Gestione Vacanze

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 7

4 Test 8

4.1 Protocollo di test 8

4.2 Risultati test 8

4.3 Mancanze/limitazioni conosciute 8

5 Consuntivo 8

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 9

# Introduzione

## Informazioni sul progetto

**Titolo**: Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

**Allievi**: Mattia Toscanelli, impiegato nello svolgimento del progetto.

**Classe**: I4AC

**Docenti**: Massimo Sartori

**Sezione scuola**: Scuola Arti e Mestieri Trevano, Informatica

**Data inizio**: 03.09.2019

**Fine**: 20.12.2019

## Abstract

E’ una breve e accurata rappresentazione dei contenuti di un documento, senza notazioni critiche o valutazioni. Lo scopo di un abstract efficace dovrebbe essere quello di far conoscere all’utente il contenuto di base di un documento e metterlo nella condizione di decidere se risponde ai suoi interessi e se è opportuno il ricorso al documento originale.

Può contenere alcuni o tutti gli elementi seguenti:

* **Background/Situazione iniziale**
* **Descrizione del problema e motivazione**: Che problema ho cercato di risolvere? Questa sezione dovrebbe includere l'importanza del vostro lavoro, la difficoltà dell'area e l'effetto che potrebbe avere se portato a termine con successo.
* **Approccio/Metodi**: Come ho ottenuto dei progressi? Come ho risolto il problema (tecniche…)? Quale è stata l’entità del mio lavoro? Che fattori importanti controllo, ignoro o misuro?
* **Risultati**: Quale è la risposta? Quali sono i risultati? Quanto è più veloce, più sicuro, più economico o in qualche altro aspetto migliore di altri prodotti/soluzioni?

Esempio di abstract:

*As the size and complexity of today’s most modern computer chips increase, new techniques must be developed to effectively design and create Very Large Scale Integration chips quickly. For this project, a new type of hardware compiler is created. This hardware compiler will read a C++ program, and physically design a suitable microprocessor intended for running that specific program. With this new and powerful compiler, it is possible to design anything from a small adder, to a microprocessor with millions of transistors. Designing new computer chips, such as the Pentium 4, can require dozens of engineers and months of time. With the help of this compiler, a single person could design such a large-scale microprocessor in just weeks.*

## Scopo

Lo scopo di questo progetto è quello di realizzare un sito web che permetta ai docenti registrati di inserire delle lezioni al di fuori dell’orario scolastico, cioè durante le vacanze. In pratica esistono delle scuole particolari che procedono per tutto l’anno e che dunque non si fermano durante le vacanze scolastiche. Le vacanze possono essere i sabati, i ponti o i giorni di festa. Ogni docente avrà la capacità di riservare le proprie ore su un calendario ben prestabilito, assegnato ogni inizio di anno scolastico, fino al raggiungimento del proprio monte-ore. C’è anche una pagina di visualizzazione il quale permetterà di vedere il calendario con le varie ore riservate dai docenti senza la capacità di poterla modificare, ma se si vuole si può stampare.

# Analisi

## Analisi del dominio

Bisogna trovare un metodo che permetta ai docenti di prenotare le ore di lezione durante le vacanze scolastiche. L’obiettivo di questo progetto è dunque quello di creare un applicativo web che permetta di fare ciò senza l’impiego di una persona esterna che gestisca il tutto oppure di un colloquio tra colleghi. Infatti bisognerà che ogni docente deve avere la capacità di entrare nel sito web e senza nessuna difficoltà potersi prenotare le proprie ore. Queste ore verranno prenotate in base ad un calendario condiviso sul quale verranno mostrate le ore già occupate da altri docenti e le altre disponibili. Ci sarà anche un gestore che avrà il potere di gestire quali giorni di vacanza si potrà lavorare e per quante ore. Ogni docente che vuole lavorare durante le vacanze avrà un numero di ore che dovrà eseguire durante l’arco di tutto l’anno e il gestore dovrà avere la capacità di visualizzare un resoconto (cioè quante ore di lavoro mancano per ognuno) in qualsiasi momento. Ogni docente che vorrà usufruire di questo servizio dovrà registrarsi al sito e il gestore dovrà avere la funzione che gli permetterà di accettare o rifiutare la richiesta di registrazione. L’interfaccia della applicazione deve essere molto semplice e intuitiva, in poche parole deve essere comprensibile anche al meno esperto di informatica. Al momento il committente non dispone di un mezzo su cui gestire questo problema e sul mercato non si trova niente che possa risolvere il problema.

## Analisi e specifica dei requisiti

|  |  |
| --- | --- |
| **ID: REQ-001** | |
| **Nome** | Sito web per la gestione delle vacanze dei docenti |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | - |
| **Sotto requisiti** | |
| **001** | Deve essere facilmente accessibile da tutti. |
| **002** | Ogni funzione del sito non deve presentare imprevisti. |
| **003** | Il sito deve essere responsive, cioè deve essere adattato a qualsiasi tipo di schermo. |

|  |  |
| --- | --- |
| **ID: REQ-002** | |
| **Nome** | Pagina di registrazione |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | - |
| **Sotto requisiti** | |
| **001** | Tutti possono accedere alla pagina di registrazione. |
| **002** | Ogni registrazione dovrà poi essere confermata da un Gestore. |
| **003** | I dati da inserire per la registrazione sono i seguenti: Nome, Cognome, Indirizzo e-mail, numero di telefono, password. |
| **004** | Ogni utente registrato verrà assegnato a un tipologia di utente: Docente, il quale potrà interagire con il calendario oppure Visualizzatore, il quale potrà appunto solamente visualizzare il calendario. |

|  |  |
| --- | --- |
| **ID: REQ-003** | |
| **Nome** | Gestione degli utenti |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | - |
| **Sotto requisiti** | |
| **001** | Ci sono tre tipologie di utenti: Gestore, Docente e Visualizzatore |
| **002** | Per ogni utente di registrano i seguenti dati: Nome, Cognome, Indirizzo e-mail, numero di telefono, password. |
| **003** | La password deve avere almento 8 caratteri con almeno 1 numero/carattere speciale. |
| **004** | Il Gestore è l’utente con più poteri. Esso decide il calendario scolastico, decide chi può registrarsi al sito, decide quante ore lavorative a ciascun docente e vedere una situazione globale di ciascun docente. |
| **005** | Il Docente deve poter interagire con il calendaro creato dal Gestore, può inserire le proprie ore di lezione nel calendario e stampare il calendario con acnhe gli altri docenti. |

|  |  |
| --- | --- |
| **ID: REQ-004** | |
| **Nome** | Pagina principale con calendario scolastico |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | - |
| **Sotto requisiti** | |
| **001** | Solo il Gestore e i Docenti possono modificare il calendario scolastico, invece i Visualizzatori potrenno solamente stamparlo. |
| **002** | Il calendario scolatico deve essere disponibile sia in formato testuale si in formato calendario. |
| **003** | Possono esserci solamente due lezioni al giorno. |
| **004** | Ogni lezione ha la durata minima di 1 ora con un massimo di 3 ore. Questa deve essere definita dalle 08:00 fino alle 16:00. |
| **005** | Non ci possono essere due lezioni nello stesso momento. |
| **006** | Ogni lezione aggiunta si può spostare o elliminare. |

|  |  |
| --- | --- |
| **ID: REQ-005** | |
| **Nome** | Pagina di stampa |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | - |
| **Sotto requisiti** | |
| **001** | È possibile stampare il calendario scolastico avendo una previsualizzazione di stampa. |
| **002** | I docenti avranno la possibilità di stampare solamente le proprie ore. |

|  |  |
| --- | --- |
| **ID: REQ-006** | |
| **Nome** | Pagina d’amministratore |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | - |
| **Sotto requisiti** | |
| **001** | Questa pagina è accessibilie solamente al Gestore. |
| **002** | C’è una tabella dove vengono mostarti tutti gli utenti registarti al sito. |
| **003** | Ci deve essere la possibilità di aggiungere o elliminare gli utenti. |
| **004** | Ci deve essere la possibilità di modificare il calendario. |
| **005** | Ci deve visualizzare le ore di lavoro rimanenti dei docenti. |

## Use case

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

## Pianificazione

Prima di stabilire una pianificazione bisogna avere almeno una vaga idea del modello di sviluppo che si intende adottare. In questa sezione bisognerà inserire il modello concettuale di sviluppo che si seguirà durante il progetto. Gli elementi di riferimento per una buona pianificazione derivano da una scomposizione top-down della problematica del progetto.

La pianificazione può essere rappresentata mediante un diagramma di Gantt:

|  |
| --- |
| gantt  Figura 2: Esempio di diagramma di Gantt. |

Se si usano altri metodi di pianificazione (es scrum), dovranno apparire in questo capitolo.

## Analisi dei mezzi

Elencare e descrivere i mezzi disponibili per la realizzazione del progetto. Ricordarsi di sempre descrivere nel dettaglio le versioni e il modello di riferimento.

### Software

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

### Hardware

Su quale piattaforma dovrà essere eseguito il prodotto? Che hardware particolare è coinvolto nel progetto? Che particolarità e limitazioni presenta? Che hw sarà disponibile durante lo sviluppo?

# Progettazione

Questo capitolo descrive esaustivamente come deve essere realizzato il prodotto fin nei suoi dettagli. Una buona progettazione permette all’esecutore di evitare fraintendimenti e imprecisioni nell’implementazione del prodotto.

## Design dell’architettura del sistema

Descrive:

* La struttura del programma/sistema lo schema di rete...
* Gli oggetti/moduli/componenti che lo compongono.
* I flussi di informazione in ingresso ed in uscita e le relative elaborazioni. Può utilizzare *diagrammi di flusso dei dati* (DFD).
* Eventuale sitemap

## Design dei dati e database

Descrizione delle strutture di dati utilizzate dal programma in base agli attributi e le relazioni degli oggetti in uso.

Schema E-R, schema logico e descrizione.

Se il diagramma E-R viene modificato, sulla doc dovrà apparire l’ultima versione, mentre le vecchie saranno sui diari.

## Design delle interfacce

Descrizione delle interfacce interne ed esterne del sistema e dell’interfaccia utente. La progettazione delle interfacce è basata sulle informazioni ricavate durante la fase di analisi e realizzata tramite mockups.

## Design procedurale

Descrive i concetti dettagliati dell’architettura/sviluppo utilizzando ad esempio:

* Diagrammi di flusso e Nassi.
* Tabelle.
* Classi e metodi.
* Tabelle di routing
* Diritti di accesso a condivisioni …

Questi documenti permetteranno di rappresentare i dettagli procedurali per la realizzazione del prodotto.

# Implementazione

In questo capitolo dovrà essere mostrato come è stato realizzato il lavoro. Questa parte può differenziarsi dalla progettazione in quanto il risultato ottenuto non per forza può essere come era stato progettato.

Sulla base di queste informazioni il lavoro svolto dovrà essere riproducibile.

In questa parte è richiesto l’inserimento di codice sorgente/print screen di maschere solamente per quei passaggi particolarmente significativi e/o critici.

Inoltre dovranno essere descritte eventuali varianti di soluzione o scelte di prodotti con motivazione delle scelte.

Non deve apparire nessuna forma di guida d’uso di librerie o di componenti utilizzati. Eventualmente questa va allegata.

Per eventuali dettagli si possono inserire riferimenti ai diari.

# 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

## Bibliografia per articoli di riviste:

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo dell’articolo (tra virgolette),
3. Titolo della rivista (in italico),
4. Anno e numero
5. Pagina iniziale dell’articolo,

## Bibliografia per libri

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo del libro (in italico),
3. ev. Numero di edizione,
4. Nome dell’editore,
5. Anno di pubblicazione,
6. ISBN.

## Sitografia

1. URL del sito (se troppo lungo solo dominio, evt completo nel diario),
2. Eventuale titolo della pagina (in italico),
3. Data di consultazione (GG-MM-AAAA).

**Esempio:**

* http://standards.ieee.org/guides/style/section7.html, *IEEE Standards Style Manual*, 07-06-2008.

# 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
* …