Marek Malý C4C

18.4.2023

Spše Ječná , školní projekt

Mail : maly@spsejecna.cz

Secret Data

Požadavky na práci s aplikací:

stažený python stažený PyCharm nebo jiný program, který umožňuje práci s pythonem

Představení projektu:

Secret data je oknový Python projekt, který umožňuje uživatelům šifrovat svá data pomocí metod XOR a Fernet. Součástí aplikace je i jednoduchý slovník, který si uživatel může naplnit vlastními slovy a který samozřejmě umí i dešifrovat. Data jsou ukládána v šifrované podobě v databázi MySQL.

Slovíčka v slovníku jsou uložena také v databázi MySQL a uživatel je může snadno přidávat, upravovat nebo odstraňovat. Slovník umožňuje uživatelům vyhledávat slova v angličtině a zobrazit jejich český překlad.

Secret data je navržena tak, aby byla snadno použitelná pro každého uživatele, ať už jde o běžného uživatele počítače, studenta nebo profesionála.

Architektura aplikace : architektura Three tier

Architektura Three tier odděluje aplikaci do tří vrstev: prezentační, logickou a datovou vrstvu. Prezentační vrstva zahrnuje uživatelské rozhraní aplikace, logická vrstva řídí a vykonává různé operace na základě uživatelských požadavků a datová vrstva uchovává data potřebná pro aplikaci.

Knihovny které program používá: tkinter filedialog , messagebox , logging , re , Toolbox , itertools

,mailbox , PIL , unnitest , json , mysql.connector

Ukázky kódu:

```python
def Encrypt(text, key):
    # Převod řetězců text a key na bajty
    text_bytes = text.encode()
    key_bytes = key.encode()

    # Kontrola typu textu a klíče
    if not isinstance(_text_, str_) or not isinstance(_key_, str_):
        messagebox.showerror(_"Error"_, "imput text musí být string"_)
        return
    if len(_key.strip(_)_) == 0_:
        messagebox.showerror(_"Error"_, "klíč je prázdný"_)
        return

    if len(_key_bytes_) < 4_:
        messagebox.showerror(_"Error"_, "klíč je krátký"_)
        return

    if len(_text.strip(_)_) == 0_:
        messagebox.showerror(_"Error"_, "text je prázdný"_)
        return

    # Pokud je text delší než klíč, klíč se zopakuje pomocí cyklu a bude stejně dlouhý jako text

    if len(text_bytes) > len(key_bytes):
        repeated_key = bytes(itertools.islice(itertools.cycle(key_bytes), len(text_bytes)))
    else:
        repeated_key = key_bytes[:len(text_bytes)]

    # Pro každý bajt textu se provede operace XOR s odpovídajícím bajtem klíče
    # Výsledkem této operace je nový bajt, který je uložen do výsledného bajtového řetězce
    result = bytearray(x ^ y for x, y in zip(text_bytes, repeated_key))
    return result.hex()
```

```python
def Encrypt(text):
    if len(_text.strip(_)_) == 0_:
        messagebox.showerror(_"Error"_, "text je prázdný"_)
        return
    try:
        key = Fernet.generate_key()  # Vygenerování náhodného klíče pro šifrování
        f = Fernet(key)  # Inicializace instance třídy Fernet s vygenerovaným klíčem
        encrypted_message = f.encrypt(text.encode())  # Zašifrování zadaného textu pomocí třídy Fernet
        encrypted_message_str = encrypted_message.decode()  # Převedení zašifrovaného textu na řetězec
        root = tk.Tk()  # Vytvoření instance hlavního okna pro dialogové okno souboru
        root.withdraw()  # Skrytí hlavního okna aplikace
        file_path = filedialog.asksaveasfilename(defaultextension='.key', filetypes=[('Key files', '*.key')])
        if file_path:
            with open(file_path, 'wb') as file:  # Otevření souboru s klíčem pro zápis bytů
                file.write(key)  # Zápis klíče do souboru
        return encrypted_message_str  # Vrácení zašifrovaného textu v řetězcové podobě
    except Exception as e:  # Výjimka pro případ chyby
        log.log_Error_('Enc'_, 'chyba šifrování '_)
        return None
```

Xor šifrovaní

Funkce přijímá dva argumenty: **`text`** a **`key`**.

Nejprve se provádí převod řetězců **`text`** a **`key`** na bajty. Poté se provádí kontrola typu argumentů, aby byly oba řetězce typu **`str`**. Dále se kontroluje, zda je klíč neprázdný a dostatečně dlouhý. Pokud jsou některé z kontrolních podmínek porušeny, funkce vrací chybové hlášky.

Pokud je text delší než klíč, klíč se zopakuje pomocí cyklu tak, aby byl stejně dlouhý jako text. Poté se pro každý bajt textu provádí operace XOR s odpovídajícím bajtem klíče. Výsledkem této operace je nový bajt, který je uložen do výsledného bajtového řetězce.

Funkce vrací hexadecimální reprezentaci výsledného bajtového řetězce.

Fernet šifrovaní

Tato funkce "Encrypt" slouží k zašifrování zadaného textu pomocí třídy Fernet, která využívá symetrické šifrování s použitím klíče. Pokud je text prázdný, funkce vrátí chybovou zprávu pomocí dialogového okna. Pokud dojde k úspěšnému zašifrování, klíč bude uložen do souboru vybraného uživatelem prostřednictvím dialogového okna pro výběr umístění souboru s klíčem. Funkce vrací zašifrovaný text v řetězcové podobě nebo None, pokud dojde k chybě.

Chybové stavy:

pokud uživatel nezadá text k zašifrovaní aplikace ho upozorní

pokud uživatel nezadá key k zašifrovaní aplikace ho upozorní

Pokud je klíč moc krátký program uživatele upozorní

Diagram db secret_data

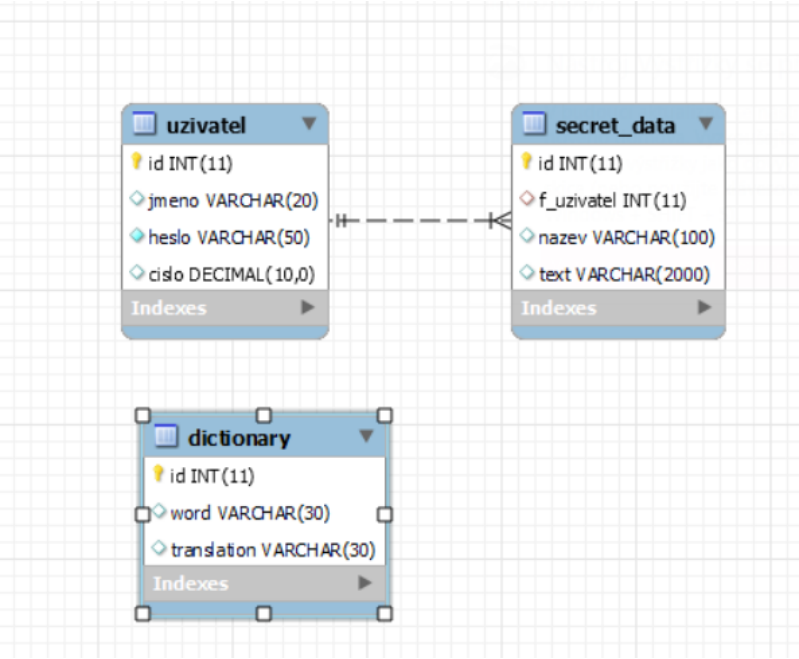

Table uživatel : pro uživatelská data

Table secret_data : pro ukládaní názvů zašifrovaných textů a zašifrovaných textů

Table dictonary : pro ukládaní slovíček a překladů

Závěrečné resumé:

Secret data je jednoduchá aplikace napsaná v Pythonu, která umožňuje uživatelům šifrovat svá data pomocí metod XOR a Fernet a ukládat je v šifrované podobě v databázi MySQL. Součástí aplikace je i slovník, který si uživatelé mohou snadno přidávat, upravovat nebo odstraňovat. Slovník umožňuje vyhledávat slova v angličtině a zobrazovat jejich český překlad. Aplikace je navržena tak, aby byla snadno použitelná pro každého uživatele a nabízí jednoduché uživatelské rozhraní. Plánujeme přidat další šifrovací metody a celkově vylepšit aplikaci.