**Lab Exercise 19- Vertical Translation Animation using PySide6**

To create a vertical translation animation in a straight line using PySide, you can modify the QPoint values to move the widget vertically. Here is an example demonstrating the vertical translation animation:

import sys

from PySide2.QtWidgets import QApplication, QMainWindow, QPushButton

from PySide2.QtCore import QPoint, QPropertyAnimation

class MainWindow(QMainWindow):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.setGeometry(100, 100, 300, 200)

self.setWindowTitle('PySide Vertical Translation Animation')

self.button = QPushButton('Animate Me', self)

self.button.setGeometry(50, 50, 100, 30)

self.button.clicked.connect(self.start\_animation)

def start\_animation(self):

start\_point = self.button.pos()

end\_point = QPoint(self.button.x(), 150)

self.anim = QPropertyAnimation(self.button, b"pos")

self.anim.setDuration(1000)

self.anim.setStartValue(start\_point)

self.anim.setEndValue(end\_point)

self.anim.start()

if \_\_name\_\_ == '\_\_main\_\_':

app = QApplication(sys.argv)

window = MainWindow()

window.show()

sys.exit(app.exec\_())

This code creates a PySide application with a button that, when clicked, triggers a vertical translation animation. The button moves vertically from its initial position to the target position within the window.

Save the code to a file named vertical\_animation.py and run it using the following command:

python vertical\_animation.py

Upon running the script, a window will appear with a button. Clicking the button will trigger the vertical translation animation, moving the button vertically from its initial position to the target position. You can customize the animation by modifying the coordinates and duration in the setStartValue, setEndValue, and setDuration methods according to your requirements.