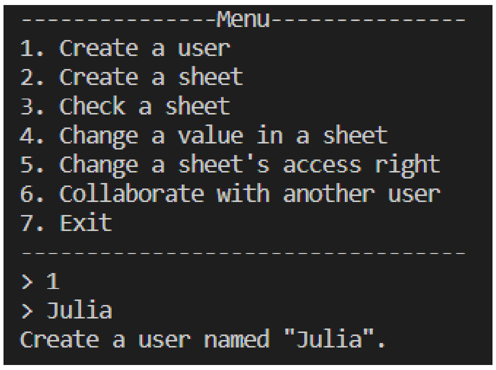
程式語言設計 <Final Project>

109502543 林怡萱

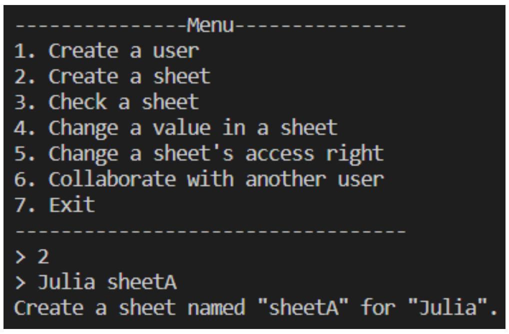
1. How does it work (基本與範例相同)

   1) create users: 輸入 “1” 後輸入 “user”



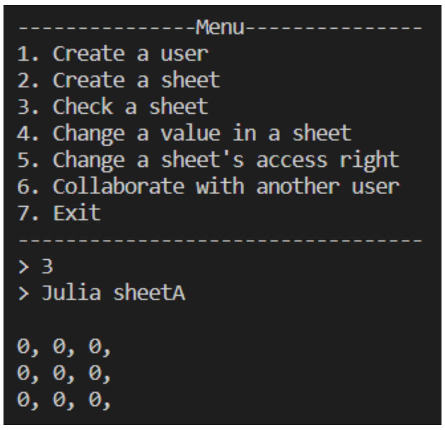
```
---------------Menu---------------
1. Create a user
2. Create a sheet
3. Check a sheet
4. Change a value in a sheet
5. Change a sheet's access right
6. Collaborate with another user
7. Exit
----------------------------------
> 1
> Julia
Create a user named "Julia".
```

   2) create sheet: 輸入 “2” 後輸入 “user sheet”

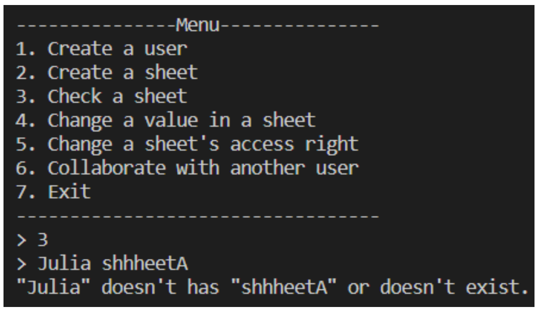


```
---------------Menu---------------
1. Create a user
2. Create a sheet
3. Check a sheet
4. Change a value in a sheet
5. Change a sheet's access right
6. Collaborate with another user
7. Exit
----------------------------------
> 2
> Julia sheetA
Create a sheet named "sheetA" for "Julia".
```

   3) print out the sheet: 輸入 “3” 後輸入 “user sheet”



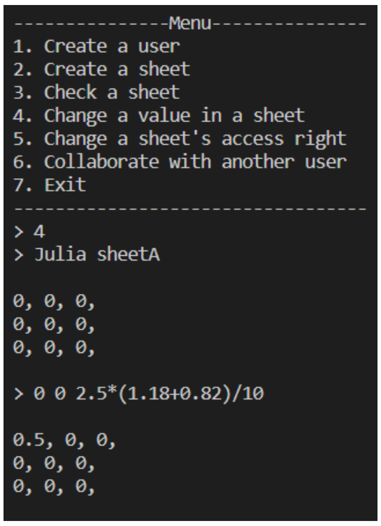
```
---------------Menu---------------
1. Create a user
2. Create a sheet
3. Check a sheet
4. Change a value in a sheet
5. Change a sheet's access right
6. Collaborate with another user
7. Exit
----------------------------------
> 3
> Julia sheetA

0, 0, 0,
0, 0, 0,
0, 0, 0,
```



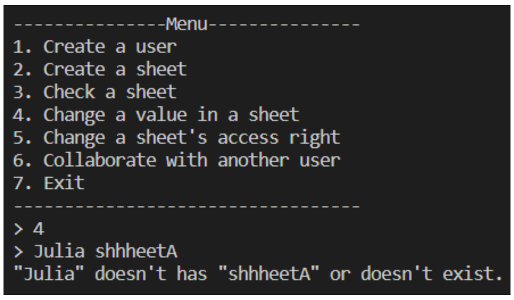
```
---------------Menu---------------
1. Create a user
2. Create a sheet
3. Check a sheet
4. Change a value in a sheet
5. Change a sheet's access right
6. Collaborate with another user
7. Exit
----------------------------------
> 3
> Julia shhheetA
"Julia" doesn't has "shhheetA" or doesn't exist.
```

輸入未創建的 user 或 sheet 時

4) change the content: 輸入 “4” 後輸入 “user sheet” 再輸入 “row column value” (value 可以是算式，包含括號與加減乘除)



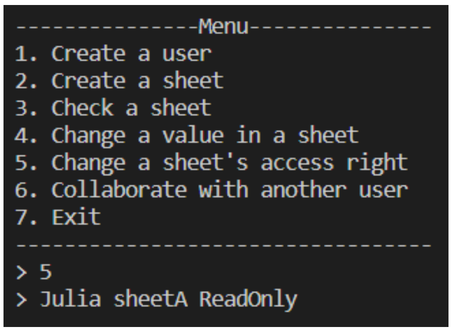
```
--------------Menu--------------
1. Create a user
2. Create a sheet
3. Check a sheet
4. Change a value in a sheet
5. Change a sheet's access right
6. Collaborate with another user
7. Exit
--------------------------------
> 4
> Julia sheetA

0, 0, 0,
0, 0, 0,
0, 0, 0,

> 0 0 2.5*(1.18+0.82)/10

0.5, 0, 0,
0, 0, 0,
0, 0, 0,
```



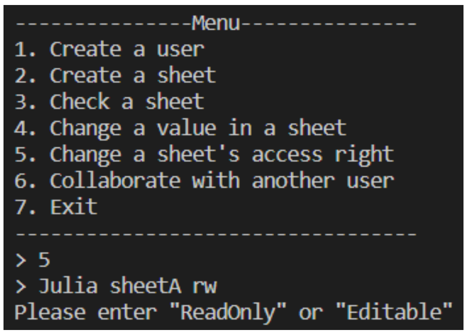
```
--------------Menu--------------
1. Create a user
2. Create a sheet
3. Check a sheet
4. Change a value in a sheet
5. Change a sheet's access right
6. Collaborate with another user
7. Exit
--------------------------------
> 4
> Julia shhheetA
"Julia" doesn't has "shhheetA" or doesn't exist.
```

輸入未創建的 user 或 sheet 時

5) change access rights: 輸入 “5” 後輸入 “user sheet ReadOnly/Editable”



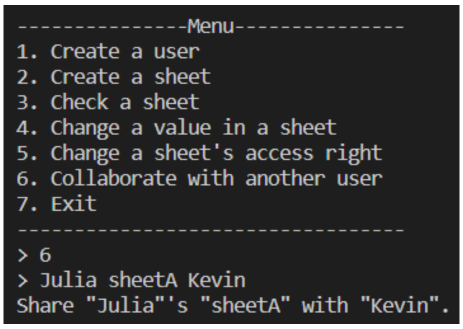
```
--------------Menu--------------
1. Create a user
2. Create a sheet
3. Check a sheet
4. Change a value in a sheet
5. Change a sheet's access right
6. Collaborate with another user
7. Exit
--------------------------------
> 5
> Julia sheetA ReadOnly
```



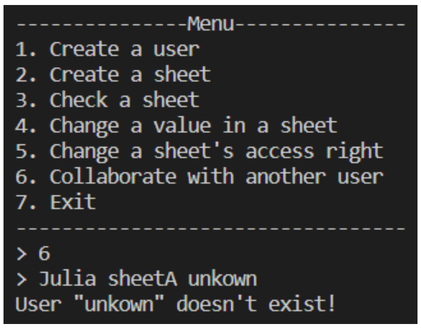
```
--------------Menu--------------
1. Create a user
2. Create a sheet
3. Check a sheet
4. Change a value in a sheet
5. Change a sheet's access right
6. Collaborate with another user
7. Exit
--------------------------------
> 5
> Julia sheetA rw
Please enter "ReadOnly" or "Editable"
```

(p.s.輸入未創建的 user 或 sheet 時也有同之前的提示)

6) share sheet with other users: 輸入 “6” 後輸入 “user1 sheet user2” (user1 與 user2 可以具有不同的編輯權限，分享時不論 user1 權限為何，user2 預設為 Editable(可用 5 變更))



```
--------------Menu--------------
1. Create a user
2. Create a sheet
3. Check a sheet
4. Change a value in a sheet
5. Change a sheet's access right
6. Collaborate with another user
7. Exit
--------------------------------
> 6
> Julia sheetA Kevin
Share "Julia"'s "sheetA" with "Kevin".
```

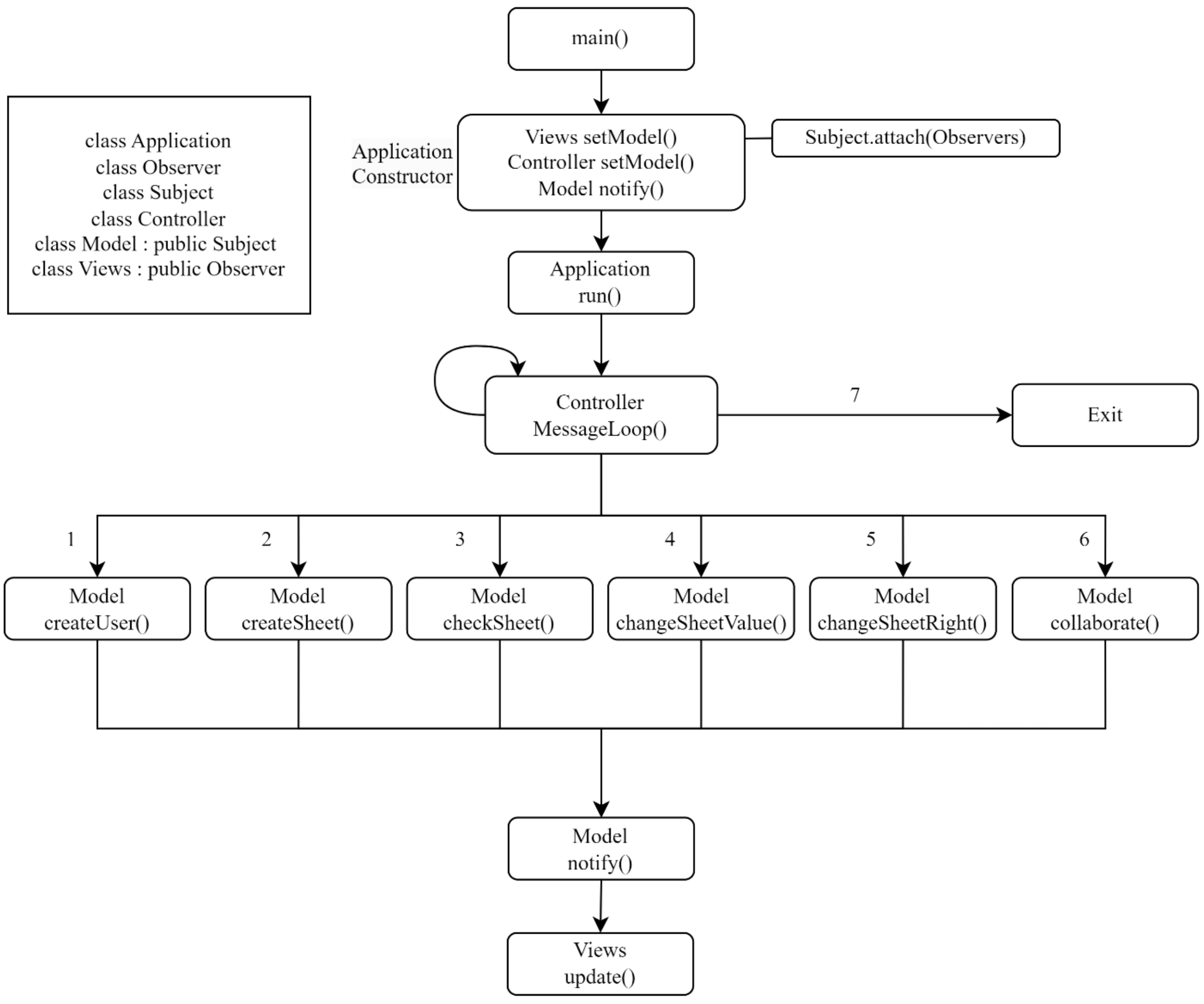


```
--------------Menu--------------
1. Create a user
2. Create a sheet
3. Check a sheet
4. Change a value in a sheet
5. Change a sheet's access right
6. Collaborate with another user
7. Exit
--------------------------------
> 6
> Julia sheetA unkown
User "unkown" doesn't exist!
```

(p.s.輸入未創建的 user1 或 sheet 時也有同之前的提示)

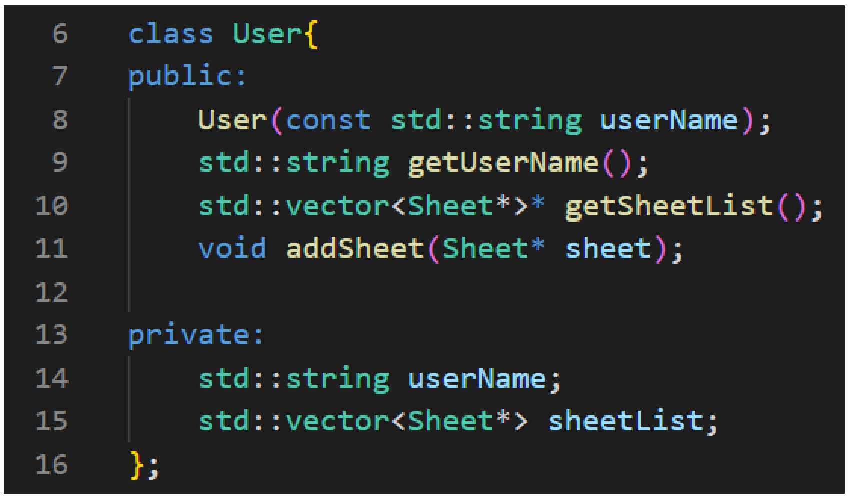
7) exit: 輸入 “7” 離開

2. Overview of the source code



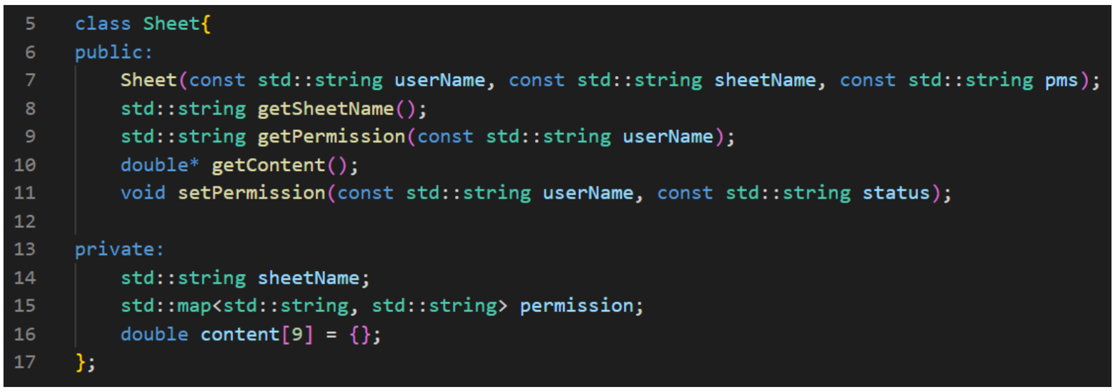
3. Data structures

class User 儲存 user 的名字及其所擁有的 sheet list



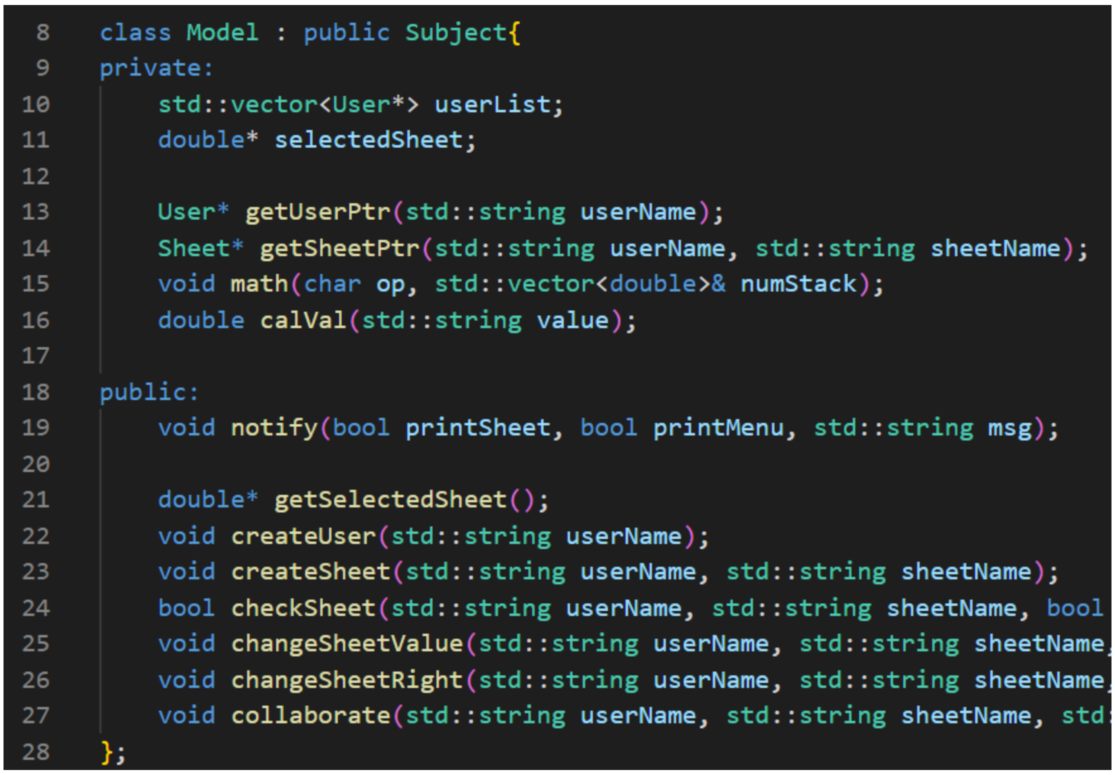
```cpp
class User{
public:
    User(const std::string userName);
    std::string getUserName();
    std::vector<Sheet*>* getSheetList();
    void addSheet(Sheet* sheet);

private:
    std::string userName;
    std::vector<Sheet*> sheetList;
};
```

class Sheet 儲存 sheet 的名字、擁有它的 user 所對應的權限 (ReadOnly/Editable)的字典以及 sheet 的內容



```cpp
class Sheet{
public:
    Sheet(const std::string userName, const std::string sheetName, const std::string pms);
    std::string getSheetName();
    std::string getPermission(const std::string userName);
    double* getContent();
    void setPermission(const std::string userName, const std::string status);

private:
    std::string sheetName;
    std::map<std::string, std::string> permission;
    double content[9] = {};
};
```

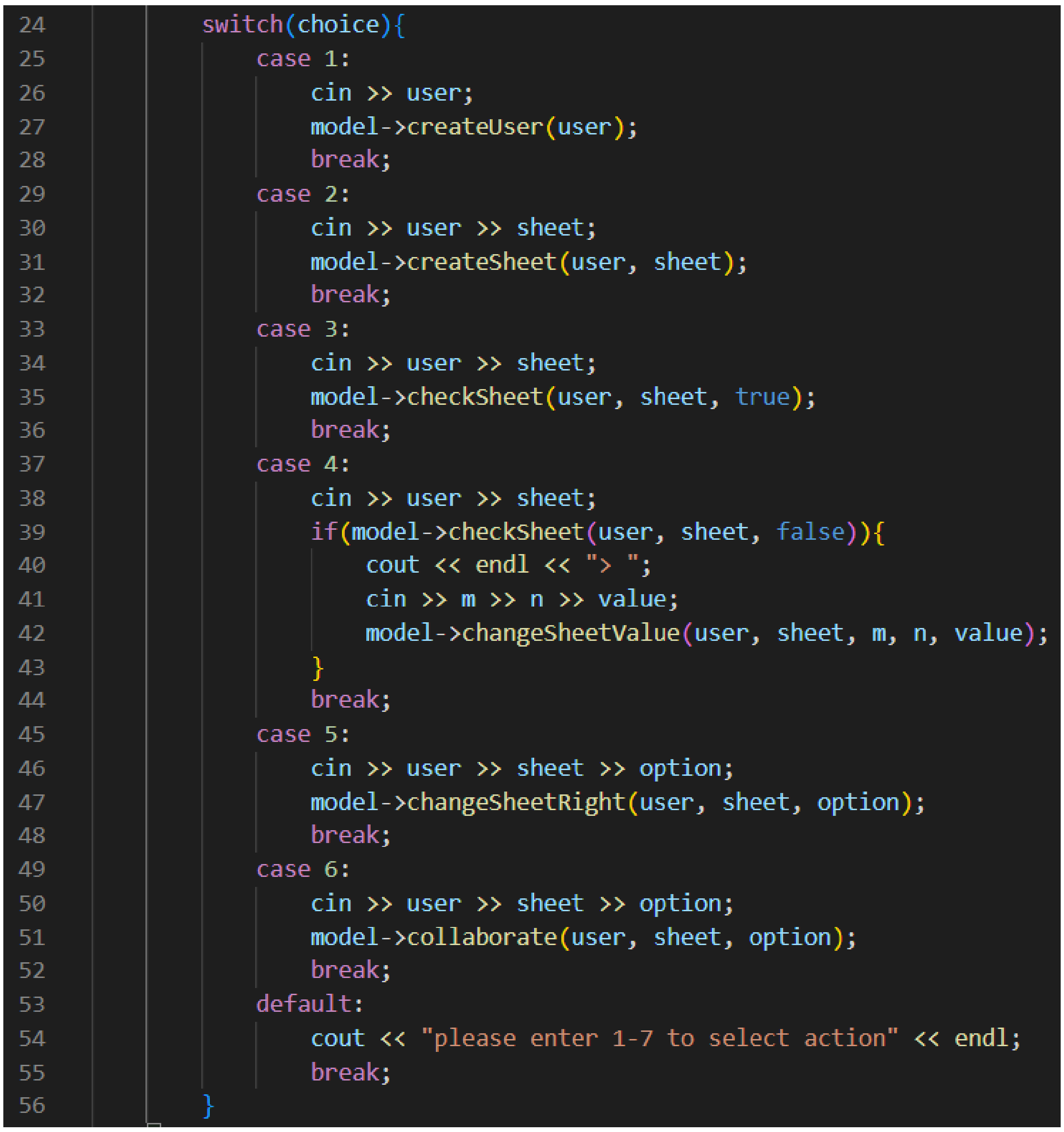
class Model 儲存 user list 並在 controller call 它的 function 時變更資料，然後 notify 所有的 observers(views)，由 views 輸出對應的訊息



```cpp
class Model : public Subject{
private:
    std::vector<User*> userList;
    double* selectedSheet;

    User* getUserPtr(std::string userName);
    Sheet* getSheetPtr(std::string userName, std::string sheetName);
    void math(char op, std::vector<double>& numStack);
    double calVal(std::string value);

public:
    void notify(bool printSheet, bool printMenu, std::string msg);

    double* getSelectedSheet();
    void createUser(std::string userName);
    void createSheet(std::string userName, std::string sheetName);
    bool checkSheet(std::string userName, std::string sheetName, bool
    void changeSheetValue(std::string userName, std::string sheetName
    void changeSheetRight(std::string userName, std::string sheetName
    void collaborate(std::string userName, std::string sheetName, std
};
```

4. how to switch on/off some functionalities

可以簡單的把 Controller 中 MessageLoop 的 case 拿掉，讓其無法呼叫對應功能的 function，亦或是將 Model 中執行該功能的 function 拿掉



```
        switch(choice){
            case 1:
                cin >> user;
                model->createUser(user);
                break;
            case 2:
                cin >> user >> sheet;
                model->createSheet(user, sheet);
                break;
            case 3:
                cin >> user >> sheet;
                model->checkSheet(user, sheet, true);
                break;
            case 4:
                cin >> user >> sheet;
                if(model->checkSheet(user, sheet, false)){
                    cout << endl << "> ";
                    cin >> m >> n >> value;
                    model->changeSheetValue(user, sheet, m, n, value);
                }
                break;
            case 5:
                cin >> user >> sheet >> option;
                model->changeSheetRight(user, sheet, option);
                break;
            case 6:
                cin >> user >> sheet >> option;
                model->collaborate(user, sheet, option);
                break;
            default:
                cout << "please enter 1-7 to select action" << endl;
                break;
        }
```

5. Design Patterns

主要是使用 MVC + Observer 的概念構成，由 controller 負責處理輸入，model 負責管理變更資料，再由 views 進行輸出，model 同時也作為 subject，views 作為 observer，在 model 更新資料時 call notify 通知所有觀察它的 views 進行輸出