

To the Graduate Council:

I am submitting herewith a thesis written by David Hua Huang entitled “Bridging User Preferences and Security Demands: A User-Centric Approach to Password Generation.” I have examined the final paper copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

---

Dr. Scott Ruoti, Major Professor

We have read this thesis  
and recommend its acceptance:

---

Dr. Scott Ruoti

---

Dr. Michael Berry

---

Dr. Catherine Schuman

Accepted for the Council:

---

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

To the Graduate Council:

I am submitting herewith a thesis written by David Hua Huang entitled “Bridging User Preferences and Security Demands: A User-Centric Approach to Password Generation.” I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

Dr. Scott Ruoti, Major Professor

We have read this thesis  
and recommend its acceptance:

Dr. Scott Ruoti

---

Dr. Michael Berry

---

Dr. Catherine Schuman

---

Accepted for the Council:

Dixie L. Thompson

---

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

# **Bridging User Preferences and Security Demands: A User-Centric Approach to Password Generation**

A Thesis Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

David Hua Huang

August 2024

© by David Hua Huang, 2024  
All Rights Reserved.

*To my family, friends, fiancée, and God for all the constant support, guidance, and  
love throughout this endeavor.*

# Acknowledgements

I would like to thank my advisor Dr. Scott Ruoti who made this all possible. His continuous support and guidance was instrumental throughout all stages of conducting my research and writing this thesis. I would also like to thank my fellow lab members of the User Lab for their advice and feedback. This work is based upon research supported by the National Science Foundation under award CNS-2226404.

*Trust in the LORD with all your heart and lean not on your own understanding; in all your ways submit to Him, and He will make your paths straight. - Proverbs 3:5-6*

# Abstract

This thesis introduces a novel password generation algorithm that aligns user-specified password composition policies (PCPs) with those required by websites, aiming to enhance security and usability. Traditional password generators focus on maximizing entropy but often neglect user ease, producing passwords that are either too complex to remember or too simple to be secure. Our research proposes a user-centric interface and algorithm that integrates the PCPs articulated by users with website requirements, facilitating a balance between security and convenience. We developed a system architecture that includes a baseline interface inspired by existing password generators and an advanced, user-centric interface that collects comprehensive user data, such as sensitivity preferences and device usage. Our methodology involves experimental testing to evaluate the algorithm's security and functionality. Initial tests confirm that our algorithm can merge different PCPs and produce compliant, secure passwords. Our work not only demonstrates the feasibility of a user-centric approach to password generation but also highlights its practical benefits. By emphasizing enhanced security and user satisfaction without overcomplicating the user experience, our approach paves the way for a more secure and user-friendly digital landscape, instilling optimism about its potential implementation.

**Keywords:** password generation, user-centric interface, password composition policies (PCPs), security, usability, system architecture



# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Background</b>	<b>4</b>
2.1	Password Managers (PWMs) . . . . .	4
2.1.1	Introduction and Core Functionality . . . . .	4
2.1.2	Features . . . . .	5
2.1.3	The Life Cycle of a Password Manager . . . . .	7
2.1.4	Benefits . . . . .	7
2.1.5	Popular Examples . . . . .	8
2.1.6	Security Mechanisms . . . . .	8
2.2	Password Composition Policies (PCPs) . . . . .	10
2.2.1	Introduction and Core Principles . . . . .	10
2.2.2	Effective Implementation of PCPs . . . . .	10
2.2.3	Challenges and Innovations . . . . .	11
2.3	Related Works . . . . .	12
<b>3</b>	<b>System Design</b>	<b>15</b>
3.1	Architectural Overview . . . . .	15
3.2	User Interface Design . . . . .	16
3.2.1	Baseline User Interface . . . . .	16
3.2.2	Advanced User-Centric Interfaces . . . . .	18
3.2.3	System Summary . . . . .	23

<b>4</b>	<b>Algorithm Design</b>	<b>25</b>
4.1	User Input Translation . . . . .	26
4.2	PCP Construction . . . . .	27
4.3	Merging PCPs . . . . .	29
4.3.1	Merging Length Requirements . . . . .	29
4.3.2	Merging Maximum Consecutive Character Limits . . . . .	32
4.3.3	Combining Prohibited Substrings . . . . .	32
4.3.4	Incorporating Required Character Sets . . . . .	32
4.3.5	Merging Subset Requirements . . . . .	32
4.3.6	Merging Character Set Requirements . . . . .	33
4.4	Password Generation . . . . .	34
4.5	Validation . . . . .	39
4.6	Algorithm Summary . . . . .	39
<b>5</b>	<b>Discussion</b>	<b>42</b>
5.1	Lessons Learned . . . . .	42
5.2	Implications for Cybersecurity Practices . . . . .	43
5.3	Future Work . . . . .	44
5.3.1	Novel Interfaces . . . . .	45
5.3.2	Comprehensive User Study . . . . .	47
5.3.3	Exploration of Passphrase Compatibility . . . . .	50
<b>6</b>	<b>Conclusion</b>	<b>53</b>
	<b>Vita</b>	<b>59</b>

# Chapter 1

## Introduction

In the digital age, passwords are the primary defense of personal and organizational data across the internet. Their ubiquity and continuation as the predominant form of authentication underscore a crucial yet often precarious balance between security and usability [1]. Despite the increase of alternative authentication mechanisms (e.g., multi-factor authentication, biometric authentication, push notification authentication), passwords remain unparalleled in their widespread adoption and accessibility. However, this reliance on passwords introduces significant challenges, notably in balancing the need for solid and secure passwords against the human limitations of memorability and convenience.

Our research addresses the inherent dilemma users face: strong passwords, essential for robust security, often result in predictable and weak choices due to the challenges of managing them effectively. Dell'Amico et al. demonstrate that, despite using advanced password attack strategies, user-chosen passwords often show significant predictability, indicating a trade-off between password strength and memorability [3]. Furthermore, user tendencies towards password reuse across multiple platforms exacerbate this paradox, increasing security risks. Florencio and Herley found that this behavior can dramatically heighten security risks; if one account is compromised, all other accounts using the same password become

vulnerable to unauthorized access [4]. Riley highlights the frequent reliance on personally meaningful but potentially insecure password elements, contributing to the predictability and weakness of user-chosen passwords [21]. Additionally, strategies such as creating variations of the same password highlight the significant challenges in maintaining secure and unique passwords. Pearman et al. report that users often resort to exact and partial password reuse, driven by the daunting task of memorizing distinct passwords for an average of over 26 web domains, with different website categories influencing password practices, revealing a nuanced landscape of security behaviors and perceptions [19].

Password managers have emerged as a technological solution to mitigate the challenges of generating, storing, and auto-filling complex passwords. While these tools offer the potential to enhance security and convenience, research by Pearman et al. reveals that their adoption and utilization, particularly of password generation features, still need to be improved. Users of built-in, browser-based password managers often prioritize convenience, whereas users of separately installed managers emphasize security [20]. Furthermore, Oesch et al. provide insights into the real-world use of password managers through observational interviews with 32 users. They find that many users employ both browser-based and third-party managers concurrently, using each as a backup for the other. This mixed usage pattern highlights a complex interplay between convenience and security concerns and reveals that users frequently eschew generated passwords when these are difficult to enter or recall without manager support [17]. Huaman et al. also identify significant interaction problems between password managers and websites, primarily due to misalignment between automatically generated passwords and websites' Password Composition Policies (PCPs). This misalignment often leads to password rejections, undermining the utility of password managers. Their systematic analysis of user reviews and GitHub issues for 30 password managers uncovers 39 distinct interaction problems, underscoring the need for improved compliance between password manager outputs and website requirements to enhance the effectiveness of these security solutions [8].

Building upon the work of Gautam et al., who sought to facilitate compliant password generation by developing a Password Composition Policy description language, this thesis extends their framework by incorporating a user-centric perspective. Gautam’s research achieved syntactical harmony between password managers and website-enforced PCPs [6]. Our work further develops this by addressing the gap between the technical constraints of PCPs and users’ varied needs and preferences. We have modified the system architecture for password generation and developed interfaces and algorithms that support the creation of secure, personalized passwords. A key feature of our approach is a dual-interface architecture, which includes a baseline inspired by existing password generators and an advanced, user-centric interface. This interface collects user inputs such as sensitivity preferences, device usage, memorability needs, and password manager access, translating them into human-centric PCPs. We then dynamically map these human PCPs to their machine-readable counterparts, integrate multiple PCPs (user-defined and website-specific), and implement a password generation algorithm to produce compliant, user-tailored passwords.

The research methodology employs comprehensive experimental testing to evaluate the robustness and practicality of the password generation algorithm. By analyzing password security, user behavior patterns, and password manager effectiveness, this thesis contributes to the broader discussion on digital security and password management. It underscores the importance of a user-centric approach in addressing persistent password generation and management challenges. Ultimately, the research aims to enhance the security framework of digital authentication systems and improve the process of creating strong passwords that better align with users’ daily digital interactions.

# Chapter 2

## Background

In this chapter, we explore the concept of a password manager, including its functions and duties. We also examine password composition policies in similar detail. Additionally, the chapter delves into previous studies and research related to Password Managers (PWMs) and Password Composition Policies (PCPs).

### 2.1 Password Managers (PWMs)

At its core, a password manager is a tool for storing user credentials, such as website and app usernames and passwords, to help the user relieve cognitive burdens associated with remembering login credentials [11].

#### 2.1.1 Introduction and Core Functionality

Password Managers (PWMs) are digital vaults designed to securely store login credentials, reducing the need for users to remember different passwords for numerous accounts. This section outlines their key features, lifecycle, benefits, PWM examples, and security implications.

## Essentials of Password Managers

At their heart, PWMs act as secure repositories, encrypting users' login details with a master password. This encryption ensures that only the user can access their stored information. Additionally, their automation of password generation, secure storage, and autofill capabilities are essential for user security and convenience [2]. Modern PWMs also offer cloud storage, allowing password synchronization across devices for easy access anywhere.

### 2.1.2 Features

Password managers provide a range of features, categorized into essential, recommended, and extended use cases based on their ubiquity and importance in the password management lifecycle, as systematized by Simmons et al. [23].

#### Essential Features

All password managers support these essential features, which include the core functionalities necessary for essential password management:

- **Credential Editing:** Users have full control over the credentials stored within the manager, meaning they can register, update, or remove any credentials.
- **Password Generation:** Integral to strengthening user security by creating unique, complex passwords that meet specific criteria.
- **Autofill Capabilities:** Facilitates a smoother user experience by automatically filling in credentials.
- **Credential Syncing:** Allows users to access their passwords across various devices, essential for modern multi-device environments.

## Recommended Features

Recommended features are not universally supported but provide significant improvements in usability and security:

- **Credential Audit:** Helps users identify and address security issues such as reused, weak, or leaked passwords, which is vital for maintaining password health.
- **Modify settings:** Allows users to customize the manager within their settings to meet their preferences and ensure secure behavior.
- **Recover Access:** Provides mechanisms for users to regain access to their vault in case of lost credentials (e.g., forgetting master password).

## Extended Features

Extended features are less common and cater to specific or advanced user needs:

- **Manager Migration:** Users sometimes switch password managers, so they would need the ability to export credentials from their old manager to import them into the new manager.
- **Share Credentials:** Facilitates secure sharing of credentials among users, useful for families or teams who need access to common accounts.
- **Manage Identities:** Allows users to segment credentials into different profiles, enhancing organizational capabilities and privacy.
- **Store Sensitive Data:** Provides the ability to securely store passwords and other sensitive information like payment details and personal notes, extending the utility of password managers beyond just password storage.

By leveraging Simmons et al.'s classification, we can better understand the scope and scale of features offered by modern password managers. Doing so enables users to make more informed decisions based on their specific security and usability needs.



### 2.1.3 The Life Cycle of a Password Manager

From account creation to autofill access, the PWM has several vital stages within its lifecycle, which include:

1. **Account Creation:** The user creates an account within the PWM, setting up a unique, strong master password to encrypt their vault.
2. **Secure Storage:** Login credentials for various accounts are encrypted and stored within the vault, with an encryption key commonly derived from the master password.
3. **Autofill Access:** When visiting a site or application, the PWM auto-fills the login details automatically or through a user input like ctrl + L. If the user has multiple accounts for a certain site, the PWM presents a list of login credential options.
4. **Password Updating:** PWMs can prompt users to update their passwords periodically or when the PWM senses a new password different from the currently stored one, assisting in maintaining strong security practices.

### 2.1.4 Benefits

Using a PWM appropriately offers many potential benefits:

1. **Cognitive Relief:** It helps relieve users of the cognitive burden of remembering login credentials.
2. **Password Reuse Mitigation:** By making it easier to assign different passwords to different websites and not having to remember them, password reuse is less likely.
3. **Stronger Passwords:** With built-in password generators, users can generate secure passwords (resilient to offline and online attacks) a lot easier.

### 2.1.5 Popular Examples

Several PWMs stand out for their robust security features and user-friendly design. Here is a look at a few notable examples:

1. **LastPass:** Offers a user-friendly interface and a variety of features such as secure password sharing, emergency access, and two-factor authentication. Its ability to generate strong passwords and its secure notes feature for storing sensitive information make it a comprehensive security tool.
2. **1Password:** Known for its strong focus on security and privacy, 1Password provides features like the Watchtower service, which alerts users about security breaches and reused passwords. It also supports secure document storage and family-sharing plans.
3. **Bitwarden:** As an open-source password manager, Bitwarden appeals to those who prioritize transparency in their security tools. It offers end-to-end encryption, cross-platform compatibility, and options for self-hosting, catering to both individual users and organizations.

Each PWM brings unique strengths, from LastPass's ease of use and comprehensive feature set to 1Password's additional security layers and Bitwarden's open-source flexibility and transparency. These notable PWMs offer extended features such as secure sharing and two-factor authentication, which are crucial for enhancing security while maintaining usability [7].

### 2.1.6 Security Mechanisms

The foundation of PWM security lies in a combination of advanced cryptographic techniques and user authentication methods, which collectively safeguard user data against a broad spectrum of threats:

- **End-to-end Encryption:** PWMs utilize end-to-end encryption to ensure that passwords and sensitive information are encrypted on the user's device before they get sent to the server for storage. Doing so means that even if an adversary intercepts the data, the data remains unreadable without the decryption key of the data's owner.
- **Zero-Knowledge Architecture:** This security model means the PWM's servers never store the decryption keys or raw user data. Essentially, even the service providers cannot access the user's stored passwords, enhancing privacy and security against data breaches.
- **Biometric Authentication:** Many PWMs integrate biometric authentication methods, such as fingerprint or facial recognition, adding another layer of security. This authentication method not only simplifies the login process for the user but also provides a strong defense against unauthorized access attempts.
- **Two-Factor Authentication (2FA):** For an added layer of security, 2FA requires a second form of verification, beyond just the master password, to access the PWM, significantly reducing the risk of unauthorized access.

These mechanisms, among others, equip PWMs to effectively protect user data from external hacking attempts and potential internal vulnerabilities. By adopting such comprehensive security measures, PWMs streamline password management and protect user data from external and internal threats [12].

As observed, PWMs have numerous features designed to assist users and enhance security protocols. However, realizing these benefits is contingent upon proper implementation and utilization of the PWMs. It is important to note that PWMs have flaws; this research will further explore some of the mentioned usability aspects and present improvements.

## 2.2 Password Composition Policies (PCPs)

PCPs dictate the required criteria for password creation on websites and applications, aiming to enhance security by compelling users to create passwords that are difficult to guess or brute-force.

### 2.2.1 Introduction and Core Principles

Designers create Password Composition Policies (PCPs) to improve account security by specifying requirements for password strength, such as length, complexity, and uniqueness. This section outlines the foundational aspects, objectives, and rationale behind PCPs and their impact on user behavior and security.

#### Foundational Aspects of PCPs

- **Definition and Objectives:** PCPs set forth rules for creating passwords, including minimum length, mandatory inclusion of different character types, and prohibitions against common passwords. The aim is to mitigate the risk of password compromise, thereby securing user accounts against unauthorized access [25].
- **Evolution and Rationale:** Initially simple, PCPs have evolved to counteract sophisticated cybersecurity threats, incorporating complex requirements for character variety and password unpredictability to better defend against attacks [3].

### 2.2.2 Effective Implementation of PCPs

Implementing PCPs involves balancing security with user experience to encourage compliance without compromising usability.

## User Experience Considerations

- **The Balance Between Security and Usability:** Effective PCPs should secure user accounts without causing undue frustration or encouraging insecure workarounds, such as password reuse, which undermines online safety [7].

## Technological Support for Compliance

- **The Role of Password Managers:** PWMs support adherence to PCPs by generating and managing complex passwords, thus alleviating the burden of remembering them and fostering better security practices [12].

### 2.2.3 Challenges and Innovations

Addressing the challenges in PCP design requires a thoughtful approach that considers user behavior and the latest cybersecurity research.

#### Balancing Complexity and Memorability

- **Design Challenges:** Creating PCPs that encourage strong, memorable passwords while avoiding patterns that attackers can exploit is a significant challenge. Excessive complexity can lead to less secure user behavior [3].

#### Adaptive Policies for Enhanced Security

- **Innovative Approaches:** Emerging trends in PCP design advocate for adaptive, user-friendly policies. For instance, initiatives like the PCP Description Language propose frameworks that facilitate easy implementation of diverse and effective PCPs, aiming to improve compliance and security [6].

As observed, PCPs are critical in securing online accounts by mandating robust password criteria. However, the effectiveness of these policies closely relates to their design and implementation, which must prioritize security and usability. Future

developments in PCP design are poised to address current limitations, offering more sophisticated and user-centric approaches to password security.

## **2.3 Related Works**

This section reviews pertinent literature that informs the broader context of our research, mainly focusing on the challenges and developments in password management practices. Through this exploration, we aim to highlight the critical interplay between usability and security that underpins this thesis’s focus on advancing password management systems.

### **User-Centric Challenges and Password Manager Usability**

A recurring theme in password security literature is the tension between stringent security requirements and the practical abilities of users to comply with these demands. Chaudhary et al. delve into this conflict, illustrating how the complexity inherent in password practices undermines security and user trust [2]. Similarly, Grobler et al. criticize traditional password generation methods for overlooking user needs, advocating for a cybersecurity approach that includes various user demographics [7]. Kaur and Mustafa reinforce this perspective, arguing for authentication systems that accommodate the user, highlighting the gap between technical policies and user accessibility [9].

### **The Dichotomy of Security and Convenience**

The literature consistently highlights a fundamental dichotomy in password management: the balance between security and convenience. Taneski et al. address the widespread issue of weak passwords and risky password behaviors, attributing these trends to a lack of user-centric design in security measures [25]. The studies by Stobert and Biddle and Melicher et al. further elaborate on user strategies like password reuse

and simplification, prioritizing convenience at the expense of security, particularly on mobile devices where these challenges are exacerbated [24, 14].

## **Innovations and Shortcomings in Password Management Tools**

Exploring solutions within the domain, Maqbali and Mitchell evaluate the effectiveness of various password generators, proposing AutoPass as a synthesis of security and usability [13]. The study by Shay et al. on more substantial password requirements reveals users' tendencies to modify existing passwords, underlining the necessity for more intuitive password creation aids [22]. Oesch and Ruoti's critical analysis of password managers pinpoints significant vulnerabilities in their design and functionality, suggesting targeted improvements to bolster their security and usability [16].

## **Password Behavior Insights and Manager Effectiveness**

Empirical studies by Florencio and Herley and Dell'Amico et al. offer insights into password habits, noting the compromise users make between password strength and memorability and the prevalent issue of password reuse [4, 3]. Pearman et al. research into password reuse across different web categories provides a nuanced understanding of user strategies, emphasizing the need for security measures that reflect user practices [19].

## **Emerging Research and Future Directions**

Recent investigations, such as those by Lyastani et al. and Pearman et al., emphasize the positive influence of password managers on security, albeit moderated by user engagement and specific tool functionalities [12, 20]. Huaman et al. and Oesch call attention to the interaction problems between password managers and websites, advocating for standardized practices to enhance security and user experience [8, 18].

Exploring these themes underscores the significance of our research focus on improving password management systems. By addressing the challenges highlighted in the literature and leveraging the advancements in password manager functionalities, our work aims to contribute meaningful insights and solutions that balance the critical aspects of usability and security in password management's password generation.



# Chapter 3

## System Design

To address the challenge of balancing password security and user experience, we developed a password generation system with the user at the center of the design. The system uses the Django web framework, providing seamless templating and server capabilities. The system's front end is developed using HTML, JavaScript, and CSS, with Bootstrap used to ensure a responsive design that works on various devices. The system underwent an iterative design process that included feedback from both technical and non-technical users, resulting in a visually appealing and functionally robust system.

### 3.1 Architectural Overview

The password generation system is built around a central server and implemented using Django. The server processes user inputs according to a password generation algorithm and delivers those generated passwords through a user interface. We chose Django for its robustness, scalability, and ease of integrating complex functionalities with a templating system that allows for the rapid development of user interfaces.

## **3.2 User Interface Design**

We developed the user interface (UI) as a critical component of the system design, emphasizing usability, accessibility, and password security. We sought feedback from acquaintances of various ages and technical backgrounds to refine the design through multiple iterations.

### **3.2.1 Baseline User Interface**

The Baseline UI is the foundational interface designed to replicate the essential functionalities of existing password generators, specifically Bitwarden, while seamlessly integrating with the system’s overall design language. This interface provided a benchmark for usability and functionality, setting the stage for developing more advanced, user-centric interfaces.

#### **Design and Functionality**

The Baseline UI design is minimalist, focusing on the core functionalities of password generation without user-specific customization options. It features a simple form for specifying password requirements like length, character sets, and additional settings, mirroring conventional password generators while adopting visual and interactive elements consistent with the rest of the system.



#### **Limitations of Baseline UI and Advancements through User-Centric Design**

As shown in Figure 3.1, the Baseline UI, while functional, presents a set of challenges for the average user. It requires users to make decisions on technical settings that directly impact the security of their password without necessarily understanding the security implications. For instance, selecting character sets, determining password length, and including/excluding symbols or numbers are technical decisions that can be daunting for non-technical users.

# Password Generator


Create your new password easily — just choose your preferences below!

w67F6XT9775aN!wn\$ho4QLHr#i4  
3R&rhE\*ZgJxb9gzz\*z6Ybbi7xWR9  
XqZp!Miuw



### OPTIONS

Password Type:  Password  Passphrase

Length  

A-Z

a-z

0-9

!@#\$\$%^&\*

Minimum numbers

Minimum special

Avoid ambiguous characters

Figure 3.1: The Baseline User Interface.

The User-Centric UI addresses these shortcomings by abstracting the technical complexities and presenting the user with intuitive questions. This approach simplifies the user experience and ensures that the generated passwords meet security standards appropriate for the user’s needs.

### 3.2.2 Advanced User-Centric Interfaces

Building on the Baseline UI, we developed an advanced interface to incorporate user-specific preferences and requirements into the password generation process, significantly enhancing the customization and relevance of generated passwords.

#### User-Centered UI + Password/Passphrase Generation



This interface, as shown in Figure 3.2, extends the baseline model by introducing options for users to specify their sensitivity level towards the generated password, the devices on which the user will use the password, and whether the password needs to be memorable or if the user will rely on a password manager. This advanced UI hides the technical variables from the baseline UI in the "more options" tab, which, when expanded, can be seen in Figure 3.3. This design approach facilitates the generation of passwords that are not only secure but also tailored to the user’s specific context and preferences.

**Translating User Inputs to Technical Settings** The system translates user input into technical settings, ensuring a secure password without requiring the user to understand the complexity.


- **Sensitivity Selection:** The user’s choice of sensitivity translates to a password strength value that the system uses to determine the complexity of the generated password. These values— $10^6$  for 'Not very' to survive online attacks,  $10^{14}$  for 'Moderately' to survive offline attacks, and  $10^{24}$  for 'Highly'—are derived from security standards that consider both offline and online attack scenarios [4, 5].

# Password Generator

Create your new password easily — just choose your preferences below!

**1R<4+d6@**  

Password Type:  Password  Passphrase

How sensitive is this password? ⓘ  
  
Not very                      Moderately                      Highly

Which devices will you be using this password on? ⓘ

Laptop/PC       Phone/Tablet

TV       Gaming System

Other

Do you need to memorize this password? ⓘ

Yes  
 No

Will you have access to your password manager? ⓘ

Yes  
 No

More Options ▼

Figure 3.2: The User-Centric Standard User Interface.

More Options ▲

Length 8

Uppercase Letters (A-Z)

Lowercase Letters (a-z)

Numbers (0-9)

Symbols (!@#\$%^&\*)

Minimum Numbers 1

Minimum Special Characters 1

Avoid Ambiguous Characters

**Figure 3.3:** The User-Centric User Interface's More Options.

These settings shown in Table 3.1 are adjustable in the system’s code to adapt to evolving security landscapes.

- **Device Consideration:** Our selection of device-specific settings significantly shapes the algorithm’s password entry approach, optimizing usability and security. For example, on PCs and mobile devices with standard Qwerty keyboards, the algorithm allows using uppercase, lowercase, numbers, and symbols to maximize password strength. However, we avoid ambiguous characters on mobile devices to minimize input errors and enhance typing speed, given mobile keyboards’ smaller size and higher error likelihood. Conversely, for TVs, where remote controls offer limited agility for complex inputs, the algorithm restricts passwords to lowercase letters and avoids ambiguous characters. This simplification reduces the cognitive load and input time, essential for devices not supporting rapid typing. Similarly, the password settings for gaming systems and other devices, such as smart watches, ATMs, and IoT devices, incorporate lowercase letters and numbers while excluding ambiguous characters. This configuration considers the limited input capabilities of these devices, striving to balance security with practicality in settings where typing is often challenging [10]. The specific settings for each device type, outlined in Table 3.2, are informed by extensive research on the usability of different input methods across various platforms. This strategy ensures that our password generation is secure and user-friendly, accommodating the distinctive features of each device type.
- **Memorability and Password Manager Access:** When a user requires a memorable password or lacks access to a password manager, the system adapts the password generation settings accordingly as seen in Table [? ], such as excluding numbers and symbols and avoiding ambiguous characters, to create a password that is easier to remember and input.

**Table 3.1:** Sensitivity to Password Security Level Mapping.

Sensitivity Level	Required Password Security Level
Not very	$10^6$
Moderately	$10^{14}$
Highly	$10^{24}$

**Table 3.2:** Device to Password Setting Mapping.

User Setting	PC / Laptop	Phone / Tablet	TV	Gaming System / Other
Uppercase (A-Z)	Yes	Yes	No	No
Lowercase (a-z)	Yes	Yes	Yes	Yes
Numbers (0-9)	Yes	Yes	No	Yes
Symbols (!@#\$\$%^&*)	Yes	Yes	No	No
Minimum Numbers	1	1	0	1
Minimum Symbols	1	1	0	0
Avoid Ambiguous Characters	No	Yes	Yes	Yes



**Integration with Password Composition Policies (PCPs)** This interface uniquely integrates user-defined PCPs with those required by websites, employing a sophisticated algorithm to generate a combined PCP that aligns with both requirements. This capability demonstrates the system’s flexibility in adapting to varied security standards and user preferences.

### **3.2.3 System Summary**

The development of this password generation system underscores the importance of a user-centered approach in designing security tools. We refined the system through an iterative process incorporating feedback from diverse users to meet technical security standards and address user needs and preferences. By integrating advanced user interfaces with the flexibility to adapt to various password composition policies, the system is committed to enhancing security and usability.

Furthermore, the system’s design, from its architectural foundations to the detailed considerations in the user interface, showcases the potential for technical solutions to bridge the gap between complex security requirements and the practical realities of everyday use. The subsequent section on the algorithm design chapter will delve deeper into the technical underpinnings that enable this balance, revealing the intricate backend functions that allow the system to translate user inputs into secure, customized passwords.

**Table 3.3:** Memorability and Password Manager Access to Technical Setting Mapping.

<b>Memorability or Password Manager Access</b>	<b>Technical Settings</b>
Memorable No Password Manager Access	No Symbols No Numbers Avoid Ambiguous Characters
Not Memorable Yes Password Manager Access	Device Settings

# Chapter 4

## Algorithm Design

After providing a detailed explanation of the system architecture and user interfaces, this section delves into the essential computational mechanisms that drive the system, notably the password generation algorithm. This algorithm is a user-friendly bridge between user inputs and creating secure and personalized PCP-compliant passwords. The algorithm comprises four fundamental components, each tailored to ensure the reliability and functionality of password generation:

1. **User Input Translation:** This component converts user inputs into precise technical settings that the algorithm can interpret. It involves analyzing the user's preferences regarding password sensitivity, device usage, memorability, and PWM access to establish a set of password requirements that are both secure and tailored to the user's needs. This translation is crucial for ensuring that the passwords generated align with the user's expectations and specific use cases.
2. **PCP Construction:** Based on the technical settings derived from user inputs, this step involves constructing a Password Composition Policy (PCP) that outlines the specific requirements for the password. These requirements include character types, length, and other constraints to ensure the password's strength

and security. The constructed PCP is a blueprint that guides the subsequent password generation process.

3. **Merging PCPs:** Often, there is a need to reconcile the PCP derived from user preferences with another PCP that a website or an application may impose. This component merges multiple PCPs to produce a unified set of rules that accommodate the most stringent aspects of each. It ensures the final password meets higher security standards and complies with all requirements.
4. **Password Generation:** The final component is the algorithm's core, which generates a password according to the rules established by the merged PCP. This process involves selecting appropriate characters, ensuring diversity, and avoiding prohibited patterns to create a password that is not only secure but also as unique as possible. This step is critical for producing a password that effectively safeguards user accounts against unauthorized access.

## 4.1 User Input Translation

User preferences are translated into algorithm parameters by a detailed process that maps user-friendly input selections to technical settings, ultimately defining the password requirements. This process is foundational to creating a Password Composition Policy (PCP) that aligns with the user's security needs and respects the contextual requirements necessitated by the required password sensitivity, devices of password use indicated, desire for password memorability, and password manager access. Below is an expanded explanation of each step involved in this translation process:

1. **Sensitivity Level:** The sensitivity level is a user-defined setting that indicates the importance or sensitivity of the account or service for which the user will use the password. This setting could range from low sensitivity (e.g., a forum

account) to high sensitivity (e.g., online banking). As mentioned, the algorithm adjusts the complexity and strength of the generated password accordingly.

2. **Device Selection:** Users can specify the devices they plan to use the password, such as smartphones, laptops, TVs, gaming systems, or others. This selection influences the password’s usability criteria, considering the ease or difficulty of entering specific passwords on different devices.
3. **Memorability:** Some users prefer easier-to-remember passwords, especially if they do not use a password manager. This preference affects the balance between password complexity and memorability, so the algorithm adjusts the password generation settings accordingly to avoid ambiguous characters, numbers, or symbols.
4. **Password Manager Access:** Users with access to password managers might opt for more complex passwords since the manager mitigates the need to remember the password. The algorithm considers this factor in determining whether the password should prioritize complexity or memorability.

We systematically translate these inputs into a set of technical settings, such as minimum length, character set requirements, and other constraints that form the basis of the PCP. The following pseudocode illustrates the algorithmic procedure for this translation, turning abstract user preferences into concrete technical specifications for password generation. The pseudocode in Algorithm 1 outlines the steps in translating user inputs into technical settings, preparing the groundwork for constructing a PCP that aligns with the user’s preferences and security needs.

## 4.2 PCP Construction

Constructing a Password Composition Policy (PCP) is a critical step that follows the translation of user inputs into technical settings. The PCP acts like a blueprint

outlining all the requirements a generated password must meet to align with user preferences and system security standards. These requirements include but are not limited to minimum and maximum length, maximum consecutive, allowed character sets with optional requirements, prohibited substrings, and any other constraints that ensure the strength and efficacy of the password.

Creating a PCP involves a detailed analysis and synthesis of the technical settings derived from the user input translation phase. This process ensures that the resulting passwords are secure, compliant with the necessary standards, and practical for user needs and use cases. Below is a detailed breakdown of the steps involved in constructing a PCP:

1. **Defining Character Sets:** The algorithm first identifies the character sets used in password generation based on user settings. Character sets include determining whether uppercase and lowercase letters, numbers, and symbols will be part of the password.
2. **Specifying Prohibited Substrings:** To enhance security and avoid common vulnerabilities, particular substrings may be explicitly prohibited in the generated passwords. This step involves listing those substrings based on user preferences such as device selection or memorability desire.
3. **Determining Required Character Sets:** Based on the selected character sets and minimum requirements, the algorithm specifies which sets must be represented in the password to ensure a diverse and robust character composition.
4. **Establishing Character Set Requirements:** Further refining the character set usage involves setting minimum counts for characters from specific charsets, ensuring that the password achieves the desired complexity and security level.

These steps culminate in the formulation of a PCP that governs the password generation process, ensuring that the generated passwords are both secure and aligned

with user expectations. The following pseudocode segments shown by Algorithms 2, 3, and 4 detail the algorithmic procedures for each of these critical steps in the PCP construction process.

## 4.3 Merging PCPs

The necessity of merging Password Composition Policies (PCPs) arises when user preferences and website-specific security requirements must coexist harmoniously within a single password policy. This process is critical for generating passwords that are not only secure but also compliant with both user expectations and external constraints imposed by online services. The merging algorithm inputs two PCPs: one reflecting the user’s preferences and the other encapsulating the website’s requirements. The goal is to synthesize a comprehensive policy that satisfies the more stringent aspects of both inputs.

In developing the algorithm for merging Password Composition Policies (PCPs), our primary objective was to ensure that the final merged policy adhered strictly to the most restrictive requirements from both user and website-specific PCPs. This meticulous approach guarantees that the resultant passwords are secure and comply with all constraints. Here, we detail the methodical process of merging each component of two PCPs into a single, more restrictive policy.

### 4.3.1 Merging Length Requirements

The first step involves merging the minimum and maximum length requirements from each PCP. We choose the maximum of the two minimum lengths to ensure the password is sufficiently long to meet the higher security standards of both policies. Conversely, we take the minimum of the two maximum lengths or set it to undefined if either is undefined to restrict the password length within a manageable and secure

---

**Algorithm 1** Translate User Inputs to Technical Settings

---

```
1: procedure TRANSLATEINPUTS
2:   sensitivity ← GETVALUEFROMDOM("sensitivity")
3:   deviceSelections ← GETCHECKEDVALUES("device-options")
4:   memorability ← GETCHECKEDVALUE("memorability")
5:   pwmAccess ← GETCHECKEDVALUE("PWM")
6:   settings ←
       DETERMINESETTINGS(deviceSelections, memorability, pwmAccess)
7:   return settings
8: end procedure

9: function DETERMINESETTINGS(deviceSelections, memorability, pwmAccess)
10:  Initialize settings with default values
11:  for each device in deviceSelections do
12:    Adjust settings based on device type
13:  end for
14:  if memorability is "yes" or pwmAccess is "no" then
15:    Adjust settings for memorability
16:  end if
17:  return settings
18: end function
```

---

---

**Algorithm 2** Construct PCP from Settings

---

```
1: procedure CONSTRUCTPCP(settings)
2:   charsets ← DEFINECHARSETS
3:   prohibitedSubstrings ← DEFINEPROHIBITEDSUBSTRINGS(settings)
4:   requiredCharsets ← DETERMINEREQUIREDCHARSETS(settings)
5:   charsetRequirements ← DETERMINECHARSETREQUIREMENTS(settings)
6:   PCPRule ← new PCPRule with settings.length, requiredCharsets, and
       charsetRequirements
7:   PCP ← new PCP with PCPRule and charsets
8:   return PCP
9: end procedure
```

---



---

**Algorithm 3** Determine what are the required subsets

---

```
1: function DETERMINEREQUIREDCHARSETS(settings)
2:   Initialize an empty set requiredCharsets
3:   if settings include uppercase then
4:     ADD(requiredCharsets, "uppercase")
5:   end if
6:   if settings include lowercase then
7:     ADD(requiredCharsets, "lowercase")
8:   end if
9:   if settings include numbers then
10:    ADD(requiredCharsets, "numbers")
11:  end if
12:  if settings include symbols then
13:    ADD(requiredCharsets, "symbols")
14:  end if
15:  return requiredCharsets
16: end function
```

---

---

**Algorithm 4** Determine what are the charset requirements

---

```
1: function DETERMINECHARSETREQUIREMENTS(settings)
2:   Initialize an empty map charsetRequirements
3:   if settings.minNumbers > 0 then
4:     charsetRequirements["numbers"] ← settings.minNumbers
5:   end if
6:   if settings.minSpecial > 0 then
7:     charsetRequirements["symbols"] ← settings.minSpecial
8:   end if
9:   return charsetRequirements
10: end function
```

---

range. This approach prevents overly lengthy passwords that might be impractical or less secure due to potential vulnerabilities in handling or storage.

### **4.3.2 Merging Maximum Consecutive Character Limits**

Similarly, we select the minimum of the limits provided by the two PCPs for maximum consecutive character limits. This restriction is crucial as it reduces the risk of easily guessable passwords due to repeated characters. If either PCP does not define this limit, we consider it undefined, which implies no specific restriction from that policy.

### **4.3.3 Combining Prohibited Substrings**

Merging the sets of prohibited substrings is straightforward: we unite the lists from both PCPs. This union ensures that the generated passwords avoid any sequence deemed insecure or undesirable by either policy, thus enhancing security against common attack patterns.

### **4.3.4 Incorporating Required Character Sets**

The required character sets from both PCPs are also combined. This inclusion ensures that the password includes a diverse range of characters, increasing its resistance to brute-force attacks. The merged PCP must not inadvertently exclude any character sets one policy might require for security purposes.

### **4.3.5 Merging Subset Requirements**

Merging subset requirements is a complex operation that requires careful consideration of the overlapping and unique character sets between two `PCPSubsetRequirements`, denoted as `subsetA` and `subsetB`. If both subsets are undefined, the result is naturally undefined, as there are no requirements to merge. If only one subset is defined, that subset is returned as the merged result since there is no conflict.

When both subsets are defined, we identify the familiar character sets that both subsets specify and note the unique character sets from each subset. This differentiation helps in understanding the diversity and commonality of requirements:

- Common character sets are those that appear in both subsets. These sets form the basis of the merged requirement, indicating characters we must include due to their presence in both original subsets.
- Unique character sets from either subset indicate characters that one policy deems necessary but the other does not explicitly require.

The merging process then involves combining these character sets into a more comprehensive character requirement. If there are standard character sets, the merged requirement count starts with the higher count from the two subsets, ensuring the more stringent requirement. Additional counts for diversity are calculated based on unique character options, ensuring the password is diverse without exceeding the total number of available character options. The final count is the sum of these values but capped at the number of unique and standard options to avoid exceeding the logical limit of character diversity.

### 4.3.6 Merging Character Set Requirements

The merging of character set requirements focuses on combining the settings from two rules to enforce the most restrictive conditions from each. For each character set that appears in either of the original PCPs, we perform the following operations:

- **Minimum Required Characters:** We determine the maximum required characters from both subsets. Doing so ensures that the merged policy adheres to the stricter requirements, guaranteeing a higher security level.
- **Maximum Allowed Characters:** We take the minimum of the maximum allowed characters unless both are undefined. This operation ensures that the

merged policy does not allow more characters than the most restrictive limit set by either policy.

- **Maximum Consecutive Characters:** Similarly, we take the minimum of the maximum consecutive characters allowed, treating undefined values as infinite. Doing so limits sequences of repeated characters, enhancing the password’s resistance to certain types of brute-force attacks.
- **Required and Prohibited Locations:** Both required and prohibited locations for each character set are combined. This combination ensures that all specific location-based restrictions from both policies are preserved in the merged policy, maintaining all designated security measures.

These operations collectively ensure that the merged character set requirements are comprehensive and strictly enforce the most restrictive elements from both input policies. This meticulous approach to merging character set requirements is critical for crafting a final PCP that is robust, secure, and adherent to user preferences and necessary security standards.

Through this process, the algorithm meticulously constructs a PCP that respects the user’s usability preferences without compromising the website’s security protocols. The following pseudocode segments detail the step-by-step procedure used to merge two PCPs into a singular, unified policy that is comprehensive and restrictive. The pseudocode segments are outlined in Algorithms 5, 6, 7, and 8.

## 4.4 Password Generation

Generating a compliant password following a carefully constructed PCP is a critical phase where theoretical security measures come into practice. This process is not merely about random character assembly; it involves an algorithm that respects the predefined rules of the PCP while ensuring the final password is secure and usable. This section describes the systematic approach of the password generation algorithm,

---

**Algorithm 5** Merge Two PCP Rules into a More Restrictive PCP Rule

---

```
1: function MERGEPCPRULES(ruleA, ruleB)
2:   minLength  $\leftarrow$ 
      MAX(ruleA.minLength, ruleB.minLength)
3:   maxLength  $\leftarrow$ 
      MINORUNDEFINED(ruleA.maxLength, ruleB.maxLength)
4:   maxConsecutive  $\leftarrow$ 
      MINORUNDEFINED(ruleA.maxConsecutive, ruleB.maxConsecutive)
5:   prohibitedSubstrings  $\leftarrow$ 
      ruleA.prohibitedSubstrings  $\cup$  ruleB.prohibitedSubstrings
6:   requiredCharsets  $\leftarrow$ 
      ruleA.requiredCharsets  $\cup$  ruleB.requiredCharsets
7:   reqSubset  $\leftarrow$ 
      MERGEREQUIRESUBSET(ruleA.requireSubset, ruleB.requireSubset)
8:   charReqs  $\leftarrow$ 
      MERGECHARSETREQUIREMENTS(ruleA.charsetRequirements,
      ruleB.charsetRequirements)
9:   return
      NEWPCPRULE(minLength, maxLength, maxConsecutive,
      prohibitedSubstrings, requiredCharsets, reqSubset, charReqs)
10: end function
```

---

---

**Algorithm 6** Finding the minimum value unless both are undefined

---

```
1: function MINORUNDEFINED(a, b)
2:   if a = undefined  $\wedge$  b = undefined then
3:     return undefined
4:   else
5:     return MIN(a, b)
6:   end if
7: end function
```

---

---

**Algorithm 7** Merge Two Require Subset Requirements into a More Restrictive Require Subset Requirement

---

```

1: function MERGEREQUIRESUBSET(subsetA, subsetB)
2:   commonOptions  $\leftarrow$ 
      INTERSECTION(subsetA.options, subsetB.options)
3:   uniqueOptions  $\leftarrow$ 
      DIFFERENCE(subsetA.options, subsetB.options)  $\cup$ 
      DIFFERENCE(subsetB.options, subsetA.options)
4:   mergedCount  $\leftarrow$ 
      CALCULATEMERGEDCOUNT(subsetA.count, subsetB.count,
      commonOptions)
5:   return
      NEWSUBSETREQUIREMENT(mergedCount, commonOptions  $\cup$ 
      uniqueOptions)
6: end function

```

---



---

**Algorithm 8** Merge Two Charset Requirements into a More Restrictive Charset Requirement

---

```

1: function MERGECHARSETREQUIREMENTS(reqsA, reqsB)
2:   Initialize mergedReqs as an empty collection
3:   for each charset in reqsA  $\cup$  reqsB do
4:     minRequired  $\leftarrow$ 
      MAX(reqsA[charset].minRequired, reqsB[charset].minRequired)
5:     maxAllowed  $\leftarrow$ 
      MINORUNDEFINED(reqsA[charset].maxAllowed,
      reqsB[charset].maxAllowed)
6:     maxConsecutive  $\leftarrow$ 
      MINORUNDEFINED(reqsA[charset].maxConsecutive,
      reqsB[charset].maxConsecutive)
7:     requiredLocations  $\leftarrow$ 
      reqsA[charset].requiredLocations  $\cup$  reqsB[charset].requiredLocations
8:     prohibitedLocations  $\leftarrow$ 
      reqsA[charset].prohibitedLocations  $\cup$  reqsB[charset].prohibitedLocations
9:     mergedReqs[charset]  $\leftarrow$ 
      NEWCHARSETREQUIREMENT(minRequired, maxAllowed,
      maxConsecutive, requiredLocations, prohibitedLocations)
10:  end for
11:  return mergedReqs
12: end function

```

---

which comprises selecting a compliant rule from the PCP, determining the appropriate length for the password, and populating it with characters that meet the specified criteria.

The algorithm unfolds in several stages:

1. **Validation of the PCP:** Initially, the PCP undergoes a validation check to ensure that it contains viable rules for password generation. This step is crucial to avoid any attempts to generate passwords based on flawed or incomplete policies.
2. **Rule Selection:** Among the various rules outlined in the PCP, one is randomly selected. This rule dictates the core structure of the password, including minimum and maximum lengths, required character sets, and any specific sequences the password must avoid.
3. **Determining Password Length:** The algorithm then decides on the password's length. This decision is influenced by the selected rule's specifications, aiming to balance security with memorability and usability.
4. **Character Assignment:** With a length established, the algorithm fills the password with characters. We meticulously designed this step to ensure that each character position contributes to the password's overall security, adhering to the character set requirements and avoiding any prohibited sequences.
5. **Randomization and Finalization:** The character positions are shuffled to introduce unpredictability, after which we concatenate the array of characters into the final password string.

This structured approach ensures that the generated password not only complies with the technical stipulations of the PCP but also aligns with the user's preferences and the security standards of the relevant website. The pseudocode outlined in Algorithms 9, 10, and 11 provides a detailed algorithmic representation of this process.

---

**Algorithm 9** Generate Password Compliant with PCP

---

```
1: function GENERATEPASSWORD(pcp)
2:   VALIDATEPCP(pcp) ▷ Ensure PCP is valid
3:   randomRule ← SELECTRANDOMRULE(pcp.rules)
4:   password ← GENERATEPASSWORDFORRULE(randomRule, pcp)
5:   return password
6: end function
```

---

---

**Algorithm 10** Generate Password for a Selected Rule

---

```
1: function GENERATEPASSWORDFORRULE(rule, pcp)
2:   length ← rule.minLength
3:   while length ≤ rule.maxLength or length ≤
   MAX_PASSWORD_LENGTH do
4:     password ← GENERATEPASSWORDWITHLENGTH(length, rule, pcp)
5:     return password ▷ Assumes validation is part of password generation
6:   end while
7:   return "" ▷ Fallback return an empty string
8: end function
```

---

---

**Algorithm 11** Generate Password with Specified Length

---

```
1: function GENERATEPASSWORDWITHLENGTH(length, rule, pcp)
2:   Prepare charset mappings and fulfill charset requirements
3:   Shuffle mappings for randomness
4:   Convert mappings to a password string
5:   return password
6: end function
```

---



## 4.5 Validation

The algorithm is subjected to an extensive suite of test cases to validate the efficacy and correctness of these security measures. These tests serve a critical purpose: they verify that the algorithms have intended behaviors, that the merged PCP rule accurately represents both PCP rules, or that the generated passwords comply with the intricate requirements of the PCPs. This rigorous validation process allows for identifying and rectifying bugs and potential vulnerabilities, thus ensuring that the algorithm remains robust and reliable.

The validation process consisted of a total of 4 test suites and around 20 tests, which consisted of tests created by Gautam et al. to validate the PCP object and the validation function and the tests we created to validate generated merged PCPs and passwords [6]. All of these tests resulted in passes, which verified the functionality and intended behavior of the algorithms.

The pseudocode segments detailed in Algorithms 12 and 13 illustrate two crucial aspects of the algorithm’s security and validation mechanisms: the merging of PCPs to form a unified policy that satisfies user and website requirements and the validation tests that ensure each generated password is secure, compliant, and functional.

## 4.6 Algorithm Summary

This chapter presented an in-depth look at the computational backbone of the password generation system, showcasing an algorithm that adeptly bridges user preferences with the stringent requirements of Password Composition Policies (PCPs). We articulate the algorithm’s design through a multi-faceted approach, encompassing user input translation, PCP construction, merging of PCPs, final password generation process, and validation. Each component is pivotal in ensuring the generated passwords are secure, compliant with various PCPs, and tailored to the user’s specific needs and usage contexts.

A noteworthy aspect of the algorithm is its capacity to intelligently merge the derived user-defined preferences with mandatory website-specific requirements, crafting a unified PCP that guarantees security without compromising usability. Using pseudocode to detail the algorithmic steps provides clarity and insight into the operational mechanisms, enabling a clear understanding of how intuitive user inputs are transformed into technically precise PCPs to generate secure passwords.

Furthermore, the rigorous validation process, comprising several test suites, underscores the reliability and functionality of the algorithm. By successfully passing these tests, the algorithm demonstrates its capability to generate passwords that are both secure and aligned with predefined PCPs, ensuring adherence to the highest security standards.

As we transition to the subsequent chapters, we will discuss this algorithm's implications for current cybersecurity practices and future developments in password management. Exploration of potential enhancements and user studies will form the basis of the discussion of future work aimed at further refining and advancing the password generation system to meet the evolving complexity and usability needs of digital security.

---

**Algorithm 12** Merge PCP Test

---

```
1: function MERGEPCPTEST
2:   Initialize PCPRules ruleA and ruleB with specific constraints
3:   Merge ruleA and ruleB into mergedRule using MERGEPCPRULES(ruleA,
   ruleB)
4:   assert mergedRule's constraints meet expected outcomes
5: end function
```

---

---

**Algorithm 13** Password Generation Validation Test

---

```
1: function PASSWORDVALIDATIONTEST
2:   Initialize a PCP with specific constraints
3:   Generate a password using GENERATEPASSWORD(PCP)
4:   Validate the generated password against the PCP using
   CHECK_PASSWORD(password, PCP)
5:   assert the validation passes all specified tests
6: end function
```

---

# Chapter 5

## Discussion

This chapter explores insights from developing and examining a user-centric password generation system. It discusses the challenges encountered, the strategic solutions employed, and the implications of these experiences. Additionally, it outlines avenues for future research and enhancements to password management solutions.

### 5.1 Lessons Learned

The journey of designing, implementing, and validating a user-centric password generation system has resulted in many valuable lessons:

1. **User-Centric Design's Critical Role:** The project reaffirmed the indispensable role of user-centric design in developing cybersecurity tools. By centering the system around user needs and behaviors, it was possible to create a solution that strengthens security and significantly improves usability. This approach emphasizes that understanding and accommodating user preferences are paramount in encouraging secure practices, as a system's effectiveness and usability do not have to be mutually exclusive.
2. **Security and Usability Balance:** One of the most notable challenges encountered was distinguishing a delicate balance between generating solid and

secure passwords and ensuring their usability for diverse user groups. This balance is crucial for promoting secure password habits, as overly complex passwords may deter users, leading to insecure workarounds. This research highlighted the need for security mechanisms that users can realistically incorporate and adopt into their daily routines, thereby supporting a security culture.

3. **Iterative Feedback Importance:** Adopting an iterative design process, which included feedback from both technical and non-technical users, was instrumental in refining the system into one that is secure and usable. This iterative approach facilitated the identification of diverse user requirements and led to developing a more inclusive and accessible system. The feedback loops were especially vital in highlighting usability issues and security misconceptions, which were implemented and addressed to enhance the system's effectiveness.

## 5.2 Implications for Cybersecurity Practices

The findings from this project carry significant implications for the broader field of cybersecurity, particularly in the realms of password management and policies:

1. **Enhanced Security Awareness:** This project underscores the vital importance of integrating the end-user perspective into the design of security tools, proving the viability and effectiveness of a user-centric approach advocates for developers and designers to prioritize user experience as an integral facet of cybersecurity solutions. A tool that fails to resonate with users will likely see limited adoption, underscoring the necessity of aligning security mechanisms with user habits and preferences.
2. **Simplification of Password Policies:** One of the contributions of this work is the demonstration that we can implement password policies without overwhelming users with technical jargon or complex requirements. The

system facilitates compliance with robust password policies by abstracting the technicalities into user-friendly interfaces and questions, enhancing overall security without sacrificing usability.

3. **Harmonization of Website and User Requirements:** The project highlights the feasibility of creating password policies that cater to website security mandates and user preferences. This dual compatibility ensures that passwords generated are secure and practical for users, addressing a common pitfall where stringent website policies can lead to user frustration or the adoption of insecure practices.
4. **Compliance with Varied Password Policies:** A significant implication of this system is its ability to dynamically generate passwords compliant with a wide range of password policies. This flexibility is paramount in a digital landscape where users frequently navigate between services with distinctive security requirements. The system's capability to adapt to these varying needs while maintaining a user-centric design philosophy paves the way for more universally applicable password management solutions.

The insights gained from this project illuminate a path forward in cybersecurity, advocating for a more inclusive approach to password policy design and implementation. By focusing on user experience, simplifying compliance with security policies, and ensuring adaptability to both user and website needs, this research contributes to a more secure and user-friendly digital environment.

### 5.3 Future Work

The project's subsequent phases will extend its functionalities, refine algorithms, and rigorously test the system's utility through comprehensive user studies. This work aims to enhance the current system and explore innovative interfaces and integration

with existing passwords or item banks, mainly focusing on enhancing usability and memorability without compromising security.

### 5.3.1 Novel Interfaces

The evolution of our user-centric password generation system envisions expanding its capabilities to enhance current functionalities and introduce novel interfaces to further personalize the password generation process. Two promising directions for these advancements are the integration of the **Existing Password** and **Word Bank** interfaces. These concepts represent a significant leap toward blending user preferences with secure password-generation practices.

#### **Existing Password Interface Concept**



The **Existing Password** interface got its inspiration from the typical user inclination to leverage familiar passwords. This interface aims to allow users to input an existing passphrase, which the system will then "mangle" or transform into a new, secure version that retains some elements of the original for ease of memorability. This approach addresses the challenge of creating secure and easy passwords for the user to remember, offering a personalized touch to the password generation process. This interface is visualized in Figure 5.1, illustrating a potential design based on my User-Centric UI.

#### **Word Bank Interface Concept**

The **Word Bank** interface concept stemmed from allowing users to input significant dates, names, phrases, or any other items of personal importance. The system would then utilize these inputs to generate a secure password or passphrase incorporating these elements. This method aims to produce secure passwords that are meaningful and memorable, thereby enhancing the overall user experience and engagement with

# Password Generator

Create your new password easily — just choose your preferences below!

existingDgaJz,\_fKd  

How sensitive is this password? ⓘ

Not very Moderately Highly

Which devices will you be using this password on? ⓘ

Laptop/PC  Phone/Tablet  
 TV  Gaming System  
 Other

Do you need to memorize this password? ⓘ

Yes  
 No

Will you have access to your password manager? ⓘ

Yes  
 No

Have a password to start with? ⓘ

Yes  
 No

Existing Passphrase:

.....

More Options ▼

Figure 5.1: The User-Centric Mangling User Interface.



the system. This interface is visualized in Figures 5.2 and 5.3, illustrating a potential design based on my User-Centric UI.

Following the conceptualization of these interfaces, the subsequent phases of the project will focus on their development, integration, and rigorous evaluation:

- **Development and Algorithmic Integration:** Future work will involve the actual coding and algorithm development for transforming existing passwords and integrating user-provided item bank items into secure passwords, followed by comprehensive testing for security and user engagement.
- **Algorithmic Enhancements:** Efforts will also be concentrated on refining the algorithms for password transformations, ensuring they provide robust security measures while maintaining the personal relevance and memorability of the generated passwords.

### 5.3.2 Comprehensive User Study



A detailed user study will be pivotal in evaluating the system's effectiveness, mainly focusing on the newly developed interfaces:

1. **Study Design:** The study will gather diverse participants with varying demographics and technical expertise. Participants will engage with the system, utilizing the new interfaces to generate passwords under specified requirements. Real-life password application scenarios will help assess system performance.
2. **Usability Evaluation:** After interacting with the system, participants will complete the System Usability Scale (SUS) questionnaire to assess each interface's usability quantitatively. The SUS provides a reliable, universally recognized method for evaluating the usability of various systems and technologies. A 10-item questionnaire with five response options ranging from "Strongly disagree" to "Strongly agree" offers a quick and effective way to measure the perceived ease of use and user satisfaction. The SUS score, which ranges from

# Password Generator


Create your new password easily — just choose your preferences below!

8-10-  
2024aO9rwTH2!\$TK#zLCr\*!&UXU\$  
WfZOJPGUb#g&SuxXKh5iEOz@X!  
8UIZX



How sensitive is this password? ⓘ

Not very      Moderately      Highly



Which devices will you be using this password on? ⓘ

Laptop/PC       Phone/Tablet

TV       Gaming System

Other

Do you need to memorize this password? ⓘ

- Yes
- No

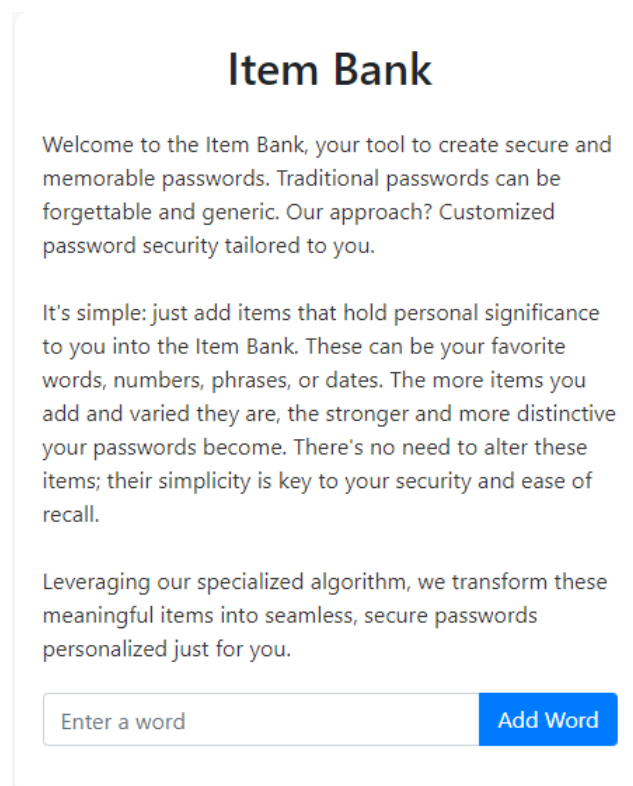
Will you have access to your password manager? ⓘ

- Yes
- No

Manage Item Bank

More Options ▼

Figure 5.2: The User-Centric Item Bank User Interface.



**Figure 5.3:** The User-Centric Item Bank User Interface's Item Bank.

0 to 100, serves as a global assessment of the subjective usability of the system, allowing for comparisons across different user interfaces and systems. This assessment aims to yield insights into the user experience, highlighting areas for improvement and verifying the system’s usability standards.

3. **Memory Test for Password Retention:** To assess the memorability of passwords, particularly those generated under memorability or no-password-manager-access conditions, a follow-up session may occur (e.g., after one week) to test if participants recall their passwords. This approach evaluates the practical memorability of passwords generated through the system.

### 5.3.3 Exploration of Passphrase Compatibility

Integrating Gautam et al.’s PCP Description Language into our user-centric UI laid the groundwork for a flexible and dynamic approach to password generation [6]. The next logical step in this evolution is to expand the system’s capabilities to support the generation of passphrases. Passphrases, typically longer than passwords and can include spaces and natural language phrases, offer distinct advantages regarding memorability and security, especially when designed to be random and complex [15]. The exploration into passphrase compatibility involves several key areas:

- **PCP Description Language Enhancement:** A critical aspect of supporting passphrases involves enhancing Gautam’s PCP Description Language to accommodate the unique characteristics of passphrases. This enhancement would enable the system to understand and apply security criteria specifically tailored to passphrase generation, such as minimum word count, inclusion of numbers or special characters within phrases, and avoidance of common phrase patterns that could be vulnerable to attacks.
- **Usability and Security Balancing:** Adapting the system to support passphrases will necessitate a careful balance between usability and security.

Passphrases are inherently more user-friendly, particularly for users who find traditional complex passwords challenging to remember. However, we must ensure that we randomly generate these passphrases and make them sufficiently complex to withstand brute force and dictionary attacks. This balance will involve algorithmic innovations that can generate secure and memorable passphrases.

- **User-Centric Design Principles:** Extending the system to include passphrase generation will further embody user-centric design principles by providing users with more options based on their preferences and security needs. This expansion could also include user interface adjustments to guide users in creating secure passphrases.
- **Integration with Existing Security Protocols:** Ensuring that generated passphrases are compatible with existing security protocols and requirements across various platforms will be vital. Compatibility with systems that may not traditionally support space characters or have length limitations could also impact passphrase efficacy.

Furthermore, research by Mukherjee et al. underlines the nuanced challenges associated with passphrase use [15]. While user-chosen passphrases often fail to meet security standards, machine-generated alternatives tend to sacrifice memorability. Mukherjee et al. introduce MASCARA. This system employs a constrained Markov generation process to produce passphrases that are both difficult to guess and easier to remember compared to those generated by current state-of-the-art systems [15]. This work is particularly relevant for exploring passphrase compatibility, as it highlights the potential for algorithmically generated passphrases to achieve a balance between security and usability that user-chosen passphrases have yet to find.

Their findings are pivotal for enhancing the PCP Description Language to support passphrases. By integrating principles similar to those of MASCARA, it is possible to

generate passphrases that adhere to security standards and are designed with human memory and recall in mind. This approach could dramatically improve the usability and security of passphrases, making them a more viable option for users, especially in scenarios where password managers are not accessible or preferred.

# Chapter 6

## Conclusion

This thesis embarked on a journey to redefine password generation through a user-centric lens, aiming to bridge the gap between the stringent security demands of digital systems and users' practical usability concerns. By developing a sophisticated password generation system and delving into the subtleties of Password Composition Policies (PCPs), this research offered a solution that accommodates both the technical constraints of online platforms and the everyday challenges users encounter. This chapter examines this work's key accomplishments, challenges, broader implications, and future directions.

The primary accomplishment of this thesis is designing and implementing a system that centers on user needs in the password generation process. Creating a Baseline User Interface and an Advanced User-Centric Interface showcased the feasibility of intuitive, user-friendly tools for generating secure passwords.

Another notable accomplishment is developing a novel method for integrating various Password Composition Policies (PCPs), effectively balancing user preferences with specific website password requirements. This approach significantly reconciles the often opposing demands of secure password creation and user convenience. By consolidating these diverse requirements into a unified policy, the system ensures that

the generated passwords meet the rigorous security standards of websites and resonate with users' preferences and habits.

Additionally, the thesis introduces a functionality capable of producing passwords that are compliant with these integrated PCPs. This feature is essential for the system's practical application, as it facilitates the automatic generation of passwords that satisfy a broad range of security criteria without user intervention. These developments highlight the thesis' contribution to enhancing the adaptability and user-friendliness of password management tools, significantly advancing digital security accessibility and management for users in various contexts.

In conclusion, this thesis advocates for a shift towards viewing password generation as a user-centered process, promoting a model where security and usability harmoniously coexist. As digital security threats evolve and become increasingly complex, the demand for innovative, accessible, and effective security tools will increase. This research paves the way for creating a safer and more user-friendly digital environment by prioritizing user needs and preferences.



# Bibliography

- [1] J. Bonneau, C. Herley, P. C. V. Oorschot, and F. Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *IEEE Symposium on Security and Privacy*, page 553–567, 2012.
- [2] S. Chaudhary, T. Schafeitel-Tähtinen, M. Helenius, and E. Berki. Usability, security and trust in password managers: A quest for user-centric properties and features. *Computer Science Review*, 33:69–90, 2019.
- [3] M. Dell’Amico, P. Michiardi, and Y. Roudier. Password strength: An empirical analysis. In *IEEE INFOCOM*, page 1–9, 2010.
- [4] D. Florencio and C. Herley. A large-scale study of web password habits. In *Proceedings of the 16th international conference on World Wide Web*, page 657–666, 2007.
- [5] D. Florêncio, C. Herley, and P. C. V. Oorschot. An administrator’s guide to internet password research. In *Large Installation System Administration Conference (LISA)*, page 44–61, 2014.
- [6] A. Gautam, S. Lalani, and S. Ruoti. Improving password generation through the design of a password composition policy description language. In *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*, pages 541–560, Boston, MA, 2022. USENIX Association.

- [7] M. Grobler, R. Gaire, and S. Nepal. User, usage and usability: Redefining human centric cyber security. *Front. Big Data*, 4, 2021.
- [8] N. Huaman, S. Amft, M. Oltrogge, Y. Acar, and S. Fahl. They would do better if they worked together: The case of interaction problems between password managers and websites. In *IEEE Symposium on Security and Privacy (SP)*, page 1626–1640, 2021.
- [9] A. A. Kaur and K. K. Mustafa. A critical appraisal on password based authentication. *International Journal of Computer Network and Information Security*, 11(1):47–61, 2019.
- [10] Y. Li, F. You, X. You, and M. Ji. Smartphone text input: Effects of experience and phrase complexity on user performance, physiological reaction, and perceived usability. *Applied Ergonomics*, 80:200–208, 2019.
- [11] Z. Li, W. He, D. Akhawe, and D. Song. The emperor’s new password manager: Security analysis of web-based password managers. In *USENIX Security Symposium*, page 465–479, 2014.
- [12] S. G. Lyastani, M. Schilling, S. Fahl, M. Backes, and S. Bugiel. Better managed than memorized? studying the impact of managers on password strength and reuse. In *USENIX Security Symposium*, page 203–220, 2018.
- [13] F. A. Maqbali and C. J. Mitchell. Autopass: An automatic password generator. In *Proceedings of the International Carnahan Conference on Security Technology (ICCST)*, pages 1–6, 2017.
- [14] W. Melicher, D. Kurilova, S. M. Segreti, P. Kalvani, R. Shay, B. Ur, L. Bauer, N. Christin, L. F. Cranor, and M. L. Mazurek. Usability and security of text passwords on mobile devices. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI ’16)*, pages 527–539, 2016.

- [15] A. Mukherjee, K. Murali, S. K. Jha, N. Ganguly, R. Chatterjee, and M. Mondal. Mascara: Systematically generating memorable and secure passphrases. In *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security*, ASIA CCS '23, page 524–538, New York, NY, USA, 2023. Association for Computing Machinery.
- [16] S. Oesch and S. Ruoti. That was then, this is now: A security evaluation of password generation, storage, and autofill in browser-based password managers. In *29th USENIX Security Symposium (USENIX Security 20)*, page 2165–2182, Aug 2020.
- [17] S. Oesch, S. Ruoti, J. Simmons, and A. Gautam. “it basically started using me:” an observational study of password manager usage. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery.
- [18] T. Oesch. *An Analysis of Modern Password Manager Security and Usage on Desktop and Mobile Devices*. PhD thesis, The University of Tennessee, 2021.
- [19] S. Pearman, J. Thomas, P. E. Naeini, H. Habib, L. Bauer, N. Christin, L. F. Cranor, S. Egelman, and A. Forget. Let’s go in for a closer look: Observing passwords in their natural habitat. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, page 295–310, 2017.
- [20] S. Pearman, S. A. Zhang, L. Bauer, N. Christin, and L. F. Cranor. Why people (don’t) use password managers effectively. In *Symposium On Usable Privacy and Security (SOUPS)*, page 319–338, 2019.
- [21] S. Riley. Password security: What users know and what they actually do. *Usability News*, 8(1):2833–2836, 2006.
- [22] R. Shay, S. Komanduri, P. G. Kelley, P. G. Leon, M. L. Mazurek, L. Bauer, N. Christin, and L. F. Cranor. Encountering stronger password requirements:

- User attitudes and behaviors. In *Proceedings of the Sixth Symposium on Usable Privacy and Security (SOUPS)*, pages 1–20, 2010.
- [23] J. Simmons, O. Diallo, S. Oesch, and S. Ruoti. Systematization of password manager use cases and design paradigms. In *Proceedings of the 37th Annual Computer Security Applications Conference, ACSAC '21*, page 528–540, New York, NY, USA, 2021. Association for Computing Machinery.
- [24] E. Stobert and R. Biddle. The password life cycle: User behaviour in managing passwords. In *Proceedings of the Symposium on Usable Privacy and Security (SOUPS)*, pages 9–11, Menlo Park, CA, 2014.
- [25] V. Taneski, M. Heričko, and B. Brumen. Systematic overview of password security problems. *Acta Polytechnica Hungarica*, 16(3):143, 2019.

# Vita

David Huang was born in Manhattan, New York, and has lived in Nashville, Tennessee, for most of his life. He is set to graduate with a Master of Science in Computer Science, focusing on cybersecurity, from the University of Tennessee Knoxville's Tickle College of Engineering in May 2024, following a Bachelor of Science in the same field, where he graduated with a 4.00 GPA and numerous honors including the Dean's List, Distinguished Tennessean, and the Volunteer of Distinction awards. Professionally, David has varied experiences, serving as a Graduate Teaching Assistant, managing business operations as an Assistant Manager at Speed Queen Wash & Dry, and honing his technical skills as a Junior Salesforce Administrator at Persist Nashville and an IT Student Assistant. His technical expertise spans programming languages such as C, C++, Java, and Python; frameworks like Django and React; and tools including Git and Visual Studio Code. An active leader in the academic community, David is Vice President and former Treasurer of the Association for Computing Machinery at UTK, a member of the Tau Beta Phi and Phi Kappa Phi honor societies, and was involved in the Heath Integrated Business and Engineering Program. With a passion for technology consultation, cybersecurity, and software engineering, David is enthusiastic about leveraging his dual strengths in technical and business realms to foster organizational growth and innovation. Upon completing his Master's degree, he plans to return to Nashville to continue contributing to its technological and business development.