Formulario di valutazione LPI

1 Introduzione 4

1.1 Informazioni sul progetto 4

1.2 Abstract 4

1.3 Scopo 4

2 Analisi 5

2.1 Analisi del dominio 5

2.2 Analisi e specifica dei requisiti 5

2.3 Use case 8

2.4 Pianificazione 8

2.4.1 Sezione analisi 10

2.4.2 Sezione progettazione 10

2.4.3 Sezione implementazione 10

2.4.4 Sezione test 11

2.5 Analisi dei mezzi 11

2.5.1 Software 11

2.5.2 Hardware 12

3 Progettazione 12

3.1 Design dell’architettura del sistema 12

3.2 Design dei dati e database 13

3.2.1 Schema ER 13

3.2.2 Schema logico 14

3.2.3 Specifica dei dati 14

3.3 Design delle interfacce 16

3.3.1 Pagina di login 16

3.3.2 Pagina di registrazione 17

3.3.3 Pagina di reset della propria password 18

3.3.4 Pagina contenente lo storico 18

3.3.5 Pagina di aggiunta di un formulario 19

3.3.6 Pagina di aggiunta delle motivazioni 20

3.3.7 Pagina di resoconto finale 21

3.3.8 Pagina di amministrazione 22

4 Implementazione 23

4.1 Lumen 23

4.1.1 Installazione 23

4.1.2 La struttura delle cartelle 24

4.1.3 Blade template 26

4.1.4 Gestione delle dipendenze 26

4.2 Gestione delle versioni 26

4.3 Database 26

5 Test 27

5.1 Protocollo di test 27

5.2 Risultati test 27

5.3 Mancanze/limitazioni conosciute 27

6 Consuntivo 28

7 Conclusioni 28

7.1 Sviluppi futuri 28

7.2 Considerazioni personali 28

8 Bibliografia 28

8.1 Sitografia 28

9 Allegati 28

# Introduzione

## Informazioni sul progetto

* Titolo: Formulario di valutazione LPI
* Durata: 23.01.2020-06.04.2020
* Allievo coinvolto: Matteo Forni
* Classe: Info 4AA
* Anno scolastico: 2019-2020
* Docente responsabile: Massimo Sartori

## Abstract

The aim of this project is to create a website that allows the creation and management of evaluation forms for the LPIs carried out by computer science students in their last year of school. Thanks to this product, it will be possible to add motivations, combined with the evaluation points that will be used later by the teachers to justify the assigned grades. The site will therefore be useful for those responsible for evaluations but also for students who will have clearer explanations. Teachers will be able to select the most suitable explanations through filters and these will be inserted in the document and will modify the final evaluation which can then be saved in PDF and printed out.

## Scopo

Lo scopo di questo progetto è quello di creare un sito web che consenta di creare e gestire dei formulari di valutazione degli LPI svolti dagli apprendisti di informatica all’ultimo anno di formazione. I moduli saranno redatti secondo il nuovo modello definito nell’ordinanza del 2014. Grazie a questo prodotto potrà essere possibile l’aggiunta di motivazioni, abbinate ai punti di valutazione, da parte di un amministratore che verranno, in seguito, utilizzate dai docenti per giustificare le note assegnate. Il sito tornerà dunque utile ai responsabili delle valutazioni ma anche agli allievi che avranno delle spiegazioni più chiare.

I docenti potranno selezionare tramite dei filtri le motivazioni più adatte al caso ed esse verranno inserite nel documento e andranno a modificare la valutazione finale che potrà poi venire salvata in formato PDF e anche stampata.

# Analisi

## Analisi del dominio

Il prodotto dovrà essere un formulario per la valutazione degli LPI degli informatici in forma digitale, attualmente esiste unicamente la versione sia digitale che cartacea dei moduli ma sono da compilare a mano. Il sito ha quindi come obiettivo quello di velocizzare le operazioni di assegnazione delle note e di eliminare gli errori umani, le valutazioni finali verranno infatti calcolate automaticamente dal sistema che si occuperà anche di formattarle e inserirle in un PDF che potrà poi venire stampato e/o salvato. Il risultato finale generato dai docenti dovrà essere conforme allo standard deciso nell’ordinanza del 2014.

Gli utenti saranno docenti, formatori e periti di esame con buone conoscenze di informatica ma il sito dovrà comunque essere di facile utilizzo e intuitivo. Il bisogno principale degli utenti è di eliminare le incongruenze tra più valutazioni diverse utilizzando delle motivazioni standard che verranno inserite dagli amministratori del sistema.

## Analisi e specifica dei requisiti

|  |  |
| --- | --- |
| **ID: REQ-01** | |
| **Nome** | Creazione database |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | - |
| **Sotto requisiti** | |
| **001** | Bisognerà inserire in maniera hard-coded i punti di valutazione |

|  |  |
| --- | --- |
| **ID: REQ-02** | |
| **Nome** | Creazione interfaccia di registrazione |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | - |
| **Sotto requisiti** | |
| **001** | Bisognerà inserire i campi nome, cognome, email e telefono |

|  |  |
| --- | --- |
| **ID: REQ-03** | |
| **Nome** | Creazione interfaccia di login |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Si dovrà avere un account per poter accedere |

|  |  |
| --- | --- |
| **ID: REQ-04** | |
| **Nome** | Creazione interfaccia di amministrazione |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Per accedervi si dovranno avere i permessi di amministratore (REQ-03) |
| **Sotto requisiti** | |
| **001** | Si dovrà poter inserire/modificare/eliminare le motivazioni |
| **002** | Le motivazioni potranno essere abbinate a un solo punto |
| **003** | Si dovrà poter inserire/modificare/eliminare i docenti |

|  |  |
| --- | --- |
| **ID: REQ-05** | |
| **Nome** | Creazione interfaccia per i docenti |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Per accedervi si dovrà aver creato un account |
| **Sotto requisiti** | |
| **001** | Si dovrà poter creare/modificare/eliminare dei formulari |
| **002** | Si dovrà poter visualizzare lo storico dei propri formulari |

|  |  |
| --- | --- |
| **ID: REQ-06** | |
| **Nome** | Creazione interfaccia di sviluppo del formulario |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Bisognerà aver creato un nuovo formulario per accedervi |
| **Sotto requisiti** | |
| **001** | Si dovrà poter scegliere le motivazioni in base alla sezione, punto o parola chiave |
| **002** | Si dovranno poter inserire i dati dei periti/docenti/allievi coinvolti |

|  |  |
| --- | --- |
| **ID: REQ-07** | |
| **Nome** | Calcolo dei risultati |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Bisognerà calcolare i risultati secondo il nuovo modello |
| **Sotto requisiti** | |
| **001** | Si dovranno calcolare i risultati in base alle motivazioni inserite |

|  |  |
| --- | --- |
| **ID: REQ-08** | |
| **Nome** | Creazione dei PDF |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | - |
| **Sotto requisiti** | |
| **001** | Si dovranno generare i PDF contenenti i risultati con il formato del nuovo modello |
| **002** | Si dovrà poter salvare i file |

## Use case



Figura 1 Diagramma degli use case

La foto soprastante rappresenta le use case del prodotto, come si può vedere vi saranno due attori che saranno rispettivamente gli amministratori e i docenti. Entrambi gli attori rappresentati nel diagramma avranno già creato un account valido ed effettuato il login con esso.

Gli admin del sistema potranno aggiungere, modificare ed eliminare gli utenti dei docenti oltre ad avere la possibilità di inserire, modificare ed eliminare delle motivazioni ed assegnarle ai rispettivi punti. I docenti potranno invece generare dei formulari a cui andranno ad assegnare delle motivazioni scelte fra quelle inserite in precedenza dagli amministratori ai vari punti di valutazione. Inoltre potranno, una volta terminata la valutazione, salvare il risultato in PDF.

## Pianificazione



Figura 2 Diagramma di Gantt preventivo

Nella figura superiore si può vedere il diagramma di Gantt completo, esso rappresenta in maniera indicativa come dovrebbe andare il progetto. Esso è stato diviso in quattro grandi categorie che sono Analisi, Progettazione, Implementazione e Test, oltre ad esse vi è la documentazione che procede lungo tutto il progetto dato che verrà completata con l’avanzare delle altre attività.

### Sezione analisi

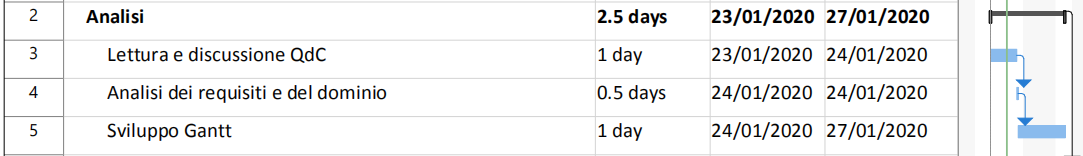


Figura 3 Diagramma Gantt, sezione analisi

La sezione iniziale è dedicata alla parte di analisi che verrà effettuata ad inizio progetto e comprenderà tre capitoli fondamentali che sono: la lettura e discussione del quaderno dei compiti con il docente responsabile, l’analisi dei requisiti e del dominio e la creazione del diagramma di Gantt. Questi tre punti sono fondamentali per comprendere tutti gli agenti e particolari del progetto. Queste tre attività sono piuttosto veloci da eseguire ed ho quindi previsto di metterci tre giorni lavorativi per portarle a termine.

### Sezione progettazione

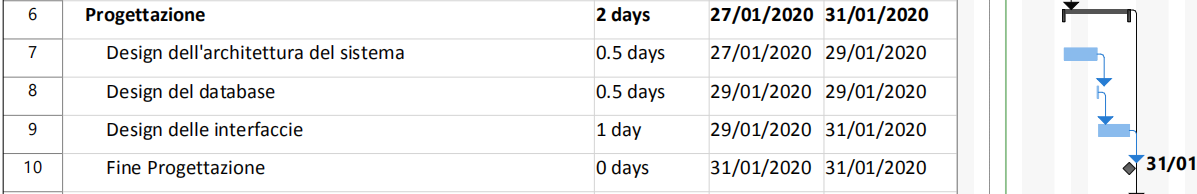


Figura 4 Diagramma Gantt, sezione progettazione

La seconda sezione del diagramma è la progettazione che comprende lo sviluppo teorico del sistema così da facilitarne l’implementazione effettiva. Questo sotto capitolo è stato suddiviso in tre attività che sono: il design dell’architettura del sistema, il design del database e il design delle interfacce. Inoltre esso termina con una milestone che rappresenta il termine della progettazione e l’inizio dello sviluppo e si situa al 31 di gennaio.

Le tre attività della sezione sono piuttosto veloci da completare ed ho quindi stimato di impiegarci mezza giornata lavorativa per le prime due e una intera per la terza.

### Sezione implementazione

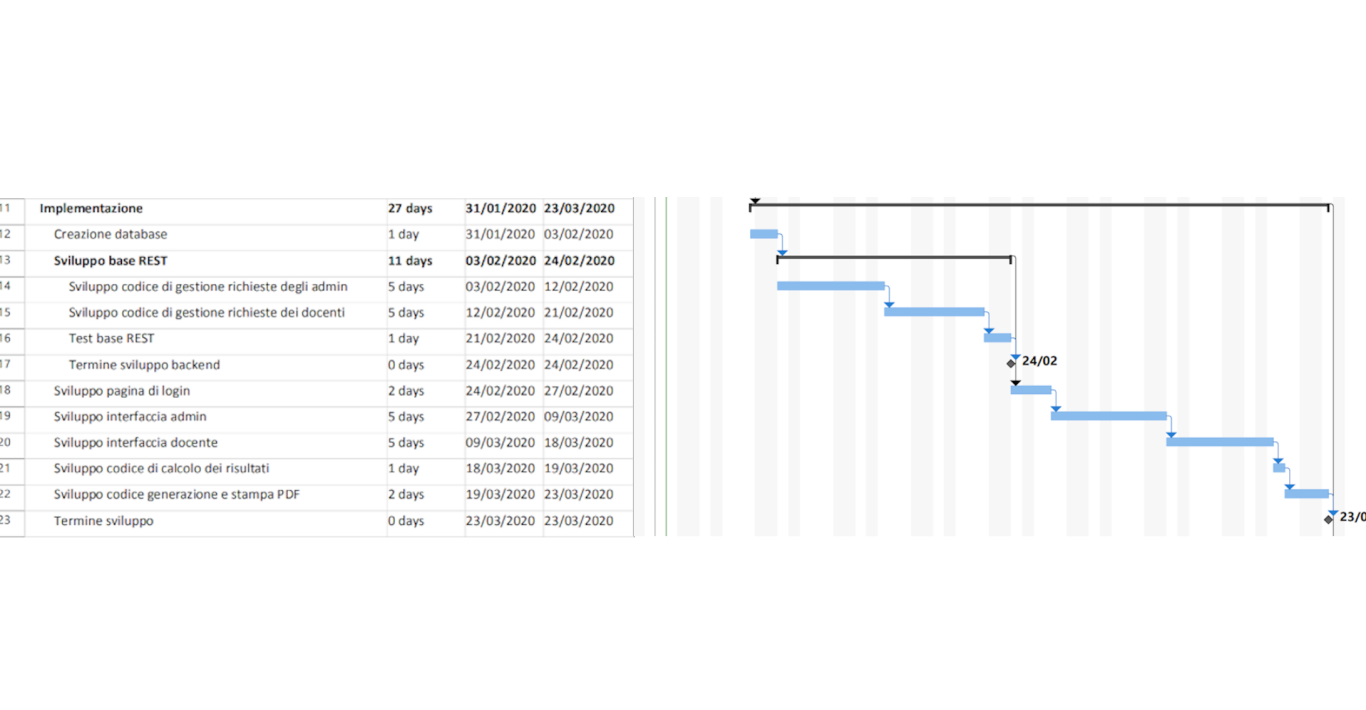


Figura 5 Diagramma Gantt, sezione implementazione

La terza sezione rappresenta l’implementazione, essa sarà la parte più lunga ed impegnativa del progetto e durerà circa un mese lavorativo. Essa inizia con lo sviluppo del database su MySQL che verrà seguito dalla creazione del codice di gestione delle richieste. Una volta che esso sarà stato testato si inizierà a sviluppare le pagine front-end e a collegarle con il back-end, infine si svilupperà la sezione di creazione dei PDF.

La sezione termina con una milestone che rappresenta il termine dell’implementazione e si situa il 23 marzo.

### Sezione test



Figura 6 Diagramma Gantt, sezione test

L’ultima parte del progetto sarà riservata ai test di funzionamento del sistema inizialmente specifici per le pagine ed infine generali per verificare l’integrazione dei vari componenti. Essi occuperanno circa una settimana lavorativa ed al loro termine finirà pure il progetto come rappresentato dalla milestone situata al 3 aprile.

## Analisi dei mezzi

### Software

SDK, librerie, tools utilizzati per la realizzazione del progetto e eventuali dipendenze.

### Hardware

L’hardware su cui verrà svolto il progetto e su cui potrà essere utilizzato in futuro è un normale PC, in questo caso un laptop Dell XPS 15 9570, su cui vi è installato un webserver funzionante che funzioni con il pattern MVC di PHP, quindi con il modulo di rewrite attivato e l’override delle cartelle consentito.

# Progettazione

## Design dell’architettura del sistema

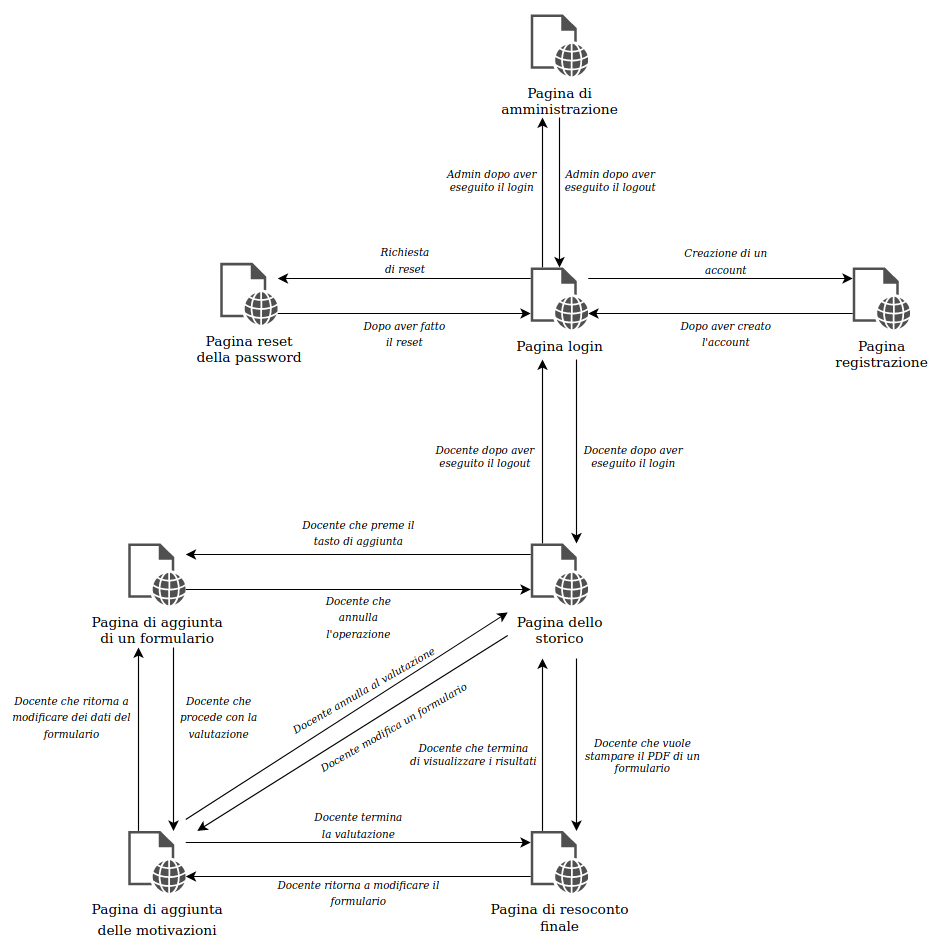


Figura 7 Diagramma dell'architettura del sistema

Il diagramma soprastante rappresenta le interazioni basilari e più importanti che saranno presenti nel prodotto, esso contiene ogni pagina con le operazioni che verranno eseguite maggiormente e la spiegazione di come gli utenti navigheranno tra esse. Questo non vuol dire che se nel diagramma le pagine non sono collegate esse non saranno accessibili fra loro ma che sarebbe un’operazione che non segue il flusso abituale del sito.

La pagina più centrale sarà quella di login dato che per utilizzare il prodotto bisognerà essere loggati, se l’utente non ha un account potrà accedere da essa alla sezione per registrarsi mentre se avrà dimenticato la propria password avrà la possibilità di reimpostarla tramite la pagina apposita. Una volta eseguito l’accesso l’utente si troverà nella prima pagina predefinita per un account con i suoi permessi: se amministratore verrà reindirizzato alla pagina riservata agli admin altrimenti a quella contenente lo storico dei propri formulari.

Dalla pagina che mostra la lista delle valutazioni eseguite in passato si potrà modificare, creare o stampare un formulario. Se si sceglierà la prima opzione si verrà reindirizzati alla pagina in cui si potranno gestire le motivazioni in un formulario, premendo il tasto di aggiunta si verrà reindirizzati alla pagina di creazione, che porterà in seguito a quella di gestione delle motivazioni, mentre se si vorrà stampare un formulario si verrà trasportati nella pagina di resoconto della valutazione dove vi saranno i bottoni per la stampa.

## Design dei dati e database

### Schema ER

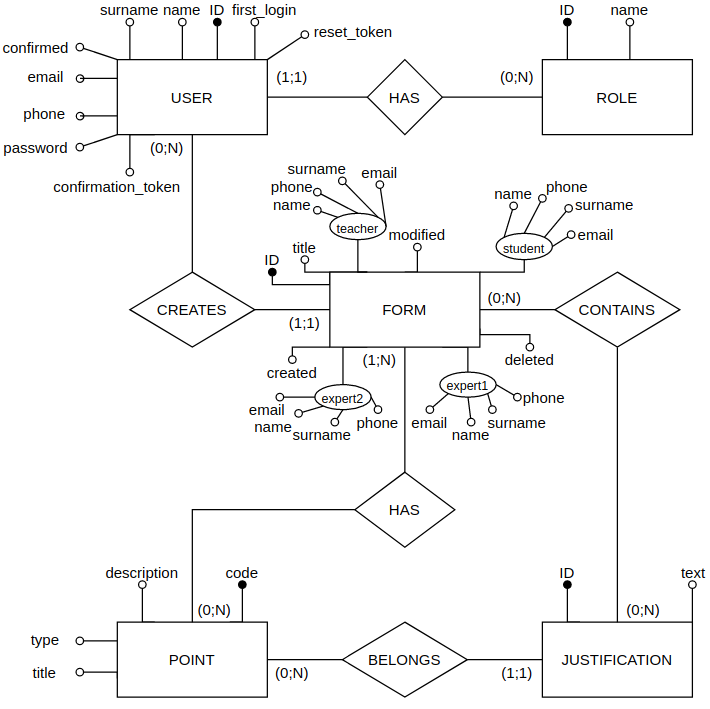


Figura 8 Diagramma ER del database

Il diagramma ER del database è piuttosto semplice e, come si può vedere dallo schema soprastante, comprende cinque tabelle che rappresentano rispettivamente: gli utenti del sistema, il ruolo che ricoprono, i punti di valutazione del LPI, le motivazioni a loro collegate e i formulari generati.

La prima tabella contiene le informazioni relative agli utenti che utilizzano il prodotto con le loro informazioni di base, essa prende il ruolo di ogni account dalla tabella role. Questo poteva essere sostituito con un attributo nella tabella user compilato manualmente alla creazione di ogni account ma avrebbe reso piuttosto complicato un futuro sviluppo del progetto. Il campo confirmedcontiene un booleano che indicherà se l’utente avrà eseguito la conferma della propria email o meno. La tabella pointcontiene le informazioni dei punti di valutazione presi dai criteri di valutazione LPI (estesi) e li identifica secondo il codice a loro assegnata nel file di provenienza, il campo non sarà quindi un intero ma una stringa. Ogni punto avrà un tipo che sarà o generaleoppure specificoed esso viene specificato nel campo typeche verrà settato a truese il punto sarà uno dei sette che potranno aggiungere i docenti o falsese sarà uno generico di base.

La tabella delle motivazioni conterrà soltanto un identificatore e il nome di essa, che è la stringa che verrà poi stampata nel file finale.

Infine la tabella più grande e complessa, quella relativa ai formulari, conterrà tutte le informazioni delle persone coinvolte nel LPI e quindi: studente, docente responsabile e i due periti. Oltre a ciò essa conterrà pure le date di creazione, ultima modifica e eliminazione così da rendere possibile la creazione dello storico di ogni docente.

### Schema logico

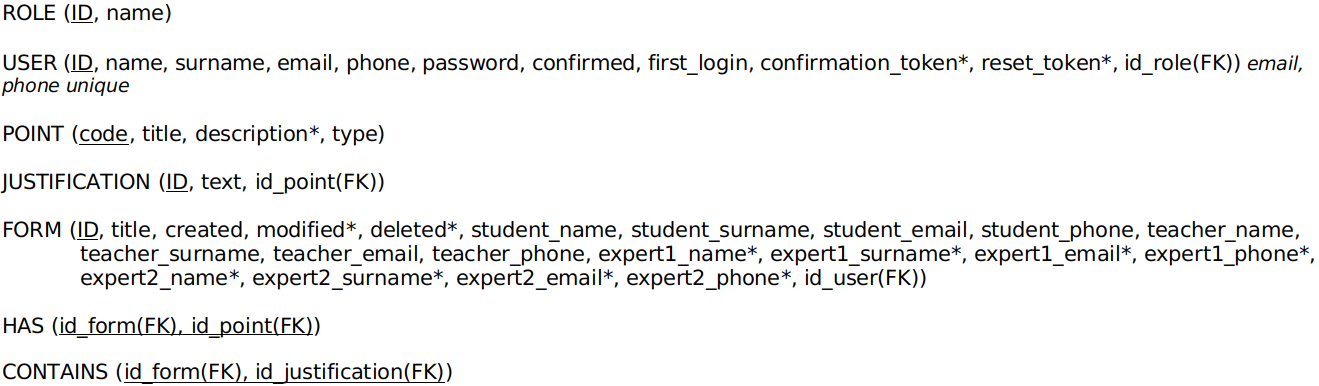


Figura 9 Schema logico del database

Lo schema logico rende più chiari alcuni punti del database e di come andrà a diventare effettivamente una volta generato in MySQL. Dall’immagine si può capire che la tabella user riceverà la chiave del ruolo così come justification da pointeformda user.Questo fa inoltre comprendere che la relazione situata fra i formulari e le motivazioni diverrà una tabella a sé che conterrà gli identificatori delle motivazioni con quelli dei formulari alla quale sono collegate, la stessa situazione è presente tra i punti di valutazione e i formulari.

### Specifica dei dati

|  |  |  |
| --- | --- | --- |
| Tabella ROLE | | |
| Attributo | Tipo | Descrizione |
| ID | int | Chiave primaria |
| name | varchar(50) | Il nome del ruolo |

|  |  |  |
| --- | --- | --- |
| Tabella USER | | |
| Attributo | Tipo | Descrizione |
| ID | int | Chiave primaria |
| name | varchar(100) | Il nome dell’utente |
| surname | varchar(100) | Il cognome dell’utente |
| email | varchar(100) | L’email dell’utente, dovrà essere unica fra tutti i record (unique) |
| phone | varchar(50) | Il num. di telefono, dovrà essere unico fra tutti i record (unique) |
| password | varchar(255) | La password, verrà salvato l’hash di quella inserita dall’utente |
| confirmed | boolean | Booleano specificante se l’utente ha confermato la sua email |
| id\_role | int | L’id del ruolo, se si eliminerà il ruolo accadrà anche all’utente |
| first\_login | boolean | Se l’utente è stato creato da un admin e si collega la prima volta |
| confirmation\_token | varchar(100) | Il token di conferma dell’email dell’account |
| reset\_token | varchar(100) | Il token di reset della password |

|  |  |  |
| --- | --- | --- |
| Tabella POINT | | |
| Attributo | Tipo | Descrizione |
| code | varchar(10) | Chiave primaria, riprende i codici dei criteri di valutazione |
| title | varchar(255) | Il titolo del punto di valutazione |
| description | longtext | La descrizione del punto di valutazione, può essere nulla |
| type | boolean | Il tipo di punto (se true: specifico altrimenti: generico) |

|  |  |  |
| --- | --- | --- |
| Tabella JUSTIFICATION | | |
| Attributo | Tipo | Descrizione |
| ID | int | Chiave primaria |
| text | longtext | Il nome della giustificazione |
| id\_point | varchar(100) | La chiave del punto, se si elimina il punto succede anche alla motivazione |

|  |  |  |
| --- | --- | --- |
| Tabella FORM | | |
| Attributo | Tipo | Descrizione |
| ID | Int | Chiave primaria |
| title | varchar(255) | Il titolo del progetto che si andrà a valutare |
| created | datetime | La data di creazione del formulario |
| modified | datetime | La data di ultima modifica del formulario, può essere nulla |
| deleted | datetime | La data di eliminazione del formulario, può essere nulla |
| student\_name | varchar(100) | Il nome dello studente |
| student\_surname | varchar(100) | Il cognome dello studente |
| student\_email | varchar(100) | L’email dello studente |
| student\_phone | varchar(50) | Il numero di telefono dello studente |
| teacher\_name | varchar(100) | Il nome del docente |
| teacher\_surname | varchar(100) | Il cognome del docente |
| teacher\_email | varchar(100) | L’email del docente |
| teacher\_phone | varchar(50) | Il numero di telefono del docente |
| expert1\_name | varchar(100) | Il nome del primo perito |
| expert1\_surname | varchar(100) | Il cognome del primo perito |
| expert1\_email | varchar(100) | L’email del primo perito |
| expert1\_phone | varchar(50) | Il numero di telefono del primo perito |
| expert2\_name | varchar(100) | Il nome del secondo perito |
| expert2\_surname | varchar(100) | Il cognome del secondo perito |
| expert2\_email | varchar(100) | L’email del secondo perito |
| expert2\_phone | varchar(50) | Il numero di telefono del secondo perito |
| id\_user | int | L’id dell’utente che ha creato il formulario |

Tutti gli attributi dei periti possono essere lasciati nulli in caso il formulario sia di valutazione di un progetto di semestre e non di un LPI.

|  |  |  |
| --- | --- | --- |
| Tabella CONTAINS | | |
| Attributo | Tipo | Descrizione |
| id\_justification | int | Chiave esterna della motivazione e chiave primaria |
| id\_form | int | Chiave esterna del formulario e chiave primaria |

|  |  |  |
| --- | --- | --- |
| Tabella HAS | | |
| Attributo | Tipo | Descrizione |
| id\_point | varchar(10) | Chiave esterna del punto di valutazione e chiave primaria |
| id\_form | int | Chiave esterna del formulario e chiave primaria |

## Design delle interfacce

### Pagina di login



Figura 10 Mockup della pagina di login

La pagina di login dovrà essere semplice e chiara, conterrà semplicemente un form di inserimento che consentirà all’utente di inserire la propria email e la relativa password. Sotto i due campi vi saranno due bottoni che consentiranno rispettivamente di accedere all’account o di modificare la propria password. Il form verrà validato sia in maniera interattiva tramite Javascript sia nel backend con PHP.

Per chi non avesse un account vi sarà un link in nel menu che consentirà di spostarsi alla pagina di registrazione.

### Pagina di registrazione



Figura 11 Mockup della pagina di registrazione

La pagina di registrazione sarà piuttosto chiara e intuitiva e conterrà unicamente un form che consentirà di inserire tutti i campi richiesti per compilare il formulario di valutazione degli LPI più una password. Il form verrà validato sia in maniera interattiva tramite Javascript sia nel backend con PHP.

### Pagina di reset della propria password



Figura 12 Mockup della pagina di reset della password

La pagina di reset della password consentirà di inserire il proprio indirizzo email alla quale verrà inviato un messaggio contenente un link, seguendo tale link l’utente si troverà in una pagina contenente due campi d’inserimento che gli consentiranno di inserire una nuova password e ripeterla.

### Pagina contenente lo storico

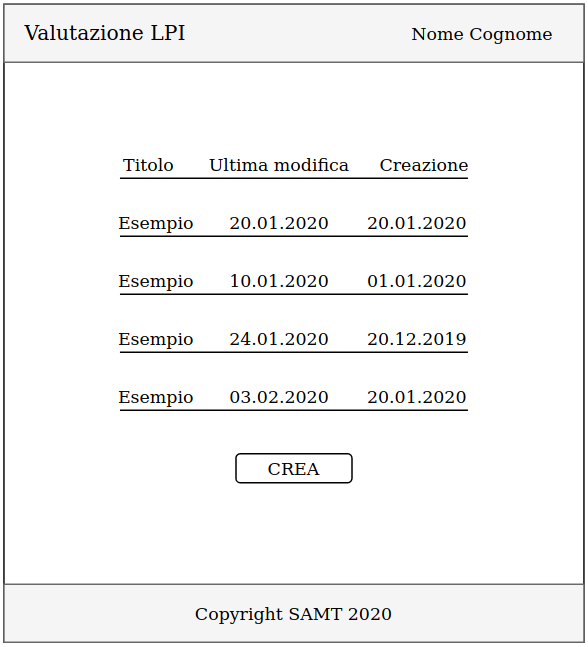


Figura 13 Mockup della pagina contenente lo storico

La pagina contenente lo storico sarà la prima che verrà visualizzata dai docenti quando accederanno al loro account. Essa mostrerà tutti i formulari da loro creati con le date di modifica, creazione e, se è il caso, eliminazione. Il docente avrà la possibilità di eliminarli, modificarli oppure crearne dei nuovi tramite il tasto crea*.*

### Pagina di aggiunta di un formulario



Figura 14 Mockup della pagina di aggiunta di un formulario

La pagina di aggiunta di un formulario sarà abbastanza complessa e carica di elementi, essa conterrà due sezioni distinte per i vari tipi di informazioni richieste:

* Dati del progetto
* Criteri specifici

Nella prima sezione si dovranno inserire tutti i dati dell’allievo coinvolto, dei periti e del superiore responsabile nel progetto che saranno: nome, cognome, email e numero di telefono. Nella seconda vi sarà invece la sezione in cui il docente potrà selezionare i criteri di valutazione specifici che potranno essere ricercati tramite codice o nome.

### Pagina di aggiunta delle motivazioni



Figura 15 Mockup della pagina di aggiunta delle motivazioni ad un formulario

La pagina dove sarà possibile collegare delle motivazioni ai formulari conterrà una sezione centrale che consentirà di ricercare le motivazioni tramite tre metodi diversi che saranno:

* Ricerca in base ad una parola chiave
* Ricerca in base ad una delle tre sezioni di valutazione
* Ricerca in base ad un punto di valutazione

Ciò significa che scrivendo qualcosa nel campo di testo appariranno tutte le motivazioni che contengono quella/e parola/e, inserendo una delle tre sezioni si visualizzeranno solamente le motivazioni di essa così come per un punto.

Sotto a questa sezione ve ne sarà una più piccola che conterrà i bottoni per procedere e terminare l’aggiunta, ritornare alla pagina precedente e salvare il lavoro fatto fino a quel momento.

### Pagina di resoconto finale



Figura 16 Mockup della pagina di resoconto

La pagina di resoconto finale conterrà i punteggi assegnati nelle tre sezioni con la possibilità di visualizzare quali motivazioni saranno parte di ogni sezione. Sotto di esse vi sarà la nota finale del LPI insieme alla possibilità di salvare il tutto in formato PDF con lo stile deciso nell’ordinanza del 2014.

### Pagina di amministrazione



Figura 17 Mockup della pagina di amministrazione

La pagina riservata agli amministratori sarà suddivisa in due parti relative alle due categorie di informazioni da gestire: gli utenti e le motivazioni. Nella prima sezione vi sarà infatti una tabella mostrante tutti gli utenti del sistema con la possibilità di aggiungerne, modificarli e eliminarli. Nella seconda sezione vi sarà una tabella rappresentante tutte le motivazioni con la possibilità di eseguirvi le stesse operazioni disponibili per gli utenti.

# Implementazione

## Lumen

Il progetto è stato interamente sviluppato utilizzando il micro-framework in PHP chiamato Lumen, esso è un discendente di Laravel e quindi ne riporta i tratti più caratteristici ma ha delle possibilità e componenti in meno. Esso è stato definito dai suoi sviluppatori come *“Decisamente Laravel. Deliziosamente minimale”* ed è stato valutato come uno dei micro-framework più veloci esistenti posizionandosi davanti a Silex e Slim 3.

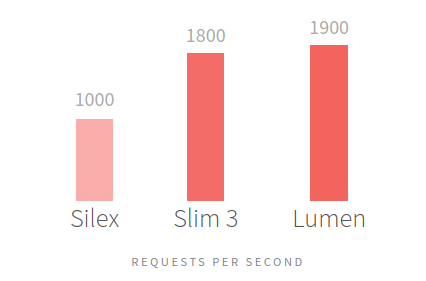


Figura 18 Lumen richieste soddisfatte per secondo

Lumen è pensato per fare delle piccole API come in questo caso e contiene già delle librerieche consentono di facilitare la scrittura del codice ad esempio Facadesche da la possibilità di eseguire query in maniera sicura e semplice sul database. Il vantaggio che esso ha su Laravel è che è molto più incentrato sulla velocità e quindi alcune cose sono gestite in maniera differente e facilitata per ridurre il tempo di risposta, ad esempio il routing è gestito dalla libreria FastRouteal posto di IlluminateRouting*.*

Il framework si basa su una struttura di lavoro del tipo MVC (Models Views Controllers), essi hanno i seguenti ruoli nel funzionamento dell’infrastruttura:

* Models: si occupano di caricare i dati dal database e di fornirli in un formato standard
* Views: sono le pagine web, si occupano di mostrare il risultato finale all’utente
* Controllers: si occupano di richiedere dei dati ai models e passarli alle views

Infine ci sono le routes che sono i mezzi tramite cui l’utente può eseguire le richieste al prodotto. A seguire uno schema del funzionamento di Laravel che è lo stesso di Lumen.

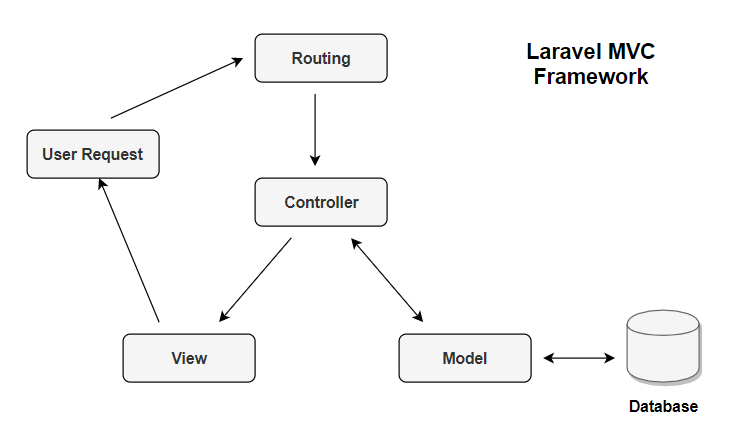


Figura 19 Schema di funzionamento MVC

### Installazione

L’installazione di Lumen è molto veloce e semplice e si racchiude in un unico comando di composer che genera tutta la base del progetto:



Una volta fatto ciò basterà far partire il progetto tramite un webserver, il metodo più comodo e veloce è quello di usare il server integrato di PHP indicando quale è la base del progetto:



Fatto ciò il framework sarà funzionante e questo sarà visibile collegandosi al link scritto in precedenza.



Figura 20 Lumen schermata iniziale

### La struttura delle cartelle

Lumen alla sua installazione conta svariate cartelle, a seguire uno schema delle cartelle e files alla radice del progetto:

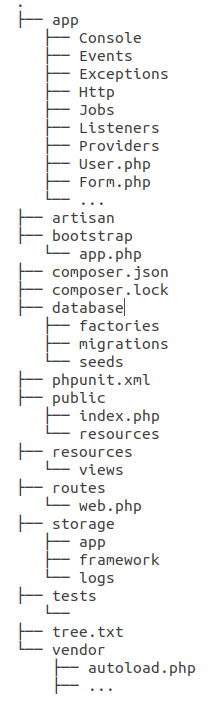


Figura 21 Schema di files e cartelle di Lumen

Le cartelle degne di nota sono quelle in cui verrà effettivamente scritto codice dallo sviluppatore, esse sono principalmente: app dove verranno inseriti i models, Http/Controllers dove verranno aggiunti i controllers, resources/views dove verranno salvate le views, routes che contiene il file dove andranno salvate le routes ed infine Http/Middleware dove verranno salvati i middleware. Un’altra cartella degna di nota è public/resources dove potranno venire salvate tutte le risorse esterne come librerie, estensioni, ...

### Blade template

Oltre a ciò il micro-framework eredita da Laravel la possibilità di utilizzare i template blade, essi sono dei files contenenti codice HTML e/o Javascript che possono venire riutilizzati e richiamati per evitare la scrittura di codice già esistente. Consentono anche di eseguire delle operazioni con delle strutture comuni come gli if o i for che semplificano lo sviluppo di pagine modulari e dinamiche. Per utilizzare questo motore di templating basta creare i files delle views con l’estensione .blade.php.

Il vantaggio principale è l’ereditarietà che consente di richiamare una pagina in molte altre così da ridurre il codice, questo può essere utilizzato ad esempio per inserire sempre header e footer senza copiarne ed incollarne il codice in tutte le pagine.

### Gestione delle dipendenze

Lumen si basa, per gestire le proprie dipendenze, su Composer, un software di gestione di dipendenze di PHP che consente di definire delle librerie da implementare nel file composer.json.



In questo caso si può notare come sono state implementate le librerie per utilizzare i JWT, il framework di base, PHPMailer per l’invio di email e TCPDF per la creazione dei PDF.

## Gestione delle versioni

Le versioni del progetto sono state gestite costantemente tramite una repository di GitHub su cui venivano fatti dei push più volte al giorno. Affianco ad ‘essa c’è stata, per gran parte del tempo, una repository di GitLab su un server interno della scuola sulla quale ho però smesso di eseguire dei push in quanto, dal 12.03.2020, le lezioni sono state eseguite da casa e non ho più avuto possibilità di collegarmi al server interno della scuola. Il link della cartella aggiornata costantemente è il seguente: <https://github.com/matteoforni/Valutazione-LPI>.

## Database

Il database su cui si basa il progetto è stato sviluppato sulla base dello schema ER e di quello logico che si trovano rispettivamente al capitolo 3.2.1 e al capitolo 3.2.2. I punti fondamentali dello sviluppo di esso sono stati i collegamenti con le varie tabelle dato che essi implicano delle chiavi esterne che rendono necessari dei vincoli di integrità referenziale. Quest’ultimi sono stati implementati in tutte le tabelle che riportano una chiave esterna e consentono di gestire meglio i dati all’eliminazione di un record. Nella tabella user la chiave esterna id\_role ha impostato che, all’eliminazione di un ruolo, nell’utente che lo possedeva, il campo viene impostato a null così da far continuare l’esistenza dell’utente, lo stesso comportamento avviene nella tabella form sul campo id\_user.



Tutte le altre chiavi esterne hanno invece impostato che all’eliminazione del record padre esse vengono eliminate in cascata. Questo viene fatto nei casi in cui un campo non ha più senso di esistere se non ha l’attributo padre.



L’accesso ad esso da parte del programma viene eseguito con un utente chiamato valutazionelpi che ha permessi solamente di scrittura, lettura, modifica e eliminazione sul database del progetto. Questo è stato fatto per evitare che, se qualcuno riuscisse ad aggirare il software, abbia pieno controllo di tutto il database.



## Connessione con il database

Per effettuare la connessione con il database in Lumen è molto semplice e veloce, basta infatti inserire nel file .env, situato alla base della cartella del progetto, tutte le informazioni necessarie nei loro campi riservati.



Fatto ciò il framework penserà a tutto il resto e, se i dati inseriti sono corretti, la connessione verrà effettuata.

## Model

Per comunicare con il database è necessario creare dei models, essi verranno salvati nella cartella app e conterranno poco codice dato che essi estenderanno per l’appunto la classe Model della libreria Eloquent che contiene tutti i metodi necessari. Per utilizzare Eloquent andrà però de commentata la riga 26 del file bootstrap/app.php.



Una volta fatto ciò basterà, nel model appena creato, estendere la classe e specificare alcuni attributi necessari al funzionamento. Bisognerà pe l’appunto specificare se la tabella relativa al model utilizza i timestamps, il nome della tabella e quali colonne si potranno modificare, questo verrà fatto specificando degli attributi nella classe.



Di classi così ve ne sono otto di cui sette sono quelle relative alle sette tabelle del database del progetto mentre l’ultima è quella che gestisce l’invio delle mail di modifica della password o di notifica agli utenti. Quest’ultima è diversa dalle altre in quanto il suo compito non è quello di comunicare con il database, essa contiene un solo metodo che consente, passando il destinatario, il corpo del messaggio e il soggetto, di inviare un’email tramite il server SMTP di Gmail basandosi sulla libreria phpmailer.

Nella classe Mailer inizialmente si genera una nuova istanza di PHPMailer dalla quale si richiameranno tutte le funzioni necessarie per l’invio del messaggio che verrà, infine, inviato tramite il metodo Send().

Gli utenti riceveranno il messaggio da parte di [valutazionelpi@gmail.com](mailto:valutazionelpi@gmail.com), un indirizzo generato appositamente per il progetto.



## Routes

La gestione delle routes, come detto in precedenza, è tutta fatta nel file routes/web.php. Esse rappresentano le richieste che possono essere fatte ai controllers del prodotto e, grazie al micro-framework, nel file vengono raggruppate per pagina, questo vuol dire che ogni pagina del sito avrà un suo gruppo di richieste fattibili e ben distinte.

La prima richiesta che viene effettuata, collegandosi al sito, è di tipo get e richiama la pagina di login.



Le richieste per caricare le pagine del sito non necessitano di aver effettuato il login e sono raggruppate per un prefisso che specifica la pagina in uso come si può vedere nell’esempio seguente che mostra le routes della pagina di registrazione.



Tutti i gruppi di richieste che necessitano di privilegi speciali hanno collegato un middleware che verifica e valuta se far procedere l’utente o fermarlo. Ci sono quindi, per alcuni casi, due gruppi della stessa pagina: uno con middleware e uno senza.



## Middleware

Il progetto conta un unico middleware, queste classi si occupano di filtrare gli accessi e le richieste in base, ad esempio, a delle autenticazioni. Esso si occupa di gestire le richieste che necessitano un login verificando che in esse vi sia impostato il token JWT, se esso è impostato la richiesta viene fatta passare, altrimenti viene bloccata e ritornato il codice 401 Unauthorized.

La classe riceve la richiesta http e carica il token dall’header, più precisamente dalla voce Authorization. Se esso è presente cerca di decodificarlo e se ci riesce manda avanti la richiesta, altrimenti ritorna un errore 401.



## Controllers

Le classi controller del progetto sono situate nella cartella app/Http/Controllers e servono a elaborare e passare i dati tra views e models e viceversa. Ogni gruppo di routes corrisponde ad un controller e quindi ve ne è uno per ogni sezione del sito: uno per il login, uno per la registrazione, uno per i docenti, uno per gli admin e uno per la creazione dei PDF.

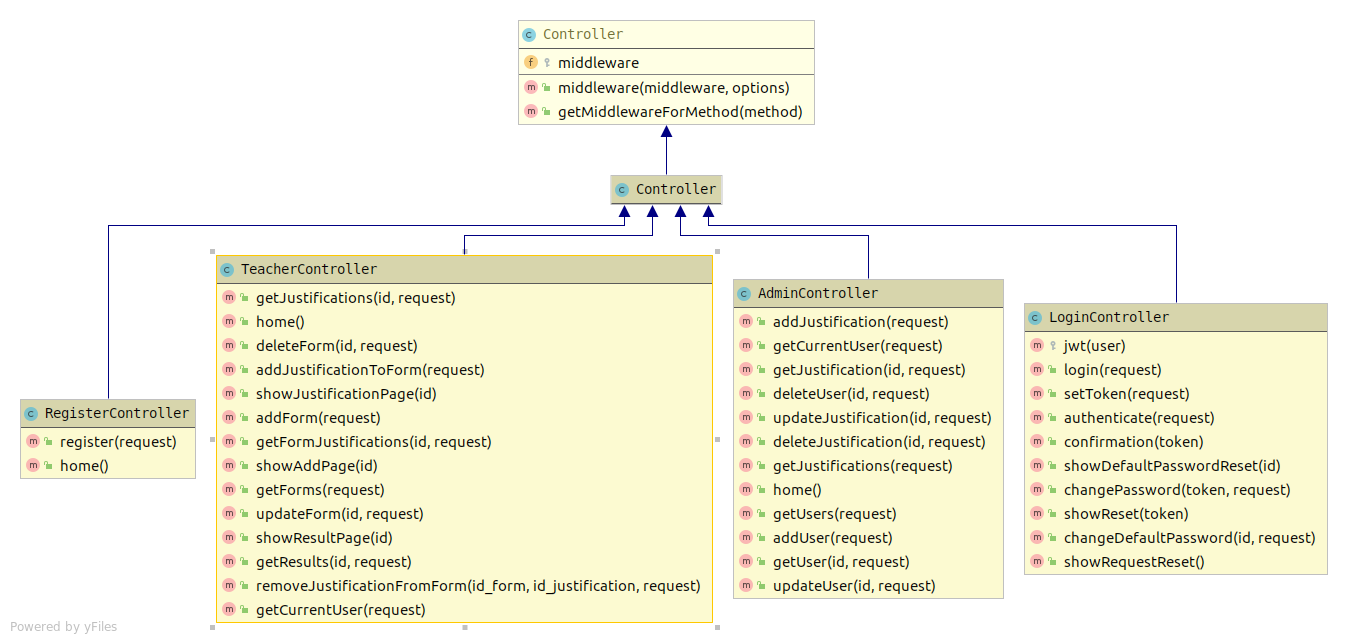


Figura 22 Schema UML della cartella Controllers

Come si può notare dallo schema soprastante tutti i controller vanno ad estendere una classe Controller globale che definisce una base standard per tutti gli altri. Nell’immagine non è presente il controller che si dedica a generare i PDF.

Tutti i metodi delle classi ritornano i dati in formato JSON e si aspettano anche di ottenerlo nello stesso formato.

### Validazione dei dati

La validazione dei dati con Lumen è molto semplice ed avviene tramite la classe Validator presente nel framework. Essa consente, richiamando il metodo make() a cui va passata la richiesta, l’array di valori che dovrà contenere la richiesta con tutti i punti che devono soddisfare e i messaggi di errore, personalizzati, che verranno poi mostrati.



Come si può notare dalla schermata superiore vi sono due metodi per specificare i requisiti che deve rispettare un campo: il primo e più utilizzato è passando un’unica stringa con tutti i punti separati da una barra verticale | mentre il secondo è passando un array di stringhe, questo si utilizza unicamente se si necessita di applicare un’espressione regolare al campo.

Per verificare che sia tutto andato bene si può utilizzare il metodo fails() ritornando gli errori:



La validazione di Lumen verifica autonomamente che i dati non siano tentativi di attacchi e quindi elimina autonomamente le possibilità di SQL Injection, cross-site scripting e molti altri attacchi comuni.

## Lettura di un elemento

### Aggiunta di un elemento

L’aggiunta di un elemento al database è relativamente semplice e veloce, le caratteristiche che accomunano tutte le operazioni di questo tipo indifferentemente dalla tabella alla quale si riferiscono sono una validazione dei dati come mostrato nel capitolo precedente, se necessario l’inserimento o la rimozione di dati aggiuntivi o che non verranno salvati nel database come ad esempio la ripetizione della password e il comando di aggiunta.

Grazie al micro-framework per inserire una riga nel database basta richiamare il metodo create() tramite il model relativo della tabella alla quale bisogna passare la richiesta ricevuta.



#### Aggiunta di un utente da parte di un amministratore

L’unica aggiunta diversa dalle altre è la creazione di un utente da parte dell’amministratore, essa infatti comprende che la password non venga inserita ma creata sul momento in maniera casuale e poi spedita per email all’indirizzo dell’account.

La creazione della stringa casuale viene fatta nel seguente modo:



Viene appunto creata una stringa di caratteri alfanumerici lunga 10 e in seguito criptata tramite il metodo password\_hash e inserita nella richiesta. Fatto ciò l’utente potrà essere creato e notificato via email.



### Modifica di un elemento

La modifica di una riga del database comporta la sovrascrittura di essa e può essere fatta in due modi diversi.

#### Modifica totale

Con modifica totale si intende quando si vanno a sovrascrivere tutti i dati contenuti nel record attuale, anche se quelli nuovi sono uguali a quelli vecchi. Questo metodo viene utilizzato nel progetto quando vi sono dei form di modifica che consentono di cambiare tutti i dati, così da evitare inutili verifiche su quale campo abbia subito un cambiamento e quale no.

Per fare ciò con Lumen basta inviare nella richiesta tutti i dati, come se si volesse aggiungere il campo, validarli ed infine passarli al metodo update del model di riferimento della tabella.



#### Modifica parziale

Con modifica parziale si intende quando si va a modificare solamente un campo della riga salvata nel database, questo è utile se si sa che si sta modificando unicamente un valore ed evita quindi la sovrascrittura di tutti gli altri.

Per eseguire questa operazione in Lumen si deve solamente caricare il record dal database con uno dei metodi disponibili e modificare unicamente il campo scelto.



Infine bisognerà sempre ricordarsi di salvare per rendere effettive le modifiche.



### Eliminazione

### Query strutturate

### Creazione dei PDF

## Views

# Test

## Protocollo di test

Definire in modo accurato tutti i test che devono essere realizzati per garantire l’adempimento delle richieste formulate nei requisiti. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-012 | **Nome:** | Import a card with KIC, KID and KIK keys, but not shown with the GUI |
| **Descrizione:** | Import a card with KIC, KID and KIK keys with no obfuscation, but not shown with the GUI | | |
| **Prerequisiti:** | Store on local PC: Profile\_1.2.001.xml (appendix n\_n) and Cards\_1.2.001.txt (appendix n\_n).  PIN (OTA\_VIEW\_PIN\_PUK\_KEY) and ADM (OTA\_VIEW\_ADM\_KEY) user right not set. | | |
| **Procedura:** | 1. Go to “Cards manager” menu,  in main page click “Import Profiles” link, Select the “1.2.001.xml” file, Import the Profile 2. Go to “Cards manager” menu,  in main page click “Import Cards” link, Select the “1.2.001.txt” file, Delete the cards,  Select the “1.2.001.txt” file, Import the cards 3. Research the “41795924770” Card, Click the imsi card link Check the card details 4. Execute the SQL: SELECT imsi, dir, keyset, cntr, rawtohex(kickey), rawtohex(kidkey), rawtohex(kikkey), rawtohex(chv), rawtohex(dap)FROM otacardkey a where imsi='340041795924770' ORDER BY keyset; | | |
| **Risultati attesi:** | Keys visible in the DB (OtaCardKey) but not visible in the GUI (Card details) | | |

## Risultati test

Tabella riassuntiva in cui si inseriscono i test riusciti e non del prodotto finale. Se un test non riesce e viene corretto l’errore, questo dovrà risultare nel documento finale come riuscito (la procedura della correzione apparirà nel diario), altrimenti dovrà essere descritto l’errore con eventuali ipotesi di correzione.

## Mancanze/limitazioni conosciute

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

# Consuntivo

Consuntivo del tempo di lavoro effettivo e considerazioni riguardo le differenze rispetto alla pianificazione (cap 1.7) (ad esempio Gannt consuntivo).

# Conclusioni

Quali sono le implicazioni della mia soluzione? Che impatto avrà? Cambierà il mondo? È un successo importante? È solo un’aggiunta marginale o è semplicemente servita per scoprire che questo percorso è stato una perdita di tempo? I risultati ottenuti sono generali, facilmente generalizzabili o sono specifici di un caso particolare? ecc

## Sviluppi futuri

Migliorie o estensioni che possono essere sviluppate sul prodotto.

## Considerazioni personali

Cosa ho imparato in questo progetto? ecc

# Bibliografia

## Sitografia

* <https://stackoverflow.com/questions/,> *Stackoverflow*, 06.02.2020
* <https://lumen.laravel.com/docs/6.x>, *Lumen 6 documentation*, 27.01.2020
* <https://laravel.com/docs/6.x>, *Laravel 6 documentation,* 29.01.2020
* <https://laracasts.com/discuss/>, *Laracasts discussion,* 31.01.2020
* <https://php.net>, *PHP documentation,* 31.01.2020
* <https://api.jquery.com>*, JQuery documentation,* 03.02.2020

# Glossario

|  |  |
| --- | --- |
| Vocabolo | Definizione |
| API | Insieme di funzioni e protocolli utilizzabili in un software |
| Composer | Software di gestione di dipendenze di PHP |
| Controllers | Classi che uniscono i models con i controllers |
| Footer | Parte finale di una pagina web |
| Framework | Struttura/architettura sulla quale sviluppare un software |
| Git | Software di gestione delle versioni |
| Header | Parte superiore di una pagina web, solitamente contiene il menu |
| Integrità referenziale | Mantenimento dell’affidabilità dei dati di un database |
| JSON | Formato per lo scambio di dati |
| JWT | (JSON Web Token) Token di verifica basato su JSON |
| Middleware | Classi che consentono di filtrare gli accessi e le richieste |
| Models | Classi di comunicazione con il database |
| PHP | Linguaggio di programmazione orientato alle pagine web dinamiche |
| Query | Richiesta al database |
| Timestamp | Tipo di dato che rappresenta un data |
| Token | Insieme di caratteri alfanumerici utilizzato per identificare qualcuno |
| Views | Parte grafica che vedrà l’utente finale |

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Istruzioni di installazione del prodotto (con credenziali di accesso) e/o di eventuali prodotti terzi
* Documentazione di prodotti di terzi
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o Qdc
* Prodotto
* …