PPS-22-energy-mng

Antonio Iannotta: antonio.iannotta2@studio.unibo.it
Andrea Catani: andrea.catani5@studio.unibo.it
Demetrio Andriani: demetrio.andriani@studio.unibo.it
Carlo Di Raddo: carlo.diraddo@studio.unibo.it

Introduzione

Questo report è relativo allo sviluppo dell'applicazione Energy Management.
L'applicazione consente il monitoraggio dei consumi energetici e del costo ad essi associato(energia idrica, energia termica ed energia elettrica) da parte degli utenti, siano essi utenti private o aziende.
Mediante questa applicazione l'utente può monitorare i consumi e il conseguente impatto economico che ne deriva sia in misura personale e sia per una specifica città o una specifica regione.
L'applicazione consente anche di effettuare delle previsioni sulla base dell'andamento storico dei consumi e dei costi in modo da fornire all'utente una panoramica più ampia di quella che potrà essere la situazione futura.

Processo di sviluppo adottato

Il processo di sviluppo adottato per questo sistema è guidato dal framework SCRUM per la gestione di progetti. In particolar modo, con riferimento a SCRUM, l'organizzazione è basata su sprint settimanali ripetuti.

- **Sprint settimanali:**
    - Sprint planning: Lunedì mattina
    - Durata Sprint: 20 ore
    - Daily Sprint
    - Sprint Review: Venerdì pomeriggio

Supporto a questo processo di sviluppo è fornito dal software Trello, utilizzato per l'organizzazione strutturata del lavoro.

Scelta strumenti di build/test/continuous integration

Strumenti di build: **sbt**
Strumenti di test: **ScalaTest**
Strumenti di continuous integration: **GitHub**

Requisiti

Requisiti business

ID	REQUISITO
1.1	Il software Energy Management consente di monitorare l'andamento dei consumi energetici ed i costi connessi ai consumi di un singolo utente, di una specifica città, di una specifica regione, dell'intero paese.
1.2	Il software Energy Management consente di effettuare previsioni sull'andamento futuro di consumi energetici e dei costi associati ai consumi di un singolo utente, di una specifica città, di una specifica regione, dell'intero paese.
1.3	Il software Energy Management consente di effettuare il confronto tra i consumi energetici e i costi associati ai consumi tra due città o tra due regioni
1.4	Il software Energy Management fa fronte alla difficoltà che gli utenti hanno di avere una visione completa ed approfondita dei propri consumi energetici
1.5	Il software Energy Management può essere utilizzato da utenti privati per il monitoraggio dei propri consumi energetici domestici e dei costi associati ad essi
1.6	Il software Energy Management può essere utilizzato da aziende per il monitoraggio dei consumi energetici dell'azienda stessa e dei costi associati ad essi.
1.7	Il software Energy Management fornisce dati sui consumi energetici e sui costi associati ad essi con un alto livello di affidabilità

Requisiti utente

ID	REQUISITO
2.1	L'utente può essere un utente privato o un'azienda
2.2	L'utente è presente nel sistema con un identificativo univoco
2.3	L'utente deve risiedere in una specifica città e in una specifica regione. La regione e la città devono essere esistenti e la città deve essere presente nella regione
2.4	L'utente visualizzerà i propri consumi energetici e i costi ad essi associati
2.5.0	L'utente visualizzerà i consumi energetici relativi alla città in cui risiede ed i

	costi ad essi associati
2.5.1	Nel caso di utente privato l'utente visualizzerà una media dei consumi energetici e dei costi ad essi associati di tutti gli utenti privati presenti nella città in cui risiede.
2.5.2	Nel caso di azienda l'utente visualizzerà una media dei consumi energetici e dei costi ad essi associati di tutte le aziende presenti nella città in cui risiede.
2.6.0	L'utente visualizzerà i consumi energetici relativi alla regione in cui risiede ed i costi ad essi associati
2.6.1	Nel caso di utente privato l'utente visualizzerà una media dei consumi energetici e dei costi ad essi associati di tutti gli utenti privati presenti nella regione in cui risiede.
2.6.2	Nel caso di utente privato l'utente visualizzerà una media dei consumi energetici e dei costi ad essi associati di tutti gli utenti privati presenti nella regione in cui risiede.
2.7	L'utente può monitorare i consumi di energia termica, idrici ed energia termica relativi ad un'unica linea intestata.

Requisiti funzionali

ID	**REQUISITO**
3.1	L'utente sceglie se registrarsi al sistema come utente privato o come azienda
3.1.1	L'utente inserisce i seguenti dati: ● UserID ● Password ● Tipologia Utente ● Regione ● Città
3.1.2	Il sistema verifica che: ● L'userID selezionato dall'utente non sia già presente all'interno del sistema ● La regione sia sia valida ● La città sia valida ● La città si trovi nella regione selezionata
3.1.3	Nel caso in cui i dati fossero corretti il sistema registra l'utente
3.1.4	Nel caso in cui una delle precedenti verifiche non andasse a buon fine la registrazione fallisce e il sistema riporta un messaggio specificando la causa dell'errore
3.2	Per accedere al sistema l'utente inserisce UserID e Password

3.2.1	Il sistema verifica la correttezza dei dati inseriti e: ● nel caso di dati corretti l'utente ottiene accesso al sistema ● nel caso di dati non corretti il sistema genera un messaggio d'errore specificando la natura dell'errore
3.3	Il sistema deve consentire la visualizzazione dei dati richiesti dall'utente
3.3.1	Il sistema deve recuperare i dati sui consumi energetici con annessi costi relativi ad un utente.
3.3.2	Il sistema deve mostrare i dati all'utente relativi alle diverse tipologie di consumi e ai costi connessi a ciascun consumo.
3.3.3	Il sistema deve consentire all'utente di visualizzare i dati relativi ai consumi della propria città (come specificato in 2.5.0).
3.3.4	Il sistema deve consentire all'utente di visualizzare i dati relativi ai consumi della propria regione (come specificato in 2.6.0).
3.3.5	Il sistema deve consentire una visualizzazione dei dati su base mensile, raggruppando e calcolando la media di consumi per città/regione inerenti allo stesso mese.
3.3.6	L'utente può scegliere una città o una regione diversa dalla propria per la quale visualizzare i dati.
3.3.7	Il sistema deve mostrare all'utente consumi e costi relativi alle diverse tipologie di consumi, con i costi associati ad essi, rispetto alla città/regione scelta dall'utente.
3.3.8	Il sistema consente di visualizzare previsioni sull'andamento dei consumi
3.3.8.1	Il sistema consente all'utente di scegliere la tipologia di consumo e l'anno su cui effettuare la previsione. Il sistema sulla base di questa scelta e sulla base dei consumi storici dell'utente mostra le previsioni per l'anno selezionato.
3.3.8.2	Il sistema consente all'utente di scegliere la città, la tipologia di consumo e l'anno su cui effettuare la previsione. Il sistema sulla base di questa scelta e sulla base dei consumi storici per quella specifica città mostra all'utente le previsioni per l'anno selezionato.
3.3.8.3	Il sistema consente all'utente di scegliere la regione, la tipologia di consumo e l'anno su cui effettuare la previsione. Il sistema sulla base di questa scelta e sulla base dei consumi storici per quella specifica regione mostra all'utente le previsioni per l'anno selezionato.
3.3.9	Il sistema deve consentire all'utente di visualizzare dati, tipologia di consumi, e previsioni relative ad un certo anno dell'intero paese.
3.4	Il sistema deve presentare i consumi di ciascun utente in forma compatta, di modo che sia possibile tenere traccia della tipologia di consumi, del consumo effettivo, del costo relativo al consumo, la data di riferimento, la regione, la città e l'UserID a cui fa riferimento

3.5	Il sistema utilizza i dati relativi alla registrazione di ciascun utente per generare in maniera automatica i consumi relativi all'utente stesso.

Requisiti non funzionali

ID	REQUISITO
4.1	Il sistema deve memorizzare i dati in maniera consistente
4.2	Il sistema deve ridurre al minimo le situazioni in cui i dati visualizzati dall'utente non corrispondano effettivamente ai dati visualizzati o generati per l'utente stesso.
4.3	Il sistema deve generare i dati relativi ai consumi con intervalli temporali ben definiti.
4.4	Il sistema deve essere in grado di gestire molteplici utenti.
4.5	I dati devono essere memorizzati e presentati agli utenti in un formato leggibile e con le opportune unità di misura.

Requisiti di implementazione

ID	REQUISITO
5.1	Il sistema deve essere in grado di girare su di una generica JVM, quindi con una forte indipendenza rispetto alla macchina sottostante.
5.2	Per la memorizzazione il sistema deve utilizzare un database NoSQL dal momento che risulta più appropriata la lavorazione di dati di questa tipologia con un database basato sui documenti
5.3	Per l'implementazione si utilizza l'IDE IntelliJ di modo da poter sfruttare tutte le peculiarità di un ambiente integrato.

Design architetturale

Architettura generale del sistema

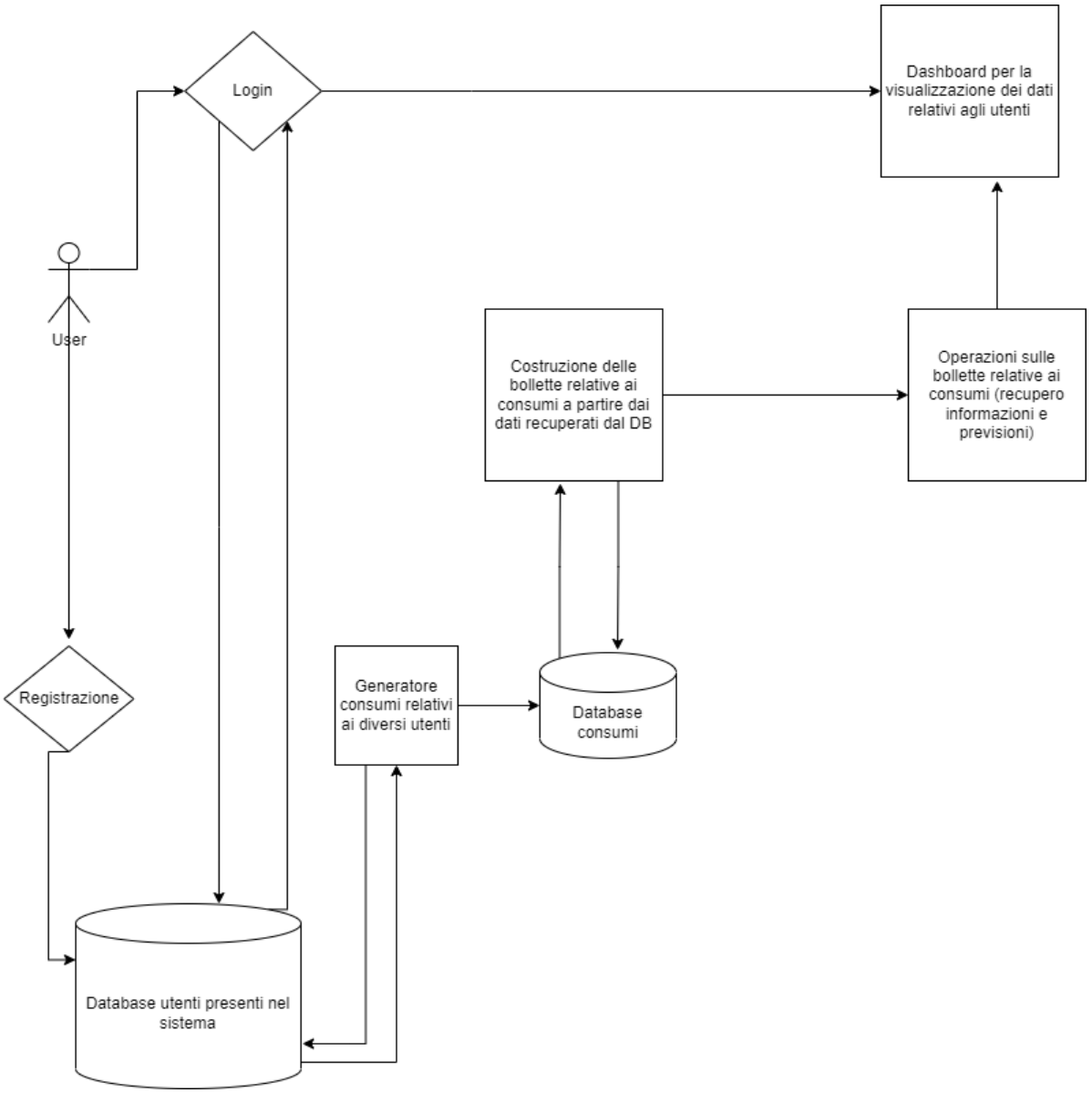
Elemento centrale dell'architettura di Energy Management è l'utente (sia esso utente privato piuttosto che azienda).
Dall'utente partono due flussi operazionali differenti:

- Registrazione: con la registrazione l'utente memorizza le sue informazioni all'interno del database. Successivamente queste informazioni sono prese da un generatore di consumi che genera i consumi, come specificato nel requisito funzionale 3.5. Questi consumi generati vengono memorizzati all'interno del DB e saranno poi recuperati da un generatore di bollette relative che, partendo dai consumi recuperati dal database, costruisce le bollette in modo opportuno di modo che queste possano essere gestite dal componente predisposto alle operazioni. Tale componente recupera le bollette costruite e sulla base di queste esegue delle operazioni, queste operazioni saranno poi utilizzate per visualizzare i dati specifici richiesti dall'utente dopo il login.
- Login: con il login l'utente ottiene accesso al sistema ottenendo la possibilità di scegliere quali informazioni visualizzare, relative a quali consumi a quale città. Con il login l'utente ottiene la possibilità di operare ed interfacciarsi con le varie operazioni fornite dal sistema.

Operazioni centrali del sistema sono i seguenti:

- Costruzione delle bollette: questa operazione è cruciale dal momento che recupera in maniera continuativa dati dal database relativo ai consumi e costruisce le relative bollette. La costruzione di tali bollette è un'operazione complessa dal punto di visto di interfacciamento con il DB in quanto risulta fondamentale per il soddisfacimento del requisito non funzionale 4.2.
- Operazioni sulle bollette costruite: queste operazioni risultano cruciali per poter permettere all'utente di visualizzare i dati richiesti. Si tratta quindi di operazioni di computazione che espongono i risultati in modo che questi siano visibili all'utente.

Di seguito è riportata l'architettura generale del sistema:



Componenti del sistema

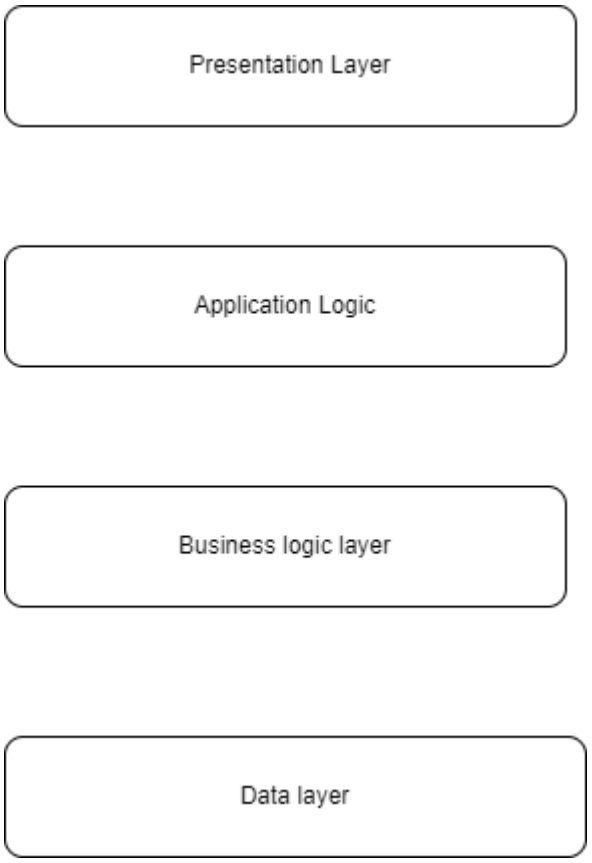
- **Registrazione**: la registrazione riceve in input i dati dell'utente quando questo vuole registrarsi al sistema. Esegue una serie di controlli di validità sui dati forniti in input come specificato nei requisiti 3.1.22 e in caso di successo registra l'utente al sistema memorizzando i dati all'interno del database degli utenti.
- **Login**: Il login riceve in input lo UserID e la password ed esegue i controlli sul database degli utenti per verificare se lo specifico userID è presente. Nel caso in cui questo sia presente allora verifica la password e nel caso di corrispondenza consente all'utente di accedere al sistema.
- **Generatore consumi**: Il generatore dei consumi recupera i dati memorizzati all'interno del database degli utenti con l'obiettivo di generare, sulla base di questi dati recuperati, dei consumi random. E' importante, con l'obiettivo di garantire consistenza e nell'ottica di rispettare il requisito non funzionale 4.2 che la richiesta al DB avvenga con una frequenza sufficientemente elevata. Il generatore, per ogni

utente recuperato ad intervalli regolari genera dei consumi random che vengono successivamente memorizzati all'interno del database dei consumi.

- **Database**
    - **Database utenti**: il database utenti memorizza le informazioni relative agli utenti che hanno effettuato la registrazione al sistema.
    - **Database consumi**: il database dei consumi memorizza i consumi generati dal generatore dei consumi relativi agli utenti che sono presenti all'interno del sistema.
- **Costruttore bollette**: il costruttore delle bollette recupera i dati memorizzati all'interno del database dei consumi e li utilizza per generare le relative bollette. Le bollette hanno una propria struttura ed è opportuno che siano generate in modo conforme alla struttura stessa nell'ottica di poter essere utilizzate in modo ottimale dal componente predisposto alle operazioni. Con l'obiettivo di ridurre al minimo problemi di inconsistenza non generando bollette per consumi effettivamente presenti e nell'ottica di soddisfare quindi il requisito non funzionale 4.2 l'interazione tra il costruttore delle bollette e il database dei consumi avviene con una frequenza elevata.
- **Operazioni bollette**: le operazioni vengono effettuate sulle bollette costruite a partire dai dati recuperati dal database dei consumi.
- **Dashboard**: è il componente che consente all'utente di interagire con il sistema. Mediante questo componente l'utente riesce ad esprimere quali sono i dati che intende visualizzare e quali sono le previsioni con relativi parametri che intende eseguire. Questo componente sfrutta, quindi, le operazioni derivanti dal componente **Operazioni bollette**

Pattern

Energy Management è un'applicazione con un forte flusso informativo, in cui è possibile individuare diversi livelli. Per questa ragione il pattern architetturale scelto è il **Layered Pattern**.



La scelta di questo pattern è stata dettata dalla possibilità fornita dallo stesso di poter posizionare le varie componenti precedentemente elencate in un livello specifico.

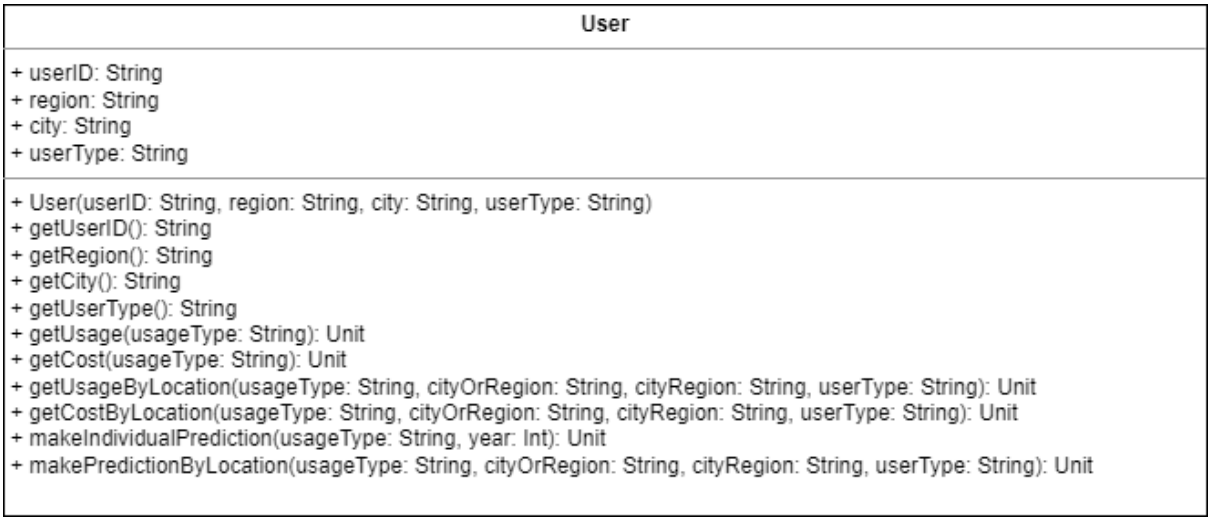
- **Data layer**: in questo livello troviamo i seguenti componenti:
    - Database utenti
    - Database consumi
    - Registrazione
    - Login
    - Generatore consumi
- **Business logic layer**: in questo livello è posizionato il costruttore delle bollette.
- **Application logic layer**: in questo livello è posizionato il componente predisposto alle operazioni sulle bollette.
- **Presentation layer**: in questo livello è posizionato il componente Dashboard che consente l'interazione con l'utente per le specifiche operazioni.

Design di dettaglio

Si presenta ora il design di dettaglio relativo alle varie componenti architetturali precedentemente presentate:

Utente

L'utente è l'elemento centrale dell'applicazione. Infatti è l'utente che effettua le scelte relative alla tipologia di dati da visualizzare e a quali previsioni effettuare all'interno della Dashboard. All'interno del sistema sono presenti due tipologie di utenti: l'utente privato e l'azienda. Tuttavia questa distinzione viene effettuata all'atto di registrazione. Al momento del login l'utente viene creato con i dati rilevanti recuperati dal DB in corrispondenza del proprio username e viene istanziato con un campo che riporta la tipologia di utente a cui ci si riferisce. Quest'ultimo punto è estremamente importante per riportare i consumi relativi ad una certa località con riferimento agli utenti privati o alle aziende. E' per questa ragione che is è scelto di rappresentare l'utente per mezzo di una **classe**.
Il seguente diagramma UML mostra e formalizza quanto detto:



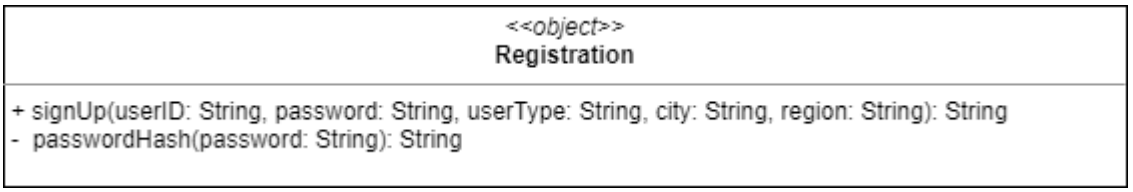
L'utente si trova nella Dashboard, in particolar modo, come spiegato successivamente, la Dashboard è istanziata ricevendo in ingresso un Utente.
Dal diagramma riportato è possibile notare come un utente venga istanziato tenendo conto del suo userID (univoco all'interno del sistema) la propria città, la propria regione e la tipologia di utente. In particolar modo la tipologia di utente risulta importante sia per effettuare le previsioni relative ad una certa località geografica (le previsioni sull'andamento dei consumi privati o aziendali relativi ad una certa città o regione) e sia per mostrare l'andamento dei consumi in una certa località geografica (andamento dei consumi privati o aziendali relativi ad una certa città o regione)

Registrazione

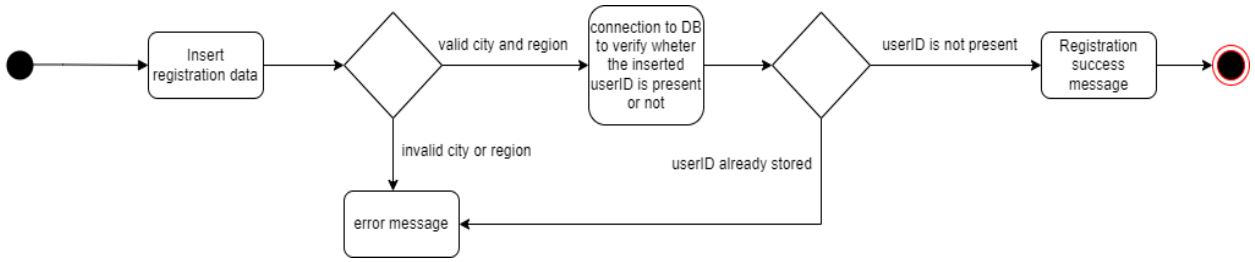
La registrazione è la prima operazione che un utente esegue quando vuole interfacciarsi all'applicazione.
L'operazione di registrazione ha uno stretto legame con il database degli utenti già presentato nella fase di architettura, dal momento che è necessario verificare l'unicità dello UserID (che è il modo con cui un utente viene verificato all'interno del sistema).
Il componente registrazione esegue solo due operazioni:

- Ricava l'hash code relativo alla password ricevuta in input
- Invia i dati presentati in input con l'hash code della password al database.

In virtù del fatto che si tratta di un componente che esegue solo un'operazione verso l'esterno si è deciso di realizzarlo come **object**.
Il seguente diagramma UML mostra e formalizza quando detto:

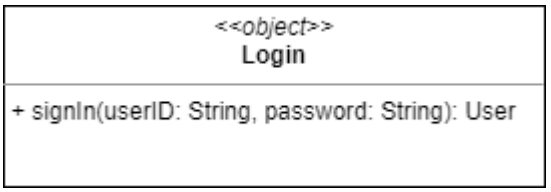


Come mostrato da questo diagramma il metodo register riporta una stringa contenente il risultato dell'operazione di registrazione. Nel caso di errore riscontrato in violazione delle verifiche esplicitate nel requisito 3.1.2 la stringa riporta anche la motivazione dell'errore.
Dal momento che tale componente prevede l'interazione con il database degli utenti per la verifica dell'unicità dello userID di seguito è riportato il corrispondente diagramma delle attività:

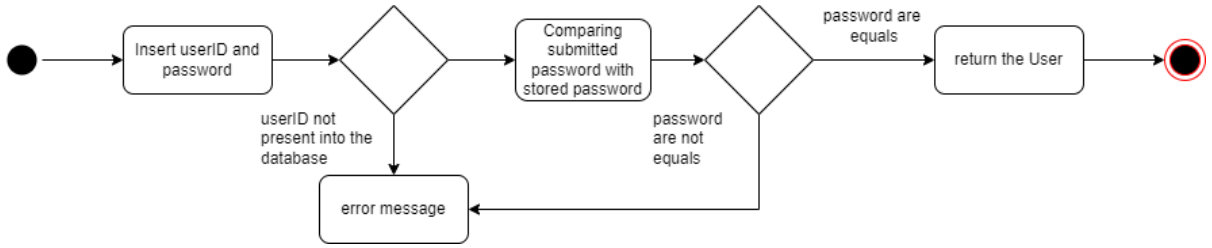


Login

L'operazione di login consente all'utente di guadagnare accesso al sistema verificando prima che l'userID inserito dall'utente sia presente all'interno del database degli utenti e successivamente verificando che la password memorizzata per lo specifico userID sia effettivamente quella inserita dall'utente.
Allo scopo di fare questo la password recuperata dal database viene hashata e successivamente viene confrontata con l'hash code relativo alla password inserita dall'utente.
Nel caso in cui le password combaciassero dal database degli utenti vengono recuperati tutti i dati utili per ritornare l'oggetto utente associato a quello userID.
Dal momento che tale componente deve eseguire un'unica operazione è stato scelto di utilizzare un **object** per incapsularne il comportamento.
Il seguente diagramma UML mostra e formalizza quanto detto:



Dal momento che l'operazione di login prevede l'interazione con il database e diverse verifiche il seguente diagramma delle attività mostra l'operatività del componente.

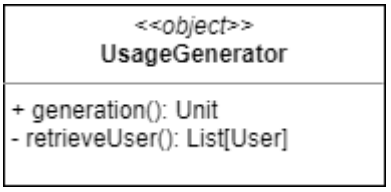


Generatore consumi

Questo componente è pensato per essere un componente sempre attivo in grado di generare dei consumi con cadenza periodica. Per poter generare i consumi il generatore recupera la lista di utenti dal database degli utenti. Ad ogni aggiornamento di questa lista vengono generati i consumi per tutti gli utenti e vengono inseriti all'interno del database dei consumi. Il flusso di esecuzione di questo componente è il seguente:

- recupero degli utenti presenti all'interno del database degli utenti
- generazione dei consumi a partire dalla lista e invio dei consumi al database dei consumi. In particolar modo, si attende il completamente del recupero dei dati e poi si effettua la generazione.
- dopo il termine della generazione dei consumi si attende una definita quantità di tempo e si effettua la richiesta al database per il recupero degli utenti.

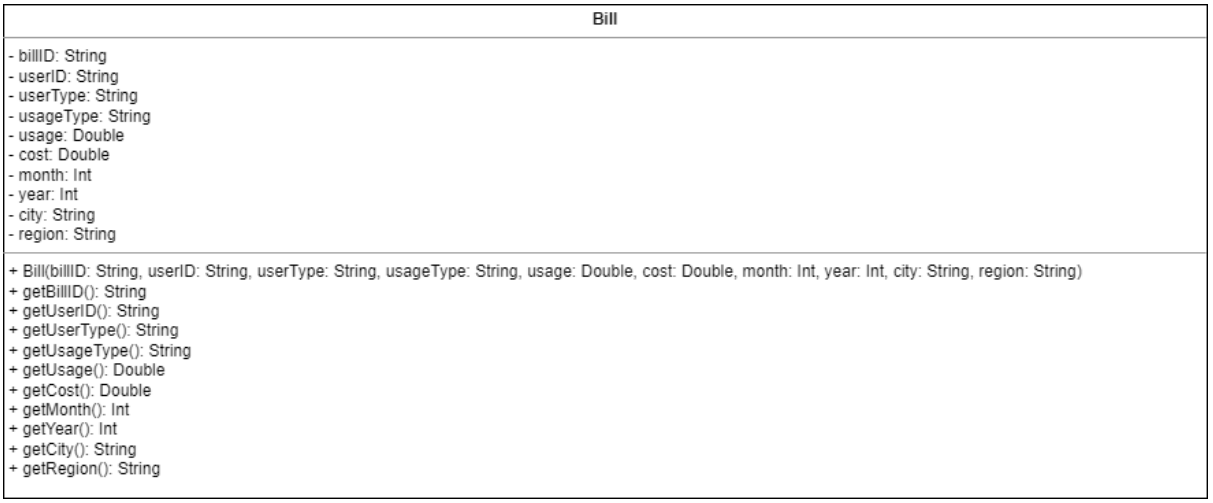
Risulta evidente da quanto detto che le operazioni eseguite da questo componente si eseguono ciclicamente.
La generazione avviene per ogni tipologia di consumo.
Ogni generazione, oltre ai dati tipici dei consumi quali consumo effettivo e costo ad esso associato, considera anche i dati recuperati dal database degli utenti come userID, città, regione.
Ulteriore elemento importante è assegnare ad ogni consumo generato un proprio id. Questo, come successivamente descritto, è importante per la costruzione della lista di bollette realizzata con i dati recuperati dal database dei consumi.
Questo componente espone verso l'esterno una sola operazione, la generazione, la quale prevede contestualmente l'invio dei dati al database dei consumi.
Per questa ragione si è deciso di realizzarlo come **object**.



Aspetto importante per quanto riguarda la generazione dei consumi e che questi devono essere generati con un mese ed un anno definiti.

Costruttore bollette

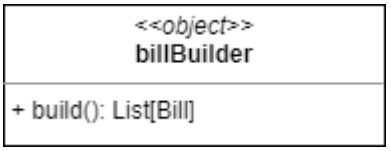
Il costruttore delle bollette ritorna la lista delle bollette ricavata interrogando in un certo istante il database dei consumi.
Il primo elemento da considerare riguarda la frequenza con cui questo componente interroga il database con l'obiettivo di costruire la lista.
Tuttavia, trattandosi di un costruttore di bollette è necessario considerare anche l'entità bolletta.
Questa entità si è deciso di modellarla come una **classe**, di modo da rendere più robusta e coerente la costruzione della lista da parte del costruttore.
Il seguente diagramma UML riporta e formalizza la classe relativa alle bollette.



Una volta definita l'entità bolletta è opportuno definire il costruttore delle bollette.
Come già riportato tale costruttore interroga periodicamente il database dei consumi, recupera i dati in esso contenuti e sulla base di questi dati costruisce le bollette.
Tutte le bollette vengono salvate, man mano che la costruzione procede, all'interno di una lista.
In virtù della periodicità con cui questo componente opera è da ritenersi un componente sempre attivo.
Questo passaggio è estremamente importante dal momento che il sistema prevede un binding molto forte tra questa lista ed il componente che si occupa di effettuare le operazioni sulle bollette.
Il flusso di attività che il costruttore delle bollette esegue è quindi il seguente:

- recupero dei dati contenuti all'interno del database dei consumi
- costruzione della lista delle bollette a partire dai dati recuperati

Un aspetto importante da rimarcare è che questo è un flusso di attività ciclico. In particolar modo il componente attende di aver recuperato i dati, costruisce le bollette e dopo la costruzione esegue un'altra interrogazione al database.
Dal momento che il costruttore di bollette esegue un'unica operazione si è deciso di modellarlo come un **object**.
Il seguente diagramma UML mostra e formalizza il costruttore delle bollette:



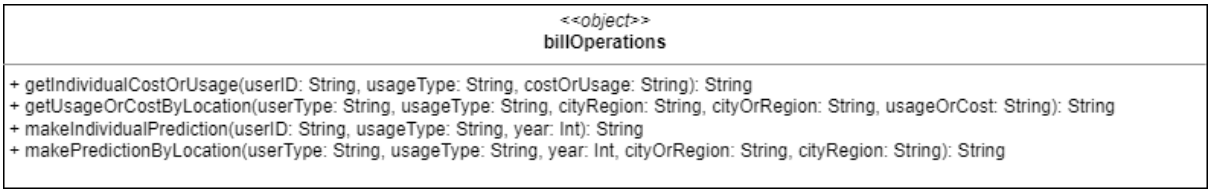
Operazioni bollette

Il componente preposto alle operazioni sulle bollette si trova nel mezzo tra la Dashboard (che richiama le operazioni su uno specifico utente) e il costruttore delle bollette che fornisce la lista delle bollette aggiornate su cui poter eseguire le operazioni.
Le operazioni che vengono effettuate da tale componente si suddividono in due categorie:

- Operazioni di semplice ritorno e filtraggio rispetto a parametri richiesti:
    - utilizzo su base mensile per una specifica utenza
    - costo su base mensile per una specifica utenza
    - utilizzo su base mensile per una specifica località
    - costo su base mensile per una specifica località
- Operazioni di previsioni sulla base di determinate specifiche.
    - Previsioni individuali per una specifica utenza relativamente ad uno specifico anno
    - Previsioni per una specifica località relativamente ad uno specifico anno

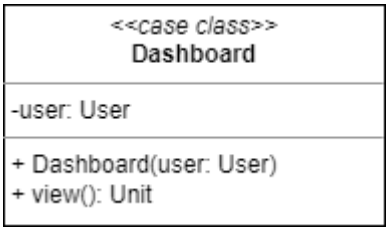
Le operazioni definite da questo componente trovano applicazione in quelli che sono i metodi che ciascun utente può richiamare per avere una visualizzazione delle informazioni richieste.
Dal momento che questo è un metodo che espone semplicemente operazioni si è deciso di realizzarlo come **object**.

Il seguente diagramma UML mostra e formalizza quanto detto.



Dashboard

La Dashboard è il punto in cui il sistema viene utilizzato dall'utente. In particolar modo si occupa di ricevere gli input dall'utente che ha effettuato il login e di visualizzare i risultati delle richieste ottenute.
La Dashboard si è scelto di modellarla come un **case class** che prende in ingresso un utente ed espone come unico metodo quello che consente di visualizzare i dati sulla base delle operazioni richieste dall'utente.
All'interno del metodo esposto la dashboard consente l'interazione con l'utente sfruttando i metodi definiti all'interno della classe User.
Il seguente diagramma UML mostra e formalizza quanto detto:



Implementazione

Antonio Iannotta

Implementazione dei seguenti componenti:

- ***case class*** Dashboard
- ***object*** billOperations

billOperations

Per l'implementazione dell'object billOperations si è utilizzato un approccio KISS e DRY. Di seguito vengono riportati i metodi implementati:

- getIndividualCostOrUsage: questo metodo restituisce i consumi o i costi di un utente relativamente ad una certa utenza
- getUsageOrCostByLocation: questo metodo restituisce i consumi o i costi mensili relativi ad una certa utenza, ad una certa regione o città. In particolar modo per ogni mese viene riportata la media dei consumi relativi ad una certa utenza e ad una specifica città.
- makeIndividualPrediction: questo metodo restituisce la previsione relativa ad uno specifico consumo o costo individuale, relativo ad un certo anno e ad una certa utenza.
- makePredictionByLocation: questo metodo restituisce la previsione relativa ad un certo consumo, per una specifica tipologia di utenti, con riferimento ad un certo anno per quanto riguarda una specifica utenza.

Dopo aver elencato i metodi che sono stati implementati parte rilevante è l'algoritmo utilizzato per le previsioni, riportato nel seguito:

- Si recuperano le medie dei consumi e costi relativi ad una specifica utenza per tutti gli anni.
- Si ipotizza che per un numero di anni successivi pari al numero di anni per cui la media è stata calcolata i consumi e i costi rimangano costanti.
- Passato questo periodo ogni anno i consumi e costi aumentano o diminuiscono di un certo valore random.

Dashboard

Andrea Catani

Carlo Di Raddo

Demetrio Andriani

Retrospettiva