Homework 2

Software Engineering Principle

Software Engineering Program,

Department of Computer Engineering,

School of Engineering, KMITL

67011352 Theepakorn Phayonrat

Features

Gantt Chart

Used for planning project plan and tasks duration or deadline.

Class Diagram

Used for designing classes in the projects.

Interaction Diagram

Used for designing how classes interact each others in the projects.

Markdown Renderer for the task assignment page

How it works:

- As mentioned earlier, we can use markdown to express the task, $\therefore$ we need a markdown renderer.

Implementation Approach:

- Use QEngineWebView Module in PyQt.

VS-Code Extension (OPTIONAL)

`TODO` extension in VS-Code with better description for the task and with team member(s) assigned to that task.
How it works:

- If you have comment with `TODO` in the front, you can add description of the task in a different entry and also in a markdown file.
- If you want to add a person in charge for that task (OPTIONAL), you can use `@TEMP`, where `TEMP` can be either role or team member names.
- After saved, you can access the `TODO` description as you hover and click to inspect task in the comment.

Implementation Approach:

- Scan through the file looking for comment with `TODO` in the front then keep the entry into the DB.
- We can edit the `TODO` description inside a external markdown file.

Page included in this homework

- **Task Assignment Page**: Page to edit `TODO` for task assignments with markdown supported for better view. User can choose whether to edit in the manual mode or external text editor and save file because it can also fetch from real `.md` files in the real program.

Code:

TaskAssignmentPage.py

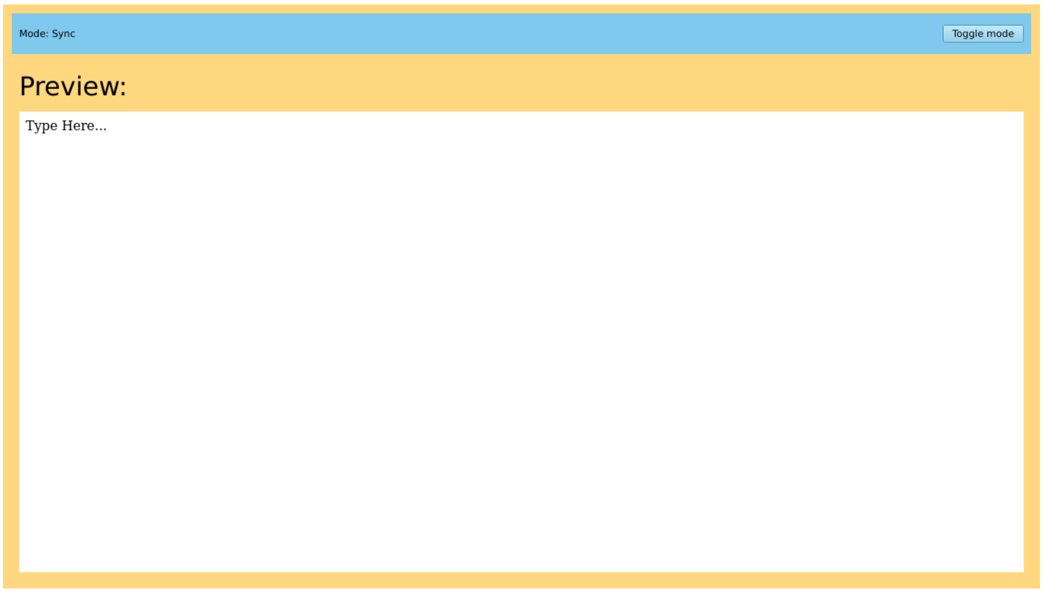
```python
import sys
from PySide6.QtWidgets import *
from PySide6.QtCore import *
from PySide6.QtWebEngineWidgets import *
import markdown as md

class TeskAssignmentPage(QWidget):
    def __init__(self) -> None:
        QWidget.__init__(self, None)
        v_box = QVBoxLayout()
        nav_bar = QHBoxLayout()

        self.cur_mode_label = QLabel(self)
        self.current_mode_text = "Sync"
        self.cur_mode_label.setText(f"Mode:
           ↪  {self.current_mode_text}")
        nav_w = QWidget()
        nav_w.setLayout(nav_bar)
        nav_bar.addWidget(self.cur_mode_label)
        nav_w.setFixedHeight(50)
        nav_w.setStyleSheet("""
            background-color: #7ec9ed;
        """)
        self.setWindowTitle("Task Assignment Page")
        self.resize(1280, 720)
        self.show()
        self.tab_stack = QStackedWidget()
        self.setStyleSheet("""
            background-color: #ffd77f;
        """)

        self.text = "Type Here..."


        toggle_mode_btn = QPushButton()
        toggle_mode_btn.setText("Toggle mode")
        toggle_mode_btn.setFixedWidth(100)
        toggle_mode_btn.clicked.connect(self.toggle_mode)
        nav_bar.addWidget(toggle_mode_btn)

        # Sync Mode: Sync with file and re-render after saved

        sync_tab = QWidget()
        sync_layout = QVBoxLayout(sync_tab)
        sync_preview_label = QLabel()
        sync_preview_label.setText("Preview: ")
        sync_preview_label.setFixedHeight(50)
        sync_preview_label.setStyleSheet("""
            font-size: 32px;
        """)
        self.sync_preview_html = QWebEngineView()
        self.sync_preview_html.setHtml(self.text)
        self.sync_preview_html.setStyleSheet("""
            background-color: #ffffff
        """)
        sync_layout.addWidget(sync_preview_label)
        sync_layout.addWidget(self.sync_preview_html)

        # Manual Mode: Type inside the TextArea on the LHS of the
           ↪  screen

        manual_tab = QWidget()
        manual_layout = QHBoxLayout(manual_tab)
        manual_code_w = QWidget()
        manual_code = QVBoxLayout(manual_code_w)
        manual_code_label = QLabel()
        manual_code_label.setText("Code:")
        manual_code_label.setStyleSheet("""
            font-size: 32px;
        """)
        self.manual_code_editor = QTextEdit()
        self.manual_code_editor.setPlaceholderText(self.text)
        self.manual_code_editor.setUndoRedoEnabled(True)
        self.manual_code_editor.setStyleSheet("""
            background-color: #ffffff
        """)
        manual_code_btn = QPushButton()
        manual_code_btn.clicked.connect(self.convert)
```

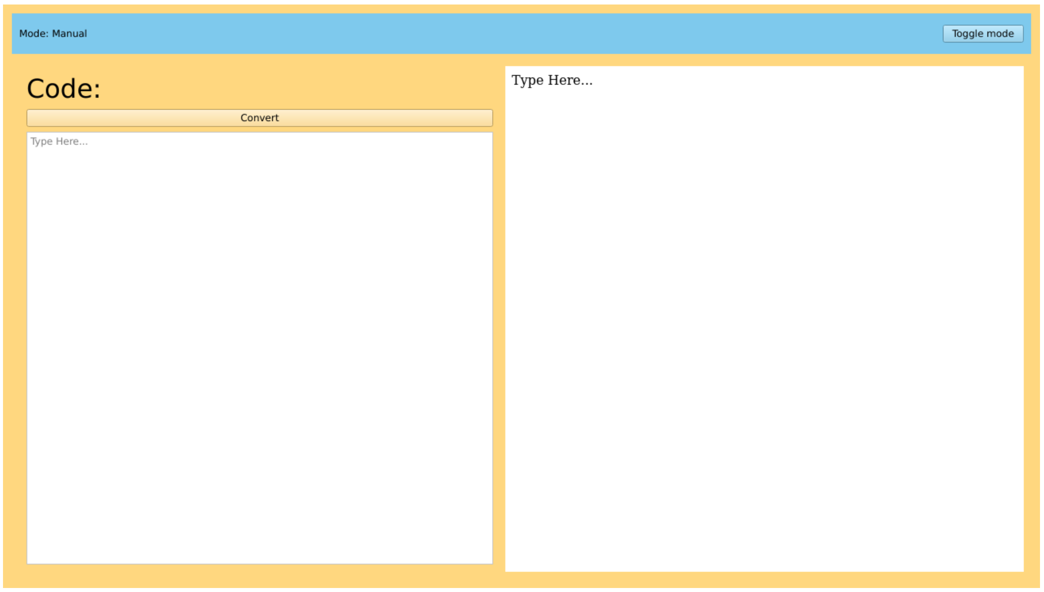
```
        manual_code_btn.setText("Convert")
        manual_code.addWidget(manual_code_label)
        manual_code.addWidget(self.manual_code_editor)
        manual_code.addWidget(manual_code_btn)

        manual_code.addWidget(self.manual_code_editor)
        manual_code_w.setLayout(manual_code)
        self.manual_preview_html = QWebEngineView()
        self.manual_preview_html.setHtml(self.text)
        self.manual_preview_html.setStyleSheet("""
            background-color: #ffffff
        """)
        manual_layout.addWidget(manual_code_w)
        manual_layout.addWidget(self.manual_preview_html)

        self.tab_stack.addWidget(sync_tab)
        self.tab_stack.addWidget(manual_tab)
        self.sync_preview_html.setHtml(self.text)
        self.manual_preview_html.setHtml(self.text)

        v_box.addWidget(nav_w)
        v_box.addWidget(self.tab_stack)
        self.setLayout(v_box)

    def toggle_mode(self) -> None:
        if self.current_mode_text == "Sync":
            self.current_mode_text = "Manual"
            self.tab_stack.setCurrentIndex(1)
            self.cur_mode_label.setText(f"Mode:
              ↪  {self.current_mode_text}")
        else:
            self.current_mode_text = "Sync"
            self.tab_stack.setCurrentIndex(0)
            self.cur_mode_label.setText(f"Mode:
              ↪  {self.current_mode_text}")

    def convert(self) -> None:
        self.text = self.manual_code_editor.toPlainText()
        self.sync_preview_html.setHtml(md.markdown(self.text))
        self.manual_preview_html.setHtml(md.markdown(self.text))
```

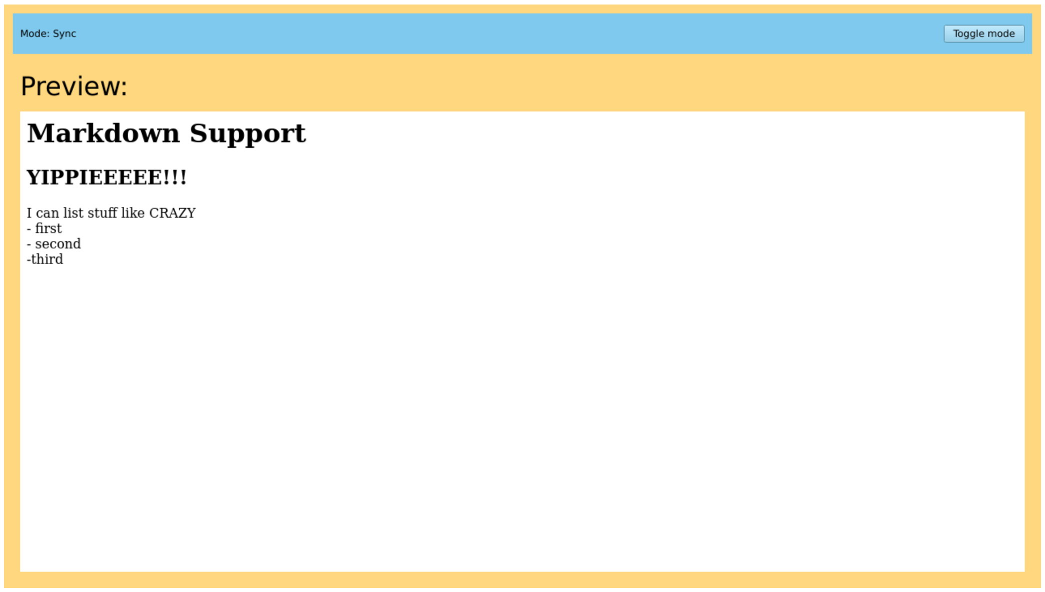
```python
def main() -> None:
    app = QApplication(sys.argv)
    _ = TeskAssignmentPage()
    sys.exit(app.exec())

if __name__ == "__main__":
    main()
```

Output:

Task Assignment Page



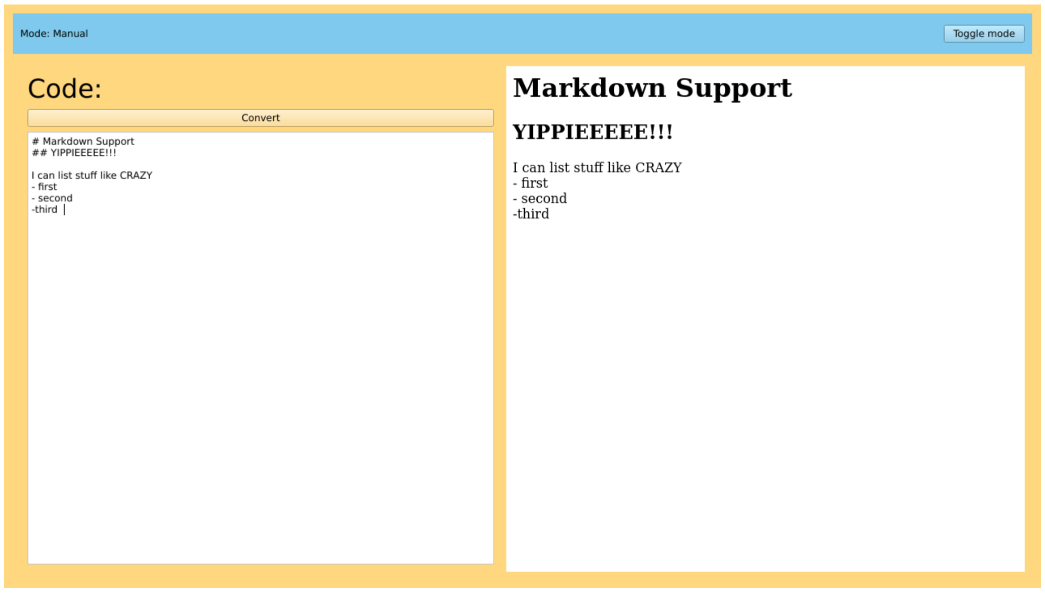




Markdown Support

YIPPIEEEEE!!!

I can list stuff like CRAZY
- first
- second
-third



```
# Markdown Support
## YIPPIEEEEE!!!

I can list stuff like CRAZY
- first
- second
-third
```

Markdown Support

YIPPIEEEEE!!!

I can list stuff like CRAZY
- first
- second
-third

External Plugin

- `pip install markdown`