UNIVERSITÀ
DI PARMA

Dipartimento di Scienze Matematiche, Fisiche e Informatiche

Corso di Laurea [Triennale in Informatica]

Titolo in Italiano

Titolo in Inglese

Candidato:
Dennis Turco

Relatore:
Prof. Roberto Alfieri

Correlatori:
Prof. co-Nome co-Cognome
Prof. co-Nome2 co-Cognome2

Anno Accademico 2020–2023

Indice

Elenco delle figure

Listings

Introduzione

Nel seguente elaborato tratterò una relazione relativa al progetto di tirocinio svolto presso un'azienda software house "ISolutions" di Noceto.

Il progetto si concentra sulla creazione di una piattaforma di e-learning per la gestione del processo di OnBoarding aziendale. L'obiettivo è di fornire ai nuovi dipendenti un'esperienza guidata, strutturata e personalizzata attraverso la piattaforma web.

Attualmente, l'azienda gestisce tra 20 e 30 processi di OnBoarding dei nuovi dipendenti ogni anno, grazie al personale HR, utilizzando un foglio di calcolo Excel per la gestione dei dati e del completamento dei vari task. Tuttavia questo metodo si basa su un funzionamento manuale di aggiornamento dei task completati dai vari dipendenti e richiede una supervisione umana costante, il che lo rende inefficiente e non scalabile. Inoltre, le informazioni raccolte attraverso il modulo post-OnBoarding non sono sempre esaustive e dettagliate. A causa dell'espansione dell'azienda, questo sistema sta diventando sempre più inefficiente e rischioso. Per questo motivo risulta importante l'implementazione di un software dedicato al processo di OnBoarding dei dipendenti, con lo scopo di garantire una riduzione del rischio di errori e di migliorare l'esperienza generale. Inoltre, il software deve permettere di automatizzare alcune attività ripetitive, liberando tempo prezioso per il team HR, che potrà concentrarsi su attività di maggiore valore aggiunto per l'azienda. Infine, il nuovo sistema di OnBoarding ha la necessità di consentire l'acquisizione e l'archiviazione in maniera più sicura e organizzata dei dati dei dipendenti, rispettando le normative sulla privacy e semplificando le attività di audit interno.

Il processo di OnBoarding avrà una durata di circa un mese per uno sviluppatore junior e due settimane per uno sviluppatore senior. La piattaforma guiderà i nuovi dipendenti attraverso i vari task previsti in modo strutturato e personalizzato, fornendo loro gli strumenti necessari per acquisire le cono-

scenze e le competenze richieste per il loro ruolo.
Inoltre, la piattaforma fornirà un modulo per la raccolta di feedback post-OnBoarding. Il modulo sarà strutturato in modo da raccogliere informazioni dettagliate e utili per migliorare continuamente il processo di OnBoarding. Questo feedback sarà utilizzato per aggiornare e migliorare costantemente la piattaforma.
Infine, poiché l'azienda è certificata ISO 9001 (certificazione della qualità), la piattaforma fornirà un'ulteriore valutazione del processo di OnBoarding. Dopo che il dipendente ha completato il processo, verrà chiesto di esprimere un giudizio attraverso un modulo dedicato. Ciò permetterà all'azienda di comprendere se il processo di OnBoarding sta funzionando correttamente e se ci sono aree che possono essere migliorate.
In sintesi, la piattaforma di e-learning per la gestione del processo di OnBoarding aziendale rappresenta un'importante innovazione per l'azienda. Grazie alla sua efficienza e scalabilità, la piattaforma migliorerà l'esperienza del dipendente e ridurrà il carico di lavoro del personale HR. Inoltre, la raccolta di feedback dettagliati post-OnBoarding e la valutazione attraverso il modulo dedicato forniranno all'azienda le informazioni necessarie per migliorare costantemente il processo di OnBoarding.

Il processo pricincipale di OnBoarding aziendale deve prevedere le seguenti fasi.

- Introduzione (˜10 tasks);
- Setup della work station (˜20 tasks);
- Documentazione, sia amministrativa che tecnica (˜30 tasks);
- Video introduttivi, tecnici e orientati ai prodotti sviluppati (˜10 tasks);
- Overview dell'organizzazione aziendale e degli strumenti utilizzati (˜10 tasks);
- Hands On degli applicativi aziendali (˜20 tasks);
- Formazione sui principali linguaggi di programmazione utilizzati (˜20 tasks);
- Overview struttura codice delle soluzioni software (˜10 tasks);
- Test finale con revisione e valutazione.

Sono previsti inoltre alcuni step preliminari, in carico al team HR, per la predispozione di quanto necessario (creazione utenza, preparazione pc etc. . .), e una fase finale di retrospettiva per raccogliere feedback riguardo il processo dal neo assunto.

Alcune desiderate del progetto:

- Integrazione autenticazione con modulo di login e gestione livelli di autorizzazione;
- Pannello amministrativo per il team HR per gestire (creazione, modifica, eliminazione) dei vari tasks;
- Tracciamento tempo impiegato sui vari task e reportistica per team HR;
- Possibilità di avviare, sospendere e saltare un task;
- Integrazione fase finale di test, revisione e valutazione.

Il progetto di creazione della piattaforma di e-learning per la gestione del processo di OnBoarding aziendale è stato realizzato come web application .NET MVC 6.0 utilizzando molteplici tecnologie e linguaggi di programmazione tra cui: C#, HTML, Css, Javascript, SQL.

Capitolo 1

Le caratteristiche dell'OnBoarding

Capitolo 2

Il modello dell'OnBoarding nell'azienda

Capitolo 3

Le tecnologie utilizzate

Il progetto di creazione della piattaforma di e-learning per la gestione del processo di OnBoarding aziendale è stato realizzato utilizzando molteplici tecnologie, servizi e linguaggi:

3.1 Tecnologie e Servizi

1. ASP.NET MVC 6.0;
2. Autenticazione;
3. GitHub;
4. Git;

3.1.1 ASP.NET MVC 6.0

ASP.NET Core MVC è un ricco framework creato da Microsoft per la creazione di applicazioni web e API utilizzando il modello di progettazione Model-View-Controller.

- Model: Il back-end che contiene tutta la logica dei dati;
- View: interfaccia utente front-end o grafica (GUI);
- Controller: Il "cervello" dell'applicazione che controlla come vengono visualizzati i dati.

3.1.2 Autenticazione con "Identity"

Siccome il progetto in questione riguarda la creazione di una piattaforma web, è stato necessario implementare il servizio di Autenticazione attraverso il framework ASP.NET Core Identity, al fine di otterenere non soltanto la possibilità di permettere le azioni di login e registrazione da parte degli utenti, ma anche di farlo ottenendo una sicurezza dei dati sensibili.

In particolare la forma di Autenticazione utilizzata nella piattaforma è una "Forms Authentication", ovvero, è quel tipo di autenticazione nella quale l'utente deve fornire le proprie credenziali attraverso un form dedicato.

ASP.NET Core Identity è un API che supporta funzionalità di login lato UI. Gestisce utenti, password, dati del profilo, ruoli, email di conferma e molto altro.
In particolare, gli utenti possono creare un account con le informazioni di accesso memorizzate in Identity o possono utilizzare un provider di accesso esterno (es. Facebook, Google, ecc...).

Nota: L'algoritmo utilizzato dal servizio Identity per criptare le password è PBKDF2 (Password-Based Key Derivation Function 2):

```
1  public static string HashPassword(string password)
2  {
3      byte[] salt;
4      byte[] bytes;
5      if (password == null)
6      {
7          throw new ArgumentNullException("password");
8      }
9      using (Rfc2898DeriveBytes rfc2898DeriveByte = new Rfc2898DeriveBytes(
       password, 16, 1000))
10     {
11         salt = rfc2898DeriveByte.Salt;
12         bytes = rfc2898DeriveByte.GetBytes(32);
13     }
14     byte[] numArray = new byte[49];
15     Buffer.BlockCopy(salt, 0, numArray, 1, 16);
16     Buffer.BlockCopy(bytes, 0, numArray, 17, 32);
17     return Convert.ToBase64String(numArray);
18 }
19
```

Codice 3.1: algoritmo di hashing per criptare le password usato dal servizio Identity

3.1.3 GitHub

Il progetto durante il suo sviluppo è stato salvato in un'apposità repository privata su GitHub.
GitHub è stato fondamentale per tenere una traccia storica di tutte le modifiche apportate nel corso del tempo, in risposta all'esecuzione dei vari task assegnati.
Esso non si è limitato ad essere un semplice strumento a d'uso esclusivamente individuale, ma ha permesso la condivisione del lavoro sia con il tutor aziendale che con il resto del personale.

3.1.4 Git

In parallelo all'utilizzo di GitHub è stato impiegato il software di controllo di versione Git con l'obbiettivo di agevolare l'aggiornamento dei file del progetto dalla workspace locale verso la repository privata che ospita il progetto, tramite appositi commit.

3.2 Linguaggi

1. SQL;
2. C#;
3. HTML, CSS, JavaScript;

3.2.1 SQL

Il linguaggio SQL è stato utilizzato per la creazione del database del sito web. Il database contiene tutte le informazioni necessarie per la corretta gestione della piattaforma, tra cui i dati relativi agli utenti registrati, ai corsi e alle categorie dei corsi.

3.2.2 C#

Il linguaggio C# è stato utilizzato per la gestione della parte back-end del sito, creando un dialogo tra le informazioni contenute nel database e l'interfaccia utente. L'accesso ai dati del database tramite il linguaggio C# è stato possibile grazie al framework "Dapper", che consente di eseguire interrogazioni in modo efficace ed efficiente. In questo modo, il sito è più responsivo anche in caso di grandi quantità di dati da gestire, eliminando elementi come le procedure, che risultano meno manutenibili e più complicate da gestire a causa di una complicazione del codice nella parte back-end per il loro utilizzo.

Il linguaggio C# è stato anche utilizzato per la gestione degli eventi, la navigazione tra le pagine e i controlli di vario genere. L'utilizzo di molteplici tecnologie ha permesso di ottenere un prodotto completo e funzionale, in grado di soddisfare le esigenze dell'azienda e dei nuovi dipendenti.

3.2.3 HTML, CSS, JavaScript

Per la gestione dell'interfaccia utente, sono stati utilizzati il linguaggio HTML, il linguaggio CSS e il linguaggio JavaScript. Il linguaggio HTML è stato utilizzato per la creazione della struttura del sito, il linguaggio CSS per la gestione dello stile e il linguaggio JavaScript per la gestione degli eventi.

Capitolo 4

Il progetto e il suo sviluppo

4.1 Tree Structure

La tree structure 4.1 del progetto si suddivide in diverse sottocartelle.
Di seguito, commenterò solo le cartelle che risultano rilevanti per la discussione:

- **wwwroot**: Contiene file per la gestione della parte grafica, in particolare, include, in apposite sottocartelle, i file HTML, CSS, JavaScript e le immagini utilizzate.
- **Areas**: Contiene i file generati automaticamente dal framework ASP.NET Core Identity.
- **Controllers**: Contiene i file scritti in linguaggio C# che svolgono un ruolo fondamentale nella gestione delle richieste HTTP e nell'orchestrazione delle azioni dell'applicazione. Questi file fungono da punto di ingresso per le richieste HTTP, coordinano le azioni dell'applicazione, manipolano i dati e restituiscono risposte HTTP appropriate.
- **Migrations**: Contiene i file in linguaggio C# relativi alle migrations.
- **Models**: Contiene solo file C#, che vengono utilizzati per rappresentare le entità e i dati all'interno dell'applicazione, contribuendo così a mantenere il codice organizzato e coeso.
- **Query**: Contiene esclusivamente file SQL, il cui utilizzo è ormai superato. I file contenuti in questa cartella non influenzano l'esecuzione della piattaforma. In particolare, questi file erano stati originariamente utilizzati per conservare le procedure, ma sono state successivamente rimpiazzate dall'uso del framework Dapper.

- **View**: Contiene numerosi file di tipo CSHTML, organizzati in diverse sottocartelle. Questi file sono responsabili di come i dati vengono presentati all'utente, gestendo l'aspetto, la struttura e l'interfaccia utente dell'applicazione.

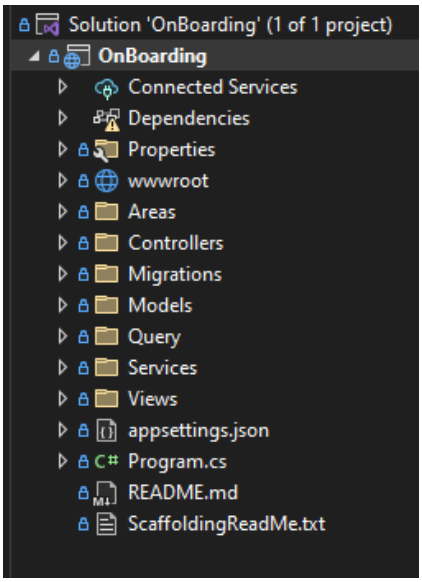


Figura 4.1: tree structure del progetto

4.2 Database

Una parte sostanziale del lavoro impiegato per portare a compimento il progetto si è concentrata in modo dettagliato sulla creazione del database, ritenuto elemento cruciale per assicurare il pieno funzionamento della piattaforma.

La creazione del database è stata articolata in diverse fasi chiave:

- Progettazione della struttura del database, un processo attentamente studiato.
- Scrittura del database in linguaggio SQL, una tappa essenziale per l'implementazione della struttura.
- Implementazione delle interrogazioni al database nella sezione di backend, facendo uso della libreria Dapper.

Nota: il database è stato creato e gestito unicamente in locale per questioni di semplicità e per l'esecuzione di test sulla correttezza della struttura e dell'implementazione del database stesso.

4.2.1 Progettazione

La fase di progettazione 4.2 del database ha occupato un ruolo fondamentale nel corso di questo processo, coinvolgendo un'analisi costante e una riflessione profonda. L'obbiettivo principale è stato quello di plasmare un database estremamente completo, in grado di soddisfare appieno le esigenze della piattaforma di e-learning, ponendosi lo scopo aggiuntivo di:

- Migliorare la leggibilità del database;
- Rendere più agevole la manutenzione;
- Fornire ampie possibilità di estensione del sistema.

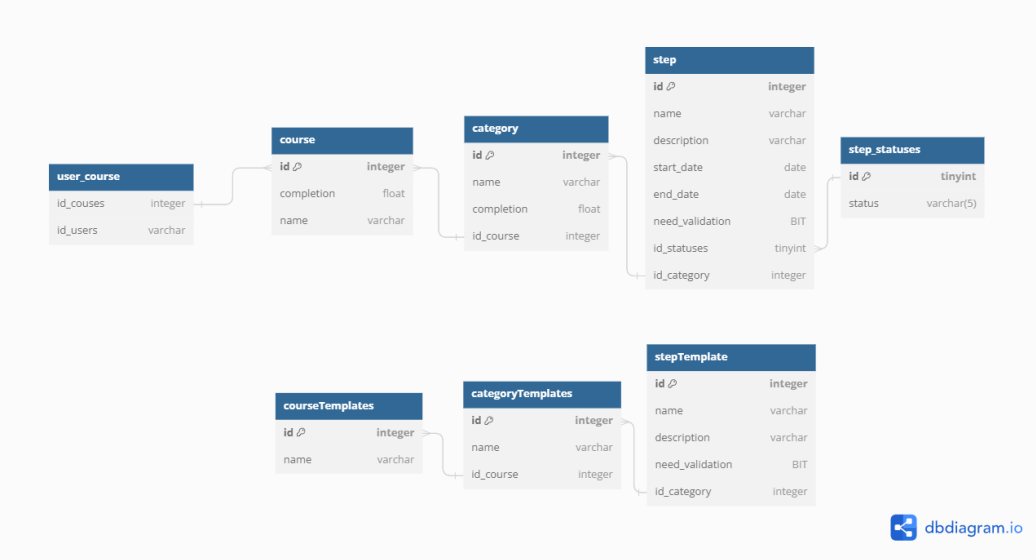


Figura 4.2: progettazione del database

Nota: la progettazione mostrata non tiene traccia delle tabelle utente, perchè vengono gestite automaticamente dal progetto .NET grazie al servizio di autenticazione già incluso (libreria `Microsoft.AspNet.Identity.EntityFramework;`).

Si noti, inoltre, che non è stato creato un collegamento tra le tabelle "*Templates" con il resto del tabelle perchè esse vengono gestite in maniera del tutto indipendente con le altre informazioni e relazioni presenti.

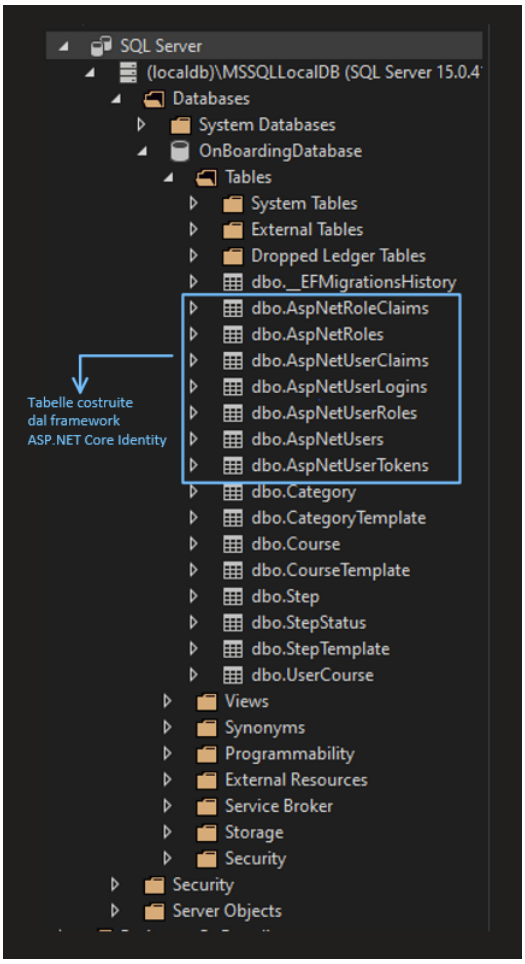


Figura 4.3: Tree structure del database

4.2.2 Scrittura nel linguaggio SQL

Il risultato finale di questa fase è il seguente A (non sono riportate le tabelle generate dal servizio di Autenticazione, perchè gestite automaticamente dalla libreria a disposizione)

4.2.3 Interrogazioni al Database

Inizialmente, le interrogazioni al database erano state sviluppate attraverso stored procedure. Tra i numerosi vantaggi proposti, l'idea principale era ottenere:

- Riutilizzo del codice;

- Semplificazione della manutenzione;
- Prestazioni migliorate;

Tuttavia, durante lo sviluppo dell'applicativo, grazie al suggerimento di alcuni membri dell'azienda, si è preferito sostituire le procedure con interrogazioni dirette dal codice tramite il framework Dapper, al fine di poter migliorare:

- Prestazioni in termini di tempo;
- Semplificare il debugging del codice;
- Centrallizzare l'intera logica in un unico posto.

All'interno della piattaforma si possono distinguere 3 macro categorie di interrogazioni per:

- Inserimento dei dati riguardanti:
  - utenti
  - corsi;
  - categorie (sotto gruppo dei corsi);
  - step (sotto gruppo delle categorie);
  - corsi template;
  - categorie template (sotto gruppo dei corsi template);
  - step template (sotto gruppo delle categorie template).
- Eliminazione dei dati riguardanti:
  - utenti
  - corsi;
  - categorie (sotto gruppo dei corsi);
  - step (sotto gruppo delle categorie);
  - corsi template;
  - categorie template (sotto gruppo dei corsi template);
  - step template (sotto gruppo delle categorie template).
- Lettura dei dati per permettere di ottenere tutte le informazioni necessarie per la generazione corretta della pagina dinamica dato un determinato utente connesso;

4.3 Scrittura del codice C#

La parte del codice back-end rappresenta il nucleo vitale dell'intero progetto, che ne garantisce il corretto funzionamento. Questa componente è stata completamente sviluppata utilizzando il linguaggio di programmazione C#.

L'obbiettivo principale è stato garantire una gestione efficiente del database direttamente attraverso il codice, sfruttando appieno le potenzialità del framework Dapper.
Utilizzando C# e il framework Dapper, è stato possibile realizzare un corretto collegamento tra dati e utenti. Questo ha reso possibile la generazione dinamica e accurata delle pagine web, consentendo un'esperienza utente ottimale.

Per eseguire delle query al database è stato necessario creare una connessione dal codice C# al database attraverso la libreria `Microsoft.Data.SqlClient;`

```
1  using Microsoft.Data.SqlClient;
2  using Microsoft.EntityFrameworkCore;
3
4  namespace OnBoarding.Services
5  {
6    public class ConnectionService
7    private SqlConnection _connection = new SqlConnection();
8    private SqlCommand _command = new SqlCommand();
9
10   public static IConfiguration? Configuration { get; set; }
11
12   public string GetconnectionString()
13   {
14     var builder = new ConfigurationBuilder().SetBasePath(Directory.
       GetCurrentDirectory()).AddJsonFile("appsettings.json");
15
16     Configuration = builder.Build();
17     return Configuration.GetConnectionString("
       ApplicationDBContextConnection");
18   }
19
20
21   public SqlConnection GetConnection()
22   {
23     return _connection;
24   }
25
```

```
26    public SqlCommand GetCommand()
27    {
28      return _command;
29    }
30
31    }
32  }
```

Codice 4.1: classe per ottenere la stringa di connessione per il collegamento al database

In questo modo, una volta che è stata stabilita una connessione, è stato reso possibile il processo di scrittura di query. A titolo esemplificativo, si può menzionare la query nella funzione denominata "GetCourses" presente all'interno del file "CourseManager.cs". Questa query, quando viene fornito l'ID dell'utente come input, ha lo scopo principale di recuperare e restituire l'insieme completo dei corsi associati all'utente specificato.

```
1  public List<CourseModel> GetCourses(string userid)
2  {
3    using (_connection = new SqlConnection(connectionService.
       GetconnectionString()))
4    {
5      _connection.Open();
6
7      string sql =
8        @"SELECT DISTINCT Course.*
9        FROM Course, AspNetUsers, UserCourse
10       WHERE UserCourse.UserId = @userId
11       AND UserCourse.CourseId = Course.Id";
12     var coursesList = _connection.Query<CourseModel>(sql, new { userId =
       userid }).ToList();
13
14     _connection.Close();
15
16     return coursesList;
17   }
18 }
```

Codice 4.2: esempio funzione per l'esecuzione di una query da codice tramite il framework Dapper

Si osserva che ciascun utente può essere associato a più corsi, stabilendo così una relazione uno a molti. Pertanto, è fondamentale che il valore restituito

dalla funzione sia di tipo `List<CourseModel >`, in quanto questa struttura dati può contenere più di un corso associato a un utente specifico. Inoltre, grazie all'uso di Dapper, è possibile effettuare un'interrogazione al database mediante una stringa appositamente creata (come mostrato nelle righe 7-11 del codice). Successivamente, è possibile ottenere il risultato dell'interrogazione corrispondente al parametro "userid" fornito alla funzione e al campo "userId" nella tabella "Course" (riga 12 del codice).
Notare che siccome la funzione ritorna in questo caso una lista, il risultato della query dovrà essere convertito in una collezione di oggetti dello stesso tipo (attraverso alla funzione `ToList()`).

Un punto di particolare rilevanza che merita menzione è rappresentato dallo sviluppo di una classe denominata "StatisticsController.cs". Il suo scopo principale consiste nell'elaborare in modo esaustivo e dettagliato tutte le statistiche associate a un utente specifico. Questa classe si avvale della sofisticata libreria `ClosedXML.Excel;` per generare dati statistici inerenti al completamento dei corsi che sono stati non solo avviati, ma anche portati a termine dall'utente in questione. È essenziale sottolineare che questa funzionalità è rigorosamente riservata agli utenti che detengono il privilegiato ruolo di "Admin".

```
1  [HttpGet]
2  [Authorize(Roles = "Admin")]
3  public IActionResult ExportToExcel(int id)
4  {
5    using (var workbook = new XLWorkbook())
6    {
7      CourseModel course = _courseManager.GetCourseByID(id);
8
9      var ws = workbook.Worksheets.Add(course.Name); // Sheet Name
10     int i = 1;
11
12     ////////////////////////////////////// Course
13     ws.Range($"A{i}:E{i}").Merge();
14     ws.Cell(i, 1).Value = "Course";
15     ws.Cell(i, 1).Style.Font.Bold = true;
16     ws.Cell(i, 1).Style.Alignment.Horizontal = XLAlignmentHorizontalValues.
       Center;
17     ws.Cell(i, 1).Style.Font.FontSize = 30;
18
19     // Header
20     i++;
21     ws.Cell(i, 1).Value = "Id";
```

```
22      ws.Cell(i, 2).Value = "Name";
23      ws.Cell(i, 3).Value = "Completion";
24      ws.Cell(i, 4).Value = "StartDate";
25      ws.Cell(i, 5).Value = "EndDate";
26      ws.Range($"A{i}:E{i}").Style.Fill.BackgroundColor = XLColor.Alizarin;
27
28      // Body
29      i++;
30      ws.Cell(i, 1).Value = course.Id;
31      ws.Cell(i, 2).Value = course.Name;
32      ws.Cell(i, 3).Value = course.Completion;
33      if (course.StartDate != DateTime.MinValue) ws.Cell(i, 4).Value = course.
        StartDate.ToString();
34      else { ws.Cell(i, 4).Value = "NULL"; ws.Cell(i, 4).Style.Font.FontColor =
        XLColor.Red; }
35      if (course.EndDate != DateTime.MinValue) ws.Cell(i, 5).Value = course.
        EndDate.ToString();
36      else { ws.Cell(i, 5).Value = "NULL"; ws.Cell(i, 5).Style.Font.FontColor =
        XLColor.Red; }
37      ws.Range($"A{i}:E{i}").Style.Fill.BackgroundColor = XLColor.AliceBlue;
38      i += 1;
39
40      // ...
41      // Stesso procedimento appena descritto per i Corsi, viene fatto
42      // per le Categorie e gli Step
43
44      using (var stream = new MemoryStream())
45      {
46          workbook.SaveAs(stream);
47          var content = stream.ToArray();
48          return File(
49              content,
50              "application/vnd.openxmlformats-officedocument-spreadsheetml.
        sheet",
51              course.Name + ".csv"
52          );
53      }
54  }
55 }
```

Codice 4.3: funzione utilizzata per l'esportazione in file exel

Un aspetto di notevole rilevanza è stato il processo di gestione dei dati statistici all'interno della pagina denominata "/Views/Statistics/Index.cshtml".

Questa pagina ha una funzione fondamentale poiché consente agli admin di visualizzare l'insieme completo delle statistiche. Ciò è reso possibile grazie alla capacità di ordinare agevolmente le informazioni presenti all'interno delle tabelle. Questa caratteristica permette agli utenti di organizzare i dati in base alle loro specifiche esigenze e criteri di ricerca.
Inoltre, per rendere l'esperienza ancora più interattiva e informativa, è stata inclusa una sezione di codice dedicata direttamente all'interno della pagina stessa, utilizzando il tag $< \text{script} > \ldots < /\text{script} >$. Questo approccio ha consentito di incorporare efficacemente dei grafici che offrono una rappresentazione visiva dei dati statistici. Questi grafici sono stati sviluppati in modo da offrire quattro diverse visualizzazioni, ognuna delle quali fornisce un'analisi dettagliata dei dati statistici, permettendo agli utenti di comprendere meglio i pattern e le tendenze sottostanti.

- due grafici a barre;
  - Il primo rappresenta la somma totale dei corsi suddivisi per stato di completamento (done, running, todo);
  - Il secondo rappresenta il numero di corsi completati in relazione al mese dell'anno.
- due grafici a torta;
  - Il primo rappresenta il tempo medio di completamento necessario per terminare uno specifico corso, suddiviso per corso;
  - Il secondo rappresenta il tempo medio di completamento necessario per teminare uno specifico corso, suddiviso per utenti.

L'implementazione del codice per la creazione dei grafici è stata effettuata mediante l'uso del linguaggio di programmazione JavaScript. Un aspetto interessante da sottolineare è che i dati numerici necessari per la creazione dei grafici non sono stati ottenuti direttamente tramite il codice JavaScript, ma sono stati acquisiti attraverso apposite funzioni getter scritte in codice C#. Questo approccio è stato adottato per garantire l'accuratezza e la coerenza dei dati, nonché per sfruttare le funzionalità di gestione dei dati messe a disposizione dal linguaggio C#. In sostanza, il processo è stato il seguente: il codice JavaScript si è interfacciato con il codice C# attraverso apposite funzioni getter, ottenendo così i dati numerici necessari per alimentare i grafici.

```
1 <script>
2   // ...
3
```

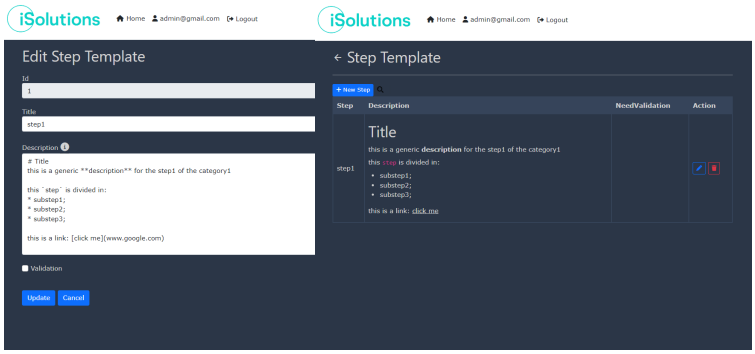
```
4   // -------------------------------- BAR CHART
5     var xValues = ["To Do", "Running", "Done"];
6     var yValues = [@todo, @running, @done, 0];
7     var barColors = ["red", "#e5e600", "green"];
8
9     new Chart("BarChart", {
10        type: "bar",
11        data: {
12            labels: xValues,
13            datasets: [{
14                backgroundColor: barColors,
15                data: yValues
16            }]
17        },
18        options: {
19            legend: { display: false },
20            title : {
21                display: false,
22                text: ""
23            }
24        }
25    });
26
27  // ...
28 </script>
```

Codice 4.4: esempio sezione codice Javascript per la creazione di un grafico a barre per la rappresentazione dei dati

4.4 Layout della piattaforma

Nonostante alcune limitazioni evidenti nel design della piattaforma, che potrebbe essere descritto come non completamente moderno e, in alcuni aspetti, forse un po' troppo minimalista, il layout della piattaforma è stato progettato e realizzato con l'obiettivo di creare un servizio che sia estremamente intuitivo e facile da utilizzare, sia per gli utenti che per gli amministratori responsabili della gestione degli accessi e del progresso degli utenti durante il loro percorso nell'esperienza. Grazie a questa attenzione all'usabilità, è stato possibile assicurarsi che l'intera esperienza sia accessibile e fruibile in modo agevole e soddisfacente per tutti i suoi utilizzatori.

Un'aggiunta di notevole interesse dal punto di vista del layout è stata l'integrazione della visualizzazione in sintassi Markdown. Questa nuova funzionalità mira a migliorare l'esperienza dell'utente durante la visualizzazione delle descrizioni delle informazioni testuali presenti all'interno dei corsi. L'obiettivo è ottenere una visualizzazione che favorisca una maggiore leggibilità delle informazioni e permetta l'integrazione di elementi come immagini, titoli, sottotitoli e molto altro all'interno delle descrizioni dei task. Questo offre maggiore flessibilità e personalizzazione agli amministratori incaricati della creazione dei corsi.



(a) scrittura in Markdown

(b) visualizzazione in Markdown

Figura 4.4: esempio di scrittura e visualizzazione in Markdown

Inoltre, per agevolare l'utilizzo di questa funzionalità, è stato incluso un apposito messaggio pop-up 4.5 che fornisce una semplice guida all'uso, completa di esempi pratici, al fine di consentire agli utenti di sfruttarla al meglio.

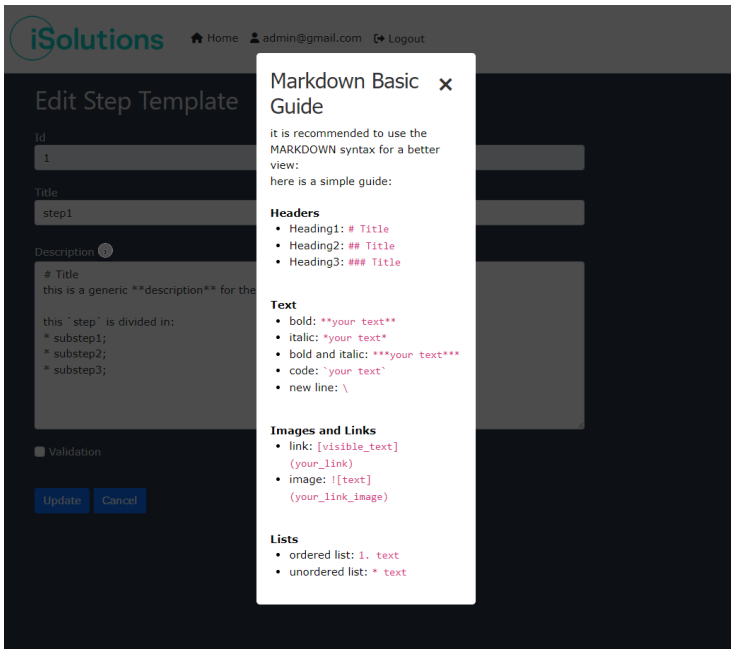


Figura 4.5: Guida basilare al Markdown

Capitolo 5

Risultati e sviluppi futuri

5.1 Risultati

5.2 Sviluppi fututi

5.2.1 Pagina statistiche

La pagina delle statistiche deve subire un notevole potenziamento, mirando a offrire un'esperienza più completa ed esaustiva. Questo miglioramento deve concentrarsi sulla possibilità di confrontare una quantità più ampia di dati, garantendo al contempo un maggiore spazio dedicato all'analisi dei dati. Questa evoluzione si rivolge in particolare al personale del reparto HR dell'azienda, che ha la responsabilità di gestire il processo di OnBoarding per una varietà di utenti.

Grazie alla capacità di consentire un confronto più dettagliato dei dati, questa pagina deve diventare uno strumento prezioso per l'HR. Inoltre, dovrebbe essere possibile implementare un numero maggiore di grafici interattivi, che contribuiranno significativamente alla comprensione e all'analisi dei dati.

Questo potenziamento non solo agevolerà il processo decisionale dell'HR, ma permetterà anche di individuare con precisione le aree che richiedono interventi e miglioramenti. In definitiva, mira a trasformare la pagina delle statistiche in uno strumento fondamentale per l'ottimizzazione dell'OnBoarding all'interno del contesto dell'azienda.

5.2.2 Ruoli utente

Un'ulteriore evoluzione e arricchimento del sistema potrebbe senz'altro consistere nell'espandere l'attuale struttura dei ruoli, che attualmente comprende solo due categorie: "User" e "Admin". Questa espansione comporterebbe l'aggiunta di ulteriori ruoli, consentendo così una maggiore granularità nei permessi di visualizzazione all'interno delle pagine generate.

L'implementazione di ruoli aggiuntivi potrebbe essere estremamente vantaggiosa, poiché permetterebbe di adattare l'accesso alle informazioni in base alle specifiche responsabilità e autorizzazioni di ciascun utente. In questo modo, si potrebbe garantire un maggiore controllo sull'accesso ai dati sensibili e una gestione più efficiente delle risorse aziendali.

Questo approccio fornirebbe ai decision-maker una maggiore flessibilità nella configurazione dei permessi e consentirebbe di assegnare ruoli intermedi, ad esempio "Supervisor" o "Manager", con diritti di accesso mirati alle informazioni rilevanti per il loro ruolo. Ciò migliorerebbe la sicurezza dei dati e l'efficienza operativa, consentendo a ciascun membro del team di accedere solo alle risorse necessarie per svolgere le proprie mansioni.

5.2.3 Creazione dei Corsi

Un aspetto che potrebbe essere considerato di importanza secondaria, ma che indubbiamente contribuirebbe al miglioramento significativo dell'esperienza utente, soprattutto dal punto di vista dell'amministratore, riguarda la possibilità di creare Corsi o CorsiTemplate in modo più efficiente, attraverso l'utilizzo di un nuovo processo/i di creazione. Attualmente, questa operazione richiede un processo manuale, ma esistono alternative che potrebbero semplificarne notevolmente l'implementazione.

Una di queste opzioni sarebbe consentire agli amministratori di importare direttamente i dati relativi ai Corsi o ai CorsiTemplate da fogli Excel. Questo approccio eliminerebbe gran parte del lavoro manuale, consentendo di caricare rapidamente una quantità significativa di informazioni nel sistema.

Inoltre, potrebbe essere utile considerare l'integrazione di funzionalità native all'interno del sito web per la creazione di Corsi o CorsiTemplate. Queste funzionalità integrate potrebbero offrire un ambiente più intuitivo e user-friendly per la progettazione e la gestione dei Corsi, migliorando ulteriormente l'esperienza dell'amministratore.

In definitiva, anche se questa caratteristica potrebbe essere considerata di importanza secondaria, l'implementazione di strumenti per l'importazione da Excel o funzionalità native all'interno del sito rappresenterebbe un passo avanti significativo nella semplificazione delle attività legate alla creazione di Corsi e CorsiTemplate, contribuendo in modo tangibile all'efficienza operativa complessiva e al miglioramento dell'esperienza dell'utente amministratore.

Nota: attualmente l'unica modalità per la creazione di nuovi Corsi e CorsiTemplate completi di relative Categorie e Step è la seguente:

1. creazione del corso;
2. apertura del corso;
3. creazione della categoria;
4. apertura della categoria;
5. creazione dello step;

5.2.4 Text editor

Attualmente, la feature che consente la scrittura in sintassi Markdown è piuttosto limitata e semplice. Potrebbe essere opportuno valutare l'implementazione di un vero e proprio editor di testo dedicato, in aggiunta alla possibilità di scrivere il Markdown manualmente. Questa aggiunta consentirebbe agli utenti di sfruttare gli stili di formattazione e i vantaggi offerti dalla sintassi Markdown in modo molto più facile, inclusivo ed intuitivo per tutti. Grazie alla presenza di appositi tasti funzione e a un ambiente di scrittura appositamente progettato, si potrebbero sfruttare al meglio tutte le potenzialità della formattazione Markdown senza la necessità di conoscere a fondo la sua sintassi. Questo migliorerebbe notevolmente l'esperienza degli utenti e renderebbe l'utilizzo di questa funzionalità ancora più accessibile e versatile. Di seguito un esempio preso dal programma "Slack":

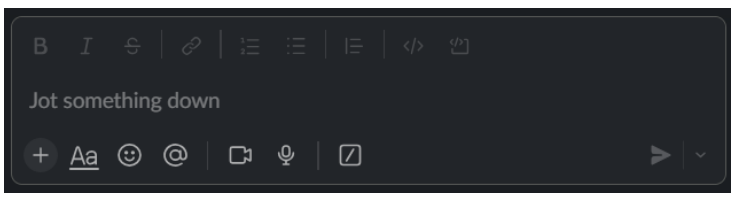


Figura 5.1: esempio di un possibile text editor

5.2.5 Reportistica errori e commenti

Reportistica errori

Una possibile feature futura che potrebbe rivelarsi estremamente utile riguarderebbe l'integrazione di un meccanismo avanzato che consenta agli utenti incaricati di svolgere i corsi all'interno della piattaforma di segnalare errori o imprecisioni nei task assegnati. Questa innovativa funzionalità darebbe agli utenti la preziosa possibilità di comunicare direttamente agli amministratori eventuali problemi o lacune, come ad esempio link non funzionanti dovuti a modifiche nel corso del tempo o informazioni errate nelle assegnazioni. Inoltre, potrebbe includere anche la segnalazione di descrizioni mancanti o insufficientemente complete, offrendo un quadro completo delle aree da migliorare.

L'introduzione di questa avanzata funzionalità consentirebbe agli amministratori di ricevere feedback costanti da parte degli utenti, contribuendo così in modo significativo a elevare la qualità del servizio offerto. Inoltre, fornirebbe un meccanismo efficace per mantenere sempre aggiornati e corretti i contenuti presenti sulla piattaforma, garantendo agli utenti una esperienza di apprendimento completa e senza interruzioni.

Commenti

Potrebbe rivelarsi di di grande utilità considerare l'aggiunta di un sistema di assistenza diretta agli utenti tramite appositi commenti, in modo che gli utenti possano richiedere aiuto agli amministratori in maniera immediata. Sistema che potrebbe essere implementato in modo da consentire la comunicazione in threads dedicati, dove ogni nuovo messaggio può essere gestito in modo organizzato e pertinente. Questa funzionalità potrebbe essere integrata insieme alla feature di segnalazione degli errori o essere inserita in una sezione separata, per garantire un supporto completo e altamente efficiente per tutti gli utenti finali, migliorando notevolmente l'esperienza globale sulla piattaforma.

5.2.6 Layout della piattaforma

Attualmente, il layout della piattaforma si presenta con un design estremamente minimalista, concentrandosi esclusivamente su ciò che è strettamente necessario per la corretta visualizzazione dei dati e delle informazioni presenti nella piattaforma. Questo stile, sebbene efficiente, può essere considerato

non particolarmente moderno. Pertanto, una delle possibili migliorie potrebbe consistere nell'apportare alcune modifiche significative al layout stesso, con l'obiettivo di migliorare l'esperienza dell'utente.

Una proposta consisterebbe nell'introduzione di navbar più complete e informative, specialmente per gli amministratori. Queste navbar potrebbero offrire un accesso rapido alle funzionalità e agli strumenti di amministrazione, semplificando così le attività di gestione e supervisione dell'intero sistema.

Inoltre, potrebbe essere benefico considerare l'implementazione di specifiche side-navbar per gli utenti, soprattutto quando si trovano nella pagina di visualizzazione dei corsi. Attualmente, la ricerca di task specifici da completare all'interno di un corso potrebbe risultare disorientante per l'utente finale, in quanto potrebbe richiedere uno sforzo eccessivo per individuare le informazioni rilevanti. L'aggiunta di una side-navbar dedicata potrebbe semplificare notevolmente questa operazione, consentendo agli utenti di accedere rapidamente ai alle categorie o ai task all'interno del corso senza difficoltà.

In conclusione, apportare modifiche al layout della piattaforma attraverso l'implementazione di navbar più esaustive per gli admin e di side-navbar specifiche per gli utenti, soprattutto nella pagina di visualizzazione dei corsi, rappresenterebbe un passo importante per migliorare l'usabilità complessiva del sistema e l'esperienza dell'utente finale.

Conclusione

Conclusione che riassume il lavoro svolto ed eventuali lavori futuri.

Ringraziamenti

Desidero ringraziare sinceramente le seguenti persone dell'azienda ISolutions che mi hanno accompagnato e guidato durante l'esecuzione del mio progetto di tirocinio:

- Alessandro Bardini (Line Manager) & Gialuca Bellini (Line Manager - NOC Team Lead). Per la loro preziosa guida nell'ambito dell'esecuzione dei vari compiti del progetto.
- Ivan Anselmi (Architect, R&D Dev). Per l'assistenza nell'applicazione del servizio di Autenticazione, per la guida su come implementare al meglio il framework Dapper e per i consigli e le linee guida forniti per ottenere la corretta gestione delle query string nella realizzazione della piattaforma web sviluppata.
- Roberto Isca (Senior Web Designer & Web Developer). Per i preziosi consigli e l'aiuto fornito in numerosi problemi di layout.
- Marco Chiesa (System Engineer). Per aver reso possibile il mio lavoro in remoto grazie alla configurazione della VPN e dei servizi aziendali.

Ringrazio, inoltre, coloro che mi hanno fornito preziosi consigli e che mi hanno sostenuto sia durante il progetto di tirocinio sia nella stesura di questo elaborato:

- Donatello Larocca.
- Mariachiara Michelini.

Il loro sostegno e la loro guida sono stati fondamentali. Grazie di cuore a tutti loro.

Appendice A

Scrittura del Database

```
CREATE TABLE [dbo].[Course] (
    [Id]           INT             IDENTITY (1, 1) NOT NULL,
    [Name]         NVARCHAR (255) NOT NULL,
    [Completion] REAL          DEFAULT ((0)) NULL,
    [StartDate]   DATETIME2 (7)  NULL,
    [EndDate]    DATETIME2 (7)  NULL,
    CONSTRAINT [PK_Course] PRIMARY KEY CLUSTERED ([Id] ASC)
);

CREATE TABLE [dbo].[Category] (
    [Id]           INT             IDENTITY (1, 1) NOT NULL,
    [Name]         NVARCHAR (255) NOT NULL,
    [Completion] REAL          DEFAULT ((0)) NULL,
    [IdCourse]    INT             NOT NULL,
    CONSTRAINT [PK_Category] PRIMARY KEY CLUSTERED ([Id] ASC),
    CONSTRAINT [FK_Category_Course_IdCourse]
    FOREIGN KEY ([IdCourse]) REFERENCES [dbo].[Course] ([Id])
    ON DELETE CASCADE
    );
    GO
    CREATE NONCLUSTERED INDEX [IX_Category_IdCourse]
        ON [dbo].[Category]([IdCourse] ASC);

-- 3 stati:
    -- 1. done
    -- 2. todo
    -- 3. check
CREATE TABLE [dbo].[StepStatus] (
    [Id]       INT             IDENTITY (1, 1) NOT NULL,
```

```
    [Status] NVARCHAR (10) NOT NULL,
    CONSTRAINT [PK_StepStatus] PRIMARY KEY CLUSTERED ([Id] ASC)
);

CREATE TABLE [dbo].[Step] (
    [Id]              INT            IDENTITY (1, 1) NOT NULL,
    [Name]            NVARCHAR (255) NOT NULL,
    [Description]     NVARCHAR (MAX) NULL,
    [StartDate]       DATETIME2 (7)  NULL,
    [EndDate]         DATETIME2 (7)  NULL,
    [Lock]            BIT            DEFAULT ((1)) NULL,
    [NeedValidation] BIT             DEFAULT ((0)) NULL,
    [IdStatus]        INT            DEFAULT ((1)) NULL,
    [IdCategory]      INT            NOT NULL,
    CONSTRAINT [PK_Step] PRIMARY KEY CLUSTERED ([Id] ASC),
    CONSTRAINT [FK_Step_StepStatus_IdStatus]
FOREIGN KEY ([IdStatus]) REFERENCES [dbo].[StepStatus] ([Id])
ON DELETE CASCADE,
    CONSTRAINT [FK_Step_Category_IdCategory]
FOREIGN KEY ([IdCategory]) REFERENCES [dbo].[Category] ([Id])
ON DELETE CASCADE
);
GO
CREATE NONCLUSTERED INDEX [IX_Step_IdCategory]
    ON [dbo].[Step]([IdCategory] ASC);
GO
CREATE NONCLUSTERED INDEX [IX_Step_IdStatus]
    ON [dbo].[Step]([IdStatus]  ASC);

CREATE TABLE [dbo].[UserCourse] (
    [UserId]    NVARCHAR (450) NOT NULL,
    [CourseId] INT              NOT NULL,
    CONSTRAINT [FK_UserCourse_Course_CourseModel]
FOREIGN KEY ([CourseId]) REFERENCES [dbo].[Course] ([Id])
ON DELETE CASCADE,
    CONSTRAINT [FK_UserCourse_AspNetUsers_UserId]
FOREIGN KEY ([UserId]) REFERENCES [dbo].[AspNetUsers] ([Id])
ON DELETE CASCADE
);
GO
CREATE NONCLUSTERED INDEX [IX_UserCourse_CourseModel]
    ON [dbo].[UserCourse]([CourseId] ASC);
GO
```

```
73 CREATE NONCLUSTERED INDEX [IX_UserCourse_UserId]
74     ON [dbo].[UserCourse]([UserId] ASC);
75
76 CREATE TABLE [dbo].[CourseTemplate] (
77     [Id]               INT            IDENTITY (1, 1) NOT NULL,
78     [Name]             NVARCHAR (255) NOT NULL,
79     [Creator]          NVARCHAR (255) NOT NULL,
80     [CreationDate]     DATETIME2 (7)  NULL,
81     [LastUpdateDate] DATETIME2 (7)  NULL,
82     CONSTRAINT [PK_CourseTemplate] PRIMARY KEY CLUSTERED ([Id]
       ASC)
83 );
84
85 CREATE TABLE [dbo].[CategoryTemplate] (
86     [Id]                 INT         IDENTITY (1, 1) NOT NULL,
87     [name]               NCHAR (255) NOT NULL,
88     IDCourseTemplate INT         NOT NULL,
89     PRIMARY KEY CLUSTERED ([Id] ASC),
90     FOREIGN KEY (IDCourseTemplate)
91     REFERENCES [dbo].[CourseTemplate] ([Id])
92     ON DELETE CASCADE
93 );
94
95 CREATE TABLE [dbo].[StepTemplate] (
96     [Id]                 INT            IDENTITY (1, 1) NOT NULL,
97     [Name]               NVARCHAR (255) NOT NULL,
98     [Description]        NVARCHAR (MAX) NOT NULL,
99     [NeedValidation]     BIT            DEFAULT ((0)) NULL,
100    [IDCategoryTemplate] INT            NOT NULL,
101    CONSTRAINT [PK_StepTemplate] PRIMARY KEY CLUSTERED ([Id]
       ASC),
102    CONSTRAINT [FK_StepTemplate_CategoryTemplate_IDCategoryTemplate]
103    FOREIGN KEY ([IDCategoryTemplate]) REFERENCES [dbo].[
       CategoryTemplate] ([Id])
104    ON DELETE CASCADE
105 );
106 GO
107 CREATE NONCLUSTERED INDEX [IX_StepTemplate_IDCategoryTemplate]
108     ON [dbo].[StepTemplate]([IDCategoryTemplate] ASC);
```

Codice A.1: tabelle per la creazione del database utilizzato