CasaNote

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Abstract 3

1.3 Scopo 3

1.4 Analisi del dominio 4

1.5 Analisi e specifica dei requisiti 4

1.5.1 Spiegazione elementi tabella dei requisiti: 7

1.6 Use case 8

1.7 Pianificazione 9

1.8 Analisi dei mezzi 1

1.8.1 Software 1

1.8.2 Hardware 1

2 Progettazione 2

2.1 Design dell’architettura del sistema 2

2.1.1 Swimlanes 3

2.2 Design dei dati e database 6

2.3 Design delle interfacce 7

2.3.1 Registrazione utente 7

2.3.2 Login utente 8

2.3.3 Home 9

2.3.4 Creazione nota 10

2.4 Design procedurale 10

3 Implementazione 11

4 Test 11

4.1 Protocollo di test 11

4.2 Risultati test 12

4.3 Mancanze/limitazioni conosciute 12

5 Consuntivo 12

6 Conclusioni 12

6.1 Sviluppi futuri 12

6.2 Considerazioni personali 12

7 Glossario 12

8 Bibliografia 13

8.1 Bibliografia per articoli di riviste: 13

8.2 Bibliografia per libri 13

8.3 Sitografia 13

9 Allegati 13

# Introduzione

## Informazioni sul progetto

* Allievi: Andrea Casamatta, Paolo Comes, Matvej Rossi
* Supervisore: Geo Petrini
* Classe: 3BB SAM Trevano, sezione informatica
* Data di inizio progetto: 8.01.2025
* Data di consegna: 05.05.2025

## Abstract

È una breve e accurata rappresentazione dei contenuti di un documento, senza notazioni critiche o valutazioni. Lo scopo di un abstract efficace dovrebbe essere quello di far conoscere all’utente il contenuto di base di un documento e metterlo nella condizione di decidere se risponde ai suoi interessi e se è opportuno il ricorso al documento originale.

Può contenere alcuni o tutti gli elementi seguenti:

* **Background/Situazione iniziale**
* **Descrizione del problema e motivazione**: Che problema ho cercato di risolvere? Questa sezione dovrebbe includere l'importanza del vostro lavoro, la difficoltà dell'area e l'effetto che potrebbe avere se portato a termine con successo.
* **Approccio/Metodi**: Come ho ottenuto dei progressi? Come ho risolto il problema (tecniche…)? Quale è stata l’entità del mio lavoro? Che fattori importanti controllo, ignoro o misuro?
* **Risultati**: Quale è la risposta? Quali sono i risultati? Quanto è più veloce, più sicuro, più economico o in qualche altro aspetto migliore di altri prodotti/soluzioni?

Esempio di abstract:

*As the size and complexity of today’s most modern computer chips increase, new techniques must be developed to effectively design and create Very Large-Scale Integration chips quickly. For this project, a new type of hardware compiler is created. This hardware compiler will read a C++ program, and physically design a suitable microprocessor intended for running that specific program. With this new and powerful compiler, it is possible to design anything from a small adder, to a microprocessor with millions of transistors. Designing new computer chips, such as the Pentium 4, can require dozens of engineers and months of time. With the help of this compiler, a single person could design such a large-scale microprocessor in just weeks.*

## Scopo

Lo scopo del progetto CasaNote è quello di fornire un'applicazione web interattiva, che consenta all'utente di accedere, di creare, visualizzare e esportare note, dove si può oltre a scrivere, disegnare e registrare audio. (da finire in seguito)Analisi

## Analisi del dominio

CasaNote è un progetto ispirato a OneNote di Office, ma mira a colmare alcune lacune delle app attualmente disponibili, come Evernote, Notion o OneNote stesso. Anche se queste applicazioni sono molto utilizzate per la gestione delle informazioni, CasaNote si focalizza su esigenze specifiche di utenti come famiglie, studenti e professionisti, offrendo una soluzione semplice, sicura e rispettosa della privacy.

A differenza di altre app, CasaNote non richiede la raccolta di dati personali (come i cookie) e garantisce un accesso facile e gratuito. È progettata per essere utilizzata in vari contesti, come a casa, a scuola, al lavoro o in viaggio, con un'attenzione particolare alla sincronizzazione tra dispositivi e alla facilità di accesso. Gli utenti possono personalizzare i propri appunti e organizzare documenti e informazioni senza complicazioni.

CasaNote offre un'esperienza più user-friendly e un'attenzione particolare all'accessibilità e sicurezza dei dati.

## Analisi e specifica dei requisiti

|  |  |
| --- | --- |
| **ID: REQ-01** | |
| **Nome** | Sito web multiutente |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-02** | |
| **Nome** | Autenticazione |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Autenticazione con nome e password. |

|  |  |
| --- | --- |
| **ID: REQ-03** | |
| **Nome** | Registrazione |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | o Non è richiesta email  o Non è richiesta conferma dell’autenticità |
| **Sotto requisiti** | |
| **001** | captcha v3 |

|  |  |
| --- | --- |
| **ID: REQ-04** | |
| **Nome** | Utente gestisce blocco note |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |
| **Sotto requisiti** | |
| **001** | Scrivere del testo |
| **002** | Disegnare |
| **003** | Registrazione audio |
| **004** | Importare file di qualunque tipo |
| **005** | Esportare il blocco note come testo (txt), pdf o immagine (es: png) |

|  |  |
| --- | --- |
| **ID: REQ-05** | |
| **Nome** | Utente elimina blocco note |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |
| **Sotto requisiti** | |
| **001** | Conferma di eliminazione. |

|  |  |
| --- | --- |
| **ID: REQ-0** | |
| **Nome** | Sito deve essere responsive |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-07** | |
| **Nome** | Dati nel db tutti cifrati |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-08** | |
| **Nome** | Deploy/hosting |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Hosting che supporta Java 3.3.6 |

|  |  |
| --- | --- |
| **ID: REQ-09** | |
| **Nome** | Interfaccia semplice e intuitiva |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |
| **Sotto requisiti** | |
| **001** | Boostrap |

### Spiegazione elementi tabella dei requisiti:

**ID**: identificativo univoco del requisito

**Nome**: breve descrizione del requisito

**Priorità**: indica l’importanza di un requisito nell’insieme del progetto, definita assieme al committente. Ad esempio, poter disporre di report con colonne di colori diversi ha priorità minore rispetto al fatto di avere un database con gli elementi al suo interno. Solitamente si definiscono al massimo di 2-3 livelli di priorità.

**Versione**: indica la versione del requisito. Ogni modifica del requisito avrà una versione aggiornata.

Sulla documentazione apparirà solamente l’ultima versione, mentre le vecchie dovranno essere inserite nei diari.

**Note**: eventuali osservazioni importanti o riferimenti ad altri requisiti.

**Sotto requisiti**: elementi che compongono il requisito.

## Use case

Questo è lo Use Case che descrive il funzionamento della mia applicazione. Come si può vedere l’utente può visualizzare le sue note puo modificarle e eliminarle. La nota puo essere esportata in formato pdf o txt.

L’utente nella nota puo allegare file, disegnare o registrare audio, formattare il testo oltre a scriverlo.

Per fare tutto ciò l’utente deve aver prima effettuato l’autentiacazione, altrimenti non potrà usare il sito.



Figura 1 - Use Case

## Pianificazione

Il seguente Gantt è stato utilizzato per dare una indicazione massima delle attività da svolgere e delle loro tempistiche, siccome noi utilizziamo il metodo Agile.



Figura 2 - Gantt

## Analisi dei mezzi

### Software

* Html
* Boostrap
* Intellij IDEA
  + JavaSpring
    - Spring Web
    - Thymeleaf
    - Spring data JPA
    - H2 Database
    - Spring Validation
    - Spring Boot DevTools
    - Lombok (facoltativo)

### Hardware

Per lo sviluppo del sito web avremmo bisogno di 1 computer della scuola con mouse e tastiera, il computer ha le seguenti caratteristiche:

* Versione 22H2 (build SO 19045.4780)
* Windows: Windows 10 Education
* RAM: 32 GB
* CPU: Intel® Core™ i7-13700 CPU @ 2.10 GHz
* GPU: NVIDIA T400 4GB

# Progettazione

## Design dell’architettura del sistema



Figura 3 - Diagramma Architettura

L’architettura qua sopra descrive in linea di massima quanto e come dovrà essere implementata la architetta del nostro progetto. Come si può vedere dall’immagine c’è un server centrale che gestirà le varie richieste e comunicazioni tra il browser ed esso. All’interno del server abbiamo il database, e le varie cartella per la gestione e il funzionamento del sito, tra le quali i template, contenente i file html delle pagine, i controller per la gestione delle richieste, i repository per interfacciarsi con il database e poi le classi di dominio.

### Swimlanes



Figura 4 - Swimlanes registrazione

Questa swimlane è stata ideata per la registrazione di un nuovo utente. L’utente per fare l’accesso e poter usufruire del sito, dovrà prima essere registrato tramite nome utente e password. La password dovrà coincidere e il nome dovrà essere almeno di un carattere. Se la password non coincide allora verrà presentato un errore e l’utente ritorna alla pagina di registrazione. In caso di registrazione effettuata con successo l’utente visualizzerà la pagina home. Il captcha V3 verrà attivato alla pagina di registrazione per controllare che l’utente che ceca di registrarsi sia un umano e non un bot.



Figura 5 - Swimlanes login

Nello swimlane soprastante andiamo a analizzare il processo di login. Inizialmente l’utente interessato a logagrsi avrà già completato il form di registrazione. Nella pagina di login, l’utente procede ad inserire il suo nome utente e password, dopodiché cliccherà sul pulsante conferma o salva, e i dati verranno inviati al server he controllerà se corrispondono ad un utente salvato nel DB, se non viene trovata una corrispondenza verranno inviati degli errori che poi verranno visualizzati, e lune verrà riportato alla pagina di login, mentre in caso contrario l’utente verrà fatto entrare nel sito e visualizzerà la pagina home.



Figura 6 - Swimlanes creazione/salvataggio nota

In questo schema di swimlane viene mostrato come è stato pensato il salvataggio e creazione di una nota.

In questo schema abbiamo tre componenti, l’utente il FE (front-end) e il BE (back-end). L’utente inizierà creando una nuova nota che potrà successivamente riempirla con vari contenuti (testuali, audio, immagini o file). Se l’utente chiude la nota verrà fato un controllo se i dati sono salvati nel DB, e se non lo sono l’utente verrà chiesta la conferma di salvataggio. L’utente decide se salvarla o meno, se decide di no, allora torna a modificarla altrimenti la pagina della modifica della nota si chiude e l’utente visualizza la notifica del salvataggio della nota e torna alla home. Se l’utente mentre sta modificando la nota decide di cliccare sul pulsante salva allora il BE riceve i dati li cripta e li inserisce nel DB e l’utente viene notificato del salvataggio.

## Design dei dati e database



Figura 7 - Diagramma Classi

Sopra possiamo vedere come è stata progettata la gestione delle classi in questo progetto. A sinistra abbiamo la classe Note, che contiene vari attributi, tra cui l'id, che rappresenta la chiave primaria con cui verrà identificata ogni singola nota, il titolo (di tipo string), il contenuto (di tipo string), il formato (che può essere immagine o testo), l'audio\_file (nel caso si necessiti di salvare un audio), il data\_canvas (nel caso di un'immagine o disegni), la data\_creazione di tipo date (che indica la data in cui è stata creata la nota), e infine esportato, che è un attributo di tipo boolean e avrà valore true se la nota è stata esportata.

## Design delle interfacce

Queste interfacce non sono uguali a quelle effettive del progetto, ma sono simili.

### Registrazione utente

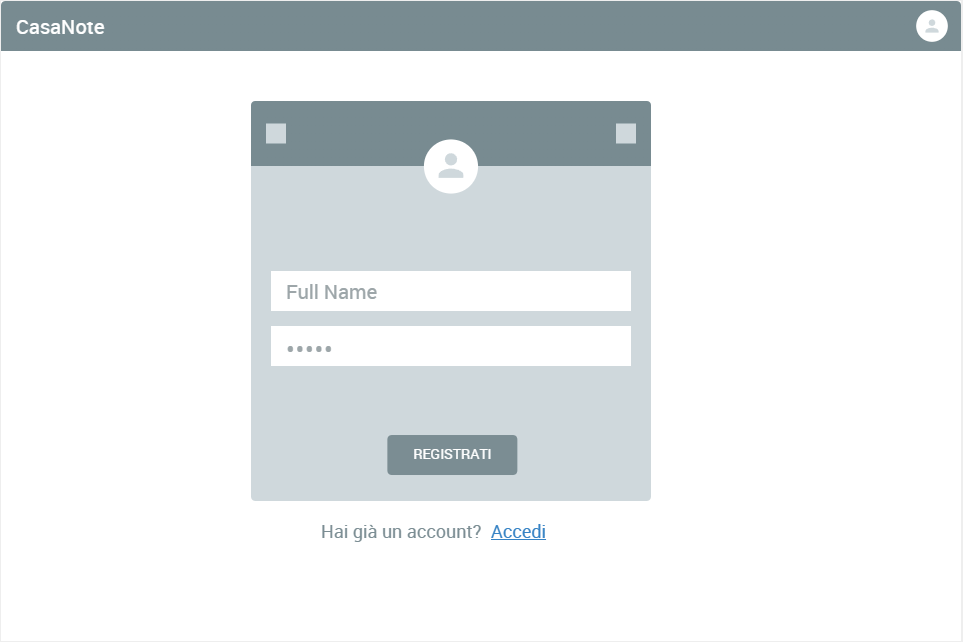


Figura 8 - Design registrazione utente

### Login utente

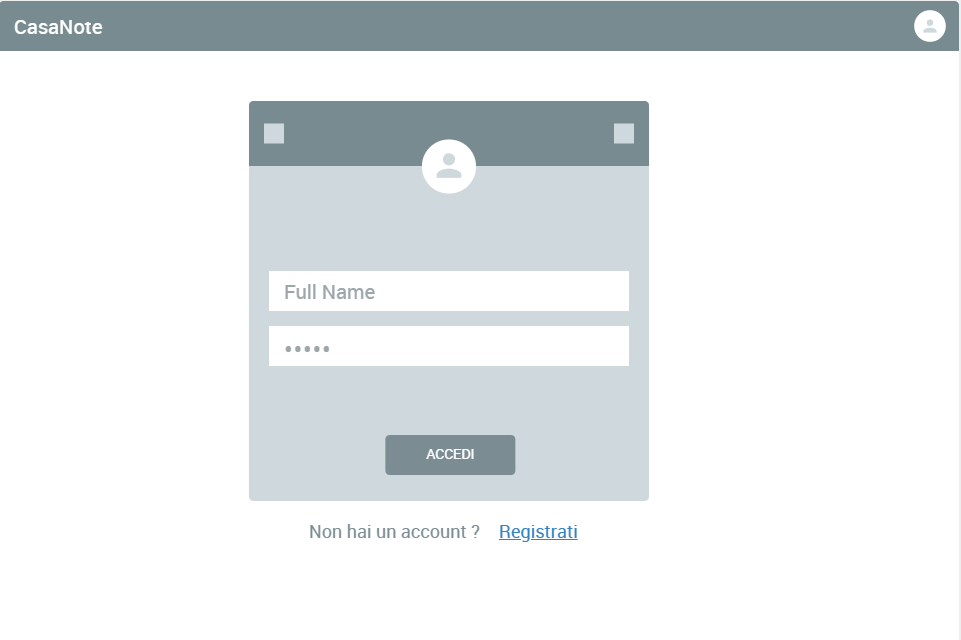


Figura 9 - Design login utente

### Home

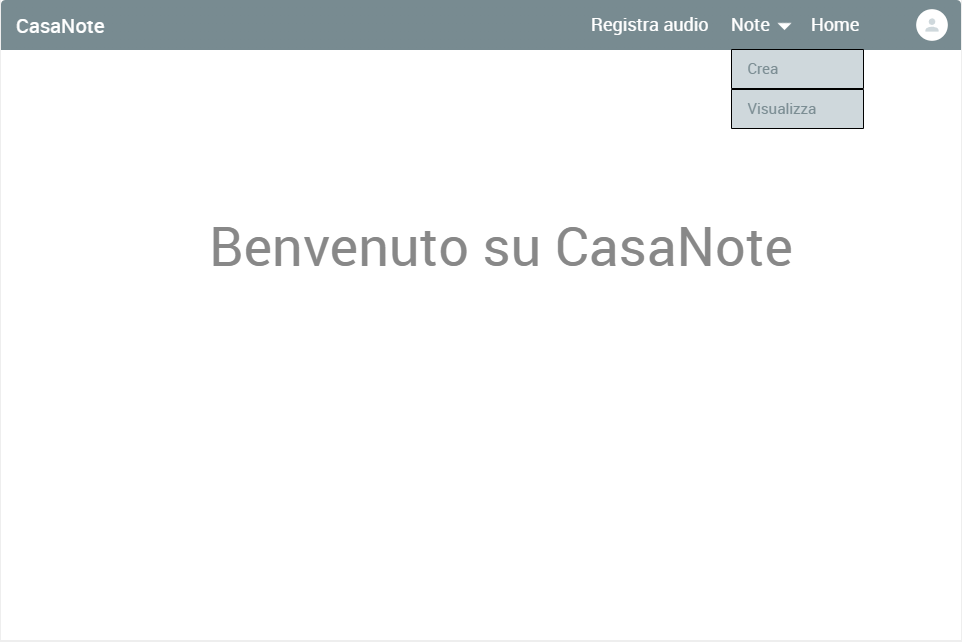


Figura 10 - Design home

### Creazione nota

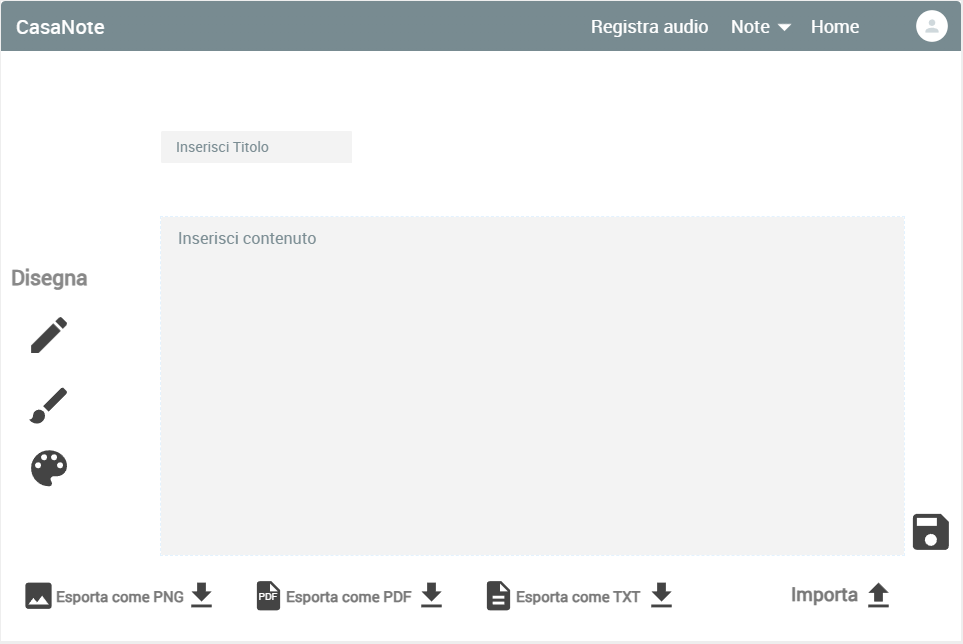


Figura 11 - Design creazione nota

### Visualizzazione note



Figura 12 - Design visualizzazione note

### Registrazione audio

## Design procedurale

Descrive i concetti dettagliati dell’architettura/sviluppo utilizzando ad esempio:

* Diagrammi di flusso e Nassi.
* Tabelle.
* Classi e metodi.
* Tabelle di routing
* Diritti di accesso a condivisioni …

Questi documenti permetteranno di rappresentare i dettagli procedurali per la realizzazione del prodotto.

# Implementazione

In questo capitolo dovrà essere mostrato come è stato realizzato il lavoro. Questa parte può differenziarsi dalla progettazione in quanto il risultato ottenuto non per forza può essere come era stato progettato.

Sulla base di queste informazioni il lavoro svolto dovrà essere riproducibile.

In questa parte è richiesto l’inserimento di codice sorgente - Print Screen - di maschere solamente per quei passaggi particolarmente significativi e/o critici.

Inoltre, dovranno essere descritte eventuali varianti di soluzione o scelte di prodotti con motivazione delle scelte.

Non deve apparire nessuna forma di guida d’uso di librerie o di componenti utilizzati. Eventualmente questa va allegata.

Per eventuali dettagli si possono inserire riferimenti ai diari.

# Test

## Protocollo di test

Definire in modo accurato tutti i test che devono essere realizzati per garantire l’adempimento delle richieste formulate nei requisiti. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-012 | **Nome:** | Import a card with KIC, KID and KIK keys, but not shown with the GUI |
| **Descrizione:** | Import a card with KIC, KID and KIK keys with no obfuscation, but not shown with the GUI | | |
| **Prerequisiti:** | Store on local PC: Profile\_1.2.001.xml (appendix n\_n) and Cards\_1.2.001.txt (appendix n\_n).  PIN (OTA\_VIEW\_PIN\_PUK\_KEY) and ADM (OTA\_VIEW\_ADM\_KEY) user right not set. | | |
| **Procedura:** | 1. Go to “Cards manager” menu,  in main page click “Import Profiles” link, Select the “1.2.001.xml” file, Import the Profile 2. Go to “Cards manager” menu,  in main page click “Import Cards” link, Select the “1.2.001.txt” file, Delete the cards,  Select the “1.2.001.txt” file, Import the cards 3. Research the “41795924770” Card, Click the imsi card link Check the card details 4. Execute the SQL: SELECT imsi, dir, keyset, cntr, rawtohex(kickey), rawtohex(kidkey), rawtohex(kikkey), rawtohex(chv), rawtohex(dap)FROM otacardkey a where imsi='340041795924770' ORDER BY keyset; | | |
| **Risultati attesi:** | Keys visible in the DB (OtaCardKey) but not visible in the GUI (Card details) | | |

## Risultati test

Tabella riassuntiva in cui si inseriscono i test riusciti e non del prodotto finale. Se un test non riesce e viene corretto l’errore, questo dovrà risultare nel documento finale come riuscito (la procedura della correzione apparirà nel diario), altrimenti dovrà essere descritto l’errore con eventuali ipotesi di correzione.

## Mancanze/limitazioni conosciute

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

# Consuntivo

Consuntivo del tempo di lavoro effettivo e considerazioni riguardo le differenze rispetto alla pianificazione (cap. 1.7) (ad esempio Gantt consuntivo).

# Conclusioni

Quali sono le implicazioni della mia soluzione? Che impatto avrà? Cambierà il mondo? È un successo importante? È solo un’aggiunta marginale o è semplicemente servita per scoprire che questo percorso è stato una perdita di tempo? I risultati ottenuti sono generali, facilmente generalizzabili o sono specifici di un caso particolare? ecc.

## Sviluppi futuri

Migliorie o estensioni che possono essere sviluppate sul prodotto.

## Considerazioni personali

Cosa ho imparato in questo progetto? ecc.

# Glossario

Inserite una semplice tabella con due colonne che spieghi i termini specifici del progetto (lista dei termini in ordine alfabetico A-Z)

Esempio:

|  |  |
| --- | --- |
| **Termine** | **Descrizione** |
| AJAX | **Asynchronous JavaScript And XML**: una tecnica che permette di eseguire richieste ed ottenere dati da una pagina web in modo asincrono. |
| CSS | **Cascading Style Sheets**: linguaggio che permette di definire il layout e la grafica di una pagina web. |

# Bibliografia

## Bibliografia per articoli di riviste:

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo dell’articolo (tra virgolette),
3. Titolo della rivista (in italico),
4. Anno e numero
5. Pagina iniziale dell’articolo.

## Bibliografia per libri

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo del libro (in italico),
3. ev. Numero di edizione,
4. Nome dell’editore,
5. Anno di pubblicazione,
6. ISBN.

## Sitografia

1. URL del sito (se troppo lungo solo dominio, evt completo nel diario),
2. Eventuale titolo della pagina (in italico),
3. Data di consultazione (GG-MM-AAAA).

**Esempio:**

* http://standards.ieee.org/guides/style/section7.html, *IEEE Standards Style Manual*, 07-06-2008.

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Istruzioni di installazione del prodotto (con credenziali di accesso) e/o di eventuali prodotti terzi
* Documentazione di prodotti di terzi
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o QdC
* Prodotto
* …