

UNIVERSITÀ DI PISA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Laurea Triennale in Ingegneria Informatica

**Progettazione di un'architettura ad oggetti
e sviluppo in ambiente WordPress di servizi
software per la generazione di prospetti di
laurea**

Relatori:

Prof: Mario G.C.A. Cimino

Dott.ssa: Rose Rossiello

Candidato:

Marco Lampis

Indice

1	Introduzione	3
1.1	Ambiente di sviluppo	3
1.1.1	WordPress	4
1.1.2	Moodle	5
2	Configurazione	6
2.1	filtroesami.json	6
2.1.1	Struttura	7
2.2	votolaurea.json	7
2.2.1	Struttura	7
2.3	mediainf.json	8
2.3.1	Struttura	8
3	Interfaccia Web	9
3.1	Homepage	9
3.2	Gestione Laurea	9
3.3	Modifica Configurazione	11
4	Redesign	14
4.1	Riprogettazione dei file di configurazione	14
4.1.1	Ridenominazione	14
4.1.2	formule-e-messaggi.json	15
4.1.3	filtro-esami.json	16
4.1.4	esami-informatici.json	17
4.2	Riprogettazione del codice	18
4.2.1	GestioneCarrieraStudente	21
4.2.2	ParametriConfigurazione	21
4.2.3	CarrieraLaureando	23
4.2.4	CarrieraLaureandoInformatica	24
4.2.5	SimulazioneVotoLaureando	25
4.2.6	SimulazioneVotoLaureandoInformatica	26
4.2.7	ReportPDFLaureando	26
4.2.8	ReportPDFLaureandoConSimulazione	27
4.2.9	ReportPDFCommissione	28

4.2.10 InviaReportPDFLaureandi	28
4.2.11 GUIParametriConfigurazione	29
4.2.12 GUISimulazioneVotoLaurea	29
A Ringraziamenti	31

Capitolo 1

Introduzione

L’Università di Pisa ha rilasciato nel 2012 il portale *Laureandosi*, un servizio innovativo nato con lo scopo di aiutare gli studenti e il coordinamento didattico nei giorni che precedono la cerimonia di laurea.

Il portale soffriva però di alcune criticità, prima tra tutte la necessità di collaborazione diretta da parte degli studenti per il caricamento dei dati necessari allo svolgimento della cerimonia di laurea.

Perciò, nel 2021, nasce il progetto *Laureandosi2*, con lo scopo di automatizzare completamente il meccanismo di recupero dati senza il coinvolgimento dei laureandi o della commissione didattica. Il servizio consente la generazione di prospetti, la modifica delle configurazioni dei Corsi di Studio, l’invio di email per la laurea e varie ulteriori funzionalità. Ciò è stato possibile grazie all’utilizzo del database di ateneo ESSE3 e della piattaforma Moodle per la gestione dei file.

Laureandosi è stato progettato con un approccio per processi ed implementato dal personale tecnico di analisi dati e dai tesisti in Ingegneria Informatica e Ingegneria Gestionale. Le istruzioni per l’uso dell’applicativo sono fornite via email dal Coordinamento Didattico, a pochi giorni dalla cerimonia di laurea.

1.1 Ambiente di sviluppo

Il progetto vede il cuore del suo sviluppo nell’utilizzo di WordPress, opportunamente configurato mediante l’utilizzo di alcuni plugin come:

- XYZ PHP Code
- WP File Manager

1.1.1 WordPress

WordPress è un Content Management System, ovvero una piattaforma che consente la gestione dei contenuti di un applicativo web, rilasciata sotto licenza open-source. Navigando alla pagina *wp-admin* è possibile accedere al pannello di amministrazione. WordPress utilizza una struttura di tipo modulare in quanto consente l'aggiunta di plugin con funzionalità peculiari, raggiungibili dal menu a sinistra della Figura 1.1.

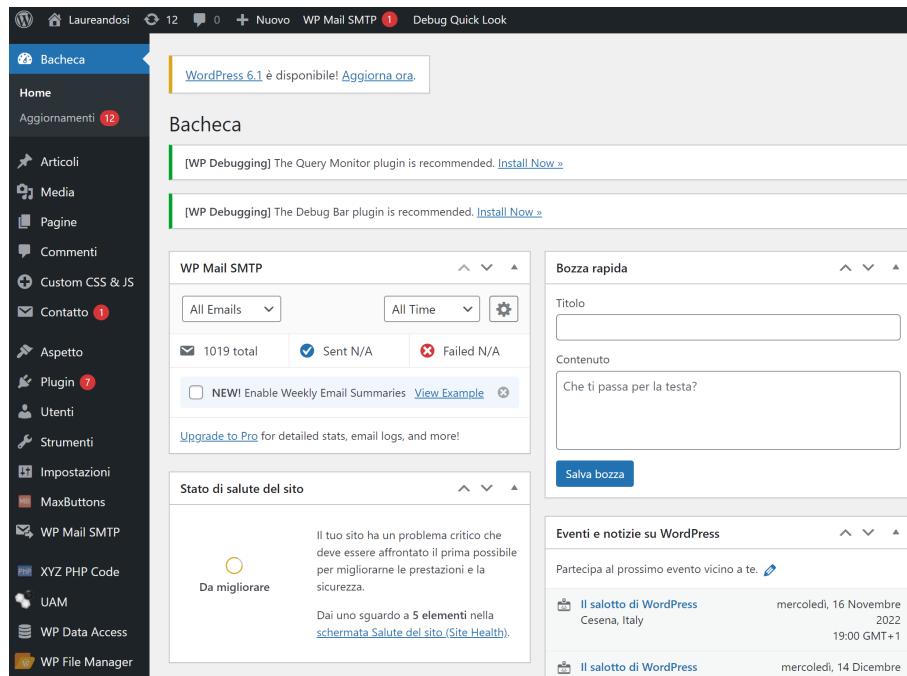


Figura 1.1: Pannello di amministrazione

XYZ PHP Code

XYZ PHP Code è un plugin per WordPress che permette di aggiungere porzioni di codice PHP all'interno delle pagine web. Per ciascuno script è possibile associare un nome che lo identifica e il codice stesso. Quando si ha la necessità di far comunicare snippet di codice differenti, è sufficiente inserirli all'interno della stessa pagina senza utilizzare metodi `require`; ciò è fondamentale per ottenere la modularità del codice. La pagina di modifica è mostrata in Figura 1.2.

Tracking Name	Snippet Short Code	Status	Action
CarrieraLaureandi	[xyz-ips snippet="CarrieraLaureandi"]	Active	
Laureando	[xyz-ips snippet="Laureando"]	Active	
GestioneCarrieraStudente	[xyz-ips snippet="GestioneCarrieraStudente"]	Active	
Modifica-JSON-Test	[xyz-ips snippet="Modifica-JSON-Test"]	Active	
PaginaCoordinamento	[xyz-ips snippet="PaginaCoordinamento"]	Active	
Upgrade	[xyz-ips snippet="Upgrade"]	Active	

Figura 1.2: XYZ PHP Code

WP File Manager

WP File Manager è un plugin per WordPress che permette di navigare il filesystem. Ciò è particolarmente utile per gestire i file di configurazione *json*. E' possibile scaricare, visualizzare, eliminare e rinominare i file. Per modificare un file è necessario cancellare la versione presente e ricaricarla. L'interfaccia è mostrata in [Figura 1.3](#).

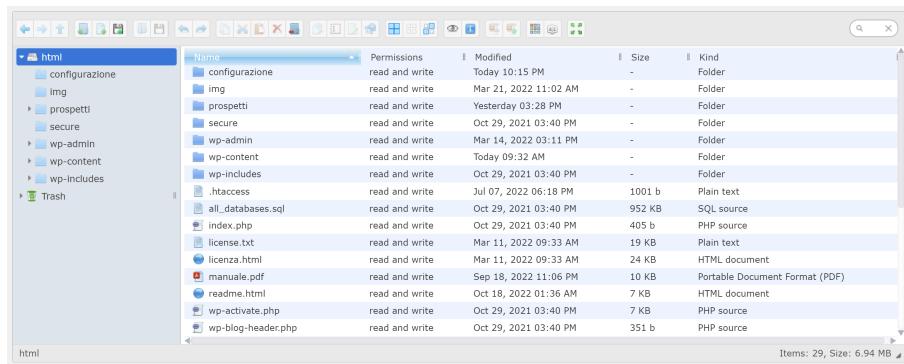


Figura 1.3: WP File Manager

1.1.2 Moodle

Moodle è una piattaforma di Learning Management System (LMS) che permette la gestione dei corsi in forma telematica. Il portale è utilizzato per il caricamento delle tesi da parte dei laureandi, nella [Figura 1.4](#) è mostrato un esempio.

The screenshot shows a Moodle course registration page. The top navigation bar includes links for 'Home', 'Corsi', 'Spazio Corsi di Laurea', 'Lauree DII', 'Cdl T. Ing. Informatica', 'Iscrivimi in questo corso', and 'Opzioni di iscrizione'. The left sidebar has a 'Navigazione' section with links for 'Home', 'Dashboard', 'Pagine del sito', 'I miei corsi', 'Reti Informatiche 2021-22', 'Comunicazioni Numeriche 2020-21', 'Comunicazioni Numeriche 2019-20', and 'Corsi'. The main content area is titled 'Opzioni di iscrizione' and shows course details: 'Titolare: MARIO GIOVANNI COSIMO ANTONIO CIMINO', 'Titolare: ROSE ROSSILO', and 'Insegnante: VITTORIA MARIAGRAZIA DATILLO'. Below this, there's a section for 'Iscrizione spontanea (Student)' with a note 'Non è necessaria una chiave di iscrizione' and a blue 'Iscrivimi' button.

Figura 1.4: Esempio portale Moodle

Capitolo 2

Configurazione

Laureandosi2 utilizza file di configurazione di tipo *json*, memorizzati all'interno della cartella **configurazione**. Nello specifico sono d'interesse: **filtroesami.json**, **votol aurea.json** e **mediainf.json**.

```
  configurazione
    ├── filtroesami.json
    ├── votol aurea.json
    └── mediainf.json
  ├── img
  ├── prospetti
  ├── secure
  ├── wp-admin
  ├── wp-content
  └── wp-includes
```

2.1 filtroesami.json

Il file **filtroesami.json** consente alla commissione di laurea di impostare dei filtri, relativi agli esami, su tutte le matricole (se indicato **"*"**) oppure su singoli laureandi. Il filtro può essere di due tipi:

- **Esami non curriculari:** si indicano nel campo **esamiNonCurr** gli esami che non devono comparire all'interno del prospetto.
- **Esami non in media:** si indicano nel campo **esamiNonInMedia** gli esami che devono comparire all'interno del prospetto ma non devono essere considerati per la media.

2.1.1 Struttura

Di seguito è mostrato un esempio di struttura dati contenuta nel file `filetroesami.json`:

```

1 {
2   "CdL": "T. Ing. Biomedica",
3   "matricole": [
4     {
5       "matricola": "*",
6       "esamiNonCurr": ["esame1", "3"],
7       "esamiNonInMedia": ["esame2", "9"]
8     ]
9 }
```

2.2 votolaurea.json

Il file `votolaurea.json` contiene le informazioni e i criteri attraverso i quali vengono calcolati i voti di laurea per i candidati, riferiti al CdL frequentato.

2.2.1 Struttura

Di seguito è mostrato un esempio di struttura dati contenuta nel file `votolaurea.json`:

```

1 {
2   "CdL": "T. Ing. Informatica",
3   "CdLshort": "t-inf",
4   "votoLaurea": "M*3+18+T+C",
5   "CFUEsamiRichiesti": 177,
6   "Tmin": 0,
7   "Tmax": 0,
8   "Tstep": 0,
9   "Cmin": 1,
10  "Cmax": 7,
11  "Cstep": 1,
12  "Lode": 33,
13  "OggettoMessaggio": "...",
14  "TestoMessaggio": "...",
15  "Note": "..."
```

2.3 mediainf.json

Ogni Corso di Laurea stabilisce autonomamente quali modalità adoperare per assegnare un voto ai propri candidati. Nel caso specifico di Ingegneria Informatica, la commissione assegna da 1 a 7 punti bonus in base alla media dei voti nell'ambito disciplinare *ing-inf 05*. Il file sopra citato si occupa, dunque, di individuare quali sono gli esami informatici.

2.3.1 Struttura

Di seguito è mostrato un esempio di struttura dati contenuta nel file `mediainf.json`:

```
1 [  
2     "FONDAMENTI DI PROGRAMMAZIONE",  
3     "ALGORITMI E STRUTTURE DATI",  
4     "BASI DI DATI",  
5     "RETI LOGICHE",  
6     "CALCOLATORI ELETTRONICI",  
7     "PROGETTAZIONE WEB",  
8     "INGEGNERIA DEL SOFTWARE",  
9     "SISTEMI OPERATIVI",  
10    "RETI INFORMATICHE",  
11    "PROGETTAZIONE DI RETI INFORMATICHE",  
12    "PROGRAMMAZIONE AVANZATA",  
13    "PROGRAMMAZIONE",  
14    "FONDAMENTI DI INFORMATICA I",  
15    "FONDAMENTI DI INFORMATICA II"  
16 ]
```

Capitolo 3

Interfaccia Web

L’interfaccia web di *Laureandosi2* si suddivide in tre pagine, distinte per funzionalità e privilegi di accesso:

- Homepage
- Gestione Laurea
- Modifica Configurazione

3.1 Homepage

La homepage è la pagina in cui l’utente accede quando arriva sul portale. Contiene i riferimenti, per ogni corso di laurea, attraverso i quali i laureandi possono caricare il proprio elaborato di tesi ([Figura 3.1](#)). E’ inoltre presente un link a *Simulazione del voto di laurea*, il cui accesso è limitato agli amministratori, per il caricamento dei dati. Infine, è possibile accedere al manuale per il coordinamento didattico che illustra come adoperare l’applicativo.

3.2 Gestione Laurea

La pagina *Gestione Laurea* consente alla segreteria didattica di inserire le matricole dei laureandi di un appello di laurea, il relativo Corso di Laurea e la data della cerimonia ([Figura 3.2](#)). Una volta inseriti i dati è possibile generare i prospetti mediante il pulsante *Crea Prospetti*, aprirli attraverso *Apri Prospetti* e inviarli per email ai laureandi premendo su *Invia Prospetti*. Infine, è presente il pulsante *Configurazione* attraverso il quale si viene reindirizzati alla pagina di configurazione.

**Accesso esclusivo per il Coordinamento didattico**

- [Manuale](#)
- [Simulazione del voto di laurea](#)

Caricamento di tesi e presentazioni

- [CdL_T_Ing_Biomedica](#)
- [CdL_T_Ing_Elettronica](#)
- [CdL_T_Ing_Informatica](#)
- [CdL_T_Ing_delle Telecomunicazioni](#)
- [CdL_M_Ing_Biomedica_Bionics_Engineering](#)
- [CdL_M_Computer_Engineering_Artificial_Intelligence_and_Data_Engineering](#)
- [CdL_M_Cybersecurity](#)
- [CdL_M_Ing_Elettronica](#)
- [CdL_M_Ing_Robotica_e_della_Automazione](#)
- [CdL_M_Ing_delle Telecomunicazioni](#)

Figura 3.1: Schermata di Homepage

Laureandosi 2 - Gestione Lauree

CdL: <input style="border: 1px solid #ccc; padding: 2px; width: 150px; height: 20px;" type="button" value="Selezione CdL"/> Data Laurea: <input style="width: 100px; height: 25px; border: 1px solid #ccc;" type="text" value="gg/mm/aaaa"/> <input style="border: 1px solid #ccc; width: 20px; height: 25px;" type="button" value="Calendario"/> <input style="background-color: #ccc; color: black; border: 1px solid #ccc; padding: 2px; width: 150px;" type="button" value="Configurazione"/>	Matricole: <div style="border: 1px solid #ccc; width: 150px; height: 150px; margin-bottom: 10px;"></div>	<input style="background-color: #0070C0; color: white; border: 1px solid #0070C0; padding: 2px; width: 150px;" type="button" value="Crea Prospetti"/> apri prospetti <input style="background-color: #0070C0; color: white; border: 1px solid #0070C0; padding: 2px; width: 150px;" type="button" value="Invia Prospetti"/>
---	--	---

Nuova Laurea:

- Seleziona CdL
- Inserisci Matricole separate da spazi/accapo
- Seleziona Data Laurea
- Clicca 'Crea Prospetti' (mostra "prospetti creati" oppure "errore di creazione")
- Clicca 'apri prospetti' (apre la cartella con i pdf)
- Clicca 'Invia Prospetti' (mostra "invia prospetto n. 1/30", "invia prospetto n. 2/30", inviando un prospetto ogni 13 secondi, oppure "errore invio prospetto 5/30")

Figura 3.2: Schermata di Gestione Laurea

3.3 Modifica Configurazione

La pagina *Modifica Configurazione* consente la modifica, da parte della segreteria didattica, dei parametri inerenti a un CdL ([Figura 3.3](#)). Ciascuno di questi parametri viene modificato selezionando dal menu a tendina *Amministrazione del CdL* il Corso di Laurea e scegliendo la sezione di interesse, che può essere:

- *Calcolo e Reportistica*: formule e messaggi inerenti al Corso di Laurea con relativi parametri per il calcolo del voto.
- *Filtro Esami*: consente di filtrare manualmente esami curricolari o non in media per tutte le matricole o per alcune specificate singolarmente.
- *Esami informatici*: consente la modifica e aggiunta dei nomi degli esami che saranno identificati come esami informatici (necessario per il calcolo del bonus di Ingegneria Informatica).

Modifica Configurazione

Figura 3.3: Schermata di modifica

T. Ing. Informatica CARRIERA E SIMULAZIONE DEL VOTO DI LAUREA				
Matricola:				
Nome:	MARCO			
Cognome:				
Email:	m.████████@studenti.unipi.it			
Data:	2022-11-18			
Bonus:	SI			
ESAME	CFU	VOT	MED	INF
FONDAMENTI DI PROGRAMMAZIONE	9	25	X	X
ANALISI MATEMATICA I	12	25	X	
ALGEBRA LINEARE E ANALISI MATEMATICA II	12	30	X	
ALGORITMI E STRUTTURE DATI	6	30	X	X
PROVA DI LINGUA INGLESE B2	3	0		
BASI DI DATI	9	27	X	X
INGEGNERIA DEL SOFTWARE	6	30	X	X
CALCOLO NUMERICO	6	26	X	
FISICA GENERALE I	12	24	X	
ECONOMIA E ORGANIZZAZIONE AZIENDALE	6	29	X	
ELETROTECNICA	6	18	X	
PROGETTAZIONE WEB	6	27	X	X
FONDAMENTI DI AUTOMATICA	9	28	X	
PROGRAMMAZIONE DI INTERFACCIE	6	30	X	
RICERCA OPERATIVA	9	18		
RETI LOGICHE	9	25	X	X
CRITTOGRAFIA	6	28	X	
CALCOLATORI ELETTRONICI	9	21	X	X
ELETTRONICA DIGITALE	9	18	X	
SISTEMI OPERATIVI	9	24	X	X
COMUNICAZIONI NUMERICHE	9	24	X	
RETI INFORMATICHE	9	26	X	X
Media Pesata (M):	25.564			
Crediti che fanno media (CFU):	165			
Crediti curriculari conseguiti:	177/177			
Voto di tesi (T):	0			
Formula calcolo voto di laurea:	$M*3+18+T+C$			
Media pesata esami INF:	25.75			

Figura 3.4: Esempio di un prospetto per lo studente

T. Ing. Informatica CARRIERA E SIMULAZIONE DEL VOTO DI LAUREA					
Matricola:					
Nome:	MARCO				
Cognome:					
Email:	m.████████@studenti.unipi.it				
Data:	2022-11-18				
Bonus:	SI				
ESAME		CFU	VOT	MED	INF
FONDAMENTI DI PROGRAMMAZIONE		9	25	X	X
ANALISI MATEMATICA I		12	25	X	
ALGEBRA LINEARE E ANALISI MATEMATICA II		12	30	X	
ALGORITMI E STRUTTURE DATI		6	30	X	X
PROVA DI LINGUA INGLESE B2		3	0		
BASI DI DATI		9	27	X	X
INGEGNERIA DEL SOFTWARE		6	30	X	X
CALCOLO NUMERICO		6	26	X	
FISICA GENERALE I		12	24	X	
ECONOMIA E ORGANIZZAZIONE AZIENDALE		6	29	X	
ELETROTECNICA		6	18	X	
PROGETTAZIONE WEB		6	27	X	X
FONDAMENTI DI AUTOMATICA		9	28	X	
PROGRAMMAZIONE DI INTERFACCIE		6	30	X	
RICERCA OPERATIVA		9	18		
RETI LOGICHE		9	25	X	X
CRITTOGRAFIA		6	28	X	
CALCOLATORI ELETTRONICI		9	21	X	X
ELETTRONICA DIGITALE		9	18	X	
SISTEMI OPERATIVI		9	24	X	X
COMUNICAZIONI NUMERICHE		9	24	X	
RETI INFORMATICHE		9	26	X	X
Media Pesata (M):	25.564				
Crediti che fanno media (CFU):	165				
Crediti curriculari conseguiti:	177/177				
Voto di tesi (T):	0				
Formula calcolo voto di laurea:	$M*3+18+T+C$				
Media pesata esami INF:	25.75				
SIMULAZIONE DI VOTO DI LAUREA					
VOTO COMMISSIONE (C)		VOTO LAUREA			
1		95.691			
2		96.691			
3		97.691			
4		98.691			
5		99.691			
6		100.691			
7		101.691			

VOTO DI LAUREA FINALE: scegli voto commissione, prendi il corrispondente voto di laurea e somma il voto di tesi tra 1 e 3, quindi arrotonda

Figura 3.5: Esempio di un prospetto per la commissione

Capitolo 4

Redesign

La struttura attuale di *Laureandosi2* è di tipo monolitico: il codice è disposto in un unico file contenente le funzioni atte a rispondere alle necessità dell'utente. Tale scelta si è dimostrata la migliore nelle prime fasi di progettazione, in quanto ha consentito uno sviluppo rapido e organizzato, ma è divenuta una limitazione in seguito al raggiungimento di una fase più matura. Per tale motivo si è operata la scelta di riprogettazione completa della struttura del codice e dei file di configurazione.

4.1 Riprogettazione dei file di configurazione

I file di configurazione soffrivano di alcune scelte implementative che rendevano necessarie, al fine di ottenere le informazioni, lo scorrimento completo dei dati. Ciò causava dei rallentamenti che, seppur non significativi al fine dell'applicativo, rendevano particolarmente complesso il recupero dei dati.

4.1.1 Ridenominazione

I file *json* di configurazione sono stati rinominati al fine di renderli più coerenti con le informazioni riportate. La nuova denominazione utilizza il carattere "-" come separatore.

- `votolaurea.json` → `formule-e-messaggi.json`
- `filtroesami.json` → `filtro-esami.json`
- `mediainf.json` → `esami-informatici.json`

La nuova struttura del portale è dunque la seguente:

```

configurazione
├── formule-e-messaggi.json
├── filtro-esami.json
└── esami-informatici.json
├── img
├── prospetti
├── secure
├── wp-admin
├── wp-content
└── wp-includes

```

Da ora in avanti si farà riferimento ai file seguendo la nuova denominazione.

4.1.2 formule-e-messaggi.json

La struttura del file `formule-e-messaggi.json` è stata modificata rendendo il campo `CdLshort` chiave per gli attributi del Corso di Laurea. La scelta è giustificata dalla necessità di accedere in modo immediato alle informazioni di un CdL senza dover utilizzare un ciclo `for` per scorrere i corsi e individuare quello corretto. Ha, inoltre, portato un miglioramento in termini di prestazioni e pulizia del codice. È stato scelto di utilizzare il campo `CdLshort` per semplicità e comodità di utilizzo.

```

1  {
2      "t-bio": {
3          "CdL": "T. Ing. Biomedica",
4          "CdLshort": "t-bio",
5          "votoLaurea": "(110/27.17)*(M*CFU+T*3)/(CFU+3)",
6          "CFUEsamiRichiesti": 177,
7          "Tmin": 18,
8          "Tmax": 30,
9          "Tstep": 1,
10         "Cmin": 0,
11         "Cmax": 0,
12         "Cstep": 0,
13         "Lode": 33,
14         "OggettoMessaggio": "...",
15         "TestoMessaggio": "...",
16         "Note": "..."
17     },
18
19     "t-ele": {

```

```

20     "CdL": "T. Ing. Elettronica",
21     "CdLshort": "t-ele",
22     "votoLaurea": "2+4*(M*CFU+T*3)/(CFU+3)",
23     "CFUEsamiRichiesti": 177,
24     "Tmin": 18,
25     "Tmax": 33,
26     "Tstep": 1,
27     "Cmin": 0,
28     "Cmax": 0,
29     "Cstep": 0,
30     "Lode": 33,
31     "OggettoMessaggio": "...",
32     "TestoMessaggio": "...",
33     "Note": "..."
34 },
35
36 ...
37
38 }
```

4.1.3 filtro-esami.json

La struttura del file `filtro-esami.json` è stata modificata rendendo il campo `CdLshort` chiave per individuare i filtri da applicare. La scelta è giustificata dalla necessità di accedere in modo immediato alle informazioni di un filtro senza dover utilizzare un ciclo `for` per scorrere i corsi e individuare quello corretto. Ha inoltre portato ad un miglioramento in termini di prestazioni e pulizia del codice. E' stato scelto di utilizzare il campo `CdLshort` per semplicità e comodità di utilizzo.

```

1 {
2   "t-bio": [
3     {
4       "matricola": "*",
5       "esamiNonCurr": [ "", "" ],
6       "esamiNonInMedia": [ "", "" ]
7     },
8     {
9       "matricola": "123456",
10      "esamiNonCurr": [ "", "" ],
11      "esamiNonInMedia": [ "", "" ]
12    }
]
```

```
13 ],
14
15 "t-inf": [
16   {
17     "matricola": "654321",
18     "esamiNonCurr": [ "", "" ],
19     "esamiNonInMedia": [ "", "" ]
20   }
21 ],
22
23 ...
24 }
```

4.1.4 esami-informatici.json

La struttura del file `esami-informatici.json` è stata modificata rendendo il campo `CdLshort` chiave. Questo individua gli esami informatici per il Corso di Laurea indicato. Ciò è stato fatto per rendere i file di configurazione coerenti nell'impostazione e agevolare la lettura degli stessi, in quanto viene utilizzata una struttura comune.

```
1 {
2   "t-inf": [
3     "FONDAMENTI DI PROGRAMMAZIONE",
4     "ALGORITMI E STRUTTURE DATI",
5     "BASI DI DATI",
6     "RETI LOGICHE",
7     "CALCOLATORI ELETTRONICI",
8     "PROGETTAZIONE WEB",
9     "INGEGNERIA DEL SOFTWARE",
10    "SISTEMI OPERATIVI",
11    "RETI INFORMATICHE",
12    "PROGETTAZIONE DI RETI INFORMATICHE",
13    "PROGRAMMAZIONE AVANZATA",
14    "PROGRAMMAZIONE",
15    "FONDAMENTI DI INFORMATICA I",
16    "FONDAMENTI DI INFORMATICA II"
17  ]
18 }
```

4.2 Riprogettazione del codice

Durante l'attività di ristrutturazione del codice, si è operato focalizzandosi su tre caratteristiche principali:

1. **Modularità**: il codice è suddiviso in più moduli distinti e indipendenti tra di loro. Ciò viene fatto al fine di rendere l'organizzazione più efficace e ridurre le attività di manutenzione.
2. **Orientato agli oggetti**: il codice è suddiviso in classi e oggetti che interagiscono tra di loro.
3. **Model-View-Controller**: il codice utilizza il pattern *Model-View-Controller* per la distinzione della responsabilità tra le classi; in particolare, tale scelta è operata con lo scopo di separare la logica di business (i modelli), dalla logica di presentazione dei dati (la vista).

Le classi e gli attributi utilizzano una nomenclatura di tipo **upperCamelCase**.

Classi Modello	
Nome	Descrizione
GestioneCarrieraStudente	Esegue le query verso il database ESSE3
ParametriConfigurazione	Recupera i parametri dai file di configurazione
CarrieraLaureando	Elaborazione dei dati della carriera di un laureando
CarrieraLaureandoInformatica	Elaborazione dei dati della carriera di un laureando in Ingegneria Informatica
SimulazioneVotoLaureando	Esegue la simulazione di voto di un laureando
SimulazioneVotoLaureandoInformatica	Esegue la simulazione di voto di un laureando in Ingegneria Informatica

Classi Vista	
Nome	Descrizione
ReportPDFLaureando	Genera il pdf per un laureando
ReportPDFLaureandoConSimulazione	Genera la pagina di un laureando per la commissione
ReportPDFCommissione	Genera il pdf per la commissione
GUIParametriConfigurazione	Interfaccia per la modifica della configurazione
GUISimulaVotoLaurea	Interfaccia per la realizzazione dei prospetti

Classi Controller	
Nome	Descrizione
InviaReportPDFLaureandi	Invia i report pdf ai laureandi

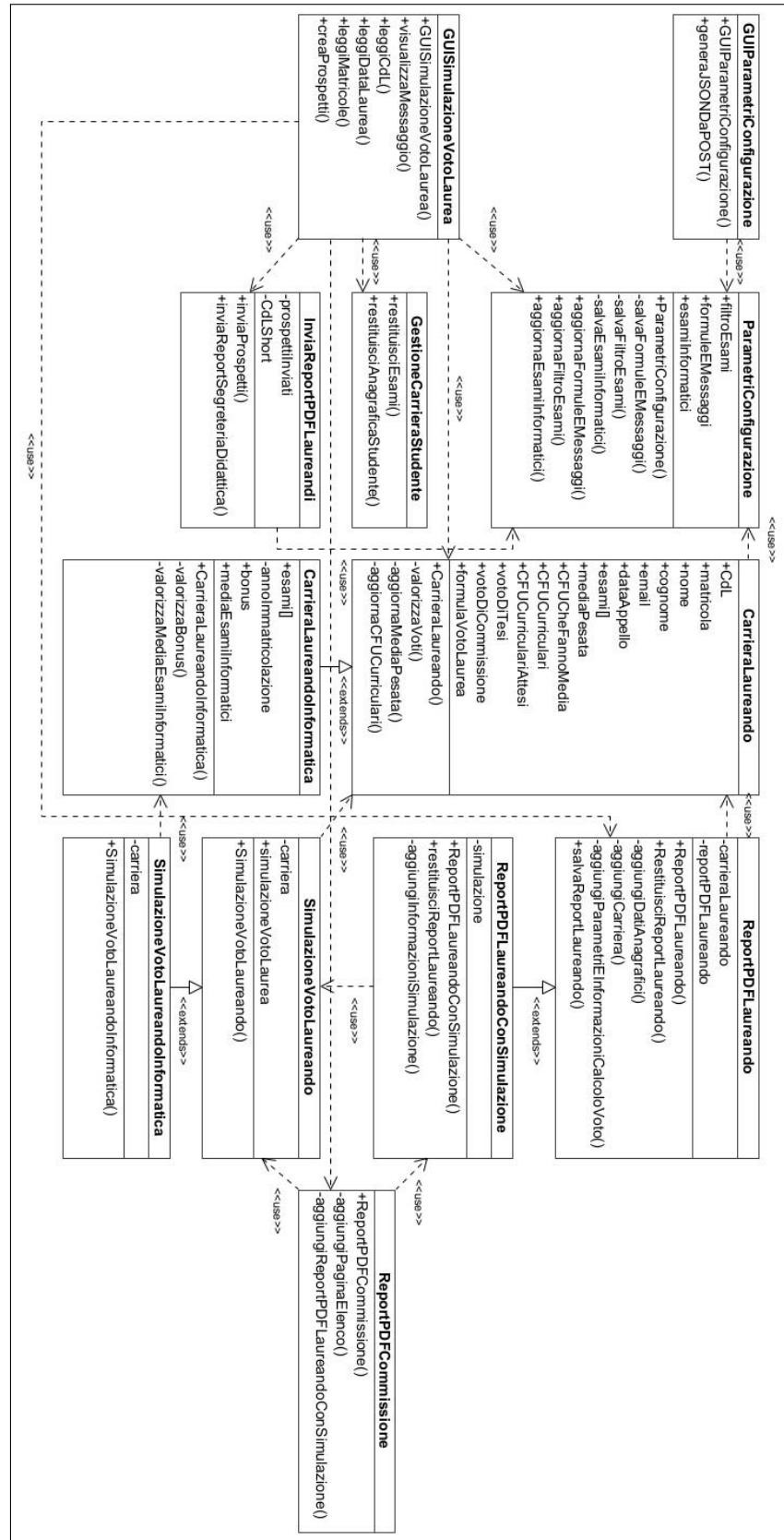


Figura 4.1: Diagramma di relazione tra classi

4.2.1 GestioneCarrieraStudente

La classe `GestioneCarrieraStudente` ha la responsabilità di eseguire le query verso il database di ateneo ESSE3 e restituire i dati ricevuti nel corrispettivo oggetto `json`. Le informazioni non vengono filtrate ma solo rielaborate in misura parziale, limitandosi a correggere la visualizzazione di accenti e caratteri speciali.

Metodi

- `public restituisciEsami(string matricola, boolean salvataggio=false)`
Restituisce un oggetto `json` con la lista di esami relativa a una matricola. Se il parametro `salvataggio` è impostato a `true`, il risultato viene salvato su file in una cartella predefinita (unicamente a scopo di debug durante la progettazione). Di default `salvataggio` vale `false`, in modo da non doverlo specificare nella chiamata in produzione.
- `public restituisciAnagraficaStudente(string matricola, boolean salvataggio=false)`
Restituisce un `json` contenente l'anagrafica di uno studente data la sua matricola. Se il parametro `salvataggio` è impostato a `true`, il risultato viene salvato su file in una cartella predefinita (unicamente a scopo di debug durante la progettazione). Di default `salvataggio` vale `false`, in modo da non doverlo specificare nella chiamata in produzione.

4.2.2 ParametriConfigurazione

La classe `ParametriConfigurazione` ha la responsabilità di recuperare tutte le informazioni presenti dai file di configurazione in formato `json`, consentendone la modifica e il salvataggio. Le impostazioni adottate risiedono nei seguenti file:

1. `formule-e-messaggi.json`, contenente le formule e messaggi per calcolare il prospetto degli studenti.
2. `filtro-esami.json`, file contenente i filtri applicati ai laureandi.
3. `esami-informatici.json`, contenente l'elenco degli esami informatici necessari per il bonus in Ing. Informatica.

Attributi

- `public object formuleEMessaggi`, contiene l'oggetto `json` del file `formule-e-messaggi.json`
- `public object filtroEsami`, contiene l'oggetto `json` del file `filtro-esami.json`

- `public object esamiInformatici`, contiene l'oggetto *json* del file `esami-informatici.json`

E' stato scelto di non utilizzare un unico campo generico "parametri" in quanto, in alcune classi, è necessario accedere al contenuto dei diversi file contemporaneamente.

Metodi

- `costruttore()`

Carica il contenuto dei tre file di configurazione in formato *json* nei rispettivi attributi pubblici.

- `private salvaFormuleEMessaggi()`

Sovrascrive completamente il file `formule-e-messaggi.json` con il contenuto dell'attributo `formuleEMessaggi`.

- `private salvaFiltroEsami()`

Sovrascrive completamente il file `filtro-esami.json` con il contenuto dell'attributo `filtroEsami`.

- `private salvaEsamiInformatici()`

Sovrascrive completamente il file `esami-informatici.json` con il contenuto dell'attributo `esamiInformatici`.

- `public aggiornaFormuleEMessaggi(object formuleEMessaggi)`

Ha come argomento un oggetto *json* contenente le formule e i messaggi relative a un CdL specifico, quindi aggiorna l'attributo `formuleEMessaggi` sostituendo le informazioni e chiama infine `salvaFormuleEMessaggi()` per salvare su file l'oggetto completo modificato. Questo metodo verrà chiamato da `GUIParametriConfigurazione` al momento del salvataggio dei campi inseriti.

- `public aggiornaFiltroEsami(object filtroEsami)`

Ha come argomento un oggetto *json* contenente i filtri per gli esami per un CdL specifico, quindi aggiorna l'attributo `filtroEsami` sostituendo le informazioni e chiama infine `salvaFiltroEsami()` per salvare su file l'oggetto completo modificato. Questo metodo verrà chiamato da `GUIParametriConfigurazione` al momento del salvataggio dei campi inseriti.

- `public aggiornaEsamiInformatici(object esamiInformatici)`

Ha come argomento un oggetto *json* contenente i filtri per gli esami per un CdL specifico, quindi aggiorna l'attributo `esamiInformatici` sostituendo le informazioni e chiama infine `salvaEsamiInformatici()` per salvare su file l'oggetto completo modificato. Questo metodo verrà chiamato al momento del salvataggio dei campi inseriti dalla classe `GUIParametriConfigurazione`.

4.2.3 CarrieraLaureando

La classe **CarrieraLaureando** ha la responsabilità di mantenere ed elaborare i dati relativi a un laureando. Il costruttore legge nome, cognome, l'indirizzo di email dall'anagrafica, matricola e analizzerà il *json* ottenuto da **GestioneCarrieraStudente** ricavando i dati necessari, ovvero: DES, DATA_ESAME, VOTO, PESO, CORSO e SOVRAN_FLG. Inoltre, si preoccupa di valorizzarne gli attributi (trasformare lode e idoneità in valore numerico).

Attributi

- `public string CdL`, corso di laurea (*preso da json config*).
- `public string matricola`, matricola del laureando.
- `public string nome`, nome del laureando.
- `public string cognome`, cognome del laureando.
- `public string email`, email del laureando.
- `public date dataAppello`, data dell'appello di laurea.
- `public object esami[]`, array di oggetti composti dai seguenti attributi: data (utilizzato per l'ordinamento), nome, cfu, voto, fmedia.
- `public float mediaPesata`, media pesata del laureando.
- `public int CFUCheFannoMedia`, crediti del laureando che fanno media.
- `public int CFUCurriculari`, crediti curriculari del laureando.
- `public int CFUCurriculariAttesi`, crediti che il laureando dovrebbe avere per laurearsi (*preso da json config*).
- `public int votoDiTesi`, voto della tesi del laureando di valore 0 (*appare solo se Tmin:0, Tstep:0, Tmax:0 e T è nella formula*).
- `public int votoDiCommissione`, voto della commissione di valore 0 (*appare solo se Cmin:0, Cstep:0, Cmax:0 e C è nella formula*).
- `public string formulaVotoLaurea`, formula per calcolare il voto del laureando (*presa da json config*).

Metodi

- **costruttore** (`ConfigurazioneParametri conf, string dataAppello, string nome, string cognome, string email, string matricola, object esami, object filtroesami`)
Carica i valori all'interno della struttura, filtra i campi superflui ed eventuali esami non curriculari.
- **private valorizzaVoti()**
Converte lode e idoneità in valori numerici. Questa funzione è richiamata dal costruttore.
- **private aggiornaMediaPesata()**
Aggiorna l'attributo `mediaPesata` e `crediticheFannoMedia` (calcolandoli).
- **private aggiornaCFUCurriculari()**
Aggiorna l'attributo `CFUCurriculari` eseguendone il calcolo.

4.2.4 CarrieraLaureandoInformatica

La classe `CarrieraLaureandoInformatica` estende `CarrieraLaurando`, aggiungendo alcune informazioni necessarie per eseguire il calcolo del voto di laurea per i laureandi in Ingegneria Informatica. Le modifiche comprendono:

- Aggiunta di un attributo booleano `bonus` (visualizzato nel report pdf dopo la formula del voto di laurea) posto a `true` nel caso in cui il laureando concluda il percorso di studio entro il mese di maggio del quarto anno. Il bonus consiste nel escludere dalla media l'esame con voto più basso (a parità di esami con voto più basso si prende quello con crediti maggiori).
- Aggiunta di un attributo intero `mediaEsamiInformatici` che calcola la media degli esami informatici considerando il bonus. Gli esami vengono riconosciuti attraverso l'aggiunta di un attributo `inf` per ogni esame, posto a `true` se presente nel file `esami-informatici.json` passato nel costruttore.

Attributi

- **public object esami[]**, è un array con oggetti composti dai seguenti attributi: `data` (solo per ordinamento), `nomeEsame`, `cfu`, `voto`, `famedia`, `flag inf`. Il flag `inf` è necessario per individuare quali esami sono informatici per il calcolo del bonus.
- **private int annoImmatricolazione**, indica l'anno di immatricolazione del laureando e viene utilizzato per verificare la corretta decorrenza dei termini.
- **public boolean bonus**, indica se il laureando ha diritto al bonus.
- **public float mediaEsamiInformatici**, rappresenta media degli esami informatici per il calcolo del bonus.

Metodi

- **costruttore** (ConfigurazioneParametri conf, string dataAppello, string nome, string cognome, string email, string matricola, json esami, json filtroesami, json esamiInf)
Carica i valori all'interno della struttura, filtra i campi superflui ed eventuali esami non curriculari e, infine, imposta l'attributo `inf` di ogni esame a `true` se informatico.
- **private valorizzaBonus()**
Verifica se sono soddisfatte le condizione per l'ottenimento del bonus, in caso positivo imposta a `false` il flag `faMedia` all'esame più sconveniente ai fini del calcolo della media. La funzione è richiamata dal costruttore.
- **private valorizzaMediaEsamiInformatici()**
Aggiorna il valore della media degli esami informatici. La funzione è richiamata dal costruttore.

4.2.5 SimulazioneVotoLaureando

La classe `SimulazioneVotoLaureando` ha la responsabilità di calcolare la simulazione di voto di laurea per un laureando. Viene utilizzata per la realizzazione della proiezione del voto dalla classe `ReportPDFLaureandoConSimulazione`.

Attributi

- **private CarrieraLaureando carriera**, carriera di un laureando
- **public object simulazioneVotoLaurea**, oggetto contenente le proiezioni di voto in relazione ai parametri `Tmin`, `Tmax`, `Tstep`, `Cmin`, `Cmax`, `Cstep`.

Metodi

- **costruttore(ConfigurazioneParametri parametri, CarrieraLaureando carriera)**
Prende i parametri necessari per la simulazione, i dati della carriera del laureando e valorizza il contenuto `simulazioneVotoLaurea`.

4.2.6 SimulazioneVotoLaureandoInformatica

La classe `SimulazioneVotoLaureandoInformatica` ha la responsabilità di eseguire la simulazione di voto per un laureando in ingegneria informatica, in accordo alle modalità del Corso di Laurea.

Attributi

- `CarrieraLaureandoInformatica carriera`, attributo con le informazioni di un laureando in ingegneria informatica (sovrascrive l'attributo `carriera` nella classe base).

Metodi

- `costruttore(CarrieraLaureando carriera, object esami_informatici)`
Imposta l'attributo `carriera` con la carriera di un laureando in Ingegneria Informatica. Inoltre, riceve un oggetto contenente gli esami informatici e imposta i relativi flag della classe `CarrieraLaureandoInformatica`.

4.2.7 ReportPDFLaureando

La classe `ReportPDFLaureando` ha la responsabilità di costruire il report pdf di un laureando a partire dai dati contenuti nella classe `CarrieraLaureando`.

Attributi

- `private CarrieraLaureando carrieraLaureando`, è la carriera di un laureando.
- `private FPDF reportPDFLaureando`, contiene i dati relativi a un laureando per la generazione del pdf.

Metodi

- `costruttore(CarrieraLaureando laureando)`
Imposta l'attributo `carrieraLaureando` e realizza il report pdf assegnando all'attributo `reportPDFLaureando` i dati necessari (mediante i metodi privati `stampaDatiAnagrafici()`, `stampaCarriera()`, `stampaInformazioniCalcoloVoto()`).
- `public RestituisciReportLaureando()`
Realizza e restituisce un oggetto di tipo `FPDF` contenente tutti i dati per creare il file pdf. Questo non viene salvato immediatamente in quanto, quando viene generato un report per la commissione, non esiste un file per ogni laureando ma un unico documento che li racchiude tutti.

- **private aggiungiDatiAnagrafici()**

Funzione di utilità, aggiunge i dati anagrafici di un laureando all'interno di una pagina pdf.

- **private aggiungiCarriera()**

Funzione di utilità, aggiunge i dati relativi alla carriera di un laureando all'interno di una pagina pdf.

- **private aggiungiParametriEInformazioniCalcoloVoto()**

Funzione di utilità, aggiunge le informazioni e i parametri utilizzati per il calcolo del voto di un laureando all'interno di una pagina pdf.

- **public salvaReportLaureando()**

Salva il report come file pdf descritto dall'attributo `reportPDFLaureando` nel sistema in formato pdf.

4.2.8 ReportPDFLaureandoConSimulazione

La classe `ReportPDFLaureandoConSimulazione` estende `ReportPDFLaureando` aggiungendo i dati ottenuti mediante la classe `SimulazioneVotoLaureando`. Ha la responsabilità di restituire la pagina di un laureando come oggetto FPDF con la simulazione che verrà poi adoperata dalla commissione di laurea.

Attributi

- **private SimulazioneVotoLaureando simulazione**, attributo che contiene le informazioni di simulazione di voto per un laureando.

Metodi

- **costruttore(CarrieraLaureando carriera, SimulazioneVotoLaureando sim)**
Si occupa di impostare gli attributi della classe.

- **public restituisceReportLaureando()**

Restituisce un oggetto di tipo FPDF con la simulazione. Dunque, richiama il metodo della classe padre aggiungendo i dati ottenuti da `SimulazioneVotoLaureando`.

- **private aggiungiInformazioniSimulazione()**

Funzione di utilità, aggiunge la tabella con la simulazione all'interno del report pdf del laureando.

4.2.9 ReportPDFCommissione

La classe `ReportPDFCommissione` ha la responsabilità di salvare il file pdf per la commissione, costruendo le pagine basandosi su `ReportPDFLaureandoConSimulazione`.

Metodi

- `costruttore (object conf, SimulazioneVotoLaureando simulazioni[])`
Genera il riepilogo dei prospetti per la commissione, ovvero il file contenente l'elenco dei laureandi e le informazioni di ciascuno con simulazione. Salva il file risultante nel file system.
- `private aggiungiPaginaElenco()`
Costruisce la prima pagina contenente l'elenco dei laureandi.
- `private aggiungiReportPDFLaureandoConSimulazione()`
Costruisce le pagine dei laureandi con la visualizzazione della proiezione di voto, mediante la classe di appoggio `ReportPDFLaureandoConSimulazione`.

4.2.10 InviaReportPDFLaureandi

La classe `InviaReportPdfLaureandi` ha la responsabilità di inviare le carriere via mail a ciascun laureando, utilizzando l'omonimo attributo all'interno della classe `CarrieraLaureando`. Il report pdf generato utilizza il nome breve del Corso di Laurea formato come segue: `nomeBreve-stato-invio.json`

Attributi

- `private boolean prospettiInviati`, flag che identifica se è stato completato l'invio delle mail.
- `private string CdLShort`, nome breve del Corso di Laurea.

Metodi

- `public inviaProspetti(ParametriConfigurazione configurazione, CarrieraLaureando laureandi[])`
Recupera da `ConfigurazioneParametri` il nome breve del corso di laurea e inizializza l'attributo `cdlShort`. Inoltre viene letto, se esiste, il file `json` che indica eventuali laureandi che non hanno ancora ricevuto la mail. Se questo non esiste, viene generato un nuovo file contenente l'elenco di mail di laureandi e il corrispettivo file pdf.
- `public inviaReportSegreteriaDidattica()`
Notifica la segreteria didattica del corretto invio (se avvenuto) dei prospetti ai laureandi.

4.2.11 GUIParametriConfigurazione

La classe `GUIParametriConfigurazione` ha la responsabilità di costruire l'interfaccia grafica della pagina di configurazione, generando un'etichetta e un'area di testo per ogni attributo json, dimensionato opportunamente in base alla lunghezza del contenuto. Le parti html vengono mostrate attraverso l'utilizzo di PHPdom.

Metodi

- **costruttore(ConfigurazioneParametri conf, string cdL, string sezione)**
Metodo che, preso un json generico, costruisce la pagina dedicata html con i form per tutti i campi relativi a un unico Corso di Laurea. Il campo action del pulsante submit è collegato a una pagina php che si occuperà di ricostruire il nuovo json modificato a partire dai parametri POST ricevuti.
- **public generaJSONDaPOST()**
Metodo richiamato al momento della sottomissione del form da parte del pulsante di submit. Legge un parametro GET nella url che rappresenta quale file json dovrà costruire. Una volta elaborato, lo inoltra a uno dei seguenti metodi, in modo che vengano salvate le modifiche nel file corretto:
 - `ParametriConfigurazione.aggiornaFiltroEsami()`
 - `ParametriConfigurazione.aggiornaEsamiInformatici()`
 - `ParametriConfigurazione.aggiornaFormuleEMessaggi()`

4.2.12 GUISimulazioneVotoLaurea

La classe `GUISimulazioneVotoLaurea` ha la responsabilità di costruire l'interfaccia grafica della pagina d'inserimento dati per l'appello di laurea, aggiungendo il form e gli oggetti json necessari in posizione prefissata. Le parti html vengono mostrate attraverso l'utilizzo di PHPdom.

Metodi

- **costruttore()**
Disegna la pagina per l'inserimento delle matricole.
- **public visualizzaMessaggio(string msg)**
Mostra a schermo un messaggio `msg` informativo (*esempio: corretto invio delle mail*).
- **public leggiCdL()**
Legge il contenuto del campo “CDL”, restituendolo come stringa.
- **public leggiDataLaurea()**
Legge il contenuto del campo “Data laurea”, restituendolo come stringa.

- **public leggiMatricole()**

Legge il contenuto del campo “Matricole”, restituendo un vettore di stringhe.

- **public creaProspetti()**

Genera i prospetti per i laureandi e la commissione attraverso l'utilizzo delle classi ReportPDFLaureando e ReportPDFCommissione.

Appendice A

Ringraziamenti

Vai su GitHub.



Figura A.1: Marco Lampis ti punta il ferro, che fai?