

## **Abstract**

Due to the fact that more people and companies are relying on online platforms to hold sensitive information, cybersecurity has become an extremely important worry in this era of digital technology. The complete cryptographic solution known as ShadowHash was developed with the intention of satisfying the ever-increasing need for strong security procedures. ShadowHash is a platform that provides a suite of cryptographic services to increase data security and integrity. This article describes the development and implementation of ShadowHash, discussing its development and deployment. ShadowHash is a program that offers users the ability to secure their passwords, encrypt their data, verify the integrity of their data, and identify potential threats. It is built using Java for the backend and HTML/CSS for the frontend, and it is hosted on GitHub and Amazon Web Services Amplify. A virus detection tool, an email breach checker, a password breach checker, a password generator, encryption and decryption techniques, hash generation and comparison tools, and a password generator are among the key features. This document provides an overview of the project's goals, characteristics, and implementation details, as well as the relevance of the project's contributions to the field of cybersecurity.

# Chapter 1

## Introduction

### **Brief Introduction:**

Today, cybersecurity plays an essential role in the digital age. As we all flock to online platforms and store our data inside some server somewhere, the demand for resilient security solutions is also growing. In order to address this growing need a new environment, ShadowHash makes it possible for many cryptographic needs within the enterprise in one place. This paper is about ShadowHash, a comprehensive cryptographic solution intended to offer its users with secure and reliable means of authenticating them along with password storage protection that also protects the files they save on their local disk.

### **Project Objectives:**

ShadowHash's primary goal is to create and roll out a platform that is user-friendly and incorporates essential cryptographic services. This platform seeks to address the diversified security requirements of individuals and organizations by providing a centralized center for encrypting data, verifying data integrity, and detecting potential threats.

### **Key Features:**

ShadowHash offers a comprehensive array of features that are specifically designed to meet the needs of a diverse spectrum of users. Here's a breakdown of its eight primary functionalities:

1. **Email Breach Checker:** This feature utilizes the Hackcheck Woventams API to compare user-supplied email addresses with databases that have been known to contain data breaches. ShadowHash alerts the user and suggests that they immediately change their passwords if a match is found, thereby promoting proactive account security.

2. **Password Breach Checker:** This feature utilizes the HaveIBeenPwned API to verify if a user's password has been compromised in any recent data breaches. To ensure user privacy, ShadowHash employs a secure verification method. It generates a SHA-1 hash of the password and transmits only the first five characters for comparison. This approach effectively safeguards the user's complete password from unauthorized access, adhering to a strict no-log policy for enhanced security.
3. **Password Creator:** ShadowHash integrates a password generation utility that accommodates to both preferences and security best practices. Users can elect for randomly generated strong passwords comprising uppercase, lowercase, alphanumeric characters, and lengths ranging from 8 to 20 characters. Alternatively, the platform allows users to construct personalized passwords based on their specific input, ensuring a balance between memorability and complexity.
4. **Encryption Algorithms:** ShadowHash offers a versatile range of encryption algorithms, encompassing both asymmetric and symmetric methods. Users can leverage the well-established RSA algorithm for asymmetric encryption and decryption. Additionally, ShadowHash provides access to a suite of symmetric algorithms including AES, 3DES, Blowfish, and a custom-developed, robust ShadowHash special algorithm. This empowers users to select the encryption method that best suits their specific security requirements and data sensitivity.
5. **Decryption Algorithms:** Complementing the encryption functionalities, ShadowHash offers decryption capabilities using the same set of algorithms: RSA for asymmetric decryption and AES, 3DES, Blowfish, and the ShadowHash special algorithm for symmetric decryption. Users have the flexibility to generate new encryption keys or utilize existing ones for decryption purposes, ensuring seamless administration of their encrypted data.
6. **Hash Creator:** ShadowHash integrates a cipher generation utility that accommodates to numerous user requirements. Users can generate message digests using prominent hashing algorithms like MD5, SHA-1, and SHA-256. The platform

also provides a generic hash creator, allowing users to investigate various hashing algorithms for specific use cases.

7. **Hash Comparison Checker:** This feature enables users to validate the integrity of their data. By comparing a calculated hash value with a previously generated hash, ShadowHash can identify any potential data modifications or manipulation attempts, ensuring the authenticity and trustworthiness of the information.
8. **Virus Detector:** Recognizing the criticality of malware protection, ShadowHash integrates the VirusTotal API. Users can upload files for scanning, and ShadowHash leverages the extensive VirusTotal database to detect and identify potential infections or malicious code embedded within the files.

By incorporating these exhaustive features, ShadowHash positions itself as a valuable tool for individuals and organizations seeking a centralized platform to manage their cryptographic requirements. The platform accommodates to a broad range of security requirements, empowering users to safeguard their credentials, encrypt sensitive data, verify file integrity, and defend against potential malware threats.

## Chapter 2

### Terminology

#### Cryptography <sup>[1]</sup>

The study of safeguarding data and communication by mathematical methods is known as cryptography. It includes a range of techniques for converting readable data into an unreadable format so that only those with permission may view it. The three main operations of cryptography are hashing, decryption, and encryption, each of which is essential to the protection of data.

#### 1. Encryption and Decryption <sup>[1]</sup>

Data is transformed from plaintext into ciphertext via the process of encryption, rendering it unintelligible to unauthorized users. The opposite procedure, known as decryption, transforms ciphertext back into plaintext. Both symmetric and asymmetric cryptographic techniques may be used to accomplish encryption and decryption.

##### 1.1.Symmetric Encryption:

The same key is used in symmetric encryption for both encryption and decryption. It is renowned for being quick and effective, which makes it appropriate for encrypting big data.

##### 1.1.1. AES (Advanced Encryption Standard) <sup>[2]</sup>

1.1.1.1. **Key Sizes:** 128, 192, or 256 bits.

1.1.1.2. **Block Size:** 128 bits.

1.1.1.3. **Summary:** The popular encryption technique AES uses numerous transformation rounds and works with a 4x4 column-major order matrix of data. There are 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys,

depending on the size of the key.

- 1.1.1.4. **Safety:** AES is a highly secure protocol that is used worldwide to protect sensitive data and is impervious to all known practical threats.

### 1.1.2. 3DES (Triple Data Encryption Standard) <sup>[3]</sup>

- 1.1.2.1. **Key Sizes:** 168 bits (effectively 112 bits due to meet-in-the-middle attacks).
- 1.1.2.2. **Block Size:** 64 bits.
- 1.1.2.3. **Summary:** Using three 56-bit keys, 3DES runs the DES algorithm three times over each data block. Using the first key to encrypt, using the second to decode, and using the third key to encrypt again are the steps in the procedure.
- 1.1.2.4. **Safety:** Said to be slower and less safe than more recent algorithms like AES, but still more secure than DES.

### 1.1.3. Blowfish <sup>[4]</sup>

- 1.1.3.1. **Key Sizes:** 32-448 bits.
- 1.1.3.2. **Block Size:** 64 bits.
- 1.1.3.3. **Summary:** A symmetric block cipher with a focus on speed and security is called Blowfish. It employs 16 encryption rounds with a changeable key length, where each round consists of a key- and data-dependent replacement as well as a key-dependent permutation.
- 1.1.3.4. **Safety:** Quick and safe, but limited by a 64-bit block size, which makes it less appropriate for certain contemporary applications.

## 1.2. Asymmetric Encryption:

A public key is used for encryption in asymmetric encryption, whereas a private key is used for decryption. When secure communication between parties is required without previous key sharing, this approach is often used to allow secure key exchange.

#### 1.2.1. **RSA (Rivest-Shamir-Adleman)** <sup>[5]</sup>

1.2.1.1. **Key Sizes:** 1024, 2048 or 4096 bits.

1.2.1.2. **Summary:** Based on the mathematical characteristics of big prime numbers, RSA is one of the first public-key cryptosystems. Large integer factoring is a challenging problem, which is the foundation of RSA's security.

1.2.1.3. **Safety:** Larger keys result in poorer performance, while RSA's security rises with key size. In order to maintain security against increases in processing power, current standards advocate for keys of at least 2048 bits.

## 2. **Hashing** <sup>[1]</sup>

Any amount of input data may be transformed via the process of hashing into a fixed-size output called a hash. Passwords are usually securely stored in hashes and used for data integrity verification. Hashing is a one-way procedure that cannot be undone, in contrast to encryption.

### 2.1. **MD5 (Message Digest Algorithm 5)** <sup>[1]</sup>

2.1.1. **Key Sizes:** 1024, 2048 or 4096 bits

2.1.2. **Summary:** From the supplied data, MD5 generates a 128-bit hash value. It is extensively used for data integrity verification and checksums.

2.1.3. **Safety:** Hash collision issues render MD5 cryptographically flawed and unfit for further use.

### 2.2. **Secure Hash Algorithm 1 (SHA-1):** <sup>[6]</sup>

2.2.1. **Key Sizes:** 160 bits

2.2.2. **Summary:** A 160-bit hash value is generated using SHA-1. It was often used to data integrity checks and information security.

2.2.3. **Safety:** Because of its flaws and vulnerability to collision attacks, SHA-1 is no longer regarded as secure.

### 2.3. Secure Hash Algorithm 256 (SHA-256) <sup>[6]</sup>

2.3.1. **Key Sizes:** 256 bits

2.3.2. **Summary:** a member of the SHA-2 family, produces a hash value that is 256 bits long. For cryptographic applications, such as digital signatures and certificate verification, it is extensively used.

2.3.3. **Safety:** SHA-256 is frequently used for a variety of security applications and is thought to be safe.



## Chapter 3

### Project Objective

The main objective of ShadowHash is to develop and launch a platform that is easy to use and integrates necessary cryptographic services. This platform offers a single center for data encryption, data integrity verification, and threat detection in an effort to meet the various security needs of people and organizations. Among the goals are:

1. **Password Security:** Offering resources for creating and verifying secure passwords. In order to improve overall password hygiene, the platform attempts to assist users in creating passwords that adhere to high security requirements, storing them safely, and routinely checking for possible breaches.
2. **Data Encryption:** Providing a variety of encryption techniques to safeguard private data while it is being sent and stored. To keep data private and safe from unwanted access, ShadowHash will support cutting-edge encryption technologies including AES, 3DES, Blowfish, and RSA along with a unique self-made ShadowHash special cryptographic algorithm. <sup>[2] [3] [4] [5]</sup>
3. **Data Integrity:** Making use of hashing algorithms to confirm data accuracy and guard against manipulation. Data will be digitally fingerprinted using hash algorithms like MD5, SHA-1, and SHA-256, allowing users to identify any changes and verify the accuracy of their data. <sup>[1] [6]</sup>
4. **Threat Detection:** By integrating APIs, security risks may be proactively identified and mitigated by checking for compromised files, passwords, and emails. With the usage of tools like VirusTotal API, HaveIBeenPwned API, and Hackcheck Woventams API, ShadowHash will assist customers in quickly identifying and addressing any malware infections and breaches.
5. **User Experience:** Making sure the platform has a smooth, easy-to-navigate interface that makes it simple for users to access all security capabilities. This involves creating a web application that is adaptable to many screen sizes and devices.
6. **Scalability and Reliability:** Constructing a strong backend architecture for the platform to manage growing user demands and guarantee steady performance. With the implementation and scalability of cloud services like as AWS Amplify, ShadowHash

will be able to provide dependable and effective service to its global user base.

7. **Updates and Maintenance:** Upholding a commitment to continual improvement via the constant addition of new features, security updates, and performance boosts to the platform. By doing this, ShadowHash is able to keep ahead of new threats and adapt to its customers' changing demands.

By accomplishing these goals, ShadowHash hopes to provide a complete security solution that makes cryptographic job management easier while improving user data security overall. Through the use of a single, integrated solution, the platform will enable people and businesses to safeguard their confidential data, confirm the accuracy of their data, and be alert for any possible cyber threats.

# Chapter 4

## Software Used

### Languages Used

#### 1.1.Front-end

##### 1.1.1. HyperText Markup Language (HTML)

- 1.1.1.1. **Structure and Content:** A web page's structure and content are specified using HTML. It offers a variety of features, including lists, headers, paragraphs, links, and photos.
- 1.1.1.2. **Semantic Elements:** Logical content organization and enhanced accessibility are facilitated by semantic elements found in HTML, such as `<header>`, `<footer>`, `<article>`, and `<section>`.
- 1.1.1.3. **Forms and Input:** Text fields, radio buttons, checkboxes, and submit buttons are just a few of the form features that HTML forms utilize to enable user input.
- 1.1.1.4. **Hyperlinks:** To enable navigating between separate web pages or parts inside a single page, utilize the HTML `<a>` element to build hyperlinks.

##### 1.1.2. Cascading Style Sheets (CSS)

- 1.1.2.1. **Styling and Layout:** The display of HTML components, such as layout, color, font, and spacing, is managed by CSS. It improves websites' usability and aesthetic appeal.
- 1.1.2.2. **Responsive Design:** CSS makes it possible to create responsive web designs, which employ media queries to adjust to different screen sizes and devices while maintaining a consistent user experience on all platforms.
- 1.1.2.3. **Flexbox and Grid:** Complex, adaptable, and responsive layouts are made easier to create with CSS thanks to its strong layout components, Flexbox and CSS Grid.

- 1.1.2.4. **Animations and Transitions:** CSS makes it possible to apply animations and transitions to elements, which improves user interaction and creates a dynamic environment.
- 1.1.2.5. **Selectors and Specificity:** Specificity rules dictate which styles are used in cases when several rules clash, while CSS selectors allow the styling of certain HTML elements.

## 2. Back-end

### 2.1. JavaScript

#### 2.1.1. JavaScript Vanilla

Ordinary JavaScript

- 2.1.1.1. **Simplicity:** Writing server-side code using vanilla JavaScript is easy and doesn't need any other libraries or frameworks.
- 2.1.1.2. **Control:** gives total control over the code, enabling accurate and efficient implementation of certain features.
- 2.1.1.3. **Adaptability:** Fit for smaller jobs and situations requiring specialized, low-maintenance solutions.

#### 2.1.2. Express

Authentication on the server side

- 2.1.2.1. **Routing:** Express manages routing for distinct web application endpoints, allowing routes to be established for varied functionality.
- 2.1.2.2. **Middleware:** To process requests, manage errors, and carry out operations like authentication and logging, Express includes middleware.
- 2.1.2.3. **Security:** To provide strong user management, Express offers secure authentication methods including OAuth, JWT, and session-based authentication.

#### 2.1.3. Node.js

Management and Creation of Servers

- 2.1.3.1. **Scalability:** Node.js is perfect for high-traffic apps since it can manage several connections at once with ease.
- 2.1.3.2. **Asynchronous I/O:** The event-driven, non-blocking I/O mechanism of Node.js guarantees excellent responsiveness and speed.
- 2.1.3.3. **Dependency Management:** Adding functionality and managing dependencies in Node.js applications is made easier by the extensive library and toolkit that the npm (Node Package Manager) ecosystem offers.

## 2.2. Java

Java was initially used to create the ShadowHash backend because to its reliability, security features, and extensive library support. Apart from guaranteeing superior performance and offering a robust structure for building a safe and expandable program, it further simplified complex cryptographic processes. Due to issues encountered throughout the front-end integration process, JavaScript was ultimately employed for the application's back end.

## 3. Hosting

### 3.1. AWS <sup>[7]</sup>

A comprehensive and extensively used cloud platform, Amazon Web Services (AWS) provides a broad range of services, including networking, storage, and processing power. It is intended to support corporate growth and expansion by offering dependable, scalable, and reasonably priced cloud solutions. Because of its extensive worldwide infrastructure, which guarantees high availability and redundancy, AWS is a top option for managing and deploying applications of all sizes. Simple websites to intricate, large-scale business applications are supported by AWS, which offers services designed for a variety of use cases.

- 3.1.1. **AWS Amplify:** The project is hosted by AWS Amplify via an integration with GitHub. This makes continuous integration/continuous deployment (CI/CD) processes simple to implement and guarantees that project modifications are automatically distributed.

- 3.1.2. **AWS CloudWatch:** AWS CloudWatch is used to track metrics and keep an eye on the performance of the application. It enables proactive management and troubleshooting by giving real-time insights into the system's condition, resource use, and any problems.
- 3.1.3. **AWS SNS (Simple Notification Service):** When CloudWatch detects any problems, AWS SNS is utilized to deliver alerts. By doing this, it is made sure that the development team is notified as soon as any abnormalities or serious problems arise, allowing for quick action to preserve system uptime and integrity.
- 3.1.4. **AWS Shield:** It is an application is shielded against Distributed Denial of Service (DDoS) assaults. By offering thorough threat detection and mitigation, this solution improves the project's security posture and guarantees continuous service availability.

## Libraries used

### 1. JavaScript Libraries

- 1.1.1. **node-fetch**: It is a small module that integrates window.fetch into Node.js, enabling quick and easy network resource fetching. It makes it simple to create HTTP queries and handle API replies in Node.js programs.
- 1.1.2. **Crypto**: Cryptographic functions such as hashing, encrypting, decrypting, and creating digital signatures are provided by the Node.js `crypto` module. Encryption using AES and hashing with MD5, SHA-1, SHA-256, and other methods is supported, which makes it crucial for safe data management in applications.
- 1.1.3. **express-fileupload**: A middleware for Express.js called `express-fileupload` makes file upload management easier. It makes it simple to handle multipart/form-data queries, manage file storage, and process uploaded files, making it easier to integrate file upload features into Node.js applications.
- 1.1.4. **Path**: Node.js's `path` module offers tools for manipulating file and directory paths. It provides ways to extract file extensions, resolve and normalize paths, work with file paths, and more. In Node.js applications, it is essential for guaranteeing platform-independent path processing.

### 2. Java Libraries (Deprecated)

- 1.1.5. **java.security Package**: It is a necessary component for cryptography functions.
  - 1.1.5.1. **KeyPairGenerator**: This tool generates RSA key pairs.
  - 1.1.5.2. **MessageDigest**: computes the input data's SHA-256 hash.
- 1.1.6. **javax.crypto Package**: It offers decryption and encryption features.
  - 1.1.6.1. **Cipher**: Blowfish, AES, 3DES, and RSA algorithms supported.

- 1.1.6.2. **SecretKeySpec**: Generates secret key constructs.
- 1.1.6.3. **SecretKeyFactory**: Manages the creation of DESede keys.
- 1.1.7. **java.util Package**: Provides utility classes, such as random number generation and Base64 encoding.
  - 1.1.7.1. Data is encoded and decoded using Base64.
  - 1.1.7.2. Random: Produces arbitrary integers.
  - 1.1.7.3. Scanner: Reads data from conventional input as well as files.
  - 1.1.7.4. ArrayList: Handles lists that are dynamic.
- 1.1.8. **The java.io Package**: Enables file management and input/output operations.
  - 1.1.8.1. Writes strings to files using FileWriter.
- 1.1.9. **okhttp3 Package**: Robust Java HTTP client.
  - 1.1.9.1. OkHttpClient: Forwards requests for HTTP.
  - 1.1.9.2. Response: Controls the processing of HTTP responses.
  - 1.1.9.3. Request: Creates requests via HTTP.
  - 1.1.9.4. MultipartBody: Generates requests for files in multiple parts.
- 1.1.10. **java.nio.file Package**: Facilitates sophisticated file input/output operations.
  - 1.1.10.1. Files: Reads bytes from files and verifies their existence.
  - 1.1.10.2. Paths: Generates `Path` objects from strings.
- 1.1.11. **java.lang Package**: Core classes and interfaces are provided by the same.
- 1.1.12. **Thread**: Controls how a thread is executed.

## APIs used

The integration of numerous APIs in Java enhances the functionality of the ShadowHash platform. These APIs allow for the substantiation of email and password vulnerabilities and



the detection of infections in files, ensuring comprehensive security for all the users.

### **1. Hackcheck Woventeams API**

- a. **URL:** <https://hackcheck.woventeams.com/api/v4/breachedaccount/>
- b. **Purpose:** Checks if an email address has been implicated in a data breach.

### **2. HaveIBeenPwned API**

- a. **URL:** <https://api.pwnedpasswords.com/range/>
- b. **Purpose:** In order to determine if a password has been hacked, the first five characters of its SHA-1 hash are checked.

### **3. VirusTotal API**

- a. **URL:** <https://www.virustotal.com/vtapi/v2/file/scan>
- b. **Purpose:** Scans files for infections, leveraging VirusTotal's comprehensive database for identifying malicious software.

## **Tools Used**

### **1. VS Code**

Front-end code is created and edited using Visual Studio Code (VS Code). The creation of HTML, CSS, and JavaScript is supported, which makes it easier to integrate front-end and back-end systems.

### **2. IntelliJ**

For Java programming, IntelliJ is a feature-rich integrated development environment (IDE). It makes Java code development and administration efficient by offering sophisticated coding aid, debugging, and integration tools.

### **3. Figma**

Wireframing and UI/UX design are done collaboratively using Figma. It facilitates the development and prototyping of user interfaces, guaranteeing an efficient design process and clear idea communication.

### **4. GitHub**

Version control and repository management are supported by GitHub. It is used to host the codebase, promote teamwork, and ultimately carry out project deployment.

### **5. AWS (Amazon Web Services)**

AWS is utilized for hosting the application through AWS Amplify, which integrates seamlessly with the GitHub repository for continuous deployment. AWS Simple Notification Service (SNS) and CloudWatch are set up for constant monitoring and alerting. Additionally, AWS Shield is configured to provide enhanced security, including protection against Distributed Denial of Service (DDoS) attacks, ensuring high availability and resilience of the application.

## **Chapter 5**

### **Methodology**

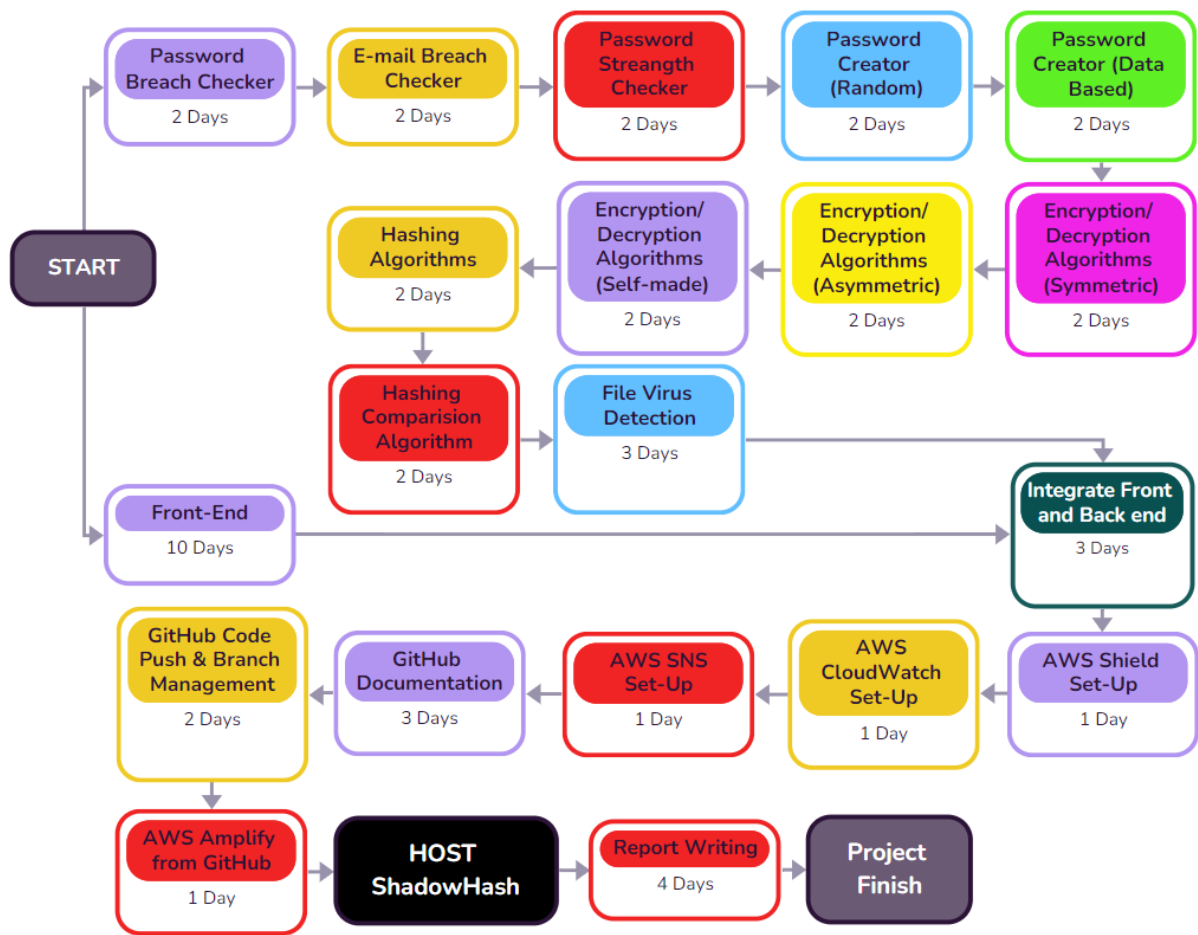
The development of ShadowHash involves a structured approach incorporating various cryptographic and security functionalities to guarantee comprehensive user protection. The methodology can be divided into several phases:

#### **1. Requirements Analysis:**

- Determine which major features are required for the platform, such as virus detection, hash generation, encryption, decryption, email breach detection, password creation, and hash checking.
- Gather requirements for the user interface and user experience to assure simplicity of use and accessibility.

#### **2. Design:**

- Create the system architecture, keeping the front-end user interface and the back-end services apart.
- Plan the integration of APIs such as Hackcheck Woventams API for email breach checking, HaveIBeenPwned API for password breach checking, and VirusTotal API for file virus infection detection.
- Define data flow and interaction between various components of the system.



**Image 1**  
**(Pert Chart)**

### 3. Development:

#### ➤ Backend Development:

- Assure safe HTTP requests and replies while implementing email breach checking using the Hackcheck Woventeams API.
- Develop password breach verification by creating a SHA-1 hash of the user's password and querying the HaveIBeenPwned API with only the first 5 characters of the hash with a strict no log policy.
- Create a password generator that produces robust credentials adhering to best practices.
- Utilizing symmetric (AES, 3DES, Blowfish) and asymmetric (RSA) methods in conjunction with a unique ShadowHash special algorithm, provide encryption and decryption functions.

- Develop hash construction and verifying functionalities using MD5, SHA-1, SHA-256, and a generic hash creator.
- Integrate the VirusTotal API to examine files for viruses.

➤ **Frontend Development:**

- Design a user-friendly interface using HTML and CSS, allowing users to interact with the platform seamlessly.
- Ensure responsive design for compatibility across various devices.

#### **4. Testing:**

- Perform unit testing and integration testing to ensure each component functions accurately and integrates seamlessly.
- Conduct security testing to verify the efficacy of cryptographic implementations and API integrations.
- Perform user acceptance testing (UAT) to ensure the platform meets user requirements and expectations.

#### **5. Deployment:**

- Host the interface on GitHub Pages for public access.
- Scalability and dependability should be ensured by deploying the backend services on AWS Amplify. By directly connecting with the GitHub repository, AWS Amplify enables continuous deployment by guaranteeing that any modifications made to the codebase are immediately pushed. Real-time insights and alerts on the performance and status of the application are provided by AWS Simple Notification Service (SNS) and CloudWatch, which are configured for continuous monitoring and alerting. AWS Shield is also used to improve the security and resilience of the application by offering additional defense against Distributed Denial of Service (DDoS) assaults.

## **6. Maintenance and Updates:**

- Continuously monitor the platform for any security vulnerabilities or performance issues.
- Provide regular updates to enhance features and resolve any identified flaws or security concerns.

This exhaustive methodology guarantees the successful development and deployment of ShadowHash, providing users with a secure and reliable cryptographic solution.

## Chapter 6

### ShadowHash Model

**ShadowHash – All in One Crypto Solution** is the platform which is easy to use and integrates necessary cryptographic services. This platform offers a single center for data encryption, data integrity verification, and threat detection in an effort to meet the various security needs of people and organizations in a pool of website (27 webpages divided into 11 sections). These sections include are:

#### Home Page:

The “home page” of ShadowHash includes four sub-parts (linked to one another via header).


1. **Landing Page:** This section includes a header connecting to all parts of the home page along with the landing page. It also includes links to our “About” and “Terms of Use” webpage.



Image 2

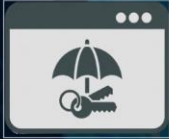
2. **Benefits:** This section highlights the benefits of using our ShadowHash Model briefly.

# Benefits



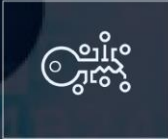
**Breach Monitoring**

Stay ahead of security threats with our Email Breach Checker and Password Breach Checker. We'll notify you if your email or password has been compromised in a data leak, allowing you to take immediate action and change your credentials.



**Unbreakable Passwords**


Generate strong, unique passwords for all your accounts using our Password Creator. Choose between random passwords or customize them with acronyms, ensuring they meet the highest security standards with a mix of uppercase, lowercase, numbers, and symbols, all within a secure length of 8-20 characters.



**Ironclad Encryption**


Protect your sensitive data with our robust Encryption & Decryption Algorithms. Choose from industry-standard Asymmetric (RSA) encryption with user-generated or imported keys, or leverage Symmetric encryption with AES, 3DES, Blowfish, or our very own, state-of-the-art ShadowHash Algorithm for an extra layer of security.

Image 3




**Verification Made Easy**

Our Hash Creator allows you to generate various hashes, like SHA-1 and MD5, for data integrity checks. The Hash Checker functionality lets you verify the authenticity of your data by comparing it against a known hash value.



**Organize Securely**

Simplify your online life with our Password Manager. Store your passwords securely and access them conveniently whenever needed.



**Virus Protection**

Scan files for potential threats using our Virus Detector powered by VirusTotal API. Ensure your device and data stay safe from malware and other malicious software.

Image 4

**3. Services:** This section includes links to all the services provided by ShadowHash.



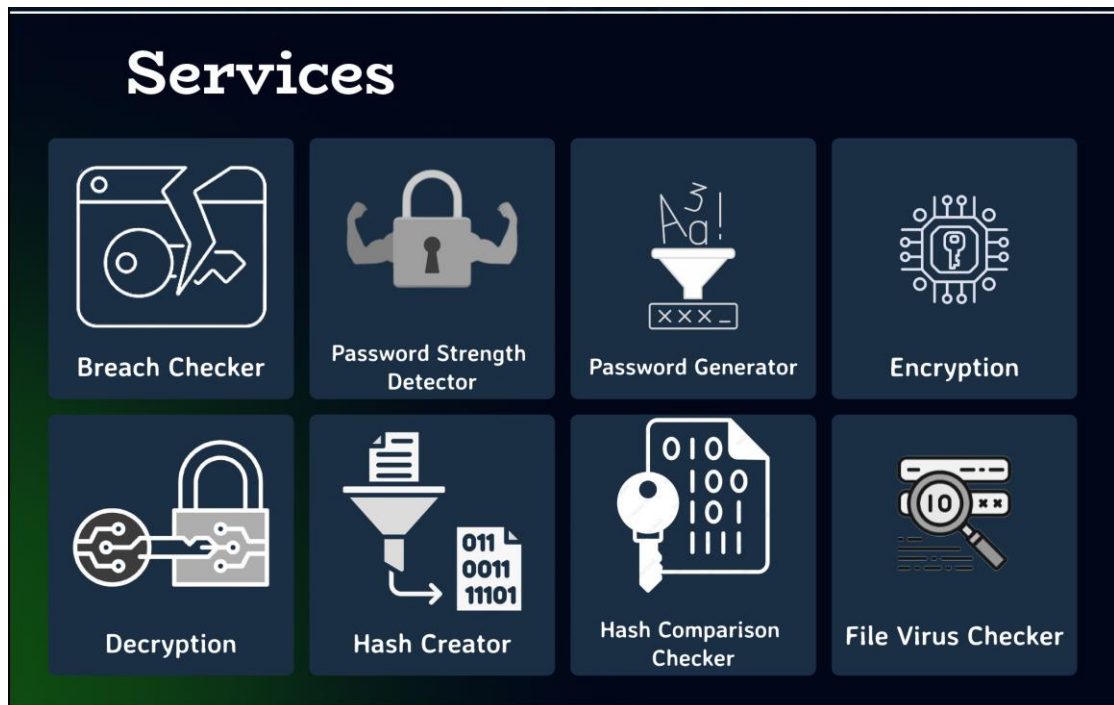


Image 5

4. **Resources:** This section specifically designed for ardent readers includes links to cryptography and cyber-security related blogs and articles which are constantly being updated along with a footer connecting to the landing page along with address and mailto format mail id for contact purposes.

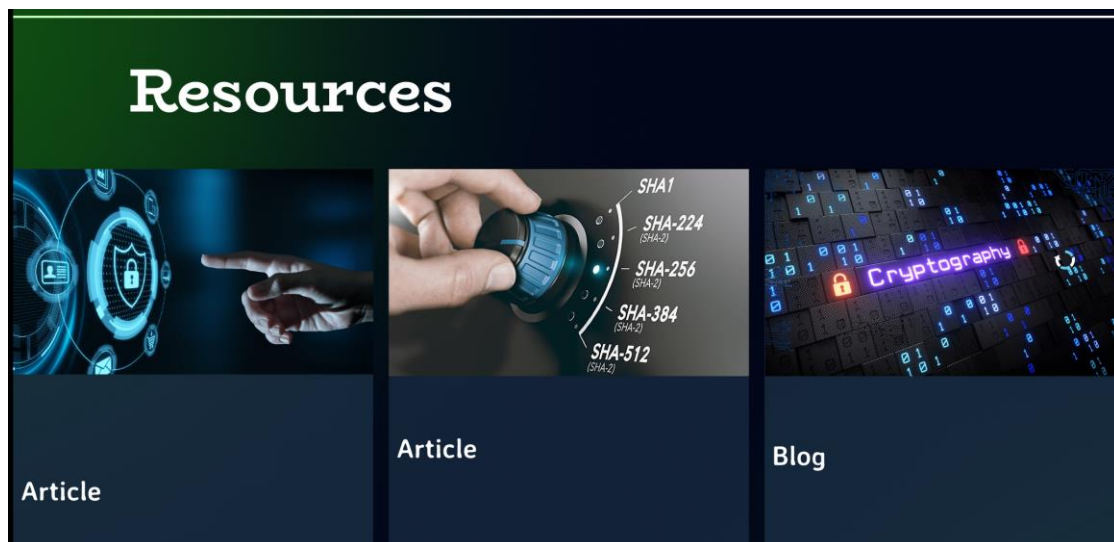


Image 6

<p><b>Block Cipher Techniques as per recommendations by NIST Frameworks</b></p> <p>NIST SP 800-131A recommends AES (FIPS 197) for new applications, while Triple DES (SP 800-67 Rev 2) is still allowed for legacy use. DES and Skipjack are no longer approved.</p>	<p><b>Hash Functions Techniques as per recommendations by NIST Frameworks</b></p> <p>NIST SP 800-131A lists approved hash functions for data integrity.  SHA-1: Deprecated due to security weaknesses, avoid using.  SHA-2 family (SHA-224, 256, 384, 512): Currently recommended for most applications.  SHA-3 family (SHA3-224, 256, 384, 512): Newer alternative to SHA-2 family.</p>	<p><b>Cryptography: Features, Types and Applications</b></p> <p>The document explores cryptography, a cornerstone of computer security, and its various methods for securing information during communication and storage. It leverages algorithms and mathematical principles to transform data into an unreadable format, ensuring confidentiality, integrity, and authenticity.</p>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Image 7



Image 8

## About:

The “About” page of ShadowHash describes about our vision, mission along with details about our team. Alongside the header connecting to the landing page as well as footer with same functionalities.

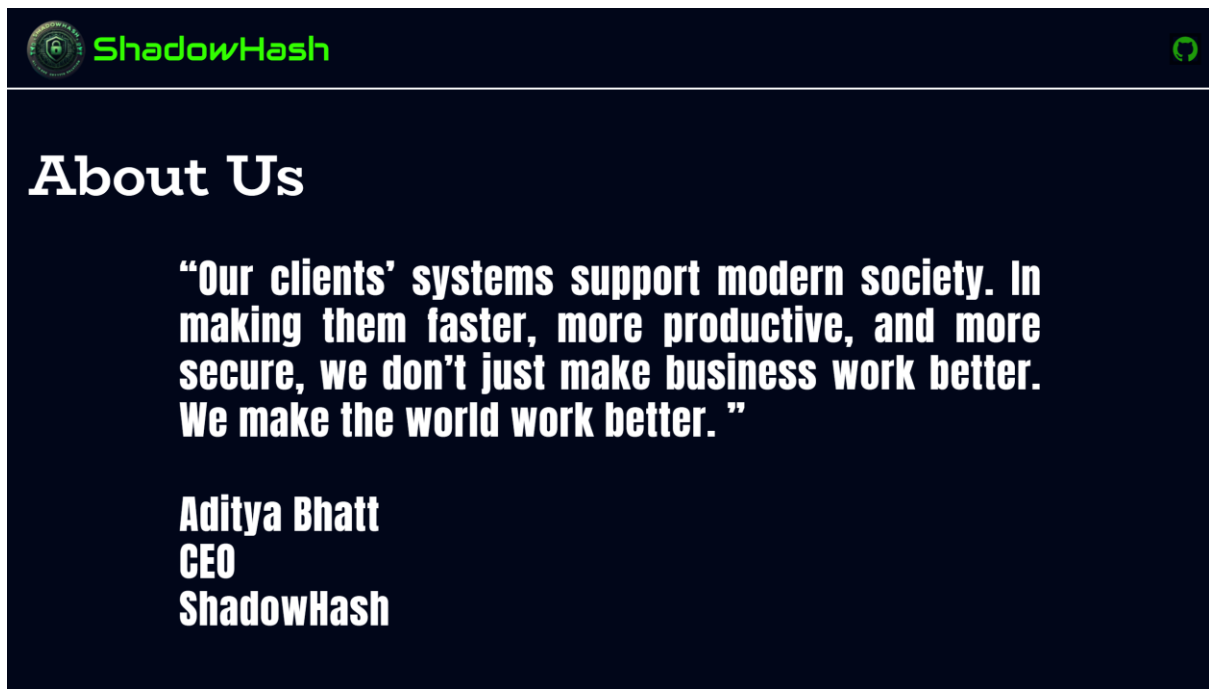


Image 9



Image 10

## Our Mission

Our mission is to make best-in-class security tools accessible to everyone. We strive to demystify cryptography and provide you with the resources you need to protect your data, passwords, and online identity.

## The ShadowHash Team

We are a team of security enthusiasts and software developers dedicated to building innovative solutions. We combine our expertise in cryptography, data protection, and user experience to bring you ShadowHash.

Image 11

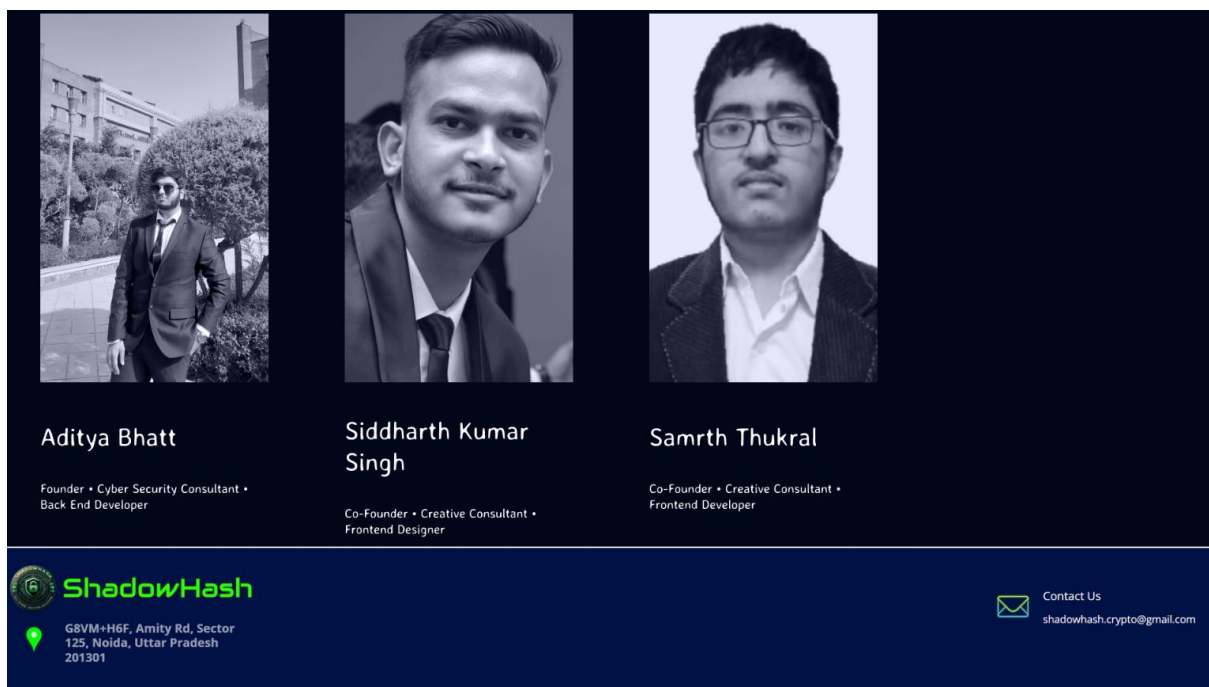


Image 12

## Terms of Use:

The “Terms of Use” page of ShadowHash describes about the detailed “Terms and Conditions” of using the webpage including the disclaimer, service usage, user content, third-party APIs governing as well of penetration testing rights. Alongside the header connecting to the landing page and GitHub repository as well as footer with same functionalities.

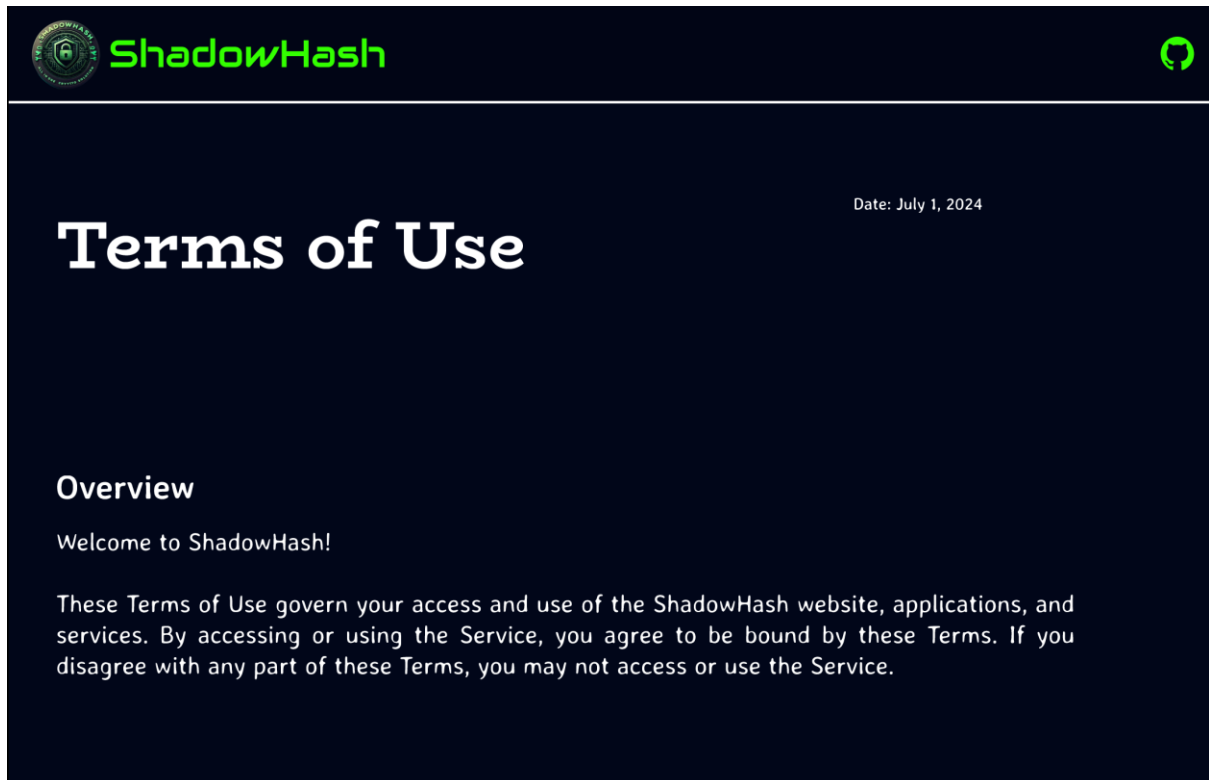


Image 13

## Use of the Service

- You must be at least 13 years old to use the Service.
- You are responsible for protecting your account information.
- You agree to use the Service for lawful purposes only.
- You agree not to disrupt or harm the Service or interfere with others' use.
- You agree to comply with all applicable laws and regulations.

## User Content

- You are responsible for any content you share through the Service.
- You warrant the right to share such content.
- ShadowHash does not claim ownership of User Content, but you grant a non-exclusive, worldwide license for its use in connection with the Service.

## Third-Party APIs

- The Service may use third-party APIs.
- Your use of the Service is also subject to the terms of such APIs.
- ShadowHash is not responsible for their availability or functionality.

Image 14

## Disclaimer

FROM TIME TO TIME, THIS WEB SITE MAY CONTAIN TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS, AND WE DO NOT WARRANT THE ACCURACY OF ANY POSTED INFORMATION. PLEASE CONFIRM YOU ARE USING THE MOST UP-TO-DATE PAGES ON THIS WEB SITE, AND CONFIRM THE ACCURACY AND COMPLETENESS OF INFORMATION BEFORE USING IT TO MAKE DECISIONS RELATING TO SERVICES, PRODUCTS, OR OTHER MATTERS DESCRIBED IN THIS WEB SITE.

IF ANY TERM IN THIS TERMS OF USE IS FOUND BY COMPETENT JUDICIAL AUTHORITY TO BE UNENFORCEABLE IN ANY RESPECT, THE VALIDITY OF THE REMAINDER OF THIS TERMS OF USE WILL BE UNAFFECTED, PROVIDED THAT SUCH UNENFORCEABILITY DOES NOT MATERIALLY AFFECT THE PARTIES' RIGHTS UNDER THIS TERMS OF USE.

## Governing Law

These Terms are governed by the laws of India.

## Changes to the Terms

ShadowHash reserves the right to change these Terms at any time. We will notify you by posting the revised Terms on the Service. Your continued use constitutes your acceptance of the changes.

Image 15

## Penetration Testing Rights

Penetration testing of the Service is prohibited without prior written authorization from ShadowHash.

## Contact Us

If you have any questions, please contact us at [shadowhash.crypto@gmail.com](mailto:shadowhash.crypto@gmail.com)



**ShadowHash**



G8VM+H6F, Amity Rd, Sector 125,  
Noida, Uttar Pradesh 201301



Contact Us  
[shadowhash.crypto@gmail.com](mailto:shadowhash.crypto@gmail.com)

Image 16



## Breach Checker:

The “Breach Checker” of ShadowHash includes two sub-parts for email and password breach checking along with a header linking the home page and GitHub repository of the same.

- 1. Email Breach Checker:** This section includes an email breach detector which takes user email address as input and detects if it was part of any recent data breach. “Hackcheck Woventeams API” is used for the same which returns status code as 200 if the email is found in a breach while status code as 404 in the other case.

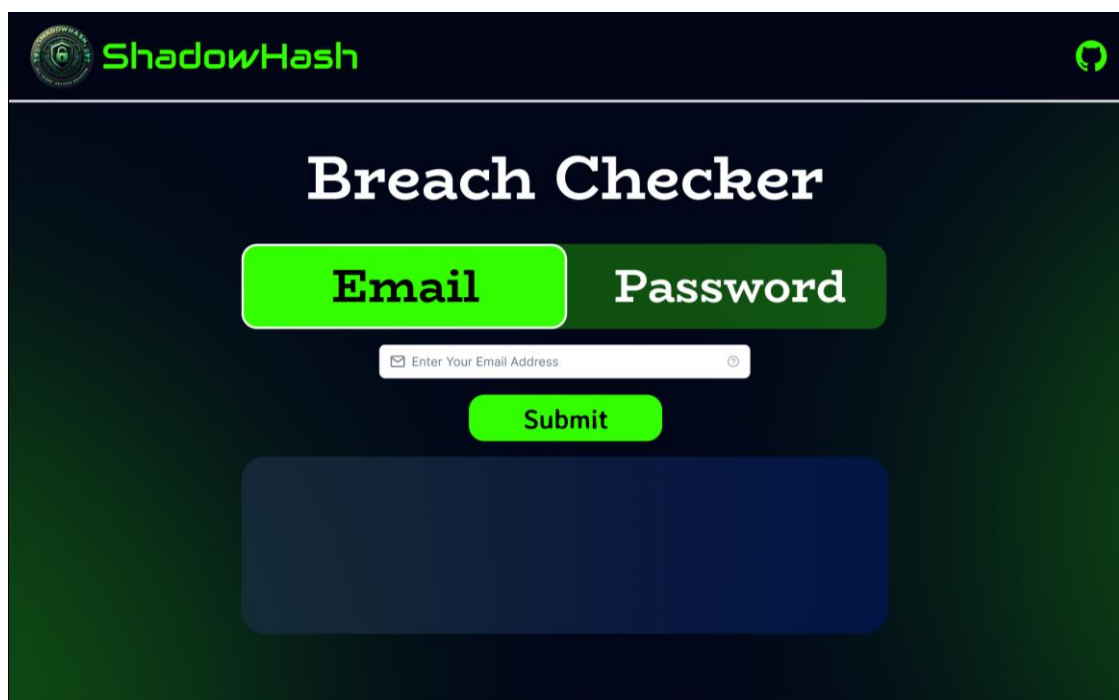


Image 17

### Getting all breaches for an account

The most common use of the API is to return a list of all breaches a particular account has been involved in. The API takes a single parameter which is the account to be searched for. Account should be a valid email address or alphanumeric username. The account is not case sensitive. This is an unauthenticated API.

`GET https://hackcheck.woventeams.com/api/v4/breachedaccount/{account}`

### Response codes

Semantic HTTP response codes are used to indicate the result of the search:

Code	Description
200	OK — everything worked and here's a JSON array of hacked sites for the account
400	Bad request — the account does not comply with an acceptable format (i.e. it's an empty string)
404	Not found — the account could not be found and has therefore not been hacked to our knowledge
500	Server error

Image 18



2. **Password Breach Checker:** This section includes a password breach detector which takes user password as input, converts it into SHA-1 hash, sends only first five characters of the hash for checking via the “HaveIBeenPwned API” and checks for remaining part of the hash if the password exists in a data breach with a strict zero log policy, thus making it completely safe.

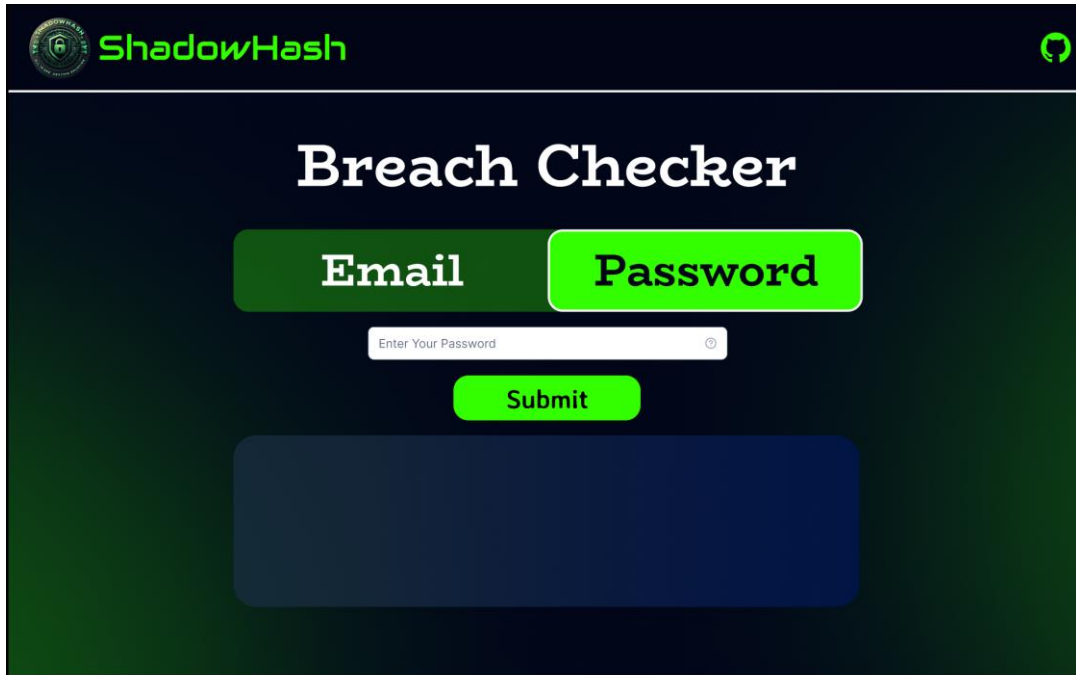
The image shows a web application interface for "ShadowHash". At the top, there is a logo on the left and the text "ShadowHash" in a green font. On the right, there is a green circular icon. The main heading is "Breach Checker" in a large, white, serif font. Below this, there are two buttons: "Email" in a dark green box and "Password" in a bright green box. The "Password" button is selected. Below these buttons is a white input field with the placeholder text "Enter Your Password" and a small eye icon on the right. Below the input field is a bright green "Submit" button. At the bottom, there is a large, empty, dark blue rectangular box.

Image 19

## Searching by range

In order to protect the value of the source password being searched for, Pwned Passwords also implements a [k-Anonymity model](#) that allows a password to be searched for by partial hash. This allows the first 5 characters of either a SHA-1 or an NTLM hash (not case-sensitive) to be passed to the API:

[click here to test](#)

```
GET https://api.pwnedpasswords.com/range/{first 5 hash chars}
```

When a password hash with the same first 5 characters is found in the Pwned Passwords repository, the API will respond with an HTTP 200 and include the *suffix* of every hash beginning with the specified prefix, followed by a count of how many times it appears in the data set. The API consumer can then search the results of the response for the presence of their source hash and if not found, the password does not exist in the data set. A sample SHA-1 response for the hash prefix "21BD1" would be as follows:

```
0018A45C4D1DEF81644B54AB7F969B88D65:1
00D4F6E8FA6EECAD2A3AA415EEC418D38EC:2
011053FD0102E94D6AE2F8B83D76FAF94F6:1
012A7CA357541F0AC487871FEEC1891C49C:2
0136E006E24E7D152139815FB0FC6A50B15:2
...
```

A range search typically returns approximately 800 hash suffixes, although this number will differ depending on the hash prefix being searched for and will increase as more passwords are added. There are 1,048,576 different hash prefixes between 00000 and FFFFF (16^5) and every single one will return HTTP 200; there is no circumstance in which the API should return HTTP 404.

Code	Body	Description
200	Hash suffixes counts	Ok — all password hashes beginning with the searched prefix are returned alongside prevalence counts

[Read more about how k-Anonymity and the Pwned Passwords range search protects searched passwords.](#)

Image 20

## Password Strength Detector:

The “Password Strength Detector” of ShadowHash detects the strength of password via rating them on a plethora of factors like:

1. Password length
2. Use of uppercase characters
3. Use of lowercase characters
4. Use of digits
5. Use of special characters
6. Use of consecutive characters
7. Use of passwords found in breaches (via of the famous rockyou.txt which contains 14,341,564 unique passwords, used in 32,603,388 accounts. <https://www.kaggle.com/datasets/wjburns/common-password-list-rockyoutxt>)
8. Use of sequential characters

Also a header linking the home page and GitHub repository of the same is present here as well.

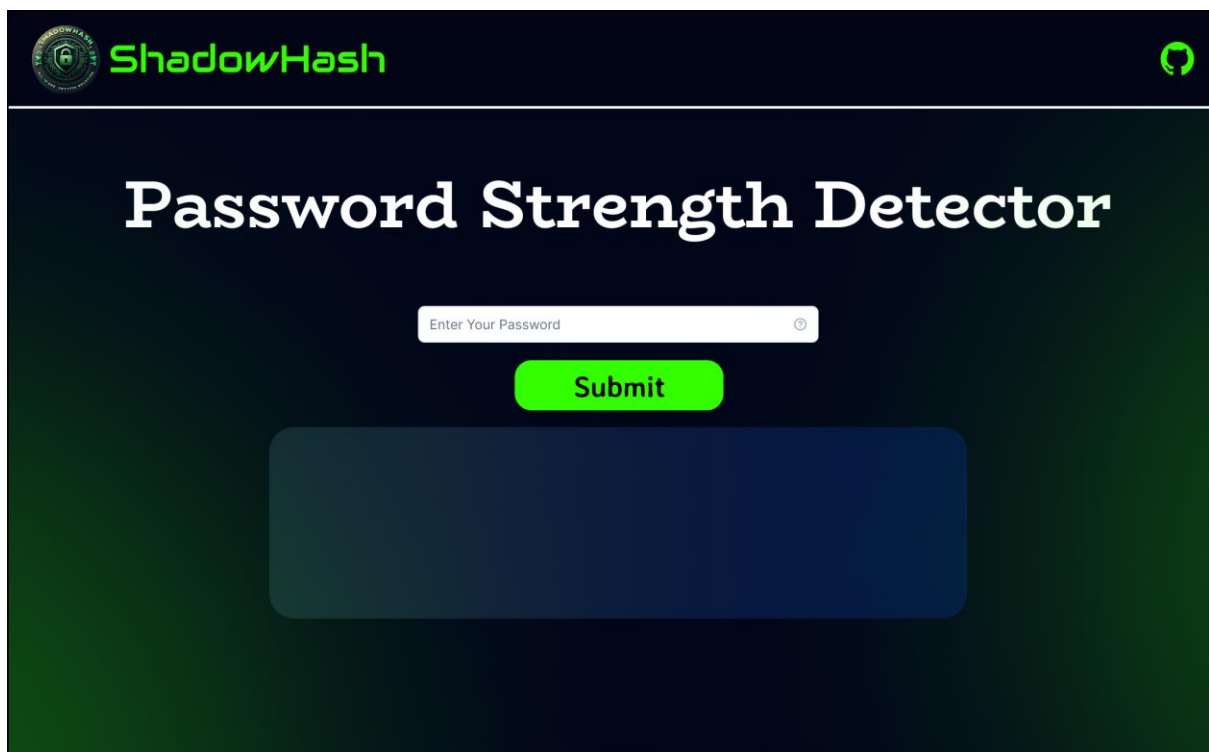


Image 21

## Password Generator:

The “Password Generator” of ShadowHash includes two sub-parts along with a header linking the home page and GitHub repository of the same.

1. **Random:** Generate a random secure password to be used while keeping the following considerations in mind:
  - Password length (random between 8-15)
  - Use of uppercase characters
  - Use of lowercase characters
  - Use of digits
  - Use of special characters
  - No use of consecutive characters

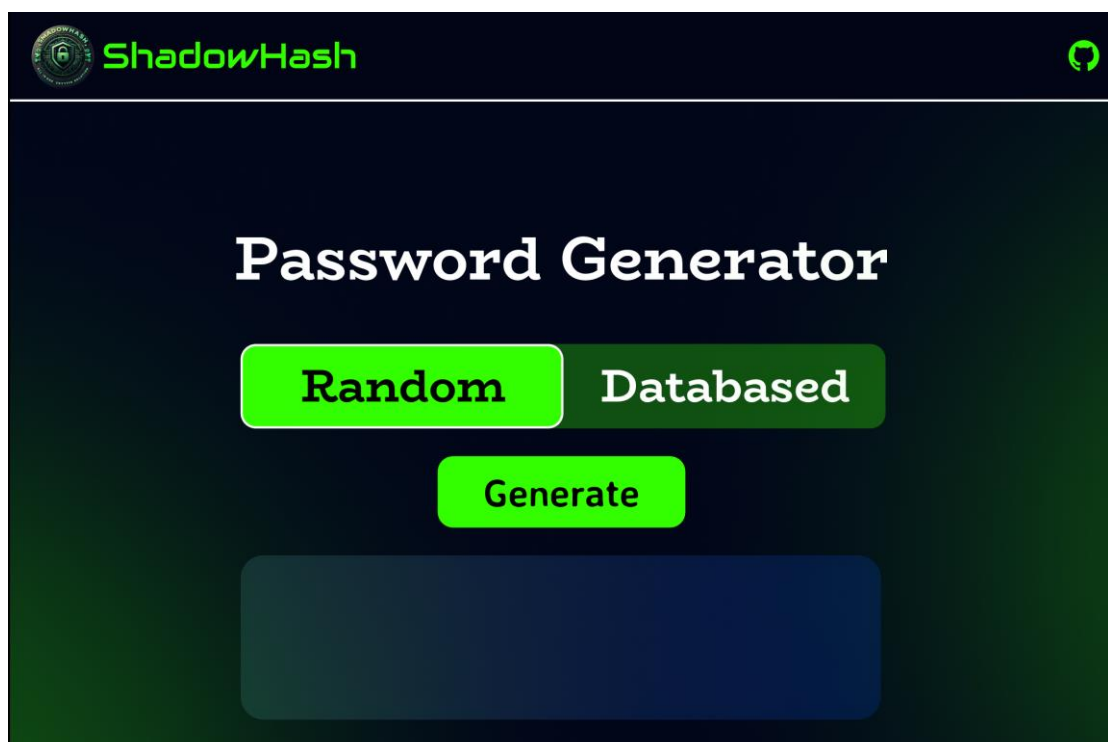


Image 22

2. **Data-based:** Generate a secure password based on user information while ensuring randomness at the same time so that the password is easy to remember for the user but more importantly, cannot be cracked by any person-specific dictionary attack.

ShadowHash

# Password Generator

Random

Databased

<small>Your Name</small> <input type="text" value="Enter Your Name"/>	<small>Pet's Name</small> <input type="text" value="Enter Your Pet's Name"/>
<small>Mother's Name</small> <input type="text" value="Enter Your Mother's Name"/>	<small>Address</small> <input type="text" value="Enter Your Address"/>
<small>Father's Name</small> <input type="text" value="Enter Your Father's Name"/>	<small>Date of Birth</small> <input type="text" value="Enter Your Date of Birth (DD/MM/YYYY)"/>
<small>Spouse's Name</small> <input type="text" value="Enter Your Spouse's Name"/>	<small>Keywords</small> <input type="text" value="Enter Keywords (Comma Separated)"/>

Image 23

## Encryption Tool:

The “Encryption Tool” of ShadowHash includes five sub-parts having different encryption algorithms along with a header linking the home page and GitHub repository of the same.

1. **AES:** Advanced Encryption Standard (symmetric encryption algorithm) converts the user given string message via user given string key. The key is first digested using the SHA-256 hashing algorithm (one-way function) which is later used for encrypting the data.



Image 24

2. **3DES:** Triple Data Encryption Standard (symmetric encryption algorithm) converts the user given message using the key which is first digested using the SHA-256 hashing algorithm (one-way function) which is later used for encrypting the data.



Image 25

3. **Blowfish:** This (symmetric encryption algorithm) converts the user given string message via user given string key. The key is first digested using the SHA-256 hashing algorithm (one-way function) which is later used for encrypting the data.



Image 26

4. **SH Special:** ShadowHash Special is a symmetric encryption algorithm designed by us

which takes user input of message and key. Firstly, five new keys are generated with random integer values between (3-9), (0-3), (0-3), (0-2147483647) and (0-2147483647) respectively. The message is treated character by character, converted from ASCII to decimal to binary. Out of 8 bits, K1 bits are left untouched while the rest are swapped. K1 is then left rotated by K2. K2 is then right rotated by K3. K3 is then encrypted via Triple DES algorithm with K4 key. K4 being encrypted via AES algorithm with K5 key. Lastly, K5 is encrypted by user given key via Blowfish algorithm. This compilation of keys is also given to the user as a second key necessary for decryption.



Image 27

5. **RSA:** Rivest, Shamir, Adleman (asymmetric algorithm) is used to encrypt user data by using public key of the receiver. There's also a "Generate Keys" feature for a new user.



**ShadowHash**

# Encryption Tool

AES3DESBlowfishSH SpecialRSA

Enter your message...

Enter Public Key

Encrypt

Generate Keys

Click on the button above to generate key pair (public and private).....

Image 28

## Decryption Tool:

The “Decryption Tool” of ShadowHash includes five sub-parts having different encryption algorithms along with a header linking the home page and GitHub repository of the same.

1. **AES:** Advanced Encryption Standard (symmetric encryption algorithm) converts the user given string message via user given string key. The key is first digested using the SHA-256 hashing algorithm (one-way function) which is later used for decrypting the data.



Image 29

2. **3DES:** Triple Data Encryption Standard (symmetric encryption algorithm) converts the user given message using the key which is first digested using the SHA-256 hashing algorithm (one-way function) which is later used for decrypting the data.

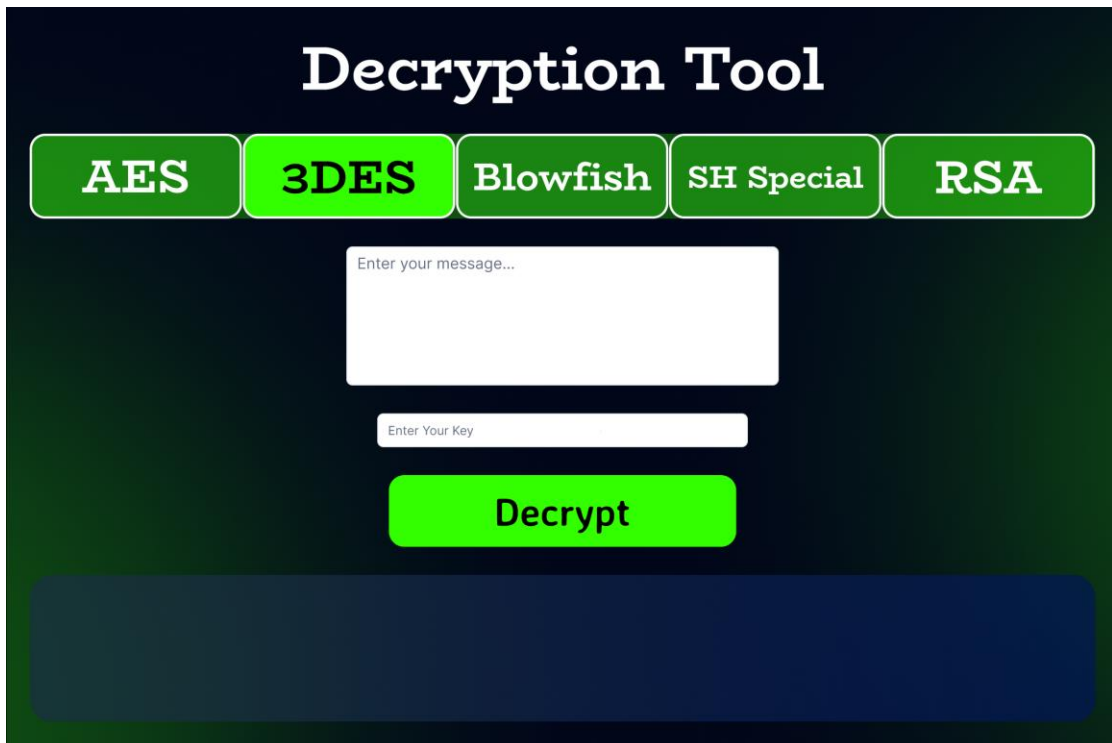


Image 30

3. **Blowfish:** This (symmetric encryption algorithm) converts the user given string message via user given string key. The key is first digested using the SHA-256 hashing algorithm (one-way function) which is later used for decrypting the data.

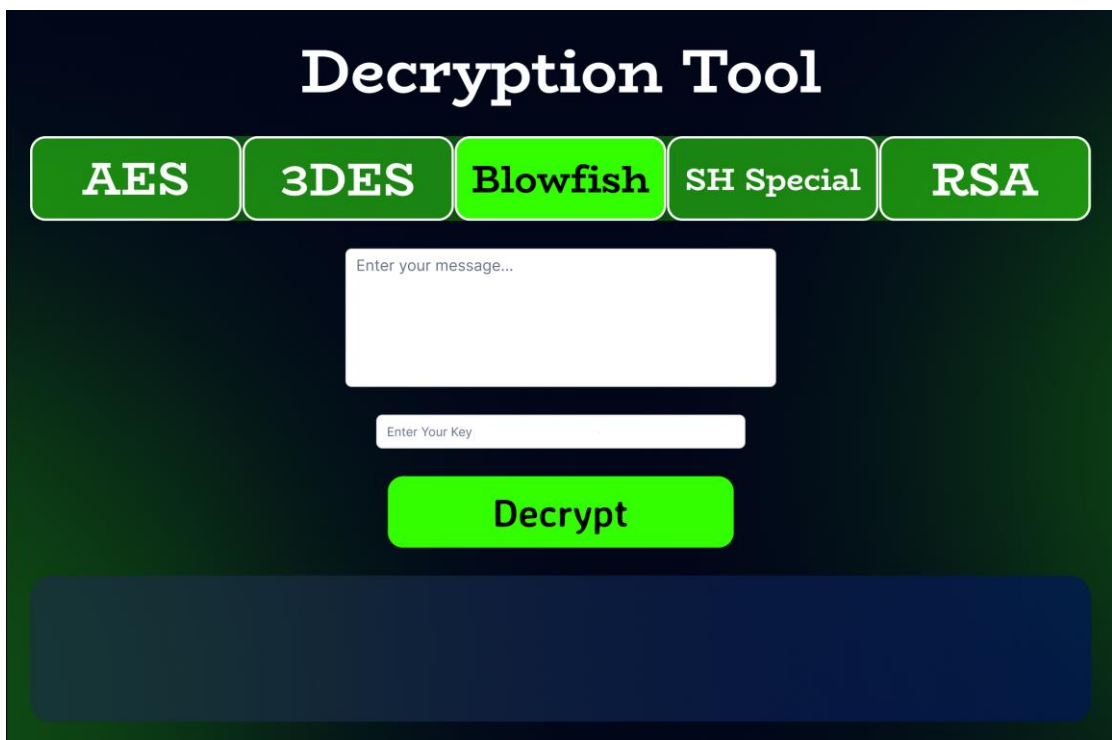


Image 31

4. **SH Special:** ShadowHash Special is a symmetric encryption algorithm designed by us which takes user input of message and key. Firstly, the encryption process involves five new keys generated with random integer values between (3-9), (0-3), (0-3), (0-2147483647) and (0-2147483647) respectively. The message is treated character by character, converted from ASCII to decimal to binary. Out of 8 bits, K1 bits are left untouched while the rest are swapped. K1 is then left rotated by K2. K2 is then right rotated by K3. K3 is then encrypted via Triple DES algorithm with K4 key. K4 being encrypted via AES algorithm with K5 key. Lastly, K5 is encrypted by user given key via Blowfish algorithm.

This compilation of keys is also given to the user as a second key necessary for decryption as this process is simply reversed for decryption algorithm.



Image 32

5. **RSA:** Rivest, Shamir, Adleman (asymmetric algorithm) is used to decrypt user data by using public key of the receiver.

# Decryption Tool

AES

3DES

Blowfish

SH Special

RSA

Enter your message...

Enter Private Key

Decrypt

Image 33

## Hash Creator:

The “Hash Creator” of ShadowHash includes four sub-parts having different hashing algorithms (MD5, SHA1 and SHA256 which are the most commonly used along with a generic one for any other algorithm) along with a header linking the home page and GitHub repository of the same.

1. **MD5:** Message Digest 5 hashes (one-way function) the user given string message and displays the digest for further usage. MD5 is a popularly known hashing algorithm which still used quite popularly yet is vulnerable to many attack vectors.

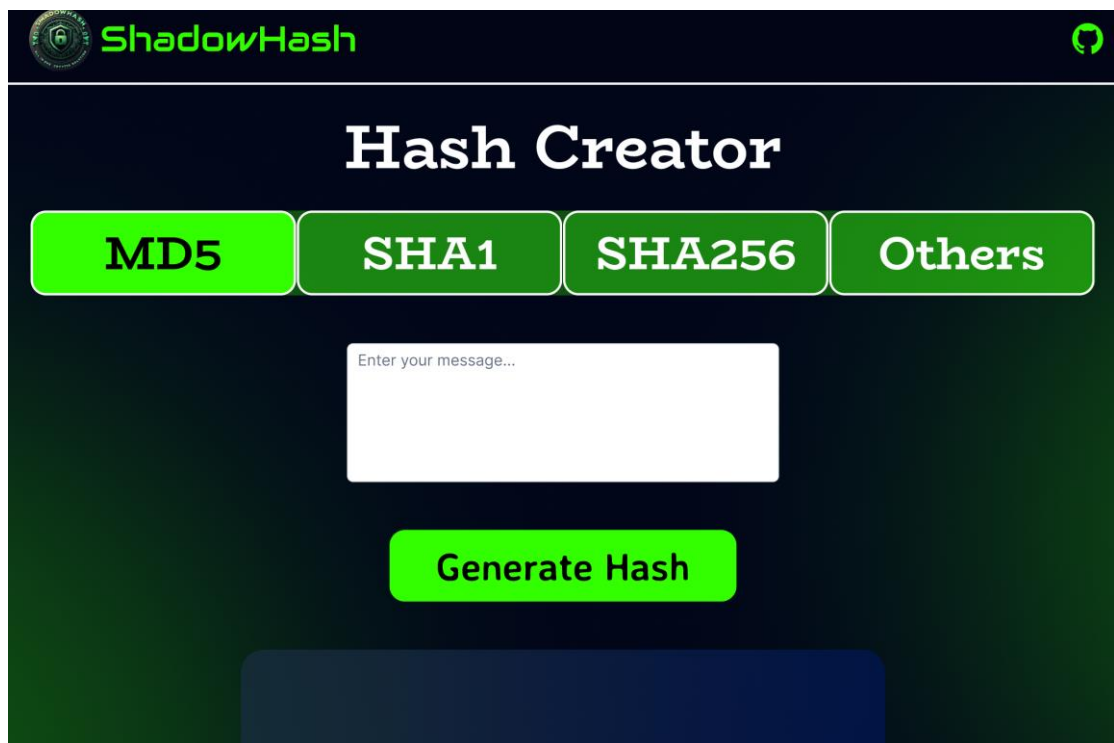


Image 34

2. **SHA1:** Secure Hash Algorithm 1 hashes (one-way function) the user given string message and displays the hash value for further usage. SHA1 is a popularly known hashing algorithm which used popular and more secure as compared to MD5.

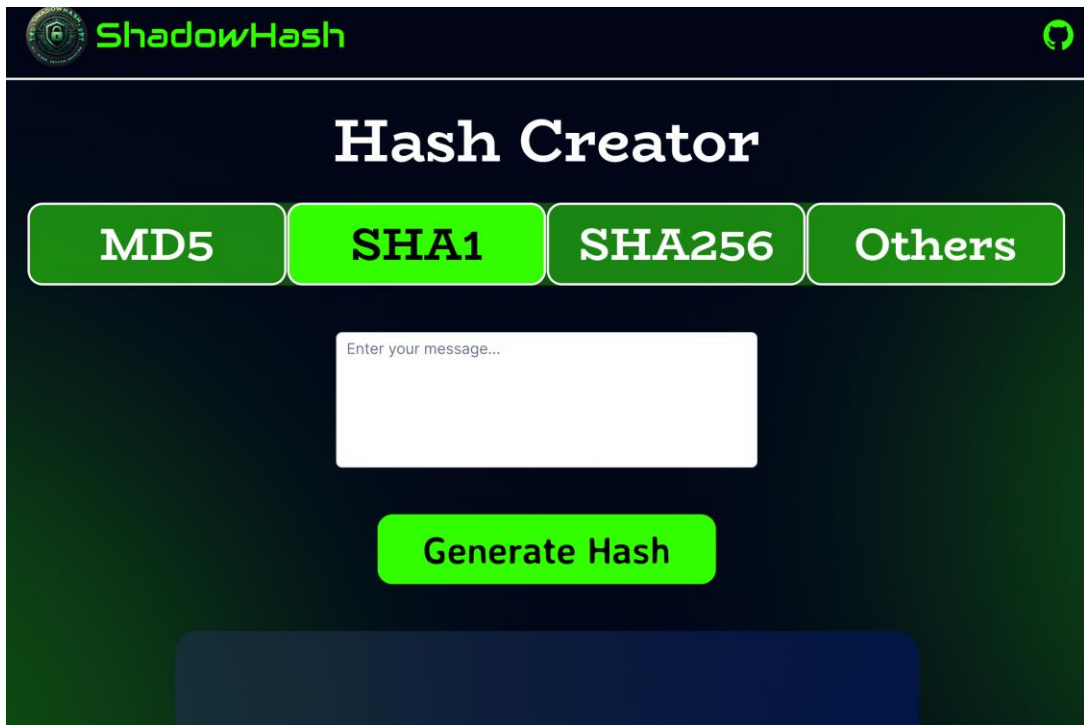


Image 35

3. **SHA256:** Secure Hash Algorithm 256 hashes (one-way function) the user given string message and displays the hash value for further usage. SHA256 yet an algorithm stronger than both the above mentioned ones is also a widely used.

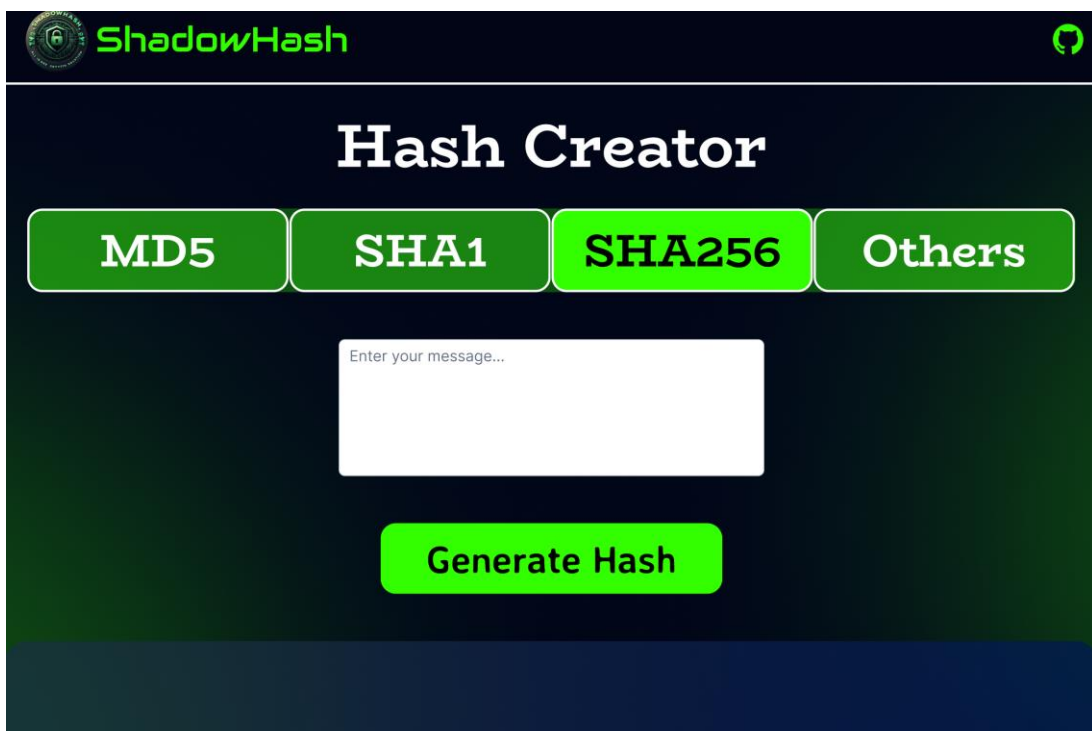
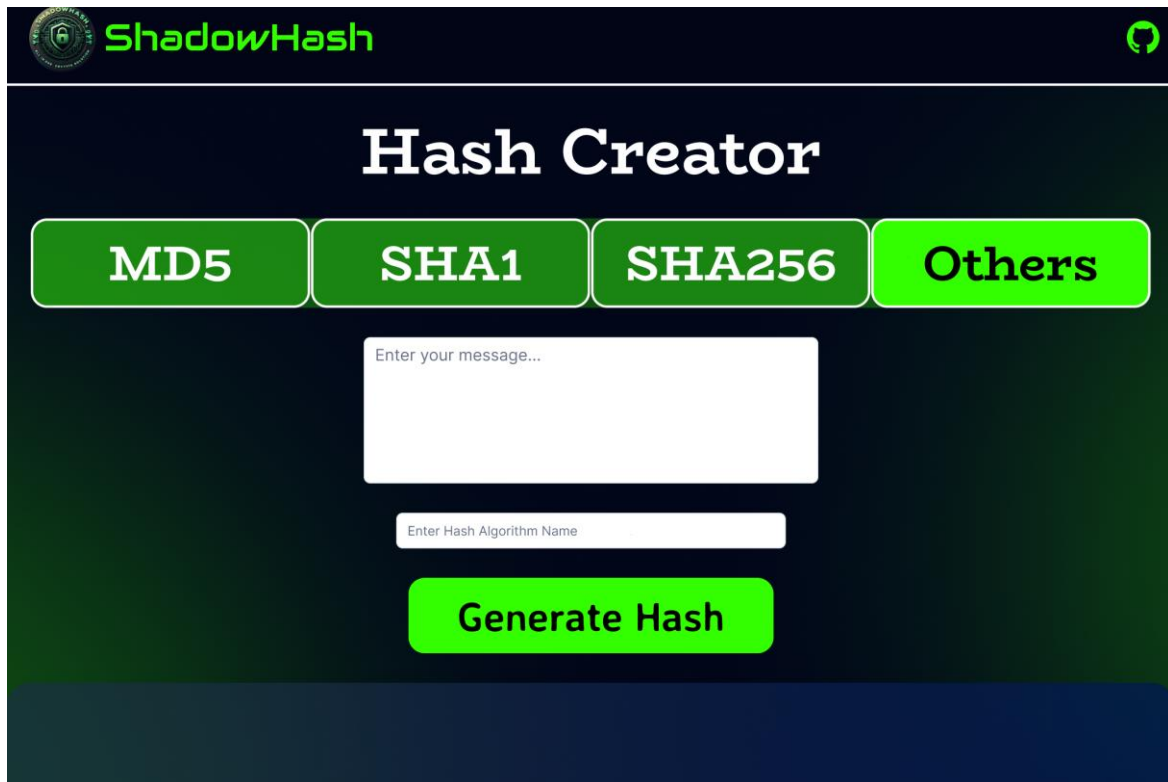


Image 36

4. **Others:** This section hashes (one-way function) the user given string message using the user given hashing algorithm and displays the hash for further use.



The image shows a web application titled "ShadowHash" with a logo on the left and a refresh icon on the right. The main heading is "Hash Creator". Below this, there are four buttons: "MD5", "SHA1", "SHA256", and "Others". The "Others" button is highlighted in red. Below the buttons is a large text input field with the placeholder "Enter your message...". Below that is a smaller text input field with the placeholder "Enter Hash Algorithm Name". At the bottom is a red button labeled "Generate Hash".

Image 37



## Hash Comparison Checker:

The “Hash Comparison Checker” of ShadowHash includes four sub-parts having different hashing algorithms (MD5, SHA1 and SHA256 which are the most commonly used along with a generic one for any other algorithm) to check for data authenticity along with a header linking the home page and GitHub repository of the same.

1. **MD5:** Message Digest 5 hashes (one-way function) the user given string message and compares the digest with the user given hash to check for data authenticity. MD5 is a popularly known hashing algorithm which still used quite popularly yet is vulnerable to many attack vectors.

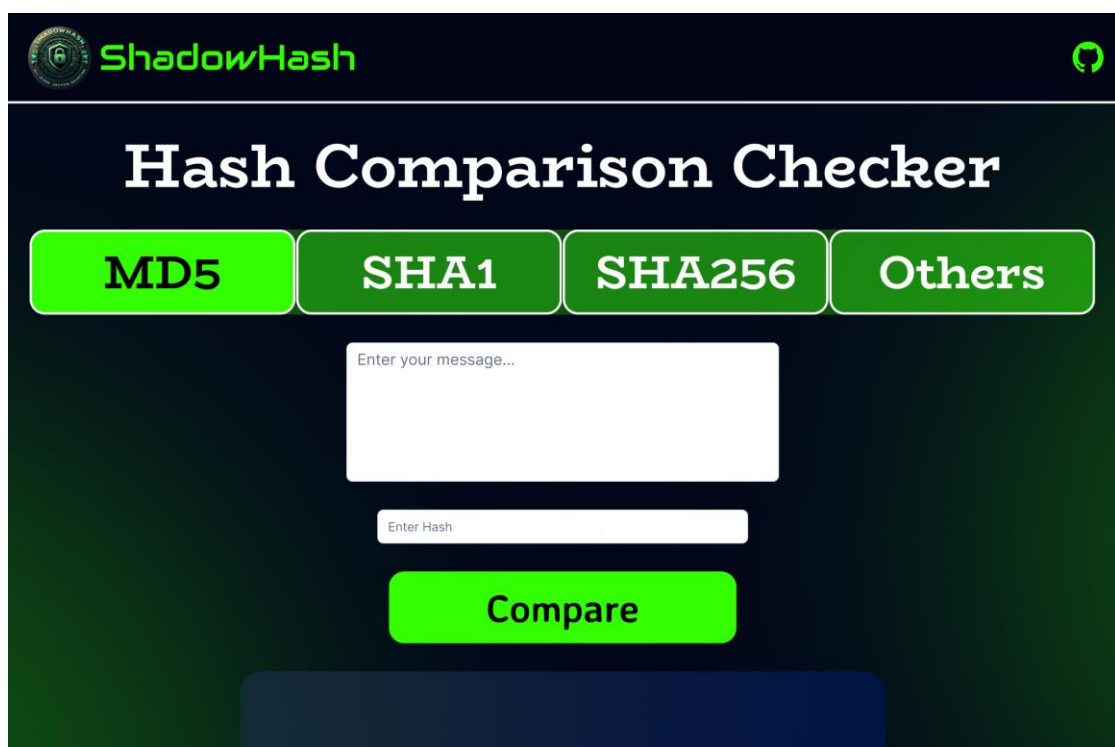


Image 38

2. **SHA1:** Secure Hash Algorithm 1 hashes (one-way function) the user given string message and compares the digest with the user given hash to check for data authenticity. SHA1 is a popularly known hashing algorithm which used popular and more secure as compared to MD5.

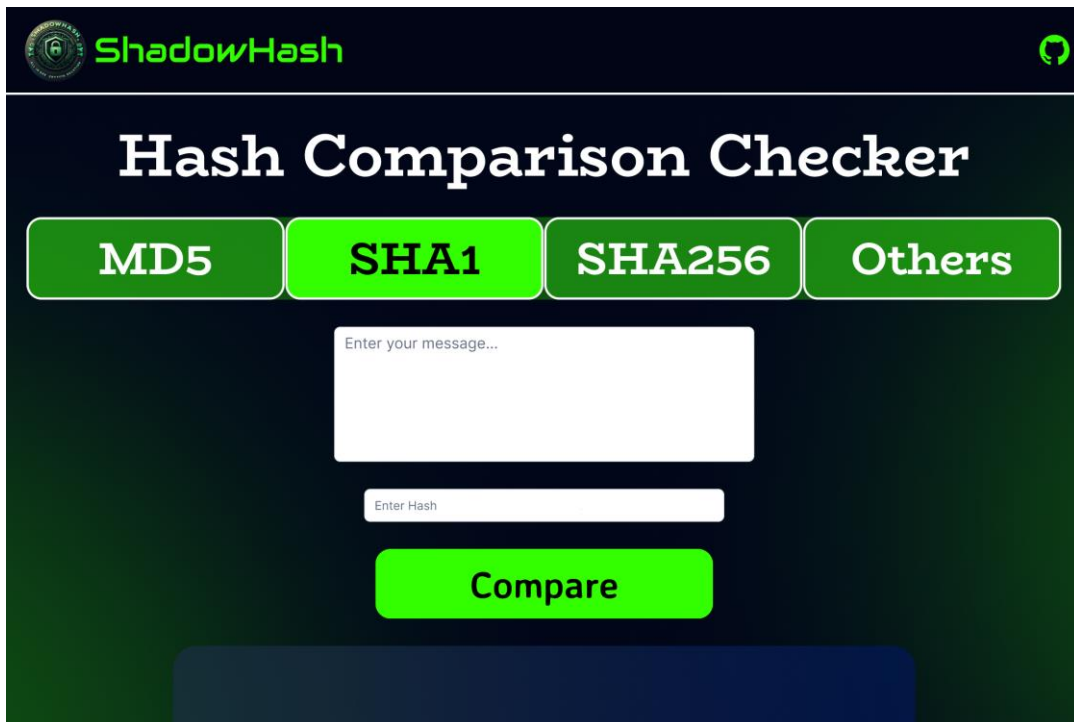


Image 39

3. **SHA256:** Secure Hash Algorithm 256 hashes (one-way function) the user given string message and compares the digest with the user given hash to check for data authenticity. SHA256 yet an algorithm stronger than both the above mentioned ones is also a widely used.

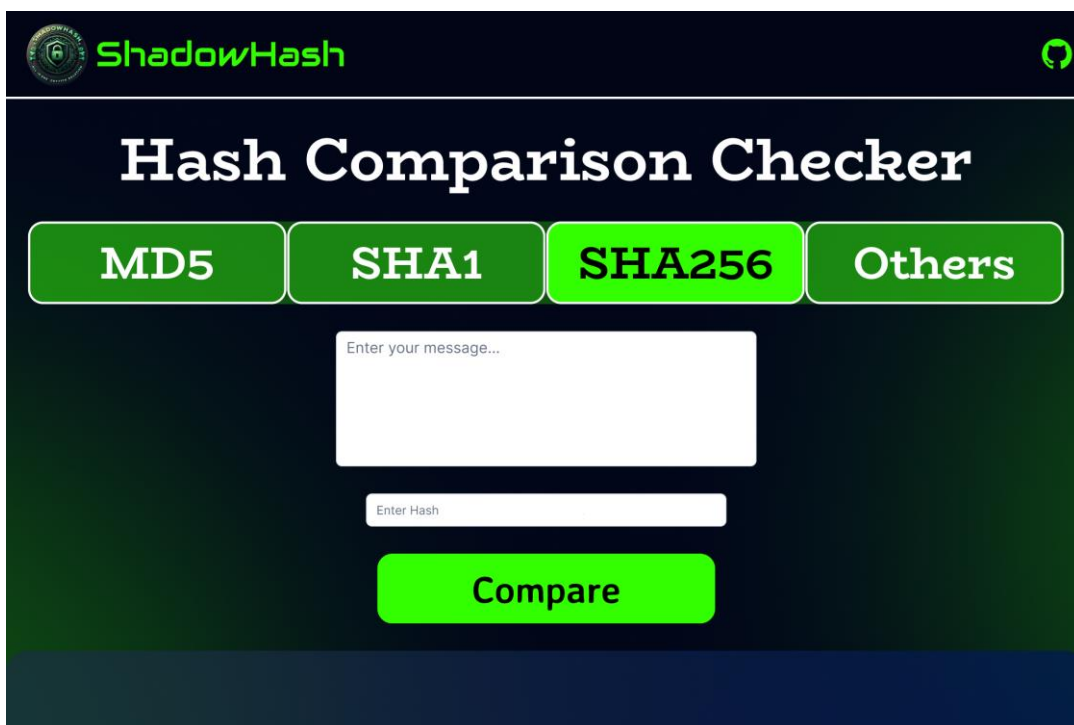


Image 40

4. **Others:** This section hashes (one-way function) the user given string message using the user given hashing algorithm and compares the digest with the user given hash to check for data authenticity.



The image shows a web application interface for "ShadowHash". At the top, there is a logo on the left and a green circular icon on the right. The main heading is "Hash Comparison Checker". Below the heading, there are four buttons: "MD5", "SHA1", "SHA256", and "Others". The "Others" button is highlighted in red. Below these buttons, there is a large text input field labeled "Enter your message...". Below this field are two smaller input fields: "Enter Hash Algorithm Name" and "Enter Hash". At the bottom, there is a red button labeled "Compare".

Image 41

## File Virus Checker:

The “File Virus Checker” of ShadowHash checks if the user file virus infested or not which is done via the VirusTotal API. Alongside, the webpage also contains a header linking the home page and GitHub repository of the same.



Image 42

## Chapter 7

### Results and Discussions

ShadowHash's development has made notable progress toward its goals of developing an approachable platform that incorporates necessary cryptographic functions. The backend, written in Java, uses effective and safe algorithms to carry out a range of cryptographic tasks. The frontend, which was created using HTML and CSS, guarantees a user interface that is responsive and easy to use, improving the user experience.

1. **Email Breach Detector:** This tool, which makes use of the Hackcheck Woventeams API, alerts users to potential password breaches by identifying compromised email addresses. By integrating this API, account security was greatly improved by ensuring real-time checks against known data breaches.
2. **Password Breach Checker:** To determine if user credentials had been hacked, the HaveIBeenPwned API was used. User privacy was protected by the secure SHA-1 hashing technique, which sent just the hash's first five characters. This method worked well for accurately detecting breaches and safeguarding user credentials.
3. **Password Creator:** The tool for creating passwords showed adaptability and a commitment to security best practices. Users might create secure passwords or alter them to suit their tastes while striking a balance between difficulty and memorability. The tool added to the password strength by preventing consecutive characters.
4. **Encryption and Decryption Algorithms:** ShadowHash included a number of encryption techniques, such as Blowfish, RSA, AES, 3DES, and a proprietary algorithm. Because of its adaptability, users were able to choose encryption techniques according to what they needed. The safe transmission and storage of data was made possible by the effective use of these algorithms.
5. **Comparison Checker and Hash Generator:** Users may create message digests and compare hash values to ensure data integrity by using the platform's hashing tools. ShadowHash gave users strong tools for data verification by supporting algorithms including MD5, SHA-1, and SHA-256, which efficiently detected any alterations or manipulations.

6. **Virus Detector:** By integrating the VirusTotal API, ShadowHash gave customers the ability to check files for malware and detect possible dangers by using an extensive database. This feature shielded users against malicious code, which made a substantial contribution to the entire security architecture.

## **Chapter 8**

### **Conclusion**

ShadowHash exemplifies a comprehensive and user-centric approach to cryptographic security, addressing the multifaceted needs of individuals and organizations alike. By integrating essential features such as email breach detection, password security, encryption, decryption, hash generation, and virus detection, ShadowHash provides a robust and unified platform for safeguarding digital assets.

The project's transition from a Java backend to a JavaScript-based solution, utilizing Express, Vanilla JavaScript, and Node.js, underscores our commitment to flexibility and compatibility. This change enhances the platform's integration with the front-end technologies, ensuring a seamless user experience.

ShadowHash's deployment on AWS Amplify, coupled with advanced monitoring and protection services like SNS, CloudWatch, and AWS Shield, guarantees scalability, reliability, and security. The careful design and implementation of the platform's various components reflect a meticulous approach to both functionality and user experience.

In conclusion, ShadowHash stands as a testament to the power of integrated cryptographic solutions, offering a reliable and efficient means for users to secure their digital interactions. By continuously monitoring and updating the platform, we ensure that ShadowHash remains at the forefront of cryptographic security, meeting the evolving needs of its users and providing peace of mind in an increasingly digital world.

## Chapter 9

### Future Scope

Although many cryptographic aspects have been successfully implemented by ShadowHash, there are still a number of areas that might want improvement and growth in the future. Among them are:

1. **Mobile Application:** Creating an application for mobile devices to enhance the online platform. This will improve usability and convenience by giving users access to ShadowHash's capabilities when they're on the road.
2. **Advanced Threat Detection:** Using machine learning methods to improve threat detection skills. ShadowHash might provide proactive security measures by more accurately anticipating and mitigating any attacks via the analysis of patterns and behaviors.
3. **Biometric Authentication:** Including biometric authentication techniques like face recognition and fingerprint authentication. By adding another layer of security and enhancing user comfort, this will improve user authentication.
4. **Multi-Factor Authentication (MFA):** Adding MFA to user accounts will increase account security. Through the use of several verification mechanisms, such SMS codes or authenticator applications, ShadowHash can guarantee that access is only granted to authorized individuals.
5. **Expanded Encryption Algorithms:** New and cutting-edge encryption algorithms are added to the platform on a regular basis. ShadowHash is kept at the forefront of cryptography by keeping up with new developments.
6. **Improved User Interface:** Making constant improvements to the user interface in response to user input. Maintaining the platform's usability and intuitiveness is essential to encouraging user adoption and happiness.
7. **Integrating Global Threat information:** Including feeds of global threat information to remain current on security risks. This real-time data may improve ShadowHash's capacity to identify and address emerging problems.



ShadowHash can grow and provide even more reliable and complete security solutions by pursuing these improvements. The platform will continue to be a useful resource for people and businesses looking to protect their digital assets because of its dedication to ongoing development and adaptability to new technologies.

## Chapter 10

### List of Images

1. **Image 1:** Pert Chart
2. **Image 2:** ShadowHash Home Page (Landing Page)
3. **Image 3:** ShadowHash Home Page (Benefits)
4. **Image 4:** ShadowHash Home Page (Benefits)
5. **Image 5:** ShadowHash Home Page (Services)
6. **Image 6:** ShadowHash Home Page (Resources)
7. **Image 7:** ShadowHash Home Page (Resources)
8. **Image 8:** ShadowHash Home Page (Footer)
9. **Image 9:** ShadowHash About Us Page (Our Aim)
10. **Image 10:** ShadowHash About Us Page (What Does ShadowHash Do)
11. **Image 11:** ShadowHash About Us Page (Our Mission)
12. **Image 12:** ShadowHash About Us Page (Our Team)
13. **Image 13:** ShadowHash Terms of Use Page (Overview)
14. **Image 14:** ShadowHash Terms of Use Page (Use of Service, User Content and Third-Party APIs)
15. **Image 15:** ShadowHash Terms of Use Page (Disclaimer, Governing Law and Changes to Terms)
16. **Image 16:** ShadowHash Terms of Use Page (Penetration Testing Rights and Contact Us)
17. **Image 17:** ShadowHash Breach Checker Page (Email Breach Checker)
18. **Image 18:** Hackcheck Woventeams API information.

Credits: <https://hackcheck.woventeams.com/api/v4/>

19. **Image 19:** ShadowHash Breach Checker Page (Password Breach Checker)

20. **Image 20:** HaveIBeenPwned API information.

Credits: <https://haveibeenpwned.com/API/v3#SearchingPwnedPasswordsByRange>

21. **Image 21:** ShadowHash Password Strength Detector Page

22. **Image 22:** ShadowHash Password Generator Page (Random Password Generator)

23. **Image 23:** ShadowHash Password Generator Page (Data-based Password Generator)

24. **Image 24:** ShadowHash Encryption Tool Page (AES Encrpytion)

25. **Image 25:** ShadowHash Encryption Tool Page (3DES Encrpytion)

26. **Image 26:** ShadowHash Encryption Tool Page (Blowfish Encrpytion)

27. **Image 27:** ShadowHash Encryption Tool Page (SH Special Encrpytion)

28. **Image 28:** ShadowHash Encryption Tool Page (RSA Encrpytion)

29. **Image 29:** ShadowHash Decryption Tool Page (AES Decrpytion)

30. **Image 30:** ShadowHash Decryption Tool Page (3DES Decrpytion)

31. **Image 31:** ShadowHash Decryption Tool Page (Blowfish Decrpytion)

32. **Image 32:** ShadowHash Decryption Tool Page (SH Special Decrpytion)

33. **Image 33:** ShadowHash Decryption Tool Page (RSA Special Decrpytion)

34. **Image 34:** ShadowHash Hash Creator Page (MD5 Hashing Algorithm)

35. **Image 35:** ShadowHash Hash Creator Page (SHA1 Hashing Algorithm)

36. **Image 36:** ShadowHash Hash Creator Page (SHA256 Hashing Algorithm)

37. **Image 37:** ShadowHash Hash Creator Page (Other Hashing Algorithms)

38. **Image 38:** ShadowHash Hash Creator Checker Page (MD5 Hashing Algorithm)

39. **Image 39:** ShadowHash Hash Creator Checker Page (SHA1 Hashing Algorithm)

40. **Image 40:** ShadowHash Hash Creator Checker Page (SHA256 Hashing Algorithm)

41. **Image 41:** ShadowHash Hash Creator Checker Page (Other Hashing Algorithms)

42. **Image 42:** ShadowHash File Virus Checker Page

## References

1. EC-Council, Certified Ethical Hacker (CEH) Version 12. Available from: VitalSource Bookshelf, (12th Edition). International Council of E-Commerce Consultants (EC Council), 2022.
2. “Advanced encryption standard (AES),” Nov. 2001. doi: 10.6028/nist.fips.197.
3. E. Barker and N. Mouha, “Recommendation for the Triple Data Encryption Algorithm (TDEA) block cipher,” Nov. 2017. doi: 10.6028/nist.sp.800-67r2.
4. Samsung/atsec information security, “Samsung FIPS BC for Mobile Phone and Tablet FIPS 140-2 Security Policy,” report, Feb. 2014. [Online]. Available: <https://csrc.nist.gov/csrc/media/projects/cryptographic-module-validation-program/documents/security-policies/140sp2092.pdf>
5. National Institute of Standards and Technology, “FIPS 186-5,” U.S. Department of Commerce, Feb. 2023. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf>
6. Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, U.S. Department of Commerce, “Hash Functions | CSRC | CSRC.” <https://csrc.nist.gov/Projects/Hash-Functions>
7. “AWS Academy Student guide.” <https://awsacademy.instructure.com/courses/68568/modules/items/6098904>