

SAPIENTIA ERDÉLYI MAGYAR TUDOMÁNYEGYETEM
MAROSVÁSÁRHELYI KAR,
INFORMATIKA SZAK



SAPIENTIA
ERDÉLYI MAGYAR
TUDOMÁNYEGYETEM

Cryptorithm

DIPLOMADOLGOZAT

Témavezető:
Dr. Márton Gyöngyvér,
Egyetemi adjunktus

Végzős hallgató:
Füzi Zsolt

2023

UNIVERSITATEA SAPIENTIA DIN CLUJ-NAPOCA
FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE,
SPECIALIZAREA INFORMATICĂ



UNIVERSITATEA
SAPIENTIA

Cryptorithm

LUCRARE DE DIPLOMĂ

Coordonator științific:
Dr. Márton Gyöngyvér,
Lector universitar

Absolvent:
Füzi Zalan

2023

SAPIENTIA HUNGARIAN UNIVERSITY OF
TRANSYLVANIA
FACULTY OF TECHNICAL AND HUMAN SCIENCES
COMPUTER SCIENCE SPECIALIZATION



SAPIENTIA
HUNGARIAN UNIVERSITY
OF TRANSYLVANIA

Cryptorithm

BACHELOR THESIS

Scientific advisor:
Dr. Márton Gyöngyvér,
Lecturer

Student:
Füzi Zalán

2023

LUCRARE DE DIPLOMĂ

Coordonator științific:
Conf. dr. Márton Gyöngyvér

Candidat: **Füzi Zalán**
Anul absolvirii: **2023**

a) Tema lucrării de licență:

Proiectarea și dezvoltarea unei aplicații web multilingve, cu numele Cryptorithm, pentru învățarea despre criptosistemele populare.

b) Problemele principale tratate:

- Studiu bibliografic privind criptosistemelor populare și aplicațiilor web similare.
- Studiu al pachetului Cryptography în Python
- Proiectarea și dezvoltarea unei aplicații web pentru utilizarea și învățarea despre diferite sisteme criptografice
- Documentarea adecvată a criptosistemelor utilizate

c) Desene obligatorii:

- Schema bloc a paginii web
- Diagrame UML ale software-ului.

d) Softuri obligatorii:

- Proiectarea și implementarea logicii în limbajul de programare Python
- Proiectarea și implementarea front-end-ului
- Integrarea logicii dezvoltate în aplicația web

e) Bibliografia recomandată:

- [1] Introduction to Cryptography: Principles and Applications / Hans Delfs and Helmut Knebl (Springer, 2nd edition, 2007)
[2] Kriptográfiai alapismeretek / Márton Gyöngyvér 2008

f) Termene obligatorii de consultații: săptămânal

g) Locul și durata practicii: Universitatea „Sapientia” din Cluj-Napoca,
Facultatea de Științe Tehnice și Umaniste din Târgu Mureș

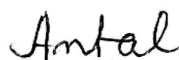
Primit tema la data de: 20.11.2022

Termen de predare: 02.07.2023

Semnătura Director Departament



**Semnătura responsabilului
programului de studiu**



Semnătura coordonatorului



Semnătura candidatului



Declarație

Subsemnatul/a FÜZI ZALÁN, absolvent(ă) al/a specializării
INFORMATICA, promoția 2023, cunoscând
prevederile Legii Educației Naționale 1/2011 și a Codului de etică și deontologie profesională a
Universității Sapientia cu privire la furt intelectual declar pe propria răspundere că prezenta
lucrare de licență/proiect de diplomă/disertație se bazează pe activitatea personală,
cercetarea/proiectarea este efectuată de mine, informațiile și datele preluate din literatura de
specialitate sunt citate în mod corespunzător.

Localitatea, TÂRGU MUREȘ
Data: 2023-06-08.

Absolvent

Semnătura [Signature]

Kivonat

A Cryptorithm egy online platform, ahol a felhasználók kriptográfiai rendszereket próbálhatnak ki, és egy kiterjedt tudásbázist kínál, hogy szabadon tanulhassanak a rendszerek eredetéről, működéséről és felhasználásáról.

A szoftver választékot kínál a népszerű és ismert módszerekből az adatok titkosításához, visszafejtéséhez és transzformációjához, lehetővé téve a felhasználóknak, hogy szabadon válogathassanak és kipróbáljanak különböző lehetőségeket. Emellett rendelkezik egy tanulói felülettel, amely hozzájárul a felhasználók számára a kriptográfiai rendszerek mélyebb megértésében. Itt információkat olvashatnak, és gyakorlati példákon keresztül sajátíthatják el ezeket a rendszereket.

Kiemelt figyelmet fordítottam a fejlesztés során a felhasználói élmény biztosítására, így a szoftver könnyedén használható felhasználói felülettel rendelkezik. Emellett fontos szempont volt az, hogy az oldal támogassa a többnyelvűséget is, így az oldal elérhetővé válik magyar és angol nyelvű felhasználók számára is.

A diplomadolgozatom további részében részletesen bemutatom a szoftver architektúráját, a megvalósított funkciókat, valamint a hozzá tartozó tudnivalókat. Emellett a felhasználói dokumentáció is részét képezi a dolgozatnak.

Rezumat

Cryptorithm este o platformă online unde utilizatorii pot încerca sisteme criptografice și oferă o bază de cunoștințe extinsă pentru a învăța în mod liber despre originea, funcționarea și utilizarea acestor sisteme.

Software-ul oferă o selecție de metode populare și bine cunoscute de criptare, decriptare și transformare a datelor, permițând utilizatorilor să aleagă și să încerce în mod liber diferite opțiuni. De asemenea, dispune de o interfață de învățare pentru a ajuta utilizatorii să dobândească o înțelegere mai profundă a sistemelor criptografice. Aici aceștia pot citi informații și pot învăța despre aceste sisteme prin exemple practice.

Am acordat o atenție deosebită asigurării unei experiențe de utilizare în timpul dezvoltării, astfel încât software-ul are o interfață ușor de utilizat. În plus, a fost important ca site-ul să fie compatibil cu multilingvismul, astfel încât să fie accesibil atât pentru utilizatorii englezi, cât și pentru cei maghiari.

În restul tezei mele voi descrie în detaliu arhitectura software-ului, caracteristicile implementate și cunoștințele aferente. În plus, documentația pentru utilizatori este, de asemenea, parte a tezei.

Abstract

Cryptorithm is an online platform where users can try out cryptographic systems and offers an extensive knowledge base to freely learn about the origin, operation and use of these systems.

The software offers a selection of popular and well-known methods for encrypting, decrypting and transforming data, allowing users to freely choose and try out different options. It also has a learning interface to help users gain a deeper understanding of cryptographic systems. Here they can read information and learn about these systems through practical examples.

I have paid particular attention to ensuring a user experience during development, so the software has an easy-to-use interface. In addition, it was important that the site supports multilingualism so that it is accessible to both English and Hungarian users.

In the rest of my thesis I will describe in detail the architecture of the software, the implemented features and the related knowledge. In addition, the user documentation is also part of the thesis.

Tartalomjegyzék

1. Bevezető	10
1.1. Motiváció és Célkitűzések	11
2. Programok, technológiák bemutatása	12
2.1. Az alkalmazás felépítése	12
2.1.1. Frontend	12
2.1.2. Backend	13
2.2. Hasonló platformok, szoftverek összehasonlítása	14
3. Kriptográfiai alapok és rendszerek	15
3.1. Kriptográfiai alapfogalmak	15
3.1.1. Titkosítás	15
3.1.2. Kulcs	15
3.1.3. Titkosítási módok	16
3.1.4. Rejtjelezések	16
3.1.5. Hash függvények	17
3.2. Kriptográfia rendszerek	17
3.2.1. Klasszikus titkosítók bemutatása	17
3.2.2. Szimmetrikus titkosítók bemutatása	18
3.2.3. Hash függvények bemutatása	18
3.3. Feltörési módszerek	19
4. Szoftver	20
4.1. Felhasználói követelmények	20
4.2. Rendszerkövetelmény	21
4.2.1. Funkcionális	21
4.2.2. Nem funkcionális	22
4.3. A rendszer architektúrája	24
5. Tervezés és megvalósítás	25

5.1. Könyvtárak	26
5.2. Jinja2	29
6. Felhasználói dokumentáció	30
6.1. Kezdőoldal	30
6.2. Tanulói felület	30
6.3. Crypt felület	30
7. Továbbfejlesztési lehetőségek	32
Összefoglaló	33
Köszönetnyilvánítás	34
Ábrák jegyzéke	35
Irodalomjegyzék	36

1. fejezet

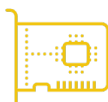
Bevezető

A digitális korban a kriptográfia egyre növekvő jelentőséggel bír, hiszen az adatbiztonság és az adatvédelem kulcsfontosságú tényezők az információtechnológia területén. A kriptográfiai rendszerek megértése és hatékony használata elengedhetetlen az adatok titkosításához, visszafejtéséhez és biztonságos átvételéhez.

Célja a dolgozatomnak, hogy bemutassa a Cryptorithm nevű webes alkalmazást, amely egy kiterjedt platformot kínál a kriptográfiai rendszerek gyakorlati kipróbálására és tanulására. Az alkalmazás lehetővé teszi a felhasználók számára, hogy interaktív módon felfedezzék a különböző kriptográfiai algoritmusokat és azok alkalmazási területeit, mint például az SHA, Whirlpool és bcrypt hash függvények, a Caesar, Affin és ChaCha20 folyamtitkosítók vagy az AES és Blowfish blokktitkosítók. Emellett a Cryptorithm egy részletes tudásbázist is nyújt, amely segíti a felhasználókat az algoritmusok eredetének, működésének és felhasználásának megértésében.

A Cryptorithm alkalmazás a Flask, egy Python alapú könnyű súlyú webes keretrendszerre épül, amely lehetővé teszi a gyors és hatékony fejlesztést. Az alkalmazás intuitív és felhasználóbarát felülettel rendelkezik, hogy könnyű legyen a navigáció és a kriptográfiai rendszerek gyakorlati alkalmazása. Az alkalmazás többnyelvű támogatást is biztosít, így bárki, függetlenül az anyanyelvétől, elérheti és használhatja a platformot.

Cryptorithm



1.1. ábra. Cryptorithm logó

1.1. Motiváció és Célkitűzések

Kriptográfiai tudatosság növelése: Az alkalmazás célja, hogy növelje a felhasználók kriptográfiai tudatosságát és ismereteit. A projekt fő motivációja a felhasználók tájékozottságának növelése a digitális biztonság terén, segítve őket a kriptográfiai rendszerek megértésében és alkalmazásában.

Gyakorlati tapasztalat nyújtása: Lehetőséget kapnak a felhasználók, hogy gyakorlatban is kipróbálják és megtapasztalják a kriptográfiai rendszerek működését.

Tanulás és oktatás támogatása: Egy tanulói felületet van biztosítva, ahol a felhasználók elmélyülhetnek a kriptográfia terén. Motivációs tényező a tudásátadás és az oktatás támogatása, amely segít a felhasználóknak a kriptográfiai rendszerek megértésében és elsajátításában.

Nyitottság és közösség támogatása: Az alkalmazás lehetőséget nyújt az idegen anyanyelvű felhasználóknak is a használatra, hogy bővítsék tudásukat a kriptográfiai rendszerekről.

2. fejezet

Programok, technológiák bemutatása

2.1. Az alkalmazás felépítése

2.1.1. Frontend

A frontend fejlesztéséhez HTML-t, CSS-t és JavaScriptet használtam, amelyek lehetővé teszik az interaktív felhasználói felület létrehozását és a dinamikus funkcionalitás implementálását. Ezek a technológiák biztosítják a felhasználók számára az intuitív böngészési élményt és a kriptográfiai rendszerek kipróbálását. Valamint a Jinja2 sablonmotort használtam, hogy segítsen a dinamikus tartalmú weboldalak generálásával, amelynek használatát egy külön szekcióba fejttem ki. (5.2)

HTML: Egy szabványosított jelölőnyelv, amelyet a weboldal strukturálására és tartalma megjelenítésére használtam. Segítségével definiáltam az elemeket, például címeket, szövegeket, képeket, hivatkozásokat stb.

CSS: Egy stílusleíró nyelv, amelyet a weboldal megjelenítési formázására használtam. Azáltal, hogy különböző stílusokat és tulajdonságokat adtam meg az elemeknek, mint például a szín, a betűtípus, a méret, a margók, a pozíció stb., a CSS lehetővé tette a weboldal testreszabását és a vizuális vonzereje növelését.

JavaScript: Egy programozási nyelv, amelyet az alkalmazás interaktív funkcióinak megvalósítására használtam. A JavaScript lehetővé teszi a weboldalam dinamikus működését, a felhasználói interakciókat, az adatok feldolgozását és a weboldalhoz kapcsolódó események kezelését.



2.1. ábra. HTML - CSS - JS logók

2.1.2. Backend

Backend oldalon a szerveroldali programozáshoz a Python programozási nyelvet választottam, amely nagy népszerűségnek örvend Flask keretrendszerrel kombinálva. A Python és a Flask együtt lehetővé teszik a hatékony és gyors fejlesztést, valamint az API-k létrehozását és a kérések kezelését.

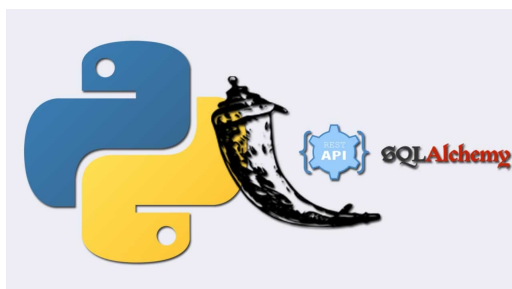
Az adatbázis kezeléséhez az SQLAlchemy nevű Python könyvtárat használtam, melynek segítségével könnyedén lehet adatmodelleket definiálni és adatbázisműveleteket végezni. Ezáltal hatékony és strukturált adatkezelést biztosít az alkalmazásban.

Ezen technológiák kombinációja lehetővé teszi az alkalmazás teljeskörű működését, az interaktív felhasználói felülettől kezdve az adatok kezeléséig és tárolásáig. Az alkalmazás fejlesztése során a modern frontend és backend technológiák összekapcsolása révén egy hatékony és felhasználóbarát környezetet hoztam létre a kriptográfiai rendszerek ki próbálásához és tanulásához.

Python: A Python egy magas szintű programozási nyelv, amelyet egyszerű és olvasható szintaxisa jellemez. A Python nagyon rugalmas és sokoldalú, és széles körben használják a webfejlesztés, adatelemzés, mesterséges intelligencia, gépi tanulás és sok más területen. A Pythonban kifejezőerő és kényelem található, ami lehetővé teszi a gyors és hatékony fejlesztést. Emellett a Python gazdag könyvtárökoszisztémával rendelkezik, amely számos előre elkészített funkcióval és modullal bővíti a fejlesztési lehetőségeket.

Flask: A Flask egy könnyűsúlyú és rugalmas webes keretrendszer Pythonban. A Flask lehetővé teszi a webalkalmazások gyors és hatékony fejlesztését, minimális konfigurációval és egyszerű szintaxisával. A Flask kiváló választás olyan projektekhez, amelyek kisebb méretűek vagy kevesebb komplexitást igényelnek, ugyanakkor rugalmasságot és skálázhatóságot biztosítanak. A Flask lehetőséget nyújt a beépített funkciók, mint például útvonalak, nézetek, sablonok és adatbázis kezelése egyszerű implementálására.

SQLAlchemy: Az SQLAlchemy egy Python alapú ORM (Object-Relational Mapping) keretrendszer, amely lehetővé teszi az adatbázisokhoz való könnyű és hatékony hozzáférést. Az ORM segítségével a fejlesztők Python objektumokat tudnak kezelni és manipulálni, miközben azokat az adatbázisban tárolják. Ezáltal az SQLAlchemy elrejtí az adatbázis-specifikus részleteket és absztrakciós réteget biztosít a programozó számára.



2.2. ábra. Python - Flask - SQLAlchemy logók

2.2. Hasonló platformok, szoftverek összehasonlítása

Cryptii [Cry]: A Cryptii egy online titkosítási és dekódolási weboldal, amely algoritmusok széles skáláját támogatja. Különböző szimmetrikus és aszimmetrikus titkosítási algoritmusok közül választhat. A weboldal emellett lehetőséget biztosít a kódolásra, a hashelésre és a tömörítésre is. A bevétel és kimenet között lehetőség van különböző rétegeket beiktatni, így a két végpont között több algoritmus is alkalmazható.

Encode-Decode [EnD]: Az Encode-Decode egy olyan weboldal, amely különböző algoritmusokhoz kínál titkosítási és dekódolási szolgáltatásokat. A weboldal egy egyszerű felületet biztosít, ahol beírhatja a szöveget, és kiválaszthatja a kívánt titkosítási vagy visszafejtési módszert.

A Cryptii és Encode-Decode platform sem ad leírást a használt algoritmusokról, így ezeket csak úgy lehet hatékonyan használni amennyiben ismerjük azokat, vagy más forrásokból tájékozódunk róluk.

CyberChef [CyC]: A CyberChef egy hatékony online eszköz az adatmanipulációhoz és a titkosításhoz/dekódoláshoz. A műveletek széles skáláját kínálja, beleértve az olyan titkosítási algoritmusokat, mint az AES, RSA, XOR és még sok más. Támogatja továbbá a kódolást, a hashelést, a tömörítést és más adattranszformációkat.

Rumkin [Rum]: A Rumkin egy olyan weboldal, amely kriptográfiai eszközök széles gyűjteményét kínálja. Különböző algoritmusokhoz kínál lehetőségeket, mint például a Caesar rejtjelezés, a Rail Fence rejtjelezés, a Playfair rejtjelezés és még sok más. Emellett kódolásra, hashelésre és steganográfiára szolgáló eszközöket is tartalmaz.

dCode [dCo]: A dCode egy olyan online platform, amely titkosítási és dekódolási eszközök hatalmas gyűjteményét kínálja. Algoritmusok, titkosítások és kódolások széles skáláját fedi le, beleértve a klasszikus titkosításokat, modern titkosítási módszereket és speciális kódolásokat. Emellett kriptóanalízishez is kínál eszközöket.

Az utolsó három alkalmazás esetében már több, részletesebb leírást lehet kapni az algoritmusokról, bár már kevésbé felhasználóbarátok, mint az első két platform.

3. fejezet

Kriptográfiai alapok és rendszerek

3.1. Kriptográfiai alapfogalmak

A Cryptorithm rendszer használata során néhány alapvető kriptográfiai fogalom ismerete előnyös lehet. Az alábbiakban néhány ilyen fogalmat tisztázok, amelyek segíthetnek a rendszer hatékony használatában.

3.1.1. Titkosítás

A titkosítás olyan folyamat, amely során az eredeti üzenetet (nyílt szöveget) átalakítjuk egy titkosított formává, hogy csak a jogosultak tudják elolvasni. Az alkalmazásban található kriptográfiai rendszerek segítségével a felhasználók titkosíthatnak és visszafejthetnek üzeneteket.

3.1.2. Kulcs

A kulcs a kriptográfiában egy olyan titkosítási vagy visszafejtési folyamat során használt információdarab, amely alapvetően befolyásolja a kriptográfiai algoritmus működését és a végeredményt. Biztosítják az adatok biztonságát és védelmet nyújtanak a jogosulatlan hozzáférésekkel szemben. Ugyanakkor fontos megjegyezni, hogy az erős és biztonságos kulcs kiválasztása és ennek tárolása, kezelése elengedhetetlen a megfelelő kriptográfiai rendszerek kialakításában.

3.1.3. Titkosítási módok

A titkosítási módok meghatározzák, hogy a titkosítás hogyan történik az üzenetek blokkjainak kezelése során. A gyakran használt titkosítási módok a következők: ECB(Electronic Codebook), CBC(Cipher Block Chaining), CFB(Cipher Feedback), OFB(Output Feedback) és a CTR(Counter). A felsorolt üzemmódok mellett számos más speciális üzemmód létezik, amelyek különböző funkciókat és jellemzőket kínálnak. A megfelelő kiválasztása és alkalmazása az adott kriptográfiai algoritmustól függ.

A Cryptorithm esetében az AES blokk titkosító CTR üzemmódban próbálható ki, ami azt jelenti, hogy a blokkok titkosítása egy folyamatosan növekvő számláló alapján történik. A számláló értékét XOR-olják az adatblokkal a blokk titkosítása előtt. Ez a módszer párhuzamosításra alkalmas, mivel a blokkok függetlenül titkosíthatóak.

3.1.4. Rejtjelezések

A dolgozatomban 2 fajta rejtjelezést, avagy titkosítást különböztetek meg, a **klasszikus** és a **szimmetrikus** titkosítókat. A klasszikus titkosítókhoz tartoznak a Caesar és Affin rejtjelezések, ezeket a rendszereket ma már csak kizárólag példaképpen használják a bevezetéshez a kriptográfia világába. A szimmetrikus titkosítás az jelenti, hogy mind a titkosításra, mind a visszafejtésre ugyanazt a kulcsot használják, ide tartoznak a ChaCha20, az AES és a Blowfish rejtjelezések. Továbbá mindegyik rejtjelezés két féle lehet, blokk- vagy folyamtitkosító.

A **blokktitkosítók** a bemeneti adatot blokkokra osztják, majd ezeket a blokkokat titkosítják. A blokkok mérete általában azonos, és ugyanazt a titkosítási kulcsot és algoritmust használják. Ilyen titkosító például az AES és a Blowfish. Előnyei a hatékonyság, mivel a blokkok egyszerre titkosíthatóak és dekódolhatóak, továbbá hibajavítási képessége, mert egy hibás blokk nem terjed át a többi blokkra.

A **folyamtitkosítók** bitenként vagy bitek kis csoportjaival dolgozik, és a titkosítás során egy folyamot generál. A folyamtitkosítás az adatokat a generált kulcs folyammal kombinálja. Előnyei közé tartozik a sebesség és az adatfolyam feldolgozásának hatékonysága, mivel az adatokat bitek szintjén lehet titkosítani és dekódolni. Emellett a folyamtitkosítás ellenállóbb lehet a blokktitkosítással szembeni kriptóanalitikai támadásokkal szemben, mivel nem állít elő azonos bemenetekhez azonos kimeneteket. Ilyen titkosító például a ChaCha20.

3.1.5. Hash függvények

Az alkalmazás a rejtjelezők mellett még lehetőséget ad a hash függvények tanulmányozására is. A hash-függvények olyan matematikai algoritmusok, amelyek egy tetszőleges hosszúságú bemenetet (pl. üzenetet) átalakítanak egy fix hosszúságú, látszólag véletlenszerű kimenetbe. A hashelés segít a felhasználóknak az adatok hitelességének ellenőrzésében. Továbbá erejük és megbízhatóságuk abban rejlik, hogy irreverzibilisek, nem lehet visszaállítani a hash értékéből a bemeneti adatot eredeti formájában és ugyanakkor ütközmentesek, vagyis nagyon alacsony valószínűséggel található két különböző bemenet, amelyek ugyanahhoz a hash értékhez vezetnek. Ilyen függvény például az SHA, a Whirlpool és a bcrypt.

3.2. Kriptográfia rendszerek

A következőkben megemlítem azokat a kriptográfiai módszereket amelyek elérhetőek a Cryptorithm alkalmazásban.

3.2.1. Klasszikus titkosítók bemutatása

A **Caesar** rejtjelezés Julius Caesarról, az ókori római katonai és politikai vezetőről kapta a nevét, aki a feltételezések szerint ezt a titkosítási technikát használta. Egy egyszerű eltolásos titkosítási módszer, amelyben az összes betűt egy adott számmal, a kulcsként használt eltolással helyettesítik. Például, ha a kulcs 3, akkor az "A" betűt a "D" betűre cserélik, a "B" betűt az "E" betűre stb. A Caesar rejtjelezés könnyen feltörhető, mivel csak 26 lehetséges eltolási kulcs létezik, amelyeket egyszerűen végig lehet próbálni.

Az **Affin** rejtjelezés egy egyszerű szubsztitúciós rejtjelezési módszer, amely a Caesar rejtjelezésre épül. Az Affin rejtjelezés egy lineáris transzformációt alkalmaz a betűkön, amely egy egyenlet alapján helyezi át azokat. Az Affin rejtjelezés a kulcsként használt két paraméter segítségével végez transzformációt az üzeneten. Az Affin rejtjelezés gyenge pontja, hogy az egyszerű frekvenciaanalízis módszerekkel feltörhető lehet, különösen kis méretű kulcsok esetén.

Fontos megjegyezni, hogy az Affin és Caesar rejtjelezések gyakorlati alkalmazásban már nem számítanak biztonságosnak, mivel könnyen feltörhetők.

3.2.2. Szimmetrikus titkosítók bemutatása

A **ChaCha20** elsősorban szimmetrikus titkosítási algoritmusként és folyamtitkosítóként használatos. Gyakran alkalmazzák hálózati protokollokban, biztonságos kommunikációban és nagy teljesítményű titkosítást igénylő alkalmazásokban. A ChaCha20 a sebességéről és biztonságáról ismert, és a régebbi titkosítók, például az RC4 alternatívájaként vált népszerűvé. [Kno19]

Az **AES** egy szimmetrikus blokktitkosítási algoritmus, amelyet a biztonságos adatátvitel és tárolás céljából használnak. Az AES algoritmus blokkokat titkosít a bemeneti üzenetből, és ezek a titkosított blokkok kombinálódnak a kimeneti titkosított üzenet létrehozásához. Az AES-t az Egyesült Államok Kormánya ajánlja a kormányzati és ipari alkalmazásokban. Az AES több különböző kulcsmérettel (128, 192, 256 bites) és különböző üzemmódokkal (pl. CTR, CBC, ECB) használható. [oST01]

A **Blowfish** egy szimmetrikus kulcsú blokktitkosító, amely 64 bites blokkokkal dolgozik, és változó kulcsméreteket támogat (32 bittől 448 bitig). Gyakran használják olyan alkalmazásokban, amelyek érzékeny adatok titkosítását és visszafejtését igénylik. A Blowfish-t széles körben alkalmazzák protokollokban, szoftverekben és hardveres rendszerekben a titkosság és az adatvédelem biztosítása érdekében. [Sch93]

3.2.3. Hash függvények bemutatása

A **Whirlpool** egy hash-függvény, amelyet általában adatintegritás-ellenőrzésre és digitális aláírásra használnak. Fix méretű, jellemzően 512 bit hosszú hash-értéket állít elő. A Whirlpoolt különböző biztonsági alkalmazásokban használják, például a fájlok integritásának ellenőrzésére, jelszavak tárolására és üzenethitelesítésre. [Bar03]

Az **SHA** egy hash függvény család, amelyeket a digitális adatok integritásának ellenőrzésére és az adatok egyedi azonosítására használnak. Az SHA hash függvények, például az SHA-1, SHA-256 stb., egy adott bemeneti üzenetet átalakítanak egy fix hosszú hash kóddá. Az SHA algoritmusok irreverzibilisek, vagyis a hash értékből nem lehet visszaállítani az eredeti üzenetet. Ez a tulajdonságuk hasznos a jelszavak, digitális aláírások és az üzenetek integritásának védelmében.

A **bcrypt** egy hash függvény, amelyet leginkább jelszavak biztonságos tárolására és ellenőrzésére használnak. Az algoritmus tervezésekor arra törekedtek, hogy lassú legyen a hash függvények hasítási sebessége, hogy megnehezítse a jelszavak visszafejtését a hash értékből. Az algoritmus egy sor iterációt végez a jelszó hasításához, és egy só (salt) értéket használ, amely tovább növeli a biztonságot. A jelszó karakterlánc, egy számokban kifejezett költség és egy sóérték a bcrypt algoritmus bemenete. Ezeket a bemeneteket a bcrypt függvény egy 24 bájtos (192 bites) hash létrehozására használja. [PN99]

3.3. Feltörési módszerek

A kriptográfia algoritmusok célja, hogy megvédjék az adatokat az illetéktelen hozzáféréstől. Azonban az informatikai fejlődéssel és az idő múlásával párhuzamosan megjelentek bizonyos feltörési módszerek is, melyeknek célja, hogy az adatokat megpróbálják visszafejteni eredeti, bemeneti formájukra. Az alábbiakban bemutatok néhány gyakran használt kriptográfiai feltörési módszert:

- **Brute force:** Ez a módszer az összes lehetséges kulcs kipróbálását jelenti, amíg megtaláljuk a helyes kulcsot. Bár hatékony kriptográfiai algoritmusok esetén ez időigényes és gyakran megvalósíthatatlan feladat, bizonyos esetekben hatékony lehet.
- **Frekvenciaanalízis:** Ez a módszer az adott nyelven használt betűk gyakoriságának elemzésén alapul. A gyakrabban előforduló betűk azonosítása segíthet a titkosított szöveg visszafejtésében, különösen, ha az adott nyelvre jellemző mintázatokat lehet azonosítani. Bizonyos algoritmusok esetében ez teljesen kivitelezhetetlen mivel a titkosított szövegben nincsen nyelvi jellegű információ, amely alapján ki lehetne használni a betűk gyakoriságát.

4. fejezet

Szoftver

4.1. Felhasználói követelmények

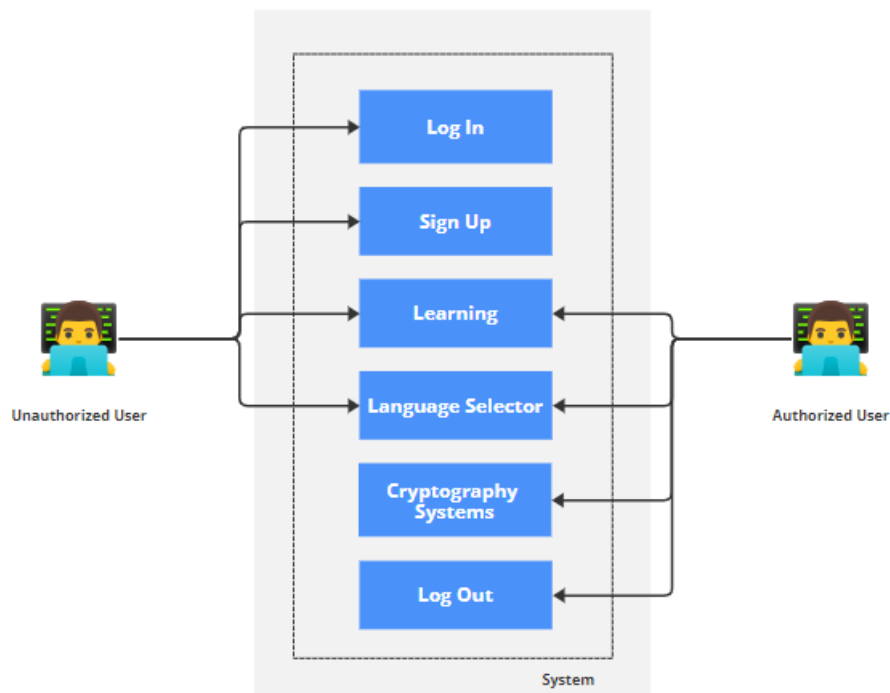
- **Regisztráció és bejelentkezés:** Lehetőség van regisztrációra az alkalmazásban, amely után a felhasználók bejelentkezhetnek a személyes fiókjukba. Ez lehetővé teszi számukra a mentett adatokhoz való hozzáférést.
- **Kriptográfiai rendszerek kiválasztása:** Különböző kriptográfiai rendszerek elérhetőek, amelyeket bejelentkezés után lehet kipróbálni. Ehhez megjelenik egy menü, ahol kényelmesen navigálhatnak a rendszerek között.
- **Rendszerek kipróbálása és tesztelése:** A felhasználóknak lehetősége van arra, hogy kipróbálják és teszteljék a választott kriptográfiai rendszereket. Ehhez felhasználóbarát és interaktív felületet van biztosítva, ahol megadhatják a bemeneti adatokat, és láthatják a kimeneti eredményeket.
- **Tanulási anyagok és információk elérhetősége:** Az alkalmazásnak tartalmaz egy tanulói felületet, ahol a felhasználók elérhetik a kriptográfiai rendszerekhez kapcsolódó részletes információkat és szemléltető anyagot. Ennek az oldalnak az eléréséhez nem szükséges a bejelentkezés.
- **Nyelvi támogatás:** Az alkalmazás lehetőséget ad a felhasználóknak arra, hogy különböző nyelveken használják az alkalmazást. Ehhez egy egyszerű nyelv kiválasztó van biztosítva, amely lehetővé teszi a felhasználóknak a kívánt nyelv kiválasztását. A nyelvek rövidítve, ISO 639.1-es kódjuk szerint vannak feltüntetve.

4.2. Rendszerkövetelmény

A Cryptorithm rendszerkövetelménye funkcionális és nem funkcionális részekre oszlik. A funkcionális része tartalmazza az alkalmazás fő céljait és funkcionalitását, hogy hogyan kéne működjön a rendszer és milyen lehetőségei vannak a felhasználónak, míg a nem funkcionális rész kitér a rendszerrel szemben támasztott követelményekre, mint például a felhasználói élmény, többnyelvűség és a rendszer architektúrája.

4.2.1. Funkcionális

- **Kriptográfiai eszközök:** Az alkalmazás választékot kínál a népszerű és ismert eszközökből az adatok titkosításához, visszafejtéséhez és transzformációjához. A felhasználók szabadon válogathatnak és kipróbálhatnak különböző lehetőségeket.
- **Tanulói felület:** Az alkalmazás rendelkezik egy tanulói felülettel, ahol a felhasználók elmélyülhetnek a kriptográfiai rendszerek megértésében. Itt információkat olvashatnak és gyakorlati példákon keresztül elsajátíthatják ezeket a rendszereket.
- **Többnyelvűség:** Az alkalmazás támogatja a többnyelvűséget, így elérhetővé válik idegen nyelvű felhasználók számára is. A felhasználók kiválaszthatják az anyanyelvüket vagy egyéb preferált nyelvet az alkalmazás használatához.
- **Visszajelzés:** Amennyiben ismert kivétel keletkezik a rendszerben ezt felugró üzeneten keresztül van leköszölve, ami ismerteti a hibát. Kivétel keletkezhet nem megfelelő beviteli adatok nyomán, például rossz vagy nem megfelelő kulcsméret megadása, vagy olyan oldalra való navigáció ahova a felhasználónak nincsen jogosultsága. Az előző esetet a hibának megfelelő üzenettel közli, míg utóbbi egy az érvényben levő jogosultságnak megfelelő oldalra navigálja a felhasználót.
- **Szerepkörök:** A rendszert két fajta felhasználó számára elérhető, bejelentkezett és nem bejelentkezett. A bejelentkezett felhasználónak elérhető a weboldal összes funkcionalitása, ezzel szemben a nem bejelentkezett felhasználó nem jogosult a kriptográfia rendszerek használatára.



4.1. ábra. Use case diagram

4.2.2. Nem funkcionális

- **Felhasználói élmény:** Az alkalmazás könnyen használható felhasználói felülettel rendelkezik. Az intuitív navigáció és a felhasználóbarát tervezés lehetővé teszi a felhasználók számára a könnyű kezelést és a zökkenőmentes interakciót az alkalmazással. Továbbá megfelelő felugró üzenetekkel jelez vissza a rendszer minden művelet elvégzése után.
- **Teljesítmény:** Az alkalmazás működése gyors és hatékony. Az oldalak betöltése és az adatbázis lekérdezések időigénye egyáltalán vagy csak enyhén észlelhetőek.
- **Biztonság:** Felhasználói adatok védve vannak, megfelelően vannak tárolva. A felhasználók jelszava SHA-256-os hash formájukban vannak eltárolva az adatbázisban, valamint a kulcsok és inicializáló vektorok kigenerálása a kritériumoknak megfelelő.
- **Skálázhatóság:** Az alkalmazás képes kezelni a megnövekedett felhasználói forgalmat és rugalmasan skálázódni. Szükség esetén növelhetőek az erőforrások, például adatbázis kapacitása vagy a processzor teljesítménye.

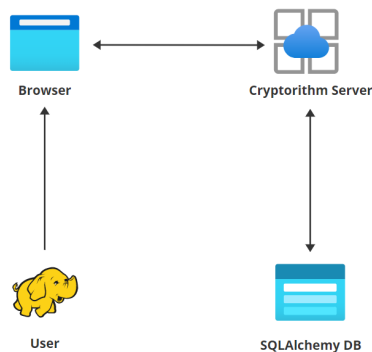
- **Hibatűrés:** Hibakezelés és hibajavítás mechanizmusainak megléte, az alkalmazásnak ellenálló képessége van a hibákhoz. Rendszer leállás esetén az újraindítás nem időigényes. Az ismert, egyedi vagy lehetséges hibafaktorokra a megfelelő reakciók biztosítása, például nem megfelelő paraméterek megadása esetén hibaüzenetek kiírása és kivételek kezelése.
- **Kompatibilitás:** A weboldal felülete főként asztali számítógépek, laptopok képernyőjére van optimalizálva, továbbá támogatja a népszerű böngészőket.



4.2. ábra. Népszerűbb Böngészők

4.3. A rendszer architektúrája

Az alkalmazás architektúrája a következő komponenseket kombinálja, hogy a felhasználók kényelmesen használhassák a Cryptorithm alkalmazást. A felhasználói interfész réteg lehetővé teszi a felhasználók interakcióját az alkalmazással, az üzleti logika réteg hajtja végre a szükséges műveleteket és feldolgozza a kéréseket, míg az adatbázis réteg biztosítja az adatok tartós tárolását és kezelését. Az architektúra segít a rendszer komponenseinek szétválasztásában és a fejlesztés hatékonyságának növelésében.



4.3. ábra. Rendszer Architektúra

Felhasználói interfész: Ez a rész felelős az alkalmazás felhasználói felületének megjelenítéséért és a felhasználóval való interakcióért. Megvalósítása az alkalmazásban HTML, CSS és JavaScript segítségével van kivitelezve. Ez a réteg jeleníti meg a kriptográfiai rendszerek kiválasztására, tesztelésére és a tanulói anyagokhoz való hozzáférésre szolgáló felületeket.

Üzleti logika: Az üzleti logika réteg felelős az alkalmazás működéséért, a felhasználók által végrehajtott műveletek feldolgozásáért és az eredmények előállításáért. Itt találhatóak a szerveroldali Python fájlok, amelyek a kriptográfiai műveletek, adatfeldolgozás és adatbázis-interakciók végrehajtásáért felelősek. Flask keretrendszert használva ez a réteg kezeli a HTTP kéréseket és válaszokat.

Adatbázis: Az alkalmazás adatbázisában tárolódnak a felhasználói adatok, például a regisztrált felhasználók adatai és előzményei. Az adatbázis kezelésére az SQLAlchemy Python könyvtárat használtam, amely lehetővé teszi a könnyű adatbázis-műveletek végrehajtását és az adatmodell definiálását.

5. fejezet

Tervezés és megvalósítás

A projektet Visual Studio Code segítségével fejlesztettem, amely egy ingyenes és nyílt forráskódú fejlesztői környezet. Bár alapvetően könnyűsúlyú, ennek ellenére erőteljes szövegszerkesztő, mivel számos programozási nyelvet és technológiát kínál. A telepítése után nagyban testreszabhatjuk saját igényeink szerint és számos egyéb funkcióhoz juthatunk a bővítmények telepítésével.

6 részre bontottam a projektet: instance, cryptorithms, languages, static, templates, cryptorithm

- **Instance:** Az instance mappában van eltárolva az adatbázis lokálisan, melynek modelje a models.py fájlban van definiálva.
- **Cryptorithms:** A cryptorithms mappában levő fájlokban kerültek kivitelezésre és implementálásra a kriptográfiai rendszerek.
- **Languages:** A nyelvekért felelős .json kiterjesztésű fájlok itt vannak tárolva, itt van egy meghatározott struktúrában lefordított tartalma az oldalnak.
- **Static:** A static alatt a .css és .js fájlok kerültek létrehozásra, melyek a weboldal dizájnért felelősek.
- **Templates:** A template mappába kerültek az alap .html fájlok amelyek a tartalom megjelenítéséért felelősek.
- **Cryptorithm:** A fő mappa a Cryptorithm amely magába foglalja az előbb említett részeket, valamint az API kéréseket és válaszokat, a szessziókezeléshez, a projekt inicializálásához szükséges fájlokat, továbbá az adatbázis adatmodelljét és a projektet indító main fájlt.

```

Cryptorithm
├── app.py
├── instance
│   └── database.db
├── website
│   ├── cryptorithms
│   │   ├── streamCipher.py
│   │   ├── blockCipher.py
│   │   └── hashing.py
│   ├── language
│   │   ├── en.json
│   │   └── hu.json
│   ├── static
│   │   ├── images
│   │   ├── base.js
│   │   ├── cryptorithm.css
│   │   ├── learning.css
│   │   └── learning.js
│   ├── templates
│   │   ├── base.html
│   │   ├── cryptorithm.html
│   │   ├── home.html
│   │   ├── learning.html
│   │   └── learnItems
│   │       ├── aes.html
│   │       ├── caesar.html
│   │       ├── blowfish.html
│   │       ├── whirlpool.html
│   │       ├── bcrypt.html
│   │       ├── afffin.html
│   │       ├── caesar.html
│   │       ├── sha1.html
│   │       └── sha2.html
│   ├── auth.py
│   ├── views.py
│   ├── models.py
│   └── __init__.py

```

5.1. ábra. Rendszer fájl struktúrája

5.1. Könyvtárak

A következőekben bemutatom a jelentősebb Python könyvtárakat, melyek segítettek az alkalmazás implementálásában.

- **Cryptography:** Kriptográfiai funkciókat és algoritmusokat kínál. Ez a könyvtár ideális választás a projektben, mivel számos kriptográfiai műveletet valósíthatunk meg vele. A Cryptography megbízható és jól dokumentált eszköz a kriptográfiai műveletek biztonságos végrehajtásához. Ennek a könyvtárnak a segítségével lett kivitelezve az AES rendszer.
- **Flask:** A projektben a Flask keretrendszert használom a webes alkalmazás felépítéséhez és a kérések kezeléséhez. A Flask könnyen tanulható és rendkívül rugalmas, ami lehetővé tette, hogy könnyedén kialakítsam a vágyott funkciókat és testreszabhasam az alkalmazást.

A következő kódrészlet a tanulói oldal betöltésének kérését mutatja be:

```
@views.route('/learning/<language>', methods=['GET', 'POST'])
def learning(language):
    if(language not in languages):
        language = app_language
    return render_template("learning.html", user=current_user,
        language=language, **languages[language])
```

5.1. kódrészlet. Learning oldal renderelése

- **FlaskSQLAlchemy:** A FlaskSQLAlchemy egy könnyen használható és hatékony ORM (Object-Relational Mapping) könyvtár, amely lehetővé teszi az adatbázis műveletek kezelését a Flask alkalmazásban. Az SQLAlchemy révén a FlaskSQLAlchemy segít az adatbázis kapcsolatok kezelésében, az adatmodell osztályok definiálásában és az adatbázis műveletek végrehajtásában. A FlaskSQLAlchemy használata átlátható és hatékony adatbázis-interakciókat tesz lehetővé a projektben.

A következő kódrészlet az adatbázis modellek implementálását mutatja be:

```
class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    email = db.Column(db.String(150), unique=True)
    password = db.Column(db.String(150))
    first_name = db.Column(db.String(150))
    datas = db.relationship('Data')

class Data(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    data = db.Column(db.String(10000))
    cryptype = db.Column(db.String(100))
    paramA = db.Column(db.String(100), nullable=True)
    paramB = db.Column(db.String(100), nullable=True)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))
```

5.2. kódrészlet. Adatbázis modellek

- **FlaskLogin:** A FlaskLogin egy hasznos kiegészítő a Flask keretrendszerhez, amely segít az autentikáció és az azonosítás kezelésében a webes alkalmazásban. Ennek segítségével egyszerűen implementálhattam a felhasználói regisztrációt, bejelentkezést és kijelentkezést a rendszerben. Ez a könyvtár felelős a felhasználói munkamenetek és a hozzáférési jogosultságok kezeléséért.

A következő kódrészletek a flask login modulja egy szemléltetése a regisztrációra és kijelentkezésre:

```
new_user = User(email=email, first_name=first_name,
    password=generate_password_hash(password1, method='sha256'))
db.session.add(new_user)
db.session.commit()
login_user(new_user, remember=True)
```

5.3. kódrészlet. Flask bejelentkezés kódrészlet

```
@auth.route('/logout/<language>', methods=['GET'])
@login_required
def logout(language):
    logout_user()
    return redirect(url_for('auth.login', language=language))
```

5.4. kódrészlet. Flask kijelentkezés kódrészlet

- **Json:** A JSON (JavaScript Object Notation) egy könnyen olvasható és írható adatformátum, amely széles körben használatos az adatok strukturált tárolására és átvitelére. A JSON könyvtárat használva a projektben könnyedén kezelhettem a JSON adatokat.

A alábbiakban látható a megvalósítás angol nyelvhez:

```
"cryptorithm": {
    "login": "Login",
    "email": "Email",
    "password": "Password",
    "no_account": "Don't have an account?",
    "signup": "Sign Up",
    "already_account": "Already have an account?"
}
```

5.5. kódrészlet. JSON angol kódrészlet

- **Egyebek:** Mindezekután szeretném megemlíteni azokat a könyvtárakat amelyek csak néhány művelet elvégzésére voltak igénybevéve, de ennek ellenére fontos részét képezik a rendszernek.

- Base64
- Glob
- Os
- Math
- Hashlib
- Random

5.2. Jinja2

A Jinja2 egy erőteljes és rugalmas sablonmotor Python nyelvhez. Arra tervezték, hogy segítsen a dinamikus tartalmú weboldalak generálásában a sablonok és adatok kombinálásával. A Jinja2 a Flask keretrendszer alapértelmezett sablonmotorja, de függetlenül is használható más projektekben. Egyszerű és kifejező szintaxissal rendelkezik, amely hasonlít a HTML-hez, viszont tartalmaz kiegészítő jelöléseket és vezérlőstruktúrákat.

Fő alapvető jellemzője és funkciója amit használtam a következő: **Sablonok:** Lehetővé teszi a HTML-szerű sablonok létrehozását, amelyekben helytartókat (placeholder) és vezérlőstruktúrákat használhatunk. A helytartók fogadják a dinamikus adatokat, amelyeket a sablonmotor behelyettesít az adott oldalra. Ezzel a funkcióval valósítottam meg, hogy a weboldal tartalma igény szerint idegen nyelvűek számára is használható legyen. Az alábbi kód szemlélteti ennek használatát, ahol a dupla-kapcsos zárójelek közötti változók a Jinja2 elemei amelyeket a nekik megfelelő json fájlok tartalmával tölt fel:

```
<button type="submit" class="btn" value="login"
    name="submit_button">{{cryptorithm.login}}</button>
<div class="login-signup">
    <p>{{cryptorithm.no_account}}</p>
    <a href="#" class="signup-link">{{cryptorithm.signup}}</a>
</div>
```

5.6. kódrészlet. Jinja2 sablon kódrészlet

6. fejezet

Felhasználói dokumentáció

6.1. Kezdőoldal

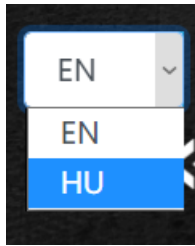
Az oldal látogatóit egy üdvözlőszöveg köszönti, amely röviden összefoglalja a felhasználó lehetőségeit az oldalon. Amennyiben nem szeretne a felhasználó egy új profilt létrehozni, vagy bejelentkezni, akkor lehetősége van elnavigálni a tanulói felületre, ahol szabadon böngészheti a jelen levő kriptográfiai rendszereket. Viszont, ha a felhasználó él a bejelentkezés lehetőségével, akkor hozzáférése lesz a műveletek elvégzéséért felelős felülethez is. Továbbá minden oldalon jelen van, egy nyelv választó, mely lehetőséget ad, hogy a felhasználó más nyelven is böngészhessen a platformon (6.1).

6.2. Tanulói felület

A tanulói felület bal oldalán egy menün keresztül lehet navigálni a módszerek között, a kiválasztás után megjelenik 2 (vissza, következő) gomb amely segít az adott rendszerhez tartozó tartalom navigálásában. Ha a felhasználó úgy dönt, hogy még is hozzáférést szeretne az oldal többi funkciójához akkor ezt a belépés fülre kattintva, egy gyors bejelentkezés vagy új profil létrehozása után megkapja.

6.3. Crypt felület

A Crypt felületen lehet kipróbálni a rendszerbe beépített kriptográfiai műveleteket, ehhez rendelkezésre áll egy kiválasztó, ahol jelen vannak az elérhető lehetőségek. A választás után megjelenhet bizonyos számú beviteli mező, ezek a kiválasztott művelet bemeneti paramétereit várják (6.2). Egy művelet sikeres elvégzése után az előzmények táblában megjelenik az eredmény, ezt megelőzően pedig a felhasználó által végrehajtott műveletek előzményei (6.3).



6.1. ábra. Nyelv választó

Choose an algortihm: AES-128 CTR DE

key: nonce:

6.2. ábra. Példa beviteli mezőre - AES-128 CTR DE

History

769f83d0b3662f7493212ec85676cd80d11bb4df6dca7e8ba171868e2322 2fa54a3f790acdc2c9595f9c2be5735c4401 Type: SHA-384	×
9ba25e0c2ba93e2a6682a67bf29d2b07425f7a4b Type: SHA-1	×
Vdslhqwld Type: CAESAR_EN Parameter A: 3	×

Type the plaintext here...

Choose an algortihm: None

Submit

6.3. ábra. Crypt felület

7. fejezet

Továbbfejlesztési lehetőségek

Bővített kriptográfiai rendszerek: Az alkalmazásba további kriptográfiai rendszerek, mint például Salsa20, ElGamal vagy ECC (Elliptic Curve Cryptography) integrálása lehetőséget adna a felhasználóknak a különböző algoritmusok kipróbálására és tanulására. Továbbá megfontolandó lenne az AES titkosító további módjai beépítésére.

Interaktív gyakorló feladatok: Interaktív gyakorló feladatok készítése, amelyek segítségével a felhasználók a tanulói felületen gyakorolhatják a különböző kriptográfiai rendszerek alkalmazását. Ez javítaná a gyakorlati tapasztalatok szerzését és a tanulás hatékonyságát.

Felhasználói profilok és eredmények nyomon követése: Egy felhasználói profil rendszer létrehozása, ahol a felhasználók módosíthatják adataikat és nyomon követhetik a teljesítményt, például az elért eredményeket vagy az elvégzett feladatokat. Ez segít a felhasználóknak a fejlődésük nyomon követésében és motivációjuk fokozásában.

További nyelvi támogatás: Az alkalmazást további nyelvek támogatásával való kibővítése, hogy a nem angol vagy magyar anyanyelvű felhasználók is könnyen használhassák és megértsék a rendszert. Ez növelné a felhasználói kört és a felhasználói elégedettséget.

Reszponzív dizájn: Az alkalmazás biztosítása a különböző eszközökön történő használathoz.

Ezek a továbbfejlesztési lehetőségek hozzájárulhatnak az alkalmazás funkcionalitásának és felhasználói élményének javításához, valamint a felhasználók kriptográfiai ismereteinek bővítéséhez.

Összefoglaló

A Cryptorithm egy webes alkalmazás, amely lehetővé teszi a felhasználók számára különböző kriptográfiai rendszerek kipróbálását és tanulását. Az alkalmazásban számos beépített kriptográfiai eszköz áll rendelkezésre, mint például az SHA, Whirlpool és bcrypt hashelés, a Blowfish, ChaCha20 titkosítók, az AES titkosítás CTR módban, valamint az Affin és Caesar rejtjelezések. A felhasználók a felületen keresztül kiválaszthatják, hogy melyik rendszert szeretnék alkalmazni, és gyakorolhatnak a megértéséhez. Az alkalmazás intuitív felhasználói felülettel rendelkezik és támogatja a többnyelvűséget. A Cryptorithm célja, hogy segítse a felhasználókat a kriptográfiai ismeretek bővítésében, miközben egy interaktív és tanulást támogató környezetet biztosít.

GitHub link: <https://github.com/zali07/Cryptorithm>

Köszönetnyilvánítás

Szeretném kifejezni hálámat a családomnak és barátaimnak, akik támogattak és bátorítottak engem ezen az úton. Köszönet illeti a vezető tanáromat is, aki értékes útmutatást és irányítást nyújtott a projekt fejlesztése során.

Ábrák jegyzéke

1.1. Cryptorithm logó	10
2.1. HTML - CSS - JS logók	13
2.2. Python - Flask - SQLAlchemy logók	14
4.1. Use case diagram	22
4.2. Népszerűbb Böngészők	23
4.3. Rendszer Architektúra	24
5.1. Rendszer fájl struktúrája	26
6.1. Nyelv választó	31
6.2. Példa beviteli mezőre - AES-128 CTR DE	31
6.3. Crypt felület	31

Irodalomjegyzék

- [Bar03] Vincent Barreto, Paulo S. L. M. & Rijmen. The whirlpool hashing function. 2003.
- [Cry] Cryptii. <https://cryptii.com>.
- [CyC] Cyberchef. <https://gchq.github.io/CyberChef>.
- [dCo] dcode. <https://www.dcode.fr>.
- [EnD] Encode-decode. <https://encode-decode.com>.
- [Kno19] Heiko Knospe. *A Course in Cryptography*. American Mathematical Society, 2019.
- [oST01] National Institute of Standards and Technology. Announcing the advanced encryption standard (aes). 2001.
- [PN99] Mazières D Provos N. A future-adaptable password scheme. 1999.
- [Rum] Rumkin. <https://rumkin.com>.
- [Sch93] Bruce Schneier. Description of a new variable-length key, 64-bit block cipher (blowfish). 1993.