**WEEK 2**

**Name: KAVIN KISHORE SG**

**Roll Number: 7376221CS191**

**1. Create a QT label and line edit.**

**Code:**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

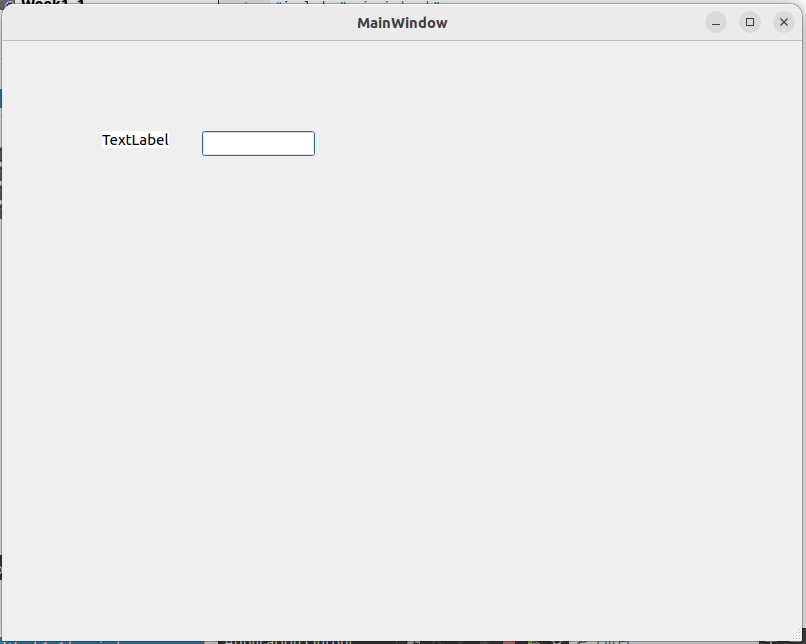
}

void MainWindow::on\_lineEdit\_textChanged(const QString &arg1)

{

}

Output:



**2.Create a QT Push button line edit label for addition.**

Code:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

float number1, number2, result = 0;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_lineEdit\_textChanged(const QString &arg1)

{

number1 = arg1.toFloat();

}

void MainWindow::on\_lineEdit\_2\_textChanged(const QString &arg1)

{

number2 = arg1.toFloat();

}

void MainWindow::on\_pushButton\_clicked()

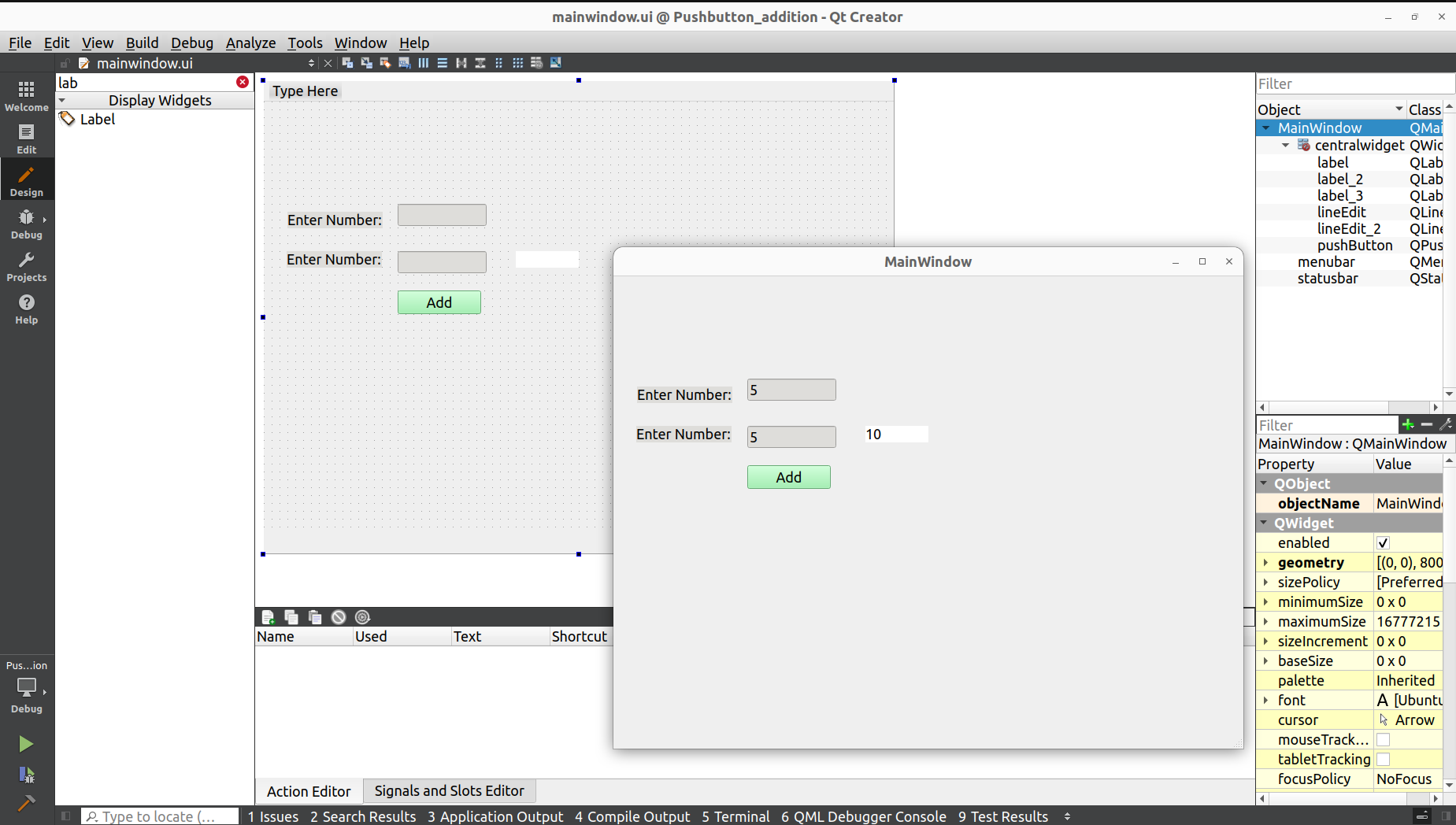
{

result = number1 + number2;

ui->label->setText(QString::number(result));

}

Output:



**3. Create a simple calculator by collecting two numbers from users and**

**perform (+,-,\* and /) and show the results in a line edit.**

Code:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

float num1,num2, add, sub, mult, divi;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_lineEdit\_textChanged(const QString &arg1)

{

num1 = arg1.toFloat();

}

void MainWindow::on\_lineEdit\_2\_textChanged(const QString &arg1)

{

num2 = arg1.toFloat();

}

void MainWindow::on\_pushButton\_clicked()

{

add = num1 + num2 ;

ui->label->setText(QString::number(add));

}

void MainWindow::on\_pushButton\_2\_clicked()

{

sub = num1 - num2 ;

ui->label->setText(QString::number(sub));

}

void MainWindow::on\_pushButton\_3\_clicked()

{

mult = num1 \* num2;

ui->label->setText(QString::number(mult));

}

void MainWindow::on\_pushButton\_4\_clicked()

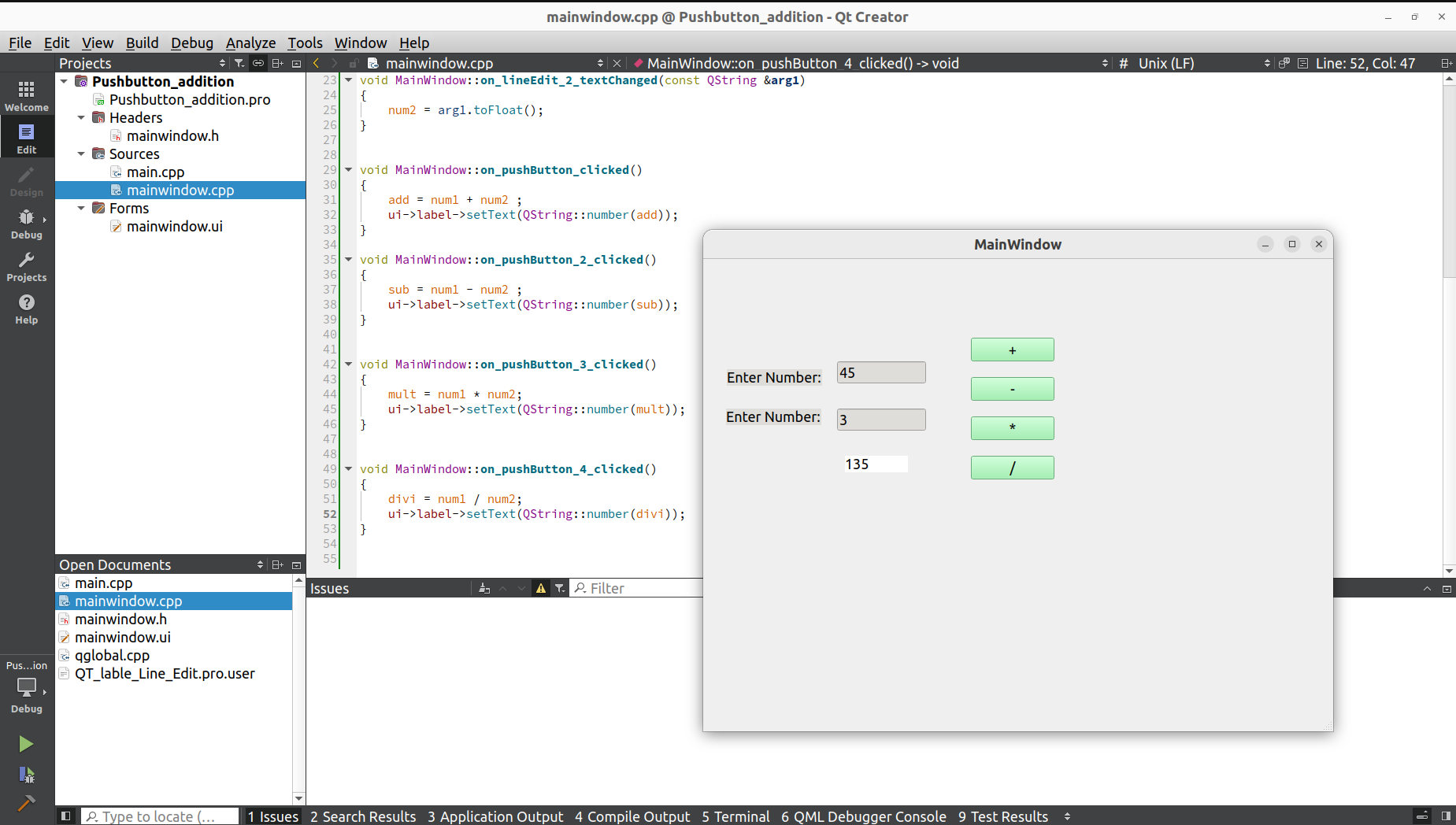
{

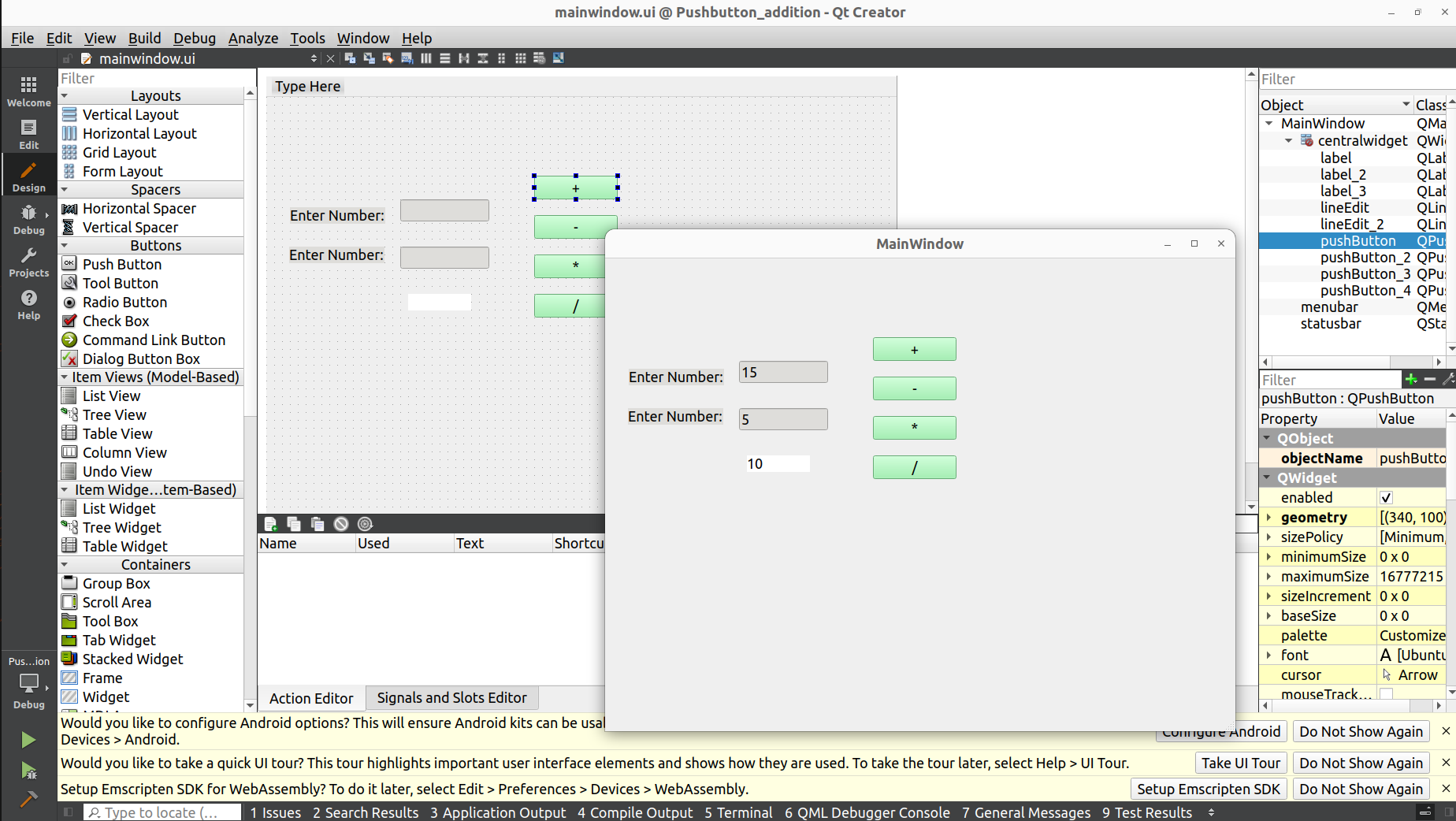
divi = num1 / num2;

ui->label->setText(QString::number(divi));

}

Output:





4. Create a List Widget for Insert, Delete Button.

Code:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

QString s;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_pushButton\_clicked()

{

ui->listWidget->addItem(s);

}

void MainWindow::on\_pushButton\_2\_clicked()

{

qDeleteAll(ui->listWidget->selectedItems());

}

void MainWindow::on\_pushButton\_3\_clicked()

{

ui->listWidget->clear();

}

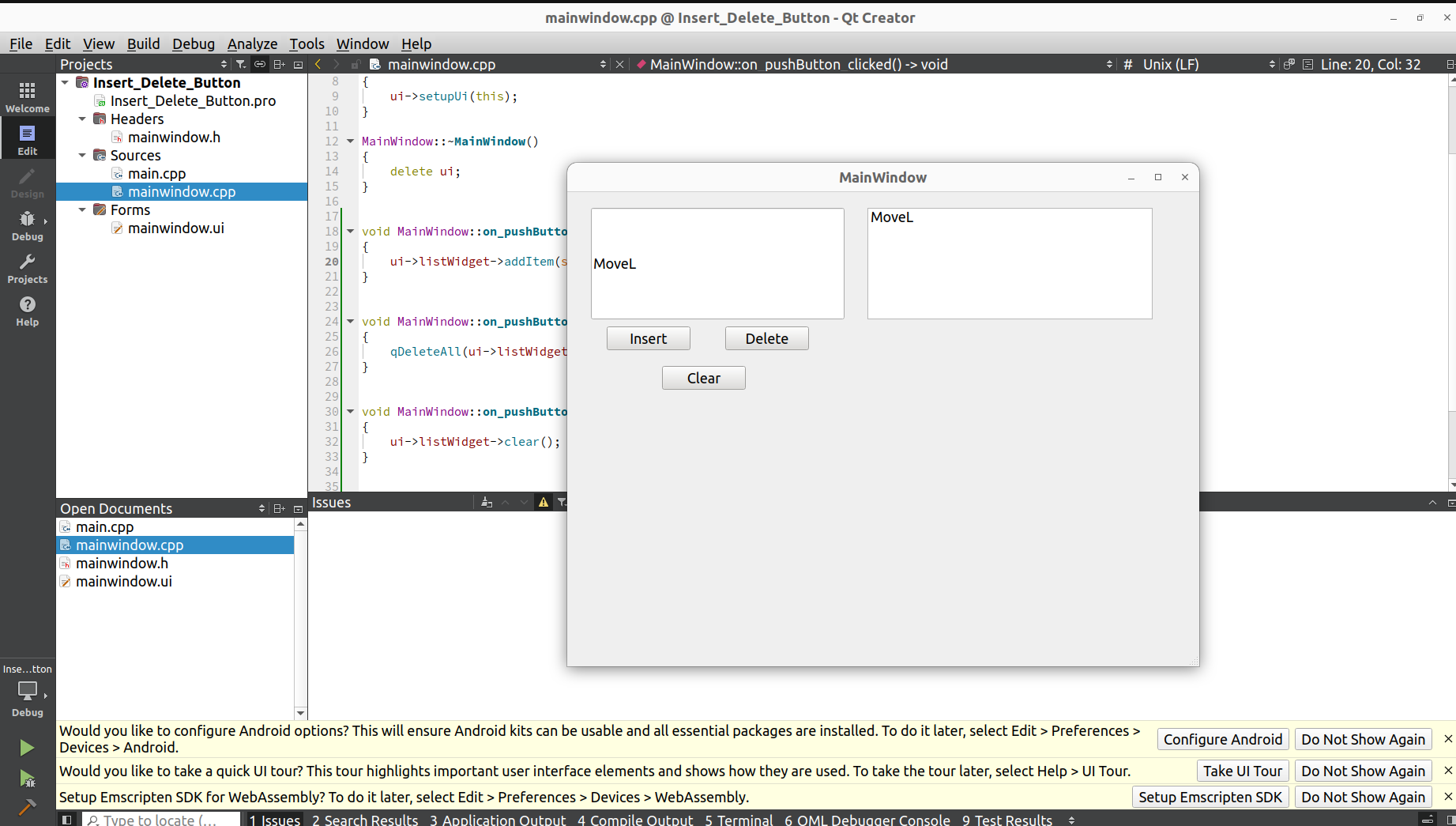
void MainWindow::on\_lineEdit\_textChanged(const QString &arg1)

{

s = arg1;

}

Output:





5. Create a three QT list widget .In the first list widget add the elements,second

widget has a line edit and updates its value in the list widget.In the third one

add or clear the values.

Code:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

float number1, number2;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_lineEdit\_textChanged(const QString &arg1)

{

number1 = arg1.toFloat();

}

void MainWindow::on\_pushButton\_4\_clicked()

{

for(int num = 0;num < 15; num++)

{

ui->listWidget->addItem(QString::number(num));

}

}

void MainWindow::on\_pushButton\_2\_clicked()

{

ui->listWidget\_2->addItem(QString::number(number1));

}

void MainWindow::on\_lineEdit\_2\_textChanged(const QString &arg1)

{

number2 = arg1.toFloat();

}

void MainWindow::on\_pushButton\_clicked()

{

ui->listWidget\_3->addItem(QString::number(number2));

}

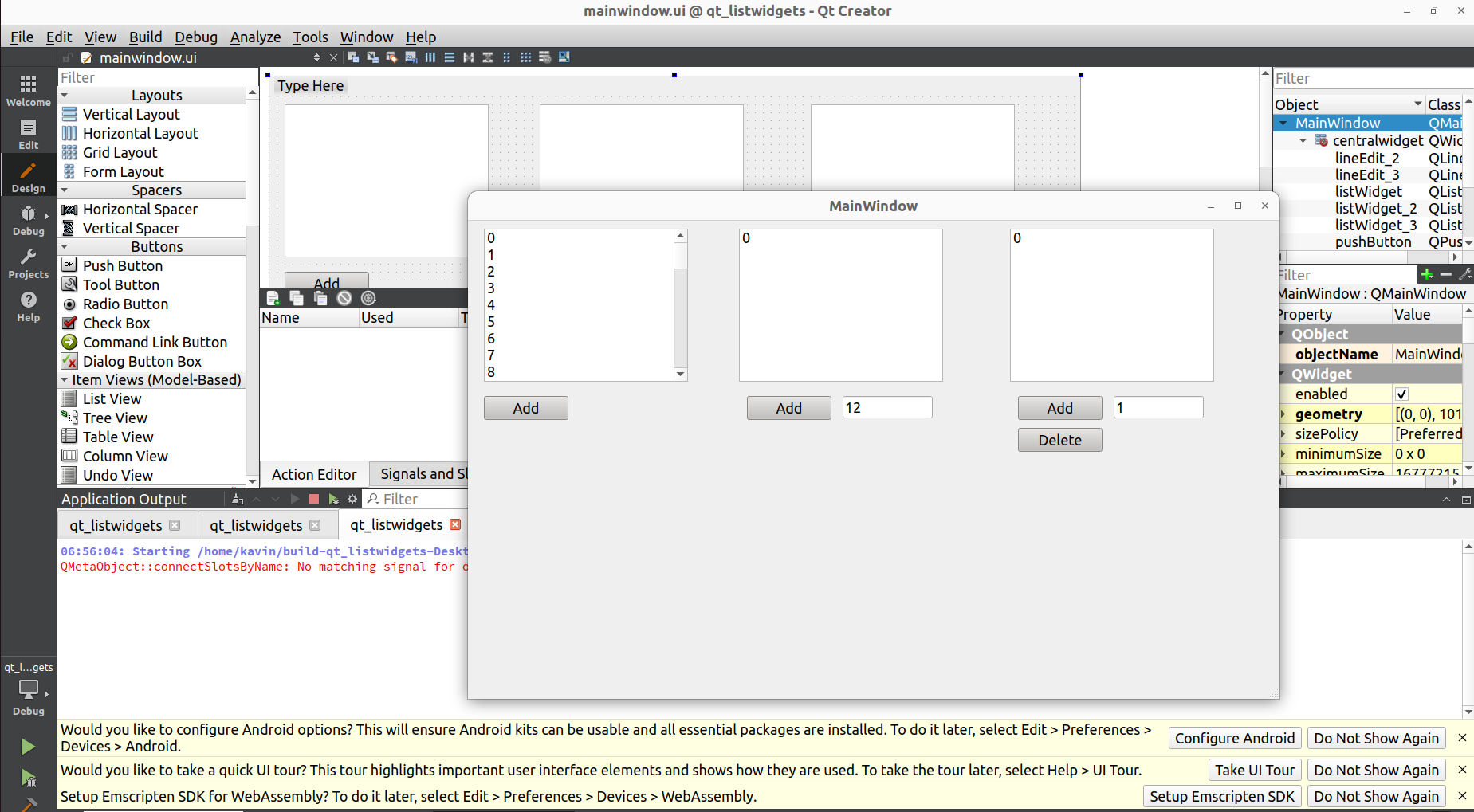
void MainWindow::on\_pushButton\_3\_clicked()

{

ui->listWidget\_3->clear();

}

Output:



6. Create a combo box,list widget and three push buttons.While insert the

values in the combo box it should be displayed in the list widget and also

perform delete ,clear operations.

Code:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

QString s;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_pushButton\_clicked()

{

s = ui->comboBox->currentText();

ui->listWidget->addItem(s);

}

void MainWindow::on\_pushButton\_2\_clicked()

{

qDeleteAll(ui->listWidget->selectedItems());

}

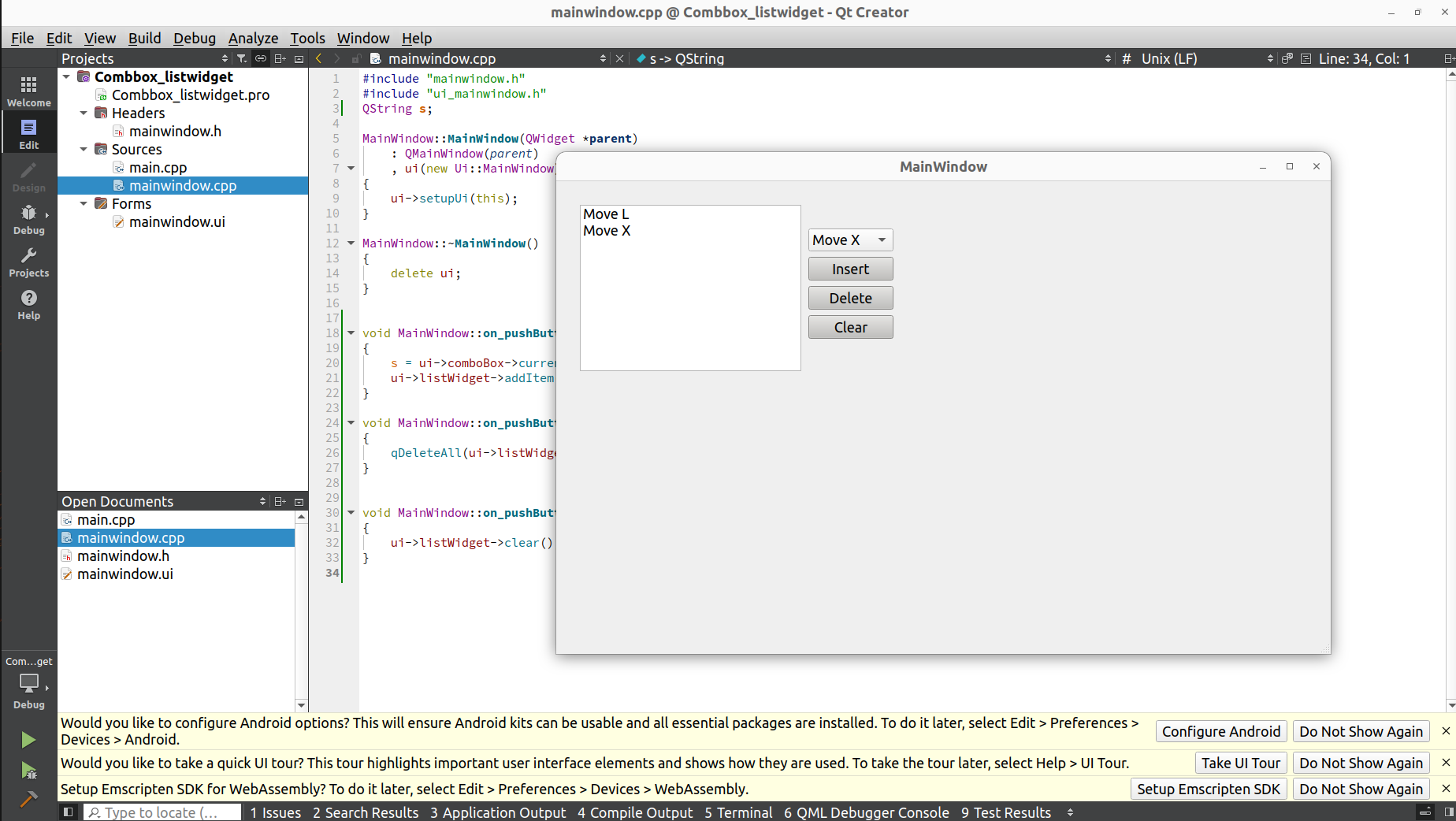
void MainWindow::on\_pushButton\_3\_clicked()

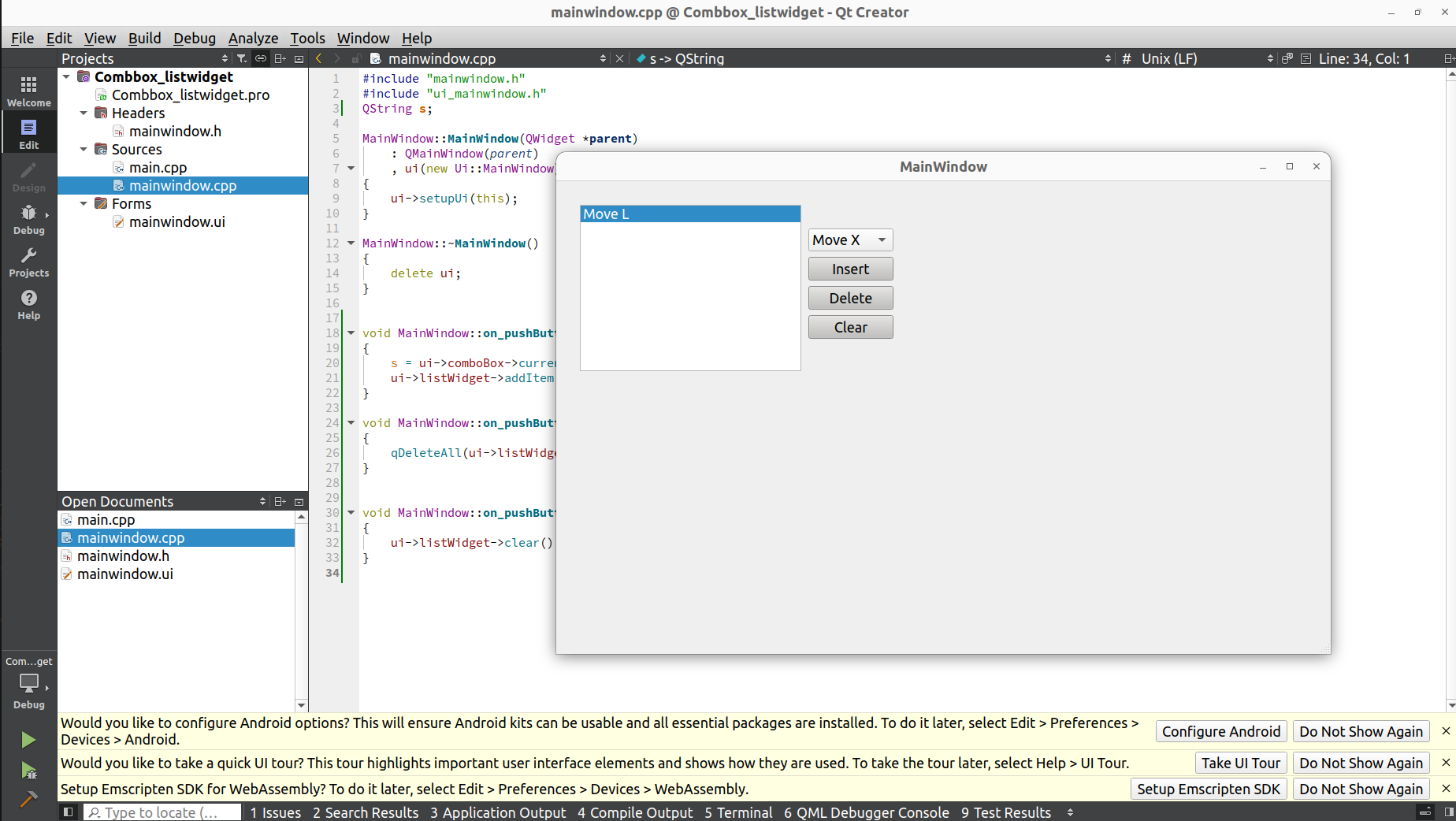
{

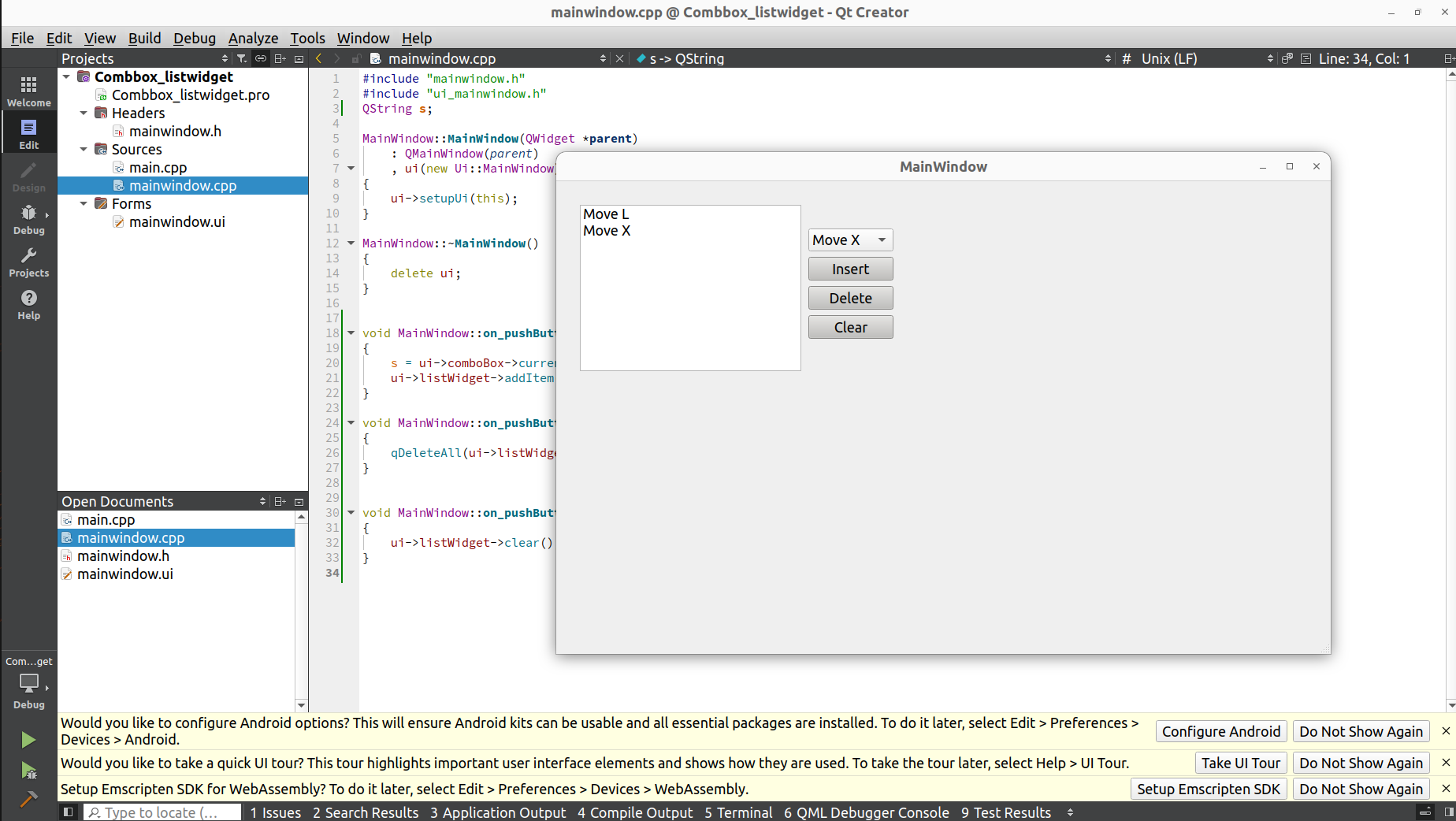
ui->listWidget->clear();

}

Output:







7. Create a Qt application that converts temperatures from Celsius to

Fahrenheit.

1) Design a GUI with the following components:

● QLabel displaying "Enter Temperature in Celsius:"

● QLineEdit for the user to input the temperature in

Celsius.

● QPushButton labeled "Convert."

● Another QLabel that will display the result in Fahrenheit.

2) Implement the logic for temperature conversion:

● When the user clicks the "Convert" button, the program

should read the value entered in the QLineEdit.

● Convert the temperature from Celsius to Fahrenheit using

the formula: Fahrenheit = (Celsius \* 9/5) + 32.

● Display the result in the second QLabel.

Code:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

float number1, number2;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_lineEdit\_textChanged(const QString &arg1)

{

number1 = arg1.toFloat();

}

void MainWindow::on\_pushButton\_clicked()

{

number2 = (number1\*9/5)+32;

ui->label\_3->setText(QString::number(number2));

}

void MainWindow::on\_pushButton\_2\_clicked()

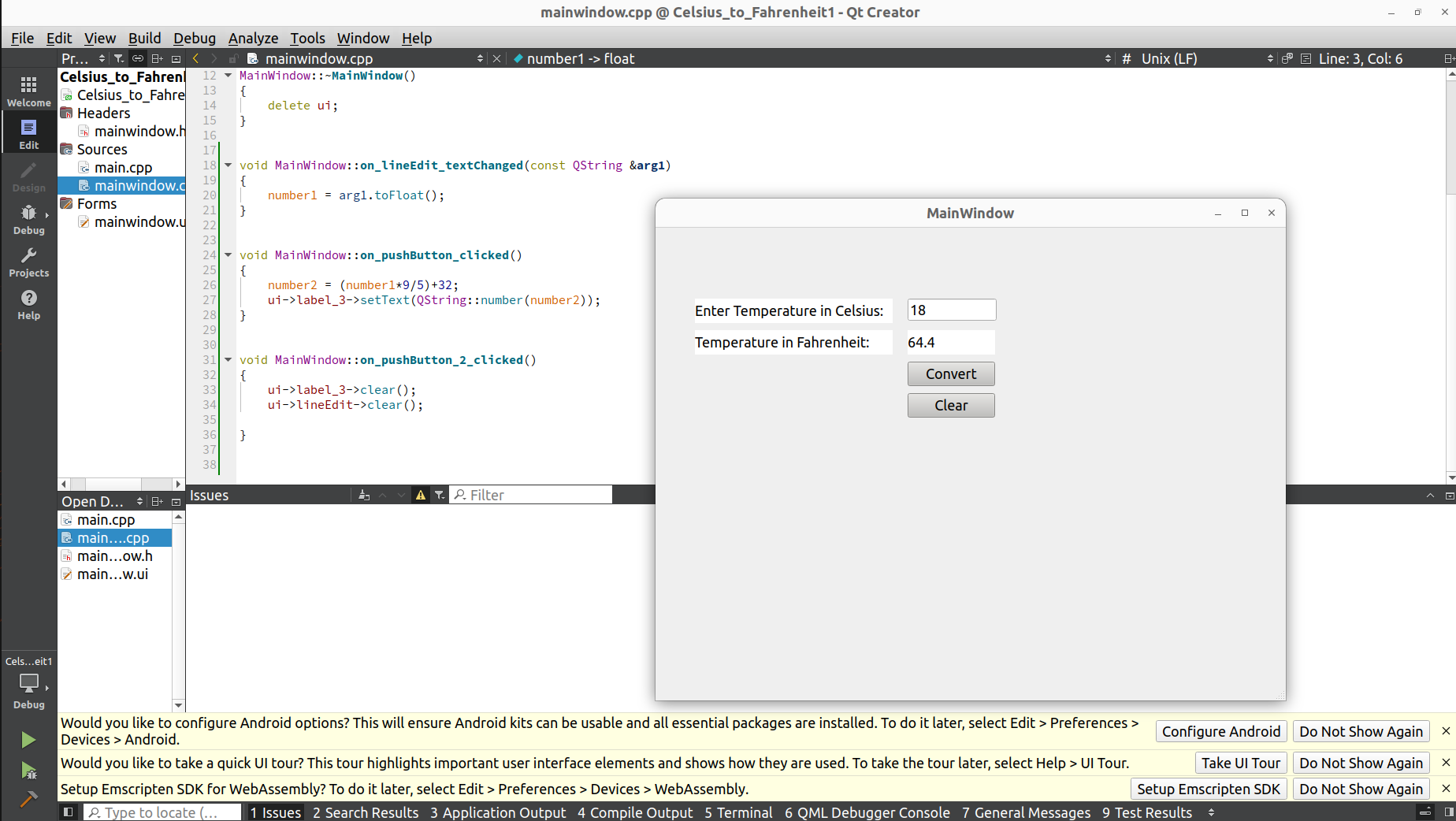
{

ui->label\_3->clear();

ui->lineEdit->clear();

}

Output:



8. Design a password strength checker application. Include a QLabel to

display a password input field using QLineEdit and a QPushButton to check

the strength. Implement a function that analyzes the password's strength

(e.g., length, complexity) and displays the result in the QLabel.

Code:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

QString password;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_lineEdit\_textChanged(const QString &arg1)

{

password = ui->lineEdit->text();

}

void MainWindow::on\_pushButton\_clicked()

{

int length= password.length();

QString result;

if(length<6){

result = "Weak(Too Short)";

}else if (length<10){

result = "Moderate";

}else{

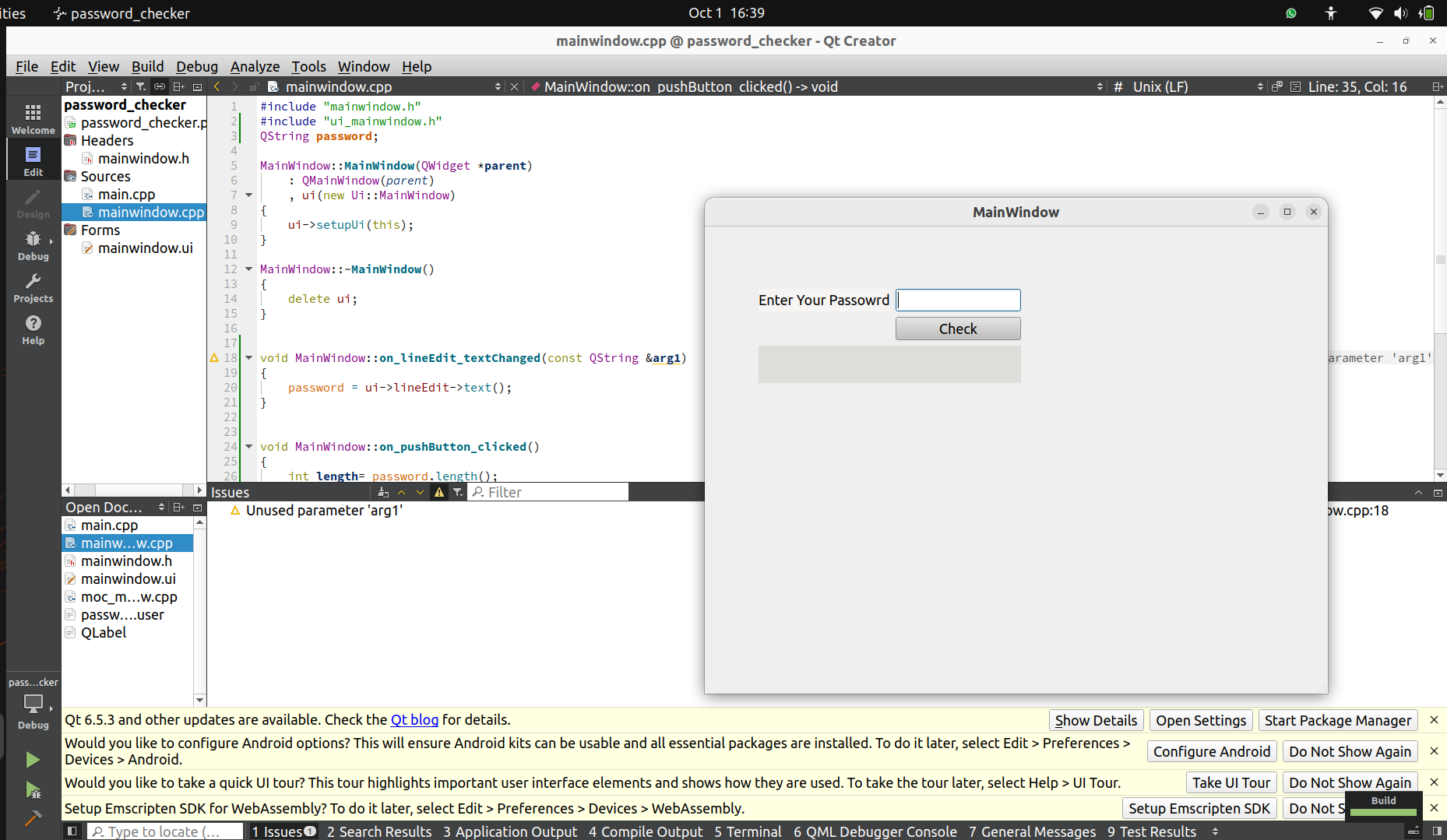
result = "Strong";

}

ui->label\_2->setText(result);

}

Output:



9. Create a word counter application with a QLabel to display a text input field

using QLineEdit and a QPushButton to count words. When the user clicks

the "Count Words" button, count and display the number of words in the

input text in the QLabel.

Code:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

QString text;

QStringList words;

int count;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_lineEdit\_textChanged(const QString &arg1)

{

text=ui->lineEdit->text();

}

void MainWindow::on\_pushButton\_clicked()

{

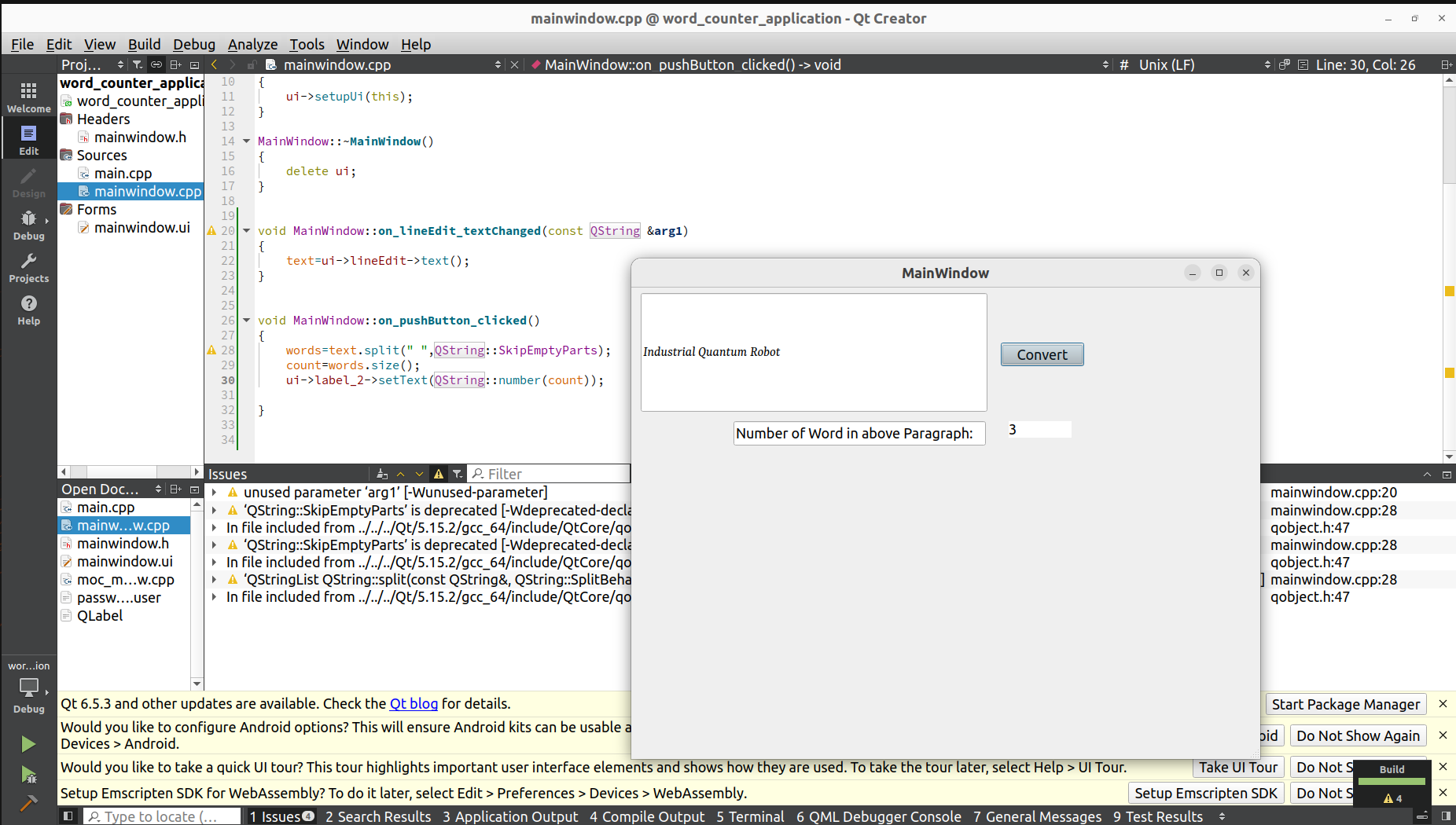
words=text.split(" ",QString::SkipEmptyParts);

count=words.size();

ui->label\_2->setText(QString::number(count));

}

Output:



10.Develop a temperature converter application with QLabel components to

display labels, QLineEdit for input, and QPushButton to perform the

conversion. Implement the logic to convert temperatures between Celsius

and Fahrenheit when the user clicks the appropriate button.

Code:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

float temp, celcius, fahrenheit;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_lineEdit\_3\_textChanged(const QString &arg1)

{

temp = arg1.toFloat();

}

void MainWindow::on\_pushButton\_clicked()

{

celcius = (temp -32)\*5/9;

ui->label\_4->setText(QString::number(celcius));

}

void MainWindow::on\_pushButton\_2\_clicked()

{

fahrenheit = (temp\*9/5) + 32;

ui->label\_5->setText(QString::number(fahrenheit));

}

void MainWindow::on\_pushButton\_3\_clicked()

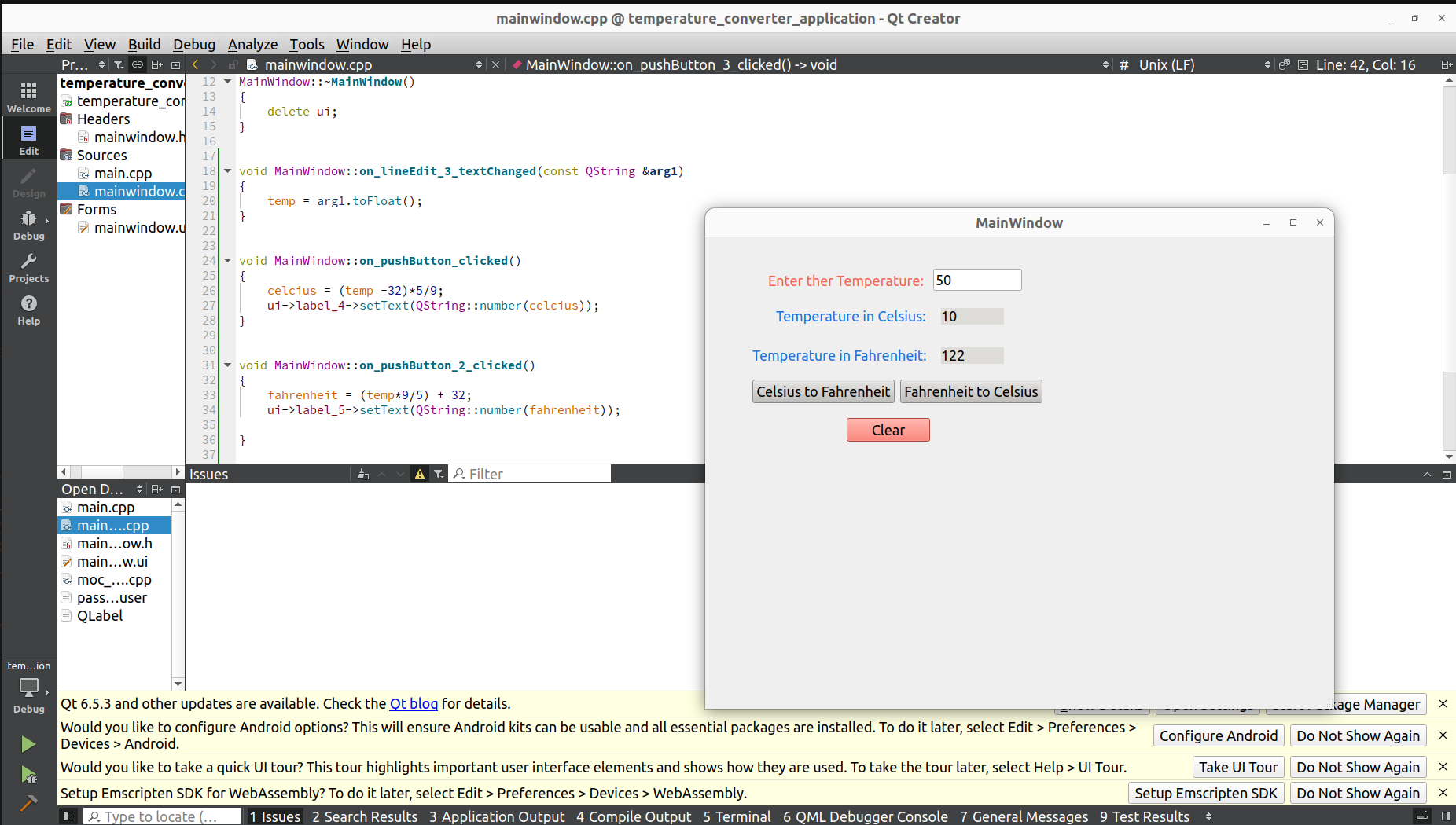
{

ui->label\_4->clear();

ui->label\_5->clear();

}

Output:



11.Design a grocery shopping list application using a QListWidget. Allow users

to add grocery items (e.g., "Milk," "Bread") with a quantity field

(QLineEdit) and an "Add" button. Include a "Delete" button to remove

selected items from the list.

Code:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

QString items;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_lineEdit\_textChanged(const QString &arg1)

{

items = arg1;

}

void MainWindow::on\_pushButton\_3\_clicked()

{

ui->listWidget->addItem(items);

ui->lineEdit->clear();

}

void MainWindow::on\_pushButton\_2\_clicked()

{

qDeleteAll(ui->listWidget->selectedItems());

}

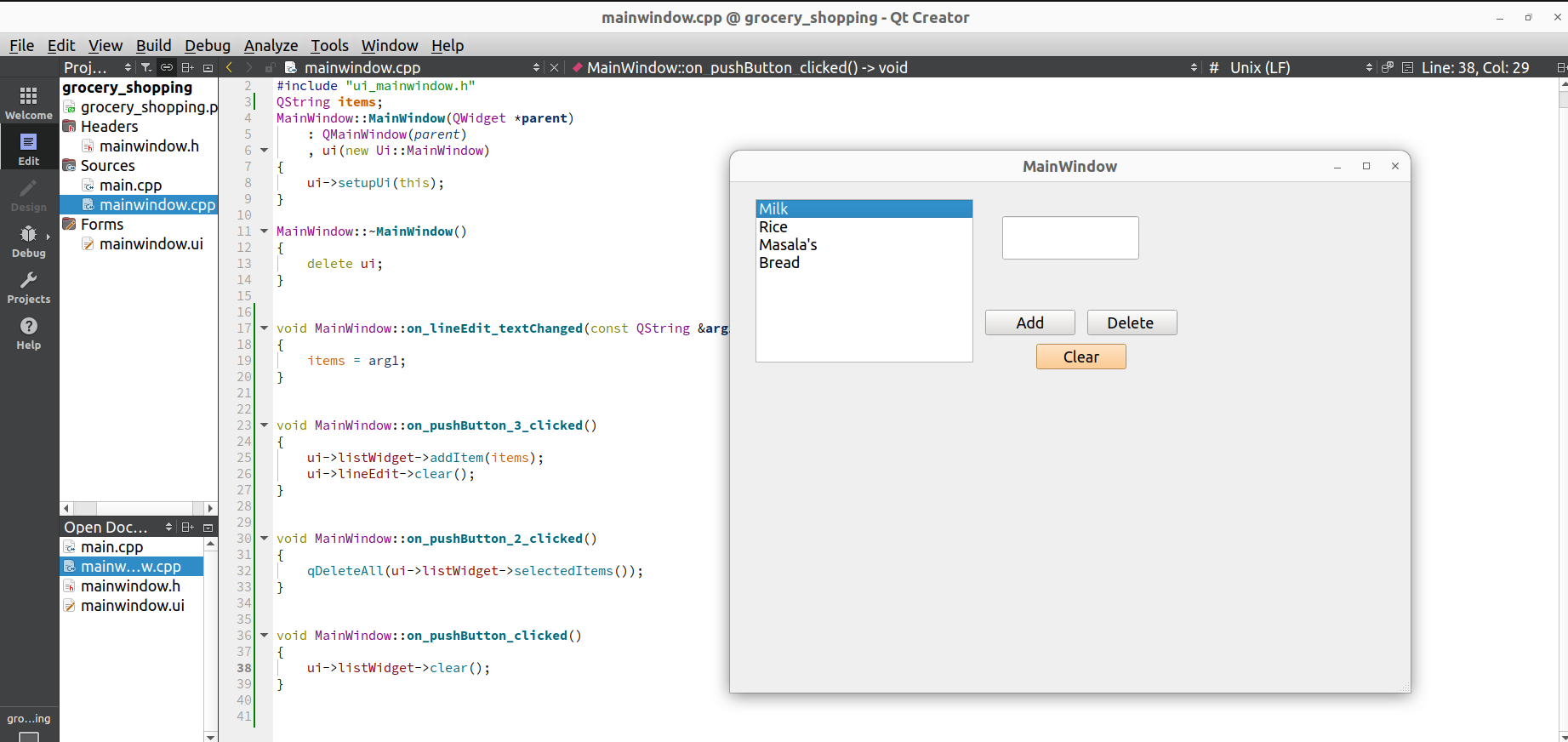
void MainWindow::on\_pushButton\_clicked()

{

ui->listWidget->clear();

}

Output:



12.Build a contact list application using a QListWidget. Allow users to enter

contact names and phone numbers. Use two QListWidgets side by side: one

for displaying contact names and another for displaying phone numbers.

Include buttons to add new contacts, delete selected contacts, and edit

existing contacts.

Input:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QInputDialog>

QString name, number;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_lineEdit\_textChanged(const QString &arg1)

{

name = arg1;

}

void MainWindow::on\_lineEdit\_2\_textChanged(const QString &arg1)

{

number = arg1;

}

void MainWindow::on\_pushButton\_clicked()

{

ui->listWidget->addItem(name);

ui->listWidget\_2->addItem(number);

}

void MainWindow::on\_pushButton\_2\_clicked()

{

qDeleteAll(ui->listWidget->selectedItems());

qDeleteAll(ui->listWidget\_2->selectedItems());

}

void MainWindow::on\_pushButton\_3\_clicked()

{

QListWidgetItem\* selectedItem1 = ui->listWidget->currentItem();

if (selectedItem1) {

QString currentName = selectedItem1->text();

QString newName = QInputDialog::getText(this, "Edit Name", "Edit name:", QLineEdit::Normal, currentName);

if (!newName.isEmpty() && newName != currentName) {

selectedItem1->setText(newName);

}

}

QListWidgetItem\* selectedItem2 = ui->listWidget\_2->currentItem();

if (selectedItem2) {

QString currentName = selectedItem2->text();

QString newName = QInputDialog::getText(this, "Edit Name", "Edit name:", QLineEdit::Normal, currentName);

if (!newName.isEmpty() && newName != currentName) {

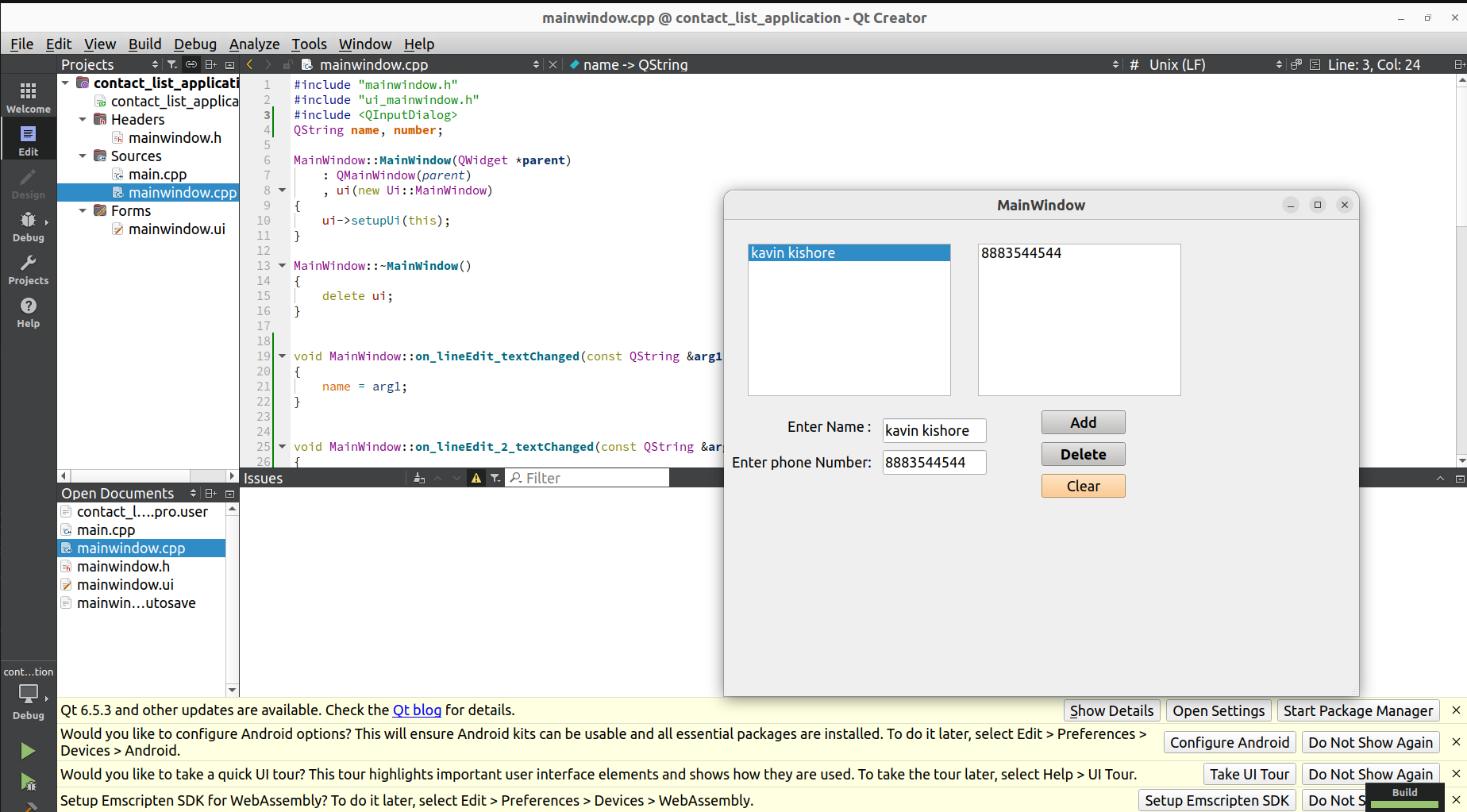
selectedItem2->setText(newName);

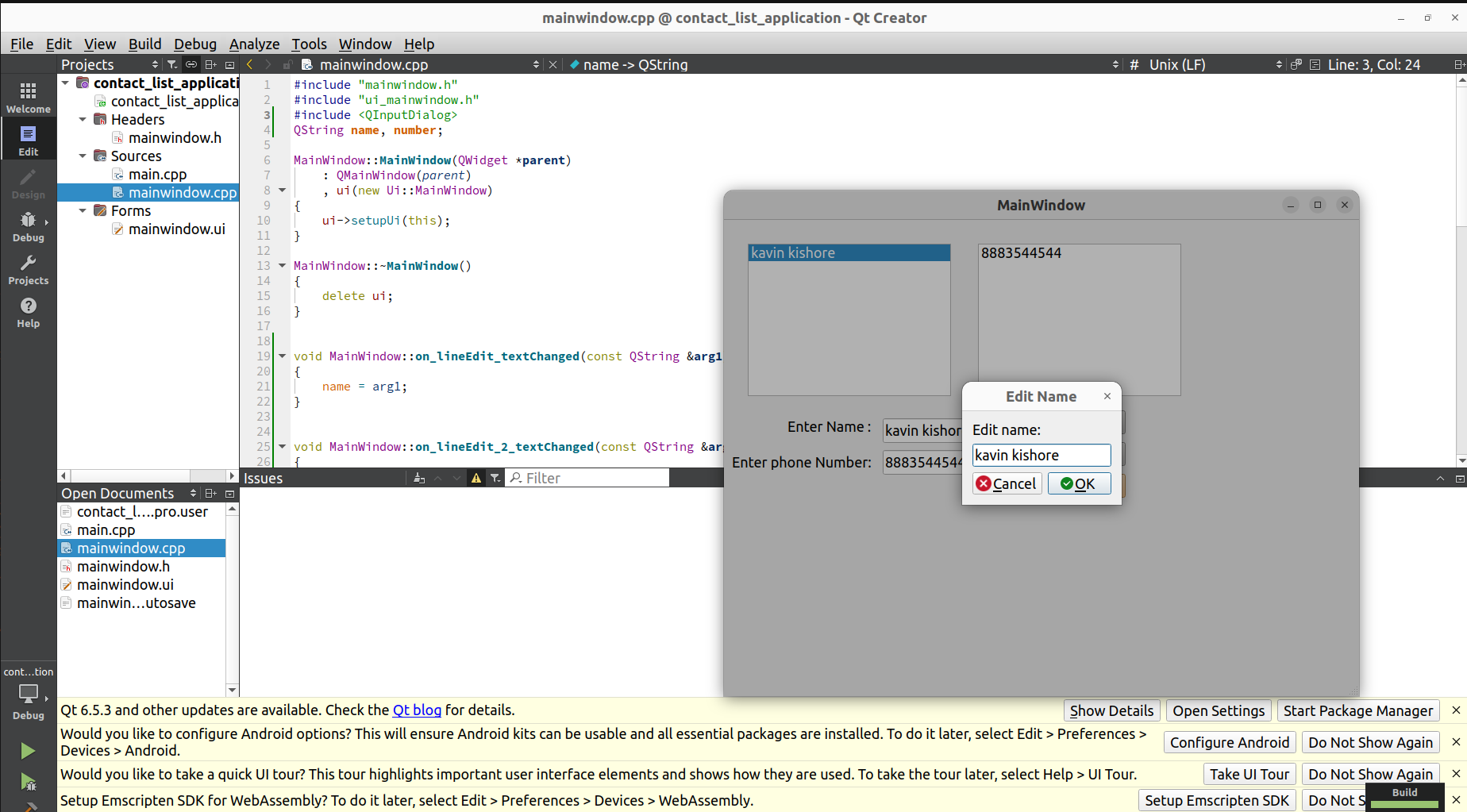
}

}

}

Output:





13. Create a basic to-do list application using a QListWidget. Design the UI with

an input field (QLineEdit), an "Add" button (QPushButton), and a "Remove"

button (QPushButton). When the user enters a task and clicks "Add," it

should add the task to the QListWidget. When a task is selected and the

"Remove" button is clicked, the selected task should be deleted.

Input:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

QString task;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_lineEdit\_textChanged(const QString &arg1)

{

task=arg1;

}

void MainWindow::on\_pushButton\_clicked()

{

ui->listWidget->addItem(task);

}

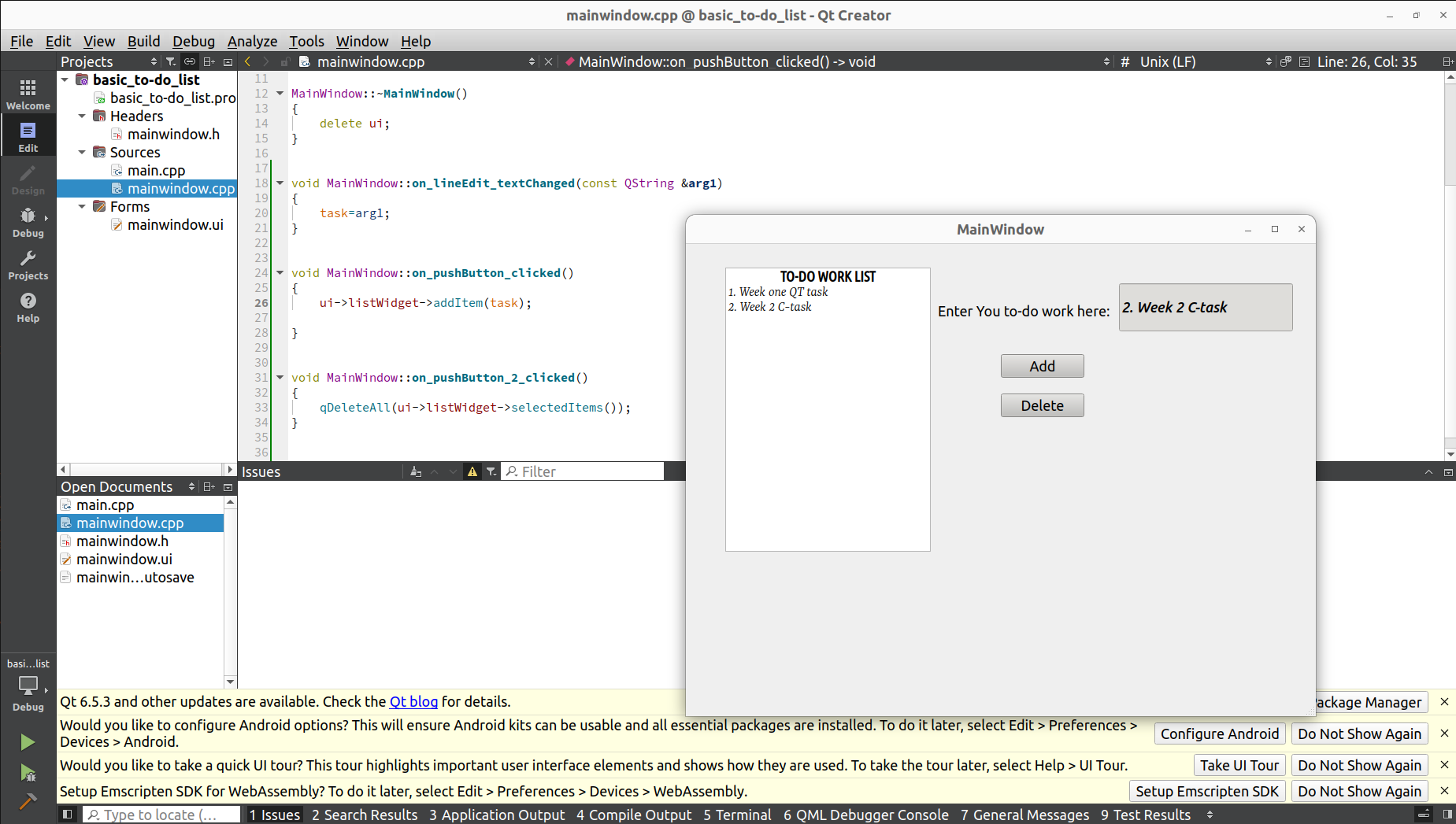
void MainWindow::on\_pushButton\_2\_clicked()

{

qDeleteAll(ui->listWidget->selectedItems());

}

OutPut:



14.Design an interactive quiz application with Qt. Create a user-friendly interface that displays multiple-choice questions and options using QLabel and QRadioButton. Allow users to select their answers, and display the correct answer after submission using QMessageBox. Implement a scoring system to keep track of users' scores.

**Coding:**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QMessageBox>

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

score=0;

connect(ui->pushButton, &QPushButton::clicked, this, &MainWindow::on\_pushButton\_2\_clicked);

QString question = "Which was the first video game produced by Nintendo?";

QString answer1 = "Super Mario Bros";

QString answer2 = "Luigi’s Mansion";

QString answer3 = "EVR Race";

QString answer4 = "Donkey Kong";

QString correctAnswer = "Donkey Kong";

ui->label\_3->setText(question);

ui->radioButton->setText(answer1);

ui->radioButton\_2->setText(answer2);

ui->radioButton\_3->setText(answer3);

ui->radioButton\_4->setText(answer4);

ui->lineEdit\_3->setText(correctAnswer);

ui->lineEdit\_3->setVisible(false);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_pushButton\_2\_clicked()

{

QString selectedAnswer;

if (ui->radioButton->isChecked()) {

selectedAnswer = ui->radioButton->text();

} else if (ui->radioButton\_2->isChecked()) {

selectedAnswer = ui->radioButton\_2->text();

} else if (ui->radioButton\_3->isChecked()) {

selectedAnswer = ui->radioButton\_3->text();

} else if (ui->radioButton\_4->isChecked()) {

selectedAnswer = ui->radioButton\_4->text();

}

QString correctAnswer = ui->lineEdit\_3->text();

if (selectedAnswer == correctAnswer) {

score++;

ui->lineEdit\_2->setText("Correct");

ui->lineEdit\_2->setStyleSheet("background-color:green;color:white;");

ui->lineEdit->setText(QString::number(score));

} else {

ui->lineEdit\_2->setText("Wrong");

ui->lineEdit\_2->setStyleSheet("background-color:red;color:white;");

ui->lineEdit->setText(QString::number(score));

}

ui->lineEdit\_3->setText(correctAnswer);

ui->lineEdit\_3->setVisible(true);

QMessageBox::information(this, "Result", "Your answer: " + selectedAnswer + "\nCorrect answer: " + correctAnswer + "\nYour score: " + QString::number(score));

ui->radioButton->setChecked(false);

ui->radioButton\_2->setChecked(false);

ui->radioButton\_3->setChecked(false);

ui->radioButton\_4->setChecked(false);

}

Output:

