Rajiv Gandhi University Of Knowledge and Technologies Nuzvid

(A.P. Government Act 18 of 2008)
RGUKT-NUZVID, Krishna Dist. - 521202
Tele Fax: 08656 – 235557/235150

A Mini Project

On

"Virtual Try On Using Image Processing "

Submitted by

N181056	D.Gayathri
N180261	MD.Salma
N180263	Sk.Sameena
N180655	G.Krishnakumari
N181031	J.Gayathri

Department of Computer Science Engineering

2021-2022

CERTIFICATE OF COMPLETION

Certified that the mini-project work entitled"Virtual Try On Using Image Processing "is bonafied work carried out by

N181056	D.Gayathri
N180261	MD.Salma
N180263	Sk.Sameena
N180655	G.Krishnakumari
N181031	J.Gayathri

The report has been approved as it satisfies the academic requirements in respect of mini-project work prescribed for the course.

Mrs.Krishna Priya

Mini-Project Coordinator

Head of the Department
RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES
(A.P. Government Act 18 of 2008)
RGUKT-NUZVID, Krishna Dist - 521202
Tele Fax : 08656 – 235557/235150

CERTIFICATE OF EXAMINATION

This is to certify that the work entitled, " Virtual Try On Using Image Processing " is the Bonafide work of (N181056) D.Gayathri,(N180261) MD.Salma, (N180263) Sk.Sameena, (N180655) G.Krishnakumari, (N181031) J.Gayathri and here by accord our approval of it as a study carried out and presented in a manner required for its acceptance in pre final year of Bachelor of Technology for which it has been submitted. This approval does not necessarily endorse or accept every statement made an opinion expressed or conclusion drawn, as a recorded in this thesis. It only signifies the acceptance of this thesis for the purpose for which it has been submitted.

Mrs.K.Krishna Priya

Project supervisor

Assist. Prof Dept .of CSE

Examiner

Project supervisor

Lecturer Dept .of CSE

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

(A.P. Government Act 18 of 2008)
RGUKT-NUZVID, Krishna Dist. - 521202
Tele Fax: 08656 – 235557/235150

DECLARATION

We, hereby declare that the project report entitled as " Virtual Try On Using Image Processing " done by us under the guidance of Mrs Kona Krishna Priya is submitted for pre final year of Bachelor of Technology in Computer Science and Engineering at RGUKT - Nuzvid. We also declare that this project is a result of our own effort and has not been copied or imitated from any source. Citations from any websites are mentioned in the references. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

Acknowledgment

We would like to express our deepest appreciation to all those who provided us the possibility to complete the project.

A special gratitude, we give to our mini project guidance Mrs K.Krishna Priya whose contribution in simulations, suggestions have encouraged us a lot.

N181056	D.Gayatri(N181056)
N180261	MD.Salma(N180261)
N180263	Sk.Sameena(N180263)
N181031	J.Gayatri(N181031)
N180655	G.Krishnakumari(N180655)

Virtual Try On Using Image Processing

CONTENTS

ABSTRACT

The virtual try-on project is a computer vision-based application that aims to revolutionize the retail industry by allowing customers to virtually try on accessories before making a purchase. Traditional shopping often poses challenges for customers, such as limited stock availability, sizing discrepancies, and the inability to physically try on products. Virtual try-on technology addresses these issues by leveraging computer vision algorithms to superimpose virtual accessories onto a user's image in real-time, creating an immersive and interactive shopping experience.

1.INTRODUCTION

Importance Of Virtual Try On Using Image Processing in Today's Era:

Virtual try-on project have completely transformed the way we approach shopping for glasses and accessories. Traditionally, customers had to rely on physical stores to try on different outfits, leading to time-consuming and sometimes frustrating experiences. However, with the introduction of virtual try-on project, customers can now enjoy the convenience of trying on items virtually, without leaving their homes.

By leveraging these technology , customers can visualize themselves wearing different products in real-time, providing them with a more immersive and interactive shopping experience.

One of the key advantages of virtual try-on projects is the enhanced customer experience they offer. Customers can now make more informed decisions by seeing how a particular glasses or accessories will look on them before making a purchase. This leads to increased customer satisfaction and reduced product returns, as customers have a clearer understanding of how the items will fit and suit them.

Additionally, virtual try-on projects have been proven to increase conversion rates. When customers can virtually try on clothes and see themselves wearing them, they are more likely to feel confident in their purchase decisions. This bridging of the gap between online and offline shopping experiences helps businesses drive sales and boosts their bottom line.

Another significant benefit is the level of personalization and customization virtual try-on projects provide. Customers can explore different styles, colors, and sizes tailored to their individual preferences and body types. This level of personalization enhances the overall shopping experience and helps customers find products that truly match their unique tastes.

Virtual try-on projects also contribute to cost and time savings for both customers and businesses. Customers no longer need to physically visit multiple stores to try on clothes, saving them time and transportation costs. Moreover, businesses can reduce the expenses associated with managing physical inventory, as virtual try-on allows them to showcase their entire product range digitally.

As technology continues to advance, This project expected to become even more sophisticated and widely adopted. They have already revolutionized the way we shop, offering enhanced customer experiences, increased conversion rates, personalization, cost and time savings, trend exploration, and sustainability. These projects are reshaping the fashion and retail industries and are poised to play a pivotal role in the future of shopping.

EXISTING SYSTEM:

The existing system is basically a Web based system. In that we can't try on Necklaces but only glasses.

PROPOSED SYSTEM:

In our System along with the Spectacles we have added extra functionality of trying necklaces.

This Was a Desktop Application With the attracting User Interface

2 .Problem Definition

Technologies used:

IDE: PyCharm

PyCharm is an Integrated Development Environment (IDE) specificallydesigned for Python development. It is developed by JetBrains, a renowned software development company. PyCharm provides a wide range of features and tools to facilitate efficient and productive Python programming.

PyCharm is available in two editions: PyCharm Community Edition (free and open-source) and PyCharm Professional Edition (commercial with additional advanced features).

Overall, PyCharm is a comprehensive and feature-rich IDE that empowers Python developers with a seamless development experience, boosting productivity and code quality

Programming Language: Python

Python's versatility and rich ecosystem make it suitable for a wide range of applications, including web development, data analysis, scientific computing, artificial intelligence, machine learning, automation, and more. Its simplicity, readability, and extensive libraries have made Python a preferred choice for both beginners and experienced developers alike.

Python is a high-level, interpreted programming language known for its simplicity, readability, and versatility. It was created by Guido van Rossum and first released in 1991. Python has gained immense popularity in various domains, including web development, data analysis, machine learning, scientific computing, and automation.

Open cv :

OpenCV (Open Source Computer Vision) is an open-source computer vision and machine learning library. It provides a comprehensive set of functions and algorithms for image and video processing, object detection and tracking, feature extraction, and more. OpenCV is widely used in various domains, including robotics, augmented reality, healthcare, surveillance, and automotive.

OpenCV is primarily written in C++, but it also provides APIs for Python, Java, and other programming languages. This allows developers to use OpenCV with their preferred programming languages and environments. Overall, OpenCV is a powerful and versatile library for computer vision tasks, offering a wide range of capabilities to developers and researchers in the field.

Dlib(Face_Recognition):

Dlib is a popular open-source library for computer vision, machine learning, and image processing. It provides a comprehensive set of tools and algorithms for various tasks, including face detection, facial landmark detection, face recognition, object detection and tracking, image segmentation, and more.

dlib includes highly accurate face detection algorithms, including a frontal face detector (get_frontal_face_detector())

dlib provides a powerful facial landmark detection algorithm (shape_predictor()) that can localize key points on a face, such as eyes, nose, mouth, and jawline. This allows for various applications, including face alignment, expression analysis, and facial feature-based recognition.

Python Framework: (PyQt5):

PyQt5 is a Python binding for the Qt framework, which is a popular and powerful cross-platform application development framework. PyQt5 allows developers to create graphical user interfaces (GUIs) for desktop applications using Python.

PyQt5 offers extensive documentation, tutorials, and community support, making it a popular choice for developing desktop applications with Python and Qt. It provides a robust and flexible framework for creating modern and user-friendly graphical interfaces for a wide range of use cases.

PyQt5 enables the development of cross-platform applications that can run on different operating systems, such as Windows, macOS, and Linux, without significant modifications. This is because it is built on the Qt framework, which provides a consistent API and functionality across platforms.

Software Requirements

- Coding Language : Python
- Operating system microprocessor : Windows operating system (64-bit) with 8 gb
- Tools: PyCharm,Idle Python,Visual Studio Code

Hardware Requirements

- System Processor : Intel(R) Core(TM) i5-3320M CPU @2.60GHz
- Hard Disk : 512 GB SSD
- System Type : 64-bit Operating system,x64-based Processor
- Ram : 16 GB

3.SYSTEM DESIGN

UML DIAGRAMS:

The Unified Modeling Language (UML) is a standard language for specifying, visualizing, constructing, and documentingthe artifacts of software systems, as well as for business modeling and other non-software systems.

- These methods were already evolving toward each other independently. It made sense to continue that evolution together rather than apart, eliminating the potential for any unnecessary andgratuitous differences that would further confuse users.

- By unifying the semantics and notation, they could bring somestability to the object-oriented marketplace, allowing projects to settle on one mature modeling language and letting tool builders focus on delivering more useful features.

- They expected that their collaboration would yield improvementsin all three earlier methods, helping them to capture lessons learnedand to address problems that none of their methods previously handled well

- Provide users with a ready-to-use, expressive visual modelinglanguage so they can develop and exchange meaningful models.
- Provide extensibility and specialization mechanisms to extend thecore concepts.

- Be independent of particular programming languages anddevelopment processes.

- Provide a formal basis for understanding the modeling language.Encourage the
- growth of the OO tools market.

- Support higher-level development concepts such as collaborations,frameworks, patterns and components.

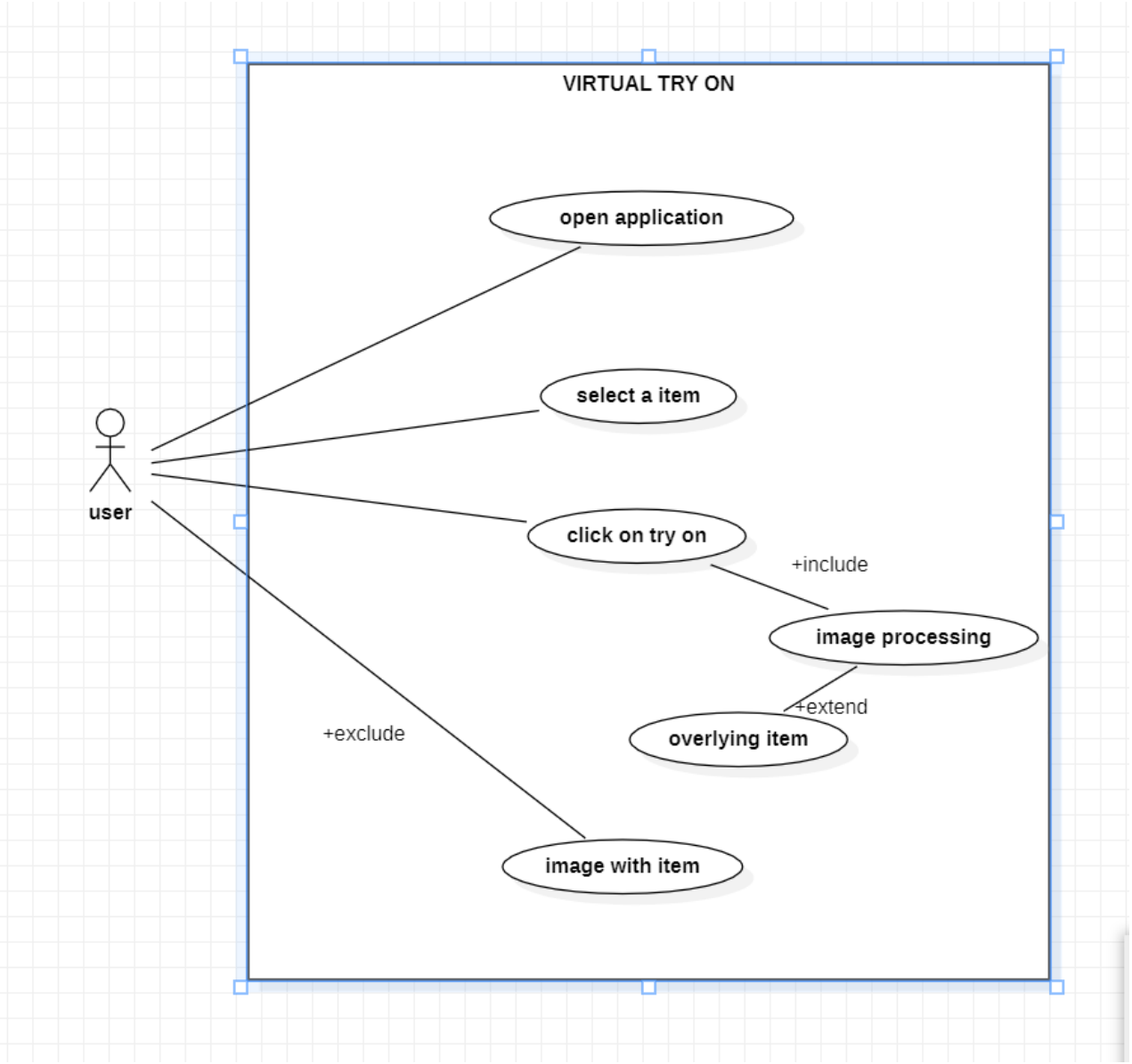
- Integrate best practices. UML has three building blocks Things-The basic building elements of model Relationships-How the things are tied together Diagrams-
- Graphical representation of element collection
- Diagramming techniques

- Each UML diagram is designed to let developers and customers view a software system from a different perspectiveand in varying degrees of abstraction. UML diagrams commonlycreated in visual modeling tools contains the following

Analysis phase

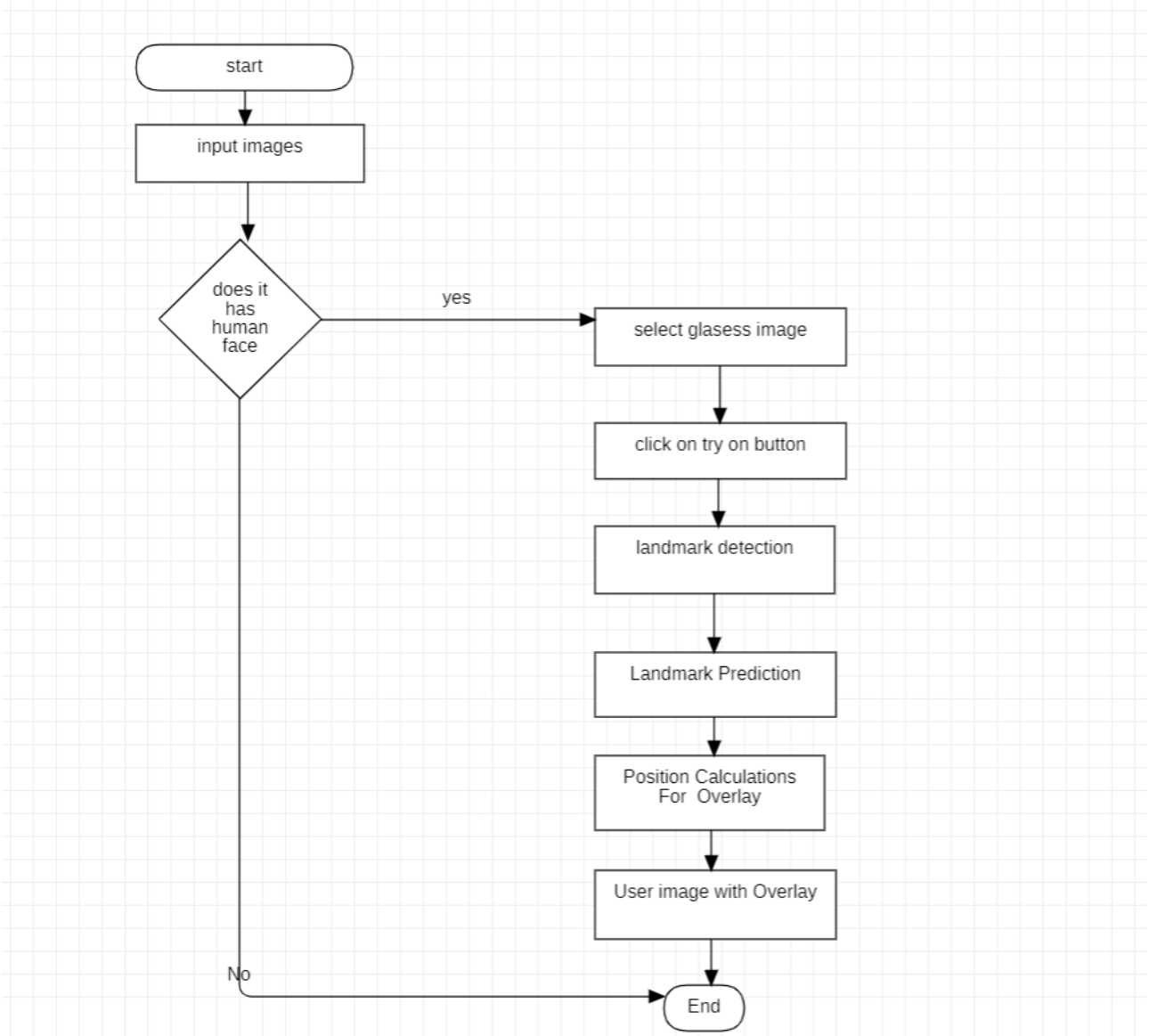
Use Case Diagram:

Use case diagrams help to capture the functional requirements of a system and illustrate the interactions between actors (users or external entities) and the system itself. They provide a high-level view of the system's behaviour and the functionalities it offers.



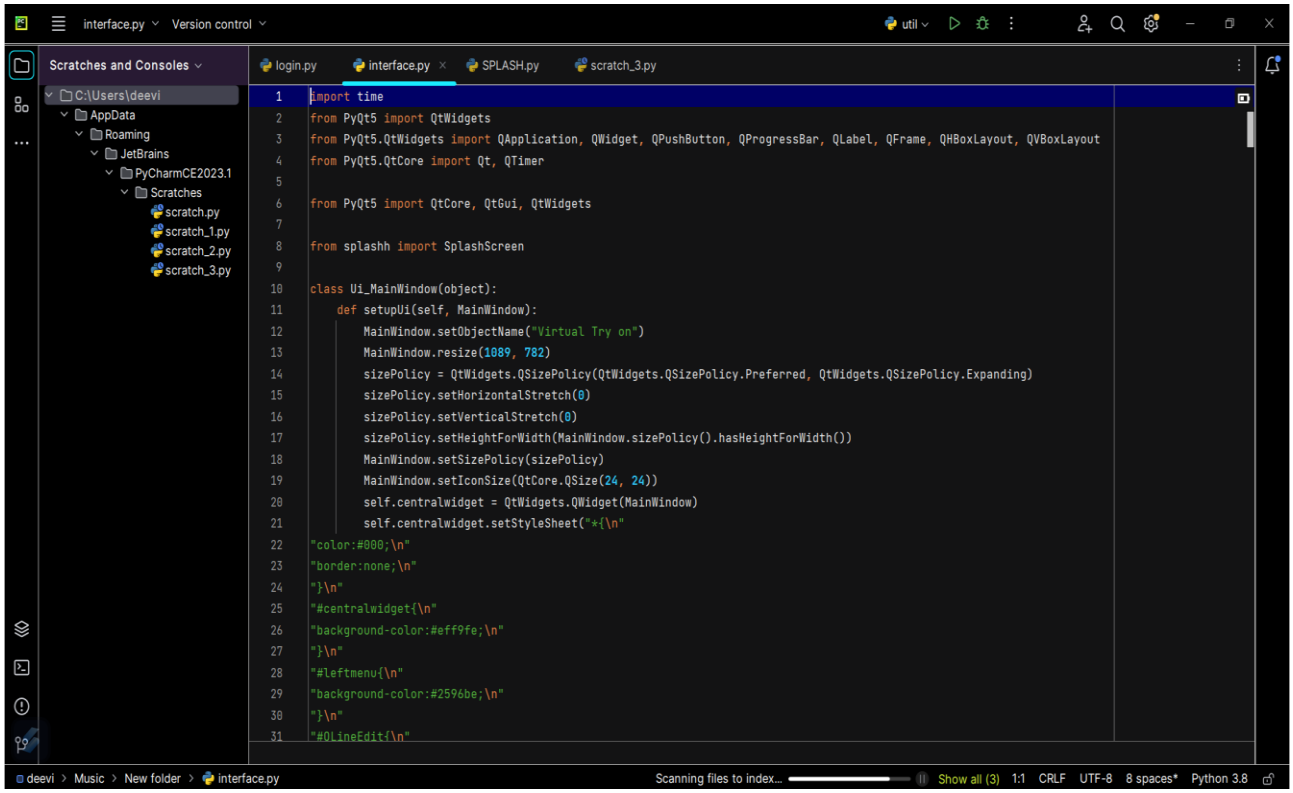
FLOW DIAGRAM

A flowchart is a picture of the separate steps of a process in sequential order. It is a generic tool that can be adapted for a wide variety of purposes, and can be used to describe various processes, such as a manufacturing process, an administrative or service process, or a project plan.

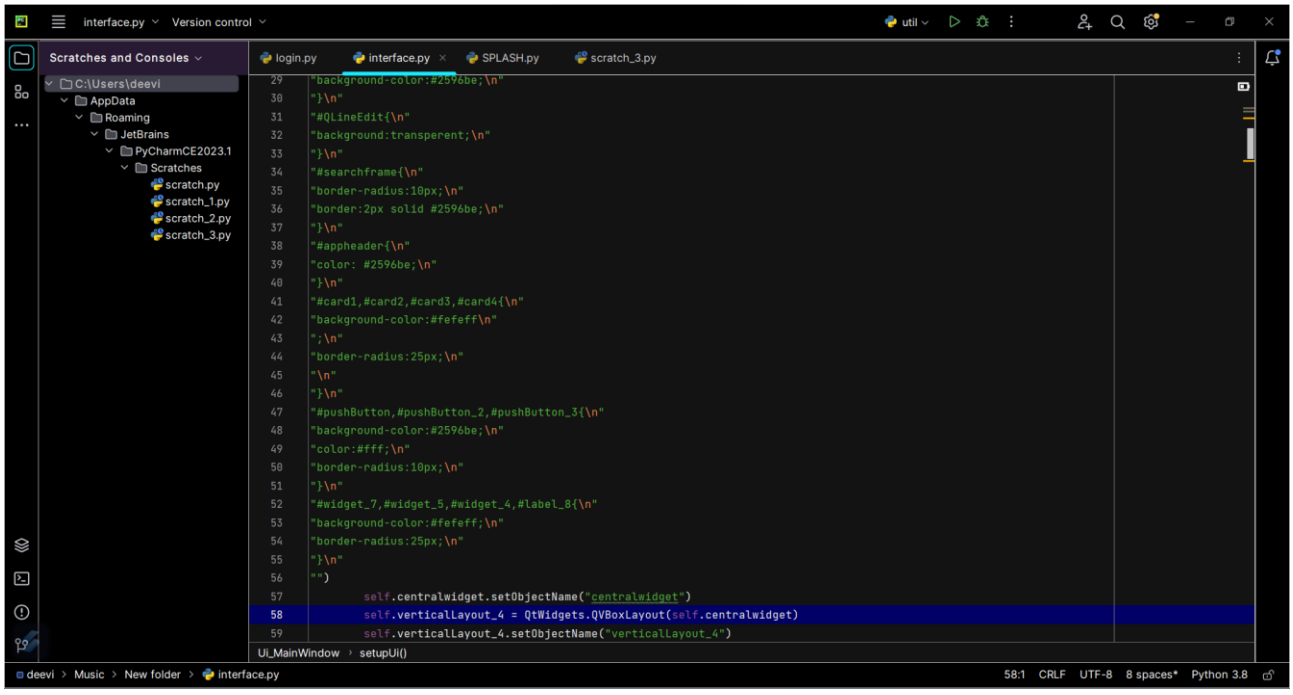


4.SAMPLE CODE

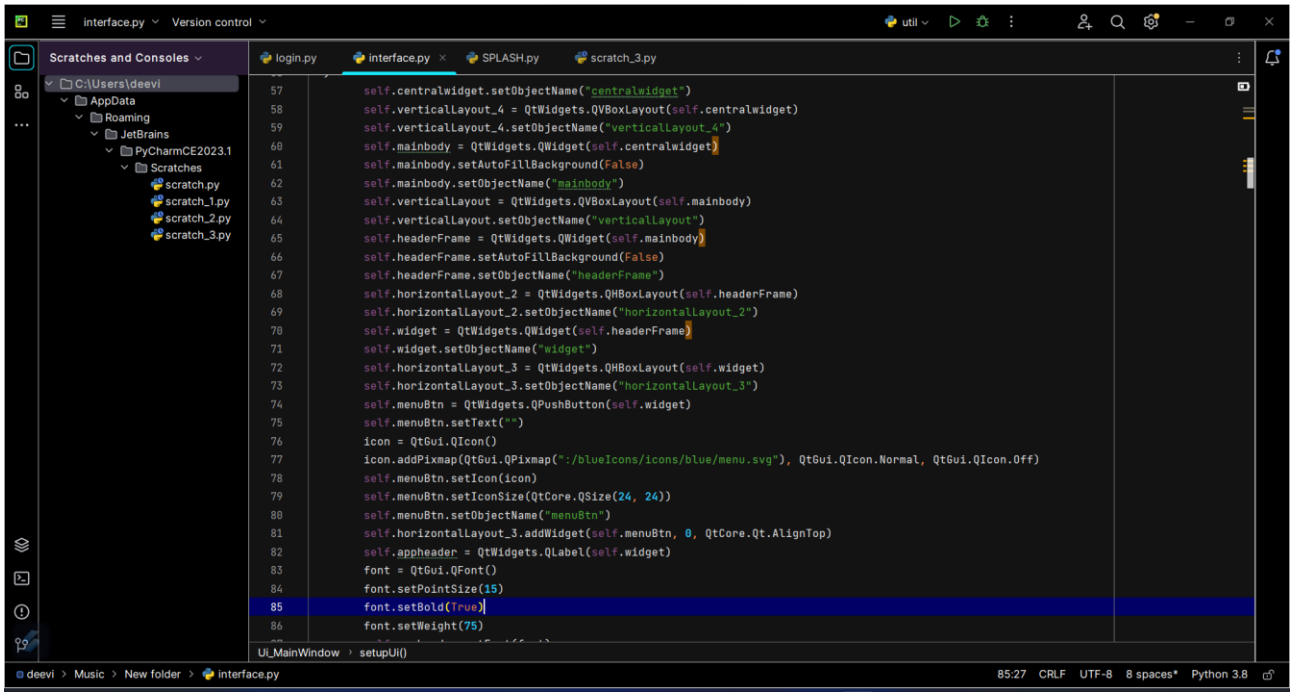
@User Interface

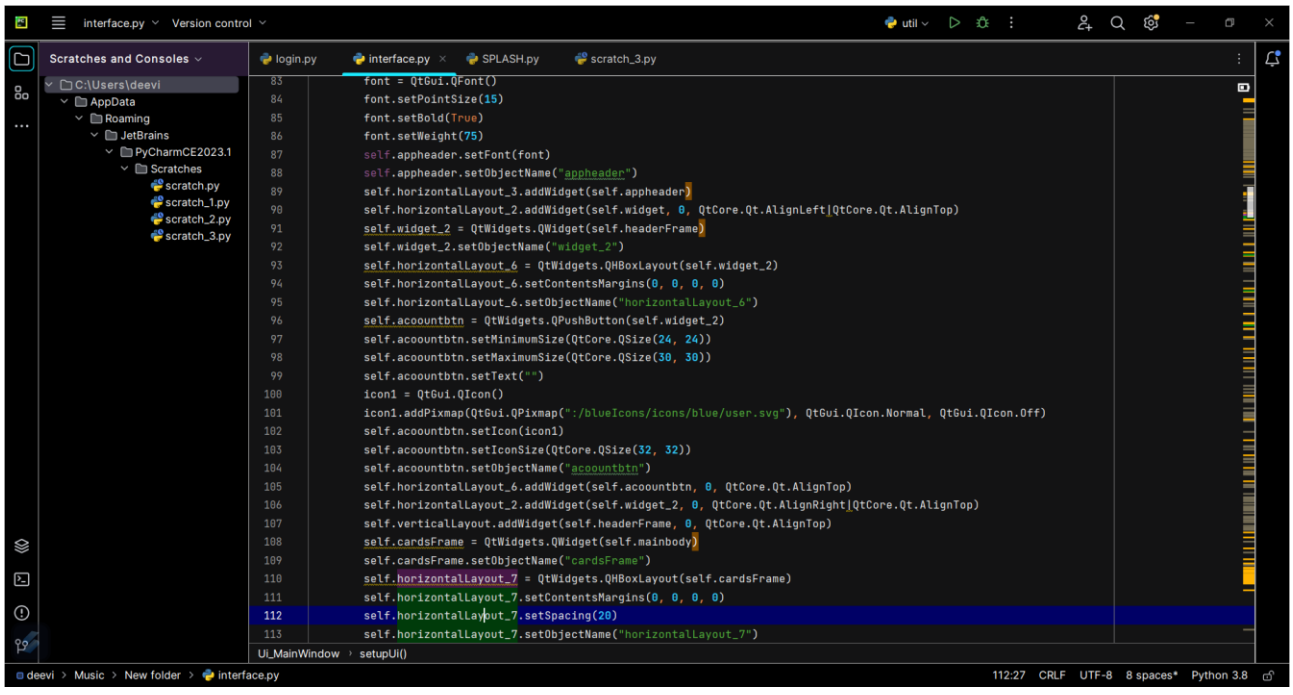


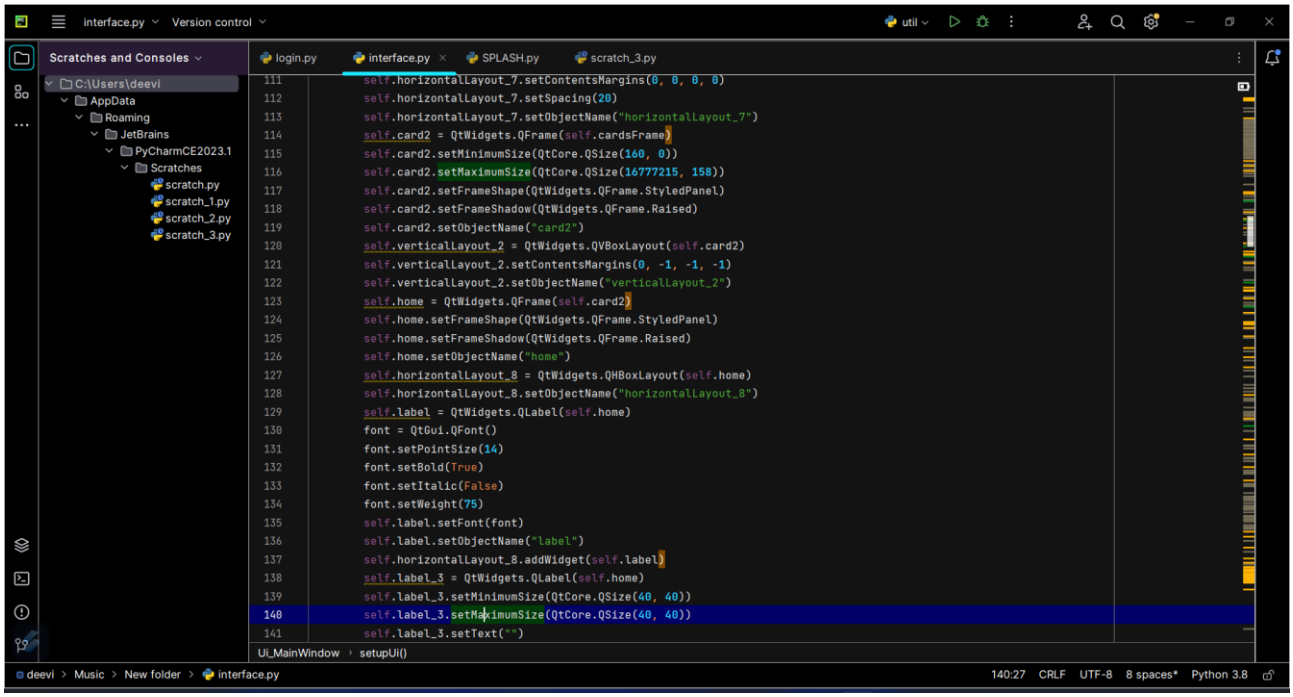
```
import time
from PyQt5 import QtWidgets
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton, QProgressBar, QLabel, QFrame, QHBoxLayout, QVBoxLayout
from PyQt5.QtCore import Qt, QTimer

from PyQt5 import QtCore, QtGui, QtWidgets

from splashh import SplashScreen

class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("Virtual Try on")
        MainWindow.resize(1089, 782)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred, QtWidgets.QSizePolicy.Expanding)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(MainWindow.sizePolicy().hasHeightForWidth())
        MainWindow.setSizePolicy(sizePolicy)
        MainWindow.setIconSize(QtCore.QSize(24, 24))
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setStyleSheet("*{\n"
"color:#000;\n"
"border:none;\n"
"}\n"
"#centralwidget{\n"
"background-color:#eff9fe;\n"
"}\n"
"#leftmenu{\n"
"background-color:#2596be;\n"
"}\n"
"#QLineEdit{\n"
```

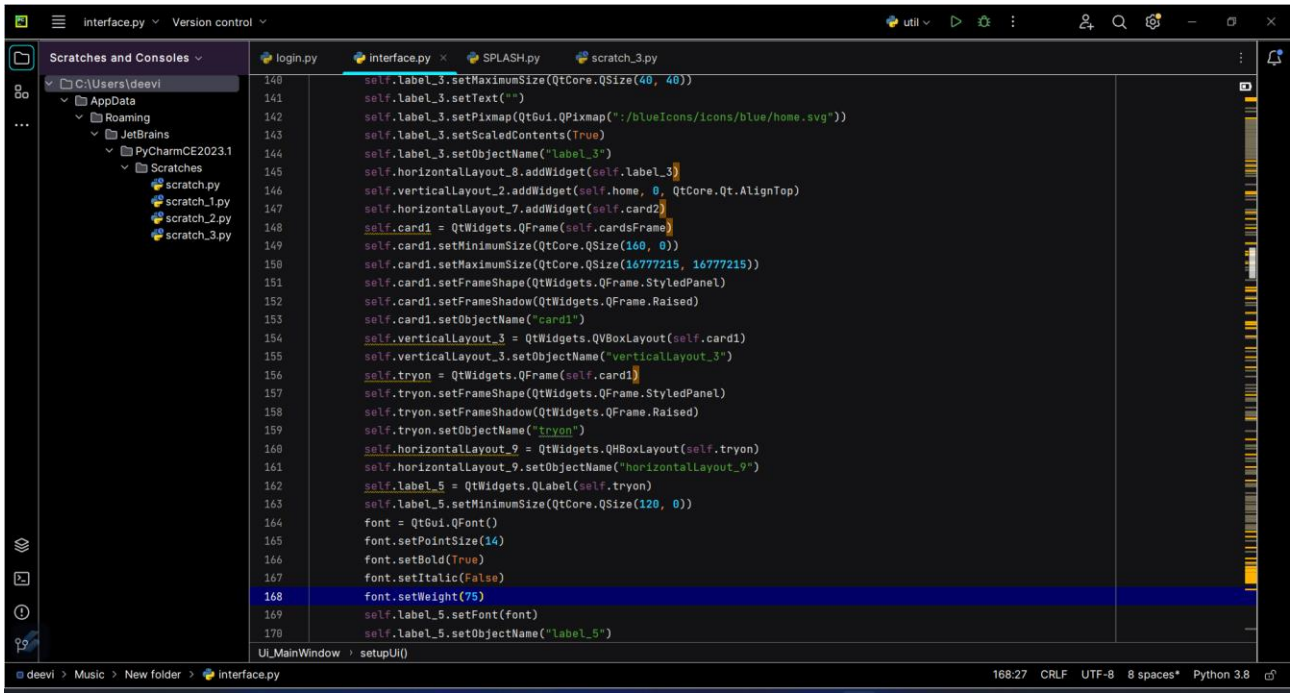


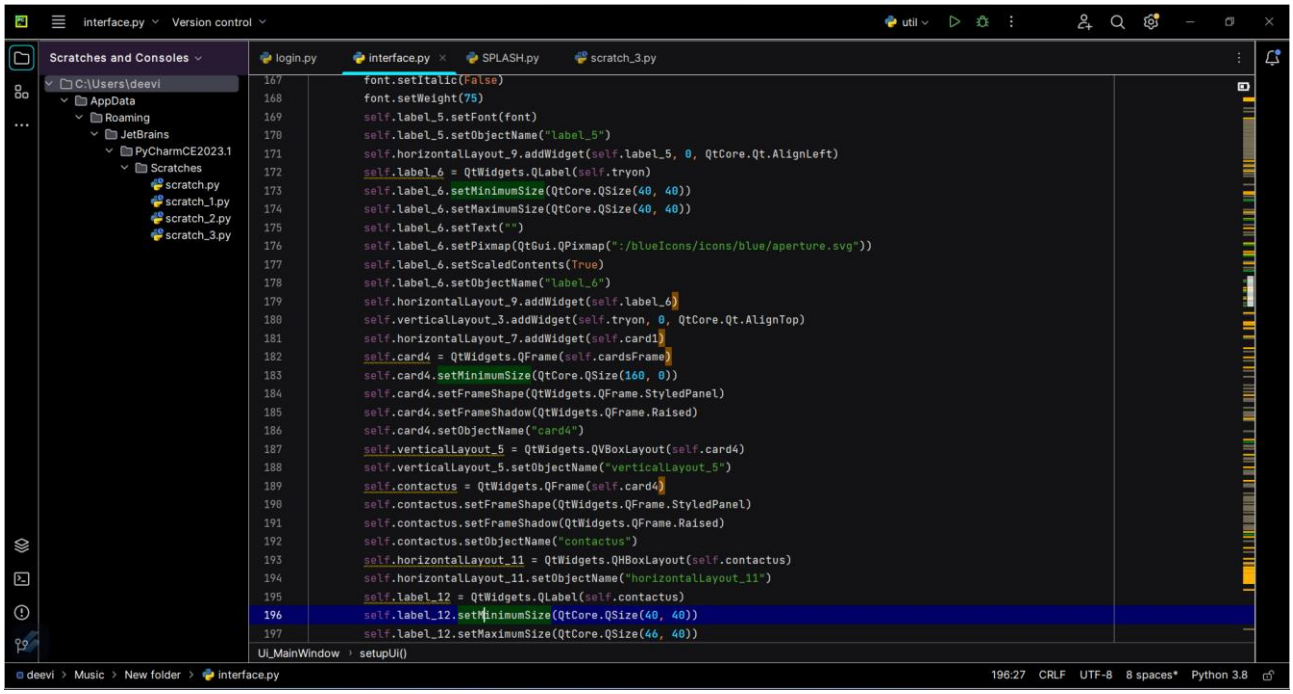
```
"background-color:#2596be;\n"
"}\n"
"#QLineEdit{\n"
"background:transperent;\n"
"}\n"
"#searchframe{\n"
"border-radius:10px;\n"
"border:2px solid #2596be;\n"
"}\n"
"#appheader{\n"
"color: #2596be;\n"
"}\n"
"#card1,#card2,#card3,#card4{\n"
"background-color:#fefeff\n"
";\n"
"border-radius:25px;\n"
"\n"
"}\n"
"#pushButton,#pushButton_2,#pushButton_3{\n"
"background-color:#2596be;\n"
"color:#fff;\n"
"border-radius:10px;\n"
"}\n"
"#widget_7,#widget_5,#widget_4,#label_8{\n"
"background-color:#fefeff;\n"
"border-radius:25px;\n"
"}\n"
"")
        self.centralwidget.setObjectName("centralwidget")
        self.verticalLayout_4 = QtWidgets.QVBoxLayout(self.centralwidget)
        self.verticalLayout_4.setObjectName("verticalLayout_4")
```

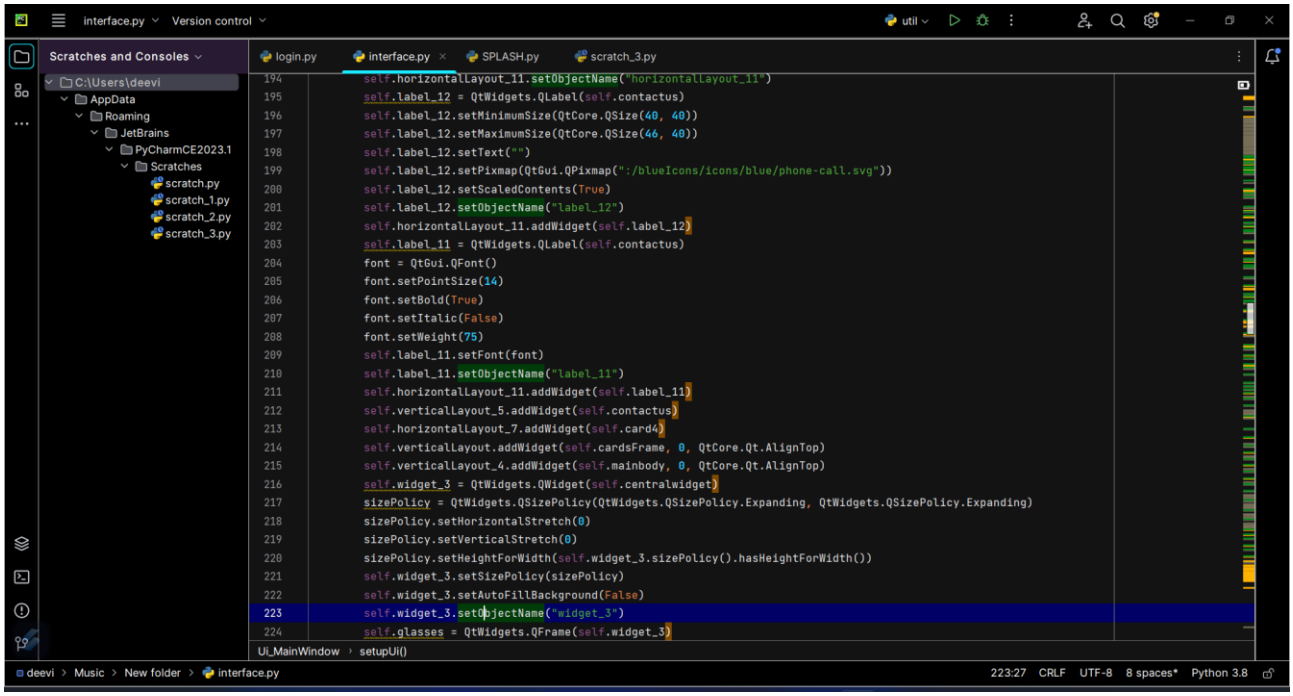

interface.py
Version control
util
Scratches and Consoles
login.py
interface.py
SPLASH.py
scratch_3.py
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
self.centralwidget.setObjectName("centralwidget")
self.verticalLayout_4 = QtWidgets.QVBoxLayout(self.centralwidget)
self.verticalLayout_4.setObjectName("verticalLayout_4")
self.mainbody = QtWidgets.QWidget(self.centralwidget)
self.mainbody.setAutoFillBackground(False)
self.mainbody.setObjectName("mainbody")
self.verticalLayout = QtWidgets.QVBoxLayout(self.mainbody)
self.verticalLayout.setObjectName("verticalLayout")
self.headerFrame = QtWidgets.QWidget(self.mainbody)
self.headerFrame.setAutoFillBackground(False)
self.headerFrame.setObjectName("headerFrame")
self.horizontalLayout_2 = QtWidgets.QHBoxLayout(self.headerFrame)
self.horizontalLayout_2.setObjectName("horizontalLayout_2")
self.widget = QtWidgets.QWidget(self.headerFrame)
self.widget.setObjectName("widget")
self.horizontalLayout_3 = QtWidgets.QHBoxLayout(self.widget)
self.horizontalLayout_3.setObjectName("horizontalLayout_3")
self.menuBtn = QtWidgets.QPushButton(self.widget)
self.menuBtn.setText("")
icon = QtGui.QIcon()
icon.addPixmap(QtGui.QPixmap(":/blueIcons/icons/blue/menu.svg"), QtGui.QIcon.Normal, QtGui.QIcon.Off)
self.menuBtn.setIcon(icon)
self.menuBtn.setIconSize(QtCore.QSize(24, 24))
self.menuBtn.setObjectName("menuBtn")
self.horizontalLayout_3.addWidget(self.menuBtn, 0, QtCore.Qt.AlignTop)
self.appheader = QtWidgets.QLabel(self.widget)
font = QtGui.QFont()
font.setPointSize(15)
font.setBold(True)
font.setWeight(75)
Ui_MainWindow
setupUi()
deevi
Music
New folder
interface.py
85:27
CRLF
UTF-8
8 spaces*
Python 3.8

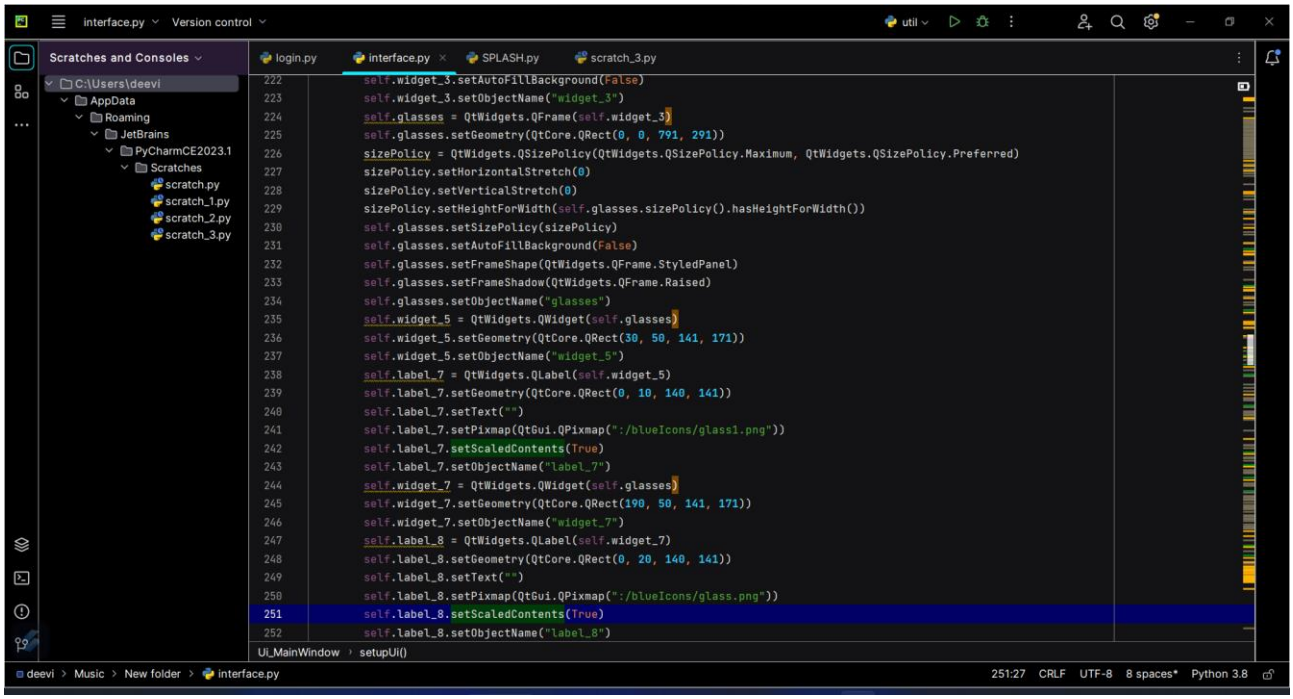

interface.py
Version control
util
Scratches and Consoles
login.py
interface.py
SPLASH.py
scratch_3.py
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
font = QtGui.QFont()
font.setPointSize(15)
font.setBold(True)
font.setWeight(75)
self.appheader.setFont(font)
self.appheader.setObjectName("appheader")
self.horizontalLayout_3.addWidget(self.appheader)
self.horizontalLayout_2.addWidget(self.widget, 0, QtCore.Qt.AlignLeft|QtCore.Qt.AlignTop)
self.widget_2 = QtWidgets.QWidget(self.headerFrame)
self.widget_2.setObjectName("widget_2")
self.horizontalLayout_6 = QtWidgets.QHBoxLayout(self.widget_2)
self.horizontalLayout_6.setContentsMargins(0, 0, 0, 0)
self.horizontalLayout_6.setObjectName("horizontalLayout_6")
self.acoountbtn = QtWidgets.QPushButton(self.widget_2)
self.acoountbtn.setMinimumSize(QtCore.QSize(24, 24))
self.acoountbtn.setMaximumSize(QtCore.QSize(30, 30))
self.acoountbtn.setText("")
icon1 = QtGui.QIcon()
icon1.addPixmap(QtGui.QPixmap(":/blueIcons/icons/blue/user.svg"), QtGui.QIcon.Normal, QtGui.QIcon.Off)
self.acoountbtn.setIcon(icon1)
self.acoountbtn.setIconSize(QtCore.QSize(32, 32))
self.acoountbtn.setObjectName("acoountbtn")
self.horizontalLayout_6.addWidget(self.acoountbtn, 0, QtCore.Qt.AlignTop)
self.horizontalLayout_2.addWidget(self.widget_2, 0, QtCore.Qt.AlignRight|QtCore.Qt.AlignTop)
self.verticalLayout.addWidget(self.headerFrame, 0, QtCore.Qt.AlignTop)
self.cardsFrame = QtWidgets.QWidget(self.mainbody)
self.cardsFrame.setObjectName("cardsFrame")
self.horizontalLayout_7 = QtWidgets.QHBoxLayout(self.cardsFrame)
self.horizontalLayout_7.setContentsMargins(0, 0, 0, 0)
self.horizontalLayout_7.setSpacing(20)
self.horizontalLayout_7.setObjectName("horizontalLayout_7")
Ui_MainWindow
setupUi()
deevi
Music
New folder
interface.py
112:27
CRLF
UTF-8
8 spaces*
Python 3.8

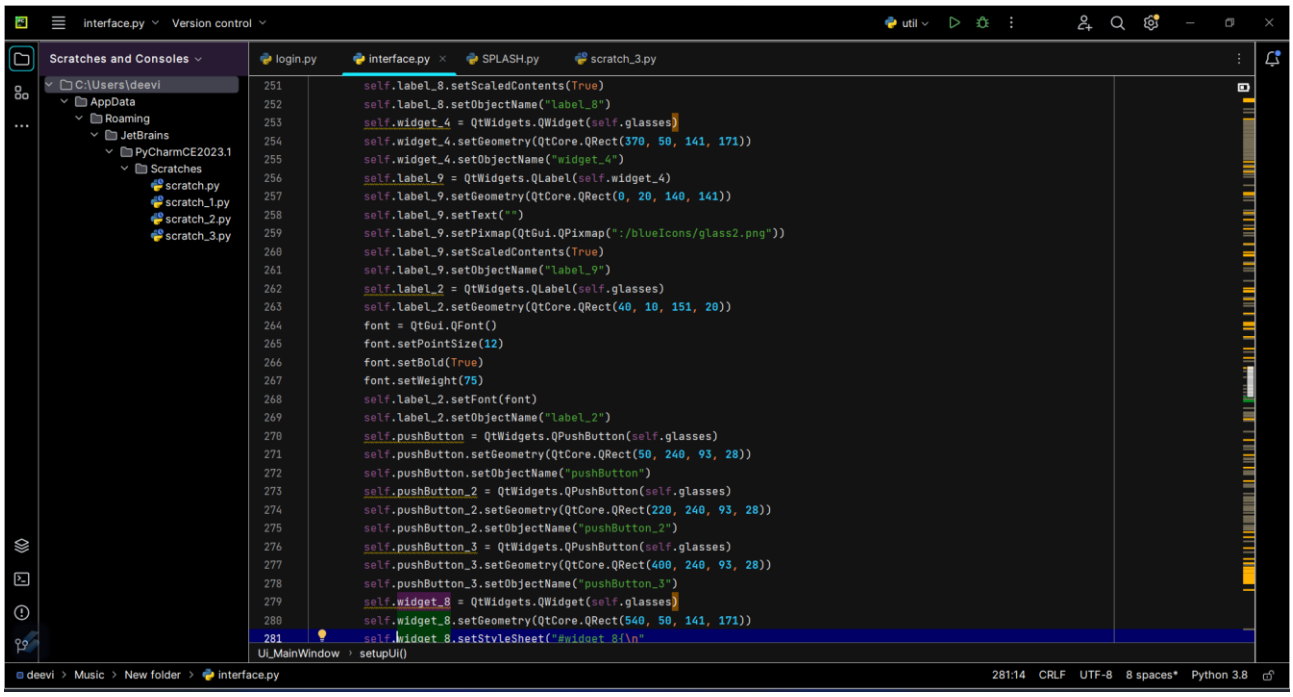

interface.py
Version control
util
Scratches and Consoles
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
login.py
interface.py
SPLASH.py
scratch_3.py
111 self.horizontalLayout_7.setContentsMargins(0, 0, 0, 0)
112 self.horizontalLayout_7.setSpacing(20)
113 self.horizontalLayout_7.setObjectName("horizontalLayout_7")
114 self.card2 = QtWidgets.QFrame(self.cardsFrame)
115 self.card2.setMinimumSize(QtCore.QSize(160, 0))
116 self.card2.setMaximumSize(QtCore.QSize(16777215, 158))
117 self.card2.setFrameShape(QtWidgets.QFrame.StyledPanel)
118 self.card2.setFrameShadow(QtWidgets.QFrame.Raised)
119 self.card2.setObjectName("card2")
120 self.verticalLayout_2 = QtWidgets.QVBoxLayout(self.card2)
121 self.verticalLayout_2.setContentsMargins(0, -1, -1, -1)
122 self.verticalLayout_2.setObjectName("verticalLayout_2")
123 self.home = QtWidgets.QFrame(self.card2)
124 self.home.setFrameShape(QtWidgets.QFrame.StyledPanel)
125 self.home.setFrameShadow(QtWidgets.QFrame.Raised)
126 self.home.setObjectName("home")
127 self.horizontalLayout_8 = QtWidgets.QHBoxLayout(self.home)
128 self.horizontalLayout_8.setObjectName("horizontalLayout_8")
129 self.label = QtWidgets.QLabel(self.home)
130 font = QtGui.QFont()
131 font.setPointSize(14)
132 font.setBold(True)
133 font.setItalic(False)
134 font.setWeight(75)
135 self.label.setFont(font)
136 self.label.setObjectName("label")
137 self.horizontalLayout_8.addWidget(self.label)
138 self.label_3 = QtWidgets.QLabel(self.home)
139 self.label_3.setMinimumSize(QtCore.QSize(40, 40))
140 self.label_3.setMaximumSize(QtCore.QSize(40, 40))
141 self.label_3.setText("")
Ui_MainWindow › setupUi()
deevi › Music › New folder › interface.py
140:27 CRLF UTF-8 8 spaces* Python 3.8

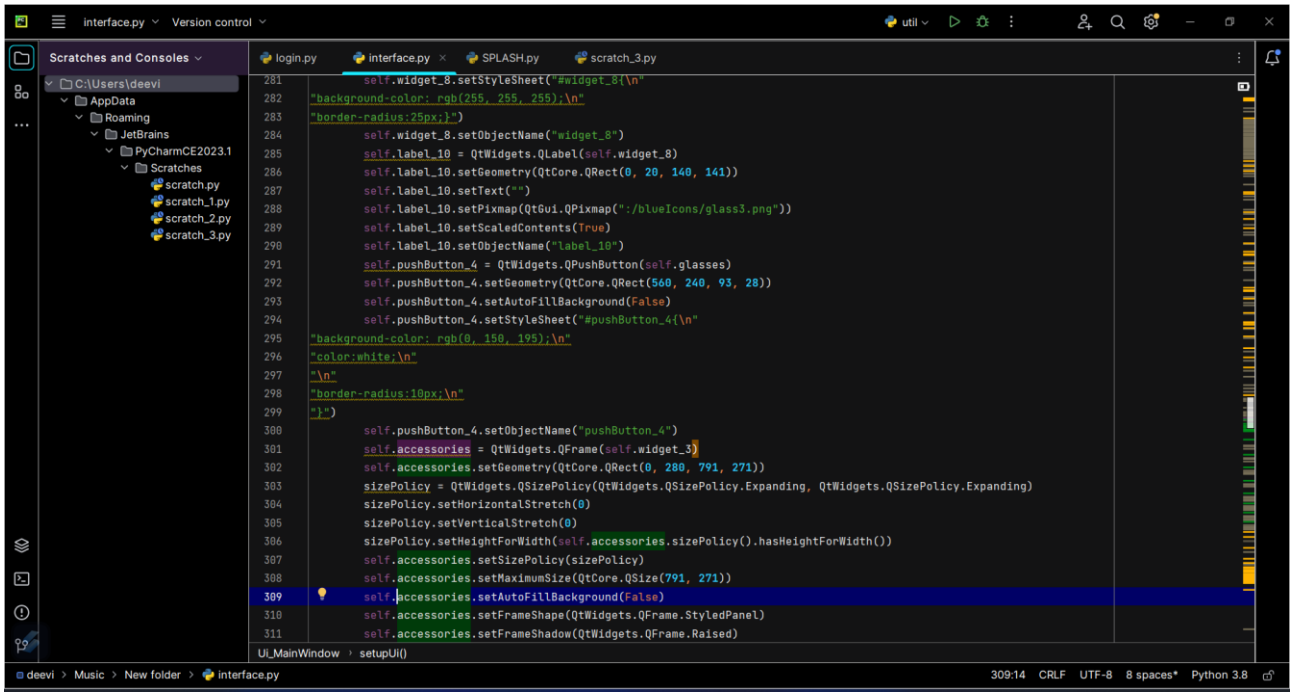

interface.py
Version control
util
Scratches and Consoles
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
login.py
interface.py
SPLASH.py
scratch_3.py
140 self.label_3.setMaximumSize(QtCore.QSize(40, 40))
141 self.label_3.setText("")
142 self.label_3.setPixmap(QtGui.QPixmap(":/blueIcons/icons/blue/home.svg"))
143 self.label_3.setScaledContents(True)
144 self.label_3.setObjectName("label_3")
145 self.horizontalLayout_8.addWidget(self.label_3)
146 self.verticalLayout_2.addWidget(self.home, 0, QtCore.Qt.AlignTop)
147 self.horizontalLayout_7.addWidget(self.card2)
148 self.card1 = QtWidgets.QFrame(self.cardsFrame)
149 self.card1.setMinimumSize(QtCore.QSize(160, 0))
150 self.card1.setMaximumSize(QtCore.QSize(16777215, 16777215))
151 self.card1.setFrameShape(QtWidgets.QFrame.StyledPanel)
152 self.card1.setFrameShadow(QtWidgets.QFrame.Raised)
153 self.card1.setObjectName("card1")
154 self.verticalLayout_3 = QtWidgets.QVBoxLayout(self.card1)
155 self.verticalLayout_3.setObjectName("verticalLayout_3")
156 self.tryon = QtWidgets.QFrame(self.card1)
157 self.tryon.setFrameShape(QtWidgets.QFrame.StyledPanel)
158 self.tryon.setFrameShadow(QtWidgets.QFrame.Raised)
159 self.tryon.setObjectName("tryon")
160 self.horizontalLayout_9 = QtWidgets.QHBoxLayout(self.tryon)
161 self.horizontalLayout_9.setObjectName("horizontalLayout_9")
162 self.label_5 = QtWidgets.QLabel(self.tryon)
163 self.label_5.setMinimumSize(QtCore.QSize(120, 0))
164 font = QtGui.QFont()
165 font.setPointSize(14)
166 font.setBold(True)
167 font.setItalic(False)
168 font.setWeight(75)
169 self.label_5.setFont(font)
170 self.label_5.setObjectName("label_5")
Ui_MainWindow › setupUi()
deevi › Music › New folder › interface.py
168:27 CRLF UTF-8 8 spaces* Python 3.8

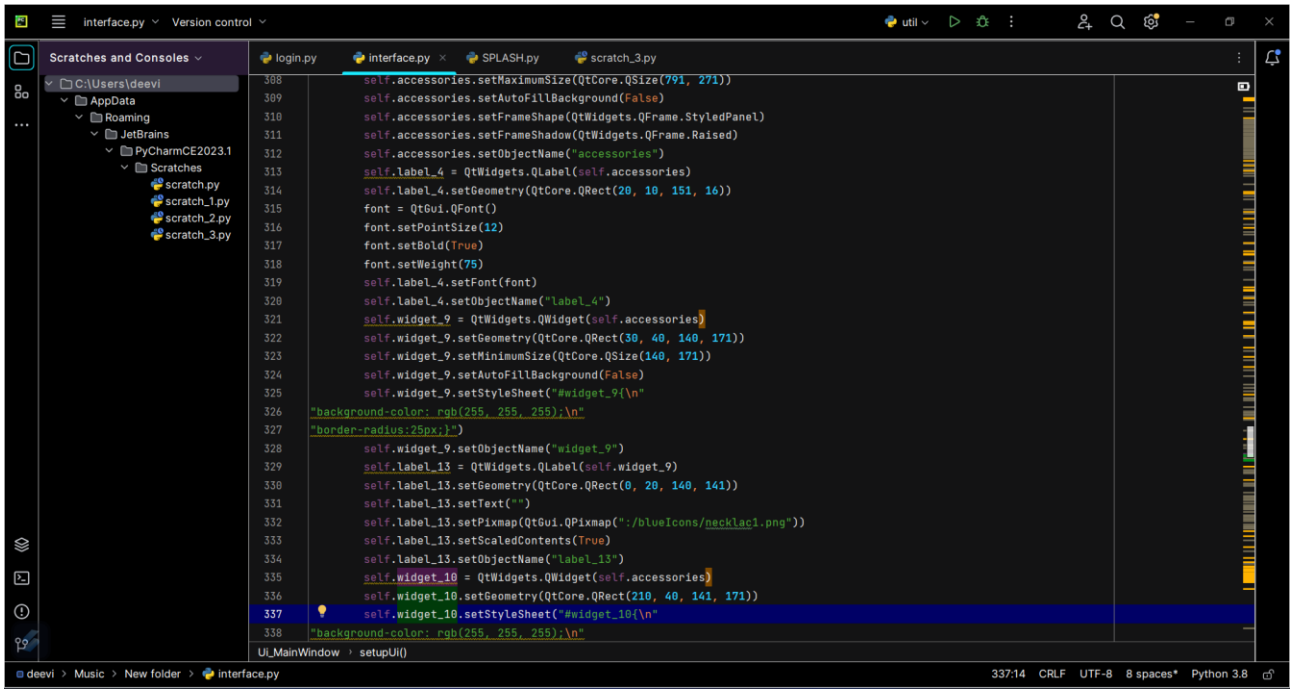

interface.py
Version control
util
Scratches and Consoles
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
login.py
interface.py
SPLASH.py
scratch_3.py
font.setItalic(False)
font.setWeight(75)
self.label_5.setFont(font)
self.label_5.setObjectName("label_5")
self.horizontalLayout_9.addWidget(self.label_5, 0, QtCore.Qt.AlignLeft)
self.label_6 = QtWidgets.QLabel(self.tryon)
self.label_6.setMinimumSize(QtCore.QSize(40, 40))
self.label_6.setMaximumSize(QtCore.QSize(40, 40))
self.label_6.setText("")
self.label_6.setPixmap(QtGui.QPixmap(":/blueIcons/icons/blue/aperture.svg"))
self.label_6.setScaledContents(True)
self.label_6.setObjectName("label_6")
self.horizontalLayout_9.addWidget(self.label_6)
self.verticalLayout_3.addWidget(self.tryon, 0, QtCore.Qt.AlignTop)
self.horizontalLayout_7.addWidget(self.card1)
self.card4 = QtWidgets.QFrame(self.cardsFrame)
self.card4.setMinimumSize(QtCore.QSize(160, 0))
self.card4.setFrameShape(QtWidgets.QFrame.StyledPanel)
self.card4.setFrameShadow(QtWidgets.QFrame.Raised)
self.card4.setObjectName("card4")
self.verticalLayout_5 = QtWidgets.QVBoxLayout(self.card4)
self.verticalLayout_5.setObjectName("verticalLayout_5")
self.contactus = QtWidgets.QFrame(self.card4)
self.contactus.setFrameShape(QtWidgets.QFrame.StyledPanel)
self.contactus.setFrameShadow(QtWidgets.QFrame.Raised)
self.contactus.setObjectName("contactus")
self.horizontalLayout_11 = QtWidgets.QHBoxLayout(self.contactus)
self.horizontalLayout_11.setObjectName("horizontalLayout_11")
self.label_12 = QtWidgets.QLabel(self.contactus)
self.label_12.setMinimumSize(QtCore.QSize(40, 40))
self.label_12.setMaximumSize(QtCore.QSize(46, 40))
Ui_MainWindow
setupUi()
deevi
Music
New folder
interface.py
196:27
CRLF
UTF-8
8 spaces*
Python 3.8

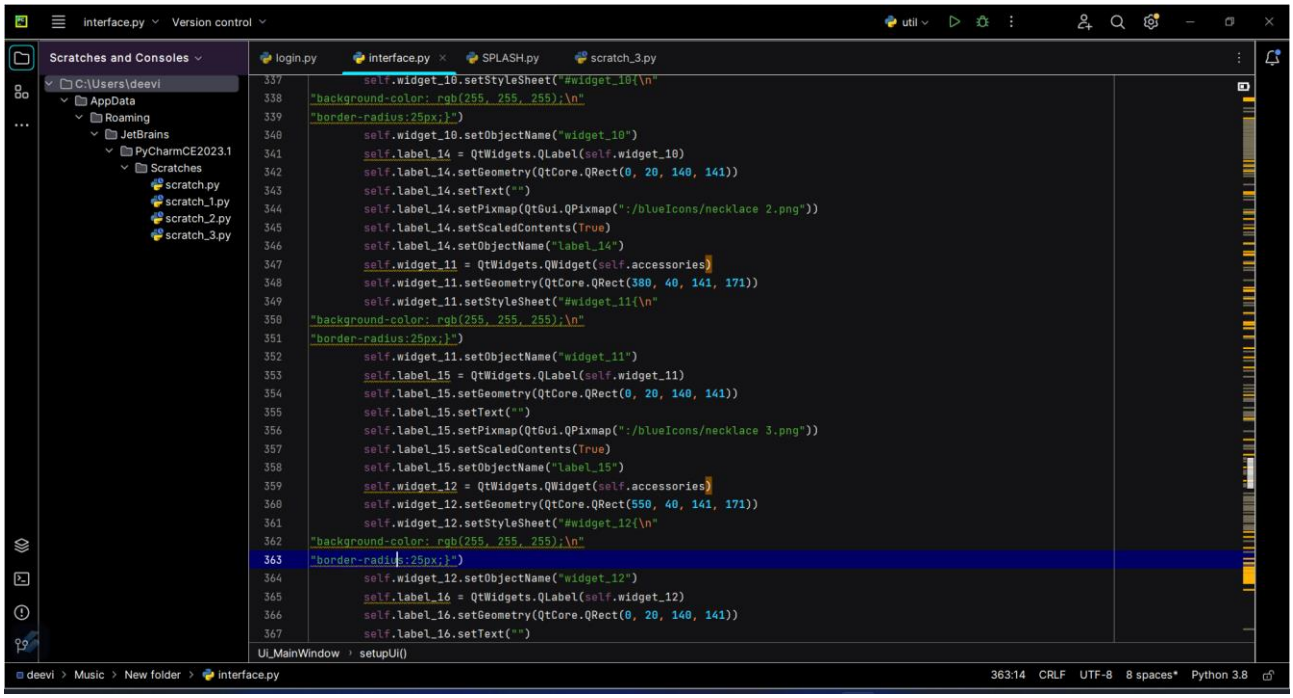

interface.py
Version control
util
Scratches and Consoles
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
login.py
interface.py
SPLASH.py
scratch_3.py
self.horizontalLayout_11.setObjectName("horizontalLayout_11")
self.label_12 = QtWidgets.QLabel(self.contactus)
self.label_12.setMinimumSize(QtCore.QSize(40, 40))
self.label_12.setMaximumSize(QtCore.QSize(46, 40))
self.label_12.setText("")
self.label_12.setPixmap(QtGui.QPixmap(":/blueIcons/icons/blue/phone-call.svg"))
self.label_12.setScaledContents(True)
self.label_12.setObjectName("label_12")
self.horizontalLayout_11.addWidget(self.label_12)
self.label_11 = QtWidgets.QLabel(self.contactus)
font = QtGui.QFont()
font.setPointSize(14)
font.setBold(True)
font.setItalic(False)
font.setWeight(75)
self.label_11.setFont(font)
self.label_11.setObjectName("label_11")
self.horizontalLayout_11.addWidget(self.label_11)
self.verticalLayout_5.addWidget(self.contactus)
self.horizontalLayout_7.addWidget(self.card4)
self.verticalLayout.addWidget(self.cardsFrame, 0, QtCore.Qt.AlignTop)
self.verticalLayout_4.addWidget(self.mainbody, 0, QtCore.Qt.AlignTop)
self.widget_3 = QtWidgets.QWidget(self.centralwidget)
sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Expanding)
sizePolicy.setHorizontalStretch(0)
sizePolicy.setVerticalStretch(0)
sizePolicy.setHeightForWidth(self.widget_3.sizePolicy().hasHeightForWidth())
self.widget_3.setSizePolicy(sizePolicy)
self.widget_3.setAutoFillBackground(False)
self.widget_3.setObjectName("widget_3")
self.glasses = QtWidgets.QFrame(self.widget_3)
Ui_MainWindow
setupUi()
deevi
Music
New folder
interface.py
223:27
CRLF
UTF-8
8 spaces*
Python 3.8

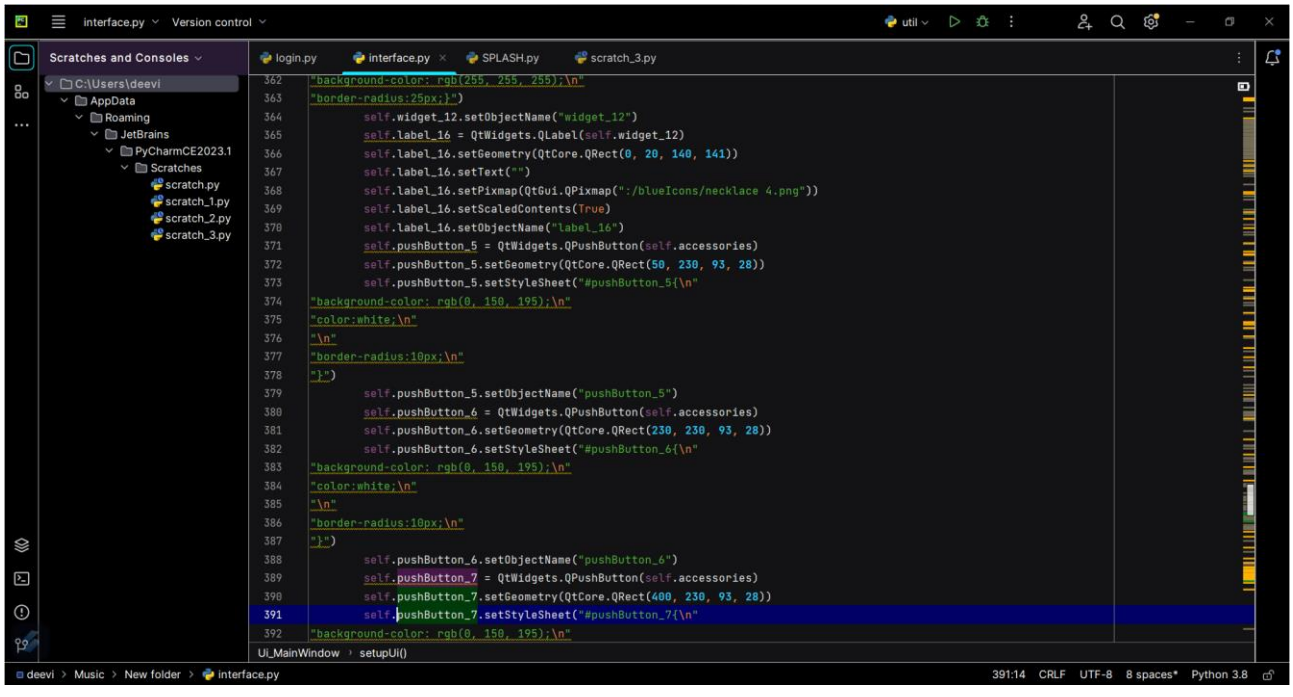

interface.py
Version control
util
Scratches and Consoles
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
login.py
interface.py
SPLASH.py
scratch_3.py
self.widget_3.setAutoFillBackground(False)
self.widget_3.setObjectName("widget_3")
self.glasses = QtWidgets.QFrame(self.widget_3)
self.glasses.setGeometry(QtCore.QRect(0, 0, 791, 291))
sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Preferred)
sizePolicy.setHorizontalStretch(0)
sizePolicy.setVerticalStretch(0)
sizePolicy.setHeightForWidth(self.glasses.sizePolicy().hasHeightForWidth())
self.glasses.setSizePolicy(sizePolicy)
self.glasses.setAutoFillBackground(False)
self.glasses.setFrameShape(QtWidgets.QFrame.StyledPanel)
self.glasses.setFrameShadow(QtWidgets.QFrame.Raised)
self.glasses.setObjectName("glasses")
self.widget_5 = QtWidgets.QWidget(self.glasses)
self.widget_5.setGeometry(QtCore.QRect(30, 50, 141, 171))
self.widget_5.setObjectName("widget_5")
self.label_7 = QtWidgets.QLabel(self.widget_5)
self.label_7.setGeometry(QtCore.QRect(0, 10, 140, 141))
self.label_7.setText("")
self.label_7.setPixmap(QtGui.QPixmap(":/blueIcons/glass1.png"))
self.label_7.setScaledContents(True)
self.label_7.setObjectName("label_7")
self.widget_7 = QtWidgets.QWidget(self.glasses)
self.widget_7.setGeometry(QtCore.QRect(190, 50, 141, 171))
self.widget_7.setObjectName("widget_7")
self.label_8 = QtWidgets.QLabel(self.widget_7)
self.label_8.setGeometry(QtCore.QRect(0, 20, 140, 141))
self.label_8.setText("")
self.label_8.setPixmap(QtGui.QPixmap(":/blueIcons/glass.png"))
self.label_8.setScaledContents(True)
self.label_8.setObjectName("label_8")
Ui_MainWindow
setupUi()
deevi
Music
New folder
interface.py
251:27
CRLF
UTF-8
8 spaces*
Python 3.8

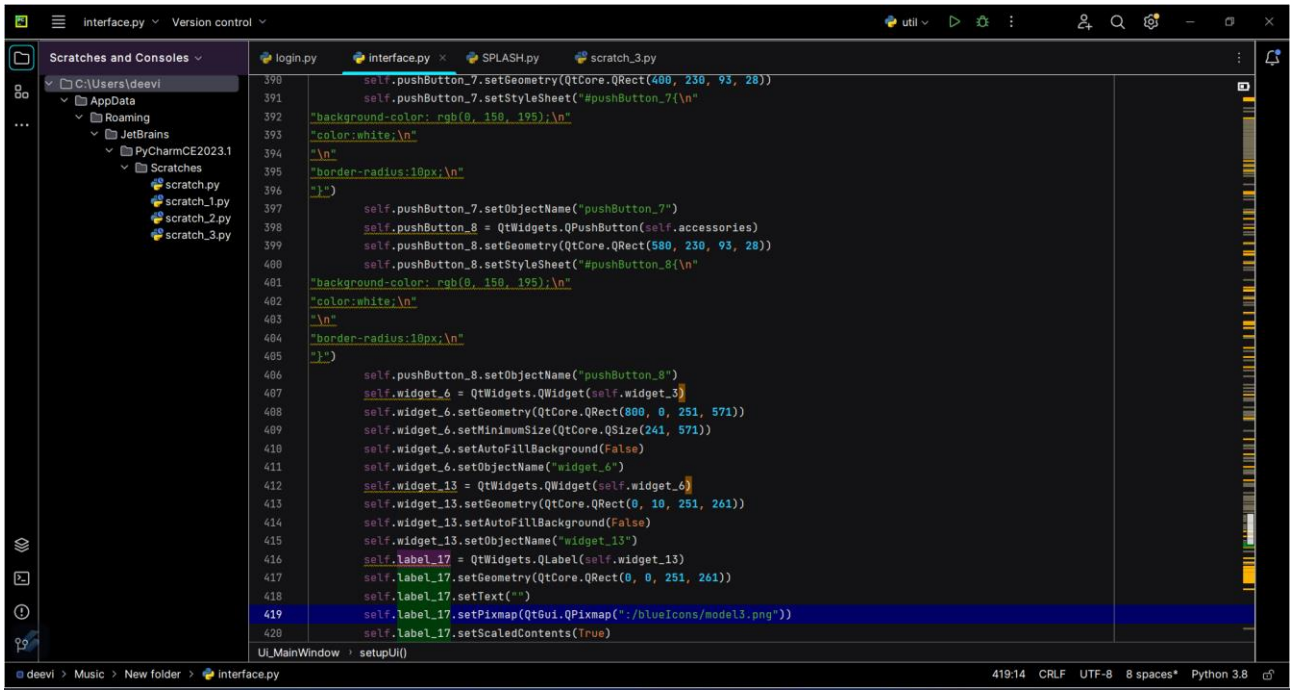

interface.py
Version control
util
Scratches and Consoles
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
login.py
interface.py
SPLASH.py
scratch_3.py
self.label_8.setScaledContents(True)
self.label_8.setObjectName("label_8")
self.widget_4 = QtWidgets.QWidget(self.glasses)
self.widget_4.setGeometry(QtCore.QRect(370, 50, 141, 171))
self.widget_4.setObjectName("widget_4")
self.label_9 = QtWidgets.QLabel(self.widget_4)
self.label_9.setGeometry(QtCore.QRect(0, 20, 140, 141))
self.label_9.setText("")
self.label_9.setPixmap(QtGui.QPixmap(":/blueIcons/glass2.png"))
self.label_9.setScaledContents(True)
self.label_9.setObjectName("label_9")
self.label_2 = QtWidgets.QLabel(self.glasses)
self.label_2.setGeometry(QtCore.QRect(40, 10, 151, 20))
font = QtGui.QFont()
font.setPointSize(12)
font.setBold(True)
font.setWeight(75)
self.label_2.setFont(font)
self.label_2.setObjectName("label_2")
self.pushButton = QtWidgets.QPushButton(self.glasses)
self.pushButton.setGeometry(QtCore.QRect(50, 240, 93, 28))
self.pushButton.setObjectName("pushButton")
self.pushButton_2 = QtWidgets.QPushButton(self.glasses)
self.pushButton_2.setGeometry(QtCore.QRect(220, 240, 93, 28))
self.pushButton_2.setObjectName("pushButton_2")
self.pushButton_3 = QtWidgets.QPushButton(self.glasses)
self.pushButton_3.setGeometry(QtCore.QRect(400, 240, 93, 28))
self.pushButton_3.setObjectName("pushButton_3")
self.widget_8 = QtWidgets.QWidget(self.glasses)
self.widget_8.setGeometry(QtCore.QRect(540, 50, 141, 171))
self.widget_8.setStyleSheet("#widget_8{\n"
Ui_MainWindow
setupUi()
deevi
Music
New folder
interface.py
281:14
CRLF
UTF-8
8 spaces*
Python 3.8

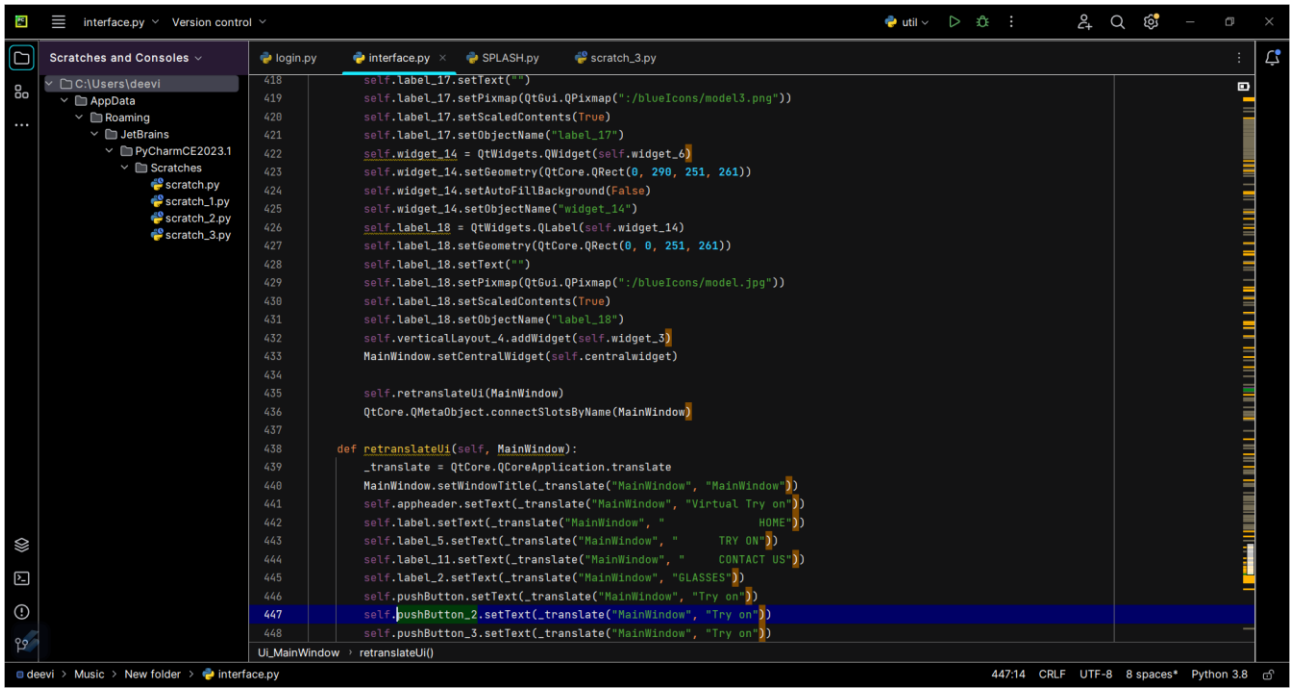

interface.py
Version control
util
Scratches and Consoles
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
login.py
interface.py
SPLASH.py
scratch_3.py
281 self.widget_8.setStyleSheet("#widget_8{\n"
282 "background-color: rgb(255, 255, 255);\n"
283 "border-radius:25px;}")
284 self.widget_8.setObjectName("widget_8")
285 self.label_10 = QtWidgets.QLabel(self.widget_8)
286 self.label_10.setGeometry(QtCore.QRect(0, 20, 140, 141))
287 self.label_10.setText("")
288 self.label_10.setPixmap(QtGui.QPixmap(":/blueIcons/glass3.png"))
289 self.label_10.setScaledContents(True)
290 self.label_10.setObjectName("label_10")
291 self.pushButton_4 = QtWidgets.QPushButton(self.glasses)
292 self.pushButton_4.setGeometry(QtCore.QRect(560, 240, 93, 28))
293 self.pushButton_4.setAutoFillBackground(False)
294 self.pushButton_4.setStyleSheet("#pushButton_4{\n"
295 "background-color: rgb(0, 150, 195);\n"
296 "color:white;\n"
297 "\n"
298 "border-radius:10px;\n"
299 "}")
300 self.pushButton_4.setObjectName("pushButton_4")
301 self.accessories = QtWidgets.QFrame(self.widget_3)
302 self.accessories.setGeometry(QtCore.QRect(0, 280, 791, 271))
303 sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Expanding)
304 sizePolicy.setHorizontalStretch(0)
305 sizePolicy.setVerticalStretch(0)
306 sizePolicy.setHeightForWidth(self.accessories.sizePolicy().hasHeightForWidth())
307 self.accessories.setSizePolicy(sizePolicy)
308 self.accessories.setMaximumSize(QtCore.QSize(791, 271))
309 self.accessories.setAutoFillBackground(False)
310 self.accessories.setFrameShape(QtWidgets.QFrame.StyledPanel)
311 self.accessories.setFrameShadow(QtWidgets.QFrame.Raised)
Ui_MainWindow › setupUi()
deevi › Music › New folder › interface.py
309:14 CRLF UTF-8 8 spaces* Python 3.8

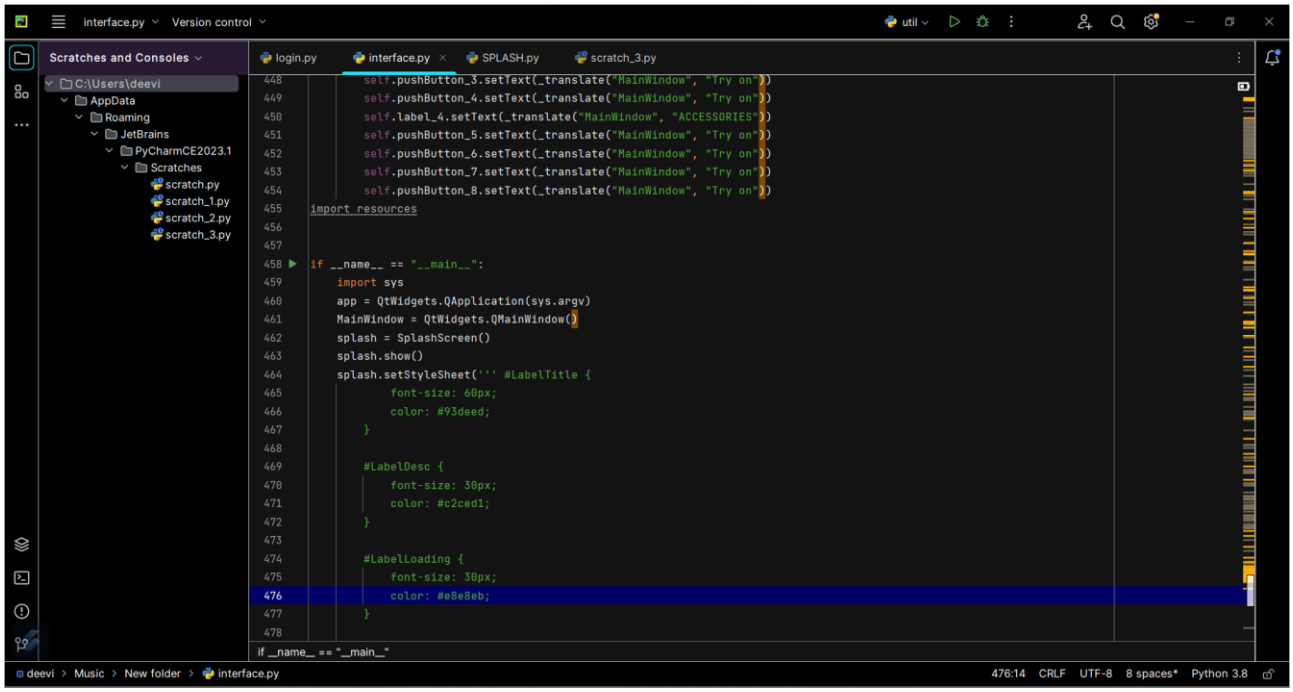

interface.py
Version control
util
Scratches and Consoles
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
login.py
interface.py
SPLASH.py
scratch_3.py
308 self.accessories.setMaximumSize(QtCore.QSize(791, 271))
309 self.accessories.setAutoFillBackground(False)
310 self.accessories.setFrameShape(QtWidgets.QFrame.StyledPanel)
311 self.accessories.setFrameShadow(QtWidgets.QFrame.Raised)
312 self.accessories.setObjectName("accessories")
313 self.label_4 = QtWidgets.QLabel(self.accessories)
314 self.label_4.setGeometry(QtCore.QRect(20, 10, 151, 16))
315 font = QtGui.QFont()
316 font.setPointSize(12)
317 font.setBold(True)
318 font.setWeight(75)
319 self.label_4.setFont(font)
320 self.label_4.setObjectName("label_4")
321 self.widget_9 = QtWidgets.QWidget(self.accessories)
322 self.widget_9.setGeometry(QtCore.QRect(30, 40, 140, 171))
323 self.widget_9.setMinimumSize(QtCore.QSize(140, 171))
324 self.widget_9.setAutoFillBackground(False)
325 self.widget_9.setStyleSheet("#widget_9{\n"
326 "background-color: rgb(255, 255, 255);\n"
327 "border-radius:25px;}")
328 self.widget_9.setObjectName("widget_9")
329 self.label_13 = QtWidgets.QLabel(self.widget_9)
330 self.label_13.setGeometry(QtCore.QRect(0, 20, 140, 141))
331 self.label_13.setText("")
332 self.label_13.setPixmap(QtGui.QPixmap(":/blueIcons/necklac1.png"))
333 self.label_13.setScaledContents(True)
334 self.label_13.setObjectName("label_13")
335 self.widget_10 = QtWidgets.QWidget(self.accessories)
336 self.widget_10.setGeometry(QtCore.QRect(210, 40, 141, 171))
337 self.widget_10.setStyleSheet("#widget_10{\n"
338 "background-color: rgb(255, 255, 255);\n"
Ui_MainWindow › setupUi()
deevi › Music › New folder › interface.py
337:14 CRLF UTF-8 8 spaces* Python 3.8

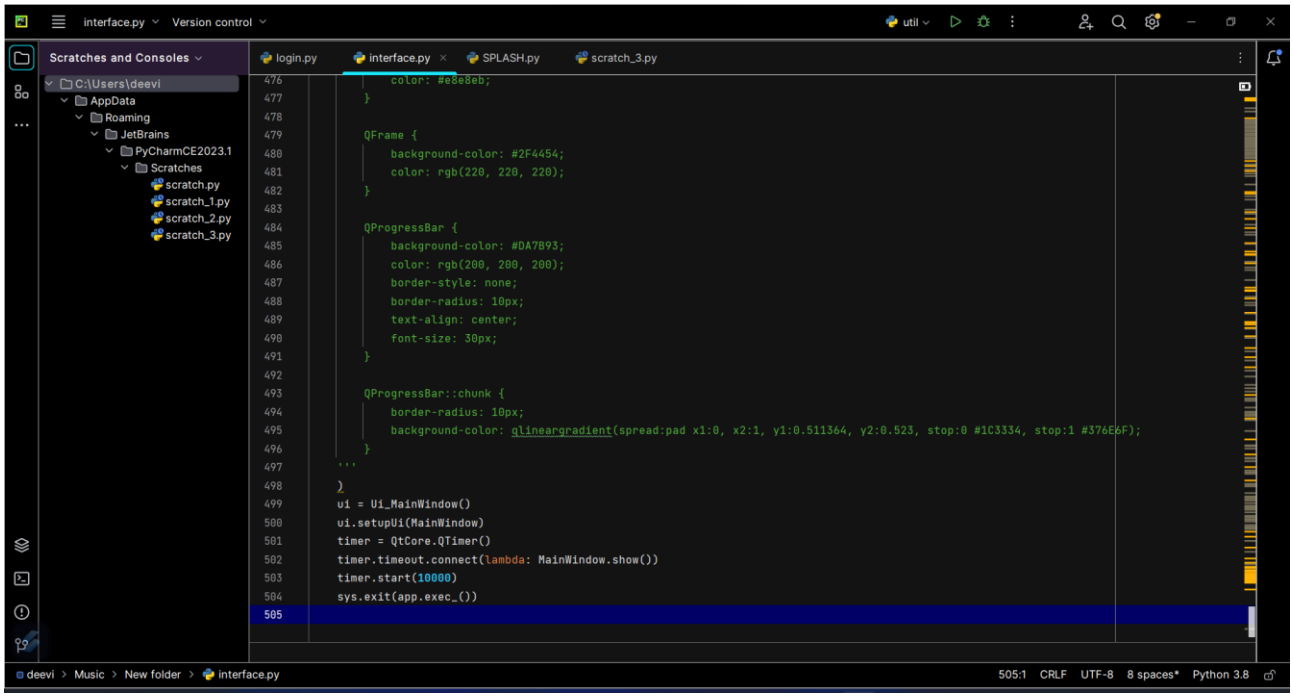

interface.py
Version control
util
Scratches and Consoles
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
login.py
interface.py
SPLASH.py
scratch_3.py
self.widget_10.setStyleSheet("#widget_10{\n"
"background-color: rgb(255, 255, 255);\n"
"border-radius:25px;}")
self.widget_10.setObjectName("widget_10")
self.label_14 = QtWidgets.QLabel(self.widget_10)
self.label_14.setGeometry(QtCore.QRect(0, 20, 140, 141))
self.label_14.setText("")
self.label_14.setPixmap(QtGui.QPixmap(":/blueIcons/necklace 2.png"))
self.label_14.setScaledContents(True)
self.label_14.setObjectName("label_14")
self.widget_11 = QtWidgets.QWidget(self.accessories)
self.widget_11.setGeometry(QtCore.QRect(380, 40, 141, 171))
self.widget_11.setStyleSheet("#widget_11{\n"
"background-color: rgb(255, 255, 255);\n"
"border-radius:25px;}")
self.widget_11.setObjectName("widget_11")
self.label_15 = QtWidgets.QLabel(self.widget_11)
self.label_15.setGeometry(QtCore.QRect(0, 20, 140, 141))
self.label_15.setText("")
self.label_15.setPixmap(QtGui.QPixmap(":/blueIcons/necklace 3.png"))
self.label_15.setScaledContents(True)
self.label_15.setObjectName("label_15")
self.widget_12 = QtWidgets.QWidget(self.accessories)
self.widget_12.setGeometry(QtCore.QRect(550, 40, 141, 171))
self.widget_12.setStyleSheet("#widget_12{\n"
"background-color: rgb(255, 255, 255);\n"
"border-radius:25px;}")
self.widget_12.setObjectName("widget_12")
self.label_16 = QtWidgets.QLabel(self.widget_12)
self.label_16.setGeometry(QtCore.QRect(0, 20, 140, 141))
self.label_16.setText("")
Ui_MainWindow
setupUi()
deevi
Music
New folder
interface.py
363:14
CRLF
UTF-8
8 spaces*
Python 3.8


interface.py
Version control
util
Scratches and Consoles
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
login.py
interface.py
SPLASH.py
scratch_3.py
"background-color: rgb(255, 255, 255);\n"
"border-radius:25px;}")
self.widget_12.setObjectName("widget_12")
self.label_16 = QtWidgets.QLabel(self.widget_12)
self.label_16.setGeometry(QtCore.QRect(0, 20, 140, 141))
self.label_16.setText("")
self.label_16.setPixmap(QtGui.QPixmap(":/blueIcons/necklace 4.png"))
self.label_16.setScaledContents(True)
self.label_16.setObjectName("label_16")
self.pushButton_5 = QtWidgets.QPushButton(self.accessories)
self.pushButton_5.setGeometry(QtCore.QRect(50, 230, 93, 28))
self.pushButton_5.setStyleSheet("#pushButton_5{\n"
"background-color: rgb(0, 150, 195);\n"
"color:white;\n"
"\n"
"border-radius:10px;\n"
"}")
self.pushButton_5.setObjectName("pushButton_5")
self.pushButton_6 = QtWidgets.QPushButton(self.accessories)
self.pushButton_6.setGeometry(QtCore.QRect(230, 230, 93, 28))
self.pushButton_6.setStyleSheet("#pushButton_6{\n"
"background-color: rgb(0, 150, 195);\n"
"color:white;\n"
"\n"
"border-radius:10px;\n"
"}")
self.pushButton_6.setObjectName("pushButton_6")
self.pushButton_7 = QtWidgets.QPushButton(self.accessories)
self.pushButton_7.setGeometry(QtCore.QRect(400, 230, 93, 28))
self.pushButton_7.setStyleSheet("#pushButton_7{\n"
"background-color: rgb(0, 150, 195);\n"
Ui_MainWindow
setupUi()
deevi
Music
New folder
interface.py
391:14
CRLF
UTF-8
8 spaces*
Python 3.8

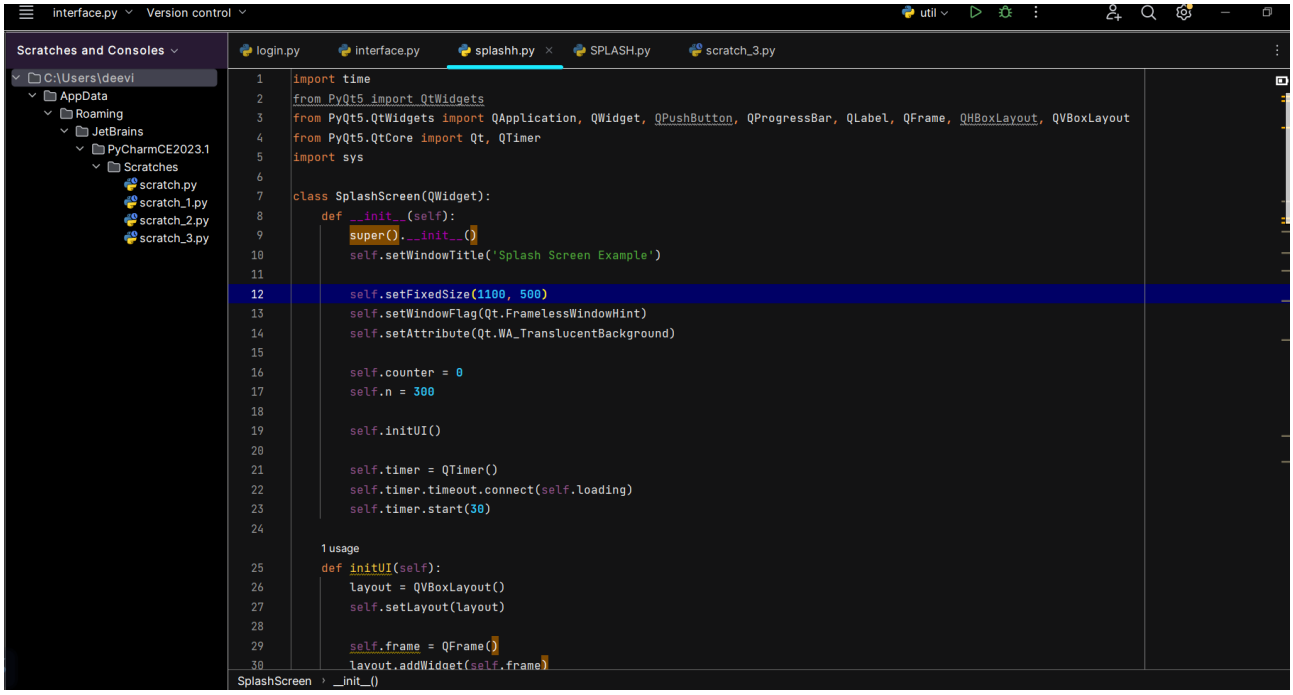

interface.py
Version control
util
Scratches and Consoles
login.py
interface.py
SPLASH.py
scratch_3.py
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
self.pushButton_7.setGeometry(QtCore.QRect(400, 230, 93, 28))
self.pushButton_7.setStyleSheet("#pushButton_7{\n"
"background-color: rgb(0, 150, 195);\n"
"color:white;\n"
"\n"
"border-radius:10px;\n"
"}")
self.pushButton_7.setObjectName("pushButton_7")
self.pushButton_8 = QtWidgets.QPushButton(self.accessories)
self.pushButton_8.setGeometry(QtCore.QRect(580, 230, 93, 28))
self.pushButton_8.setStyleSheet("#pushButton_8{\n"
"background-color: rgb(0, 150, 195);\n"
"color:white;\n"
"\n"
"border-radius:10px;\n"
"}")
self.pushButton_8.setObjectName("pushButton_8")
self.widget_6 = QtWidgets.QWidget(self.widget_3)
self.widget_6.setGeometry(QtCore.QRect(800, 0, 251, 571))
self.widget_6.setMinimumSize(QtCore.QSize(241, 571))
self.widget_6.setAutoFillBackground(False)
self.widget_6.setObjectName("widget_6")
self.widget_13 = QtWidgets.QWidget(self.widget_6)
self.widget_13.setGeometry(QtCore.QRect(0, 10, 251, 261))
self.widget_13.setAutoFillBackground(False)
self.widget_13.setObjectName("widget_13")
self.label_17 = QtWidgets.QLabel(self.widget_13)
self.label_17.setGeometry(QtCore.QRect(0, 0, 251, 261))
self.label_17.setText("")
self.label_17.setPixmap(QtGui.QPixmap(":/blueIcons/model3.png"))
self.label_17.setScaledContents(True)
Ui_MainWindow › setupUi()
deevi › Music › New folder › interface.py
419:14 CRLF UTF-8 8 spaces* Python 3.8


interface.py
Version control
util
Scratches and Consoles
login.py
interface.py
SPLASH.py
scratch_3.py
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
self.label_17.setText("")
self.label_17.setPixmap(QtGui.QPixmap(":/blueIcons/model3.png"))
self.label_17.setScaledContents(True)
self.label_17.setObjectName("label_17")
self.widget_14 = QtWidgets.QWidget(self.widget_6)
self.widget_14.setGeometry(QtCore.QRect(0, 290, 251, 261))
self.widget_14.setAutoFillBackground(False)
self.widget_14.setObjectName("widget_14")
self.label_18 = QtWidgets.QLabel(self.widget_14)
self.label_18.setGeometry(QtCore.QRect(0, 0, 251, 261))
self.label_18.setText("")
self.label_18.setPixmap(QtGui.QPixmap(":/blueIcons/model.jpg"))
self.label_18.setScaledContents(True)
self.label_18.setObjectName("label_18")
self.verticalLayout_4.addWidget(self.widget_3)
MainWindow.setCentralWidget(self.centralwidget)
self.retranslateUi(MainWindow)
QtCore.QMetaObject.connectSlotsByName(MainWindow)
def retranslateUi(self, MainWindow):
_translate = QtCore.QCoreApplication.translate
MainWindow.setWindowTitle(_translate("MainWindow", "MainWindow"))
self.appheader.setText(_translate("MainWindow", "Virtual Try on"))
self.label.setText(_translate("MainWindow", "HOME"))
self.label_5.setText(_translate("MainWindow", "TRY ON"))
self.label_11.setText(_translate("MainWindow", "CONTACT US"))
self.label_2.setText(_translate("MainWindow", "GLASSES"))
self.pushButton.setText(_translate("MainWindow", "Try on"))
self.pushButton_2.setText(_translate("MainWindow", "Try on"))
self.pushButton_3.setText(_translate("MainWindow", "Try on"))
Ui_MainWindow › retranslateUi()
deevi › Music › New folder › interface.py
447:14 CRLF UTF-8 8 spaces* Python 3.8

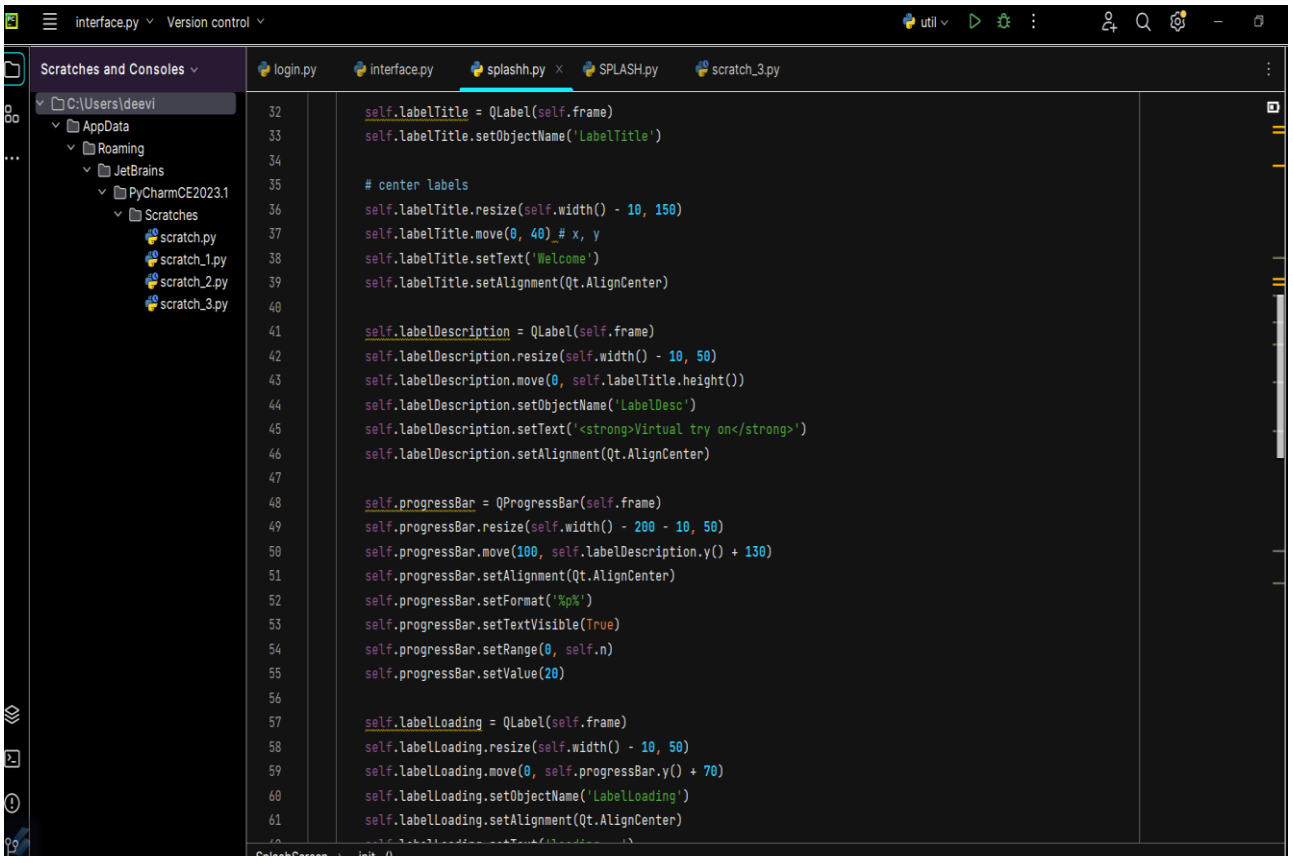

interface.py
Version control
util
Scratches and Consoles
login.py
interface.py
SPLASH.py
scratch_3.py
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
self.pushButton_3.setText(_translate("MainWindow", "Try on"))
self.pushButton_4.setText(_translate("MainWindow", "Try on"))
self.label_4.setText(_translate("MainWindow", "ACCESSORIES"))
self.pushButton_5.setText(_translate("MainWindow", "Try on"))
self.pushButton_6.setText(_translate("MainWindow", "Try on"))
self.pushButton_7.setText(_translate("MainWindow", "Try on"))
self.pushButton_8.setText(_translate("MainWindow", "Try on"))
import resources
if __name__ == "__main__":
import sys
app = QtWidgets.QApplication(sys.argv)
MainWindow = QtWidgets.QMainWindow()
splash = SplashScreen()
splash.show()
splash.setStyleSheet(''' #LabelTitle {
font-size: 60px;
color: #93deed;
}
#LabelDesc {
font-size: 30px;
color: #c2ced1;
}
#LabelLoading {
font-size: 30px;
color: #e8e8eb;
}
if __name__ == "__main__"
deevi
Music
New folder
interface.py
476:14
CRLF
UTF-8
8 spaces*
Python 3.8


interface.py
Version control
util
Scratches and Consoles
login.py
interface.py
SPLASH.py
scratch_3.py
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
color: #e8e8eb;
}
QFrame {
background-color: #2F4454;
color: rgb(220, 220, 220);
}
QProgressBar {
background-color: #DA7B93;
color: rgb(200, 200, 200);
border-style: none;
border-radius: 10px;
text-align: center;
font-size: 30px;
}
QProgressBar::chunk {
border-radius: 10px;
background-color: qlineargradient(spread:pad x1:0, x2:1, y1:0.511364, y2:0.523, stop:0 #1C3334, stop:1 #376E6F);
}
'''
)
ui = Ui_MainWindow()
ui.setupUi(MainWindow)
timer = QtCore.QTimer()
timer.timeout.connect(lambda: MainWindow.show())
timer.start(10000)
sys.exit(app.exec_())
deevi
Music
New folder
interface.py
505:1
CRLF
UTF-8
8 spaces*
Python 3.8

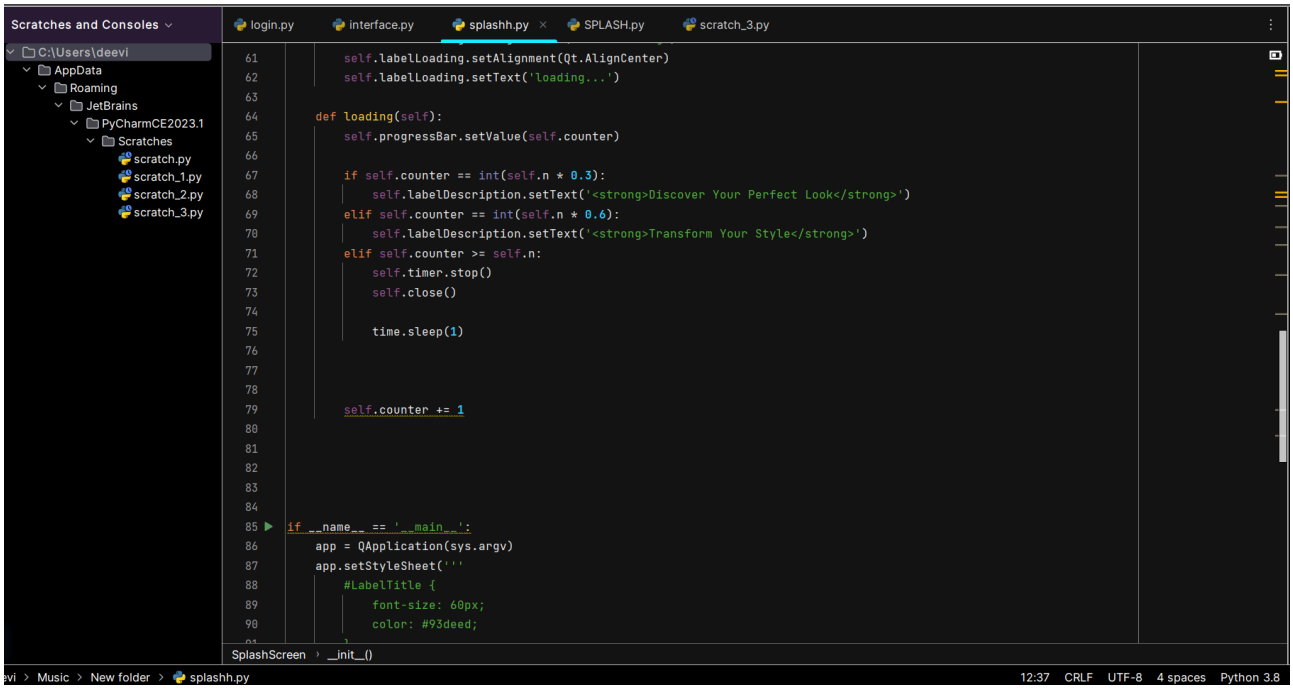
@Splashscreen

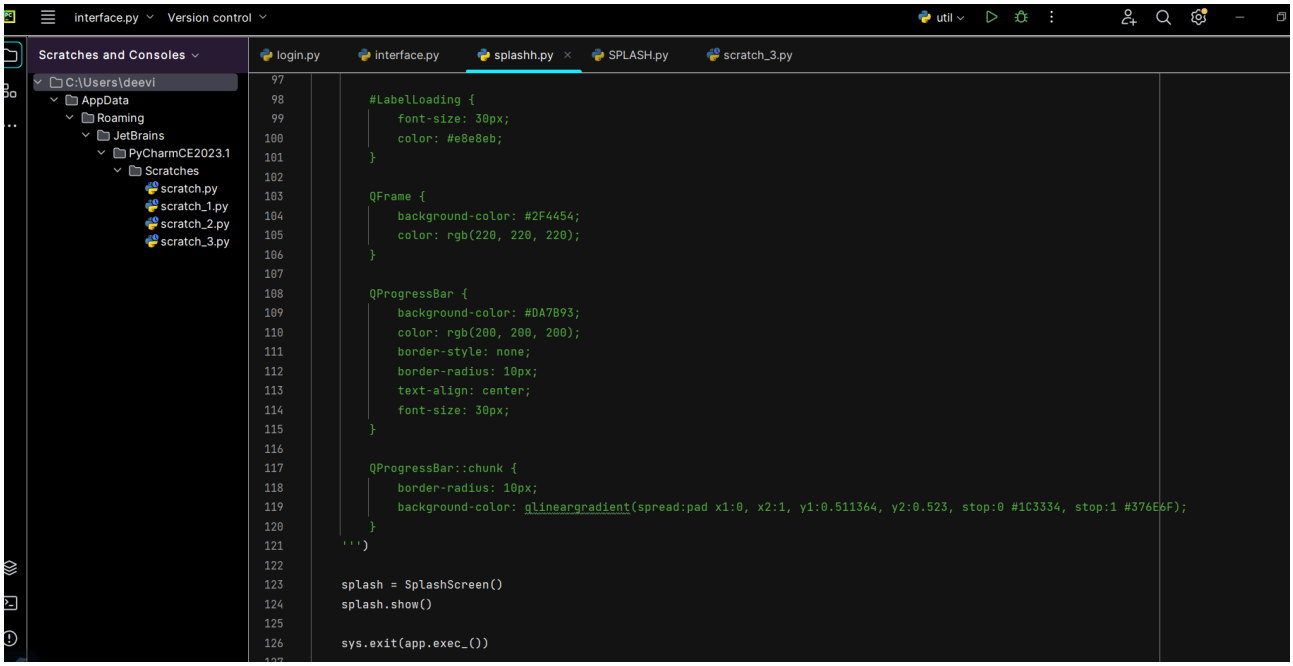


```python
import time
from PyQt5 import QtWidgets
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton, QProgressBar, QLabel, QFrame, QHBoxLayout, QVBoxLayout
from PyQt5.QtCore import Qt, QTimer
import sys

class SplashScreen(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle('Splash Screen Example')

        self.setFixedSize(1100, 500)
        self.setWindowFlag(Qt.FramelessWindowHint)
        self.setAttribute(Qt.WA_TranslucentBackground)

        self.counter = 0
        self.n = 300

        self.initUI()

        self.timer = QTimer()
        self.timer.timeout.connect(self.loading)
        self.timer.start(30)

    1 usage
    def initUI(self):
        layout = QVBoxLayout()
        self.setLayout(layout)

        self.frame = QFrame()
        layout.addWidget(self.frame)
```



```python
        self.labelTitle = QLabel(self.frame)
        self.labelTitle.setObjectName('LabelTitle')

        # center labels
        self.labelTitle.resize(self.width() - 10, 150)
        self.labelTitle.move(0, 40) # x, y
        self.labelTitle.setText('Welcome')
        self.labelTitle.setAlignment(Qt.AlignCenter)

        self.labelDescription = QLabel(self.frame)
        self.labelDescription.resize(self.width() - 10, 50)
        self.labelDescription.move(0, self.labelTitle.height())
        self.labelDescription.setObjectName('LabelDesc')
        self.labelDescription.setText('<strong>Virtual try on</strong>')
        self.labelDescription.setAlignment(Qt.AlignCenter)

        self.progressBar = QProgressBar(self.frame)
        self.progressBar.resize(self.width() - 200 - 10, 50)
        self.progressBar.move(100, self.labelDescription.y() + 130)
        self.progressBar.setAlignment(Qt.AlignCenter)
        self.progressBar.setFormat('%p%')
        self.progressBar.setTextVisible(True)
        self.progressBar.setRange(0, self.n)
        self.progressBar.setValue(20)

        self.labelLoading = QLabel(self.frame)
        self.labelLoading.resize(self.width() - 10, 50)
        self.labelLoading.move(0, self.progressBar.y() + 70)
        self.labelLoading.setObjectName('LabelLoading')
        self.labelLoading.setAlignment(Qt.AlignCenter)
```


Scratches and Consoles
login.py
interface.py
splashh.py
SPLASH.py
scratch_3.py
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
self.labelLoading.setAlignment(Qt.AlignCenter)
self.labelLoading.setText('loading...')
def loading(self):
self.progressBar.setValue(self.counter)
if self.counter == int(self.n * 0.3):
self.labelDescription.setText('Discover Your Perfect Look')
elif self.counter == int(self.n * 0.6):
self.labelDescription.setText('Transform Your Style')
elif self.counter >= self.n:
self.timer.stop()
self.close()
time.sleep(1)
self.counter += 1
if __name__ == '__main__':
app = QApplication(sys.argv)
app.setStyleSheet('''
#LabelTitle {
font-size: 60px;
color: #93deed;
SplashScreen
__init__()
Music
New folder
splashh.py
12:37
CRLF
UTF-8
4 spaces
Python 3.8

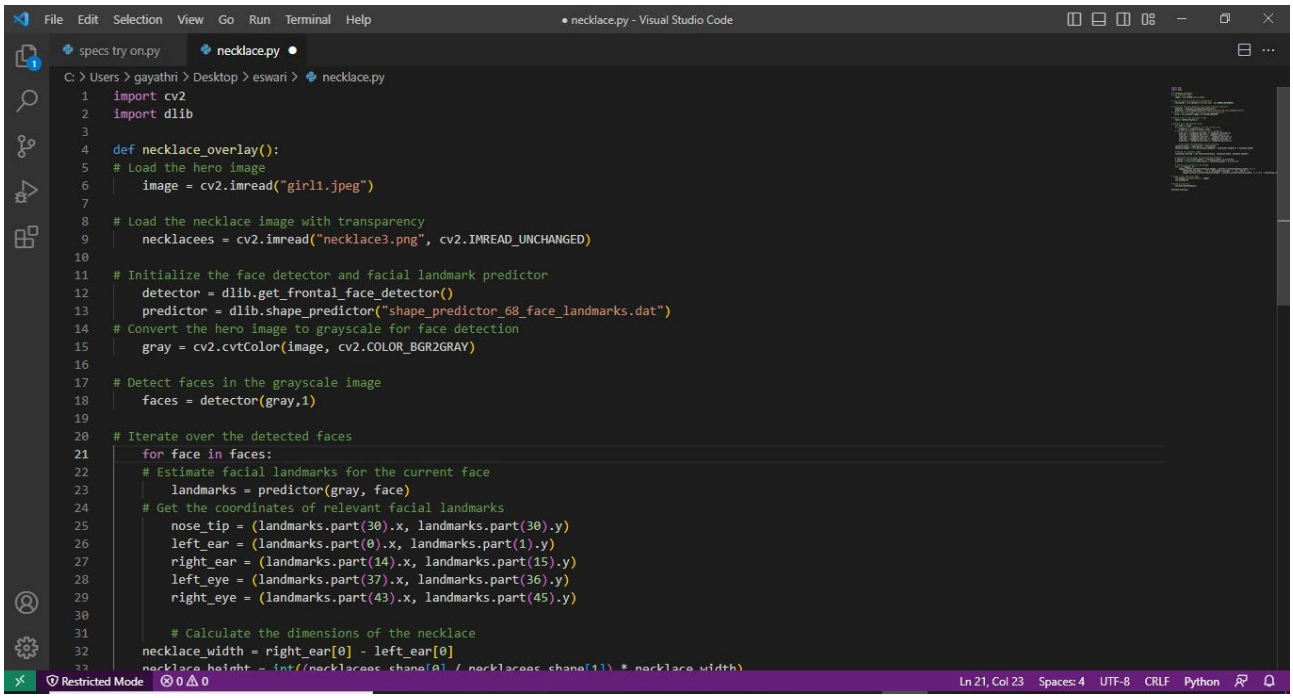

interface.py
Version control
util
Scratches and Consoles
login.py
interface.py
splashh.py
SPLASH.py
scratch_3.py
C:\Users\deevi
AppData
Roaming
JetBrains
PyCharmCE2023.1
Scratches
scratch.py
scratch_1.py
scratch_2.py
scratch_3.py
#LabelLoading {
font-size: 30px;
color: #e8e8eb;
}
QFrame {
background-color: #2F4454;
color: rgb(220, 220, 220);
}
QProgressBar {
background-color: #DA7B93;
color: rgb(200, 200, 200);
border-style: none;
border-radius: 10px;
text-align: center;
font-size: 30px;
}
QProgressBar::chunk {
border-radius: 10px;
background-color: qlineargradient(spread:pad x1:0, x2:1, y1:0.511364, y2:0.523, stop:0 #1C3334, stop:1 #376E6F);
}
''')
splash = SplashScreen()
splash.show()
sys.exit(app.exec_())

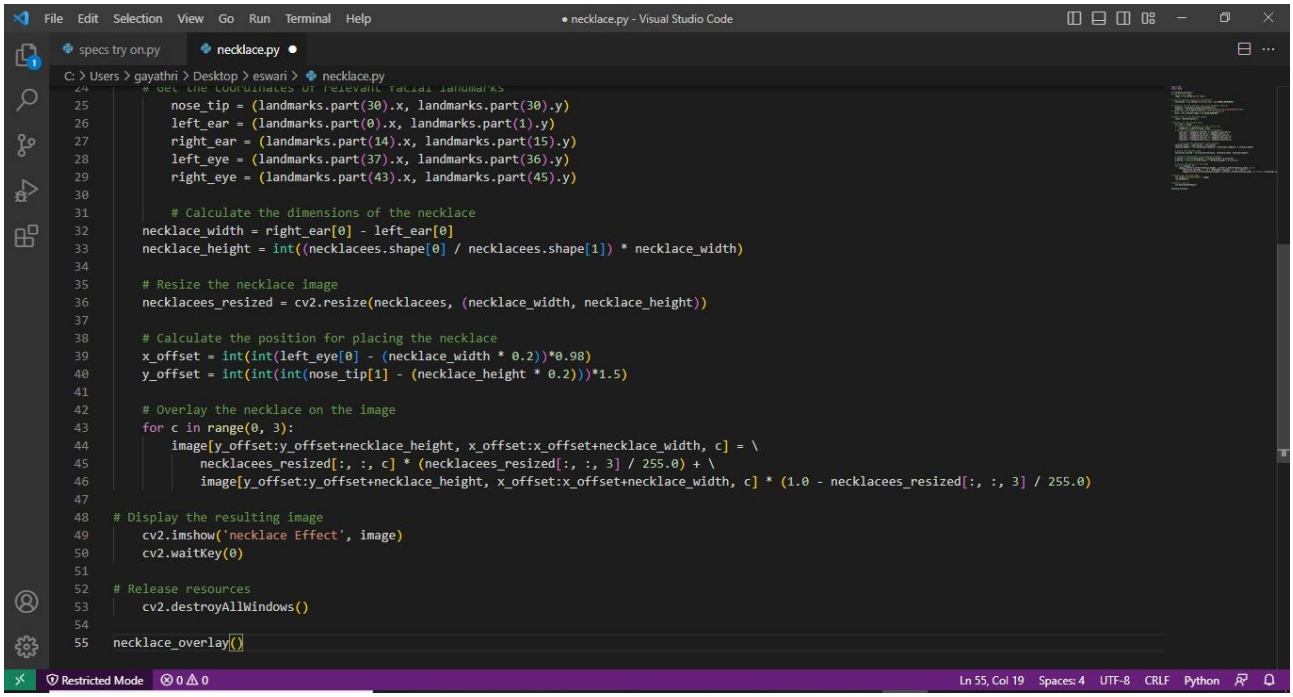
@Spectacle Try On

```
> Users > gayathri > Desktop > eswari > specs try on.py
import cv2
import dlib


image = cv2.imread("lee.jpg",cv2.IMREAD_COLOR)

sunglasses = cv2.imread("glass2.png", cv2.IMREAD_UNCHANGED)

detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Detect faces in the grayscale image
faces = detector(gray,1)

# Iterate over the detected faces
for face in faces:
    # Estimate facial landmarks for the current face
    landmarks = predictor(gray, face)
    # Get the coordinates of relevant facial landmarks
    nose_tip = (landmarks.part(29).x, landmarks.part(29).y)
    left_ear = (landmarks.part(0).x, landmarks.part(0).y)

    right_ear = (landmarks.part(16).x, landmarks.part(16).y)
    left_eye = (landmarks.part(36).x, landmarks.part(36).y)
    right_eye = (landmarks.part(45).x, landmarks.part(45).y)

```

```
    # Calculate the dimensions of the sunglasses
    sunglass_width = right_ear[0] - left_ear[0]
    sunglass_height = int((sunglasses.shape[0] / sunglasses.shape[1]) * sunglass_width)

    # Resize the sunglasses image
    sunglasses_resized = cv2.resize(sunglasses, (sunglass_width, sunglass_height))

    # Calculate the position for placing the sunglasses
    x_offset = int(left_eye[0] - (sunglass_width * 0.18))
    y_offset = int(nose_tip[1] - (sunglass_height * 0.7))

    # Overlay the sunglasses on the image
    for c in range(0, 3):
        image[y_offset:y_offset+sunglass_height, x_offset:x_offset+sunglass_width, c] = \
            sunglasses_resized[:, :, c] * (sunglasses_resized[:, :, 3] / 255.0) + \
            image[y_offset:y_offset+sunglass_height, x_offset:x_offset+sunglass_width, c] * (1.0 - sunglasses_resized[:, :, 3] / 255.0)

# Display the resulting image
cv2.imshow('Sunglass Effect', image)
cv2.waitKey(0)

# Release resources
cv2.destroyAllWindows()
```

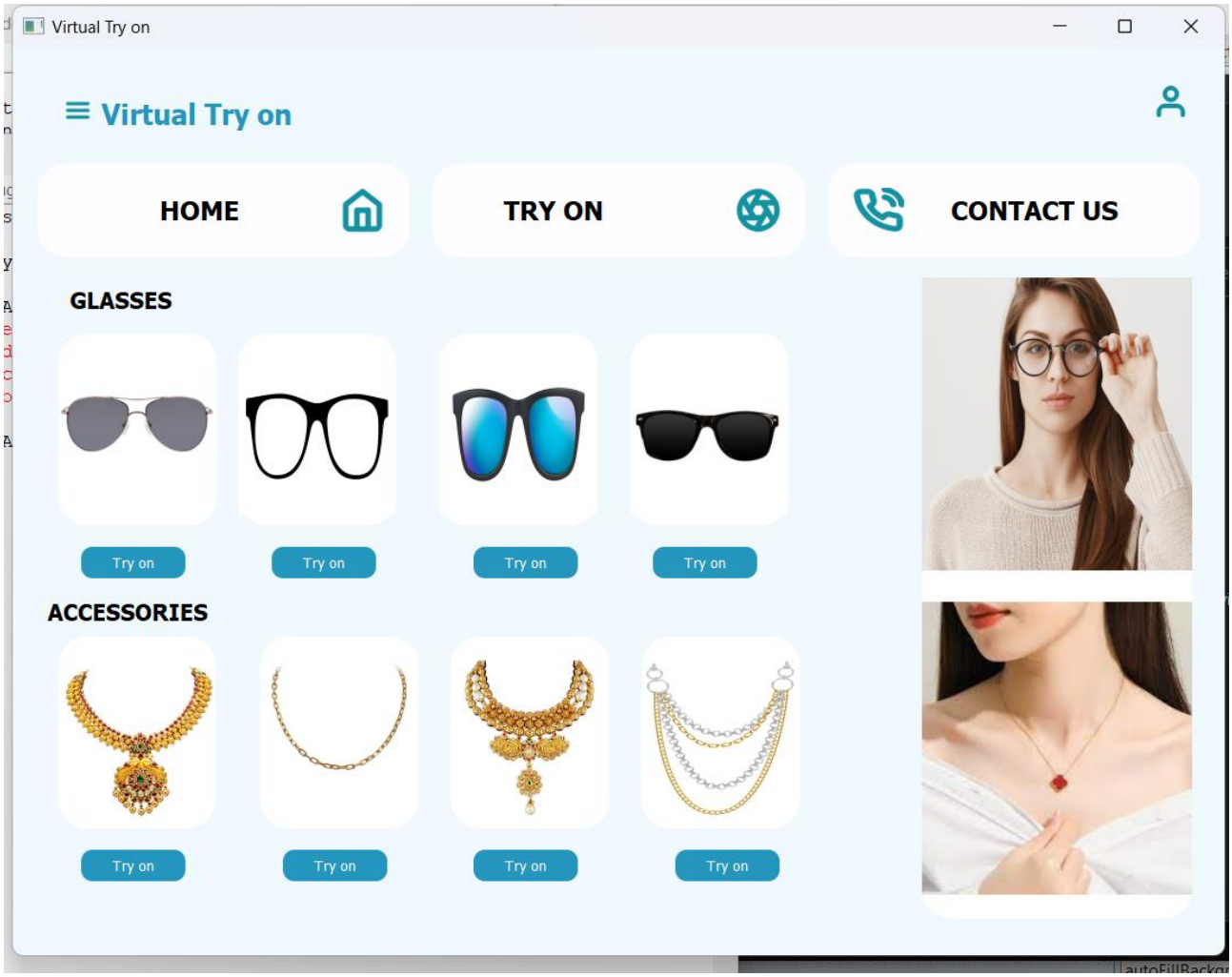
@Necklace Try On



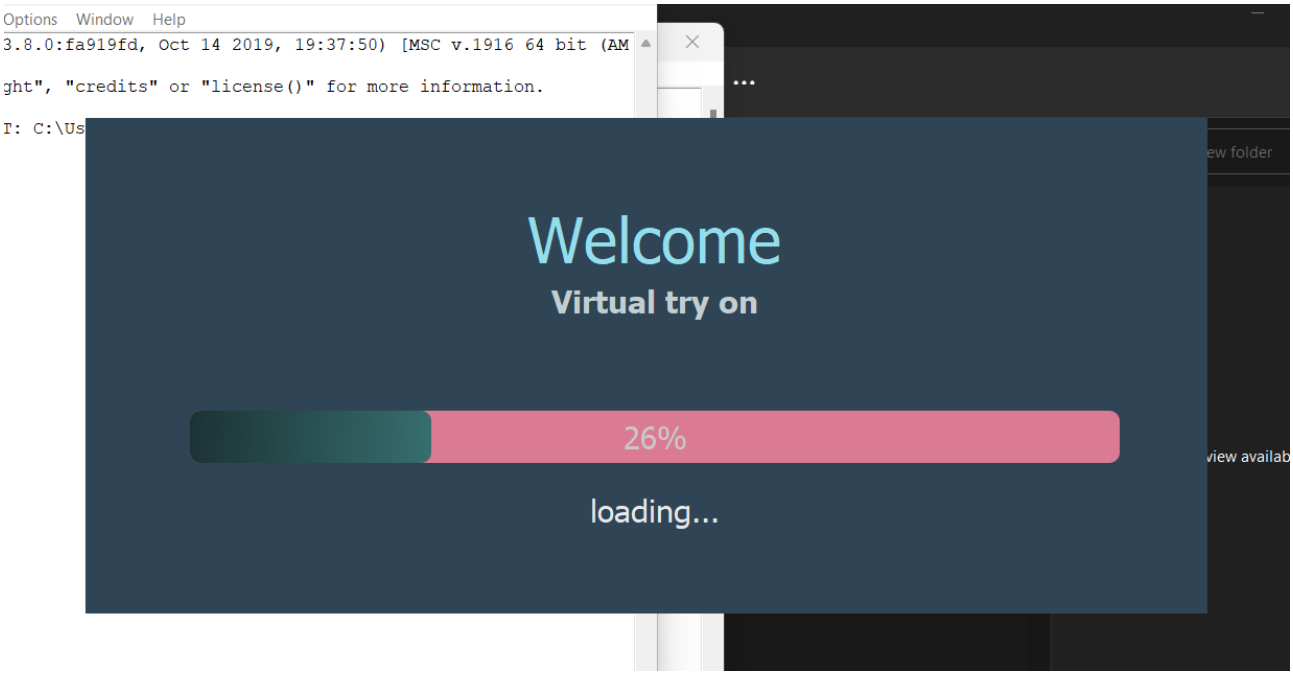


5.OUTPUT SCREENSHOTS

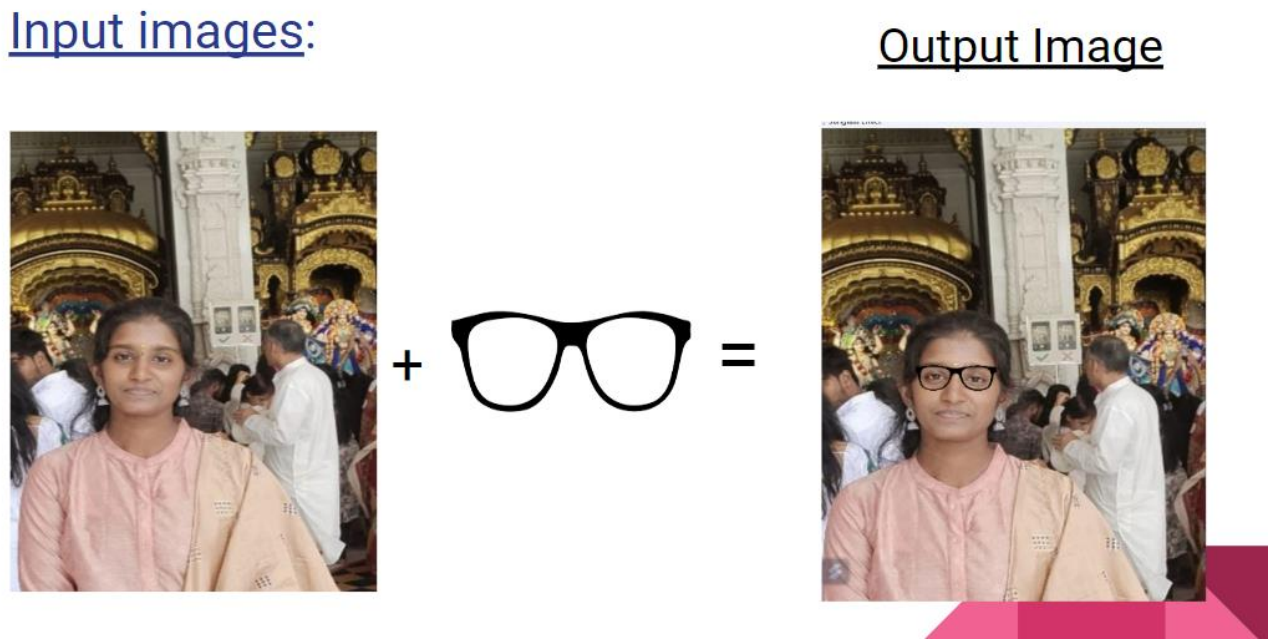
@UserInterface



@SplashScreen



@Try On Glasses Output



@Necklace Try On

6.CONCLUSION

In conclusion, virtual try-on using 2D image overlaying is a game changer for online shopping of glasses and necklaces. It helps solve the problem of buying these items without trying them on first, and provides an easy and convenient way to do so from the comfort of your own home.

By using this technology, you can save time and money, and have a better shopping experience overall. So next time you're in the market for new glasses or necklaces, give virtual try-on using 2D image overlaying a try. You won't be disappointed!

REFERENCES:

https://chat.openai.com/

https://contribute.qt-project.org/

https://www.qt.io/

www.google.comm

www.youtube.com (Code First with Hala, Jie Jenn)