Documentazione Progetto 1

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Abstract 3

1.3 Scopo 3

Analisi 4

1.4 Analisi del dominio 4

1.5 Analisi e specifica dei requisiti 4

1.6 Use case 6

1.7 Pianificazione 6

1.8 Analisi dei mezzi 6

1.8.1 Software 6

1.8.2 Hardware 6

2 Progettazione 7

2.1 Design dell’architettura del sistema 7

2.2 Design dei dati e database 7

2.3 Design delle interfacce 7

2.4 Design procedurale 7

3 Implementazione 8

4 Test 8

4.1 Protocollo di test 8

4.2 Risultati test 9

4.3 Mancanze/limitazioni conosciute 9

5 Consuntivo 9

6 Conclusioni 9

6.1 Sviluppi futuri 9

6.2 Considerazioni personali 9

7 Bibliografia 9

7.1 Bibliografia per articoli di riviste: 9

7.2 Bibliografia per libri 9

7.3 Sitografia 9

8 Allegati 10

# Introduzione

## Informazioni sul progetto

Questo è il primo progetto guidato fatto da Fadil Smajilbasic per la SAMT. I docenti responsabili sono Muggiasca Luca, Mussi Francesco, Elisa Nannini.

Il progetto fa parte del modulo 306.

## Abstract

The client needed an application to store user data in a specific format. There are currently other solutions from Google or Microsoft, but the client needs to store the data in a specific format not used by the solutions from other companies. The client wants a to have a global file where all the registrations will be stored and a file for every day with the registrations of that day. Both files must be of the CSV format with the default separator “;”. Making the actual programs isn’t a big deal, the focus of this project is the documentation that comes with the finished product.

## Scopo

Lo scopo di questo progetto è di imparare a come scrivere una documentazione, analizzare i requisiti e progettare un progetto.

## Analisi

## Analisi del dominio

Al momento esistono delle soluzioni di Google e soluzioni della Microsoft ma soddisfano i bisogni del committente. Il prodotto deve essere accessibile da un client web e che da la possibilità a degli allievi di un istituto di registrarsi a esso. I dati degli allievi verranno salvati in un file CSV dopo una convalidazione di tutti i dati inseriti. Questo sito è previsto di essere usato da un utente standard con conoscenze basilari di come si usa un pc. Per questo sito esistono delle convenzioni per la convalidazione dei dati come lo standard per la forma delle e-mail, per la selezione del sesso, la forma del numero telefonico e la struttura del NAP.

## Analisi e specifica dei requisiti

Il committente ha bisogno di un formulario di registrazione che salva i dati degli alunni di una scuola, i dati dovranno essere salvati in un file CSV giornaliero e uno generale che contiene tutti i dati. Ci dovrà essere un controllo della esistenza di quelli due file. Dopo l’inserimento dei dati l’utente verrà portato ad una pagina riassuntiva dove potrà con un bottone ritornare al formulario per correggere dei dati oppure continuare. Dopo la pagina che riassume i dati, essi verranno salvati nei file e verrà presentata una pagina con i dati letti da uno dei file.

|  |  |
| --- | --- |
| **ID: REQ-01** | |
| **Nome** | Form di registrazione |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |
| **Sotto requisiti** | |
| **001** | Controllo della validità dei dati inseriti |
| **002** | Si dovranno poter immettere nuovi allievi |
| **003** | Tasto “Avanti” convalida i dati |
| **004** | Campi obbligatori dovranno essere segnati con l’asterisco |
| **005** | Alcuni dati avranno una formattazione specifica che dovrà essere rispettata |

|  |  |
| --- | --- |
| **ID: REQ-02** | |
| **Nome** | Pagina di benvenuto |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** |  |
| **Sotto requisiti** | |
| **001** | Tasto “avanti” che porta al form |

|  |  |
| --- | --- |
| **ID: REQ-03** | |
| **Nome** | Salvataggio dei dati |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |
| **Sotto requisiti** | |
| **001** | Se il file non è esistente crea il file |
| **002** | Salvataggio dei dai in 2 file CSV, uno Generale che contiene tutte le registrazioni e uno del giorno corrente |

|  |  |
| --- | --- |
| **ID: REQ-04** | |
| **Nome** | Tabella Riassuntiva |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** |  |
| **Sotto requisiti** | |
| **001** | Riassumere i dati in una tabella |
| **002** | Tasto con la possibilità di modificare i dati |
| **003** | Tasto Registra che salva i file |

|  |  |
| --- | --- |
| **ID: REQ-05** | |
| **Nome** | Struttura dei file |
| **Priorità** | 3 |
| **Versione** | 1.0 |
| **Note** | Avere una gerarchia dei file salvati |
| **Sotto requisiti** | |

## Use case

I casi d’uso rappresentano l’interazione tra i vari attori e le funzionalità del prodotto.

## Pianificazione

|  |
| --- |
| Figura 1: Diagramma di Gantt. |

## Analisi dei mezzi

### Software

I seguenti software sono stati utilizzati:

* Microsoft Word 2016
* Microsoft Visual Studio Code
* Apache 2.4.27
* PHP 7.0.23
* Google Chrome 69.0.3497.100
* GanttProject 2.8.9
* Windows 10 Version 1803 (Build 17134.286)

### Hardware

Per lo sviluppo e testing è stata usato un Asus ROG GL502VS con le seguenti specifiche:

CPU: i7-7700HQ

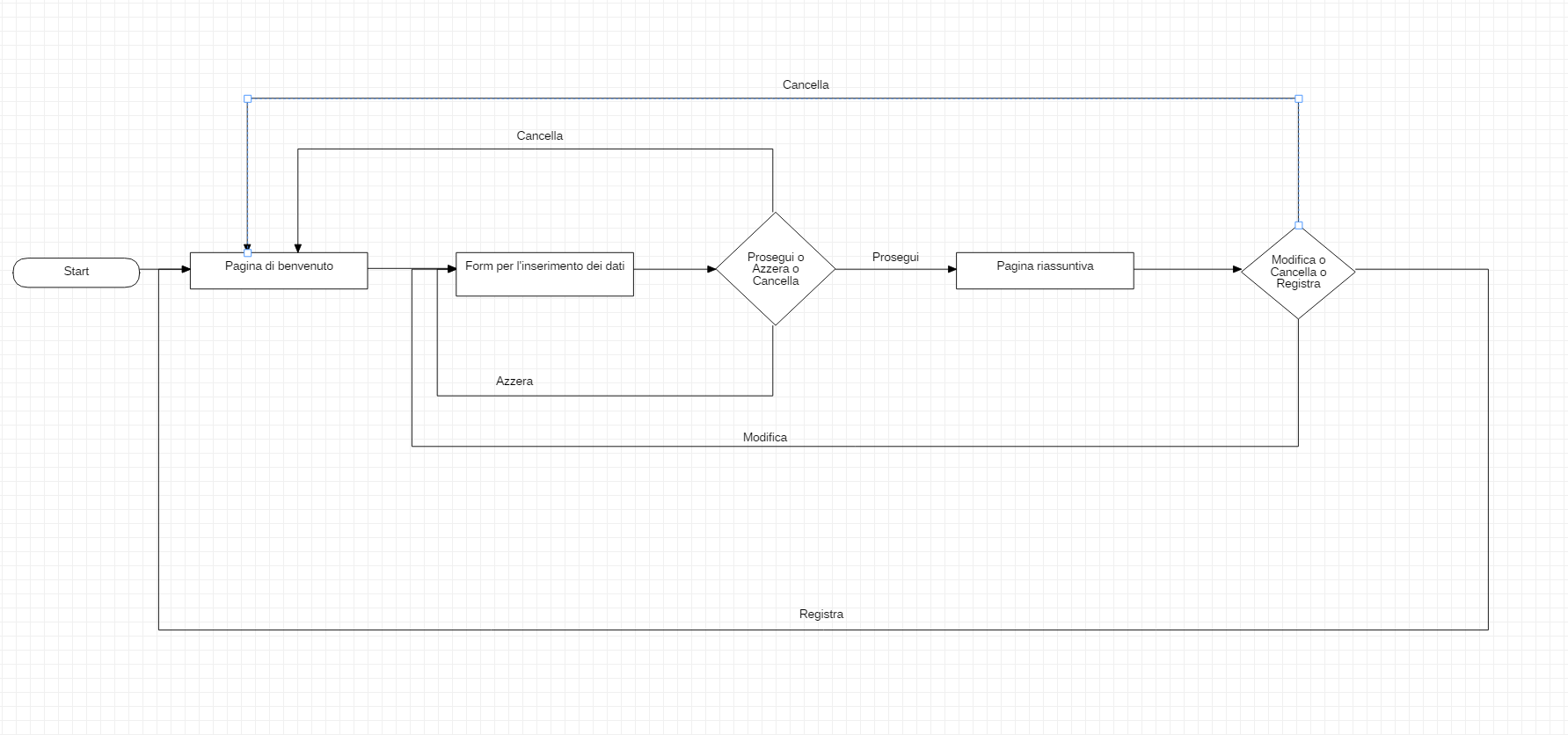
GPU: Nvidia GTX 1070

RAM: 16GB

Scheda Madre Proprietaria

# Progettazione

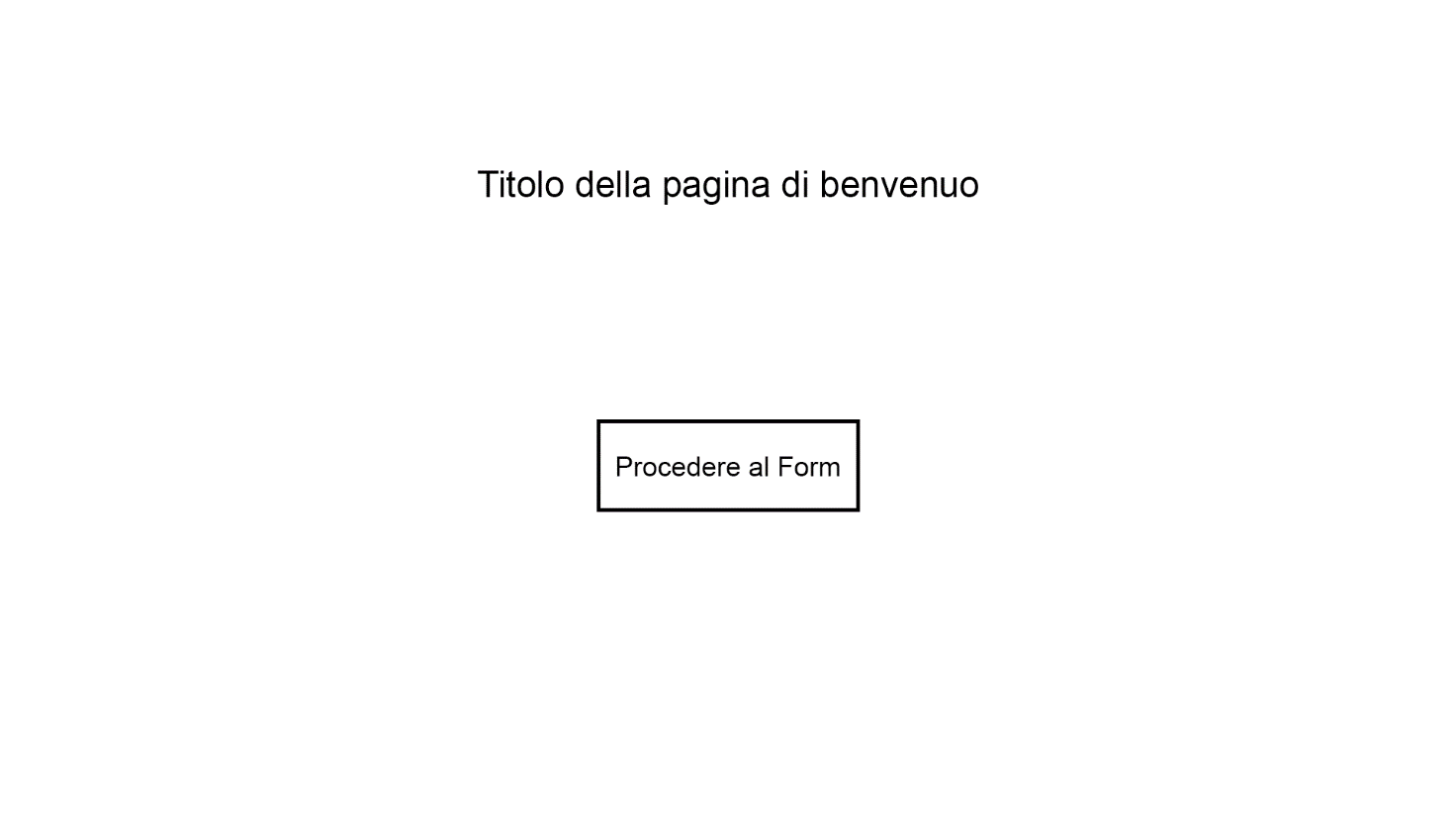
## Design dell’architettura del sistema

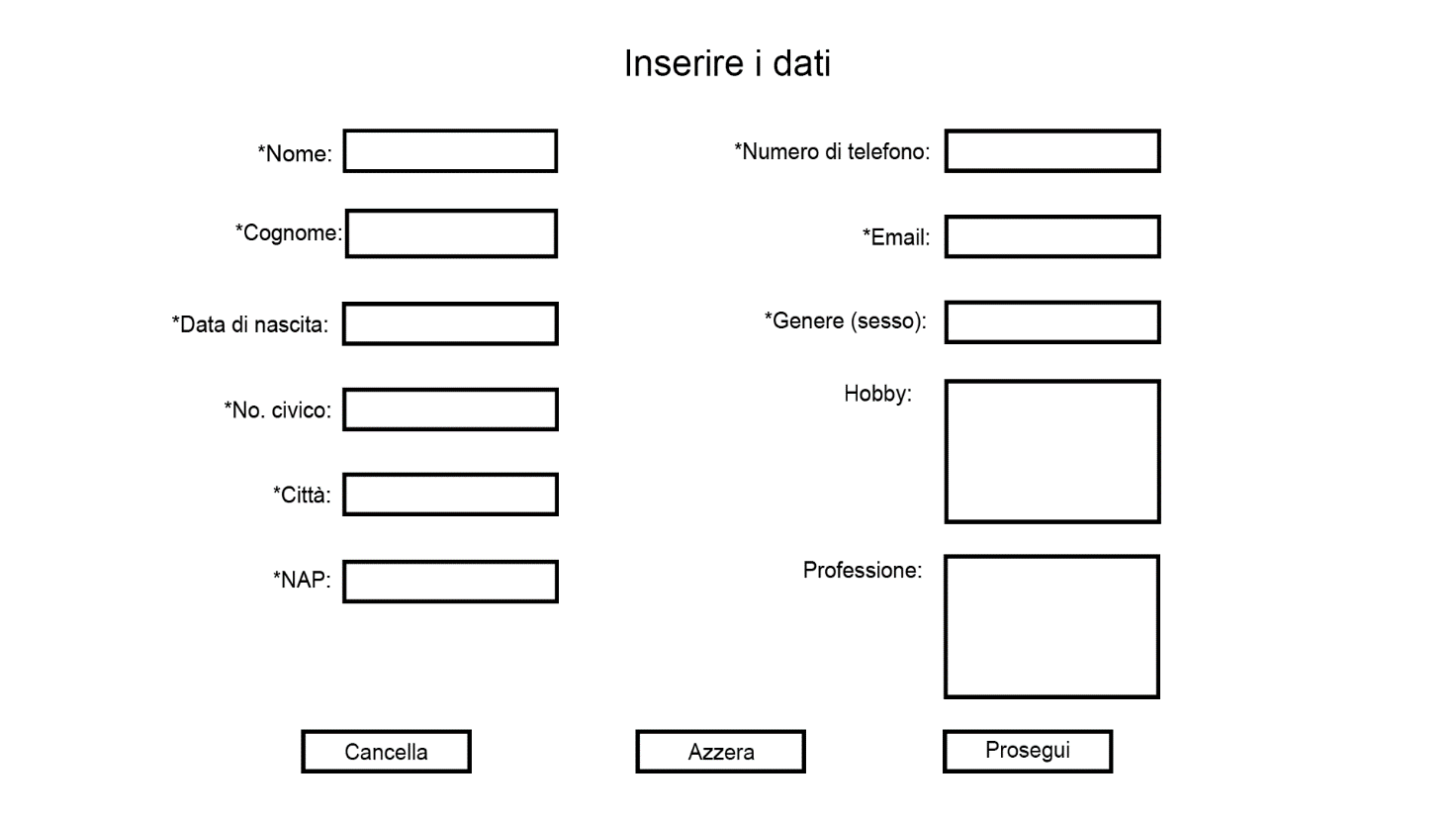


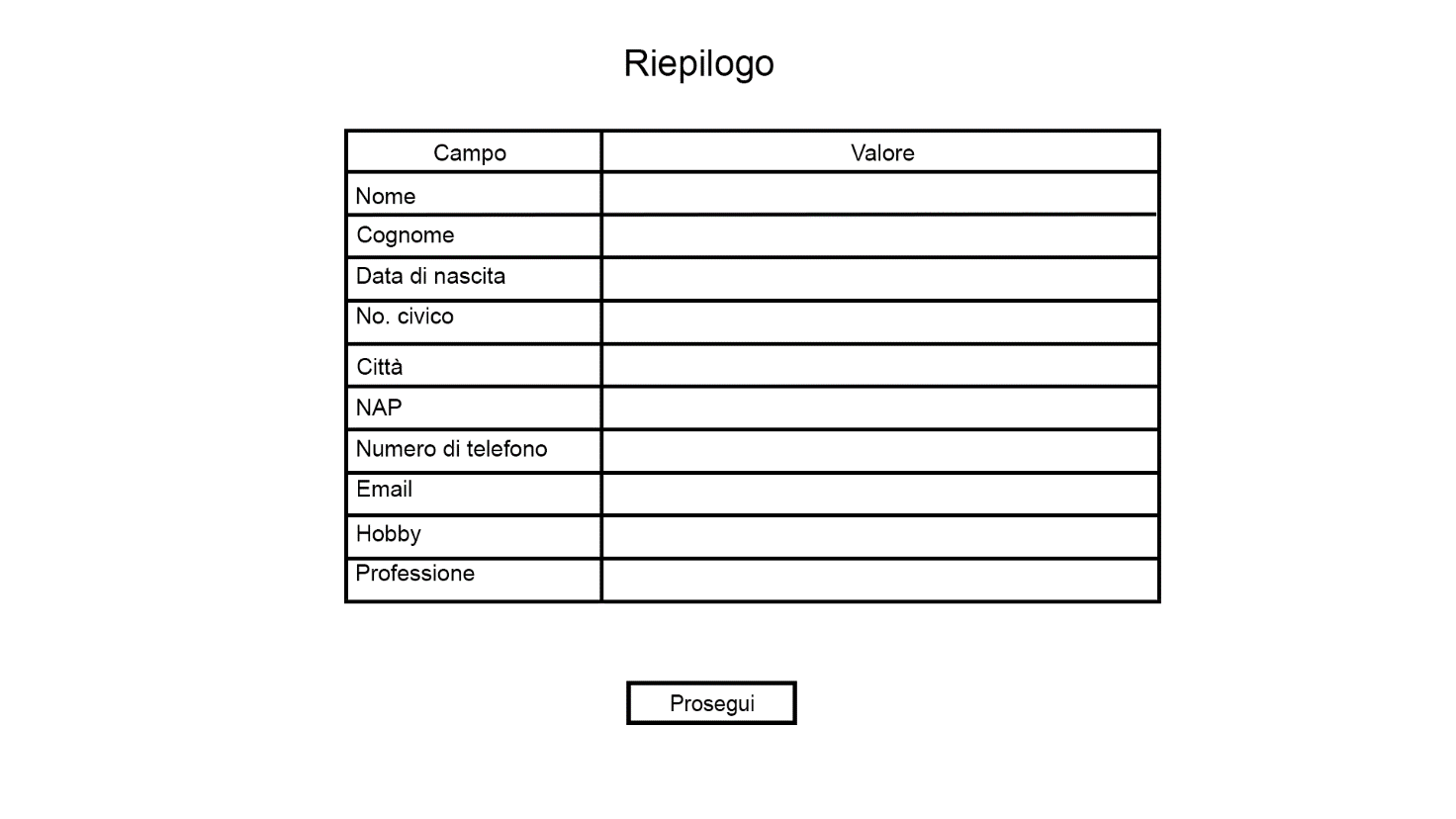
## Design dei dati e database

Il prodotto non usa nessun database, invece usa dei file csv per salvare le registrazioni. Il isto salva i dati dei utenti in 2 file separati, un file è quello globale dove vendono salvati tutte le registrazioni fatte, l’altro è il file giornaliero dove ci sono solo le iscrizioni dello stesso giorno.

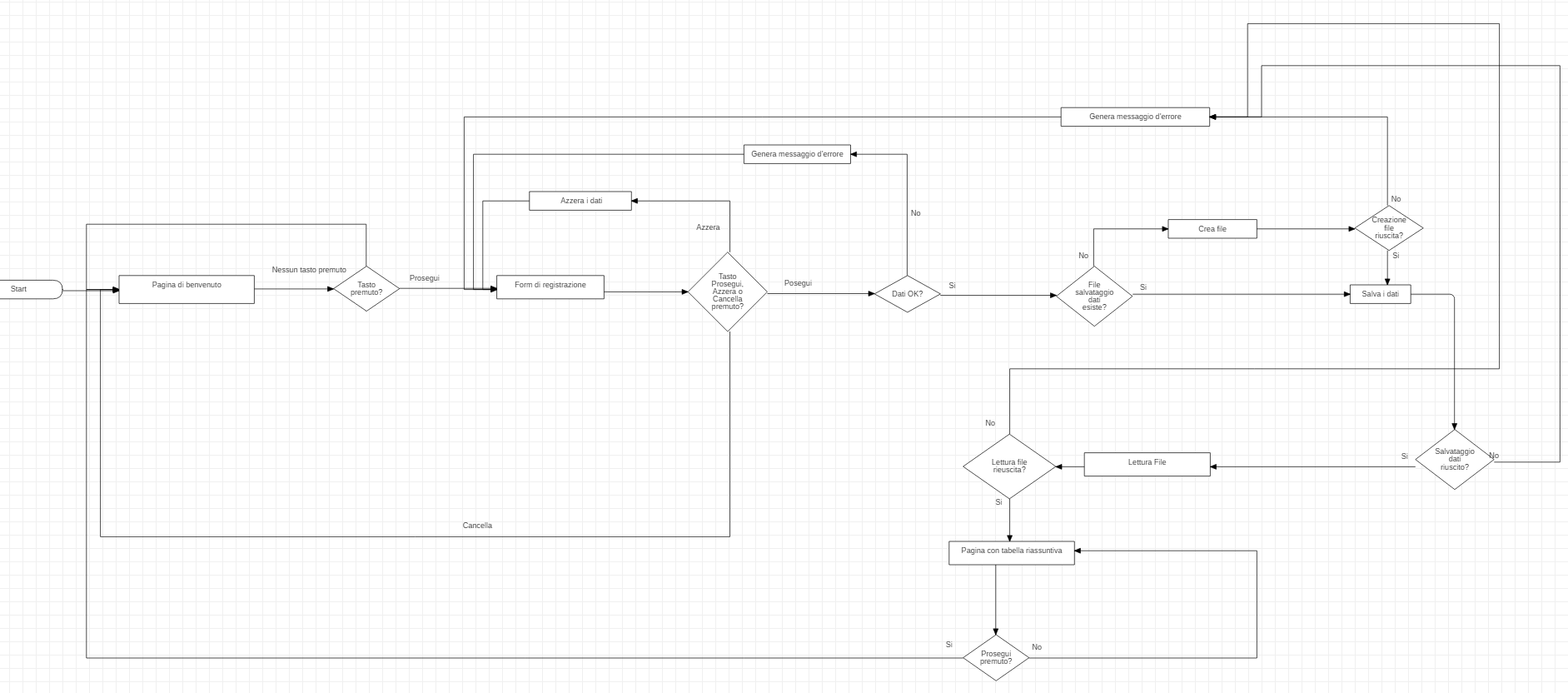
## Design delle interfacce







## Design procedurale



# Implementazione

Il sito è stato sviluppato per Google Chrome e Mozilla Firefox

Per eventuali dettagli si possono inserire riferimenti ai diari.

Questo input si trova alla fine della pagina del form, quando viene cliccato quando viene premuto richiama il metodo next() con un parametro messo su true.

*<*input type="button" class="nextPage" value="Prosegui" onclick="next(true);"*>*

La funzione next() chiama la funzione checkValues() che controlla la validità dei dati, se essi sono validi procede a chiamare la funzione makeTable con lo stesso parametro passato a next(), se i dati non dovessero essere validi la funzione checkValues segnala i campi sbagliati e il programma non procede.

function next(flag) {

if (checkValues()) {

makeTable(flag);

} else {

}

}

La funzione, makeTable() crea la tabella riassuntiva dei dati inseriti da parete del utente. In base al parametro passato può creare o distruggere la tabella dove si trovano tutti i campi.

Se il parametro è true:

* Toglie lo stile ad ogni input
* Imposta ogni input a sola lettura
* Disabilita i menu a tendina
* Mette a sola lettura i campi job e hobby
* Cambia lo stile della tabella che contiene tutti i campi
* Modifica il bottone “Prosegui” in “Modifica” che se cliccato richiama il metodo next ma con il parametro impostato su false
* Crea un nuovo bottone di tipo submit che fa la richiesta alla pagina di scrittura e lettura dei dati
* Elimina il bottone “Azzera”

Se il parametro passato è false (nel caso del bottone “Modifica”):

* Aggiunge lo stile normale agli input
* Abilita la modifica dei campi
* Modifica il bottone “Modifica” in prosegui che quando viene cliccato chiama il metodo next() con il parametro true
* Ricrea il Bottone azzera con tutte le sue proprietà

Per le notifiche (presenti in basso nel centro del form) uso NotifyJS, una libreria di javascript , cha a sua volta usa jQuery

function makeTable(flag) {

var inputs = document.querySelectorAll('input'); //contiene tutti i campi normali

var selects = document.querySelectorAll('select'); //contiene tutti i menu a tendina

var textareas = document.querySelectorAll('textarea'); //contiene I campi hobby e job

var tables = document.querySelectorAll('childTable'); // contiene le tabelle

// contiene gli elementi delle tabelle

var tableElements = document.querySelectorAll('tr, td');

//il bottone che serve per proseguire alla pagina che salva e legge i dati

var proseguiButton = document.getElementsByClassName("nextPage")[0];

var azzeraButton = document.getElementsByClassName("azzeraButton")[0];

// il bottone che azzera tutti i dati

if (flag) {

inputs.forEach(element => {

if (element.type != "button") {

element.readOnly = true;

element.className = "";

}

});

selects.forEach(element => {

element.disabled = true;

});

textareas.forEach(element => {

element.readOnly = true;

element.className = "long";

});

tables.forEach(element => {

if (element.className != "parentTable") {

element.style = "border-collapse: collapse;border: solid black 1px";

}

});

tableElements.forEach(element => {

if (element.parentElement.parentElement.className != "parentTable" || element.parentElement.parentElement.parentElement.className != "parentTable") {

element.style = "border: solid black 1px";

}

});

var newbutton = document.createElement("INPUT");

newbutton.type = "submit";

newbutton.value = "Prosegui";

proseguiButton.parentElement.appendChild(newbutton);

proseguiButton.value = "Modifica";

proseguiButton.setAttribute("onClick", "next(false);");

azzeraButton.parentElement.removeChild(azzeraButton);

} else {

inputs.forEach(element => {

element.readOnly = false;

element.className = "normalInput";

});

textareas.forEach(element => {

element.readOnly = false;

element.className = "long normalTextarea";

});

tables.forEach(element => {

element.removeAttribute("style");

});

selects.forEach(element => {

element.disabled = false;

});

tableElements.forEach(element => {

element.removeAttribute("style");

});

var proseguiParent = proseguiButton.parentElement;

if (proseguiParent.children.length == 2) {

proseguiParent.removeChild(

proseguiParent.children[proseguiParent.children.length - 1]

);

}

proseguiButton.value = "Prosegui";

proseguiButton.className = "nextPage"

proseguiButton.setAttribute("onClick", "next(true)");

var newbutton = document.createElement("INPUT");

newbutton.type = "button";

newbutton.value = "Azzera";

newbutton.className = "azzeraButton";

newbutton.setAttribute("onClick", "clearInputs()");

proseguiButton.parentElement.appendChild(newbutton);

}

Il seguente pezzo di codice serve a scrivere i dati nei due file separati.

Prende il percorso dei file tramite la variabile $GLOBALS dove sono impostati il percorso e i nomi dei file dove deve scrivere i dati. Se il percorso non esiste lo crea. Se i file non esistono crea anche loro.

Se l’utente dove viene eseguito il sito non ha permessi su file, il sito ritorna al form con un messaggio d’errore. Alla fine, richiamo il metodo clearstatcache() che cancella i file scritti la cache.

function writeData()

{

$path = $GLOBALS['path'];

if (!file\_exists($path)) {

mkdir($path, 0777, true);

}

$globalFilename = $GLOBALS['globalFilename'];

$dailyFilename = $GLOBALS['dailyFilename'];

$globalHandle = (file\_exists($globalFilename)) ? fopen($globalFilename, "a+") : fopen($globalFilename, "w+");

$dailyHandle = (file\_exists($dailyFilename)) ? fopen($dailyFilename, "a+") : fopen($dailyFilename, "w+");

$result = getFormattedData();

$colNames = getColNames() . "\n";

if (is\_writable($globalFilename) && is\_writable($dailyFilename)) {

if (filesize($globalFilename) == 0) {

fwrite($globalHandle, $colNames);

}

if (filesize($dailyFilename) == 0) {

fwrite($dailyHandle, $colNames);

}

} else {

goBack();

}

fwrite($globalHandle, ($result . "\n"));

fwrite($dailyHandle, ($result . "\n"));

fclose($globalHandle);

fclose($dailyHandle);

clearstatcache(false, $globalFilename);

clearstatcache(false, $dailyFilename);

}

# 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-003 | **Nome:** | Salvataggio dei dati inseriti dall’utente |
| **Descrizione:** | Salvare I dati convalidati in un file giornaliero e poi risalvare quelli dati in un file globale. Entrambi i file sono di tipo csv | | |
| **Prerequisiti:** | La comunicazione deve rimanere attiva tra server e client.  Assicurare che l’utente abbia i permessi per creare/scrivere su dei file.  I dati inseriti devono essere validi. | | |
| **Procedura:** | Aprire la pagina web, proseguire al form e inserire dei dati validi.  Schiacciare il tasto registra.  Il programma controlla l’esistenza dei file, se non esistono gli crea, se si il programma prosegue  Scrive i dati nel file giornaliero e quello globale. | | |
| **Risultati attesi:** | La pagina salva I dati, convalidati, in 2 file diversi.  L’utente viene portato alla pagina di riassunto dove i dati vengono letti dal file giornaliero. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-002  REQ-04 | **Nome:** | Tabella riassuntiva con i dati dell’utente |
| **Descrizione:** | Riassumere i dati inseriti dall’utente nel form in una tabella riassuntiva prima di mandarli alla pagina che salva i dati | | |
| **Prerequisiti:** | I dati inseriti devono essere validi. | | |
| **Procedura:** | Aprire la pagina web, proseguire al form e inserire dei dati validi.  Schiacciare il tasto registra. | | |
| **Risultati attesi:** | Il sito crea una tabella riassuntiva formattata in modo corretto con i tasti modifica e prosegui. | | |

## Risultati test

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Risultato**: | TC-002  Risultato-01 | **Nome:** | Tabella riassuntiva con i dati dell’utente |
| **Descrizione:** | Riassumere i dati inseriti dall’utente nel form in una tabella riassuntiva prima di mandarli alla pagina che salva i dati | | |
| **Risultato:** | Test passato | | |
| **Procedura:** | scrivere i dati giusti | | |
| **Risultati attesi:** | Il sito crea una tabella riassuntiva formattata in modo corretto con i tasti modifica e prosegui. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Risultato**: | TC-001  Risultato-02 | **Nome:** | Salvataggio dei dati inseriti dall’utente |
| **Descrizione:** | Controllare la scrittura dei dati dell’utente nei file in modo corretto con le separazioni fatte con il segno “;” | | |
| **Risultato:** | Test passato | | |
| **Procedura:** | Inserire i dati validi nel form così passano la validazione. | | |
| **Risultati attesi:** | I dati dell’utente sotto le colonne giuste nei file CSV e come separatore il segno “;” | | |

## Mancanze/limitazioni conosciute

I campi che contengono dei dati non validi non hanno un messaggio d’errore che spiega quale è il problema

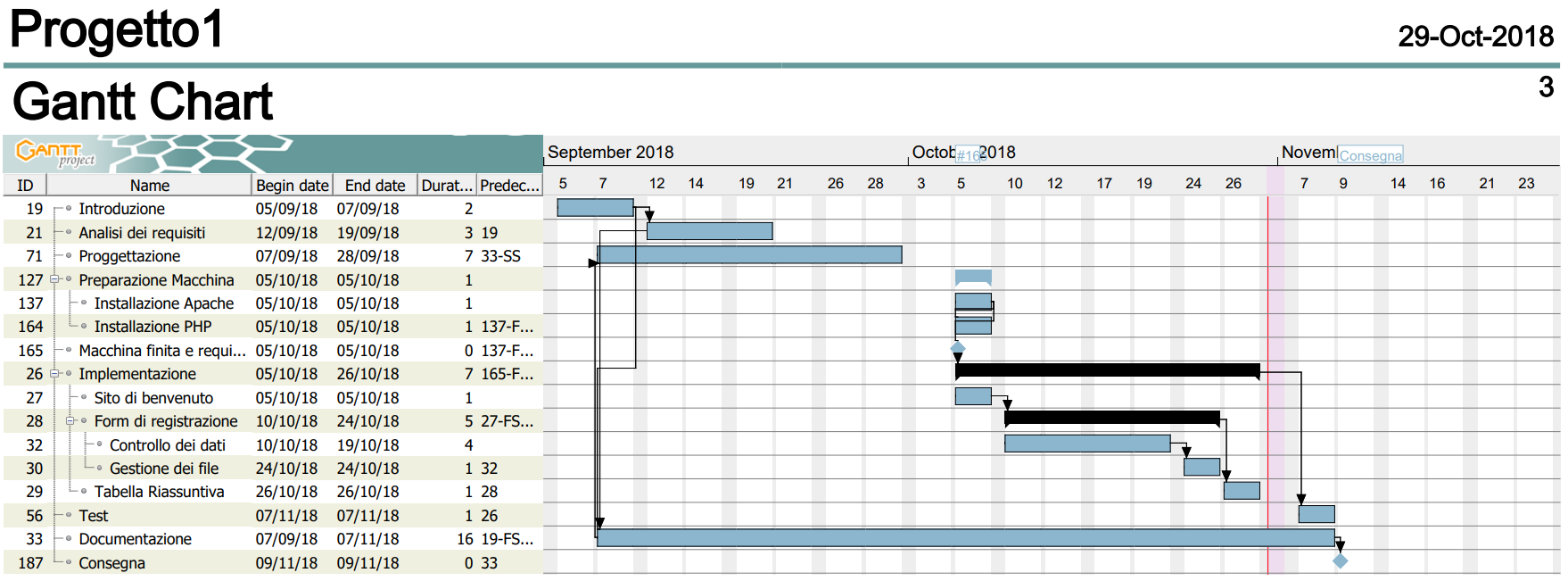
Il bottone azzera non reimposta il valore di default dei menu a tendina

I campi non sono responsive.

Il sito è usabile su Internet Explorer 11 con la mancanza del date picker per la data di nascita, perciò bisogna scriverla manualmente con il formato “YYYY-mm-dd”.

Il sito non fa i controlli se nei dati c’è presente il carattere ;

# Consuntivo



# Conclusioni

Questa soluzione essendo come esercizio fatto in classe non avrà nessun impatto.

## Sviluppi futuri

Nel futuro si potrebbero fare più controlli sulla validità dei dati inseriti dall’utente: la validità della email tramite un email di conferma, la validità del numero di telefono tramite un sms di conferma, la validità del NAP rispetto alla città e il paese. Migliorare la grafica con una libreria apposta come bootstrap, materialise o altri e fare il sito responsive per sopportare più tipi di dispositivi

## Considerazioni personali

Con questo progetto ho imparato come

# Bibliografia

## Bibliografia per articoli di riviste:

Non ho usato le riviste per questo progetto

## Bibliografia per libri

Non ho usato nessun libro per questo progetto

## Sitografia

|  |  |  |
| --- | --- | --- |
| URL del sito | Titolo | Data |
| http://jsfiddle.net/bMf9M/2/ | Stile CSS form | 05.10.2018 |
| https://www.gradient-animator.com/ | CSS Gradient Animator | 05.10.2018 |
| https://gist.github.com/andyj/7108917 | HTML <select> international calling codes for each country | 12.10.2018 |
| https://stackoverflow.com/questions/46155/how-to-validate-an-email-address-in-javascript | How to validate an email address in JavaScript? | 12.10.2018 |
| https://stackoverflow.com/questions/368813/html-form-readonly-select-tag-input | HTML form readonly SELECT tag/input | 24.10.2018 |
| https://www.w3schools.com/jsref/met\_his\_back.asp | History back() Method | 26.10.2018 |
| http://php.net/manual/en/function.clearstatcache.php | PHP: clearstatcache - Manual | 28.10.2018 |
| https://notifyjs.jpillora.com/ | Notify.js Documenatation | 01.11.2018 |

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o Qdc
* Prodotto