National Research University Higher School of Economics
Faculty of Computer Science
Bachelor's Program "HSE University and University of London Double Degree
Program in Data Science and Business Analytics"

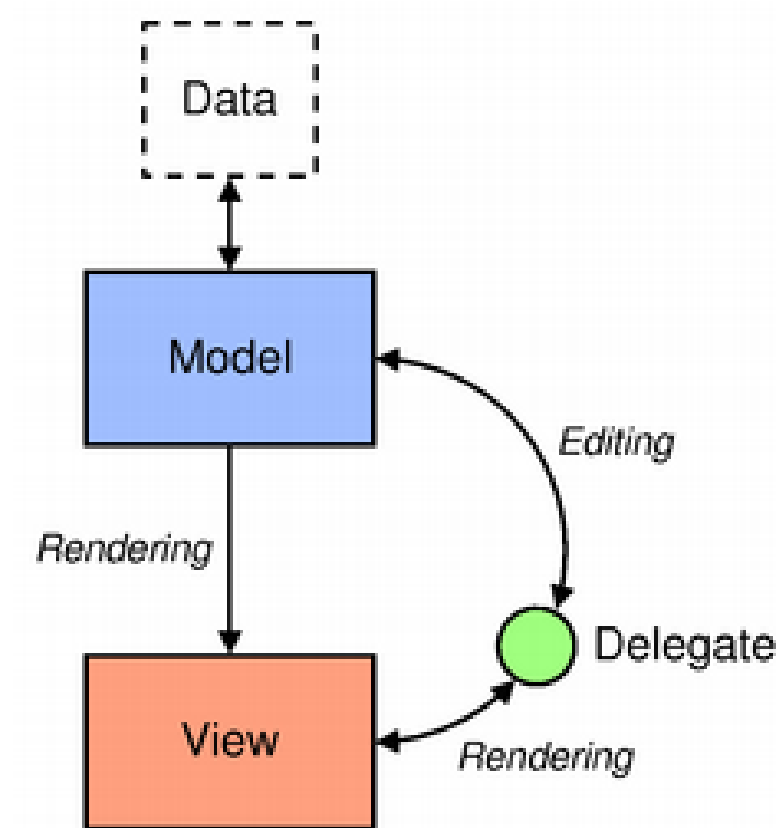# Introduction to Programming

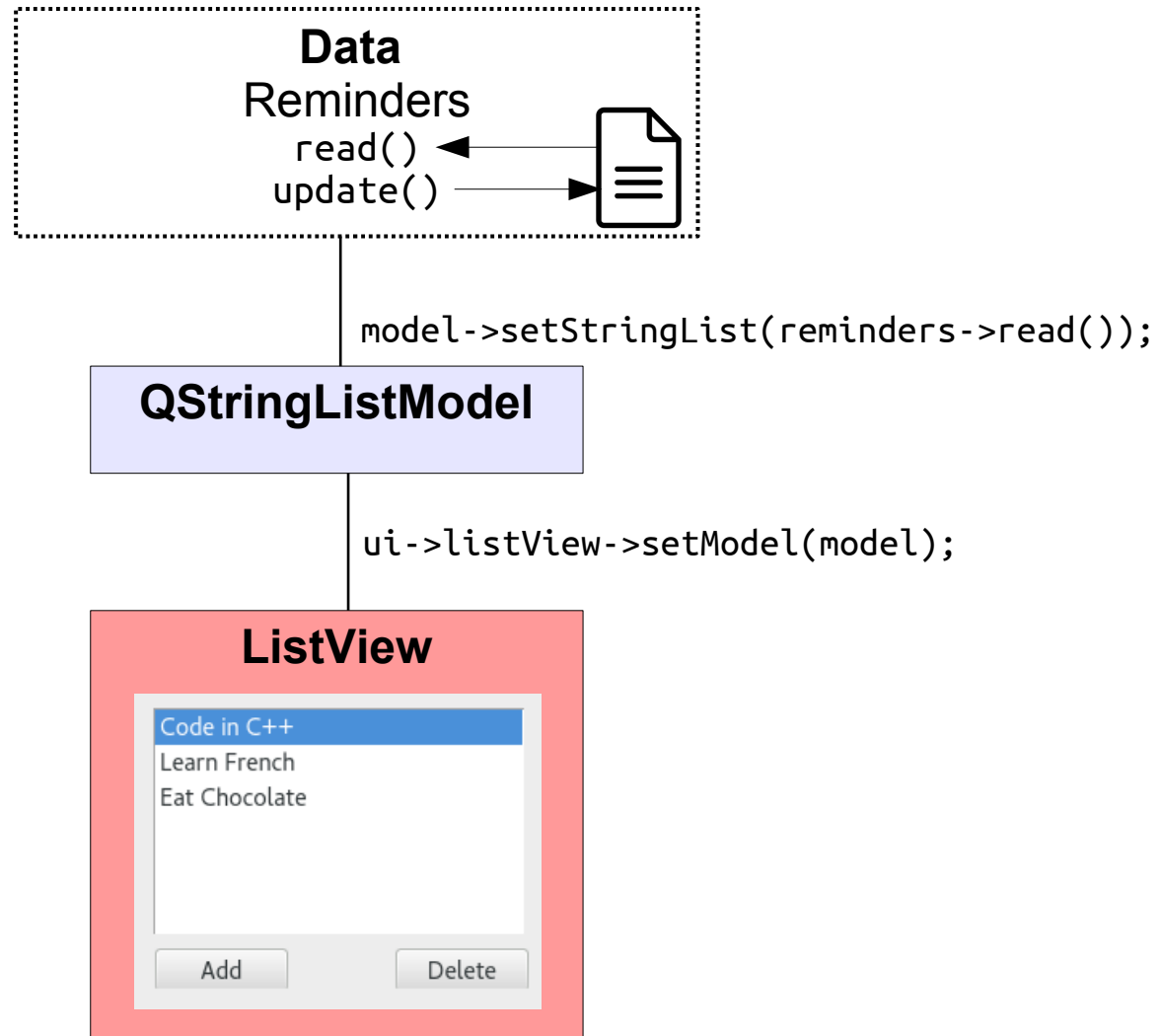# Workshop #33

Mon 17.05.2021

Julio Carrasquel

NATIONAL RESEARCH
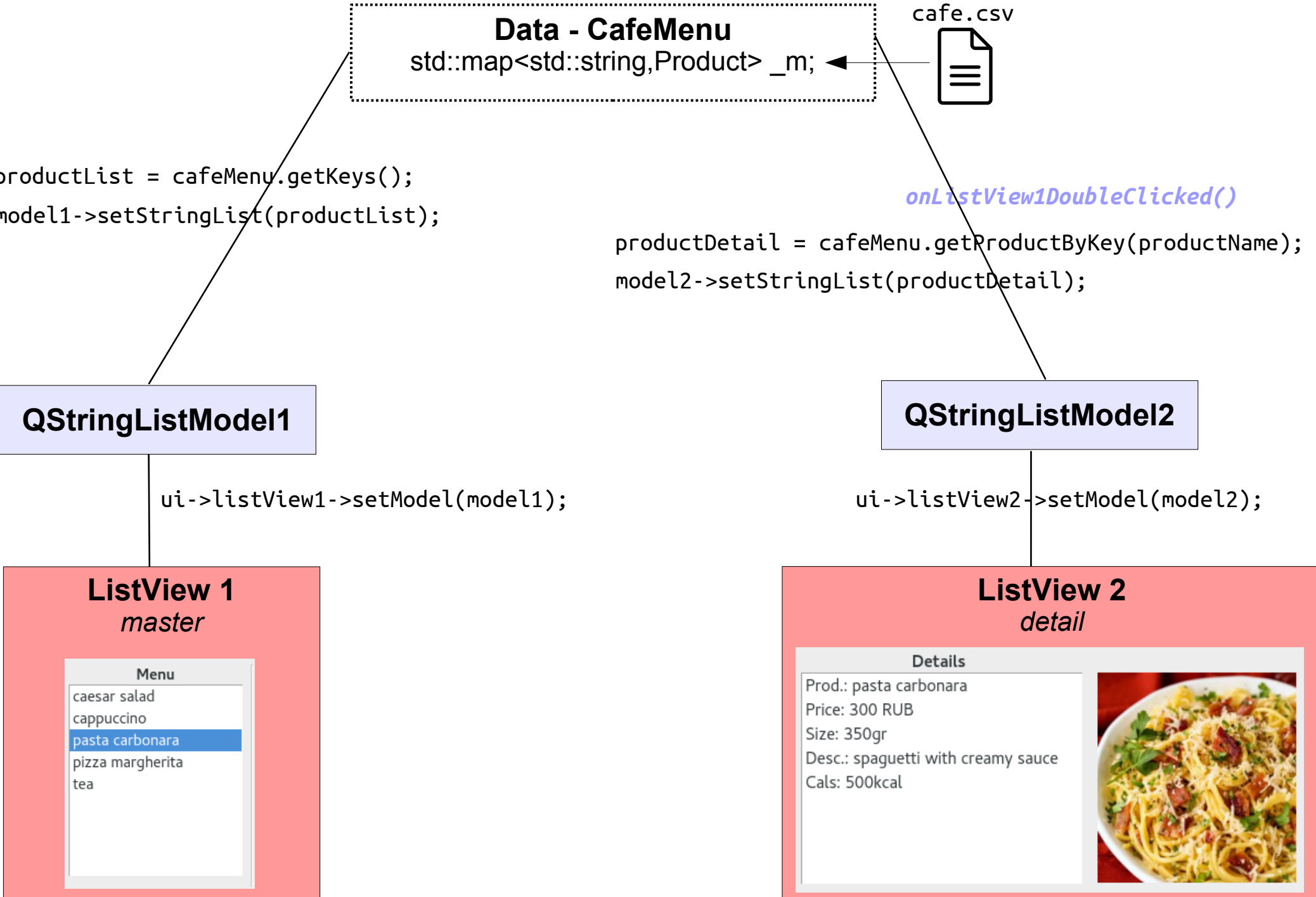UNIVERSITY

# Model-View programming

# Example #1: Reminders

```
QObject::connect(model, &QStringListModel::dataChanged, reminders, &Reminders::update);
QObject::connect(model, &QStringListModel::rowsRemoved, reminders, &Reminders::update);
```

**Data**
Reminders
read()
update()

`model->setStringList(reminders->read());`

**QStringListModel**

`ui->listView->setModel(model);`

**ListView**

Code in C++
Learn French
Eat Chocolate

Add        Delete

# Example #2: Master-Detail

**Data - CafeMenu**
std::map<std::string,Product> _m;

cafe.csv

productList = cafeMenu.getKeys();
model1->setStringList(productList);

*onListView1DoubleClicked()*

productDetail = cafeMenu.getProductByKey(productName);

model2->setStringList(productDetail);

**QStringListModel1**

**QStringListModel2**

ui->listView1->setModel(model1);

ui->listView2->setModel(model2);

**ListView 1**
*master*

| Menu |
|---|
| caesar salad |
| cappuccino |
| pasta carbonara |
| pizza margherita |
| tea |

**ListView 2**
*detail*

**Details**
Prod.: pasta carbonara
Price: 300 RUB
Size: 350gr
Desc.: spaguetti with creamy sauce
Cals: 500kcal

# Example #3: Table and *Custom* Delegate
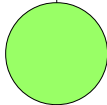
```
model = new QStandardItemModel(5, 2, this);
for(int row = 0; row < 5; ++row){
    for(int col = 0; col < 2; ++col){
        QModelIndex index = model->index(row, col, QmodelIndex());
        model->setData(index, 0); // initialize a cell in the model with zero value
    }
}
```

**QStandardItemModel**

ui->tableView->setModel(model)

**TableView**
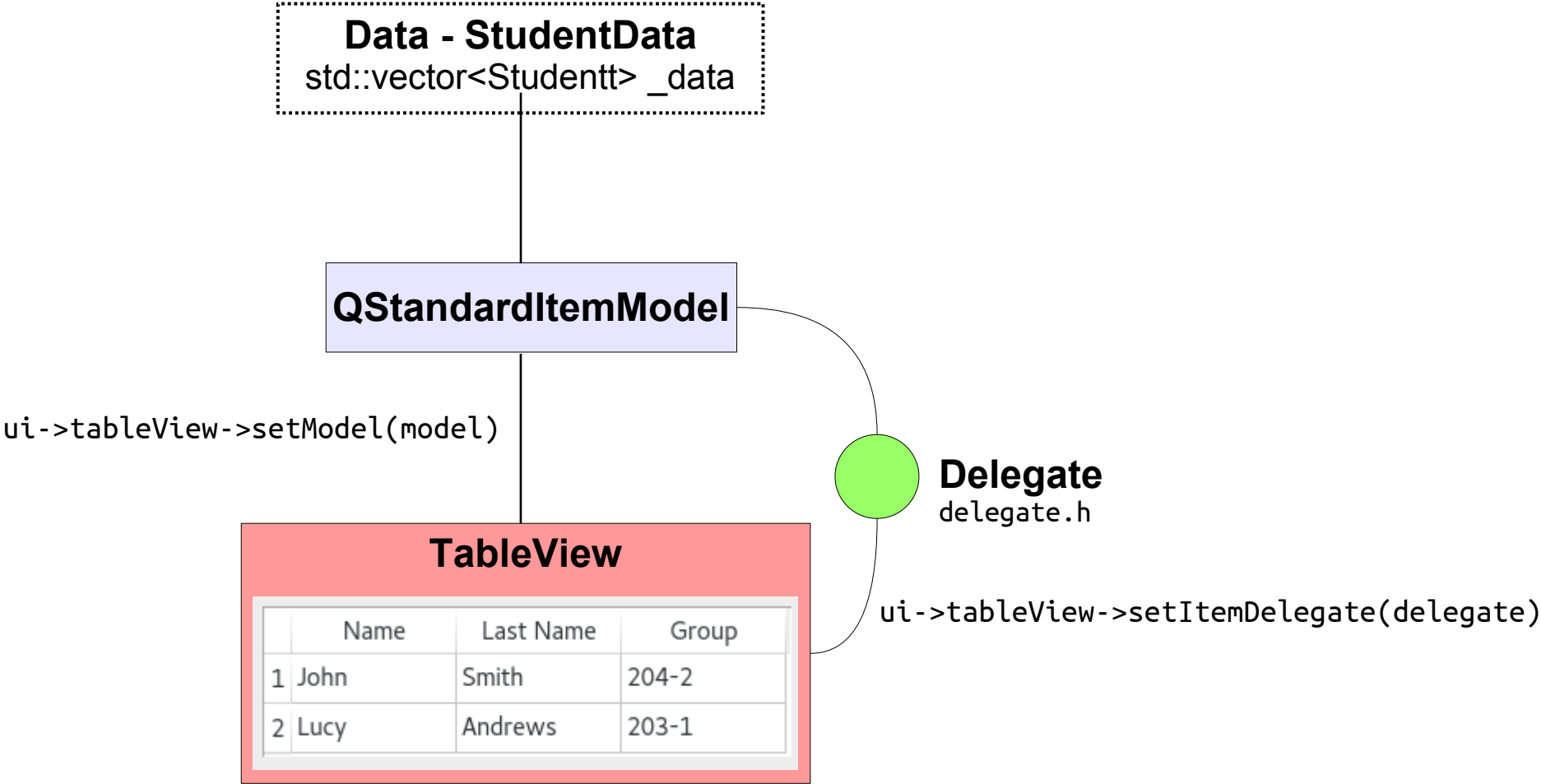
**Delegate**
delegate.h

ui->tableView->setItemDelegate(delegate)

| | 1 | 2 |
|---|---|---|
| 1 | 0 | 0 |
| 2 | 0 | 0 |
| 3 | 0 | 0 |
| 4 | 0 | 0 |
| 5 | 0 | 0 |

# Example #4: Student Table

**Data - StudentData**
std::vector<Studentt> _data

**QStandardItemModel**

ui->tableView->setModel(model)

**TableView**

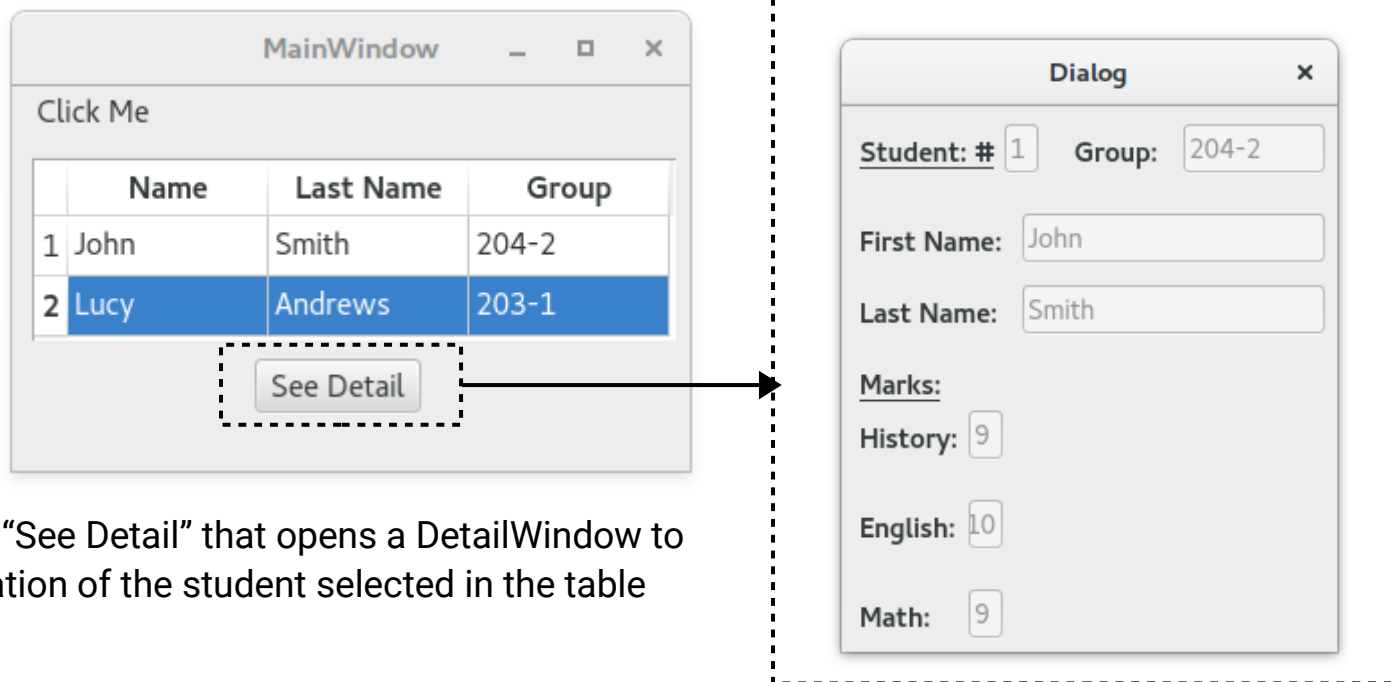| | Name | Last Name | Group |
|---|---|---|---|
| 1 | John | Smith | 204-2 |
| 2 | Lucy | Andrews | 203-1 |

🟢 **Delegate**
delegate.h

ui->tableView->setItemDelegate(delegate)

# Exercise



Implement a button "See Detail" that opens a DetailWindow to show all the information of the student selected in the table view.

# Exercise



Implement a button "See Detail" that opens a DetailWindow to show all the information of the student selected in the table view.

## Hints

1) The DetailWindow inherits from QDialog. Create the window via "New File/Project → Qt → QtDesigner Form Class". This will create the three files for DetailWindow (.h, .cpp, .ui). Remember to include the files in the CMakeLists.txt

2) Design detailwindow.ui with the designer as the example image.

3) In the class DetailWindow, implement the method setStudentDetail(int number, Student student) where

Number is the number of the table row in the MainWindow, and student is the student to display to data.

4) In mainwindow.ui create the button SeeDetail. Implement a slot (*go to slot*) that when the button "See Detail" is clicked it will create and open the DetailWindow. You can use the code below.

```cpp
int selectedRow = ui->tableView->currentIndex().row();
std :: cout << "[SEE DETAIL] of element in row " << selectedRow << std :: endl;
DetailWindow detailWindow;
detailWindow.setStudentDetail(selectedRow, studentData->getStudent(selectedRow));
detailWindow.setModal(true);
detailWindow.exec();
```

# Homeworks (per group)
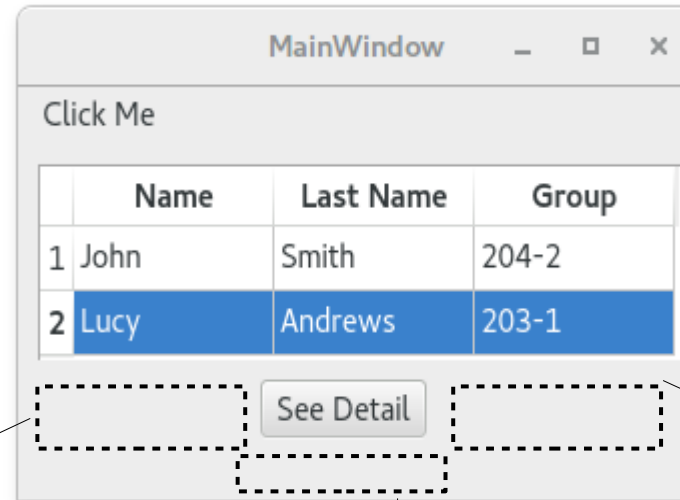


**Homework 1**
Implement a button "**Add**" that will open a new window with a form, to insert a new student in the view/model. Remember also to insert the new student in the vector inside the StudentDetail data.

**Homework 2**
Implement a button "**Delete**" that will delete the selected student from the view/model. Remember also to delete the student in the vector inside the StudentDetail data

**Homework 3**
Load the vector of students inside the StudentDetail data from a CSV file.
1) Implement a button **Save** that when clicked it saves the current content of the vector in the file.
2) Implement also the variant that when a student is modified, inserted or deleted from the table it saves the content of the vector in the file.