FINAL PROJECT GROUP 3 PROGRESS REPORT

Work done so far:

1. [✓] Login selection screen.
2. [ ] Registration Screen.
3. [✓] Login screen for both Teacher and student.
4. [✓] The Calendar layout.
5. [✓] Data Storage.
6. [✓] References in the paper.
7. [✓] The General and specific objectives in the documentation paper.
8. [ ] Flow Chart
9. [ ] Pseudo Code

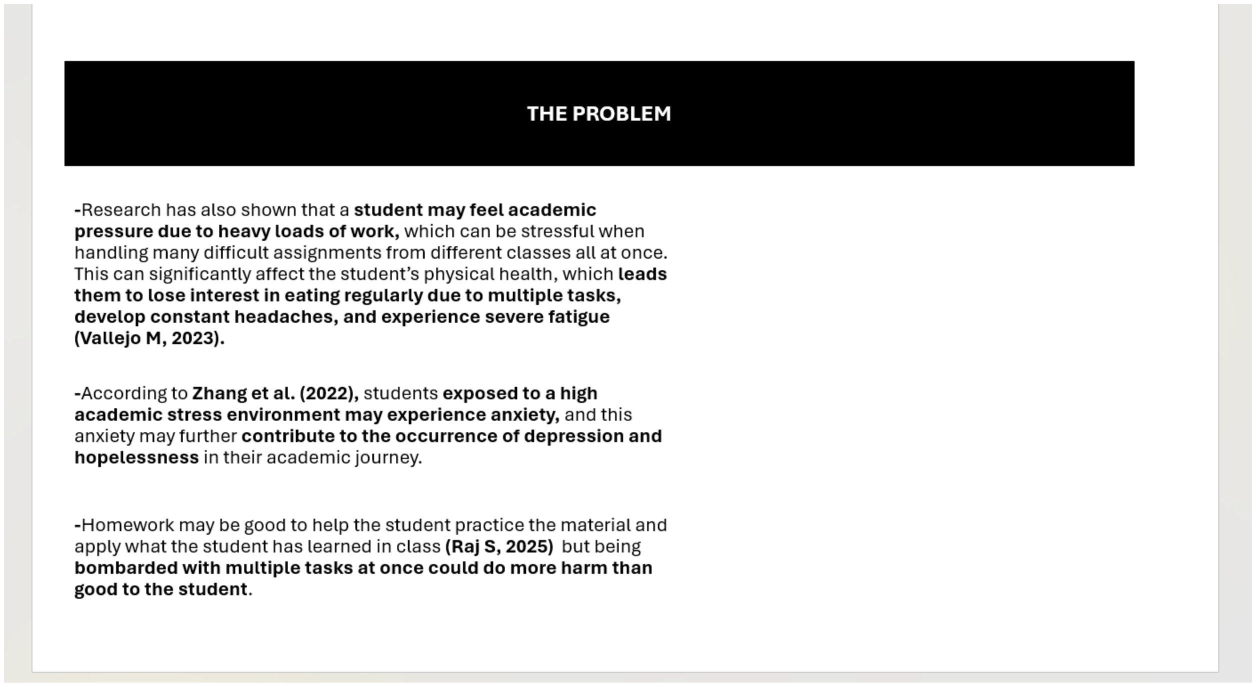
Contributions:

Braganza, Ralph Angelov F. – working the documentation paper (Finding references and general and specific objectives on the paper).
Condino, Niel Vincent B. – Created the calendar, data storage system, printing layout and started the PowerPoint creation.
Lisud, Ian Paulo S - Created the base login system and the menus.
Lopez, Andrei Dion C – Made the login system menus dynamically interactive.

FINAL PROJECT GROUP 3 PROGRESS REPORT

PROOF OF PROGRESS

PowerPoint creation (Condino)

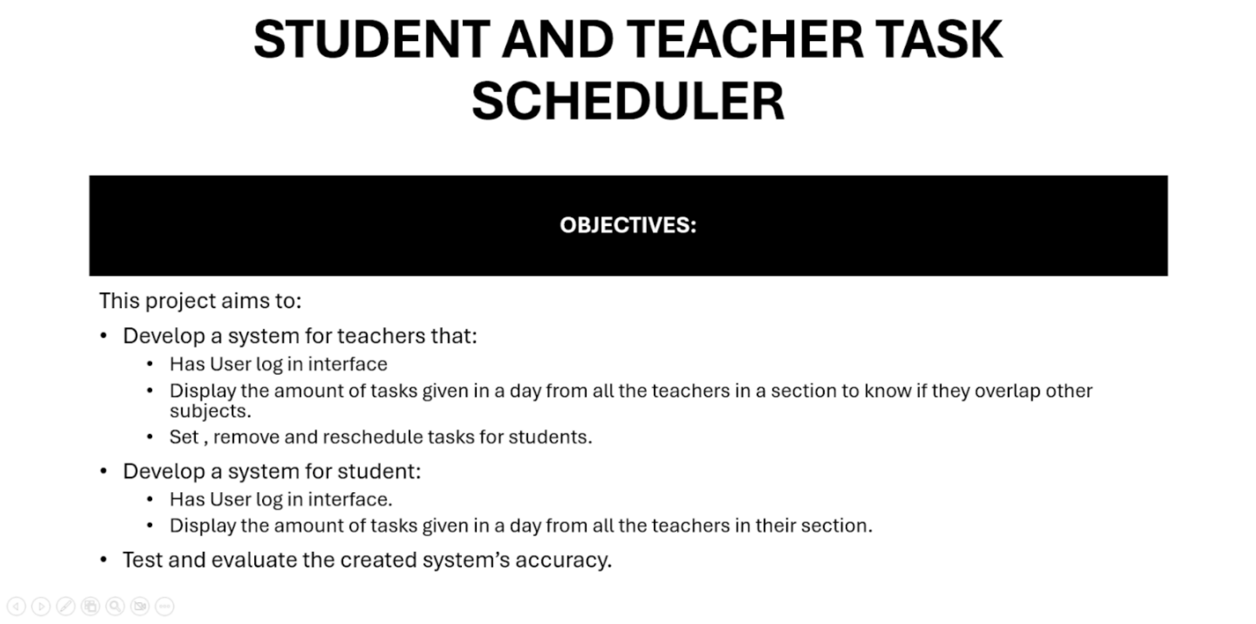


THE PROBLEM

-Research has also shown that a **student may feel academic pressure due to heavy loads of work,** which can be stressful when handling many difficult assignments from different classes all at once. This can significantly affect the student's physical health, which **leads them to lose interest in eating regularly due to multiple tasks, develop constant headaches, and experience severe fatigue (Vallejo M, 2023).**

-According to **Zhang et al. (2022),** students **exposed to a high academic stress environment may experience anxiety,** and this anxiety may further **contribute to the occurrence of depression and hopelessness** in their academic journey.

-Homework may be good to help the student practice the material and apply what the student has learned in class **(Raj S, 2025)** but being **bombarded with multiple tasks at once could do more harm than good to the student**.



STUDENT AND TEACHER TASK SCHEDULER

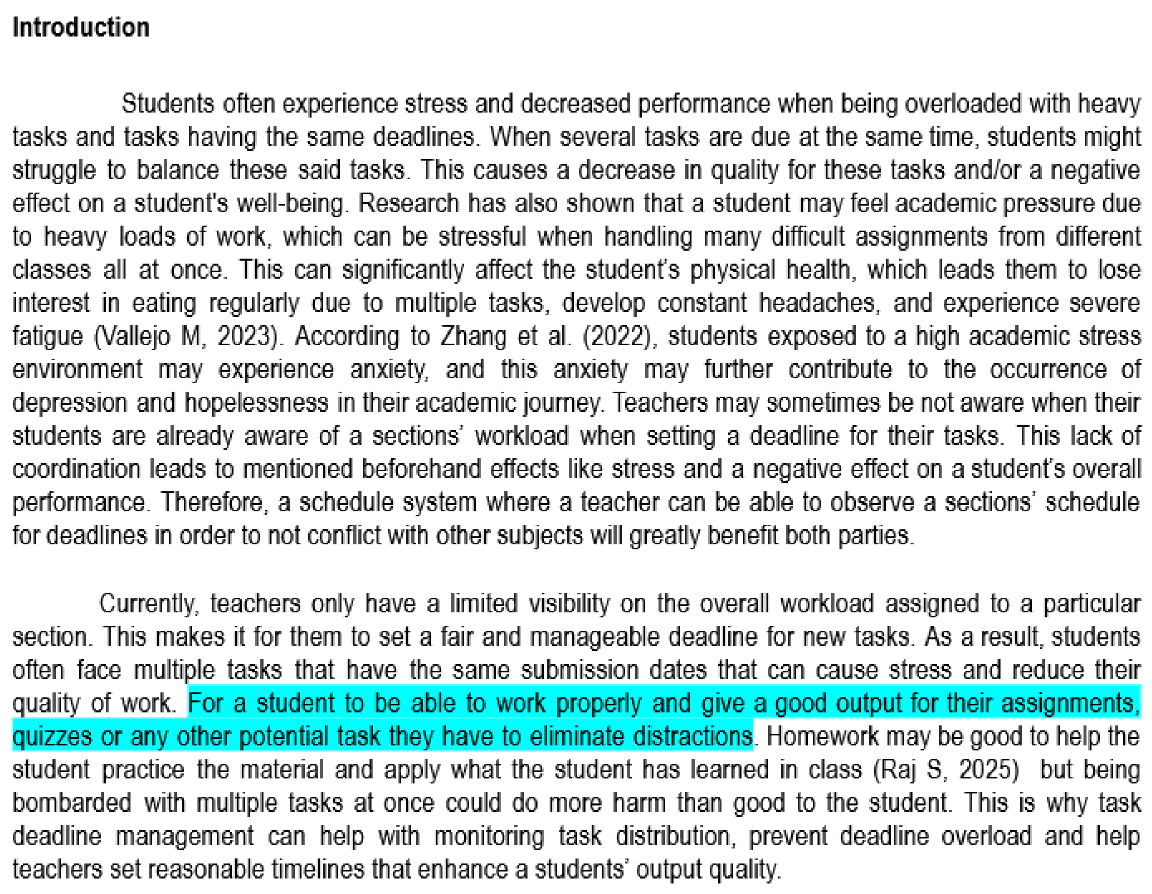
OBJECTIVES:

This project aims to:

- Develop a system for teachers that:
  - Has User log in interface
  - Display the amount of tasks given in a day from all the teachers in a section to know if they overlap other subjects.
  - Set , remove and reschedule tasks for students.
- Develop a system for student:
  - Has User log in interface.
  - Display the amount of tasks given in a day from all the teachers in their section.
- Test and evaluate the created system's accuracy.

FINAL PROJECT GROUP 3 PROGRESS REPORT

Fixed the recency of our documentation (Braganza)



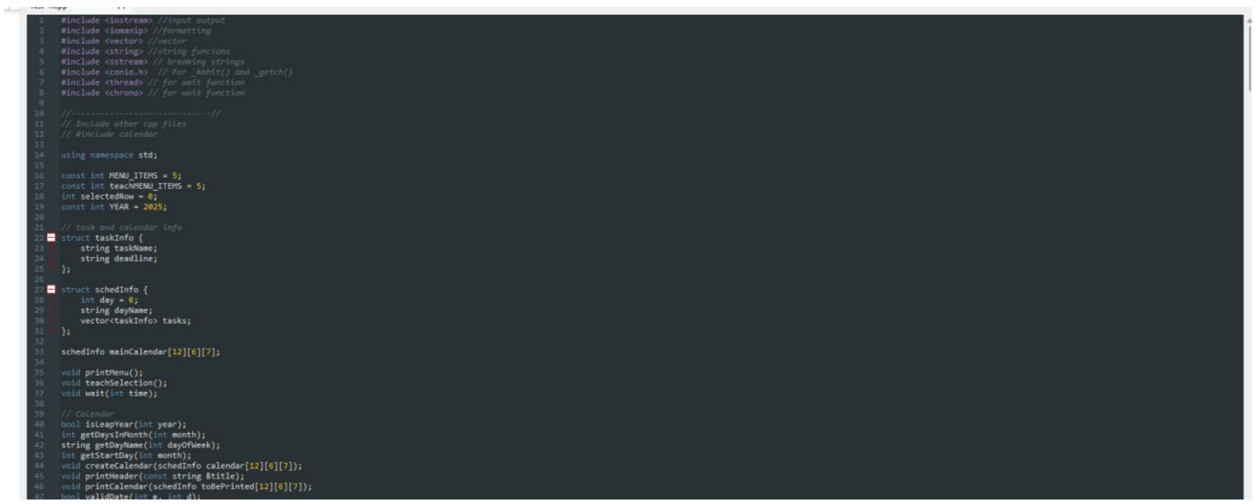
Introduction

Students often experience stress and decreased performance when being overloaded with heavy tasks and tasks having the same deadlines. When several tasks are due at the same time, students might struggle to balance these said tasks. This causes a decrease in quality for these tasks and/or a negative effect on a student's well-being. Research has also shown that a student may feel academic pressure due to heavy loads of work, which can be stressful when handling many difficult assignments from different classes all at once. This can significantly affect the student's physical health, which leads them to lose interest in eating regularly due to multiple tasks, develop constant headaches, and experience severe fatigue (Vallejo M, 2023). According to Zhang et al. (2022), students exposed to a high academic stress environment may experience anxiety, and this anxiety may further contribute to the occurrence of depression and hopelessness in their academic journey. Teachers may sometimes be not aware when their students are already aware of a sections' workload when setting a deadline for their tasks. This lack of coordination leads to mentioned beforehand effects like stress and a negative effect on a student's overall performance. Therefore, a schedule system where a teacher can be able to observe a sections' schedule for deadlines in order to not conflict with other subjects will greatly benefit both parties.

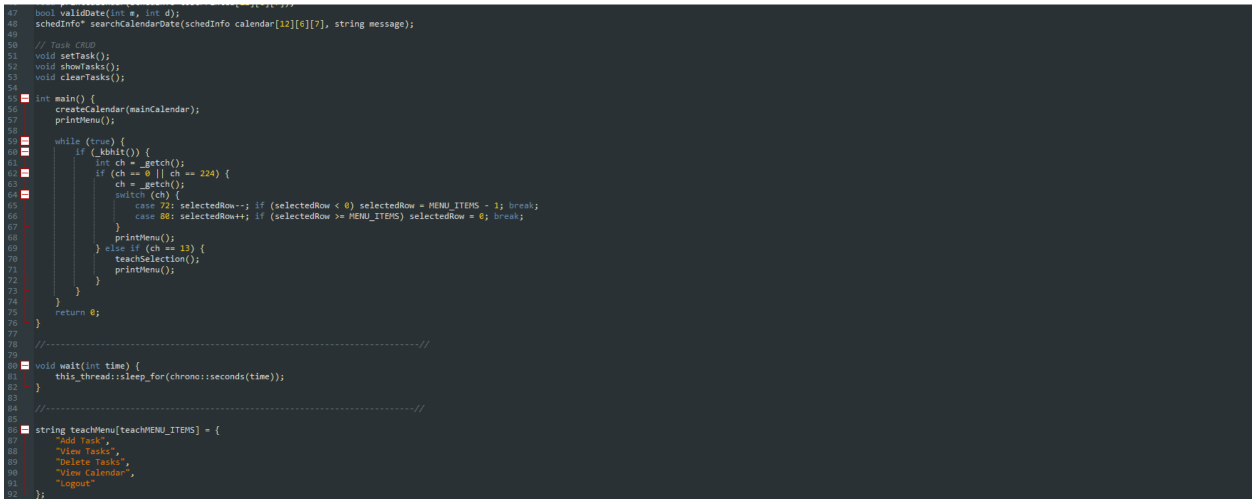
Currently, teachers only have a limited visibility on the overall workload assigned to a particular section. This makes it for them to set a fair and manageable deadline for new tasks. As a result, students often face multiple tasks that have the same submission dates that can cause stress and reduce their quality of work. For a student to be able to work properly and give a good output for their assignments, quizzes or any other potential task they have to eliminate distractions. Homework may be good to help the student practice the material and apply what the student has learned in class (Raj S, 2025) but being bombarded with multiple tasks at once could do more harm than good to the student. This is why task deadline management can help with monitoring task distribution, prevent deadline overload and help teachers set reasonable timelines that enhance a students' output quality.

Code progress (Lisud & Lopez)

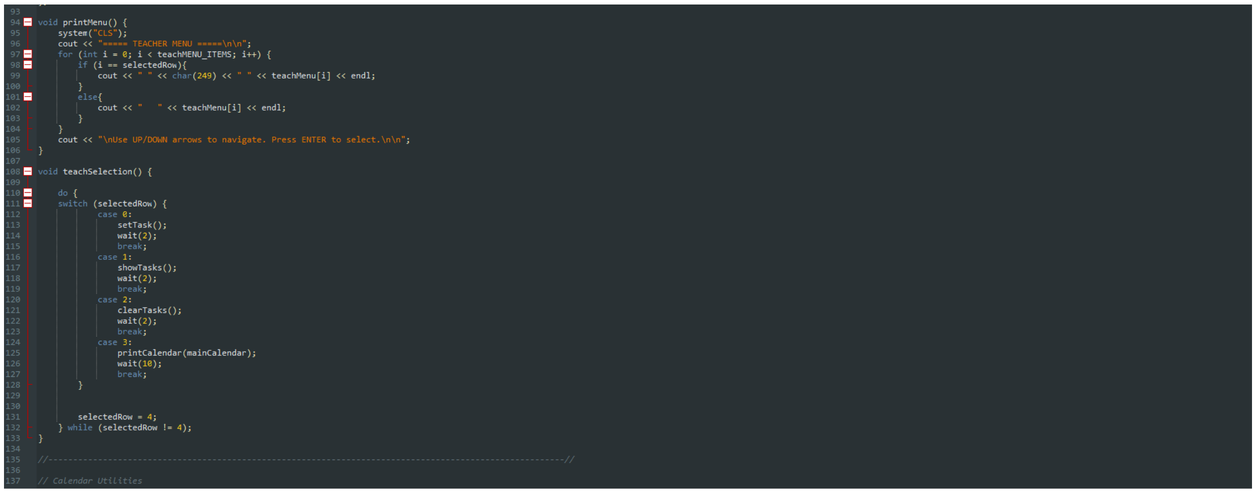
FINAL PROJECT GROUP 3 PROGRESS REPORT



```cpp
#include <iostream> //input output
#include <iomanip> //formatting
#include <vector> //vector
#include <string> //string funcions
#include <sstream> // breaking strings
#include <conio.h>  // For _kbhit() and _getch()
#include <thread> // for wait function
#include <chrono> // for wait function

//----------------------------//
// Include other cpp files
// #include calendar

using namespace std;

const int MENU_ITEMS = 5;
const int teachMENU_ITEMS = 5;
int selectedRow = 0;
const int YEAR = 2025;

// task and calendar info
struct taskInfo {
    string taskName;
    string deadline;
};

struct schedInfo {
    int day = 0;
    string dayName;
    vector<taskInfo> tasks;
};

schedInfo mainCalendar[12][6][7];

void printMenu();
void teachSelection();
void wait(int time);

// Calendar
bool isLeapYear(int year);
int getDaysInMonth(int month);
string getDayName(int dayOfWeek);
int getStartDay(int month);
void createCalendar(schedInfo calendar[12][6][7]);
void printHeader(const string &title);
void printCalendar(schedInfo toBePrinted[12][6][7]);
```

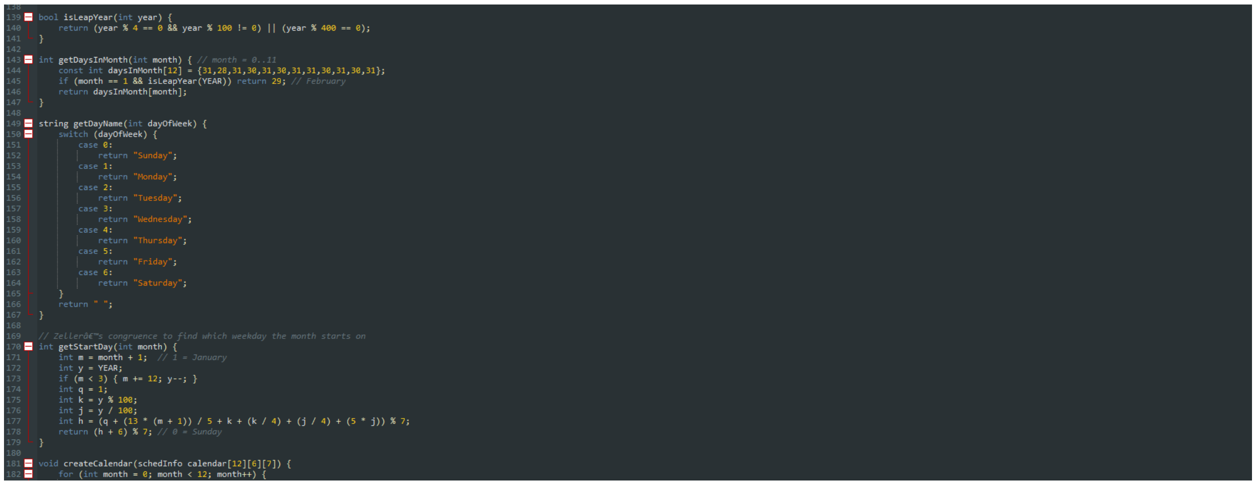


```cpp
bool validDate(int m, int d);
schedInfo* searchCalendarDate(schedInfo calendar[12][6][7], string message);

// Task CRUD
void setTask();
void showTasks();
void clearTasks();

int main() {
    createCalendar(mainCalendar);
    printMenu();

    while (true) {
        if (_kbhit()) {
            int ch = _getch();
            if (ch == 0 || ch == 224) {
                ch = _getch();
                switch (ch) {
                    case 72: selectedRow--; if (selectedRow < 0) selectedRow = MENU_ITEMS - 1; break;
                    case 80: selectedRow++; if (selectedRow >= MENU_ITEMS) selectedRow = 0; break;
                }
                printMenu();
            } else if (ch == 13) {
                teachSelection();
                printMenu();
            }
        }
    }
    return 0;
}

//----------------------------------------------------------------------//

void wait(int time) {
    this_thread::sleep_for(chrono::seconds(time));
}

//----------------------------------------------------------------------//

string teachMenu[teachMENU_ITEMS] = {
    "Add Task",
    "View Tasks",
    "Delete Tasks",
    "View Calendar",
    "Logout"
};
```

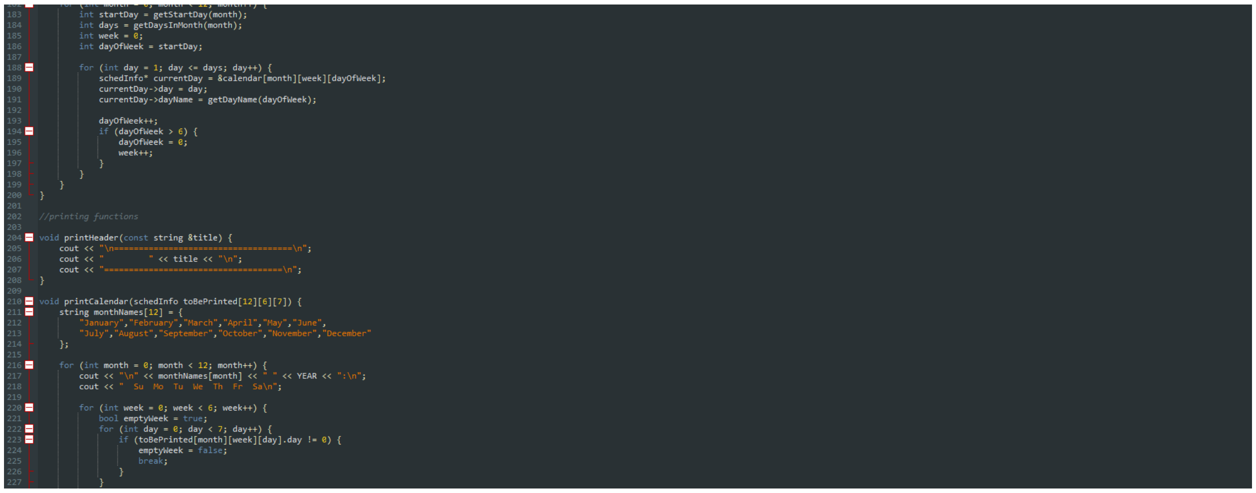


```cpp

void printMenu() {
    system("CLS");
    cout << "===== TEACHER MENU =====\n\n";
    for (int i = 0; i < teachMENU_ITEMS; i++) {
        if (i == selectedRow){
            cout << " " << char(249) << " " << teachMenu[i] << endl;
        }
        else{
            cout << "   " << teachMenu[i] << endl;
        }
    }
    cout << "\nUse UP/DOWN arrows to navigate. Press ENTER to select.\n\n";
}

void teachSelection() {

    do {
    switch (selectedRow) {
            case 0:
                setTask();
                wait(2);
                break;
            case 1:
                showTasks();
                wait(2);
                break;
            case 2:
                clearTasks();
                wait(2);
                break;
            case 3:
                printCalendar(mainCalendar);
                wait(10);
                break;
        }


        selectedRow = 4;
    } while (selectedRow != 4);
}

//------------------------------------------------------------------------------------------//

// Calendar Utilities
```

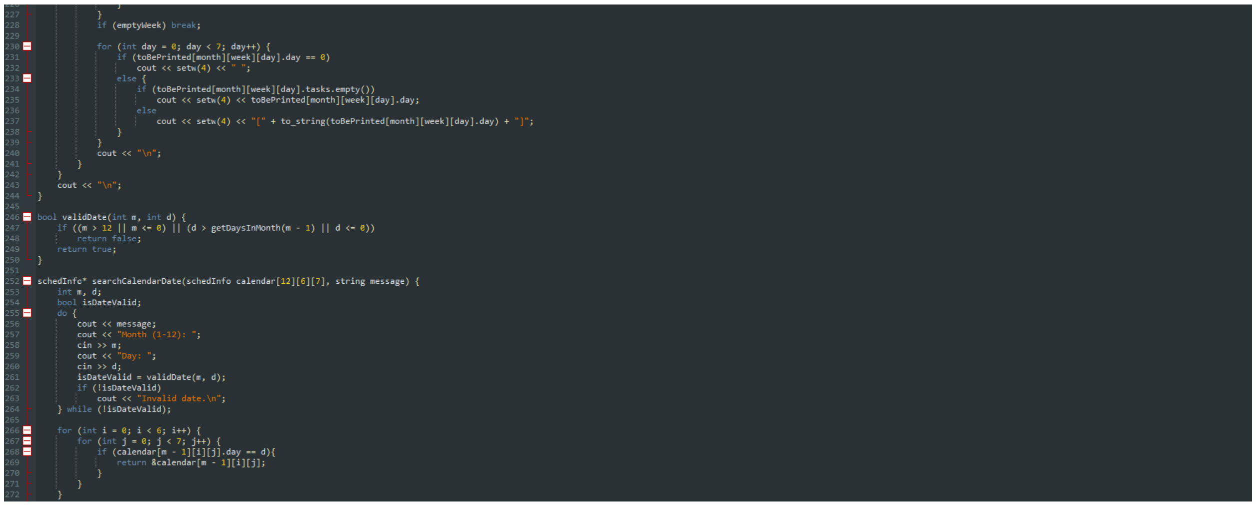
FINAL PROJECT GROUP 3 PROGRESS REPORT



```cpp
bool isLeapYear(int year) {
    return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
}

int getDaysInMonth(int month) { // month = 0..11
    const int daysInMonth[12] = {31,28,31,30,31,30,31,31,30,31,30,31};
    if (month == 1 && isLeapYear(YEAR)) return 29; // February
    return daysInMonth[month];
}

string getDayName(int dayOfWeek) {
    switch (dayOfWeek) {
        case 0:
            return "Sunday";
        case 1:
            return "Monday";
        case 2:
            return "Tuesday";
        case 3:
            return "Wednesday";
        case 4:
            return "Thursday";
        case 5:
            return "Friday";
        case 6:
            return "Saturday";
    }
    return " ";
}

// Zellerâ€™s congruence to find which weekday the month starts on
int getStartDay(int month) {
    int m = month + 1;  // 1 = January
    int y = YEAR;
    if (m < 3) { m += 12; y--; }
    int q = 1;
    int k = y % 100;
    int j = y / 100;
    int h = (q + (13 * (m + 1)) / 5 + k + (k / 4) + (j / 4) + (5 * j)) % 7;
    return (h + 6) % 7; // 0 = Sunday
}

void createCalendar(schedInfo calendar[12][6][7]) {
    for (int month = 0; month < 12; month++) {
```



```cpp
        int startDay = getStartDay(month);
        int days = getDaysInMonth(month);
        int week = 0;
        int dayOfWeek = startDay;

        for (int day = 1; day <= days; day++) {
            schedInfo* currentDay = &calendar[month][week][dayOfWeek];
            currentDay->day = day;
            currentDay->dayName = getDayName(dayOfWeek);

            dayOfWeek++;
            if (dayOfWeek > 6) {
                dayOfWeek = 0;
                week++;
            }
        }
    }
}

//printing functions

void printHeader(const string &title) {
    cout << "\n====================================\n";
    cout << "          " << title << "\n";
    cout << "====================================\n";
}

void printCalendar(schedInfo toBePrinted[12][6][7]) {
    string monthNames[12] = {
        "January","February","March","April","May","June",
        "July","August","September","October","November","December"
    };

    for (int month = 0; month < 12; month++) {
        cout << "\n" << monthNames[month] << " " << YEAR << ":\n";
        cout << "  Su  Mo  Tu  We  Th  Fr  Sa\n";

        for (int week = 0; week < 6; week++) {
            bool emptyWeek = true;
            for (int day = 0; day < 7; day++) {
                if (toBePrinted[month][week][day].day != 0) {
                    emptyWeek = false;
                    break;
                }
            }
```



```cpp
            }
            if (emptyWeek) break;

            for (int day = 0; day < 7; day++) {
                if (toBePrinted[month][week][day].day == 0)
                    cout << setw(4) << " ";
                else {
                    if (toBePrinted[month][week][day].tasks.empty())
                        cout << setw(4) << toBePrinted[month][week][day].day;
                    else
                        cout << setw(4) << "[" + to_string(toBePrinted[month][week][day].day) + "]";
                }
            }
            cout << "\n";
        }
    }
    cout << "\n";
}

bool validDate(int m, int d) {
    if ((m > 12 || m <= 0) || (d > getDaysInMonth(m - 1) || d <= 0))
        return false;
    return true;
}

schedInfo* searchCalendarDate(schedInfo calendar[12][6][7], string message) {
    int m, d;
    bool isDateValid;
    do {
        cout << message;
        cout << "Month (1-12): ";
        cin >> m;
        cout << "Day: ";
        cin >> d;
        isDateValid = validDate(m, d);
        if (!isDateValid)
            cout << "Invalid date.\n";
    } while (!isDateValid);

    for (int i = 0; i < 6; i++) {
        for (int j = 0; j < 7; j++) {
            if (calendar[m - 1][i][j].day == d){
                return &calendar[m - 1][i][j];
            }
        }
    }
```