Before you learn to convert a Tab widget into a Tool Box or Stacked widget, let's explain the two terms:

Tool Box: A Tool Box is an instance of the QToolBoxclass and provides a column of tabbed widget items, one above the next. The widgets of the current tab are displayed below it.

Stacked Widget: A Stacked widget is an instance of QStackedWidgetand provides a stack of widgets where only one widget is visible at a time. Again, it can be used to display large chunks of information in the Tab widget. By default, the Stacked widget doesn't have a way to switch pages, so to switch pages, you must use another widget, such as a Combo Box or a List widget.

Converting a Tab Widget into a Tool Box

1. To convert a Tab widget to a Tool Box, right-click on it and select the **Morph Into** option

2. You will see two suboptions: QStackedWidget and QToolBox. Select QToolBox to convert the Tab widget into a Tool Box.

3. The Tab widget will be converted into a Tool Box widget

4. The Tab buttons of the Tab widget will change to a column of tabs, and the widgets inside each Tab button of the Tab widget will appear as widgets of the respective tabs in Tool Box.

5. The default text of each tab will be Page. Change the tab text to Food, Drinks using the **currentItemText** property.

6. Save the application with the name fileName.ui.

7. CONVERT .UI FILE INTO .PY (In the .py file is where all coding automatically takes place)

8. Create the .pyw with just the basic structure only creating the self.ui. Heres the structure:

```
import sys
from PyQt5.QtWidgets import QApplication, QDialog
from tabwidgettoolbox import Ui_Dialog

class MyForm(QDialog):
    def __init__(self, parent=None):
        super(MyForm, self).__init__(parent)
        self.ui = Ui_Dialog()
        self.ui.setupUi(self)
```

```
if __name__ == "__main__":
    app = QApplication(sys.argv)
    myapp = MyForm()
    myapp.show()
    sys.exit(app.exec_())
```

You don't need to add those unnecessary things of setText(), checked() etc…
Just the basic structure here of self.ui because the .py file from converting the .ui has the code automatically in it.