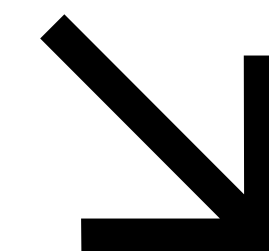


Python project: Artificial Intelligence



F.R.I.D.A.Y.

F.R.I.D.A.Y. (Female Replacement Intelligent Digital Assistant Youth) is a fictional artificial intelligence appearing in comic books published by Marvel Comics, usually depicted as the personal assistant and ally of the superhero Iron Man (Tony Stark).

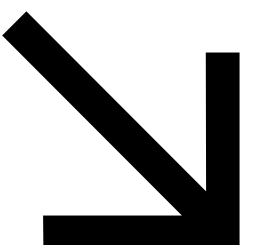


pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline, and is compatible with both Python 2 and 3.

```
engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[1].id)
engine.setProperty('rate', 150)
```

```
|
```

```
def speak(audio):
    engine.say(audio)
    engine.runAndWait()
```



SpeechRecognition is a library for performing speech recognition, with support for several engines and APIs, online and offline.

Speech recognition engine/API support:

- CMU Sphinx (works offline)
- Google Speech Recognition
- Google Cloud Speech API
- Wit.ai
- Microsoft Bing Voice Recognition
- Houndify API
- IBM Speech to Text

```
def takeCommand(self):  
    r = sr.Recognizer()  
    with sr.Microphone() as source:  
        print("I'm Listening...")  
        r.pause_threshold = 1  
        audio = r.listen(source)  
  
    try:  
        print("Recognizing...")  
        query = r.recognize_google(audio, language='en-in')  
        print(f"User said: {query}\n")  
    except Exception as e:  
        print("Repeat please...")  
        return "None"  
  
    return query
```

FRIDAY

Python project

Open

Programs

Tell

a joke

Shut

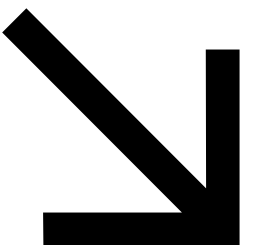
Down the computer

```
def TaskExecution(self):  
    wishMe()  
    while True:  
        self.query = self.takeCommand().lower()  
        if 'wikipedia' in self.query:  
            speak('Searching Wikipedia...')  
            self.query = self.query.replace("wikipedia", "")  
            results = wikipedia.summary(self.query, sentences=2)  
            speak("According to Wikipedia")  
            print(results)  
            speak(results)  
  
        elif 'open chrome' in self.query:  
            os.startfile('C:\Program Files\Google\Chrome\Application\chrome.exe')  
        elif 'close chrome' in self.query:  
            os.system("taskkill /f /im chrome.exe")  
  
        elif 'open new window' in self.query:  
            pyautogui.hotkey('ctrl', 'n')  
        elif 'open incognito window' in self.query:  
            pyautogui.hotkey('ctrl', 'shift', 'n')
```

Pyjokes

One line jokes for programmers

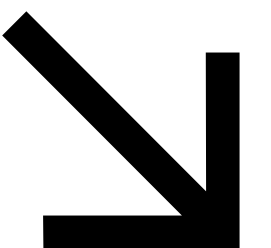
```
elif "tell me a joke" in self.query:  
    jokes = pyjokes.get_joke()  
    speak(jokes)
```



Shutting down the computer

FRIDAY

```
elif "shut down the system" in self.query:  
    os.system("shutdown /s /t 5")  
elif "restart the system" in self.query:  
    os.system("shutdown /r /t 5")  
elif "Lock the system" in self.query:  
    os.system("rundll32.exe powrprof.dll,SetSuspendState 0,1,0")
```

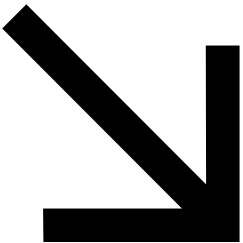


Creation of

GUI

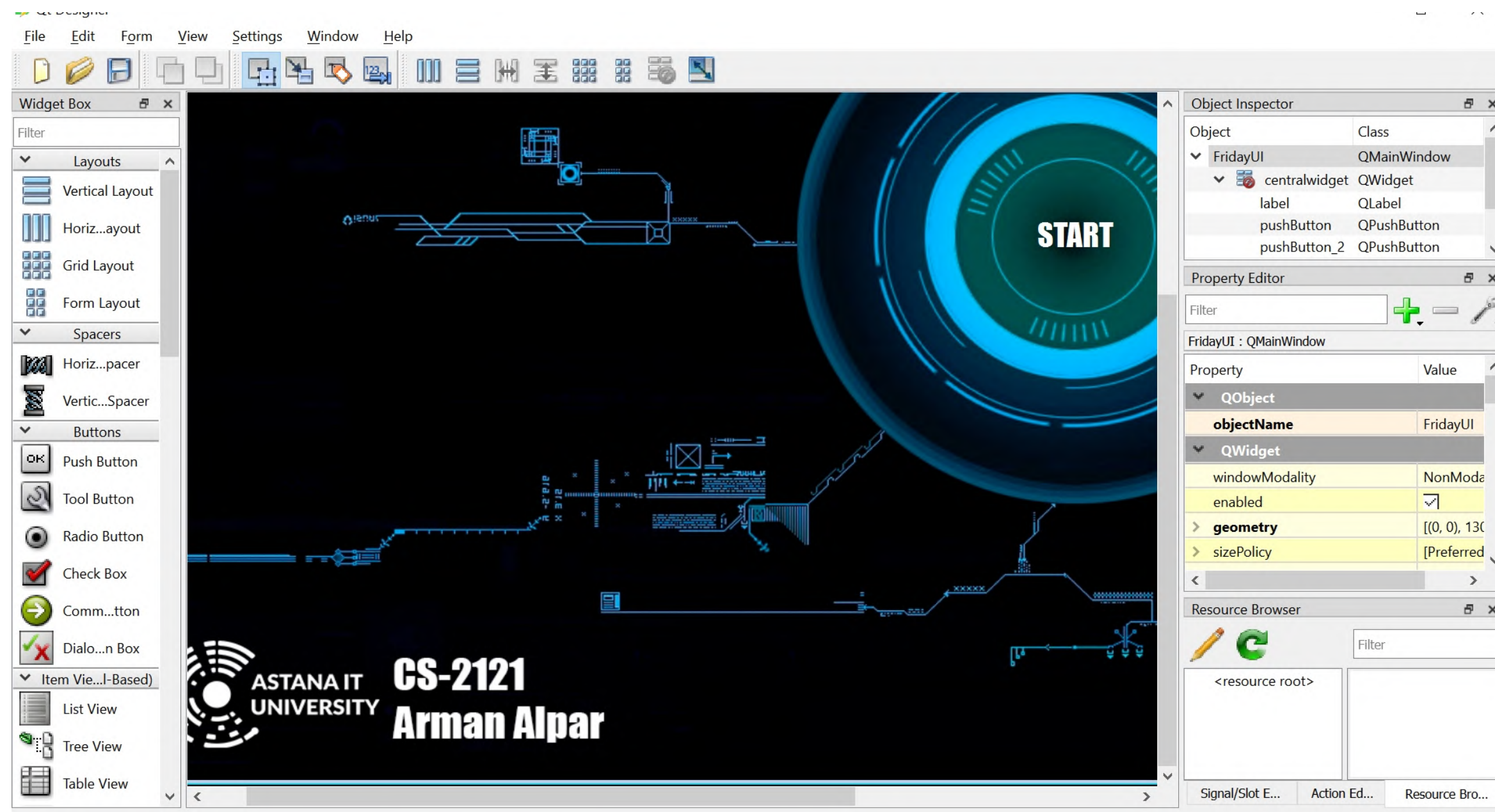
FRIDAY

Pitch



What is Qt Designer?

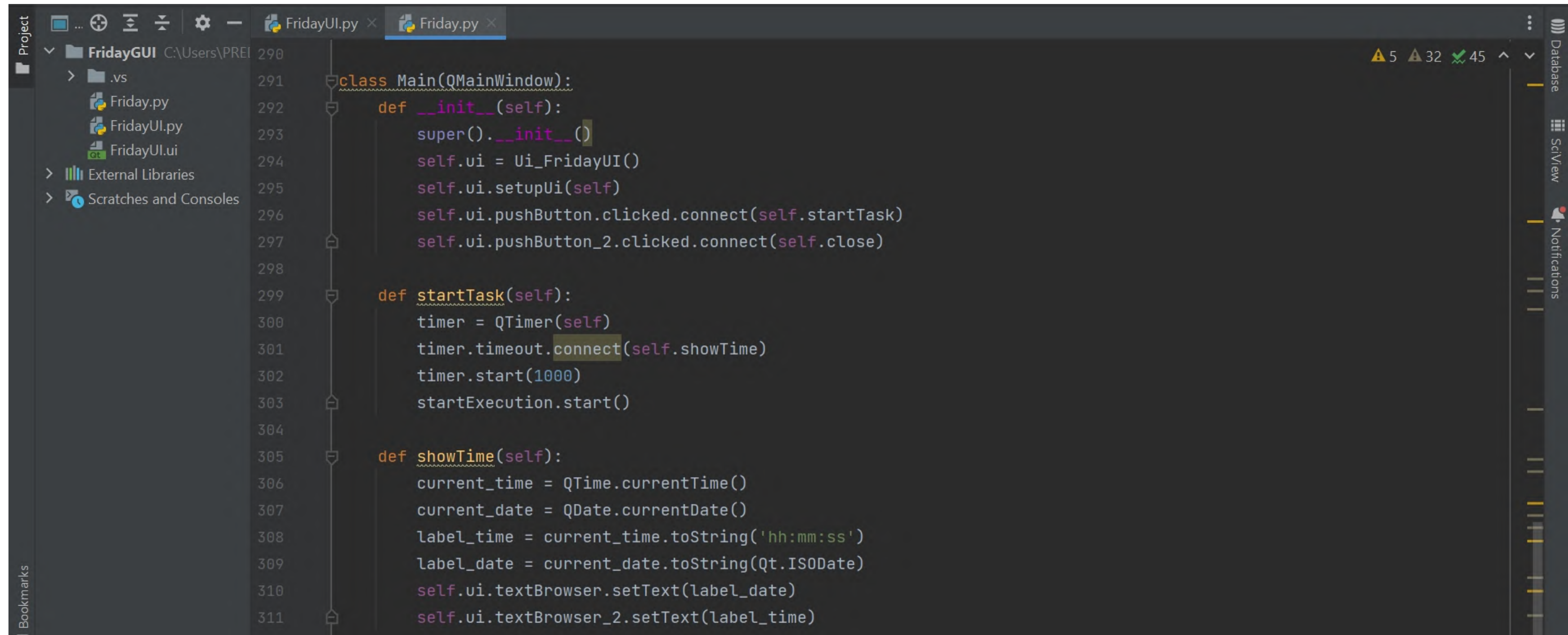
Qt Designer is a tool for quickly building graphical user interfaces with widgets from the Qt GUI framework. It gives you a simple drag-and-drop interface for laying out components such as buttons, text fields, combo boxes and more.



Converting into py

```
class Ui_FridayUI(object):
    def setupUi(self, FridayUI):
        FridayUI.setObjectName("FridayUI")
        FridayUI.resize(1301, 701)
        self.centralwidget = QtWidgets.QWidget(FridayUI)
        self.centralwidget.setObjectName("centralwidget")
        self.label = QtWidgets.QLabel(self.centralwidget)
        self.label.setGeometry(QtCore.QRect(0, 0, 1301, 701))
        self.label.setText("")
        self.label.setPixmap(QtGui.QPixmap("../Friday.png"))
        self.label.setScaledContents(True)
        self.label.setObjectName("label")
        self.pushButton = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton.setGeometry(QtCore.QRect(610, 320, 81, 61))
        self.pushButton.setStyleSheet("font: 15pt \"Impact\";\n"
"background-color: transparent;\n"
"color: rgb(255, 255, 255);")
        self.pushButton.setObjectName("pushButton")
        self.pushButton_2 = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton_2.setGeometry(QtCore.QRect(1050, 610, 71, 21))
        self.pushButton_2.setStyleSheet("font: 10pt \"Impact\";\n"
"background-color: rgb(0, 188, 253);\n"
"color: rgb(255, 255, 255);")
```


Putting Everything Together

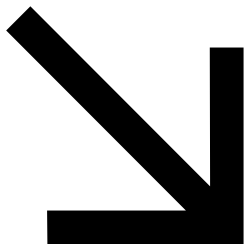


```
290
291 class Main(QMainWindow):
292     def __init__(self):
293         super().__init__()
294         self.ui = Ui_FridayUI()
295         self.ui.setupUi(self)
296         self.ui.pushButton.clicked.connect(self.startTask)
297         self.ui.pushButton_2.clicked.connect(self.close)
298
299     def startTask(self):
300         timer = QTimer(self)
301         timer.timeout.connect(self.showTime)
302         timer.start(1000)
303         startExecution.start()
304
305     def showTime(self):
306         current_time = QTime.currentTime()
307         current_date = QDate.currentDate()
308         label_time = current_time.toString('hh:mm:ss')
309         label_date = current_date.toString(Qt.ISODate)
310         self.ui.textBrowser.setText(label_date)
311         self.ui.textBrowser_2.setText(label_time)
```

FRIDAY

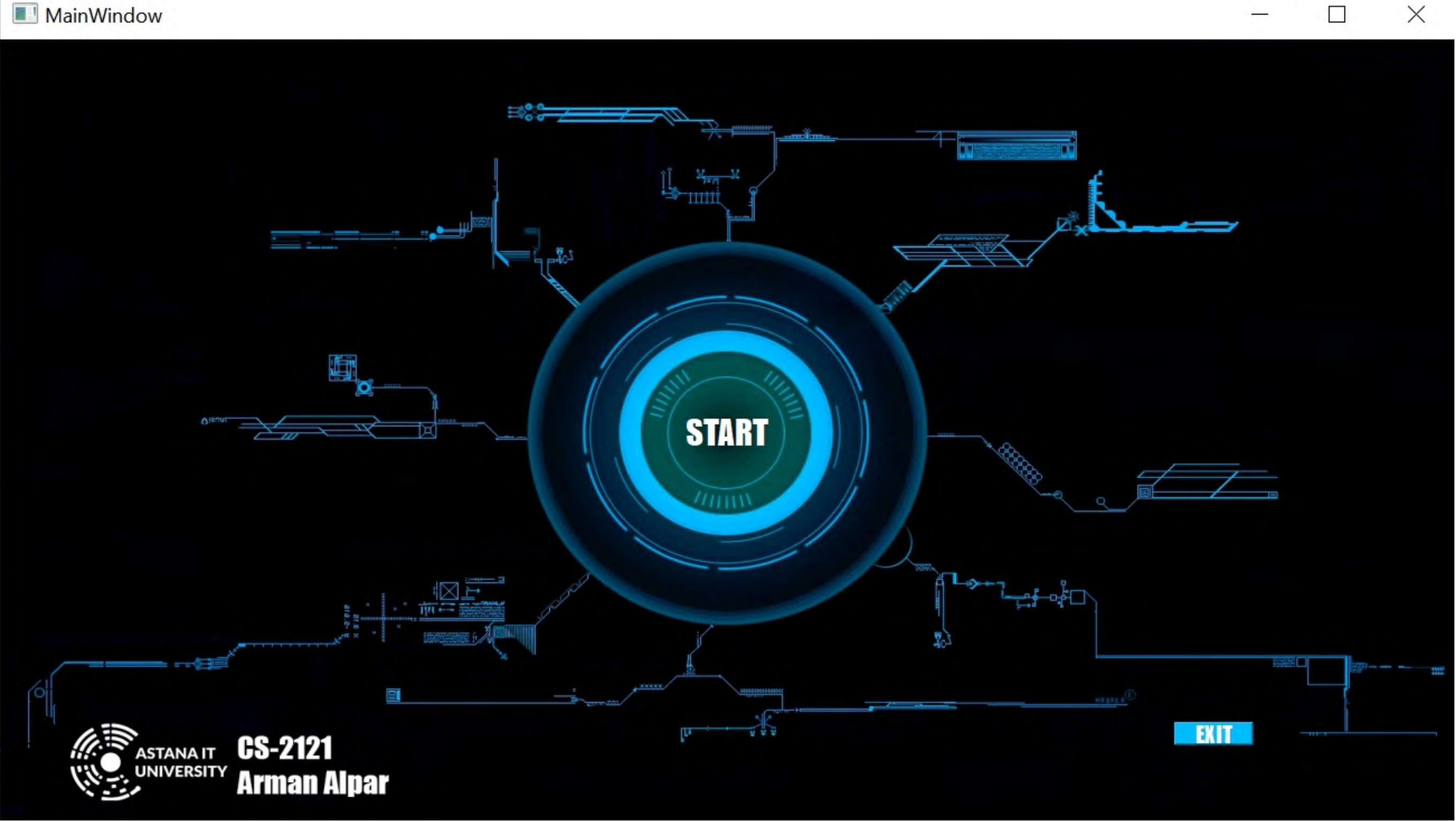
Python

Final product



Friday

Python



Final

Project

End

Thanks for your attention

