implementa le operazioni per creare oggetti concreti product di AbstractProduct. Per garantire che nel sistema esiste un'unica istanza di ciascuna ConcreteFactory, è buona norma definire ciascuna di esse come *Singleton*;
- **AbstractProduct**: interfaccia che definisce la struttura base dei product che la factory può instanziare;
- **ConcreteProduct**: nel sistema possono essere creati n ConcreteProduct, ciascuno dei quali dovrà implementare l'interfaccia AbstractProduct;
- **Client**: utilizza l'AbstractFactory per generare i prodotti concreti all'interno del sistema.

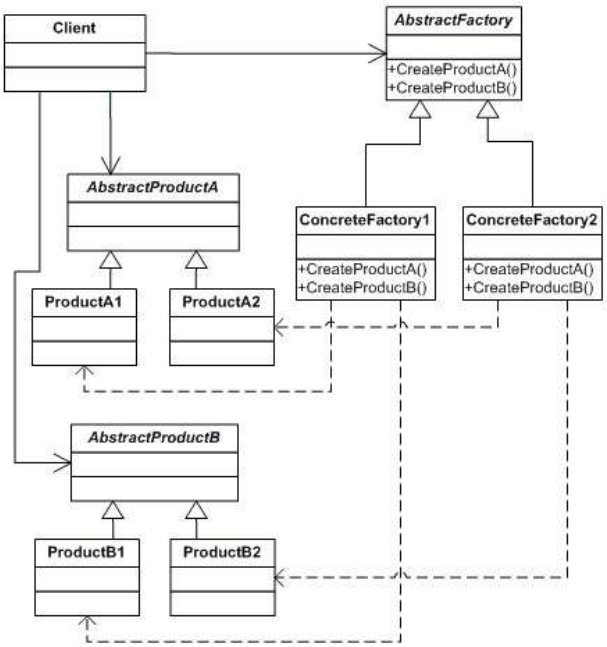

Figura 4: Design pattern Abstract Factory

I punti di forza di questo design pattern sono:

- Isola i client dall'implementazione delle classi concrete;
- Rafforza il raggruppamento di prodotti in famiglie;
- Rende possibile interscambiare facilmente le famiglie di prodotti, perché ogni factory concreta genera una famiglia di prodotti.

4.1.1 Campo di utilizzo

Nel sistema realizzato, il design pattern *Abstract Factory* verrà utilizzato per rendere il sistema indipendente dalla creazione delle classi concrete ed essere aperto all'estensione tramite la definizione di nuovi tipi.

4.2 Singleton

Il ***Singleton*** è un design pattern creazionale che ha lo scopo di garantire che venga creata una e una sola istanza di una determinata classe, e di fornire un punto di accesso globale a tale istanza. Gli elementi caratteristici fondamentali per l'implementazione di questo pattern sono due:

- Un costruttore privato, per evitare che la classe possa essere istanziata arbitrariamente;
- Un metodo statico per accedere alla singola istanza.

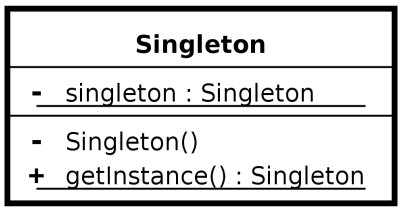

Figura 5: Design pattern Singleton

I punti di forza di questo design pattern sono:

- L'accesso controllato all'istanza;
- È possibile consentire più istanze, sempre in modo controllato.

4.2.1 Campo di utilizzo

Ogni servizio in AngularJS è un *Singleton*, perchè ogni servizio è istanziato non più di una volta. In API Market saranno presenti dei servizi per gestire le interazioni tra la web app e i database contenenti i dati. Il design pattern *Singleton* è stato utilizzato anche per il back-end della piattaforma, infatti ogni microservizio realizzato in Jolie è istanziato solo una volta all'avvio e rimane in attesa di input per operare sul database e fornire risultati al servizio chiamante del front-end.

4.3 Facade

Il ***Facade*** è un design pattern strutturale che permette, attraverso un'interfaccia più semplice, l'accesso a sottosistemi che espongono interfacce complesse e molto diverse tra loro, nonché a blocchi di codice complessi. Questo rende una libreria più facile da capire, usare e testare, inoltre permette di diminuire le dipendenze tra sottosistemi senza nascondere le funzionalità di basso livello.

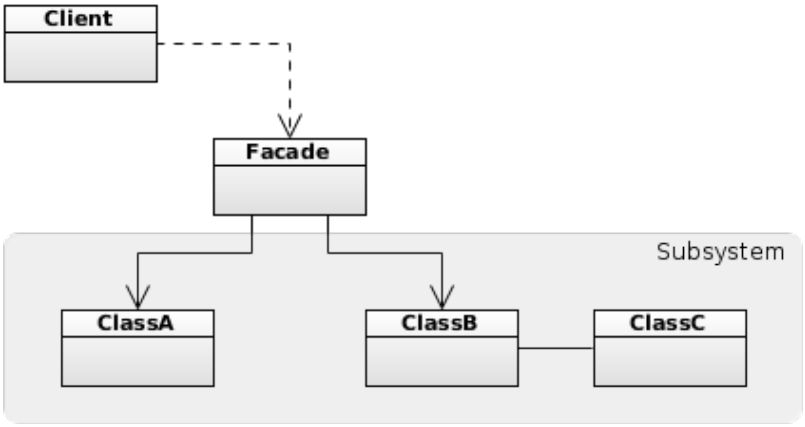

Figura 6: Design pattern Facade

I punti di forza di questo design pattern sono:

- Nascondere la complessità dell'operazione all'interno di un sistema complesso;
- Fornisce un'interfaccia semplificata al client.

4.3.1 Campo di utilizzo

Nella parte front-end dell'applicazione, il design pattern Facade viene utilizzato nel file principale *app.js*. Questo file definisce le routes dell'applicazione e viene utilizzato per associare un URL alle varie *view* dell'applicazione.
All'interno del file è utilizzata la funzione *$locationProvider* e *$routeProvider* che gestiscono il routing e specificano quale pagina utilizzare in caso non venga specificato l'URL.
Inoltre, il pattern Facade viene utilizzato nella parte di back-end dall'API Gateway quando un utente sviluppatore inserisce un microservizio sulla piattaforma. Infatti, esso può essere composto da un numero arbitrario di interfacce e sarà compito del gateway fornire ai clienti del marketplace un'interfaccia unificata, in modo da rendere user friendly l'utilizzo.

4.4 Decorator

Il ***Decorator*** è un design pattern strutturale che permette di aggiungere dinamicamente nuove responsabilità ad un oggetto. In questo modo, si possono estendere le funzionalità di oggetti particolari senza coinvolgere complete classi (senza dover utilizzare il subclassing).

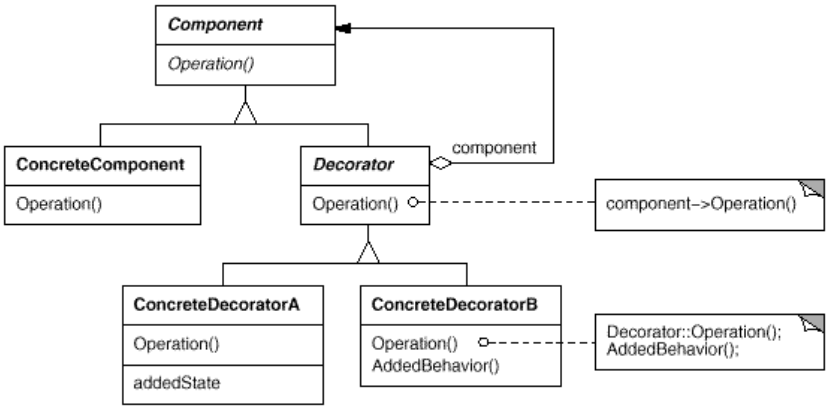

Figura 7: Design pattern Decorator

Questa tecnica si applica quando:

- Occorre aggiungere funzionalità dinamicamente ad un oggetto in modo trasparente;
- Si vogliono "circoscrivere" le funzionalità;
- Non è possibile estendere una classe per subclassing.

4.4.1 Campo di utilizzo

Il pattern Decorator viene utilizzato nella parte di back-end dall'API Gateway quando avviene la richiesta di utilizzo di un microservizio da parte di un utente cliente. Il gateway deve aggiungere dinamicamente (decorare) delle informazioni alla richiesta del cliente, che ha a disposizione la sola interfaccia del microservizio target. I dati aggiunti servono per il monitoring dei dati di SLA (Service Level Agreement) e per la parte di comunicazione vera e propria con il microservizio Jolie (location e protocol).

5 Diagrammi dei packages e delle classi

5.1 Introduzione

Questa sezione racchiude la descrizione dei diagrammi dei packages, progettati tramite un approccio top-down.
N.B.: raggiunto il sub-package minimo, si procederà con la descrizione delle classi implementate al suo interno.
Considerando l'approccio adottato e il particolare applicativo realizzato con un pattern a microservizi, il team non si discosterà dalla nomenclatura tradizionale di *Classe*, intesa come una categoria di entità in grado di svolgere uno specifico compito. Nei diagrammi delle classi del back-end, è rappresentato graficamente come classe, ciò che in realtà è un microservizio. I suoi metodi mostrano l'interfaccia da esso esposta, mentre i campi dati saranno presenti solo nell'implementazione del microservizio.
Trattandosi di una progettazione con un livello di dettaglio intermedio, le classi non verranno definite, bensì verrà data una descrizione della loro funzionalità, lo scopo e le relazioni che intercorrono con le altre classi. Ogni componente individuato, quindi, verrà descritto adottando il seguente schema:

- **Padre**: indica il package padre che contiene il componente in analisi;
- **Descrizione**: fornisce una breve descrizione riguardo funzionalità e scopo del componente;
- **Relazioni d'uso con altri componenti**: indica le eventuali varie relazioni d'uso con altri componenti, interne o esterna al package in analisi;
- **Package/classi contenuti/e**: elenco dei package e/o delle classi eventualmente contenuti dal componente in analisi.

L'applicativo API Market è strutturato, come di consueto, in un lato front-end ed un lato back-end, come mostrato dal diagramma seguente:

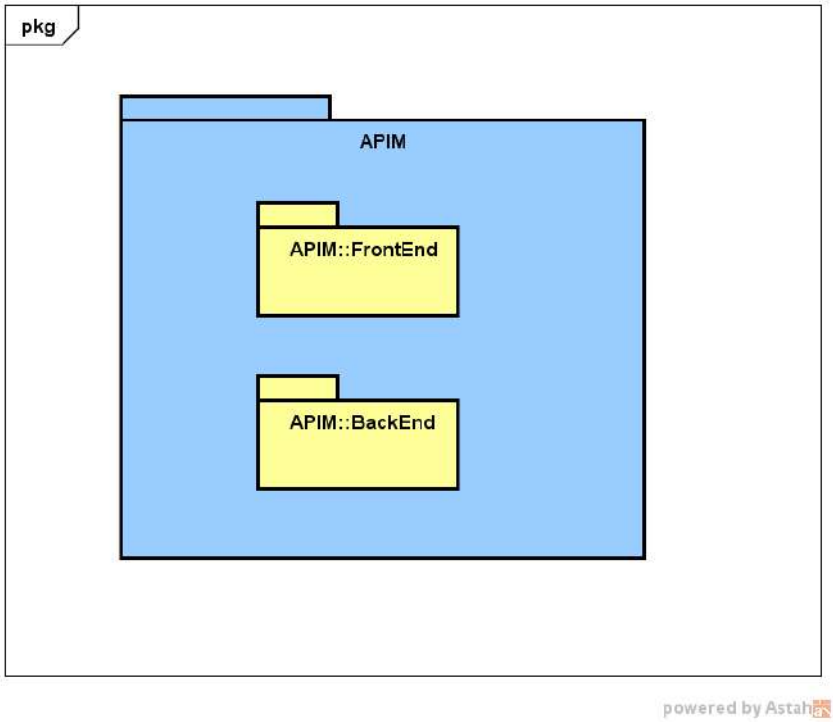

Figura 8: Package APIM

5.2 Front-end

Nella parte front-end sono presenti i package *node_ modules* e *App*, che derivano dall'architettura dei componenti e framework scelti (AngularJS).

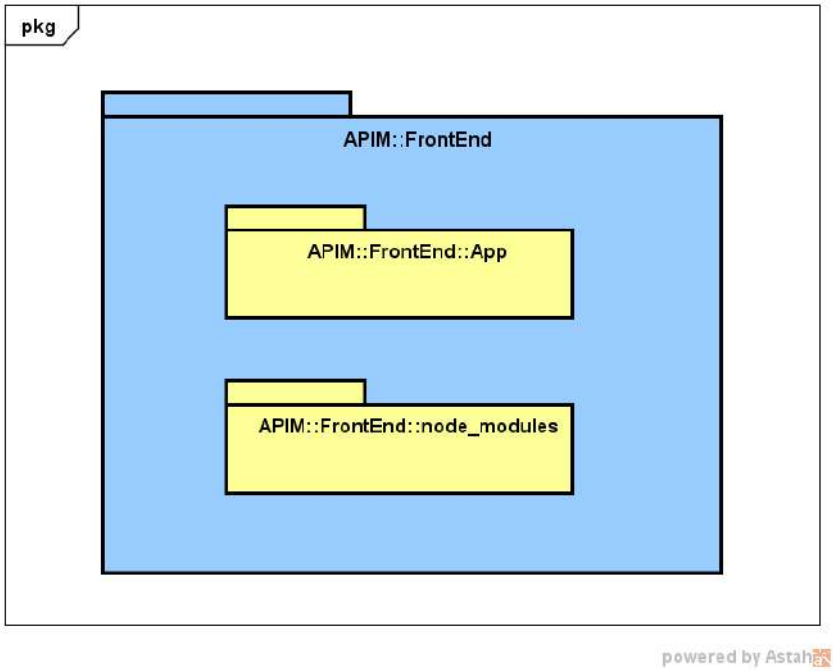

Figura 9: Package APIM::FrontEnd

- **Padre**: APIM;
- **Descrizione**: package contenente le componenti del front-end dell'applicazione;
- **Package contenuti**:
  - ***node_ modules***: package contenente i riferimenti alle librerie esterne di JavaScript;
  - ***App***: package contenente le componenti principali dell'applicazione.

5.2.1 node_modules

- **Padre**: FrontEnd;
- **Descrizione**: package che raccoglie le librerie esterne di JavaScript, installate tramite *npm* di Node.js, e fornisce le funzionalità necessarie alla parte di front-end dell'applicazione;
- **Relazioni d'uso con altri componenti**: questo package si relaziona con i *controllers* presenti nel sub-package *Controllers* del package *App*.

5.2.2 App

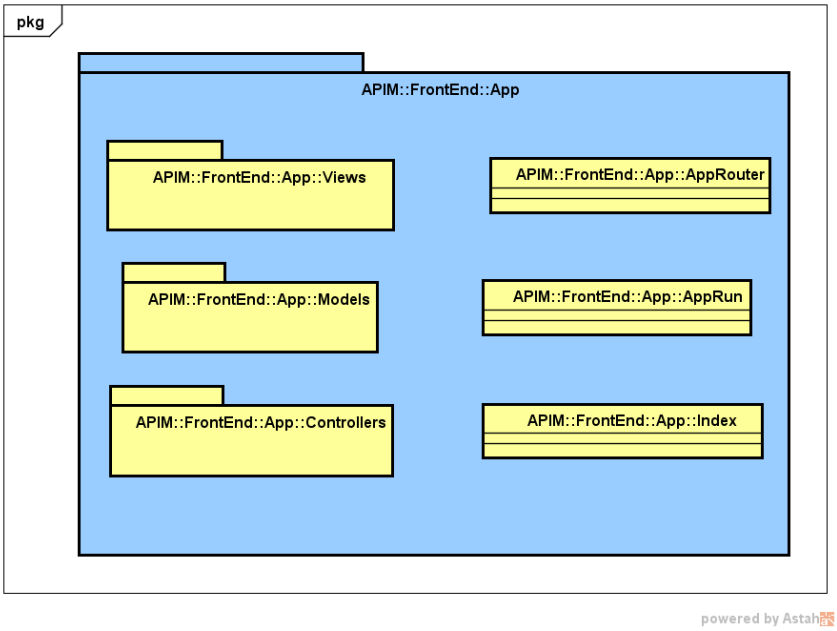

Figura 10: Package APIM::FrontEnd::App

- **Padre**: FrontEnd;
- **Descrizione**: package che raccoglie le componenti principali del front-end dell'applicazione web, secondo il pattern architetturale Model-View-Controller (MVC);
- **Relazioni d'uso con altri componenti**: questo package si relaziona con le librerie presenti nel sub-package *node_ modules* del package padre;
- **Package contenuti**:
  - ***Views***: package contenente le *views* del front-end dell'applicazione;
  - ***Models***: package contenente i *models* del front-end dell'applicazione;
  - ***Controllers***: package contenente i *controllers* del front-end dell'applicazione.
- **Classi contenute**:
  - ***AppRun***: classe che istanzia l'applicazione;
  - ***AppRouter***: classe che gestisce i routes dell'applicazione;
  - ***Index***: view generale dell'applicazione (single page app).

5.2.2.1 AppRun

- **Padre**: App;
- **Descrizione**: classe che istanzia l'applicazione e che viene utilizzata per indicare le dipendenze tra l'applicazione e i packages esterni.

5.2.2.2 AppRouter

- **Padre**: App;
- **Descrizione**: classe che gestisce i routes dell'applicazione al fine di associare ad ogni route un controller e una view (associa un URL alle varie view dell'applicazione).

5.2.2.3 Index

- **Padre**: App;
- **Descrizione**: classe che rappresenta la view generale dell'applicazione e che contiene gli elementi che saranno presenti in ogni pagina dell'applicazione.

5.2.2.4 Views

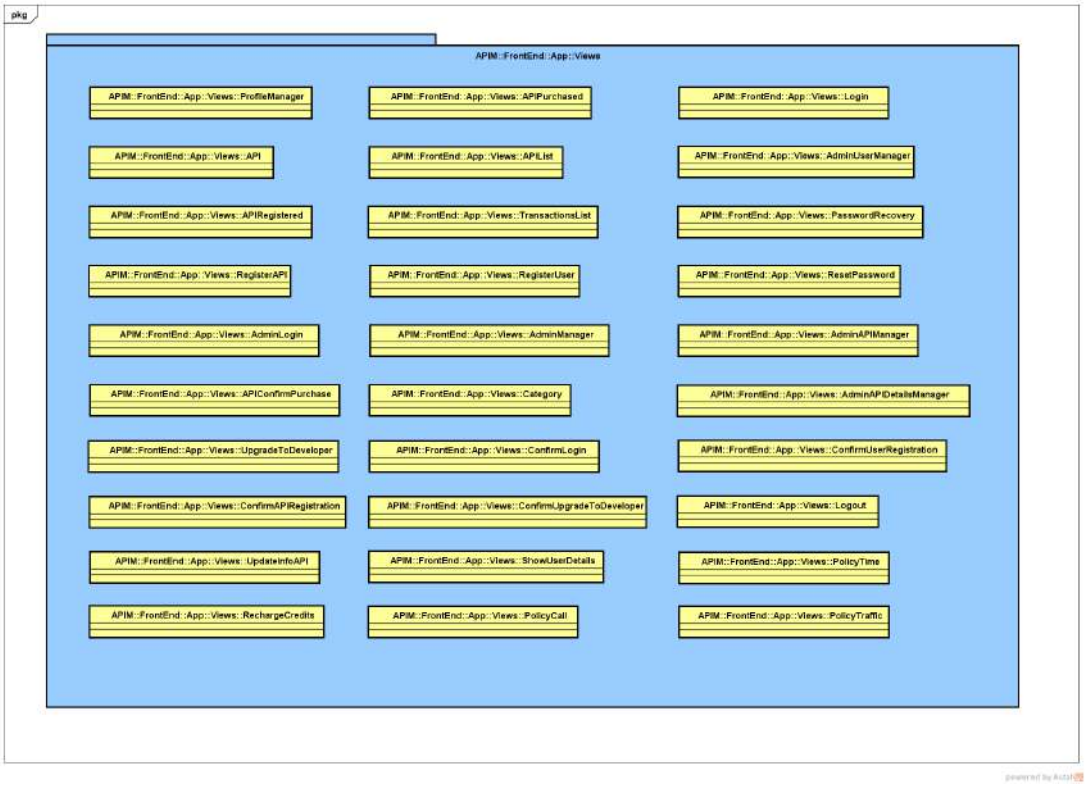

Figura 11: Package APIM::FrontEnd::App::Views

- **Padre**: App;
- **Descrizione**: package contenente tutte le classi che rappresentano i vari template HTML per le pagine web dell'applicazione;
- **Relazioni d'uso con altri componenti**: questo package si relaziona con i *controllers* presenti nel package *Controllers*;
- **Classi contenute**:
  - ***RegisterUser***: view contenente il form dedicato alla registrazione di un utente, il quale può inserire i campi necessari e registrarsi così alla piattaforma. Contiene, inoltre, un link alla pagina di login.
    Si relaziona con le seguenti componenti: *UserRegistrationController*.
  - ***Login***: view contenente il form necessario affinchè l'utente possa effettuare il login ed autenticarsi al sistema. Contiene, inoltre, un link alla pagina di registrazione e uno alla pagina per il recupero della password.
    Si relaziona con le seguenti componenti: *LoginController*.

- ***PasswordRecovery***: view contenente il form dedicato al recupero della password di un utente, il quale può inserire l'indirizzo email e ricevere una nuova password con la quale autenticarsi al sistema. Contiene, inoltre, un link alla pagina di login.
  Si relaziona con le seguenti componenti: *PasswordRecoveryController*.
- ***API***: view contenente i risultati della ricerca effettuata, che permette di selezionare un risultato presente al suo interno.
  Si relaziona con le seguenti componenti: *APIController*.
- ***ProfileManager***: view contenente le informazioni del profilo personale di un utente registrato. Contiene, inoltre, l'informazione relativa al saldo del proprio conto virtuale.
  Si relaziona con le seguenti componenti: *ProfileManagerController*.
- ***ResetPassword***: view contenente il form dedicato al cambio di password di un utente autenticato, il quale può inserire la nuova password che intende utilizzare per i futuri login al sistema.
  Si relaziona con le seguenti componenti: *ResetPasswordController*.
- ***RegisterAPI***: view contenente il form per l'inserimento di una API da parte di un utente sviluppatore. Lo sviluppatore può inserire tutti i dati relativi al microservizio che intende esporre sul marketplace.
  Si relaziona con le seguenti componenti: *APIRegistrationController*.
- ***PolicyCall***: view contenente le informazioni relative alla policy per chiamate.
  Si relaziona con le seguenti componenti: *PolicyCallController*.
- ***PolicyTime***: view contenente le informazioni relative alla policy per tempo.
  Si relaziona con le seguenti componenti: *PolicyTimeController*.
- ***PolicyTraffic***: view contenente le informazioni relative alla policy per traffico dati.
  Si relaziona con le seguenti componenti: *PolicyTrafficController*.
- ***APIRegistered***: view contenente le informazioni di una API registrata sulla piattaforma.
  Si relaziona con le seguenti componenti: *APIRegisteredController*.
- ***APIPurchased***: view contenente le informazioni delle API acquistate da un cliente della piattaforma API Market.
  Si relaziona con le seguenti componenti: *APIPurchasedController*.
- ***APIList***: view contenente l'elenco delle API disponibili sul marketplace API Market.
  Si relaziona con le seguenti componenti: *APIListController*.
- ***TransactionsList***: view contenente l'elenco delle transazioni effettuate da un utente sul marketplace API Market.
  Si relaziona con le seguenti componenti: *TransactionsListController*.
- ***APIConfirmPurchase***: view contenente la conferma di un acquisto di una API.
  Si relaziona con le seguenti componenti: *APIConfirmPurchaseController*.
- ***AdminManager***: view contenente le operazioni per la gestione del profilo amministratore API Market.
  Si relaziona con le seguenti componenti: *AdminManagerController*.
- ***Category***: view contenente l'elenco delle categorie nell'API Market.
  Si relaziona con le seguenti componenti: *CategoryController*.
- ***ConfirmUpgradeToDeveloper***: view contenente la conferma dell'upgrade di un utente a sviluppatore nell'API Market.
  Si relaziona con le seguenti componenti: *ConfirmUpgradeToDeveloperController*.
- ***ConfirmLogin***: view contenente la conferma di login all'API Market.
  Si relaziona con le seguenti componenti: *ConfirmLoginController*.

- ***ConfirmUserRegistration***: view contenente la conferma di registrazione all'API Market.
  Si relaziona con le seguenti componenti: *ConfirmUserRegistrationController*.
- ***ConfirmAPIRegistration***: view contenente la conferma di registrazione di una API all'API Market.
  Si relaziona con le seguenti componenti: *ConfirmAPIRegistrationController*.
- ***UpgradeToDeveloper***: view contenente il modulo per diventare sviluppatore all'interno dell'API Market.
  Si relaziona con le seguenti componenti: *UpgradeToDeveloperController*.
- ***AdminAPIManager***: view contenente le operazioni dell'amministratore sulle API dell'API Market.
  Si relaziona con le seguenti componenti: *AdminAPIManagerController*.
- ***AdminAPIDetailsManager***: view contenente le statistiche di una API dell'API Market.
  Si relaziona con le seguenti componenti: *AdminAPIDetailsManagerController*.
- ***AdminUserManager***: view contenente le operazioni di un ammistratore sugli utenti dell'API Market.
  Si relaziona con le seguenti componenti: *AdminUserManagerController*.
- ***AdminLogin***: view contenente il form necessario affinchè l'amministratore possa effettuare il login ed autenticarsi al sistema.
  Si relaziona con le seguenti componenti: *AdminLoginController*.
- ***Logout***: view contenente la conferma di logout dall'API Market.
  Si relaziona con le seguenti componenti: *LogoutController*.
- ***UpdateInfoAPI***: view contenente il form dedicato modifica delle informazioni di una API presente nel marketplace API Market.
  Si relaziona con le seguenti componenti: *UpdateInfoAPIController*.
- ***RechargeCredits***: view contenente la possibilità di scegliere quanti crediti ricaricare sul conto personale tra i tagli disponibili.
  Si relaziona con le seguenti componenti: *RechargeCreditsController*.

  item ***ShowUserDetails***: view contenente le informazioni personali di un utente visualizzabili da altri fruitori del marketplace API Market.
  Si relaziona con le seguenti componenti: *ShowUserDetalisController*.
- ***AdminManager***: view contenente il form dedicato alla moderazione di un utente o di una API/microservizio da parte di un amministratore della piattaforma API Market.
  Si relaziona con le seguenti componenti: *AdminManagerController*.

5.2.2.5 Models

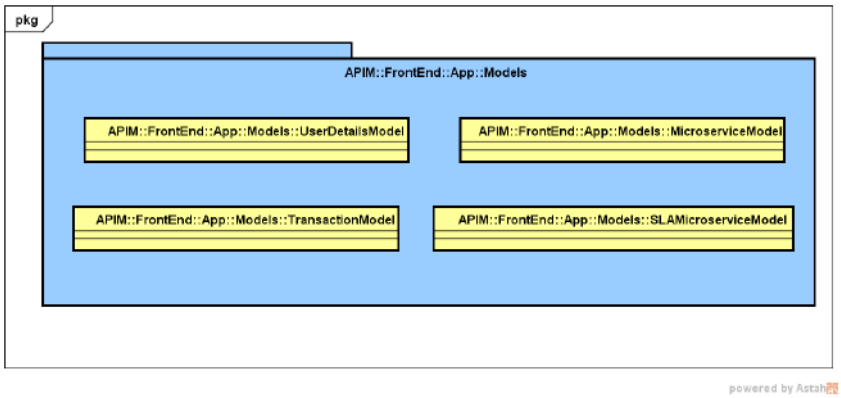

Figura 12: Package APIM::FrontEnd::App::Models

- **Padre**: App;
- **Descrizione**: package che contiene le classi che definiscono la business logic dell'applicazione;
- **Relazioni d'uso con altri componenti**: questo package si relaziona con il package *Controllers*;
- **Classi contenute**:
  - ***UserDetailsModel***: classe che rappresenta un utente e che contiene tutte le informazioni necessarie alla presentazione del contenuto di un utente, sia nella visualizzazione che nella gestione di un profilo.
    Si relaziona con le seguenti componenti: *LoginController*, *APIController*, *ProfileManagerController*, *ResetPasswordController*, *PasswordRecoveryController*.
  - ***MicroserviceModel***: classe che rappresenta un microservizio e che contiene tutte le informazioni necessarie alla presentazione del contenuto di un microservizio, sia nella visualizzazione che nella gestione.
    Si relaziona con le seguenti componenti: *APIRegiteredController*, *APIRegistrationController*, *PolicyCallController*, *PolicyTimeController*, *PolicyTrafficController*, *APIPurchasedController*, *APIListController*.
  - ***TransactionModel***: classe che rappresenta una transazione avvenuta e che contiene tutte le informazioni necessarie alla presentazione del contenuto di una transazione, sia nella visualizzazione che nella gestione.
    Si relaziona con le seguenti componenti: *TransactionsListController*, *APIPurchasedController*.
  - ***SLAMicroserviceModel***: classe che rappresenta la SLA di un microservizio e che contiene tutte le informazioni necessarie alla presentazione del contenuto di SLA di un microservizio, sia nella visualizzazione che nella gestione.
    Si relaziona con le seguenti componenti: *PolicyCallController*, *PolicyTimeController*, *PolicyTrafficController*, *APIRegistrationController*.

5.2.2.6 Controllers

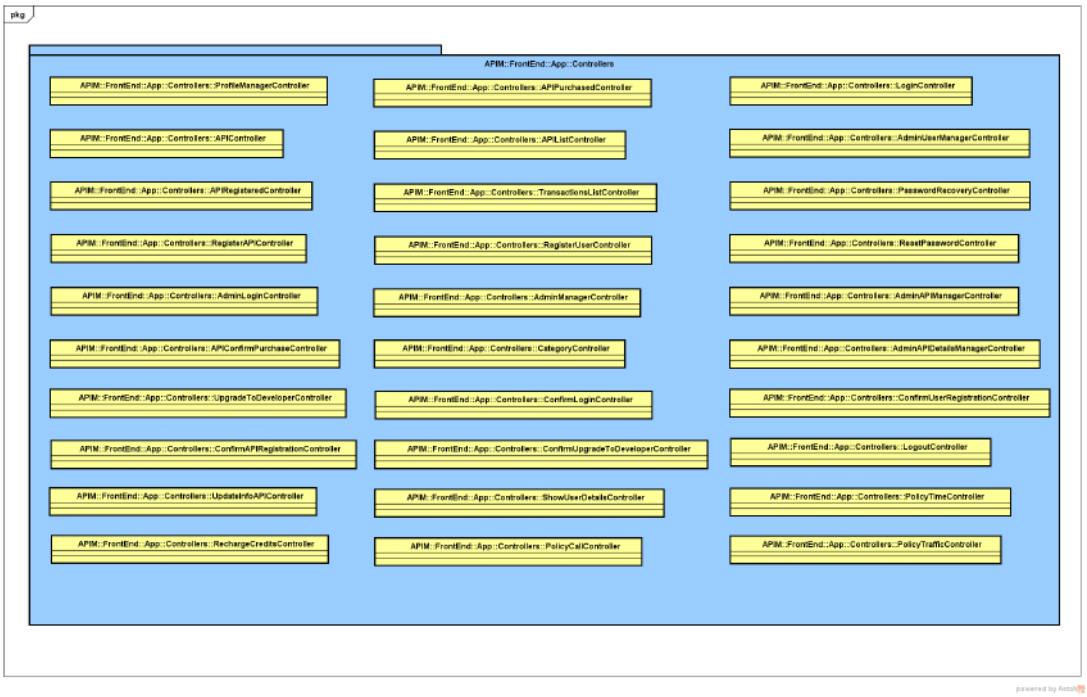

Figura 13: Package APIM::FrontEnd::App::Controllers

- **Padre**: App;
- **Descrizione**: package che contiene i *controllers* individuati per la parte front-end dell'applicazione, i quali consentono la gestione delle azioni utente dell'applicazione web;
- **Relazioni d'uso con altri componenti**: questo package si relaziona con i package *Views* e *Models*.
- **Classi contenute**:
  - ***RegisterUserController***: classe che permette di gestire la registrazione di un utente al sistema, fornendone le funzionalità preposte.
    Si relaziona con le seguenti componenti: *RegisterUser*.
  - ***LoginController***: classe che permette di gestire il login di un utente alla piattaforma API Market, fornendo le funzionalità di autenticazione al sistema, compresa la gestione di situazioni di errore di autenticazione.
    Si relaziona con le seguenti componenti: *UserDetailsModel*, *Login*.
  - ***PasswordRecoveryController***: classe che permette di gestire il ripristino della password dimenticata da un utente, fornendo tutte le funzionalità per il recupero della password dopo aver verificato l'identità dell'utente.
    Si relaziona con le seguenti componenti: *UserDetailsModel*, *PasswordRecovery*.
  - ***APIController***: classe che permette di gestire la ricerca di microservizi all'interno del marketplace API Market, fornendo all'utente le funzionalità di ricerca tramite categorie e keywords per sviluppatori e microservizi.
    Si relaziona con le seguenti componenti: *API*.
  - ***ProfileManagerController***: classe che permette di gestire il profilo personale di un utente, fornendo le funzionalità all'utente per poter modificare i propri dati.
    Si relaziona con le seguenti componenti: *UserDetailsModel*, *ProfileManager*.

- ***ResetPasswordController***: classe che permette di gestire il cambio password di un utente autenticato al sistema, fornendo le funzionalità per il salvataggio di una nuova password.
  Si relaziona con le seguenti componenti: *UserDetailsModel*, *ResetPassword*.
- ***RegisterAPIController***: classe che permette di gestire l'inserimento di una API, fornendo tutte le funzionalità atte alla corretta esposizione di un microservizio di uno sviluppatore, utente della piattaforma API Market.
  Si relaziona con le seguenti componenti: *MicroserviceModel*, *RegisterAPI*.
- ***PolicyCallController***: classe che permette di gestire le policy di vendita dei microservizi per chiamate.
  Si relaziona con le seguenti componenti: *PolicyCall*, *SLAMicroserviceModel*.
- ***PolicyTimeController***: classe che permette di gestire le policy di vendita dei microservizi per tempo.
  Si relaziona con le seguenti componenti: *PolicyTime*, *SLAMicroserviceModel*.
- ***PolicyTrafficController***: classe che permette di gestire le policy di vendita dei microservizi per traffico dati.
  Si relaziona con le seguenti componenti: *PolicyTraffic*, *SLAMicroserviceModel*.
- ***APIRegisteredController***: classe che permette di gestire le informazioni di una API precedentemente inserita.
  Si relaziona con le seguenti componenti: *MicroserviceModel*, *APIRegistered*.
- ***APIPurchasedController***: classe che permette di gestire l'acquisto e le relative informazioni di una API, fornendo l'API Key per l'utilizzo del cliente.
  Si relaziona con le seguenti componenti: *APIPurchased*, *MicroserviceModel*, *TransactionModel*.
- ***APIListController***: classe che permette di gestire l'elenco dei microservizi presenti sul marketplace API Market.
  Si relaziona con le seguenti componenti: *APIList*, *MicroserviceModel*.
- ***TransactionsListController***: classe che permette di gestire lo storico delle transazioni di un utente del marketplace API Market.
  Si relaziona con le seguenti componenti: *TransactionsList*, *TransactionsModel*.
- ***AdminManagerController***: classe che permette di gestire il profilo di un amministratore della piattaforma API Market, fornendo le funzionalità per poter modificare i propri dati e moderare utenti ed API.
  Si relaziona con le seguenti componenti: *AdminManager*.
- ***CategoryController***: classe che permette di gestire l'elenco delle categoria nell'API Market.
  Si relaziona con le seguenti componenti: *Category*.
- ***ConfirmUpgradeToDeveloperController***: classe che permette di gestire la conferma dell'upgrade di un utente a sviluppatore nell'API Market.
  Si relaziona con le seguenti componenti: *ConfirmUpgradeToDeveloper*.
- ***ConfirmLoginController***: classe che permette di gestire la conferma di login all'API Market.
  Si relaziona con le seguenti componenti: *ConfirmLogin*.
- ***ConfirmUserRegistrationController***: classe che permette di gestire la conferma di registrazione all'API Market.
  Si relaziona con le seguenti componenti: *ConfirmUserRegistration*.
- ***ConfirmAPIRegistrationController***: classe che permette di gestire la conferma di registrazione di una API all'API Market.
  Si relaziona con le seguenti componenti: *ConfirmAPIRegistration*.

- ***UpgradeToDeveloperController***: classe che permette di gestire il modulo per diventare sviluppatore all'interno dell'API Market.
  Si relaziona con le seguenti componenti: *UpgradeToDeveloper*.
- ***AdminAPIManagerController***: classe che permette di gestire le operazioni dell'amministratore sulle API dell'API Market.
  Si relaziona con le seguenti componenti: *AdminAPIManager*.
- ***AdminAPIDetailsManagerController***: classe che permette di gestire le statistiche di una API dell'API Market.
  Si relaziona con le seguenti componenti: *AdminAPIDetailsManager*.
- ***AdminUserManagerController***: classe che permette di gestire le operazioni di un ammistratore sugli utenti dell'API Market.
  Si relaziona con le seguenti componenti: *AdminUserManager*.
- ***AdminLoginController***: classe che permette di gestire il form necessario affinchè l'amministratore possa effettuare il login ed autenticarsi al sistema.
  Si relaziona con le seguenti componenti: *AdminLogin*.
- ***LogoutController***: classe che permette di gestire la conferma di logout dall'API Market.
  Si relaziona con le seguenti componenti: *Logout*.
- ***UpdateInfoAPIController***: classe che permette di gestire il form dedicato modifica delle informazioni di una API presente nel marketplace API Market.
  Si relaziona con le seguenti componenti: *UpdateInfoAPI*.
- ***RechargeCreditsController***: classe che permette di gestire la possibilità di scegliere quanti crediti ricaricare sul conto personale tra i tagli disponibili.
  Si relaziona con le seguenti componenti: *RechargeCredits*.
- ***ShowUserDetalisController***: classe che permette di gestire le informazioni personali di un utente visualizzabili da altri fruitori del marketplace API Market.
  Si relaziona con le seguenti componenti: *ShowUserDetalis*
- ***AdminModerationController***: classe che permette di gestire la moderazione di un utente (cliente o sviluppatore che sia) e di API, fornendo le funzionalità per la sospensione e rimozione.
  Si relaziona con le seguenti componenti: *AdminModeration*.

5.3 Back-end

Nella parte back-end sono presenti i package *Gateway* e *Services* strutturati secondo un'architettura a microservizi, i quali hanno una dipendenza verso l'interfaccia *ServiceInteractionHandler*.

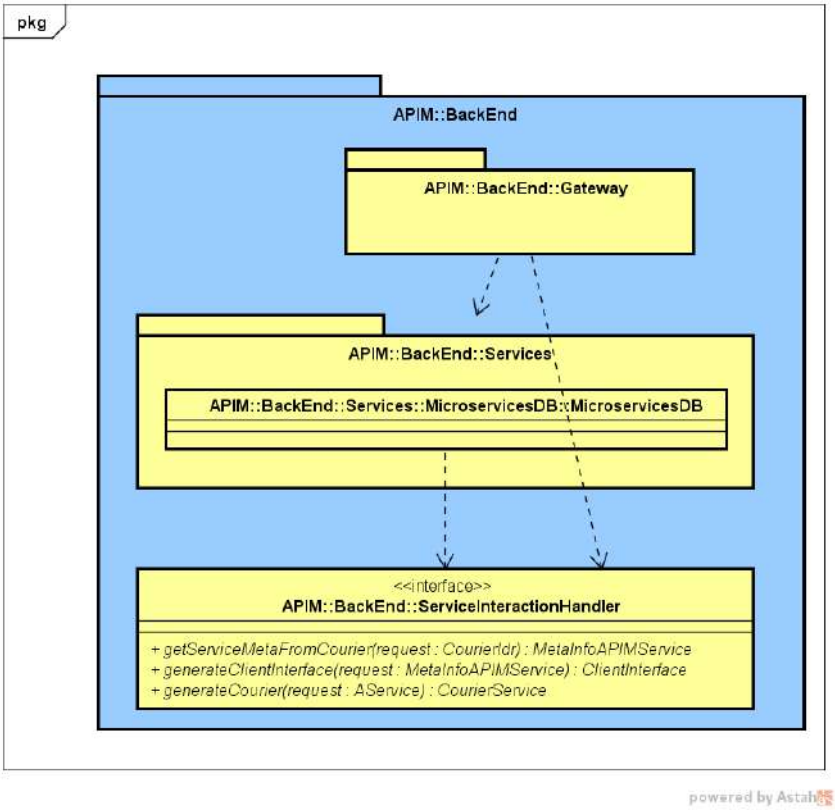

Figura 14: Package APIM::BackEnd

- **Padre**: APIM;
- **Descrizione**: package contenente le componenti del back-end dell'applicazione;
- **Package contenuti**:
  - ***Gateway***: package contenente le classi e le interfacce per il funzionamento dell'API Gateway;
  - ***Services***: package contenente tutti i microservizi per le comunicazioni con i database.
- **Classi contenute**:
  - ***ServiceInteractionHandler***: interfaccia per la gestione delle comunicazioni dell'API Gateway e dei *services*.

5.3.1 Gateway

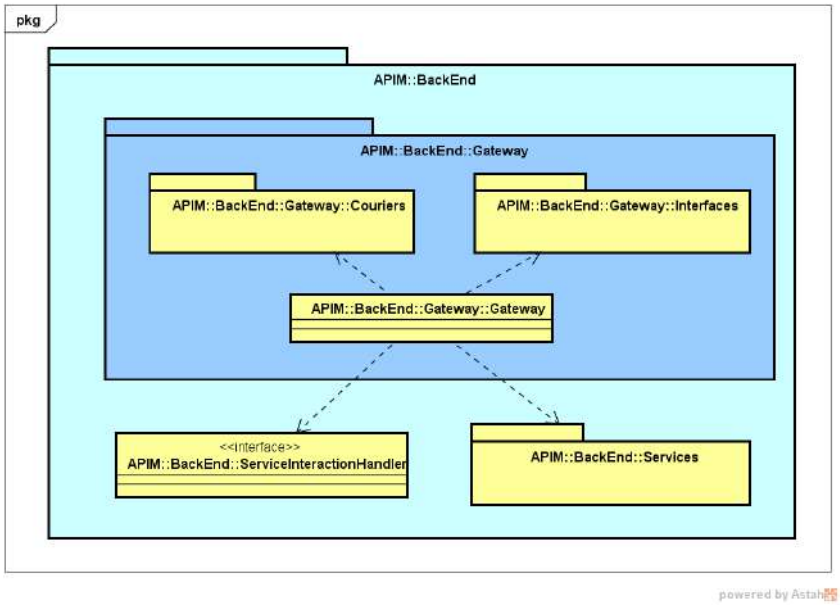

Figura 15: Package APIM::BackEnd::Gateway

- **Padre**: BackEnd;
- **Descrizione**: package contenente la componente principale del lato back-end. Pur essendo integrato nella sistema, è un modulo che svolge funzioni separate alla piattaforma vera e propria: infatti, l'API Gateway si occupa di controllare e reindirizzare le chiamate effettuate dai clienti ai microservizi (prodotti) inseriti nel marketplace API Market, verificandone la validità (credenziali, chiave) e monitorandone l'uso (SLA, utilizzi rimanenti).
- **Relazioni d'uso con altri componenti**: la classe *Gateway* comunica con le classi contenute all'interno del package *Services*. Inoltre, necessita delle interfacce contenute nel package *Interfaces* e crea le sessioni *couriers* Jolie, archiviandole nel package *Couriers*.
- **Package contenuti**:
  - ***Couriers***: package contenente l'archivio delle sessioni couriers dei microservizi Jolie. Le sessioni couriers vengono create dal gateway ed utilizzate da *ServiceInteractionHandler*. Inoltre, forniscono i dati necessari al funzionamento del package *Interfaces*. Le sessioni *couriers* permettono l'overloading dei messaggi inviati nelle chiamate dei microservizi al gateway, così da allegare alla richiesta le informazioni per raggiungere il microservizio target.
  - ***Interfaces***: package contenente le interfacce necessarie al funzionamento dell'API Gateway. Si occupa dei dati riguardanti le chiamate ai microservizi, in particolar modo, il tipo di operazione richiesta e le informazioni riguardanti l'interfaccia del microservizio target.
- **Classi contenute**:
  - ***Gateway***: classe che rappresenta la struttura dell'API Gateway della piattaforma API Market.

5.3.2 Services

Tale package contiene tutti i servizi appartenenti al lato back-end della piattaforma, eccezion fatta per il gateway già descritto nella sezione soprastante. *Services* contiene tutte le classi di comunicazione con i database che permetteranno il corretto funzionamento dell'applicazione.

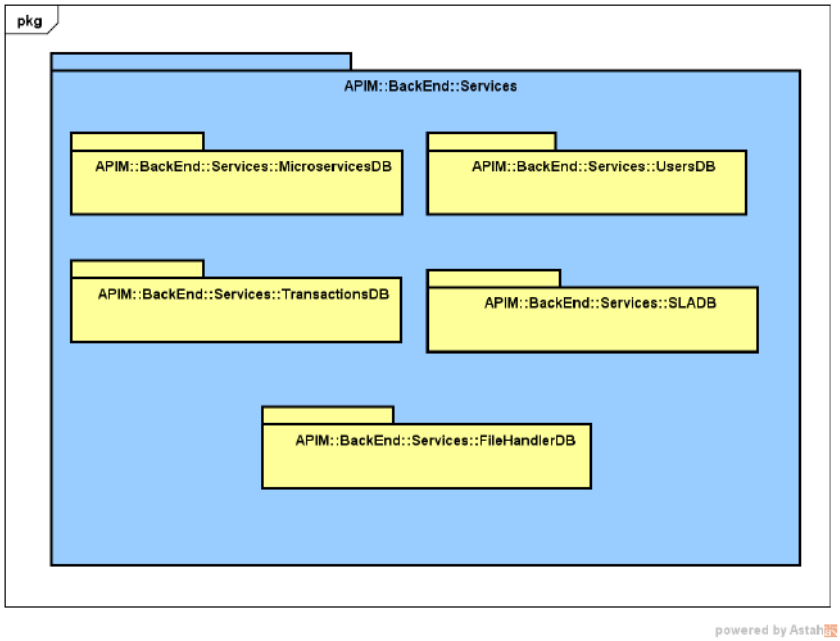

Figura 16: Package APIM::BackEnd::Services

- **Padre**: BackEnd;
- **Descrizione**: package che contiene tutti i package relativi ai microservizi del back-end dell'applicazione;
- **Relazioni d'uso con altri componenti**: questo package si relaziona con l'API Gateway;
- **Package contenuti**:
  - ***UsersDB***: package contenente le classi che permettono la comunicazione con il database relativo agli utenti del sistema;
  - ***MicroservicesDB***: package contenente le classi che permettono la comunicazione con il database relativo ai microservizi registrati sulla piattaforma;
  - ***TransactionsDB***: package contenente le classi che permettono la comunicazione con il database relativo alle transazioni degli utenti del sistema;
  - ***SLADB***: package contenente le classi che permettono la comunicazione con il database relativo alle informazioni di SLA dei microservizi utilizzati tramite l'API Gateway della piattaforma API Market;
  - ***FileHandlerDB***: package contenente le classi che permettono la gestione dei file.

5.3.2.1 UsersDB

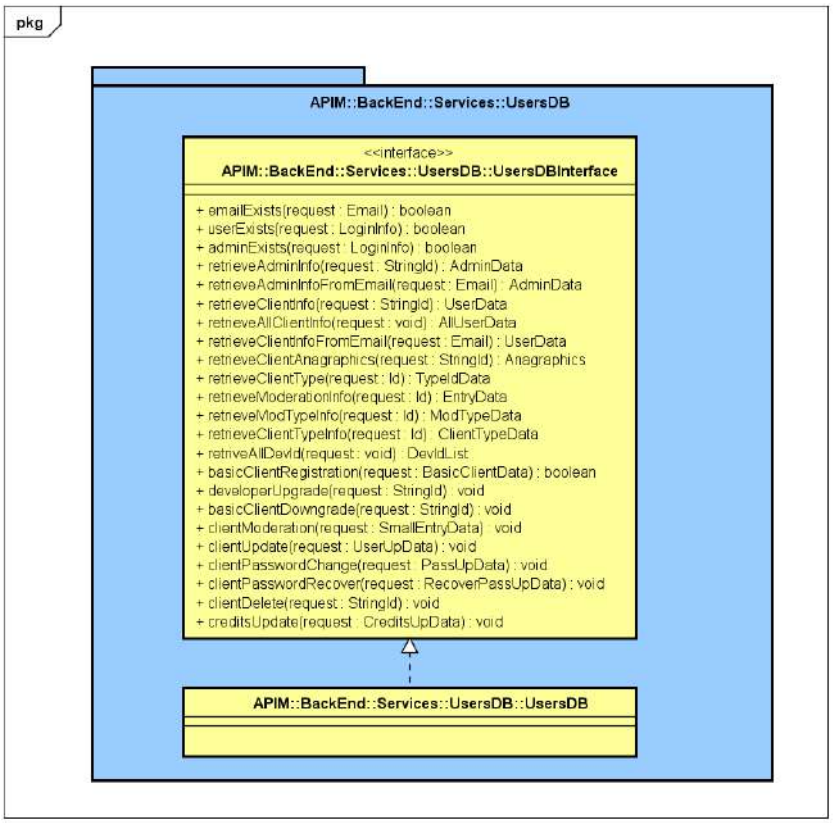

Figura 17: Package APIM::BackEnd::Services::UsersDB

- **Padre**: Services;
- **Descrizione**: package che contiene le classi di comunicazione con il database dell'applicazione relativo alle informazioni (anagrafiche e personali) degli utenti (admin, clienti, sviluppatori) del sistema;
- **Relazioni d'uso con altri componenti**: comunica con il *Gateway* per permettere l'identificazione, l'autenticazione e la gestione degli utenti, clienti e sviluppatori della piattaforma API Market;
- **Classi contenute**:
  - ***UsersDBInterface***: interfaccia che raccoglie tutte le *operation* riguardanti il database degli utenti;
  - ***UsersDB***: classe che implementa l'interfaccia *UsersDBInterface*.

5.3.2.2 MicroservicesDB

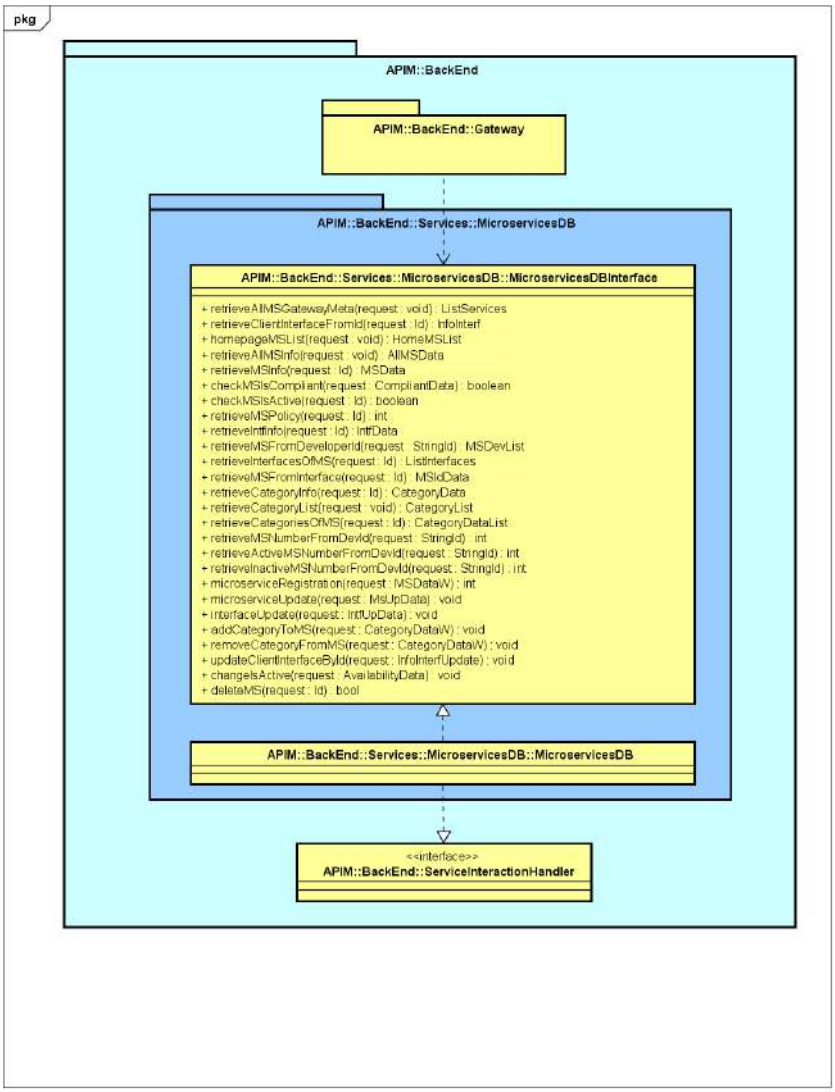

Figura 18: Package APIM::BackEnd::Services::MicroservicesDB

- **Padre**: Services;
- **Descrizione**: package che contiene le classi di comunicazione con il database dell'applicazione relativo ai microservizi registrati nella piattaforma;
- **Relazioni d'uso con altri componenti**: comunica con il *Gateway* per permettere l'identificazione e l'utilizzo dei microservizi, e garantire correttezza, efficienza e validità delle chiamate;
- **Classi contenute**:
  - ***MicroservicesDBInterface***: interfaccia che raccoglie tutte le *operation* riguardanti il database dei microservizi;
  - ***MicroservicesDB***: classe che implementa le interfacce *MicroservicesDBInterface* e *ServiceInteractionHandler*.

5.3.2.3 TransactionsDB

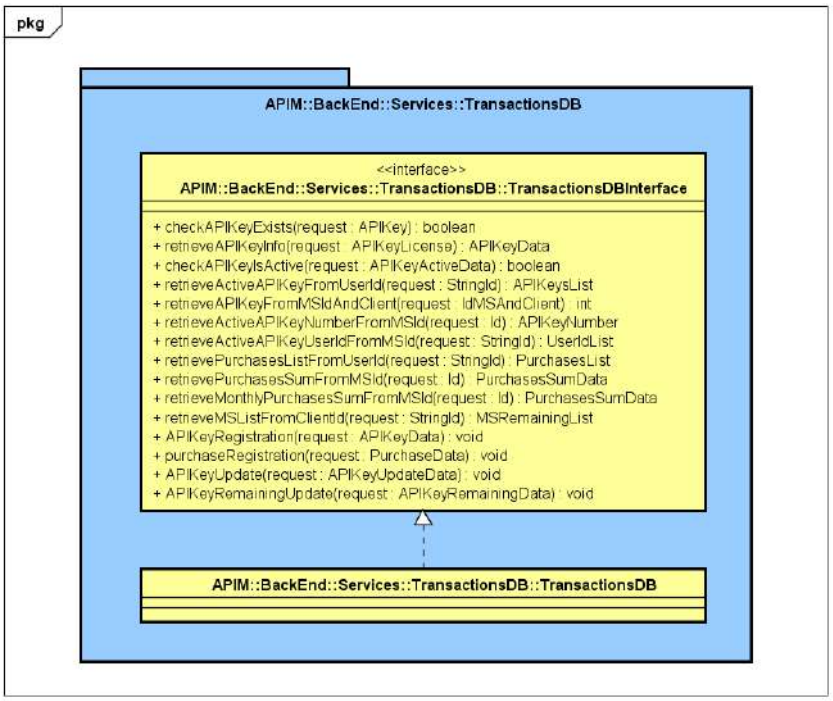

Figura 19: Package APIM::BackEnd::Services::TransactionsDB

- **Padre**: Services;
- **Descrizione**: package che contiene le classi di comunicazione con il database dell'applicazione che si occupa di immagazzinare i dati sulle transazioni;
- **Relazioni d'uso con altri componenti**: comunica con il *Gateway* per verificare la validità delle API Key e/o aggiornare i dati relativi (usi residui della chiave) alle chiamate ai microservizi effettuate;
- **Classi contenute**:
  - ***TransactionsDBInterface***: interfaccia che raccoglie tutte le *operation* riguardanti il database delle transazioni;
  - ***TransactionsDB***: classe che implementa l'interfaccia *TransactionsDBInterface*.

5.3.2.4 SLADB

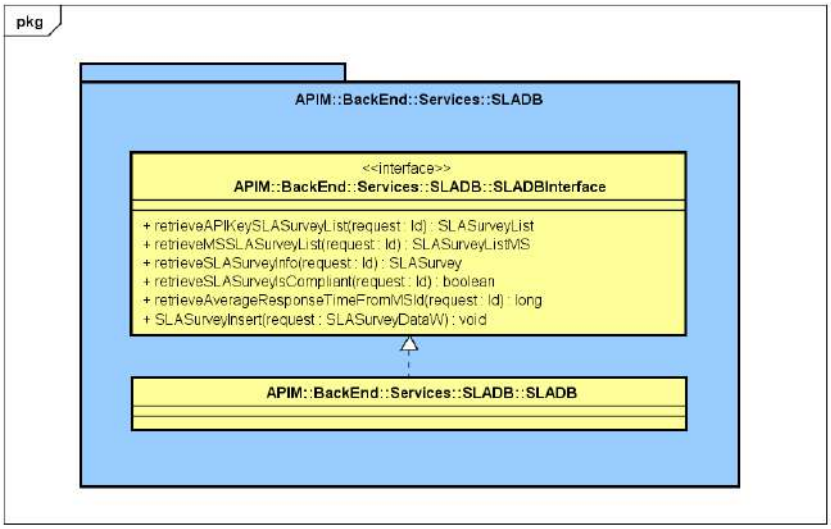

Figura 20: Package APIM::BackEnd::Services::SLADB

- **Padre**: Services;
- **Descrizione**: package che contiene le classi di comunicazione con il database che si occupa di immagazzinare e trattare i dati relativi al Service Level Agreement (SLA);
- **Relazioni d'uso con altri componenti**: comunica con il *Gateway* per aggiornare i dati delle performance di risposta di ogni microservizio utilizzato;
- **Classi contenute**:
  - ***SLADBInterface***: interfaccia che raccoglie tutte le *operation* riguardanti il database delle informazioni di SLA;
  - ***SLADB***: classe che implementa l'interfaccia *SLADBInterface*.

5.3.2.5 FileHandlerDB

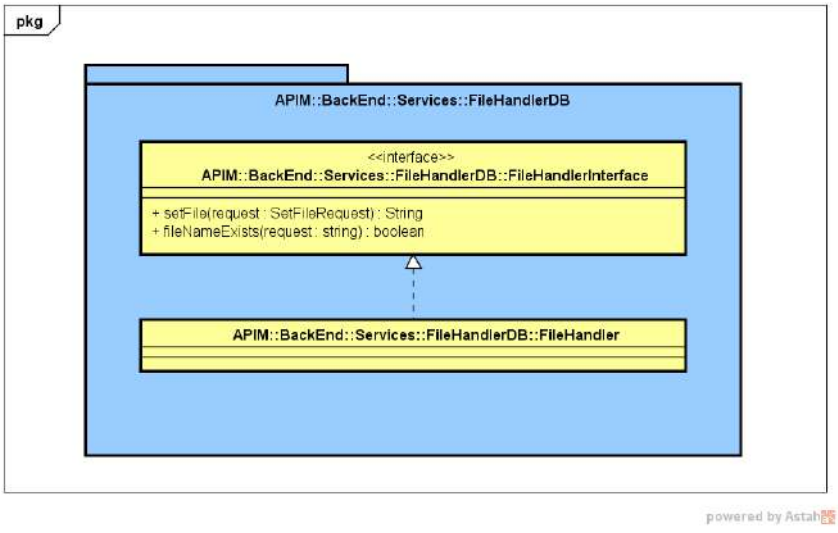

Figura 21: Package APIM::BackEnd::Services::FileHandlerDB

- **Padre**: Services;
- **Descrizione**: package che si occupa della gestione dei file, e presenta le classi di comunicazione con il database per tale finalità;
- **Relazioni d'uso con altri componenti**: comunica con il *Gateway* per recuperare i file legati alle interfacce dei microservizi;
- **Classi contenute**:
  - ***FileHandlerInterface***: interfaccia che raccoglie tutte le *operation* riguardanti la gestione dei file;
  - ***FileHandler***: classe che implementa l'interfaccia *FileHandlerInterface*.

6 Diagrammi di sequenza

6.1 Introduzione

Un diagramma di sequenza è un diagramma previsto dallo standard UML, utilizzato per descrivere un particolare scenario. Uno scenario è una determinata sequenza di azioni in cui tutte le scelte sono state già effettuate. In pratica, nel diagramma non compaiono nè scelte, né flussi alternativi.
Di seguito, verranno presi in considerazione i diagrammi di sequenza per i principali scenari che il team ha individuato nell'utilizzo dell'applicazione API Market:

- Ricerca di una API;
- Acquisto di una API;
- Registrazione di una API;
- Ricarica del saldo del conto virtuale.

N.B.: i diagrammi di sequenza presentati sono descritti ad alto livello.

6.2 Ricerca API

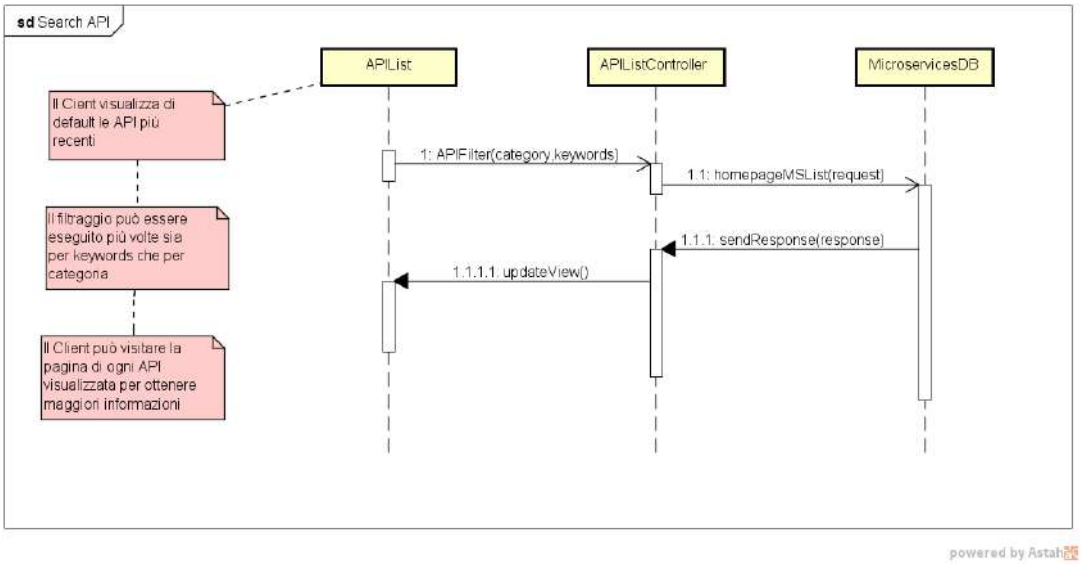

Figura 22: Diagramma di sequenza per la ricerca di una API

- **Pre-condizioni**: l'utente si trova nella homepage dell'applicazione;
- **Post-condizioni**: l'utente ha visualizzato le API secondo la categoria e le keywords inserite;
- **Descrizione**: di default, la homepage di API Market mostra le API registrate più recentemente. L'utente può immettere delle keywords nell'apposito riquadro per alterare i risultati, e scegliere di restringere la ricerca ad una singola categoria. I risultati visualizzati cambiano dinamicamente non appena l'applicazione web riceve risposta dai servizi che la collegano ai database.

6.3 Acquisto API

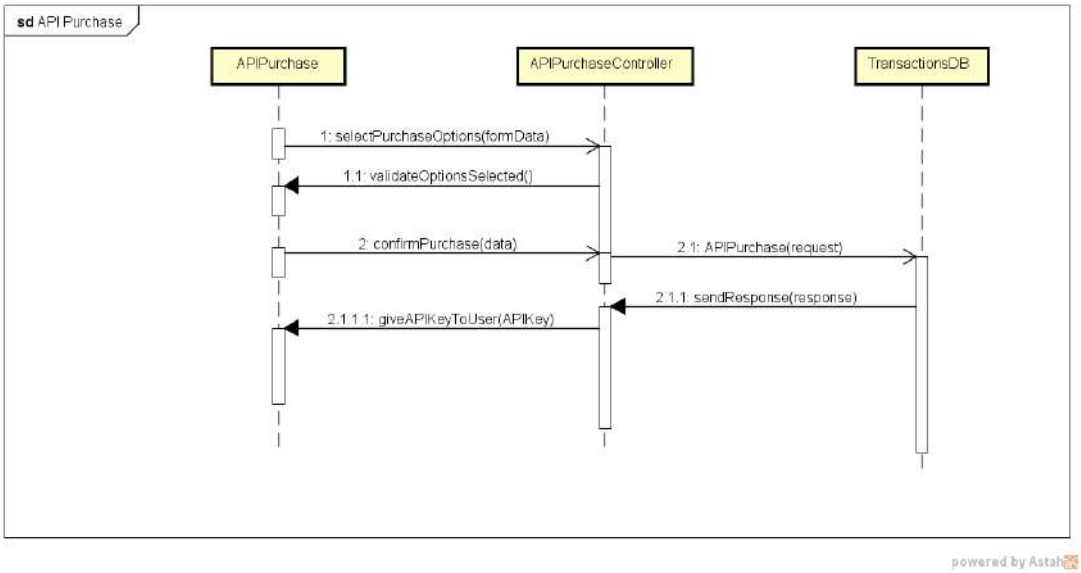

Figura 23: Diagramma di sequenza per l'acquisto di una API

- **Pre-condizioni**: il cliente si trova nella schermata di acquisto di una specifica API;
- **Post-condizioni**: il cliente ha ricevuto l'API Key per l'API acquistata, secondo le modalità da lui scelte, e gli sono stati sottratti i corrispondenti crediti;
- **Descrizione**: il cliente compila il form per l'acquisto dell'API desiderata, visualizzando il preventivo di crediti spesi in base ai parametri scelti. Confermando l'acquisto (che può essere anche un rinnovo), i services di API Market provvedono a generare una apikey ed a scalare i crediti dall'account utente. L'utente potrà visualizzare l'API Key ed un messaggio di ringraziamento.

6.4 Registrazione API

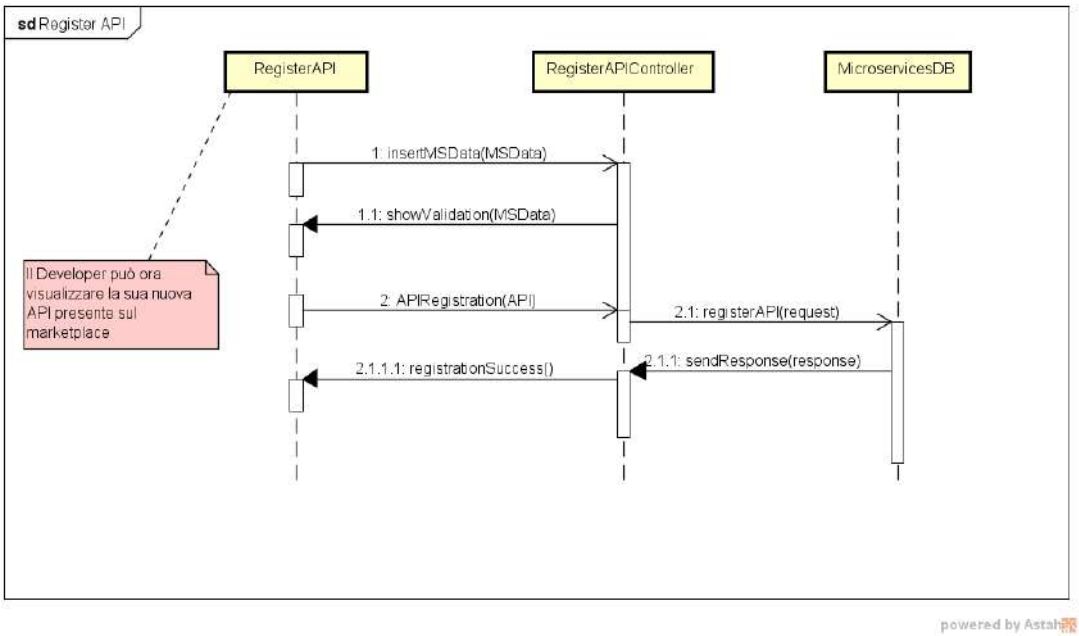

Figura 24: Diagramma di sequenza per la registrazione di una API

- **Pre-condizioni**: lo sviluppatore si trova nella schermata di registrazione di una nuova API;
- **Post-condizioni**: lo sviluppatore ha registrato la sua nuova API ed essa è ora disponibile in API Market;
- **Descrizione**: Lo sviluppatore compila il form per la registrazione di una nuova API, provvedendo anche a caricare sul server di API Market i file del logo e della documentazione PDF. Confermando la registrazione, i services di API Market provvedono ad inserire nel database la nuova API e renderla accessibile attraverso il Gateway. A registrazione avvenuta, lo sviluppatore riceve un messaggio di successo.

6.5 Ricarica saldo conto virtuale

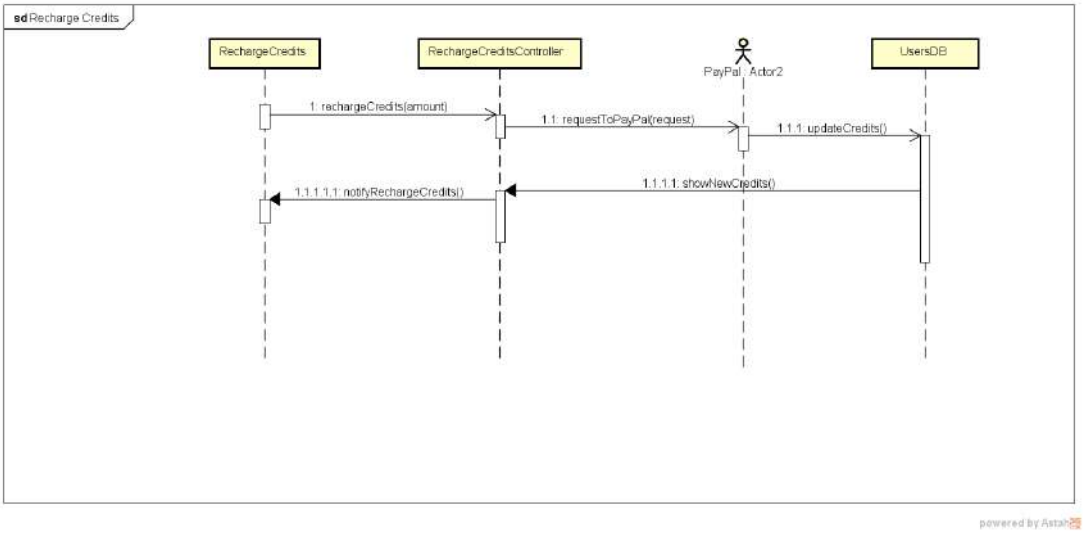

Figura 25: Diagramma di sequenza per la ricarica del saldo del conto virtuale

- **Pre-condizioni**: il cliente si trova nella schermata di ricarica del proprio conto virtuale;
- **Post-condizioni**: il cliente ha effettuato la ricarica dei propri crediti;
- **Descrizione**: il cliente stabilisce l'ammontare dei crediti da ricaricare nel proprio conto virtuale. A decisione avvenuta, verrà reindirizzato alla procedura di acquisto di PayPal, che lo guiderà nell'acquisto. Portata a termine la transazione, PayPal avvertirà i services di API Market che si occuperanno di incrementare i crediti del conto virtuale del cliente.

7 Tracciamento

7.1 Tracciamento Classi - Requisiti

Nome Classe	Codice Requisito
APIM::FrontEnd::App::Index	RFO1
	RFO2
	RFO4
	RFO4.1
	RFO4.2
	RFO4.3
	RFO4.3.1
	RFO4.3.2
	RFO4.3.3
	RFO4.3.4
	RFO4.3.5
	RFO11
APIM::FrontEnd::App::Views::RegisterUser	RFO1
	RFO1.1
	RFO1.2
	RFO1.3
	RFO1.4
	RFO1.5
	RFO1.6
	RFO1.7
	RFO1.8
	RFO1.9
	RFO1.10
APIM::FrontEnd::App::Views::Login	RFO2
	RFO2.1
	RFO2.1.1
	RFO2.1.2
	RFO2.1.3

	RFO2.1.4
APIM::FrontEnd::App::Views::Logout	RFO11
APIM::FrontEnd::App::Views::PasswordRecovery	RFD3
	RFD3.1
	RFD3.2
	RFD3.3
	RFD3.4
APIM::FrontEnd::App::Views::API	RFO4.3.1
	RFO4.3.2
	RFO4.3.3
	RFO4.3.4
	RFO4.3.5
	RFO5
	RFO5.1
	RFO5.2
	RFO5.3
	RFO5.4
	RFO5.5
	RFO5.5.1
	RFO5.5.2
	RFO5.6
	RFO5.6.1
	RFO5.6.2
	RFO5.7
	RFD5.7.1
	RFO5.7.2
	RFO5.8
	RFO5.9
	RFO5.10
	RFO5.11
	RFO5.12
	RFO5.13

	RFO7
	RFO7.1
	RFO7.1.1
	RFO7.1.2
	RFO7.1.3
	RFO7.2
	RFO7.3
	RFO7.4
	RFO7.5
	RFD7.5.1
	RFO7.5.2
	RFO7.5.3
	RFO7.6
APIM::FrontEnd::App::Views::ProfileManager	RFO6
	RFO10
	RFO10.1
	RFO10.1.1
	RFO10.1.1.1
	RFO10.1.1.2
	RFO10.1.1.3
	RFO10.1.1.4
	RFO10.1.1.5
	RFO10.1.1.6
	RFO10.1.2
	RFO10.1.2.1
	RFO10.1.2.2
	RFO10.1.2.3
	RFO10.1.2.4
	RFO10.1.2.5
	RFD10.1.2.6
	RFO10.1.2.7
	RFO10.1.2.8

	RFD10.1.2.8
	RFO10.2
	RFO10.2.1
	RFO10.2.2
	RFO10.2.2.1
	RFO10.2.2.2
	RFO10.3.3
	RFO10.3.3.1
	RFO10.3.3.2
	RFO10.3.3.3
	RFO10.3.3.4
	RFO11
APIM::FrontEnd::App::Views::ResetPassword	RFD3
	RFD3.1
	RFD3.2
	RFD3.3
	RFD3.4
APIM::FrontEnd::App::Views::RegisterAPI	RFO9
	RFO9.1
	RFO9.2
	RFO9.3
	RFO9.4
	RFO9.5
	RFD9.6
	RFD9.7
	RFO9.8
	RFO9.8.1
	RFO9.8.2
	RFO9.8.3
	RFO9.9
	RFD9.10
	RFD9.11

	RFO9.12
	RFD9.13
APIM::FrontEnd::App::Views::PolicyCall	RFD5.7.1
	RFO5.8
	RFO5.12
APIM::FrontEnd::App::Views::PolicyTime	RFD5.7.1
	RFO5.8
	RFO5.12
APIM::FrontEnd::App::Views::PolicyTraffic	RFD5.7.1
	RFO5.8
	RFO5.12
APIM::FrontEnd::App::Views::APIRegistered	RFO8
	RFO8.1
	RFO8.2
	RFO8.2.1
	RFO8.2.2
	RFO8.2.3
	RFO8.2.4
	RFO8.2.4.1
	RFO8.2.4.2
	RFD8.2.4.3
	RFO8.2.4.4
	RFO8.2.4.5
	RFO8.2.4.6
	RFO8.2.4.7
	RFO8.2.4.8
	RFO8.2.4.9
	RFO8.2.4.10
	RFO8.2.4.11
	RFO8.2.5
	RFO8.2.6
	RFO8.2.7

	RFO8.2.8
	RFO8.2.9
	RFO8.2.9.1
	RFO8.2.9.2
	RFO9
	RFO9.1
	RFO9.2
	RFD9.3
	RFO9.4
	RFO9.5
	RFD9.6
	RFD9.7
	RFO9.8
	RFO9.8.1
	RFO9.8.2
	RFO9.8.3
	RFO9.9
	RFD9.10
	RFD9.11
	RFO9.12
	RFD9.13
APIM::FrontEnd::App::Views::APIPurchased	RFO6
	RFO6.1
	RFO6.2
	RFO6.2.1
	RFO6.2.2
	RFO6.2.3
	RFO6.2.4
	RFO6.2.5
	RFD6.2.6
	RFD6.2.7
	RFO7.5.2

	RFO7.5.3
APIM::FrontEnd::App::Views::APIList	RFO4.3.1
	RFO4.3.2
	RFO4.3.3
	RFO4.3.4
	RFO4.3.5
	RFO5
	RFO5.1
	RFO5.2
	RFO5.3
	RFO5.4
	RFO5.5
	RFO5.5.1
	RFO5.5.2
	RFO5.6
	RFO5.6.1
	RFO5.6.2
	RFO5.7
	RFD5.7.1
	RFO5.7.2
	RFO5.8
	RFO5.9
	RFO5.10
	RFO5.11
	RFO5.12
	RFO5.13
	RFO7
APIM::FrontEnd::App::Views::TransactionsList	RF010.3
	RF010.3.1
	RF010.3.1.1
	RF010.3.1.2
	RF010.3.2

	RF010.3.2.1
	RF010.3.2.2
	RF010.3.2.3
	RF010.3.2.4
	RF010.3.2.5
APIM::FrontEnd::App::Views::AdminManager	RFO12
	RFO12.1
	RFO12.1.1
	RFO12.1.1.1
	RFO12.1.1.1.1
	RFO12.1.1.1.2
	RFD12.1.1.1.3
	RFD12.1.1.1.4
	RFO12.1.1.1.5
	RFO12.1.1.1.5.1
	RFF12.1.1.1.5.2
	RFD12.1.1.1.6
	RFO12.1.1.2
	RFO12.1.1.2.1
	RFO12.1.1.2.2
	RFD12.1.1.3
	RFO12.1.1.3.1
	RFD12.1.1.3.2
	RFO12.1.1.3.3
APIM::FrontEnd::App::Views::AdminUserManager	RFO12.2
	RFO12.2.1
	RFO12.2.1.1
	RFO12.2.1.1.1
	RFO12.2.1.1.2
	RFO12.2.1.2
	RFO12.2.1.2.1
	RFO12.2.1.2.2

	RFO12.2.1.3
	RFO12.2.1.4
	RFO12.2.1.5
	RFO12.2.1.5.1
	RFO12.2.1.5.2
APIM::FrontEnd::App::Views::AdminAPIManager	RFO12.2
	RFO12.2.1
	RFO12.2.1.1
	RFO12.2.1.1.1
	RFO12.2.1.1.2
	RFO12.2.1.2
	RFO12.2.1.2.1
	RFO12.2.1.2.2
	RFO12.2.1.3
	RFO12.2.1.4
	RFO12.2.1.5
	RFO12.2.1.5.1
	RFO12.2.1.5.2
APIM::FrontEnd::App::Models::UserDetailsModel	RFO1
	RFO1.1
	RFO1.2
	RFO1.3
	RFO1.4
	RFO1.5
	RFO1.6
	RFO1.7
	RFO1.8
	RFO1.9
	RFO1.10
	RFO6
	RFO10
	RFO10.1

	RFO10.1.1
	RFO10.1.1.1
	RFO10.1.1.2
	RFO10.1.1.3
	RFO10.1.1.4
	RFO10.1.1.5
	RFO10.1.1.6
	RFO10.1.2
	RFO10.1.2.1
	RFO10.1.2.2
	RFO10.1.2.3
	RFO10.1.2.4
	RFO10.1.2.5
	RFD10.1.2.6
	RFO10.1.2.7
	RFO10.1.2.8
	RFD10.1.2.8
	RFO10.2
	RFO10.2.1
	RFO10.2.2
	RFO10.2.2.1
	RFO10.2.2.2
	RFO10.3.3
	RFO10.3.3.1
	RFO10.3.3.2
	RFO10.3.3.3
	RFO10.3.3.4
	RFO11
APIM::FrontEnd::App::Models::MicroserviceModel	RFO4.3.1
	RFO4.3.2
	RFO4.3.3
	RFO4.3.4

RFO4.3.5
RFO5
RFO5.1
RFO5.2
RFO5.3
RFO5.4
RFO5.5
RFO5.5.1
RFO5.5.2
RFO5.6
RFO5.6.1
RFO5.6.2
RFO5.7
RFD5.7.1
RFO5.7.2
RFO5.8
RFO5.9
RFO5.10
RFO5.11
RFO5.12
RFO5.13
RFO6
RFO6.1
RFO6.2
RFO6.2.1
RFO6.2.2
RFO6.2.3
RFO6.2.4
RFO6.2.5
RFD6.2.6
RFD6.2.7
RFO7

	RFO7.1
	RFO7.1.1
	RFO7.1.2
	RFO7.1.3
	RFO7.2
	RFO7.3
	RFO7.4
	RFO7.5
	RFD7.5.1
	RFO7.5.2
	RFO7.5.3
	RFO7.6
	RFO8
	RFO8.1
	RFO8.2
	RFO8.2.1
	RFO8.2.2
	RFO8.2.3
	RFO8.2.4
	RFO8.2.4.1
	RFO8.2.4.2
	RFD8.2.4.3
	RFO8.2.4.4
	RFO8.2.4.5
	RFO8.2.4.6
	RFO8.2.4.7
	RFO8.2.4.8
	RFO8.2.4.9
	RFO8.2.4.10
	RFO8.2.4.11
	RFO8.2.5
	RFO8.2.6

	RFO8.2.7
	RFO8.2.8
	RFO8.2.9
	RFO8.2.9.1
	RFO8.2.9.2
	RFO9
	RFO9.1
	RFO9.2
	RFD9.3
	RFO9.4
	RFO9.5
	RFD9.6
	RFD9.7
	RFO9.8
	RFO9.8.1
	RFO9.8.2
	RFO9.8.3
	RFO9.9
	RFD9.10
	RFD9.11
	RFO9.12
	RFD9.13
APIM::FrontEnd::App::Models::TransactionModel	RFO10.2
	RFO10.2.1
	RFO10.2.2
	RFO10.2.2.1
	RFO10.2.2.2
	RFO10.3.3
	RFO10.3.3.1
	RFO10.3.3.2
	RFO10.3.3.3
	RFO10.3.3.4

	RF010.3
	RF010.3.1
	RF010.3.1.1
	RF010.3.1.2
	RF010.3.2
	RF010.3.2.1
	RF010.3.2.2
	RF010.3.2.3
	RF010.3.2.4
	RF010.3.2.5
APIM::FrontEnd::App::Models::SLAMicroserviceModel	RFO5.7
	RFD5.7.1
	RFO5.7.2
APIM::FrontEnd::App::Controllers::RegisterUserController	RFO1
	RFO1.1
	RFO1.2
	RFO1.3
	RFO1.4
	RFO1.5
	RFO1.6
	RFO1.7
	RFO1.8
	RFO1.9
	RFO1.10
APIM::FrontEnd::App::Controllers::LoginController	RFO2
	RFO2.1
	RFO2.1.1
	RFO2.1.2
	RFO2.1.3
	RFO2.1.4
APIM::FrontEnd::App::Controllers::LogoutController	RFO11
APIM::FrontEnd::App::Controllers::PasswordRecoveryController	RFD3

	RFD3.1
	RFD3.2
	RFD3.3
	RFD3.4
APIM::FrontEnd::App::Controllers::APIController	RFO4.3.1
	RFO4.3.2
	RFO4.3.3
	RFO4.3.4
	RFO4.3.5
APIM::FrontEnd::App::Controllers::ProfileManagerController	RFO6
	RFO10
	RFO10.1
	RFO10.1.1
	RFO10.1.1.1
	RFO10.1.1.2
	RFO10.1.1.3
	RFO10.1.1.4
	RFO10.1.1.5
	RFO10.1.1.6
	RFO10.1.2
	RFO10.1.2.1
	RFO10.1.2.2
	RFO10.1.2.3
	RFO10.1.2.4
	RFO10.1.2.5
	RFD10.1.2.6
	RFO10.1.2.7
	RFO10.1.2.8
	RFD10.1.2.8
	RFO10.2
	RFO10.2.1
	RFO10.2.2

	RFO10.2.2.1
	RFO10.2.2.2
	RFO10.3.3
	RFO10.3.3.1
	RFO10.3.3.2
	RFO10.3.3.3
	RFO10.3.3.4
	RFO11
APIM::FrontEnd::App::Controllers::ResetPasswordController	RFD3
	RFD3.1
	RFD3.2
	RFD3.3
	RFD3.4
APIM::FrontEnd::App::Controllers::RegisterAPIController	RFO9
	RFO9.1
	RFO9.2
	RFO9.3
	RFO9.4
	RFO9.5
	RFD9.6
	RFD9.7
	RFO9.8
	RFO9.8.1
	RFO9.8.2
	RFO9.8.3
	RFO9.9
	RFD9.10
	RFD9.11
	RFO9.12
	RFD9.13
APIM::FrontEnd::App::Controllers::PolicyCallController	RFD5.7.1
	RFO5.8

	RFO5.12
APIM::FrontEnd::App::Controllers::PolicyTimeController	RFD5.7.1 RFO5.8 RFO5.12
APIM::FrontEnd::App::Controllers::PolicyTrafficController	RFD5.7.1 RFO5.8 RFO5.12
APIM::FrontEnd::App::Controllers::APIRegisteredController	RFO8 RFO8.1 RFO8.2 RFO8.2.1 RFO8.2.2 RFO8.2.3 RFO8.2.4 RFO8.2.4.1 RFO8.2.4.2 RFD8.2.4.3 RFO8.2.4.4 RFO8.2.4.5 RFO8.2.4.6 RFO8.2.4.7 RFO8.2.4.8 RFO8.2.4.9 RFO8.2.4.10 RFO8.2.4.11 RFO8.2.5 RFO8.2.6 RFO8.2.7 RFO8.2.8 RFO8.2.9

	RFO8.2.9.1
	RFO8.2.9.2
	RFO9
	RFO9.1
	RFO9.2
	RFD9.3
	RFO9.4
	RFO9.5
	RFD9.6
	RFD9.7
	RFO9.8
	RFO9.8.1
	RFO9.8.2
	RFO9.8.3
	RFO9.9
	RFD9.10
	RFD9.11
	RFO9.12
	RFD9.13
APIM::FrontEnd::App::Controllers::APIPurchasedController	RFO6
	RFO6.1
	RFO6.2
	RFO6.2.1
	RFO6.2.2
	RFO6.2.3
	RFO6.2.4
	RFO6.2.5
	RFD6.2.6
	RFD6.2.7
	RFO7.5.2
	RFO7.5.3
APIM::FrontEnd::App::Controllers::APIListController	RFO4.3.1

	RFO4.3.2
	RFO4.3.3
	RFO4.3.4
	RFO4.3.5
	RFO5
	RFO5.1
	RFO5.2
	RFO5.3
	RFO5.4
	RFO5.5
	RFO5.5.1
	RFO5.5.2
	RFO5.6
	RFO5.6.1
	RFO5.6.2
	RFO5.7
	RFD5.7.1
	RFO5.7.2
	RFO5.8
	RFO5.9
	RFO5.10
	RFO5.11
	RFO5.12
	RFO5.13
	RFO7
APIM::FrontEnd::App::Controllers::TransactionsListController	RF010.3
	RF010.3.1
	RF010.3.1.1
	RF010.3.1.2
	RF010.3.2
	RF010.3.2.1
	RF010.3.2.2

	RF010.3.2.3
	RF010.3.2.4
	RF010.3.2.5
APIM::FrontEnd::App::Controllers::AdminManagerController	RFO12
	RFO12.1
	RFO12.1.1
	RFO12.1.1.1
	RFO12.1.1.1.1
	RFO12.1.1.1.2
	RFD12.1.1.1.3
	RFD12.1.1.1.4
	RFO12.1.1.1.5
	RFO12.1.1.1.5.1
	RFF12.1.1.1.5.2
	RFD12.1.1.1.6
	RFO12.1.1.2
	RFO12.1.1.2.1
	RFO12.1.1.2.2
	RFD12.1.1.3
	RFO12.1.1.3.1
	RFD12.1.1.3.2
	RFO12.1.1.3.3
APIM::FrontEnd::App::Controllers::AdminUserManagerController	RFO12.2
	RFO12.2.1
	RFO12.2.1.1
	RFO12.2.1.1.1
	RFO12.2.1.1.2
	RFO12.2.1.2
	RFO12.2.1.2.1
	RFO12.2.1.2.2
	RFO12.2.1.3
	RFO12.2.1.4

	RFO12.2.1.5
	RFO12.2.1.5.1
	RFO12.2.1.5.2
APIM::FrontEnd::App::Controllers::AdminAPIManagerController	RFO12.2
	RFO12.2.1
	RFO12.2.1.1
	RFO12.2.1.1.1
	RFO12.2.1.1.2
	RFO12.2.1.2
	RFO12.2.1.2.1
	RFO12.2.1.2.2
	RFO12.2.1.3
	RFO12.2.1.4
	RFO12.2.1.5
	RFO12.2.1.5.1
	RFO12.2.1.5.2
APIM::BackEnd::Services::UsersDB	RFO1
	RFO1.1
	RFO1.2
	RFO1.3
	RFO1.4
	RFO1.5
	RFO1.6
	RFO1.7
	RFO1.8
	RFO1.9
	RFO1.10
	RFO6
	RFO10
	RFO10.1
	RFO10.1.1
	RFO10.1.1.1

	RFO10.1.1.2
	RFO10.1.1.3
	RFO10.1.1.4
	RFO10.1.1.5
	RFO10.1.1.6
	RFO10.1.2
	RFO10.1.2.1
	RFO10.1.2.2
	RFO10.1.2.3
	RFO10.1.2.4
	RFO10.1.2.5
	RFD10.1.2.6
	RFO10.1.2.7
	RFO10.1.2.8
	RFD10.1.2.8
	RFO10.2
	RFO10.2.1
	RFO10.2.2
	RFO10.2.2.1
	RFO10.2.2.2
	RFO10.3.3
	RFO10.3.3.1
	RFO10.3.3.2
	RFO10.3.3.3
	RFO10.3.3.4
	RFO11
APIM::BackEnd::Services::MicroservicesDB	RFO4.3.1
	RFO4.3.2
	RFO4.3.3
	RFO4.3.4
	RFO4.3.5
	RFO5

RFO5.1
RFO5.2
RFO5.3
RFO5.4
RFO5.5
RFO5.5.1
RFO5.5.2
RFO5.6
RFO5.6.1
RFO5.6.2
RFO5.7
RFD5.7.1
RFO5.7.2
RFO5.8
RFO5.9
RFO5.10
RFO5.11
RFO5.12
RFO5.13
RFO6
RFO6.1
RFO6.2
RFO6.2.1
RFO6.2.2
RFO6.2.3
RFO6.2.4
RFO6.2.5
RFD6.2.6
RFD6.2.7
RFO7
RFO7.1
RFO7.1.1

	RFO7.1.2
	RFO7.1.3
	RFO7.2
	RFO7.3
	RFO7.4
	RFO7.5
	RFD7.5.1
	RFO7.5.2
	RFO7.5.3
	RFO7.6
	RFO8
	RFO8.1
	RFO8.2
	RFO8.2.1
	RFO8.2.2
	RFO8.2.3
	RFO8.2.4
	RFO8.2.4.1
	RFO8.2.4.2
	RFD8.2.4.3
	RFO8.2.4.4
	RFO8.2.4.5
	RFO8.2.4.6
	RFO8.2.4.7
	RFO8.2.4.8
	RFO8.2.4.9
	RFO8.2.4.10
	RFO8.2.4.11
	RFO8.2.5
	RFO8.2.6
	RFO8.2.7
	RFO8.2.8

	RFO8.2.9
	RFO8.2.9.1
	RFO8.2.9.2
	RFO9
	RFO9.1
	RFO9.2
	RFD9.3
	RFO9.4
	RFO9.5
	RFD9.6
	RFD9.7
	RFO9.8
	RFO9.8.1
	RFO9.8.2
	RFO9.8.3
	RFO9.9
	RFD9.10
	RFD9.11
	RFO9.12
	RFD9.13
APIM::BackEnd::Services::TransactionsDB	RFO10.2
	RFO10.2.1
	RFO10.2.2
	RFO10.2.2.1
	RFO10.2.2.2
	RFO10.3.3
	RFO10.3.3.1
	RFO10.3.3.2
	RFO10.3.3.3
	RFO10.3.3.4
	RF010.3
	RF010.3.1

	RF010.3.1.1
	RF010.3.1.2
	RF010.3.2
	RF010.3.2.1
	RF010.3.2.2
	RF010.3.2.3
	RF010.3.2.4
	RF010.3.2.5
APIM::BackEnd::Services::SLADB	RFO5.7
	RFD5.7.1
	RFO5.7.2
APIM::BackEnd::Services::FileHandlerDB	RFD4.3.4
	RFO5.5.2
	RFO5.6
	RFO5.6.1
	RFD6.2.3
	RFD6.2.7
	RFO8.2.4.5
	RFO8.2.4.7
	RFD9.10
	RFD9.11
	RFD10.1.1.5
	RFD10.1.2.6

Tabella 1: Tracciamento Classi-Requisiti

7.2 Tracciamento Requisiti - Classi

Codice Requisito	Nome Classe
RF01	APIM::FrontEnd::App::Index APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::RegisterUserController APIM::BackEnd::Services::UsersDB
RFO1.1	APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::RegisterUserController APIM::BackEnd::Services::UsersDB
RFO1.2	APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::RegisterUserController APIM::BackEnd::Services::UsersDB
RFO1.3	APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::RegisterUserController APIM::BackEnd::Services::UsersDB
RFO1.3	APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::RegisterUserController APIM::BackEnd::Services::UsersDB
RFO1.4	APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::RegisterUserController APIM::BackEnd::Services::UsersDB
RFO1.5	APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::RegisterUserController APIM::BackEnd::Services::UsersDB

RFO1.6	APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::BackEnd::Services::UsersDB
RFO1.7	APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::RegisterUserController APIM::BackEnd::Services::UsersDB
RFO1.8	APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::RegisterUserController APIM::BackEnd::Services::UsersDB
RFO1.9	APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::RegisterUserController APIM::BackEnd::Services::UsersDB
RFO1.10	APIM::FrontEnd::App::Views::RegisterUser APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::RegisterUserController APIM::BackEnd::Services::UsersDB
RFO2	APIM::FrontEnd::App::Index APIM::FrontEnd::App::Views::Login APIM::FrontEnd::App::Controllers::LoginController
RFO2.1	APIM::FrontEnd::App::Views::Login APIM::FrontEnd::App::Controllers::LoginController
RFO2.1.1	APIM::FrontEnd::App::Views::Login APIM::FrontEnd::App::Controllers::LoginController
RFO2.1.2	APIM::FrontEnd::App::Views::Login APIM::FrontEnd::App::Controllers::LoginController
RFO2.1.3	APIM::FrontEnd::App::Views::Login APIM::FrontEnd::App::Controllers::LoginController
RFO2.1.4	APIM::FrontEnd::App::Views::Login

	APIM::FrontEnd::App::Controllers::LoginController
RFD3	APIM::FrontEnd::App::Views::PasswordRecovery
	APIM::FrontEnd::App::Views::ResetPassword
	APIM::FrontEnd::App::Controllers::PasswordRecoveryController
	APIM::FrontEnd::App::Controllers::ResetPasswordController
RFD3.1	APIM::FrontEnd::App::Views::PasswordRecovery
	APIM::FrontEnd::App::Views::ResetPassword
	APIM::FrontEnd::App::Controllers::PasswordRecoveryController
	APIM::FrontEnd::App::Controllers::ResetPasswordController
RFD3.2	APIM::FrontEnd::App::Views::PasswordRecovery
	APIM::FrontEnd::App::Views::ResetPassword
	APIM::FrontEnd::App::Controllers::PasswordRecoveryController
	APIM::FrontEnd::App::Controllers::ResetPasswordController
RFD3.3	APIM::FrontEnd::App::Views::PasswordRecovery
	APIM::FrontEnd::App::Views::ResetPassword
	APIM::FrontEnd::App::Controllers::PasswordRecoveryController
	APIM::FrontEnd::App::Controllers::ResetPasswordController
RFD3.4	APIM::FrontEnd::App::Views::PasswordRecovery
	APIM::FrontEnd::App::Views::ResetPassword
	APIM::FrontEnd::App::Controllers::PasswordRecoveryController
	APIM::FrontEnd::App::Controllers::ResetPasswordController
RFO4	APIM::FrontEnd::App::Index
RFO4.1	APIM::FrontEnd::App::Index
RFO4.2	APIM::FrontEnd::App::Index
RFO4.3	APIM::FrontEnd::App::Index
RFO4.3.1	APIM::FrontEnd::App::Index
	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIController
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB

RFO4.3.2	APIM::FrontEnd::App::Index
	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIController
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO4.3.3	APIM::FrontEnd::App::Index
	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIController
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO4.3.4	APIM::FrontEnd::App::Index
	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIController
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO4.3.5	APIM::FrontEnd::App::Index
	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIController
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO5	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
RFO5.1	APIM::FrontEnd::App::Views::API

	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO5.2	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO5.3	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO5.4	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO5.5	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO5.5.1	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
R	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel

	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
	APIM::BackEnd::Services::FileHandlerDB
RFO5.5.6	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
	APIM::BackEnd::Services::FileHandlerDB
RFO5.6.1	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
	APIM::BackEnd::Services::FileHandlerDB
RFO5.6.2	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO5.7	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Models::SLAMicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
	APIM::BackEnd::Services::SLADB
RFD5.7.1	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::PolicyCall
	APIM::FrontEnd::App::Views::PolicyTime
	APIM::FrontEnd::App::Views::PolicyTraffic
	APIM::FrontEnd::App::Views::APIList

	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Models::SLAMicroserviceModel
	APIM::FrontEnd::App::Controllers::PolicyCallController
	APIM::FrontEnd::App::Controllers::PolicyTimeController
	APIM::FrontEnd::App::Controllers::PolicyTrafficController
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
	APIM::BackEnd::Services::SLADB
RFO5.7.2	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Models::SLAMicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
	APIM::BackEnd::Services::SLADB
RFO5.8	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::PolicyCall
	APIM::FrontEnd::App::Views::PolicyTime
	APIM::FrontEnd::App::Views::PolicyTraffic
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::PolicyCallController
	APIM::FrontEnd::App::Controllers::PolicyTimeController
	APIM::FrontEnd::App::Controllers::PolicyTrafficController
	APIM::BackEnd::Services::MicroservicesDB
RFO5.9	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
RFO5.10	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel

	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO5.11	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::BackEnd::Services::MicroservicesDB
RFO5.12	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::PolicyCall
	APIM::FrontEnd::App::Views::PolicyTime
	APIM::FrontEnd::App::Views::PolicyTraffic
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::PolicyCallController
	APIM::FrontEnd::App::Controllers::PolicyTimeController
	APIM::FrontEnd::App::Controllers::PolicyTrafficController
	APIM::FrontEnd::App::Controllers::APIListController
RFO5.13	APIM::FrontEnd::App::Views::API
	APIM::FrontEnd::App::Views::APIList
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIListController
	APIM::BackEnd::Services::MicroservicesDB
RFO6	APIM::FrontEnd::App::Views::ProfileManager
	APIM::FrontEnd::App::Views::APIPurchased
	APIM::FrontEnd::App::Models::UserDetailsModel
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::ProfileManagerController
	APIM::FrontEnd::App::Controllers::APIPurchasedController
	APIM::BackEnd::Services::UsersDB
	APIM::BackEnd::Services::MicroservicesDB
RFO6.1	APIM::FrontEnd::App::Views::APIPurchased
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIPurchasedController

	APIM::BackEnd::Services::MicroservicesDB
	APIM::FrontEnd::App::Views::APIPurchased
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIPurchasedController
	APIM::BackEnd::Services::MicroservicesDB
	APIM::FrontEnd::App::Views::APIPurchased
	APIM::FrontEnd::App::Controllers::APIPurchasedController
RFO6.2	APIM::FrontEnd::App::Views::APIPurchased
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIPurchasedController
	APIM::BackEnd::Services::MicroservicesDB
	APIM::FrontEnd::App::Views::APIPurchased
RFO6.2.1	APIM::FrontEnd::App::Views::APIPurchased
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIPurchasedController
	APIM::BackEnd::Services::MicroservicesDB
RFO6.2.2	APIM::FrontEnd::App::Views::APIPurchased
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIPurchasedController
	APIM::BackEnd::Services::MicroservicesDB
RFO6.2.3	APIM::FrontEnd::App::Views::APIPurchased
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIPurchasedController
	APIM::BackEnd::Services::MicroservicesDB
RFO6.2.4	APIM::FrontEnd::App::Views::APIPurchased
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIPurchasedController
	APIM::BackEnd::Services::MicroservicesDB
RFO6.2.5	APIM::FrontEnd::App::Views::APIPurchased
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIPurchasedController
	APIM::BackEnd::Services::MicroservicesDB

RFD6.2.6	APIM::FrontEnd::App::Views::APIPurchased APIM::FrontEnd::App::Models::MicroserviceModel APIM::FrontEnd::App::Controllers::APIPurchasedController APIM::BackEnd::Services::MicroservicesDB
RFD6.2.7	APIM::FrontEnd::App::Views::APIPurchased APIM::FrontEnd::App::Models::MicroserviceModel APIM::FrontEnd::App::Controllers::APIPurchasedController APIM::BackEnd::Services::MicroservicesDB APIM::BackEnd::Services::FileHandlerDB
RFO7	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Views::APIList APIM::FrontEnd::App::Models::MicroserviceModel APIM::FrontEnd::App::Controllers::APIListController APIM::BackEnd::Services::MicroservicesDB
RFO7.1	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Models::MicroserviceModel APIM::BackEnd::Services::MicroservicesDB
RFO7.1.1	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Models::MicroserviceModel APIM::BackEnd::Services::MicroservicesDB
RFO7.1.2	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Models::MicroserviceModel APIM::BackEnd::Services::MicroservicesDB
RFO7.1.3	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Models::MicroserviceModel APIM::BackEnd::Services::MicroservicesDB
RFO7.2	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Models::MicroserviceModel APIM::BackEnd::Services::MicroservicesDB
RFO7.3	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Models::MicroserviceModel

	APIM::BackEnd::Services::MicroservicesDB
RFO7.4	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Models::MicroserviceModel APIM::BackEnd::Services::MicroservicesDB
RFO7.5	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Models::MicroserviceModel APIM::BackEnd::Services::MicroservicesDB
RF7.5.1	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Models::MicroserviceModel APIM::BackEnd::Services::MicroservicesDB
RFO7.5.2	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Views::APIPurchased APIM::FrontEnd::App::Models::MicroserviceModel APIM::FrontEnd::App::Controllers::APIPurchasedController APIM::BackEnd::Services::MicroservicesDB
RFO7.5.3	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Views::APIPurchased APIM::FrontEnd::App::Models::MicroserviceModel APIM::FrontEnd::App::Controllers::APIPurchasedController APIM::BackEnd::Services::MicroservicesDB
RFO7.6	APIM::FrontEnd::App::Views::API APIM::FrontEnd::App::Models::MicroserviceModel APIM::BackEnd::Services::MicroservicesDB
RFO8	APIM::FrontEnd::App::Views::APIRegistered APIM::FrontEnd::App::Models::MicroserviceModel APIM::FrontEnd::App::Controllers::APIRegisteredController APIM::BackEnd::Services::MicroservicesDB
RFO8.1	APIM::FrontEnd::App::Views::APIRegistered APIM::FrontEnd::App::Models::MicroserviceModel APIM::FrontEnd::App::Controllers::APIRegisteredController APIM::BackEnd::Services::MicroservicesDB
RFO8.2	APIM::FrontEnd::App::Views::APIRegistered

	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.1	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.2	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.3	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.4	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.4.1	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.4.2	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFD8.2.4.3	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.4.4	APIM::FrontEnd::App::Views::APIRegistered

	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.4.5	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
	APIM::BackEnd::Services::FileHandlerDB
RFO8.2.4.6	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.4.7	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.4.8	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.4.9	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.4.10	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.4.11	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController

	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.5	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO8.2.6	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
FO8.2.7	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
FO8.2.8	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
FO8.2.9	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
FO8.2.9.1	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
FO8.2.9.2	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO9	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel

	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO9.1	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO9.2	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
RFO9.3	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Controllers::RegisterAPIController
RFO9.4	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::BackEnd::Services::MicroservicesDB
RFO9.5	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFD9.6	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB

RFD9.7	APIM::FrontEnd::App::Views::ProfileManager
	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::BackEnd::Services::MicroservicesDB
RFO9.8	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO9.8.1	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO9.8.2	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO9.8.3	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO9.9	APIM::FrontEnd::App::Views::RegisterAPI

	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFD9.10	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::BackEnd::Services::MicroservicesDB
	APIM::BackEnd::Services::FileHandlerDB
RFD9.11	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::BackEnd::Services::MicroservicesDB
RFO9.12	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFD9.13	APIM::FrontEnd::App::Views::RegisterAPI
	APIM::FrontEnd::App::Views::APIRegistered
	APIM::FrontEnd::App::Models::MicroserviceModel
	APIM::FrontEnd::App::Controllers::RegisterAPIController
	APIM::FrontEnd::App::Controllers::APIRegisteredController
	APIM::BackEnd::Services::MicroservicesDB
RFO10	APIM::FrontEnd::App::Views::ProfileManager
	APIM::FrontEnd::App::Models::UserDetailsModel
	APIM::FrontEnd::App::Controllers::ProfileManagerController
	APIM::BackEnd::Services::UsersDB

RFO10.1	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB
RFO10.1.1	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB
RFO10.1.2	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB
RFO10.1.2.1	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsMode APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB
RFO10.1.2.2	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsMode APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB
RFO10.1.2.3	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsMode APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB
RFO10.1.2.4	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsMode APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB
RFO10.1.2.5	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsMode APIM::FrontEnd::App::Controllers::ProfileManagerController

	APIM::BackEnd::Services::UsersDB
RFO10.1.2.6	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsMode APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB
RFO10.1.2.7	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsMode APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB
RFO10.1.2.8	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsMode APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB
RFO10.2	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Models::TransactionModel APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB
RFO10.2.1	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Models::TransactionModel APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB APIM::BackEnd::Services::TransactionsDB
RFO10.2.2	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsModel APIM::FrontEnd::App::Models::TransactionModel APIM::FrontEnd::App::Controllers::ProfileManagerController APIM::BackEnd::Services::UsersDB APIM::BackEnd::Services::TransactionsDB
RFO10.2.2.1	APIM::FrontEnd::App::Views::ProfileManager APIM::FrontEnd::App::Models::UserDetailsModel

	APIM::FrontEnd::App::Models::TransactionModel
	APIM::FrontEnd::App::Controllers::ProfileManagerController
	APIM::BackEnd::Services::UsersDB
	APIM::BackEnd::Services::TransactionsDB
RFO10.2.2.2	APIM::FrontEnd::App::Views::ProfileManager
	APIM::FrontEnd::App::Models::UserDetailsModel
	APIM::FrontEnd::App::Models::TransactionModel
	APIM::FrontEnd::App::Controllers::ProfileManagerController
	APIM::BackEnd::Services::UsersDB
	APIM::BackEnd::Services::TransactionsDB
RFO10.3.3	APIM::FrontEnd::App::Views::ProfileManager
	APIM::FrontEnd::App::Models::UserDetailsModel
	APIM::FrontEnd::App::Models::TransactionModel
	APIM::FrontEnd::App::Controllers::ProfileManagerController
	APIM::BackEnd::Services::UsersDB
	APIM::BackEnd::Services::TransactionsDB
RFO10.3.3.1	APIM::FrontEnd::App::Views::ProfileManager
	APIM::FrontEnd::App::Models::UserDetailsModel
	APIM::FrontEnd::App::Models::TransactionModel
	APIM::FrontEnd::App::Controllers::ProfileManagerController
	APIM::BackEnd::Services::UsersDB
	APIM::BackEnd::Services::TransactionsDB
RFO10.3.3.2	APIM::FrontEnd::App::Views::ProfileManager
	APIM::FrontEnd::App::Models::UserDetailsModel
	APIM::FrontEnd::App::Models::TransactionModel
	APIM::FrontEnd::App::Controllers::ProfileManagerController
	APIM::BackEnd::Services::UsersDB
	APIM::BackEnd::Services::TransactionsDB
RFO10.3.3.3	APIM::FrontEnd::App::Views::ProfileManager
	APIM::FrontEnd::App::Models::UserDetailsModel
	APIM::FrontEnd::App::Models::TransactionModel
	APIM::FrontEnd::App::Controllers::ProfileManagerController

	APIM::BackEnd::Services::UsersDB
	APIM::BackEnd::Services::TransactionsDB
RFO10.3.3.4	APIM::FrontEnd::App::Views::ProfileManager
	APIM::FrontEnd::App::Models::UserDetailsModel
	APIM::FrontEnd::App::Models::TransactionModel
	APIM::FrontEnd::App::Controllers::ProfileManagerController
	APIM::BackEnd::Services::UsersDB
	APIM::BackEnd::Services::TransactionsDB
RFO11	APIM::FrontEnd::App::Index
	APIM::FrontEnd::App::Views::Logout
	APIM::FrontEnd::App::Models::UserDetailsModel
	APIM::FrontEnd::App::Controllers::LogoutController
RFO12	APIM::FrontEnd::App::Views::AdminManager
	APIM::FrontEnd::App::Controllers::AdminManagerController
RFO12.1	APIM::FrontEnd::App::Views::AdminAPIManager
	APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1	APIM::FrontEnd::App::Views::AdminAPIManager
	APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.1	APIM::FrontEnd::App::Views::AdminAPIManager
	APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.1.1	APIM::FrontEnd::App::Views::AdminAPIManager
	APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.1.2	APIM::FrontEnd::App::Views::AdminAPIManager
	APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.1.3	APIM::FrontEnd::App::Views::AdminAPIManager
	APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.1.4	APIM::FrontEnd::App::Views::AdminAPIManager
	APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.1.5	APIM::FrontEnd::App::Views::AdminAPIManager
	APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.1.5.1	APIM::FrontEnd::App::Views::AdminAPIManager
	APIM::FrontEnd::App::Controllers::AdminAPIManagerController

RFO12.1.1.1.5.2	APIM::FrontEnd::App::Views::AdminAPIManager APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFD12.1.1.1.6	APIM::FrontEnd::App::Views::AdminAPIManager APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.2	APIM::FrontEnd::App::Views::AdminAPIManager APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.2.1	APIM::FrontEnd::App::Views::AdminAPIManager APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.2.2	APIM::FrontEnd::App::Views::AdminAPIManager APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.3	APIM::FrontEnd::App::Views::AdminAPIManager APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.3.1	APIM::FrontEnd::App::Views::AdminAPIManager APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.3.2	APIM::FrontEnd::App::Views::AdminAPIManager APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.1.1.3.3	APIM::FrontEnd::App::Views::AdminAPIManager APIM::FrontEnd::App::Controllers::AdminAPIManagerController
RFO12.2	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1.1	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1.1.1	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1.1.2	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1.2	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1.2.1	APIM::FrontEnd::App::Views::AdminUserManager

	APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1.2.2	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1.3	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1.4	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1.5	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1.5.1	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController
RFO12.2.1.5.2	APIM::FrontEnd::App::Views::AdminUserManager APIM::FrontEnd::App::Controllers::AdminUserManagerController

Tabella 2: Tracciamento Requisiti-Classi