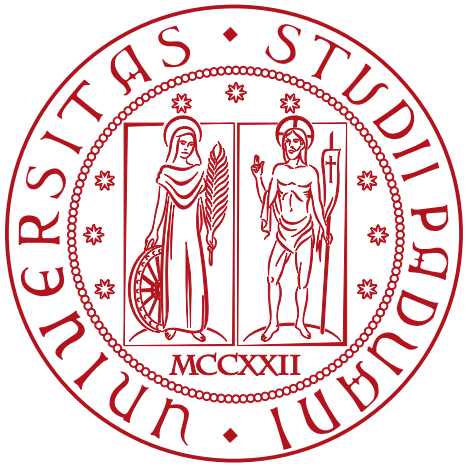
Università degli studi di Padova

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica



Vision Ticketing: progettazione e sviluppo di un portale web integrato per l'automazione dei servizi di assistenza clienti

Tesi di laurea triennale

Relatore
Prof. Marco Zanella

Laureanda
Valentina Schivo

ANNO ACCADEMICO 2024-2025

© Valentina Schivo, Dicembre 2025. Tutti i diritti riservati. Tesi di laurea triennale: *Vision Ticketing: progettazione e sviluppo di un portale web integrato per l'automazione dei servizi di assistenza clienti.*

Sommario

La gestione tradizionale delle richieste di assistenza comporta tempi elevati e un notevole carico di lavoro per il personale. La digitalizzazione di questi processi rappresenta un passaggio strategico per le aziende di software gestionali, poiché permette di migliorare la qualità del servizio e ottimizzare le risorse disponibili.
Il progetto Vision Ticketing, oggetto del presente elaborato, si inserisce nel percorso di stage curriculare presso VisioneImpresa S.r.l., con l'obiettivo di ideare e implementare un portale di ticketing, integrato al gestionale Vision Enterprise.
La trattazione descrive le principali fasi progettuali del portale di ticketing volto ad automatizzare la gestione delle richieste di assistenza, limitando le operazioni manuali e mantenendo continuità e coerenza con le procedure aziendali esistenti.
L'integrazione di tale sistema al gestionale Vision Enterprise, supportata dallo sviluppo di API in C# .NET, da modifiche al database SQL Server e da un attento adattamento del frontend tramite tecnologie containerizzate come Docker e DDEV, ha garantito consistenza dei dati, tracciabilità delle operazioni e un significativo miglioramento dell'efficienza complessiva del processo di assistenza.
Il risultato è una piattaforma di ticketing operativa, in grado di offrire un servizio più rapido e trasparente ai clienti e, al tempo stesso, di valorizzare le risorse interne attraverso l'automazione dei processi.

Abstract

The traditional management of support requests involves long response times and a considerable workload for staff. The digitalization of these processes represents a strategic step for companies developing management software, as it allows them to improve service quality and optimize available resources.
The Vision Ticketing project, which is the subject of this thesis, was carried out during a curricular internship at VisioneImpresa S.r.l., with the objective of designing and implementing a ticketing portal integrated with the Vision Enterprise management system.
The document describes the main design and development phases of the ticketing portal, aimed at automating the handling of support requests, reducing manual operations, and maintaining continuity and consistency with the company's existing procedures.
The integration of this system with Vision Enterprise, supported by the development of APIs in C# .NET, modifications to the SQL Server database, and a careful adaptation of the frontend through containerized technologies such as Docker and DDEV, ensured data consistency, operational traceability, and a significant improvement in the overall efficiency of the support process.
The result is a fully functional ticketing platform capable of providing faster and more transparent service to customers while enhancing internal resources through process automation.

Convenzioni tipografiche

La presente trattazione adotta il seguente schema di convenzioni tipografiche:

- Gli acronimi, le abbreviazioni e i termini di uso non comune menzionati vengono definiti nel glossario, situato alla fine del documento (p. 71).
  Per la prima occorrenza dei termini riportati nel glossario viene utilizzata la seguente nomenclatura: *termine*$_G$;
- I termini facenti parti del gergo tecnico sono evidenziati con il carattere *corsivo*;
- I nomi di funzioni o variabili appartenenti ad un linguaggio di programmazione vengono scritte con un carattere `monospaziato`;
- Le citazioni ad un libro o ad una risorsa presente nella bibliografia (p. 76) saranno affiancate dal rispettivo numero identificativo, es. [1]. É inoltre disponibile la lista dei siti web consultati nella sitografia (p. 77);
- I blocchi di codice sono rappresentati nel seguente modo:

```c
int main() {
    printf("Tesi di Laurea in Informatica\n");
    return 0;
}
```

Codice 1: Codice d'esempio.

Indice

1 Introduzione 1

1.1 L'azienda 1
1.2 L'idea e l'interesse 2

2 Descrizione dello stage 3

2.1 Il progetto Vision Ticketing 3
2.2 Contesto operativo e svolgimento delle attività 4
2.3 Finalità 5
2.3.1 Uso Previsto 5
2.3.2 Utenti finali 5
2.3.3 Prodotti attesi 6
2.4 Analisi preventiva dei rischi 6
2.4.1 Resistenza al cambiamento da parte degli utenti 6
2.4.2 Integrazione con il gestionale esistente 6
2.4.3 Appredimento delle nuove tecnologie 7
2.4.4 Ritardi nella consegna delle attività 7
2.5 Strumenti e tecnologie 7
2.5.1 Strumenti 7
Microsoft Visual Studio 7
Microsoft SQL Server Management Studio 8
Internet Information Services 8
Outlook / Posta elettronica aziendale 8
SoapUI / Postman 9
Azure Active Directory 9
2.5.2 Tecnologie 10
C# (.NET Framework) 10
ASP.NET 10
DLL RegistraDoc.net 10
Microsoft SQL Server 11
WebService (Servizi Web SOAP) 11
PHP 12
HTML, CSS e JavaScript 12
Dependency Injection 12

Microsoft Graph API 13
OAuth2 – Client Credentials Flow 13

3 Analisi dei requisiti 15

3.1 Classificazione dei requisiti 15
3.2 Requisiti funzionali 16
3.3 Requisiti non funzionali 18

4 Il gestionale Vision 21

4.1 Panoramica e gestione delle richieste 21
4.2 Introduzione delle API 22
4.3 Database 23

5 Il portale web 27

5.1 Architettura 27
5.2 Funzionalità del portale 27
5.2.1 Accesso e autenticazione 27
5.2.2 Area anagrafica 28
5.2.3 Area ticket 29
Creazione ticket 30
Lista ticket 31
Dettaglio ticket 31
5.3 Integrazione con il WebService 33
5.3.1 Flusso di esecuzione delle API 33
5.3.2 Processo di integrazione 33
5.3.3 La classe WebServiceNf_Ticketing 34
5.3.4 La classe WebService 35
5.4 Sviluppo e adattamento del portale 36
5.4.1 Adattamento del codice esistente 36
5.4.2 Design e ottimizzazioni 37
5.5 Evoluzione del portale 41
5.5.1 Pagina di login 42
5.5.2 Pagina dell'area anagrafica 43
5.5.3 Pagina di ricerca e visualizzazione lista 44

6 Il modulo EmailTicketReader 47

6.1 Introduzione e obiettivi 47
6.2 Architettura e funzionamento 47
6.3 Implementazione 49
6.3.1 Componenti principali 49
Program.cs 49

OAuth2EmailService.cs 51
6.4 Analisi e prospettive del modulo 52
6.4.1 Stato attuale 52
6.4.2 Sviluppi futuri 53

7 Test e validazione 55

7.1 Obiettivo dei test 55
7.2 Ambiente di test 56
7.3 Classificazione dei test 56
7.4 Test funzionali 56
7.5 Test di integrazione 58
7.6 Requisiti soddisfatti 59
7.6.1 Requisiti funzionali 60
7.6.2 Requisiti non funzionali 62

8 Conclusioni 65

8.1 Risultati 65
8.2 Sviluppi futuri 66
8.2.1 Integrazione con moviTICKET 66
8.2.2 Integrazione del VoIP con speech-to-text 66
8.3 Valutazione personale 66

Ringraziamenti 69

Glossario 71

Bibliografia 76

Sitografia 77

Elenco delle Figure

Figura 1 Logo di VisioneImpresa. 2
Figura 2 Flusso della gestione delle richieste di assistenza. 22
Figura 3 Tabella WebReparto. 24
Figura 4 Tabella Comm. 25
Figura 5 Tabella DocTes2. 25
Figura 6 Tabella ImpostaB2B. 26
Figura 7 Schermata di login del portale. 28
Figura 8 Area anagrafica del portale. 29
Figura 9 Area ticket del portale. 29
Figura 10 Area creazione dei ticket. 30
Figura 11 Visualizzazione lista dei ticket. 31
Figura 12 Schermata dettaglio del ticket. 32
Figura 13 Banner inserimento del sollecito. 32
Figura 14 Banner chiusura del ticket. 33
Figura 15 UML di WebServiceNf_Ticketing.cs. 35
Figura 16 UML di WebService.cs. 36
Figura 17 Confronto tra palette di OfficeGroup e VisioneImpresa. . 37
Figura 18 Pagina di login del vecchio portale. 42
Figura 19 Pagina di login del nuovo portale. 43
Figura 20 Area anagrafica del portale di test. 44
Figura 21 Area anagrafica del nuovo portale. 44
Figura 22 Sezione lista del portale di test. 45
Figura 23 Sezione lista del nuovo portale. 45
Figura 24 Diagramma di sequenza EmailTicketReader. 49
Figura 25 Dettaglio documenti CH creati con *EmailTicketReader*. . 53

Elenco delle Tabelle

Tabella 1 Tracciamento dei requisiti funzionali obbligatori. 16
Tabella 2 Tracciamento dei requisiti funzionali desiderabili 17
Tabella 3 Tracciamento dei requisiti funzionali opzionali. 17
Tabella 4 Tracciamento dei requisiti non funzionali obbligatori. . . . 18
Tabella 5 Tracciamento dei requisiti non funzionali desiderabili. . . 19
Tabella 6 Riepilogo dei requisiti. 19
Tabella 7 Tracciamento dei test funzionali. 57
Tabella 8 Tracciamento dei test di integrazione. 58
Tabella 9 Stato dei requisiti funzionali obbligatori. 60
Tabella 10 Stato dei requisiti funzionali desiderabili 61
Tabella 11 Stato dei requisiti funzionali opzionali. 62
Tabella 12 Stato dei requisiti non funzionali obbligatori. 62
Tabella 13 Stato dei requisiti non funzionali desiderabili. 63

Elenco dei Codici Sorgente

Codice 1 Codice d'esempio. v
Codice 2 Configurazione global.php. 37
Codice 3 Dettaglio codice view.ctp 39
Codice 4 Dettaglio codice add.js 40
Codice 5 Dettaglio codice TicketsController.php 41
Codice 6 Dettaglio codice WebServiceNf_Ticketing.cs 41
Codice 7 Configurazione appsettings.json. 51

Capitolo 1

Introduzione

In questo capitolo viene presentata l'azienda e le motivazioni che hanno portato alla nascita del progetto Vision Ticketing.

1.1 L'azienda

VisioneImpresa S.r.l. si configura come una realtà consolidata nel panorama italiano della consulenza aziendale e dello sviluppo software gestionale. Situata nella zona industriale di Pernumia (PD), la sua missione centrale consiste nell'affiancare le imprese nel processo di innovazione digitale, fornendo soluzioni tecnologiche avanzate e servizi di consulenza, mirati a ottimizzare i processi gestionali e operativi.
Fondata negli anni '80, l'offerta di VisioneImpresa si sviluppa principalmente in due ambiti complementari.
Da un lato, l'azienda è specializzata nello sviluppo e nella distribuzione di software gestionali, con la propria suite modulare *Vision Enterprise* come prodotto di punta. Dall'altro lato, VisioneImpresa offre un insieme di servizi di consulenza e supporto, che comprendono attività di consulenza organizzativa e fiscale, l'outsourcing di processi come paghe e contabilità, e formazione e assistenza tecnica continuativa.
Il gestionale, sviluppato internamente, è progettato per rispondere in modo flessibile alle esigenze di aziende di diverse dimensioni, supportando funzioni fondamentali quali contabilità, gestione del magazzino, produzione, fatturazione elettronica, *CRM*$_G$ e business intelligence.
Attualmente le aziende in contratto di assistenza gestite direttamente da VisioneImpresa sono circa 340 mentre le licenze attive presso i rivenditori sono circa 850.
Grazie a questo approccio integrato l'azienda non si limita a fornire un software, ma accompagna il cliente come partner affidabile, seguendolo dall'analisi dei bisogni fino all'implementazione e al mantenimento della soluzione, garantendo così un supporto concreto ed efficienza operativa.

Figura 1: Logo di VisioneImpresa.

1.2 L'idea e l'interesse

Negli ultimi quattro anni il flusso medio di richieste annuali di assistenza ammonta a 6.300 richieste, motivo per cui è di rilevante importanza per VisioneImpresa ottimizzare i sistemi di gestione per la consulenza e il supporto tecnico.
L'idea del progetto nasce, quindi, dall'esigenza di digitalizzare e automatizzare la gestione delle richieste di assistenza, sostituendo i processi manuali, basati su telefonate ed email, con un sistema integrato di web ticketing che migliori l'efficienza operativa, la tracciabilità delle segnalazioni per i clienti e la qualità del servizio offerto.
L'interesse alla realizzazione del presente progetto nasce da due prospettive. Da un lato, l'azienda VisioneImpresa ha manifestato la necessità di innovare i propri processi gestionali, riflettendo la volontà dell'azienda di mantenere la propria competitività sul mercato e rendersi più appetibile per nuovi clienti, senza rinunciare all'esperienza maturata nel corso degli anni.
Dall'altro lato, vi è un interesse personale che si concentra sull'analisi e la comprensione dei processi di innovazione all'interno di realtà consolidate, come introdurre nuove metodologie in modo graduale e sostenibile senza compromettere le strutture esistenti e quali strategie consentono di rinnovarsi senza dover ripartire completamente da zero.
Questo duplice interesse ha costituito il punto di partenza per lo sviluppo del progetto *Vision Ticketing*, con l'obiettivo di conciliare esigenze di innovazione e continuità operativa.

Capitolo 2

Descrizione dello stage

Una volta definite l'azienda e le ragioni del progetto, in questo capitolo si descrivono gli obiettivi del progetto di stage e i prodotti software attesi, congiuntamente al contesto operativo e all'analisi preventiva dei rischi.

2.1 Il progetto Vision Ticketing

L'obiettivo generale del progetto di stage *Vision Ticketing* consiste nell'automatizzare la fase di ricezione e gestione delle richieste di assistenza, presso l'azienda VisioneImpresa, al fine di ridurre l'intervento del personale, liberare risorse da destinare ad attività di maggior valore e migliorare la qualità complessiva del servizio clienti.
A partire da un'analisi dei processi esistenti, che prevedono l'inserimento manuale delle richieste tramite telefonate o email nel gestionale *Vision Enterprise*, il progetto di stage si propone di introdurre strumenti digitali integrati, capaci di ottimizzare il flusso di lavoro e garantire una maggiore tracciabilità e tempestività nelle comunicazioni con i clienti.

Gli obiettivi individuati sono i seguenti:

1. Realizzare un nuovo portale di web ticketing, derivato da una soluzione già sperimentata dall'azienda consociata, denominata OfficeGroup, che permetta l'inserimento diretto delle richieste da parte dei clienti;

2. Sviluppare nuove funzionalità applicative, tramite *API*$_G$ *SOAP*$_G$, necessarie all'integrazione dei sistemi coinvolti e al miglioramento dell'automazione dei processi di registrazione delle richieste di assistenza;

3. Integrare il portale web con il gestionale *Vision*, attraverso l'utilizzo di *RegistraDoc.net*, una libreria personalizzata di VisioneImpresa;

4. Automatizzare la gestione delle email in ingresso destinate alla casella *assistenza@vsh.it*, in modo che le richieste vengano automaticamente convertite in documenti all'interno del gestionale *Vision*;

5. Rivedere la libreria di interfacciamento utilizzata per la registrazione delle richieste provenienti dall'app mobile *moviTICKET*, al fine di integrarne pienamente il funzionamento con il gestionale aziendale;

6. Integrare l'impianto telefonico VoIP per consentire la creazione automatica di chiamate a partire dalla scheda anagrafica del cliente nel gestionale, implementando tecniche di *speech-to-text* attraverso l'utilizzo di librerie già disponibili in commercio, trasformando le chiamate vocali in testo da archiviare nei documenti del gestionale.

2.2 Contesto operativo e svolgimento delle attività

Lo stage si è svolto all'interno del contesto operativo di VisioneImpresa S.r.l., e le attività di stage sono state svolte in presenza, sotto la supervisione del tutor aziendale Francesco Turra, amministratore dell'azienda, con meeting periodici per monitorare l'avanzamento dei lavori, risolvere eventuali criticità e definire le priorità operative.

Le attività di progetto sono state organizzate secondo una metodologia strutturata, comprendente:

- Analisi dei processi esistenti e del database aziendale;
- Definizione dei requisiti funzionali e tecnici in collaborazione con il tutor e il personale aziendale;
- Progettazione e sviluppo delle funzionalità, con iterazioni periodiche per raccogliere feedback;
- Test delle soluzioni implementate e verifica dell'integrazione con il gestionale *Vision Enterprise*;
- Documentazione delle attività svolte.

Questa organizzazione delle attività ha permesso un inserimento graduale nel contesto aziendale, facilitando un apprendimento tecnico costante. Le attività di analisi, progettazione e testing si sono svolte in modo ciclico, dividendo il progetto in più fasi incrementali tramite un approccio di sviluppo agile, per perfezionare le funzionalità implementate.

Durante lo svolgimento dello stage è stata fondamentale la collaborazione con diversi membri del personale:

- Stefania Baracco, addetta all'help desk e al reparto di assistenza *Vision*, per il supporto nella comprensione delle procedure operative di registrazione e le modalità di comunicazione con i clienti;
- Dario Minella, programmatore del reparto sviluppo software, per il supporto tecnico e collaboratore nella definizione e implementazione delle funzionalità richieste;
- Filippo Donà, programmatore dell'azienda OfficeGroup, per lo sviluppo del frontend e l'integrazione delle componenti software del progetto.

2.3 Finalità

2.3.1 Uso Previsto

Il portale *Vision Ticketing* consente ai clienti di VisioneImpresa di inserire in autonomia le proprie richieste di assistenza, allegando eventuali documenti o screenshot utili alla comprensione del problema.
Le segnalazioni vengono automaticamente registrate all'interno del gestionale come documenti di tipo CH_G, riducendo la necessità di interventi manuali da parte del personale aziendale. L'interfaccia web offre inoltre agli utenti la possibilità di monitorare lo stato di avanzamento dei ticket e visualizzare gli aggiornamenti effettuati dal reparto tecnico.
Dal lato aziendale, invece, il reparto assistenza può gestire l'assegnazione delle attività e aggiornare in tempo reale le informazioni relative ai ticket mentre la direzione tecnica può analizzare i dati raccolti per individuare le aree più critiche e valutare la qualità del servizio offerto.

2.3.2 Utenti finali

Il portale *Vision Ticketing* è progettato per soddisfare due categorie di utenti distinte.
Per VisioneImpresa, il sistema consente una gestione centralizzata e automatizzata delle richieste di supporto, integrando le informazioni direttamente nel gestionale *Vision Enterprise*.
Per i clienti e i rivenditori, invece, il portale *Vision Ticketing* costituisce un canale diretto e intuitivo per comunicare con il servizio di assistenza, favorendo una maggiore trasparenza e autonomia nella gestione delle richieste.

2.3.3 Prodotti attesi

Al termine dello stage, dallo sviluppo del progetto *Vision Ticketing* sono attesi i seguenti prodotti:

- **Soluzione software**: realizzazione e implementazione del sistema di web ticketing, comprensivo delle integrazioni con il gestionale aziendale, dello sviluppo delle API e del portale web dedicato ai clienti;

- **Documentazione tecnica e materiale formativo**: produzione della documentazione progettuale e tecnica, a supporto del sistema sviluppato, comprendente la descrizione delle funzionalità aggiunte, le procedure dei test e il manuale utente.

2.4 Analisi preventiva dei rischi

Durante la fase di analisi del progetto sono stati individuati alcuni potenziali rischi con le relative strategie di mitigazione.

2.4.1 Resistenza al cambiamento da parte degli utenti

Rischio: Medio.
Uno dei principali rischi individuati riguarda la possibile resistenza degli utenti all'adozione delle nuove procedure digitalizzate. Poiché VisioneImpresa è un'azienda con processi consolidati e fortemente basati su comunicazioni telefoniche ed email, l'introduzione di un nuovo portale web per la gestione delle richieste di assistenza può generare iniziali difficoltà di adattamento sia da parte del personale aziendale sia da parte della clientela di VisioneImpresa. Per mitigare questo rischio, è stato previsto il coinvolgimento attivo degli operatori sin dalle fasi iniziali del progetto, mediante momenti di confronto e discussione, ed è stata mantenuta la possibilità di assistenza tradizionale ai clienti, per consentire un periodo di adattamento e non compromettere la qualità del servizio percepita.

2.4.2 Integrazione con il gestionale esistente

Rischio: Medio.
Un rischio tecnico rilevante è rappresentato dalle possibili criticità nell'integrazione tra il nuovo sistema web di ticketing e il gestionale *Vision Enterprise*, già ampiamente utilizzato in azienda. La riscrittura delle API e l'aggiornamento dei database possono introdurre incompatibilità o anomalie nei processi. Per prevenirle sono stati effettuati

frequenti test per garantire la continuità operativa e l'allineamento dei dati con il gestionale.

2.4.3 Appredimento delle nuove tecnologie

Rischio: Alto.
Durante lo svolgimento del progetto è stato necessario acquisire familiarità con tecnologie e strumenti nuovi, come framework di sviluppo o metodologie operative adottate in azienda. Per mitigare il rischio, sono stati dedicati dei momenti di formazione autonoma e guidata, attraverso la consultazione di documentazione online e confronto con i colleghi, che assieme all'adozione di un approccio iterativo ha consentito di apprendere progressivamente le nuove tecnologie.

2.4.4 Ritardi nella consegna delle attività

Rischio: Alto.
A causa della complessità del progetto e il numero di componenti software coinvolti, è stata considerata la possibilità di ritardi nella realizzazione delle varie funzionalità e incompletamento di alcuni degli obiettivi del progetto. Questo rischio è stato mitigato attraverso una pianificazione settimanale delle attività di progetto e momenti di confronto periodici con il tutor aziendale, al fine di garantire l'avanzamento degli obiettivi principali del progetto.

2.5 Strumenti e tecnologie

2.5.1 Strumenti

Microsoft Visual Studio

Motivazione della scelta
Visual Studio[1] è un ambiente di sviluppo *IDE*$_G$ largamente utilizzato per applicazioni .NET e in particolare per lo sviluppo in C#. Essendo il portale ticketing sviluppato in C#, l'utilizzo di Visual Studio garantisce pieno supporto al linguaggio, agli strumenti di debugging e alla gestione dei file dell'intero progetto.

Ruolo nello sviluppo
- Scrittura del codice sorgente in C#;
- Gestione del progetto;

[1] https://visualstudio.microsoft.com/it/

- Supporto al debugging.

Microsoft SQL Server Management Studio

Motivazione della scelta

La scelta di *SQL Server Management Studio*[2] è risultata naturale poiché rappresenta lo strumento già in uso all'interno dell'azienda per la gestione del database relazionale. L'adozione di *SSMS*$_G$ ha garantito continuità con le pratiche operative aziendali e piena compatibilità con il nuovo sistema di ticketing.

Ruolo nello sviluppo

- Creazione delle tabelle di integrazione con il gestionale;
- Manutenzione e monitoraggio della base dati con backup e query;
- Verifica e testing dell'integrazione dati.

Internet Information Services

Motivazione della scelta

Per l'hosting del portale web di ticketing e delle API in ambiente Windows Server è stato scelto *IIS*$_G$, nativo per sistemi Windows. Questo web server assicura compatibilità con ASP.NET/C# e integrazione con l'ambiente Windows, semplificando il deployment, la configurazione della sicurezza e la gestione dell'ambiente di sviluppo.

Ruolo nello sviluppo

- hosting del portale web di ticketing;
- pubblicazione e gestione delle API;
- configurazione dell'ambiente.

Outlook / Posta elettronica aziendale

Motivazione della scelta

Outlook[3] rappresenta il sistema di posta elettronica già in uso all'interno dell'azienda e l'adozione di questa piattaforma ha permesso di integrare in modo diretto il flusso di comunicazioni con il sistema di ticketing, sfruttando le funzionalità native offerte dall'ambiente Microsoft, come la gestione delle regole di inoltro e le automazioni per l'elaborazione dei messaggi.

[2] https://learn.microsoft.com/en-us/ssms/sql-server-management-studio-ssms
[3] https://www.microsoft.com/it-it/microsoft-365/outlook/

Ruolo nello sviluppo

- Intercettazione automatica delle email;
- Elaborazione dei messaggi ricevuti e conversione automatica in ticket nel sistema;
- Integrazione con i servizi Microsoft per la gestione sicura delle credenziali e dei flussi di autenticazione.

SoapUI / Postman

Motivazione della scelta

Per testare la corretta configurazione delle chiamate API durante il loro sviluppo sono stati utilizzati gli strumenti *SoapUI*[4] e *Postman*[5], che consentono di simulare richieste, verificare risposte e testare gli scenari delle API.

Ruolo nello sviluppo

- Validazione e test delle API.

Alternative considerate

Durante la fase di valutazione sono stati presi in considerazione anche strumenti alternativi come ReadyAPI e Katalon Studio, anch'essi utilizzati nel campo del testing di API. Tuttavia, Postman e SoapUI sono stati preferiti poiché già integrati nel flusso di lavoro aziendale e supportano un'ampia varietà di protocolli.

Azure Active Directory

Motivazione della scelta

Per la gestione delle credenziali del servizio email e per abilitare l'autenticazione tramite il flusso OAuth2 Client Credentials è stato utilizzato *Azure Active Directory*[6], denominato ora Entra ID. Questa piattaforma cloud Microsoft consente di gestire identità, autorizzazioni, e flussi di autenticazione standard rendendo sicura l'integrazione del servizio mail con il gestionale e il database.

Ruolo nello sviluppo

- Registrazione dell'applicazione del servizio email in Azure;
- Configurazione del flusso OAuth2 per ottenere token di accesso da usare nei servizi backend;
- Gestione autorizzazioni e protezione degli endpoint API.

[4]https://www.soapui.org/

[5]https://www.postman.com/

[6]https://azure.microsoft.com/it-it/products/active-directory-external-identities

2.5.2 Tecnologie

C# (.NET Framework)

Motivazione della scelta
C# è un linguaggio di programmazione maturo e profondamente integrato con l'ecosistema Microsoft, e l'impiego del framework .NET come piattaforma di runtime è risultato coerente con l'infrastruttura aziendale esistente e con il team di sviluppo, facilitando la scrittura della logica applicativa lato server e delle API di comunicazione tra il portale ticketing e il gestionale *Vision*.

Ruolo nello sviluppo
- Sviluppo dei servizi lato server;
- Implementazione delle API.

ASP.NET

Motivazione della scelta
Il framework ASP.NET, basato sul pattern *MVC*$_G$, è stato scelto per lo sviluppo del portale web poiché consente una chiara separazione tra i livelli di presentazione, logica e accesso ai dati. L'utilizzo di ASP.NET si è rivelato una scelta naturale in quanto il codice del portale web era già stato parzialmente sviluppato e condiviso dall'azienda consociata OfficeGroup, consentendo il riutilizzo e l'adattamento dei componenti esistenti.

Ruolo nello sviluppo
- Costruzione del portale web;
- Comunicazione con il database e le API;
- Riutilizzo e adattamento del codice per ridurre duplicazioni e mantenere la coerenza architetturale.

DLL RegistraDoc.net

Motivazione della scelta
Nel gestionale *Vision*, la libreria *RegistraDoc.net* rappresenta un componente interno sviluppato in ambiente .NET che gestisce la registrazione automatica dei documenti nel sistema, come le chiamate di assistenza. Questa DLL incapsula le logiche di validazione e aggiornamento dei dati, garantendo coerenza tra le varie entità gestionali.
Nel progetto *Vision Ticketing*, tuttavia, la DLL non è stata utilizzata direttamente, in quanto l'integrazione con il gestionale è avvenuta tramite tabelle su Microsoft SQL Server e servizi web, in particolare

WebService, che a loro volta invocano in modo indiretto le funzioni di *RegistraDoc.net*.

Ruolo nello sviluppo
- Gestione indiretta dei documenti CH.

Microsoft SQL Server

Motivazione della scelta
Microsoft SQL Server offre strumenti consolidati per l'archiviazione e la gestione dei dati. La scelta di questa tecnologia è risultata naturale in quanto nell'azienda le informazioni relative ai documenti CH erano già archiviate all'interno delle tabelle del database SQL Server del gestionale *Vision*.

Ruolo nello sviluppo
- Archiviazione dei dati relativi ai ticket;
- Integrazione diretta con le tabelle del gestionale *Vision Enterprise*;
- Utilizzo di query e viste.

WebService (Servizi Web SOAP)

Motivazione della scelta
I *WebService* rappresentano il principale meccanismo di comunicazione tra il portale *Vision Ticketing* e il gestionale *Vision Enterprise*. Basati sul protocollo SOAP, consentono lo scambio strutturato di dati tramite *XML*$_G$ su protocollo *HTTP*$_G$. La scelta di utilizzare i *WebService* esistenti è risultata naturale, poiché questi costituiscono il canale ufficiale di integrazione previsto dal gestionale *Vision*.

Ruolo nello sviluppo
- Interfaccia di comunicazione tra il portale e il gestionale *Vision*;
- Invocazione di metodi esposti dal gestionale per la lettura e la registrazione dei documenti;
- Scambio di informazioni strutturate in formato XML.

Alternative considerate
In fase di progettazione sono state considerate API *REST*$_G$ come possibile alternativa, che avrebbero permesso un'integrazione più leggera e basata su *JSON*$_G$. Tuttavia, la presenza di un'infrastruttura SOAP già esistente e perfettamente compatibile con *Vision Enterprise* ha reso preferibile mantenere i *WebService* aziendali.

PHP

Motivazione della scelta

Nel progetto, PHP è stato impiegato per la gestione della logica lato server del portale web, in particolare nei controller, responsabili dell'elaborazione delle richieste provenienti dal frontend e della comunicazione con le API e il database. La scelta di questa tecnologia è stata dettata sia dalla presenza di codice preesistente sviluppato in PHP, sia dalla sua flessibilità e semplicità d'integrazione con i servizi già in uso.

Ruolo nello sviluppo

- Gestione della logica lato server del portale;
- Integrazione con le API e con il database aziendale.

HTML, CSS e JavaScript

Motivazione della scelta

Per lo sviluppo del frontend del portale è stato adottato un approccio basato sulle tecnologie standard del web: HTML per la struttura, CSS per lo stile e JavaScript per l'interattività. Questa scelta è stata dettata sia dalla volontà di mantenere parte del codice frontend già esistente, sviluppato in ASP.NET, sia dall'obiettivo di garantire un'interfaccia utente moderna e accessibile.

Ruolo nello sviluppo

- Definizione della struttura delle pagine web;
- Applicazione dello stile grafico e design dell'interfaccia;
- Implementazione dell'interattività lato client.

Dependency Injection

Motivazione della scelta

L'adozione del pattern di architettura *DI*$_G$ ha permesso di costruire il modulo *EmailTicketReader* in grado di leggere email, eseguire l'autenticazione e l'accesso al database in modo scalabile.
L'approccio Dependency Injection consente di separare i servizi principali di lettura, autenticazione e database, rendendo il sistema più manutenibile e testabile.

Ruolo nello sviluppo

- Separazione dei componenti;
- Possibilità di sostituire implementazioni concrete in fase di test, senza modificare il codice principale.

Alternative considerate
Nel modulo *EmailTicketReader* si è valutato l'utilizzo di approcci alternativi alla Dependency Injection per la gestione dei servizi applicativi, come l'istanziazione diretta delle dipendenze all'interno delle classi, una soluzione più semplice ma che avrebbe introdotto un forte accoppiamento tra i componenti e ridotto la possibilità di sostituire o testare i servizi in modo isolato.

Microsoft Graph API

Motivazione della scelta
La scelta della Microsoft Graph API è stata dettata dalla necessità di accedere in maniera sicura e moderna alle caselle di posta di Microsoft 365/Exchange Online. Graph API fornisce un endpoint unificato per interagire con servizi Microsoft cloud e consente di implementare flussi automatizzati per la lettura delle email, il filtraggio dei messaggi e l'autenticazione tramite token.

Ruolo nello sviluppo
- Lettura delle email inviate all'indirizzo di assistenza;
- Filtraggio dei messaggi in base a criteri;
- Gestione dell'autenticazione e dell'autorizzazione per l'accesso alle mailbox tramite token OAuth2.

Alternative considerate
In fase di analisi è stata valutata anche la possibilità di utilizzare la libreria Mail.dll, che consente di gestire in modo diretto l'autenticazione e l'invio di email tramite protocolli standard come *SMTP*$_G$ e *IMAP*$_G$, includendo anche il supporto a OAuth2. Tuttavia, questa soluzione avrebbe richiesto una gestione manuale delle credenziali e una configurazione più complessa della sicurezza.

OAuth2 – Client Credentials Flow

Motivazione della scelta
L'utilizzo del flusso di autenticazione OAuth 2.0 Client Credentials Flow è stato scelto per permettere al servizio backend di accedere autonomamente alle risorse Microsoft 365, in particolare la casella di posta e Graph API, senza intervento dell'utente. Questo protocollo garantisce un accesso sicuro e automatizzato, adatto per scenari server-to-server [1].

Ruolo nello sviluppo
- Registrazione dell'applicazione in Azure AD con permessi application;

- Ottenimento del token di accesso via client credentials;
- Utilizzo del token per effettuare chiamate verso Microsoft Graph API.

Capitolo 3

Analisi dei requisiti

Questo capitolo descrive in dettaglio i requisiti emersi durante la fase di analisi del progetto.

3.1 Classificazione dei requisiti

I requisiti individuati per il progetto *Vision Ticketing* sono stati suddivisi in funzionali e non funzionali, e ulteriormente classificati come obbligatori, desiderabili e opzionali.
La nomenclatura utilizzata segue la struttura:

[Tipo][Priorità]-[Numero progressivo]

Dove per Tipo si indica:

F : funzionale;
NF : non funzionale.

e per Priorità si indica:

O : obbligatorio
D : desiderabile
P : opzionale

Nelle tabelle sottostanti sono riassunti i requisiti e il loro tracciamento.

3.2 Requisiti funzionali

Obbligatori

Codice	Descrizione	Fonti
FO-1	Il sistema deve consentire l'accesso autenticato ai clienti e ai rivenditori tramite credenziali personali.	Piano di lavoro
FO-2	Il sistema deve permettere l'inserimento di nuove richieste di assistenza (ticket) da parte degli utenti autenticati.	Piano di lavoro
FO-3	Il sistema deve permettere il sollecito, la chiusura e la visualizzazione in dettaglio dei ticket già creati.	Piano di lavoro
FO-4	Ogni richiesta di assistenza deve essere associata a un cliente.	Piano di lavoro
FO-5	Il sistema deve registrare automaticamente le richieste di assistenza all'interno del gestionale *Vision Enterprise* come documenti di tipo CH.	Piano di lavoro
FO-6	Il sistema deve permettere l'inserimento degli allegati durante la creazione delle richieste di assistenza.	Piano di lavoro
FO-7	Il sistema deve permettere la visualizzazione degli allegati sia dal portale sia dal gestionale.	Piano di lavoro
FO-8	Il personale interno deve poter modificare, chiudere o riaprire i ticket direttamente dal gestionale.	Piano di lavoro
FO-9	Il sistema dovrebbe consentire ai tecnici di aggiungere commenti o allegati ai ticket in lavorazione.	Piano di lavoro

Codice	Descrizione	Fonti
FO-10	Il sistema dovrebbe consentire la visualizzazione di uno storico delle richieste di assistenza.	Piano di lavoro

Tabella 1: Tracciamento dei requisiti funzionali obbligatori.

Desiderabili

Codice	Descrizione	Fonti
FD-1	Il sistema dovrebbe permettere la ricerca e il filtraggio dei ticket per stato, priorità o area funzionale.	Piano di lavoro
FD-2	Il sistema dovrebbe permettere la riassegnazione dei ticket tra operatori diversi.	Riunione col tutor
FD-3	Il sistema deve essere in grado di leggere le email inviate all'indirizzo *assistenza@vsh.it* e convertirle in ticket nel gestionale	Piano di lavoro
FD-4	Il sistema dovrebbe salvare e visualizzare gli allegati delle email all'indirizzo assistenza@vsh.it.	Piano di lavoro

Tabella 2: Tracciamento dei requisiti funzionali desiderabili

Opzionali

Codice	Descrizione	Fonti
FP-1	Il sistema può integrarsi con l'applicazione *moviTICKET* per importare automaticamente richieste provenienti da dispositivi mobili.	Piano di lavoro

Codice	Descrizione	Fonti
FP-2	Il sistema può consentire la creazione automatica di ticket a partire dalle chiamate vocali ricevute tramite impianto VoIP.	Piano di lavoro
FP-3	Il sistema può consentire la trascrizione automatica delle chiamate vocali in testo da allegare al ticket.	Piano di lavoro

Tabella 3: Tracciamento dei requisiti funzionali opzionali.

3.3 Requisiti non funzionali

Obbligatori

Codice	Descrizione	Fonti
NFO-1	Il sistema deve assicurare la compatibilità con il gestionale *Vision Enterprise* e il database SQL Server.	Piano di lavoro
NFO-2	Il portale web deve essere accessibile dai principali browser web (Chrome, Firefox, Edge, Safari).	Piano di lavoro
NFO-3	Il portale deve garantire tempi di risposta accettabili.	Piano di lavoro
NFO-4	Il portale deve fornire messaggi di errore chiari in caso di richieste non realizzabili.	Piano di lavoro

Tabella 4: Tracciamento dei requisiti non funzionali obbligatori.

Desiderabili

Codice	Descrizione	Fonti
NFD-1	L'interfaccia del portale dovrebbe essere coerente con il layout e l'identità visiva del sito aziendale VisioneImpresa.	Piano di lavoro
NFD-2	Il portale dovrebbe offrire un'esperienza utente intuitiva e coerente, minimizzando gli errori di input.	Piano di lavoro
NFD-3	Il portale dovrebbe essere responsivo e adattarsi automaticamente a diverse dimensioni di schermo.	Piano di lavoro

Tabella 5: Tracciamento dei requisiti non funzionali desiderabili.

Di seguito, nella Tabella 6 viene mostrato il riepilogo dei requisiti, suddivisi per tipologia e necessità.

Tipo	Obbligatori	Desirabili	Opzionali	Somma
Funzionali	10	4	3	17
Non Funzionali	4	3	0	7
Totale	**14**	**7**	**3**	**24**

Tabella 6: Riepilogo dei requisiti.

Capitolo 4

Il gestionale Vision

Dopo aver introdotto il contesto aziendale e le finalità generali del progetto, in questo capitolo viene presentato il gestionale Vision Enterprise, con particolare attenzione al sistema di gestione delle richieste di assistenza, all'introduzione delle nuove API e alla struttura del database su cui si basa il progetto Vision Ticketing.

4.1 Panoramica e gestione delle richieste

Il gestionale *Vision* è un *ERP*$_G$, in grado di gestire i principali processi aziendali come contabilità, vendita, gestione magazzini e delle risorse. Oltre a gestire questi processi, *Vision* integra il ciclo di assistenza clienti, collegando ogni chiamata allo storico del cliente e ai suoi ordini.
Il flusso tradizionale delle segnalazioni di assistenza si basa sulla creazione di documenti CH all'interno del gestionale.
Quando un cliente contatta l'azienda, telefonicamente o via email, per segnalare un problema l'operatore addetto alla reception registra la richiesta aprendo un nuovo documento CH.
Durante la registrazione vengono inseriti i principali campi identificativi, ovvero, data e ora della segnalazione, cliente di riferimento, descrizione del problema ed eventuali allegati. Una volta creata nel gestionale, la segnalazione viene contrassegnata come 'non verificata', indicando che la richiesta è stata registrata ma non ancora analizzata.
A seguito, l'help desk verifica la segnalazione, ne valuta la priorità e assegna la richiesta al tecnico del reparto competente, che si occuperà di aggiornare il documento CH con aggiornamenti sulle attività svolte per risolvere la segnalazione.
Quando il problema risulta risolto, l'operatore di assistenza inserisce nel documento una nota risolutiva, data e ora del termine della segnalazione e il documento viene marcato come 'chiuso'.

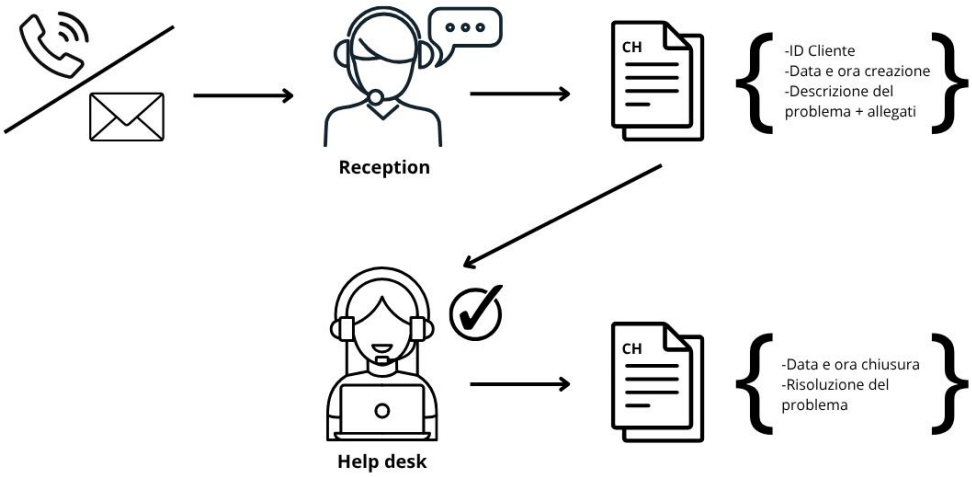


Figura 2: Flusso della gestione delle richieste di assistenza.

4.2 Introduzione delle API

Per superare le criticità del processo standard di gestione delle segnalazioni, è stato introdotto un sistema di API SOAP, sviluppato all'interno del modulo *WebService*, un servizio applicativo, sviluppato in .NET, che fa da ponte tra il gestionale *Vision* e i vari sistemi esterni aziendali.
La scelta di API SOAP, con scambio dei dati in formato XML, permette di mantenere la massima compatibilità con l'infrastruttura .NET e un elevato grado di sicurezza.
Con il nuovo sistema di ticketing, *WebService* è stato oggetto di un'estensione funzionale, per introdurre le procedure di comunicazione tra il gestionale e le applicazioni esterne, tra cui il nuovo portale web e il modulo di gestione email, ossia *EmailTicketReader*.

Le principali funzionalità introdotte riguardano:

- **Creazione automatica dei documenti CH**: aggiunta di metodi dedicati alla generazione dei documenti di chiamata direttamente da richieste provenienti dal portale web, con registrazione automatica di cliente, descrizione, data e allegati;

- **Modifiche dello stato dei ticket**: implementazione di procedure per la modifica dello stato della segnalazione e l'inserimento dei solleciti;

- **Recupero dello storico e delle informazioni dei ticket**: aggiunta di metodi per il recupero dei dati dei ticket in dettaglio e della lista completa dei ticket in uno storico ordinato;

- **Chiusura dei ticket**: possibilità di chiudere la segnalazione dal portale web per i clienti in modo autonomo;

- **Gestione degli allegati**: possibilità di ricevere, archiviare e associare file ai documenti CH, anche se provenienti da fonti diverse;

- **Integrazione con il modulo EmailTicketReader**: introduzione di un'interfaccia dedicata alla ricezione dei dati elaborati dal servizio di lettura email, per la creazione automatica di chiamate di assistenza basate sul contenuto delle email in arrivo;

- **Verifica anagrafica clienti**: integrazione con le tabelle del database aziendale per l'identificazione automatica del cliente mittente sulla base dell'indirizzo email o del codice cliente.

4.3 Database

Il gestionale *Vision* utilizza un database relazionale, implementato su Microsoft SQL Server e durante la fase di integrazione delle nuove API, si è operato su una sua istanza locale, al fine di lavorare in un ambiente isolato e garantire l'integrità dei dati.

I principali schemi di tabelle coinvolte durante lo stage sono:

- **VISIONVI**: contenente i dati del sistema relativi a clienti, fornitori e prodotti;

- **VISIONCOMMON**: contenente le informazioni condivise e di riferimento comuni per la strutturazione degli indirizzi come città, province e comuni;

- **VISIONSYS**: contenente i parametri di personalizzazione del sistema.

In particolare, le attività si sono concentrate principalmente sullo schema VISIONVI, che costituisce il nucleo operativo del gestionale.

Tra le tabelle più significative si trovano:

- **dbo.CliFor**: gestisce le anagrafiche di clienti e fornitori, includendo dati identificativi, contatti e informazioni commerciali;

- **dbo.DocTes**: raccoglie i documenti gestionali tra cui le richieste di assistenza CH e ne memorizza i relativi metadati.

Inoltre, per il corretto funzionamento delle nuove API, si è reso necessario modificare e aggiungere nuovi dati e nuove tabelle, tra cui:

- **dbo.WebReparto**: introdotto per distinguere il reparto di assistenza di *Vision* da quello di *moviDAT*, un'applicazione per tablet del gestionale;

Nome colonna	Tipo di dati
Id_WebReparto	int
DesWebReparto	varchar(40)
TimeIns	datetime
TimeMod	datetime
UserIns	varchar(40)
UserMod	varchar(40)
Email	varchar(50)

Figura 3: Tabella WebReparto.

- **dbo.Comm**: aggiunto il campo *Ecommerce* di tipo bit per controllare la visibilità delle commesse o aree funzionali nel portale web;

CONSCOST	decimal(18, 6)	☐
CONSRICA	decimal(18, 6)	☐
LavPrimus	varchar(200)	☐
SDDManId	varchar(35)	☐
SDDManDate	smalldatetime	☐
CodExport	varchar(20)	☐
Ecommerce	bit	☐
		☐

Figura 4: Tabella Comm.

- **dbo.DocTes2**: aggiunto il riferimento alla tabella WebReparto tramite chiave esterna;

ID_DocTes2	int
ID_DocTes	int
ChkAumento	bit
NoScadSim	bit
DAumento	smalldatetime
TimeIns	smalldatetime
TimeMod	smalldatetime
UserIns	varchar(40)
UserMod	varchar(40)
Id_WebReparto	int

Figura 5: Tabella DocTes2.

- **dbo.ImpostaB2B**: aggiunte le configurazioni per il sistema di ticketing per facilitare la compilazione automatica dei campi di ticketing.

19	19	UsaAllegati	1
20	20	NewCli	1
21	21	CodCliForPrefix	CB
22	22	AllDoc4Agente	1
23	23	authDocs	1
24	24	blockDocs	1
25	25	GIAC	1
26	26	NoteArticolo	0
27	27	TipoTrasporto	1
28	28	CodDocCH	CH
29	29	CodApertCH	NONVE
30	30	CodChiusCH	CHIUS

Figura 6: Tabella ImpostaB2B.

Capitolo 5

Il portale web

Dopo aver introdotto le nuove API, il seguente capitolo descrive il portale web di Vision Ticketing, illustrandone l'architettura, le funzionalità principali insieme alle principali fasi di sviluppo che hanno permesso la realizzazione della versione aggiornata del sistema.

5.1 Architettura

Il portale si basa su un'architettura web containerizzata, gestita tramite DDEV e Docker, il cui frontend è stato adattato dal portale dell'azienda consociata OfficeGroup, con personalizzazioni grafiche e strutturali per VisioneImpresa.
Il backend, invece, si interfaccia con le nuove API SOAP del *WebService*, responsabili della comunicazione con il gestionale *Vision* e del popolamento del database. L'integrazione con quest'ultimo consente di sincronizzare in tempo reale i dati tra il portale web e il gestionale.

5.2 Funzionalità del portale

5.2.1 Accesso e autenticazione

La schermata di login presenta due campi principali, username e password, e un messaggio di errore compare in caso di credenziali errate. L'accesso avviene tramite credenziali personali fornite da VisioneImpresa, e dopo l'autenticazione, l'utente accede direttamente all'area operativa.

Figura 7: Schermata di login del portale.

5.2.2 Area anagrafica

All'accesso, l'utente visualizza la sezione *Area anagrafica*, che riporta le informazioni del cliente o dell'azienda registrate nel gestionale:

- Dati anagrafici e locazione;
- Recapiti telefonici ed email;
- Banche di appoggio;
- Condizioni di pagamento.

Queste informazioni sono di sola consultazione e garantiscono l'allineamento dei dati tra cliente e gestionale aziendale.

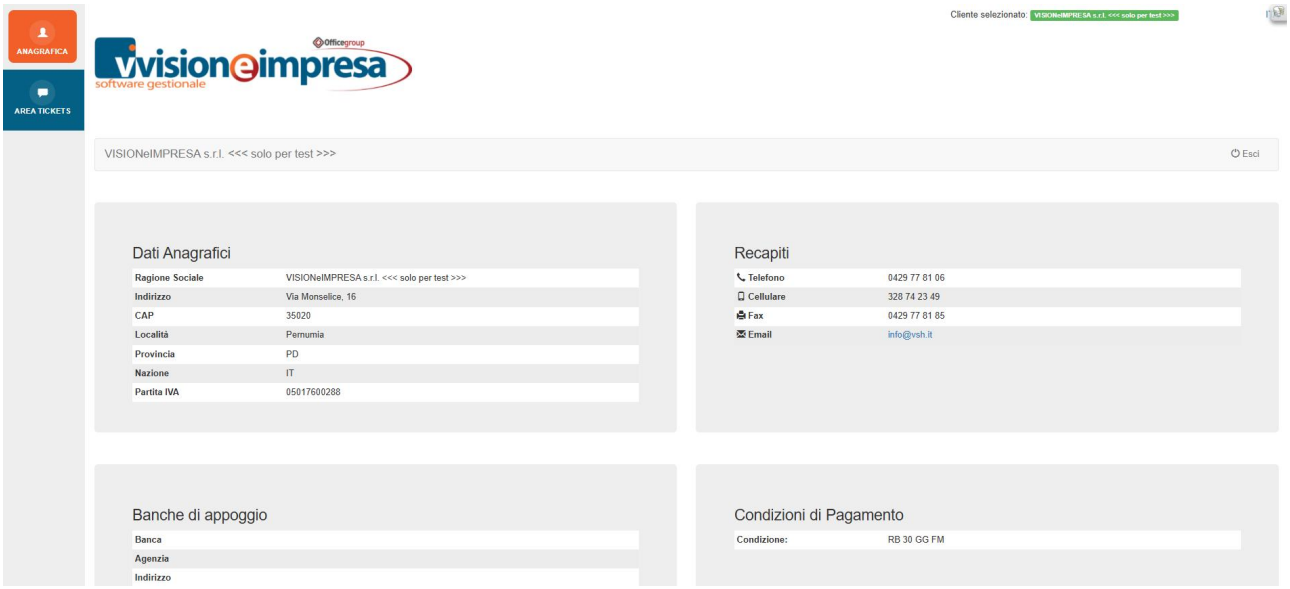


Figura 8: Area anagrafica del portale.

5.2.3 Area ticket

La sezione principale del portale, *Area Ticket*, è dedicata alla gestione dei ticket di assistenza e tramite due pulsanti principali, *Aggiungi ticket* e *Lista ticket*, l'utente può creare nuove segnalazioni o consultare quelle esistenti. Inoltre, sotto i due pulsanti è presente una barra di ricerca per filtrare i ticket creati.

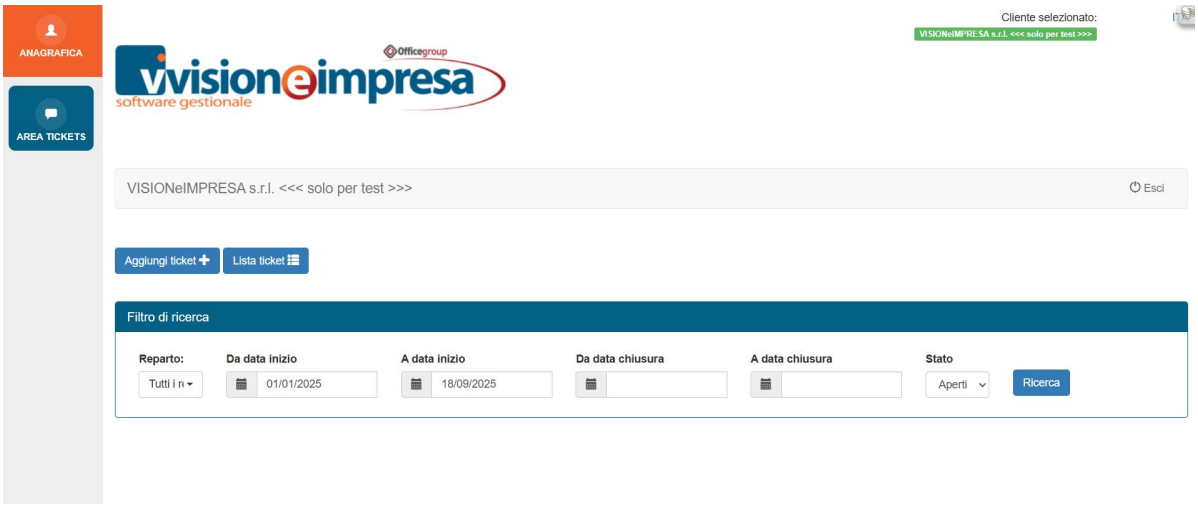


Figura 9: Area ticket del portale.

Creazione ticket

Il form di inserimento per la creazione dei ticket prevede i seguenti campi:

- Referente;
- Area Programma;
- Reparto;
- Titolo;
- Descrizione del problema;
- Allegati.

Il salvataggio del ticket genera un documento di tipo CH nel gestionale *Vision*.

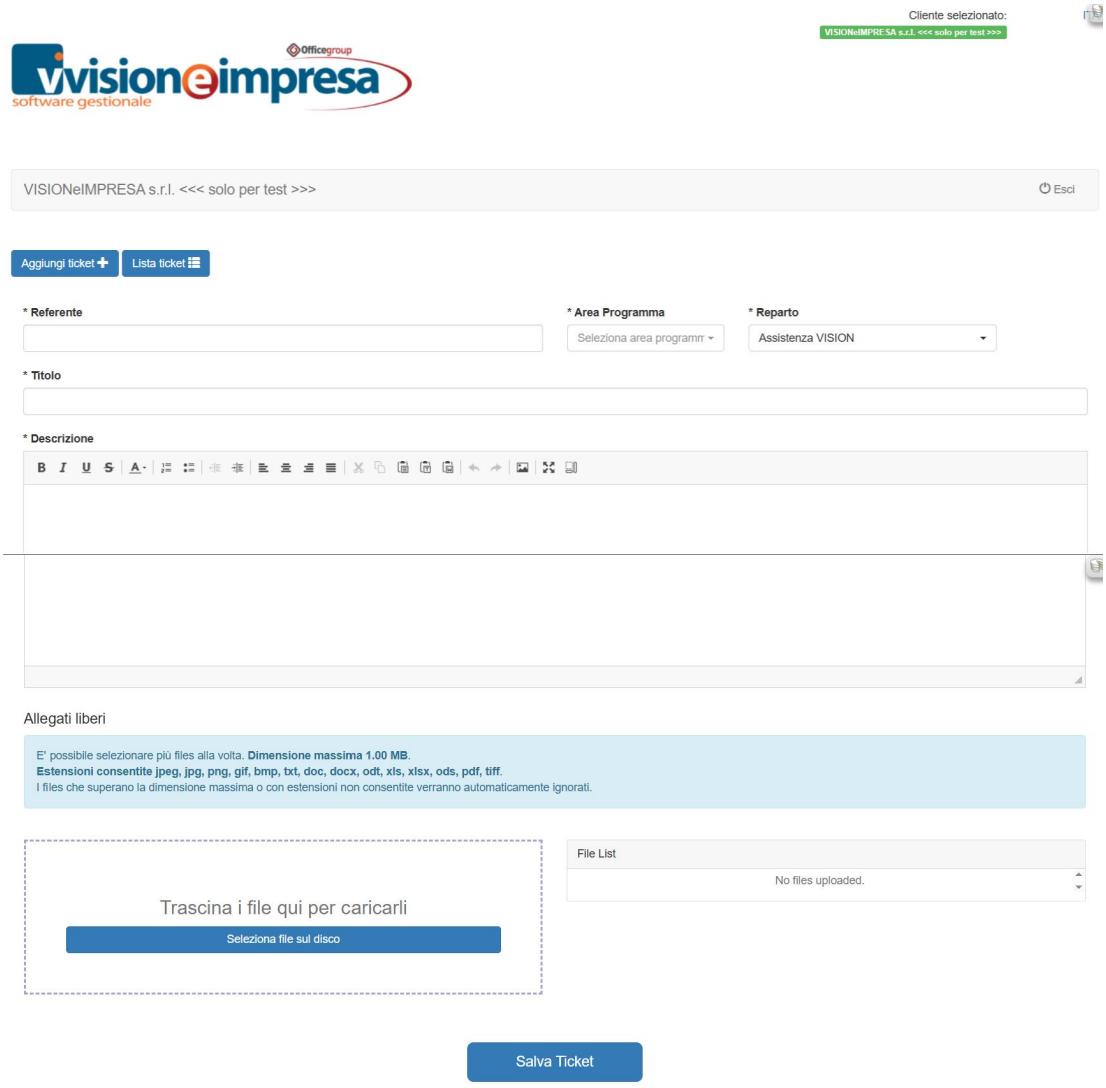


Figura 10: Area creazione dei ticket.

Lista ticket

La lista permette di consultare i ticket in dettaglio grazie ad una sezione di ricerca che permette di filtrare i ticket per reparto, intervallo di date e stato (aperto o chiuso).
Ogni record mostra le informazioni principali e consente di:

- Aprire il dettaglio completo del ticket;
- Scaricare il PDF del ticket.

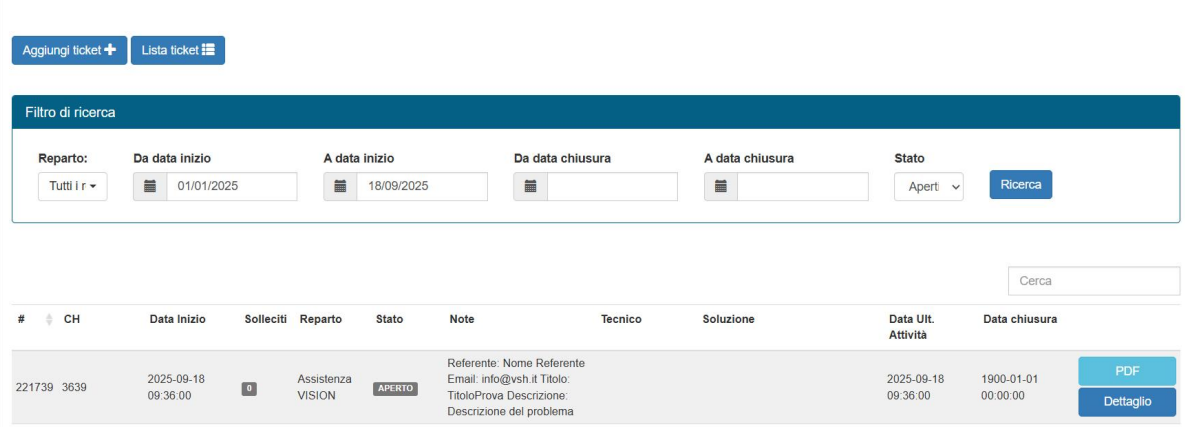


Figura 11: Visualizzazione lista dei ticket.

Dettaglio ticket

Cliccando il pulsante *Dettaglio*, visibile nella lista ticket, vengono visualizzate tutte le informazioni relative alla segnalazione, comprese le note e gli allegati caricati. In questa sezione il cliente può sollecitare l'assistenza inserendo una nota aggiuntiva, oppure chiudere la richiesta, motivando la risoluzione, tramite i rispettivi pulsanti *Sollecita* e *Chiudi come risolto*. Queste operazioni sono possibili se il ticket risulta aperto, mentre sono disattivate e non visibili se il ticket risulta già chiuso, per preservare l'integrità del flusso.

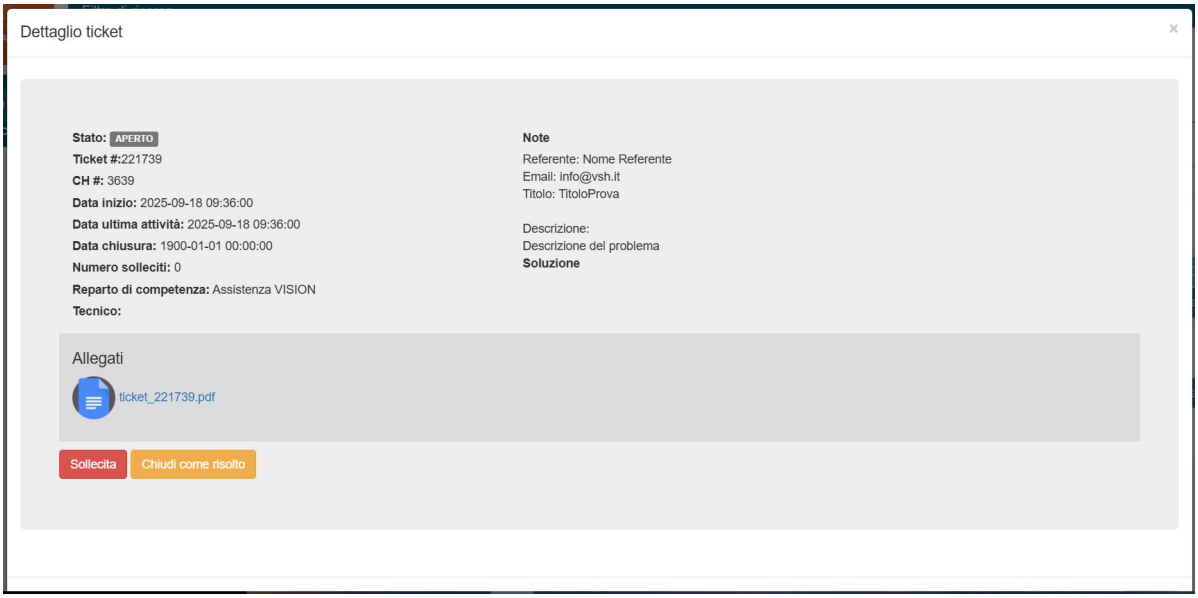


Figura 12: Schermata dettaglio del ticket.

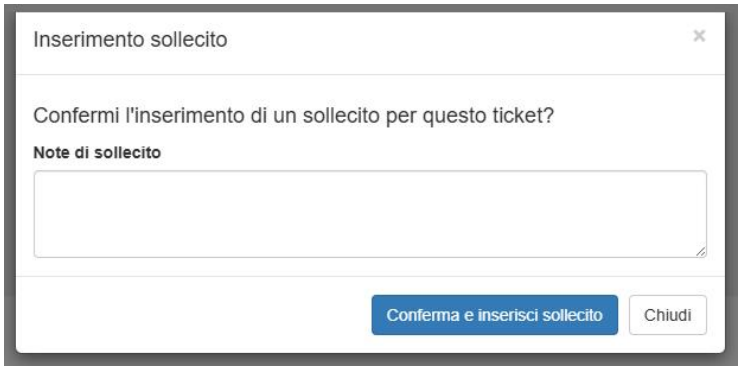


Figura 13: Banner inserimento del sollecito.

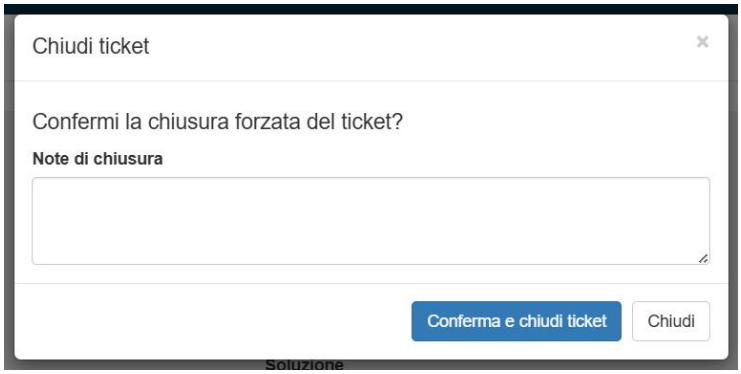


Figura 14: Banner chiusura del ticket.

5.3 Integrazione con il WebService

5.3.1 Flusso di esecuzione delle API

Avviando il *WebService* dal server IIS Express, Visual Studio apre automaticamente l'endpoint *http://localhost:5009/WebService.asmx*, che rappresenta l'interfaccia del servizio.
In questa pagina vengono elencate tutte le funzioni esposte dalla classe *WebService*, ciascuna delle quali è marcata con l'attributo `[WebMethod]`.
Il portale web invia una richiesta SOAP all'endpoint del *WebService*, specificando il metodo da invocare e i relativi parametri. Il metodo corrispondente deserializza i parametri ricevuti e li passa alla classe *WebServiceNf_Ticketing*, che si occupa di eseguire le operazioni sul database *Vision*.
Al termine dell'elaborazione, i risultati vengono restituiti al client sotto forma di messaggio SOAP/XML, pronto per essere interpretato e visualizzato dal portale.

5.3.2 Processo di integrazione

In un primo momento, il servizio web veniva eseguito in locale tramite IIS Express, ma in questa configurazione il portale non era in grado di comunicare con esso, poiché *WebService* e il cordice sorgente del portale risiedevano in ambienti distinti.
Per consentire il corretto scambio di dati e simulare l'ambiente aziendale, il progetto è stato configurato su DDEV, piattaforma che ha consentito di replicare la struttura del server di produzione.
In questa configurazione, il portale viene eseguito in un container separato, collegato al servizio tramite un endpoint di rete interna: *http://visioneimpresa.local/WebService.asmx*.

5.3.3 La classe WebServiceNf_Ticketing

La classe *WebServiceNf_Ticketing* comprende la logica del sistema di integrazione, essa non espone direttamente i metodi web, ma viene utilizzata internamente da *WebService* per gestire i dati e le query sul gestionale *Vision*.

Le principali responsabilità della classe sono:

1. **Gestione dei parametri e delle strutture dati**

Contiene le classi necessarie alla gestione degli input e output:

- `TicketParameters`: creazione o modifica di un ticket;
- `GetTicketsParameters`: filtraggio e ricerca dei ticket;
- `AttivitaCollegateParameters`: recupero delle attività associate;
- `Allegato`: rappresentazione dei file in Base64 allegati ai ticket.

2. **Logica di accesso ai dati**

Interagisce con il database *Vision*, eseguendo query e procedure per restituire oggetti come:

- `s_ticket`: lista dei ticket;
- `s_oit_attivita_collegate_row`: attività collegate;
- `s_oit_segnalazione`: dettaglio delle segnalazioni.

3. **Gestione dello stato e degli errori**

Contiene variabili come `err_code`, `err_message` e `message`, utilizzate per gestire esiti e messaggi di ritorno verso il client.

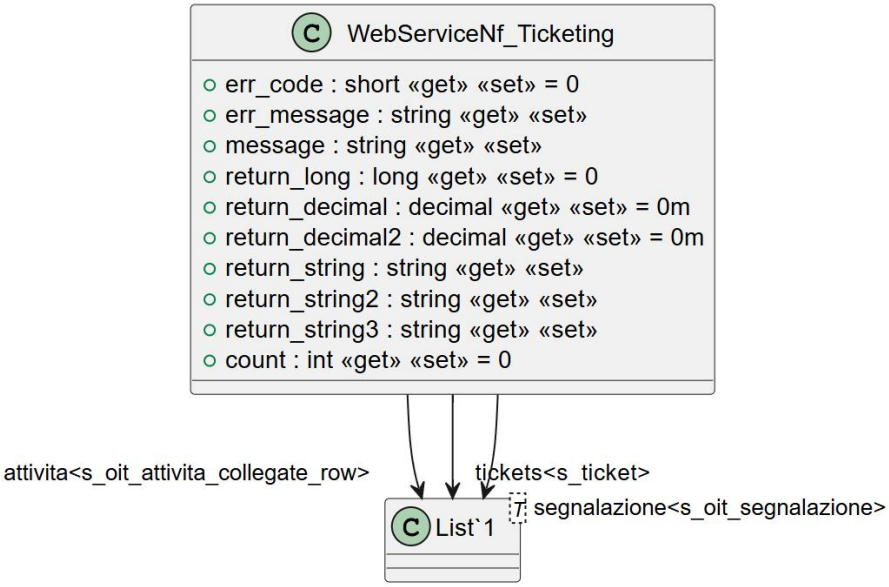


Figura 15: UML di WebServiceNf_Ticketing.cs.

5.3.4 La classe WebService

La classe *WebService* implementa i metodi che rappresentano le nuove API SOAP sviluppate per gestire le funzionalità del nuovo portale.
Ogni metodo del servizio riceve parametri strutturati, invocando *WebServiceNf_Ticketing* e restituendo il risultato al client.

Tra le nuove funzioni implementate abbiamo:

- `nf_oit_ins_ticket()` : inserisce un nuovo ticket nel sistema;
- `nf_oit_sollecito()` : registra un sollecito su un ticket aperto;
- `nf_oit_get_sett_personale()` : restituisce l'elenco dei reparti disponibili;
- `nf_oit_get_area_programma()` : restituisce le aree funzionali come contabilità, magazzino, ecc..;
- `nf_oit_get_attivita_collegate()` : recupera le attività collegate ad un ticket;
- `nf_oit_chiudi_segnalazione()` : chiude un ticket esistente;
- `nf_oit_get_tickets()` : restituisce la lista dei ticket filtrati in base a cliente, stato o data.

Nel diagramma UML riportato di seguito sono rappresentate soltanto le nuove funzionalità aggiunte, a causa delle dimensioni eccessive del file completo.

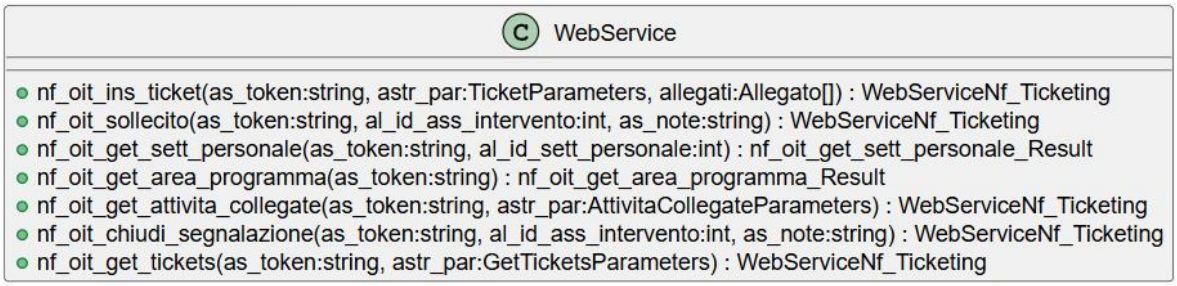


Figura 16: UML di WebService.cs.

5.4 Sviluppo e adattamento del portale

5.4.1 Adattamento del codice esistente

Il portale *Vision Ticketing* deriva dal portale originariamente sviluppato da OfficeGroup e già utilizzato per la gestione centralizzata delle loro richieste di assistenza.
L'obiettivo principale dello sviluppo è stato quello di adattare e personalizzare il codice preesistente per il contesto di VisioneImpresa, integrandolo con il nuovo *WebService* e con il gestionale *Vision*.

Il codice sorgente, basato su Model–View–Controller, è composto da file:

- `.php`: per la logica di business e i controller,
- `.ctp`: per la definizione delle viste e dei template HTML,
- `.css` e `.js`: per la parte grafica e l'interattività lato client.

Questa struttura ha facilitato l'integrazione con il nuovo *WebService*, permettendo di intervenire sulle componenti interessate senza alterare l'impianto generale del portale.

Le modifiche principali hanno riguardato la configurazione delle impostazioni di connessione, sostituendo gli endpoint originali di OfficeGroup con quelli specifici di VisioneImpresa, e la rimozione dei parametri di OfficeGroup che non avevano corrispondenza con il database di *Vision*.
Nel file di configurazione, `global.php`, è stata introdotta una nuova voce, denominata *ves*, che definisce le informazioni necessarie per la connessione al *WebService*:

```php
$config['Settings']['ws'] = [

    'ves' => [
        'endpoint' => 'http://visioneimpresa.local/
        WebService.asmx?WSDL',
        'token' => '**********',
        'folder' => 'ves',
    ],
];
```

Codice 2: Configurazione global.php.

5.4.2 Design e ottimizzazioni

Parallelamente all'integrazione con il nuovo *WebService*, è stato condotto un intervento di revisione grafica e funzionale del portale, con l'obiettivo di renderlo coerente con l'identità visiva e i requisiti operativi di VisioneImpresa. Le modifiche hanno riguardato sia aspetti estetici, tra cui colori, logo e layout, sia la riorganizzazione delle aree funzionali in un'ottica di usabilità e semplificazione dei flussi operativi.

Adeguamento grafico e identità visiva

L'interfaccia è stata uniformata al sito istituzionale di VisioneImpresa sostituendo logo, favicon, icona di caricamento e palette cromatica, rispetto al sito originale di OfficeGroup.
In particolare, i colori dei pulsanti e degli elementi interattivi sono stati adeguati alla brand identity aziendale, garantendo coerenza visiva e riconoscibilità immediata del marchio.
Anche se questo è un'intervento marginale ha un impatto diretto sull'esperienza dell'utente, che percepisce il portale come parte integrante dell'ecosistema digitale aziendale.

Figura 17: Confronto tra palette di OfficeGroup e VisioneImpresa.

Ottimizzazione della navigazione e delle aree funzionali

Dal punto di vista strutturale, la barra di navigazione laterale è stata semplificata, eliminando la voce *Area Interventi*, non prevista tra i requisiti funzionali del nuovo sistema, rendendo più immediato l'accesso alle sezioni realmente operative.
Nell'area anagrafica, nel file `view.ctp`, è stata effettuata una riprogettazione parziale della disposizione dei blocchi: il campo *Condizioni di pagamento* è stato spostato sotto la sezione *Banche di appoggio*, mentre al suo posto è stato inserito il nuovo blocco *Recapiti*.
La modifica è stata introdotta per allineare la sequenza dei campi a un flusso di lettura più logico, centrato sull'utente finale piuttosto che sulla struttura amministrativa interna.

ctp

```
<div id="anagrafica">
  <div class="row">
    <div class="col-md-6">
      <div class="jumbotron">
        <h3><?php echo __('Dati Anagrafici',true); ?></h3>
        <div id="anag"></div>
      </div>
    </div>
    <div class="col-md-6">
      <div class="jumbotron">
        <h3><?php echo __('Recapiti',true); ?></h3>
        <div id="tel"></div>
      </div>
    </div>
  </div>
```

```ctp
    <div class="row">
      <div class="col-md-6">
        <div class="jumbotron">
          <h3><?php echo __('Banche di appoggio',true); ?></h3>
          <div id="banche"></div>
        </div>
      </div>
      <div class="col-md-6">
        <div class="jumbotron">
          <h3><?php echo __('Condizioni di Pagamento',true); ?></h3>
          <div id="pag"></div>
        </div>
      </div>
    </div>
</div>
```

Codice 3: Dettaglio codice view.ctp

Semplificazione delle sezioni operative

Le aree dedicate alla creazione e gestione dei ticket sono state oggetto delle modifiche più significative.
Durante la creazione di un ticket, sono stati rimossi i campi relativi alle sedi aziendali e all'area azienda, poiché non utili alla risoluzione del ticket e ridondanti rispetto ai dati gestiti nel gestionale.
Inoltre, sempre nell'area di creazione dei ticket, è stato ridefinito l'ordine dei campi di compilazione secondo una sequenza più naturale per l'utente:

Referente → Area Programma → Reparto → Titolo → Descrizione → Allegati.

In questo flusso, il *Referente* identifica la persona che ha riscontrato il problema, l'*Area Programma* indica il contesto o la sezione applicativa in cui l'anomalia si è verificata, mentre il *Reparto* rappresenta il settore di assistenza competente a gestire la segnalazione. Seguono il *Titolo*, che sintetizza brevemente la natura del ticket, e la *Descrizione*, in cui l'utente può dettagliare il problema in modo completo e inserire eventuali screenshot o file nella sottostante area *Allegati*.
Questa riorganizzazione migliora la fluidità dell'inserimento delle infor-

mazioni del ticket e riduce il rischio di errori di comprensione durante la compilazione.

È stato rimossa la possibilità per l'utente di inserire la priorità, che viene definita di default, ed è stato rimosso il limite minimo di 100 caratteri nel campo *Descrizione*, che obbligava l'inserimento di un testo di tale lunghezza anche quando non necessario. Tuttavia, nello script `add.js` viene segnalato un errore quando il campo Descrizione è lasciato vuoto; ciò vale anche per gli altri campi del modulo.

```
1                                                    JavaScript
    if
2   (CKEDITOR.instances.as_des_breve.getData().length
    < window.app.ticketMinLength) {
3       validationErrors.push('La descrizione non
        può essere vuota. ');
4       }
```

Codice 4: Dettaglio codice add.js

Nella sezione lista dei ticket, è stato rimosso il filtro di ricerca per *Area azienda*, non disponibile per VisioneImpresa, e resi pienamente funzionanti i filtri per *Stato* e *Reparto*, che nella versione di OfficeGroup non restituivano risultati corretti a causa della gestione dei documenti CH lato backend.
Nella visualizzazione dello storico dei ticket, la colonna delle *Note* è stata semplificata rimuovendo la funzione *"Leggi di più"*, che risultava superflua rispetto alla sezione di dettaglio del ticket, dove le informazioni complete sono già visibili. L'eliminazione di questo elemento ha reso la tabella formattata e leggibile, evitando disarmonie nella visualizzazione dell'elenco.

Inoltre sono stati aggiunti due nuovi campi: il codice `num_doc`, denominato *CH* e identificativo del ticket, e il codice `cod_dipen`, ovvero il nome del tecnico a cui viene assegnato il ticket per migliorare la tracciabilità e il dialogo tra cliente e assistenza.
Questa nuova aggiunta ha richiesto le modifiche di diversi file del portale, tra cui `TicketsController.php`, e l'aggiornamento dei parametri nel *WebService*.

```php
public function nf_oit_get_tickets()
{
    $listalabel = ['data_inizio',
    'num_solleciti', 'priorita',
    'codice_sett_personale',
    'descrizione_sett_personale', 'stato',
    'note', 'soluzione_rilascio',
    'data_ultima_attivita', 'priorita_0',
    'priorita_1', 'priorita_2', 'priorita_3',
    'priorita_4', 'nodata', 'pdf', 'sollecita',
    'chiudi', 'destinazione', 'data_chiusura',
    'dettaglio', 'area_azienda',
    'num_doc','cod_dipen'];
```

Codice 5: Dettaglio codice TicketsController.php

```csharp
public class GetTicketsParameters
{
   ...
    public string num_doc { get; set; } = "";
    public string cod_dipen { get; set; } = "";

}
```

Codice 6: Dettaglio codice WebServiceNf_Ticketing.cs

Anche la sezione dettaglio ticket è stata razionalizzata, in quanto sono state eliminate funzioni ridondanti come i pulsanti *Sollecito* e *Chiudi come risolto* quando il ticket risulta già chiuso.
Queste modifiche riducono la possibilità di azioni incoerenti e semplificano l'esperienza d'uso in fase di consultazione.

5.5 Evoluzione del portale

Prima dell'introduzione del nuovo sistema, VisioneImpresa disponeva già di un portale web ma quest'ultimo era riservato solo ai rivenditori e limitato alla segnalazione di bug o richieste di migliorie, senza integrazione diretta con il gestionale aziendale.
Con la nuova versione del portale *Vision Ticketing*, invece, il cliente finale può autonomamente creare, monitorare e chiudere le proprie richieste di

assistenza, che vengono registrate automaticamente nel gestionale grazie alle nuove API SOAP.
I tempi di risposta delle richieste vengono ridotti grazie alla tracciabilità in tempo reale e alla rimozione dello stadio iniziale di creazione del documento CH, mentre, l'interfaccia semplice e intuitiva permette una facile comprensione delle possibili operazioni del portale, con colori coerenti con il sito ufficiale di VisioneImpresa.
Poiché la versione precedente del portale era in fase di rinnovo, è stato possibile visionare soltanto l'interfaccia di accesso originale e una versione di test parziale dell'ambiente interno, utilizzata per scopi dimostrativi, che non includeva tutte le funzionalità originarie come l'area dedicata ai ticket.

Dalle immagini sottostanti è possibile confrontare il portale web prima e dopo il progetto di stage.

5.5.1 Pagina di login

La versione originaria della pagina di login, datata ai primi anni 2000, si caratterizza per un aspetto essenziale e lineare, ma privo di coerenza con l'identità aziendale del sito ufficiale, anche a causa dell'utilizzo di un logo differente. La nuova versione del portale, al contrario, adotta uno stile più attuale, ispirato ai modelli contemporanei di interfaccia utente, in cui il focus visivo è posto sul modulo di autenticazione.

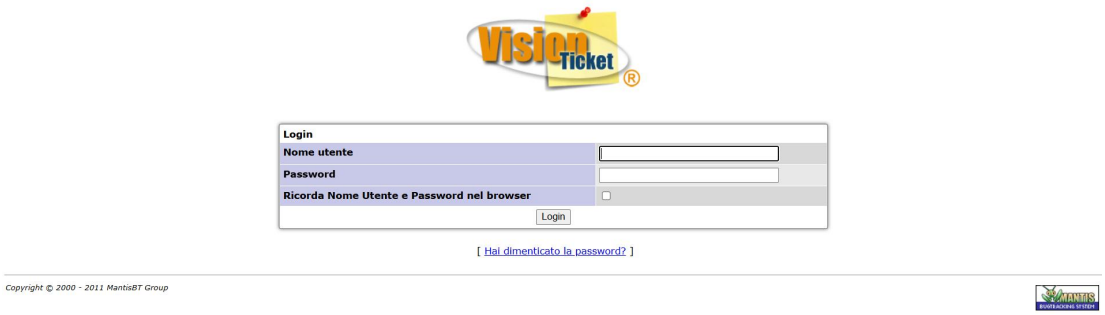


Figura 18: Pagina di login del vecchio portale.

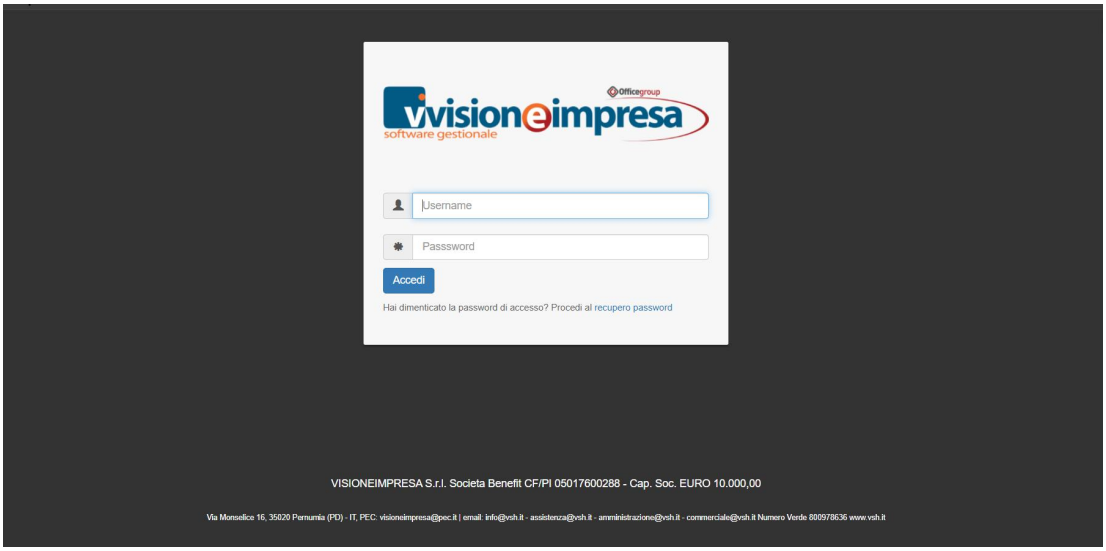


Figura 19: Pagina di login del nuovo portale.

5.5.2 Pagina dell'area anagrafica

Nella versione precedente, l'area anagrafica era articolata in più sottosezioni cliccabili, paragonabili a una rubrica, in cui i dati risultavano accessibili solo previa apertura della sezione corrispondente.
La disposizione dei contenuti, concentrata nella parte sinistra dello schermo, lasciava invece ampi spazi vuoti sul lato destro, generando una percezione di squilibrio visivo e di inefficienza nell'utilizzo dello spazio.
La nuova versione del portale unifica, invece, le informazioni anagrafiche in un'unica pagina, organizzandole per categorie e rendendole immediatamente consultabili, con un miglioramento significativo in termini di chiarezza e usabilità.

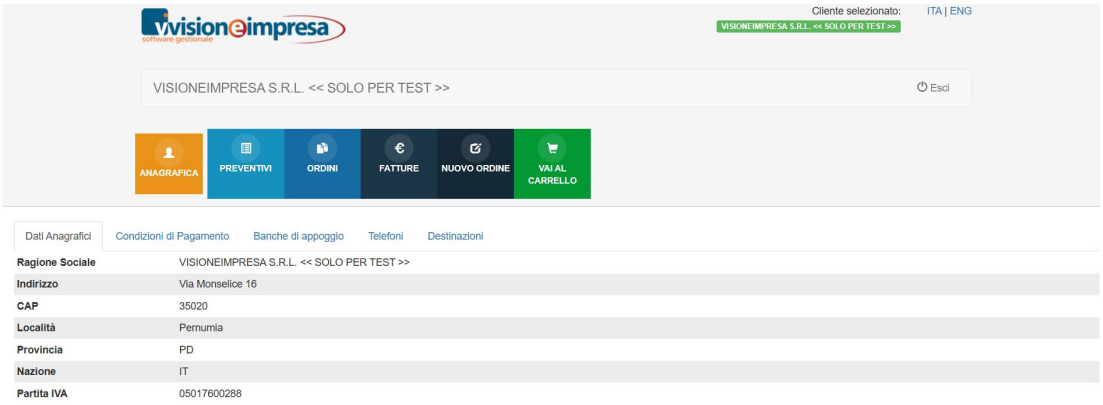


Figura 20: Area anagrafica del portale di test.

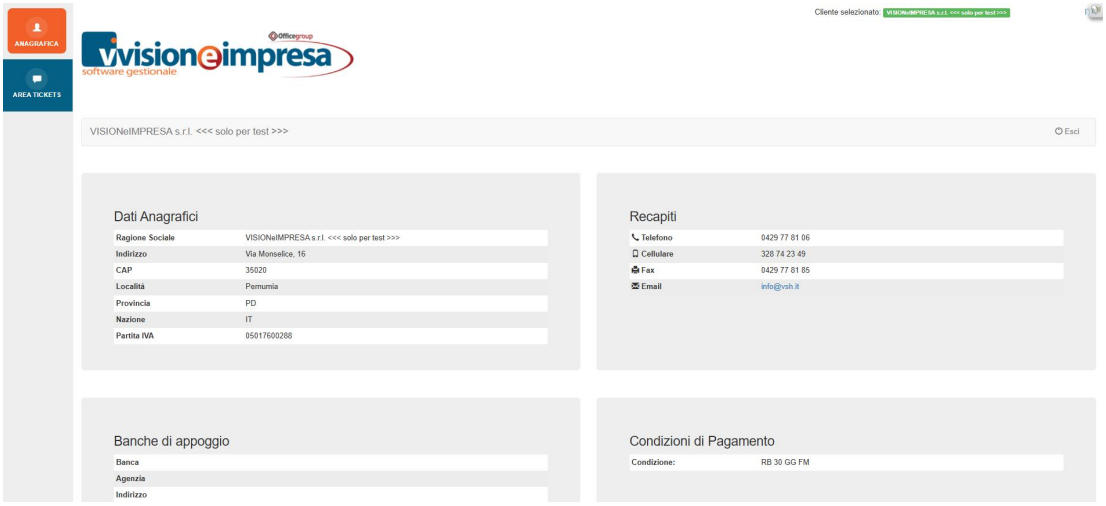


Figura 21: Area anagrafica del nuovo portale.

5.5.3 Pagina di ricerca e visualizzazione lista

L'area dedicata alla visualizzazione dello storico dei ticket non è disponibile nella versione di test, poiché non ancora implementata. È tuttavia possibile osservare che il filtro di ricerca presenta un'impostazione molto simile a quella adottata nel nuovo portale. Ciò è dovuto al fatto che la versione di test è stata sviluppata in un periodo più recente rispetto al portale originale.
Tuttavia, nella versione di test il portale non risulta coerente con l'identità aziendale. Tale incoerenza emerge chiaramente nella barra di navigazione superiore, costituita da blocchi colorati non allineati alla palette cromatica del sito ufficiale di Visioneimpresa. Questa scelta unita

all'utilizzo di colori troppo simili in elementi adiacenti può confondere l'utente nella navigazione e compromettere l'esperienza d'uso complessiva.

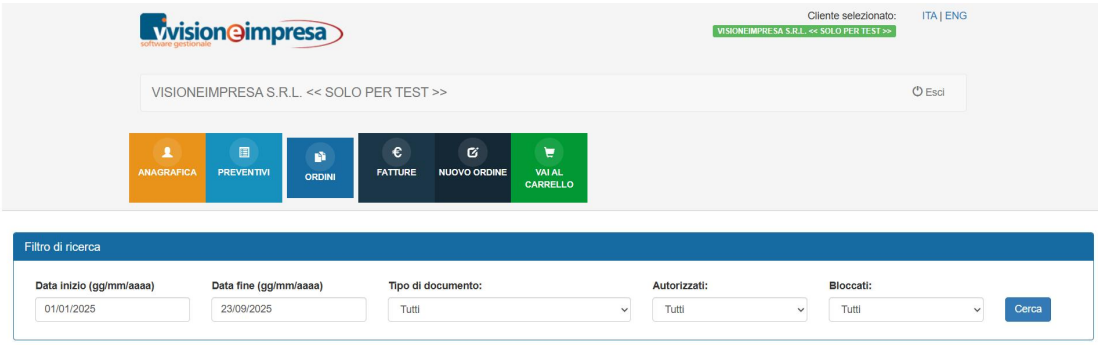


Figura 22: Sezione lista del portale di test.

Nel nuovo portale, invece, la sezione dedicata alla lista dei ticket è stata progettata con l'obiettivo di rendere la consultazione più intuitiva ed efficiente. La visualizzazione dei ticket avviene in un'unica pagina organizzata in forma tabellare, con colonne dedicate alle principali informazioni, come numero identificativo, stato, data di apertura, note e tecnico assegnato. Ogni elemento risulta facilmente distinguibile grazie all'uso di spazi ben bilanciati e di una palette cromatica coerente con l'identità visiva del portale, che contribuisce a una maggiore leggibilità complessiva.

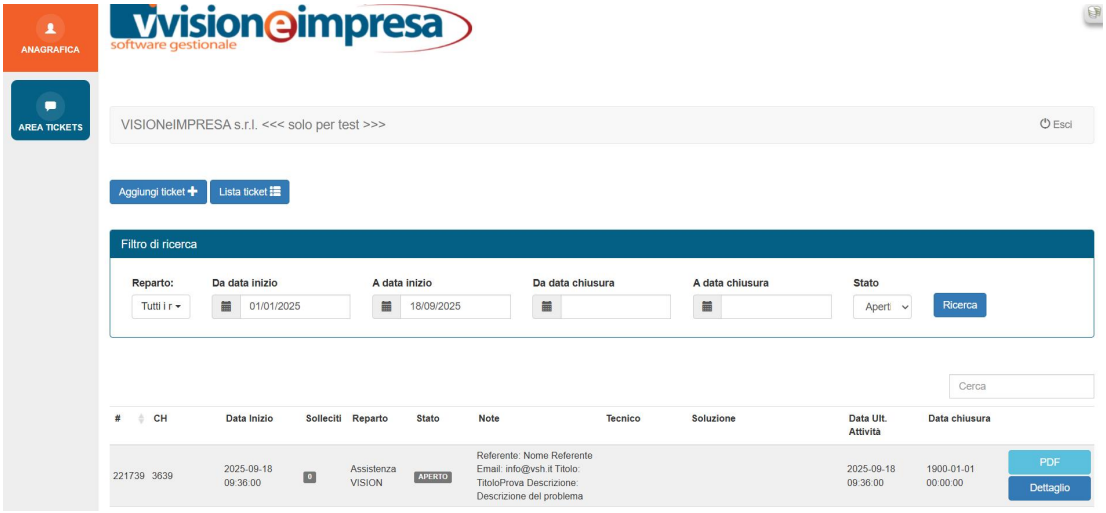


Figura 23: Sezione lista del nuovo portale.

Capitolo 6

Il modulo EmailTicketReader

In questo capitolo viene analizzato il modulo EmailTicketReader, realizzato per automatizzare la gestione delle richieste via email e integrarle nel gestionale Vision, con una panoramica sul suo funzionamento e sulle prospettive di sviluppo.

6.1 Introduzione e obiettivi

Il modulo *EmailTicketReader* rappresenta un componente sperimentale del progetto *Vision Ticketing*, sviluppato con l'obiettivo di automatizzare la ricezione e la classificazione delle richieste di assistenza provenienti da email. *EmailTicketReader* è stato ideato per intercettare automaticamente le email pertinenti all'assistenza tecnica, estrarne i dati principali e predisporli in un formato compatibile con il gestionale.
La gestione delle email non è stata integrata direttamente nel *WebService* per motivi di isolamento funzionale, ma è stata progettata per interagire con esso tramite chiamate alle sue funzioni, così da consentire in futuro l'evoluzione del componente in un microservizio dedicato.

Tra le funzioni previste, il modulo è in grado di:
- leggere e filtrare le email provenienti dalla casella dedicata all'assistenza tecnica;
- estrarre le informazioni rilevanti come mittente, oggetto e contenuto;
- inviare i dati elaborati alle funzioni esposte dal *WebService* che si occupano della creazione e gestione dei ticket nel gestionale *Vision*.

6.2 Architettura e funzionamento

Dal punto di vista architetturale, il modulo è stato progettato come applicazione console in .NET 6, strutturato in componenti indipendenti che cooperano all'interno di un flusso lineare.

Il flusso operativo del modulo prevede, in primo luogo, la connessione al servizio Microsoft Graph API, realizzata tramite autenticazione OAuth2 e l'utilizzo di credenziali fornite da Azure Active Directory.
Una volta stabilita la connessione, il sistema procede al recupero automatico delle email non lette dalla casella di posta configurata, convertendo il corpo del messaggio da formato HTML a testo semplice, in modo da facilitare le successive elaborazioni.

Segue una fase di filtraggio dei messaggi, basata su criteri sia semantici che tecnici, con l'obiettivo di identificare esclusivamente le comunicazioni pertinenti all'assistenza tecnica, escludendo elementi non rilevanti come spam o notifiche automatiche. Per ciascun messaggio ritenuto valido, il sistema effettua una ricerca del mittente all'interno del database aziendale *Vision*, utilizzando una connessione *OleDB*$_G$ e una query diretta alla tabella dei clienti. I dati, così ottenuti, vengono poi strutturati in modo coerente, includendo informazioni essenziali come indirizzo email, oggetto e contenuto del messaggio.

Infine, le informazioni elaborate vengono trasmesse al modulo di creazione ticket nel *WebService* del gestionale Vision tramite chiamata SOAP, consentendo la generazione automatica di una richiesta di assistenza nel sistema aziendale. Al termine dell'esecuzione, *EmailTicketReader* produce due tipi di output, uno in console che mostra l'esito dell'elaborazione, e un file *CSV*$_G$ che salva i risultati in formato UTF-8 con informazioni su oggetto, mittente, data e cliente associato.
Inoltre, il modulo registra informazioni di debug e log per ogni passaggio, includendo timestamp, codici di stato e descrizione dell'esito.
Questo livello di tracciamento è stato introdotto poiché, trattandosi di una fase di sperimentazione, l'obiettivo principale era verificare la fattibilità e la correttezza del flusso di integrazione tra email e gestionale, più che ottenere un prodotto immediatamente utilizzabile in produzione.

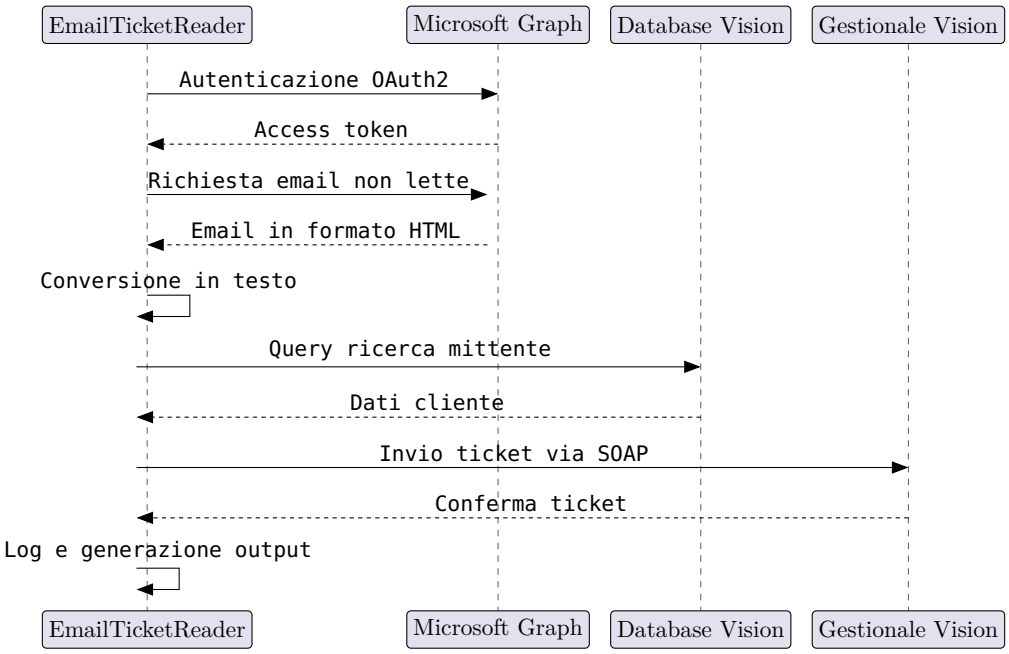


Figura 24: Diagramma di sequenza EmailTicketReader.

6.3 Implementazione

6.3.1 Componenti principali

Il sistema è suddiviso in due macro-componenti, ovvero `Program.cs`, che costituisce il punto di ingresso dell'applicazione e gestisce il flusso principale del processo e `OAuth2EmailService.cs`, che gestisce la connessione a Microsoft 365 tramite Microsoft Graph API, utilizzando il protocollo di autenticazione OAuth2.

Program.cs

Il file `Program.cs` rappresenta il punto di coordinamento principale dell'applicazione e gestisce l'intero flusso operativo del sistema.
Le principali funzioni operative sono le seguenti:

- `Main()`;
- `BuildConfiguration()`;
- `ConfigureServices()`;
- `GetUnreadMessagesFromGraphAsync()`;
- `ParseUserDataFromEmail()`;
- `RecuperaDatiUtenteDaDb()`;
- `InserisciTicketNelGestionaleDebugAsync()`.

La funzione `Main()`, che costituisce l'entrypoint del programma, ha il compito di inizializzare i servizi necessari, configurare l'autenticazione tramite OAuth2 e gestire le diverse fasi di esecuzione.

Attraverso il metodo `BuildConfiguration()`, l'applicazione legge e costruisce la configurazione di sistema a partire dal file `appsettings.json`. Quest'ultimo contiene le impostazioni di configurazione dell'applicazione, tra cui i parametri per collegarsi alla casella email, tra cui server, porta, Client ID e Client Secret per OAuth2.
Il Client ID permette di identificare in modo univoco l'applicazione presso il provider Microsoft (Azure/Office 365), mentre il Client Secret è la password privata che permette all'app di dimostrare la propria identità e accedere alle risorse in modo sicuro.
Per motivi di privacy e sicurezza, nello script sottostante, Codice 7, le variabili sensibili non vengono mostrate direttamente, ma sono riportate con nomi generici, così da non esporre dati riservati.

La funzione `ConfigureServices()` si occupa, invece, di registrare i vari servizi nel container di Dependency Injection, tra cui quelli relativi alla configurazione, al logging e all'autenticazione, garantendo così una gestione modulare e scalabile delle dipendenze.

Il metodo asincrono `GetUnreadMessagesFromGraphAsync()` è dedicato al recupero delle email non lette dalla casella Microsoft 365, utilizzando le interfacce messe a disposizione da Microsoft Graph API, mentre la funzione `ParseUserDataFromEmail()` elabora successivamente il contenuto di ciascun messaggio, estraendo le informazioni rilevanti quali mittente, oggetto e testo del corpo.

Il metodo `RecuperaDatiUtenteDaDb()` esegue poi una ricerca del cliente nel database aziendale VISIONVI, interrogando la tabella *CliFor* in base all'indirizzo email del mittente, al fine di associare correttamente la comunicazione al relativo profilo anagrafico.

Infine, la funzione `InserisciTicketNelGestionaleDebugAsync()` ha il compito di inviare i dati al WebService, permettendo la creazione automatica di un nuovo ticket di assistenza direttamente dal contenuto dell'email.

```json
{
  "ConnectionStrings": {
    "DefaultConnection":
    "Provider=SQLOLEDB;
    Data Source=SERVER_ADDRESS;
    Persist Security Info=True;
    User ID=USER;Password=PASSWORD;
    Initial Catalog=VISIONVI;
    Current Language=Italian"
  },
  "EmailSettings": {
    "MonitoredEmail": "email@vsh.it",
    "CheckIntervalMinutes": 5,
    "MarkAsReadAfterProcessing": true,
    "TicketKeywords": [
      "ticket", "supporto", "aiuto", "problema",
      "bug", "richiesta", "assistenza", "guasto"
    ],
    "ExcludeKeywords": [
      "out of office", "fuori ufficio", "re:",
      "fwd:", "automatic reply", "risposta
      automatica"
    ]
  },
  "AzureAd": {
    "ClientId": "CLIENT_ID",
    "ClientSecret": "CLIENT_SECRET",
    "TenantId": "TENANT_ID"
  },
}
```

Codice 7: Configurazione appsettings.json.

OAuth2EmailService.cs

Il file `OAuth2EmailService.cs` implementa il servizio responsabile dell'accesso sicuro alla casella email tramite OAuth2, garantendo la let-

tura automatizzata dei messaggi senza utilizzare direttamente username e password.

In dettaglio, il servizio si occupa di:

1. **Inizializzazione e autenticazione**
Dal file `appsettings.json` recupera i parametri necessari per l'autenticazione, ovvero Client ID, Client Secret e Tenant ID, e richiede al provider di posta Microsoft 365 un token di accesso temporaneo, che autentica l'applicazione in modo sicuro.

1. **Accesso e lettura della casella email**
Utilizza il token ottenuto per interrogare le API del provider e recuperare informazioni sulle email della casella monitorata. Durante questa interrogazione vengono effettuate diverse operazioni tra cui la verifica dell'esistenza dell'utente e della casella, e la lettura dei messaggi più recenti, con filtraggio su oggetto, mittente e stato di lettura.

1. **Gestione sicura dei dati**
Tutte le credenziali sensibili come Client Secret, password della casella e Tenant ID, sono lette da `appsettings.json` ma, per motivi di sicurezza e privacy, il servizio non memorizza né espone le password, utilizzando solo il token OAuth2 per le operazioni di accesso.

6.4 Analisi e prospettive del modulo

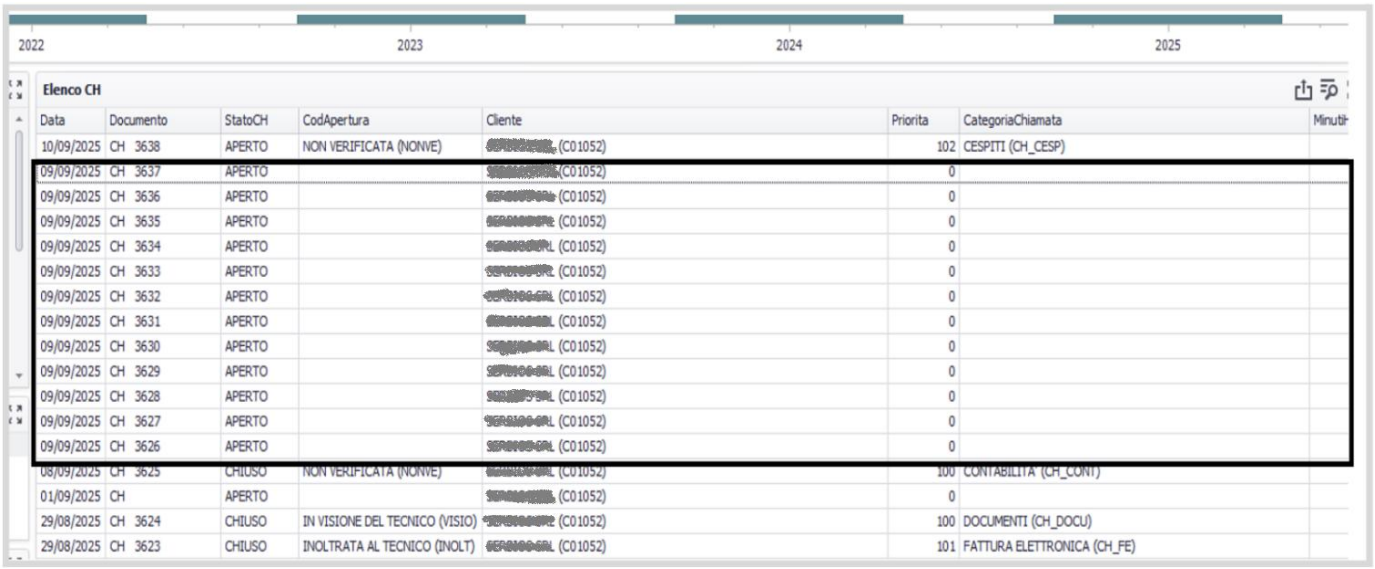
6.4.1 Stato attuale

Attualmente, il modulo *EmailTicketReader* si trova ancora in fase sperimentale e viene eseguito in ambiente controllato.
Durante lo sviluppo del modulo sono stati riscontrati alcuni limiti e problematiche che influenzano il flusso di gestione automatica dei ticket, evidenziando aree di possibile miglioramento.

Uno degli aspetti principali riguarda il database *Clifor*, che contiene le informazioni sui clienti e sulle aziende. Attualmente, il modulo associa automaticamente un ticket a un cliente, solo se l'indirizzo email del mittente è già presente nel database. Tuttavia, se un cliente invia una mail da un indirizzo non registrato, il sistema non è in grado di effettuare l'associazione automatica.
In questi casi, il ticket viene temporaneamente salvato senza collegamento a un cliente specifico, con un utente generico, ed è richiesto un intervento manuale da parte del personale di assistenza per aggiornare i dati nel gestionale.

Questo comportamento, sebbene gestibile, allunga i tempi di elaborazione delle richieste, riducendo parzialmente il beneficio dell'automazione.

Inoltre il sistema ha evidenziato alcune anomalie nella compilazione dei documenti CH seppure sia in grado di eseguire le funzioni previste durante le prove di integrazione con il gestionale *Vision*.
In particolare, tre campi non risultano valorizzati correttamente, nonostante per due di essi fossero stati definiti valori di default nel database. Nella Figura 25 è possibile vedere le tre colonne *Codice apertura*, *Priorità* e *Categoria chiamata* non valorizzate; il nome della colonna *Cliente* è stato censurato per motivi di privacy.
Tali risultati, pur non compromettendo il corretto funzionamento generale del modulo, evidenziano la necessità di ulteriori verifiche e rifiniture nella fase di integrazione completa con il gestionale *Vision*.



Data	Documento	StatoCH	CodApertura	Cliente	Priorita	CategoriaChiamata	Minutiv
10/09/2025	CH 3638	APERTO	NON VERIFICATA (NONVE)	[redacted] (C01052)	102	CESPITI (CH_CESP)	
09/09/2025	CH 3637	APERTO		[redacted] (C01052)	0		
09/09/2025	CH 3636	APERTO		[redacted] (C01052)	0		
09/09/2025	CH 3635	APERTO		[redacted] (C01052)	0		
09/09/2025	CH 3634	APERTO		[redacted] (C01052)	0		
09/09/2025	CH 3633	APERTO		[redacted] (C01052)	0		
09/09/2025	CH 3632	APERTO		[redacted] (C01052)	0		
09/09/2025	CH 3631	APERTO		[redacted] (C01052)	0		
09/09/2025	CH 3630	APERTO		[redacted] (C01052)	0		
09/09/2025	CH 3629	APERTO		[redacted] (C01052)	0		
09/09/2025	CH 3628	APERTO		[redacted] (C01052)	0		
09/09/2025	CH 3627	APERTO		[redacted] (C01052)	0		
09/09/2025	CH 3626	APERTO		[redacted] (C01052)	0		
08/09/2025	CH 3625	CHIUSO	NON VERIFICATA (NONVE)	[redacted] (C01052)	100	CONTABILITA' (CH_CONT)	
01/09/2025	CH	APERTO		[redacted] (C01052)	0		
29/08/2025	CH 3624	CHIUSO	IN VISIONE DEL TECNICO (VISIO)	[redacted] (C01052)	100	DOCUMENTI (CH_DOCU)	
29/08/2025	CH 3623	CHIUSO	INOLTRATA AL TECNICO (INOLT)	[redacted] (C01052)	101	FATTURA ELETTRONICA (CH_FE)	

Figura 25: Dettaglio documenti CH creati con *EmailTicketReader*.

6.4.2 Sviluppi futuri

In prospettiva, il modulo *EmailTicketReader* potrà essere ulteriormente potenziato e integrato nel sistema *Vision* e tra i principali sviluppi futuri si prevedono l'estensione della logica di elaborazione per includere la gestione degli allegati, la corretta valorizzazione dei campi dei documenti CH e la trasformazione del modulo in un servizio capace di eseguire, in modo autonomo e continuo, il controllo della casella di posta.

Capitolo 7

Test e validazione

Il capitolo che segue descrive le attività di test e validazione svolte sul progetto Vision Ticketing, con l'obiettivo di verificare il corretto funzionamento delle componenti sviluppate e la loro integrazione con il gestionale Vision Enterprise.

7.1 Obiettivo dei test

I test svolti hanno avuto l'obiettivo di garantire che le modifiche introdotte al portale *Vision Ticket* e al backend fossero pienamente funzionanti e coerenti con i requisiti del progetto, oltre ad essere integrate correttamente con il gestionale.

In particolare si è voluto verificare:
- la correttezza degli input e degli output delle nuove API SOAP sviluppate in ambiente .NET;
- l'integrazione tra frontend e backend, validando i flussi principali del portale web;
- la coerenza funzionale tra il nuovo portale e quello già esistente di OfficeGroup;
- l'usabilità e coerenza grafica dell'interfaccia utente dopo le personalizzazioni.

L'attività di test, svolta manualemnte, si è articolata principalmente in test funzionali e d'integrazione. I primi sono stati eseguiti con gli strumenti Postman e SoapUI, inserendo i parametri necessari o, nei casi richiesti, predisponendo payload XML specifici, e verificando il comportamento delle API in risposta a input validi e non validi. I secondi, invece, sono stati svolti sia in autonomia per verificare l'integrazione tra le API, il database e il portale, sia con il supporto dei dipendenti dell'azienda, al fine di osservare il funzionamento dell'intero flusso in condizioni operative reali e identificare eventuali mancanze funzionali non previste in fase progettuale.

7.2 Ambiente di test

L'ambiente di test è stato predisposto per rispecchiare il più fedelmente possibile quello di produzione, garantendo affidabilità ai risultati.

Componenti principali

- **Backend**: *WebService* .NET collegato al database *Vision* ERP in ambiente di sviluppo.
- **Frontend**: Portale web containerizzato tramite DDEV e Docker, eseguito localmente su *WSL*$_G$.
- **Gestione del codice**: repository Git su Azure DevOps, utilizzato per versioning e tracciabilità.
- **Strumenti di test**: SoapUI e Postman per la validazione delle API SOAP.

7.3 Classificazione dei test

Per ogni test viene adottata una nomenclatura univoca che consente di distinguere la tipologia di test eseguito.
La nomenclatura utilizzata segue la struttura:

[Tipo]-[Numero progressivo]

Dove per Tipo si indica:

TF : test funzionale;
TI : test di integrazione.

7.4 Test funzionali

I test funzionali hanno lo scopo di verificare che ogni funzione o servizio del sistema svolga correttamente le operazioni previste dai requisiti, producendo output coerenti con gli input forniti.
In particolare, per il progetto *Vision Ticketing*, sono stati testati i metodi SOAP sviluppati nel backend, inviando richieste sia con parametri validi che non validi, per controllare il comportamento in entrambi i casi.

Codice	Descrizione	Stato
TF-1	Verifica che la funzione `nf_oit_ins_ticket` crei un nuovo ticket con tutti i parametri obbligatori e restisca l'ID generato.	Superato
TF-2	Verifica che la funzione `nf_oit_ins_ticket` con input privo del campo `id_sog_commerciale` rifiuti la creazione e restituisca errore di validazione.	Superato
TF-3	Verifica che la funzione `nf_oit_chiudi_segnalazione` chiuda il ticket con aggiornamento dello stato e del timestamp di chiusura.	Superato
TF-4	Verifica che la funzione `nf_oit_sollecito` inserisca sollecito su ticket aperto e incrementi la priorità.	Superato
TF-5	Verifica del comportamento di `nf_oit_sollecito` su ticket già chiuso, il sistema deve impedire l'operazione e segnalare l'errore.	Superato
TF-6	Verifica che la funzione `nf_oit_get_tickets` ritorni correttamente la lista dei ticket.	Superato
TF-7	Test di input non valido su `nf_oit_get_tickets` con date in formato errato con gestione dell'errore.	Superato
TF-8	Verifica che la funzione `nf_oit_get_attivita_collegate` recuperi correttamente le ultime attività collegate al ticket selezionato.	Superato

Codice	Descrizione	Stato
TF-9	Verifica che la funzione `nf_oit_get_area_programma` restituisca correttamente l'elenco delle aree programma disponibili.	Superato
TF-10	Verifica che la funzione `nf_oit_get_sett_personale` recuperi i reparti disponibili.	Superato

Tabella 7: Tracciamento dei test funzionali.

7.5 Test di integrazione

I test di integrazione mirano a verificare che i diversi moduli del sistema comunichino correttamente tra loro, assicurando che i dati transitino in modo coerente e senza perdita di informazioni.
Nel caso del progetto *Vision Ticketing*, questi test hanno validato l'interazione tra il frontent del portale web, il backend *WebService* e e il database del gestionale *Vision*.

Codice	Descrizione	Stato
TI-1	Verifica del login utente con credenziali valide, l'accesso al portale è consentito.	Superato
TI-2	Verifica del login con credenziali non valide, restuisce messaggio di errore.	Superato
TI-3	Creazione di un ticket dal portale con parametri completi e corretta registrazione nel gestionale *Vision*.	Superato
TI-4	Creazione ticket con parametri mancanti, verifica corretta gestione dell'errore con mancato inserimento.	Superato

Codice	Descrizione	Stato
TI-5	Recupero e visualizzazione della lista dei ticket dal portale tramite API `nf_oit_get_tickets`.	Superato
TI-6	Recupero del dettaglio di un ticket, verifica coerenza dei dati con quelli presenti nel gestionale *Vision*.	Superato
TI-7	Inserimento sollecito su ticket aperto, l'operazione è accettata e la priorità è aggiornata; controllo operazione bloccata su ticket chiuso.	Superato
TI-8	Chiusura ticket dal portale e aggiornamento automatico dello stato nel gestionale *Vision*.	Superato
TI-9	Modifica del campo *Tecnico* nel gestionale *Vision* e verifica aggiornamento nel portale.	Superato
TI-10	Modifica del campo *Area Programma* nel gestionale *Vision* e verifica aggiornamento nel portale.	Superato
TI-11	Modifica del campo *Note Soluzioni* nel gestionale *Vision* e verifica aggiornamento nel portale.	Superato

Tabella 8: Tracciamento dei test di integrazione.

7.6 Requisiti soddisfatti

Per validare e accertare che il sistema sviluppato risponda ai requisiti definiti in fase di analisi, in basso è riportato un elenco riepilogativo dei requisiti progettuali, ciascuno associato al relativo stato di soddisfacimento, al fine di documentare in modo chiaro l'esito finale della validazione.

7.6.1 Requisiti funzionali

Obbligatori

Codice	Descrizione	Stato
FO-1	Il sistema deve consentire l'accesso autenticato ai clienti e ai rivenditori tramite credenziali personali.	Soddisfatto
FO-2	Il sistema deve permettere l'inserimento di nuove richieste di assistenza da parte degli utenti autenticati.	Soddisfatto
FO-3	Il sistema deve permettere il sollecito, la chiusura e la visualizzazione in dettaglio dei ticket già creati.	Soddisfatto
FO-4	Ogni richiesta di assistenza deve essere associata a un cliente.	Soddisfatto
FO-5	Il sistema deve registrare automaticamente le richieste di assistenza all'interno del gestionale *Vision Enterprise* come documenti di tipo CH.	Soddisfatto
FO-6	Il sistema deve permettere l'inserimento degli allegati durante la creazione delle richieste di assistenza.	Soddisfatto
FO-7	Il sistema deve permettere la visualizzazione degli allegati sia dal portale sia dal gestionale.	Soddisfatto
FO-8	Il personale interno deve poter modificare, chiudere o riaprire i ticket direttamente dal gestionale.	Soddisfatto

Codice	Descrizione	Stato
FO-9	Il sistema dovrebbe consentire ai tecnici di aggiungere commenti o allegati ai ticket in lavorazione.	Soddisfatto
FO-10	Il sistema dovrebbe consentire la visualizzazione di uno storico delle richieste di assistenza.	Soddisfatto

Tabella 9: Stato dei requisiti funzionali obbligatori.

Desiderabili

Codice	Descrizione	Stato
FD-1	Il sistema dovrebbe permettere la ricerca e il filtraggio dei ticket per stato, priorità o area funzionale.	Soddisfatto
FD-2	Il sistema dovrebbe permettere la riassegnazione dei ticket tra operatori diversi.	Soddisfatto
FD-3	Il sistema deve essere in grado di leggere le email inviate all'indirizzo assistenza@vsh.it e convertirle in ticket nel gestionale	Soddisfatto parzialmente
FD-4	Il sistema dovrebbe salvare e visualizzare gli allegati delle email all'indirizzo assistenza@vsh.it.	Non soddisfatto

Tabella 10: Stato dei requisiti funzionali desiderabili

Opzionali

Codice	Descrizione	Stato
FP-1	Il sistema può integrarsi con l'applicazione *moviTICKET* per importare automaticamente richieste provenienti da dispositivi mobili.	Non soddisfatto
FP-2	Il sistema può consentire la creazione automatica di ticket a partire dalle chiamate vocali ricevute tramite impianto VoIP.	Non soddisfatto
FP-3	Il sistema può consentire la trascrizione automatica delle chiamate vocali in testo da allegare al ticket.	Non soddisfatto

Tabella 11: Stato dei requisiti funzionali opzionali.

7.6.2 Requisiti non funzionali

Obbligatori

Codice	Descrizione	Stato
NFO-1	Il sistema deve assicurare la compatibilità con il gestionale *Vision Enterprise* e il database SQL Server.	Soddisfatto
NFO-2	Il portale web deve essere accessibile dai principali browser web (Chrome, Firefox, Edge, Safari).	Soddisfatto
NFO-3	Il sistema deve garantire tempi di risposta accettabili.	Soddisfatto

Codice	Descrizione	Stato
NFO-4	Il sistema deve fornire messaggi di errore chiari in caso di richieste non realizzabili.	Soddisfatto

Tabella 12: Stato dei requisiti non funzionali obbligatori.

Desiderabili

Codice	Descrizione	Stato
NFD-1	L'interfaccia dovrebbe essere coerente con il layout e l'identità visiva del sito aziendale VisioneImpresa.	Soddisfatto
NFD-2	Il sistema dovrebbe offrire un'esperienza utente intuitiva e coerente, minimizzando gli errori di input.	Soddisfatto
NFD-3	Il portale dovrebbe essere responsivo e adattarsi automaticamente a diverse dimensioni di schermo.	Soddisfatto

Tabella 13: Stato dei requisiti non funzionali desiderabili.

Capitolo 8

Conclusioni

A conclusione del lavoro, dopo aver analizzato il portale e illustrato le nuove funzionalità implementate, in questo capitolo vengono esposte le considerazioni finali relative ai risultati conseguiti, una riflessione personale sull'esperienza maturata e una panoramica sugli sviluppi futuri che potrebbero interessare il sistema.

8.1 Risultati

Al termine dello stage, il lavoro svolto ha prodotto due componenti principali: il portale web e le API dedicate.
Le API realizzate garantiscono un'esposizione sicura e strutturata dei dati necessari alle operazioni del nuovo sistema, mentre il portale consente di interagire con tali servizi tramite un'interfaccia moderna e coerente con gli standard attuali.
Grazie a queste componenti, i clienti possono ora usufruire di un ambiente centralizzato in cui consultare le informazioni aziendali, gestire le proprie richieste di assistenza e monitorare il loro avanzamento.
In contemporanea, il sistema semplifica le attività operative dei dipendenti di VisioneImpresa, riducendo i tempi di evasione delle richieste e migliorando la qualità del servizio di assistenza.
Il nuovo sistema *Vision Ticketing* non porta solo benefici diretti dovuti all'automazione ma contribuisce alla modernizzazione dell'azienda, che adattandosi alle nuove tecnologie attraverso l'utilizzo di processi più efficienti, mantiene la sua appetibilità nel mercato italiano.
Inoltre, l'adozione delle tecnologie aggiornate e la nuova architettura permettono di predisporre il sistema a futuri ampliamenti, come il modulo *EmailTicketReader* , che una volta ottimizzato potrà intercettare autonomamente le email di assistenza e inserirle nel gestionale, diventando un sistema sempre più completo e integrato.

8.2 Sviluppi futuri

Il progetto *Vision Ticketing* rappresenta una prima versione del sistema, con le principali funzionalità implementate e testate. Tuttavia, esistono alcune direzioni di sviluppo che, per motivi di tempo e complessità tecnica, non sono state ancora integrate. Questi sviluppi futuri potranno arricchire ulteriormente il sistema, migliorandone l'usabilità e le possibilità operative.

8.2.1 Integrazione con moviTICKET

Una delle funzionalità opzionali previste ma non implementate durante lo stage, riguarda l'integrazione con *moviTICKET*, ossia un'app mobile che permette di inserire ticket di assistenza e seguirne l'intero ciclo di vita. L'obiettivo della sua integrazione con il sistema attuale è la sincronizzazione dell'applicazione con il portale web, per gestire e visualizzare i ticket presenti nell'app direttamente dal portale, riducendo errori o duplicazioni di informazioni. L'implementazione futura con *moviTICKET* rappresenta quindi un passo fondamentale verso una gestione più completa e integrata dei flussi di lavoro.

8.2.2 Integrazione del VoIP con speech-to-text

Un ulteriore sviluppo riguarda l'integrazione di funzionalità VoIP con trascrizione automatica dei messaggi vocali in testo, denominata *speech-to-text*. Questa tecnologia permetterebbe di registrare e trascrivere le comunicazioni telefoniche in tempo reale, creando un archivio testuale delle conversazioni. Successivamente, le trascrizioni verrebbero convertite in ticket, estraendo le principali informazioni come referente, nome dell'azienda, oggetto del ticket e descrizione del problema, riducendo i tempi di elaborazione.
Purtroppo a causa della complessità tecnica e della necessità di sperimentazione con i servizi di riconoscimento vocale, come strumenti basati sull'Intelligenza Artificiale, questa funzionalità non è stata ancora integrata nella versione attuale del sistema, ma rappresenta un obiettivo per gli sviluppi futuri.

8.3 Valutazione personale

L'esperienza di stage presso VisioneImpresa mi ha permesso di ampliare in modo significativo le mie competenze tecniche e professionali. In particolare, ho avuto l'opportunità di sviluppare API per il servizio del

portale aziendale, approfondendo la progettazione di interfacce applicative, la gestione della sicurezza e l'interazione con servizi esterni.
Lo stage mi ha inoltre dato la possibilità di partecipare attivamente al processo di rinnovamento tecnologico dell'azienda, grazie alla modernizzazione degli strumenti software e all'introduzione di soluzioni orientate all'automazione dei flussi di lavoro. È stato stimolante poter assistere da vicino all'evoluzione di un'azienda storica che, attraverso l'adozione di nuove tecnologie, si proietta verso un modello gestionale più innovativo.

Durante il percorso non sono mancate alcune difficoltà, che hanno rappresentato però occasioni di crescita. La principale ha riguardato la gestione del database di *Vision*, un sistema complesso e di grandi dimensioni che, a causa della sua struttura originaria e della progressiva evoluzione nel tempo, presenta alcune incongruenze e disomogeneità nei dati. Queste caratteristiche, tipiche di sistemi nati in contesti aziendali precedenti alla diffusione di standard moderni di memorizzazione, hanno reso più complessa l'estrazione e la normalizzazione delle informazioni, richiedendo un'attenta analisi e una revisione dei processi di accesso ai dati.
Un'ulteriore difficoltà è legata alla gestione dei tempi di sviluppo, poiché la realizzazione e il perfezionamento del portale hanno richiesto un'attenta revisione e una costante collaborazione con i colleghi di VisioneImpresa. La mancanza di una documentazione completa del codice sorgente preesistente ha reso necessaria un'attività preliminare di analisi del software, allo scopo di comprenderne la logica interna e garantire la compatibilità con i nuovi moduli sviluppati. Questa fase, pur impegnativa, mi ha permesso di affinare la capacità di apprendimento autonomo e di lavorare in team in un contesto reale.
Nonostante tali difficoltà, considero questa esperienza estremamente positiva, in quanto mi ha offerto una visione concreta delle dinamiche di sviluppo software all'interno di un'azienda italiana, mettendomi in contatto diretto con problematiche e metodologie tipiche del mondo professionale. Avrei desiderato disporre di maggior tempo per approfondire ulteriori funzionalità opzionali, come l'integrazione del sistema VoIP e di strumenti *speech-to-text* basati su intelligenza artificiale e machine learning, che rappresentano un'evoluzione naturale del progetto e contribuirebbero ulteriormente al processo di modernizzazione di VisioneImpresa.

Ringraziamenti

Innanzitutto, vorrei esprimere la mia gratitudine al Prof. Marco Zanella relatore della mia tesi, per l'aiuto e il continuo sostegno fornitomi durante la stesura del lavoro.

Desidero ringraziare i miei genitori per il sostegno e per essermi stati vicini durante gli anni di studio.

Un ringraziamento speciale ad Angelo, per la sua presenza costante, la pazienza e per aver creduto in me anche nei momenti più incerti, ricordandomi ogni giorno il valore della determinazione e della serenità.

Ci tengo infine a ringraziare i colleghi di VisioneImpresa S.r.l. e il tutor aziendale Francesco Turra per avermi dato l'opportunità di lavorare a questo progetto.

Padova, Dicembre 2025 *Valentina Schivo*

Glossario

***API* – Application Programming Interface**: Insieme di regole, protocolli e strumenti che consentono a due diverse applicazioni software di comunicare tra loro e scambiarsi dati o funzionalità. 3

***CH* – Call Help**: Abbreviazione usata internamente per indicare la gestione delle richieste di assistenza clienti, con cui vengono indicate le chiamate o ticket di supporto aperti. 5

***CSV* – Comma-Separated Values**: Formato di file di testo in cui i dati sono organizzati in forma tabellare e separati da virgole o altri delimitatori. Ogni riga rappresenta un record, ed è comunemente utilizzato per l'esportazione e l'importazione di dati tra applicazioni, come fogli di calcolo e database. 48

***CRM* – Customer Relationship Management**: Filosofia gestionale e insieme di strumenti volti a costruire relazioni di lungo periodo con la clientela, attraverso la personalizzazione dell'offerta e del servizio, con l'obiettivo di incrementare la customer satisfaction e la fedeltà al brand. 1

***DI* – Dependency Injection**: Design pattern che implementa il principio dell'inversione del controllo, in cui un oggetto riceve le istanze degli oggetti da cui dipende da un agente esterno chiamato injector o container, anziché crearle autonomamente al proprio interno. 12

***ERP* – Enterprise Resource Planning**: Software integrato che aiuta le aziende a gestire e automatizzare i processi aziendali quotidiani, come finanza, contabilità, risorse umane, produzione e catena di approvvigionamento, centralizzando i dati in un'unica fonte. 21

***HTTP* – HyperText Transfer Protocol**: Protocollo standard a livello applicativo che costituisce la base del funzionamento del World Wide Web. La sua funzione principale è regolare la comunicazione e lo scambio di dati tra un client/broswer e un server web. 11

***IDE* – Integrated Development Environment**: Applicazione software che fornisce ai programmatori e agli sviluppatori un set completo di strumenti e funzionalità per semplificare e ottimizzare l'intero ciclo di sviluppo del software. 7

***IIS* – Internet Information Services**: Termine per indicare il server web e un insieme di servizi Internet-based sviluppati da Microsoft per i sistemi operativi Windows. La sua funzione principale è quella di ospitare siti e applicazioni web, gestendo le richieste provenienti dai broswer su reti locali o globali. 8

***IMAP* – Internet Message Access Protocol**: Protocollo a livello applicativo utilizzato per la ricezione e l'accesso ai messaggi di posta elettronica. 13

***JSON* – JavaScript Object Notation**: Formato standard di testo leggero e indipendente dal linguaggio, utilizzato principalmente per lo scambio di dati tra applicazioni web. JSON rappresenta i dati in modo strutturato e gerarchico utilizzando coppie chiave-valore. 11

***MVC* – Model-View-Controller**: Pattern architetturale utilizzato nello sviluppo di software, in particolare per le interfacce utente e le applicazioni web. Il suo scopo principale è quello di separare la gestione dei dati dalla logica di presentazione. 10

***OleDB* – Object Linking and Embedding Database**: API sviluppata da Microsoft che fornisce un accesso uniforme e standardizzato a una varietà di origini dati diverse, come database relazionali, fogli di calcolo e file di testo. 48

***REST* – REpresentational State Transfer**: Stile architetturale che definisce un insieme di vincoli e linee guida per la progettazione di sistemi distribuiti e servizi web scalabili. Le risposte della REST sono generalmente in JSON, ma possono anche essere in un formato di dati diverso come XML o HTML. 11

***SMTP* – Simple Mail Transfer Protocol**: Protocollo standard a livello applicativo utilizzato esclusivamente per l'invio e l'inoltro di messaggi di posta elettronica. 13

***SOAP* – Simple Object Access Protocol**: Protocollo standard utilizzato per lo scambio di messaggi strutturati in formato XML tra componenti software. Consente a diverse applicazioni, che utilizzano lingue e piattaforme eterogenee, di comunicare tra loro. 3

***SSMS* – SQL Server Management Studio**: Ambiente di sviluppo integrato e strumento di amministrazione fornito da Microsoft. Permette di gestire le infrastrutture SQL, interrogare e manipolare i dati. 8

***WSL* – Windows Subsystem for Linux**: Funzionalità in Windows che permette di eseguire un ambiente Linux direttamente all'interno del sistema operativo Windows, senza bisogno di installare una macchina virtuale. 56

***XML* – eXtensible Markup Language**: Linguaggio di markup per la codifica di documenti in un formato strutturato. La sua funzione è quella di definire regole per lo scambio e l'archiviazione strutturata dei dati. 11

Bibliografia

[1] R. Boyd, *Getting Started with OAuth 2.0, Cap. "Client Credentials Flow"*. O'Reilly Media, 2012, pp. 41–43. [Online]. Disponibile su: https://books.google.it/books?id=qcsoLHusAFsC&pg=PA41&hl=it&source=gbs_toc_r&cad=2#v=onepage&q&f=false

Sitografia

VisioneImpresa. url: https://www.vsh.it/

OfficeGroup. url: https://www.officegroup.it/

Libreria Mail.dll. url: https://www.limilabs.com/mail

Documentazione CakePHP. url: https://book.cakephp.org/5/en/index.html

Documentazione Azure Active Directory. url: https://learn.microsoft.com/en-us/azure/active-directory/

OAuth 2.0 Authorization Framework. url: https://datatracker.ietf.org/doc/html/rfc6749