🎓

Client (Mohd Zairul Mazwan Jilani)

Overview (Transaction Authorization Code (TAC) for Internet Banking Login and Fund Transfer using Vigenère Cipher)

Transaction authorization code (TAC) is commonly used in banking for completing financial transactions over the internet securely. This method is known as two-factor authentication (2-FA). The use of TAC is also extended to other online processes such as login as part of a security measure. TACs are usually sent to the user's mobile phone or email, who performs transactions online. A TAC may consist of alphabets and numerical values of length 5 characters.

Objective →

☐ Develop a proof of concept of a web application that enables users to login to an internet banking platform and perform fund transfers to recipients securely.

- The system to be developed is an internet banking application for a consumer bank namely "MZMazwan Bank."

Business Requirements

The MZMazwan Bank would like to implement a security measure for the login process and fund transfer transactions for their internet banking system using TAC. All unique TACs will be generated by the system and be sent to the user's email address. The system will use the user's email address that is securely saved in the system databases.

Main Requirements →

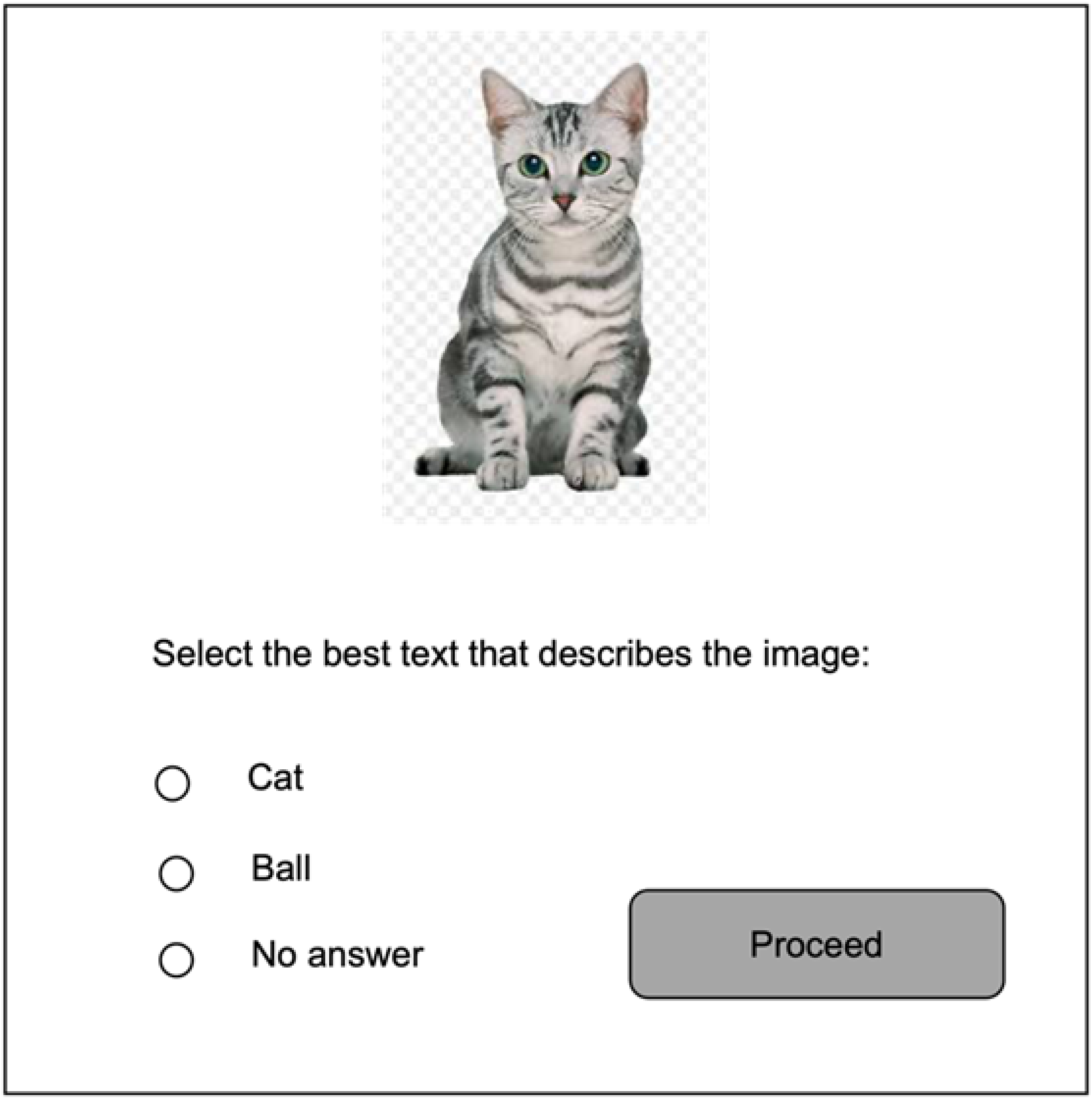
1. ***Login -*** The TAC process will be implemented to validate genuine logins to the system. However, not all logins will require a TAC. There are conditions:
    - Users who have not logged in to the system for a long interval time i.e., more than 30 days must be verified with a TAC.
    - Users who have logged in to the system frequently, more than twice within 10 minutes must be verified with a TAC.
2. ***Fund Transfer -*** All fund transfers of more than £1000 will require a TAC to complete the transactions. The implementation of TAC for this phase will only be applicable to internal fund transfers (sending money between the account holders of MZMazwan Bank).
3. ***TAC Format -*** A TAC will have five (5) characters of the following combinations:
    - Three (3) upper case alphabets. The first three characters.
    - Two (2) numeric values. The last two characters.

    Example -

    ABC56

    The code should be generated randomly by the system. The code will be encrypted (encoded) before sending it to the user's email. The encryption method is elaborated in the next section.
4. ***Encryption Method -*** All TACs will be encrypted using the Vigenère Cipher method before sending it to the user's email. The method algorithm is explained from this reference. Upon receiving the TAC (encoded) through email, the user will need to key in the TAC and click proceed. The system will then decode the encoded TAC using the same keyword so that it matches with the original value of the TAC.
5. ***Encryption Process -*** The Vigenère Cipher method needs a keyword to encode a TAC. A keyword should be provided by the system through an image (object) randomly. The user would need to select the text that best describes the object from the image

    Example →



Select the best text that describes the image:

○ Cat

○ Ball

○ No answer

From the example above, the system must validate the answer from the user. Only a valid answer will be processed. If the user selects “Ball” or “No answer”, an appropriate error message should be prompted (maximum 2 chances). If the user selects “Cat” (used as the keyword), then the following processes will be executed →

Upon clicking the proceed button (with a correct answer), the system will:

- □ generate a TAC
- □ encode the TAC using the keyword
- □ send the encoded TAC to the user’s email address.

☐ The user will be prompted to check their email which is registered with the bank for retrieving the TAC for the transaction.

☐ The user will need to input the TAC they receive and click proceed.

☐ The system will decode the TAC using the same keyword and validate it.

☐ The login/fund transfer should be flagged as successful upon process 6 (successful validation).

A TAC is only valid for a single transaction within 5 minutes. If the user inputs the TAC beyond this period, the TAC will be flagged as stale, and the user should redo the login/fund transfer process.

Incorrect input of a TAC will result in two attempts for the user to input the same TAC. Otherwise, the system will cancel the transaction/login and return to the user homepage. For login, return to the Bank’s landing page.

6. ***Technical Requirements -*** The system should be a proof of concept of a web application. A local host of a web application is acceptable for the project’s deliverable. However, using an email API for Google emails is required for TACs. It is not a requirement to use any web application frameworks. Using procedural programming of PHP or equivalents is sufficient to progress with this project.
    - A simple database design and some fabricated datasets are required for this project. The focal point of the project is not the database design but more on the algorithm and process of TAC. However, a good database design that supports the concept is sought from this project.
7. ***Other Generic Requirements -***

This project is an enhancement to the existing functionalities of the internet banking platform i.e., login and fund transfer. Therefore, the proof of concept for this project would necessitate having common internet banking features such as account holders, account balance, user status and many more. The requirements from section should be gathered from the client during the lecture/tutorial sessions.

8. ***Optional Requirement -***

Sending TACs to WhatsApp (business account) instead of email is favorable. However, this requirement is not mandatory for the proof-of-concept version.

Simplified Version→

MZMazwan Bank would like to implement a security measure for the login process and fund transfer transactions for their internet banking system using TAC.

- All unique TACs will be generated by the system and be sent to the user’s email address.
- The system will use the user’s email address that is securely saved in the system databases.

Functional Requirements (Features Of The System)→

□ The system must generate a unique TAC for each transaction.

□ The system should be able to send the TAC to the user's email address securely.

(Optional) -

□ Sending TACs to a user's WhatsApp (business account) instead of their email address is desirable but not essential for the proof-of-concept version.

Login -

□ The system must implement a login process that requires a TAC under the following conditions:

- Users who haven't logged in for more than 30 days.
- Users who have logged in more than twice within 10 minutes.

TAC Format -

□ The TAC should consist of 5 characters, 3 uppercase letters followed by 2 numerals (i.e. ABC56)

Fund Transfer -

□ The system must require a TAC for all fund transfers exceeding £1000.

Encryption Method -

- □ The system must implement a CAPTCHA challenge using an image to obtain a keyword for the Vigenère Cipher method. The CAPTCHA challenge should require the user to identify the correct object in the image.
  - □ The system must encrypt TACs before sending them to the user's email using the Vigenère Cipher method with the keyword obtained from the CAPTCHA challenge.
    - □ The system must decrypt the TAC received from the user using the same keyword used for encryption.
      - □ The system must validate the decrypted TAC to ensure it matches the original TAC.
      - □ The system must restrict the number of TAC input attempts to a maximum of three.
        - □ Upon exceeding the maximum number of incorrect TAC attempts, the system should terminate the transaction/login session and return the user to the homepage.
      - □ The system should flag a TAC as invalid if it's not used within 5 minutes.

Non-Functional Requirements (Behaviors Of The System)→

- □ The system should be a proof-of-concept web application.
- □ The system should use a local host for development purposes.
- □ The system should leverage an email API for sending TACs via email.
- □ The system should employ procedural programming language like PHP.
- □ The system should have a simple database design with fabricated data sets.

Domain Requirements (Regulations Of The System)→

- □ The system should comply with the existing functionalities of the MZMazwan Bank internet banking platform.
- □ The system should support common internet banking features like account holder information, account balance, and user status.

Resources / References

The Vigenère Cipher Encryption and Decryption →

https://pages.mtu.edu/~shene/NSF-4/Tutorial/VIG/Vig-Base.html

PHP script to send messages with WhatsApp Business API →

https://medium.com/@256cub/php-script-to-send-messages-with-whatsapp-business-api-a732d5206a0d

How to Send a Message by WhatsApp API using PHP easily →

https://blog.ultramsg.com/send-whatsapp-message-by-whatsapp-api-using-php/

Programmable Messaging for WhatsApp and PHP Quick start →

https://www.twilio.com/docs/whatsapp/quickstart/php