

Implementazione di un tool per il supporto alla valutazione automatica di progetti su GitHub

Corso di Laurea Triennale in Informatica
Tesi di Laurea

Matteo Azzarelli

ANNO ACCADEMICO 2017/2018



**Università
degli Studi
di Perugia**

- 1 Introduzione
- 2 GitHub
- 3 Lavori Correlati
- 4 Studio ed Implementazione
- 5 Conclusioni

Perché abbiamo sviluppato questa applicazione?

- Dissuadere gli studenti dal plagio.
- Scaricare i progetti assegnati su GitHub Classroom.
- Gestione di centinaia di progetti per svariati appelli all'anno.

Perché abbiamo sviluppato questa applicazione?

- 1 Dissuadere gli studenti dal plagio.
- 2 Scaricare i progetti assegnati su GitHub Classroom.
- 3 Gestione di centinaia di progetti per svariati appelli all'anno.

Perché abbiamo sviluppato questa applicazione?

- 1 Dissuadere gli studenti dal plagio.
- 2 Scaricare i progetti assegnati su GitHub Classroom.
- 3 Gestione di centinaia di progetti per svariati appelli all'anno.

Perché abbiamo sviluppato questa applicazione?

- 1 Dissuadere gli studenti dal plagio.
- 2 Scaricare i progetti assegnati su GitHub Classroom.
- 3 Gestione di centinaia di progetti per svariati appelli all'anno.

Cosa è **GitHub**?

GitHub è un servizio di hosting per il controllo delle versioni.

Il controllo delle versioni consente di tenere traccia delle modifiche apportate al codice sorgente del software.

GitHub Classroom

Consente agli insegnanti di assegnare compiti agli studenti, facendoli approcciare a GitHub.

Cosa è **GitHub**?

GitHub è un servizio di hosting per il controllo delle versioni.

Il controllo delle versioni consente di tenere traccia delle modifiche apportate al codice sorgente del software.

GitHub Classroom

Consente agli insegnanti di assegnare compiti agli studenti, facendoli approcciare a GitHub.

GitHub è basato sul software **Git**, il quale permette di:

- Tenere traccia dei cambiamenti dei file.
- Coordinare il lavoro di più persone sullo stesso insieme di file.
- Mantenere l'integrità dei files.
- Gestire flussi di lavoro non lineari.

GitHub è basato sul software **Git**, il quale permette di:

- Tenere traccia dei cambiamenti dei file.
- Coordinare il lavoro di più persone sullo stesso insieme di file.
- Mantenere l'integrità dei files.
- Gestire flussi di lavoro non lineari.

GitHub è basato sul software **Git**, il quale permette di:

- Tenere traccia dei cambiamenti dei file.
- Coordinare il lavoro di più persone sullo stesso insieme di file.
- Mantenere l'integrità dei files.
- Gestire flussi di lavoro non lineari.

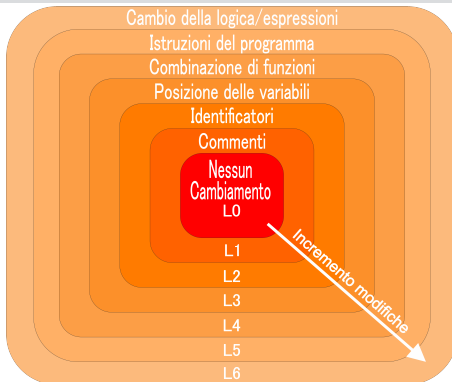
GitHub è basato sul software **Git**, il quale permette di:

- Tenere traccia dei cambiamenti dei file.
- Coordinare il lavoro di più persone sullo stesso insieme di file.
- Mantenere l'integrità dei files.
- Gestire flussi di lavoro non lineari.

Definizione

Un programma che è stato prodotto da un altro e riportato con un numero esiguo di trasformazioni di routine.

1976. Alan Parker e James O. Hamblen.



Caratteristiche dell'applicazione:

■ Sviluppata in **Java**:

- È cross-platform.
- Documentazione completa.
- Esistenza di librerie per il nostro scopo.

■ Utilizzo del pattern Model View Controller.

■ Download delle repositories di GitHub.

■ Analisi dei progetti degli studenti tramite il servizio MOSS.

Caratteristiche dell'applicazione:

■ Sviluppata in **Java**:

- È cross-platform.
- Documentazione completa.
- Esistenza di librerie per il nostro scopo.

■ Utilizzo del pattern Model View Controller.

■ Download delle repositories di GitHub.

■ Analisi dei progetti degli studenti tramite il servizio MOSS.

Caratteristiche dell'applicazione:

■ Sviluppata in **Java**:

- È cross-platform.
- Documentazione completa.
- Esistenza di librerie per il nostro scopo.

■ Utilizzo del pattern Model View Controller.

■ Download delle repositories di GitHub.

■ Analisi dei progetti degli studenti tramite il servizio MOSS.

Caratteristiche dell'applicazione:

- Sviluppata in **Java**:
 - È cross-platform.
 - Documentazione completa.
 - Esistenza di librerie per il nostro scopo.
- Utilizzo del pattern Model View Controller.
- Download delle repositories di GitHub.
- Analisi dei progetti degli studenti tramite il servizio MOSS.

Caratteristiche dell'applicazione:

- Sviluppata in **Java**:
 - È cross-platform.
 - Documentazione completa.
 - Esistenza di librerie per il nostro scopo.
- Utilizzo del pattern Model View Controller.
- Download delle repositories di GitHub.
- Analisi dei progetti degli studenti tramite il servizio MOSS.

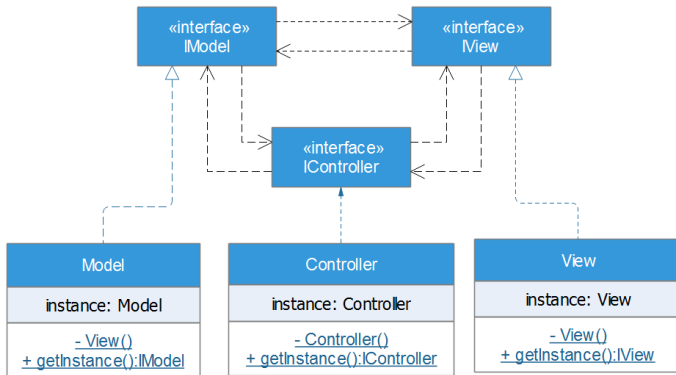
Caratteristiche dell'applicazione:

- Sviluppata in **Java**:
 - È cross-platform.
 - Documentazione completa.
 - Esistenza di librerie per il nostro scopo.
- Utilizzo del pattern Model View Controller.
- Download delle repositories di GitHub.
- Analisi dei progetti degli studenti tramite il servizio MOSS.

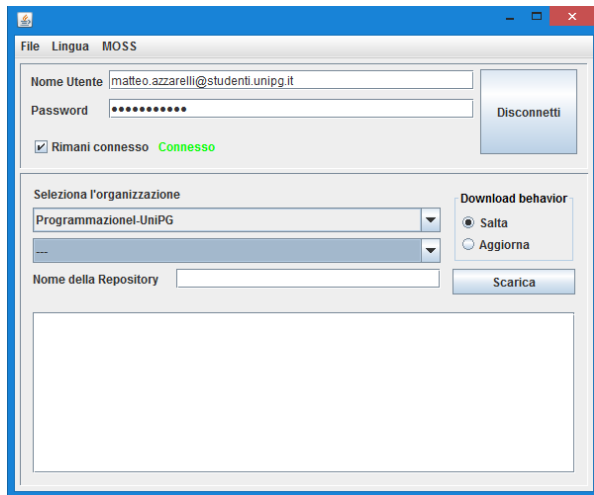
Caratteristiche dell'applicazione:

- Sviluppata in **Java**:
 - È cross-platform.
 - Documentazione completa.
 - Esistenza di librerie per il nostro scopo.
- Utilizzo del pattern Model View Controller.
- Download delle repositories di GitHub.
- Analisi dei progetti degli studenti tramite il servizio MOSS.

Diagramma UML della struttura del software:



Uno sguardo all'interfaccia:



The screenshot shows the MOSS (Modular Open Source System) interface. The window has a blue title bar with standard Windows controls. The menu bar includes 'File', 'Lingua', and 'MOSS'. The main area is divided into sections for user login, organization selection, and repository management.

User Login Section:

- Nome Utente:**
- Password:**
- ☒ Rimani connesso Connesso
- Disconnetti** button

Organization Selection Section:

- Seleziona l'organizzazione:**
- Download behavior:**
 - ☒ Salta
 - ☐ Aggiorna
- Nome della Repository:**
- Scarica** button

Repository List:

- A large empty rectangular box for displaying repository information.

MOSS (Measure Of Software Similarity) è un sistema automatico per determinare la similarità di programmi.

Input Accetta gruppi di documenti.

Output Restituisce un'insieme di pagine HTML contenenti le coppie di documenti simili.

Può analizzare molti linguaggi di programmazione, come ad esempio:

- C, C++ e C#
- Java
- Il linguaggio naturale e molti altri.

MOSS (Measure Of Software Similarity) è un sistema automatico per determinare la similarità di programmi.

Input Accetta gruppi di documenti.

Output Restituisce un'insieme di pagine HTML contenenti le coppie di documenti simili.

Può analizzare molti linguaggi di programmazione, come ad esempio:

- C, C++ e C#
- Java
- Il linguaggio naturale e molti altri.

Con poche fingerprint ottiene ottimi risultati. Ciò implica maggiore efficienza.

Algoritmo:

- 1 Costruisce una mappa delle fingerprint per tutti i documenti.
- 2 Per ogni documento viene effettuata una nuova fingerprint, ottenendo una lista di fingerprint per ciascun documento. Ora ogni documento d può contenere fingerprint di molti altri documenti d_1, d_2, \dots .
- 3 La lista delle fingerprint viene raggruppata per documento e poi vengono fatte le coppie di documenti (d, d_1) , (d, d_2) . Queste coppie vengono ordinate per numero di fingerprint uguali.

Con poche fingerprint ottiene ottimi risultati. Ciò implica maggiore efficienza.

Algoritmo:

- 1 Costruisce una mappa delle fingerprint per tutti i documenti.
- 2 Per ogni documento viene effettuata una nuova fingerprint, ottenendo una lista di fingerprint per ciascun documento. Ora ogni documento d può contenere fingerprint di molti altri documenti d_1, d_2, \dots .
- 3 La lista delle fingerprint viene raggruppata per documento e poi vengono fatte le coppie di documenti (d, d_1) , (d, d_2) . Queste coppie vengono ordinate per numero di fingerprint uguali.

Con poche fingerprint ottiene ottimi risultati. Ciò implica maggiore efficienza.

Algoritmo:

- 1** Costruisce una mappa delle fingerprint per tutti i documenti.
- 2** Per ogni documento viene effettuata una nuova fingerprint, ottenendo una lista di fingerprint per ciascun documento. Ora ogni documento d può contenere fingerprint di molti altri documenti d_1, d_2, \dots .
- 3** La lista delle fingerprint viene raggruppata per documento e poi vengono fatte le coppie di documenti (d, d_1) , (d, d_2) . Queste coppie vengono ordinate per numero di fingerprint uguali.

Generare le fingerprint:

- Divide un documento in k-grams, cioè in sottostringhe di lunghezza k

Esempio k-gram

A do run run run, a do run run

(a) un po di testo [The Crystals. Da do run run, 1963]

adorunrunrunadorunrun

(b) il testo senza alcune caratteristiche irrilevanti

adoru dorun orunr runru unrun nrunr runru
unrun nruna runad unado nador adoru dorun
orunr runru unrun

(c) la sequenza di 5-grams derivata dal testo

- Utilizza una funzione hash per ogni k-gram e selezioniamo tra essi qualche sottoinsieme, che sarà la fingerprint.

Generare le fingerprint:

- Divide un documento in k-grams, cioè in sottostringhe di lunghezza k

Esempio k-gram

A do run run run, a do run run

(a) un po di testo [The Crystals. Da do run run, 1963]

adorunrunrunadorunrun

(b) il testo senza alcune caratteristiche irrilevanti

adoru dorun orunr runru unrun nrunr runru
unrun nruna runad unado nador adoru dorun
orunr runru unrun

(c) la sequenza di 5-grams derivata dal testo

- Utilizza una funzione hash per ogni k-gram e selezioniamo tra essi qualche sottoinsieme, che sarà la fingerprint.

Generare le fingerprint:

- Divide un documento in k-grams, cioè in sottostringhe di lunghezza k

Esempio k-gram

A do run run run, a do run run

(a) un po di testo [The Crystals. Da do run run, 1963]

adorunrunrunadorunrun

(b) il testo senza alcune caratteristiche irrilevanti

adoru dorun orunr runru unrun nrunr runru
unrun nruna runad unado nador adoru dorun
orunr runru unrun

(c) la sequenza di 5-grams derivata dal testo

- Utilizza una funzione hash per ogni k-gram e selezioniamo tra essi qualche sottoinsieme, che sarà la fingerprint.

Generare le fingerprint:

- Divide un documento in k-grams, cioè in sottostringhe di lunghezza k

Esempio k-gram

A do run run run, a do run run

(a) un po di testo [The Crystals. Da do run run, 1963]

adorunrunrunadorunrun

(b) il testo senza alcune caratteristiche irrilevanti

adoru dorun orunr runru unrun nrunr runru
unrun nruna runad unado nador adoru dorun
orunr runru unrun

(c) la sequenza di 5-grams derivata dal testo

- Utilizza una funzione hash per ogni k-gram e selezioniamo tra essi qualche sottoinsieme, che sarà la fingerprint.

Generare le fingerprint:

- Divide un documento in k-grams, cioè in sottostringhe di lunghezza k

Esempio k-gram

A do run run run, a do run run

(a) un po di testo [The Crystals. Da do run run, 1963]

adorunrunrunadorunrun

(b) il testo senza alcune caratteristiche irrilevanti

adoru dorun orunr runru unrun nrunr runru
unrun nruna runad unado nador adoru dorun
orunr runru unrun

(c) la sequenza di 5-grams derivata dal testo

- Utilizza una funzione hash per ogni k-gram e selezioniamo tra essi qualche sottoinsieme, che sarà la fingerprint.

MOSS mette a disposizione un webservice accessibile tramite vari script e librerie.

Per il nostro progetto abbiamo utilizzato la libreria Java **MOJI** che consente l'accesso al servizio.

Questa libreria permette di:

- 1 Connettersi al server con il proprio numero utente.
- 2 Inviare la cartella contenente tutti i progetti degli studenti.
- 3 Inviare la cartella contenente il modello del progetto.
- 4 Ricevere il link per visualizzare la pagina HTML dei risultati.

MOSS mette a disposizione un webservice accessibile tramite vari script e librerie.

Per il nostro progetto abbiamo utilizzato la libreria Java **MOJI** che consente l'accesso al servizio.

Questa libreria permette di:

- 1 Connettersi al server con il proprio numero utente.
- 2 Inviare la cartella contenente tutti i progetti degli studenti.
- 3 Inviare la cartella contenente il modello del progetto.
- 4 Ricevere il link per visualizzare la pagina HTML dei risultati.

MOSS mette a disposizione un webservice accessibile tramite vari script e librerie.

Per il nostro progetto abbiamo utilizzato la libreria Java **MOJI** che consente l'accesso al servizio.

Questa libreria permette di:

- 1 Connettersi al server con il proprio numero utente.
- 2 Inviare la cartella contenente tutti i progetti degli studenti.
- 3 Inviare la cartella contenente il modello del progetto.
- 4 Ricevere il link per visualizzare la pagina HTML dei risultati.

MOSS mette a disposizione un webservice accessibile tramite vari script e librerie.

Per il nostro progetto abbiamo utilizzato la libreria Java **MOJI** che consente l'accesso al servizio.

Questa libreria permette di:

- 1 Connettersi al server con il proprio numero utente.
- 2 Inviare la cartella contenente tutti i progetti degli studenti.
- 3 Inviare la cartella contenente il modello del progetto.
- 4 Ricevere il link per visualizzare la pagina HTML dei risultati.

MOSS mette a disposizione un webservice accessibile tramite vari script e librerie.

Per il nostro progetto abbiamo utilizzato la libreria Java **MOJI** che consente l'accesso al servizio.

Questa libreria permette di:

- 1 Connettersi al server con il proprio numero utente.
- 2 Inviare la cartella contenente tutti i progetti degli studenti.
- 3 Inviare la cartella contenente il modello del progetto.
- 4 Ricevere il link per visualizzare la pagina HTML dei risultati.

Struttura cartella dei progetti

```
solution_directory
|- student1
    |- main.c
    |- ...
|- student2
    |- ...
|- student3
    |- ...
```

Moss Results

Wed Jul 4 13:31:37 PDT 2018

Options -l c -d -m 10

ESAME PROGRAMMAZIONE 1

[[How to Read the Results](#) | [Tips](#) | [FAQ](#) | [Contact](#) | [Submission Scripts](#) | [Credits](#)]

File 1	File 2	Lines Matched
C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████c98/ (96%)	C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████s2/ (97%)	1035
C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████ni4/ (74%)	C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████elli/ (78%)	776
C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████ce/ (96%)	C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████trdo/ (97%)	669
C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████iv/ (85%)	C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████p0/ (92%)	831
C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████etta/ (84%)	C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████etti/ (86%)	759
C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████etto/ (76%)	C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████sae/ (80%)	722
C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████c0/ (80%)	C:/ProgrammazioneI-UniPG/prova-finale-di-programmazione-secondo-appello/██████sae/ (78%)	729

C:\Programmi\UniPG\prova-finale-di-programmazione-secondo-appello\██████████\2: (57%)	C:\Programmi\UniPG\prova-finale-di-programmazione-secondo-appello\██████████\3: (65%)
144-311	231-301
182-243	168-229
323-377	317-369
390-428	379-415

C:\Programmi\UniPG\prova-finale-di-programmazione-secondo-appello\██████████\2:

```

>>> file: gamelib.c
#include "gamelib.h"

static Zona "prima_zona" = NULL;
static Zona "ultima_zona" = NULL;

static Giocatore "giocatore[numero_giocatori];

int benza = 0;
int flagChange = 0;
int difficolta = 0;
int turno;

//funzioni che gestiscono la Happa
//
void gest_map() //funzione che gestisce il menu "crea mappa"
{
    char op_map = ' ';
    int i_zona = 0;

    if (flag_ins == 0)
    {
        printf("\nSeleziona il livello di difficoltà\n1_Facile\n2_Normale\n3_Difficile\n");
        scanf("%d", &difficolta);
        system("clear");
    }

    do
    {
        printf("\n1_Inserisci Zona\n2_Elimina Ultima Zona\n3_Elimina Happa\n4_Stampa Happa\n5_Chiudi Happa\n6_Salva Happa\n7_Ricarica Happa\n8_Restaura Happa\n9_Exit\n");
        scanf("%d", &op_map);

        switch(op_map)
        {
            case '1':
                if (flag_ins == 0)
                {
                    printf("Inserisci il nome della zona: ");
                    char nome_zona[50];
                    scanf("%s", nome_zona);

                    if (strlen(nome_zona) > 0)
                    {
                        i_zona++;
                        zona[i_zona] = nome_zona;
                    }
                }
            case '2':
                if (i_zona > 0)
                {
                    printf("Seleziona la zona da eliminare: ");
                    int i_elimina;
                    for (i_elimina = 0; i_elimina < i_zona; i_elimina++)
                    {
                        printf("%d: %s\n", i_elimina, zona[i_elimina]);
                    }
                    scanf("%d", &i_elimina);

                    if (i_elimina < i_zona)
                    {
                        zona[i_elimina] = NULL;
                        i_zona--;
                    }
                }
            case '3':
                if (i_zona > 0)
                {
                    printf("Seleziona la zona da eliminare: ");
                    int i_elimina;
                    for (i_elimina = 0; i_elimina < i_zona; i_elimina++)
                    {
                        printf("%d: %s\n", i_elimina, zona[i_elimina]);
                    }
                    scanf("%d", &i_elimina);

                    if (i_elimina < i_zona)
                    {
                        zona[i_elimina] = NULL;
                        i_zona--;
                    }
                }
            case '4':
                printf("Stampa Happa\n");
                for (i = 0; i < i_zona; i++)
                {
                    printf("%s\n", zona[i]);
                }
            case '5':
                printf("Chiudi Happa\n");
                break;
            case '6':
                printf("Salva Happa\n");
                break;
            case '7':
                printf("Ricarica Happa\n");
                break;
            case '8':
                printf("Restaura Happa\n");
                break;
            case '9':
                printf("Exit\n");
                break;
        }
    } while (op_map != 5);
}

```

C:\Programmi\UniPG\prova-finale-di-programmazione-secondo-appello\██████████\3:

```

>>> file: gamelib.c
#include "gamelib.h"
#include <stdlib.h>

static Zona "prima_zona" = NULL;
static Zona "ultima_zona" = NULL;

static Giocatore "giocatore[numero_giocatori];

//funzioni che gestisce il "Gestione Happa"
void gest_map()
{
    int sceltaOpzioniMenu = 0;
    int sceltaMenuZona = 0;
    char sceltaEliminazioneHappa = ' '; //Variabile in cui andrà il carattere dell'eliminazione mappa

    do
    {
        printf("\n--Opzioni Crea Happa--\n");
        printf("\n1-Inserisci Zona\n2-Cancella Zona\n3-Stampa Happa\n4-Elimina Happa\n5-Chiudi Happa\n6-Salva Happa\n7-Ricarica Happa\n8-Restaura Happa\n9-Exit\n");
        scanf("%d", &sceltaOpzioniMenu);

        switch(sceltaOpzioniMenu)
        {
            case 1:
                if (flagHappaCrea == 0)
                {
                    do
                    {
                        printf("Inserisci almeno otto zone\n1-Cucina\n2-Soggiorno\n3-Salotto\n4-Bagno\n5-Corridoio\n6-Terrazza\n7-Garage\n8-Portico\n9-Altro\n");
                        scanf("%d", &sceltaMenuZona);

                        if (sceltaMenuZona < 1 || sceltaMenuZona > 9)
                        {
                            printf("Seleziona una zona valida\n");
                            continue;
                        }

                        if (sceltaMenuZona > 0)
                        {
                            zona[sceltaMenuZona] = NULL;
                            sceltaMenuZona--;
                        }
                    } while (sceltaMenuZona > 0);

                    printf("Inserisci il nome della zona: ");
                    char nome_zona[50];
                    scanf("%s", nome_zona);

                    if (strlen(nome_zona) > 0)
                    {
                        i_zona++;
                        zona[i_zona] = nome_zona;
                    }
                }
            case 2:
                if (i_zona > 0)
                {
                    printf("Seleziona la zona da eliminare: ");
                    int i_elimina;
                    for (i_elimina = 0; i_elimina < i_zona; i_elimina++)
                    {
                        printf("%d: %s\n", i_elimina, zona[i_elimina]);
                    }
                    scanf("%d", &i_elimina);

                    if (i_elimina < i_zona)
                    {
                        zona[i_elimina] = NULL;
                        i_zona--;
                    }
                }
            case 3:
                printf("Stampa Happa\n");
                for (i = 0; i < i_zona; i++)
                {
                    printf("%s\n", zona[i]);
                }
            case 4:
                printf("Elimina Happa\n");
                for (i = 0; i < i_zona; i++)
                {
                    zona[i] = NULL;
                    i_zona = 0;
                }
            case 5:
                printf("Chiudi Happa\n");
                break;
            case 6:
                printf("Salva Happa\n");
                break;
            case 7:
                printf("Ricarica Happa\n");
                break;
            case 8:
                printf("Restaura Happa\n");
                break;
            case 9:
                printf("Exit\n");
                break;
        }
    } while (sceltaOpzioniMenu != 9);
}

```


L'applicazione è stata testata sul campo riportando risultati positivi.

Futuri ampliamenti:

- Download dei risultati di MOSS (MOSS li elimina dopo 14 giorni).
- Analisi statica dei progetti (es. Valgrind), per fornire un aiuto al docente con una pre-valutazione del progetto.
- Ottenere il nome e il cognome degli studenti dal file ReadMe delle repositories e costruire dinamicamente un gestionale degli strumenti.

Grazie per l'attenzione