Analisi WikiStudenti

Avveduto, Barani, Brambilla, Cecire

[**Scopo del progetto 1**](#_ie5ofsrrt8gk)

[Analisi funzionale 1](#_gjg5ztrawt5k)

[Gestione dei servizi 3](#_r5f5o4b11lux)

[Analisi sistemistica 3](#_k8r0vhji5fmj)

[Casi d’uso 5](#_uqlzt3lvzenn)

[Schema E/R 7](#_2oyek35bnnd5)

[Schema logico 7](#_in9vkoqmvpdg)

[**Comparazione 8**](#_lada8459otk0)

[Soluzione custom vs MediaWiki 8](#_mvtqxtqy5x1g)

[Copertura dei requisiti (soluzione custom) 8](#_s8frpn9mn3dx)

[Affinità e differenze 9](#_5tt1kej6dds8)

[Affinità 9](#_ehjqj53ue1ti)

[Differenze 9](#_lm73lsr6fq1y)

[Tempi di sviluppo (soluzione custom) 10](#_ycea2lc859pt)

[Tempi di installazione e configurazione (soluzione MediaWiki) 10](#_g1r0rrv62z61)

# Scopo del progetto

Si vuole realizzare un server web che possa gestire un’applicazione wiki dove gli studenti e i professori della scuola possono accedere e creare contenuti relativi alle materie studiate in modo che siano sempre accessibili.  
Il progetto ha un duplice obiettivo: lo sviluppo di un’applicazione full-stack (utilizzando HTML, JavaScript e il framework Bulma per la parte client-side e PHP per la parte server-side e di connessione al database) e la predisposizione di un ambiente di esecuzione fisico sicuro (il server).

## Analisi funzionale

L’applicazione web permette di visualizzare e creare articoli inerenti specifiche materie.

Ogni **articolo** è composto da (in ordine di apparizione):

* Titolo
* Abstract (riassunto dei contenuti dell’articolo)
* Immagine di presentazione
* Contenuto testuale
* Slider di immagini relative all’articolo → Numero di immagini non fisso
* Link ad altri contenuti (sia della stessa applicazione che di altri siti web)

Ogni articolo può avere più versioni differenti, in base alle modifiche effettuate, ma solo una (l’ultima salvata o quella scelta da un utente moderatore) sarà quella visualizzabile da tutti.

Un utente **visualizzatore** può leggere i contenuti presenti nell’applicazione e utilizzare una form di registrazione dove può **candidarsi** ad un determinato ruolo (come **scrittore**, per poter creare contenuti o come **moderatore**, per revisionare ed approvare i contenuti creati dagli scrittori). L’applicazione invia una mail all’**amministratore** che approverà o meno la richiesta dell’utente.  
La **form** avrà i seguenti campi:

* Nome
* Cognome
* Classe
* Anno di nascita
* Email (identifica l’account dell’utente) → Successivamente criptata
* Password → Successivamente criptata
* Checkboxes con i ruoli a cui ci si vuole candidare

Uno **scrittore** utilizza la form di compilazione dell’articolo, che contiene tutti i campi di cui esso è composto, per creare i contenuti. Una volta che lo scrittore ha invitato la form (che sia una modifica o l’aggiunta di un nuovo documento), l’articolo diventa una bozza (in attesa di approvazione) e viene inviata una mail a tutti i moderatori con il seguente testo:

**Oggetto:** Bozza da approvare <Nome documento> <ID\_Editor>

**Corpo della mail:**

Gentile moderatore,

E’ presenta una nuova bozza da approvare del documento <Nome\_documento> dell’editor <ID\_Editor><Nome\_Editor>.

La invitiamo a dare un’occhiata!

Cordialmente,

Lo staff di Wiki Studenti

Le bozze in attesa di approvazione non sono più modificabili fino a quando non vengono approvate o rifiutate dai moderatori.

Ogni documento scritto ha un numero non limitato di immagini e di link correlati.

Lo scrittore non ha la possibilità di ridefinire il layout grafico dell’articolo.  
Una volta approvata la bozza, le immagini verranno salvate nel file system del web server.

Un **moderatore**, attraverso un’apposita interfaccia grafica, può approvare o rifiutare qualsiasi bozza inviata dagli scrittori e ripristinare ad una determinata versione qualsiasi articolo.

Tutti gli utenti possono candidarsi a determinate posizioni e privilegi di gestione della wiki. Tali candidature possono essere approvate o rifiutate esclusivamente dall’amministratore, che riceverà la seguente mail:

**Oggetto:** Candidatura di <mail\_candidato>

**Corpo della mail:**

Gentile Amministratore,

E’ presente una nuova candidatura da approvare; mail del candidato: <mail\_candidato>.

La invitiamo a dare un’occhiata!

Cordialmente,

Lo staff di Wiki Studenti

L’**amministratore**, attraverso un’apposita interfaccia grafica, approva la creazione o la rimozione degli utenti e ne gestisce i privilegi.

Ogni volta un utente si logga nell’applicazione, viene visualizzata una schermata principale con le azioni possibili in base al ruolo (es. creare, modificare, approvare bozze, gestione utenti, ecc.).

### Gestione dei servizi

L’applicazione utilizzerà l’architettura SOA (Service Oriented Architecture), sfruttando la tecnologia REST, che prevede uno scambio di risorse in formato JSON.  
Verrà creato un web service, scritto in PHP, che farà le opportune query, in base al metodo HTTP corrispondente (GET/POST/PUT/DELETE), al database e ne restituirà il risultato in formato JSON al client

## Analisi sistemistica

Lo scopo principale nell’organizzazione della rete dell’ambiente di produzione è quello di **non** permettere nessuna **intrusione** nella rete.

Di seguito la **topologia logica** dell’organizzazione della rete.

# 

Il web server si trova nella **DMZ** (zona demilitarizzata), quindi è esposto rispetto al server MySQL/Application server che si trova oltre il firewall (di base), di conseguenza è piùprotetto.

**ENTRAMBI I SERVER AVRANNO INSTALLATA L’ULTIMA VERSIONE SERVER DI UBUNTU.**

In tale server sarà installato il database ed il software PHP che sarà accessibile tramite un web server Apache che processerà i dati in ingresso così da poter effettuare query al database in localhost, senza uscire dalla propria rete.

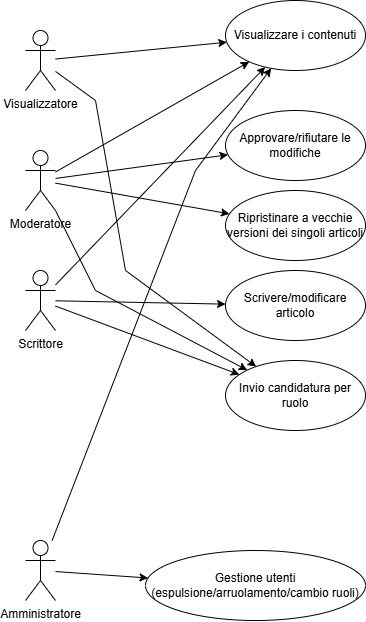
I due server si troveranno su reti diverse ed Il collegamento sarà protetto tramite tunnel SSL con l’implementazione open source “OpenSSL”.

Entrambi i server utilizzeranno Apache2 per ospitare rispettivamente la web app ed il servizio php per l’accesso al DB.

Sarà implementato un autenticatore (server RADIUS) per gestire gli utenti e i ruoli di ciascuno.

Verrà utilizzato solo ed esclusivamente il protocollo HTTPS, generando autocertificati.

## Casi d’uso



| **Caso d’uso: Visualizzare contenuti** |
| --- |
| **ID:** UC1 |
| **Attori:** Visualizzatore, Moderatore, Scrittore, Amministratore |
| **Precondizioni:** Aver avviato la pagina |
| **Sequenza degli eventi:**   1. Ricerca dei contenuti 2. Visualizzazione risultati ricerca 3. Selezione articolo 4. Visualizzazione effettiva dell’articolo |
| **Postcondizioni:** Navigazione del sito |

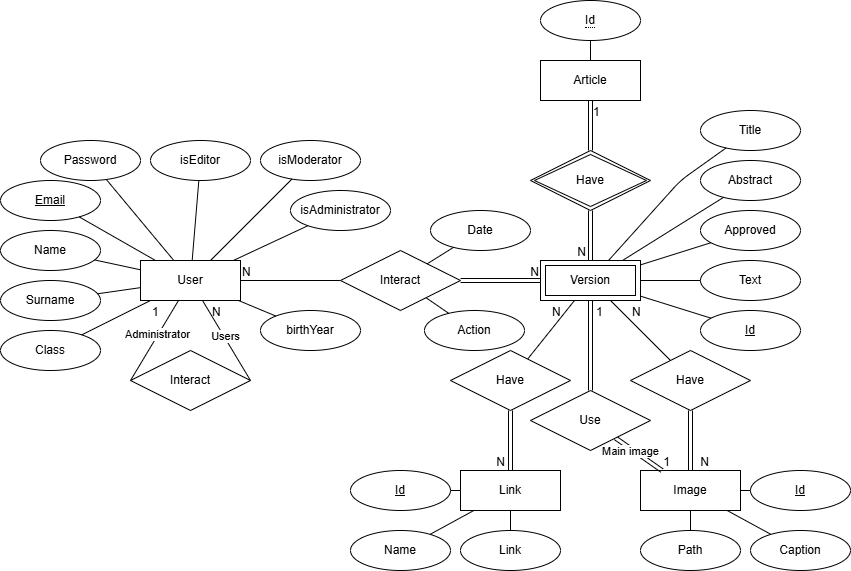
| **Caso d’uso: Approvare/rifiutare modifiche** |
| --- |
| **ID:** UC2 |
| **Attori:** Moderatore |
| **Precondizioni:** Richiesta modifica del testo da parte di uno scrittore. Ricezione mail di avviso |
| **Sequenza degli eventi:**   1. Il moderatore ricerca i contenuti 2. Visualizzazione risultati ricerca 3. Seleziona modifica da approvare 4. Il moderatore analizza il contenuto modificato 5. Il moderatore può accettare o rifiutare una richiesta di modifica del testo richiesta da uno scrittore 6. Se il moderatore accetta la modifica essa viene modificata nel database 7. Se il moderatore rifiuta la modifica viene inviata una mail |
| **Postcondizioni:** Viene inviata una mail. Visualizzazione dei nuovi contenuti se accettati, altrimenti il testo rimane invariato se la richiesta viene rifiutata |

| **Caso d’uso: Ripristinare vecchie versioni** |
| --- |
| **ID:** UC3 |
| **Attori:** Moderatore |
| **Precondizioni:** Modifiche precedenti presenti |
| **Sequenza degli eventi:**   1. Il moderatore ricerca i contenuti 2. Il moderatore naviga nella pagina 3. Il moderatore ripristina una versione precedente dell’articolo |
| **Postcondizioni:** I contenuti visualizzati diventano quelli ripristinati |

| **Caso d’uso: Scrivere/modificare articoli** |
| --- |
| **ID:**UC4 |
| **Attori:** Scrittore |
| **Precondizioni:** Necessità di scrivere o modificare un articolo |
| **Sequenza degli eventi:**   1. Lo scrittore ricerca i contenuti 2. Lo scrittore naviga nella pagina 3. Lo scrittore scrive o modifica un articolo utilizzando il form predefinito 4. Se i campi non sono compilati correttamente viene mostrato un errore 5. Se i campi sono compilati correttamente viene aggiunta al database 6. Invia la modifica in attesa di approvazione |
| **Postcondizioni:** Viene inviata una mail a tutti i moderatori |

| **Caso d’uso: Inviare candidatura per ruolo** |
| --- |
| **ID:** UC5 |
| **Attori:** Visualizzatore, Scrittore |
| **Precondizioni:** Visualizzare la pagina di registrazione |
| **Sequenza degli eventi:**   1. L’utente va nella pagina di registrazione 2. L’utente compila il form 3. Se non è compilato correttamente viene mostrato un errore 4. Se è compilato correttamente viene fatto un controllo delle credenziali sul database 5. Se le credenziali sono corrette L’utente invia una candidatura per uno o più ruoli 6. viene inviata una mail di candidatura all'amministratore |
| **Postcondizioni:** L’amministratore provvederà ad accettare o rifiutare la candidatura |

## Schema E/R



Un utente può interagire con una o più versioni (entità debole) e per ogni interazione ne viene salvato il tipo (scrittura, rimozione, approvazione, rifiuto) e la data.  
La creazione di un nuovo articolo non presente prima comporta la creazione di una prima versione e di un'istanza dell’entità articolo.

## Schema logico

User(Email, Name, Surname, Class, Password, isEditor, isModerator, isAdministrator, AdministratorEmail(FK))

Link(Id, Name, Link)

Image(Id, Path, Caption)

Article(Id)

Version(Id, ReferdToId, Title, Text, Abstract, Approved, MainImage(FK))

UserInteractsVersion(UserEmail(FK), VersionId(FK), Date, Action)

VersionHasLinks(VersionId(FK), LinkId(FK))

VersionHasImages(VersionId(FK), ImageId(FK))

# Comparazione

## Soluzione custom vs MediaWiki

Il nostro progetto prevede un’applicazione con un database relazionale (basato su MySQL) e un web server Apache con motore PHP. A differenza di un sistema come MediaWiki, la nostra soluzione offre maggiore flessibilità nella gestione dei dati e degli utenti.

Dal punto di vista della sicurezza, verranno implementate tecnologie avanzate come il tunnel SSL e restrizioni di accesso, garantendo un ambiente più sicuro rispetto alle impostazioni predefinite di altri sistemi. L’integrazione di un sistema personalizzato per la gestione dei permessi degli utenti (tramite un server RADIUS per l’autenticazione) ci consente di offrire un livello di personalizzazione più elevato.

### Copertura dei requisiti (soluzione custom)

* Utilizzo di Ubuntu per sfruttare il suo firewall per garantire una maggiore sicurezza.
* Utilizzo di HTTPS per lo scambio di dati criptati.
* Autenticazione tramite server RADIUS che gestisce i permessi di ogni utente loggato.
* Crittare il collegamento tra i due server tramite tunnel SSL.
* Deve essere tenuta traccia di ogni modifica.
* Per ogni documento deve essere possibile ripristinare ad una determinata versione.
* Gli utenti possibili sono quattro:
  + l’editor, ha come unico ruolo quello di creare/modificare documenti.
  + Il moderatore ha come compito approvare/declinare modifiche in ingresso dagli editor. Inoltre ha la possibilità di ripristinare specifici documenti a determinate versioni.
  + Utente generico ha l’unico ruolo di visualizzatore.
  + L'amministratore (una sola persona) ha il ruolo di gestire gli utenti ed i relativi permessi/ruoli.
* Ad ogni invio di modifica degli editor verrà recapitata una mail ai moderatori.
* Ogni utente generico ha la possibilità di candidarsi ad uno o più ruoli inserendo nome, cognome, email e password (criptata), classe e anno di nascita. All’invio della candidatura verrà recapitata una mail all'amministratore.
* Le immagini aggiunte ai documenti vengono salvate nel file system.

### Affinità e differenze

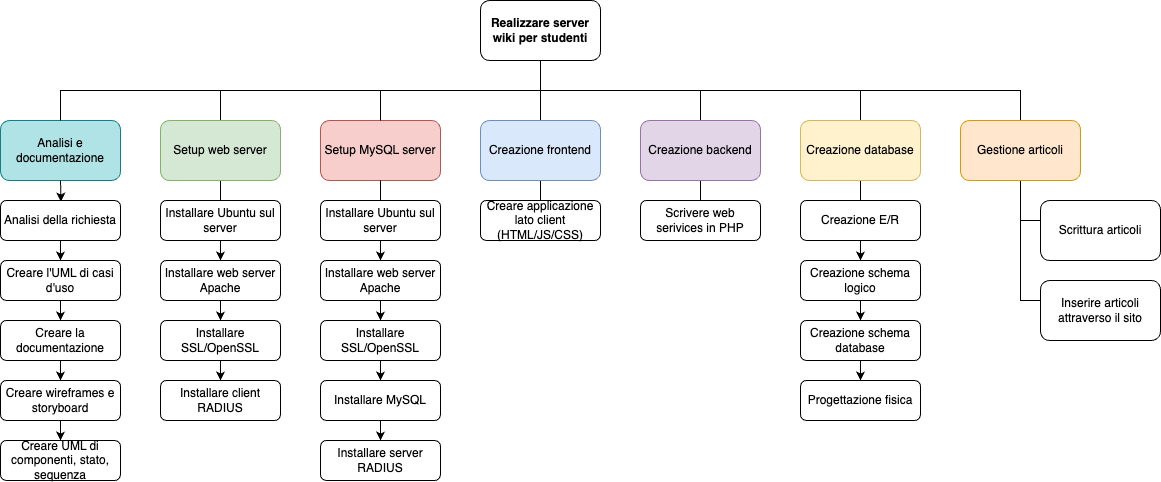
#### Affinità

* Entrambe le soluzioni si basano su un database MySQL per gestire e organizzare le informazioni.
* Il sistema di gestione degli utenti e permessi è molto simile: sia il nostro progetto che MediaWiki prevedono livelli di accesso per controllare chi può modificare o approvare i contenuti e chi può gestire gli utenti.
* Dal punto di vista tecnico, entrambi fanno affidamento su un web server Apache su cui gira un’applicazione PHP.
* La sicurezza è un altro punto in comune: in entrambi i casi si cerca di proteggere i dati chiudendo tutte le porte superflue e utilizzando strumenti come il openSSL per criptare i collegamenti.
* Entrambe le soluzioni offrono la possibilità di notificare determinati utenti tramite email.
* Entrambe le soluzioni offrono la “page history”, ovvero la possibilità di vedere la cronologia delle verifiche di ogni documento con la possibilità di ripristinare ad una determinata versione.

#### Differenze

* MediaWiki ha un sistema predefinito con permessi standard (lettura/scrittura) mentre nel nostro progetto c’è una maggiore flessibilità e personalizzazione dei ruoli.
* MediaWiki supporta ben 140 lingue mentre la nostra soluzione custom supporta solo la lingua italiana.
* MediaWiki offre una vera e propria interfaccia di editing del testo offrendo anche la possibilità di formattarlo (grassetto, corsivo…), mentre la nostra soluzione custom permette solo di incollare del testo.
* MediaWiki offre la possibilità di importare/esportare documenti mentre la nostra soluzione no.
* MediaWiki permette di visualizzare statistiche come per esempio i documenti in tendenza o i visitatori recenti mentre la nostra soluzione non prevede questa funzionalità.
* MediaWiki offre la possibilità di sfruttare diversi formati di testo (come html), mentre come detto la nostra soluzione permette solo di incollare del testo.
* Infine mediawiki offre dei servizi extra come l’integrazione di un calendario o di forum tramite plugin, servizio da noi non offerto.

### Tempi di sviluppo (soluzione custom)



Tempi previsti per ogni macro attività:

* Analisi e documentazione → 8 ore (1 giorno/uomo)
* Setup web server → 6 ore (0.75 giorni/uomo)
* Setup MySQL server → 4 ore (0.5 giorni/uomo)
* Creazione frontend → 20 ore (2.5 giorni/uomo)
* Creazione backend → 20 ore (2.5 giorni/uomo)
* Creazione DB → 5 ore (0.6 giorni/uomo)

### Tempi di installazione e configurazione (soluzione MediaWiki)

In base alla nostra esperienza l’installazione di mediawiki e la sua configurazione “semplice”, richiede all’incirca una o due ore. Invece lo stesso procedimento con una configurazione “complessa” richiede una decina d’ore, per quanto non sia possibile fare un’accurata stima del tempo impiegato.