

UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Dipartimento di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea in Informatica

Progetto e Sviluppo di un applicativo web per
la gestione e archiviazione di analisi EDXRF

Simone Biagini
Tesi di Laurea

Relatore:
Prof. Riccardo Martoglia

Anno Accademico 2014/2015

RINGRAZIAMENTI

Ringrazio l'Ing. Riccardo Martoglia per la continua disponibilità e assistenza.

Un ringraziamento speciale va alla mia famiglia, la mia ragazza e ai miei amici che in questi anni mi hanno sempre sostenuto e sopportato.

PAROLE CHIAVE

Python

Django

Analisi Chimiche

Basi di Dati

Lavoro su Stringhe

Indice

Introduzione	pag. 1
---------------------	---------------

I – Il Caso di studio	pag. 3
------------------------------	---------------

1 Background	pag. 4
---------------------	---------------

1.1	Richiesta iniziale	pag. 4
1.2	Dettagli concettuali	pag. 5
1.2.1	Specifiche dei requisiti software	pag. 6
1.2.2	Introduzione e obiettivi	pag. 6
1.2.3	Descrizione generale	pag. 7
1.3	Problematiche affrontate	pag. 9

2 Approccio risolutivo	pag. 10
-------------------------------	----------------

2.1	Soluzione proposta	pag. 10
2.2	Tecnologie usate	pag. 14
2.3	Requisiti funzionali	pag. 15
2.3.1	Inserimento Elementi	pag. 16
2.3.2	Inserimento Materiali e Fam.Materiali	pag. 16
2.3.3	Inserimento Fornitori, Scopo Analisi, Tag	pag. 17
2.3.4	Inserimento del risultato delle Analisi	pag. 18

II – Applicazione web based per lo studio di analisi chimiche	pag. 20
3 Progettazione database	pag. 21
3.1 Diagramma E/R	pag. 21
3.2 Traduzione da schema E/R a schema logico	pag. 25
3.3 Creazione DB	pag. 26
3.4 Dalla realtà al DB	pag. 32
4 Implementazione	pag. 34
4.1 Inserimento risultati	pag. 34
4.2 Salvataggio	pag. 37
4.3 Visualizzazione dati	pag. 38
4.4 Libreria Openpyxl	pag. 40
Conclusione e sviluppi futuri	pag. 44

Indice delle figure

Figura 1.1: Input software proprietario	pag
Figura 1.2: Risultati del sw proprietario	pag
Figura 1.3: Organizzazione precedente	pag
Figura 1.4: Output impaginato	pag
Figura 3.1: Schema ER scheletro	pag
Figura 3.2: Entità Analisi	pag
Figura 3.3: Relazione Fornitore – Analisi	pag
Figura 3.4: Relazione Tag - Analisi	pag
Figura 3.5: Relazione Tabella_Elementi – Elenco_Materiali	pag
Figura 3.6: Relazione Famiglia_Materiali – Elenco_Materiali	pag
Figura 3.7: Relazione Elenco_Materiali - Analisi	pag
Figura 3.8: Input ElencoMateriali	pag
Figura 3.9: Input Tabella_Elementi	pag
Figura 3.10: Input Famiglia_Materiali	pag
Figura 3.11 Form Input Fornitore	pag
Figura 3.12 Display results precedente	pag
Figura 3.13: Stringa risultato	pag
Figura 3.14: Template Tabella risultati	pag
Figura 4.1: Template ElencoAnalisi	pag
Figura 4.2: Template filtri risultati	pag
Figura 4.3: Output grafico	pag

Introduzione

Il lavoro di tirocinio svolto riguarda la realizzazione di un software che possa dare supporto alla gestione dei risultati delle analisi chimiche effettuate dallo spettrofotometro¹. Questo macchinario è situato nel laboratorio geologico della ceramica Italgraniti Group. All'interno del laboratorio ceramico viene creato l'impasto che, dopo diverse lavorazioni, andrà a formare il prodotto principale di tale azienda: la mattonella. Il motivo di successo o di fallimento del prodotto finale dipende proprio dalla qualità di tale lavorazione, la quale deve chiaramente raggiungere elevati standard. Per questo motivo sono numerosi i controlli che vengono effettuati su campioni di materia attraverso analisi EDXRF. Questa particolare tecnica di analisi permette di conoscere la composizione elementare di un campione attraverso lo studio della radiazione di fluorescenza X.

Lo spettrofotometro è provvisto di nove slot (alloggiamenti) disponibili. Per questo motivo si possono effettuare analisi su massimo nove campioni alla volta, solitamente tre per ogni materiale.

Quindi i risultati generati saranno composti dal risultato dell'analisi su ogni campione. Questi risultati sono i dati che dovranno essere gestiti del software che si è creato.

La necessità derivatane è dunque quella di poter accorgersi in breve tempo di eventuali anomalie presenti nei campioni, ma anche di poter effettuare controlli sui risultati risalenti a giorni, settimane o mesi prima, poiché non sempre i difetti possono emergere immediatamente.

Il lavoro svolto consiste nell'aver creato un applicativo web che permetta di svolgere queste operazioni velocemente e da un unico software, garantendo la persistenza dei dati nel tempo.

¹ **Spettrofotometro:** Strumento per effettuare analisi chimiche non distruttiva che permette di conoscere la composizione elementare di un campione attraverso lo studio della radiazione di fluorescenza X.

Grazie all'utilizzo di interfacce semplici e intuitive l'operatore di laboratorio, nonché utente del software, potrà copiare i risultati generati dallo spettrofotometro ed incollarli in un'area dedicata del nuovo programma, che riorganizzerà il testo inserito in maniera consona al salvataggio nel database, e da quel momento in poi, grazie a template² dedicati sarà possibile visualizzare i dati inseriti sotto diversi punti di vista. Sarà possibile ad esempio esportare su excel i risultati di un'unica analisi, oppure scegliere i risultati da visualizzare grazie a filtri personalizzati che selezioneranno tra tutti i risultati presenti nel database, quelli che soddisfano determinate richieste. Di questi ultimi, per poterli confrontare tra di loro è possibile crearne e visualizzarne il loro grafico, così da evidenziarne eventuali variazioni nel tempo.

Sono state inoltre predisposte altre tabelle in cui inserire dati descrittivi, il processo di lavorazione per creare i campioni (tag), un elenco dei materiali creati, e degli elementi che li compongono, e infine descrizioni e informazioni dei fornitori di tali materiali.

Questo lavoro è stato effettuato utilizzando come framework Django³ e come linguaggio di programmazione Python, con l'ausilio di qualche libreria esterna per gestire per esempio la creazione dei file excel (libreria Openpyxl). Tutti gli strumenti utilizzati sono free ed open source⁴.

Una volta testato il software con un'ampia gamma di dati, verificandone il comportamento in qualsiasi situazione si presenti, potrà essere installato sui server aziendali per utilizzarlo come strumento principale di salvataggio e consultazione delle analisi effettuate.

² **Template:** documento o programma nel quale, come in un foglio semicompiato cartaceo, su una struttura generica o standard esistono spazi temporaneamente "bianchi" da riempire successivamente.

³ **Django:** framework che ha il compito di facilitare la creazione di siti web dinamici o applicazioni web.

⁴ **Free e Open Source:** si dice dei software gratuiti, dei quali si può reperire il codice sorgente online e non sono coperti da licenza.

Parte I

Il caso di studio

Capitolo 1

Background

In questa sezione saranno descritti i passaggi che hanno preceduto la realizzazione vera e propria del progetto partendo dalle richieste iniziali del committente, procedendo con uno studio di fattibilità utilizzato per verificare se tutte le richieste potevano essere soddisfatte, e in che modo.

Momenti fondamentali di queste fasi sono risultati essere i colloqui con il committente (nel caso specifico utente del futuro prodotto) per capire quali fossero i punti fondamentali su cui porre maggiore attenzione durante il lavoro.

Ultima fase preliminare è stata la scelta delle tecnologie da utilizzare per soddisfare i requisiti richiesti dal cliente e il controllo dei requisiti imposti dalle tecnologie con cui andava ad interagire il software.

1.1 Richiesta iniziale

Il progetto iniziale prevedeva la creazione di un sistema di gestione in ambiente Windows, a supporto dello spettrofotometro EDXFR presente nel laboratorio ceramico utilizzato per le analisi chimiche.

In particolare era richiesto che il sistema organizzasse i dati per la preparazione delle prove e dovesse interfacciarsi con il software dello spettrofotometro nell'elaborazione e nell'output dei risultati delle prove.

Una volta terminate le prove, l'utente deve poter avere libero accesso alla totalità dei risultati prodotti. Dopo averli riportati sul software creato per poterli osservare, comparare tra loro, e infine in grado di poterli filtrare in base agli attributi che più preferisce per far emergere incongruità o difetti sui campioni.

In caso di evidenti problematiche deve poter impaginare i risultati in maniera leggibile e comprensibile, magari supportati da grafici, per poterli mostrare e far

comprendere ai responsabili superiori, non in grado di tradurre i dati elaborati della macchina (percentuali di ossidi presenti nel campione) nel problema reale.

Entrando più nello specifico è stato necessario aggiungere alcuni accorgimenti risultati da piccole richieste tra i quali: il riempimento automatico di campi standard quando si procede ad inserire una nuova analisi che richiede un numero progressivo (procedura da ripetere ogni volta), la data odierna, la data del campione, e altri dati utili ai fini del programma.

É possibile far precompilare dal sistema quasi tutti i suddetti campi, attendendo l'inserimento manuale da parte dell'operatore dei risultati generati dalla macchina con un semplice "copia-incolla".

1.2 Dettagli concettuali

Lo scopo di questa sezione è quello di comprendere il problema, le condizioni di risoluzione ed i vincoli, partendo da una generale definizione della necessità dei potenziali utenti.

Saranno espone e descritte qui di seguito le motivazioni e le necessità che hanno portato all'ideazione, definizione e creazione del software per la gestione delle analisi. Saranno inoltre riportate in modo corretto e non ambiguo le specifiche dei requisiti del software, senza descrivere ancora alcun dettaglio progettuale o implementativo.

In altre parole verrà stabilito e descritto il "che cosa" il sistema deve fare, senza parlare ancora del "come" dovrà farlo.

Siccome si utilizzerà spesso e volentieri il termine "requisito" è opportuno specificarne il significato. Con il termine "requisito" in italiano si intende una caratteristica o proprietà che una data persona o cosa è tenuta a possedere.

Noi ci interesseremo ai requisiti del prodotto, in particolare ai requisiti del software dove il termine "requisito" assume un significato più ampio, vale a dire qualcosa che il prodotto deve fare o una caratteristica che deve possedere.

1.2.1 Specifica dei requisiti del software

Si andrà ora a elencare ed esplicitare quelli che sono i requisiti non funzionali come per esempio le modalità con cui il sistema interagisce con l'esterno, con gli utenti, con l'hardware e con altri software.

Si descriveranno le prestazioni, gli attributi (come la condizione di portabilità), la sicurezza e così via.

Si valuterà che tipo di interfaccia esterna sarà necessario sviluppare, si analizzeranno gli aspetti riguardanti la sicurezza e si cercherà di giudicare se il tempo di risposta sia una caratteristica saliente o trascurabile.

1.2.2 Introduzione e obiettivo

La presente sezione ha lo scopo di riportare la visione globale dell'intero documento sottostante di "specifica dei requisiti". La struttura del documento è quella suggerita dallo standard IEEE 830 noto come SRS⁵ (*Software Requirements Specifications*).

Lo scopo di tale documento è quindi di rappresentare, nel modo più preciso possibile, consistente, non ambiguo e comprensibile, l'impatto che la soluzione studiata e adottata per la gestione delle analisi chimiche effettuate dallo spettrofotometro EDXRF abbia sul procedimento attuale di gestione. Si tenga presente che l'approccio seguito per lo sviluppo del software è di tipo prototipale, quindi nuovi requisiti potranno essere introdotti in seguito, oltre a tutti i vincoli che fino a questo momento non sono ancora stati individuati.

Le conseguenze del suo utilizzo saranno certamente il risparmio di tempo sul procedimento standard, la certezza di avere un archivio sempre aggiornato sul server aziendale, e la comodità per l'operatore di avere tutte le funzionalità racchiuse in un unico software. Successivamente si potranno aggiungere funzionalità ora non disponibili.

⁵ **SRS:** è una descrizione completa del comportamento di un sistema software da sviluppare.

Definizioni, acronimi e abbreviazioni

Spettrofotometro EDXRF	Macchina che utilizza la spettrofotometria: tecnica di analisi non distruttiva che permette di conoscere la composizione multielementare di un campione attraverso lo studio della radiazione di fluorescenza X
Operatore	Rappresenta un geologo del laboratorio ceramico, che ha il compito di fare l'impasto per le piastrelle, e dunque di fare le analisi, quando richiesto, su campioni dello stesso
Analisi	Sono il risultato delle analisi svolte dallo spettrofotometro su un massimo di 9 campioni di materiali

1.2.3 Descrizione generale

Il software richiesto si propone come supporto alla gestione dell'attività di analisi dei risultati ottenuti dallo spettrofotometro che non permette una manipolazione libera da parte dell'utente della macchina.

La specifica dei requisiti del sistema si riferisce alla gestione dei risultati in modo indipendente dal software proprietario dello spettrofotometro, e quindi del pc di laboratorio dal quale verrà utilizzato.

Il beneficio principale che deriva dall'utilizzo del software è la molteplicità di operazioni che si possono effettuare senza ricorrere all'utilizzo di altri software, funzionalità che può sembrar banale, ma che offre all'utente la possibilità di velocizzare il procedimento di gestione di attività che precedentemente venivano svolte utilizzando diversi programmi di base del computer.

Gli utenti che operano nel laboratorio ceramico geologico all'interno del quale è presente lo spettrofotometro non sono obbligati ad avere una preparazione elevata nell'uso di strumenti informatizzati, se non per quelli necessari a svolgere il loro lavoro specifico.

Vale a dire che sono sicuramente esperti del problema, ma in generale non di applicazioni software. Per questo motivo durante la realizzazione del software saranno frequenti i colloqui con un operatore per tener conto dell'accessibilità dell'interfaccia utente, che dovrà essere confermata dagli utenti stessi. Per questi motivi si è deciso di prestare grande attenzione alla progettazione di un design semplice, leggero e intuitivo.

L'usabilità quindi implica che le informazioni debbano essere organizzate e strutturate in maniera da garantire la massima fruibilità, eventualmente anche all'utente meno esperto.

In altre parole, deve essere realizzata un'interfaccia non soggetta a particolari vincoli grafici, ma con l'obiettivo dell'essenzialità. L'informazione deve essere organizzata in modo tale che gli utenti siano in grado di realizzare i loro obiettivi in modo semplice.

Per creare un buon applicativo, nonostante non sia esplicitamente richiesto dal committente, è buona regola rispettare il requisito della navigabilità: rendere l'accesso alle informazioni ridondante, ossia prevedere almeno due strade diverse per accedere allo stesso contenuto.

Non è stato posto alcun vincolo all'occupazione di memoria in quanto i dati in questione sono unicamente stringhe e dati di testo di dimensioni molto contenute.

Inoltre, la frequenza con la quale vengono inseriti nuovi dati è molto bassa: circa un'analisi, composta al massimo da nove stringhe, al giorno.

Gli operatori saranno coloro che sono incaricati di effettuare le analisi sui campioni di materiale.

Non sono stati posti vincoli di processo, cioè sul metodo di sviluppo, né sul linguaggio di programmazione, né sugli strumenti da utilizzare.

L'unico vincolo è quello di compatibilità con i sistemi utilizzati in azienda (Windows 7), e con il server aziendale su cui sarà caricato il software. Quest'ultimo non impone molti vincoli.

1.3 Problematiche affrontate

Più che veri e propri problemi progettuali è stato necessario comprendere a fondo i requisiti richiesti per poi “tradurli” in linguaggio informatico poiché il committente, esperto del proprio lavoro, esprimeva quesiti specifici dell'impresa spesso lontani dalla comprensione dei non addetti al settore chimico.

È bastato un primo breve colloquio per rivedere le richieste iniziali che pretendevano un interfacciamento con il software dello spettrofotometro:

software proprietario della casa produttrice dell'apparecchiatura.

Dopo aver studiato la documentazione disponibile sul software, e aver appurato che non era possibile nessuna interazione con esso, è subito stata evidente la necessità di modificare i requisiti.

Durante i successivi colloqui, finalizzati allo studio di fattibilità delle altre richieste, e a conoscere il comportamento del software proprietario, per analizzare il suo output, sono stati ritrattati anche altri requisiti.

Capitolo 2

Approccio risolutivo

In questo capitolo si osserveranno tutti i singoli step che sono stati percorsi per arrivare alla risoluzione del problema.

Si tratteranno in primo luogo le linee guida teoriche, per poi passare ai dettagli pratici nel capitolo successivo.

2.1 Soluzione proposta

Appreso quali sono le reali condizioni, e il modo in cui agisce il sistema si è potuta fare una stima di ciò che era possibile realizzare.

Prima di tutto è stato eliminato il requisito dell'interazione tra i due sistemi: quello da creare e quello proprietario dello spettrofotometro, a causa della sua riservatezza. Se eventualmente fosse possibile migliorare qualche aspetto del software proprietario, questo è di competenza della casa produttrice.

Descrivendo ancora la fase precedente all'avvio delle analisi, per quanto riguarda la richiesta di preparazione dei dati, dopo aver capito quali fossero quelli che necessitavano di essere inseriti in precedenza all'avvio dell'elaborazione, è stata proposta un'adeguata soluzione. Il problema è stato infatti risolto formando automaticamente una sigla identificativa e progressiva nel pannello di inserimento di una nuova analisi, come quella mostrata nella **figura 1.1**, creata concatenando un numero progressivo, la data del campione, e il materiale su cui si sta svolgendo l'analisi.

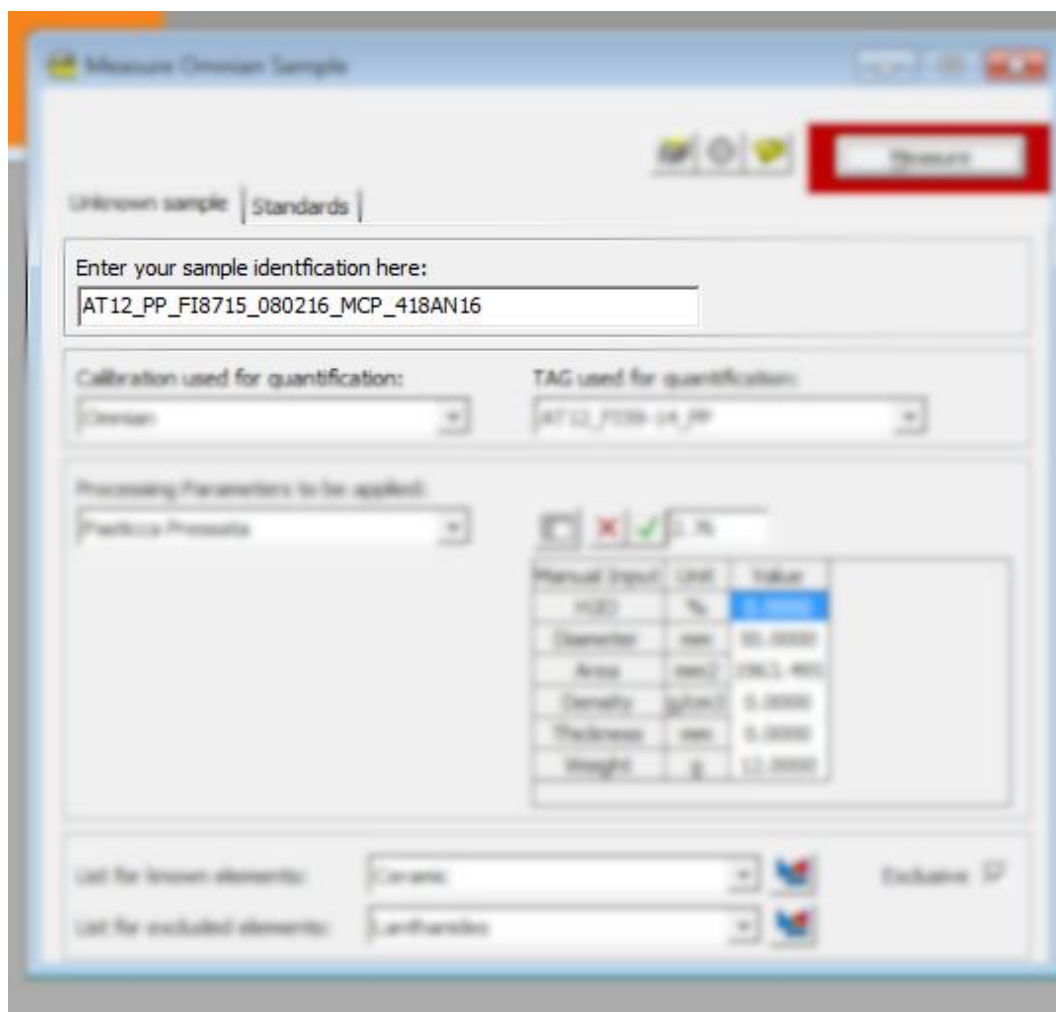


Figura 1.1: Input software proprietario

Copiando questo identificativo nel pannello di input della macchina, ai risultati generati verranno assegnati gli stessi identificativi con il suffisso _A, _B, _C, etc fino al numero di campioni inseriti (**figura 1.2**). Inoltre il software calcolerà altre due righe: la media dei campioni inseriti per ogni campo, e la deviazione standard. Concetto che verrà spiegato più precisamente nel capitolo successivo.

Nr	Ident	Seq
1	AT12_COTTO_P_FI8715_150216_417AN16_A	1 of 1
2	AT12_COTTO_P_FI8715_150216_417AN16_B	1 of 1
3	AT12_COTTO_P_FI8715_150216_417AN16_C	1 of 1
4	AT12_COTTO_P_FI8715_150216_417AN16_?	Ave of 3
5	AT12_COTTO_P_FI8715_150216_417AN16_?	SDev of 3

Figura 1.2: Risultati del sw proprietario

Durante il ciclo di sviluppo è risultato evidente che la richiesta di aprire il file excel precedentemente utilizzato, e dopo l'analisi, aggiungervi un collegamento ipertestuale ad un file contenente il risultato di tale analisi come il file presente in **figura 1.3**, potesse essere un lavoro superfluo, o sostituibile da tutt'altro ragionamento.

	A	B	C	D	E	F	G	H
1	ELENCO ANALISI 2015							
2	N° PROVA	DATA ANALISI	DATA CAMPIONE	IDENTIFICATIVO XRF	MATERIALE	FORNITORE	ID	COLLEGAMENTO RISULTATI
3	1AN15		24/02/2015	ML1 170215 LOOSE	ML1			C:\Users\User\Desktop\DAT\ML1 170215.xlsx
4	2AN15	30/03/2015	11/04/2014	P9550 PP STD 26	P9550			P9550\P9550 PP STD 26 2AN15.xlsx
5	3AN15	30/03/2015	03/12/2015	P9550 PP BOX 27	P9550			P9550\P9550 PP BOX 27 3AN15.xlsx
6	4AN15		26/03/2015	P9550 PP controllo	P9550			P9550\P9550 PP 26-03-2015 4AN15.xlsx

Figura 1.3:Organizzazione precedente

È stato quindi sostituito dalla creazione di template html sostitutivi del file excel, contenenti anch'essi un link, con un altro template del progetto in cui veniva stampato il risultato delle analisi impaginato secondo le preferenze dell'operatore di laboratorio.

Ed eventualmente, come anticipato prima, da questo template c'è la possibilità di stampare, o meglio esportare, questi risultati, in un file excel [2] in cui vengono disposti ordinatamente, con evidenziati i valori che non rispettano gli standard (**figura 1.4**).

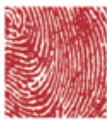
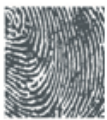
	A	B	C	D	E	F	G	H	I
1	ITALGRANITI GROUP		LABORATORIO ITALGRANITI GROUP					ANALISI CHIMICHE	
2	 IMPRONTA CERAMICHE	 ITALGRANITI							
3									
4			CAMPIONE					N° Prova	
5			ML1					64AN14	
6									
7	DATA PRELIEVO:		25/11/2014						
8	NOME FILE:		ML1_PP_STD_251114_9						
9	FORNITORE:								
10									
11									
12	Strumento:		epsilon3 PANalytical					Metodo:	XRF
13									
14									
15									
16			ANALISI CHIMICA						
17									
18			Componenti	Conc. (%)	Tolleranza				
19					Min.	Max.			
20			Na2O	0,83					
21			MgO	0,15					
22			Al2O3	8,05					
23			SiO2	83,49					
24			K2O	4,40					
25			CaO	1,20					
26			TiO2	0,07					
27			Fe2O3	0,26					
28			P. F.	1,37					
29									
30									
31									
32			Tecnico Operatore		Data Analisi:				
33					28/04/2015				
34									

Figura 1.4: Output impaginato

Questo file excel dovrebbe essere in grado di far capire, anche ai meno esperti, le anomalie emerse dall'analisi.

2.2 Tecnologie usate

Come descritto precedentemente non essendo stato posto nessun vincolo alle tecnologie da usare, si è optato per rimanere su tecnologie interamente open source.

Django (versione 1.8)

Django è un framework, scritto interamente in python, free, open source che ha l'intento di semplificare la costruzione di siti dinamici e applicazioni web, fornendo una struttura logica e modulare per sistemi fortemente interattivi e complessi.

Lo schema di programmazione che esso definisce prende il nome MTV dalle iniziali delle sue tre componenti principali: *Model*, *Template*, *View*. Queste componenti mirano a tenere separati su tre livelli di astrazione diversi i dati (Model), le funzioni che agiscono sui dati (View) e come questi vengono presentati all'utente (Template).

Django è ormai sempre più diffuso, ed alcuni dei più noti siti che lo utilizzano sono: Instagram, Mozilla, Pinterest, The Washington Times e Public Broadcasting Service. [2]

Python (3.5)

Python è un linguaggio di programmazione interpretato, interattivo e orientato agli oggetti. Incorpora al suo interno moduli, eccezioni, tipizzazione dinamica, tipi di dato di altissimo livello e classi. Python combina una eccezionale potenza con una sintassi estremamente chiara. Ha interfacce verso molte chiamate di sistema, oltre che verso diversi ambienti grafici, ed è estensibile in C e C++. È inoltre usabile come linguaggio di configurazione e di estensione per le applicazioni che richiedono un'interfaccia programmabile. Da ultimo, python è portabile, e sempre più conosciuto, nel caso si affidasse un futuro ampliamento del progetto ad altri. Questi sono alcuni dei motivi che hanno portato alla scelta dell'utilizzo di python per la realizzazione del software. [3]

SQLite

La scelta del database ricadeva principalmente tra MySQL e SQLite, ma viste le ancora presenti incompatibilità tra MySQL e Windows 10 (sistema operativo su cui è stato scritto il codice), e la non enorme quantità di dati da inserire, e di accessi contemporanei si è potuto optare per SQLite che è un ottimo database soprattutto per progetti di dimensioni medie, oltre che da usare per i test. I suoi autori sostengono che sia l'SQL engine più usato al mondo dato che si trova in moltissime applicazioni su desktop oltre che su cellulari, palmari, lettori mp3 e altro ancora. Il sorgente di SQLite è di dominio pubblico [4].

Bootstrap

Si tratta di una raccolta di strumenti liberi per la creazione di siti e applicazioni per il web. Essa contiene modelli di progettazione basati su HTML e CSS, sia per la tipografia, che per le varie componenti dell'interfaccia, come moduli, bottoni e navigazione, e altri componenti dell'interfaccia.

È nato dagli sviluppatori di Twitter, e ora, per chi cerca di costruire un sito dal design convenzionale, utilizzando uno strumento che possa fornire un punto di partenza solido nella strutturazione del progetto, Bootstrap è lo strumento ideale.

È stato utilizzato soprattutto per dare un aspetto professionale ai template [5].

2.3 Requisiti funzionali

Di seguito verranno descritte dettagliatamente le principali operazioni che si possono svolgere all'interno dell'applicazione web.

Non essendo stati posti grossi vincoli dal committente, interessato al risultato finale e non al procedimento per ottenerlo, è stato possibile costruire questo documento sui requisiti funzionali strada facendo.

2.3.1 Inserimento Elementi

L'operazione di inserimento degli elementi verrà effettuata una sola volta al momento del primo utilizzo del software. Il motivo di tale operazione si capirà dopo la descrizione delle entità e soprattutto delle relazioni che costituiranno lo schema E/R.

Si tratta dell'inserimento di tutti gli elementi chimici della tavola periodica, poiché è di questi che sono composti i campioni che si andranno ad analizzare.

Dato che questa operazione è da svolgere una sola volta, può essere fatta dall'interfaccia di amministrazione messa a disposizione da Django, non troppo curata, ma estremamente efficace e semplice da personalizzare.

RF01	Popolazione tabella <i>Elementi</i>
Input	Sigla e Nome degli elementi chimici
Processo	Operazione di salvataggio
Output	Reindirizzamento al riepilogo delle istanze inserite.(Sempre all'interno dell'interfaccia di Amministrazione)

2.3.2 Inserimento Materiali e Famiglie Materiali

Per quanto riguarda l'inserimento dei materiali e delle famiglie dei materiali può essere fatto un discorso analogo al precedente, con la differenza che questi inserimenti verranno effettuati più frequentemente rispetto agli elementi. Si capirà meglio dalle relazioni dello schema E/R, ma possiamo già accennare in che modo sono in relazione tra di loro queste tabelle:

Una *Famiglia di materiali* comprende uno o più *Materiali*.

Ogni *Materiale* è formato da uno o più *Elementi*

La cardinalità della relazione verrà descritta più precisamente nell'apposita sezione.

RF02	Popolazione tabella <i>Materiale e Fam.Materiale</i>
Input	1.Nome del <i>Materiale</i> , ed associazione degli <i>Elementi</i> chimici che ne fanno parte. 2. Nome della <i>Fam.Materiali</i> , ed associazione dei <i>Materiali</i> che ne fanno parte.
Processo	Operazione di salvataggio
Output	Reindirizzamento al riepilogo delle istanze inserite.(Sempre all'interno dell'interfaccia di Amministrazione)

2.3.3 Inserimento Fornitori/Scopo Analisi/Tag

L'inserimento dei fornitori, dello scopo e del tag sono semplici campi descrittivi delle analisi. Il sistema, oltre a elaborare i dati e convertirli nella forma più leggibile e utilizzabile, riveste un importante ruolo di archivio di rilevanti informazioni circa i processi chimici con i quali si ottengono i campioni analizzati, l'elenco dei propri fornitori e la motivazione dell'analisi.

RF03	Popolazione database
Input	Dati relativi alla tabella da popolare.
Processo	Operazione di salvataggio dei dati all'interno del database.
Output	Se l'operazione va a buon fine, si viene reindirizzati ad una pagina di riepilogo di tutte le istanze inserite in tale tabella

2.3.4 Inserimento del risultato delle analisi

L'inserimento del risultato delle analisi è l'operazione più importante: dal software esterno vengono importati i risultati delle analisi effettuate.

RF04	Inserimento
Input	Si copiano i risultati generati dal software esterno, e si incollano nella text area dedicata.
Processo	Il sistema provvederà a ripulire il testo, separarlo in righe, e dalla riga relativa alla media delle analisi (unica riga rilevante per l'operatore), ricavare i singoli elementi.
Output	Rappresentazione tabellare della riga rappresentante la media.

Prima di procedere con i capitoli più significativi si vuol precisare la differenza tra il termine "analisi", e il termine "risultati" diffusamente usati da qui in avanti.

Fisicamente, quando viene effettuata un'analisi, si inseriscono fino a nove campioni di materiale nello spettrofotometro. Per non aspettare troppo tempo (su tre campioni la macchina impiega circa 45 minuti), ma assicurandosi anche un buon risultato, le analisi si effettuano su tre campioni, così da poter evitare analisi su materiali accidentalmente contaminati. Infatti capita spesso che nel campione cadano residui di colla posizionata nei pressi della macchina in laboratorio. Una volta terminata l'analisi lo spettrofotometro produce i risultati, che sono i valori degli ossidi di ogni campione (in questo caso tre), più altre due righe, una rappresentante la media dei valori, e l'altra la deviazione standard (nel caso questa fosse molto alta, è chiara la presenza di un campione contaminato).

Quindi parlando di analisi, si intende anche l'insieme di tutti i dati descrittivi (es. data, identificativo, materiale ecc), mentre per ogni analisi sono presenti dei risultati di $n + 2$ righe (n = numero di campioni).

Parte II

Applicazione web based per lo studio di analisi chimiche

Capitolo 3

Progettazione database

Come punto di partenza per la realizzazione del software, è stato necessario, comprendere quali fossero le entità, i loro attributi, e soprattutto le relazioni tra loro per poter progettare il database. Da esso dipenderà la buona riuscita e la difficoltà nel programmare tutte le operazioni.

Si procederà mostrando prima l'intero schema E/R (**figura 3.1**), per motivarne successivamente la relazione tra loro. Si descriverà infine brevemente il loro significato e il loro scopo, attributi compresi.

3.1 Diagramma E/R

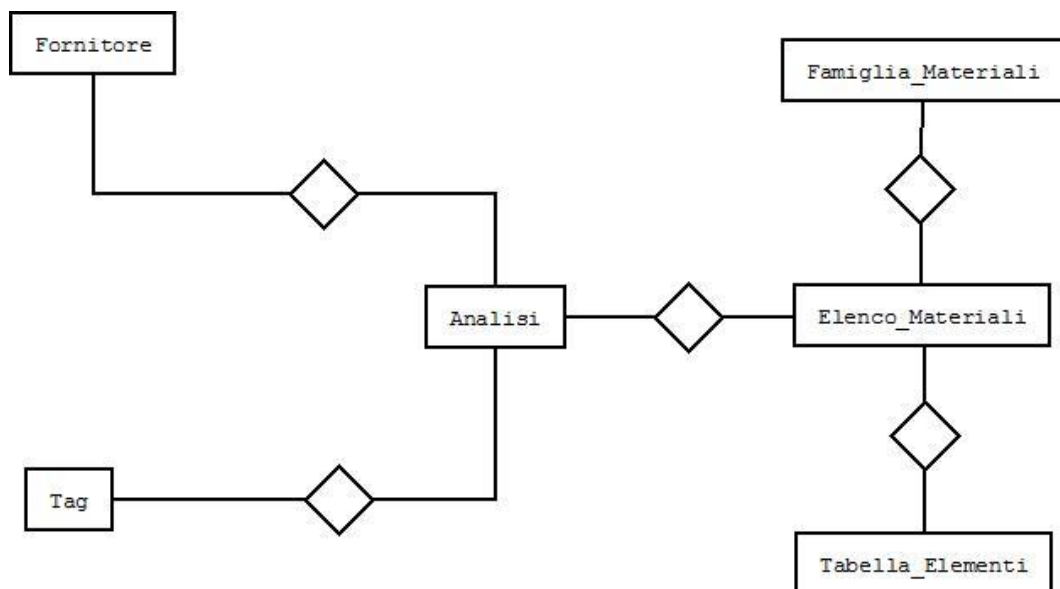


Figura 3.1: Schema ER scheletro

Vediamo quindi quali saranno gli attributi dell'entità **Analisi**, nonché quella principale e più importante (**figura 3.2**).

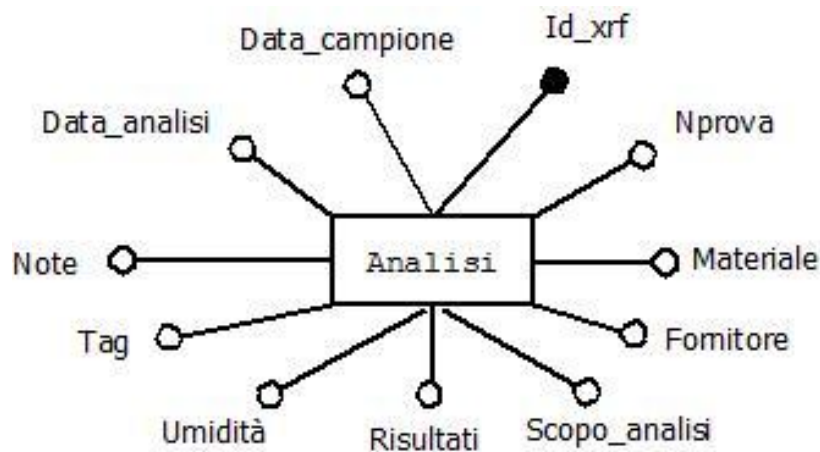


Figura 3.2: Entità Analisi

Data_analisi, **Data_campione**, **Note**, **Umidità**, **Scopo_analisi**, **Fornitore** sono attributi descrittivi dell'analisi effettuata e non hanno bisogno di spiegazioni particolari.

L'analisi viene fatta su un campione di un certo **materiale**.

Id_xrf e **Nprova** sono entrambi identificativi dell'analisi, in particolare **Id_xrf** è proprio la chiave primaria ed è formato dalla concatenazione di un numero progressivo + "AN" + l'anno dell'analisi (es. 25AN16), mentre **Nprova** riassume un po' i dati salienti dell'analisi e a sua volta è formato dal materiale + data_analisi + id_xrf (es. AT12_COTTO_P_051115_302AN15).

La relazione con la tabella **Fornitore** è tra le più banali: si riferisce al fornitore del materiale. Un fornitore fornisce uno o più materiali. Un materiale è ricevuto da uno e un solo fornitore.

Gli attributi di **Fornitore** sono il suo nome e i suoi recapiti (**figura 3.3**).

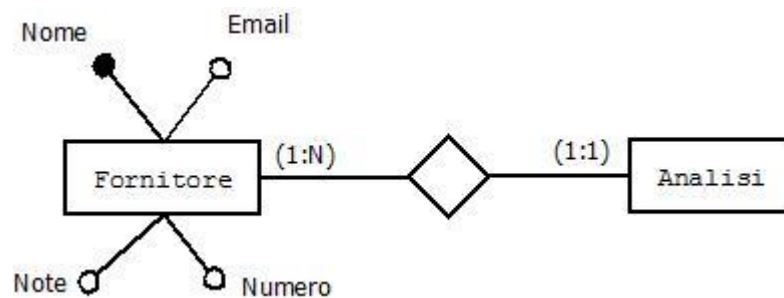


Figura 3.3: Relazione Fornitore - Analisi

La relazione *Tag - Analisi* ha anch'essa principalmente scopo descrittivo.

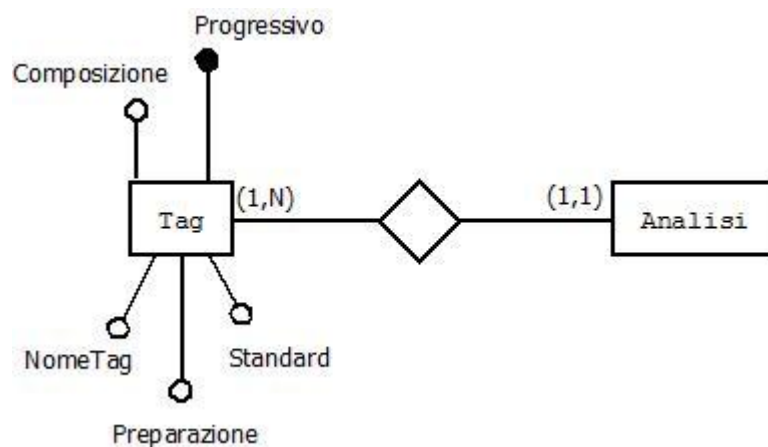


Figura 3.4: Relazione Tag - Analisi

Gli attributi di **Tag** hanno lo scopo di descrivere il processo di preparazione del campione sottoposto all'analisi.

Un tag deve caratterizzare una o più analisi (**figura 3.4**). Un' analisi ha un solo tag caratterizzante il suo materiale.

La **Tabella_Elementi** è l'intera tavola periodica degli elementi⁶ chimici, con un nome e la sigla (es. "K, Potassio", "Ca, Calcio")

⁶ **Tavola periodica degli elementi:** è lo schema con il quale vengono ordinati gli elementi chimici sulla base del loro numero atomico e del numero di elettroni presenti nell'orbitale atomico più energetico.

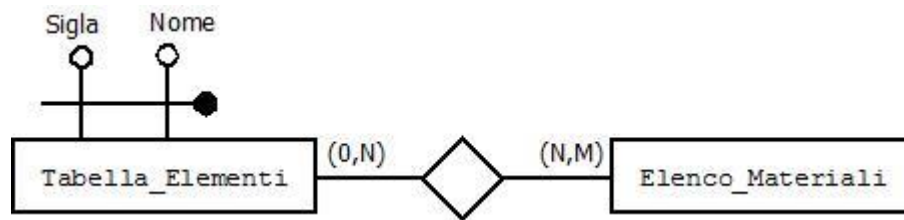


Figura 3.5: Relazione Tabella_Elementi – Elenco_Materiali

Un elemento può essere contenuto in uno o più materiali. Un materiale è formato da più elementi (**figura 3.5**).

Per facilitare la comprensione, e anche il compito al programmatore e a utenti meno esperti, è stata introdotta un'altra entità al puro scopo chiarificatore. La relazione che c'è tra **Famiglia_Materiali** ed Elenco_Materiale è analoga a quella sopracitata (**figura 3.6**).

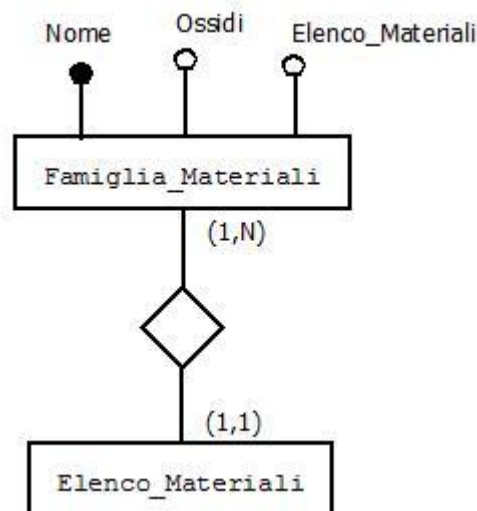


Figura 3.6: Relazione Famiglia_Materiali – Elenco_Materiali

Una **Famiglia_Materiali** (es. Calce, Engobbi) può comprendere uno o più materiali. Un materiale fa parte di una sola famiglia di materiali.

È stato inserito l'attributo ossidi (tipo *integer*), poiché la classificazione in famiglie è stata fatta proprio in base al numero di ossidi contenuti (e quindi

rilevati) nel materiale che sono noti a priori dell'analisi. Questo numero, sarà proprio il numero dei campi che lo spettrofotometro analizzerà e fornirà in output.

Dopo aver chiarito il discorso relativo ai materiali viene di seguito riportata la relazione più scontata: l'analisi è svolta su un campione di un materiale.

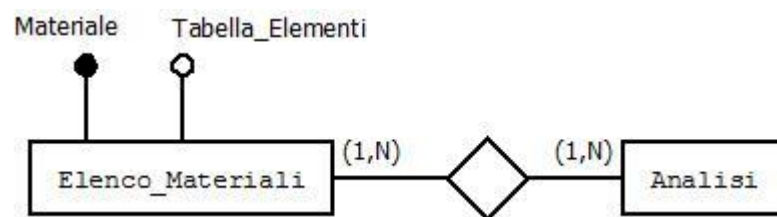


Figura 3.7: Relazione Elenco_Materiali - Analisi

Un materiale (es. APLITE A3, PI_COTTO, DBK12E) può essere sottoposto a una o più analisi. Un'analisi può essere fatta su uno o più materiali (**figura 3.7**). Come accennato prima, l'attributo **Tabella_Elementi**, è la chiave esterna riferita a più elementi che compongono tale materiale.

3.2 Traduzione da schema E/R a progetto logico

Il progetto logico rappresenta la traduzione schematica di quanto proposto sottoforma di E/R comprensivo degli attributi delle entità e delle chiavi (primarie ed esterne).

Le chiavi primarie sono state sottolineate, mentre le chiavi esterne saranno indicate con *FK*.

ANALISI (ID_XREF, NPROVA, DATA_ANALISI, DATA_CAMPIONE, MATERIALE_ID, FORNITORE_ID, RISULTATI, SCOPO_ANALISI_ID, UMIDITÀ, TAG_ID, NOTE)

AK⁷: NPROVA

FK: MATERIALE_ID REFERENCES MATERIALE

FK: FORNITORE_ID REFERENCES FORNITORI

FK: SCOPO_ANALISI_ID REFERENCES SCOPO_ANALISI

FK: TAG_ID REFERENCES TAG

TAG (NOMETAG, PROGRESSIVO, COMPOSIZIONE, PREPARAZIONE, STANDARD)

TABELLA_ELEMENTI (SIGLA, NOME)

ELENCO_MATERIALI (NOME, TABELLA_ELEMENTI)

FK: TABELLA_ELEMENTI REFERENCES TABELLA_ELEMENTI

FAMIGLIA_MATERIALI (NOME, ELENCO_MATERIALI, OSSIDI)

ELENCO_MATERIALI REFERENCES ELENCO_MATERIALI

FORNITORE(NOME, NUMERO, MAIL, NOTE)

SCOPO_ANALISI (DESCRIZIONE)

3.3 Creazione Database

A questo punto verrà fatta una rapida panoramica sulle tipologie di dati assegnati ai campi più rilevanti ai fini della creazione del database in Python.

⁷ **AK**: Chiave primaria alternativa (Alternative Key)

ANALISI

```
class Analisi(models.Model):
    nprova = models.CharField(unique=True, max_length=20)
    data_analisi = models.DateField(null=False)
    data_campione = models.DateField(null=False)
    id_xrf = models.CharField(unique=True, max_length=50)
    materiale = models.ForeignKey(ElencoMateriali)
    fornitore = models.ForeignKey(Fornitore)
    tag = models.ForeignKey(Tag)
    risultati = models.CharField(max_length=100)
    ris_completo = models.CharField(max_length=100000)
    scopo_analisi = models.ForeignKey(ScopoAnalisi, default='Altro')
    umidita = models.IntegerField(validators=[MinValueValidator(0), MaxValueValidator(100)], null=True)
    note = models.CharField(max_length=100, null=True, blank=True)
```

Questa classe identifica un'analisi. Per questo motivo contiene i dati e la descrizione di tutto ciò che la caratterizza.

Il significato di ogni attributo è stato già analizzato in precedenza, di seguito si farà riferimento unicamente alla tipologia di dato associatagli.

Oltre a tutti i *CharField*⁸ che sono semplici campi di testo, sono presenti **materiale**, **fornitore**, **tag** e **scopo_analisi** che sono *ForeignKey* poiché si riferiscono alla chiave della loro tabella, in cui sono presenti tutte le istanze possibili per quella data entità. **Nprova** e **id_xrf**, hanno settato **unique = True**⁹, perché dovranno identificare l'analisi in modo univoco. Mentre **umidità** è sottoposto al controllo di **validators**¹⁰ che convalida il campo solamente se viene inserito un numero intero compreso tra 0 e 100, essendo una percentuale [6].

ELENCO MATERIALI

Andando in ordine tra gli attributi di *Analisi* la prima entità che troviamo è **materiale** che referencia la tabella *ElencoMateriali*.

⁸ **CharField**: Tipo di dato Char, accetta una stringa in input.

⁹ **Unique=True**: Controllo sul database che accetterà un solo campo con un determinato valore, non accetta duplicati.

¹⁰ **Validators**: controllo sul valore immesso

```
class ElencoMateriali(models.Model):
    nome = models.CharField(unique=True, max_length=30, null=False)
    TabellaElementi = models.ManyToManyField(TabellaElementi)
```

ElencoMateriali ha il nome del materiale, e poi un `ManyToManyFields` [7] che si riferisce a *TabellaElementi*. Questo significa che ogni materiale può avere, anzi ha più elementi. In pratica traduce quella che è una relazione $N:N^{11}$ nello schema E/R.

Amministrazione Django Benvenuto, asd. Visualizza il sito / Modifica password / Annulla l'accesso

Pagina iniziale > Italgraniti > Elenco Materiali > Aggiungi elenco materiali

Aggiungi elenco materiali

Nome: DBK12E

TabellaElementi:

TabellaElementi disponibili

Filtro

K, potassio
Ba, Bario
Al, Alluminio
Si, Silicio

TabellaElementi scelti

Na, Sodio
Mg, Magnesio
Zn, Zinco
Br, Bromo
Li, Litio

Scegli tutto Elimina tutti

Tieni premuto "Control", o "Command" su Mac, per selezionarne più di uno.

Salva e aggiungi un altro Salva e continua le modifiche Salva

Figura 3.8: Input *ElencoMateriali*

Un esempio di questa tabella è rappresentato dal materiale *DBK12E* che è formato dagli elementi *Sodio*, *Magnesio*, *Potassio*, *Calcio*, *Zirconio*, *Ferro*, *Zinco*, *Alluminio*, *Silicio*, *Fosforo*, *Piombo* e *Bario* (figura 3.8).

¹¹ **N:N** : Relazione molti a molti

TABELLA ELEMENTI

Proseguendo verticalmente la descrizione delle entità incontrate abbiamo ora *TabellaElementi*.

```
class TabellaElementi(models.Model):  
    sigla = models.CharField(unique=True, max_length=10, null=False)  
    nome = models.CharField(unique=True, max_length=20, null=False)
```

Il suo scopo è stato esplicitato dall'esempio del paragrafo precedente, e i suoi campi sono semplicemente Sigla e Nome di tutti gli elementi della tavola periodica (es. K, Potassio). Poiché l'inserimento degli elementi sarà un'operazione da svolgere solamente al primo utilizzo, sarà fattibile solamente tramite interfaccia di Admin di Django (**figura 3.9**).

Amministrazione Django Benvenuto, asd. Visualizza il sito / Modifica password / Annulla l'accesso

Pagina iniziale > Italgraniti > Tabella Elementi > Aggiungi tabella elementi

Aggiungi tabella elementi

Sigla:	<input type="text" value="Ca"/>
Nome:	<input type="text" value="Calcio"/>

Figura 3.9:Input *Tabella_Elementi*

FAMIGLIA MATERIALE

Riportando l'attenzione sull'argomento "materiali", è necessario precisare che per gestire in modo più facile e preciso questa tabella è stato necessario implementare anche l'entità *FamigliaMateriale* che funziona in similmodo a *TabellaElementi*.

```
class FamigliaMateriale(models.Model):  
    nome = models.CharField(max_length=30)  
    ElencoMateriali = models.ManyToManyField(ElencoMateriali)  
    ossidi = models.IntegerField(null=True)
```

Ad ogni famiglia è stato ovviamente assegnato un nome, un numero di ossidi che caratterizza tale famiglia, e i materiali che ne fanno parte, scegliendo tra quelli presenti nell'*ElencoMateriali* (figura 3.10).

The screenshot shows the Django Admin interface for the 'Famiglie Materiali' app. The page title is 'Aggiungi famiglia materiale'. The form includes a 'Nome:' field with the value 'Engobbi'. Below it is a section for 'ElencoMateriali' with two lists: 'ElencoMateriali disponibili' (containing 'PK7525') and 'ElencoMateriali scelti' (containing 'Aplite A3' and 'DBK12E'). There are buttons for 'Scegli tutto' and 'Elimina tutti'. Below the lists is a note: 'Tieni premuto "Control", o "Command" su Mac, per selezionarne più di uno.' At the bottom is an 'Ossidi:' field with the value '10'. The page footer has three buttons: 'Salva e aggiungi un altro', 'Salva e continua le modifiche', and 'Salva'.

Figura 3.10: Input Famiglia_Materiali

Come in *TabellaElementi* anche qui l'*ElencoMateriali* è un *ManyToManyField*.

TAG, FORNITORE e SCOPO ANALISI

Per quanto riguarda “tag”, “fornitore” e “scopo analisi” poiché le tabelle risultano molto intuitive e simili tra loro, si è deciso di spiegare le entità in una sola volta.

```
class Fornitore(models.Model):
    nome = models.CharField(max_length=30)
    numero = models.CharField(max_length=15, null=True, blank=True)
    email = models.CharField(max_length=30, null=True, blank=True)
    note = models.CharField(max_length=100, null=True, blank=True)

class Tag(models.Model):
    idtag = models.CharField(max_length=30)
    progressivo = models.CharField(max_length=15, null=True, blank=True)
    composizione = models.CharField(max_length=30, null=True, blank=True)
    preparazione = models.CharField(max_length=100, null=True, blank=True)
    standard = models.CharField(max_length=100, null=True, blank=True)

class ScopoAnalisi(models.Model):
    descrizione = models.CharField(max_length=60)
```

Hanno tutte e tre solo dei *CharField* i quali hanno uno scopo descrittivo delle relative tabelle. Dato l'utilizzo e l'aggiornamento più frequente rispetto alle tabelle dei materiali, è stato implementato un piccolo form per aggiungere nuove istanze direttamente tramite l'interfaccia web.

Inserisci Nuovo Fornitore

Nome Fornitore	<input type="text"/>
Numero Fornitore	<input type="text"/>
Email Fornitore	<input type="text"/>
Note Fornitore	<input type="text"/>

Salva

Figura 3.11 Form Input Fornitore

Come scritto precedentemente gli altri due form [5] sono analoghi a quello riportato qui sopra (**figura 3.11**).

3.4 Dalla realtà al database

In questa sezione ci si soffermerà a riflettere sul motivo di alcune scelte fatte, che risulteranno ancora più chiare dopo il prossimo capitolo in cui si farà riferimento anche al codice scritto.

Le scelte più delicate da fare sono state sull'implementazione dei materiali, ma dopo alcuni colloqui con il committente l'attuazione di questa soluzione è stata ovvia. Come riportato in precedenza, il problema iniziale era risutato essere più comunicativo che implementativo.

Un'altra scelta delicata è stata quella relativa ai risultati, che è infine ricaduta sulla creazione di due campi per mantenere in memoria il risultato completo con tutte le analisi, e un campo per il suo identificativo (ricavato proprio dal risultato completo). L'alternativa presa in considerazione inizialmente era di creare una tabella anche per i risultati, e una volta inserito in input il risultato completo, estrapolare ogni campo e salvarlo nel suo database. Questa scelta è stata scartata perché in base al tipo di materiale analizzato varia il numero degli ossidi in esso contenuto, e parlando di database, sarebbe variato il numero degli attributi da salvare per ogni analisi. Discorso troppo complesso, e soprattutto, visto che avere il valore di ogni ossido presente nei materiali non è strettamente necessario, poco utile..

È stato quindi deciso di tenere salvato in `ris_completo` l'intera stringa grezza contenente tutte le righe di ogni singola analisi, e quando sarebbe tornato utile avere o un singolo campo, o tutte le righe, si sarebbe richiamata tutte le volte la funzione che ripulisce la stringa. In questo modo le analisi inserite in origine, anche se in modo grezzo, saranno per sempre salvate in un campo e mai modificate, così da poter sempre effettuare controlli in caso di anomalie, poiché queste non verranno mai modificate, ma solo consultate, e manipolate, ma al solo scopo di stamparle.

Per capire meglio il ragionamento sopra riportato, quello che lo spettrofotometro produce appare così:

Figura 3.12 Display results precedente

Copiato ed incollato nel nuovo software, non risulta altro che un'unica stringa lunghissima, senza caratteri di fine riga, piena di tabulazioni e campi inutili illustrata in **figura 3.13**.

Nr	Ident	Seq	Time	Pos	UTC	Username	Norm F	C (Na2O)	Unit (Na2O)	C (MgO)	Unit (MgO)	C (Al2O3)	Unit (Al2O3)	C
(S102)	Unit (S102)													
1	AT12_COTTO_P_FI8715_150216_417AN16_A		16-Feb-2016 13:29:29		UTC+01:00	1 of 1	1.262	6.275	0.104	16.420	72.876	3.254	0.569	0.187
2	AT12_COTTO_P_FI8715_150216_417AN16_B		16-Feb-2016 13:42:46		UTC+01:00	1 of 1	1.252	6.614	0.160	16.574	72.441	3.166	0.563	0.176
3	AT12_COTTO_P_FI8715_150216_417AN16_C		16-Feb-2016 13:56:01		UTC+01:00	1 of 1	1.240	6.795	0.142	16.440	72.434	3.150	0.562	0.173
4	AT12_COTTO_P_FI8715_150216_417AN16_AVG		16-Feb-2016 13:56:01		UTC+01:00	Ave of 3	1.252	6.561	0.136	16.478	72.583	3.190	0.565	0.179
5	AT12_COTTO_P_FI8715_150216_417AN16_SDEV		16-Feb-2016 13:56:01		UTC+01:00	SDev of 3	0.011	0.264	0.029	0.084	0.253	0.056	0.004	0.007

Figura 3.13: Stringa risultato

Dopo il passaggio dalla funzione di pulizia diventa molto più leggibile e interpretabile come è ben visibile dalla **figura 3.14** (sempre da occhi esperti):

Figura 3.14: Template Tabella risultati

Abbiamo descritto le prime decisioni che si possono ancora definire di tipo progettuali.

Capitolo 4

Implementazione

Dopo aver accennato in diverse occasioni ad alcune funzionalità richieste dal committente, il lettore dovrebbe essersi fatto ormai un'idea del software che verrà esposto nel cuore di questa tesi.

Si esporranno dettagliatamente le principali funzioni implementate con riferimento al codice.

Si cercherà di seguire il percorso in ordine temporaneo della procedura di inserimento e consultazione del risultato inserito.

Riprendendo i requisiti del software, si ricorda che le motivazioni principali che hanno portato alla sua creazione sono state le esigenze di poter gestire i risultati delle analisi fatte negli anni, in modo semplice, veloce, e soprattutto da un unico software. Questo lavoro ha quindi creato un grosso vantaggio in termini di comodità e risparmio di tempo.

Una volta che l'utente copia il risultato (**figura 3.13**) dal pannello di output del programma proprietario dello spettrofotometro, e lo incolla nel nuovo software dedicato, il risultato sarà completamente a disposizione del software, dal quale potrà essere richiamato, visualizzato o stampato.

Faremo qualche esempio con annesso codice nei successivi paragrafi.

4.1 Inserimento Risultati

Riprendendo l'inserimento del risultato all'interno del software, come è stato già accennato, vengono richiesti altri campi a scopo descrittivo, che non sono ricavabili dal risultato.

Sempre per velocizzare la procedura, è stato implementato un pulsante *Compila*, che ha il compito, appunto, di precompilare tutti i campi che sarebbero

obbligatorie, lasciando all'utente il solo compito di inserire il risultato. Ovviamente questi campi saranno modificabili dal committente, poiché il valore generato automaticamente, in alcuni casi può essere sbagliato. Vengono di seguito mostrati nel dettaglio:

- *Nprova*: è il numero identificativo dell'analisi. Il valore generato automaticamente dal pulsante *Compila* comprende un numero progressivo concatenato alla stringa 'AN' (analisi) e all'anno attuale;
- *Data_analisi* e *Data_Campione*: la prima come si può intuire è riferita alla data in cui è stata fatta l'analisi, mentre la seconda risale alla creazione del campione. "Compila" li imposta automaticamente entrambi sulla data corrente. Poiché la trascrizione dal software dello spettrofotometro al nuovo software, può non essere immediata, queste verranno spesso modificate;
- *Id_Xrf*: analogamente al *Nprova*, rappresenta un sunto dell'intera analisi, per fornire qualche informazione in più a prima lettura. È formato dal *materiale* + *data_analisi* + *Nprova*;
- *Materiale*: da scegliere tra tutti i materiali presenti.
Nota bene: si ricorda che ogni materiale appartiene ad una famiglia di materiali diversa. Ciò significa che avrà caratteristiche diverse, e in base alla famiglia a cui appartiene, verrà invocata una funzione specifica per poter gestire il risultato in maniera corretta;
- *Risultati*: è solamente un identificato dei risultati prodotto, che viene estrapolato contemporaneamente al salvataggio nel database dei risultati.
- *Scopo_Analisi e Fornitori*: anch'essi da scegliere tra le opzioni già inserite nel DB, hanno meramente scopo descrittivo;
- *Tag*: questo campo, dal punto di vista implementativo è uguale ai due precedenti, ma per l'operatore ha un ruolo ben più importante, poiché contiene le informazioni su come è stata prodotto tale campione di materiale.
- *Umidità*: è un valore intero obbligatorio che deve essere inserito dall'operatore.
- *Note*: semplice campo descrittivo, utile per appuntare qualche dettaglio in caso di necessità.

Di seguito viene riportato il codice di come sono stati salvati, e compilati questi campi:

```
if 'compila' in request.POST:
    # tmp è l'indice della prossima analisi che inserisco
    tmp = count() + 1
    # auto costruisce il nprova e lo assegna
    nprova = auto(tmp)
    oggi = date.today()
    data_an_auto = oggi.strftime('%d/%m/%y')
    data_ca_auto = oggi.strftime('%d/%m/%y')
    id_xrf = xrf(nprova,data_an_auto)

    newanal = Analisi(nprova=nprova, data_analisi=data_an_auto, data_campione=data_ca_auto, id_xrf=id_xrf)

    return render_to_response('italgraniti/addanalisi.html',
                              {'nprova': nprova, 'data_an_auto': data_an_auto, 'id_xrf': id_xrf,
                               'data_ca_auto': data_ca_auto, 'materiale': materiale,
                               'fornitore': fornitore, 'tag': tag, 'scopoanalisi': scopoanalisi},
                              context_instance=RequestContext(request))
```

Questo accade nel caso venga cliccato il pulsante *Compila*, altrimenti viene direttamente effettuata la procedura di salvataggio:

4.2 Salvataggio

La procedura di salvataggio è implementata dal seguente codice:

```
def addanalisi(request):
    template = 'italgraniti/addanalisi.html'
    analisi = Analisi.objects.all()
    materiale = ElencoMateriali.objects.all()
    fornitore = Fornitore.objects.all()
    tag = Tag.objects.all()
    scopoanalisi = ScopoAnalisi.objects.all()

    if 'salva' in request.POST:
        forni = request.POST.get('fornitore')
        t = request.POST.get('tag')
        scopo = request.POST.get('scopoanalisi')
        mat = request.POST.get('materiale')
        nprova = request.POST.get('nprova')
        note = request.POST.get('note')
        umidita = request.POST.get('umidita')
        ris_completo = request.POST.get('ris_completo')
        risultati = get_id_ris(ris_completo)
        fornitore = Fornitore.objects.get(id=forni)
        tag = Tag.objects.get(id=t)
        scopoan = ScopoAnalisi.objects.get(id=scopo)
        materiale = ElencoMateriali.objects.get(id=mat)
        data_ca = datetime.strptime(request.POST['data_analisi'], "%d/%m/%y")
        data_an = datetime.strptime(request.POST['data_campione'], "%d/%m/%y")
        id_xrf = str(materiale) + "_" + (request.POST.get('id_xrf'))

        newanal = Analisi(nprova=nprova, data_analisi=data_an, data_campione=data_ca, id_xrf=id_xrf,
                          materiale=materiale, risultati=risultati, ris_completo=ris_completo,
                          umidita=umidita, scopo_analisi=scopoan, fornitore=fornitore, tag=tag, note=note, )
        newanal.save()
        return redirect('elencoanalisi')
```

Al termine di questa funzione si viene reindirizzati alla pagina in cui sono elencate tutte le analisi già effettuate (**figura 4.1**), per controllare che l'operazione appena eseguita sia andata a buon fine..



NPROVA	DATA ANALISI	DATA CAMPIONE	ID XRF	MATERIALE	RISULTATI	SCOPO ANALISI	FORNITORE	TAG	UMIDITA
1AN2016	01/03/16	29/02/16	E310_PP_180316_1AN2016	E310_PP	E310_PP_010216_W5_392AN16	Controllo	Sasil	ML1 LOOSE	20
2AN2016	10/03/16	09/03/16	Aplite A3_180316_2AN2016	Aplite A3	APLITE_PP25_260116_W4_381AN16	Controllo	Sasil	DBK12E	60
3AN2016	18/03/16	18/03/16	FTPD_PP5_180316_3AN2016	FTPD_PP5	FTPD_PP5_130116_358AN16	Controllo	Kerakoll	DBK12E	10

Figura 4.1: Template ElencoAnalisi

4.3 Visualizzazione dati

Una volta posizionati nella pagina di riepilogo, cliccando su un’analisi si potranno vedere tutti i risultati. Da questa pagina, cliccando sul link, viene passato alla view associata al template che apparirà tutto il risultato grezzo, e per la “prima” volta viene ripulito, e salvato in una apposita struttura dati permettendone la stampa pulita ed ordinata (**Figura 3.14**) grazie a questo codice:

```
def create_dict_ceramic(risraw):

    risraw = risraw[242:] #rimuove la riga di intestazione
    risraw = risraw.replace("\t", " ") #sostituisce le tabulazioni con spazi
    list_raw = []
    p = 0
    p_max = famiglia.ossidi
    raw = []
    risraw = risraw.replace("ppm", "%")

    for x in risraw:
        if (p < p_max):
            raw.append(x)
        if (x == "%"):
            p += 1
        if (p == p_max):
            r = "".join(raw)
            r = r.replace("?", "AVG")
            r = r.replace("%", "")
            list_raw.append(str(r))
            raw = []
            p = 0 #se arrivo al 9° '%' sono a fine riga, resetto il contatore dei % e la riga

    list_raw[len(list_raw) - 1] = list_raw[len(list_raw) - 1].replace("AVG", "SDEV")
    list = []
    list_raw_value = list_raw[:len(list_raw) - 2]
    list_raw_stat = list_raw[len(list_raw) - 2:]

    #cancellazione campi non utili dalle righe relative ai campioni
    for raw in list_raw_value:
        list_tmp = raw.split()
        del list_tmp[2:11]
        list.append(list_tmp)

    #cancellazione campi non utili dalle righe relative alle statistiche
    for raw in list_raw_stat:
        list_tmp = raw.split()
        del list_tmp[2:8]
        list.append(list_tmp)

    return list
```

Vi è una funzione di questo tipo per ogni famiglia di materiali. Non è sicuramente la soluzione più portatile possibile, ma affermando che le 4 famiglie di materiali gestite racchiudono circa il 90% dei materiali presenti, e il 100% dei materiali fino ad ora analizzati (e quindi di cui erano presenti esempi da poter studiare), può rappresentare una buona soluzione temporanea.

Quindi in base alla famiglia alcuni parametri cambieranno.

In ordine le operazioni effettuate sulla stringa grezza (*ris_completo*) sono:

- rimozione della prima riga di intestazione (contenente i nomi delle colonne);
- sostituzione di tutte le tabulazione con dei semplici spazi;
- gestione eccezioni: in questo caso al posto del simbolo % è stato trovato 'ppm', e quindi avrebbe sballato la successiva funzione;
- divisione della stringa rimasta per righe: questa è la funzione principale di questa operazione: ha il compito di dividere la stringa rimasta in tante righe quante sono le analisi, in modo preciso. Non essendo presente nessun carattere di fine riga, nessun campo fisso che indicasse l'inizio di ogni nuova riga, si è risaliti ad uno stratagemma: per ogni famiglia di materiali è stato richiesto di inserire anche il numero di ossidi in essa contenuti. L'analisi prodotta, calcola proprio la percentuale di ognuno di questi ossidi. L'ultimo carattere di ogni riga è stato provato che sia sempre il simbolo '%'. Quindi scansionando l'intera stringa, si è riusciti a separare la stringa andando 'a capo' ogni n-esimo '%', dove n è il numero di ossidi ricavato dalla famiglia del materiale in questione. Fortunatamente anche quando i valori sono nulli (es. nella deviazione standard) è sempre presente il simbolo percentuale, rendendo sempre efficace questa soluzione.

Ciò che esce da questa procedura è una lista di n liste, rappresentante le righe delle analisi di cui le ultime due righe indicano la media dei primi $n-2$ campioni, e la deviazione standard;

- creazione dell'identificativo: la macchina aggiunge automaticamente una lettera progressiva alla fine di ogni identificativo delle analisi, ma alle ultime due righe, non essendo delle analisi, bensì le statistiche su di esse, le attribuisce la desinenza '_?'. Il problema sta nel fatto che dovendo passare

come parametro l'identificativo in coda all'url, per mostrare in seguito il risultato specifico, verrebbe scambiato come carattere di fine stringa dal parser¹² che lo scansiona alla ricerca di un match nel file degli url, non trovando nulla. Quindi prima di inviarlo insieme all'url, semplicemente viene sostituito da `'_AVG'` il primo, e da `'_SDEV'` il secondo;

- eliminazione di alcuni campi inutili. Generando un numero di campi differente le righe relative alle analisi, dalle ultime due, questa operazione viene fatta in due cicli differenti;

Il risultato ora ottenuto viene ritornato ed è pronto per essere stampato su */ris_spec/identificativo*.

4.4 Libreria Openpyxl

Da questa schermata è possibile esportare tale risultato su excel. Nel file che viene generato, vengono rappresentati solamente i valori medi. Nel caso questi valori non rispettino i valori standard (inseriti, ed aggiornati dall'operatore sullo stesso file base di excel), verrà evidenziata l'anomalia.

Per creare questo file è stata utilizzata la libreria *openpyxl* di Django che permette la creazione, e (come in questo caso) la modifica di file excel già esistenti. Si è deciso di caricare un file già esistente, poiché questa funzione ha solamente lo scopo di mostrare ai capi dell'azienda,, attraverso un'impaginazione più user friendly (anche grazie al supporto di grafici, spiegati più avanti) una panoramica sui risultati. Potendo quindi creare lo scheletro del file di excel base manualmente, si ha molta più libertà. Successivamente, per ogni risultato che si vuole stampare verrà aperto il file base (senza i dati), verranno scritti i dati, e successivamente salvato con l'identificativo dell'analisi stampata [8].

¹² **Parser:** software in grado di effettuare l'analisi sintattica di un testo rispetto ad un determinato linguaggio.

```

from openpyxl import load_workbook

def crea_excel(analisi, avg):
    wb = load_workbook('output/file_base.xlsx')
    ws = wb.active
    stat = avg[3:]
    id = str(analisi.risultati)
    id.replace("_AVG", "")

    ws['H5'] = str(analisi.nprova)
    ws['C7'] = str(analisi.data_campione)
    ws['C8'] = id
    ws['C9'] = str(analisi.fornitore)
    ws['C5'] = str(analisi.materiale)
    ws['E33'] = str(analisi.data_analisi)

    for row, entry in enumerate(stat, start=20):
        ws.cell(row=row, column=4, value=entry)

    id = str(analisi.nprova)
    est = ".xlsx"
    path = "output/"
    filename = path+id+est

    wb.save(filename)
    return 1

```

Queste poche righe di codice all'interno della funzione invocata quando viene cliccato “Stampa” non fanno altro che aprire il file (scritto manualmente) *'file_base.xlsx'* come workbook¹³, e dopo aver aggiustato e pulito ulteriormente la stringa da stampare, vengono scritti alcuni campi descrittivi in celle selezionate manualmente, per poi scrivere attraverso il ciclo, tutti i dati relativi al risultato. Fatto ciò si costruisce la stringa che rappresenterà il nome del file e lo si salva.

Una delle funzioni richieste con più enfasi, data la sua importanza, è stata quella di poter consultare le analisi eseguite, ma in seguito ad una personale selezione (**figura 4.2**). Questo è stato realizzato grazie all'implementazione di alcuni filtri

¹³ **Workbook:** insieme di fogli di excel

tramite i quali si può scegliere di visualizzare solamente i risultati delle analisi che soddisfano tali filtri, sul materiale e sulla data [9].

RIASSUNTO RISULTATI										
MATERIALE	E310_PP	DATA >=	01/03/16	E <= DI	31/03/16	Visualizza				
Identificativo	Norm F	Na2O	MgO	Al2O3	SiO2	K2O	CaO	TiO2	Fe2O3	H2O
E310_PP_100316_W5_1AN16_AVG	1,252	6,561	0,136	16,478	72,583	3,190	0,565	0,179	0,308	0,0
E310_PP_170316_W5_22AN16_AVG	1,017	1,622	0,721	15,404	70,586	7,351	1,442	0,340	0,414	2,120
E310_PP_280316_W5_30AN16_AVG	1,014	0,114	0,404	22,305	65,905	2,404	0,188	1,515	1,326	5,84

Figura 4.2: Template filtri risultati

Ovviamente il confronto ha senso se le diverse analisi si riferiscono allo stesso campione, magari in date diverse. Per evidenziare tali eventuali discrepanze, lo strumento più utile è un grafico, costruibile con il pulsante “Crea Grafico”.

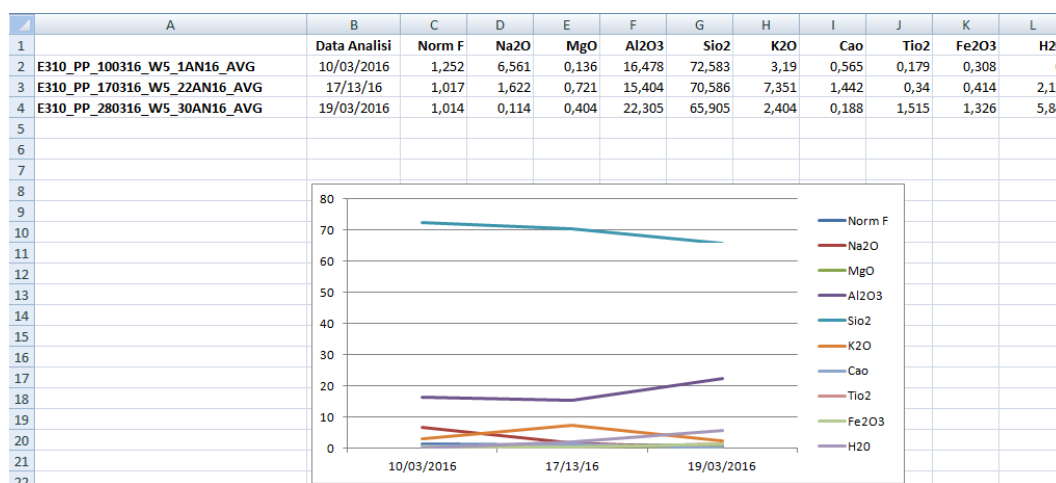


Figura 4.3: Output grafico

Dal grafico si evincono i cambiamenti di ogni ossido contenuto nel campione, e la sua variazione nelle date selezionate dall’utente (**figura 4.3**).

Tale grafico è realizzato sempre grazie a poche e semplici istruzioni della libreria *openpyxl*, per poter lasciare la libertà all’utente in seguito di aggiungere sue note personali sul file, o come in questo caso, poter creare un altro grafico a supporto del precedente, eliminando i valori molto fuori scala. Poiché per molti ossidi una

variazione (che in un grafico con valori molto maggiori non è facile notare) anche solo di una unità può essere molto significativa, questa operazione risulta molto utile.

Queste sono state le maggiori funzionalità implementate a supporto delle analisi. È stato aggiunto un menù anche per il DataBase che comprende un riepilogo delle tabelle presenti in “database” e la possibilità di inserire nuove istanze attraverso interfaccia web.

Operazioni ugualmente realizzabili tramite l'apposita interfaccia di amministrazione di Django, che è stata ugualmente linkata, ma che si cerca di utilizzare il meno possibile, dando la possibilità di intervenire direttamente sul database.

Conclusioni e migliorie

In questa tesi si è cercato di descrivere il lavoro effettuato, enfatizzando in particolar modo come è stato possibile abbassare enormemente le perdite di tempo presenti prima dell'utilizzo di tale software, nonostante la sua semplicità di realizzazione, e si spera, di utilizzo.

Ne ha guadagnato anche la sicurezza, avendo automaticamente ogni dato immediatamente online, mentre prima veniva effettuato un backup periodico sul server aziendale, ora ogni singolo dato inserito nel programma è salvato istantaneamente.

Sicuramente il lavoro da svolgere è ancora molto. Sono molti gli aspetti che si possono migliorare, primo su tutti la gestione di famiglie di materiali multiple, con conseguente procedura di ottenimento dei risultati personalizzati per famiglia.

Per mostrare al committente quali e quante erano le possibilità di sviluppo, ci si è concentrati più sulla quantità che sulla qualità, lasciando qualche lacuna nella gestione e reazione agli errori. Per queste eventualità è stato lasciato il libero accesso all'interfaccia di amministrazione dalla quale si può intervenire manualmente sul database in caso di errori.

Anche per questo motivo non è stato gestito il discorso dei login e degli accessi, soprattutto perché il programma è destinato ad essere utilizzato da un unico operatore, il solo in grado di utilizzare lo spettrofotometro, e di conseguenza il software a lui dedicato.

Sottolineate le mancanze e possibili migliorie, non rimane che riavvalorare, in antitesi, quanto invece risulta essere un punto di forza del progetto: essere riusciti a racchiudere le stesse funzionalità e non solo, a portata di clic. Le stesse funzionalità che prima erano raggiungibili in seguito a diversi copia-incolla e infiniti passaggi tra excel, access, creazione grafici, creazioni pdf, consultazioni manuali dei dati ottenuti e confronto con standard cartacei.

Tutto ciò ora è fruibile grazie ad una semplicissima e intuitiva interfaccia web.

Bibliografia

- [1] Wikipedia Foundation.
 <http://www.wikipedia.org>.

- [2] Marco Beri. Sviluppare applicazioni web con Django.
 Apogeo. 2009.

- [3] Python.
 <http://www.python.org/>.

- [4] SQLite.
 <http://www.sqlite.org/index.html>.

- [5] Bootstrap Components
 <http://getbootstrap.com/components/>

- [6] Django.
 <https://docs.djangoproject.com/en/1.8/>

- [7] Python.
 <http://docs.python.org/2/>.

- [8] Openpyxl
 <https://openpyxl.readthedocs.org/en/2.3.3/>

- [9] SQLite3 for Python.
 <http://docs.python.org/2/library/sqlite3.html>.

- [10] HTML Tutorial:
 <http://www.w3schools.com/html/default.asp>

[11] Dizionario Treccani.
<http://www.treccani.it/>.