GRUPPO

JereMIE FWAPA KORANGI

PARFAIT WESILAKENDA KUBIKULAVE

STERLING MOMO NGUIMASTA

**PROGETTO 32 Contenitore di evento per i ricercatori**

2025

**Indice**

1. **Introduzione** 
   1. EventApp
2. **Analisi dei requisiti**
   1. L’idea
   2. Glossario
   3. Requisiti Funzionali
   4. Requisiti non funzionali
   5. Casi d’uso
   6. Diagramma delle attività
3. **Progettazione**
   1. Tecnologie utilizzate
   2. Database
4. **Sviluppo e test**
   1. Struttura dell’applicazione web
   2. Navigazione
5. **Conclusioni**

**Capitolo 1**

**Introduzione**

Come introdotto nel corso di ingegneria del software, lo scopo principale di questo lavoro è di applicare tutti i concetti imparati a lezione per realizzare un software talmente grande e complesso che sarà realizzato dalla squadra Team PSJ. L’obiettivo principale è quindi di analizzare, di progettare, di implementare e di testare un software, di preciso un’applicazione web, avendo lavorato con questa tipologia di software.  
  
Nel nostro caso, ci è richiesto di creare un contenitore di evento per i ricercatori, si vuole sviluppare una applicazione web per la diffusione di eventi legati alla ricerca. In particolare, si vuole creare un portale di eventi, tipo conferente o workshop, che i ricercatori o altri entri creano e vogliono diffondere. Agli eventi possono partecipare i ricercatori iscritti alla piattaforma. Si deve poter creare un evento con tutti i suoi dettagli: data, luogo, programma, etc… Nel caso in cui ci si iscrive ad un evento, il ricercatore riceve tutti gli aggiornamenti sul suo cruscotto.

Per migliorare l’interazione con l’app, abbiamo deciso di darle un nome, EventApp.

* 1. **EventApp**

Come anticipato nell’introduzione, abbiamo deciso di chiamarla EventApp, essendo un software, dovrebbe avere un nome per essere identificato nel mercato e per favorire principalmente questa fase di sviluppo per non richiamarla ogni volta con la sua descrizione “contenitore di evento per i ricercatori”.  
  
EventApp si tratta di un’applicazione molto complessa che sarà sviluppata con tutte le nuove tecnologie che abbiamo a disposizione al giorno di oggi e soprattutto grazie alla maggiore flessibilità di queste tecnologie che cercano sempre di migliorarsi e di produrre delle versioni più robuste e sofisticate.

**Capitolo 2**

**Analisi dei requisiti**

L’analisi dei requisiti è quel processo di comprensione di ciò che è richiesto a un sistema. Sviluppare un’applicazione web non significa scrivere delle righe di codice, ma ha un altro significato che gli sviluppatori considerano come la base dello sviluppo. Si tratta di scrivere più o meno formale e rigorosa di una caratteristica del sistema.

La gestione dei requisiti (acquisizione, analisi, negoziazione, specifica, validazione) è il primo passo del processo di sviluppo e una delle fasi più critiche dello sviluppo software, perché influenza in modo diretto il successo o il fallimento dei progetti.

**2.1 L’idea**

In questo progetto, si vuole realizzare un’applicazione web based che permetta ai ricercatori ed altri enti di creare degli eventi che vogliono diffondere. Lo scenario tipico è quello di avere una pagina principale che mostra tutti gli eventi ai quali possono iscriversi delle tipologie di utenti, questi eventi vengono creati dagli utenti che si registrano nella piattaforma. Questi utenti sono principalmente ricercatori ma anche altri enti. Hanno a disposizione una pagina personale dove possono visualizzare, inserire, modificare e rimuovere degli eventi. In piu, devono ricevere degli aggiornamenti sugli eventi in cui sono iscritti. Insomma, l’applicazione deve consentire l’inserimento, la visualizzazione, la modifica e la rimozione degli eventi, fornendo anche aggiornamento in tempo reale, e permettendo a tutte le tipologie di utenti di iscriversi agli eventi che possono essere creati solo dai ricercatori ed altri enti.

Il nome della web-app sarà “EventApp”.

Le funzionalità richieste sono:

* **Visualizza** eventi
* **Registrazione** dell’account (ricercatori o altri enti)
* **Login** dell’utente nel sistema
* **Iscrizione** ad un evento
* **Ricerca** di un evento tramite hashtag
* **Inserimento**, **modifica** e **rimozione** degli eventi
* **Notifica** sugli eventi iscritti
* **Logout** dell’utente dall’applicazione

Esempio. Un ricercatore sta organizzando un evento che vorrebbe diffondere, ma non ha conoscenza dell’applicazione. Entra in EventApp, e visualizza tutti gli eventi che gli altri ricercatori hanno già diffuso, ma si registra prima nella piattaforma e poi fa l’accesso e crea un evento che poi pubblica nell’applicazione. E poi potrebbe iscriversi ad un altro evento creato da un altro utente.

**2.2 Glossario**

|  |  |  |
| --- | --- | --- |
| **TERMINE** | **DESCRIZIONE** | **SINONIMI** |
| Home Page | Pagina principale dell’applicazione dove è visualizzata le tre azioni importanti da fare | Pagina principale |
| Ricercatore | Attore del sistema | Ricercatore, utente |
| Atri enti | Attore del sistema | Altri enti, |
| Login | Azione dell’utente per essere autenticato nel sistema | Accesso al sistema, connessione, autenticazione |
| Logout | Azione dell’utente per uscire dal sistema | Uscita dal sistema, disconnessione |
| Registrazione | Azione di creare un conto nel sistema | Registrarsi, Creazione conto |
| Visualizzazione eventi | Lista degli eventi sulla home page | Manage event |
| Hashtag | Filtro che consente di visualizzare gli eventi di un determinato tipo | filtro |
| Password | Sequenza di caratteri alfanumerici scelta dall’utente per accedere al sistema | Parola segreta |

**2.3 Requisiti funzionali**

I requisiti funzionali descrivono le funzionalità ed i servizi forniti dal sistema (cosa deve essere fatto). Nel ciclo di sviluppo software i requisiti funzionali rappresentano i casi d’uso. Di seguito sono riportate nel dettaglio le funzionalità richieste.

**Registra account**

Rappresenta la pagina di benvenuto dell’applicazione. Ogni utente che intende utilizzare l’applicazione per diffondere gli eventi, che sia ricercatore o utenti, deve essere registrato nel database. Al primo accesso all’applicazione, per poter fare il Login nel sistema, l’utente deve inserire dei dati personali di registrazione:

* Nome utente
* Indirizzo e-mail
* Password
* Conferma password
* Tipologia utente (ricercatori o altri enti)

Una volta che l’utente ha inserito i suoi dati, il sistema li memorizza nel database per poterli utilizzare successivamente in fase di login. Per l’autenticazione nel sistema, l’utente dovrà inserire indirizzo e-mail e password.

**Login**

Questa funzionalità permette al sistema di autenticare l’utente tramite l’account creato precedentemente nel caso d’uso registra account.

In fase di login, il sistema mostra all’utente i campi in cui inserire indirizzo mail e password, verifica la correttezza dei dati inseriti confrontandoli con i dati presenti nel database.

Se i dati inseriti risultano corretti, viene autenticato nel sistema e può usare l’applicazione, se i dati non sono corretti, il sistema propone all’utente di modificarli.

**Visualizza eventi**

Questo requisito è uno dei più importanti perché permette a tutti gli utenti dell’applicazione, di iscriversi ad un evento anche senza essere registrato nella piattaforma. La visualizzazione degli eventi deve prevedere una vista:

1. Il sistema deve prevedere una sezione “hashtag” da cui è possibile visualizzare la tipologia degli eventi che scelta dall’utente

**Iscrizione**

L’utente che ha scelto l’evento per cui vuole iscriversi ha a disposizione dei dati che vengono generati dal sistema, tra i quali: il titolo, la data e l’ora, il luogo dell’evento, la descrizione.

**Inserimento – Modifica -Rimozione**

L’utente che si è registrato nel sistema, per diffondere un evento, ha la possibilità di aggiungerlo, di modificarlo e di rimuoverlo

**Ricerca con l’hashtag**

Un qualsiasi utente dell’applicazione deve essere capace di trovare un evento specifico attraverso l’uso della barra di ricerca proposta dal sistema con l’hashtag che corrisponde all’evento per il quale corrisponde in funzione dei dati inseriti dal creatore dell’evento stesso.

**Notifica sugli eventi**

Dopo essere iscritto ad un evento, i ricercatori, gli altri enti e gli utenti esterni devono ricevere dal sistema, una notifica di aggiornamento in tempo reale dell’evento per il quale sono iscritti.

**Logout**

Il sistema deve fornire la procedura di uscita dall’applicazione. Effettuando il logout, l’utente viene scollegato dall’applicazione.

**2.4 Requisiti non funzionali**

I requisiti non funzionali non sono collegati direttamente con le funzioni implementate dal sistema, ma piuttosto alle modalità operative, di gestione. Definiscono vincoli sullo sviluppo del sistema. In seguito, verranno descritti tali requisiti.

**Responsive web design**

Nello sviluppo dell’applicazione è richiesto l’utilizzo della tecnica di web design responsive, in modo che le pagine web adattino automaticamente il layout per fornire una visualizzazione ottimale in funzione dell’ambiente nei quali vengono visualizzati: pc, tablet, smartphone sono i principali.  
 **Durata sessione**

Se l’utente non esegue alcuna azione nel sistema, la sessione deve interrompersi dopo 10minuti, quindi scollegare l’utente.

**Visualizzazione degli eventi**

Gli eventi vengono elencati in una lista in ordine random con tutte le informazioni sull’evento (data, programma, luogo…)

**Linguaggio di markup**

Il linguaggio di markup da utilizzare è HTML5

**Database**

Le informazioni principali da salvare nel database riguardano gli account degli utilizzatori del sistema e gli eventi.

**Lingua**

L’applicazione deve essere in lingua italiana.

**Sicurezza**

Il sistema deve prevedere delle norme di sicurezza:

* Le password salvate nel database devono essere criptate
* Controllare le variabili che arrivano dai form per evitare attacchi di SQL Injection, Form injection, Variable injection.

**Concorrenza**

L’applicazione deve prevedere l’utilizzo di più account contemporaneamente, e uno stesso account deve poter essere utilizzato da più utenti allo stesso momento.

**2.5 Casi d’uso**

Un caso d’uso specifica cosa ci aspetta da un sistema (“what?”) ma nasconde il suo comportamento (“how?”). È una sequenza di azioni, con varianti, che producono un risultato osservabile da un attore e rappresenta un requisito funzionale. Ogni sequenza (detta scenario) rappresenta l’interazione di entità esterne al sistema (attori) con il sistema stesso o sue componenti. Il flusso principale degli eventi viene separato dalle varianti alternative.

L’attore rappresenta un soggetto o un’entità che non fa parte del sistema, ma interagisce in qualche modo con esse. Individua un ruolo che l’utente ricopre nell’interagire con il sistema. Gli attori eseguono i casi d’uso.

Nel nostro caso, l’attore del sistema è l’utente (Ricercatori, Altri enti, utente esterno), i casi d’uso sono: Visualizzazione eventi, Login, Registra account, Iscrizione, Inserimento-modifica-rimozione, Ricerca con l’hashtag, Notifica, Logout.

**2.5.1 Diagramma dei casi d’uso**

Il diagramma dei casi d’uso è il modello che descrive i requisiti del sistema in termini di funzionalità (casi d’uso) e ambiente circonstante (attori). Mostra cosa deve fare il sistema.

Immagine che contiene diagramma, testo, linea, schizzo

Descrizione generata automaticamente

Figura 1: Diagramma dei casi d'uso

Di seguito vengono descritte le specifiche dei casi d’uso con scenari principali e alternativi. Uno scenario è una sequenza di passi che descrivono l’interazione tra un sistema e un attore (che dovrebbe trarre vantaggio dall’interazione).

**2.5.2 Registra account**

* **Nome caso d’uso**: Registra account
* **Id**: UC1
* **Attori**: Utente (ricercatore o altri enti o utente esterni)
* **Precondizioni** 
  + L’utente non è registrato
* **Scenario principale**
  + L’utente accede per la prima volta al sistema
  + Il sistema visualizza la scelta tra Login e Registra account
  + L’utente sceglie Registra account
  + Il sistema visualizza i campi da compilare per registrare l’account utente
  + L’utente inserisce i dati di registrazione
  + Il sistema verifica la correttezza dei dati
    - Se i dati non sono corretti, il sistema evidenza i campi da modificare e ritorna al punto in cui visualizza i campi da compilare
  + Se i dati sono corretti, il sistema conclude la fase di registrazione
  + L’utente può effettuare il login
* **Postcondizioni:**
  + L’utente è registrato
* **Scenario secondario:**
  + L’utente accede al caso d’uso registra account
  + Il sistema visualizza i campi da compilare per la registrazione dell’account
  + L’utente inserisce i dati
  + Il sistema verifica che i dati sono già presenti nel database
  + Il sistema avvisa l’utente che i dati inseriti sono già stati utilizzati
* **Postcondizioni:**
  + Inizia il caso d’uso Login

**2.5.3 Login**

* **Nome caso d’uso**: Login
* **Id**: UC2
* **Attori**: Utente (ricercatore o altri enti)
* **Precondizioni**
  + L’utente ha avviato il sistema
* **Scenario principale:**
  + Il caso d’uso inizia quando l’utente avvia l’applicazione da un browser
  + Il sistema visualizza i campi dove inserire indirizzo mail e password dell’utente
  + L’utente inserisce indirizzo mail e password dell’utente
  + Il sistema verifica se indirizzo mail e password sono corretti
    - Se non sono corretti, si ritorna al punto dove visualizza i campi
  + Se indirizzo mail e password sono corretti, il sistema visualizza lo spazio utente
* **Postcondizioni:**
  + L’utente è registrato nel sistema
* **Scenario secondario:**
  + L’utente non è registrato nel sistema
  + Il sistema avvia il caso d’uso Registra account
* **Postcondizioni:**
  + Inizia il caso d’uso Registra account

**2.5.4 Visualizza eventi**

* **Nome caso d’uso**: Visualizza eventi
* **Id**: UC3
* **Attori**: Ricercatori, altri enti e utenti esterni
* **Precondizioni**:
  + L’utente ha fatto accesso all’applicazione
* **Scenario principale:**
  + Il sistema mostra la lista degli eventi in modo random disponibili per un certo periodo
  + L’utente visualizza tutti questi eventi
* **Postcondizioni**
  + L’utente può selezionare un evento specifico

**2.5.5 Iscrizione**

* **Nome caso d’uso**: Iscrizione
* **Id**: UC4
* **Attori**: Ricercatori, altri enti e utenti esterni
* **Precondizioni**
  + L’utente ha scelto l’evento per cui vuole fare l’iscrizione
* **Scenario principale**
  + Il caso d’uso inizia quando l’utente ha realmente scelto per quale evento vuole partecipare
  + Il sistema chiede all’utente se non è connesso di connettersi o di continuare compilando una sezione dei dati
  + Il sistema elabora la domanda ed invia una mail di avvenuta iscrizione all’evento
* **Postcondizioni**:
  + L’utente può visualizzare, modificare o cancellare l’iscrizione e ricevere aggiornamento

**2.5.6 Inserimento – Modifica – Rimozione**

* **Nome caso d’uso**: Inserimento, Modifica, Rimozione
* **Id**: UC5
* **Attori**: Ricercatori o altri enti
* **Precondizioni**:
  + Non esiste nessun evento
* **Scenario principale**
  + Il caso d’uso inizia quando l’utente si è connesso
  + Il sistema permette di aggiungere un evento
  + L’utente sceglie questa azione
  + Il sistema controlla la correttezza dell’operazione e procede
* **Postcondizioni**
  + Esiste un evento
* **Scenario secondario**
  + L’utente vuole modificare o rimuovere un evento esistente
  + Il sistema chiede che azione vuole fare
  + L’utente sceglie un’azione
  + Il sistema controlla la correttezza dell’operazione e procede

**2.5.7 Ricerca con hashtag**

* **Nome caso d’uso**: Ricerca con hashtag
* **Id**: UC6
* **Attori**: Utente
* **Precondizioni**
  + L’utente ha fatto accesso alla pagina di visualizzazione degli eventi
* **Scenario principale**
  + Il caso d’uso inizia quando l’utente vuole ha accesso alla pagina
  + Il sistema apre una barra di ricerca che permette di inserire l’hashtag
  + L’utente digita l’hashtag
  + Il sistema rilascia gli eventi che corrispondo all’hashtag richiesto
* **Postcondizioni**
  + Il sistema non trova l’evento ricercato

**2.5.8 Notifica**

* **Nome caso d’uso**: Notifica
* **Id**: UC7
* **Precondizioni**:
  + L’utente è già connesso al sistema
* **Scenario principale:** 
  + Il caso d’uso inizia quando l’utente è connesso
  + Il sistema fornisce all’utente un aggiornamento dell’evento
* **Postcondizioni**:
  + L’utente non è connesso al sistema
* **Scenario secondario**
  + L’utente fa accesso al suo account
  + Il caso d’uso inizia
  + Il sistema fornisce all’utente un aggiornamento in tempo reale sull’evento

**2.5.9 Logout**

* **Nome caso d’uso**: Logout
* **Id**: UC8
* **Attori**: Utente (ricercatori o altri enti)
* **Precondizioni**
  + L’utente ha effettuato il login
* **Scenario principale**:
  + Il caso d’uso inizia quando l’utente vuole uscire dal sistema
  + L’utente chiede al sistema di disconnettersi
  + Il sistema disconnette l’utente
* **Postcondizioni**
  + L’utente è disconnesso dal sistema

**2.6 Diagramma delle attività**

Il diagramma delle attività, in UML, è un diagramma di flusso (con alcuni elementi aggiuntivi) che mostra una sequenza di attività. Viene utilizzato per rappresentare i passi (le transazioni) che compongono il flusso di un caso d’uso, descrivono quindi il comportamento dinamico di un sistema. La rappresentazione delle attività è comoda in quanto consente di rappresentare sinteticamente flusso principale e flussi alternativi. Il diagramma in figura2 mostra il flusso delle attività dell’utente.

Immagine che contiene testo, diagramma, Piano, Disegno tecnico

Descrizione generata automaticamente

Figura 2 - Diagramma delle attività

**Capitolo 3**

**Progettazione**

In questa fase si definisce l’architettura software su cui su baserà la realizzazione del prodotto EventApp. Abbiamo scelto una strutturazione tipica su tre livello logico-funzionali ma che possono essere distribuiti anche su più livelli.  
 L’applicazione web Three-Tier si sviluppa su 3 livelli:

* Presentazione (cliente): costituisce l’interfaccia utente dell’applicazione e corrisponde al browser web. Le tecnologie scelte da utilizzare sono HTML e CSS
* Applicazione (Servente Web): è il livello logico della web-app, la componente elaborativa. Dal lato server (Server-side) utilizza la tecnologia PHP, dal lato client (Client-side) utilizza il linguaggio di scripting Javascript.
* Dati (Servente RDBMS): consente di modellare e gestire il contenuto informativo dell’applicazione. La tecnologia usata a questo livello è il Database relazione MySQL.

Immagine che contiene testo, diagramma, schermata, Biglietto Post-it

Descrizione generata automaticamente

Figura 3: Struttura web application

L’applicazione ha una architettura di comunicazione Client-Server di tipo MVP REST-full API CRUD.

Gli strumenti che abbiamo utilizzato sono:

Docker è uno strumento per la creazione, la gestione e l’orchestrazione di piu container di applicazioni. Il suo obiettivo è quello di ottimizzare il processo di sviluppo, test, consegne e distribuzione delle applicazioni, mediante la creazione di pacchetti di ogni componente dell’applicazione in un contenitore separato.

Nel nostro caso, l’abbiamo utilizzato per il lato server installando 3 immagini:

* **Web**: php:8.2-apache, questa immagine ci consente di lanciare la nostra applicazione come se fosse un server reale
* **Database**: mysql:8.1.0, ci consente di creare e di gestire il database
* **Phpmyadmin**: phpmyadmin/phpmyadmin, consente di gestire php e di avviare lo strumento di gestione di MySQL

Si accede all’applicazione, collegandosi all’indirizzo <http://localshot>, per accedere invece a phpmyadmin, <http://localhost:8080>

**3.1 Tecnologie utilizzate**

In questa sezione sono descritte in dettaglio le tecnologie utilizzare per la realizzazione dell’applicazione. Lo studio di esse ha occupato una parte rilevante del nostro lavoro in quanto sono tecnologie di nuova generazione.

**HTML**

L'**HyperText Markup Language** (lett. "linguaggio di marcatura d'ipertesto"), comunemente noto con l'acronimo **HTML**, è il linguaggio di marcatura più usato per i documenti web. Nato per la formattazione e impaginazione di documenti ipertestuali disponibili nel web 1.0, oggi è utilizzato principalmente per il disaccoppiamento della struttura logica di una pagina web (definita appunto dal markup) e la sua rappresentazione, gestita tramite gli stili CSS per adattarsi alle nuove esigenze di comunicazione e pubblicazione all'interno di Internet.[[](https://it.wikipedia.org/wiki/HTML#cite_note-Bochiccio-Mostarda-3)

**CSS**

**Cascading Style Sheets**, meglio noto come **CSS** (in italiano **fogli di stile a cascata**), è un linguaggio usato per definire la formattazione di documenti HTML, XHTML e XML, ad esempio i siti web e relative pagine web. Le regole per comporre il CSS sono contenute in un insieme di direttive (*Recommendations*) emanate a partire dal 1996 dal W3C.

**JAVASCRIPT**

**JavaScript** è un linguaggio di programmazione multi paradigma orientato agli eventi, utilizzato sia nella programmazione lato client web che lato server (Node.js) per la creazione di RESTful API, applicazioni desktop e embedded, siti web e applicazioni web, di effetti dinamici interattivi tramite funzioni di script invocate da *eventi* innescati a loro volta in vari modi dall'utente sulla pagina web in uso (mouse, tastiera, caricamento della pagina ecc...).

**PHP**

**PHP** (acronimo ricorsivo di "PHP: HyperText Preprocessore", preprocessore di ipertesti; originariamente acronimo di "Personal Home Page") è un linguaggio di scripting interpretato, originariamente concepito per la programmazione di pagine web dinamiche. L'interprete PHP è un software libero distribuito sotto la licenza PHP.

**DOCKER**

**Docker** è un popolare[[1]](https://it.wikipedia.org/wiki/Docker#cite_note-1) software libero che utilizza la virtualizzazione a livello di sistema operativo per eseguire applicazioni in ambienti isolati chiamati container.

**WEB SERVER APACHE**

**Apache** è il nome di un server web libero sviluppato dalla Apache Software Foundation. È la piattaforma server Web modulare più diffusa, in grado di operare su una grande varietà di sistemi operativi, tra cui UNIX/Linux, Microsoft Windows e OpenVMS. È un software che realizza le funzioni di trasporto delle informazioni, di internet work e di collegamento, ed ha il vantaggio di offrire funzioni di controllo per la sicurezza come quelle effettuate da un proxy.

**MYSQL**

**MySQL** o **Oracle MySQL** è un *relational database management system* (RDBMS) composto da un client a riga di comando e un server. Ambo i costituenti sono multipiattaforma e sono disponibili ufficialmente su tutte le distribuzioni conosciute, quali Debian, Ubuntu e CentOS, sebbene lo abbiano sostanzialmente sostituito con MariaDB a partire dal [2012](https://it.wikipedia.org/wiki/2012).

**BOOTSTRAP**

**Bootstrap** è una libreria di strumenti liberi per la creazione di siti e applicazioni per il Web. Essa contiene modelli di progettazione basati su HTML e CSS, sia per la tipografia, che per le varie componenti dell'interfaccia, come moduli, pulsanti e navigazione, così come alcune estensioni opzionali di JavaScript.

**Software utilizzati**

* Windows 11 Home x64: sistema operativo
* Docker: web server
* Visual studio: Ambiente di sviluppo
* DrawIO: per la realizzazione dei diagrammi UML
* Mozilla Firefox, Google Chrome, Opera, Internet Explorer: browser usati per ricerca, sviluppo e test.

**3.2 Database**

Quando i dati sono molti e salvarli su filesystem risulta inefficiente, bisogna usare il supporto di una base di dati, nel nostro caso l’abbiamo chiamata “contenitore”.

L’analisi dei requisiti ha portato all’individuazione di cinque entità fondamentali: “Account”, “Eventi”, “Iscrizione”, “Notifica”, e “Hashtag”. Le entità e le associazioni che intercorrono tra esse sono rappresentate nel seguente schema Entity/Relationship.

Immagine che contiene schermata, diagramma, Carattere, design

Descrizione generata automaticamente

Figura 4: Schema Entity/Relationship

Lo schema logico ottenuto è il seguente:

ACCOUNT (id\_account, nome\_utente, indirizzo\_mail, password, tipologia)

EVENTI (id\_evento, titolo, descrizione, data\_svolgimento, luogo\_svolgimento, autore, data\_creazione, image, status)

ISCRIZIONE (id\_iscrizione, data\_iscrizione, status)

HASHTAG (id\_hashtag, parola)

NOTIFICA (id\_notifica, descrizione, data\_creazione, status)

Nelle sezioni seguenti verrà approfondita l’analisi delle entità, precisando il ruolo di ciascuna.

**3.2.1 Entità “Account”**

L’entità “Account” serve per tenere traccia di tutti gli utilizzatori del sistema (ricercatore, altri enti e utenti esterni). Il compito principale è la gestione delle credenziali per poter effettuare il login tramite indirizzo mail e password scelti dall’utente in fase di registrazione.  
 Gli attributi di Account sono i seguenti:

**id\_account** serve a identificare univocamente un utente del sistema (chiave primaria)

**nome\_utente** serve a conoscere il nome completo del proprietario del conto.

**indirizzo\_mail** serve principalmente nella fase di login e nella fase di notifica tramite mail

**password** serve anche in fase di login per autenticare l’utente nel sistema. Questo campo

come detto nella fase di analisi, viene salvato nel database criptato per motivi di sicurezza

**tipologia** serve per conoscere di che tipologia è l’utente che si è registrato nel sistema

L’entità “Account” risulta la seguente:

Immagine che contiene schermata, testo, Carattere

Descrizione generata automaticamente

**3.2.2 Entità “Eventi”**

Questa entità ci permette di tenere traccia di tutti gli eventi che vengono aggiunti nell’applicazione.  
 Gli attributi di Eventi sono i seguenti:

**id\_evento** serve per identificare in modo univoco l’evento

**titolo** serve per il titolo dell’evento

**descrizione** serve a descrivere l’evento

**data\_svolgimento** serve a conoscere quando si terrà l’evento

**luogo\_svolgimento** serve a conoscere invece l’ora dell’evento

**autore** serve a identificare chi ha creato l’evento

**data\_creazione** tiene traccia dell’ora esatta della creazione dell’evento

**image** serve per aggiungere un’immagine legata all’evento

**status** ci da l’indicazione sulla validità dell’evento

L’entità risulta quindi la seguente:

Immagine che contiene schermata, nero

Descrizione generata automaticamente

**3.2.3 Entità “Iscrizione”**

L’entità ‘Iscrizione’ tiene traccia di tutte le iscrizioni effettuate dagli utenti.  
 Gli attributi di iscrizione sono i seguenti:

**id\_iscrizione** serve a identificare univocamente le iscrizioni degli utenti

**data\_iscrizione** serve a memorizzare la data dell’iscrizione ad un evento

**status** indica lo status dell’iscrizione, se è valido o no

L’entità risulta la seguente:

Immagine che contiene testo, schermata, Carattere, design

Descrizione generata automaticamente

**3.2.4 Entità “Hashtag”**

L’entità “hashtag” tiene traccia di tutte le parole associate ad un evento creato da un utente, per permettere una ricerca rapida in funzione del tag corrispondente.

Gli attributi di hashtag sono i seguenti:

**id\_hashtag** serve a identificare univocamente un hashtag

**parola** serve a memorizzare la parola corrispondente

L’entità risulta la seguente:

Immagine che contiene testo, schermata, biglietto da visita, Carattere

Descrizione generata automaticamente

**3.2.5 Entità “Notifica”**

Questa entità tiene traccia di tutte le notifiche che gli utenti del sistema ricevono in funzione dell’evento sul quale viene effettuate delle azioni.  
 Gli attributi di Notifica sono i seguenti:

**id\_notifica** serve a identificare univocamente la notifica

**descrizione** fornisce un piccolo messaggio della notifica

**data\_creazione** indica il momento della creazione della notifica

**status** indica la validità della notifica in funzione del fatto che l’utente lo apre o no.

L’entità risulta la seguente:

Immagine che contiene testo, schermata, Carattere, biglietto da visita

Descrizione generata automaticamente

Capitolo 4

Implementazione e test

E la fase realizzativa. Ogni modulo, come unità indipendente, viene implementato e controllato per assicurarne la correttezza.