ZIQUIZ

# Indice

1 Indice 2

2 Introduzione 3

2.1 Informazioni sul progetto 3

2.2 Abstract 3

2.3 Scopo 4

3 Analisi 5

3.1 Analisi del dominio 5

3.2 Analisi e specifica dei requisiti 5

3.3 Use case 9

3.4 Pianificazione 10

3.5 Descrizione Gantt 11

3.5.1 Analisi 11

3.5.2 Progettazione 11

3.5.3 Implementazione 11

3.5.4 Test 11

3.5.5 Documentazione 11

3.6 Analisi dei mezzi 12

3.6.1 Software 12

3.6.2 Hardware 12

4 Progettazione 13

4.1 Design dei dati e database 13

4.2 Design dell’architettura del sistema 13

4.3 Design delle interfacce 16

4.4 Design procedurale 26

5 Implementazione 27

6 Test 27

6.1 Protocollo di test 27

6.2 Risultati test 29

6.3 Mancanze/limitazioni conosciute 29

7 Consuntivo 29

8 Conclusioni 31

8.1 Sviluppi futuri 31

8.2 Considerazioni personali 31

9 Bibliografia 32

9.1 Bibliografia per articoli di riviste: 32

9.2 Bibliografia per libri 32

9.3 Sitografia 32

10 Glossario 33

11 Indice delle figure 33

12 Allegati 33

# Introduzione

## Informazioni sul progetto

Allievo: Robin Sartore

Docente: Michel Palucci

Scuola: SAM Trevano

Sezione: Informatica

Modulo:306

Data inizio:11.09.2024

Data fine: 18.12.2024

## 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 consiste nella creazione di un’applicazione per un quiz di nome Ziquiz, con varie modalità di domande, difficoltà, e tipologie e si può usare per creare un quiz personalizzato e nel caso usarlo per studiare si visualizzerà anche un proprio punteggio per le varie categorie e difficoltà e verrà salvato localmente un top score.

Un altro scopo è quello di iniziare a fare progetti dato che l’anno prossimo dovrò fare il LPI e quindi è meglio iniziare a prepararsi prima per arrivare al meglio preparati e con abbastanza esperienza per riuscire a fare un buon progetto.

# Analisi

## Analisi del dominio

Il prodotto potrà essere usato come svago nel tempo libero e sfidare gli amici a chi fa il punteggio più alto e vedere chi ha più conoscenze sulle varie categorie o un allenamento per studiare una qualsiasi materia o formazione con domande personalizzate, chiaramente esistono già vari quiz online su cose diverse ma questo si differenzia dagli altri perché oltre ad avere una modalità studio i quiz sono su varie categorie con varie difficoltà e mi sembra molto innovativo.

Gli utenti potranno migliorare e sviluppare una conoscenza su vari argomenti e migliorarsi ogni volta superando il loro top score.

I quiz che ci sono già nel web sono pieni di animazioni e effetti quindi devo provare ad avere anch’io qualche animazione e non qualcosa di monotono.

## Analisi e specifica dei requisiti

|  |  |
| --- | --- |
| **Req-01** | |
| **Nome** | Creazione interfaccia Menu |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Comprensibile all’utente |

|  |  |
| --- | --- |
| **Req-02** | |
| **Nome** | Possibilità di scegliere la tipologia di quiz |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Comprensibile all’utente |

|  |  |
| --- | --- |
| **Req-03** | |
| **Nome** | Possibilità di creare un quiz |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Comprensibile all’utente |

|  |  |
| --- | --- |
| **Req-04** | |
| **Nome** | Possibilità di scegliere la difficolta del quiz |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Comprensibile all’utente |

|  |  |
| --- | --- |
| **Req-05** | |
| **Nome** | Possibilita di vedere il Topscore |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Comprensibile all’utente |

|  |  |
| --- | --- |
| **Req-06** | |
| **Nome** | Studiare su quiz creati |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Comprensibile all’utente |

|  |  |
| --- | --- |
| **Req-07** | |
| **Nome** | Musica di sottofondo |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Comprensibile all’utente |

|  |  |
| --- | --- |
| **Req-08** | |
| **Nome** | Visualizzare se si ha sbagliato |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Comprensibile all’utente |

|  |  |
| --- | --- |
| **Req-09** | |
| **Nome** | Aggiunta animazioni |
| **Priorità** | 2 |
| **Versione** | 1.0 |

|  |  |
| --- | --- |
| **Req-010** | |
| **Nome** | Scegliere una modalità di studio |
| **Priorità** | 2 |
| **Versione** | 1.0 |

## Use case

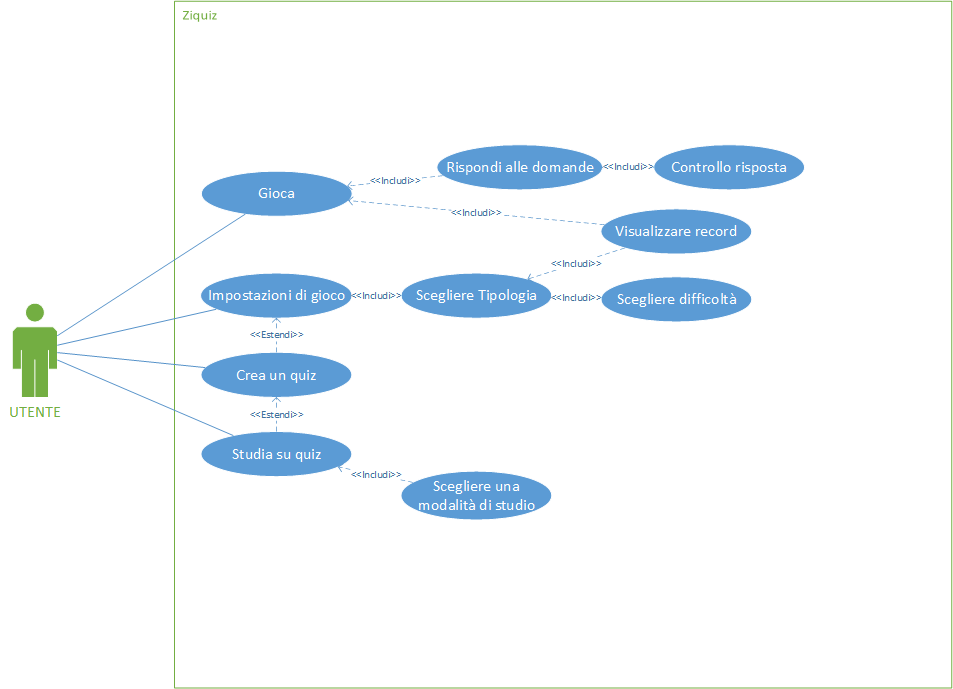


Figura use case

Questo è lo schema delle use case del progetto. In pratica l’utente può scegliere le impostazioni del quiz, la tipologia e la difficolta di cui può vedere il suo record in quella specifica modalità, può anche creare un quiz e studiare sul quiz con varie diverse modalità di studio e può anche giocare rispondendo alle domande e visualizzando gli errori.

## Pianificazione

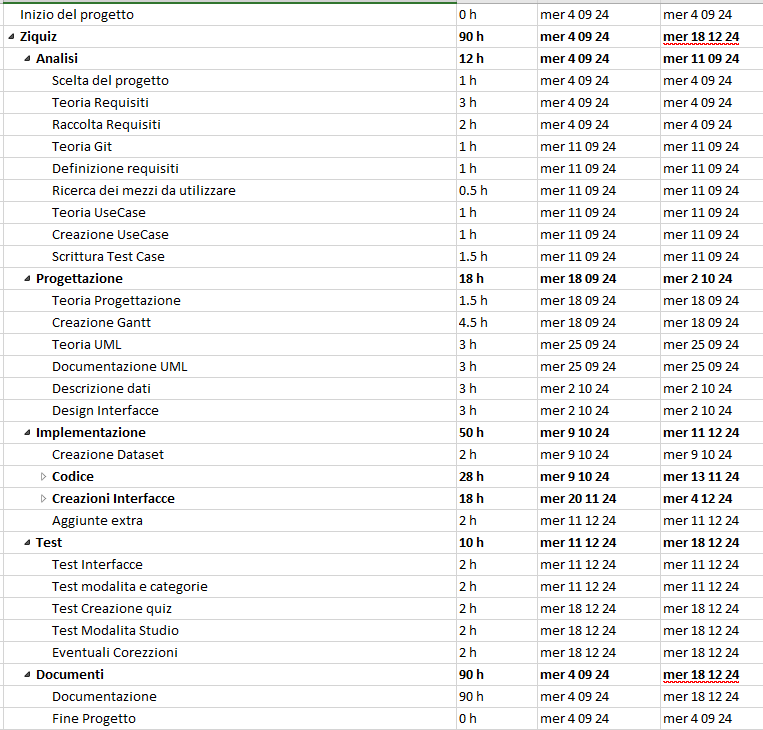


Figura Esempio di diagramma di Gantt

Questo è il Gantt di 90 ore del mio progetto, ho applicato una metologia waterfall.

## Descrizione Gantt

### Analisi

Per l’analisi ho programmato 2 giornate lavorative da 6 ore includendo la teoria fatta a lezione e ho ipotizzato dei tempi per le varie parti.

### Progettazione

Per la progettazione ho calcolato più ore della Analisi, ho messo una lunga tempistica soprattutto per la costruzione del gantt.

### Implementazione

L’implementazione è la parte più importante del progetto per questo ho previsto 50 ore di lavoro dato che devo ancora bene capire come leggere le domande e come crearle.

### Test

Per i test ho previsto 10 ore per i vari test e le eventuali correzioni ho preso più di un giorno perché non riuscirei magari a correggere molti errori in un giorno.

### Documentazione

Per la documentazione ho calcolato la durata intera quindi 90 ore anche se in realtà prenderà il 305 del tempo ma verrà fatta man mano.

## Analisi dei mezzi

Per la realizzazione del progetto è stato utilizzato un PC scolastico con Visual Studio 2022

Elencare e descrivere i mezzi disponibili per la realizzazione del progetto. Ricordarsi di sempre descrivere nel dettaglio le versioni e il modello di riferimento.

### Software

Visual studio 2022

Hosting servito dalla scuola

### Hardware

Computer fornito dalla scuola:

32 gb di RAM

Windows 10 Education 22H2

Processore 13th Gen Intel(R) Core(TM) i7-13700 2.10 GHz

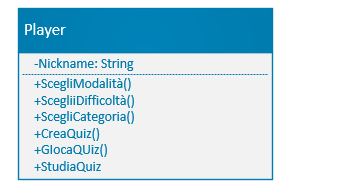
# Progettazione

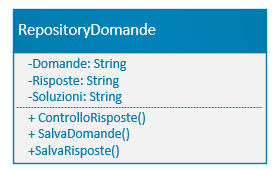
## Design dei dati e database

L’architettura del sistema si basa su un file json in cui ci saranno le domande e le risposte per le domande e un altro file in cui verrano salvate localmente le domande e le risposte del utente

## Design dell’architettura del sistema

Diagramma di classi





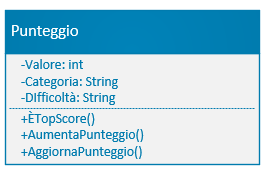


Figura 6 Diagramma di classi

## Design delle interfacce

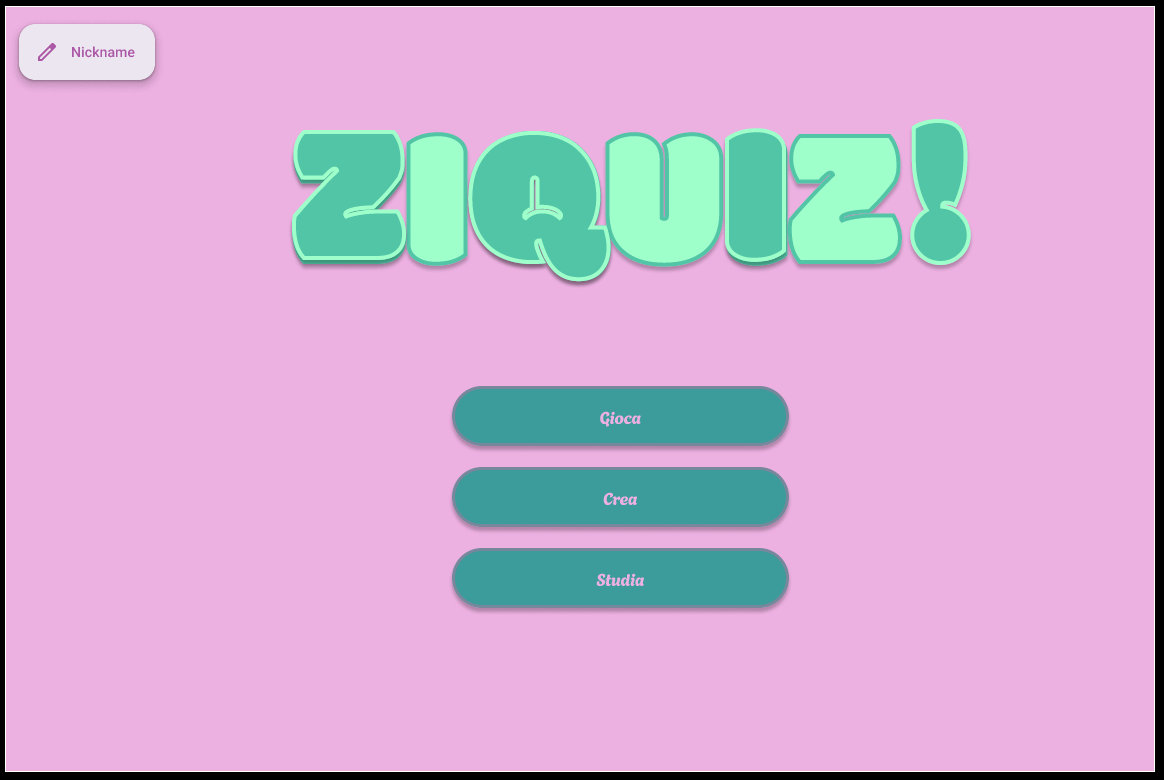
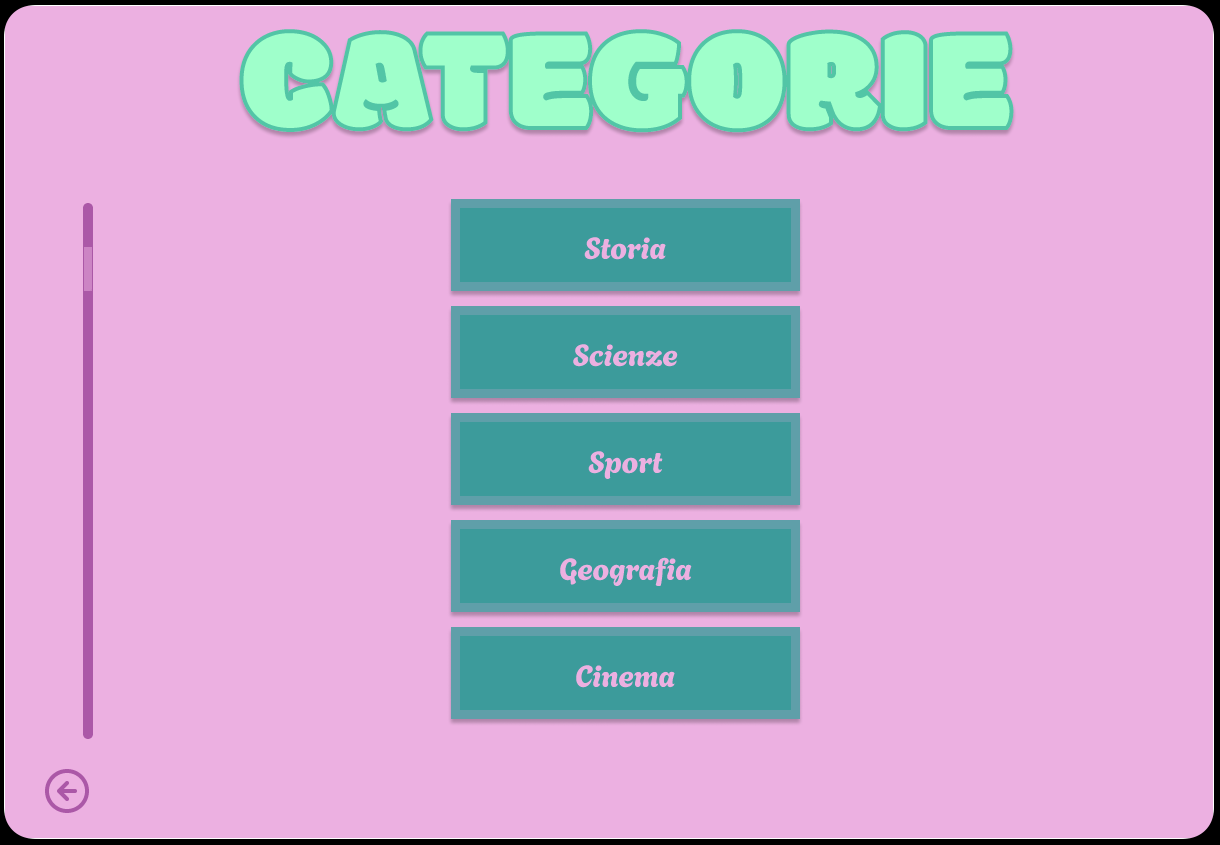


Figura 6 Interfaccia menù

Questa è l’interfaccia del menu principale qui l’utente potrà scegliere cosa fare e inserire il suo nickname, e si può leggere il titolo dell’applicazione.

Per vedere quali colori stavano meglio insieme ho usato questo sito [link](https://color.adobe.com/it/create/color-wheel).



Figuura 7 Categoria

Qua si potrà selezionare la categoria che più ci piace, si potrà scegliere anche un quiz creato dall’utente.



Figura 8 difficoltà

Qui invece si sceglierà la difficoltà e si potrà vedere il topscore in base a ogni difficoltà.



Figura 9 esempio domanda

In Questa interfaccia si può vedere come viene mostrato il quiz con un punteggio in alto a destra



Figura 10 esempio errore

Qua si può vedere come viene mostrato un errore e la risposta correttà.

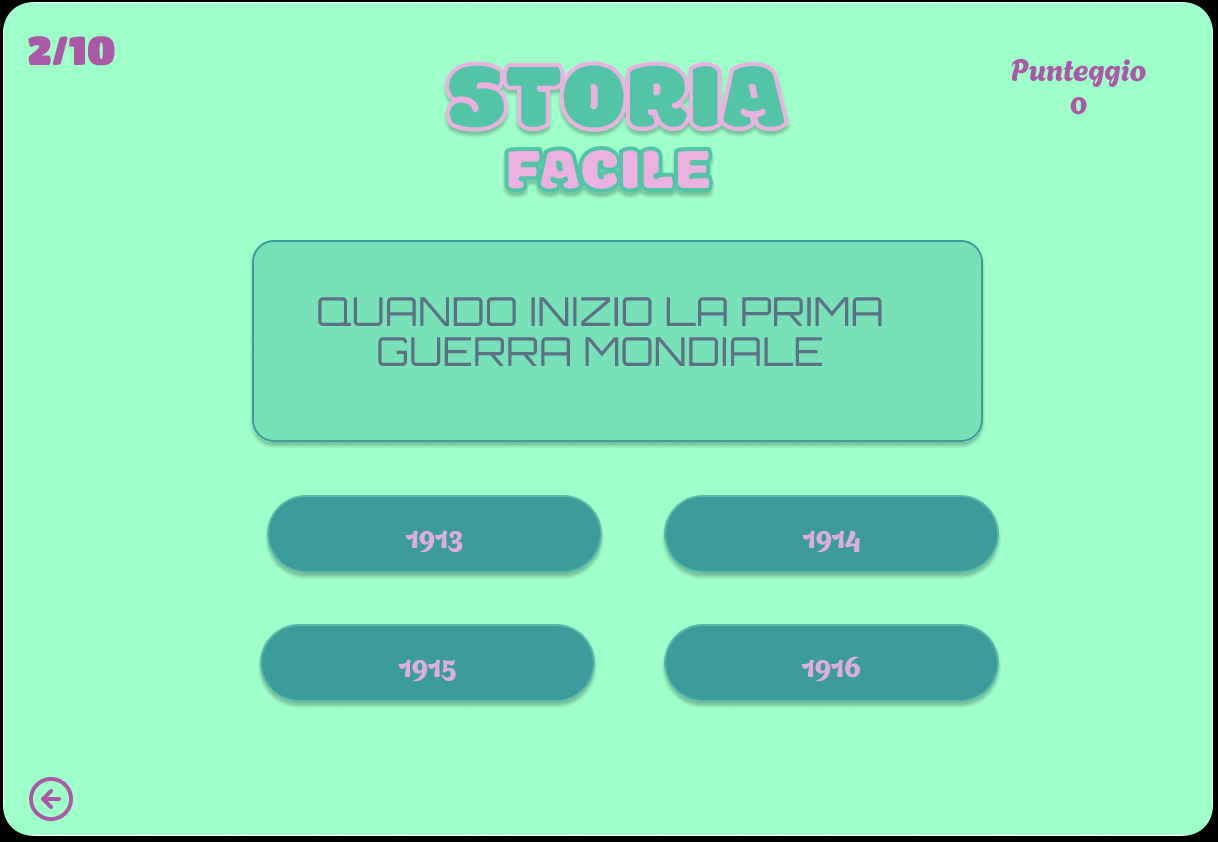
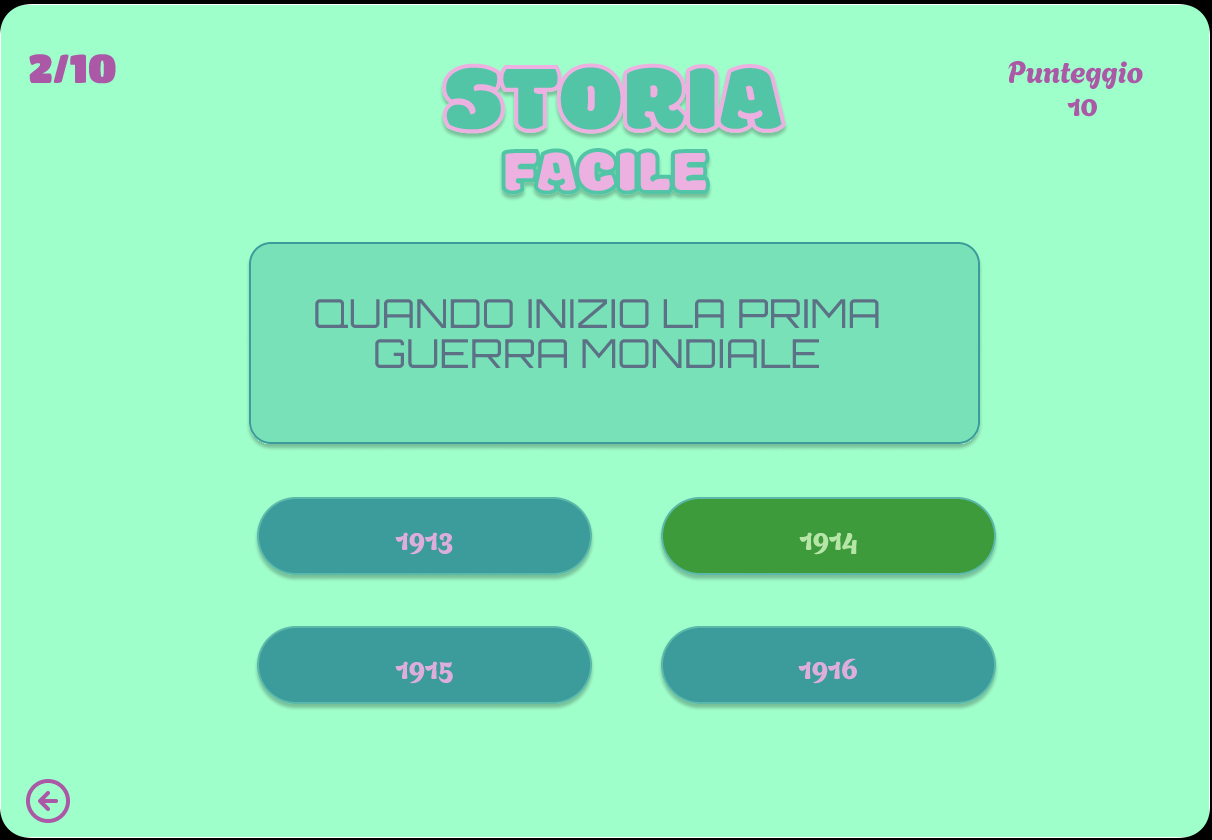


Figura 11 esempio domanda 2

Questa è un'altra domanda e si può vedere in alto a sinistra che è stat modificato il counter delle domande.

Figura 12 Esempio correzione 2

In questa interfaccia si può vedere invece quando viene inserita una risposta corretta , e si vede anche il punteggio aumentare.



Figura 13 Uscita da quiz

Qui viene mostrato il messaggio di quando si vuole uscire dal quiz

Figura 14 Modalità Crea

Qua si vede la schermata principale in cui si vedono i vari dati da inserire per creare un quiz.

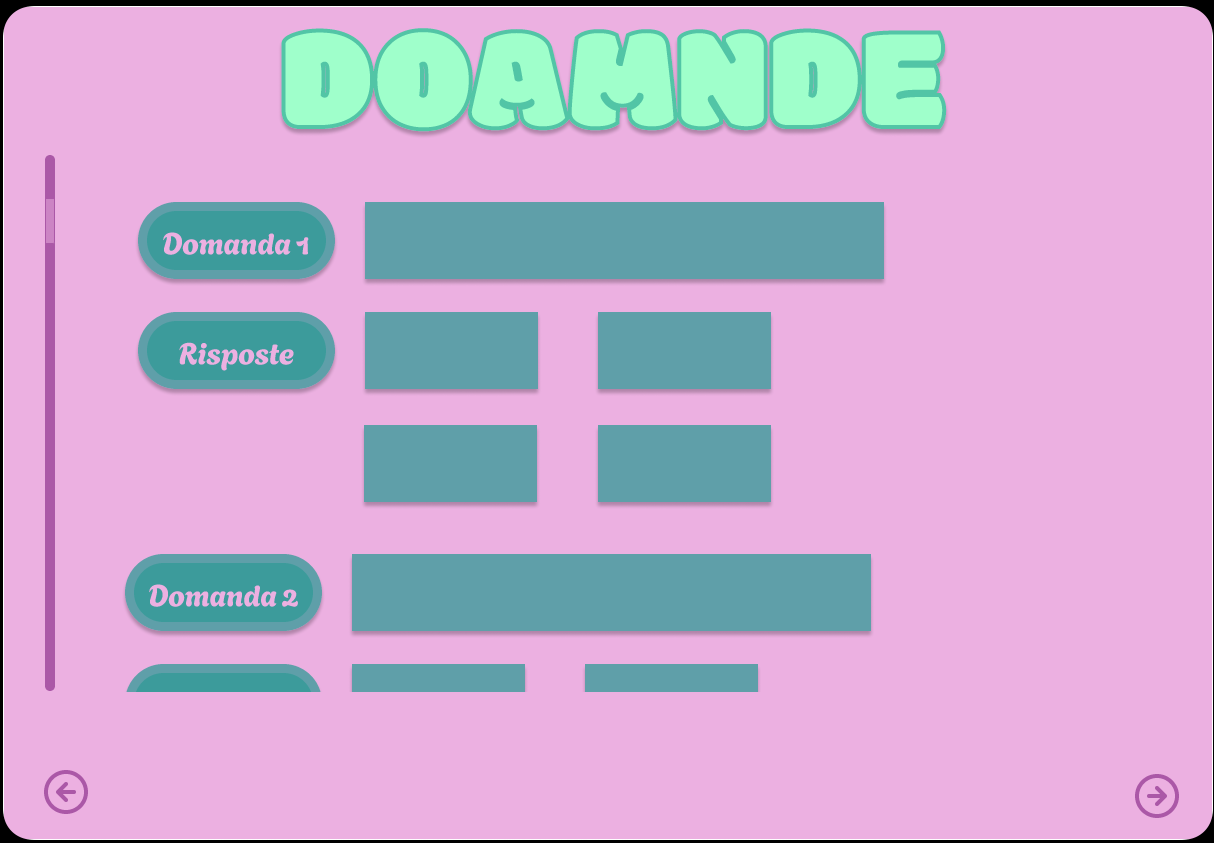


Figura 15 domande modalita crea

Qua invece viene mostrata la schermata in cui si selezioneranno le domande e le risposte da inserire nel nostro quiz personalizzato.

## Design procedurale

Diagramma Swimlanes

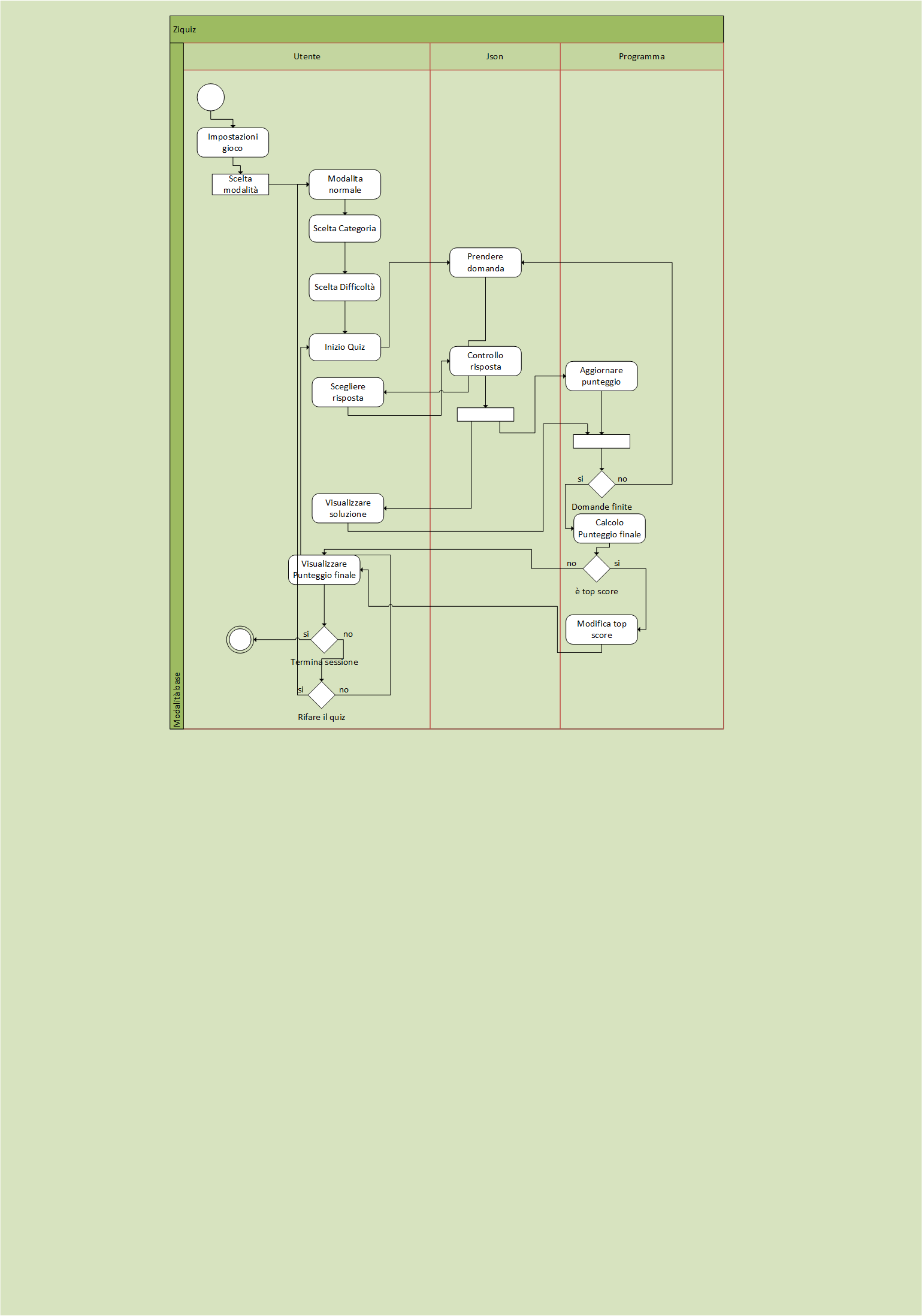


Figura Diagramma Swimlanes base

Diagramma Swimlanes 2

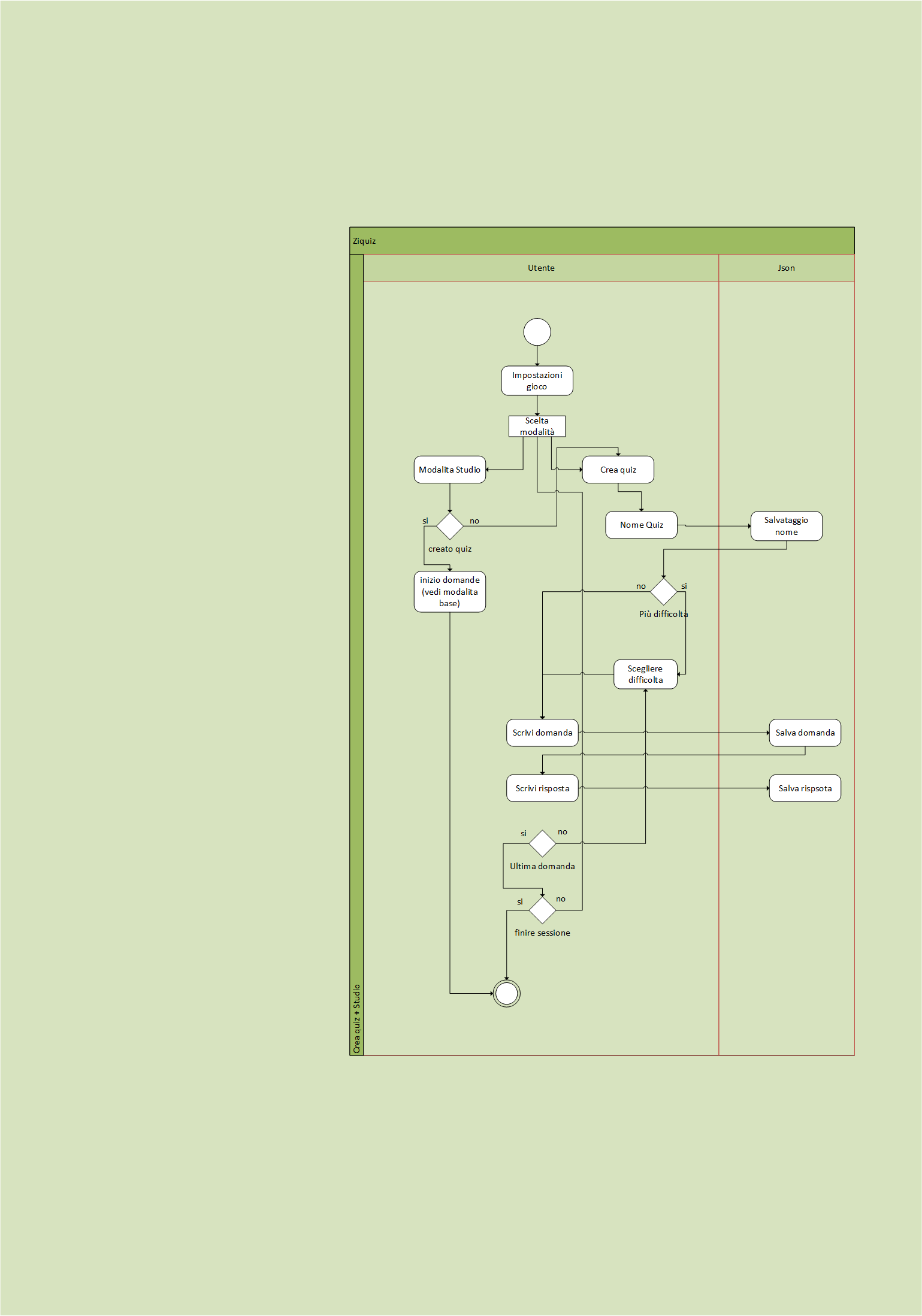


Figura Diagramma Swimlanes (studio + creaquiz)

# Implementazione

## Menù

Per ora il menù lo strutturato così:

<!--BUG:

-->

<!DOCTYPE HTML>

<html lang="it">

<head>

  <meta charset="UTF-8">

  <meta name="description" content="Quiz">

  <meta name="keywords" content="Quiz">

  <meta name="author" content="Robin Sartore">

  <title>ROBIN SARTORE</title>

  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH" crossorigin="anonymous">

  <link href="menu.css" rel="stylesheet" type="text/css">

  <link rel="icon" type="image/x-icon" href="image/favicon.ico">

<script src="menu.js"></script>

<script src="nickname.js"></script>

</head>

<body onload="refresh()">

    <div id="nicknamefinal" class="nickname">nuc</div>

    <div class="titolo">

        <SPAN>Z<D>I</D>Q<D>U</D>I<D>Z</D>!</SPAN>

    </div>

    <button class="nicknamebutton" onclick="nickname()" >nickname</button>

    <button class="gioca" onclick="gioca()">GIOCA</button>

</body>



## Nickname:

Per il nickname ho usato le sessioni per salvarmelo, lo salvo su una variabile locale e poi quando viene ricaricata una nuova pagina il nome si inserirà da solo.

let nick;

function insert(){

    const prendinickname = document.getElementById("insertnickname").value;

    if(prendinickname=="")

        {

            alert("Inserire nickname");

         }

         else{

            sessionStorage.setItem("1",prendinickname);

            nick = sessionStorage.getItem("1");

            close();

            var cat=window.open("Ziquiz.html");

         }

}

    function refresh(){

    nick = sessionStorage.getItem("1");

    document.getElementById("nicknamefinal").innerHTML= nick;

}

## Tasto back

Per il tasto back ho inserito un png e quando viene cliccato torna alla pagina precedente

function back(){

    close();

    var pag=window.open("Ziquiz.html");

}

## Shuffle

Per avere le risposte in modo casuale ho creato un array da uno a quattro e ho associato i numeri alle classi dei bottoni contenenti le risposte per assegnarlo sempre casuale ho creato un metodo mescola che mescola l’array

function mescola(arr) {

    arr.sort(() => Math.random() - 0.5);

    }

## Mandare i dati a php

Al posto del Top score ho deciso di fare un intera classifica.

Per creare la leaderboard mando i dati del nickname e del punteggio del giocatore e la difficolta e la categoria per sapere in quale leaderboard salvarlo.

Per passare questi dati ho usato un fetch

function salvaPunteggio() {

       // Dati da inviare

       const dati = {

        nome: nickname,

        punteggio: punti,

        categoria: categoria,

        difficolta: difficolta

    };

    // Invia i dati a PHP usando fetch

    fetch('../php/record.php', {

        method: 'POST', // metodo di invio

        headers: {

            'Content-Type': 'application/json'  // Tipo di contenuto

        },

        body: JSON.stringify(dati)  // Dati da inviare, convertiti in formato JSON

    })

    .then(response => response.json())  // Assumiamo che il server ritorni una risposta JSON

    .then(data => {

        console.log('Dati salvati con successo', data);

    })

    .catch(error => {

        console.error('Errore nella richiesta:', error);

    });

}

## PHP salvataggio record

Questo è il codice php che uso per prendere i nuovi dati salvarli nel json giusto e fa anche in modo che i dati vengano messi in base al punteggio e possono essere presi.

<?php

// Abilitare la visualizzazione degli errori per il debug

ini\_set('display\_errors', 1);

error\_reporting(E\_ALL);

// Impostazioni per evitare caching

header("Cache-Control: no-store, no-cache, must-revalidate, max-age=0");

header("Pragma: no-cache");

header("Expires: Thu, 01 Jan 1970 00:00:00 GMT");

// Funzione per ottenere il percorso del file JSON

function getFilePath($categoria, $difficolta) {

    return \_\_DIR\_\_ . '/../json/record/' . $categoria . '\_' . $difficolta . '\_dati.json';

}

// Se è una richiesta POST (per aggiungere punteggi)

if ($\_SERVER['REQUEST\_METHOD'] == 'POST') {

    $data = json\_decode(file\_get\_contents('php://input'), true); // Prende i dati dal corpo della richiesta

    // Controlla se i dati necessari sono presenti

    if ($data && isset($data['nome']) && isset($data['punteggio']) && isset($data['categoria']) && isset($data['difficolta'])) {

        $nome = $data['nome'];

        $punteggio = $data['punteggio'];

        $categoria = $data['categoria'];

        $difficolta = $data['difficolta'];

        // Percorso del file JSON

        $filePath = getFilePath($categoria, $difficolta);

        // Crea la directory se non esiste

        $dirPath = dirname($filePath);

        if (!is\_dir($dirPath)) {

            mkdir($dirPath, 0777, true);

        }

        // Crea il file se non esiste

        if (!file\_exists($filePath)) {

            $created = file\_put\_contents($filePath, json\_encode([]));  // Crea un file vuoto

            if ($created === false) {

                echo json\_encode(['status' => 'error', 'message' => 'Errore nella creazione del file']);

                exit;

            }

        }

        // Leggi i dati esistenti

        $datiCorrenti = json\_decode(file\_get\_contents($filePath), true);

        if (!is\_array($datiCorrenti)) {

            $datiCorrenti = [];

        }

        // Aggiungi il nuovo punteggio al file JSON

        $datiCorrenti[] = [

            'nome' => $nome,

            'punteggio' => $punteggio

        ];

        // Salva i dati nel file JSON

        $result = file\_put\_contents($filePath, json\_encode($datiCorrenti, JSON\_PRETTY\_PRINT));

        // Risposta JSON

        if ($result === false) {

            echo json\_encode(['status' => 'error', 'message' => 'Errore nella scrittura del file']);

        } else {

            echo json\_encode(['status' => 'success', 'message' => 'Punteggio salvato correttamente']);

            exit;

        }

    } else {

        // Se i dati sono incompleti

        echo json\_encode(['status' => 'error', 'message' => 'Dati mancanti o non validi']);

    }

}

?>

## Login

Per avere un unicita dei record ho deciso di fare un login cosi ogni utente puo vedere i propri record e accedere da piu dispositivi. Per fare ciò ho usato Firebase.

### Firebsae

Firebase è una piattaforma sviluppata da google che permette ai suoi utenti di creare app mobile e web.

### Motivo scelta

Ho scelto di utilizzare Firebase perché rende disponibile un database i cuoi potevo salvare gli utenti ed è facilmente usabile per esempio mettere un recupero della password

### Configurazione

Dopo aver fatto il login con l’account di google su firbase ho collegato il mio progetto per farlo ho dovuto mettere questa configurazione in un file js nel mio applicativo:

// Firebase configuration

const firebaseConfig = {

  apiKey: "AIzaSyBVAPvEBg-9KnKMgrE5cFqxZrQiaCIlGsU",

  authDomain: "ziquiz-cb376.firebaseapp.com",

  databaseURL: "https://ziquiz-cb376-default-rtdb.europe-west1.firebasedatabase.app/",  // Aggiungi questa riga

  projectId: "ziquiz-cb376",

  storageBucket: "ziquiz-cb376.firebasestorage.app",

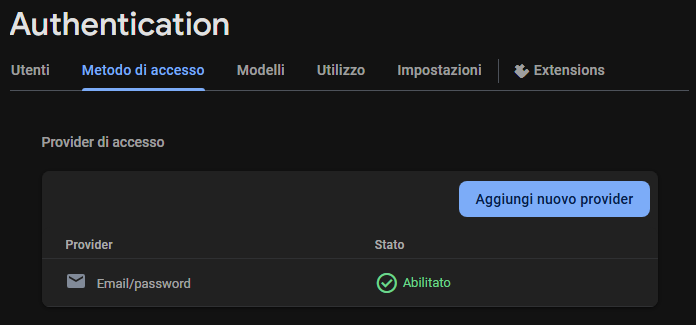
  messagingSenderId: "971536143420",

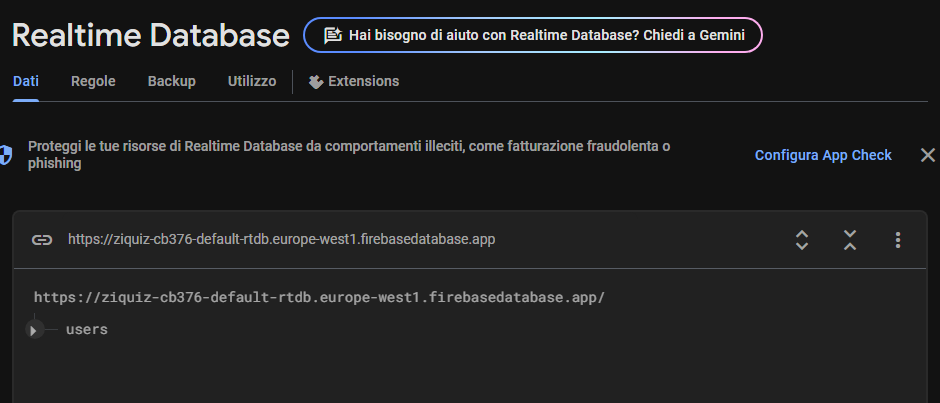
  appId: "1:971536143420:web:9534385c390bc939339f39",

  measurementId: "G-G3W2P5JLVZ"

};

poi ho dovuto creare un database Realtime e configurare come metodo di accesso Email/password





# 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-001 | **Nome** | Verifica funzionalità Menu |
| **Descrizione** | Si controlla se il menu sia utilizzabile | | |
| **Prerequisiti** | - | | |
| **Procedura** | 1. Accedere al sito 2. Premere bottone gioca   3.tornare indietro | | |
| **Risultati attesi** | Il bottone del menu funziona e ti fa andare avanti e poi si può tornare indietro | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-002  REQ-001 | **Nome** | Verifica funzionalità nickname |
| **Descrizione** | Si controlla se il nickname si imposta | | |
| **Prerequisiti** | - | | |
| **Procedura** | 1. Accedere al sito 2. Premere il bottone nickname e impostare un nuovo nickname | | |
| **Risultati attesi** | Una volta impostato il nickname | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-003  REQ-002 | **Nome** | Verifica funzionalità scelta categorie |
| **Descrizione** | Si controlla se è possibile scegliere tra le varie categorie | | |
| **Prerequisiti** | - | | |
| **Procedura** | 1. Accedere al sito 2. Premere gioca 3. Premere su una categoria 4. Tornare indietro e premerne un altra | | |
| **Risultati attesi** | L’utente potrà scegliere varie categorie e tornare indietro quando vuole | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-004  REQ-004 | **Nome** | Verifica funzionalità scelta difficoltà |
| **Descrizione** | Si controlla se è possibile scegliere tra le varie categorie | | |
| **Prerequisiti** | - | | |
| **Procedura** | 1. Accedere al sito 2. Premere gioca 3. Premere su una categoria 4. Scegliere la difficoltà 5. Tornare indietro | | |
| **Risultati attesi** | L’utente potrà scegliere varie difficoltà e tornare indietro al menù | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-005  REQ-006 | **Nome** | Verifica corezzione domanda |
| **Descrizione** | Si controlla se si visualizza la corezzione | | |
| **Prerequisiti** | - | | |
| **Procedura** | 1. Accedere al sito 2. Premere gioca 3. Premere su una categoria 4. Scegliere una difficoltà facile 5. Sbagliare apposta una domanda | | |
| **Risultati attesi** | Si dovrà visualizzare l’errore | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-006  REQ-005 | **Nome** | Verifica funzionalità top score |
| **Descrizione** | Si controlla se il top score viene salvato | | |
| **Prerequisiti** | - | | |
| **Procedura** | 1. Accedere al sito 2. Premere gioca 3. Premere su una categoria 4. Scegliere una difficoltà 5. Finire il quiz 6. Tornare alla pagina dela scelta delle dificoltà | | |
| **Risultati attesi** | Ci dovrà essere salvato il top score | | |

## 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.

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Risultato ottenuto** | **Stato** |
| TC-001 | Testo o immagine a conferma del test | Passato |
| TC-002 | Testo o immagine a conferma del test | Fallito |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## 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).

Stile orizzontale



CONSUNTIVO

Figura Esempio di diagramma di 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.

# 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.

# Glossario

|  |  |
| --- | --- |
| **Termine** | **Significato** |
| ASCII | American Standard Code for Information Interchange, codice per la codifica di caratteri. |
|  |  |
|  |  |
|  |  |
|  |  |
| ZFS | Zettabyte File System, filesystem opernsource a 128 bit con funzionalità avanzate. |

# Indice delle figure

[Figura 1 Esempio di diagramma di Gantt 7](#_Toc124775984)

[Figura 2 Esempio di diagramma di Gantt 8](#_Toc124775985)

[Figura 3 Diagramma ER Chen 10](#_Toc124775986)

[Figura 4 Diagramma ER Barker 11](#_Toc124775987)

[Figura 5 Esempio di diagramma di Gantt consuntivo 16](#_Toc124775988)

# 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
* …