Міністерство освіти і науки

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Кафедра автоматизації проектування енергетичних процесів і систем

ЗВІТ   
про виконання практичної роботи №2  
з дисципліни «Безпека програмного забезпечення»

Тема: «Захист веб-ресурсів від ботів та спаму за допомогою механізму CAPTCHA»

Варіант 15

Виконав студент 3-го курсу

групи ТІ-92 Черноусов Денис

Перевірила Смаковська Г. М.

КИЇВ 2022

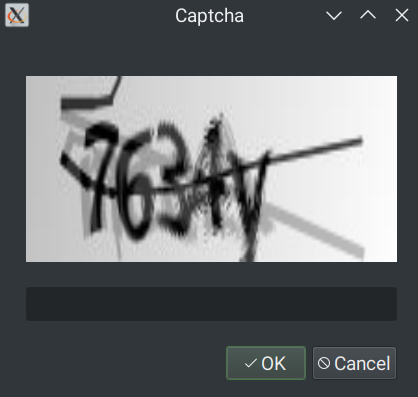
**1. Мета роботи**: отримати практичні навики реалізації механізму CAPTCHA.

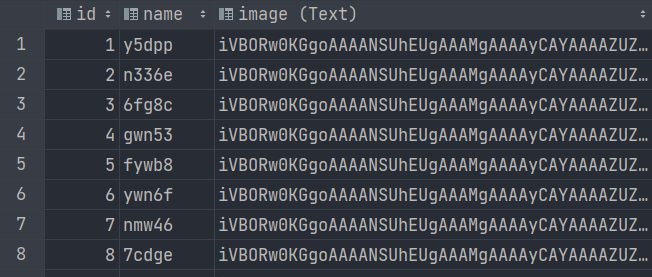
**2. Завдання на виконання**:

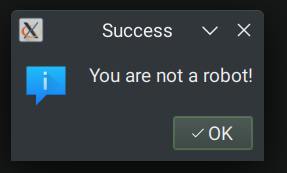
Удосконалити ПЗ розроблене в попередній практичній роботі в частині захисту однієї із розроблених форм за допомогою механізму CAPTCHA відповідно до індивідуального завдання. Для оцінки «відмінно» запрограмувати механізм CAPTCHA у вигляді запиту відповіді на зображення із накладенням фону та шумів, яке в довільному порядку обирається з бази даних ПЗ. Розмістити на окремій формі reCaptcha Google.

**4. Тестовий приклад**

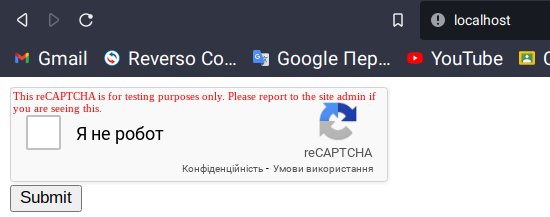
Каптча в програмі з шумами та фоном

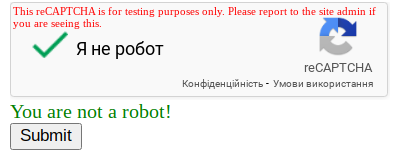
Капча та фон випадковим чином вибираються з бази даних

Фото записане у вигляді закодованих рядків з байтів

****

Повідомлення на коректно введену капчу

Recaptcha на окремій формі (її тестова версія)

****

**5. Програмний код що реалізує індивідуальне завдання:**

UiCaptcha.py

*import* base64  
  
*from* PyQt5 *import* QtWidgets, QtCore  
*from* PyQt5.QtGui *import* QImage, QPixmap, QPainter  
*from* PyQt5.QtWidgets *import* QDialog  
  
  
*class* UiCaptcha:  
 \_\_dialog: QDialog  
  
 *def* \_\_init\_\_(*self*, image\_coded: bytes, background\_coded: bytes):  
 *self*.\_\_dialog = QDialog()  
 *self*.\_\_dialog.setObjectName("dialog")  
 *self*.\_\_dialog.resize(425, 366)  
 *self*.\_\_dialog.setWindowTitle('Captcha')  
 *self*.gridLayoutWidget = QtWidgets.QWidget(*self*.\_\_dialog)  
 *self*.gridLayoutWidget.setGeometry(QtCore.QRect(29, 40, 371, 304))  
 *self*.gridLayoutWidget.setObjectName("gridLayoutWidget")  
 *self*.gridLayout = QtWidgets.QGridLayout(*self*.gridLayoutWidget)  
 *self*.gridLayout.setContentsMargins(0, 0, 0, 0)  
 *self*.gridLayout.setHorizontalSpacing(10)  
 *self*.gridLayout.setVerticalSpacing(25)  
 *self*.gridLayout.setObjectName("gridLayout")  
 *self*.label = QtWidgets.QLabel(*self*.gridLayoutWidget)  
 *self*.label.setObjectName("label")  
 *self*.label.setPixmap(*self*.\_\_draw\_captcha(base64.b64decode(image\_coded), base64.b64decode(background\_coded)))  
 *self*.label.setScaledContents(*True*)  
 *self*.gridLayout.addWidget(*self*.label, 0, 0, 1, 1)  
 *self*.buttonBox = QtWidgets.QDialogButtonBox(*self*.gridLayoutWidget)  
 *self*.buttonBox.setOrientation(QtCore.Qt.Horizontal)  
 *self*.buttonBox.setStandardButtons(QtWidgets.QDialogButtonBox.Cancel | QtWidgets.QDialogButtonBox.Ok)  
 *self*.buttonBox.setObjectName("buttonBox")  
 *self*.gridLayout.addWidget(*self*.buttonBox, 3, 0, 1, 1)  
 *self*.lineEdit = QtWidgets.QLineEdit(*self*.gridLayoutWidget)  
 *self*.lineEdit.setEnabled(*True*)  
 *self*.lineEdit.setObjectName("lineEdit")  
 *self*.gridLayout.addWidget(*self*.lineEdit, 1, 0, 1, 1)  
 *self*.buttonBox.accepted.connect(*self*.\_\_dialog.accept)  
 *self*.buttonBox.rejected.connect(*self*.\_\_dialog.reject)  
 QtCore.QMetaObject.connectSlotsByName(*self*.\_\_dialog)  
  
 *def* \_\_draw\_captcha(*self*, image\_bytes: bytes, background\_bytes: bytes) -> QPixmap:  
 image = QPixmap().fromImage(QImage().fromData(image\_bytes))  
 painter = QPainter()  
 painter.begin(image)  
 painter.setOpacity(0.25)  
 painter.drawPixmap(0, 0, QPixmap().fromImage(QImage().fromData(background\_bytes)))  
 painter.end()  
 *return* image  
  
 *def* exec(*self*) -> str:  
 *if self*.\_\_dialog.exec():  
 *return self*.lineEdit.text()

MainWindow.py

*def* \_\_test\_captcha(*self*) -> int:  
 samples = get\_random\_rows\_from\_table(CAPTCHA, 2)  
 answer: str = samples.iloc[0, 1]  
 image: bytes = samples.iloc[0, 2]  
 background: bytes = samples.iloc[1, 2]  
 response: str = UiCaptcha(image, background).exec()  
 *if None is* response:  
 *return* QMessageBox.Cancel  
 *if* response == answer:  
 *return self*.\_\_messanger.show\_message('You are not a robot!', SUCCESS, QMessageBox.Information)  
 *return self*.\_\_test\_captcha()

index.html

<html>  
<head>  
 <title>reCAPTCHA demo: Simple page</title>  
 <script *src*="https://www.google.com/recaptcha/api.js" *async defer*></script>  
  
 <script>  
 *var* recaptcha\_response = '';  
  
 *function* submitUserForm() {  
 *if* (recaptcha\_response.length === 0) {  
 document.getElementById('g-recaptcha-error').innerHTML = '<span style="color:red;">This field is required.</span>';  
 *return false*;  
 }  
 *return true*;  
 }  
  
 *function* verify(token) {  
 recaptcha\_response = token;  
 document.getElementById('g-recaptcha-error').innerHTML = '<span style="color: green">You are not a robot!</span>';  
 }  
 </script>  
*<!--6LeIxAcTAAAAAJcZVRqyHh71UMIEGNQ\_MXjiZKhI-->*  
</head>  
<body>  
<form *method*="GET" *action*="" *onsubmit*="*return* submitUserForm();">  
 <div *class*="g-recaptcha" *data-sitekey*="6LeIxAcTAAAAAJcZVRqyHh71UMIEGNQ\_MXjiZKhI" *data-callback*="verify"></div>  
 <div *id*="g-recaptcha-error"></div>  
 <input *type*="submit" *name*="submit" *value*="Submit" />  
</form>  
</body>  
  
</html>

**6. Висновок**

LibreOffice Writer