**PyQt6 Final Project - Submission 1**

# 1. Project Title and Team Members

**Project Title: Unit Converter + Simple Calculator**

**Team Member: Xingzuo Li**

**Student ID: 2295275**

# 2. Application Purpose and Target Users

This desktop application combines two essential tools into one: a unit converter and a basic arithmetic calculator. The purpose is to provide users—particularly students, teachers, and professionals—with a simple, efficient way to perform unit conversions and quick arithmetic calculations without switching between multiple applications.  
  
The application is designed for users who need to perform metric-to-imperial conversions (e.g., meters to feet, Celsius to Fahrenheit), and basic math operations like addition, subtraction, multiplication, and division in a user-friendly graphical interface.

# 3. Planned Features and Functionality

* - Main window to choose between unit converter and calculator.
* - Unit Converter window: supports categories like Length (meters, feet) and Temperature (Celsius, Fahrenheit).
* - Calculator window: supports +, -, \*, / operations with two numeric inputs.
* - Clear and Convert/Calculate buttons.
* - Input validation (no empty or non-numeric input).
* - Display of result in a read-only output field.
* - Menus for switching views and accessing help/about.
* - Keyboard shortcuts (e.g., Ctrl+U for unit converter, Ctrl+C for calculator, Ctrl+Q to quit).
* - Basic file logging of conversion/calculation history (optional feature).
* - Proper error handling and user feedback dialogs.

# 4. Meeting the Technical Requirements

This project will meet all specified technical requirements:  
- Multiple Windows: MainWindow, UnitConverterDialog, CalculatorDialog  
- Widget Variety: QPushButton, QLabel, QLineEdit, QComboBox, QTextEdit, QSlider  
- Layout Management: QVBoxLayout, QFormLayout, QHBoxLayout  
- Signal Handling: Built-in button clicks, textChanged; Custom signal when result is updated  
- Menus & Shortcuts: File, Tools, Help menus with QAction and shortcuts  
- Data Management: All handled with Python lists and strings  
- File Operations: Save result history to text file (optional bonus)  
- Input Validation: Reject empty or non-numeric input with error dialogs  
- Code Organization: Class-based structure with comments and docstrings

# 5. Timeline and Task Distribution

As a solo student, I will follow this proposed timeline:  
  
- June 13- 14: Submit project proposal and UI design  
- June 14–18: Build UI components (MainWindow and dialogs)  
- June 19–23: Implement logic for conversions and calculations  
- June 24–26: Add file operations, menu bar, and keyboard shortcuts  
- June 27–28: Final testing, bug fixing, and video recording  
- June 29: Final submission with code, documentation, and video presentation