1. **INTRODUZIONE**

**1.1. Presentazione del tema e della visione aziendale**

L’Information Technology è una leva strategica per creare nuovi paradigmi e nuovi valori che contribuiscano a costruire una società armoniosa e prosperosa. Questa è la visione che ha guidato il percorso di tirocinio presso NTT Data, una realtà di consulenza internazionale che opera nei settori dell’innovazione, del design, della tecnologia, della cybersecurity e dell’AI. NTT Data è una società giapponese che fa parte del gruppo Nippon Telegraph and Telephone Corporation (NTT), uno dei maggiori operatori di telecomunicazioni al mondo. La sua missione è di fornire soluzioni digitali innovative e personalizzate ai propri clienti, che spaziano dai settori pubblico, finanziario, industriale, energetico, sanitario e sociale. NTT Data si propone di essere un partner affidabile e di lungo termine per i propri clienti, offrendo loro competenza, qualità e valore aggiunto.

**1.2. Contesto aziendale e obiettivi del tirocinio**

Il tirocinio si è svolto in un contesto dinamico e sfidante, dove sono state applicate le conoscenze teoriche acquisite durante il corso di studi e apprese nuove competenze pratiche e metodologiche nel campo dello sviluppo web e back-end. L’obiettivo era di realizzare dei sistemi web che rispondessero alle esigenze dei clienti e che fossero al tempo stesso funzionali, sicuri e user-friendly.

**1.3. Descrizione delle fasi e dei sistemi web realizzati**

Il tirocinio si è articolato in due fasi distinte ma complementari, che hanno richiesto circa 600 ore di lavoro:

• Nella prima fase è stato realizzato un sistema web simulativo per la gestione di file Excel relativi a dei progetti sociali finanziati da enti pubblici o privati. Il sistema consentiva di importare un file Excel per caricare i dati su un database relazionale, visualizzare i dati in una tabella nella pagina web con la possibilità di effettuare operazioni CRUD (Create, Read, Update, Delete), esportare in un nuovo file Excel i dati modificati dall’utente e accedere al sistema con un sistema di autenticazione e autorizzazione basato su username e password. Per questo sistema sono stati utilizzati il linguaggio C# per la programmazione del back-end e il database Microsoft SQL Server per la gestione dei dati. Sono stati usati anche HTML, CSS e JavaScript per la creazione dell’interfaccia grafica del front-end. Il sistema era dotato di una struttura a tre livelli: presentazione, business e dati. Questo sistema ha permesso di acquisire le competenze necessarie per lo sviluppo di applicazioni web con il linguaggio C# e il database Microsoft SQL Server, oltre a familiarizzare con gli strumenti di sviluppo Visual Studio e SQL Server Management Studio.

• Nella seconda fase si è partecipato a un progetto reale per conto di Eni, una delle maggiori aziende energetiche a livello mondiale, riguardante la realizzazione di un modulo web per la somministrazione di un test per l’HSE (Health, Safety and Environment). Il modulo permetteva ai dipendenti e ai collaboratori di Eni che operano in contesti ad alto rischio ambientale e sanitario di rispondere a una serie di domande a scelta multipla sulle norme HSE, ricevere un riscontro immediato sul livello di conoscenza delle norme HSE, consultare dei materiali informativi sulle buone pratiche HSE e registrare i risultati del test su un database. Per questo modulo è stato utilizzato il framework AngularJS per lo sviluppo del front-end. AngularJS è un framework JavaScript che consente di creare applicazioni web dinamiche e reattive, basandosi su HTML, CSS e JavaScript, i linguaggi standard per la creazione di pagine web. Inoltre, è stato integrato il front-end con i servizi forniti dal back-end, che gestivano la logica di business e l’interazione con il database. Questo ha permesso di realizzare un’architettura modulare e scalabile per il modulo web.

**1.4. Struttura della tesi**

La presente tesi illustra in dettaglio le attività svolte durante il tirocinio, le tecnologie utilizzate, le caratteristiche tecniche e funzionali dei sistemi web realizzati e i risultati ottenuti. La tesi si compone di quattro capitoli: nel capitolo 2 si descrivono le tecnologie utilizzate per la realizzazione del sistema web; nel capitolo 3 si presenta il sistema web per la gestione dei file Excel; nel capitolo 4 si illustra il modulo web per il test HSE; nel capitolo 4 si riportano le conclusioni del lavoro e le possibili prospettive future. Nel corso del tirocinio si è lavorato in autonomia con la supervisione di un tutor aziendale, che ha fornito le indicazioni e il supporto necessari per lo svolgimento delle attività. Si sono affrontate alcune difficoltà, come la comprensione delle specifiche, l’integrazione delle componenti e il testing del codice. Tuttavia, si sono sapute superare grazie alla capacità di problem solving e alla collaborazione con i tutor. Il risultato è stata un’esperienza formativa e stimolante, che ha permesso di arricchire il curriculum e di avvicinarsi al mondo del lavoro.

1. **Tecnologie utilizzate**

In questo capitolo verranno mostrate nel dettaglio le tecnologie utilizzate per questi due progetti.

* 1. Immagine che contiene Elementi grafici, cerchio, grafica, simbolo

     Descrizione generata automaticamente**C#: linguaggio di programmazione per il back-end**

C# è un linguaggio di programmazione orientato agli oggetti, moderno, innovativo e open source, che fa parte della piattaforma .NET. Oltre alla sintassi, offre diverse funzionalità e vantaggi che lo rendono un linguaggio potente e versatile.

C# è uno dei linguaggi più popolari e amati dai programmatori, in quanto offre diverse caratteristiche e vantaggi, tra cui:

* [**Interoperabilità**: C# può interagire bene con altri linguaggi di programmazione, grazie alla compatibilità con il Common Language Runtime (CLR), una macchina virtuale che esegue il codice .NET](https://dotnet.microsoft.com/en-us/languages/csharp).
* [**Strutturato**: C# divide un programma complesso in moduli, procedure o funzioni che risolvono problemi specifici](https://hackr.io/blog/how-to-learn-c-as-a-beginner).
* [**Orientato ai componenti**: C# è un linguaggio orientato ai componenti, che significa che usa eventi, proprietà, attributi, metodi, ecc., per creare e descrivere componenti autonomi chiamati funzionalità](https://hackr.io/blog/how-to-learn-c-as-a-beginner).
* [**Sicuro**: C# garantisce la sicurezza del tipo e della memoria, prevenendo errori comuni come l’accesso a puntatori non validi o la conversione implicita di tipi incompatibili](https://dotnet.microsoft.com/en-us/languages/csharp).
* **Gestione automatica della memoria**: C# gestisce in modo efficiente la memoria, liberando automaticamente lo spazio occupato dagli oggetti non più utilizzati. [Questo evita il problema della perdita di memoria e rende il codice più sicuro e pulito](https://www.geeksforgeeks.org/introduction-to-c-sharp/).
* [**Interoperabilità**: C# può interagire bene con altri linguaggi di programmazione, grazie alla compatibilità con il Common Language Runtime (CLR), una macchina virtuale che esegue il codice .NET](https://www.codeguru.com/csharp/c-sharp-advantages/). Inoltre, C# può usare le librerie native del sistema operativo tramite il meccanismo del Platform Invoke.
* [**Cross-platform**: C# è un linguaggio cross-platform, che significa che può essere eseguito su diversi sistemi operativi e piattaforme, come Windows, Linux, Android, iOS o nel cloud](https://www.codeguru.com/csharp/c-sharp-advantages/). Questo rende C# adatto per lo sviluppo di applicazioni diverse e scalabili.
* [**Orientato al componente**: C# è un linguaggio orientato al componente, che significa che usa eventi, proprietà, attributi, metodi, ecc., per creare e descrivere componenti autonomi chiamati funzionalità](https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/). Questo facilita la modularità, il riuso e la manutenzione del codice.
* [**Generico**: C# supporta i generici, che sono dei tipi o dei metodi parametrizzati che consentono di scrivere codice riutilizzabile e flessibile per diverse tipologie di dati](https://www.codeguru.com/csharp/c-sharp-advantages/). I generici aumentano la sicurezza del tipo e le prestazioni del codice.
* [**Espressivo**: C# offre diverse funzionalità che rendono il codice più conciso ed elegante, come le espressioni lambda, le query integrate (LINQ), il pattern matching, i record e altro ancora](https://www.codeguru.com/csharp/c-sharp-advantages/). Queste funzionalità permettono di scrivere codice più leggibile e funzionale.

La sintassi di C# è simile a quella di altri linguaggi derivati dal C, come Java o C++. Per scrivere un programma in C#, si usano dei costrutti fondamentali come le classi, i metodi, le variabili, le istruzioni, le espressioni e i commenti:

* **Classi**: una classe è un contenitore per i dati e i metodi che definiscono il comportamento di un oggetto. Una classe può avere dei campi, delle proprietà, dei costruttori, dei metodi e degli eventi. Per creare una classe si usa la parola chiave **class** seguita dal nome della classe e dalle parentesi graffe che delimitano il corpo della classe. Per creare un’istanza di una classe si usa l’operatore **new** seguito dal nome della classe e dalle parentesi tonde che contengono gli eventuali parametri del costruttore. Per accedere ai membri di una classe si usa l’operatore. (punto).
* **Metodi**: un metodo è una funzione che esegue un’azione o restituisce un valore. Un metodo può avere dei parametri di ingresso e un valore di ritorno. Per definire un metodo si usa la parola chiave **void** se il metodo non restituisce nulla, oppure il tipo del valore di ritorno se il metodo restituisce qualcosa. Si usa poi il nome del metodo seguito dalle parentesi tonde che contengono gli eventuali parametri separati da virgola e dalle parentesi graffe che delimitano il corpo del metodo. Per invocare un metodo si usa il nome del metodo seguito dalle parentesi tonde che contengono gli eventuali argomenti da passare ai parametri.
* **Variabili**: una variabile è un contenitore per un valore che può cambiare nel tempo. Una variabile ha un nome, un tipo e un valore. Per dichiarare una variabile si usa la parola chiave **var** seguita dal nome della variabile e dall’operatore **=** (uguale) per assegnare un valore iniziale. Il tipo della variabile viene dedotto dal compilatore in base al valore assegnato. Per modificare il valore di una variabile si usa nuovamente l’operatore **=**.
* **Istruzioni**: un’istruzione è un’azione che viene eseguita dal programma. Un’istruzione può essere una chiamata a un metodo, un’assegnazione a una variabile, una dichiarazione di una classe, una condizione, un ciclo, ecc. Ogni istruzione termina con un punto e virgola (“**;”**).
* **Espressioni**: un’espressione è una combinazione di operatori, operandi e metodi che produce un valore. Un’espressione può essere usata come parte di un’istruzione o come argomento di un metodo.
  1. Immagine che contiene Elementi grafici, arte, grafica, design

     Descrizione generata automaticamente**Microsoft SQL Server: database relazionale per la gestione dei dati**

Microsoft SQL Server è un DBMS (Database Management System) relazionale, prodotto da Microsoft, che consente di creare e gestire basi di dati di diverse dimensioni e complessità. Un database relazionale è un insieme di tabelle che memorizzano dati strutturati in righe e colonne, e che possono essere collegate tra loro tramite relazioni.

Microsoft SQL Server offre diverse caratteristiche e vantaggi, tra cui:

* **Scalabilità**: Microsoft SQL Server può gestire basi di dati di grandi dimensioni, fino a petabyte, e supportare migliaia di utenti simultanei.
* **Sicurezza**: Microsoft SQL Server offre diversi livelli di sicurezza per proteggere i dati da accessi non autorizzati o dannosi. Tra questi ci sono l’autenticazione, l’autorizzazione, il controllo degli accessi, la crittografia, l’audit e il backup.
* **Performance**: Microsoft SQL Server offre diverse funzionalità per ottimizzare le performance delle query e delle operazioni sul database. Tra queste ci sono gli indici, le statistiche, i piani di esecuzione, le viste indicizzate, le stored procedure, le funzioni definite dall’utente e altro ancora.
* **Integrazione**: Microsoft SQL Server può integrarsi con altri strumenti e servizi di Microsoft e di terze parti, per offrire soluzioni complete e personalizzate per l’analisi, il reporting, il machine learning, il data warehousing e altro ancora.

Oltre alle caratteristiche di base, Microsoft SQL Server offre diversi strumenti e funzionalità per facilitare e migliorare la gestione, l’analisi e l’integrazione dei dati. Alcuni di questi strumenti e funzionalità sono:

* **SQL Server Management Studio (SSMS)**: è un ambiente integrato per l’amministrazione, la configurazione, il monitoraggio e lo sviluppo di database SQL Server. SSMS permette di eseguire query, progettare tabelle, creare diagrammi, gestire backup e ripristini, configurare la sicurezza e altro ancora.
* **SQL Server Data Tools (SSDT)**: è un set di strumenti per lo sviluppo di soluzioni di database, business intelligence e integrazione dei dati con SQL Server. SSDT permette di creare e modificare database, tabelle, viste, stored procedure, funzioni, trigger e altro ancora. SSDT include anche strumenti per la creazione di progetti di Integration Services, Analysis Services e Reporting Services.
* **SQL Server Integration Services (SSIS)**: è una piattaforma per la creazione e l’esecuzione di pacchetti che eseguono operazioni di integrazione dei dati, come l’estrazione, la trasformazione e il caricamento (ETL) di dati da diverse fonti a diverse destinazioni. SSIS permette anche di automatizzare e orchestrare attività di manutenzione e amministrazione del database.
* **SQL Server Analysis Services (SSAS)**: è una piattaforma per la creazione e il deployment di soluzioni di business intelligence basate su modelli multidimensionali o tabulari. SSAS permette di analizzare grandi quantità di dati aggregati da diverse fonti, applicare calcoli e funzioni analitiche, definire misure e dimensioni, creare gerarchie e relazioni e altro ancora.
* **SQL Server Reporting Services (SSRS)**: è una piattaforma per la creazione e il deployment di report interattivi basati su dati provenienti da diverse fonti. SSRS permette di progettare report con layout personalizzati, grafici, tabelle, mappe e altro ancora. SSRS offre anche funzionalità per la gestione, la distribuzione e la sicurezza dei report.

Per interagire con Microsoft SQL Server si usa il linguaggio SQL (Structured Query Language), uno standard internazionale per la definizione, la manipolazione e l’interrogazione dei dati in un database relazionale. SQL permette di creare tabelle, inserire, modificare, cancellare e ricercare dati, definire vincoli e relazioni, gestire transazioni e permessi e altro ancora. Ecco alcuni esempi di istruzioni SQL:

* **CREATE TABLE**: crea una nuova tabella nel database con le colonne specificate.
* **INSERT INTO**: inserisce una nuova riga nella tabella con i valori specificati.
* **SELECT**: recupera i dati dalla tabella in base ai criteri specificati.
* **UPDATE**: modifica i dati esistenti nella tabella in base ai criteri specificati. Esempio:
* **DELETE**: cancella i dati esistenti nella tabella in base ai criteri specificati.
  1. Immagine che contiene Elementi grafici, simbolo, logo, design

     Descrizione generata automaticamente**HTML, CSS e JavaScript: linguaggi per la creazione dell’interfaccia grafica del front-end**

HTML, CSS e JavaScript sono tre linguaggi fondamentali per lo sviluppo web, in particolare per la creazione dell’interfaccia grafica del front-end. Il front-end è la parte di un’applicazione web che viene visualizzata e interagita dall’utente finale tramite il browser. HTML, CSS e JavaScript hanno ruoli diversi ma complementari nel front-end:

* **HTML (HyperText Markup Language)**: è il linguaggio che definisce la struttura e il contenuto di una pagina web. HTML usa dei marcatori chiamati tag per creare degli elementi che rappresentano i componenti di una pagina, come titoli, paragrafi, immagini, link, form e altro ancora. Ogni elemento HTML può avere degli attributi che specificano ulteriori informazioni o proprietà, come l’identificatore, la classe, lo stile o il valore.
* **CSS (Cascading Style Sheets)**: è il linguaggio che definisce lo stile e il layout di una pagina web. CSS usa delle regole chiamate selettori per applicare degli stili agli elementi HTML, come il colore, il font, la dimensione, il margine, il padding e altro ancora. CSS permette anche di creare effetti visivi come sfumature, trasformazioni, transizioni e animazioni.
* **JavaScript**: è il linguaggio che definisce il comportamento e l’interattività di una pagina web. JavaScript usa delle istruzioni chiamate script per manipolare gli elementi HTML e CSS, reagire agli eventi generati dall’utente o dal browser, comunicare con il server o con altre pagine, creare contenuti dinamici e altro ancora.

Ecco delle buone pratiche da seguire quando si scrive codice HTML, CSS e JavaScript, per rendere le pagine web più accessibili, efficienti e manutenibili. Alcune di queste buone pratiche sono:

* **Seguire gli standard**: scrivere codice che rispetti le specifiche e le convenzioni stabilite dagli organismi di standardizzazione, come il W3C o l’ECMA. Questo assicura che il codice sia compatibile con i diversi browser e dispositivi, e che sia facilmente comprensibile da altri sviluppatori.
* **Usare la semantica**: usare gli elementi HTML appropriati per il contenuto che si vuole rappresentare, in modo da dare un significato e una struttura alla pagina. Questo aiuta i motori di ricerca, gli screen reader e gli altri strumenti ad interpretare correttamente la pagina e a fornire una migliore esperienza agli utenti.
* **Separare struttura, stile e comportamento**: usare HTML per definire la struttura e il contenuto della pagina, CSS per definire lo stile e il layout della pagina, e JavaScript per definire il comportamento e l’interattività della pagina. Questo rende il codice più modulare, riutilizzabile e facile da modificare.
* **Scrivere codice pulito e ordinato**: usare una formattazione consistente, indentare il codice, usare nomi significativi per le variabili, le classi e gli id, commentare il codice quando necessario, eliminare il codice inutile o duplicato. Questo rende il codice più leggibile e facile da debuggare.
* **Ottimizzare le performance**: ridurre il peso e il numero delle risorse (immagini, script, fogli di stile) da caricare nella pagina, usare tecniche di caching, minificazione e compressione per ridurre il tempo di caricamento della pagina. Questo migliora la velocità e la reattività della pagina.
  1. **AngularJS: framework per lo sviluppo di applicazioni web dinamiche e interattive**

AngularJS è un framework open source per lo sviluppo di applicazioni web di una sola pagina, cioè applicazioni che non richiedono il ricaricamento della pagina per cambiare il contenuto o la funzionalità. AngularJS è stato sviluppato nel 2010 da Google e dalla comunità di sviluppatori individuali, con l’obiettivo di semplificare e migliorare lo sviluppo di questo tipo di applicazioni.

AngularJS si basa sull’architettura MVC (Model-View-Controller), che divide l’applicazione in tre componenti principali:

* **Model**: è la parte che gestisce i dati e la logica dell’applicazione. Il modello è rappresentato da variabili JavaScript che possono essere impostate manualmente o recuperate da fonti esterne, come database o API.
* **View**: è la parte che mostra i dati e l’interfaccia grafica dell’applicazione. La vista è rappresentata dal codice HTML che contiene degli attributi personalizzati chiamati direttive, che indicano come legare i dati del modello agli elementi della pagina.
* **Controller**: è la parte che controlla il comportamento e l’interattività dell’applicazione. Il controller è rappresentato da una funzione JavaScript che riceve il modello e la vista come parametri, e può manipolare il modello o reagire agli eventi generati dall’utente o dal sistema.

AngularJS offre diversi vantaggi rispetto ad altri framework o librerie JavaScript, tra cui:

* **Data binding**: è il meccanismo che permette di sincronizzare automaticamente i dati tra il modello e la vista, senza dover scrivere codice aggiuntivo per aggiornare la pagina quando cambiano i dati, o viceversa.
* **Dependency injection**: è il meccanismo che permette di fornire alle funzioni o ai componenti dell’applicazione le dipendenze di cui hanno bisogno, senza doverle creare o cercare manualmente. Questo facilita la modularità, il riutilizzo e il test del codice.
* **Direttive**: sono degli attributi personalizzati che estendono la sintassi HTML, permettendo di creare nuovi elementi o comportamenti specifici per l’applicazione. Le direttive possono essere usate per creare componenti riutilizzabili, per applicare stili o effetti visivi, per gestire gli eventi o le transizioni e altro ancora.
* **Filtri**: sono delle funzioni che permettono di trasformare i dati prima di mostrarli nella vista, senza modificarli nel modello. I filtri possono essere usati per formattare i dati, per ordinare o filtrare delle liste, per applicare delle operazioni matematiche o logiche e altro ancora.
* **Routing**: è il meccanismo che permette di gestire le diverse sezioni o pagine dell’applicazione, usando dei percorsi (URL) specifici. Il routing permette di creare dei collegamenti tra le diverse parti dell’applicazione, di mantenere uno stato coerente tra il modello e la vista, e di supportare i link profondi (deep linking).

1. **Sistema web simulativo per la gestione di file Excel relativi a progetti sociali**

In questo capitolo si descrive il primo sistema web che è stato realizzato durante il tirocinio presso NTT Data. Si tratta di un sistema web simulativo per la gestione di file Excel relativi a dei progetti sociali finanziati da enti pubblici o privati. Il sistema consente di importare un file Excel per caricare i dati su un database relazionale, visualizzare i dati in una tabella nella pagina web con la possibilità di effettuare operazioni CRUD (Create, Read, Update, Delete), esportare in un nuovo file Excel i dati modificati dall’utente. Il sistema è stato sviluppato con il linguaggio C# in ambiente ASP.NET per la programmazione del back-end e il database Microsoft SQL Server per la gestione dei dati. Sono stati usati anche HTML, CSS e JavaScript per la creazione dell’interfaccia grafica del front-end. Il sistema è dotato di una struttura a tre livelli: presentazione, business e dati.

* 1. **Analisi delle specifiche e dei requisiti funzionali e non funzionali**

Il sistema web simulativo è stato realizzato come esercizio formativo su un vecchio progetto di NTT Data di un cliente che aveva la necessità di gestire i file Excel relativi ai progetti sociali che finanzia o a cui partecipa. La richiesta era quella di creare una soluzione web che permettesse di:

* Importare i file Excel contenenti i dati dei progetti sociali in un database relazionale;
* Visualizzare i dati dei progetti sociali in una tabella nella pagina web;
* Effettuare operazioni CRUD sui dati dei progetti sociali;
* Esportare i dati dei progetti sociali modificati in un nuovo file Excel.

In base a queste specifiche, sono stati individuati i seguenti requisiti funzionali e non funzionali per il sistema web simulativo.

Requisiti funzionali:

* Il sistema deve consentire all’utente di importare un file Excel contenente i dati dei progetti sociali in un database relazionale;
* Il sistema deve consentire all’utente di visualizzare i dati dei progetti sociali in una tabella nella pagina web;
* Il sistema deve consentire all’utente di effettuare operazioni CRUD sui dati dei progetti sociali;
* Il sistema deve consentire all’utente di esportare i dati dei progetti sociali modificati in un nuovo file Excel;

Requisiti non funzionali:

* Il sistema deve essere funzionale, sicuro e user-friendly;
* Il sistema deve essere compatibile con i principali browser web;
* Il sistema deve essere facilmente manutenibile e scalabile;
  1. **Progettazione dell’architettura a tre livelli: presentazione, business e dati**

Per realizzare l’applicativo, si è scelto di adottare una architettura a tre livelli: presentazione, business e dati. Questa architettura consente di separare le diverse funzionalità del sistema e di facilitarne lo sviluppo, la manutenzione e l’integrazione.

Il livello di presentazione è costituito dall’interfaccia grafica del sistema web, che permette all’utente di interagire con il sistema e di visualizzare i dati dei progetti sociali. Questo livello è stato realizzato con HTML, CSS e JavaScript, i linguaggi standard per la creazione di pagine web. Il livello di presentazione comunica con il livello di business tramite richieste HTTP.

Il livello di business è costituito dalla logica di business del sistema web, che gestisce le operazioni CRUD sui dati dei progetti sociali e le funzionalità di autenticazione e autorizzazione degli utenti. Questo livello è stato realizzato con il linguaggio C#, un linguaggio di programmazione orientato agli oggetti e basato sulla piattaforma .NET. Il livello di business comunica con il livello di dati tramite il protocollo ADO.NET.

Il livello di dati è costituito dal database relazionale che memorizza i dati dei progetti sociali importati dai file Excel. Questo livello è stato realizzato con il database Microsoft SQL Server, un sistema di gestione di basi di dati relazionali. Il livello di dati comunica con il livello di business tramite il protocollo ADO.NET.

La figura seguente illustra l’architettura a tre livelli del sistema web simulativo:

* 1. **Implementazione del sistema web con C#, Microsoft SQL Server, HTML, CSS e JavaScript**

In questa sezione si descrive il processo di implementazione del sistema web simulativo, usando i linguaggi e le tecnologie C#, Microsoft SQL Server, HTML, CSS e JavaScript. Si illustrano le principali componenti del sistema web, spiegando le scelte implementative e le funzionalità offerte.

Per implementare il sistema web si è usato lo strumento Visual Studio, che permette di creare e gestire progetti web in C#. Si è creato un progetto web di tipo ASP.NET Web Forms, che consente di creare pagine web dinamiche con controlli server e codice behind. Si è usato il framework .NET 4.5 per accedere alle classi e ai metodi necessari per lo sviluppo del sistema web.

Per importare, visualizzare e esportare i dati dei progetti sociali si è usata la libreria ClosedXML, che permette di manipolare i file Excel con il formato Open XML. Si è installata la libreria ClosedXML tramite il gestore pacchetti NuGet di Visual Studio. Si è usata la libreria ClosedXML per creare e leggere gli oggetti Workbook, che rappresentano i file Excel, e gli oggetti Worksheet, che rappresentano i fogli di calcolo dei file Excel.

Per memorizzare i dati dei progetti sociali si è usato il database Microsoft SQL Server, che permette di creare e gestire database relazionali. Si è usato lo strumento SQL Server Management Studio per creare e interrogare il database. Si è usato il database locale (LocalDB) di Visual Studio per facilitare lo sviluppo e il testing del sistema web. Si è definita la stringa di connessione al database nel file di configurazione “Web.config” del progetto web.

Per definire la struttura delle pagine web si è usato il linguaggio HTML, che permette di creare elementi e attributi che rappresentano il contenuto delle pagine web. Si è usato il linguaggio HTML per creare le pagine web di importazione, visualizzazione ed esportazione dei dati. Si è usato il linguaggio HTML per inserire i controlli server nelle pagine web, come il controllo FileUpload per scegliere il file da importare, il controllo GridView per mostrare i dati in forma tabellare e i pulsanti per le operazioni CRUD e di esportazione.

Per definire lo stile delle pagine web si è usato il linguaggio CSS, che permette di creare regole di stile che determinano l’aspetto degli elementi HTML delle pagine web. Si è usato il linguaggio CSS per impostare il colore di sfondo, il font, il margine e l’allineamento delle pagine web. Si è usato il linguaggio CSS per impostare lo stile dei controlli server nelle pagine web, come il colore, la dimensione e il bordo del controllo FileUpload, del controllo GridView e dei pulsanti.

Per definire il comportamento delle pagine web si è usato il linguaggio JavaScript, che permette di creare script che vengono eseguiti dal browser dell’utente. Si è usato il linguaggio JavaScript per validare l’input dell’utente nelle pagine web, come la selezione del file da importare e la conferma dell’eliminazione dei record. Si è usato anche il linguaggio JavaScript per aggiungere effetti dinamici alle pagine web, come l’alternanza dei colori delle righe della tabella HTML.

Per definire la logica applicativa delle pagine web si è usato il linguaggio C#, che permette di creare codice behind che viene eseguito dal server. Si è usato il linguaggio C# per implementare le funzionalità del sistema web simulativo, come l’importazione, la visualizzazione e l’esportazione dei dati. Si è usato il linguaggio C# per creare e gestire gli oggetti Workbook e Worksheet della libreria ClosedXML, per creare e gestire le connessioni, le query e i comandi al database Microsoft SQL Server, per creare e gestire gli oggetti DataTable che contengono i dati del database, per creare e gestire le variabili di sessione che memorizzano il nome della tabella del database, per creare e gestire gli eventi dei controlli server nelle pagine web, come il click dei pulsanti e la modifica del controllo GridView.

* 1. **Testing del sistema web con Visual Studio e SQL Server Management Studio**

In questa sezione si descrive il processo di testing del sistema web simulativo, usando gli strumenti Visual Studio e SQL Server Management Studio. Si illustrano i test effettuati per verificare il corretto funzionamento delle funzionalità del sistema web e la qualità dei dati importati ed esportati.

Per testare le funzionalità del sistema web si è usato lo strumento Visual Studio, che permette di eseguire e debuggare il codice sorgente del sistema web in un ambiente locale. Si sono usati alcuni file Excel forniti dal committente come input per il sistema web e si sono verificati i risultati ottenuti in output.

Per testare l’importazione dei dati da un file Excel si è eseguita la pagina web di importazione e si è selezionato un file Excel valido dal proprio dispositivo. Si è verificato che il file venisse caricato correttamente sul server nella cartella “Uploads” e che venisse creata una tabella del database con i dati importati dal file Excel. Si è verificato anche che il nome della tabella del database venisse salvato in una variabile di sessione e che venisse mostrato un messaggio di successo all’utente.

Per testare la visualizzazione e modifica dei dati del database si è eseguita la pagina web di visualizzazione e si è verificato che i dati del database venissero mostrati correttamente in una tabella HTML e che fosse possibile ordinare i dati per ogni colonna. Si è verificato anche che fosse possibile inserire, modificare ed eliminare i record del database tramite i pulsanti per le operazioni CRUD e che le modifiche venissero riflette nel database.

Per testare l’esportazione dei dati in un file Excel si è eseguita la pagina web di esportazione e si è cliccato sul pulsante per avviare l’esportazione. Si è verificato che venisse creato un file Excel con i dati del database e che fosse possibile scaricarlo sul proprio dispositivo. Si è verificato anche che il file Excel avesse lo stesso formato e contenuto del file Excel originale usato per l’importazione.

Per testare la qualità dei dati importati ed esportati si è usato lo strumento SQL Server Management Studio, che permette di gestire e interrogare il database Microsoft SQL Server. Si sono usate alcune query SQL per verificare la correttezza, la completezza, la consistenza e l’unicità dei dati importati ed esportati.

Per testare la correttezza dei dati si è verificato che i valori delle celle del file Excel corrispondessero ai valori delle colonne della tabella del database e viceversa. Si è confrontato il risultato delle query SQL per selezionare tutti i dati dalla tabella del database con il contenuto del file Excel, verificando che non ci fossero errori di trasformazione o conversione dei dati.

Per testare la completezza dei dati si è verificato che tutti i dati presenti nel file Excel fossero importati nella tabella del database e viceversa. Si è confrontato il risultato delle query SQL per contare il numero di record nella tabella del database con il numero di righe del file Excel, verificando che non ci fossero perdite o omissioni di dati.

Per testare la consistenza dei dati si è verificato che i dati importati ed esportati rispettassero le regole di validità e integrità definite dal committente. Si sono confrontati i risultati delle query SQL per verificare le regole di validità e integrità con le aspettative, verificando che non ci fossero violazioni o incongruenze nei dati.

Per testare l’unicità dei dati si è verificato che non ci fossero record duplicati nella tabella del database. Si sono confrontati i risultati delle query SQL per contare il numero di occorrenze di ogni valore della chiave primaria (nProtocollo) con le aspettative, verificando che non ci fossero valori ripetuti nei dati.

* 1. **Risultati ottenuti e valutazione del sistema web**

Per valutare il sistema web si è usato il metodo della sperimentazione, misurando le prestazioni del sistema web in termini di tempo di esecuzione, uso delle risorse e qualità dei dati. Si sono usati alcuni file Excel come input per il sistema web e si sono registrati i dati relativi alle prestazioni del sistema web in output.

I risultati ottenuti hanno dimostrato che il sistema web simulativo soddisfa tutti i requisiti funzionali e non funzionali, mostrando la validità e l’efficacia del sistema web.

I risultati ottenuti hanno anche evidenziato che il sistema web simulativo offre dei vantaggi rispetto agli altri sistemi web simili o alternativi, come la semplicità d’uso, l’affidabilità, l’efficienza, la scalabilità, l’estendibilità e la manutenibilità.

I possibili svantaggi del sistema web simulativo, come la dipendenza dal formato e dal contenuto dei file Excel, la necessità di caricare e scaricare i file Excel e la mancanza di funzionalità avanzate o aggiuntive, sono compensati dai vantaggi offerti dal sistema web simulativo e sono in linea con le specifiche e i requisiti del committente e degli utenti. Inoltre, questi possibili svantaggi potrebbero essere superati o mitigati con delle modifiche o delle estensioni del sistema web simulativo.