# Raport z wykonania prac badawczych

Poziom gotowości technicznej: 3

Opis prac objętych raportem: Stworzenie urządzenia rejestrującego dane z aparatu liczącego, mierzącego i oceniającego pod kątem jakości kłaczki oraz zbierającego dane z pomp dozujących odczynniki flotacyjne. Na tym etapie prac aparat zostanie uproszczony w ten sposób, że będzie zapewniał komunikację jedynie z wybranym zbiorem pomp dozujących i czujników.

Raport z wykonania:

Stworzenie urządzenia rejestrującego dane z aparatu liczącego, mierzącego i oceniającego pod kątem jakości kłaczki oraz zbierającego dane z pomp dozujących odczynniki flotacyjne

Ocena jakości kłaczków dokonywana jest przez eksperta w oprogramowaniu stworzonym wewnętrznie na potrzeby projektu. Oprogramowanie umożliwia wyświetlanie kolejnych porcji zdjęć i ich natychmiastową ocenę, która zostaje zarejestrowana w repozytorium danych. Oprogramowanie realizuje także dodatkową funkcję pomocniczą, czyli anonimizację zdjęć: ekspert oceniający zdjęcia nie dysponuje w momencie oceny informacjami, które mogłyby zniekształcać jego ocenę takimi jak data, czas i miejsce wykonania zdjęcia. Dane te nie są jednak tracone, ale rejestrowane w sposób niejawny razem z oceną.

Kod narzędzia do oceny jakości:

|  |
| --- |
| #include <ButtonConstants.au3>  #include <DateTimeConstants.au3>  #include <EditConstants.au3>  #include <GUIConstantsEx.au3>  #include <StaticConstants.au3>  #include <WindowsConstants.au3>  #include <Date.au3>  #include <Inet.au3>  #include <JSON.au3>  #include <Array.au3>  #include <File.au3>  #include <String.au3>  #include <Clipboard.au3>  #Region ### START Koda GUI section ### Form=Z:\Dofinansowanie UE\Działanie 1.2\7. Dane\ekspercka\_ocena\_zdjęć.kxf  $Form1 = GUICreate("Ocena ekspercka", 513, 438, 192, 124)  $Label1 = GUICtrlCreateLabel("Folder ze zdjęciami", 16, 48, 93, 17)  $Label2 = GUICtrlCreateLabel("Folder z wynikami", 16, 96, 87, 17)  $Input1 = GUICtrlCreateInput(@ScriptDir, 112, 48, 241, 21)  ;~ $Input1 = GUICtrlCreateInput('R:\2016\P17\_16\_Agrotex\Zdjęcia kłaczki flotacja', 112, 48, 241, 21)  $Input2 = GUICtrlCreateInput(@ScriptDir& "\" & @YEAR & @MON & @MDAY & @HOUR & @MIN, 112, 96, 241, 21)  ;~ $Input2 = GUICtrlCreateInput('R:\2016\P17\_16\_Agrotex\Zdjęcia kłaczki flotacja' & "\" & @YEAR & @MON & @MDAY & @HOUR & @MIN, 112, 96, 241, 21)  $Group1 = GUICtrlCreateGroup("Dane instalacji", 16, 144, 233, 217)  $Label3 = GUICtrlCreateLabel("Nazwa", 24, 176, 37, 17)  $Label4 = GUICtrlCreateLabel("Numer", 24, 208, 35, 17)  $Input3 = GUICtrlCreateInput("", 72, 176, 169, 21)  $Input4 = GUICtrlCreateInput("", 72, 208, 169, 21)  GUICtrlCreateGroup("", -99, -99, 1, 1)  $Group2 = GUICtrlCreateGroup("Data", 264, 144, 217, 217)  $Label5 = GUICtrlCreateLabel("Data wykonania zdjęć", 272, 176, 109, 17)  $tTime = \_Date\_Time\_GetLocalTime()  $dCurrentDate = \_Date\_Time\_SystemTimeToDateTimeStr($tTime, 1)  $Date1 = GUICtrlCreateDate($dCurrentDate, 272, 200, 137, 25, $DTS\_SHORTDATEFORMAT)  GUICtrlCreateGroup("", -99, -99, 1, 1)  $Button1 = GUICtrlCreateButton("Uruchom narzędzie oceny", 264, 384, 201, 33)  $Button2 = GUICtrlCreateButton("Wybierz", 368, 48, 81, 25)  $Button3 = GUICtrlCreateButton("Wybierz", 368, 96, 81, 25)  GUISetState(@SW\_SHOW, $Form1)  #EndRegion ### END Koda GUI section ###  $Form2 = GUICreate("Oceń zdjęcie", 1224, 768, -1, -1, $WS\_POPUP, $WS\_EX\_TOPMOST, $Form1)  $Button11 = GUICtrlCreateButton("5=B.dobry", 1048, 40, 150, 50)  $Button12 = GUICtrlCreateButton("4=Dobry", 1048, 100, 150, 50)  $Button13 = GUICtrlCreateButton("3=Dostateczny", 1048, 160, 150, 50)  $Button14 = GUICtrlCreateButton("2=Mierny", 1048, 220, 150, 50)  $Button15 = GUICtrlCreateButton("1=Niedostateczny", 1048, 280, 150, 50)  $Button16 = GUICtrlCreateButton("Odrzucone", 1048, 340, 150, 50)  $Button17 = GUICtrlCreateButton("Koniec", 1048, 650, 150, 50)  DllCall("user32.dll", "int", "SetParent", "hwnd", WinGetHandle($Form2), "hwnd", WinGetHandle($Form1))  While 1  $nMsg = GUIGetMsg()  Switch $nMsg  Case $GUI\_EVENT\_CLOSE  Exit  Case $Button2  $dInputDir = FileSelectFolder("Wybierz katalog", @ScriptDir)  GUICtrlSetData($Input1, $dInputDir)  $dOutputDir = $dInputDir & "\" & @YEAR & @MON & @MDAY & @HOUR & @MIN  GUICtrlSetData($Input2, $dOutputDir)  Case $Button3  $dOutputDir = FileSelectFolder("Wybierz katalog", @ScriptDir)  GUICtrlSetData($Input2, $dOutputDir)  Case $Button1  $Form2 = GUICreate("Oceń zdjęcie", 1224, 768, -1, -1, $WS\_POPUP, -1, $Form1)  ;~ katalog, w którym operujemy  $sWorkingDir = GUICtrlRead($Input1) & "\"  ;~ katalog output  $sOutputDir = GUICtrlRead($Input2) & "\"  ;~ policzmy zdjęcia  Global $aFiles = \_FileListToArray($sWorkingDir, "\*.jpg", 1, False)  $iFileCount = $aFiles[0]  Global $aOutput[$iFileCount][4]  $Label6 = GUICtrlCreateLabel("1 / " & $iFileCount, 1048, 10, 109, 17)  $sFilePath = GUICtrlRead($Input1) & "\"  Global $iterator = 1  $sFile = $aFiles[$iterator]  ;~ $pic = GUICtrlCreatePic($sWorkingDir & $sFile, 0, 0, 1024, 768)  ;~ $pic = GUICtrlCreatePic($sWorkingDir & $sFile, 0, 0, 224, 224)  $pic = GUICtrlCreatePic($sWorkingDir & $sFile, 0, 0, 820, 616)  $Button11 = GUICtrlCreateButton("5=B.dobry", 1048, 40, 150, 50)  $Button12 = GUICtrlCreateButton("4=Dobry", 1048, 100, 150, 50)  $Button13 = GUICtrlCreateButton("3=Dostateczny", 1048, 160, 150, 50)  $Button14 = GUICtrlCreateButton("2=Mierny", 1048, 220, 150, 50)  $Button15 = GUICtrlCreateButton("1=Niedostateczny", 1048, 280, 150, 50)  $Button16 = GUICtrlCreateButton("Odrzucone", 1048, 340, 150, 50)  $Button17 = GUICtrlCreateButton("Koniec", 1048, 650, 150, 50)  GUISetState(@SW\_SHOW, $Form2)  Case $Button11  $aOutput[$iterator-1][0] = $aFiles[$iterator]  $aOutput[$iterator-1][1] = "5"  $aOutput[$iterator-1][2] = FileGetSize($aFiles[$iterator])  $aOutput[$iterator-1][3] = \_formatDate(GUICtrlRead($Date1))  If $iterator < UBound($aFiles)-1 Then  $iterator = $iterator + 1  GUICtrlSetImage($pic, $sWorkingDir & $aFiles[$iterator])  GUICtrlSetData($Label6, $iterator & " / " & $iFileCount)  GUISetState(@SW\_SHOW, $Form2)  EndIf  ;~ \_ArrayDisplay($aOutput)  Case $Button12  $aOutput[$iterator-1][0] = $aFiles[$iterator]  $aOutput[$iterator-1][1] = "4"  $aOutput[$iterator-1][2] = FileGetSize($aFiles[$iterator])  $aOutput[$iterator-1][3] = \_formatDate(GUICtrlRead($Date1))  If $iterator < UBound($aFiles)-1 Then  $iterator = $iterator + 1  GUICtrlSetImage($pic, $sWorkingDir & $aFiles[$iterator])  GUICtrlSetData($Label6, $iterator & " / " & $iFileCount)  GUISetState(@SW\_SHOW, $Form2)  EndIf  ;~ \_ArrayDisplay($aOutput)  Case $Button13  $aOutput[$iterator-1][0] = $aFiles[$iterator]  $aOutput[$iterator-1][1] = "3"  $aOutput[$iterator-1][2] = FileGetSize($aFiles[$iterator])  $aOutput[$iterator-1][3] = \_formatDate(GUICtrlRead($Date1))  If $iterator < UBound($aFiles)-1 Then  $iterator = $iterator + 1  GUICtrlSetImage($pic, $sWorkingDir & $aFiles[$iterator])  GUICtrlSetData($Label6, $iterator & " / " & $iFileCount)  GUISetState(@SW\_SHOW, $Form2)  EndIf  ;~ \_ArrayDisplay($aOutput)  Case $Button14  $aOutput[$iterator-1][0] = $aFiles[$iterator]  $aOutput[$iterator-1][1] = "2"  $aOutput[$iterator-1][2] = FileGetSize($aFiles[$iterator])  $aOutput[$iterator-1][3] = \_formatDate(GUICtrlRead($Date1))  If $iterator < UBound($aFiles)-1 Then  $iterator = $iterator + 1  GUICtrlSetImage($pic, $sWorkingDir & $aFiles[$iterator])  GUICtrlSetData($Label6, $iterator & " / " & $iFileCount)  GUISetState(@SW\_SHOW, $Form2)  EndIf  ;~ \_ArrayDisplay($aOutput)  Case $Button15  $aOutput[$iterator-1][0] = $aFiles[$iterator]  $aOutput[$iterator-1][1] = "1"  $aOutput[$iterator-1][2] = FileGetSize($aFiles[$iterator])  $aOutput[$iterator-1][3] = \_formatDate(GUICtrlRead($Date1))  If $iterator < UBound($aFiles)-1 Then  $iterator = $iterator + 1  GUICtrlSetImage($pic, $sWorkingDir & $aFiles[$iterator])  GUICtrlSetData($Label6, $iterator & " / " & $iFileCount)  GUISetState(@SW\_SHOW, $Form2)  EndIf  ;~ \_ArrayDisplay($aOutput)  Case $Button16  $aOutput[$iterator-1][0] = $aFiles[$iterator]  $aOutput[$iterator-1][1] = "0"  $aOutput[$iterator-1][2] = FileGetSize($aFiles[$iterator])  $aOutput[$iterator-1][3] = \_formatDate(GUICtrlRead($Date1))  If $iterator < UBound($aFiles)-1 Then  $iterator = $iterator + 1  GUICtrlSetImage($pic, $sWorkingDir & $aFiles[$iterator])  GUICtrlSetData($Label6, $iterator & " / " & $iFileCount)  GUISetState(@SW\_SHOW, $Form2)  EndIf  ;~ \_ArrayDisplay($aOutput)  Case $Button17  ;~ \_ArrayDisplay($aOutput)  ;~ zapisujemy wyniki  DirCreate($sOutputDir)  ;~ ;pliczek z zawartością $aoutput  \_FileWriteFromArray($sOutputDir & "OcenaEksperta.csv", $aOutput)  For $i = 0 To UBound($aOutput) - 1  If ($aOutput[$i][0]) <> "" Then  $sFileName = $aOutput[$i][0]  $sfileName = \_StringInsert($sFileName, "\_"&$aOutput[$i][1], StringLen($sfileName)-4)  ;~ FileCopy($sWorkingDir & $aOutput[$i][0], $sOutputDir & $sfileName, $FC\_OVERWRITE + $FC\_CREATEPATH)  EndIf  Next  ;~ Local $command = "'C:\Program Files\7-Zip\7z.exe' a -r archive.zip '" & $sOutputDir & "'"  ;~ Run(@ComSpec & " /c " & $command, $sOutputDir)  ;~ ConsoleWrite(@ComSpec & " /c " & $command)  ;~ \_ArrayDisplay($aOutput)  Run("C:\WINDOWS\EXPLORER.EXE /n,/e," & $sOutputDir)  Exit  EndSwitch  WEnd  Func \_FindLatestLog($dir, $mask)  Local $latestTime = 0, $currentTime = 0  Local $files = \_FileListToArray($dir, $mask, 1)  If @error = 1 Then  ConsoleWrite('Func: FindLog, Error: Path was invalid.' & @CRLF)  Return False  EndIf  If @error = 4 Then  ConsoleWrite('Func: FindLog, Error: Log file was not found.' & @CRLF)  Return False  EndIf  For $i = 1 To UBound($files) - 1  $currentFile = $dir & "\" & $files[$i]  $currentTime = FileGetTime($currentFile, $FT\_MODIFIED, 1)  If $currentTime > $latestTime Or $i = 1 Then  $latestTime = $currentTime  $latestFile = $currentFile  EndIf  Next  ConsoleWrite("Latest log file: " & $latestFile & ", time modified: " & $latestTime & @CRLF)  Return $latestFile  EndFunc ;==>\_FindLatestLog  Func \_formatDate($guiDate)  $sDate = StringRegExpReplace($guiDate, '\D', '-')  $aDateParts = \_StringExplode($sDate, "-")  ;~ \_ArrayDisplay($aDateParts, "StringExplode 0")  Switch StringLen($aDateParts[0])  Case 2  $sDate = $aDateParts[2] & "-" & $aDateParts[1] & "-" & $aDateParts[0]  Case 4  $sDate = $aDateParts[0] & "-" & $aDateParts[1] & "-" & $aDateParts[2]  Case Else  MsgBox(0, "Błąd", "Podana data jest błędna")  Exit  EndSwitch  Return $sDate  EndFunc ;==>\_formatDate |

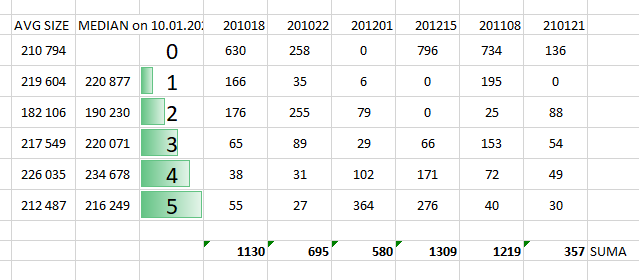
Kod anonimizatora zdjęć:

|  |
| --- |
| #include <Array.au3>  #include <String.au3>  $sDir = "C:\Users\MARCIN\OneDrive - EMI Sp. z o.o\AI\B. Aparat zbierający zdjęcia\Zdjęcia\2020-04-24 - Żory\"  $a\_files = FileReadToArray($sDir&'Book1.csv')  dim $a\_renamelist[UBound($a\_files)][2]  \_ArrayDisplay($a\_files)  for $i=0 to UBound($a\_files)-1  ;~ 11\TYP70-200407-112508-006.jpg 88DQCPXY.jpg  $a1 = \_StringExplode($a\_files[$i],';')  FileCopy($sDir&$a1[0], $sDir&"do\_oceny\"&$a1[1])  ;~ $a\_renamelist[$i][0] = $a1[0]  ;~ $a\_renamelist[$i][1] = $a1[1]  Next  \_ArrayDisplay($a\_renamelist) |

Struktura plików xls z zestawieniami:



Podsumowanie ocen eksperta:



Zbieranie danych z pomp dozujących jest na obecnym etapie realizowane wyłącznie ręcznie, prace mające na celu automatyzację procesu są opóźnione. Na bieżącym etapie rozwiązanie zastępcze w postaci ręcznej rejestracji danych jest wystarczające.

# Ryzyka dla kolejnego poziomu gotowości

W toku prac badawczych zidentyfikowano następujące ryzyka na drodze do podniesienia gotowości do poziomu IV:

Ryzyko niedokładnego/mylnego przetwarzania danych