ASSIGNMENT 1
"SRS"

AUGUST 20, 2020

AIN SHAMS UNIVERSITY – FACULTY OF ENGINEERING

AIN-SHAMS UNIVERCITY

FACULTY OF ENGINEERING

CSE426: Software Maintenance and Evolution

Assignment 1 "SRS"

Name: **Abdelrahman Amr El-Sayed Mohamed Issawi**

ID: **16P6001**

Email: **aid-issawi@hotmail.com**

Submitted to:

Prof. Dr. Ayman Bahaa.

AUGUST 20, 2020

A report for Software Maintenance and Evolution Course codded CSE426 with the requirements of Ain Shams University.

Table of Contents

1. DESCRIPTION

Develop an IDE which write, edit, compile, and run micro python codes, reformat written text according to python conventions, and allow user to open existed files and save the newly edited files.

2. SRS

2.1 Functional Requirements

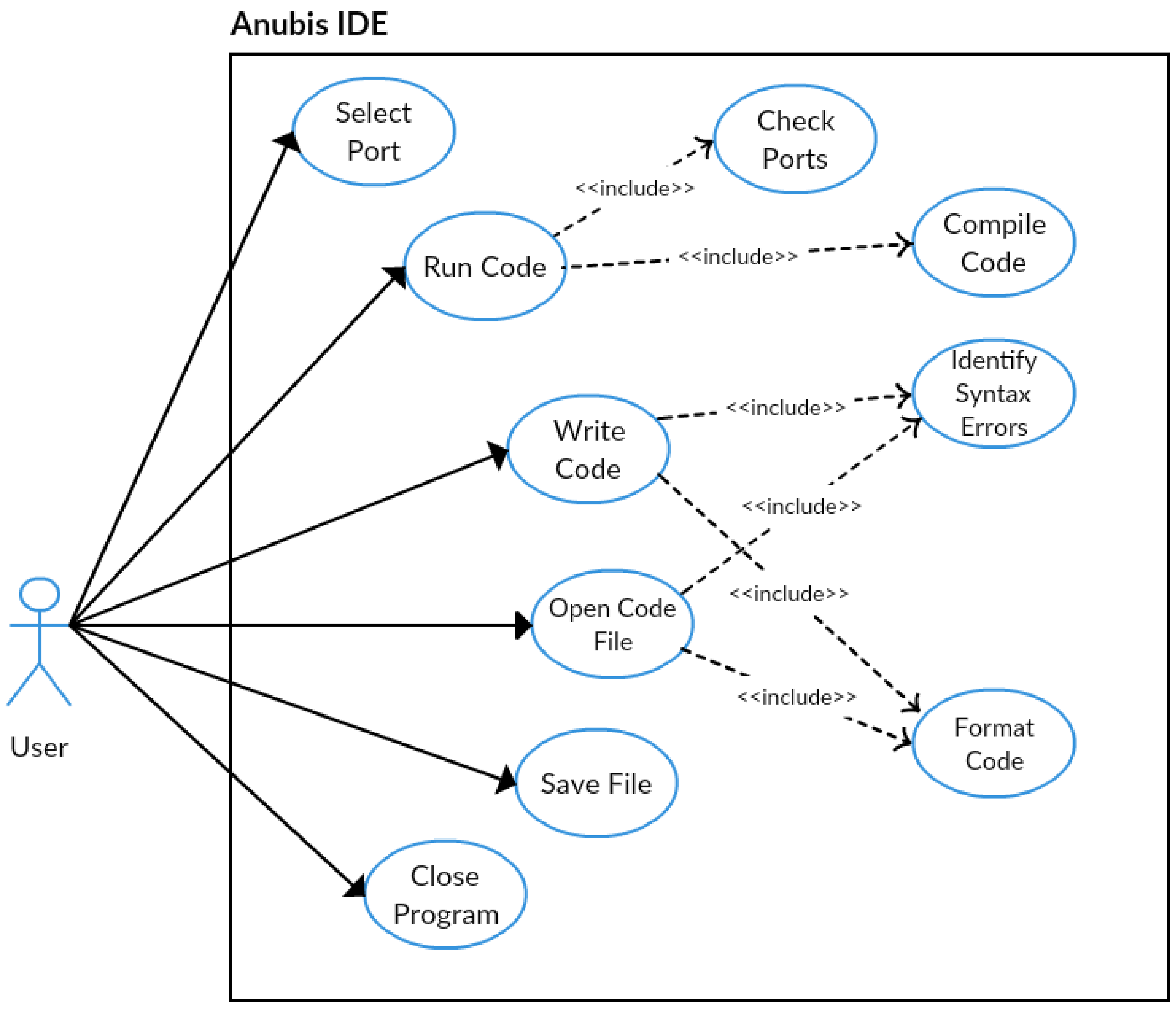
1. Develop an IDE to perform basic functionalities. The IDE should allow to write, edit, compile, and run micro python codes.
2. The IDE must be attached to a python compiler to scan and parse the written text and perform all the compiler tasks.
3. The IDE should allow the user to be able to navigate through the device folders and files to choose files.
4. The IDE should allow the user to be able to save the edited code.
5. The IDE must allow user to select the serial ports.
6. Once a reserved word for python language is detected in the text field for writing the code its formatted referred to the python colors conventions.
7. The IDE must detect the syntax errors and highlight it.

2.2 Non-Functional Requirements

1. The IDE must be written in python programming language.
2. The GUI must be developed using PYQT5 library.
3. The ports are detected automatically using pyserial library.
4. The response time of reformatting the text should not exceed 1 millisecond.
5. Split the coloring class into a separate python file.
6. Provide a detailed User manual and installation instructions.

3. USE CASE

3.1 Use Case Diagram



3.2 Use Case Description

Use Case name:	Run Code.
Goal in context:	Execute the written code and get the output.
Pre-Conditions:	A code is already written to run.
Successful end condition:	An output is returned, or the code function is done.
Failure end condition:	Error occurs, incorrect output.
Primary Actor:	User.
Secondary Actor:	None.
Trigger:	User clicked on Run item in the menu bar.
Input:	Code.
Output:	None.
Include:	Compile Code, Check ports.
Main flow:	1. User Clicks on menu bar item “Run”
Extensions:	1.1 No compiler attached. 1.2 No code written.

Use Case name:	Compile Code.
Goal in context:	Scan and parse the written code.
Pre-Conditions:	The IDE is already opened, execution started, compiler is attached.
Successful end condition:	Code is compiled successfully.
Failure end condition:	Compilation error.
Primary Actor:	None.
Secondary Actor:	None.
Trigger:	Run button is clicked and execution started.
Input:	Code.
Output:	None.
Main flow:	1. User Clicks on menu bar item “Run”. 2. Written code goes through the compiler scanner, parser and all the compilation processes. 3. The output is displayed in the text field.
Extensions:	3.1 Code contains errors. 3.2 No written code. 3.3 Compiler is not attached.

Use Case name:	Check Ports.
Goal in context:	Check that serial ports are selected.
Pre-Conditions:	The IDE is already opened, execution started.
Successful end condition:	Serial ports found.
Failure end condition:	No ports found.
Primary Actor:	None.
Secondary Actor:	None.
Trigger:	Run button is clicked and execution started.
Input:	Code.
Output:	None.
Main flow:	1. User Clicks on menu bar item “Run”. 2. System check for serial ports selected. 3. If ports are found selected the execution starts.
Extensions:	2.1 No ports found.

Use Case name:	Select Port.
Goal in context:	To select port where microcontroller is connected.
Pre-Conditions:	The IDE is already opened, ports are available.
Successful end condition:	Port is successfully selected.
Failure end condition:	No ports selected.
Primary Actor:	User.
Secondary Actor:	None.
Trigger:	User clicked on ports item in the menu bar.
Input:	Port name.
Output:	None.
Main flow:	1. User Clicks on menu bar item “Port” 2. User selects a port from displayed ones.
Extensions:	2.1 User did not find any port available.

Use Case name:	Write Code.
Goal in context:	Enter python codes to the IDE.
Pre-Conditions:	Text filed exists.
Successful end condition:	Code is written successfully.
Failure end condition:	Code is not written.
Primary Actor:	User.
Secondary Actor:	None.
Trigger:	User put cursor on text field to write.
Include:	Identify Syntax Errors, Format Code.
Input:	Text.
Output:	None.
Main flow:	1. User clicks on text field to write code.
Extensions:	1.1 Text field did not accept text.

Use Case name:	Identify Syntax Errors.
Goal in context:	Highlight to the user any found syntax errors.
Pre-Conditions:	Code must exist in the text field.
Successful end condition:	All syntax errors are identified.
Failure end condition:	No or not all syntax errors are identified.
Primary Actor:	Compiler.
Secondary Actor:	None.
Trigger:	Code is written in the text field.
Input:	Code.
Output:	None.
Main flow:	1. User write code in the text field.
Extensions:	1.1 Compiler is not attached.

Use Case name:	Format Code.
Goal in context:	Format written text and color it according to the python color conventions.
Pre-Conditions:	Code must exist in the text field.
Successful end condition:	All code parts are colored correctly according to the coloring conventions.
Failure end condition:	No or not all code parts are colored right.
Primary Actor:	Highlighter.
Secondary Actor:	None.
Trigger:	Code is written in the text field.
Input:	Code.
Output:	None.
Main flow:	1. User write code in the text field.
Extensions:	1.1 Highlighter class is not imported or not working.

Use Case name:	Open Code File.
Goal in context:	Open already existed code files on the device.
Pre-Conditions:	The IDE is already opened.
Successful end condition:	Files are opened successfully.
Failure end condition:	Files are not opened.
Primary Actor:	User.
Secondary Actor:	None.
Trigger:	User clicked on open in the menu bar.
Include:	Identify Syntax Errors, Format Code.
Input:	File path.
Output:	None.
Main flow:	1. User Clicks on menu bar item “File” 2. User selects open option. 3. User Navigates through files. 4. User selects the needed file.
Extensions:	4.1 User did not find any file available. 4.2 User choose unsupported file.

Use Case name:	Save File.
Goal in context:	Save formatted files into specific location.
Pre-Conditions:	The IDE is already opened, Code already exists.
Successful end condition:	Code is saved successfully.
Failure end condition:	Code is not saved.
Primary Actor:	User.
Secondary Actor:	None.
Trigger:	User clicked on save item in menu bar.
Input:	None.
Output:	None.
Main flow:	1. User Clicks on menu bar item “File” 2. User selects Save option. 3. User choose the location.
Extensions:	3.1 User has no memory available.

Use Case name:	Close Program.
Goal in context:	Close the IDE.
Pre-Conditions:	The IDE is already opened.
Successful end condition:	IDE is closed.
Failure end condition:	IDE is not closed and still opened.
Primary Actor:	User.
Secondary Actor:	None.
Trigger:	User clicked on Close item in menu bar.
Input:	None.
Output:	None.
Main flow:	1. User Clicks on menu bar item “File” 2. User selects Close option.
Extensions:	None.