**3rd Sem Mini Project Report on**



**PASSWORD GENERATOR**



**Submitted in partial fulfillment of the requirement for the award of the degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE & ENGINEERING (CYBERSECURITY)**

**Submitted by:**

**Student Name-Lucky Singh Rawat** **University Roll No.- 2319044**



**Department of Computer Science and Engineering**

**Graphic Era Hill University**

**Dehradun, Uttarakhand**

**2024-25**

**CANDIDATE’S DECLARATION**

I hereby certify that the work which is being presented in the project report entitled **“Password Manager ”** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering **(Cybersecurity)** in the Department of Computer Science and Engineering of the Graphic Era Hill University, Dehradun shall be carried out by the undersigned under the supervision of **Mr. Nishant Bhandari, Assistant Professor**, Department of Computer Science and Engineering, Graphic Era Hill University, Dehradun.

Name - **Lucky Singh Rawat** University Roll no - **2319044**

The above-mentioned student shall be working under the supervision of the undersigned on the **“PASSWORD MANAGER”**

**Supervisor** **Head of the Department**

**Examination**

**Name of the Examiners: Signature with Date**

1.

2.

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Description** | **Page No.** |
| Chapter 1 | Introduction and Problem Statement | **4** |
| Chapter 2 | Methodology | **5** |
| Chapter 3 | Project Work Carried Out | **7** |
| Chapter 4 | Results and Discussion | **9** |
| Chapter 5 | Conclusion and Future Work | **11** |
| Chapter 6 | References | **13** |
|  |  |  |

Chapter 1 – Introduction

Welcome to the Password Manager project! This project aims to provide a secure and efficient solution for generating, storing, and managing strong passwords. With the increasing need for online security, it is essential to have unique and complex passwords for each website you use. However, remembering all these passwords can be a daunting task. This is where our Password Manager comes into play.

Our Password Manager offers a user-friendly interface that allows you to generate strong passwords, save them securely with their corresponding website names, and verify the stored passwords when needed. The project uses the cryptography library for encryption and decryption to ensure your passwords are stored securely. The tkinter library is used to create a simple graphical user interface, making it easy for users to interact with the application.

**Features**

* **Generate Strong Passwords**: Create complex and secure passwords with a single click.
* **Save Passwords Securely**: Store your passwords in an encrypted format to protect them from unauthorized access.
* **Check Passwords**: Verify if a given password matches the stored password for a specific website.
* **No Duplicate Website Names**: Ensure that each website name is unique, preventing any potential confusion or conflicts.

**Benefits**

* **Enhanced Security**: By using strong, unique passwords for each website, you reduce the risk of cyber-attacks and unauthorized access.
* **Ease of Use**: The intuitive interface makes it easy to generate, save, and manage your passwords.
* **Reliability**: With secure encryption and decryption, you can trust that your passwords are stored safely.

Chapter 2 – Problem Statement

**Problem Statement 1 : Weak Passwords**

**Issue**: Many users tend to use weak or simple passwords for convenience, which makes their accounts vulnerable to hacking and unauthorized access.

**Solution**: The Password Manager generates strong, complex passwords using a combination of letters, digits, and special characters. This reduces the risk of password guessing and brute-force attacks.

**Problem Statement 2 : Difficulty in Remembering Multiple Passwords**

**Issue**: Users often struggle to remember multiple strong passwords for different accounts, leading to the reuse of passwords across multiple sites.

**Solution**: The Password Manager securely stores and organizes passwords for different websites, allowing users to manage them without the need to remember each one individually. The encrypted storage ensures that the passwords remain secure.

**Problem Statement 3 : Duplicate Website Names**

**Issue**: Storing multiple entries for the same website can lead to confusion and conflicts, making it difficult to manage and retrieve the correct password.

**Solution**: The Password Manager ensures that no duplicate website names are stored. It checks for existing entries and alerts the user if a duplicate is found, maintaining a clear and organized password database.

**Problem Statement 4 : Vulnerability to Unauthorized Access**

**Issue**: Storing passwords in plain text or using weak encryption methods can make them susceptible to unauthorized access and cyber-attacks.

**Solution**: The Password Manager uses strong encryption methods provided by the cryptography library to securely store passwords. This ensures that even if the password file is accessed by an unauthorized person, the encrypted passwords remain safe and unreadable.

**Problem Statement 5 : Inconsistent Password Policies**

**Issue**: Different websites have different password policies, and users may struggle to create passwords that meet all the required criteria.

**Solution**: The Password Manager allows users to generate passwords of customizable lengths, incorporating various character types to meet different website policies. This flexibility ensures that users can create passwords that comply with specific requirements.

Chapter – 3 Methodology

1. **Requirements Analysis**

* Identify the need for secure password management and determine the essential features: password generation, secure storage, retrieval, and verification.
* Gather user requirements and specify the functionalities to be implemented.

2. **Design**

* **System Architecture**: Design a modular architecture with separate components for password generation, encryption/decryption, and user interface.
* **User Interface Design**: Create a simple and intuitive GUI using tkinter for easy interaction.
* **Security Design**: Incorporate strong encryption methods using the cryptography library to ensure data security.

3. **Implementation**

* **Password Generator Module**: Develop a function to generate strong passwords using a combination of letters, digits, and special characters.
* **Encryption Manager Module**: Implement functions to encrypt and decrypt passwords using symmetric key encryption.
* **User Interface Module**: Build the GUI using tkinter to facilitate user interactions, including saving, generating, and checking passwords.
* **Data Storage**: Implement a method to store encrypted passwords in a text file and ensure no duplicate website names are stored.

4. **Testing**

* **Unit Testing**: Test individual functions and modules to ensure they work correctly. Verify the password generation, encryption, and decryption processes.
* **Integration Testing**: Test the integration of different modules to ensure seamless interaction and data flow between components.
* **User Acceptance Testing**: Conduct testing with end-users to ensure the application meets their requirements and expectations.

5. **Documentation**

* **Code Documentation**: Provide clear and concise comments within the code to explain the functionality and purpose of different sections.
* **User Documentation**: Create a user manual or README file to guide users on how to install, run, and use the application effectively.

6. **Deployment**

* Package the application and its dependencies for distribution.
* Provide installation instructions and ensure the application can be easily set up by users.

7. **Maintenance and Updates**

* Monitor the application for any issues or bugs reported by users.
* Provide regular updates and enhancements based on user feedback and evolving security standards.

Chapter 4 – Project work carried out

1. **Initial Research and Planning**
   * Conducted research on existing password managers to identify common features and security practices.
   * Defined the scope of the project and outlined the key functionalities required for the Password Manager.
2. **Design and Architecture**
   * Designed the system architecture, dividing the project into three main modules: Password Generator, Encryption Manager, and User Interface.
   * Created detailed design documents to outline the flow of data and interactions between components.
3. **Development**
   * **Password Generator Module**: Implemented the generate password function to create strong, random passwords using a combination of letters, digits, and special characters.
   * **Encryption Manager Module**: Developed functions to handle the encryption and decryption of passwords using the cryptography library. This included generating and storing a key, as well as encrypting and decrypting password strings.
   * **User Interface Module**: Built the GUI using tkinter, allowing users to interact with the application easily. This included fields for entering website names and passwords, buttons for generating and saving passwords, and functionality for checking stored passwords.
4. **Integration**
   * Integrated the Password Generator and Encryption Manager modules with the User Interface module to create a cohesive application.
   * Ensured seamless data flow and interaction between the components, allowing users to generate, save, and verify passwords efficiently.
5. **Testing and Debugging**
   * Conducted unit testing on individual functions to ensure they performed correctly.
   * Performed integration testing to verify that the modules interacted as expected.
   * Debugged any issues that arose during testing, ensuring the application was stable and reliable.
6. **User Feedback and Iteration**
   * Collected feedback from users during the testing phase to identify areas for improvement.
   * Made iterative changes based on user feedback, enhancing the application's usability and functionality.
7. **Documentation**
   * Created comprehensive documentation, including a README file with installation and usage instructions.
   * Provided detailed comments within the code to explain the functionality and purpose of different sections.
8. **Deployment and Maintenance**
   * Packaged the application for distribution, providing installation instructions to users.
   * Established a plan for ongoing maintenance and updates, addressing any issues reported by users and incorporating new features based on user feedback.

Chapter 5 - Results and Discussion

# **Results**

1. **Password Generation**:
   * Successfully implemented a function to generate strong, random passwords.
   * Users can generate passwords with a combination of letters, digits, and special characters, enhancing security.
2. **Secure Storage**:
   * Implemented encryption and decryption functions using the cryptography library.
   * Passwords are securely stored in an encrypted format in the passwords.txt file, ensuring they are protected from unauthorized access.
3. **User Interface**:
   * Developed a user-friendly interface using tkinter, making it easy for users to generate, save, and check passwords.
   * Included features to prevent duplicate website entries and ensure unique website names.
4. **Password Verification**:
   * Added functionality to verify if a given password matches the stored password for a specific website.
   * Users can easily check the correctness of their passwords, improving usability and reducing errors.
5. **No Duplicate Website Names**:
   * Ensured that the application checks for existing website names before saving new entries.
   * Prevented duplicate website entries, maintaining a clear and organized password database.

# **Discussion**

1. **Security**:
   * The use of strong, complex passwords significantly reduces the risk of unauthorized access and cyber-attacks.
   * Encrypting passwords ensures that even if the password file is accessed, the passwords remain secure and unreadable without the decryption key.
   * By storing the encryption key separately, the security of the stored passwords is further enhanced.
2. **Usability**:
   * The intuitive GUI allows users to interact with the application easily, making password management straightforward.
   * The functionality to generate and save passwords with a single click simplifies the process and encourages users to use strong passwords.
3. **Reliability**:
   * The application reliably generates, stores, and retrieves passwords, providing a dependable solution for password management.
   * By ensuring unique website names, the application avoids potential conflicts and confusion in password storage.
4. **Scalability**:
   * The modular design allows for easy extension and addition of new features.
   * Future enhancements could include more advanced features like multi-factor authentication, cloud storage integration, and password strength analysis.
5. **User Feedback**:
   * User testing and feedback revealed that the application effectively addresses common password management challenges.
   * Users appreciated the simplicity and security provided by the Password Manager.

**Chapter 6 - Conclusion and Future Work**

# **Conclusion**

The Password Manager project successfully achieved its goals of providing a secure and user-friendly solution for generating, storing, and managing passwords. The combination of strong password generation, secure storage through encryption, and an intuitive interface ensures that users can enhance their online security with minimal effort. Future improvements and additional features can further enhance the application's functionality and user experience.

# **Future Work**

1. **Multi-Factor Authentication (MFA)**
   * Implement Multi-Factor Authentication to enhance security further. This could include integrating SMS, email, or authenticator apps to verify user identity during login or password retrieval.
2. **Password Strength Analysis**
   * Develop a feature to analyse and provide feedback on the strength of user-generated passwords. This could include suggestions to improve password complexity based on common password policies.
3. **Cloud Storage Integration**
   * Integrate cloud storage options like Google Drive, Dropbox, or OneDrive to store encrypted passwords. This ensures that users can access their passwords securely from any device.
4. **Auto-Fill Capabilities**
   * Develop browser extensions or desktop applications that can auto-fill login forms with stored passwords. This improves convenience while maintaining security.
5. **Mobile Application Development**
   * Create a mobile version of the Password Manager for both Android and iOS platforms. This allows users to manage their passwords securely on the go.
6. **User Authentication System**
   * Implement a secure user authentication system to manage multiple user accounts. This could include registration, login, and password recovery features.
7. **Password Recovery Options**
   * Provide secure methods for recovering lost or forgotten passwords, such as security questions, backup codes, or biometric authentication.
8. **Periodic Security Audits**
   * Schedule regular security audits to identify and fix vulnerabilities in the application. This ensures that the Password Manager stays up-to-date with the latest security standards.
9. **Customizable Password Policies**
   * Allow users to define custom password policies based on their specific requirements. This could include setting minimum password lengths, required character types, and expiration periods.
10. **User-Friendly Tutorials and Documentation**

* Develop detailed tutorials and documentation to help users understand and utilize all features of the Password Manager. This improves user engagement and satisfaction.

**Chapter 7 – References**

1. **Python Documentation**:
   * Official Python documentation for understanding the Python programming language and its libraries.
   * Python Official Documentation
2. **Cryptography Library**:
   * Documentation and resources for the cryptography library used for encryption and decryption.
   * Cryptography Documentation
3. **tkinter Library**:
   * Official documentation for tkinter, the standard Python interface to the Tk GUI toolkit.
   * tkinter Documentation
4. **pyperclip Library**:
   * Documentation for the pyperclip library used to copy text to the clipboard.
   * pyperclip Documentation
5. **Random Number Generation**:
   * Resources and guides on using the random module for generating random numbers and passwords.
   * Python random Module