E-Cycle

**No table of contents entries found.**

# Ruoli dei Membri

I membri del gruppo di sviluppo di E-Cycle sono

* Bellotta Lorenzo,
* Bonanomi Elisa,
* Pinizzotto Filippo,
* Testa Maurizio.

La progettazione astratta previa l’implementazione del database è stata effettuata da tutti i membri del gruppo.

L’implementazione del modello logico è stata effettuata da Bellotta Lorenzo e Testa Maurizio.

La realizzazione dei model, dei repository e dei service è stata effettuata dinamicamente in base alle esigenze dei developer in relazione all’applicazione: non ci sono metodi inutilizzati.

L’implementazione dell’algoritmo di *matching* automatizzato e della pagina di home è stata effettuata da Bellotta Lorenzo e Testa Maurizio.

L’implementazione di login, registrazione, modifica del profilo, inserimento della richiesta e visualizzazione dei dettagli di una richiesta è stata effettuata Bellotta Lorenzo.

L’implementazione di inserimento dell’offerta e visualizzazione dei dettagli di un’offerta è stata effettuata dal Testa Maurizio.

L’implementazione dello styling CSS è stata effettuata da Bonanomi Elisa, Pinizzotto Filippo e Testa Maurizio.

Alla stesura della relazione hanno partecipato tutti i membri del gruppo.

# Introduzione

## Il Problema

Oggi chi vuole disfarsi di un bene ha due strade: l’accumulo, sia esso privato o in discariche, e il *riutilizzo*.

Con *riutilizzo* si intendono pratiche tre pratiche principali:

* up-cycling, con cui il bene assume una funzione diversa da quella designata;
* riciclaggio, con cui il bene viene scomposto in parti rientranti nel ciclo produttivo;
* e rivendita, con cui il bene viene ceduto intatto a un altro soggetto, che poi né farà ciò che desidera.

L’attività di riutilizzo più semplice ed efficace è senza dubbio la rivendita: il venditore non compie alcuna lavorazione sul bene e si limita a guadagnare, il compratore ottiene l’oggetto desiderato pronto all’uso che più desidera, e la collettività trae beneficio dallo scambio con la riduzione dei rifiuti.

Ancora oggi però vendere un bene inutilizzato risulta complicato. I marketplace sono dispersivi e privi di alcuna logica strutturale: la domanda e l’offerta faticano ad incontrarsi. Manca uno spazio dove si possa cercare non un prodotto specifico, ma un bene con determinate caratteristiche.

## La Soluzione

E-Cycle nasce per colmare questo vuoto: lo scopo dell’app è facilitare la rivendita dei beni inutilizzati dando loro una nuova vita.

Chi possiede beni vacanti li può offrire e chi ne ha bisogno li può cercare. Entrambi agiscono in base alle caratteristiche reali del bene, non solo in base al suo nome commerciale.

Con pochi click una richiesta può trovare una risposta, e un oggetto dimenticato un nuovo contesto d’uso.

E-Cycle è più di un semplice marketplace: è un’intelligenza collettiva applicata ai beni materiali.

# L’Applicazione

## Sotto un Punto di Vista Teorico

### Glossario

Per comprendere il funzionamento dell’applicazione, è necessario essere messi a conoscenza di alcuni termini specifici del progetto.

* *Utente*: qualsiasi soggetto registrato sull’app, sia esso privato o azienda.
* *Caratteristiche* (o Insieme Caratteristico): l’insieme delle proprietà che definiscono un bene.
* *Marca* e *Modello*: rispettivamente, il produttore e la denominazione specifica dell’asset.
* *Categoria*: macro-classe di beni a cui un asset appartiene (es. “dispositivo elettronico”, “veicolo industriale”, “mobile d’ufficio”).
* *Natura*: il nome comune con cui si identifica l’oggetto (es. “laptop”, “forno industriale”, “trapano”).
* *Interazione*: o una richiesta o un offerta.
* *Richiesta*: la formulazione, da parte di un utente, di un bisogno relativo a uno o più beni dotati di caratteristiche specifiche. Una richiesta può riguardare beni omogenei (identici tra loro) o eterogenei (diversi ma compatibili con usi distinti), e può quindi essere costituita da più richieste singole.
* *Richiesta singola*: la richiesta, da parte di un utente, di un singolo bene con determinate caratteristiche.
* *Offerta*: la disponibilità, da parte di un utente, a cedere uno o più beni dotati di caratteristiche specifiche. Un’offerta può riguardare beni omogenei o eterogenei, e può quindi essere costituita da più offerte singole.
* *Offerta singola*: l’offerta, da parte di un utente, di un singolo bene con determinate caratteristiche.
* *Matching*: il processo attraverso cui l’applicazione confronta una richiesta singola con le offerte singole disponibili, identificando quelle compatibili per caratteristiche e prezzo.

*Negoziazione*: la fase in cui un utente visualizza un’offerta singola in relazione ad una sua richiesta singola e può decidere di accettare l’acquisto o rifiutarlo.

### Modello Concettuale

A graph paper with blue lines and dots

AI-generated content may be incorrect.

#### Le Entità

* *user*, caratterizzata da *username, name, surname, email, ts\_passwordUpdate, password, address* (a sua volta caratterizzato da *state, region, province, city, street e civic*)e *ID*;
* *interaction*, caratterizzata da *title, ts\_creation, isOffer* e *ID*;
* *sing\_request*, carratterizzata da *max\_price, ts\_deletion* e *ID*;
* *sing\_offer*, caratterizzata da *price, picture\_path, description, expiration, ts\_deletion* e *ID*;
* *characteristics* (degli asset), caratterizzata da *main\_color, function, quality, prod\_year, batch* e *ID*;
* *model*, *brand*, *category* e *nature*, ciascuna delle quali caratterizzate da un *ID*;
* e *negotiation*, caratterizzata da *ts\_creation, ts\_closure, wasAccepted* e *ID*.

#### Le Relazioni

* tra *user* e *interaction*, di tipo n:1, con vincoli di cardinalità rispettivamente di “(0, n)” e “(1, 1)”;
* tra *interaction* e *sing\_request*, di tipo n:1, con vincoli di cardinalità rispettivamente di “(0, n)” e “(1, 1)”;
* tra *sing\_request* e *characteristics*, di tipo 1:n, con vincoli di cardinalità rispettivamente di “(1, 1)” e “(0, n)”;
* tra *sing\_request* e *negotiation*, di tipo n:1, con vincoli di cardinalità rispettivamente di “(0, n)” e “(1, 1)”;
* tra *interaction* e *sing\_offer*, di tipo n:1, con vincoli di cardinalità rispettivamente di “(0, n)” e “(1, 1)”;
* tra *sing\_offer* e *characteristics*, di tipo 1:n, con vincoli di cardinalità rispettivamente di “(1, 1)” e “(0, n)”;
* tra *sing\_offer* e *negotiation*, di tipo n:1, con vincoli di cardinalità rispettivamente di “(0, n)” e “(0, 1)”;
* tra *model* e *characteristics*, di tipo n:1, con vincoli di cardinalità rispettivamente di “(1, n)” e “(1, 1)”;
* tra *brand* e *model*, di tipo n:1, con vincoli di cardinalità rispettivamente di “(1, n)” e “(1, 1)”;

#### I Vincoli

Oltre ai vincoli di cardinalità, mantenuti intrinsecamente dalla struttura del database, in E-Cycle esistono vincoli ulteriori che vengono imposti dal *modus operandi* dell’applicazione.

Questi vincoli aggiuntivi sono:

* nella tabella *users* solamente *name* e *surname* sono attributi di inserimento facoltativo;
* nella tabella *users* non possono esistere due o più tuple con attributo *username* uguale;
* nella tabella *users* non possono esistere due o più tuple con attributo *email* uguale;
* nella tabella *interactions* tutti gli attributi devono essere diversi da NULL;
* nella tabella *sing\_requests* solamente *max\_price* e *ts\_deletion* possono essere NULL;
* nella tabella *sing\_offers* solamente *description*, *expiration* e *ts\_deletion* possono essere NULL;
* quando si inseriscono tuple in *sing\_offers* queste non possono avere *expiration* inferiore rispetto alla data corrente;
* nella tabella *negotiations* solamente gli attributi *ts\_closure* e *wasAccepted* possono essere NULL;
* al momento di creazione di una nuova tupla nella tabella *negotiations*, la richiesta singola che verrà associata ad essa non può avere *ts\_deletion* diverso da NULL e non può essere associata ad un’altra tupla di *negotiations* in cui l’attributo *ts\_closure* equivale a NULL;
* al momento di creazione di una nuova tupla nella tabella *negotiations*, l’offerta singola che verrà associata ad essa non può avere *expiration* pari alla data del giorno corrente, non può avere *ts\_deletion* diverso da NULL, e non può essere già associata ad un’altra tupla di *negotiations* in cui l’attributo *ts\_closure* equivale a NULL;
* tutte le tuple di *negotiations* di cui il campo *ts\_creation* sottratto al timestamp corrente produce una differenza pari o superiore a 24 ore non possono avere *ts\_closure* diverso da NULL;
* tutte le tuple di *negotiations* di cui il campo *closure* è diverso da NULL non possono avere il proprio campo *ts\_closure* pari a NULL;
* nelle tabelle *offers*, *requests*, *natures*, *models*, *characteristics*, *categories* e *brands* tutti gli attributi non possono essere NULL;
* tutte le tuple di *characteristics* non possono avere il campo *prod\_year* inferiore a 0 o maggiore rispetto all’anno corrente.

### Modello Logico

A graph paper with writing on it

AI-generated content may be incorrect.

### Funzionamento dell’Applicazione

#### La Registrazione e l’Accesso

Ogni utente, privato o azienda, accede alla piattaforma tramite un account. Dopo la registrazione, può interagire con l’applicazione in qualità di offerente (chi cede beni) o richiedente (chi li cerca).

#### La Pubblicazione di un’Offerta

Un utente che intende vendere beni inutilizzati procede come segue:

* accede all’area di creazione di un’offerta;
* dà un nome all’offerta;
* specifica le caratteristiche dei beni che desidera cedere: marca, modello, anno di produzione, lotto, categoria, natura, stato qualitativo, colore principale e funzione.;
* indica la quantità che vuole cedere di tale bene;
* assegna un prezzo unitario a ciascuno di essi;
* e salva l’offerta, che si tradurrà in più offerte singole, entranti nel sistema e rimanenti disponibile al matching fino alla loro scadenza.

#### La Pubblicazione di una Richiesta

Un utente con un bisogno definito procede come segue:

* accede all’area di creazione di una richiesta;
* descrive l’insieme caratteristico dei beni desiderato;
* indica la quantità necessaria dei beni;
* opzionalmente, inserisce un prezzo massimo disposto a pagare per ciascun asset;

e salva la richiesta, che si tradurrà in più richieste singole, entranti nel sistema e rimanenti disponibili al matching fino alla loro scadenza.

#### Il *Matching* Automatizzato

Il sistema esegue periodicamente un confronto tra richieste singole e offerte singole. Un’offerta singola viene considerata compatibile se:

* il suo insieme caratteristico coincide o è contenuto in quello richiesto;
* il prezzo richiesto è inferiore o uguale a quello massimo indicato nella richiesta (se presente);
* o l’offerta non è scaduta.

Quando vengono trovate corrispondenze:

* al richiedente viene mostrata la richiesta singola con l’offerta singola compatibile;
* si apre così una negoziazione: il richiedente ha 24 ore per accettare o rifiutare l’acquisto;
* e se accetta, l’offerta singola viene considerata assegnata. Se rifiuta o non risponde entro il termine, la trattazione decade.

#### Il Ciclo Vita delle Richieste e delle Offerte

* Un’offerta singola ha una durata temporale limitata: dopo la data di scadenza non viene più considerata nei matching.
* Una richiesta singola resta attiva finché non viene soddisfatta o cancellata dall’utente richiedente.
* Le offerte singole e non ancora accettate possono partecipare ad una singola trattazione al momento.

## Sotto un Punto di Vista Concreto

### Le Tecnologie Usate

#### L’Ecosistema SpringBoot

L’applicazione E-Cycle è costruita sull’ecosistema Spring Boot, una piattaforma Java moderna pensata per semplificare la creazione di applicazioni web robuste e scalabili.

Grazie alla sua architettura modulare, Spring Boot ha permesso di strutturare il progetto secondo il paradigma MVC (Model-View-Controller), garantendo una netta separazione tra la logica di business, l’accesso ai dati e la visualizzazione.

Spring Data JPA è stato utilizzato per l’interazione con il database, permettendo una gestione efficiente delle entità e delle operazioni CRUD attraverso repository dichiarativi.

Spring Boot ha anche facilitato l’integrazione di funzionalità avanzate, come l’esecuzione periodica del matching automatizzato, tramite l’utilizzo di task schedulati.

#### MariaDB

Come database relazionale è stato scelto MariaDB, una soluzione open-source stabile, sicura e pienamente compatibile con MySQL. MariaDB ha offerto prestazioni solide e una buona scalabilità, supportando il modello concettuale e logico definito in fase di progettazione.

Sono state definite tabelle normalizzate e vincoli di integrità referenziale per garantire coerenza e affidabilità nei dati. Il motore di database ha inoltre supportato in modo efficiente le query necessarie al matching, garantendo tempi di risposta adeguati anche su set di dati non banali.

#### L’Algoritmo SHA-256 e il Concetto dei *Sali*

Per garantire la sicurezza delle credenziali degli utenti è stato implementato un sistema di cifratura articolato su due livelli, il secondo dei quali utilizza un sale variabile.

SHA-256 è un algoritmo crittografico della famiglia SHA-2, standardizzato dal NIST, che produce un hash di 256 bit. Questo algoritmo è impiegato nel primo livello di cifratura delle password, eseguito direttamente dal client.

Il secondo livello di protezione viene applicato lato server. In questa fase, l’hash ricevuto dal client viene ulteriormente elaborato attraverso un nuovo passaggio SHA-256. A questa elaborazione viene aggiunto un sale, ovvero una stringa casuale o semi-casuale che serve a rendere unico l’hash risultante anche a fronte della stessa password. In questo caso, il sale è rappresentato da un timestamp (senza millisecondi) corrispondente al momento preciso in cui la password viene aggiornata.Questo approccio consente di rendere estremamente difficile l’uso di attacchi basati su dizionari o tabelle pre**-**computate (come le *rainbow tables*), poiché ogni aggiornamento genera un hash irripetibile anche a parità di input iniziale.

### Alcune Sezioni di Codice

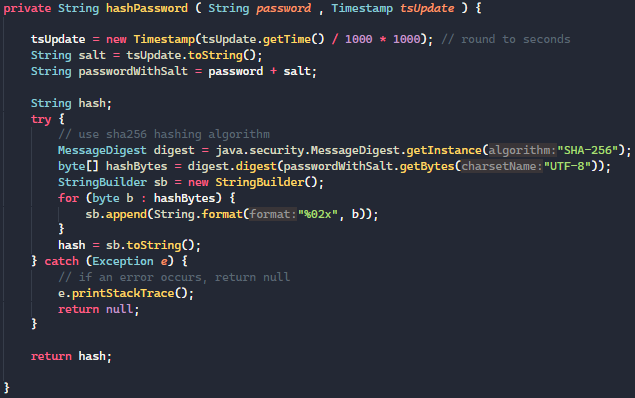
#### La Cifratura delle Password

##### Il Primo Livello

A screen shot of a computer code

AI-generated content may be incorrect.

##### Il Secondo Livello



#### A screenshot of a computer program AI-generated content may be incorrect.Il Matching Automatizzato

A computer screen shot of text

AI-generated content may be incorrect.