Research and Innovation

Terms of Reference

Name of Student_1:	Ketumile	**ID Number_1:**	NS23-021
Name of Student_2:	Bonang	**ID Number_2:**	NS23-027
Name of Student_3:	Tshiamo	**ID Number_3:**	NS23-023
Name of Student_4:	Mompoloki	**ID Number_4:**	NS23-018
Name of Student_4:	Tumelo	**ID Number_4:**	NS23-014
Supervisor:	Thobo Maruatona		
Title	Password Authenticator tool		

Problem Statement and Background

1. In today's digital landscape, students frequently create and manage multiple online accounts for academic and personal use. However, many still rely on weak and easily guessable passwords, putting their personal information, school accounts, and digital resources at risk. Cybercriminals exploit weak passwords to gain unauthorized access, leading to potential data breaches, identity theft, and academic disruptions. According to JustFirewalls, weak passwords remain a major security risk, emphasizing the need for stronger authentication measures to protect online security (JustFirewalls, n.d.).

   Research by Verizon (2023) found that 81% of hacking-related breaches stem from weak or compromised passwords, highlighting the critical role of password security in preventing cyberattacks. Additionally, a study by the National Institute of Standards and Technology (NIST) emphasizes the importance of using long, complex passwords and avoiding predictable patterns to enhance security (Grassi et al., 2017).

	To address this issue within the school community, a password authenticator tool is being developed specifically for students. This tool will evaluate the strength of passwords in real-time using a robust algorithm that analyses key security parameters such as length, character diversity, and predictability. By ensuring that students create strong and secure passwords, this solution aims to safeguard their academic and personal accounts, thereby reducing the risk of cyber threats and unauthorized access.
Aim & Objectives/	**Aim:** The aim of this project is to develop a password authenticator application tool for Botswana Accountancy College students, in effort to assist with the creation of strong and secure passwords. This solution aims to help safeguard the students' academic and personal accounts, reducing the risk of cyber threats and unauthorized access. **Objectives:** 1. To investigate and analyze password security challenges among BAC students by collecting and examining data on common password weaknesses, security risks, and user behaviours to gain deeper insights into the issue. 2. To apply the Waterfall software development methodology in systematically designing and developing a password authenticator tool, ensuring structured and reliable implementation. 3. To develop a secure password authenticator tool that generates strong passwords, evaluates password strength in real time, and enforces security policies to enhance students' online safety. 4. To evaluate and test the effectiveness of the password authenticator tool through rigorous functionality, integration, and beta testing to ensure it meets security and usability requirements. 5. To produce comprehensive project documentation that details the research findings, system development process, testing results, and final implementation of the password authenticator tool.
Requirements	1. User Authorization: The application should require authentication to access, utilizing passcodes, passwords, or biometric verification methods. 2. Secure Password Generation: Passwords will be generated based on user criteria (length, character type etc) 3. Data Encryption: Locally stored passwords data will remain local to each device and will be stored using advanced encryption standards such as Twofish and AES. These encryption standards will ensure that in the event of a fail or breach to device USB debugging authorization sensitive data will remain secure. 4. Protection against cyber threats: The app should be able to detect and prevent malicious activities such as brute force attacking.

	5. Offline Authentication: The tool must function without an internet connection 6. Password Guidelines Display: The system should display visual feedback based on password requirement in the form of a password strength meter 7. Password Policy Enforcement: Students are required/forced to change their passwords after a set period 8. Password Reset/Recovery: A way for users to reset/recover the app's access passwords. 9. Incident Response: In case of system failure/attack, monitoring systems should alert a 10. Logging: Every attempt whether successful or not should be logged for security 11. Compliance: The tool should adhere to legal and regulatory standards 12. Performance: It should have fast response time and run efficiently on low end devices 13. Multi factor authentication: It should have support for more authentication options such as one time password 14. Disaster recovery plan: procedures to restore functionality in case of system failure. 15. Auto-Lock: After inactivity, the app must automatically lock itself after a user-defined period of inactivity 16. User Experience: Authentication process should be easy to use especially for those with disabilities by using top accessibility standards. 17. Role Based Access Control: Tool should be able to differentiate users to assign correct permissions 18. Edit Password Entries: Users must be able to edit existing password entries 19. Delete Password Entries: Users must be able to permanently delete stored passwords 20. Seach and Filter Passwords: The app must allow users to quickly search for and filter saved passwords by name or category
Resources and Skills Required	Resources: 1. Pycharm 2. Py QT Design 3. Visual Studio 4. A powerful computer 5. Crackstation.net 6. anaconda

	Skills: 1. Introduction to python programming. 2. CCNA: Introduction to networks. 3. CCNA: Switching, Routing & Wireless Essentials. 4. Computer Systems Installation and Maintenance.
Outcomes and Deliverables	1. A secure and functional password authenticator tool that helps BAC students generate strong passwords and evaluate their security. 2. Improved password security practices among students, reducing the risk of cyber threats and unauthorized access. 3. Real-time password strength analysis to guide users in creating secure passwords. 4. A fully functional multiplatform password authenticator application. 5. A detailed project report covering research findings, development process, security measures, and evaluation results. 6. An inbuilt tutorial system for guiding students on the features and functions of the password authenticator tool, to educate students on how it works and how to use it. 7. Test reports from functionality, integration, and beta testing phases. 8. Enhanced user experience with a simple and intuitive interface for easy navigation and password management. 9. Offline functionality that allows students to use the tool without an internet connection. 10. A project report that summarizes development process and challenges encountered. 11. Training manuals for IT staff 12. Well organized source code that could be used by BAC if they ever want to integrate it in their system
Methodology	Waterfall methodology, this is a sequential approach to software development introduced by Winston W. Royce in 1970. The choice for

this methodology on the development of the Password Authenticator Tool is based on reasons that; This model of approach is simple and idealistic, with each phase being completed after one another. Specific details about which authentication apps were developed using waterfall model aren't typically disclosed publicly, however it has been widely used in the past for various types of software projects. According to Aiden Gallagher, Jack Dunleavy, Peter Reeves stating the advantages and disadvantages of a waterfall model, the project team members do not require consistent communication unless specific integrations are required. Team members can also work independently and often required to provide status reports. (IBM Developer, 22 April 2019).

1. Requirements Gathering

   The development team conduct comprehensive requirements gathering by engaging with BAC college students. This involves collecting detailed feedback and understanding their needs and expectations for the authenticator app. Gathered requirements are thoroughly analysed to ensure a clear understanding of the desired functionalities and security features.

2. Design

   With the requirements clearly defined, the design phase commence. The team create detailed wireframes and prototypes to visualize the app's user interface. These designs serve as blueprints, ensuring that the app's layout and navigation would be intuitive and user-friendly. The design phase also includes planning the app's architecture and selecting appropriate technologies.

3. Development

   The development phase will begin with coding of the user interface, ensuring it aligns with the wireframes and prototypes. The team will integrate the app with necessary APIs to enable core functionalities. Following this, robust encryption and other security measures will be implemented to protect user data and ensure secure authentication processes. This phase involves iterative development and regular code reviews to maintain high-quality standards.

4. Testing

   Testing will be conducted by the development team at multiple levels to ensure the app met all requirements. Individual

	components undergo rigorous functionality testing to verify their performance. Integration testing ensures that all parts of the app work together seamlessly. The project team also will conduct beta testing with a small group of students to gather real-world feedback and identify any usability issues. This feedback is crucial for making necessary adjustments and improvements. 5. Deployment In the deployment phase, the development team finalize the app, addressing any remaining bugs and ensuring all configurations were correctly set. The app is then deployed to the production environment, making it accessible to students. The deployment process includes thorough checks to ensure the app's stability and performance in the live environment. 6. Maintenance Post-deployment, the development team commit to ongoing maintenance to keep the app secure and up to date. This will involve regular updating of security features to counteract new threats, such as attempts to decrypt the current encryption methods by hackers. Continuous monitoring and periodic updates will ensure the app remain reliable and secure for all users.
Limitations & Constraints	1. The development team should ensure that the app works seamlessly across the different operating systems platforms by using a programming language that would integrate different hardware and software. 2. The development team does not have access to third-party security APIs for embedding security features into the app. To overcome this limitation, the team will implement custom security measures, including AES-256 encryption for local password storage, PBKDF2 or Argon2 hashing for securing the master password, and manual integration of biometric authentication using device-native libraries. Additionally, secure local key storage will be implemented using obfuscation techniques to prevent unauthorized access. This approach ensures strong security without relying on external APIs. 3. App maintenance and updates will be dependent on the development team being able to keep up to standard with the latest encryption technologies to stay up to date and ensure

	user privacy.
Evaluation	1. **Password Strength Analysis**; the development team will establish a criteria for evaluating password strength, including length, complexity (use of uppercase, lowercase, special characters and numbers). Creating algorithms that assess passwords based on the defined security will analyse the password's structure and compare it against known patterns and vulnerabilities. 2. **Real-time feedback**; Project team in designing user interface, will design an interface that provides real-time feedback as students type their passwords indicating with colours the strength of the password and providing suggestions for improving password strength 3. **Integration Capability**; the project team will develop APIs that allow the password strength analysis tool to integrate with various platforms and applications. Extensive tests by the team will be conducted to ensure compatability with different operating systems and devices and ensuring that the integration process adheres to security best practices and complies with relevant regulations and standards
References	1. Grassi, P., Garcia, M., & Fenton, J. (2017). Digital identity guidelines: Authentication and lifecycle management. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-63b 2. Verizon (2023). 2023 Data Breach Investigations Report. Retrieved from https://www.verizon.com/business/resources/reports/dbir/ 3. JustFirewalls (n.d.). The dangers of weak passwords and how to avoid them. Retrieved from https://www.justfirewalls.com/the-psychology-of-passwords-why-we-choose-weak-ones-and-how-to-overcome-it/ 4. Aiden Gallagher, Jack Dunleavy, Peter Reeves IBM Developer (22 April 2019). The Waterfall Model: Advantages, disadvantages, and when you should use it The Waterfall Model: Advantages, disadvantages, and when you should use it - IBM Developer 5. Waterfall Model – Software engineering (18 Oct,2024). Waterfall Model - Software Engineering - GeeksforGeeks

PROJECT SCHEDULE

Student Names:	Mompoloki Majang, Tumelo Ogaufi Bimbo, Tshiamo Dikinya, Bonang Mahalelo, Ketumile Maitlhamako
Project Title	Password Authenticator tool
Supervisor	Thobo Maruatona

Project Task		Predicted (days) [excluding weekends]	Planned Start Date	Planned End Date	Specific Task Objectives or Deliverables	Actual (days)[excluding weekends]	Actual Start Date	Actual End Date	Task Outcome Evaluation/Consequences
No	Description								
1	Planning and research	10	03/02/2025	17/02/2025		15	03/02/2025	21/02/2025	
		1	03/02/2025	04/02/2025	1. Formation of project team.	1	03/02/2025	04/02/2025	Formed a team consisting of 5 members.
		1	03/02/2025	04/02/2025	2. Brainstorming project ideas	1	03/02/2025	04/02/2025	Selection of project idea (Password authenticator tool).

		2	04/02/2025	06/02/2025	3. Define requirements-identifying features needed by the system.	2	04/02/2025	06/02/2025	compiled requirements for the system
		1	6/02/2025	07/02/2025	4. Deciding on the programming languages and tools development team will be using	1	06/02/2025	07/02/2025	Gathered essential tools such as development IDE tools to be utilised by the development team.
		5	07/02/2025	14/02/2025	5. Developing the TOR.	9	10/02/2025	21/02/2025	The project's TOR document.
		2	14/02/2025	17/02/2025	6. Literature Review		27/02/2025	05/03/2025	
2	Design	10	17/02/2025	03/03/2025		7	07/03/2025	17/03/2025	

		8 2	17/02/2025 27/02/20	27/02/2025 03/03/2025	1. Create prototypes for our app's interface 2. Map out the user journey from start to finish	7 2	07/02/2025 10/03/2025	17/03/2025 12/03/2025	Different semi completed/completed prototypes for the application interface. A detailed document detailing the step-by-step sequence the user will have to follow to fully utilise the app.
3	Development	10 7 3	03/03/2025 03/03/2025 12/03/2025	17/03/2025 12/03/2025 17/03/2025	1. Start building user interface and integrating it with the API endpoints, e.g. biometric 2. Implement encryption and other security measures		17/03/2025 17/03/2025		The coding and development of the beta app, incorporating the chosen User Interface theme and various features into the beta app. Addition of the necessary privacy and security features into the beta app.
4	Testing	10	17/03/2025	31/03/2025					

		2 4 4	17/03/2025 19/03/2025 25/03/2025	19/03/2025 25/03/2025 31/03/2025	1. Test individual components for functionality 2. Ensure all parts of the app work together seamlessly 3. Conduct beta testing with small group of users to gather feedback				
5	Deployment	10 3 1	31/03/2025 31/03/2025 03/04/2025	14/04/2025 03/04/2025 04/04/2025	1. Finalize all features and fix any remaining bugs . 2. Launch of the application.				A final and polished version of the application. Following the beta testing. Launch of the application into various application hosting sites, such as the Google Play Store, Apple App store, Microsoft App store.