

**Università degli Studi di Padova**

DIPARTIMENTO DI MATEMATICA "TULLIO LEVI-CIVITA "

CORSO DI LAUREA IN INFORMATICA



**Instant Developer: progettazione e  
implementazione di un configuratore  
catalogo/prodotti**

*Tesi di laurea triennale*

*Relatore*

Prof.Francesco Ranzato

*Laureando*

Riccardo Bernucci







# Sommario

Il presente documento descrive il lavoro svolto durante il periodo di stage, della durata di circa trecentodieci ore, dal laureando Riccardo Bernucci presso l'azienda Tepui S.r.l. Gli obiettivi da raggiungere sono stati molteplici.

In primo luogo hanno richiesto uno studio e apprendimento degli strumenti adottati dall'azienda per lo sviluppo delle applicazioni e dei data warehouse che sono rispettivamente Instant Developer e Microsoft SQL Server. In secondo luogo ho dovuto seguire una buona metodologia nello sviluppo delle unità della componente da realizzare effettuando molteplici test. Inoltre, al termine dello stage, in caso di raggiungimento dello stato di validazione e collaudo, si è prefissato il rilascio del progetto con eventuale gestione autonoma di modifiche correttive e/o adattive segnalate dal cliente.

Gli obiettivi in generale sono mirati all'apprendimento di come Tepui S.r.l. gestisce i suoi clienti e realizza i software richiesti. In aggiunta, sono serviti a perseguire l'ulteriore obiettivo di andare a migliorare le capacità acquisite nella disciplina di Ingegneria del Software.

Questa tesi si compone di 4 capitoli. Il primo presenta l'azienda, come è nata, quali tecnologie e quale metodologia di lavoro adotta. Il secondo, invece, presenta il progetto al centro delle attività svolte durante lo stage, i vincoli e gli obiettivi prefissati. Nel terzo capitolo viene presentato il progetto nel dettaglio presentando le scelte di progettazione e implementazione seguite da una piccola digressione su Instant Developer. Infine, il quarto capitolo presenta una valutazione del tirocinio, sia a livello oggettivo, considerando, ad esempio, il grado di soddisfacimento degli obiettivi, che soggettivo, esponendo, quindi, una mia valutazione personale sulle attività svolte.



“The future belongs to those who believe in the beauty of their dreams.”

— Eleanor Roosevelt

# Ringraziamenti

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

*Desidero ringraziare con affetto mia madre e mio padre per il sostegno, il grande aiuto e per essermi stati vicini in ogni momento durante gli anni di studio. Dalle superiori all'università avete sempre cercato di appoggiare le mie scelte. Avete contribuito alla mia formazione personale, rendendomi l'uomo che sono oggi.*

*Vorrei ringraziare i colleghi che ho incontrato durante il mio tirocinio presso Tepui S.r.l per la loro collaborazione. In particolare, mi rivolgo al mio supervisore, il signor Martino Vedana, vorrei ringraziarla per l'incredibile disponibilità e per tutte le opportunità che mi sono state date nel condurre la mia esperienza lavorativa.*

*Ho desiderio di ringraziare poi i miei amici per tutti i bellissimi anni passati insieme e le mille avventure vissute. Il tempo con voi ha avuto un peso determinante nel conseguimento di questo risultato, punto di arrivo e contemporaneamente di partenza della mia vita.*

*Un grazie speciale a Maria, la persona che più di tutte è stata capace di capirmi e di sostenermi nei momenti difficili.*

*Padova, Settembre 2019*

Riccardo Bernucci





# Indice

<b>1</b>	<b>L'Azienda</b>	<b>1</b>
1.1	Presentazione dell'azienda . . . . .	1
1.1.1	Tepui S.r.l . . . . .	1
1.1.2	Prodotti e servizi . . . . .	2
1.1.3	Tecnologie di riferimento . . . . .	3
1.2	Processi aziendali . . . . .	4
1.2.1	Miglioramento della qualità dei processi . . . . .	4
1.2.2	Metodologia agile . . . . .	5
1.3	Strumenti a supporto dei processi . . . . .	6
1.3.1	Gestione di progetto . . . . .	6
1.3.2	Documentazione . . . . .	7
1.3.3	Sistema di versionamento . . . . .	7
1.3.4	Ambiente di sviluppo . . . . .	8
1.3.5	Sistemi operativi . . . . .	9
1.4	Clientela . . . . .	9
<b>2</b>	<b>Stage</b>	<b>11</b>
2.1	Vantaggi dell'azienda . . . . .	11
2.2	Presentazione del progetto . . . . .	11
2.2.1	Visualizzazione dettaglio prodotto . . . . .	13
2.2.2	Inserimento attraverso modali . . . . .	13
2.2.3	Gestione delle configurazioni . . . . .	13
2.2.4	Gestione delle traduzioni . . . . .	13
2.2.5	Altri interventi . . . . .	13
2.2.6	Obiettivi . . . . .	14
2.3	I vincoli . . . . .	14
2.3.1	Vincoli temporali . . . . .	14
2.3.2	Vincoli metodologici . . . . .	16
2.3.3	Vincoli tecnologici . . . . .	16
2.4	Scelta e obiettivi personali . . . . .	17
<b>3</b>	<b>Progetto e attività di stage</b>	<b>19</b>
3.1	Formazione . . . . .	19
3.2	Progettazione . . . . .	19
3.2.1	Database . . . . .	19
3.2.2	Design Pattern . . . . .	21
3.3	Codifica . . . . .	23
3.3.1	Documenti . . . . .	23

3.3.2	Videate . . . . .	26
3.4	Verifica, validazione e collaudo . . . . .	27
3.5	Modulo RTC . . . . .	28
3.6	Altri interventi . . . . .	28
<b>4</b>	<b>Valutazione retrospettiva</b>	<b>29</b>
4.1	Instant Developer . . . . .	29
4.1.1	Instant Developer: creazione di una schermata base . . . . .	29
4.1.2	Estensioni . . . . .	30
4.2	Obiettivi . . . . .	30
4.2.1	Stage . . . . .	30
4.2.2	Personalì . . . . .	30
4.3	Considerazioni personali . . . . .	31
	<b>Glossario</b>	<b>33</b>
	<b>Acronimi e abbreviazioni</b>	<b>35</b>
	<b>Fonti bibliografiche</b>	<b>37</b>

# Elenco delle figure

1.1	Logo dell'azienda . . . . .	1
1.2	Prodotti e servizi forniti nella loro pagina web . . . . .	3
1.3	Linguaggi di programmazione . . . . .	3
1.4	Database . . . . .	4
1.5	Ciclo PDCA ( <a href="https://bit.ly/2y3khfp">https://bit.ly/2y3khfp</a> ) . . . . .	5
1.6	Conversazione su Teams . . . . .	6
1.7	Kanban dell'applicazione iDo (con nome cliente censurato) . . . . .	7
2.1	Front-end realizzato con i dati inseriti dal configuratore . . . . .	12
2.2	Diagramma di Gantt . . . . .	14
3.1	Diagramma Entity Relationship . . . . .	20
3.2	Design pattern strutturale: Facade ( <a href="https://bit.ly/2McTHsR">https://bit.ly/2McTHsR</a> ) . . . . .	22
3.3	Diagramma UML delle classi . . . . .	22

# Elenco delle tabelle



# Capitolo 1

## L'Azienda

*Questo capitolo descrive nel dettaglio l'azienda ospitante andando a definire il suo business, l'organizzazione interna e le tecnologie adottate per soddisfare i clienti.*

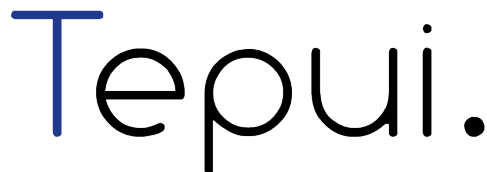
### 1.1 Presentazione dell'azienda

Questa sezione si concentra sull'azienda che ha ospitato lo stage, fornendo una chiara spiegazione dei prodotti e dei servizi forniti assieme alla descrizione delle tecnologie adottate.

#### 1.1.1 Tepui S.r.l

Tepui S.r.l è una software house specializzata nello sviluppo di applicazioni gestionali attraverso l'utilizzo di strumenti CASE.

Nasce nel 2016 dall'idea di tre persone. I fondatori hanno incontrato diverse realtà lavorative prima di portare avanti il progetto di aprire un'azienda. Due di queste hanno lavorato nell'ambito della business intelligence. Sulla base degli studi effettuati, risulta molto difficile analizzare i dati delle imprese in quanto ognuna ha una sua metodologia lavorativa, soprattutto nella gestione dei database o data warehouse. Sono arrivati alla conclusione che, se fosse possibile fornire un prodotto software standardizzato, sarebbe molto più semplice anche l'analisi. In ragione di ciò, i due hanno iniziato a cercare un'applicazione che permettesse di creare software in maniera rapida e compatibile con diversi sistemi operativi. Il risultato della loro ricerca è stato Instant Developer ([InDe](#)) ed a partire da questo momento si è unito il terzo fondatore già esperto dell'applicazione che li ha formati nel suo utilizzo.



**Figura 1.1:** Logo dell'azienda

I tre fondatori sono fortemente convinti nei vantaggi tecnologici ed economici che l'implementazione mediante strumenti di sviluppo rapido porta ai propri clienti. L'azienda nasce come evoluzione di tre progetti indipendenti dei tre soci fondatori nelle aree della consulenza direzionale, consulenza nei processi produttivi, business intelligence e produzione di software. Inizialmente, il software è stato utilizzato come strumento a supporto della consulenza e, l'esigenza di sviluppare velocemente applicazioni database driven, ha portato alla ricerca di una soluzione rapida ed efficiente che fosse focalizzata sui processi e non sulla programmazione. La potenza e versatilità della soluzione adottata ha portato a fornire anche esternamente servizi di sviluppo e software basati su tale tecnologia. Contestualmente a questo sviluppo, l'azienda si è specializzata nella realizzazione di soluzioni di business intelligence attraverso realtà aziendali separate, ma fortemente interconnesse da un punto di vista della visione imprenditoriale. Tepui si è affermata quindi come azienda che concentra queste esperienze e le propone come fornitore unico di consulenza e prodotti proponendo soluzioni a pacchetto o personalizzate. La società opera principalmente a livello nazionale con sedi in Veneto e Lombardia.

Per la realizzazione delle applicazioni, lo strumento che viene adottato maggiormente è InDe. Oltre allo sviluppo di software, l'azienda si occupa anche di data warehousing e business intelligence attraverso principalmente la suite dei prodotti Microsoft e Qlik

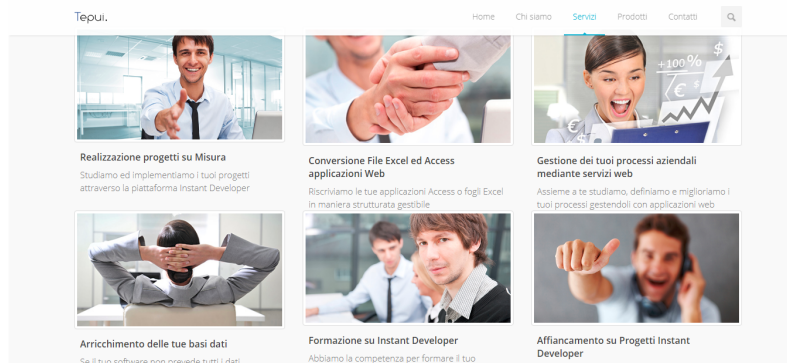
### 1.1.2 Prodotti e servizi

I principali prodotti realizzati sono software e gestione di processi aziendali e di supporto al data warehousing. Essi rappresentano l'insieme dei software che supportano l'automatizzazione dei processi aziendali e di arricchimento dei dati. Si dividono principalmente nei seguenti macro gruppi:

- \* Software di contabilità;
- \* Software per il magazzino;
- \* Software per la produzione;
- \* Software per il budgeting;
- \* Software di gestione ed analisi finanziaria;
- \* Software dedicato;
- \* Software di arricchimento dati.

Tepui S.r.l persegue due diverse soluzioni per la creazione dei prodotti software: a pacchetto e su misura. Le soluzioni a pacchetto consistono in software completi già disponibili all'interno dell'azienda destinati alla vendita. Tuttavia, la loro vendita non è immediata ma segue comunque un controllo e modifica per adattare il prodotto venduto alla realtà del cliente. L'altro tipo di soluzione consiste, invece, nella realizzazione da zero di un prodotto. Questo tipo di servizio prevede tutti i passaggi dallo studio del problema alla realizzazione del prodotto completo.

Infine, per quanto riguarda i servizi, l'azienda fornisce la manutenzione di un qualsiasi prodotto InDe, sia esso creato da Tepui S.r.l o da una qualsiasi altra ditta che faccia affidamento a tale piattaforma, purché si disponga del codice sorgente. Inoltre, tra i servizi offerti troviamo anche: formazione su InDe, affiancamento ai progetti



**Figura 1.2:** Prodotti e servizi forniti nella loro pagina web

con InDe, conversione dei file Excel ed Access in applicazioni web e la possibilità di sfruttare prodotti web per i processi aziendali.

### 1.1.3 Tecnologie di riferimento

#### Linguaggi di programmazione

L'azienda opera nell'ambito web. I prodotti realizzati si basano sui seguenti linguaggi di programmazione lato server:

- \* **C#**: linguaggio di programmazione orientato agli oggetti che consente di creare una vasta gamma di applicazioni protette e affidabili per .NET Framework. Esso può essere adottato per creare applicazioni client Windows, servizi Web XML, componenti distribuiti, applicazioni clientserver, applicazioni di database e molto altro.
- \* **Java**: linguaggio di programmazione ad alto livello, orientato agli oggetti e a tipizzazione statica, che si appoggia sull'omonima piattaforma software, specificamente progettato per essere il più possibile indipendente dalla piattaforma hardware di esecuzione.

Per quanto riguarda la componente grafica, le tecnologie adottate sono: HTML5, CSS3 e Javascript.



**Figura 1.3:** Linguaggi di programmazione

## Database

Tutte le applicazioni dell'azienda mirano alla realizzazione di software gestionale, i quali necessitano di uno o più database, o per realtà più grandi dei Data Warehouse. I principali database che hanno avuto modo di incontrare nello svolgimento delle loro attività sono stati: MySQL, DB2, PostgreSQL, Oracle e SQL Server.

Tra i differenti database disponibili, quello adottato Tepui S.r.l è principalmente SQL Server. La scelta ricade su questo dispositivo perché rappresenta un punto di incontro tra prestazioni, flessibilità e costi.

Altri fattori degni di nota riguardano il suo elevato utilizzo da parte delle aziende del territorio e per la sua popolarità visto che ancora oggi risulta essere il terzo database più usato al mondo dopo Oracle e MySQL.



Figura 1.4: Database

## 1.2 Processi aziendali

Questa sezione presenta l'organizzazione dell'azienda e come quest'ultima cerchi di migliorarsi nel corso del tempo.

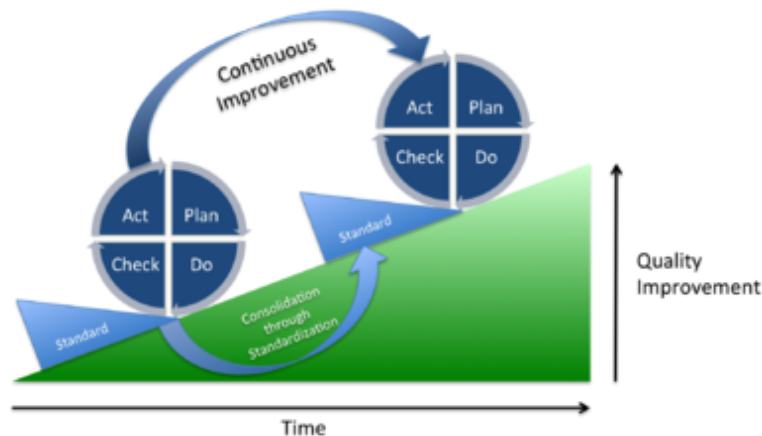
### 1.2.1 Miglioramento della qualità dei processi

Tepui S.r.l. nello svolgimento delle proprie attività opta per perseguire il costante miglioramento dei processi. Per fare questo fa affidamento al ciclo PDCA che si compone di quattro attività:

- \* **Plan:** individuazione degli obiettivi di miglioramento e creazione di un piano d'azione nello svolgimento dei lavori;
- \* **Do:** esecuzione di quanto pianificato;



- \* **Check**: analisi dei risultati ottenuti nella fase precedente e confronto con quanto pianificato;
- \* **Act**: standardizzazione delle attività andate a buon fine e rielaborazione di quelle da migliorare ricominciando con la pianificazione.



**Figura 1.5:** Ciclo PDCA (<https://bit.ly/2y3khfp>)

Durante lo stage, ho notato che l'azienda adotta questa strategia principalmente nel processo di sviluppo mirando ad ottenere un prodotto efficiente ed efficace. Negli altri processi aziendali invece, quali ad esempio la documentazione, spesso viene volontariamente scelto di dargli un'importanza marginale. Questa decisione è legata al software che in alcuni casi si presta alla prototipizzazione rapida (Sezione 1.2.2), discussione con il cliente e successiva revisione.

### 1.2.2 Metodologia agile

L'azienda nello sviluppo delle applicazioni adotta la metodologia agile. Questo metodo operativo permette una maggiore libertà rispetto ad altri tipi quali sequenziale, incrementale o a spirale. I punti fondamentali sono:

- \* privilegiare la realizzazione del software alla creazione di documentazione;
- \* collaborare con il cliente invece di dedicarsi a contrattazioni;
- \* essere pronti a reagire ad ogni situazione invece di avere un piano di gestione dei rischi.

Tepui S.r.l ha deciso di adottare questo metodo lavorativo per i suoi prodotti perché hanno osservato come nella realtà lavorativa le aziende vorrebbero avere a disposizione prodotti efficienti ed efficaci in tempi molto brevi. Inoltre, la scelta è ricaduta su questa tipologia per un motivo molto importante: conoscere i clienti, il mercato e creare un rapporto duraturo di fiducia.

Questo metodo si concretizza con degli incontri la cui scadenza può essere, settimanale, bisettimanale o mensile, con i clienti. Durante gli incontri si raccolgono task, migliorie da apportare ai progetti o addirittura

ci si ferma dal cliente per realizzare nuove funzionalità e chiedere informazioni in maniera immediata. Così facendo la comunicazione è rapida, le mail sono ridotte ed è molto più facile comprendere le necessità delle aziende cliente osservandola dall'intero.

## 1.3 Strumenti a supporto dei processi

Questa sezione illustra gli strumenti adoperati a supporto dei processi e per lo sviluppo mirati a garantire qualità dei prodotti e servizi.

### 1.3.1 Gestione di progetto

La gestione di progetto consiste nel definire ed organizzare il lavoro da svolgere in tempi e modi ben definiti. Per il seguente processo vengono utilizzati tre strumenti: Microsoft Teams, iDo e le mail.

**Microsoft Teams** è un'applicazione di comunicazione unificata multi-piattaforma. Essa permette di creare diversi gruppi con all'interno molti canali di comunicazione ai quali possono accedere solo le persone invitate. L'azienda crea un gruppo per ogni cliente e all'interno prevede un canale generale dove inserire documentazione o fare domande di natura generica. Mentre gli altri riguardano un progetto specifico o le singole funzionalità da implementare, se vi è un unico progetto.

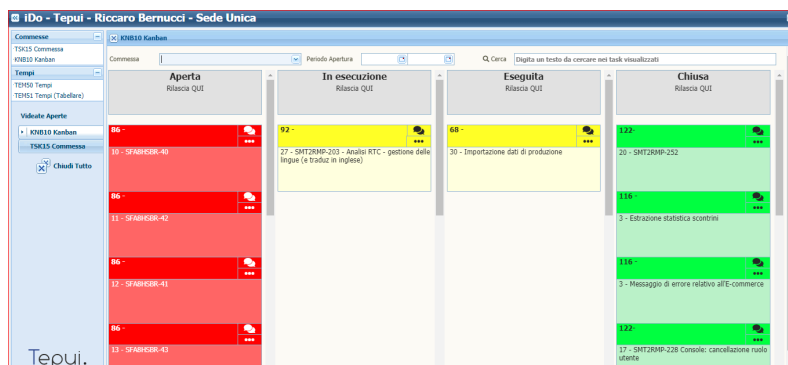


Figura 1.6: Conversazione su Teams

**Mail** ovvero la posta elettronica. Per iniziare il progetto mi è stata fornita una mail aziendale. Le mail servono per comunicare in maniera tempestiva la creazione ed assegnazione di una commessa nell'applicazione iDo. Queste ultime sono il principale mezzo di comunicazione con le aziende clienti. Una prassi interna all'azienda prevede che per informazioni da chiedere al cliente, bisogna prima discuterne internamente tra i membri del gruppo assegnato a quel progetto e poi quella di mettere in copia carbone il responsabile di progetto alla eventuale mail da inviare ai clienti.

**iDo** è un'applicazione web realizzata con InDe dove vengono assegnate le commesse, indicando tempi di inizio e fine previsti. In questo applicativo, si devono indicare le ore svolte dai lavoratori specificando le ore di inizio, fine e informazioni di quanto si è

svolto in quel periodo. Inoltre, si possono inserire commenti utili all'azienda cliente e a Tepui S.r.l. Grazie a questa applicazione viene calcolato il compenso e il consuntivo del progetto. Essa si compone di diverse sezioni la Kanban (figura 1.7), dove vengono presentate tutte le commesse con in testa il nome del cliente e del progetto; una sezione Commessa, nella quale vengono inseriti i progetti assegnati e navigando al suo interno si accede alle commesse; una sezione tempi, dedicata a sistemare eventuali errori di inserimento a fine mese o per controllare le attività svolte dai dipendenti e prendere le opportune decisioni.



**Figura 1.7:** Kanban dell'applicazione iDo (con nome cliente censurato)

### 1.3.2 Documentazione

L'azienda, sebbene abbia deciso di adottare un modello agile, non è priva di documenti. Quando deve realizzare un progetto il primo passo è quello di redigere un Piano di progetto e POC documentale con le principali caratteristiche che il prodotto finale dovrà avere. Il motivo per cui la società dà molta importanza al POC è che, con un documento nel quale è riportato la struttura del database e la grafica pressapochista che il progetto dovrà avere, permette di realizzare un prodotto completo in 3/4 settimane. La documentazione è salvata interamente su OneDrive For Business. Ogni documento ha una sua collocazione da rispettare.

### 1.3.3 Sistema di versionamento

Per il versionamento e il salvataggio dei File prodotti durante la realizzazione dei progetti è previsto l'utilizzo di una repository creata dal Project Manager su un'applicazione web ideata sempre su InDe, TeamWorks. Successivamente, vengono forniti ai dipendenti scelti nella realizzazione di uno specifico progetto i permessi di: scrittura, lettura ed eliminazione. Questa applicazione web risulta essere molto simile a GitHub, tuttavia è molto più intuitivo e semplice perché le funzionalità permesse sono controllare informazioni relative ai commit, tornare indietro di versione (rollback), scaricare progetti (Download) e creare dei derivati (Fork) premendo unicamente dei pulsanti.

Ciascun progetto deve essere soggetto a versionamento perciò chiunque lo utilizza ha una visione chiara e dettagliata della sua storia e delle sue modifiche. Ad ogni task deve corrispondere una versione. Nelle versioni viene applicato il seguente formalismo:

X.Y.Z

Dove X,Y,Z sono numeri incrementali da 0 a infinito. Z indica i singoli task e bug individuati ed risolti, Y rappresenta la implementazione di nuove funzionalità rilasciate per la fase di collaudo e X quando il progetto ha superato il collaudo e diventa operativo.

### Sistema di pubblicazione

La pubblicazione presso Tepui S.r.l corrisponde al rilascio dell'applicazione al cliente per effettuare dei test. Il collaudo viene effettuato anche internamente all'azienda, ma questa doppio controllo permette di realizzare applicazioni corrette che non necessitino di eccessive manutenzioni di tipo correttivo. Il sistema adottato per pubblicare il software è IDManager, anche essa una applicazione web di InDe, la quale permette di modificare i riferimenti del database cambiando la stringa di connessione del progetto e permette di caricare unicamente le differenze tra la versione precedente e quella attuale.

### 1.3.4 Ambiente di sviluppo

**Instant Developer** consiste in una piattaforma ad alta produttività, per lo sviluppo di applicazioni cross-platform (Web-based) creata da Pro Gamma S.p.A.. La scelta dell'azienda ricade su questo tipo di strumento per i seguenti motivi:

- \* Scrivere l'applicazione e poterla distribuire in ambiente Java o Microsoft C#;
- \* Collegare ed utilizzare più database contemporaneamente anche di tipo differente;
- \* Implementare applicazioni Desktop e Mobile;
- \* Gestire i rilasci successivi in maniera sicura e strutturata;
- \* Potersi focalizzare sui processi da gestire, sui i dati da memorizzare o modificare, evitando di dover programmare a basso livello, avendo però la possibilità, quando necessario, di poterlo fare.

Le applicazioni prodotte sono perciò nativamente multi-piattaforma, cross-browser, multi-database già nel momento in cui vengono create.

**Microsoft SQL Server** Un DBMS relazionale, prodotto da Microsoft, che usa T-SQL, una variante del linguaggio SQL Standard.

**Microsoft SQL Server Management Studio** É un'applicazione software che viene utilizzata per la configurazione, la gestione e l'amministrazione di tutti i componenti all'interno di Microsoft SQL Server. Lo strumento include sia editor di script che strumenti grafici che funzionano con oggetti e funzionalità del server.

**Qlik** É un pacchetto che include QlikView, Qlik Sense ed NPrinting. Questi software sono di visualizzazione e business intelligence che permettono il rapido sviluppo di dashboard completamente personalizzabili in grado di fornire rapidamente informazioni utili sui dati a disposizione.

**Microsoft Power BI** È un servizio di analisi aziendale di Microsoft fortemente integrato con Microsoft Office e con gli altri strumenti dell'ecosistema Microsoft. Mira a fornire visualizzazioni interattive e funzionalità di business intelligence con un'interfaccia abbastanza semplice in modo da consentire agli utenti finali di creare i propri report e dashboard.

#### **Altri strumenti**

Oltre agli strumenti appena descritti, eventuali IDE per scrivere in C#, Java e gli altri linguaggi riportati in Sezione 1.1.3, sono lasciati al programmatore. Può capitare che nel corso di un progetto siano richieste modifiche specifiche che l'applicazione InDe non permette, in quei casi vi è una modalità di inserimento personalizzato che consente di scrivere codice.

#### **1.3.5 Sistemi operativi**

L'azienda usa solo i sistemi operativi di Windows. Questo perché risultano essere gli unici compatibili con InDe. Per chi non dovesse avere a disposizione tale sistema operativo viene fornita una macchina virtuale alla quale collegarsi.

#### **VPN e desktop remoto**

In base al progetto, spesso può capitare che ci si debba affidare alla macchina virtuale del cliente. In queste occasioni l'azienda consiglia l'utilizzo di una delle seguenti applicazioni per connettersi alla VPN: openVPN, FortiClient o lo strumento di Windows. Mentre per entrare in desktop remoto: Connessione Desktop Remoto di windows oppure nRemoteNG, il quale offre anche la possibilità di creare una o più connessioni VPN ed aprire diversi schermi remoti contemporaneamente. Quando invece si effettuano delle assistenze, AnyDesk o TeamViewer sono dei software efficienti per collegarsi al desktop del cliente e risolvere problemi.

### **1.4 Clientela**

I clienti di Tepui S.r.l risiedono nei territori del nord Italia. La sede legale dell'azienda si trova a Milano. A Castelfranco veneto è collocata una delle due sedi operative, presso la quale ho svolto la mia esperienza di stage.

I clienti sono imprese di medio-grandi dimensioni. Altre imprese degne di nota sono Aton s.p.a, Sistemi s.p.a, WiseEnergy Italia s.r.l, Geox s.p.a. ed altre aziende che operano a livello internazionale.

In seguito alla compilazione dell'[Accordo di non divulgazione](#) per l'intera durata del progetto non verrà nominato il nome dell'azienda cliente.



## Capitolo 2

# Stage

*Questo capitolo spiega il motivo per cui l'azienda ha deciso di pendere uno stagista e l'utilità che potrà avere nella realizzazione del progetto. Inoltre, vengono presentati i vincoli imposti in sede di pianificazione dello stage e gli obiettivi che ci si aspetta vengano realizzati.*

### 2.1 Vantaggi dell'azienda

Tepui trae diversi vantaggi dall'attività di stage curricolare organizzata presso la sede di Castelfranco Veneto.

In primo luogo, l'aumento della forza lavoro. L'introduzione di un nuovo membro nel team di sviluppo ha permesso all'azienda di redistribuire il carico di lavoro in modo da implementare altri progetti in cantiere e di incrementare i servizi di consulenza. Inoltre, il punto di vista proveniente da un utente esterno ha permesso all'azienda di individuare nuove funzionalità di Instant Developer rilasciate dalla casa produttrice. Un esempio di nuova funzionalità è stata la realizzazione del caricamento immagini senza l'ausilio di informazioni di header in fase di upload e la possibilità di caricare i file in una cartella specifica temporanea modificando dei comandi preimpostati dall'applicazione.

In secondo luogo, lo stage ha permesso all'azienda di apprendere un metodo di implementazione del codice ordinato attraverso la catalogazione offerta da InDe che dà la possibilità di includere parti codice in cartelle e sottocartelle scrivendo a commento la loro funzionalità in modo che il codice sia più facile da capire.

In terzo luogo, il costo di uno stagista è stato minimale ed ha permesso di esplorare nuove funzionalità risparmiando. Trattandosi di un'azienda giovane, poter conoscere e migliorare la propria operatività a prezzi irrisori è considerato un'ottima occasione.

### 2.2 Presentazione del progetto

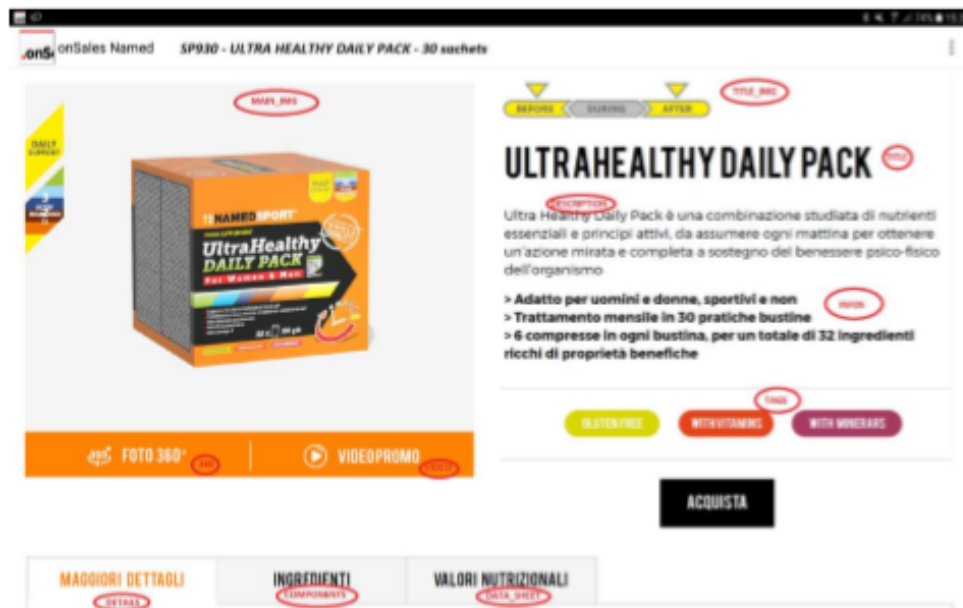
In questa sezione verranno esposte le informazioni di base relative al progetto da realizzare. Entreremo nel dettaglio del progetto nel capitolo 3.

Il progetto da realizzare rappresenta una applicazione web che permetta ad una azienda cliente di inserire e mantenere un catalogo prodotti lato back-end, ovvero inserire

informazioni che verranno utilizzate dalla pagina web alla quale accedono i consumatori B2B. Il progetto in questione quindi mira a:

- \* semplificare l'inserimento a database dei dati che verranno utilizzati da un'altra applicazione;
- \* - Arricchire dati provenienti da un sistema terzo.

Il progetto si compone di un insieme di schermate che devono offrire la possibilità di gestire tutti i prodotti di un catalogo.



**Figura 2.1:** Front-end realizzato con i dati inseriti dal configuratore

### Visualizzazione prodotti

Nella schermata di "Visualizzazione prodotti" si devono presentare i singoli articoli e alcune delle informazioni più importanti tra cui il codice identificativo, nome breve del prodotto e a quali famiglie di prodotti esso appartiene. In questa videata è richiesta la presenza delle seguenti operazioni: ricerca, cancellazione, inserimento e dettaglio. Per quanto riguarda l'inserimento esso può avvenire direttamente nell'elenco prodotti indicando gruppo di appartenenza, codice e nome breve. Entrando nel dettaglio invece, è richiesta l'apertura di una seconda schermata riempita con i contenuti del prodotto presenti nel database, quindi gestire le informazioni che si desiderano presentare nella pagina web di front-end. Un'altra operazione a cui bisogna prestare attenzione è la cancellazione di un prodotto. Il cliente ha chiesto espressamente che questa cancellazione sia unicamente logica per quanto riguarda il prodotto fatta eccezione alle informazioni interne che potranno essere cancellate fisicamente: tag, specifiche tecniche del prodotto, immagini.



### 2.2.1 Visualizzazione dettaglio prodotto

La "Visualizzazione dettaglio prodotto" rappresenta il centro del progetto. In questa schermata devono essere implementate diverse funzionalità che permettono di gestire al meglio l'inserimento e l'aggiornamento delle informazioni di un articolo. Fondamentale è nella schermata avere un'unico metodo di salvataggio e dei controlli specifici che creino unicamente i record necessari. Le informazioni che bisogna essere in grado di gestire sono:

- \* Nome esteso, descrizione ed informazioni;
- \* Inserimento e gestione tag;
- \* Inserimento e gestione immagine principale;
- \* Inserimento e gestione video;
- \* Inserimento e gestione Titolo (immagine);
- \* Inserimento e gestione immagine/i 360, ovvero la creazione di una galleria di immagini da mostrare;
- \* Informazioni aggiuntive che possono essere create, modificate e cancellate da una schermata a parte.

### 2.2.2 Inserimento attraverso modali

Gli inserimenti delle immagini, video o delle informazioni aggiuntive deve avvenire attraverso delle modali. Le modali richieste prevedono l'inserimento di queste informazioni: Numeri, Liste valori, Stringhe, Date, Booleani, Multi-selezione, URL, Email, Telefono, File, Immagine, Multi-File, Immagini 360.

### 2.2.3 Gestione delle configurazioni

Le informazioni che devono poter essere aggiunte ad un prodotto hanno dei record preimpostati non cancellabili. Inoltre con questa schermata nasce l'intento di poter creare tipi di informazioni aggiuntive a discrezione dell'utente finale. Nell'inserimento dell'informazione da gestire è necessario limitare l'utente con i tipi indicati nelle modali.

### 2.2.4 Gestione delle traduzioni

Al termine del progetto è richiesto di individuare un metodo adeguato per la gestione delle traduzioni dei prodotti in lingue differenti dall'italiano. Studio dell'eventuale utilizzo di una componente RTC presente su InDe con una licenza da acquistare.

### 2.2.5 Altri interventi

Infine, oltre alla realizzazione del progetto, l'azienda per rendere lo stage vicino alla realtà che deve affrontare ogni giorno ha chiesto di essere disponibile e pronto a fornire supporto in questioni esterne al progetto che possono verificarsi.

## 2.2.6 Obiettivi

Gli obiettivi concordati nel piano di lavoro sono stati suddivisi in tre categorie: obbligatori, desiderabili e facoltativi. L'azienda ha espresso la richiesta che gli obbligatori siano completati, mentre per i desiderabili, almeno due dei tre indicati, siano portati a termine.

Gli obiettivi si distinguono in:

\* Obbligatori

- O01: Apprendimento della piattaforma Instant Developer;
- O02: Test delle funzionalità implementate e rilascio;
- O03: Utilizzo di Microsoft SQL Server.

\* Desiderabili

- D01: Gestione di progetto;
- D02: Comunicazione con il cliente;
- D03: Scrittura delle procedure T-SQL.

\* Facoltativi

- F01: Autonomia a risolvere nuove problematiche.

## 2.3 I vincoli

### 2.3.1 Vincoli temporali

Lo stage ha una durata prevista di 310 ore complessive, distribuite in 8 settimane da 40 ore lavorative ciascuna, dal 13 Maggio 2019 al 8 Luglio 2019. L'orario di lavoro concordato con il tutor aziendale è stato dal Lunedì al Venerdì dalle 09:00 alle 18:00, con 1 ora di pausa pranzo. Prima dell'inizio dello stage è stato redatto un piano di lavoro con una scansione temporale delle attività con granularità settimanale così definita:

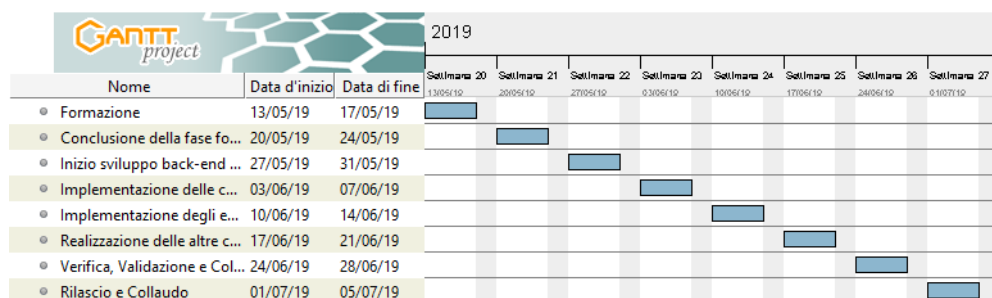


Figura 2.2: Diagramma di Gantt

**Prima Settimana - Formazione (40 ore)**

La prima settimana prevede l'apprendimento del programma Instant Developer seguendo un corso online composto da video realizzati dai produttori del programma. Inoltre, per apprendere al meglio il prodotto, è richiesto che nel seguire il video si svolgano i progetti commissionati dal corso stesso integrandoli con richieste dell'azienda per velocizzarne l'apprendimento. Durante la settimana sono entrato in contatto con gli altri membri del team di sviluppo e ho compreso le dinamiche aziendali.

**Seconda Settimana - Conclusione della fase formativa ed inizio gestione di progetto (40 ore)**

Durante la seconda settimana è stato previsto uno studio di SQL Server e delle stored procedure presenti nei database dell'azienda in modo da comprendere alcuni degli standard e saperli poi applicare in caso di necessità. Inoltre si inizia con la lettura delle specifiche del progetto che si dovrà realizzare e a seguito di una attenta analisi interna ci saranno degli incontri con il cliente per approfondire alcune caratteristiche su cui si desidera maggiore chiarezza.

Infine al termine della settimana ci si aspetta la redazione di un documento che riporti il metodo formativo dell'azienda se è stato sufficiente e dove si ritiene necessario maggior impegno in modo che tali documenti fungano da base per un loro miglioramento.

**Terza Settimana - Inizio sviluppo back-end della componente (40 ore)**

La terza settimana si entra nella progettazione che prevede una prima comprensione del problema e realizzazione di una bozza delle tabelle da creare al fine di non compromettere il sistema preesistente e integrando quello nuovo. Una volta individuate le tabelle necessarie dovranno essere implementate e in questa settimana si inizieranno a realizzare i primi documenti relativi a tempistiche e un primo aspetto grafico completo che il progetto dovrà avere.

**Quarta Settimana - Implementazione delle componenti di base del progetto (40 ore)**

Nella quarta settimana è stata prevista l'implementazione delle classi basate sulle tabelle ideate, le schermate ad esse associate e un giorno di riepilogo presso la sede del cliente dello stato di avanzamento del lavoro. Inoltre, internamente, hanno chiesto che si rediga un documento con le scelte implementative adottate per riutilizzare tale documentazione in caso di realizzazione di progetti simili o per sapere quali sono state le motivazioni che hanno portato ad una scelta implementativa rispetto ad un'altra.

**Quinta Settimana - Implementazione degli eventi del progetto (40 ore)**

Nella quinta settimana è stato previsto un lavoro intenso nell'implementazione degli eventi di caricamento e salvataggio dei dati, mantenendo una costante redazione ed aggiornamento dei documenti relativi al progetto.

**Sesta Settimana - Realizzazione delle altre componenti del progetto (40 ore)**

Durante la sesta settimana ci si aspetta che il prodotto sia sufficientemente sviluppato almeno per le componenti principali e quindi si dovranno effettuare i primi test di unità.

Se il progetto è sufficientemente stabile inoltre sarà possibile passare alla creazione degli altri oggetti e delle altre schermate da realizzare, in particolare le schermate di upload delle immagini e dei file. Nella settimana è previsto un incontro presso la sede del cliente in modo da visionare lo stato di avanzamento del prodotto commissionato.

### **Settima Settimana - Verifica, Validazione e Collaudo (40 ore)**

La settima settimana si presuppone che il progetto sia completamente realizzato e quindi si ricontrollino i documenti interni in modo che siano precisi al dettaglio per ogni singola schermata, classe e metodo implementato. Poi si prevede l'integrazione del prodotto nel sistema preesistente con relativo collaudo di ogni singola funzionalità e quindi sistemazione degli eventuali bug del software.

### **Ottava Settimana - Rilascio e Collaudo (30 ore)**

L'ultima settimana si dedica al collaudo del software, alla pubblicazione di quest'ultimo al cliente perché possa testarlo e fornire eventuali feedback. Infine, sono previste in questa settimana la gestione di richieste particolari del cliente, come: miglioramento grafico ed eliminazione di funzionalità prima ritenute importanti.

## **2.3.2 Vincoli metodologici**

In accordo con il tutor aziendale, lo stage si è svolto presso la sede dell'azienda. Questa decisione si deve principalmente a due motivi:

- \* Agevolare la comprensione, da parte dello stagista, delle dinamiche aziendali e l'interazione con il proponente del progetto;
- \* Favorire il confronto tra stagista, team e tutor aziendale.

A seguito dei servizi di consulenza offerti dall'azienda, nella seconda metà dello stage, la comunicazione con il tutor è stata meno costante ed in previsione di ciò si è adottata la politica di individuazione ad inizio settimana dei task da sviluppare e ad ogni problema o implementazione completata una comunicazione nel canale Teams predisposto.

Per l'intera durata dello stage, Tepui S.r.l ha richiesto che lo stagista redigesse delle brevi relazioni, descrivendo le problematiche affrontate, le scelte adoperate e il risultato ottenuto. Questi documenti hanno la funzione di materiale ausiliario dedicato al miglioramento della gestione degli stagisti.

Infine, è stato posto come obbligo che tutta la documentazione rimanesse in una cartella OneDrive e che ogni singola operazione svolta venisse indicata su iDo, la piattaforma con la quale riescono ad indicare le ore a consuntivo svolte per la realizzazione di ogni modulo.

## **2.3.3 Vincoli tecnologici**

Nella realizzazione del progetto l'azienda ha chiesto che si adottassero i concetti base della programmazione ad oggetti. In Instant Developer questo prevede che per ogni tabella del database debba essere creata una classe. Quindi si implementano solo ed unicamente metodi, che nel programma si distinguono in eventi e procedure, necessari.

Instant Developer presenta due diverse modalità di creazione delle pagine web: Table Oriented (TO) e Document Oriented (DO). Nel mio caso la richiesta dell'azienda

prevedeva l'utilizzo della seconda modalità applicando i concetti appresi nella programmazione ad oggetti. La programmazione DO corrisponde alla Object Oriented Programming.

La document orientation (DO) si basa su un framework ORM (Object-relational Mapping) di classe enterprise, il cui scopo è, oltre quello di automatizzare la serializzazione ed il caricamento degli oggetti dal database, quello di gestirne l'intero ciclo di vita e le relazioni con gli altri componenti dell'applicazione. Le caratteristiche enterprise della DO sono arricchite dalla presenza di un sistema estensibile di servizi ai documenti, che attraverso tecniche OOP permette di implementare caratteristiche comuni e perpendicolari alla gerarchia delle classi.

Con InDe è possibile creare manualmente le classi e poi aggiungere le proprietà e i metodi, oppure affidarsi ad un metodo più semplice, ovvero quello di effettuare un D&D della tabella del database che conterrà i dati del documento sull'applicazione, tenendo premuti i tasti shift e ctrl, generando automaticamente le classi.

Questo comportamento dell'applicazione va contro i principi dell'ingegneria del software, la quale prima prevede si progetti lo schema delle classi e poi questo lo si traduca in uno schema di database. Il metodo proposto non presuppone una maggiore importanza del database rispetto alle classi, ma è stato pensato per ottenere i seguenti vantaggi:

- \* il database è già presente; con l'importazione ottengo lo schema del database e genero classi in maniera automatica;
- \* se il database è nuovo, la sua definizione dovrebbe seguire gli stessi principi della creazione dello schema delle classi; quindi partire da un punto o dall'altro è indifferente;
- \* dal punto di vista del framework DO, esso utilizza anche alcune informazioni presenti all'interno della definizione del database, come ad esempio la struttura delle foreign key.

## 2.4 Scelta e obiettivi personali

Sono entrato in contatto con l'azienda ospitante grazie ad un amico che mi ha messo in contatto con i responsabili. Dopo un colloquio ed una spiegazione generale delle attività svolte dall'azienda, hanno suscitato il mio interesse. L'idea di interfacciarmi con il mondo del lavoro prendendo in mano la gestione di dati sensibili e la possibilità di creare un gestionale rientra perfettamente nell'impiego da me cercato.

Dopo aver studiato economia presso l'Istituto Tecnico Commerciale Statale P.F. Calvi ed informatica presso l'Università di Padova, entrare in una realtà lavorativa che concilia i due ambiti, mi sembra un buon completamento dei miei studi fino a questo momento.

Gli obiettivi che mi sono posto di raggiungere a livello personale oltre a quelli concordati con l'azienda sono:

- \* Accrescere le conoscenze in merito al mondo RAD e Data Warehouse;
- \* Migliorare le capacità di realizzazione di applicazioni seguendo il metodo Bottom-Up;
- \* Apprendere come interfacciarmi con i clienti;
- \* Migliorare le mie capacità di Problem Solving.



## Capitolo 3

# Progetto e attività di stage

*Questa capitolo parla delle attività di stage andando a descrivere il progetto nei suoi stati di avanzamento, le scelte adottate e le motivazioni che mi hanno permesso di scegliere.*

### 3.1 Formazione

All'inizio dello stage, la prima attività che ho dovuto svolgere è stata quella di studio delle applicazioni adottate dall'azienda e i formalismi, ovvero codici e abbreviazioni adottate per la realizzazione dei campi di database. Per la formazione sull'applicazione Instant Developer ho seguito sei video-corsi della durata di una o anche due ore l'uno. In aggiunta, per velocizzare l'apprendimento dell'uso dell'applicazione, è stato richiesto che venissero realizzate le applicazioni presenti nei video. Rispetto a quanto pianificato, l'apprendimento del programma è risultato più rapido della settimana preventivata e quindi, durante la stessa, ho potuto interfacciarmi con i data warehouse e lo studio di alcune stored procedure. Verso fine settimana ho iniziato lo studio dei documenti relativi al configuratore catalogo/prodotti.

### 3.2 Progettazione

Dalla seconda settimana di stage, ho iniziato ad affrontare le dinamiche aziendali concentrandomi nelle attività di progetto che è stata divisa in incontri con il cliente e lavoro presso la sede di Castelfranco.

#### 3.2.1 Database

Affiancato al tutor aziendale, il primo punto affrontato è stato il database. Il cliente ci ha dato carta bianca, ovvero potevamo decidere tra due soluzioni: riutilizzare tabelle presenti nel database oppure crearne di nuove. Questo cliente specifico dispone tra i suoi software di sei applicazioni ideate con InDe e nei mesi di stage hanno iniziato a chiedere nuovi progetti.

La scelta, dopo una attenta discussione con il cliente, è stata quella di creare delle tabelle nuove perché questa nuova componente in futuro vorrebbero integrarla in altre applicazioni oltre che a quella per cui è stata realizzata.

Per cercare di mantenere una certa uniformità con il resto del data warehouse, la

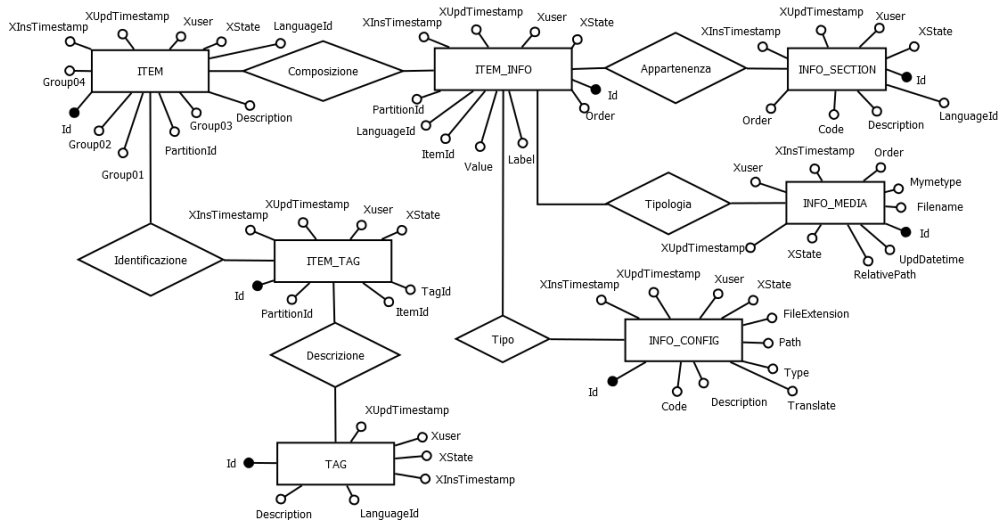
prima scelta è stata quella di non inserire vincoli di Foreign Key. Questo vincolo che avrebbe permesso query più rapide è risultato superfluo in seguito alla velocità dei server. Tuttavia, continuo a ritenere molto utile la creazione delle foreign key perché in questo modo se anche le tabelle iniziano ad avere una quantità di record spropositata la velocità resta soddisfacente. Inoltre, il problema dell'eventuale plugin dell'applicazione richiede comunque che si metta mano al codice nell'importazione della componente. Quest'ultima scelta ha portato il vantaggio di estendere i progetti in maniera più rapida. Infine, per concludere il tema vincoli gli unici adottati sono quelli di Primary Key.

La prima tabella da cui partire è quella dedicata agli articoli. Quest'ultima è già presente nel data warehouse e dispone di una vista. Partendo da questi due contenitori, si è realizzata una struttura che permetta prima di realizzare una lista prodotti, poi, se l'articolo esiste, arricchire le sue informazioni.

Gli aspetti da tenere in considerazione sono stati:

- \* ogni articolo ha delle immagini, video o dei file di vario genere;
- \* ogni articolo deve poter avere uno o più tag;
- \* ogni articolo deve avere delle informazioni di base e delle informazioni aggiuntive;
- \* ogni informazione aggiuntiva deve poter essere gestita (creata, modificata ed eliminata).

Queste informazioni sono state definite durante la prima riunione presso la sede del cliente.



**Figura 3.1:** Diagramma Entity Relationship

Il diagramma ER (i vincoli sono fittizi, mostrano i collegamenti nonostante l'assenza di foreign key) è stato pensato in modo tale che ogni articolo dispone di molte informazioni e ciascuna di esse appartiene ad una categoria.



**ITEM** Questo oggetto è una vista con alcune delle informazioni più importanti. Visto che la fase di inserimento prevede unicamente che un'utente possa inserire alcune informazioni. I dati gestiti nella vista sono i gruppi di appartenenza (ad esempio insaccati, surgelati, prodotti da forno) ed una descrizione breve dell'oggetto. In questo contesto la descrizione appare due volte una negli ITEM ed una nella ITEM\_INFO per il semplice motivo che la tabella esisteva già nel database.

**ITEM\_INFO** Nella seguente tabella vengono raccolte tutte le informazioni e rappresenta il punto di collegamento dell'intero progetto. In esso vengono inserite le etichette e i valori che rappresentano i punti fondamentali di questa tabella. Le etichette (label) sono la descrizione dei valori (value) che permettono ricerche in tabella più rapide (video, titolo, immagine). I valori sono quelli che in base alle altre tabelle saranno stampati a video dall'applicazione web di front-end (figura ??).

**INFO\_CONFIG** La tabella INFO\_CONFIG è stata realizzata al fine di inserire record con specifiche che limitano i tipi di record che possono essere gestiti all'interno di ITEM\_INFO. Inoltre definiscono le caratteristiche su cui sono fondati i record della tabella associata ad esempio è possibile definire le estensioni di file o un parametro che mi hanno chiesto di gestire in un secondo momento, ovvero la lingua; nel diagramma risulta già presente, ma in realtà è stato aggiunto in un secondo momento verso fine progetto circa a metà della penultima settimana.

**INFO\_MEDIA** INFO\_MEDIA è la tabella, come è comprensibile dal nome, che detiene tutte le informazioni relative ai file. In questo caso una prima idea è stata quella di salvare il file nel database con un blob, tuttavia ho pensato che poi nell'interrogazione di una tabella contenente molti media poteva diventare molto pesante considerando la richiesta di gestione dei video. In questa occasione ho pensato fosse più propizio ideare una cartella online dove è installata l'applicazione con un GUID (nome univoco) assegnato al file e nel database disporre unicamente del percorso relativo dell'immagine.

**INFO\_SECTION** Le informazioni relative ad un prodotto devono essere categorizzate. Questa tabella ha la funzione di indicare la categoria di appartenenza così come avviene in diversi siti e-commerce. L'idea è di poter gestire le informazioni presentandole in sezioni dinamiche tante quante si ritengano necessarie.

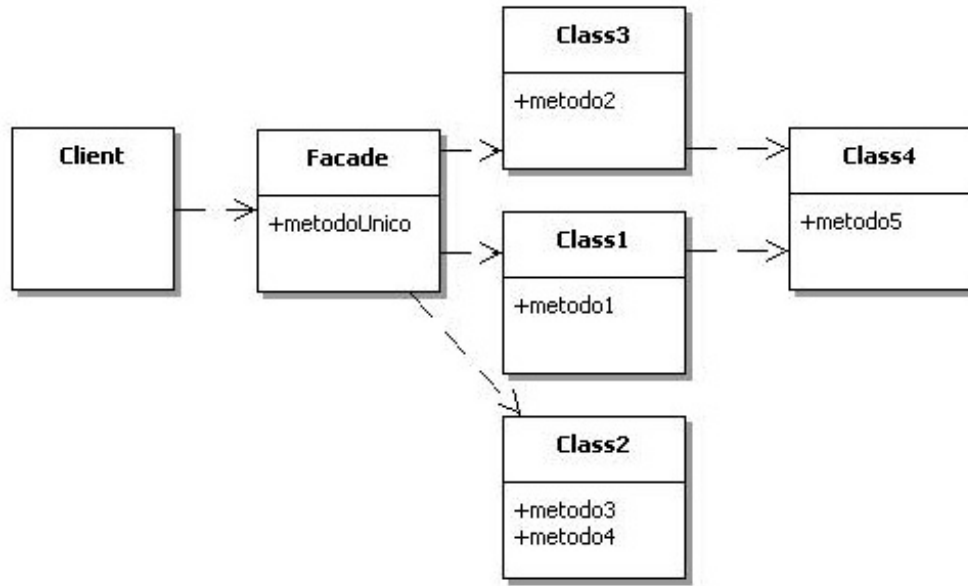
**TAG** Tabella che serve ad includere ogni tipo di parole chiavi che permettano di effettuare ricerche per arrivare ad uno specifico prodotto o gruppo di prodotti. Questa tabella non è stata creata perché è adoperata in altri contesti.

**ITEM\_TAG** Quest'ultima tabella ha lo scopo di collegare i tag con lo specifico articolo. Si è di fronte ad una relazione uno a molti che, in fase di ristrutturazione, ha comportato la creazione di una tabella intermedia tra articoli e tag.

### 3.2.2 Design Pattern

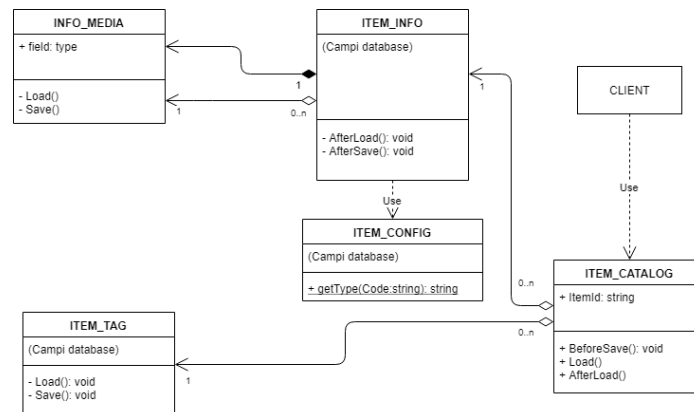
Nella realizzazione dell'applicazione per la maggior parte delle videate un singolo oggetto doveva gestire le sue operazioni nella propria videata. Vi è stato solo un caso particolare che ha richiesto l'uso di un design pattern e che già nella fase di progettazione del database è stato abbozzato: nella gestione di dettaglio del singolo

articolo. Il pattern scelto è il Facade. Tutte le informazioni partono da un articolo (record della tabella ITEM) e da questo vengono generate opportune operazioni.



**Figura 3.2:** Design pattern strutturale: Facade (<https://bit.ly/2McTHsR>)

Il grafico UML generato una volta definito il design pattern da applicare è risultato essere quello sottostante. Per realizzarlo, ho seguito la logica che doveva avere l'applicazione ed ho provato a segnare le attività di caricamento, creazione, modifica e cancellazione dei record delle diverse tabelle.



**Figura 3.3:** Diagramma UML delle classi

**Caricamento** Un dettaglio articolo non è altro che un'insieme di informazioni (ITEM\_INFO). Partendo da questo presupposto ho deciso di caricare un array di oggetti (in InDe prende il nome di IDCollection). Per definire quale articolo caricare indico

uso l'id dell'articolo; quindi carico le varie informazioni tramite query. Quando ricevo una informazione verifico immediatamente a quale configurazione (INFO\_CONFIG) appartiene e solo se necessario carico gli ulteriori dati, ovvero i media. Infine, con la classe ITEM\_TAG metto in un array tutti i tag associati all'articolo selezionato.

**Creazione** Questa operazione è molto più semplice e controllata. Prima creo un articolo (ITEM), quindi entro nel dettaglio e creo tanti ITEM\_INFO con o senza il proprio ITEM\_MEDIA. Poi se vengono inseriti creo gli ITEM\_TAG. Tutto questo viene salvato unicamente dall'oggetto ITEM\_CATALOG che fa partire gli eventi di save dei vari oggetti.

**Modifica** In questo caso la modifica è uguale all'inserimento. La differenza è che in inserimento all'inizio devo creare un articolo e poi aprire il suo dettaglio senza caricare informazioni. In questa situazione, l'oggetto deve caricare ogni singola informazione quindi è necessario un controllo sulle query se ritornano o meno informazioni. In seguito, cambiando un dato questo segnala all'oggetto, a volte forzando l'update, che devono essere aggiornati i dati.

**Cancellazione** L'ultima operazione non meno importante è la cancellazione. Questa deve generare la cancellazione di ogni singolo record di uno specifico articolo senza lasciare record "orfani". Per evitare questo fenomeno, quando cancello inizio dai figli fino al raggiungimento del padre ed infine blocco la cancellazione dell'articolo (ITEM), ma cambio lo stato di un campo del record da "I" (Insert) o "M"(Modified) a "D" (Deleted) ed aggiorno XUpdDatetime.

**TO DO** diagrammi di sequenza per le 4 operazioni

### 3.3 Codifica

Dopo la fase di progettazione, si è affrontata la codifica. Questa fase è stata differente rispetto a quanto ho sviluppato durante il corso di Programmazione ad oggetti, Programmazione concorrente e distribuita ed Ingegneria del Software. Il motivo della diversità è l'applicazione Instant Developer.

Progettare classi e videate con questo software mi ha abbastanza cambiato il modo di implementare e testare. Prima di questo cambiamento tendevo a crearmi una scaletta, implementarla e passo per passo effettuare dei test di unità basati su codice scritto (ad esempio Java associato a JUnit).

In questo contesto, invece, mi sono ritrovato a saltare i test di unità e passare immediatamente a test di sistema. Di conseguenza ho creato classi, metodi ed eventi cercando quanto più possibile di seguire una logica rigorosamente commentata. Mi attendevo molti errori, ma sono rimasto sorpreso. Il numero di errori, per il fatto che il codice generato si basa su template ben controllato e già testato dalla casa produttrice, era molto basso e, con il debugger incluso, ho potuto risolvere ogni errore commesso in poco tempo per poi passare ad altre funzionalità.

#### 3.3.1 Documenti

Instant Developer definisce con il termine documenti le classi java o c# che dovranno essere generate e da cui si basano le applicazioni ideate. Le prime classi che ho creato sono state quelle automatiche, secondo il ragionamento del framework ORM

di InDe. Ho preso le tabelle del database e implementato una classe per ciascuna. Il funzionamento di alcune delle classi è rimasto quello standard. La classe che ha richiesto un ragionamento maggiore è stata quella per la gestione del dettaglio articoli. La creazione automatica genera delle classi con tanti attributi quanti sono quelli presenti nelle tabelle di riferimento e con essi sono creati anche i metodi get e set, rispettivamente dedicati a mostrare e modificare i valori di un record.

A seguito dell'utilizzo dell'ORM, le classi vengono generate sulla base di un template. Tutte le classi basate su un database estendono una classe IDDocument la quale contiene ogni metodo necessario ed evento necessario da gestire. Nelle classi derivate, create a partire da una tabella, è possibile intravedere nel codice come solo i metodi ed eventi di cui si hanno la necessità sono valorizzati e ridefiniti, mentre gli altri metodi previsti dal programma sono solo dichiarati e riprendono il funzionamento della classe padre.

### ITEM

La classe dedicata all'articolo è rimasta pressoché invariata. È stato creato un campo decodifica gruppi al fine di mostrarlo a video e popolarlo solo in caso di esigenza e si è rivista la query da cui vengono estrapolati i dati da assegnare ad un oggetto.

Metodo e Query	Descrizione
BeforeSave()	Questo metodo rientra tra quelli generati automaticamente dal programma. Al suo normale funzionamento ho aggiunto alcune informazioni da salvare in caso di inserimento, modifica o cancellazione (XState, XInsTimestamp, XUpdTimestamp). Nella cancellazione ho bloccato l'operazione di "delete" trasformandola in una "update" in cui il valore XState passa a "D".
DecodificaGruppi()	Il metodo di decodifica gruppi è stato realizzato al fine di caricare in un campo di tipo string l'insieme dei gruppi di appartenenza dell'Articolo decodificando il codice nella descrizione corrispondente.
Master Query	Le master query sono le query di base che vengono lanciate per popolare gli attributi di una classe. In questo caso la modifica è stata quella di limitare la visibilità ai soli record il cui campo XState non fosse "D", ovvero cancellato.

### ITEM\_INFO

Per quanto riguarda le informazioni, mi sono dedicato molto al comportamento che l'oggetto dovrebbe avere durante l'esecuzione dell'applicazione web. Dopo la lettura e salvataggio dei dati nel database.

Metodo e Query	Descrizione
AfterSave()	Dopo aver salvato un oggetto di tipo ITEM_INFO nel database, viene azionato questo metodo che ho utilizzato per controllare se l'informazione salvata è un'immagine e in quel caso procedere all'inserimento di alcuni valori nell'oggetto Media e salvarlo.
AfterLoad()	Come per la AfterSave(), dopo aver caricato la mia informazione verifico di che tipo è e se si tratta di una immagine la carico nell'oggetto Media interno alla classe.

### ITEM\_CONFIG

La ITEM\_CONFIG è stata tra le prime classi ad essere realizzata. Sulla base di questa sono gestiti alcuni eventi dell'ITEM\_INFO e dell'ITEM\_CATALOG. È stata studiata per contenere le configurazioni e ottenere in maniera rapida delle informazioni da altri oggetti. Per fare ciò ho creato metodi statici che effettuano query e ritornano una specifica informazione.

Metodo e Query	Descrizione
BeforeSave()	Come per l'articolo anche in questo caso la cancellazione è stata gestita a livello logico.
GetFileExtension(string code)	Metodo statico che restituisce un valore booleano in base al tipo di estensione se accettata o meno.
GetPath(string code)	Metodo statico che restituisce il percorso dove salvare i tipo (code) di file.
HasModal(string code)	Metodo che mi permette di controllare se questo tipo di ITEM_CONFIG deve essere o meno gestito con l'uso delle modali.

### INFO\_MEDIA

La classe dedicata alla gestione dei media ha richiesto molto tempo più che al caricamento dei file al suo salvataggio. In fase di inserimento record nel database ho anche dovuto trovare un modo per salvare le immagini in una cartella dell'applicazione web con un nome univoco.

Metodo e Query	Descrizione
OnEndTransaction()	Metodo che viene richiamato ogni qual volta si crea o modifica un singolo campo del database che generalmente serve a popolare alcuni campi obbligatori ma non visibili. In questo caso è utilizzata per generare in automatico un path assoluto per permettere il caricamento dell'immagine.
BeforeSave()	Prima di salvare il record controllo se esiste già il file nella directory di destinazione e verifico che non sia uguale all'immagine che ho caricato in modo da ridurre il i byte da trasferire ai soli necessari.
AfterLoad()	Carico il percorso assoluto dell'immagine e verifico se è presente una immagine in quel percorso. La assegno ad una variabile che a video verrà immediatamente esposta.
GetMediaPath()	Metodo per ottenere il path dell'immagine.
ValorizzaImagePath(string TempPath)	Metodo dedicato per creare una immagine temporanea nella cartella temporanea del programma che appena viene chiuso vengono cancellati automaticamente.

### INFO\_SECTION e TAG

Le seguenti due classe hanno mantenuto la gestione automatica generata da InDe. Tuttavia, anticipando quanto richiesto in un secondo momento, esposto nella sezione 3.6, la classe INFO\_SECTION ha subito un cambiamento. Il cambiamento in questione

è relativo alla gestione della lingua, che come in altre classi, ha introdotto dei nuovi controlli.

### ITEM\_TAG

La classe è rimasta pressoché invariata da quella auto-generata nella gestione degli eventi fatta eccezione per l'inserimento e la modifica che ha richiesto solo l'arricchimento di dati di controllo.

Metodo e Query	Descrizione
BeforeSave	Gestisco i campi XInsTimestamp, XUpdTimestamp e XState.
ExistsInCollection(IDCollection coll)	Metodo pubblico che controlla se un elemento è presente nella IDCollection.

### ITEM\_CATALOG

Questa classe funge da tramite per tutte le operazioni reattive al dettaglio di un prodotto. Essa non è basata su alcuna tabella. Nel caricamento delle informazioni vi sono specifiche query ideate per caricare unicamente la combinazione dei dati di un articolo. Si compone di due IDCollection una di ITEM\_INFO e una di ITEM\_TAG. La prima la utilizza per salvare le informazioni che vengono passate, mentre la seconda salva i tag.

Metodo e Query	Descrizione
BeforeSave()	Questo evento ha in sé tutto il codice relativo all'inserimento, modifica e cancellazione dei dettagli articolo. Esso prende le informazioni passate dalle interfacce e richiama il salvataggio dei singoli ITEM_INFO ed ITEM_TAG indicando eventuali logiche a supporto.
BeforeLoad()	L'oggetto non si basa su una tabella o query. Esso fa riferimento a due collezioni di oggetti. Per caricare le due collezioni ho inserito dei campi obbligatori: ItemId, per riconoscere l'articolo, combinato al partitionId, anche se ad oggi non è mai stato valorizzato.

### 3.3.2 Videate

La creazione delle interfacce utente, o come le chiamano su Instant Developer "Videate", ha richiesto un lavoro di "drug and drop". Quindi l'aspetto grafico è stato molto semplice e rapido, fatta eccezione per l'aspetto responsive che ha richiesto una combinazione tra le funzionalità di InDe e l'importazione di un foglio di stile (Cascading Style Sheet) customizzato. I metodi gestiti nelle videate sono gli eventi di load, refresh, update (che comprende inserimento, modifica e cancellazione dei record).

Di tutte le videate realizzate, quelle più complesse sono state le interfacce dedicate al caricamento delle immagini. Nella loro realizzazione hanno richiesto la gestione di alcuni comportamenti a livello di applicazione generale differenti dall'usuale. Il media ha richiesto delle continue rivisitazioni. Per la gestione dei file, il programma prevede che questi vengano caricati in una cartella temporanea. Tuttavia, da quanto si è definito in progettazione, una immagine viene salvata in una cartella ad essa dedicata con un nome univoco ideato solo per quello specifico elemento. Prima di

arrivare ad una soluzione ho dovuto creare dei progetti di test e verificare pregi e difetti. La soluzione che ho individuato è stata combinare dei metodi dell'applicazione. Ho combinato il `tempPath`, metodo che restituisce il una stringa con il percorso alla cartella temporanea e il `replace`, metodo per il rimpiazzo di stringhe. Infine, con un `makeDirectory`, altro metodo di InDe ho creato il nuovo percorso dove salvare i file in maniera dinamica. L'insieme di questi metodi sono serviti a creare un nuovo percorso di salvataggio dei file, per caricarli effettivamente ho optato per una doppia gestione: trascinamento dell'immagine nell'applicazione oppure click nella sezione di drop-down e importazione standard di documenti dal computer alla pagina web. Infine, per il caricamento ho optato per caricarli nella cache in modo tale da caricare i media di un elemento una sola volta per l'intera durata della sessione.

L'ultimo aspetto di focale importanza è risultato essere il controllo dell'evento `onCommand()`, ovvero l'evento lanciato ogni qual volta si preme un pulsante presente nella intestazione della videata (`refresh`, `save`, `insert`, `duplicate`) assieme al comando `load()`, dedicato al caricamento delle videate ed a eventuali controlli prima di aprire una qualsiasi schermata. La gestione degli `onCommand()`, rispetto a quanto aspiravo di creare, è stata diversa per ogni videata tuttavia il comportamento generale dell'applicazione ho cercato di standardizzarlo seguendo i seguenti principi:

- \* da una lista posso inserire, modificare e cancellare solo le principali informazioni;
- \* da una lista posso entrare nel dettaglio di un singolo elemento;
- \* dal dettaglio posso inserire dati, modificarli e cancellarli;
- \* dal dettaglio posso caricare media o dati particolari solo attraverso modali controllate (ade esempio nelle mail viene controllato che ci siano parole separate dal simbolo di `e` da un punto).

### Modali

Le modali sono videate di dimensioni ridotte, paragonabili ai popup, nelle quali ho dovuto gestire il caricamento di diverse informazioni. Per evitare di adottare una classe non basata su una tabella per ciascuna delle videate ho adottato le tabelle IMDB. Una tabella IMDB è una sorta di contenitore temporaneo che mantiene le sue informazioni fino al termine della sessione. Queste tabelle possono avere uno o più record così come possono essere utilizzate per effettuare dei filtri quando si apre una videata. Un esempio di suo utilizzo è stato l'inserimento delle variabili `ItemId` e `PartitionId` per filtrare un particolare articolo ed entrare quindi nel dettaglio di un singolo elemento. Queste schermate sono state ideate al fine di passare dati alle videate sottostanti e si cancellano, alla chiusura, per non occupare memoria nella cache.

## 3.4 Verifica, validazione e collaudo

Le operazioni di verifica, validazione e collaudo sono risultate più semplici di quanto mi aspettassi. Per verificare il prodotto ho compilato il codice ed eseguito ogni volta l'applicazione. Una volta eseguita la app, ho seguito una scaletta pre-impostata con tutte le operazioni implementate e possibili. All'inizio ho commesso l'errore di creare molte funzionalità e poi testarle. Dopo 3 settimane ho concluso che è molto più vantaggioso concentrarsi su una singola funzionalità, farla funzionare correttamente

e poi passare alla successiva. Alla fine di ogni schermata, ho effettuato dei test generali sulle funzionalità e dopo un collaudo da parte del mio tutor, la videata si considerava validata salvo eventuali errori. Il collaudo finale è stato gestito in maniera completamente diversa. Mentre la verifica e validazione sono state interne all'azienda e seguivano le linee guida indicate dai documenti di progetto, il collaudo è stato un rilascio anticipato al cliente. Quest'ultimo ha rieseguito i test da noi effettuati e ci ha rilasciato una lista delle nuove specifiche, miglioramenti o bug da risolvere. In seguito al fatto che, l'azienda segue la metodologia agile, i collaudi da parte del cliente sono stati più di uno. Grazie alla collaborazione tra cliente e fornitore il progetto alla fine dello stage è stato realizzato per intero con anche dei miglioramenti indicati nella sezione seguente.

### 3.5 Modulo RTC

Questo aspetto del progetto non ho potuto approfondirlo a sufficienza. Quello che però ho potuto verificare è stato che gli ideatori di Instant Developer hanno previsto che ci potesse essere l'esigenza di creare una applicazione in lingua. Il modulo RTC è un componente a pagamento utilizzabile solo con le licenze che lo includono. In questo modulo ci sono tre modalità di generazione delle traduzioni: google API key, Microsoft App Key e inserimento manuale. Per quanto riguarda i primi due metodi di traduzione sono a pagamento, mentre la terza è sicuramente più economica ma richiede tempo e conoscenza delle lingue. La componente è semplice da inserire e, se si desidera, si può acquistare anche la sorgente ed apportare le modifiche desiderate al codice. Lo studio effettuato è stato cercare di inserire il modulo ad una fork del progetto al suo termine.

### 3.6 Altri interventi

Il progetto ha subito un miglioramento fondamentale, ossia la gestione della lingua. L'applicazione web ideata deve poter creare informazioni sugli articoli. In aggiunta, i dati inseriti devono apparire in pagine web ed è possibile che anche persone di nazionalità diversa da quella italiana visiti il sito. Per queste ragioni, mi è stato chiesto di tradurre o creare un metodo di inserimento delle stesse informazioni in qualsiasi altra lingua.

Per gestire la lingua, visto che la richiesta è nata in un secondo momento, ho deciso di inserire nelle tabelle un nuovo campo `LanguageId` popolato con l'identificativo della lingua individuabile in una tabella del database. Introducendo questo nuovo campo in automatico anche le classi lo hanno ereditato ed a questo punto ho cercato di studiare un modo per gestire le traduzioni. La soluzione è stata quella di introdurre un campo booleano nella tabella `INFO_CONFIG` e un metodo statico `isTranslatable(string code)`, i quali mi indicavano se un campo deve essere tradotto oppure no, come nel caso delle immagini. Per lasciare completa libertà all'utente ho deciso di inserire il campo a video nella videata dedicata alle configurazioni in modo tale che se un utente vuole anche che le immagini cambino in seguito alla lingua è possibile farlo attraverso una check-box.

Le classi che hanno delle modifiche sono state `ITEM`, `ITEM_INFO`, `INFO_SECTION`, `INFO_CONFIG`. Quella che è stata apportata a tutte è il caricamento con una aggiunta alle where delle master query del `languageId` e il salvataggio della lingua, se necessario.



## Capitolo 4

# Valutazione retrospettiva

### 4.1 Instant Developer

#### 4.1.1 Instant Developer: creazione di una schermata base

In questa sezione ho desiderio di analizzare il particolare aiuto derivante da InDe. Illustro come una pagina web viene creata da zero per permettere ai lettori di questo documento di comprendere quanto può essere semplice, con l'aiuto di strumenti RAD come questo, creare applicazioni.

##### Passo 1: Database

La prima attività da svolgere è quella di pensare al proprio database, se è necessario. Quest'ultima affermazione è legata al fatto che è possibile creare applicazioni anche senza un database. In questo contesto ideiamo un semplice database per la gestione delle corsie di un supermercato. Le tabelle che creiamo sono Corsia e Articoli.

Per creare le due tabelle è necessario conoscere basi di dati soprattutto per evitare creare fin da subito query non troppo complicate che faticano ad essere completate.

Con il pulsante destro clicco due volte sul database definisco le specifiche che lo riguardano. A questo punto premo il pulsante destro del mouse sul database e seleziono la voce "Aggiungi tabella" inserisco le specifiche di questa. Con lo stesso procedimento di creazione tabella, creo i campi. Se desidero creare delle foreign key si deve trascinare la tabella interessata verso quella di destinazione.

**TO DO** immagine e ulteriori spiegazioni

##### Passo 2: Oggetto

Una volta creata la tabella, per creare un oggetto è sufficiente trascinare sull'applicazione la tabella tenendo premuto shift e ctrl. Viene generato un documento (classe) che potremo gestire come meglio crediamo. In questo caso mi limito a creare l'oggetto.

**TO DO** immagine e ulteriori spiegazioni

##### Passo 3: Videata

Trascinando l'oggetto sull'applicazione e premendo o shift o ctrl viene creata una videata basata sull'oggetto. Poi selezionando la videata interessata la si può modificare graficamente e gestire le funzionalità.

**TO DO** immagine e ulteriori spiegazioni

### 4.1.2 Estensioni

In questo contesto non sono è risultato necessario adoperare questa funzionalità. Tuttavia, ritengo importante accennare alle possibilità offerte dall'applicazione. InDe, infatti, permette di estendere le sue librerie con delle nuove. Ho visto che è possibile creare delle funzioni in SQL. Oppure implementare esternamente un file Java o C# e quindi richiamarlo nell'applicazione. L'aspetto negativo delle estensioni è che quando creo una applicazione InDe mi permette di passare da Java a C# e viceversa in poco tempo. Con le estensioni devo prevedere di creare due librerie una per linguaggio. E lo stesso vale per i database gestiti, se creo un comando sql nuovo (esempio nullif di sql server) devo creare il corrispettivo di tutti i database che dovrò utilizzare altrimenti mi devo limitare ad un singolo tipo di database.

**TO DO** immagine e ulteriori spiegazioni

## 4.2 Obiettivi

### 4.2.1 Stage

Gli obiettivi concordati nel piano di lavoro sono stati suddivisi in tre categorie: obbligatori, desiderabili e facoltativi. L'azienda ha espresso la richiesta che gli obbligatori siano completati, mentre per i desiderabili, almeno due dei tre indicati, siano portati a termine.

Gli obiettivi si distinguono in:

\* Obbligatori

- O01: Apprendimento della piattaforma Instant Developer;
- O02: Test delle funzionalità implementate e rilascio;
- O03: Utilizzo di Microsoft SQL Server.

\* Desiderabili

- D01: Gestione di progetto;
- D02: Comunicazione con il cliente;
- D03: Scrittura delle procedure T-SQL.

\* Facoltativi

- F01: Autonomia a risolvere nuove problematiche.

### 4.2.2 Personali

Sono entrato in contatto con l'azienda ospitante grazie ad un amico che mi ha messo in contatto con i responsabili. Dopo un colloquio ed una spiegazione generale delle attività svolte dall'azienda, hanno suscitato il mio interesse. L'idea di interfacciarmi con il mondo del lavoro prendendo in mano la gestione di dati sensibili e la possibilità di creare un gestionale rientra perfettamente nell'impiego da me cercato.

Dopo aver studiato economia presso l'Istituto Tecnico Commerciale Statale P.F. Calvi

ed informatica presso l'Università di Padova, entrare in una realtà lavorativa che concilia i due ambiti, mi sembra un buon completamento dei miei studi fino a questo momento.

Gli obiettivi che mi sono posto di raggiungere a livello personale oltre a quelli concordati con l'azienda sono:

- \* Accrescere le conoscenze in merito al mondo RAD e Data Warehouse;
- \* Migliorare le capacità di realizzazione di applicazioni seguendo il metodo Bottom-Up;
- \* Apprendere come interfacciarmi con i clienti;
- \* Migliorare le mie capacità di Problem Solving.

### **4.3 Considerazioni personali**



# Glossario

**API** In informatica con il termine *Application Programming Interface API* (ing. interfaccia di programmazione di un'applicazione) si indica ogni insieme di procedure disponibili al programmatore, di solito raggruppate a formare un set di strumenti specifici per l'espletamento di un determinato compito all'interno di un certo programma. La finalità è ottenere un'astrazione, di solito tra l'hardware e il programmatore o tra software a basso e quello ad alto livello semplificando così il lavoro di programmazione

**UML** In ingegneria del software *UML, Unified Modeling Language* (ing. linguaggio di modellazione unificato) è un linguaggio di modellazione e specifica basato sul paradigma object-oriented. L'*UML* svolge un'importantissima funzione di "lingua franca" nella comunità della progettazione e programmazione a oggetti. Gran parte della letteratura di settore usa tale linguaggio per descrivere soluzioni analitiche e progettuali in modo sintetico e comprensibile a un vasto pubblico

**C#** Linguaggio di programmazione orientato agli oggetti che consente di creare una vasta gamma di applicazioni protette e affidabili per .NET Framework. Esso può essere adottato per creare applicazioni client Windows, servizi Web XML, componenti distribuiti, applicazioni clientserver, applicazioni di database e molto altro.

**Java** Linguaggio di programmazione ad alto livello, orientato agli oggetti e a tipizzazione statica, che si appoggia sull'omonima piattaforma software, specificamente progettato per essere il più possibile indipendente dalla piattaforma hardware di esecuzione.

**IDE** Un ambiente di sviluppo integrato (in lingua inglese Integrated Development Environment), è un software che, in fase di programmazione, aiuta i programmatori nello sviluppo del codice sorgente di un programma. Spesso l'IDE aiuta lo sviluppatore segnalando errori di sintassi del codice direttamente in fase di scrittura, oltre a tutta una serie di strumenti e funzionalità di supporto alla fase di sviluppo e debugging.

**Accordo di non divulgazione** Un accordo di non divulgazione (in lingua inglese Non-Disclosure Agreement, NDA) è un negozio giuridico di natura sinallagmatica che designa informazioni confidenziali e con il quale le parti si impegnano a mantenerle segrete, pena la violazione dell'accordo stesso e il decorso di specifiche clausole penali in esso contenute.

**Instant Developer** Instant Developer, in particolare la versione per desktop, Foundation è la piattaforma di sviluppo adottata per creare il progetto oggetto della tesi.

Si tratta di un RAD che permette di realizzare applicazioni in tempi molto brevi senza la necessità di conoscere il codice alla base del progetto.

# Acronimi e abbreviazioni

API - Application Program Interface

UML - Unified Modeling Language

IDE - Integrated Development Environment

NDA - Non-Disclosure Agreement

InDe - Instant Developer





# Fonti bibliografiche

- \* *Informazioni sull'azienda - consultato 23/05/2019*  
<https://www.tepui.it/servizi/>
- \* *Definizione di Software gestionale - consultato 23/05/2019*  
<https://bit.ly/2H8Cr4P>
- \* *Definizione di C# - Consultato 23/05/2019*  
<https://bit.ly/308zwjY>
- \* *Definizione di Java - Consultato 23/05/2019*  
<https://bit.ly/1dHppDG>
- \* *Definizione di Qlik - Consultato 23/05/2019*  
<https://bit.ly/2J76agL>
- \* *Definizione di SSMS - Consultato 23/05/2019*  
<https://bit.ly/2VPWZHn>
- \* *Definizione di Microsoft Power BI - Consultato 23/05/2019*  
<https://bit.ly/2FkLBeX>
- \* *Definizione di IDE - Consultato 23/05/2019*  
<https://bit.ly/20zruuv>
- \* *Definizione di NDA - Consultato 23/05/2019*  
<https://bit.ly/2Lw4sYu>
- \* *Scegliere SQL Server - Consultato 09/07/2019*  
<https://bit.ly/2YJ3RVt>
- \* *Statistiche SQL Server - Consultato 09/07/2019*  
<https://bit.ly/2XxEjhv>
- \* *Documentazione in pdf di InDe - Consultato 22/07/2019*  
<https://doc.instantdeveloper.com/inde-users-guide.pdf>