
UNIVERSITÀ
DEGLI STUDI
DI BERGAMO

Scuola di Ingegneria

Corso di Laurea Magistrale in Ingegneria Informatica

Classe LM-32 – Classe delle lauree magistrali in ingegneria informatica

# Estensione del modulo della piattaforma ENEA PELL per la valutazione economico-finanziaria degli impianti di illuminazione pubblica

Relatore

Prof.ssa Patrizia Scandurra

Correlatore

PhD Dr. Edoardo Scazzocchio – Nomos Consulting s.r.l.
PhD Dr. Fabio Moretti – ENEA Research Lab.

Studenti

Salvatore Greco matr. 1053509

Fabio Gamba matr. 1053157

A.A 2022/2023

# ABSTRACT

La piattaforma software PELL (Public Energy Living Lab) IP (Illuminazione Pubblica), proprietà di ENEA, ha come finalità il censimento e il monitoraggio dei dati relativi agli impianti di illuminazione pubblica su strada dei comuni presenti sul territorio italiano.

In particolare, il modulo applicativo SAVE (Supporto Alla Valutazione Economico finanziaria) della piattaforma PELL IP offre un servizio che consente la valutazione economico-finanziaria di nuove soluzioni illuminotecniche di riqualificazione rispetto alla situazione esistente.

L'obiettivo principale del progetto PELL (attivo da luglio 2019) IP è quello di poter rispondere alla necessità sempre più crescente di rendere le infrastrutture pubbliche più intelligenti e digitali, mettendo a disposizione delle città e delle municipalità uno strumento accessibile e chiaro, che permetta di valutare vantaggi e svantaggi di ogni fase necessaria alla riqualificazione degli impianti in chiave smart city.

Il lavoro di questa tesi è stato svolto in collaborazione con ENEA e si concentra sullo sviluppo e adattamento del modulo software SAVE.

SAVE è un modulo software di back-end già integrato ed esposto come servizio del portale PELL (https://www.pell.enea.it), ma necessita di una estensione per rispondere alle nuove esigenze e requisiti funzionali individuati da ENEA a supporto dei comuni e gestori degli impianti di illuminazione pubblica.

Nello specifico, il nuovo modulo SAVE mantiene tutte le funzionalità della versione precedente, ovvero il calcolo aggregato per impianto di varie soluzioni di finanziamento in base alle informazioni di profilazione inserite all'interno del portale, l'applicazione dei parametri di investimento in input e l'elaborazione delle caratteristiche dell'investimento, includendo il concetto completamente nuovo delle Zone Omogenee: grazie ad esse ora è possibile suddividere l'impianto in zone identificabili da un insieme di caratteristiche.

In aggiunta, nella nuova versione del modulo SAVE, tutte queste funzionalità sono state arricchite con nuove possibilità di finanziamento (tramite ESCo e Istituti Finanziari), analisi di sensitività (grazie agli indici di redditività e sostenibilità economica) e un report che aggrega tutti questi risultati in modo compatto e ordinato per facilitarne la fruizione.

In parallelo, è stata estesa la parte front-end del portale PELL, che si slega dai concetti tecnico-economico elaborati separatamente dal modulo SAVE, ma è funzionale alla presentazione dei dati dello strumento, offrendo all'utente finale del portale una serie di dati e andamenti che è possibile utilizzare per valutare la bontà dell'investimento ipotizzato e delle possibili variazioni in maniera dinamica.

# Indice

# 1 Introduzione

## 1.1. Descrizione del contesto e del problema affrontato

Nel corso degli anni i servizi offerti alla collettività hanno subito molte innovazioni, soprattutto recentemente anche grazie alla diffusione dell'Industria 4.0. Nato da progetto tedesco, l'Industria 4.0 punta a investire pesantemente in infrastrutture più moderne, sostenibili e connesse, incorporando il concetto dell'Internet of Things: le città e le loro infrastrutture diventano intelligenti e interconnesse, scambiando informazioni in tempo reale e permettendo di accedere a una enorme quantità di dati.

La gestione dell'illuminazione pubblica è un contesto estremamente importante a livello nazionale e anch'esso è fortemente influenzato da questi cambiamenti. I lampioni tradizionali sono pronti per essere superati e sostituiti da un modello che integra la ovvia funzionalità di diffusione della luce, con tutta una serie di nuovi servizi a favore delle città e dei cittadini. La raccolta e l'elaborazione delle informazioni prodotte da un sistema interconnesso garantiscono una gestione efficiente del consumo di energia poiché consentono una regolazione continua, ottimale e in tempo reale dei sistemi di illuminazione, in base al fabbisogno energetico.

ENEA partecipa a questa evoluzione mettendo a disposizione a Comuni e gestori il portale PELL (Public Energy Living Lab), che contribuisce attivamente alla digitalizzazione e ammodernamento dell'impianto pubblico.

Il PELL, Public Energy Living Lab, è una piattaforma informatica che attraverso la realizzazione di un censimento degli impianti avvia un processo di recupero, raccolta, organizzazione, gestione, elaborazione e valutazione dei dati tecnici e consumi degli impianti di pubblica illuminazione.
La sua struttura risponde ad una richiesta di digitalizzazione delle infrastrutture pubbliche che mira a trasformarle in reti intelligenti attraverso la digitalizzazione delle informazioni, il monitoraggio continuo, la elaborazione in tempo reale degli input relativi ai consumi e prestazioni, la redistribuzione aperte delle informazioni aggregate e quindi la creazione di un canale di collegamento diretto tra amministratori e amministrati in merito ad alcune tipologie d'informazioni. [0]

In particolare, il suo modulo SAVE (Supporto Alla Valutazione Economico finanziaria) consente gli utenti di effettuare valutazioni economiche nell'ipotesi di riqualificazione dell'impianto di illuminazione, senza avere conoscenze estese in campo economico. Infatti, il gestore di un impianto di illuminazione accedendo al portale PELL, può utilizzare il modulo SAVE per inserire tutti i dettagli riguardanti il proprio impianto di competenza e ottenere un report completo di analisi e valutazione.
Considerando un processo di rifacimento ed evoluzione della stessa piattaforma PELL, per stare al passo con le tecnologie, il modulo SAVE ha richiesto un aggiornamento, con l'introduzione di nuove e migliorate funzionalità, e un adattamento per facilitare l'integrare all'interno della nuova piattaforma.

## 1.2. Soluzione proposta e note metodologiche

La soluzione proposta si basa su vari concetti chiave:

- Calcolo aggregato: si parte calcolando i piccoli risultati che verranno poi utilizzati per calcoli sempre più complessi, fino al risultato finale
- Fruizione tramite web API: permette il recupero delle informazioni tramite l'esposizione pubblica di endpoint (anche chiamati link)
- Feature test: consente il testing di una funzionalità implementata per la verifica del funzionamento e l'assenza di errori durante l'operazione

Sono state utilizzate varie note metodologiche di stesura del codice, al fine di rendere più comprensibile possibile le operazioni effettuate

- Utilizzo del Camel case: consiste nella scrittura di nomi o frasi marcando la prima lettera tra uno spazio e l'altro con il maiuscolo, e successivamente eliminando gli spazi. Questo stile di scrittura è stato utilizzato per nomi dei metodi e delle funzioni, al fine di renderli più leggibili
- Utilizzo dei nomi di funzione in italiano: abbiamo scelto l'italiano per i nomi di funzione in modo da dare un'idea immediata dell'operazione svolta al lettore
- Utilizzo dei commenti in inglese: la lingua scelta per i commenti è in inglese, al fine di poter orientare anche chi non è confidente con la lingua italiana durante l'analisi delle funzioni
- Classe helper: sono un concetto di programmazione che consentono allo sviluppatore di semplificare l'utilizzo delle particolari funzionalità che esse contengono.

Per la costruzione di un iniziale database di test abbiamo adottato le seguenti note metodologiche sulle varie entità:

- Entità Plant (Impianto): rappresenta un impianto disponibile all'elaborazione per l'utente. Oltre ad essere identificato univocamente, contiene informazioni quali il CAP del comune di riferimento e un'etichetta descrittiva
- Entità Investment (Investimento): rappresenta il set di dati scelti per l'elaborazione dell'investimento complessivo sull'impianto scelto. Questa serie di dati utilizza i valori di default previsti dalla specifica, per i campi informativi relativi alla spartizione dell'ammontare dell'investimento (parte finanziata dalla banca, a carico del comune, a carico del soggetto terzo) si è deciso di attribuire la totalità dell'investimento al comune, lasciando azzerate le informazioni degli altri attori
- Entità Has (Zone omogenee): una zona omogenea rappresenta un'area all'interno dell'impianto dove i parametri di manutenzione dei dispositivi sono in comune, ovvero una determinata zona dove l'illuminazione è affidata alle stesse programmazioni di gestione per tutti i lampioni e pannelli associati.

Abbiamo scelto di utilizzare 2 zone da riqualificare (AS-IS) associate a uno a uno con 2 zone riqualificate (TO-BE), utilizzando parametri per le zone riqualificate chiaramente più vantaggiosi di quelle precedenti

- Entità Cluster (Raggruppamenti): entità all'interno della zona omogenea dove i parametri di illuminazione dei dispositivi sono in comune, ovvero una determinata zona dove l'illuminazione fa a capo a un unico pannello o contatore, con i relativi settaggi elettrici
- Entità Analysis (Analisi): entità che tiene traccia delle simulazioni dato un impianto e un investimento applicato, in modo da mantenere uno storico di simulazioni per il determinato utente

## 1.3. Organizzazione della tesi e riferimenti

Di seguito presentiamo i capitoli che comporranno questa tesi:

- La piattaforma informatica PELL IP: in questo capitolo verrà illustrato più approfonditamente la piattaforma proprietaria di ENEA, scendendo nei dettagli tecnici dell'infrastruttura e dettagliando i servizi che offre.
- Il Modulo applicativo SAVE (Supporto Alla Valutazione Economico-finanziaria): questa sezione definirà le specifiche funzionali ed i requisiti del modulo, così come design utilizzati per lo sviluppo. Verrà inoltre illustrato il nuovo modello dei dati su cui il modulo lavorerà.
- Sviluppo funzionalità della classe SaveHelper: questa sezione è dedicata ai dettagli implementativi dei requisiti del modulo SAVE
- Validazione e testing del modulo SAVE: qui verranno illustrate le metodologie di validazione utilizzati ed i risultati di testing sulle funzionalità
- Conclusione: verranno evidenziati gli stati di avanzamento, eventuali considerazioni finali e sviluppi futuri
- Appendice & Bibliografia: conterrà la matrice di tracciabilità dei requisiti e i riferimenti bibliografici utilizzati nella tesi

Tutti gli artefatti software e il materiale prodotto per la redazione di questa tesi è disponibile in una repository GitHub pubblicata al seguente indirizzo: https://github.com/fabiomor/save-module-2.0

# 2 La piattaforma informatica PELL IP

## 2.1. Architettura della piattaforma

L'architettura del portale PELL è descritta dalla *Figura 1*:

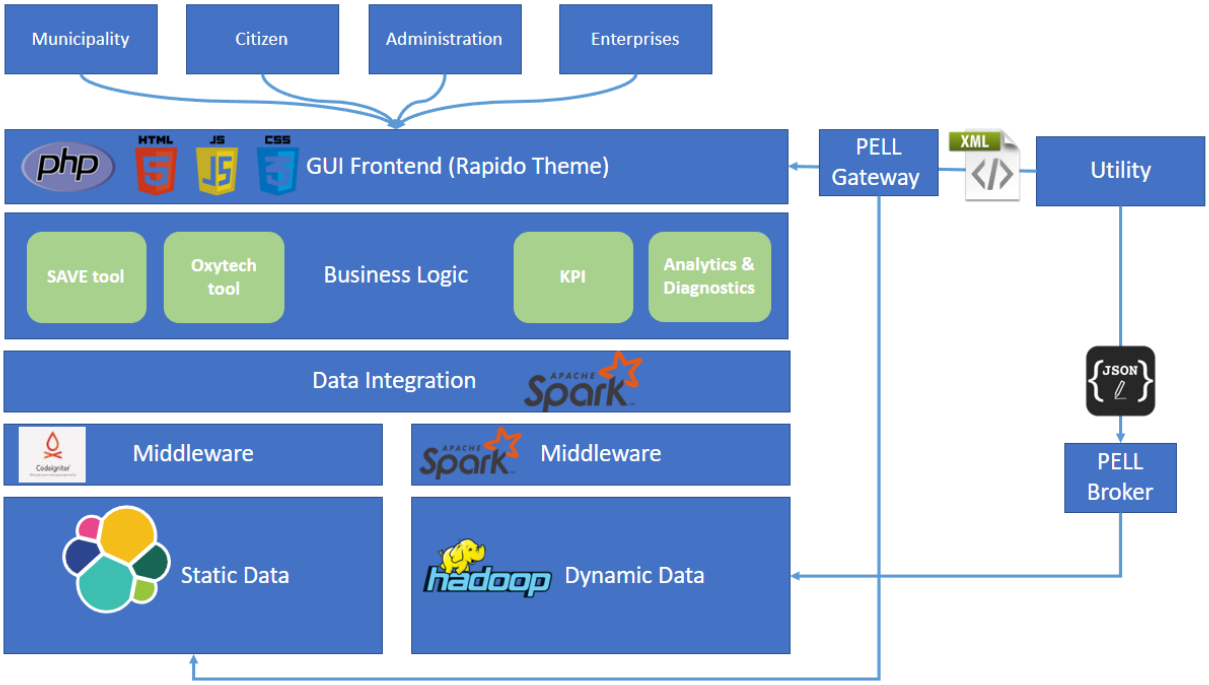


*Figura 1 Architettura portale PELL*

Questo schema racchiude la struttura di tutto il portale, mentre il focus della nostra tesi riguarda una porzione che comprende:

- GUI Frontend
- Business Logic: SAVE Tool
- Data Integration
- Middleware
- Static Data

La GUI (Graphical User Interface) rappresenta l'interfaccia grafica con la quale l'utente finale interagisce quando utilizza il servizio. Essa ha subito un completo rifacimento in modo da potersi adattare alle nostre modifiche effettuate al modulo SAVE. Attraverso questa nuova interfaccia è possibile accedere a tutti gli strumenti che il PELL mette a disposizione.

Il SAVE Tool contiene tutta la logica di elaborazione delle analisi in base ai dati degli impianti inseriti dall'utente e ai dati degli investimenti. Questo modulo, quindi, si occupa di prendere i dati, elaborarli e presentarli all'interfaccia grafica in modo da poter essere visualizzati.

Per quanto riguarda Data Integration, Middleware e Static Data, durante lo sviluppo abbiamo lavorato in un ambiente semplificato, che non rispecchia totalmente le tecnologie e strutture dell'intero portale. Abbiamo infatti utilizzato WampServer, che facilita lo sviluppo e il testing di applicazioni web, mettendo a disposizione tre componenti fondamentali:

- Server web Apache
- Sistema di gestione di database MySQL
- Linguaggio di programmazione PHP

## 2.2. Obiettivo del servizio

Lo scenario di riferimento per il modulo applicativo è relativo ad amministratori di enti locali che, in fase di pianificazione di interventi di riqualificazione illuminotecnica di un impianto di pubblica illuminazione, procedono ad un censimento dell'infrastruttura, alla quantificazione di costi e/o benefici e successivamente ad una analisi preliminare delle diverse modalità di finanziamento per proseguire con il suddetto intervento di riqualificazione, attraverso l'uso di linee guida qualitative che mirano a supportare gli amministratori locali nella scelta tra queste diverse modalità di finanziamento, a seconda delle condizioni in cui si trova l'ente di appartenenza (si rinvia al Report RdS/PAR2017/053 per una trattazione più esaustiva) [1].

Il PELL non si configura pertanto come uno smart street service in senso stretto; tuttavia, è stato incluso nell'analisi in quanto può rappresentare un importante fattore abilitante per la diffusione degli interventi di riqualificazione dell'infrastruttura di Pubblica Illuminazione in ottica smart street service.

La struttura del PELL risponde ad una logica di digitalizzazione delle infrastrutture pubbliche energivore che mira a trasformarle in reti intelligenti attraverso la digitalizzazione delle informazioni, il monitoraggio continuo, l'elaborazione in tempo reale degli input relativi ai consumi e prestazioni, la redistribuzione aperta delle informazioni aggregate e quindi la creazione di un canale di collegamento diretto tra amministratori e amministrati.

I benefici che caratterizzano l'intervento di riqualificazione in ottica smart street service denominato "Public Energy Living Lab" sono indiretti, ossia legati alle opportunità di riqualificazione dell'infrastruttura di Pubblica Illuminazione che emergono dall'utilizzo del PELL.

## 2.3. Tecnologie utilizzate

- PHP8.1/Laravel: la soluzione proposta è sviluppata utilizzando il linguaggio di programmazione PHP 8.1. In particolare, viene utilizzato il framework Laravel 8 scritto sulla versione di PHP specificata.
  Il framework in questione aiuta lo sviluppo di applicazioni web eliminando alcune operazioni comuni nello sviluppo di questa classe di applicazioni. [2]
- MySQL/PHPMyAdmin: database utilizzato per gestire i dati anagrafici dei comuni, degli impianti, e i dati inseriti dai gestori.
- diagrams.net: utilizzato per la modellazione UML degli Use Case e per la successiva estrapolazione dei requisiti funzionali
- Postman: software di sviluppo e testing di API web, utilizzato per la validazione dei risultati ottenuti dal modulo SAVE
- GitHub: nota piattaforma online per il versionamento del codice e la collaborazione tra gli addetti ai lavori

# 3 Il Modulo applicativo SAVE (Supporto Alla Valutazione Economico-finanziaria)

## 3.1. Specifica dei casi d'uso del modulo

### Use Case 1: Scelta funzionalità

**Descrizione:** l'utente all'interno del modulo SAVE sceglie una funzionalità.

**Attori:** utente PELL.

**Flusso:** il flusso di esecuzione è descritto dal seguente diagramma di attività (*Figura 2*):

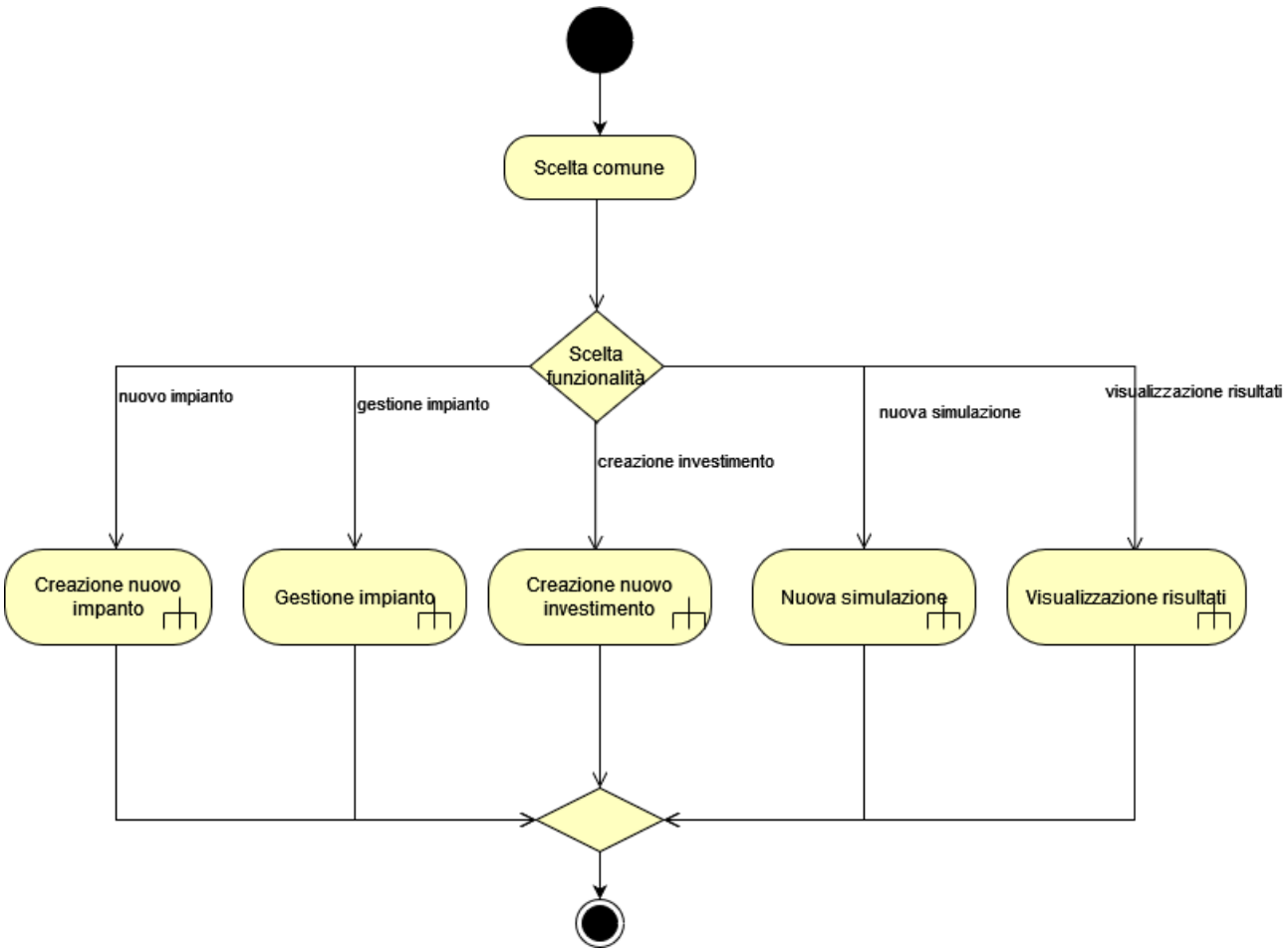


*Figura 2 UC.1 Scelta funzionalità*

**Dettagli:** l'attore attiva il caso d'uso una volta eseguito l'accesso al portale PELL e recandosi nella sezione del modulo SAVE. Può quindi decidere quale funzionalità usare, attraverso l'interfaccia.

**Precondizioni:** l'utente si è autenticato nel portale PELL ed ha selezionato il modulo SAVE.

**Post condizioni:** la funzionalità scelta viene presentata all'utente.

## Use Case 2: Gestione impianto

**Descrizione:** l'utente attiva questo use case per creare le zone omogenee dell'impianto selezionato. Oltre alla creazione può modificare zono omogenee già presenti, cancellarle o copiarle.

**Attori:** utente PELL.

**Flusso:** il flusso di esecuzione è descritto dal seguente diagramma di attività (*Figura 3*):

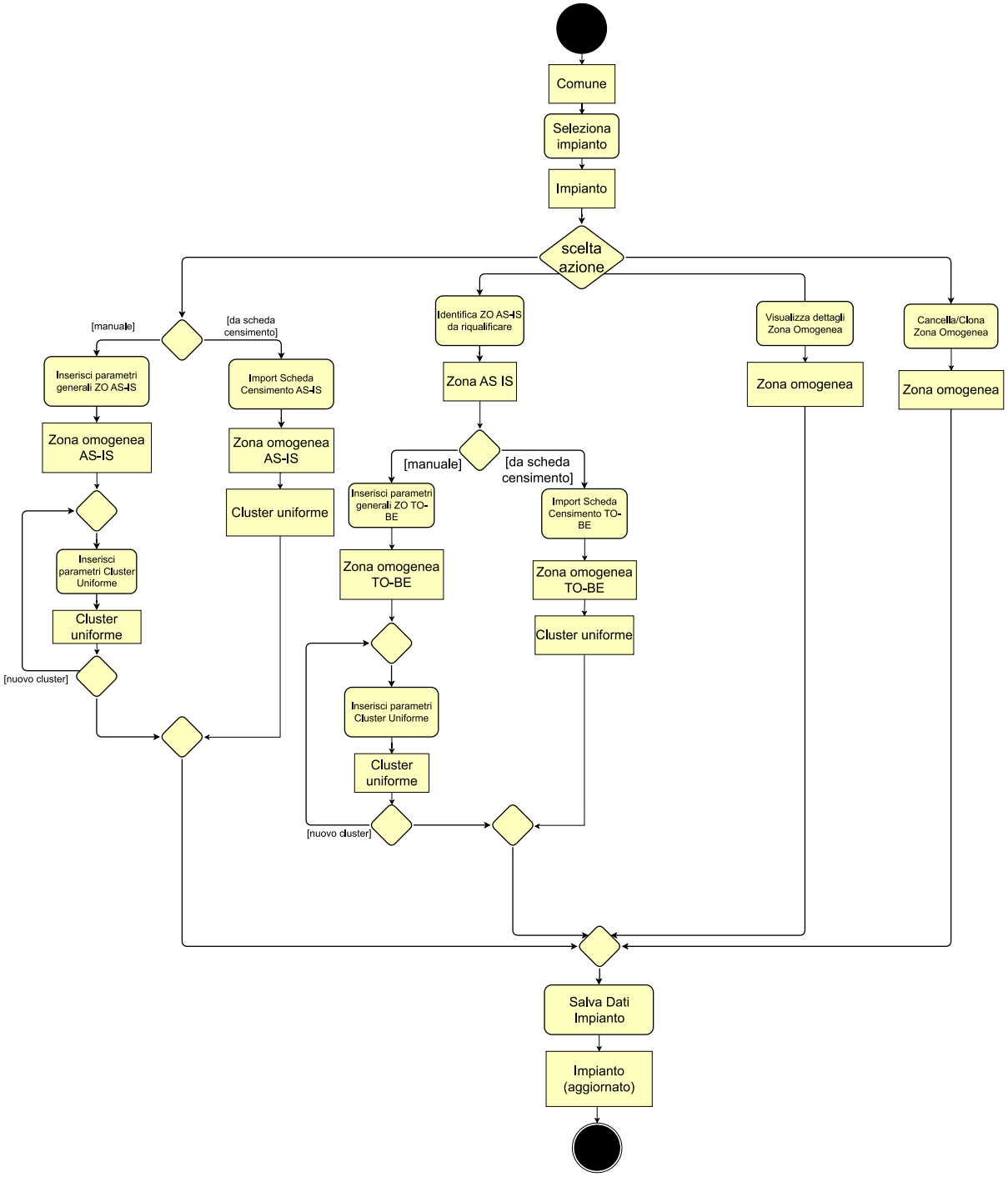


*Figura 3 UC.2 Gestione impianto*

**Dettagli:** l'utente attivando questo caso d'uso, deve inserire in appositi form i dati illuminotecnici caratteristici di ogni zona omogenea che compone l'impianto scelto. Successivamente per ogni zona omogenea registrata, bisogna inserire i dati dei cluster uniformi che la compongono.

L'inserimento di questi dati può avvenire manualmente oppure con l'ausilio di schede censimento. Nel caso di inserimento manuale, il form mostra tutti i campi necessari e si occupa della gestione dei vincoli dei dati. Mediante l'utilizzo di schede censimento, invece, l'utente deve semplicemente selezionare il file locale che verrà utilizzato per estrarre tutti i dati necessari.

**Precondizioni:** l'utente si è autenticato nel portale PELL ed ha selezionato il modulo SAVE.

**Post condizioni:** i dati vengono salvati nel database e sono liberamente consultabili dall'utente che li ha creati.

## Use Case 3: Creazione nuovo investimento

**Descrizione:** l'utente attiva questo use case per inserire i dati economici di un possibile investimento.

**Attori:** utente PELL.

**Flusso:** il flusso di esecuzione è descritto dal seguente diagramma di attività (*Figura 4*):

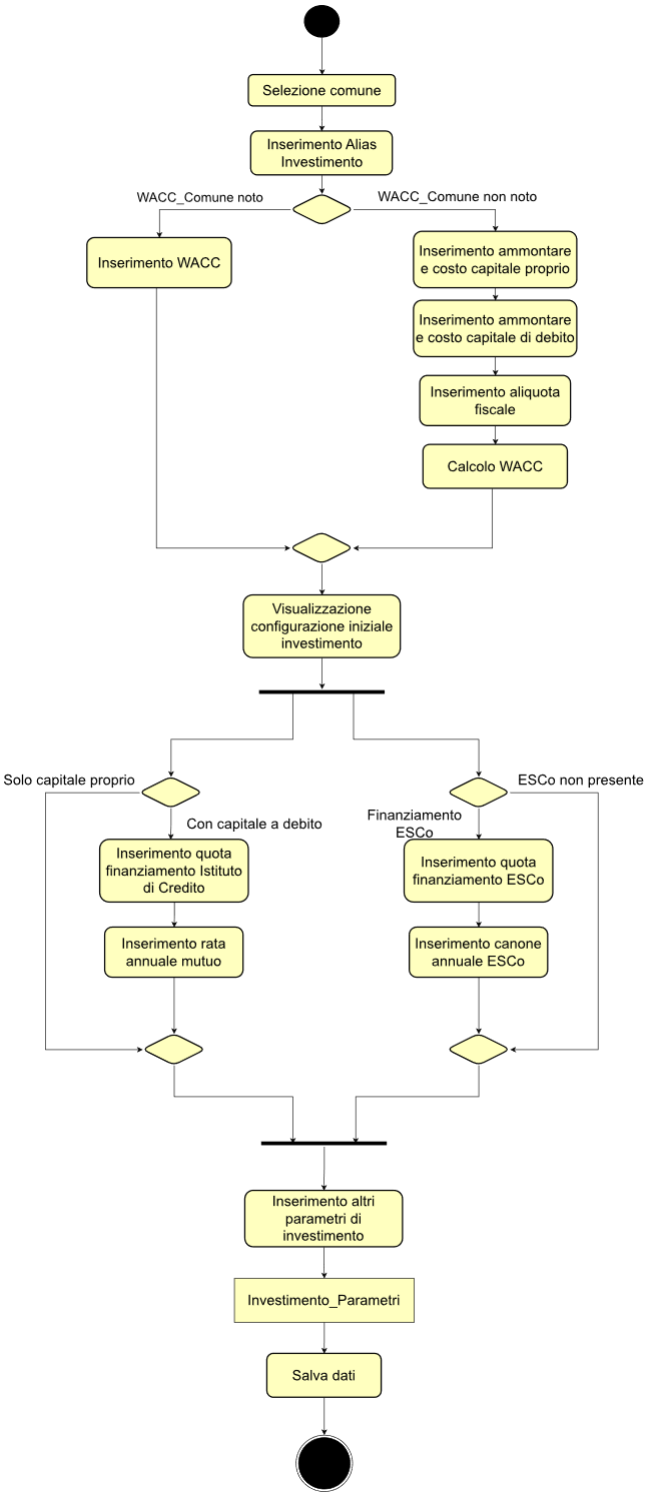


*Figura 4 UC.3 Creazione nuovo investimento*

**Dettagli:** in questo caso d'uso, l'utente, dopo aver selezionato il proprio comune e aver inserito un alias per il nuovo investimento, deve inserire una serie di parametri economici obbligatori in un apposito form, che conterrà inizialmente i valori di default. Una volta inseriti questi dati iniziali, verrà presentato all'utente un secondo form con un'altra serie di dati anch'essi obbligatori.

**Precondizioni:** l'utente si è autenticato nel portale PELL e ha selezionato il modulo SAVE.

**Post condizioni:** i dati vengono salvati nel database e sono liberamente consultabili dall'utente che li ha creati.

## Use Case 4: Nuova simulazione

**Descrizione:** l'utente attiva questo use case per analizzare pro/contro di una ipotetica riqualificazione dell'impianto, in termini di costi e ricavi.

**Attori:** utente PELL.

**Flusso:** il flusso di esecuzione è descritto dal seguente diagramma di attività (*Figura 5*) (*Figura 6*):

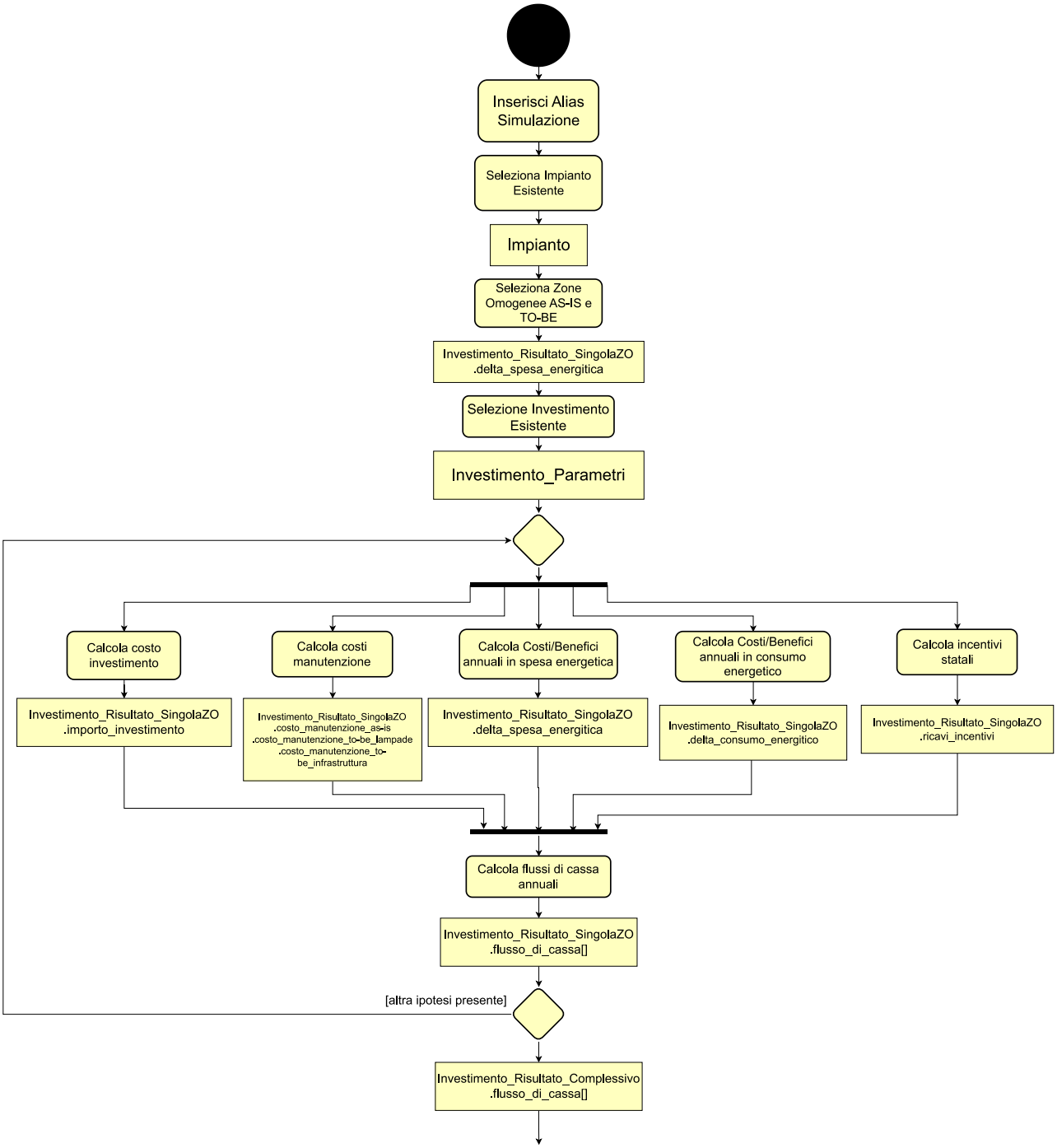


*Figura 5 UC.4 Nuova simulazione (parte 1)*

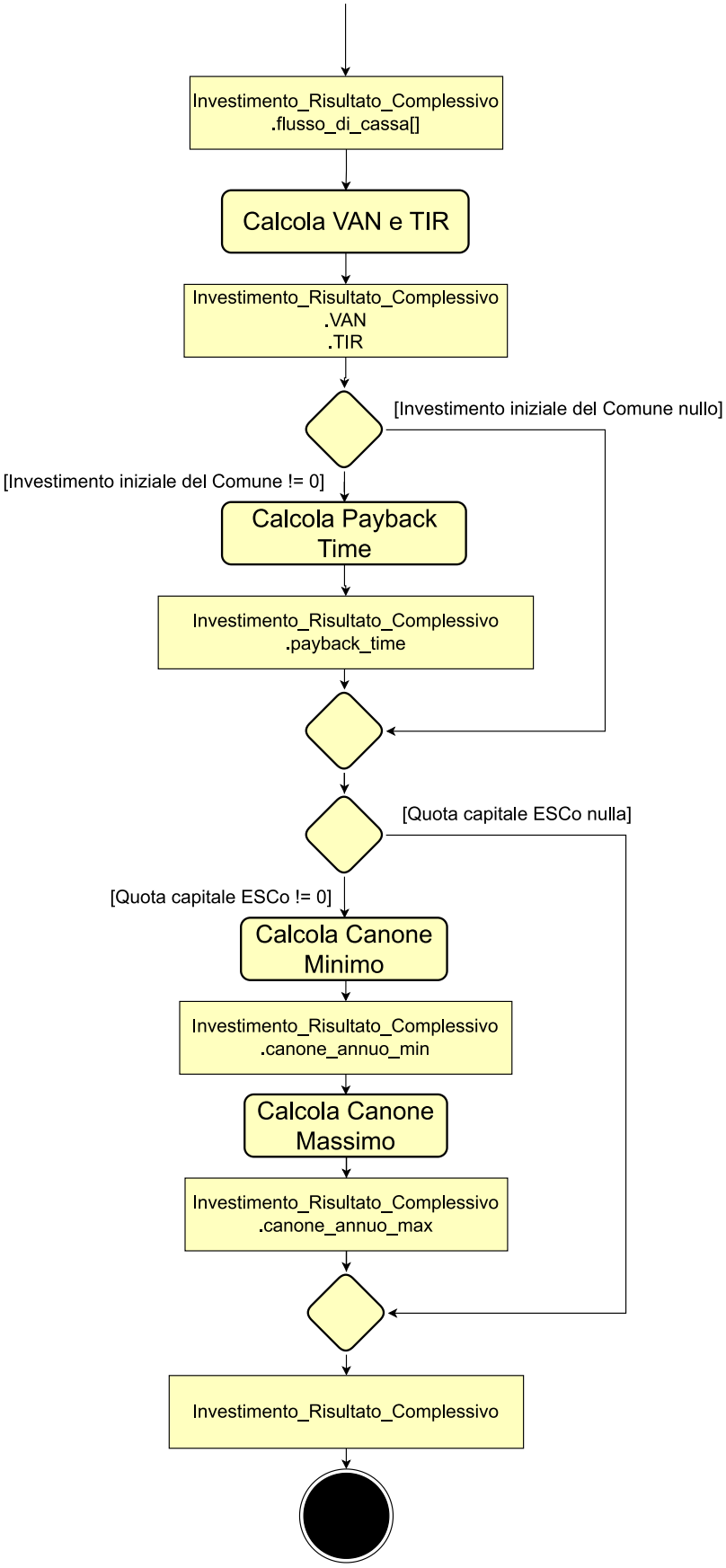


*Figura 6 UC.4 Nuova simulazione (parte 2)*

**Dettagli:** l'utente dopo aver inserito l'alias della nuova simulazione e aver scelto l'impianto da sottoporre ad analisi, deve selezionare una o più zone omogenee AS-IS e le relative zone omogenee TO-BE. Successivamente l'utente deve selezionare un investimento. Quando i dati necessari sono stati selezionati, vengono calcolati una serie di parametri economici che andranno a formare la nuova simulazione. Questi parametri sono:

- Costo investimento
- Costi manutenzione
- Costi/benefici in spesa energetica
- Costi/benefici in consumo energetico
- Incentivi statali

Una volta calcolati questi dati è possibile ottenere una stima dei flussi di cassa annuali per l'intero impianto, che verrà usata per i successivi calcoli di:

- VAN (Valore Attuale Netto), indice della redditività di un investimento, ovvero indica se il progetto genererà un valore aggiunto nel tempo.
- TIR (Tasso Interno di Rendimento), che fornisce un'indicazione del rendimento effettivo di un investimento, in combinazione con il valore del VAN
- Payback Time, ovvero il numero di anni necessari per compensare l'investimento iniziale, iniziare generare dei profitti e non ritornare in perdita. Per maggiori dettagli, ci riferiamo alla sezione 4.11 (Calcolo del Payback Time) di implementazione della funzionalità.
- Canone minimo, che rappresenta il canone che l'ente deve pagare al soggetto privato per rientrare nei costi iniziali.
- Canone massimo, che è il canone che si può pagare al soggetto privato per un progetto del simile a quello intrapreso.

Effettuati tutti i calcoli, verrà mostrato all'utente una pagina contenente tutti i risultati dell'analisi economica, con grafici e dati.

**Precondizioni:** l'utente si è autenticato nel portale PELL, ha selezionato il modulo SAVE e gli use case Gestione Impianto e Nuovo investimento sono stati eseguiti con successo.

**Post condizioni:** i dati dell'analisi economica vengono salvati nel database e sono liberamente consultabili dall'utente che li ha richiesti.

## Use Case 5: Visualizzazione risultati

**Descrizione:** l'utente attiva questo use case per visualizzare i risultati dell'analisi economica, in particolare una serie di grafici e tabelle che riassumono i vantaggi e svantaggi dell'ipotetica riqualificazione.

**Attori:** utente PELL.

**Flusso:** il flusso di esecuzione è descritto dal seguente diagramma di attività (*Figura 7*) (*Figura 8*):

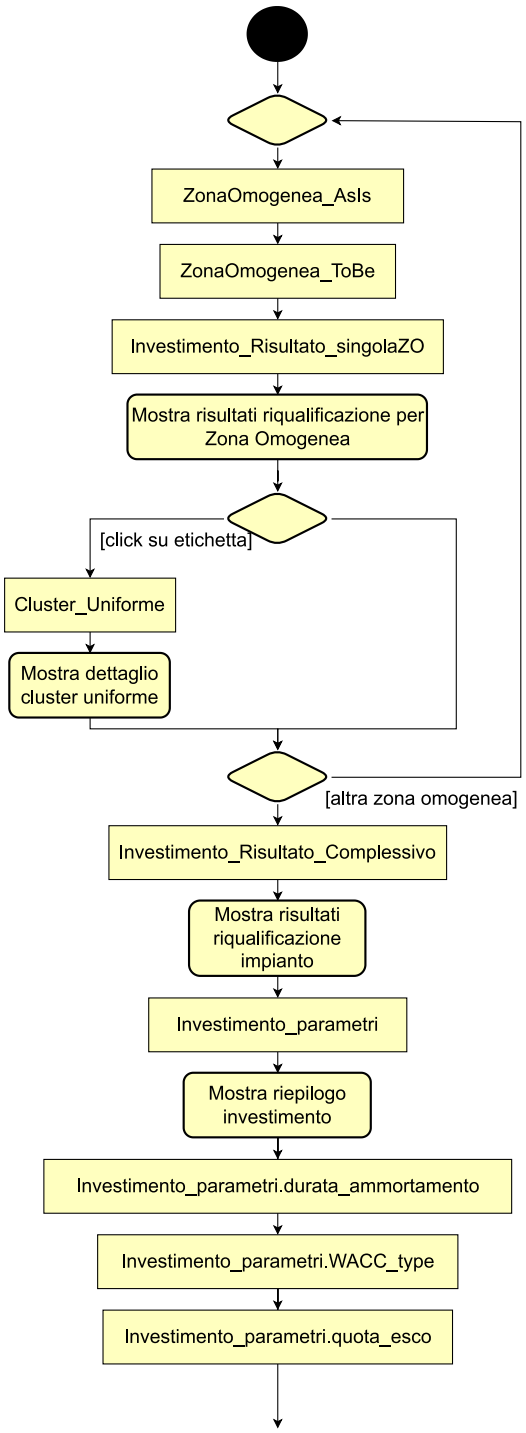


*Figura 7 UC.5 Visualizzazione risultati (parte 1)*

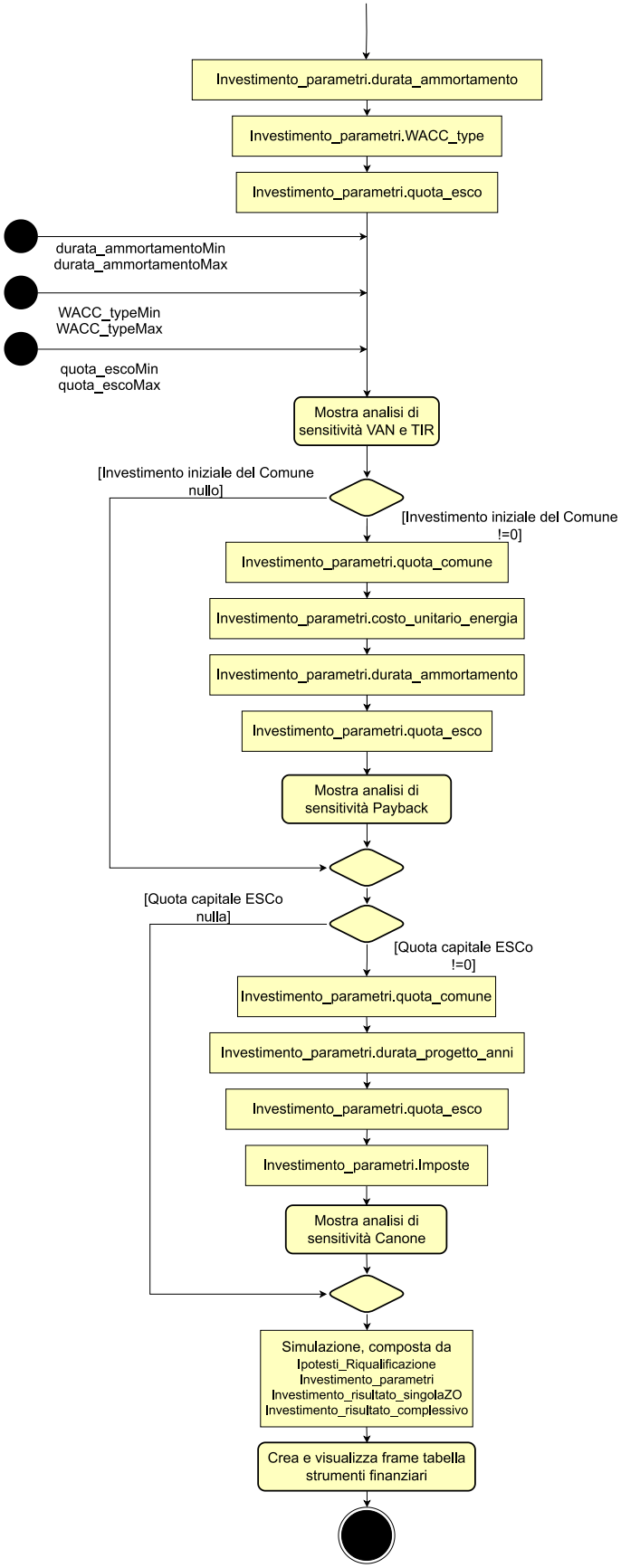


*Figura 8 UC.6 Visualizzazione risultati (parte 2)*

**Dettagli:** in questo caso d'uso vari dati vengono mostrati all'utente. Inizialmente vengono visualizzati i parametri, calcolati nella prima parte dello use case Nuova Simulazione, per ogni zona omogenea (sia AS-IS che TO-BE). In seguito, verranno presentati i dati aggregati per l'impianto, un riepilogo dei dati di investimento scelto e in coda ad essi saranno mostrati, infine, i risultati della seconda parte dello use case Nuova Simulazione, ovvero:

- Analisi di sensitività VAN e TIR
- Analisi di sensitività Payback time
- Analisi di sensitività canone minimo
- Analisi di sensitività canone massimo

**Precondizioni:** l'utente si è autenticato nel portale PELL, ha selezionato il modulo SAVE e ha eseguito con successo lo use case Nuova Simulazione.

**Post condizioni:** i dati della simulazione vengono salvati nel database e sono liberamente consultabili dall'utente che li ha richiesti.

## 3.2. Requisiti funzionali

Dopo la definizione dei casi d'uso, riportiamo tutti i requisiti funzionali del modulo SAVE.

**R.1 Calcolo Importo investimento per zona omogenea**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare il valore dell'importo dell'investimento in base alla zona omogenea TO-BE scelta dall'utente.

**R.2 Calcolo del delta consumo energetico tra zona omogenea AS-IS e TO-BE**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare la differenza tra il consumo energetico della zona omogenea AS-IS scelta dall'utente e il consumo energetico della zona omogenea TO-BE associata ad essa.

**R.3 Calcolo del delta spesa energetica tra zona omogenea AS-IS e TO-BE**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare la differenza tra la spesa energetica della zona omogenea AS-IS scelta dall'utente e la spesa energetica della zona omogenea TO-BE associata ad essa.

**R.4 Calcolo degli incentivi statali**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare gli incentivi statali per la zona omogenea scelta dall'utente per un numero di anni pari alla durata degli incentivi specifica dell'investimento.

**R.5 Calcolo dei costi di manutenzione**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare i costi di manutenzione per la zona omogenea scelta dall'utente, per tutta la durata dell'ammortamento specifico dell'investimento scelto.

**R.6 Calcolo flussi di cassa per zona omogenea**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare il vettore dei flussi di cassa per la zona omogenea AS-IS scelta dall'utente, tenendo conto della zona omogenea TO-BE associata.

**R.7 Calcolo dei flussi di cassa per impianto**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare i flussi di cassa per l'intero impianto scelto dall'utente, sommando i flussi di cassa delle singole coppie di zono omogenea che lo compongono.

**R.8 Calcolo importo investimento per impianto**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare l'importo dell'investimento necessario per la procedura di ammodernamento dell'intero impianto scelto dall'utente. Questo calcolo aggrega i singoli importi calcolati per ogni coppia di zone omogenea AS-IS/TO-BE.

**R.9 Calcolo VAN per impianto**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare il VAN (Valore Attuale Netto) per l'intero impianto scelto dall'utente.

**R.10 Calcolo del TIR per impianto**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare il TIR (Tasso Interno di Rendimento) per l'impianto scelto dall'utente.

**R.11 Calcolo del Payback Time**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare il payback time per l'impianto scelto, tenendo conto dei flussi di cassa cumulativi.

**R.12 Calcolo del canone minimo**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare il canone minimo, in base alla quota da corrispondere al soggetto privato secondo i parametri dell'investimento scelto, per l'impianto selezionato dall'utente.

**R.13 Calcolo del canone massimo**

*Descrizione:* Il modulo SAVE deve essere in grado di calcolare il canone massimo, in base alla quota da corrispondere al soggetto privato secondo i parametri dell'investimento scelto, per l'impianto selezionato dall'utente.

**R.14 Calcolo aggregato dei risultati**

*Descrizione:* Il modulo SAVE deve essere in grado di aggregare tutti i risultati delle precedenti operazioni, in modo da rispettare il formato atteso per poter essere visualizzato correttamente dall'utente.

## 3.3. Pattern architetturale MVC e classi helper

Nell'ambito dello sviluppo e design di questa soluzione software, si è reso necessario l'utilizzo di una determinata soluzione tecnica che possa rendere fattibile ed efficiente il sistema descritto dalle analisi dei requisiti e Use cases.

In particolare, le applicazioni web fanno uso di una ricorrente soluzione tecnica, che è l'utilizzo di un pattern architetturale MVC, in modo da [3] organizzare e impostare lo schema di base del sistema software: lo stesso framework Laravel utilizzato per la scrittura della applicazione è basato sull'architettura MVC.

Facciamo notare come pattern architetturale e design pattern siano in realtà due concetti sostanzialmente diversi: mentre il primo si pone l'obiettivo di dare un'organizzazione alla forma del software prodotto, il secondo è una soluzione progettuale generale ad un problema ricorrente: il design pattern opera quindi ad un livello più ristretto e specifico rispetto all'architettura, che non è

Il pattern architetturale MVC (Model View Controller) è utilizzato largamente in ambito web, poichè il vantaggio di poter separare il layer di interfaccia grafica (UI) dalle altre parti del sistema, come indicato nella *Figura 9*.

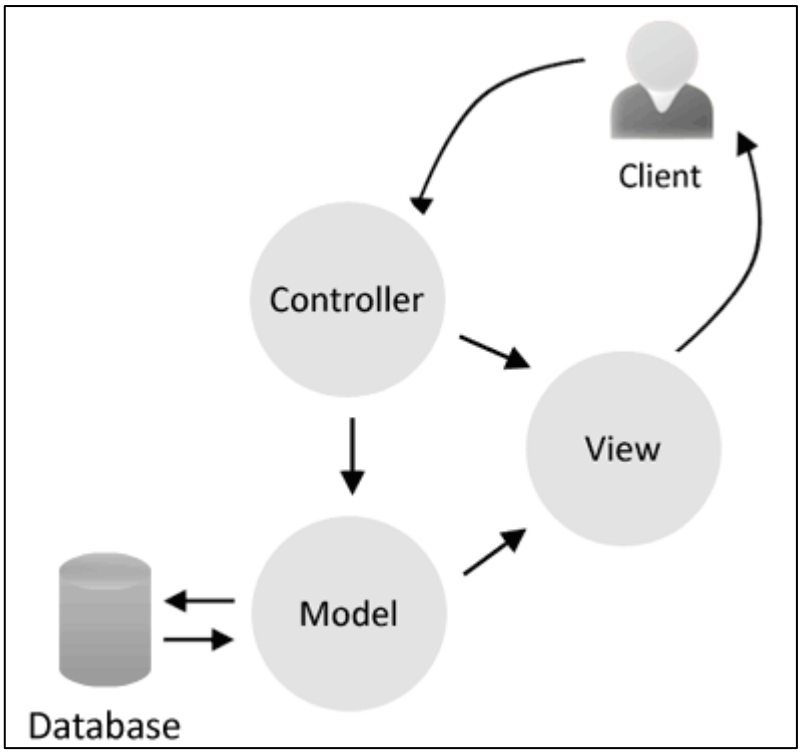


*Figura 9 Schema pattern MVC*

In particolare, come mostrato nella *Figura 10*:

- Il model contiene le varie classi di dati e informazioni, che possono essere recuperati da basi di dati e manipolati

- La view contiene gli oggetti utilizzati per mostrare l'aspetto dei dati proveniente dai model sull'interfaccia grafica
- Il controller contiene gli oggetti che controllano e guidano le interazioni dell'user tra view e model.
  Per la separazione del Model dalla View viene di solito utilizzato l'Observable design pattern, in modo da rendere asincrone e non bloccanti le varie operazioni.

Alcuni principi di design

- Divide et conquer: i tre componenti possono essere in qualche modo progettati inndipendentemente
- Aumento della coesione: i componenti dei layer avranno una coesione maggiore rispetto ad un View e Controller uniti in un singolo layer
- Accoppiamento ridotto: I canali di comunicazione tra i tre componenti sono minimi
- Aumento del riutilizzo del codice: La view e il controller normalmente fanno un uso estensivo di componenti riutilizzabili per vari controlli di UI
- Flessibilità di progettazione: è abbastanza semplice sostituire componenti di UI modificando solo la View o il Controller, o entrambi
- Progettato per essere testato: si può testare la logica e la UI in maniera separata

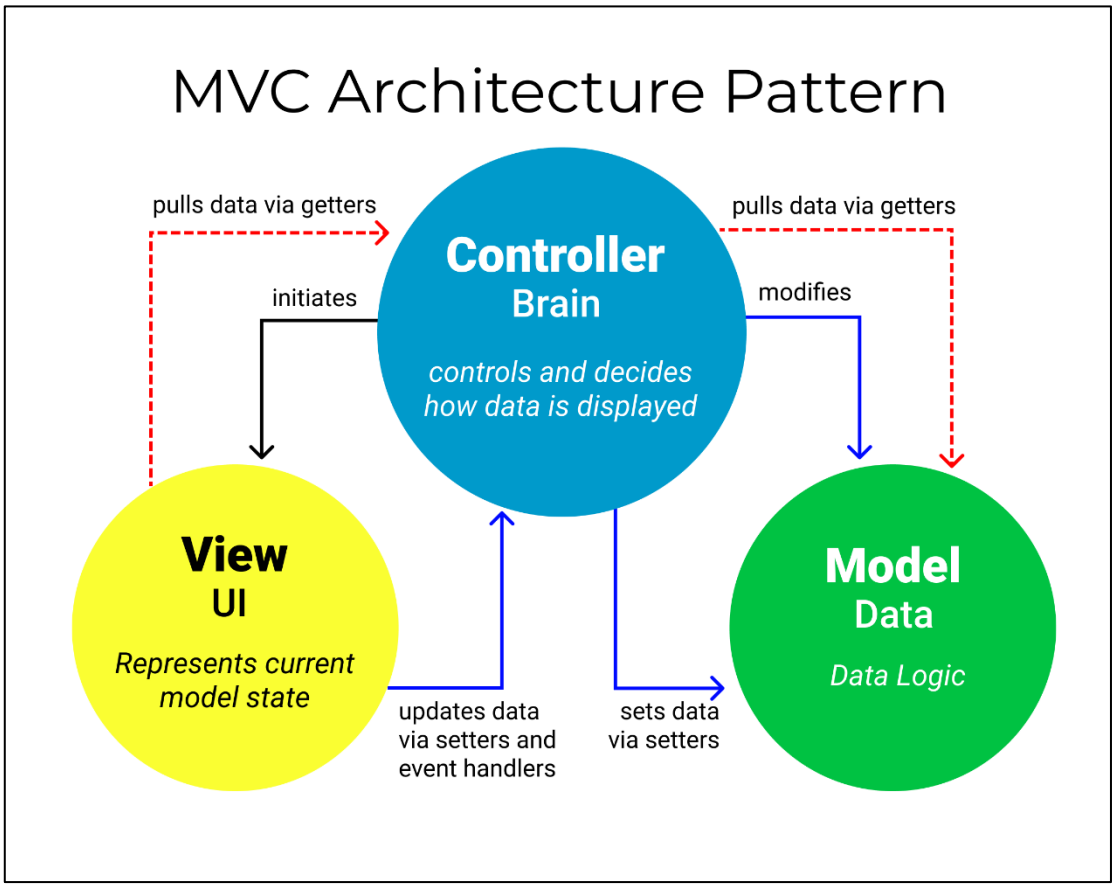


*Figura 10 Pattern architetturale MVC (dettagli)*

## 3.4. Evoluzione del modello dei dati

Rispetto alla versione precedente della piattaforma PELL IP, vi è una sostanziale differenza che riguarda il concetto di Zona Omogenea. Precedentemente, infatti, un impianto veniva diviso in varie zone omogenee, che raggruppavano gli apparecchi con caratteristiche illuminotecniche identiche.

Nella nuova versione, è stata aggiunta una entità inedita, che prende il nome di cluster uniforme. Ora le zone omogenee possono raggruppare apparecchi con differenti tecnologie, differenti lampade, differenti consumi e configurazioni, a patto che il rapporto tra il costo dell'energia dell'anno precedente e i consumi energetici dell'anno precedente sia costante.

I cluster uniformi, invece, raggruppano quegli apparecchi che hanno caratteristiche tecniche uguali, ovvero hanno assunto il ruolo che avevano le vecchie zone omogenee.

Con l'introduzione dei concetti fondamentali di zona omogenea e cluster uniforme, si è reso quindi necessario un cambiamento radicale del modello dei dati, con l'aggiunta di nuove entità e la modifica di quelle già esistenti. Dopo aver effettuato un'analisi dei dati necessari per la realizzazione di questo nuovo modello del modulo SAVE, abbiamo individuato tutti i parametri necessari di impianti, zone omogenee, cluster, investimenti e analisi. Le tabelle del database che abbiamo realizzato e utilizzato durante lo sviluppo sono:

- save_plants (*Tabella 1*): tabella degli impianti
- save_investments (*Tabella 2*): tabella degli investimenti
- save_has (*Tabella 3*): tabella delle zone omogenee
- save_clusters (*Tabella 4*): tabella dei cluster uniformi
- save_analysis (*Tabella 5*): tabella delle analisi

### Tabella save_plants

| Attributo | Min | Max | Altri Vincoli | Descrizione |
|---|---|---|---|---|
| id | - | - | Univoco | Chiave primaria |
| label_plant | - | - | - | Nome dell'impianto |
| municipality_code | - | - | - | Codice identificativo del comune |
| user_id | - | - | - | Id utente che ha creato l'impianto |

*Tabella 1 Schema tabella save_plants*

## Tabella save_investments

| Attributo | Min | Max | Altri Vincoli | Descrizione |
|---|---|---|---|---|
| id | - | - | Univoco | Chiave primaria |
| user_id | - | - | | Id utente che ha creato l'investimento |
| label_investment | - | - | - | Nome dell'investimento |
| wacc | 0 | 100 | - | Costo Medio Ponderato del Capitale |
| share_municipality | 0 | 100 | share_municipality + share_bank + share_esco = 100 | Quota parte dell'investimento risorse Comune |
| share_bank | 0 | 100 | share_municipality + share_bank + share_esco = 100 | Quota parte dell'investimento Istituto di Credito |
| mortgage_installment | 0 | - | - | Valore della rata annuale Comune verso Istituto di Credito |
| fee_esco | 0 | - | - | Canone annuale Comune verso ESCo |
| share_esco | 0 | 100 | share_municipality + share_bank + share_esco = 100 | Quota parte dell'investimento ESCo |
| energy_unit_cost | 0 | - | - | Costo per l'acquisto dell'energia |
| incentives_duration | 0 | - | - | Numeri di anni nei quali si può usufruire degli incentivi statali |
| tep_kwh | 0 | - | - | Fattore di conversione KWh TEP |
| tep_value | 0 | - | - | Valore monetario del singolo TEP |
| management_cost | 0 | - | - | Costo annuale gestione della zona omogenea |
| duration_amortization | 0 | - | - | Durata dell'ammortamento in anni |
| project_duration | 0 | - | - | Durata del progetto di finanziamento |
| taxes | 0 | 100 | - | Tasso percentuale relativo alle imposte a carico del soggetto privato |
| cost_funded | 0 | - | - | Costo annuale per la stipulazione di assicurazioni... |

*Tabella 2 Schema tabella save_investments*

## Tabella save_has

| Attributo | Min | Max | Altri Vincoli | Descrizione |
|---|---|---|---|---|
| id | - | - | Univoco | Chiave primaria |
| plant_id | - | - | - | Id impianto a cui appartiene la zona |
| label_ha | - | - | - | Nome della zona omogenea |
| type_ha | - | - | AS_IS o TO_BE | Tipo di zona omogenea |
| ref_as_is_id_ha | - | - | - | Usato solo dalle zone TO-BE, indica l'id della zona AS-IS a cui è associata |
| is_ready | - | - | - | (campo utile al front-end) |
| lamp_cost | 1 | - | > 1 | Costo lampada |
| lamp_disposal | 0 | - | > 1 | Costo smaltimento lampada |
| lamp_maintenance_interval | 0 | - | - | Intervallo manutenzione lampada, corrispondente al MTTF della tecnologia della lampada |
| panel_cost | 0 | - | - | Costo quadro |
| panel_num | 1 | - | > 1 | Numero quadri elettrici presenti in un cluster |
| prodromal_activities_cost | 0 | - | - | Costo attività preliminari per la diagnosi tecnica e la progettazione |
| system_renovation_cost | 0 | - | - | Costo di ammodernamento/rifacimento impianto elettrico |
| infrastructure_maintenance_cost | 0 | - | - | Costo medio della componente infrastrutturale relativa alla singola lampada |
| infrastructure_maintenance_interval | 0 | - | - | Intervallo medio di manutenzione della componente dell'infrastruttura |

*Tabella 3 Schema tabella save_has*

## Tabella save_clusters

| Attributo | Min | Max | Altri Vincoli | Descrizione |
|---|---|---|---|---|
| id | - | - | Univoco | Chiave primaria |
| ha_id | - | - | - | Id zona omogenea a cui appartiene il cluster |
| label_cluster | - | - | - | Nome del cluster |
| type_cluster | - | - | AS_IS o TO_BE | Tipo di cluster |
| ref_as_is_id_cluster | - | - | - | Usato solo dai cluster TO-BE, indica l'id del cluster AS-IS a cui è associato |
| is_to_be_featured | - | - | - | Indica quale cluster TO-BE verrà usato per l'ipotesi di riqualificazione |
| lamp_technology | - | - | - | Tecnologia lampada |
| lamp_num | 1 | - | > 1 | Numero apparecchi/lampadine del singolo cluster uniforme |
| device_num | 1 | - | > 1 | Numero lampioni all'interno del cluster |
| average_device_power | 0 | - | - | Potenza media ai morsetti per tutti gli apparecchi del cluster |
| dimmering | 0 | 100 | - | Riduzione di potenza rispetto al totale se funzionamento dimmering |
| hours_full_light | 0 | 4999 | hour_full_light + hour_dimmering_light < 5000 | Ore accensione a piena potenza |
| hours_dimmering_light | 0 | 4999 | hour_full_light + hour_dimmering_light < 5000 | Ore accensione a potenza ridotta |

*Tabella 4 Schema tabella save_clusters*

## Tabella save_analysis

| Attributo | Min | Max | Altri Vincoli | Descrizione |
|---|---|---|---|---|
| id | - | - | Univoco | Chiave primaria |
| label_analysis | - | - | - | Nome dell'analisi |
| user_id | - | - | - | Id utente che ha richiesto l'analisi |
| plant_id | - | - | - | Id dell'impianto analizzato |
| investment_id | - | - | - | Id dell'investimento usato per l'ipotesi di riqualificazione |

*Tabella 5 Schema tabella save_analysis*

## 3.5. Presentazione dell'output

Il risultato dell'elaborazione viene presentato sotto forma di oggetto JSON, ideale per l'utilizzo in ambito web. In particolare, mettiamo a disposizione delle API che una volta richiamate tramite protocollo http restituiscono la richiesta sulla base dei parametri specificati

### 3.5.1. API Calcolo dell'impianto

Il calcolo dell'impianto presenta tutte le informazioni relative all'impianto a partire dai dati specificati dell'investimento, senza enfasi sulle variazioni di particolari parametri di investimento.

L'oggetto restituito fornisce le seguenti informazioni:

- municipality: fornisce l'etichetta assegnata all'impianto in fase di creazione
- plants: fornisce i risultati dell'elaborazione di ognuna delle zone omogenee all'interno dell'impianto
- total: contiene i risultati di tutte le zone omogenee della sezione plants aggregati, che mostra quindi una situazione complessiva tra il prima e il dopo dell'impianto
- financement: contiene le caratteristiche dell'investimento calcolate sulla base delle caratteristiche dell'impianto e della riqualificazione scelta

### 3.5.2. API Van e TIR

Questa API è dedicata al calcolo delle caratteristiche dell'investimento del VAN (Valore Attuale Netto) e TIR (Tasso Interno di Rendimento) al variare del WACC (Costo Medio Ponderato del Capitale) e della durata dell'ammortamento

Vengono specificati tre valori possibili per i parametri in input WACC e durata ammortamento in modo da poter generare una tabella dove mostrare la variazione di queste caratteristiche di investimento.

L'oggetto restituito fornisce le seguenti informazioni:

- van: contiene un array di VAN che variano in base alla durata dell'ammortamento utilizzata per effettuare il calcolo. Ogni oggetto dell'array contiene a sua volta un array di 3 elementi, ognuno degli elementi utilizza un diverso WACC tra quelli specificati
- tir: contiene un array di TIR che variano in base alla durata dell'ammortamento utilizzato per effettuare il calcolo. Qui le celle contengono risultati singoli poiché il calcolo del TIR dipende solo dalla variazione della durata dell'ammortamento

### 3.5.3. API Payback

Questa API è dedicata al calcolo della caratteristica dell'investimento del Payback time al variare dell'unità di costo dell'energia.

Vengono specificati un minimo ed un massimo di costo per l'energia come parametri, e a partire da questo intervallo si ottengono una serie di punti configurabili che possono poi essere rappresentati su un grafico come andamento

L'oggetto restituito fornisce le seguenti informazioni:

- data: contiene l'array di punti calcolati al variare del costo di un'unità di energia, ogni punto varia in maniera incrementale dal minimo al massimo. Ogni cella di questo array contiene due elementi: il primo è il costo per unità utilizzato ed il secondo il valore ottenuto dal calcolo

### 3.5.4. API Altre modalità di finanziamento

Questa API è dedicata al calcolo per mostrare la convenienza, se esiste, delle altre modalità di finanziamento (tramite banca o soggetto terzo).

Vengono specificati un minimo ed un massimo per la durata dei costi della quota finanziata (e quindi della durata dell'investimento) e a partire da questo intervallo si ottengono una serie di punti configurabili che possono poi essere rappresentati su un grafico come andamento.

Altri parametri in input sono la percentuale tasse sull'importo dell'investimento e la percentuale della quota finanziata sul totale richiesto.

L'oggetto restituito fornisce le seguenti informazioni:

- fee_min: contiene l'array di punti calcolati al variare della durata dell'investimento, dove ognuna della cella contiene gli anni di investimento e il canone minimo calcolato per quegli anni
- fee_max: contiene l'array di punti calcolati al variare della durata dell'investimento, dove ognuna della cella contiene gli anni di investimento e il canone massimo calcolato per quegli anni

## 3.6. Interfaccia web del modulo SAVE nel portale PELL

L'interfaccia del modulo SAVE, che presenta i risultati dell'analisi economico finanziaria, è suddivisa in quattro sezioni:

- Report analisi (*Figura 11*)
- VAN e TIR (*Figura 12*)
- Payback (*Figura 13*)
- Altre modalità di finanziamento (*Figura 14*)

### Report

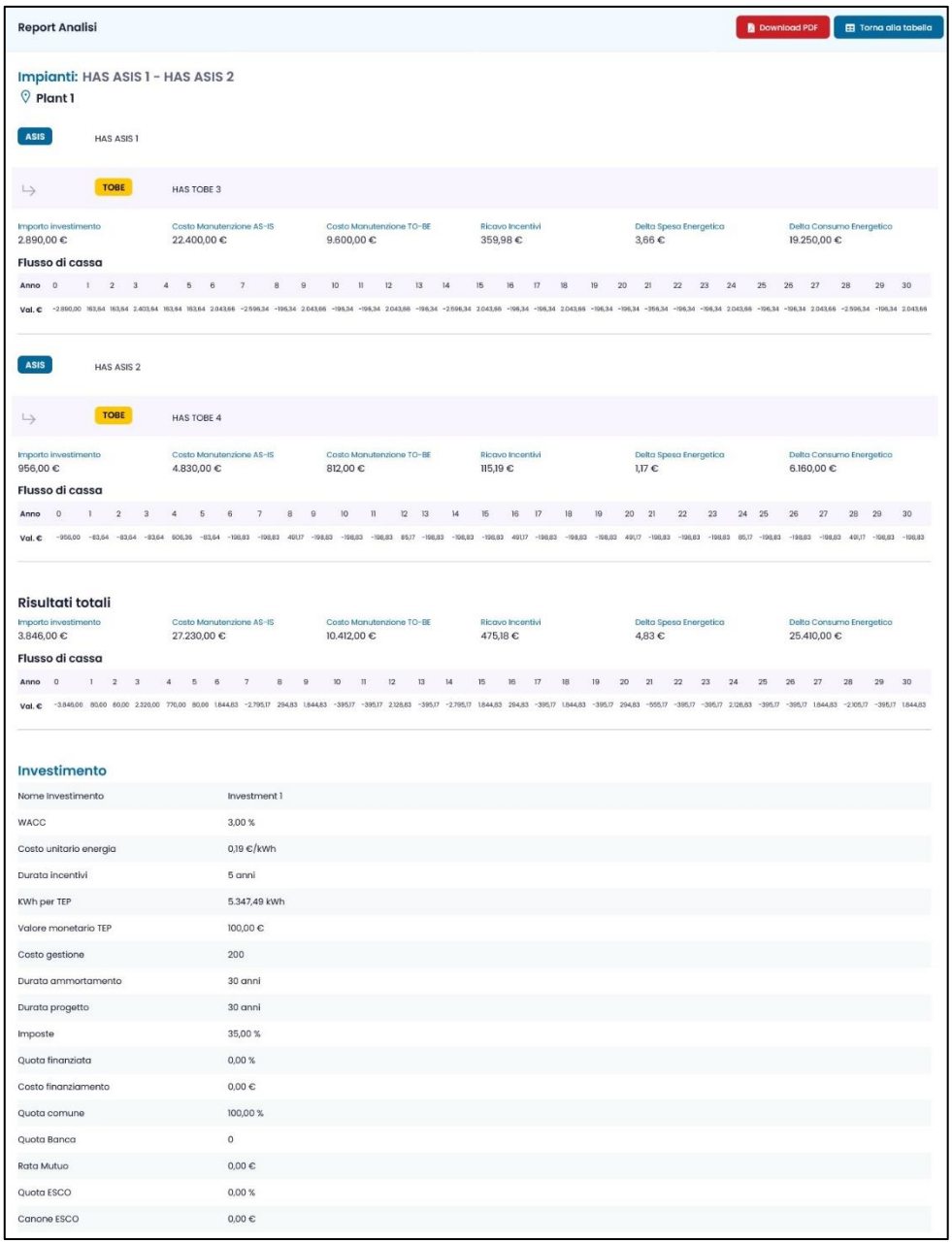


*Figura 11 Modulo SAVE: interfaccia report analisi*

## Sezione VAN E TIR

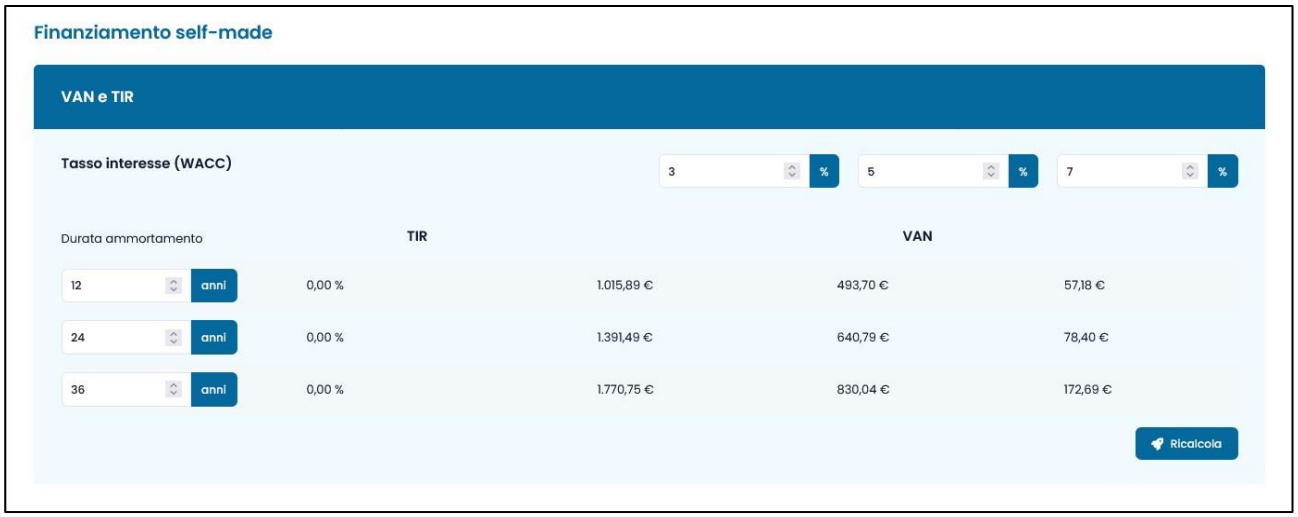


*Figura 12 Modulo SAVE: interfaccia indici VAN e TIR*

## Sezione Payback

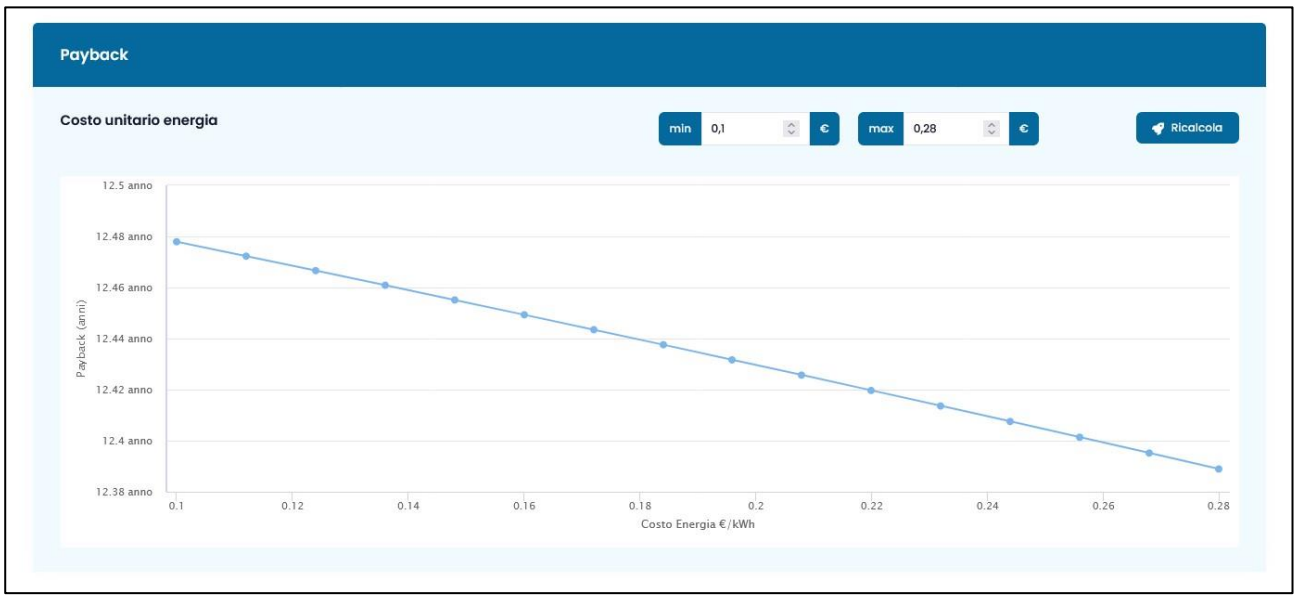


*Figura 13 Modulo SAVE. interfaccia Payback Time*

## Sezione Altre modalità di finanziamento


*Figura 14 Modulo SAVE. interfaccia canone minimo e massimo*

# 4 Sviluppo funzionalità della classe SaveHelper

Per programmazione dinamica intendiamo una tecnica di progettazione di algoritmi basata sulla divisione del problema in sotto problemi e sull'utilizzo di sottostrutture per la memorizzazione dei risultati dei precedenti [4].

In particolare, la sottostruttura utilizzata, ovvero una sotto-soluzione ottima, viene definita ottimale se può essere utilizzata per trovare una soluzione ottima dell'intero problema. Individuate quindi le sottostrutture da calcolare e memorizzandole per riutilizzarle quando sono poi disponibili è l'approccio principale della programmazione dinamica [5].

Per l'applicazione possono essere utilizzati due diversi approcci:

- Top-down: detta anche programmazione ricorsiva, il problema è risolto individuando la soluzione base ed eseguendola fino ad ottenere la soluzione completa
- Bottom-up: si risolvono innanzitutto i sotto problemi per poi ricomporli e trovare la soluzione completa.

Considerando quindi questo approccio, abbiamo prima individuato i sotto problemi di livello di più basso, necessari alla soluzione degli altri, per poi unirli in una funzione che li richiamasse tutti

In particolare, troviamo nel livello più basso:

- Calcola Importo Investimento per ogni zona omogenea
- Calcolo consumo energetico per ogni zona AS IS
- Calcolo consumo energetico per ogni zona TO BE
- Calcolo spesa energetica per ogni zona AS IS
- Calcolo spesa energetica per ogni zona TO BE
- Calcolo totale lampade per ogni zona omogenea
- Calcolo VAN per impianto
- Calcolo TIR per impianto
- Calcolo Payback time
- Calcolo Canone Minimo

Ad un livello immediatamente successivo, troviamo tutte le funzioni che utilizzano le precedenti elencate:

- Calcola Importo Investimento per Impianto
- Calcolo Delta consumo energetico per zona omogenea
- Calcolo Delta spesa energetica per zona omogenea
- Calcolo Delta spesa energetica per impianto
- Calcolo Costi di manutenzione per zona omogenea
- Calcolo costo manutenzione infrastruttura per zona omogenea
- Calcolo flussi di cassa per zona omogenea

All'ultimo livello abbiamo tutte le funzioni utilizzate dalle precedenti

- Calcolo Importo investimento per impianto

- Calcolo costi di manutenzione zone AS IS per impianto
- Calcolo costi di manutenzione zone TO BE per impianto
- Calcolo contributo incentivi per impianto
- Calcolo Delta spesa energetica per Impianto
- Calcolo Delta consumo energetico per Impianto
- Calcolo Canone Massimo
- Calcolo Flussi di cassa per Impianto

Una funzione esegue il calcolo completo di tutte le informazioni, inserendole dentro ad un'oggetto di output

- Calcolo pilota

Di seguito abbiamo riportato la lista di requisiti individuati in fase di progettazioni e per ognuno di questi abbiamo spiegato i vari metodi utilizzati per raggiungere l'obiettivo. La descrizione di ogni funzione riportata segue questo schema:

- Descrizione: una breve descrizione della funzione
- Valori validati: i valori ritenuti accettabili in output
- Formula matematica: la formula di partenza su cui si base il codice
- Pseudocodice: per mostrare la funzione abbiamo adottato l'uso di un pseudocodice PHP-like, in modo che non si discosti troppo dal codice originale, senza comunque usare costrutti specifici di Laravel.

## 4.1. Calcolo Importo investimento per zona omogenea (R.1)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare il valore dell'importo dell'investimento in base alla zona omogenea TO-BE scelta dall'utente.

### 4.1.1. Funzione calcolaImportoInvestimentoPerHA

Descrizione: la funzione calcola il costo dell'investimento necessario per effettuare l'operazione di ammodernamento per la zona omogenea TO-BE specificata, in particolare esegue un calcolo per ogni cluster della zona omogenea specificata, aggrega i risultati e somma, infine, i costi comuni associati alla zona.

Valori validati: $[0, +\infty)$

Formula matematica corrispondente:

$$Importo\_investimento_{ZO} = \sum_{i=1}^{\#Clusters} (costoMedioLampada_i + costoInfrastruttura + costoMedioSmaltimentoLampada_i) * nLampade_i + costoRifacimentoImpElett + costoAttivitàProdromiche + costoQuadro * nQuadriEl$$

Lo pseudocodice della funzione è il seguente (*Funzione 1*):

```
function calcolaImportoInvestimentoPerHA(Obj ZonaOmogenea){
    clusters = ZonaOmogenea.Clusters;
    sommaParziale = 0;
    for (i = 0; i < clusters.Length; i++)
    {
        cluster = clusters[i];
        sommaParziale += (ZonaOmogenea.lamp_cost
                + ZonaOmogenea.infrastructure_maintenance_cost
                + ZonaOmogenea.lamp_disposal) * cluster.lamp_num;
    }

    sommaParziale += ZonaOmogenea.system_renovation_cost
                  + ZonaOmogenea.prodromal_activities_cost
                  + (ZonaOmogenea.panel_cost
                  * ZonaOmogenea.panel_num);
```

```
    return sommaParziale;
}
```

*Funzione 1 Pseudocodice calcolaImportoInvestimentoPerHA*

## 4.2. Calcolo del delta consumo energetico tra zona omogenea AS-IS e TO-BE (R.2)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare la differenza tra il consumo energetico della zona omogenea AS-IS scelta dall'utente e il consumo energetico della zona omogenea TO-BE associata ad essa.

### 4.2.1. Funzione calcoloDeltaConsumoEnergeticoPerHAS

Descrizione: la funzione calcola i costi/benefici in consumo energetico risultante dalla riqualificazione. In particolare, questa funzione esegue la sottrazione tra il risultato aggregato delle zone omogenee AS IS e quello delle TO BE e, all'interno dei risultati delle singole zone omogenee, lo posiziona per label corrispondente

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$DeltaConsumoEnergetico_{HA} = ConsumoEnergetico_{HA\ AS\ IS} - ConsumoEnergetico_{HA\ TO\ BE}$$

Lo pseudocodice della funzione è il seguente (*Funzione 2*):

```
function calcoloDeltaConsumoEnergeticoPerHAS(Obj ZOASIS,
                                             Obj ZOTOBE,
                                             Obj result)
{
    delta = ConsumoEnergeticoPerHaASIS(ZOASIS)
          - ConsumoEnergeticoPerHaTOBE(ZOTOBE);

    result.foreach(risultatoSingolaZO => {
        if(risultatoSingolaZO.getAsisName() == ZOASIS.label_ha){
            risultatoSingolaZO.setDeltaEnergyConsumption(delta);
        }
    });
}
```

*Funzione 2 Pseudocodice calcoloDeltaConsumoEnergeticoPerHAS*

### 4.2.2. Funzione calcoloConsumoEnergeticoPerHaASIS

Descrizione: la funzione calcola il consumo energetico per la specificata zona omogenea. In particolare, questa funzione è specifica per le zone AS IS poiché aggrega tutti i risultati dei cluster associati alle zone (ogni zona omogenea AS IS può avere più cluster AS IS)

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$ConsumoEnergetico_{ZO\ ASIS} = \sum_{i}^{\#cluster\ AS\ IS} \left(oreAccPiena_i + \left(1 - \frac{\%dimm_i}{100}\right) * oreDimm_i\right) * Napp_i * \overline{p_{l_i}}$$

dove

- $N_{app}$ : numero di apparecchi nel Cluster uniforme
- $\overline{p_l}$ : media della potenza nel Cluster
- oreAccPiena: ore di accensione a piena potenza degli apparecchi
- oreDimm: ore di accensione in modalità dimmering
- %dimm: percentuale di riduzione della potenza in modalità dimmering

Lo pseudocodice della funzione è il seguente (*Funzione 3*):

```
function calcoloConsumoEnergeticoPerHaASIS(Obj ZonaOmogenea){

    clusters = ZonaOmogenea.Clusters;
    consumoEnergeticoHa = 0;

    //sommatoria per ogni cluster
    for (i = 0; i < clusters.Length; i++) {
        cluster = clusters[i];
        consumoEnergetico =
     (cluster.hours_full_light + (1 - (cluster.dimmering / 100))
     * cluster.hours_dimmer_light )
     * cluster.device_num
     * cluster.average_device_power;

       consumoEnergeticoHa += consumoEnergetico;
    }

    return consumoEnergeticoHa;
}
```

*Funzione 3 Pseudocodice calcoloConsumoEnergeticoPerHaASIS*

### 4.2.3. Funzione calcoloConsumoEnergeticoPerHaTOBE

Descrizione: la funzione calcola il consumo energetico per la specificata zona omogenea. In particolare, questa funzione è specifica per le zone TO BE poiché nella riqualificazione sarà presente un unico cluster.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$ConsumoEnergetico_{ZO\ TO\ BE} = \left(oreAccPiena\ + \left(1 - \frac{\%dimm}{100}\right) * oreDimm\right) * N_{app} * \overline{p_l}$$

dove

- $N_{app}$: numero di apparecchi nel Cluster uniforme
- $\overline{p_l}$ : media della potenza nel Cluster
- oreAccPiena: ore di accensione a piena potenza degli apparecchi
- oreDimm: ore di accensione in modalità dimmering
- %dimm: percentuale di riduzione della potenza in modalità dimmering

Lo pseudocodice della funzione è il seguente (*Funzione 4*):

```
function calcoloConsumoEnergeticoPerHaTOBE(Obj ZonaOmogenea){
    cluster = ZonaOmogenea.Clusters;

    return
     (cluster.hours_full_light + (1 - (cluster.dimmering / 100))
     * cluster.hours_dimmer_light)
     * cluster.device_num
     * cluster.average_device_power;
}
```

*Funzione 4 Pseudocodice calcoloConsumoEnergeticoPerHaTOBE*

## 4.3. Calcolo del delta spesa energetica tra zona omogenea AS-IS e TO-BE (R.3)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare la differenza tra la spesa energetica della zona omogenea AS-IS scelta dall'utente e la spesa energetica della zona omogenea TO-BE associata ad essa.

### 4.3.1. Funzione calcoloDeltaSpesaEnergeticaPerHAS

Descrizione: la funzione calcola i costi/benefici in spesa energetica risultante dalla riqualificazione. In particolare, questa funzione esegue la sottrazione tra il risultato aggregato delle zone omogenee AS IS e quello delle TO BE e, all'interno dei risultati delle singole zone omogenee, lo posiziona per label corrispondente

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$DeltaSpesaEnergetica_{HA} = SpesaEnergetica_{HA\ AS\ IS} - SpesaEnergetica_{HA\ TO\ BE}$$

Lo pseudocodice della funzione è il seguente (*Funzione 5*):

```
function calcoloDeltaSpesaEnergeticaPerHAS(Obj ZOASIS,
                                           Obj ZOTOBE,
                                           Obj result)
{
    delta = SpesaEnergeticaPerHaASIS(ZOASIS)
          - SpesaEnergeticaPerHaTOBE(ZOTOBE);

    result.foreach(risultatoSingolaZO => {
        if(risultatoSingolaZO.getAsisName() == ZOASIS.label_ha){
            risultatoSingolaZO.setDeltaEnergyExpenditure(delta);
        }
    });
}
```

*Funzione 5 Pseudocodice calcoloDeltaSpesaEnergaticaPerHAS*

### 4.3.2. Funzione calcoloSpesaEnergeticaPerHaASIS

Descrizione: la funzione calcola la spesa energetica per la specificata zona omogenea. In particolare, questa funzione è specifica per le zone AS IS poiché aggrega tutti i risultati dei cluster associati alla zona (ogni zona omogenea AS IS può avere più cluster AS IS).

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$SpesaEnergetica_{ZO\ AS\ IS} = \sum_{i}^{\#cluster\ AS\ IS} \left(oreAccPiena_i + \left(1 - \frac{\%dimm_i}{100}\right) * oreDimm_i\right) * Napp_i * \overline{p_{l_i}} * \frac{costo\_unitario\_energia}{1000}$$

dove

- $N_{app}$: numero di apparecchi nel Cluster uniforme
- $\overline{p_l}$: media della potenza nel Cluster
- oreAccPiena: ore di accensione a piena potenza degli apparecchi
- oreDimm: ore di accensione in modalità dimmering
- %dimm: percentuale di riduzione della potenza in modalità dimmering
- costo_unitario_energia: costo dell'energia al kWh

Lo pseudocodice della funzione è il seguente (*Funzione 6*):

```
function calcoloSpesaEnergeticaPerHaASIS(Obj ZonaOmogenea,
                                         float energy_unit_cost)
{

    clusters = ZonaOmogenea.Clusters;
    spesaEnergeticaHa = 0;

    //sommatoria per ogni cluster
    for (i = 0; i < clusters.Length; i++) {
        cluster = clusters[i];
        spesaEnergetica =
                    (cluster.hours_full_light
                    + (1 - (cluster.dimmering / 100))
                    * cluster.hours_dimmer_light )
                    * cluster.device_num
                    * cluster.average_device_power
                    * (energy_unit_cost / 1000);
```

```
        spesaEnergeticaHa += spesaEnergetica;
    }

    return spesaEnergeticaHa;
}
```

*Funzione 6 Pseudocodice calcoloSpesaEnergeticaPerHaASIS*

### 4.3.3. Funzione calcoloSpesaEnergeticaPerHaTOBE

Descrizione: la funzione calcola la spesa energetica per la specificata zona omogenea. In particolare, questa funzione è specifica per le zone TO BE poiché nella riqualificazione sarà presente un unico cluster.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$SpesaEnergetica_{ZO\ TO\ BE} = \sum_{i}^{\#cluster\ TO\ BE} \left(oreAccPiena_i + \left(1 - \frac{\%dimm_i}{100}\right) * oreDimm_i\right) * Napp_i * \overline{p_{l_i}} * \frac{costo\_unitario\_energia}{1000}$$

dove

- $N_{app}$: numero di apparecchi nel Cluster uniforme
- $\overline{p_l}$: media della potenza nel Cluster
- oreAccPiena: ore di accensione a piena potenza degli apparecchi
- oreDimm: ore di accensione in modalità dimmering
- %dimm: percentuale di riduzione della potenza in modalità dimmering
- costo_unitario_energia: costo dell'energia al kWh

Lo pseudocodice della funzione è il seguente (*Funzione 7*):

```
function calcoloSpesaEnergeticaPerHaTOBE(Obj ZonaOmogenea,
                                         float energy_unit_cost)
{
    cluster = ZonaOmogenea.Clusters;

    return (cluster.hours_full_light
           + (1 - (cluster.dimmering / 100))
           * cluster.hours_dimmer_light)
           * cluster.device_num
           * cluster.average_device_power
```

```
            * (energy_unit_cost / 1000);
}
```

*Funzione 7 Pseudocodice calcoloSpesaEnergeticaPerHaTOBE*

## 4.4. Calcolo degli incentivi statali (R.4)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare gli incentivi statali per la zona omogenea scelta dall'utente per un numero di anni pari alla durata degli incentivi specifica dell'investimento.

### 4.4.1. Funzione calcoloIncentiviStatali

Descrizione: la funzione calcola gli incentivi statali, per la specificata zona omogenea, per un totale di anni pari al valore durata_incentivi.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$ricavoIncetivi = \frac{delta\_consumo\_energetico}{kWH\_TEP} * valore\_monetario\_TEP$$

dove

- delta_consumo_energetico: è il risultato della funzione calcoloDeltaConsumoEnergeticoPerHAS
- kWH_TEP: fattore di conversione TEP
- valore_monetario_TEP: valore del singolo TEP

Lo pseudocodice della funzione è il seguente (*Funzione 8*):

```
function calcoloIncentiviStatali(float delta_energy_consumption,
                                 Obj investment)
{

    return delta_energy_consumption > 0 ?
           ((delta_energy_consumption / investment.tep_kwh)
          * investment.tep_value])
          : 0;

}
```

*Funzione 8 Pseudocodice calcoloIncentiviStatali*

## 4.5. Calcolo dei costi di manutenzione (R.5)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare i costi di manutenzione per la zona omogenea scelta dall'utente, per tutta la durata dell'ammortamento specifico dell'investimento scelto.

### 4.5.1. Funzione calcoloCostiManutenzione

Descrizione: la funzione prende il vettore dei costi di manutenzione calcolati per ogni zona omogenea e somma ogni valore, per ottenere un costo totale per tutta la durata dell'ammortamento.

N.B.: il costo per la manutenzione di una zona omogenea AS-IS deve tener conto solo dei costi derivanti dalla sostituzione delle lampade, mentre per le zone omogenee TO-BE bisogna anche tenere conto del costo di realizzazione dell'impianto.

Valori validati: $(-\infty, +\infty)$

dove

- costo_medio_lampada: costo singola lampada
- costo_medio_smaltimento_lampada: costo smaltimento singola lampada
- numero_lampade: numero lampadine nel cluster

Lo pseudocodice della funzione è il seguente (*Funzione 9*):

```
//Calcola costi manutenzione
//calcolo flussi e totale costo manutenzione ASIS e TOBE
for(j = 1; j <= durationAmortization; j++) {

    result_asis_maintenance_cost[j] =
              calcolaCostiManutezionePerHA();

    result_tobe_lamp_cost[j] = calcolaCostiManutezionePerHA();

    result_tobe_infrastructure_cost[j] =
              calcolaCostoManutenzioneInfrastrutturaPerHA()}
result[i].asis_maintenance_cost =
              array_sum(result_asis_maintenance_cost);

result[i].tobe_maintenance_cost =
              array_sum(result_tobe_infrastructure_cost)
              + array_sum(result_tobe_lamp_cost);
```

*Funzione 9 Pseudocodice calcoloCostiDiManutenzione*

### 4.5.2. Funzione calcolaCostiManutezionePerHA

Descrizione: la funzione calcola i costi di manutenzione, per la specificata zona omogenea, per quanto riguarda il costo per la sostituzione delle lampade.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$costiManutenzioneLampade = (costo\_medio\_lampada + costo\_medio\_smaltimento\_lampada) * numero\_lampade$$

dove

- costo_medio_lampada: costo singola lampada
- costo_medio_smaltimento_lampada: costo smaltimento singola lampada
- numero_lampade: numero lampadine nel cluster

Lo pseudocodice della funzione è il seguente (*Funzione 10*):

```
function calcolaCostiManutezionePerHA(obj ZonaOmogenea){
    return (ZonaOmogenea.lamp_cost
           + ZonaOmogenea.lamp_disposal])
           * calcolaTotaleLampadePerHA(ZonaOmogenea);
}
```

*Funzione 10 Pseudocodice calcolaCostiManutenzionePerHA*

### 4.5.3. Funzione calcolaCostiManutezioneInfrastrutturaPerHA

Descrizione: la funzione calcola i costi di manutenzione dell'intera infrastruttura, per la specificata zona omogenea TO BE.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$costiManutenzioneInfrastruttura_{ZO\ TO\ BE} = costo\_manutenzione\_infrastruttura * numero\_lampade$$

dove

- costo_manutenzione_infrastruttura: costo medio della componente infrastrutturale relativa alla singola lampada
- numero_lampade: numero lampadine nel cluster

Lo pseudocodice della funzione è il seguente (*Funzione 11*):

```
function calcolaCostoManutenzioneInfrastrutturaPerHA(
                                            obj ZonaOmogenea)
{
    return ZonaOmogenea.infrastructure_maintenance_cost
           * calcolaTotaleLampadePerHA(ZonaOmogenea);
}
```

*Funzione 11 Pseudocodice calcolaCostoManutenzioneInfrastrtturaPerHA*

### 4.5.4. Funzione calcolaTotaleLampadePerHA

Descrizione: la funzione calcola il numero totale di lampade di tutti i cluster che compongono una specifica zona omogenea.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$NumeroLampade_{ZO} = \sum_{i}^{\#cluster} numero\_lampade$$

dove

- numero_lampade: numero lampadine nel cluster

Lo pseudocodice della funzione è il seguente (*Funzione 12*):

```
function calcolaTotaleLampadePerHA(obj ZonaOmogenea)
{
    clusters = ZonaOmogenea.clusters;

    nLampadeTot = 0;

    for (i = 0; i < count($clusters); i++) {
        cluster = clusters[$i];
        nLampadeTot += cluster.lamp_num;
    }

    return nLampadeTot;
}
```

*Funzione 12 Pseudocodice calcolaTotaleLampadePerHA*

## 4.6. Calcolo flussi di cassa per zona omogenea (R.6)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare il vettore dei flussi di cassa per la zona omogenea AS-IS scelta dall'utente, tenendo conto della zona omogenea TO-BE associata.

### 4.6.1. Funzione calcoloFlussiDiCassaPerHA

Descrizione: la funzione calcola il vettore dei flussi di cassa annuali, per l'intera durata dell'investimento, per ogni coppia di zone omogenee AS-IS e TO-BE.

Valori validati: $(-\infty, +\infty)$

Il vettore dei flussi di cassa, dall'indice 1 fino all'indice che corrisponde alla durata dell'ammortamento, si ottiene sommando i ricavi provenienti da:

- Risparmio della spesa energetica
- Incentivi statali
- Risparmio delle spese di manutenzione della specifica zona omogenea AS-IS

E sottraendo i costi derivanti da:

- Eventuale mutuo, nel caso di prestito da un Istituto di Credito
- Eventuale canone, nel caso di finanziamento ESCo
- Spese di manutenzione nuova infrastruttura
- Costo gestione nuova zona omogenea TO-BE

Il flusso di cassa in posizione zero, invece, è pari al valore dell'importo investimento moltiplicato per la quota del comune, cambiato di segno.

Lo pseudocodice della funzione è il seguente (*Funzione 13*):

```
function calcoloFlussiDiCassaPerHA() {

    for (i = 0; i < arrayASIS.size; i++){
        //inizializzazione oggetto di output
        result[i] = new Risultato_singolaZO();

        //singola HA ASIS
        haASIS = arrayASIS[$i];
        result[$i].AsisName = haASIS.label_ha;

        //getting TOBE associata
        haTOBE = arrayTOBE.filter(tobe => {
```

```
    tobe.ref_as_is_id_ha == haASIS.id;
};

result[i].TobeName = haTOBE.label_ha;

//inizio calcolo
//calcolo costo investimento
result[i].InvestmentAmount =
       calcolaImportoInvestimentoPerHA(haTOBE)
       * (investment.share_municipality /100);

//Calcola costi/benefici annuali in consumo energetico
calcoloDeltaConsumoEnergeticoPerHAS();
//Calcola costi/benefici annuali in spesa energetica
calcoloDeltaSpesaEnergeticaPerHAS();

//Calcola incentivi statali
result[i].IncentiveRevenue =
    result[i].DeltaEnergyConsumption()
    / investment.tep_kwh
    * investment.tep_value;

//Calcola costi manutenzione
//calcolo flussi e totale costo manutenzione ASIS e TOBE
for(j = 1; j <= durationAmortization; j++) {

    //costo
    result_asis_maintenance_cost[j] =
       calcolaCostiManutezionePerHA(haASIS)
       * ((j % haASIS.lamp_maintenance_interval == 0)
       ? 1 : 0);

    result_tobe_lamp_cost[j] =
       calcolaCostiManutezionePerHA(haTOBE)
       * ((j % haTOBE.lamp_maintenance_interval == 0)
       ? 1 : 0);



    result_tobe_infrastructure_cost[$] =
       calcolaCostoManutenzioneInfrastrutturaPerHA(haTOBE)
       * ((j % haTOBE.lamp_maintenance_interval == 0)
       ? 1 : 0);
}

result[i].asis_maintenance_cost =
       array_sum(result_asis_maintenance_cost);

result[$i].tobe_maintenance_cost =
       array_sum(result_tobe_infrastructure_cost)
       + array_sum(result_tobe_lamp_cost);
```

```
        //Calcola flussi di cassa annuali
        result[i].cash_flow[0]= - result[i].InvestmentAmount();

        for(j = 1; j <= durationAmortization; j++) {
            //costo
            result[i].cash_flow[j]=+
                result[i].DeltaEnergyExpenditure()
                + result_asis_maintenance_cost[j]
                - investment.mortgage_installment
                - investment.fee_esco
                - result_tobe_lamp_cost[j]
                - result_tobe_infrastructure_cost[j]
                - investment.management_cost;
        }

        for(j = 1; j <= investment.incentives_duration; j++){
            result[i].cash_flow[j] +=
                    result[i].IncentiveRevenue();
        }
    }

    return result;
}
```

*Funzione 13 Pseudocodice calcoloFlussiDiCassaPerHA*

## 4.7. Calcolo dei flussi di cassa per impianto (R.7)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare i flussi di cassa per l'intero impianto scelto dall'utente, sommando i flussi di cassa delle singole coppie di zono omogenea che lo compongono.

### 4.7.1. Funzione calcoloFlussiDiCassaPerPlant

Descrizione: la funzione calcola la somma dei flussi di cassa di ogni singola zona omogenea che compone l'impianto.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$FlussiDiCassa_{PLANT} = \sum_{i}^{\#ZO} FlussoDiCassa_{ZO}$$

dove

- FlussoDiCassaZO: il flusso di cassa derivante dai ricavi e costi per ogni coppia di zone omogenee AS-IS e TO-BE.

Lo pseudocodice della funzione è il seguente (*Funzione 14*):

```
function calcoloFlussiDiCassaPerPlant(Obj risultatoSingolaZO)
{

    //calcolo cashflow totale
    for(i=0; i<risultatoSingolaZO.size; i++){
        for(j=0; j<(risultatoSingolaZO[i].cash_flow).size; j++){
            cashFlowTotale[j] +=
                    risultatoSingolaZO[i].cash_flow[j];
        }
    }

    return cashFlowTotale;
}
```

*Funzione 14 Pseudocodice calcoloFlussiDiCassaPerPlant*

## 4.8. Calcolo importo investimento per impianto (R.8)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare l'importo dell'investimento necessario per la procedura di ammodernamento dell'intero impianto scelto dall'utente. Questo calcolo aggrega i singoli importi calcolati per ogni coppia di zone omogenea AS-IS/TO-BE.

### 4.8.1. Funzione calcolaImportoInvestimentoPerPlant

Descrizione: la funzione calcola la somma dei vari importi investimento, calcolati per ogni coppia di zone omogenee AS-IS e TO-BE, in modo da ottenere un importo investimento per l'intero impianto.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$ImportoInvestimento_{PLANT} = \sum_{i}^{\#ZO} ImportoInvestimento_{ZO}$$

dove

- ImportoInvestimentoZO: l'importo dell'investimento necessario per il rifacimento della singola zona omogenea AS-IS.

Lo pseudocodice della funzione è il seguente (*Funzione 15*):

```
function calcolaImportoInvestimentoPerPlant(risultatiSingolaZO)
{
    result = 0;
    for(i = 0; i<risultatiSingolaZO.size; i++){
       result += risultatiSingolaZO[i].importoInvestimento;
    }
    return result;
}
```

*Funzione 15 Pseudocodice calcolaImportoInvestimentoPerPlant*

## 4.9. Calcolo VAN per impianto (R.9)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare il VAN (Valore Attuale Netto) per l'intero impianto scelto dall'utente.

### 4.9.1. Funzione calcoloVANPerImpianto

Descrizione: la funzione calcola il VAN (Valore Attuale Netto) a partire dai flussi di cassa dell'intero impianto.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$VAN = \sum_{i=0}^{n} \frac{CF_i}{(1 + WACC_{comune})^i}$$

dove

- CFi: i-esimo elemento del vettore flusso di cassa dell'impianto.
- WACC_comune: WACC del comune a cui appartiene l'impianto.

Lo pseudocodice della funzione è il seguente (*Funzione 16*):

```
function calcoloVANperImpianto(float[] cashFlow, float wacc)
{

    wacc_absolute = (float)wacc / 100;

    van = 0;
    totVal = count(cashFlow);

    for (i = 0; i < totVal; i++) {
        van += cashFlow[i] / ((1 + wacc_absolute)^i);
    }
    return van;
}
```

*Funzione 16 Pseudocodice calcoloVANPerImpianto*

## 4.10. Calcolo del TIR per impianto (R.10)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare il TIR (Tasso Interno di Rendimento) per l'impianto scelto dall'utente.

### 4.10.1. Funzione calcoloTIRPerImpianto

Descrizione: la funzione calcola il TIR (Tasso Interno di Rendimento) a partire dai flussi di cassa dell'intero impianto.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$\sum_{i=0}^{n} \frac{CF_i}{(1+TIR)^i} = 0$$

dove

- CFi: i-esimo elemento del vettore flusso di cassa dell'impianto. Pseudo codice

Lo pseudocodice della funzione è il seguente (*Funzione 17*):

```
function calcoloTIRperImpianto(float[] cashFlow)
{
    maxIterations = 100;
    tolerance = 0.00001;
    guess = 0.1;

    count = count(cashFlow);

    cashFlow[0] = investment_amount;

    positive = false;
    negative = false;
    for (i = 0; i < count; i++) {
        if (cashFlow[$i] > 0) {
            positive = true;
        } else {
            negative = true;
        }
    }
```

```
    if (!positive || !negative) {
        return null;
    }

    guess = (cashFlow == 0) ? 0.1 : guess;

    for (i = 0; i < maxIterations; i++) {
        npv = 0;
        dnpv = 0.00;

        for (j = 0; j < count; j++) {
            npv += cashFlow[j] / pow(1 + guess, j);
            dnpv -= j * cashFlow[j] / pow(1 + guess, j + 1);
        }

        if(dnpv != 0)
            newGuess = guess - (npv / dnpv);
        else
            return 0;

        if (abs(newGuess - guess) < tolerance) {
            return newGuess;
        }

        guess = newGuess;
    }

    return guess;
}
```

*Funzione 17 Pseudocodice calcoloTIRPerImpianto*

## 4.11. Calcolo del Payback Time (R.11)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare il payback time per l'impianto scelto, tenendo conto dei flussi di cassa cumulativi.

### 4.11.1. Funzione calcoloPayBackTime

Descrizione: la funzione calcola il payback time, ovvero il numero di anni necessari per rientrare nell'investimento iniziale e generare flussi di cassa positivi.

Lo pseudocodice della funzione è il seguente (*Funzione 18*):

```
function calcoloPayBackTime(float[] flussoDiCassa,
                            int durata_ammortamento)
{
    payback_time = 0;
    flusso_cumulativo[0] = flussoDiCassa[0];

    /*
     * calcolo flusso cumulativo
     */
    for (i = 1; i < durata_ammortamento + 1; i++) {
        flusso_cumulativo[i] =
                    flussoDiCassa[i]
                    + flusso_cumulativo[i - 1];
    }

    /*
     * ultimo flusso di cassa cumulativo negativo
     */
    for (i = durata_ammortamento + 1; i > 0; i--) {
        if (exist(flusso_cumulativo[i])) {
            if (flusso_cumulativo[i] < 0) {
                payback_time = i;
                break;
            }
        }
    }

    if(payback_time > 0
       && exist(flussoDiCassa[payback_time + 1])
       && flussoDiCassa[payback_time + 1] !== 0)
```

```
    {
        payback_time +=
                (abs(flusso_cumulativo[payback_time])
                / flussoDiCassa [payback_time + 1]);
     } else{
        payback_time = 0;
 }

    return payback_time;
}
```

*Funzione 18 Pseudocodice calcoloPayBackTime*

Spiegazione pseudocodice: come mostrato nella *Funzione 18*, per calcolare il tempo di Payback è necessario avere i flussi di cassa per ogni anno per tutta la durata dell'ammortamento. Da questi flussi di cassa si calcolano i flussi di cassa cumulativi, ottenuti sommando il flusso di cassa cumulativo attuale al flusso di cassa del singolo anno successivo.

Una volta ottenuto questo nuovo vettore cumulativo si cerca l'indice *i* (se esiste) dell'ultimo flusso di cassa cumulativo negativo, ovvero l'anno in cui l'investimento è ripagato e si iniziano a generare profitti fino alla fine della durata dell'ammortamento. A questo indice si somma poi il rapporto tra i-esimo flusso di cassa cumulativo negativo e il flusso di cassa dell'anno successivo (che sarà sempre maggiore o uguale). In questo modo quindi si ottiene il calcolo del Payback.

## 4.12. Calcolo del canone minimo (R.12)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare il canone minimo, in base alla quota da corrispondere al soggetto privato secondo i parametri dell'investimento scelto, per l'impianto selezionato dall'utente.

### 4.12.1. Funzione calcoloCanoneMinimo

Descrizione: la funzione calcola il canone annuo minimo da corrispondere al soggetto privato.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$canone\_annuo\_min = \frac{canone\_iniziale - \ ammortamento * \frac{imposte}{100}}{1 - \frac{imposte}{100}}$$

dove

- canone_iniziale si calcola come segue

$$canone\_iniziale = \frac{investimento\_ESCo}{(1 - (1 + WACC\_comune)^{-t})/(WACC\_comune)}$$

- t è la durata del progetto
- investimento_ESCo è così calcolato

$$investimento\_ESCo = importo\_investimento * \left(\frac{quota\_ESCo}{100}\right)$$

- ammortamento, infine, si calcola nel seguente modo

$$ammortamento = importo\_investimento/durata\_progetto$$

Lo pseudocodice della funzione è il seguente (*Funzione 19*):

```
function calcoloCanoneMinimo(float importoInvestimento, obj
investment){

    wacc_absolute = (float)investment.wacc / 100;

    investment_ESCO = importoInvestimento * (financedQuote/100);

    canoneIniziale = (investment_ESCO)
                     / ((1-(1+wacc_absolute)^(-feeDuration))
```

```
                        / wacc_absolute);

    ammortamento = investment_ESCO / feeDuration;

    return (canoneIniziale - ammortamento * taxes / 100)
            /(1-taxes / 100);
}
```

*Funzione 19 Pseudocodice calcoloCanoneMinimo*

## 4.13. Calcolo del canone massimo (R.13)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di calcolare il canone massimo, in base alla quota da corrispondere al soggetto privato secondo i parametri dell'investimento scelto, per l'impianto selezionato dall'utente.

### 4.13.1. Funzione calcoloCanoneMassimo

Descrizione: la funzione calcola il canone annuo massimo che si può corrispondere al soggetto privato.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$canone\_annuo\_max = delta\_spesa\_energetica - (ammortamento\_comune - (rata\_mutuo))$$

dove

- investimento_iniziale_comune è così calcolato

$$investimento\_iniziale\_comune = importo\_investimento * \left(\frac{quota\_comune}{100}\right)$$

- ammortamento, infine, si calcola nel seguente modo

$$ammortamento\_comune = \frac{investimento\_iniziale\_comune}{(1-(1+WACC\_comune)^{-t})/(WACC\_comune)}$$

Lo pseudocodice della funzione è il seguente (*Funzione 20*):

```
function calcoloCanoneMassimo(obj plant,
                              float importoInvestimento,
                              obj investment)
{

    $wacc_absolute = (float)investment.wacc / 100;
    investimentoIniziale_comune = importoInvestimento
                                  *(financedQuote/ 100);

    ammortamento_comune = investimentoIniziale_comune
                          / ((1-(1+wacc_absolute)^(-$feeDuration))
                          /wacc_absolute);

    return
         calcoloDeltaSpesaEnergeticaPerImpianto(plant, investment)
```

```
            - (ammortamento_comune
            - (investment.mortgage_installment));
}
```

*Funzione 20 Pseudocodice calcoloCanoneMassimo*

### 4.13.2. Funzione calcoloDeltaSpesaEnergeticaPerImpianto

Descrizione: la funzione calcola la differenza della spesa energetica tra tutte le zone omogenee AS-IS e quelle TO-BE associate.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$DeltaSpesaEnergetica_{PLANT} = \sum_{i}^{\#ZO\ AS\ IS} spesa\_energetica_{AS\ IS} - \sum_{i}^{\#ZO\ TO\ BE} spesa\_energetica_{TO\ BE}$$

dove

- spesa_energetica_asis: la spesa energetica per la zona omogenea AS-IS
- spesa_energetica_tobe: la spesa energetica per la zona omogenea TO-BE

Lo pseudocodice della funzione è il seguente (*Funzione 21*):

```
function calcoloDeltaSpesaEnergeticaPerImpianto(plant, investment)
{

    result = 0;
    for (i = 0; i < arrayASIS.size; i++) {
        //per ogni haASIS
        haASIS = arrayASIS[i];

        //cerco la HA TOBE associata
        haTOBE = arrayTOBE.filter(tobe{
            tobe.ref_as_is_id_ha == haASIS.id;
        });

        //prendo tutte le ZO AS-IS, sommatoria CU e calcolo -
              prendo solo la TO-BE associata e sottraggo
       value = calcoloSpesaEnergeticaPerHaASIS(haASIS)
                - calcoloSpesaEnergeticaPerHaTOBE(haTOBE);

        //sommo per aggregare i risultati
        result += value;
```

```
    }

    //restituisco il risultato
    return result;
}
```

*Funzione 21 Pseudocodice calcoloDeltaSpesaEnergeticaPerImpianto*

## 4.14. Calcolo aggregato dei risultati (R.14)

*Riportiamo la descrizione del requisito funzionale*

Il modulo SAVE deve essere in grado di aggregare tutti i risultati delle precedenti operazioni, in modo da rispettare il formato atteso per poter essere visualizzato correttamente dall'utente.

### 4.14.1. Funzione calcoloPilota

Descrizione: la funzione crea l'oggetto per i vari grafici e le varie tabelle del frontend. Contiene tutti i risultati dei calcoli effettuati nelle successive funzioni, aggregando questi dati secondo una struttura ben definita. Una volta creato l'oggetto, viene effettuata una conversione in formato JSON.

Lo pseudocodice della funzione è il seguente (*Funzione 22*):

```
function calcoloPilota(plant, investment){

    result["municipality"] = plant["label_plant"];

    result["plants"] = calcoloFlussiDiCassaPerHA();

    //calcolo totali
    cashFlowTotale = calcoloFlussiDiCassaPerPlant();

    result["total"]["cash_flow"] = cashFlowTotale;

    result["total"]["investment_amount"] =
         calcolaImportoInvestimentoPerPlant();

    result["total"]["asis_maintenance_cost"] =
          calcolaCostiManutezioneASISPerPlant();

    result["total"]["tobe_maintenance_cost"] =
          calcolaCostiManutezioneTOBEPerPlant();

    result["total"]["incentive_revenue"] =
          calcolaContributoIncentiviPerPlant();
    result["total"]["delta_energy_expenditure"] =
               calcolaDeltaSpesaEnergeticaPerPlant();
    result["total"]["delta_energy_consumption"] =
               calcolaDeltaConsumoEnergeticoPerPlant();

    //calcolo sommatorie parametri dell'investimento
    //Calcola VAN e TIR
    result["financement"]["van"] = calcoloVANperImpianto();
```

```
    result["financement"]["tir"] = calcoloTIRperImpianto();

    //Calcola Payback Time
    result["financement"]["payback_time"]=calcoloPayBackTime();

    //Calcola Canone Minimo
    result["financement"]["fee_min"] = calcoloCanoneMinimo();

    //Calcola Canone Massimo
    result["financement"]["fee_max"] = calcoloCanoneMassimo();

    return result;
}
```

*Funzione 22 Pseudocodice calcoloPilota*

Un esempio dell'oggetto restituito da questa funzione è:


```
{
    "plants" : {
        "municipality": "Roma",
        "plants": [{
            "asis_name": "Garibaldi",
            "tobe_name": "Garibaldi to be",
            "investment_amount": 11382,
            "asis_maintenance_cost": 2321.54,
            "tobe_maintenance_cost": 5980,
            "incentive_revenue": 164.7,
            "delta_energy_expenditure": 1673.44,
            "delta_energy_consumption": 8807.55,
            "cash_flow": [-11382, 1838.14, 1838.14, 1838.14,
     1838.14, 1838.14, 1838.14, 1838.14,
     1838.14, 1838.14, 1838.14, 1838.14,
     1838.14, 1838.14, 1838.14, 1838.14, 1
     838.14, 1838.14, 1838.14, 1838.14, 1838.14]
        }, {
            "asis_name": "Verdi",
            "tobe_name": "Verdi to be",
            "investment_amount": 17500,
            "asis_maintenance_cost": 4464.5,
            "tobe_maintenance_cost": 11500,
            "incentive_revenue": 351.17,
            "delta_energy_expenditure": 3568,
            "delta_energy_consumption": 18778.97,
            "cash_flow": [-17500, 3919.18, 3919.18, 3919.18,
     3919.18, 3919.18, 3919.18, 3919.18,
     3919.18, 3919.18, 3919.18, 3919.18,
     3919.18, 3919.18, 3919.18, 3919.18,
     3919.18, 3919.18, 3919.18, 3919.18, 3919.18]
        }],
```

```
        "total": {
        "investment_amount": 28882,
        "asis_maintenance_cost": 6786.04,
        "tobe_maintenance_cost": 17480,
        "incentive_revenue": 515.87,
        "delta_energy_expenditure": 5241.4400000000005,
        "delta_energy_consumption": 27586.52,
        "cash_flow": [-28882, 5757.32, 5757.32, 5757.32,
    5757.32, 5757.32, 5757.32, 5757.32,
    5757.32, 5757.32, 5757.32, 5757.32,
    5757.32, 5757.32, 5757.32, 5757.32,
    5757.32, 5757.32, 5757.32, 5757.32,5757.32]
        },
        "financement": {
            "van": 88580.08,
            "tir": 23,
            "payback_time": 3.9,
            "fee_min": 1295.55,
            "fee_max": 4877.52
        }
    }
}
```


### 4.14.2. Funzione calcolaCostiManutezioneASISPerPlant

Descrizione: la funzione calcola i costi di manutenzione dell'intero impianto per quanto riguarda le zone omogenee AS-IS.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$CostiManutenzione_{PLANT} = \sum_{i}^{\#ZO\ AS\ IS} costi\_manutenzione$$

dove

- costi_manutenzione: costo per la manutenzione della singola zona omogenea AS-IS dell'impianto analizzato.

Lo pseudocodice della funzione è il seguente (*Funzione 23*):

```
function calcolaCostiManutenzioneASISPerPlant(
                                        obj risultatiSingolaZO)
{
    costoManutenzione = 0;
    for(i = 0; i<risultatiSingolaZO.size; i++){
     costoManutenzione +=
               risultatiSingolaZO[i].asisMaintenanceCost;
    }
    return costoManutenzione;
}
```

*Funzione 23 Pseudocodice calcolaCostiManutenzioneASISPerPlant*

### 4.14.3. Funzione calcolaCostiManutezioneTOBEPerPlant

Descrizione: la funzione calcola i costi di manutenzione dell'intero impianto per quanto riguarda le zone omogenee TO-BE.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$CostiManutenzione_{PLANT} = \sum_{i}^{\#ZO\ TO\ BE} costi\_manutenzione$$

dove

- costi_manutenzione: costo per la manutenzione della singola zona omogenea TO-BE dell'impianto analizzato.

Lo pseudocodice della funzione è il seguente (*Funzione 24*):

```
function calcolaCostiManutenzioneTOBEPerPlant(obj
risultatiSingolaZO){
    costoManutenzione = 0;
    for(i = 0; i<count($risultatiSingolaZO); i++){
        costoManutenzione +=
                    risultatiSingolaZO[$i].tobeMaintenanceCost;
    }
    return costoManutenzione;
}
```

*Funzione 24 Pseudocodice calcolaCostiManutenzioneTOBEPerPlant*

### 4.14.4. Funzione calcolaContributoIncentiviPerPlant

Descrizione: la funzione calcola i ricavi derivanti dai contributi incentivi dell'intero impianto a seguito dell'ammodernamento.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$ContributoIncentivi_{PLANT} = \sum_{i}^{\#ZO} contributi\_incentivi$$

dove

- contributi_incentivi: ricavi derivanti dal processo di aggiornamento da zona omogenea AS-IS a zona omogenea TO-BE.

Lo pseudocodice della funzione è il seguente (*Funzione 25*):

```
function calcolaContributoIncentiviPerPlant(
                                        obj risultatiSingolaZO)
{
    contributoIncentivi = 0;
    for(i = 0; i<count($risultatiSingolaZO); i++){
        contributoIncentivi +=
                        risultatiSingolaZO[$i].incentiveRevenue;
    }
    return contributoIncentivi;
}
```

*Funzione 25 Pseudocodice calcolaContributoIncentiviPerPlant*

### 4.14.5. Funzione calcolaDeltaSpesaEnergeticaPerPlant

Descrizione: la funzione calcola la differenza della spesa energetica tra tutte le zone omogenee AS-IS e quelle TO-BE.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$DeltaSpesaEnergetica_{PLANT} = \sum_{i}^{\#ZO\ AS\ IS} spesa\_energetica_{AS\ IS} - \sum_{i}^{\#ZO\ TO\ BE} spesa\_energetica_{TO\ BE}$$

dove

- spesa_energetica_asis: la spesa energetica per la zona omogenea AS-IS
- spesa_energetica_tobe: la spesa energetica per la zona omogenea TO-BE

Lo pseudocodice della funzione è il seguente (*Funzione 26*):

```
function calcolaDeltaSpesaEnergeticaPerPlant(
                                              obj risultatiSingolaZO)
{
    deltaSpesaEnergetica = 0;
    for(i = 0; i<count($risultatiSingolaZO); i++){
        deltaSpesaEnergetica +=
                    risultatiSingolaZO[$i].deltaEnergyExpenditure;
    }
    return deltaSpesaEnergetica;
}
```

*Funzione 26 Pseudocodice calcolaDeltaSpesaEnergeticaPerPlant*

### 4.14.6. Funzione calcolaDeltaConsumoEnergeticoPerPlant

Descrizione: la funzione calcola la differenza del consumo energetico tra tutte le zone omogenee AS-IS e quelle TO-BE.

Valori validati: $(-\infty, +\infty)$

Formula matematica corrispondente:

$$DeltaConsumoEnergetico_{PLANT} = \sum_{i}^{\#ZO\ AS\ IS} consumo\_energetico_{AS\ IS} - \sum_{i}^{\#ZO\ TO\ BE} consumo\_energetico_{TO\ BE}$$

dove

- consumo_energetico_asis: il consumo energetico della i-esima zona omogenea AS-IS
- consumo_energetica_tobe: il consumo energetico della i-esima zona omogenea TO-BE

Lo pseudocodice della funzione è il seguente (*Funzione 27*):

```
function calcolaDeltaConsumoEnergeticoPerPlant(
                                              obj risultatiSingolaZO)
{
    deltaConsumoEnergetico = 0;
```

```
    for(i = 0; i<count($risultatiSingolaZO); i++){
        deltaConsumoEnergetico +=
                    risultatiSingolaZO[$i].deltaEnergyConsumption;
    }
    return deltaConsumoEnergetico;
}
```

*Funzione 27 Pseudocodice calcolaDeltaConsumoEnergeticoPerPlant*

# 5 Validazione e testing del modulo SAVE

Il modulo SAVE sviluppato è stato sottoposto ad un processo di validazione dei dati in output basandosi sia su dati impiantistici reali ma simulati, sia su dati effettivamente raccolti da un impianto di illuminazione esistente.

L'obiettivo della validazione è verificarne la completezza ed il rispetto dei requisiti software richiesti dalla specifica, pertanto, non necessariamente la validazione del software comporta la ricerca di errori che bloccano l'esecuzione delle operazioni (cosiddette eccezioni)

Inoltre, il processo di validazione non è monolitico: in caso vengano riscontrati risultati forvianti si può agire tempestivamente in modo da correggere il comportamento non desiderato.

Il testing del modulo è invece più direzionato alla ricerca di errori e/o eccezioni vere e proprie: si verifica il compimento con successo dell'operazione, verificando anche tramite dei vincoli sul risultato ottenuto.

Le due operazioni avvengono in questo caso in maniera separata.

## 5.1. Validazione mediante dati simulati

La validazione mediante dati simulati è stata effettuata confrontando i dati ottenuti richiamando l'API tramite Postman, con i rispettivi risultati attesi calcolati singolarmente. La suite di testing con dei risultati salvati è consultabile sul repository al path "/doc/SAVE.postman_collection.json"

### 5.1.1. Simulazione mediante dataset fornito dal correlatore

Il dataset è strutturato con 2 zone AS-IS, a cui sono associate rispettivamente una zona riqualificata TO-BE 1 a 1. Per ognuna delle zone omogenee, è associato rispettivamente un cluster 1 a 1 tipizzato in base al tipo della zona omogenea (se la zona omogenea è AS-IS, allora avrà associato solo cluster AS-IS). Questo dataset è richiamabile utilizzando nelle richieste un plantID = 9 ed un investmentID = 2.

In particolare, per questo dataset si evidenziano i seguenti valori

- mortgage_installment = 400
- duration_amortization = 30

All'interno della suite di testing sono presenti le seguenti richieste con i relativi risultati:

- Funzionalità "showCalcoloImpianto" richiamata tramite API di indirizzo: http://127.0.0.1:8000/CalcoloImpianto?plantId=9&investmentId=2

- Funzionalità “VanETir” richiamata tramite API di indirizzo: http://127.0.0.1:8000/VanETir?plantId=9&investmentId=2&wacc[]=3&wacc[]=5&wacc[]=7&amortization_duration[]=12&amortization_duration[]=24&amortization_duration[]=36
- Funzionalità “Payback” richiamata tramite API di indirizzo: http://127.0.0.1:8000/PayBack?plantId=9&investmentId=2&min_energy_cost=0.1&max_energy_cost=0.28
- Funzionalità altreModalità richiamata tramite API di indirizzo: http://127.0.0.1:8000/calcolaAltreModalita?plantId=9&investmentId=2&min_fee_duration=12&max_fee_duration=33&taxes=35.78&financed_quote=75.90

### 5.1.2. Simulazione mediante database generato da noi studenti

In maniera simile al dataset fornito dal correlatore, abbiamo strutturato il nostro dataset di validazione. Questo dataset è richiamabile utilizzando nelle richieste un plantID = 1 ed un investmentID = 1.

In particolare, per questo dataset si evidenziano i seguenti valori

- mortgage_installment = 0
- duration_Amortization = 12

All’interno della suite di testing sono presenti le seguenti richieste con i relativi risultati:

- Funzionalità “showCalcoloImpianto” richiamata tramite API di indirizzo: http://127.0.0.1:8000/CalcoloImpianto?plantId=1&investmentId=1
- Funzionalità “VanETir” richiamata tramite API di indirizzo: http://127.0.0.1:8000/VanETir?plantId=1&investmentId=1&wacc[]=3&wacc[]=5&wacc[]=7&amortization_duration[]=12&amortization_duration[]=24&amortization_duration[]=36
- Funzionalità “Payback” richiamata tramite API di indirizzo: http://127.0.0.1:8000/PayBack?plantId=1&investmentId=1&min_energy_cost=0.1&max_energy_cost=0.28
- Funzionalità altreModalità richiamata tramite API di indirizzo: http://127.0.0.1:8000/calcolaAltreModalita?plantId=1&investmentId=1&min_fee_duration=12&max_fee_duration=33&taxes=35.78&financed_quote=75.90

## 5.2. Validazione delle funzionalità economiche

In questa sezione validiamo singolarmente i calcoli dei VAN, TIR e Payback a partire da un flusso di cassa comune e confrontando il risultato della funzionalità implementata sul SAVE con una funzionalità disponibile pubblicamente (come, ad esempio, le formule messe a disposizione su Excel). In questo modo ci assicuriamo che i risultati siano validati anche su un ambiente slegato dal modulo SAVE e che, applicato a quest'ultimo, restituisca calcoli realistici

### 5.2.1. Validazione VAN e TIR

Per la prima simulazione abbiamo utilizzato il database di validazione generato da noi studenti, variando il campo "mortgage_installment" dell'entità Investimenti a 300€ e impostando il campo "duration_ammortization" a 30 anni. Abbiamo quindi preso i flussi di cassa totali generati dal modulo SAVE, e abbiamo calcolato VAR e TIR prima con la funzionalità di calcolo di Excel per una serie di flussi di cassa (funzione …) e successivamente abbiamo confrontato il risultato con quello ottenuto dal modulo SAVE.

| Flussi di cassa con mortgage = 300€ e 30 anni di ammortamento |
|---|
| 3846 |
| -519,9958958 |
| -519,9958958 |
| 1720,004104 |
| 170,0041042 |
| -519,9958958 |
| 1244,8279 |
| -3395,1721 |
| -305,1721 |
| 1244,8279 |
| -995,1721 |
| -995,1721 |
| 1528,8279 |
| -995,1721 |
| -3395,1721 |
| 1244,8279 |
| -305,1721 |
| -995,1721 |
| 1244,8279 |
| -995,1721 |
| -305,1721 |
| -1155,1721 |

| |
|---|
| -995,1721 |
| -995,1721 |
| 1528,8279 |
| -995,1721 |
| -995,1721 |
| 1244,8279 |
| -2705,1721 |
| -995,1721 |
| 1244,8279 |

*Tabella 6 Flussi di cassa per VAN e TIR (Excel)*

| TIR | VAN |
|---|---|
| | |
| 6% | -9.939,80 € |

*Tabella 7 Validazione VAN e TIR (Excel)*

Di seguito il risultato ottenuto dal modulo SAVE:


```
{
    "municipality": "Plant 1",
    "plants": [
        {
            "asis_name": "HAS ASIS 1",
            "tobe_name": "HAS TOBE 3",
            "investment_amount": 2890,
            "asis_maintenance_cost": 22400,
            "tobe_maintenance_cost": 9600,
            "incentive_revenue": 359.98197285081415,
            "delta_energy_expenditure": 3.6575,
            "delta_energy_consumption": 19250,
            "cash_flow": […]
        },
        {
            "asis_name": "HAS ASIS 2",
            "tobe_name": "HAS TOBE 4",
            "investment_amount": 956,
            "asis_maintenance_cost": 4830,
            "tobe_maintenance_cost": 812,
            "incentive_revenue": 115.19423131226051,
            "delta_energy_expenditure": 1.1704,
            "delta_energy_consumption": 6160,
            "cash_flow": […]
        }
    ],
    "total": {
        "cash_flow": [
            -3846,
```

```
            -519.9958958369253,
            -519.9958958369253,
            1720.0041041630743,
            170.00410416307466,
            -519.9958958369253,
            1244.8278999999998,
            -3395.1721,
            -305.1721,
            1244.8278999999998,
            -995.1721,
            -995.1721,
            1528.8278999999998,
            -995.1721,
            -3395.1721,
            1244.8278999999998,
            -305.1721,
            -995.1721,
            1244.8278999999998,
            -995.1721,
            -305.1721,
            -1155.1721000000002,
            -995.1721,
            -995.1721,
            1528.8278999999998,
            -995.1721,
            -995.1721,
            1244.8278999999998,
            -2705.1721,
            -995.1721,
            1244.8278999999998
        ],
        "investment_amount": 3846,
        "asis_maintenance_cost": 27230,
        "tobe_maintenance_cost": 10412,
        "incentive_revenue": 475.17620416307466,
        "delta_energy_expenditure": 4.8279000000000005,
        "delta_energy_consumption": 25410
    },
    "financement": {
        "van": -10237.998,
        "tir": 0.0632690773590805,
        "payback_time": 0,
        "fee_min": 0,
        "fee_max": 304.8279
    }
}
```


Dove si evidenzia che, a parità di flussi di cassa

$$VAN_{validato} = 0{,}06 \approx VAN_{SAVE} = 0{,}063269$$
$$TIR_{validato} = -9939{,}80 \approx VAN_{SAVE} = -10237{,}998$$

Per la seconda simulazione abbiamo continuato ad utilizzare il database di validazione generato da noi studenti, variando il campo “mortgage_installment” dell’entità Investimenti a 1000€ e impostando il campo “duration_ammortization” a 12 anni. Abbiamo come prima preso i flussi di cassa totali generati dal modulo SAVE, e abbiamo calcolato VAR e TIR prima con la funzionalità di calcolo di Excel per una serie di flussi di cassa e successivamente confrontato il risultato con quello ottenuto dal modulo SAVE.

Di seguito il risultato ottenuto dal modulo SAVE:


```
{
    "municipality": "Plant 1",
    "plants": [
        {
            "asis_name": "HAS ASIS 1",
            "tobe_name": "HAS TOBE 3",
            "investment_amount": 2890,
            "asis_maintenance_cost": 8960,
            "tobe_maintenance_cost": 2400,
            "incentive_revenue": 359.98197285081415,
            "delta_energy_expenditure": 3.6575,
            "delta_energy_consumption": 19250,
            "cash_flow": […]
        },
        {
            "asis_name": "HAS ASIS 2",
            "tobe_name": "HAS TOBE 4",
            "investment_amount": 956,
            "asis_maintenance_cost": 2070,
            "tobe_maintenance_cost": 406,
            "incentive_revenue": 115.19423131226051,
            "delta_energy_expenditure": 1.1704,
            "delta_energy_consumption": 6160,
            "cash_flow": […]
        }
    ],
    "total": {
        "cash_flow": [
            -3846,
            -1919.9958958369255,
            -1919.9958958369255,
            320.0041041630743,
            -1229.9958958369252,
            -1919.9958958369255,
            -155.17210000000023,
            -4795.1721,
            -1705.1721,
            -155.17210000000023,
            -2395.1721,
            -2395.1721,
            128.82789999999977
```

```
        ],
        "investment_amount": 3846,
        "asis_maintenance_cost": 11030,
        "tobe_maintenance_cost": 2806,
        "incentive_revenue": 475.17620416307466,
        "delta_energy_expenditure": 4.8279000000000005,
        "delta_energy_consumption": 25410
    },
    "financement": {
        "van": -18892.118,
        "tir": 0.37789901231518863,
        "payback_time": 0,
        "fee_min": 0,
        "fee_max": 1004.8279
    }
}
```


Dove si evidenzia che, a parità di flussi di cassa

$$VAN_{validato} = 0.38 \approx VAN_{SAVE} = 0.377899$$

$$TIR_{validato} = -18341{,}86 \approx VAN_{SAVE} = -18892{,}118$$

Quindi il calcolo del TIR e del VAN risulta validato

### 5.2.2. Validazione PayBack

Utilizziamo sempre il database di test generato da noi studenti, stavolta con "mortgage_installment" uguale a 0 e "project_amortization" uguale a 30 anni.

Per la validazione del tempo di PayBack, ci assicuriamo che il momento temporale da cui il flusso di cassa cumulativo diventa positivo, sia il valore restituito.

| Flussi di cassa (30 anni e mortgage = 0€) | Flussi di cassa cumulativi |
|---|---|
| -3846 | -3846 |
| 80 | -3.766 |
| 80 | -3.686 |
| 2.320 | -1.366 |
| 770 | -596 |
| 80 | -516 |
| 1.845 | 1.329 |
| -2.795 | -1.466 |
| 295 | -1.171 |
| 1.845 | 673 |
| -395 | 278 |
| -395 | -117 |

| | |
|---|---|
| 2.129 | 2.012 |
| -395 | 1.617 |
| -2.795 | -1.179 |
| 1.845 | 666 |
| 295 | 961 |
| -395 | 566 |
| 1.845 | 2.411 |
| -395 | 2.016 |
| 295 | 2.310 |
| -555 | 1.755 |
| -395 | 1.360 |
| -395 | 965 |
| 2.129 | 3.094 |
| -395 | 2.699 |
| -395 | 2.303 |
| 1.845 | 4.148 |
| -2.105 | 2.043 |
| -395 | 1.648 |
| 1.845 | 3.493 |

Da questi flussi di cassa abbiamo calcolato i seguenti risultati:

| | |
|---|---|
| Indice ultimo flusso di cassa cumulativo negativo | 14 |
| Valore anno successivo | 0,638828 |
| | |
| PayBack Totale (anni) | 14,63883 |

*Tabella 8 Validazione Payback (Excel)*

E di seguito il risultato ottenuto dal modulo SAVE


```
{
  "municipality": "Plant 1",
  "plants": [
    {
      "asis_name": "HAS ASIS 1",
      "tobe_name": "HAS TOBE 3",
      "investment_amount": 2890,
      "asis_maintenance_cost": 22400,
      "tobe_maintenance_cost": 9600,
      "incentive_revenue": 359.98197285081415,
      "delta_energy_expenditure": 3.6575,
      "delta_energy_consumption": 19250,
      "cash_flow": [...]
```

```
    },
    {
      "asis_name": "HAS ASIS 2",
      "tobe_name": "HAS TOBE 4",
      "investment_amount": 956,
      "asis_maintenance_cost": 4830,
      "tobe_maintenance_cost": 812,
      "incentive_revenue": 115.19423131226051,
      "delta_energy_expenditure": 1.1704,
      "delta_energy_consumption": 6160,
      "cash_flow": [...]
    }
  ],
  "total": {
    "cash_flow": [
      -3846,
      80.00410416307466,
      80.00410416307466,
      2320.0041041630743,
      770.0041041630747,
      80.00410416307466,
      1844.8278999999998,
      -2795.1721,
      294.8279,
      1844.8278999999998,
      -395.1721,
      -395.1721,
      2128.8278999999998,
      -395.1721,
      -2795.1721,
      1844.8278999999998,
      294.8279,
      -395.1721,
      1844.8278999999998,
      -395.1721,
      294.8279,
      -555.1721000000002,
      -395.1721,
      -395.1721,
      2128.8278999999998,
      -395.1721,
      -395.1721,
      1844.8278999999998,
      -2105.1721,
      -395.1721,
      1844.8278999999998
    ],
    "investment_amount": 3846,
    "asis_maintenance_cost": 27230,
    "tobe_maintenance_cost": 10412,
    "incentive_revenue": 475.17620416307466,
    "delta_energy_expenditure": 4.8279000000000005,
```

```
    "delta_energy_consumption": 25410
  },
  "financement": {
    "van": 1522.266,
    "tir": 0,
    "payback_time": 14.63882835856105,
    "fee_min": 0,
    "fee_max": 4.8279000000000005
  }
}
```


Dove si evidenzia che, a parità di flussi di cassa

$$PayBack_{Validato} \sim 14anni = PayBack_{SAVE} \sim 14anni$$

Il calcolo del PayBack è stato validato, infatti il calcolo rispetta il risultato atteso.

## 5.3. Testing di unità della funzionalità

Le funzionalità sono state testate con dei feature test consultabili nel repository e descritte brevemente nella *Tabella 12*:

| Requisito - Nome test | Vincolo | Risultato | Correzione |
|---|---|---|---|
| R.1 - test_calcoloImportoInvestimentoPerHA() | Restituisce zero se la zona non ha cluster associati, altrimenti un numero positivo | OK | - |
| R.2 - test_calcoloDeltaConsumoEnergeticoPerHAS() | Restituisce zero se non ci sono cluster associati alle zone omogenee, altrimenti non ci sono particolari vincoli | OK | - |
| R.3 - test_calcoloDeltaSpesaEnergeticaPerHAS() | Nessun errore con parametri nulli,<br>Risultato non nullo se parametri non nulli | ErrorException : Trying to access array offset on value of type null | Applicato un controllo != null sui parametri in input della funzione |
| R.4 - test_calcoloIncentiviStatali() | Nessun errore con parametri nulli,<br>Controllo divisione per 0,<br>Risultato non nullo se parametri non nulli | Error : Call to a member function setIncentiveRevenue() on null | Applicato un controllo != null sui parametri in input della funzione |
| R.5 - test_calcoloCostiManutenzione() | Nessun errore con parametri nulli,<br>Risultato non nullo se parametri non nulli | ErrorException : Undefined variable $result_asis_maintenance_cost | Applicato un controllo != null sui parametri in input della funzione e aggiunto un return value di default |

*Tabella 9 Relazione requisito - test*

# 6 Conclusione

## 6.1. Stati di avanzamento (matrice di tracciabilità dei requisiti)

Il modulo SAVE alla fine degli sviluppi richiesti risulta completamente implementato e pronto per un'integrazione all'interno della piattaforma PELL dalla quale ne deriva come requisito funzionale.

Per un resoconto delle funzionalità implementate si fa riferimento alla seguente matrice di tracciabilità dei requisiti:

| Requisito – Use case | Punto nel codice | Test | Punto nel codice |
|---|---|---|---|
| R.1 (calcoloImportoInvestimentoPerHA) - Use Case 4: Nuova simulazione | CalculateHelper, linea 30 | test_ calcoloImportoInvestimentoPerHA | SaveTest, linea 20 |
| R.2 (calcoloDeltaConsumoEnergeticoPerHAS) - Use Case 4: Nuova simulazione | CalculateHelper, linea 120 | test_ calcoloDeltaConsumoEnergeticoPerHAS | SaveTest, linea 36 |
| R.3 (calcoloDeltaSpesaEnergeticaPerHAS) - Use Case 4: Nuova simulazione | CalculateHelper, linea 195 | test_ calcoloDeltaSpesaEnergeticaPerHAS | SaveTest, linea 64 |
| R.4 (calcoloIncentiviStatali) - Use Case 4: Nuova simulazione | CalculateHelper, linea 254 | test_calcoloIncentiviStatali | SaveTest, linea 96 |
| R.5 (calcoloCostiDiManutenzione) - Use Case 4: Nuova simulazione | CalculateHelper, linea 253 | test_calcoloCostiDiManutenzione | SaveTest linea 116 |

*Tabella 10 Matrice di tracciabilità requisito - test nel codice sorgente*

## 6.2. Considerazioni finali e sviluppi futuri

Il modulo SAVE, alla fine degli sviluppi trattati, viene presentato come un modulo distinto, utilizzabile previo adattamento all'interno della piattaforma di ENEA, ma solo per la parte riguardante l'estrazione delle informazioni necessarie ai calcoli dal database.

La generazione dei requisiti funzionali specifici per il modulo SAVE ha permesso di scendere più nel dettaglio e quindi, di approfondire le funzionalità richieste da ENEA, partendo dalla specifica funzionale esistente riguardante l'intera piattaforma PELL.

È inoltre corredato ad un'attenta analisi sui risultati attesi, generata sia da un lavoro di testing ancora migliorabile, sia da un lavoro di validazione sui parametri economici e la loro definizione.

Degli sviluppi che si prospettano in futuro sono:

- Test di unità: può essere considerato l'implementazione di test di unità con l'obiettivo di applicare una non-regressione nei futuri cicli di implementazione e/o correzione dei difetti
- Confronto con altri strumenti di calcolo: nei test di validazione può rendersi necessario un confronto delle funzionalità con nuovi strumenti di calcolo con l'obiettivo di migliorare i risultati di validazione
- Implementazione degli Smart service: Per gli altri benefici i quali la relativa valorizzazione può avvenire solo a posteriori, ossia una volta identificati i dettagli degli interventi di riqualificazione dell'infrastruttura di PI in ottica smart service e dell'area presso cui insistono gli interventi stessi, viene lasciata la possibilità al comune di inserire tali benefici senza fornire una formula specifica.

# 7 Bibliografia

[0] https://www.pell.enea.it
[1] P. Bocciarelli, A. D'Ambrogio, B. Gentili, M. Facondini, L. Tiburzi. "Estensione del Modulo SAVE. Modello Economico-Finanziario e Revisione della Specifica Tecnica". Report RdS/PTR2021/XXX
[2] "The PHP Framework for Web Artisans". https://laravel.com/

[3] "Design pattern". https://it.wikipedia.org/wiki/Design_pattern

[4] DB Wagner - The Mathematica Journal, 1995 - yaroslavvb.com "Dynamic programming" http://yaroslavvb.com/papers/wagner-dynamic.pdf

[5] "Programmazione dinamica". https://it.wikipedia.org/wiki/Programmazione_dinamica