A REPORT ON

**A SOCKET CLIENT FRAMEWORK FOR TESTING MONSTAR DAQ SERVER**

Submitted to: **Mr. AYAN BANERJEE**

**Computer Division**

**Vocational Training** at

**Variable Energy Cyclotron Centre, Kolkata**



From**: 18th June** To: **18th July**

Prepared By:

**Shalmoli Neogi-MCKVIE**

# **ACKNOWLEDGEMENT**

We would like to extend our heartfelt gratitude to my project guide, ***Mr. Ayan Banerjee***. Without his valuable support and guidance this project would not have got proper shape.

We are also grateful to our training coordinator ***Dr. P Y Nabhiraj*** for his guidance and support. Without him the training program would not have been possible.

# **CONTENTS**

## 1. Introduction

## 2. Methods

* Barycentric Interpolation method
* qHull method
* Newton Conjugate Gradient method
* ROSD
* Detection and location of radioactive substances using

WSN

* Particle filter

## 3. Conclusion

## 4. Reference

# **INTRODUCTION**

The project **A socket client framework for testing MONSTER DAQ server** has been made in **C++** language.

This project consists of the following parts:

* Server Port
* Server IP
* Connect
* Set buffer size
* Set data limit
* Send

Each part performs some defined function

# **OBJECTIVE**

Main objective of this project is:

To build a client for testing the multiclient server MONSTAR DAQ which will have the following properties:

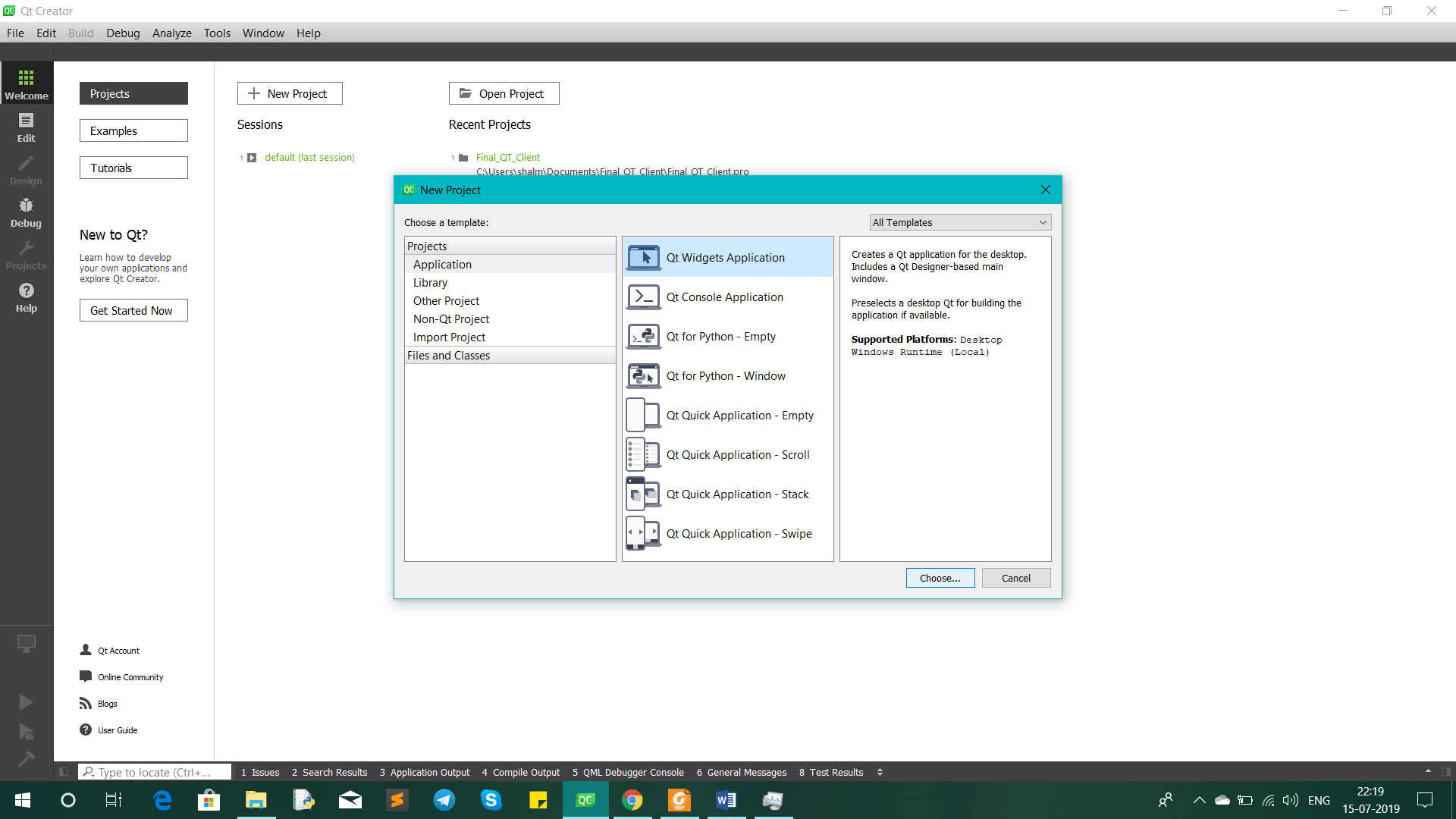
* It should write data as per size of user’s choice
* It should send data continuously until the limit specified by the user is reached
* Thread management needs to be implemented, i.e. one thread writes the data in the buffer, whereas the other one sends the buffer to the server.
* Writing and sending concurrency needs to be achieved.
* Locks needs to be implemented in order to prevent inconsistent results.

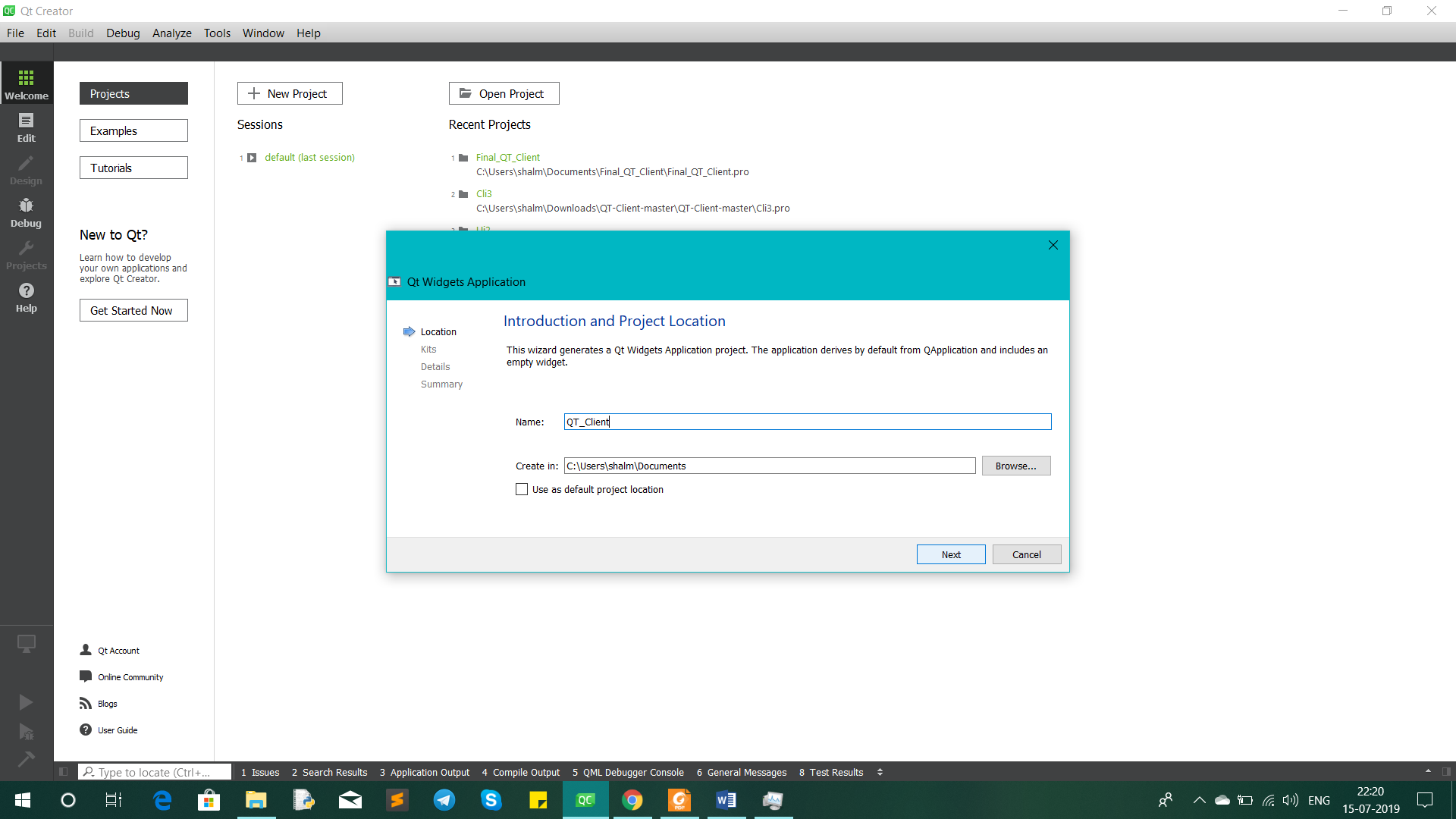
# **PREREQUICITES**

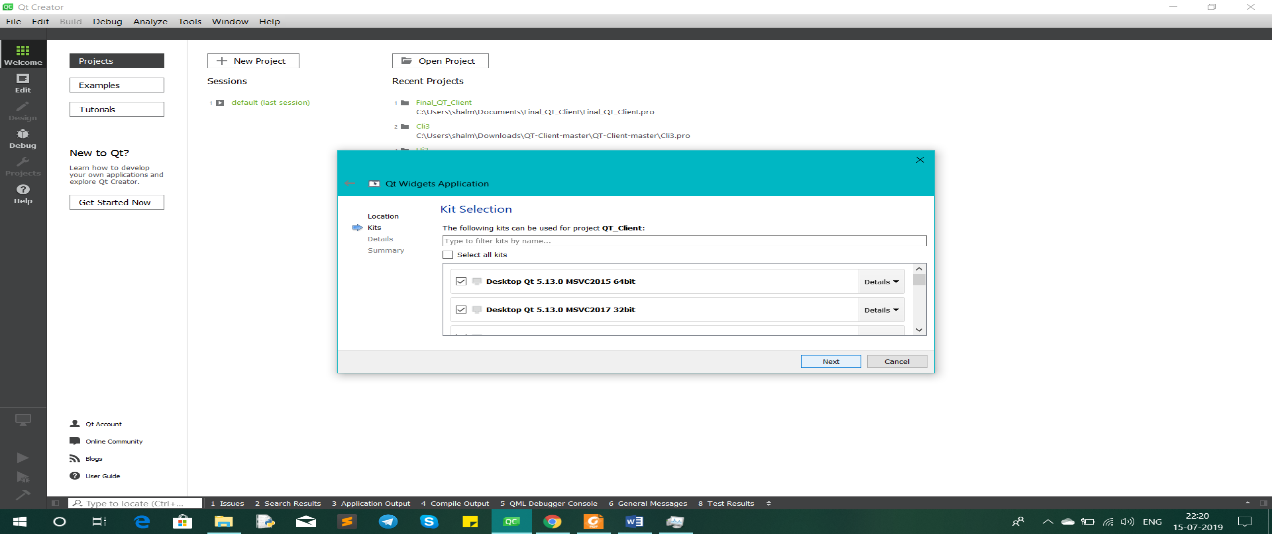
* Hardware used
* Software required:
* Qt Creator
* g++ compiler

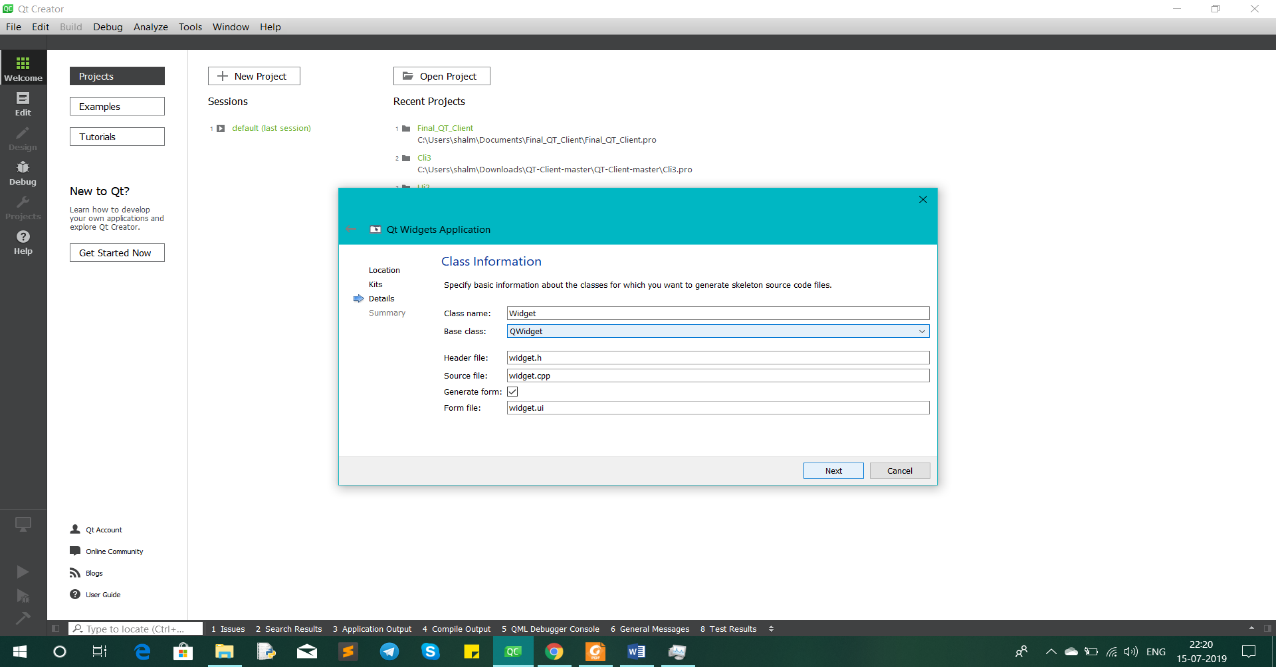
# **PROGRAM FLOWCHART**

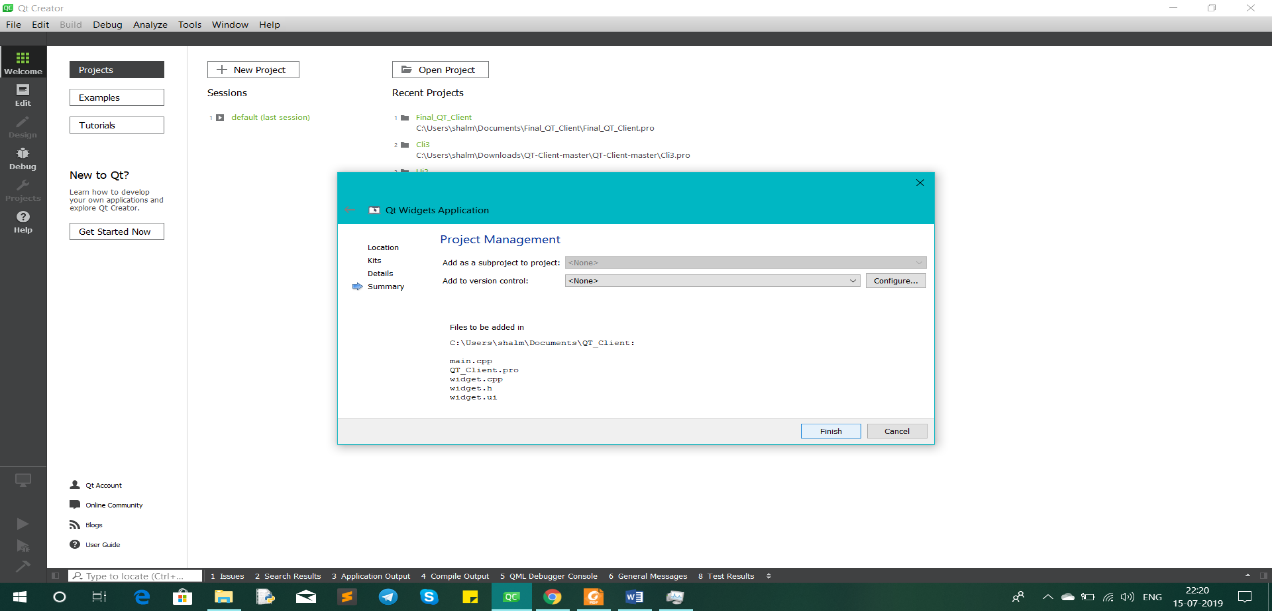
* **Step 1: Creating a project in QT**

****

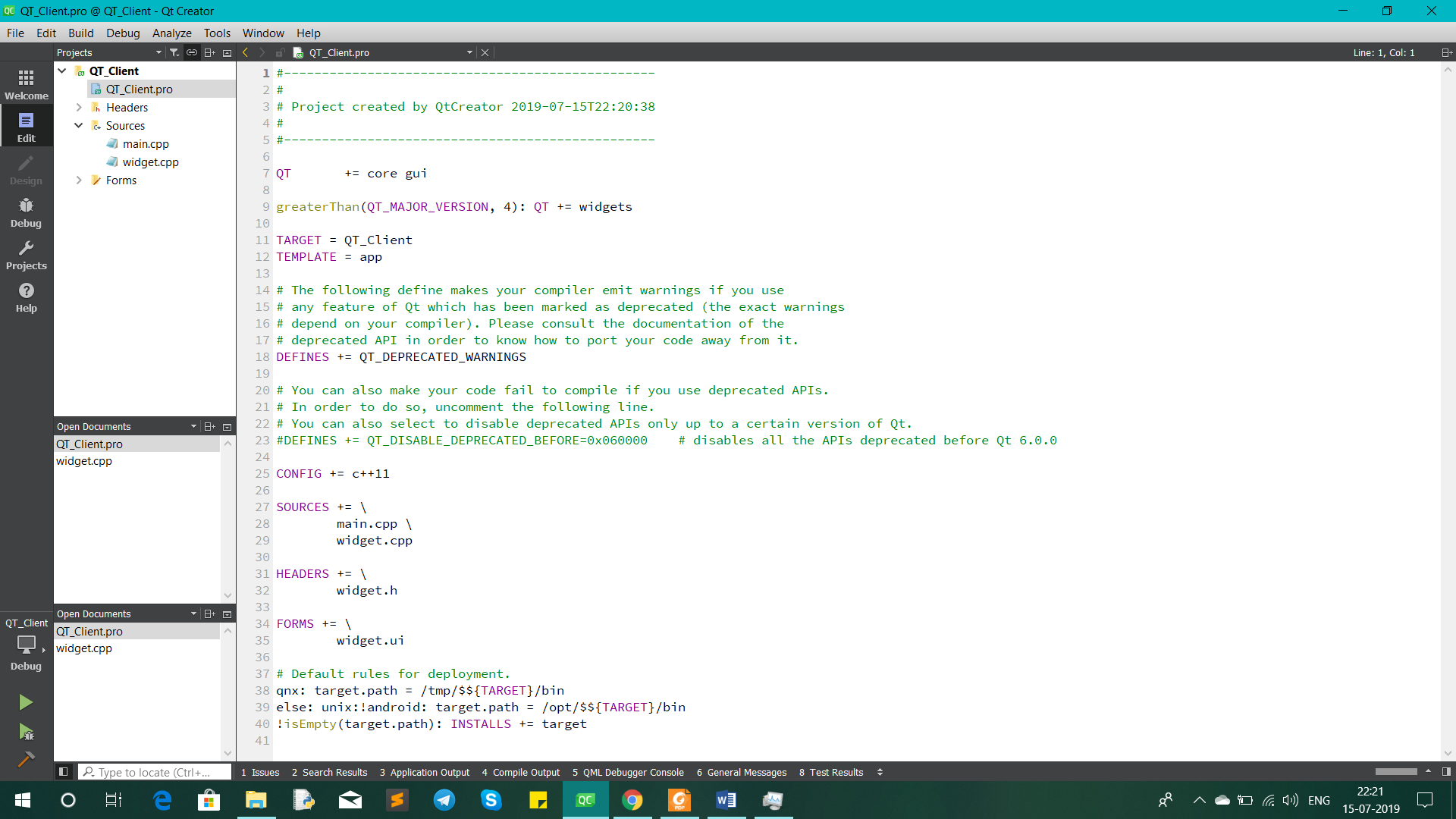
****

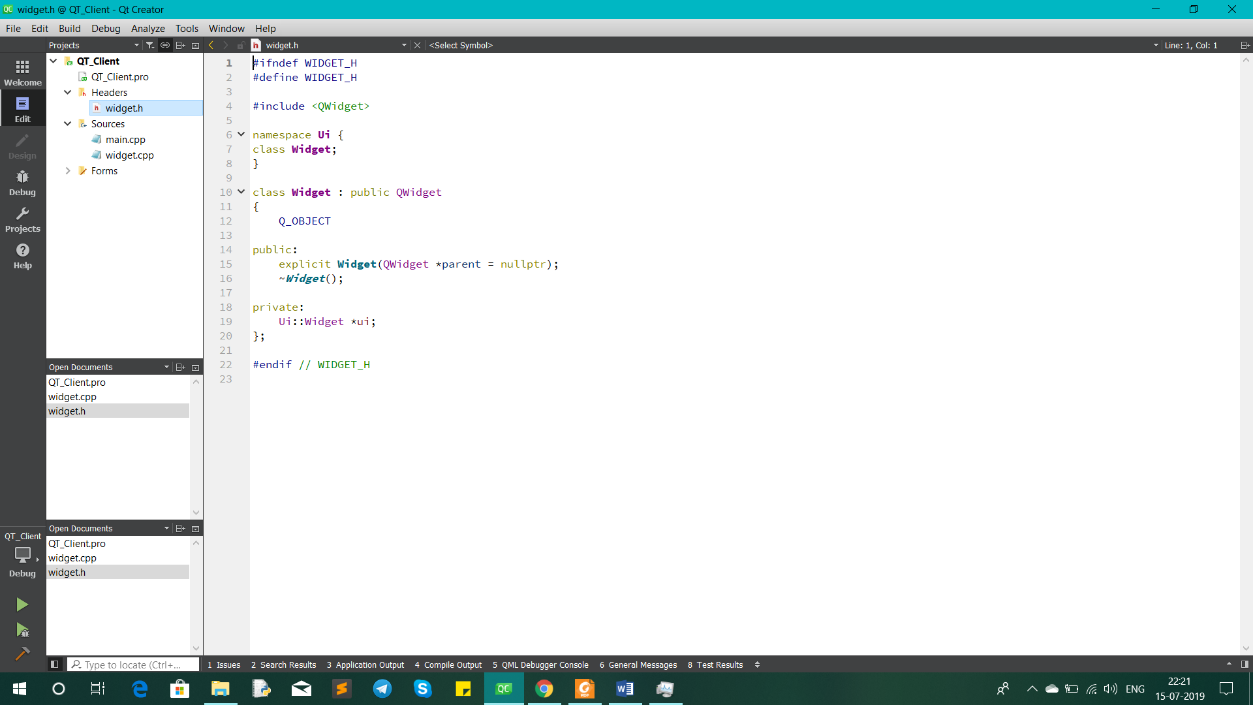
****

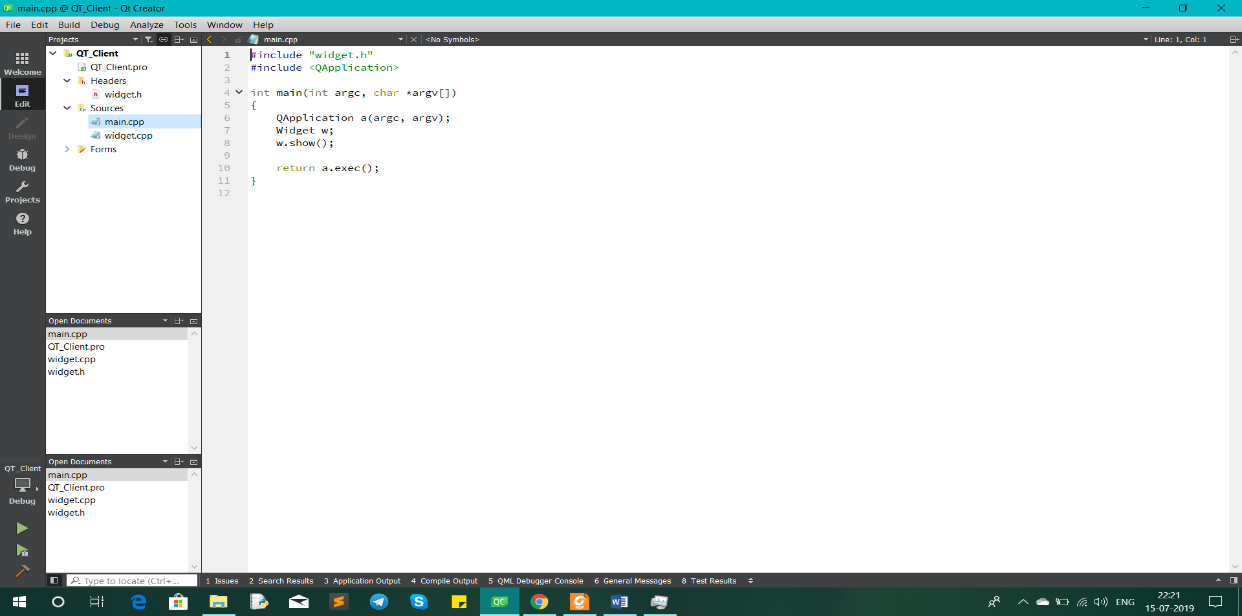
****

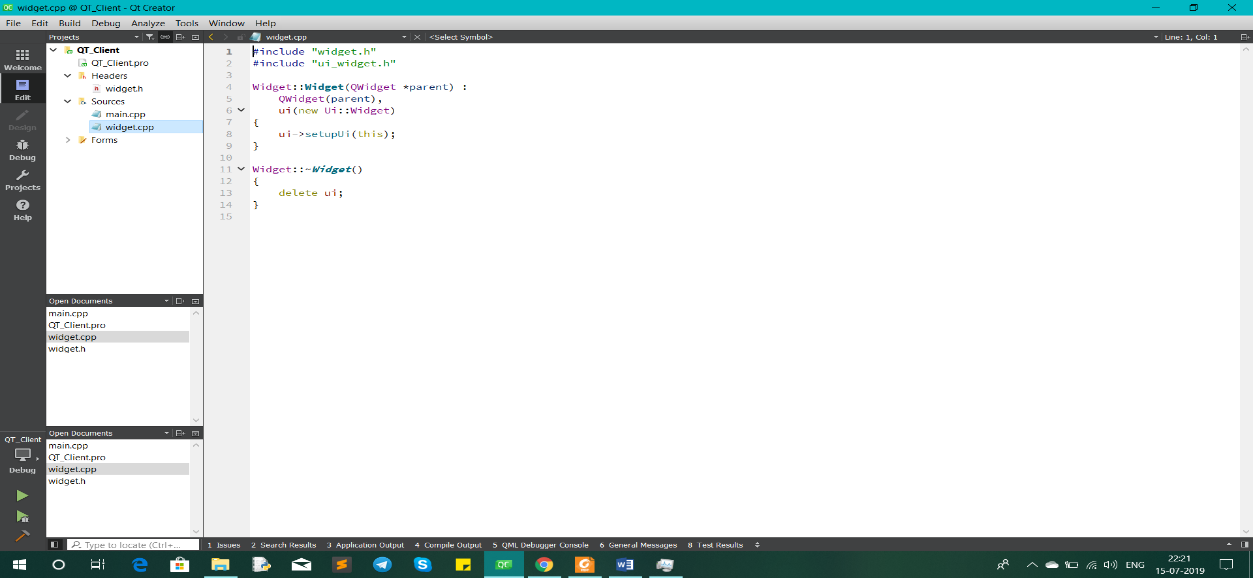
****

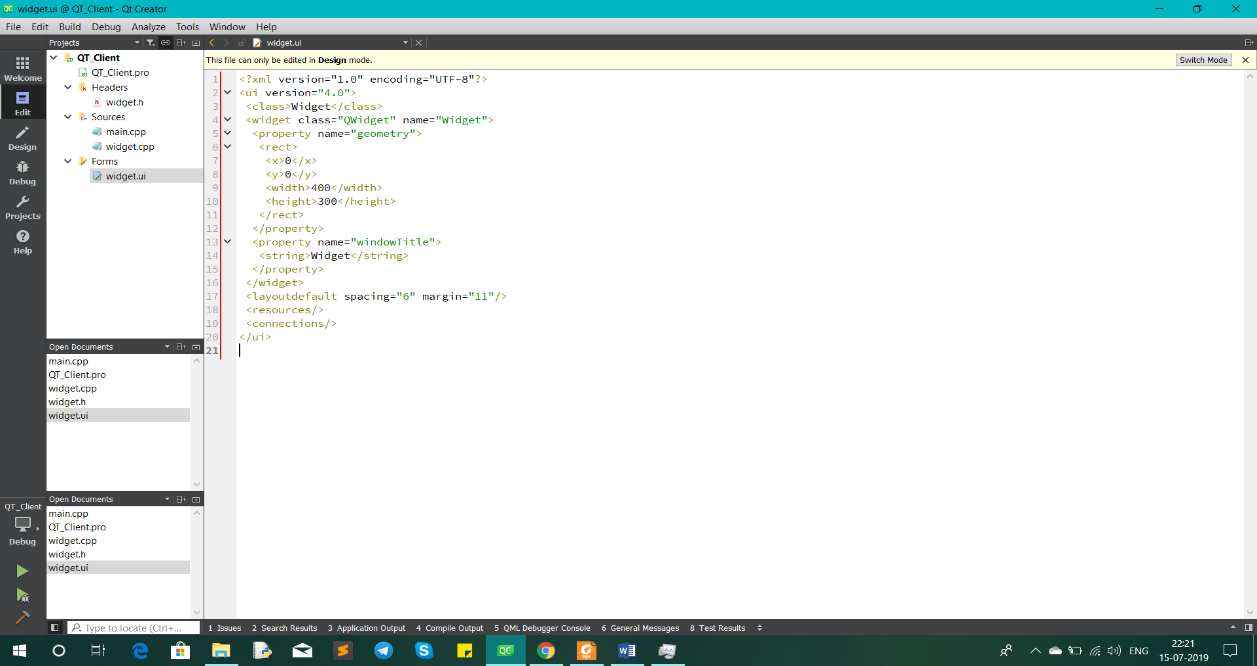
* **Step 2: Following 5 files are automatically created by QT**

****

****

****

****

****

* **Step 3: Several header files are added**

**1.Qwidget:**

The [QWidget](https://doc.qt.io/qt-5/qwidget.html) class is the base class of all user interface objects.

|  |  |
| --- | --- |
| Header: | #include <QWidget> |
| qmake: | QT += widgets |
| Inherits: | [QObject](https://doc.qt.io/qt-5/qobject.html) and [QPaintDevice](https://doc.qt.io/qt-5/qpaintdevice.html) |
| Inherited By: | [QAbstractButton](https://doc.qt.io/qt-5/qabstractbutton.html), [QAbstractSlider](https://doc.qt.io/qt-5/qabstractslider.html), [QAbstractSpinBox](https://doc.qt.io/qt-5/qabstractspinbox.html), [QCalendarWidget](https://doc.qt.io/qt-5/qcalendarwidget.html), [QComboBox](https://doc.qt.io/qt-5/qcombobox.html), [QDesktopWidget](https://doc.qt.io/qt-5/qdesktopwidget.html), [QDialog](https://doc.qt.io/qt-5/qdialog.html), [QDialogButtonBox](https://doc.qt.io/qt-5/qdialogbuttonbox.html), [QDockWidget](https://doc.qt.io/qt-5/qdockwidget.html), [QFocusFrame](https://doc.qt.io/qt-5/qfocusframe.html), [QFrame](https://doc.qt.io/qt-5/qframe.html), [QGroupBox](https://doc.qt.io/qt-5/qgroupbox.html), [QKeySequenceEdit](https://doc.qt.io/qt-5/qkeysequenceedit.html), [QLineEdit](https://doc.qt.io/qt-5/qlineedit.html), [QMacCocoaViewContainer](https://doc.qt.io/qt-5/qmaccocoaviewcontainer.html), [QMacNativeWidget](https://doc.qt.io/qt-5/qmacnativewidget.html), [QMainWindow](https://doc.qt.io/qt-5/qmainwindow.html), [QMdiSubWindow](https://doc.qt.io/qt-5/qmdisubwindow.html), [QMenu](https://doc.qt.io/qt-5/qmenu.html), [QMenuBar](https://doc.qt.io/qt-5/qmenubar.html), [QOpenGLWidget](https://doc.qt.io/qt-5/qopenglwidget.html), [QProgressBar](https://doc.qt.io/qt-5/qprogressbar.html), [QRubberBand](https://doc.qt.io/qt-5/qrubberband.html), [QSizeGrip](https://doc.qt.io/qt-5/qsizegrip.html), [QSplashScreen](https://doc.qt.io/qt-5/qsplashscreen.html), [QSplitterHandle](https://doc.qt.io/qt-5/qsplitterhandle.html), [QStatusBar](https://doc.qt.io/qt-5/qstatusbar.html), [QTabBar](https://doc.qt.io/qt-5/qtabbar.html), [QTabWidget](https://doc.qt.io/qt-5/qtabwidget.html), [QToolBar](https://doc.qt.io/qt-5/qtoolbar.html), and [QWizardPage](https://doc.qt.io/qt-5/qwizardpage.html) |

**2.QAbstractSocket:**

The [QAbstractSocket](https://doc.qt.io/qt-5/qabstractsocket.html) class provides the base functionality common to all

Socket types.

|  |  |
| --- | --- |
| Header: | #include <QAbstractSocket> |
| qmake: | QT += network |
| Inherits: | [QIODevice](https://doc.qt.io/qt-5/qiodevice.html) |
| Inherited By: | [QTcpSocket](https://doc.qt.io/qt-5/qtcpsocket.html) and [QUdpSocket](https://doc.qt.io/qt-5/qudpsocket.html) |

|  |  |
| --- | --- |
| Header: | #include <QTcpSocket> |
| qmake: | QT += network |
| Inherits: | [QAbstractSocket](https://doc.qt.io/qt-5/qabstractsocket.html) |
| Inherited By: | [QSctpSocket](https://doc.qt.io/qt-5/qsctpsocket.html) and [QSslSocket](https://doc.qt.io/qt-5/qsslsocket.html) |

**3.QTcpSocket:**

The [QTcpSocket](https://doc.qt.io/qt-5/qtcpsocket.html) class provides a TCP socket.

**4.QByteArray:**

The [QByteArray](https://doc.qt.io/qt-5/qbytearray.html) class provides an array of bytes.

|  |  |
| --- | --- |
| Header: | #include <QByteArray> |
| qmake: | QT += core |

**5.QMutex:**

The [QMutex](https://doc.qt.io/qt-5/qmutex.html) class provides access serialization between threads.

|  |  |
| --- | --- |
| Header: | #include <QMutex> |
| qmake: | QT += core |
| Inherits: | [QBasicMutex](https://doc.qt.io/qt-5/qbasicmutex.html) |

**6.QThread:**

The [QThread](https://doc.qt.io/qt-5/qthread.html) class provides a platform-independent way to manage threads.

|  |  |
| --- | --- |
| Header: | #include <QThread> |
| qmake: | QT += core |
| Inherits: | [QObject](https://doc.qt.io/qt-5/qobject.html) |

**7.QTextStream:**

The [QTextStream](https://doc-snapshots.qt.io/qt5-5.12/qtextstream.html) class provides a convenient interface for reading and writing

text..

|  |  |
| --- | --- |
| Header: | #include <QTextStream> |
| qmake: | QT += core |

**8.QString:**

The [QString](https://doc.qt.io/qt-5/qstring.html) class provides a Unicode character string.

|  |  |
| --- | --- |
| Header: | #include <QString> |
| qmake: | QT += core |

**9.QDebug:**

The [QDebug](https://doc.qt.io/qt-5/qdebug.html) class provides an output stream for debugging information.

|  |  |
| --- | --- |
| Header: | #include <QDebug> |
| qmake: | QT += core |

**10.QHostAddress:**

The [QHostAddress](https://doc.qt.io/qt-5/qhostaddress.html) class provides an IP address.

|  |  |
| --- | --- |
| Header: | #include <QHostAddress> |
| qmake: | QT += network |

**11.QCoreApplication:**

The [QCoreApplication](https://doc.qt.io/qt-5/qcoreapplication.html) class provides an event loop for Qt applications without UI.

|  |  |
| --- | --- |
| Header: | #include <QCoreApplication> |
| qmake: | QT += core |
| Inherits: | [QObject](https://doc.qt.io/qt-5/qobject.html) |

**12.QFile:**

The [QFile](https://doc.qt.io/qt-5/qfile.html) class provides an interface for reading from and writing to files.

|  |  |
| --- | --- |
| Header: | #include <QFile> |
| qmake: | QT += core |
| Inherits: | [QFileDevice](https://doc.qt.io/qt-5/qfiledevice.html) |
| Inherited By: | [QTemporaryFile](https://doc.qt.io/qt-5/qtemporaryfile.html) |

# **BIBLIOGRAPHY**

* <https://www.qt.io/>
* <https://riptutorial.com/qt/example/29873/tcp-client>