Systemic Programming – Task 7 Report

1. Task Description

The goal of this task was to create a 32-bit Windows GUI application in MASM with the following behaviour:
1) The main window contains a text input field.
2) The main window also has a button. When the user clicks the button, the program opens a pop-up window, copies the text entered in the main window to the pop-up, and adds my name to the copied text.
3) When the pop-up window is closed, the text from the pop-up window is copied back to the main window's text field.

2. Tools and Environment

- Operating System: Windows 10 (64-bit)
- IDE: Microsoft Visual Studio 2022
- Assembler: Microsoft Macro Assembler (MASM) integrated into Visual Studio
- Libraries and includes:
  - masm32_lib_inc_for_VS2022 package provided by the teacher
  - Standard Win32 includes: windows.inc, kernel32.inc, user32.inc, gdi32.inc

3. Program Design

3.1 Main Window

- Registers a window class named FirstWindowClass.
- Creates an overlapped window titled "My First MASM32 Window".
- Inside WM_CREATE of WndProc the program creates:
  - An EDIT control for text input (EDIT_MAIN_ID).
  - A BUTTON control labelled "Show" (BTN_SHOW_ID).

3.2 Popup Window

- Registers a separate window class named PopupWindowClass.
- The window procedure for the popup is PopupProc.
- Inside WM_CREATE of PopupProc the program creates an EDIT control (EDIT_POP_ID) where the copied text plus name is shown.

3.3 Data and Buffers

- MainTextBuffer – text buffer (256 bytes) used to transfer text between windows.
- UserName – string constant " - From Al Amin" appended to user input.
- Global handles: hMainEdit, hMainButton, hPopupWnd, hPopupEdit.

4. Implementation Details

In WM_COMMAND of the main window, the program checks for BTN_SHOW_ID. When the button is pressed, GetWindowText reads the text from the main edit control into MainTextBuffer. A manual loop finds the end of the string and copies UserName (" - From Al Amin") to the buffer, effectively appending the name.

After that, CreateWindowEx creates the popup window using PopupWindowClass. ShowWindow and UpdateWindow display it.

In PopupProc, WM_CREATE creates the popup EDIT control and uses SetWindowText to display the content of MainTextBuffer.

When the user closes the popup (WM_CLOSE), GetWindowText reads the current text from the popup edit into MainTextBuffer, and SetWindowText writes it back into the main window edit control. Finally, DestroyWindow closes the popup.

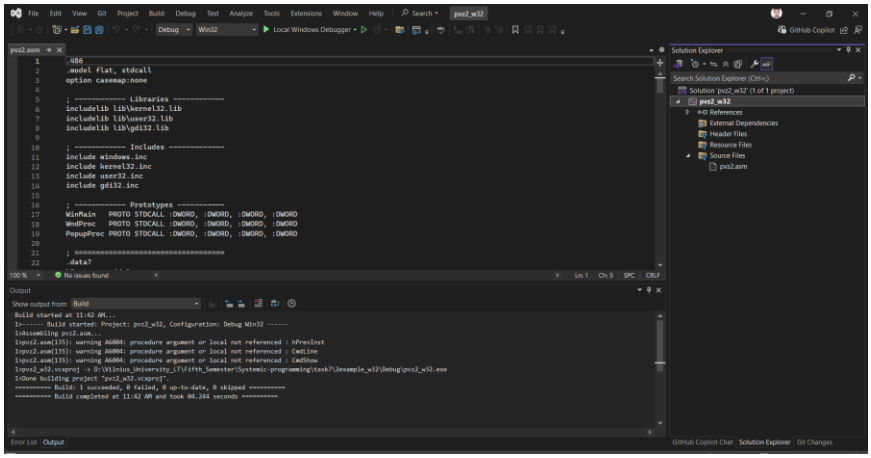
5. Key Code Fragments

The data section defines the handles and buffers, including MainTextBuffer and UserName, as in pvz2.asm. WM_CREATE in WndProc creates the main EDIT and Show button. WM_COMMAND implements the behaviour for BTN_SHOW_ID, and PopupProc handles WM_CREATE and WM_CLOSE for the popup window.

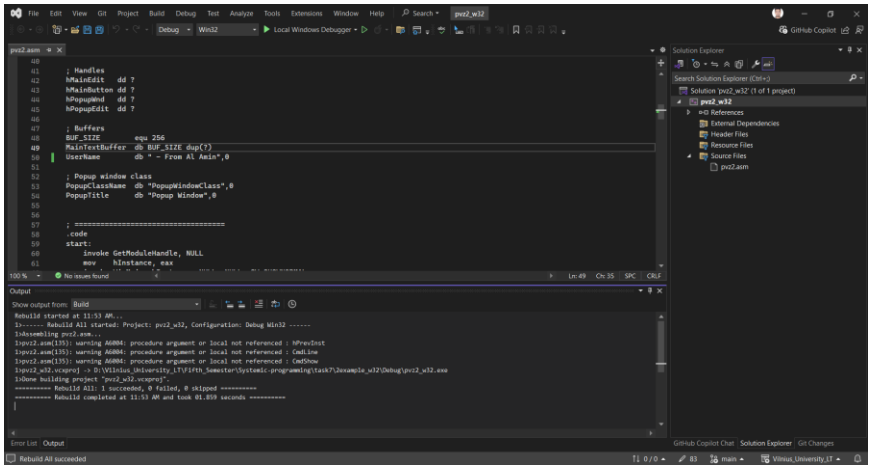
6. Testing and Screenshots

The following screenshots show that the program builds and runs correctly.

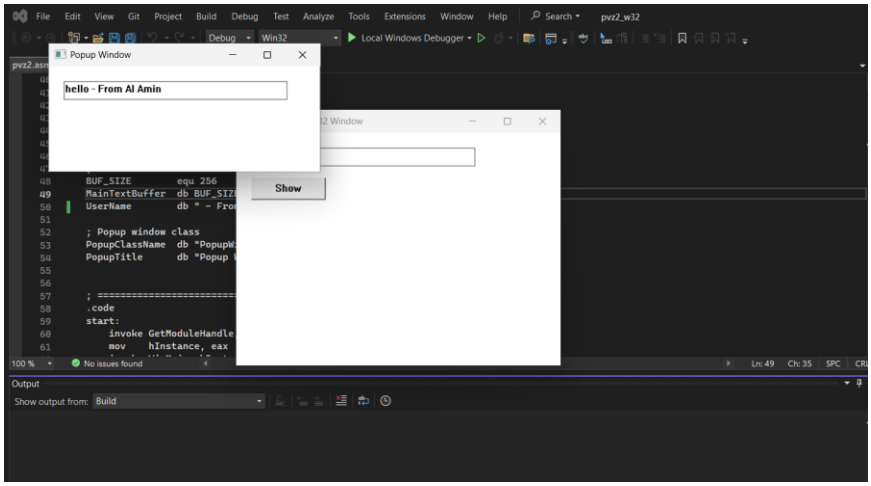
Screenshot 1 – Successful build in Visual Studio 2022.



Screenshot 2 – Main window with text input and Show button.



Screenshot 3 – Popup window showing text with appended name.



7. Conclusions

This task demonstrated how to build a small Win32 GUI program in MASM. I configured MASM with the provided 32-bit libraries, registered two window classes, created controls, and used Win32 API functions such as CreateWindowEx, GetWindowText, SetWindowText, and DestroyWindow. The final program fully satisfies the assignment: text entered in the main window is shown in a popup window with my name appended, and when the popup is closed, the text is copied back to the main window.