Contents

[Introduzione 2](#_Toc39658778)

[Informazioni sul progetto 2](#_Toc39658779)

[Abstract 2](#_Toc39658780)

[Scopo 2](#_Toc39658781)

[Analisi 2](#_Toc39658782)

[Analisi del dominio 2](#_Toc39658783)

[Use case 3](#_Toc39658784)

[Pianificazione 4](#_Toc39658785)

[Pianificazione iniziale 4](#_Toc39658786)

[Pianificazione finale 4](#_Toc39658787)

[Analisi dei mezzi 4](#_Toc39658788)

[Hardware 4](#_Toc39658789)

[Software 5](#_Toc39658790)

[Progettazione 5](#_Toc39658791)

[Design dell’architettura del sistema 5](#_Toc39658792)

[Design dei dati e database 5](#_Toc39658793)

[Design delle interfacce 5](#_Toc39658794)

[Interfaccia grafica (con le impostazioni) 5](#_Toc39658795)

[Countdown 6](#_Toc39658796)

[Design procedurale 6](#_Toc39658797)

[Implementazione 6](#_Toc39658798)

[Test 10](#_Toc39658799)

[Protocollo di test 10](#_Toc39658800)

[Risultati test 11](#_Toc39658801)

[Mancanze/limitazioni conosciute 11](#_Toc39658802)

[Consuntivo 11](#_Toc39658803)

[Conclusioni 12](#_Toc39658804)

[Sviluppi futuri 12](#_Toc39658805)

[Considerazioni personali 12](#_Toc39658806)

[Sitografia 12](#_Toc39658807)

[Allegati 13](#_Toc39658808)

# Introduzione

## Informazioni sul progetto

* **Allievi coinvolti (Progettisti e Sviluppatori):** 
  + Matteo Arena
  + Bruno Gomes
  + Luca Mazza
  + Jonas Bertossa
* **Docenti coinvolti (Consulenti e Clienti):** Geo Petrini e Luca Muggiasca
* **Scuola:** Scuola Arti e Mestieri (SAM) Trevano, sezione informatica
* **Classe:** I3AA.
* **Materia:** Modulo 306.
* **Durata progetto:** 17.01.2020 - 08.05.2020

## Abstract

Con questo progetto si intende realizzare un programma che permetta di bloccare il PC quando l'utente non si trova davanti alla webcam.

## Scopo

Questo progetto ci offre l'opportunità di provare per la prima volta a gestire un progetto in team. I punti importanti, oltre al programma effettivo, sono la documentazione e i diari che devono essere completi ed esaustivi.

# Analisi

## Analisi del dominio

Il progetto è realizzato in maniera che possa essere utilizzato da qualsiasi utente, dal più esperto a quello che usa il computer solo per navigare online. Partendo con questo intento abbiamo deciso di rendere il tutto User-Friendly e il più comprensibile possibile in modo da non far emergere ambiguità. Cercando online non abbiamo trovato alcun programma che si occupi di questo (o almeno nessuno fatto rigorosamente bene) e, in quelli che abbiamo trovato, richiedevano tutti una conoscenza tecnica abbastanza elevata.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Nome** | **Descrizione** | **Importanza** | **Versione** |
| 0 | Riconoscere Volto | Capire quando qualcuno si trova davanti alla webcam | Alta | 1.0 |
| 1 | Registrare utente | L'utente deve poter registrare il proprio volto | Alta | 1.0 |
| 2 | Associare volto all'utente | Fare in modo che in base al volto visto dalla webcam il software capisca di che utente si tratta | Alta | 1.0 |
| 3 | Intervallo di blocco | Dopo x tempo che non è presente il volto il PC si deve bloccare | Alta | 1.0 |
| 4 | Interfaccia grafica | Tutte le impostazioni devono essere modificate da un'interfaccia grafica (electron o java) desktop | Alta | 1.0 |
| 5 | Bloccare PC | Il PC si deve bloccare quando non viene riconosciuto un volto | Alta | 1.0 |
| 6 | Scelta intervallo di blocco | L'utente può scegliere dopo quanto tempo il PC si deve bloccare | Media | 1.0 |
| 7 | Modifica impostazioni in modo sicuro | La modifica delle impostazioni richiede una password | Alta | 1.0 |
| 8 | Aggiornamento progressivo volto | I volti vengono sovrascritti ogni giorno per fare in modo che sia il più duraturo possibile | Bassa | 1.0 |
| 9 | Registrazione di più volti | Più utenti (sulla macchina) possono aggiungere il proprio volto | Media | 1.0 |
| 10 | Utente corrente | L'utente da riconoscere deve essere quello dell'account | Bassa | 1.0 |
| 11 | Multipiattaformità | Il software deve poter essere eseguito sui sistemi operativi più usati (Windows, MacOS, Linux) | Alta | 1.0 |
| 12 | Cifratura dati volto | I dati del volto sono cifrati | Bassa | 1.0 |
| 13 | Countdown | Quando non viene riconosciuto un volto parte il countdown e poi si spegne | Alta | 1.0 |
| 14 | Shortcut | Tramite una shortcut deve essere possibile attivare e disabilitare le funzionalità del software | Bassa | 1.0 |
| 15 | Utenti aggiuntivi | Dentro ad un account ci possono essere anche degli utenti con il permesso di usare il PC | Media | 1.0 |
| 16 | Mail/notifica di blocco | Quando il PC viene bloccato ci arriverà una mail/notifica | Bassa | 1.0 |
| 17 | Mail/Notifica utilizzo | Quando qualcuno usa il PC e non sei tu arriva una notifica/mail | Bassa | 1.0 |

## Use case

<img src="Img/useCase.png" alt="Immagine ritraente lo use case"/>

## Pianificazione

### Pianificazione iniziale

<img src="img/GanttIniziale1.png" alt="Gantt iniziale 1"/>

<img src="img/GanttIniziale2.png" alt="Gantt iniziale 2"/>

### Pianificazione finale

<img src="img/GanttFinale1.png" alt="Gantt Finale 1"/>

<img src="img/GanttFinale2.png" alt="Gantt Finale 2"/>

Come si può notare facilmente la durata del progetto è stata estesa, questo allungamento non è stato a causa della difficoltà sottovalutata ma perché abbiamo deciso di aggiungere degli elementi aggiuntivi. Il 5 febbraio infatti avevamo già finito la grande maggioranza dei requisiti ma per rendere il programma più completo abbiamo aggiunto ancora altri requisiti.

## Analisi dei mezzi

### Hardware

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nome** | **Proprietario** | **Sistema operativo** | **RAM** | **Storage** | **Scheda grafica** | **CPU** | **Risoluzione** |
| Acer Aspire 5 | Matteo | Windows 10 1909 | 16GB | 128GB SSD 1TB HDD | NVIDIA 940mx | i7 7500U | 1920x1080 |
| MacBook Pro (15-inch, 2017) | Luca | MacOS Catalina 10.15.4 | 16GB | 256GB SSD | Radeon Pro 555 | i7 7700HQ | 2880x1800 |
| Acer Predator Orion 3000 | Matteo | Windows 10 1903 | 16GB | 500GB SSD 1TB HDD | NVIDIA Geforce RTX 2060 super | i7 9700 | 1920x1080 (monitor esterno) |
| Asus | Bruno | Windows 10 1909 | 4GB | 128GB | - | i3 7020U | 1366x768 |

### Software

|  |  |
| --- | --- |
| Nome | Versione |
| Python | 3.6.6 |
| cmake | 3.16.3 |
| dlib | 19.7.0 |
| face-recognition | 1.3.0 |
| face-recognition-models | 0.3.0 |
| numpy | 1.18.2 |
| opencv-python | 4.1.0.25 |
| pip | 20.0.2 |
| psutil | 5.7.0 |
| netbeans | 11.1 |
| JRE | 12.0.2 |
| JDK | 12.0.2 |
| NSIS | 3.05 |

# Progettazione

## Design dell’architettura del sistema

Lo script principale (quello che è sempre eseguito) si chiama faceCheck.py e si può trovare nella directory 'C:\Program Files (x86)\FaceLock\Source\faceCheck.py' su windows, '/Applications/FaceLock/Resources/faceCheck.py' su MacOS e [**METTI DIRECTORY LINUX**] su Linux. Una volta avviato lo script viene eseguito in background sulla macchina ospite e ogni secondo accende la webcam per controllare se il proprietario è davanti al computer. Dentro allo script sono presenti molte funzioni per fare in modo che il main possa richiamarle più volte avendo in questa maniera un codice estremamente pulito e molto facile da leggere.

## Design dei dati e database

I dati dei volti sono codificati tramite degli encodings di face-recognition (libreria di Python), per leggere i dati del volto o crearne uno nuovo bisogna passare tramite la funzione dedicata della libreria.

## Design delle interfacce

### Interfaccia grafica (con le impostazioni)

<img src="Img/FaceLockUI.png" width="400" height="500" style="float: left;"/>

L'interfaccia grafica (in questa schermata ideata per MacOS ma praticamente invariata per Windows e Linux) è pensata per essere molto minimale ma che comprende tutte le informazioni indispensabili. Come si può notare infatti le prime informazioni che si notano sono i nomi delle facce da riconoscere (John Zillo, Pier Telo e George Bo) con a fianco una x per eliminarli. Subito sotto troviamo la categoria General contenente le preferenze dell'utente, più specificatamente:

* Scelta dei secondi dopo il quale bloccare il computer
* Notifica quando il computer viene spento (sul telefono)
* Shortcut per abilitare o disattivare lo script che controlla se qualcuno è davanti al computer
* Dark mode del programma

Infine è presente un checkbox per scegliere se notificare (sempre sul telefono) se qualcuno che non è il proprietario sta utilizzando il computer.

### Countdown

<img src="Img/Countdown.png" width="350" height="250" style="float: left;" hspace="10"/>

In questa immagine è rappresentata come il countdown una volta che non vedrà uno degli utenti definiti in precedenza, ovviamente dovrà essere completamente funzionante e dovrà essere visualizzato unicamente quando lo script è sicuro che davanti alla webcam non ci sia nessuno.

## Design procedurale

All'interno del progetto dovranno essere presenti due file con gli script (quello in Python e quello in Java), la differenza tra i due sarà che quello in Python si occuperà di riconoscere i volti, bloccare lo schermo e gestire il countdown mentre il programma in java si occuperà di gestire le impostazioni. Le impostazioni verranno poi salvate in un file dove il file Java scriverà e quello Python leggerà.

Come primissima cosa nel file Python bisognerà leggere le immagini contenenti i volti degli utenti, poi lo script potrà partire e, se non riconoscerà alcun volto per un tot di tempo definito nelle impostazioni, esso bloccherà lo schermo.

# Implementazione

La struttura del nostro progetto è divisa in 3 file principali, più precisamente:

**“Preferences.jar”, “image.py” e “faceCheck.py”**

Image.py

Questa classe (scritta in **Python**) si occupa unicamente di scattare una foto all'utente (con un'interfaccia grafica dedicata). Questo codice è duplicato anche in **faceCheck.py**, abbiamo optato per questa opzione per non fare troppe richieste alla classe principale e poter gestire al meglio lo scatto della fotocamera.

user = ""

files = []

isempty = True

if len(sys.argv) == 2:

user = sys.argv[1]

print(user)

else:

user = "DEFAULT"

if (not os.path.isdir("Dataset/" + user + "/")):

os.mkdir("Dataset/" + user + "/")

for file in os.listdir('Dataset/' + user + "/"):

files.insert(0, file.strip(".jpg"))

isempty = False

if isempty:

files.insert(0, "0")

else:

files[0] = int(files[0]) + 1

camera = cv2.VideoCapture(0)

while True:

return\_value,image = camera.read()

cv2.imshow('image',image)

if cv2.waitKey(1)& 0xFF == ord('s'):

cv2.imwrite('' + str(files[0]) + '.jpg',image)

break

camera.release()

cv2.destroyAllWindows()

if platform == 'win32':

shutil.move('.\\' + (str(files[0]) + '.jpg'), 'Dataset\\' + user + '\\' + str(files[0]) + '.jpg')

else:

shutil.move('./' + (str(files[0]) + '.jpg'), 'Dataset/' + user + '/' + str(files[0]) + '.jpg')

FaceCheck.py

Questa classe è sicuramente la più complicata e intrinseca del progetto, tutte le operazioni complicate all'interno di essa sono state gestite in funzione, in modo che il codice eseguito potesse essere il più pulito possibile e quindi anche comprensibile e facile.

#### Riconoscimento della piattaforma

A causa di qualche problema (siccome del codice funzionava unicamente su una certa piattaforma e non su un'altra) abbiamo implementato all'inizio del codice un metodo per riconoscere facilmente la piattaforma utilizzata. Abbiamo deciso di cambiare la nomenclatura normale (Darwin per MacOS, win32 per windows e altrimenti Linux) con una più semplice per non doverci ricordare ogni volta come si chiama.

system = ""

if platform == "darwin":

system = "macOS"

elif platform == "win32":

system = "Windows"

else:

system = "Linux"

#### Riconoscimento degli utenti dalle cartelle

Con questo metodo riusciamo a capire quali utenti sono registrati (e quindi compariranno nell'interfaccia grafica), queste cartelle infatti sono all'interno della cartella Dataset (in Source/FaceLock) e le cartelle a loro volta contengono tutte le immagini che gli utenti hanno scattato.

def getUsers(self):

path = './Dataset/'

files = os.listdir(path)

for name in files:

full\_path = os.path.join(path, name + "/")

if os.path.exists(full\_path):

self.users.append(name)

#### Blocco dello schermo

Grazie all'utilizzo della funzione “**lockScreen**” il programma riesce con un solo comando a bloccare lo schermo, qua dentro il codice riconosce automaticamente il sistema operativo utilizzato basandosi sull'array creato all'inizio del codice. In caso che il sistema non venga riconosciuto esso non verrà bloccato e su linea di comando verrà stampato l'errore "Non posso ancora bloccare questo dispositivo".

def lockScreen(self, system):

if(self.system == "Windows"):

ctypes.windll.user32.LockWorkStation()

elif (system == "macOS" or system == "Linux"):

loginPF = CDLL('/System/Library/PrivateFrameworks/login.framework/Versions/Current/login')

result = loginPF.SACLockScreenImmediate()

else:

print("Non posso ancora bloccare questo dispositivo")

#### Verificare l'utilizzo simultaneo della fotocamera

Su windows abbiamo avuto un problema riguardante l'utilizzo della fotocamera su più processi, per evitare questo problema abbiamo scritto questo codice che si occupa di controllare se un processo sta venendo eseguito. Ovviamente il processo non dirà che è già eseguito dalla classe FaceCheck perché la fotocamera viene attivata e disattivata e non lasciata continuamente attiva.

def checkIfProcessRunning(self, processName):

for proc in psutil.process\_iter():

try:

if processName.lower() in proc.name().lower():

return True

except (psutil.NoSuchProcess, psutil.AccessDenied, psutil.ZombieProcess):

pass

return False

Preferences.jar

Preferences.jar fa parte di un progetto NetBeans (ovviamente in **Java**) che permette di cambiare le impostazioni legate allo script FaceCheck. Tutte queste impostazioni vengono successivamente scritte nel file settings.csv (all'interno della cartella Settings) e verrà poi interpretato da FaceCheck per capire come dovrà comportarsi.

# 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 | TC-001 |

|---------------|--------------------------------------|

|\*\*Nome\*\* |Import a card, but not shown with the GUI |

|\*\*Riferimento\*\*|REQ-012 |

|\*\*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) |

|\*\*Procedura\*\* | - Go to “Cards manager” menu, in main page click “Import Profiles” link, Select the “1.2.001.xml” file, Import the Profile - 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 |

|\*\*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 Gantt

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

# 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

- …