

Documentazione Progetto Registrazione

Titolo del progetto: Documentazione Progetto Registrazione
Alunno/a: Jari Näser
Classe: Info 3AA
Anno scolastico: 2018/2019
Docente responsabile: Adriano Barchi, Francesco Mussi e Luca Muggiasca

1	Introduzione	3
1.1	Informazioni sul progetto.....	3
1.2	Abstract.....	3
1.3	Scopo	3
1.4	Analisi del dominio	3
1.5	Analisi e specifica dei requisiti	4
1.6	Pianificazione	6
1.7	Analisi dei mezzi	7
1.7.1	Software.....	7
1.7.2	Hardware	7
2	Progettazione	7
2.1	Design dell'architettura del sistema	7
2.2	Design delle interfacce.....	8
2.3	Design procedurale	9
3	Implementazione	10
3.1	Pagina di benvenuto	10
3.2	Pagina di registrazione	11
3.2.1	Form inserimento dati e controlli.....	11
3.2.2	Tabella verifica dati	14
3.3	Pagina di salvataggio	16
3.3.1	Ricezione e controllo dati.....	16
3.3.2	Uso file csv.....	17
4	Test	19
4.1	Protocollo di test	19
4.2	Risultati test.....	22
4.3	Mancanze/limitazioni conosciute	22
5	Consuntivo	23
6	Conclusioni.....	24
6.1	Sviluppi futuri	24
6.2	Considerazioni personali.....	24
7	Bibliografia	25
7.1	Sitografia	25
8	Allegati	25

1 Introduzione

1.1 Informazioni sul progetto

Allievo coinvolto: Jari Näser

Classe: Informatica 3AA Presso la Scuola Arti e Mestieri Trevano

Docenti responsabili: Francesco Mussi, Adiano Barchi, Luca Muggiasca

Data inizio: 5-09-18

Data fine: 9-11-18

1.2 Abstract

Because in today's web pages registration forms are very popular the scope of this project was to realize an interactive web page made of various programming languages such as php, javascript, html and css where it's possible to register people in it.

It's also possible to view all the registrations that were made in the same day as you're viewing the page at the end of the registration process.

Thanks to this project it's now possible to use personal written validator methods, functions and forms in future projects.

1.3 Scopo

Lo scopo principale di questo progetto era di farci prendere dimestichezza con questo nuovo metodo di lavoro e con i software di versioning.

Inoltre abbiamo adoperato molte conoscenze apprese in moduli fatti precedentemente e unendoli abbiamo svolto il progetto richiesto.

Come secondo scopo l'obiettivo era quello di capire come arrivare da una richiesta di un cliente ad un progetto funzionante attraverso tutte le fasi che verranno specificate nel resto del documento

Analisi

1.4 Analisi del dominio

Attualmente il sistema non è ancora esistente, lo scopo è quello di creare un prodotto in formato web che si occupa di semplificare l'uso delle registrazioni.

1.5 Analisi e specifica dei requisiti

Inizialmente bisogna installare l'ambiente di sviluppo che è composto dall'IDE PhpStorm della JetBrains e del software Apache che fa da Host per il nostro progetto, inoltre necessitiamo anche dell'interprete php che ci permetterà di poter scrivere del codice nel linguaggio php.

Successivamente bisogna creare una pagina di benvenuto che contiene al suo interno una semplice scritta di benvenuto e un bottone che al suo click ci manderà alla pagina di registrazione.

La pagina di registrazione è composta da 11 campi di cui 9 sono obbligatori che l'utente dovrà compilare, inoltre in fondo alla pagina sono anche presenti i due bottoni "Cancella" e "Avanti" che si occupano di cancellare tutto il contenuto immesso nei Form oppure di farci andare alla pagina successiva di conferma nella quale appare una tabella contenente i dati appena immessi che attraverso due bottoni "Modifica" e "Salva" danno la possibilità di correggere i campi errati oppure mandando i dati immessi nei Form e controllati dai metodi validatori al server che andrà ad aggiungerli nei rispettivi file csv.

In fine verrà mostrata una pagina contenente tutte le registrazioni effettuate nel giorno stesso della propria con l'opzione di tornare alla home attraverso il bottone "Torna alla home".

ID: REQ-01	
Nome	Pagina di benvenuto
Priorità	2
Versione	1.0
Note	Pagina di introduzione che attraverso un bottone ci porta alla pagina delle registrazioni
Sotto requisiti	
01	Si necessita del bottone di benvenuto che ci porta alla pagina di registrazione

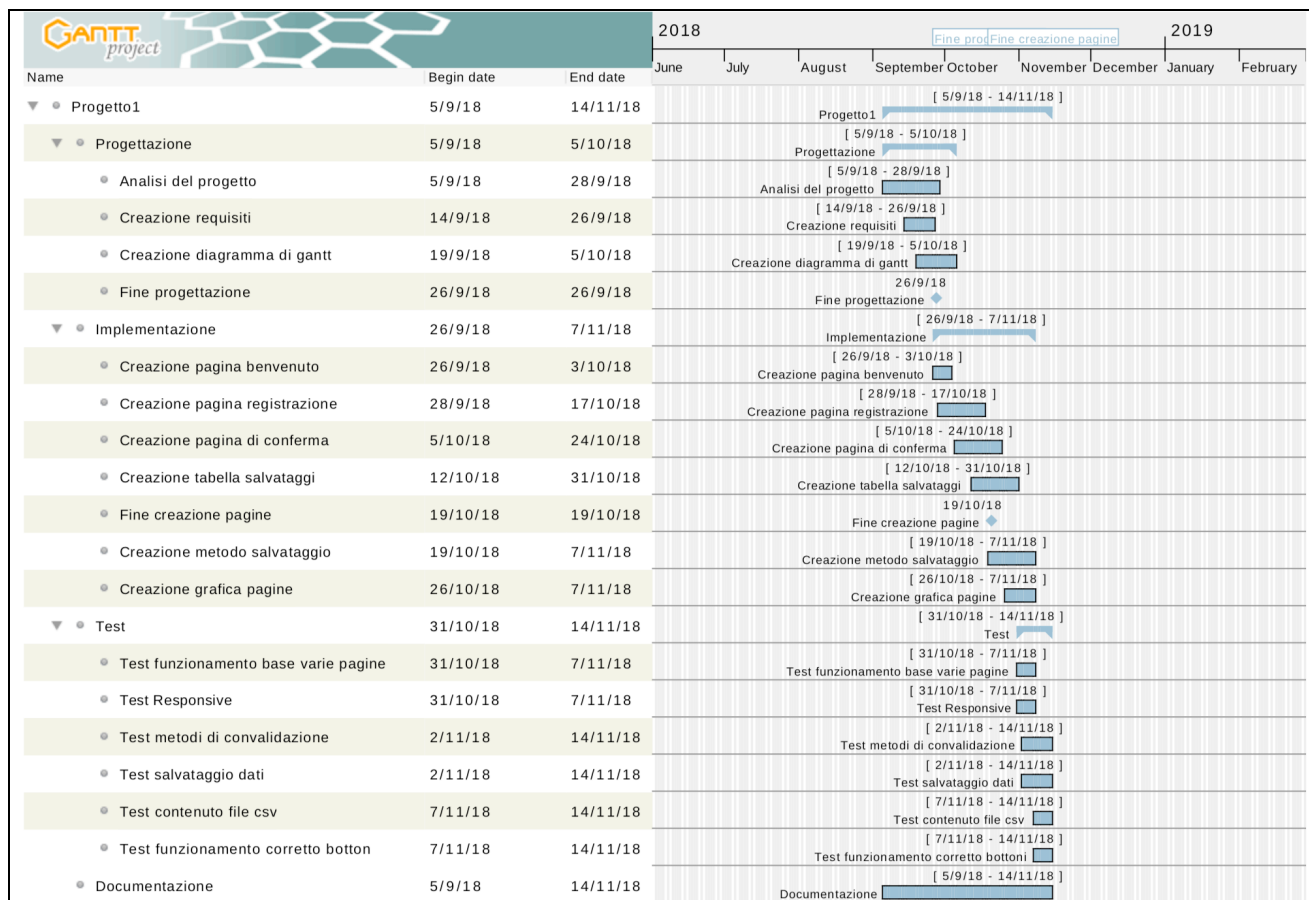
ID: REQ-02	
Nome	Pagina di registrazione
Priorità	1
Versione	1.0
Note	Pagina che gestisce i parametri ricevuti e li convalida attraverso i metodi
Sotto requisiti	
01	Si necessitano dei campi di input obbligatori, segnati con l'asterisco
02	Si necessita dei campi di input facoltativi
03	Si necessita del bottone "cancella" che al click azzerà tutti i contenuti immessi nei textbox
04	Si necessita del bottone "avanti" che al click ci porta alla pagina di conferma
05	Si necessita di metodi che si occupano di convalidare gli input ricevuti nel modo corretto

ID: REQ-03	
Nome	Pagina di conferma
Priorità	2
Versione	1.0
Note	Pagina che mostra tutti i dati dell'utente formattati nel modo corretto, è possibile correggerli
Sotto requisiti	
01	Si necessita del testo immesso nei form durante la registrazione da mostrare sulla pagina
02	Si necessita del bottone "modifica" che al click ci fa tornare alla pagina di registrazione per poter modificare i campi che riteniamo sbagliati
03	Si necessita del bottone "conferma" che al click salva tutti i dati sui file csv

ID: REQ-04	
Nome	Salvataggio su file csv esterni
Priorità	2
Versione	1.0
Note	Si necessita di una macchina lato server che si occupa di salvare i due file con la data corretta
Sotto requisiti	
01	Si dovrà prendere la data del server per poi salvare i file csv con il nome adeguato ricevendo gli input dalla pagina di conferma
02	Si necessita di salvare il primo file chiamato "Registrazioni_tutte.csv" che contiene tutte le registrazioni fatte su questa macchina attraverso questa pagina
03	Si necessita di salvare il secondo file chiamato "Registrazione_aaaa_mm_gg.csv" che contiene tutte le registrazioni che sono state fatte nel giorno contenuto nel nome del csv su questa macchina e pagina

ID: REQ-05	
Nome	Tabella che mostra i dati salvati del giorno
Priorità	2
Versione	1.0
Note	Si necessita di una tabella che prende tutti i dati salvati nel file csv "Registrazione_aaaa_mm_gg.csv" dello stesso giorno e li mostri
Sotto requisiti	
01	Si necessita di visualizzare tutti i dati salvati del giorno in una tabella
02	Si necessita del bottone "home" che al suo click ci riporta alla pagina iniziale

1.6 Pianificazione



1.7 Analisi dei mezzi

1.7.1 Software

Per la realizzazione di questo progetto ho usato come software:

- Apache 2.4.33: Software che simula un server creandone uno virtuale sulla propria macchina accessibile anche dall'esterno.
- PHP 7.2.11: Interprete che si occupa di permettere al server di lavorare con file scritti in PHP.
- JetBrains PhpStorm 2017.2.4: Software che ha fatto da IDE per tutta la creazione del progetto.

Inoltre ho usato le seguenti librerie:

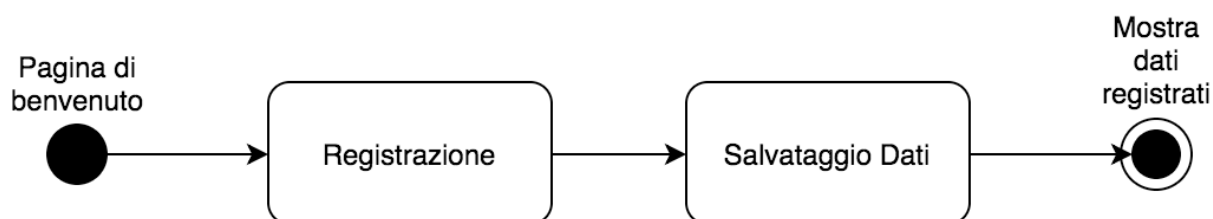
- Bootstrap 4.1.3: Libreria che si occupa di gestire la struttura delle pagine e che si occupa di tutto quello che dev'essere responsive.
- jQuery 3.3.1: Libreria che semplifica l'uso del linguaggio JavaScript.
- Bulma 0.7.2: Libreria CSS che si occupa di gestire parte di tutto l'aspetto grafico delle pagine.
- Font Awesome 5.3.1: Libreria che fornisce le varie icone come quelle presenti nei form.

1.7.2 Hardware

Per questo progetto non ho necessitato di materiale particolare, ho usato il mio portatile MacBook Pro 2015 con il sistema operativo OS X Mojave che faceva sia da client che da server grazie ai software elencati nella parte soprastante.

2 Progettazione

2.1 Design dell'architettura del sistema



2.2 Design delle interfacce

La visualizzazione della tabella di conferma e di quella dei salvataggi inizialmente era la medesima e di conseguenza è rappresentata una sola volta.

Pagina benvenuto

Benvenuto

Clicca qua per registrarti

Pagina iscrizione

Iscriviti

...	<input type="text"/>
...	<input type="text"/>
...	<input type="text"/>
...	<input type="text"/>
...	<input type="text"/>
...	<input type="text"/>
...	<input type="text"/>
...	<input type="text"/>
...	<input type="text"/>

Cancella

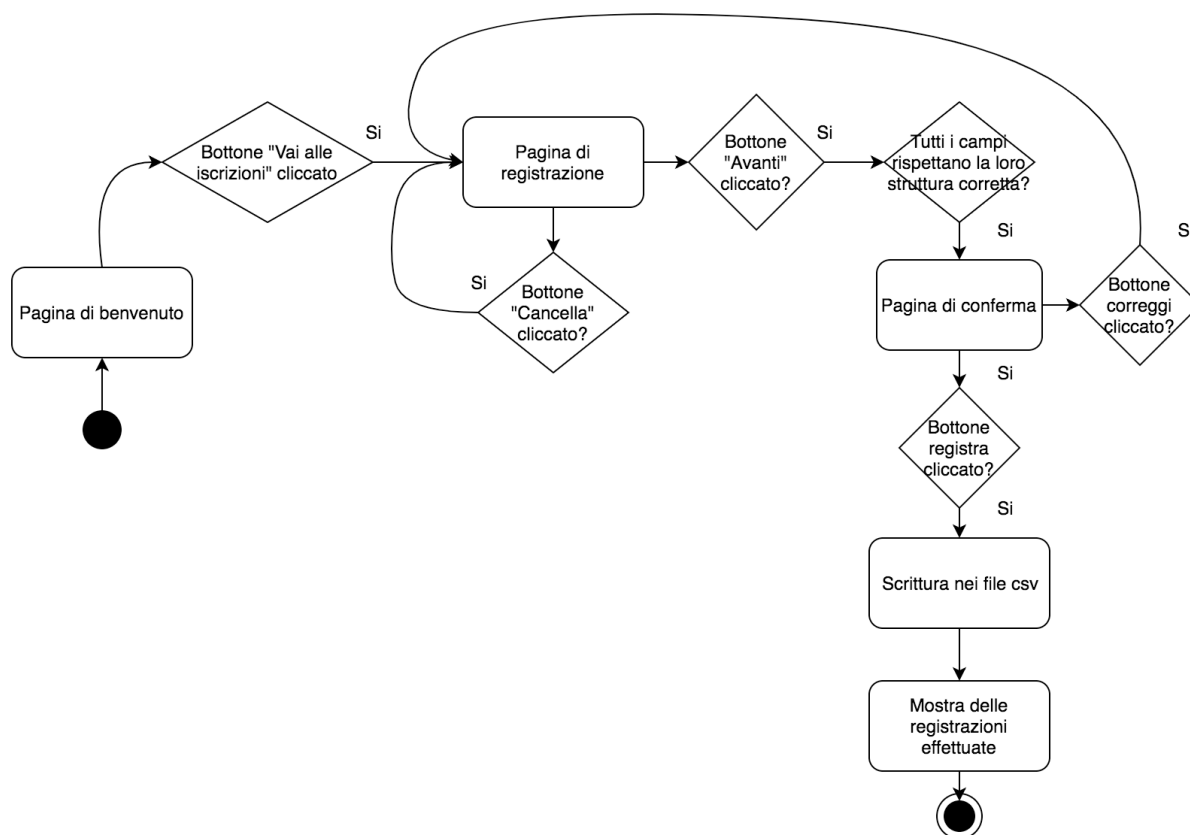
Avanti

Visualizzazione dati

...	...
...
...	...
...	...
...	...
...	...
...	...
...	...
...	...
...	...
...	...
...	...
...	...
...	...

Torna alla home

2.3 Design procedurale



3 Implementazione

3.1 Pagina di benvenuto

Pagina di benvenuto che ha lo scopo di portare l'utente alla pagina di registrazione.
La sua struttura è molto semplice di conseguenza non necessita spiegazioni approfondite, ecco come appare:

Benvenuto!



© All credits go to Jari Näser























3.2 Pagina di registrazione

La pagina di registrazione è una delle due componenti principali del progetto, infatti si occupa di ricevere tutti gli input dell'utente attraverso i suoi 11 Form.

Inoltre controlla sul lato client tutti i dati ricevuti e in caso di errore o valore non accettato lo segnala.

Infine è possibile correggere i propri dati immessi attraverso la tabella della verifica dei dati.

Registrati

 * Nome 	 * Cognome 
 * Data Nasc 	 * No. Civico 
 * Città 	 * Nap 
 * No. Telefono 	 * E-mail 
 * Genere 	
 Hobby 	
 Professione 	
<div>Cancella</div> <div>Avanti</div>	

 Help 

3.2.1 Form inserimento dati e controlli

Tutto il codice di questa pagina è scritto in HTML, CSS e JavaScript

3.2.1.1 Metodi di controllo

Questa prima parte si occupa di prendere il testo immesso negli input e di convalidarlo attraverso il metodo di convalidazione principale (testInput).

```

/*
    Questo metodo si occupa di testare il testo nell'input ad ogni
    cambiamento di esso (es. Aggiunta di un carattere) e ritorna
    il valore true se la stringa immessa è accettata e false se non
    lo è.

    Parametro regex:      Caratteri accettati nella stringa passata.
    Parametro text:       Il testo preso dall'input.
    Parametro validatedNum: Numero da 0-8 che andrà a mettere un
                           valore in un'array a quella posizione.
*/
function testInput(regex, text, validatedNum){
    //Viene testato ogni singolo carattere.
    for(var i = 0; i < text.length; i++){
        if(regex.test(text[i])){
            //Assegna il valore false alla posizione validatedNum
            //all'array validated.
            validated[validatedNum] = false;
            return false;
        }
    }
    //Assegna il valore true alla posizione validatedNum
    //all'array validated.
    validated[validatedNum] = true;
    return true;
}

```

3.2.1.2 Metodi feedback

Inoltre grazie al risultato *true* o *false* che ci da il metodo `testInput` possiamo dare un feedback all'utente, questo viene fatto attraverso due icone della Font Awesome e due colori simili al rosso e al verde:

```
//Caratteri Font Awesome
const TRUE_CHAR = 'fa-check';
const FALSE_CHAR = 'fa-times';
//Colori
const TRUE_COLOR = '#abdd92';
const FALSE_COLOR = '#f76a6a';
```



Il metodi che si occupano di assegnare l'icona corretta e del colore giusto nel suo spazio apposito nel Form sono i seguenti:

```
/*
    Questo metodo si occupa di richiamare i metodi setTrue(id)
    o setFalse(id) a dipendenza dello stato del metodo.

    Parametro id:      Definisce il nome dell'elemento da modificare.
    Parametro method:  Riceve il metodo validatore dal quale deve
                       ricevere la risposta se la stringa è accettata
                       o meno.
*/

function setIcon(id, method){
    if(method){
        setTrue(id);
    }else{
        setFalse(id);
    }
}

/*
    Questo metodo mette il carattere "visto" con il suo colore che segna
    lo stato dell'input immesso.

    Parametro id: Definisce il nome dell'elemento da modificare
*/
function setTrue(id){
    $(id).removeClass(FALSE_CHAR);
    $(id).addClass(TRUE_CHAR);
    $(id).css('color', TRUE_COLOR);
}

/*
    Questo metodo mette il carattere "croce" con il suo colore che segna
    lo stato dell'input immesso.

    Parametro id: Definisce il nome dell'elemento da modificare
*/
function setFalse(id){
    $(id).removeClass(TRUE_CHAR);
    $(id).addClass(FALSE_CHAR);
    $(id).css('color', FALSE_COLOR);
}
```

Infine i risultati di questi metodi sono i seguenti:



3.2.1.3 Bottoni Cancella e Avanti

Come ultimi elementi della pagina ci sono i due bottoni “Cancella” e “Avanti” che hanno le seguenti funzionalità:

Bottone cancella: Si occupa di ripristinare tutto il contenuto presente nei Form e di rimettere tutte le icone indicatrici a falso.

Azzeramento valori:

Essendo contenuto tutto in un Form basta mettere type=reset.

```
<button name="cancella" type="reset" class="col-md-3" id="buttonCancella">Cancella</button>
```

Ripristino icone indicatrici:

Semplicemente utilizzo il metodo setFalse(id) visto precedentemente per ogni input.

Bottone avanti: Si occupa di controllare se tutti i valori obbligatori sono stati immessi correttamente e se si al click mostra la tabella contenente i dati immessi.

3.2.2 Tabella verifica dati

Tabella che contiene i dati inseriti nei Form formattati nel modo corretto con i due bottoni “Correggi” e “Registra”, il suo scopo principale è quello di permettere all’utente di rivedere i suoi dati immessi prima di andare a salvarli dando così la possibilità di poter ancora modificarli.

Dati inseriti

Nome	Jari
Cognome	Näser
Data di nascita	2001-10-24
Numero civico	10
Città	Lugano
Nap	6900
Numero di telefono	079 123 45 67
E-mail	jari.naeser@test.ch
Genere	M
Hobby	Hockey
Professione	-

Correggi

Registra

3.2.2.1 Tabella

La tabella viene generata utilizzando il DOM e prendendo i valori convalidati dagli input.

3.2.2.2 Bottoni “Correggi” e “Registra”

Bottone correggi: Questo bottone ha lo scopo di riportarci ai Form in cui abbiamo immesso i nostri dati precedentemente mantenendo i testi immessi offrendo così la possibilità di modificare un campo che eventualmente è stato inserito nel modo sbagliato.

Quest'azione viene fatta nascondendo la tabella con il metodo .hide() di jQuery e mostrando gli input attraverso il metodo .show() della medesima libreria.

Bottone registra: Questo bottone ha lo scopo di prendere i dati convalidati e di mandarli alla classe php che andrà poi a scriverli sui file.

Questo processo viene fatto attraverso il metodo \$.ajax di jQuery.

```
$.ajax({
  type: "POST",
  url: "Saves.php",
  data: {
    nome: nome,
    cognome: cognome,
    dataNascita: dataNascita,
    noCivico: noCivico,
    citta: citta,
    nap: nap,
    noTelefono: noTelefono,
    email: email,
    genere: genere,
    hobby: hobby,
    professione: professione
  },
  success: function(data){
    console.log("Transfer of " + data + " is OK.");
    document.open();
    document.write(data);
    document.close();
  },
  error: function(data){
    console.log("Couldn't transfer " + data);
  }
});
```

3.3 Pagina di salvataggio

Questa pagina ha come scopo principale quello di ricevere tutti i dati, riconvalidarli lato server e se tutto va a buon fine scriverli in due file csv.

Inoltre alla fine di queste azioni mostra i dati salvati nella giornata stessa della propria registrazione.

REGISTRAZIONI EFFETTUATE OGGI

2018-11-4

Nome	Cognome	Data di Nascita	Numero Civico	Città	NAP	Numero di Telefono	E-Mail	Genere	Hobby	Professione	Data e Ora Salvataggio
Jari	Näser	2001-10-24	10	Lugano	6900	079 123 45 67	jari.naaser@test.ch	M	Hockey	-	2018-11-04 22:21:04

[Torna alla Home](#)

3.3.1 Ricezione e controllo dati

3.3.1.1 Ricezione dati

I dati vengono presi attraverso il tipo di request post e ognuno messo in una costante, inoltre viene anche fatto l'htmlspecialchars che interpreta i caratteri speciali diversamente dal solito facendo di modo che non sia possibile sfruttare una certa sequenza di caratteri per scopi malefici; Aumenta la sicurezza del prodotto.

```
define(NOME, htmlspecialchars($_POST['nome']));
define(COGNOME, htmlspecialchars($_POST['cognome']));
define(DATA_NASCITA, htmlspecialchars($_POST['dataNascita']));
define(NO_CIVICO, htmlspecialchars($_POST['noCivico']));
define(CITTA, htmlspecialchars($_POST['citta']));
define(NAP, htmlspecialchars($_POST['nap']));
define(NO_TELEFONO, htmlspecialchars($_POST['noTelefono']));
define(EMAIL, htmlspecialchars($_POST['email']));
define(GENERE, htmlspecialchars($_POST['genere']));
define(HOBBY, htmlspecialchars($_POST['hobby']));
define(PROFESSIONE, htmlspecialchars($_POST['professione']));
```


3.3.1.2 Controllo dati

Per ogni costante c'è un metodo che si occupa di controllare il suo contenuto, il suo funzionamento è identico a quello del metodo di controllo lato client a patto che è scritto in php e che come già menzionato ce n'è uno per ogni valore.

Esempio metodo validatore per il nome:

```
function valNome(){
    if(!isNullOrEmpty(NOME)){
        if(strlen(NOME) > 0){
            if(NOME[0] == '-' || NOME[0] == '.'){
                return false;
            }
            if(preg_match("/([A-Za-zöäüÖÄÜàèìòùÀÈÌÒÙ -])/", NOME)){
                $valStatus[0] = true;
                return true;
            }
        }
    }
    $valStatus[0] = false;
    return false;
}
```

3.3.2 Uso file csv

3.3.2.1 Creazione percorso e file

Come primo passo il programma controlla se esiste la cartella "Registrazioni" nel punto in cui si trovano anche tutte le pagine del prodotto, se esiste ci entra altrimenti la crea e ci entra.

Una volta dentro la cartella controlla se esistono già i due file "Registrazioni_tutte.csv" e "Registrazioni_aaaa-mm-gg.csv", se non sono ancora presenti li crea e li inizializza.

Inoltre ogni giorno viene creato un nuovo file "Registrazioni_aaaa-mm-gg.csv".

Questo processo viene fatto con questo metodi:

```
//Crea la cartella "Registrazioni" se non esiste ancora.
if(!file_exists('Registrazioni')){
    mkdir('Registrazioni', 0777, true);
}

//Crea il file "Registrazioni_tutte.csv" se non esiste ancora.
if(!file_exists(REG_TUTTE)){
    $fileAll = fopen(REG_TUTTE, "a") or die("Unable to open " . REG_TUTTE . "!");
    fwrite($fileAll, initializeCSV());
}

//Crea il file "Registrazioni_aaaa-mm-dd.csv" se non esiste ancora.
if(!file_exists(REG_OGGI)){
    $fileToday = fopen(REG_OGGI, "a") or die("Unable to open " . REG_OGGI . "!");
    fwrite($fileToday, initializeCSV());
}
```

3.3.2.2 Scrittura nei file csv

La scrittura viene fatta attraverso la funzione fopen(), fwrite() e fclose().

```
//Apri i due file csv in modalità append.
$fileAll = fopen(REG_TUTTE, "a") or die("Unable to open " . REG_TUTTE . "!");
$fileToday = fopen(REG_OGGI, "a") or die("Unable to open " . REG_OGGI . "!");

//Scrivi nei due file csv
//Il parametro addToCsv() è la stringa che vogliamo aggiungere ai file.
if(strlen(NOME) > 0){
    fwrite($fileAll, addToCsv());
    fwrite($fileToday, addToCsv());
}

//Chiudi i due file csv
fclose($fileAll);
fclose($fileToday);
```

3.3.2.3 Lettura dai file csv

La lettura viene fatta con la funzione fgetcsv(), inoltre i contenuti ricavati vengono aggiunti ad una tabella che poi verrà aggiunta alla pagina delle registrazioni effettuate attraverso il DOM.

Il metodo che si occupa di fare tutto ciò è il reader():

```
function reader(){
    //Preparazione tabella
    $table = "<div style='overflow-x:auto;'><table><tr><th>Nome</th><th>Cognome</th><th>Data di  
Nascita</th>  
        <th>Numero Civico</th><th>Città</th><th>NAP</th><th>Numero di Telefono</th><th>E-Mail</th>  
        <th>Genere</th><th>Hobby</th><th>Professione</th><th>Data e Ora Salvataggio</th></tr>";

    //Apri il file delle registrazioni di oggi in modalità lettura
    $file = fopen(REG_OGGI, "r");

    if(count(fgetcsv($file)) > 0){
        while (($row = fgetcsv($file, 1000, SEPARATOR)) !== FALSE) {
            //Con questo controllo le linee contenenti caratteri intercettati dal htmlspecialchars
            //non verranno indicate nella tabella (perché stiamo usando il ';' come separatore).
            if(count($row) == 12){
                $table .= "<tr>";
                for($i = 0; $i < 12; $i++){
                    $table .= "<td>" . $row[$i] . "</td>";
                }
                $table .= "</tr>";
            }
        }
        fclose($file);
    }

    $table .= "</table></div>";

    echo $table;
}
```

3.3.2.4 Bottone torna alla home

Lo scopo di questo bottone è semplicemente di riportarci alla pagina di benvenuto, non esegue azioni speciali.

4 Test

4.1 Protocollo di test

Test Case:	TC-001	Nome:	Funzionamento bottone “registrati”.
Riferimento:	REQ-001 sub01		
Descrizione:	Prova del funzionamento corretto del bottone “registrati” che sta al centro della pagina sotto la scritta “Benvenuto!”.		
Prerequisiti:	Pagina di benvenuto funzionante. Bottone “registrati” esistente. Codice di background del bottone registrati che ci porta alla pagina di registrazione.		
Procedura:	1. Andare sulla pagina di benvenuto 2. Cliccare il bottone “registrati”		
Risultati attesi:	Al click del bottone veniamo portati alla pagina “Registrazioni.html”.		

Test Case:	TC-002	Nome:	Controllo validazione input “nome”
Riferimento:	REQ-002 sub05		
Descrizione:	Prova del funzionamento della validazione del campo “nome”.		
Prerequisiti:	Pagina di registrazione funzionante. Form “nome” esistente. Codice di background del Form “nome” che si occupa di testare la validità del testo inserito.		
Procedura:	1. Andare sulla pagina di registrazione 2. Inserire il testo “Jari” nel Form “nome” per poter eseguire il test		
Risultati attesi:	L'icona alla destra del Form deve diventare verde e deve apparire un visto.		

Test Case:	TC-003	Nome:	Funzionamento bottone “cancella”.
Riferimento:	REQ-002 sub03		
Descrizione:	Prova del funzionamento corretto del bottone “cancella” che sta in fondo ai Form della pagina di registrazione.		
Prerequisiti:	Pagina di registrazione funzionante. Bottone “cancella” esistente. Codice di background del bottone “cancella” che azzerà tutti i contenuti dei vari Form.		
Procedura:	1. Andare sulla pagina di registrazione 2. Inserire nei vari Form dati per poter eseguire i test 3. Cliccare il bottone “cancella”		
Risultati attesi:	Tutti gli input immessi nei Form si azzerano e tornano al loro stato iniziale senza contenere alcun dato.		

Documentazione Progetto Registrazione

Test Case:	TC-004	Nome:	Funzionamento bottone “avanti”.
Riferimento:	REQ-002 sub04		
Descrizione:	Prova del funzionamento corretto del bottone “avanti” che sta in fondo ai Form della pagina di registrazione.		
Prerequisiti:	Pagina di registrazione funzionante. Bottone “avanti” esistente. Codice di background del bottone “avanti” che manda tutti i contenuti dei Form alla tabella di conferma.		
Procedura:	1. Andare sulla pagina di registrazione 2. Inserire nei vari Form dati per poter eseguire i test 3. Cliccare il bottone “avanti”		
Risultati attesi:	Tutti gli input immessi nei Form vengono controllati e se sono accettati vengono mandati alla tabella di conferma		

Test Case:	TC-005	Nome:	Funzionamento bottone “correggi”.
Riferimento:	REQ-003 sub02		
Descrizione:	Prova del funzionamento corretto del bottone “correggi” che sta in fondo alla tabella di conferma.		
Prerequisiti:	Tabella di conferma funzionante. Bottone “correggi” esistente. Codice di background del bottone “correggi” che ci riporta sulla pagina di registrazione.		
Procedura:	1. Andare sulla tabella di registrazione 2. Cliccare il bottone “correggi”		
Risultati attesi:	La visualizzazione torna sulla pagina di registrazione con i Form che contengono ancora i dati inseriti precedentemente		

Test Case:	TC-006	Nome:	Funzionamento bottone “registra”.
Riferimento:	REQ-003 sub03		
Descrizione:	Prova del funzionamento corretto del bottone “registra” che sta in fondo alla tabella di conferma.		
Prerequisiti:	Tabella di conferma funzionante. Bottone “registra” esistente. Codice di background del bottone “registra” che prende i dati dai Form, li controlla attraverso i metodi di convalidazione e li manda al file Saves.php.		
Procedura:	1. Andare sulla tabella di registrazione 2. Cliccare il bottone “registra”		
Risultati attesi:	Il contenuto dei Form viene mandato al file Saves.php		

Test Case:	TC-007	Nome:	Creazione file "Registrazioni_tutte.csv"
Riferimento:	REQ-004 sub02		
Descrizione:	Prova del funzionamento corretto della creazione del file "Registrazioni_tutte.csv".		
Prerequisiti:	Cartella "Registrazioni" esistente.		
Procedura:	1. Andare sulla pagina Saves.php		
Risultati attesi:	Il file "Registrazioni_tutte.csv" viene creato nella cartella "Registrazioni".		

Test Case:	TC-007	Nome:	Creazione file "Registrazioni_aaaa-mm-dd.csv"
Riferimento:	REQ-004 sub03		
Descrizione:	Prova del funzionamento corretto della creazione del file "Registrazioni_aaaa-mm-dd.csv".		
Prerequisiti:	Cartella "Registrazioni" esistente.		
Procedura:	1. Andare sulla pagina Saves.php		
Risultati attesi:	Il file "Registrazioni_aaaa-mm-dd.csv" viene creato nella cartella "Registrazioni".		

Test Case:	TC-008	Nome:	Lettura dai file csv
Riferimento:	REQ-005 sub01		
Descrizione:	Prova del funzionamento dell'algoritmo che si occupa di estrapolare i dati salvati sui file csv e di aggiungerli in una tabella.		
Prerequisiti:	Pagina di salvataggi funzionante. File csv contenente qualcosa Codice di background che si occupa di leggere i dati dal file csv		
Procedura:	1. Andare sulla pagina dei salvataggi		
Risultati attesi:	Gli elementi del file csv vengono interpretati nel modo corretto e il programma ritorna una tabella contenente essi.		

4.2 Risultati test

Test Case	Numero Passaggio	Risultato
TC-001	1-2	Il bottone ci porta alla pagina di registrazione
TC-002	1-2	La convalidazione del campo nome funziona correttamente
TC-003	1-3	Il funzionamento del bottone cancella è corretto
TC-004	1-3	Il funzionamento del bottone avanti è corretto
TC-005	1-2	Il funzionamento del bottone correggi è corretto
TC-006	1-2	Il funzionamento del bottone registra è corretto
TC-007	1	La creazione dei file csv viene eseguita nel modo corretto
TC-008	1	La lettura dei dati dai file csv viene eseguita nel modo corretto

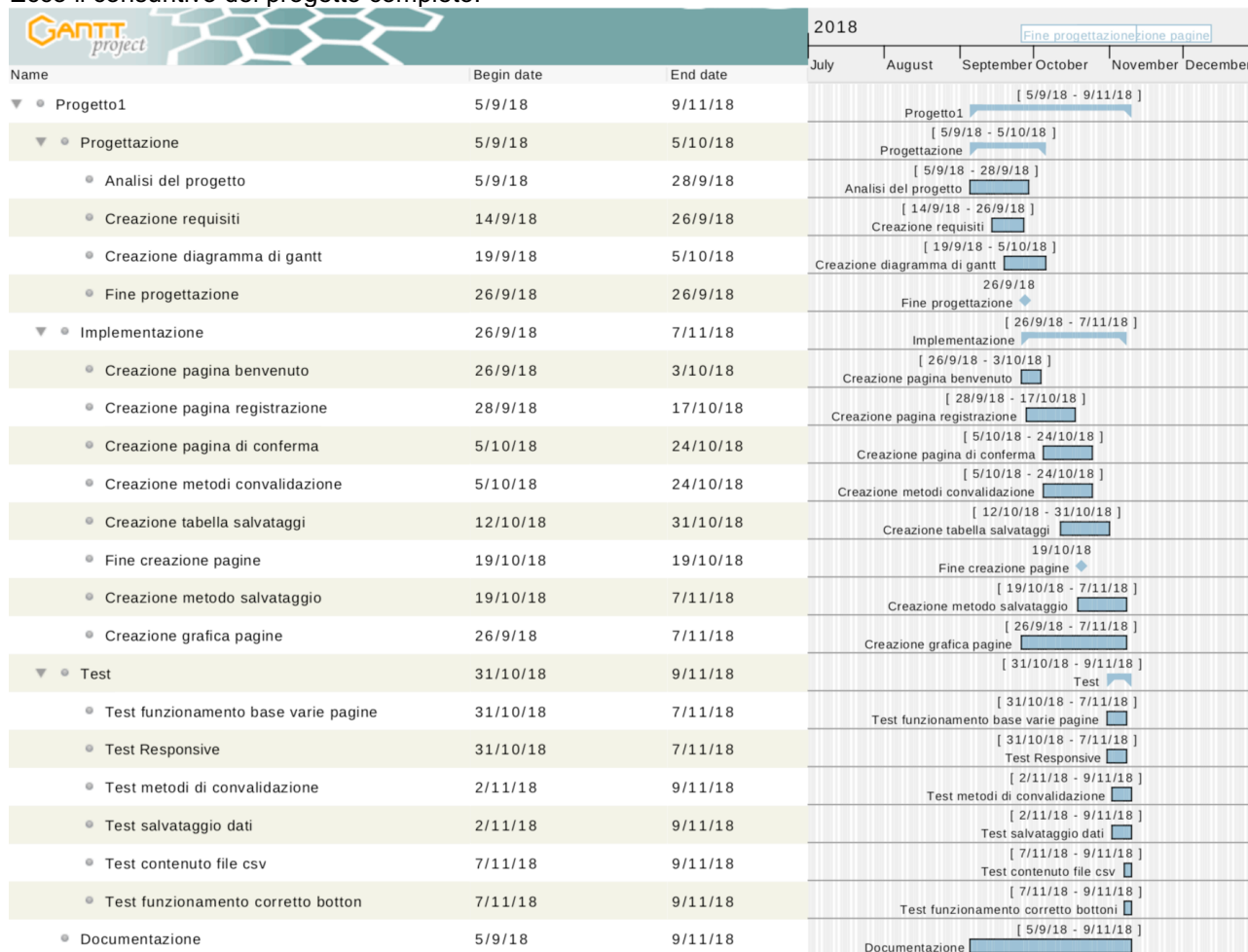
4.3 Mancanze/limitazioni conosciute

Il tipo di Form “date” che al click apre un calendario che su mobile per un qualche motivo non funziona, non lascia immettere alcun testo nel campo o selezionare una data.

Di conseguenza non l'ho potuto implementare e ho tenuto il campo di tipo testo.

5 Consuntivo

Sono riuscito a mantenere le specifiche del progetto e ad organizzarmi bene con i tempi.
Non ho sfasato con i tempi e sono riuscito a dare anche cura ai dettagli.
Ecco il consuntivo del progetto completo:



6 Conclusioni

Questo prodotto offre un sistema di registrazione solido, semplice ed intuitivo da usare che facilita molto l'uso di registrazioni in vari ambiti come quello web oppure scolastico.

6.1 Sviluppi futuri

Sicuramente è ancora possibile approfondire sulla sua sicurezza ed interfaccia ma generalmente è molto compatto e quindi abile di essere integrato in futuri progetti oppure siti web.

6.2 Considerazioni personali

Ho imparato l'importanza della progettazione, documentazione e calcolo dei tempi necessari per ottenere un progetto anche relativamente piccolo come uno di questo calibro, se si fanno bene tutte e tre è già un aiuto enorme per mettere in piedi un progetto strutturato in un modo molto buono.

Inoltre mi è piaciuta molto la parte dell'implementazione che ci ha permesso di essere liberi sull'utilizzo di componenti, librerie e linguaggi.

Ho trovato molto interessante l'aspetto di collegare vari linguaggi per ottenere un progetto singolo, di sfruttare le potenziali di ogni linguaggio per uno scopo specifico e di farli interagire assieme.

7 Bibliografia

7.1 Sitografia

- <http://stackoverflow.com/>, *StackOverFlow*, dal 05.09 al 9.11
- <http://www.w3schools.com/>, *W3Schools*, dal 05.09 al 9.11
- <https://bulma.io/>, *Bulma CSS*, dal 05.09 al 9.11
- <https://getbootstrap.com/docs/4.1/getting-started/introduction/>, *Bootstrap*, dal 05.09 al 9.11
- <http://php.net/>, *PHP*, dal 05.09 al 9.11

8 Allegati

- Diari di lavoro
- Codici sorgente
- Quaderno dei compiti
- Prodotto