REUSABLE CAPTCHA SECURITY ENGINE

A Project Report Submitted in the
Partial Fulfillment of the Requirements
for the Award of the Degree of

BACHELOR OF TECHNOLOGY

IN

Information Technology

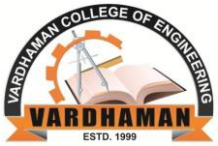
Submitted by

HARI BHARADWAJ 20881A1287
TOURIA TANAZZUM 20881A12B7
YAMSANI VAISHNAVI 20881A12C0

SUPERVISOR

Dr.Ganesh B Regulwar

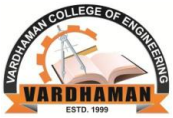
Associate professor



Department of Information Technology

VARDHAMAN COLLEGE OF ENGINEERING, HYDERABAD

An Autonomous Institute, Affiliated to JNTUH



VARDHAMAN COLLEGE OF ENGINEERING, HYDERABAD

An Autonomous Institute, Affiliated to JNTUH

Department of Information Technology

CERTIFICATE

This is to certify that the project titled **REUSABLE CAPTCHA SECURITY ENGINE** is carried out by **Hari Bharadwaj, Roll number: 20881A1287, Touria Tanazzum, Roll number: 20881A12B7 and Yamsani Vaishnavi, Roll number: 20881A12C0** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Information Technology** during the year 2022-23.

Signature of the Supervisor
Dr.Ganesh B Regulwar
Associate professor

Signature of the HOD
Dr. G Suryanarayana
Associate Professor, HOD, IT

Acknowledgement

The satisfaction that accompanies the successful completion of the task would be put incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success.

We wish to express our deep sense of gratitude to **Dr.Ganesh B Regulwar**, Associate professor and Project Supervisor, Department of Information Technology, Vardhaman College of Engineering, for his able guidance and useful suggestions, which helped us in completing the project in time.

We are particularly thankful to **Dr. G Suryanarayana**, the Head of the Department, Department of Information Technology, his guidance, intense support and encouragement, which helped us to mold our project into a successful one.

We show gratitude to our honorable Principal **Dr. J.V.R. Ravindra**, for providing all facilities and support.

We avail this opportunity to express our deep sense of gratitude and heartful thanks to **Dr. Teegala Vijender Reddy**, Chairman and **Sri Teegala Upender Reddy**, Secretary of VCE, for providing a congenial atmosphere to complete this project successfully.

We also thank all the staff members of Information Technology department for their valuable support and generous advice. Finally thanks to all our friends and family members for their continuous support and enthusiastic help.

HARI BHARADWAJ

TOURIA TANAZZUM

YAMSANI VAISHNAVI

Abstract

Completely Automated Public Turing Test to Tell Computers and Humans Apart (CAPTCHA) is an important human-machine distinction technology for website to prevent the automatic malicious program attack. CAPTCHA recognition studies can find security breaches in CAPTCHA, improve CAPTCHA technology, it can also promote the technologies of license plate recognition and handwriting recognition. In the proposed CAPTCHA security system will generate the CAPTCHA by using a new improved algorithm which will be interesting to solve at the same time it will be tougher than previous to solved by the bots while will feel easy. The humans have limitations on the speed of response then compared to any computer and hence the computer must be slower than the human and so we will make full benefit of this and use it in the proposed system. The CAPTCHA security system will contain colored graphical interface with the font is limited to two while the border line thickness and color will be fixed. The CAPTCHA security system will generate random text which will be shorter than the present system but will be difficult to hack as it will randomly generate.

Table of Contents

List of Figures

Abbreviations

Abbreviation	Description
VCE	Vardhaman College of Engineering
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
UML	Unified Modeling language
HTML	Hypertext Markup Language
DDoS	Distributed Denial-of-Service
CBIR	Content-Based Image Retrieval

CHAPTER 1

Introduction

1.1 Introduction

A reusable CAPTCHA security engine is a system designed to provide an effective and efficient method of distinguishing between human users and automated bots on online platforms. CAPTCHA stands for "Completely Automated Public Turing test to tell Computers and Humans Apart." The primary purpose of a CAPTCHA is to prevent malicious bots from accessing or interfering with online services, such as spamming, data scraping, or unauthorized account creation.

A reusable CAPTCHA security engine offers several advantages over traditional CAPTCHA systems. It aims to strike a balance between security and user experience by implementing innovative techniques that are resilient to bot attacks while minimizing inconvenience for genuine users. The term "reusable" signifies that the CAPTCHA system can be deployed and used across multiple applications or platforms without significant modifications.

In today's digital landscape, where automated bots pose a significant threat to online security and user privacy, the need for effective CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) systems has become paramount. CAPTCHAs serve as a crucial line of defense, preventing automated bots from accessing sensitive information, conducting malicious activities, or spamming online platforms. However, traditional CAPTCHA systems often present challenges that are difficult for users and may lead to frustration and poor user experiences.

1.2 Problem Definition

The problem addressed by a reusable CAPTCHA security engine is the need to effectively distinguish between human users and automated bots on online platforms while providing a seamless user experience. Traditional CAPTCHA systems often present challenges that are overly complex or time-consuming for genuine users, leading to frustration and poor user satisfaction. At the same time, malicious bots have become increasingly sophisticated, finding ways to bypass or overcome traditional CAPTCHA mechanisms, posing significant threats to online security.

The accessibility problem involves ensuring that the challenges presented are perceivable and solvable by users with visual, hearing, cognitive, or motor limitations. The usability problem extends to optimizing the usability of the CAPTCHA system. The challenges should be designed in a way that aligns with users' mental models and expectations, requiring minimal cognitive load and effort to complete. As online platforms experience varying levels of user traffic, the scalability problem extends to ensuring that the CAPTCHA system can handle high volumes of simultaneous requests without compromising performance or introducing significant latency

CHAPTER 2

Literature Survey

2.1 Existing System

A way to tell apart a human from a computer by a test is known as a Turing Test. When a computer program is able to generate such tests and evaluate the result, it is known as a CAPTCHA (Completely Automated Public test to Tell Computers and Humans Apart). In the past, Websites have often been attacked by malicious programs that register for service on massive scale. This has driven researchers to the idea of CAPTCHA-based security, to ensure that such attacks are not possible without human intervention, which in turn makes them ineffective. CAPTCHA-based security protocols have also been proposed for related issues, e.g., countering Distributed Denial-of-Service (DDoS) attacks on Web servers. A CAPTCHA acts as a security mechanism by requiring a correct answer to a question which only a human can answer any better than a random guess. Humans have speed limitation and hence cannot replicate the impact of an automated program. Thus the basic requirement of a CAPTCHA is that computer programs must be slower than humans in responding correctly. To that purpose, the semantic gap between human understanding and the current level of machine intelligence can be exploited. Most current CAPTCHAs are text-based. Commercial text-based CAPTCHAs have been broken using object-recognition techniques, with accuracies of up to 99% on EZ-Gimpy. This reduces the reliability of security protocols based on text-based CAPTCHAs. There have been attempts to make these systems harder to break by systematically adding noise and distortion, but that often makes them hard for humans to decipher as well. Image-based CAPTCHAs have been proposed as alternatives to the text media. More robust and user-friendly systems can be developed. State-of-the art content-based image retrieval (CBIR) and annotation techniques have shown great promise at automatically finding semantically similar images or naming them, both of which allow means of attacking image-based CAPTCHAs.

2.2 Limitations of Existing Systems

- Risk of Failure
- Increased Bot Sophistication
- Integration Complexity
- Adoption and Standardization

2.3 Proposed method and Advantages

In the proposed CAPTCHA security system will generate the CAPTCHA by using a new improved algorithm which will be interesting to solve at the same time it will be tougher than previous to solved by the bots while they feel easy. The humans have limitations on the speed of response then compared to any computer and hence the computer must be slower than the human and so we will make full benefit of this and use it in the proposed system. The CAPTCHA security system will contain colored graphical interface with the font is limited to two while the border line thickness and color will be fixed. The CAPTCHA security system will generate random text which will be shorter than the present system but will be difficult to hack as it will randomly generate.

We have developed a website where it contains all the information about CAPTCHA. We have different sections related to CAPTCHA .We also included two python files where we demonstrate the usage of CAPTCHA. We included two buttons which relate to the different python files. First one is the demonstration of text based CAPTCHA. And the second one in the demonstration of image based CAPTCHA.

Text-based CAPTCHA:

While clicking the button, the user is prompted to another web page which shows “Python File Executed!” and a popup is displayed to verify the CAPTCHA. We have to enter the given text based CAPTCHA in the place provided to enter the CAPTCHA. If the user enters the correct CAPTCHA it shows “Verified” in green color, else it displays “Incorrect” in red color.

Image-based CAPTCHA:

While clicking the button, the user is shown a random image with two rectangles, two triangles and two circles. You have to enter the output shown in the command prompt. Circle is denoted by ‘C’, triangle is denoted by ‘T’, and rectangle by ‘R’. If the user enters the correct output it displays “Captcha Verification Successful”, else is displays “Captcha Verification Failed”.

2.4 Advantages of proposed methodology

- Enhanced Security: A reusable CAPTCHA security engine offers improved security measures by incorporating advanced algorithms, adaptive challenge generation, and behavioral analysis techniques.
- Better User Experience: The engine aims to provide a more user-friendly experience compared to traditional CAPTCHAs.
- Adaptability to Evolving Threats: Reusable CAPTCHA security engines are designed to adapt to emerging bot tactics and techniques.
- Versatility and Compatibility: The reusability aspect of the engine allows for easy integration into various applications and platforms.
- Cost-Efficiency: Implementing a reusable CAPTCHA security engine can be cost-effective compared to developing custom CAPTCHA solutions for each individual system.
- Reduced False Positives: Through adaptive challenge generation and behavioral analysis, reusable CAPTCHA security engines can reduce false positives, minimizing the chances of genuine users being incorrectly identified as bots.

CHAPTER 3

Analysis

3.1 Functional requirements

Modules:

1. User: The User is the main entity in our proposed framework. The patient has the following main tasks:
    - View the instructions.
    - Demonstrate the usage of text and image based CAPTCHA's.
2. Admin: Admin is responsible for the following functions:
    - Authenticate all users who interact with the system.

3.2 Non-functional requirements

- **Security:** The primary purpose of a CAPTCHA is to differentiate between humans and automated bots. Therefore, one of the key challenges is to create a CAPTCHA that is difficult for bots to solve but still manageable for humans. Achieving the right balance between security and usability can be challenging.
- **Usability**: CAPTCHAs should be designed in a way that they can be easily understood and completed by humans. If a CAPTCHA is too complex or confusing, it may deter users or result in a poor user experience. Striking the right balance between security and usability is crucial to ensure that legitimate users can pass the CAPTCHA without much difficulty.
- **Accessibility**: CAPTCHAs should be accessible to all users, including those with disabilities. Developers need to consider accessibility guidelines and ensure that the CAPTCHA is perceivable and usable by individuals with different abilities, such as visually impaired users who may rely on screen readers.
- **Performance**: We make sure our website has high performance by

  1.Reducing overall load time

  2. Making the site usable as soon as possible

  3. Smoothness and interactivity

  4. We take performance measurements

- **Robustness against attacks:** CAPTCHAs are often targeted by attackers who attempt to bypass those using automated methods. Developers need to be aware of the various attack techniques, such as optical character recognition (OCR), machine learning algorithms, or crowdsourcing, and continuously improve the CAPTCHA generation process to stay ahead of these attacks.

- **User experience and aesthetics:** While security is a priority, it's also important to consider the visual appeal and user experience of the CAPTCHA. A well-designed and visually pleasing CAPTCHA can enhance the overall user experience and encourage user engagement.

3.3 Computational requirements

3.3.1 Software requirements

Operating System: Windows 10

Server: Flask, local host

Platform: Python

Technology: HTML

Client Side Technology: HTML

IDE: Visual Studio

3.3.2 Hardware requirements

Processor: Intel core

I7 RAM: 4GB

Hard Disk: 500 GB

CHAPTER 4

Design

4.1 Architectures

4.1.1 Software Architecture

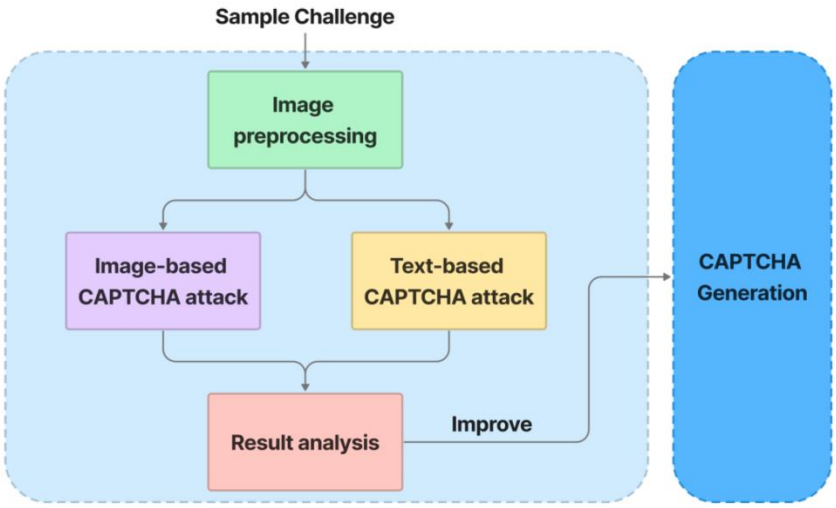


Figure 4.1: Software Architecture

4.1.2 Technical Architecture

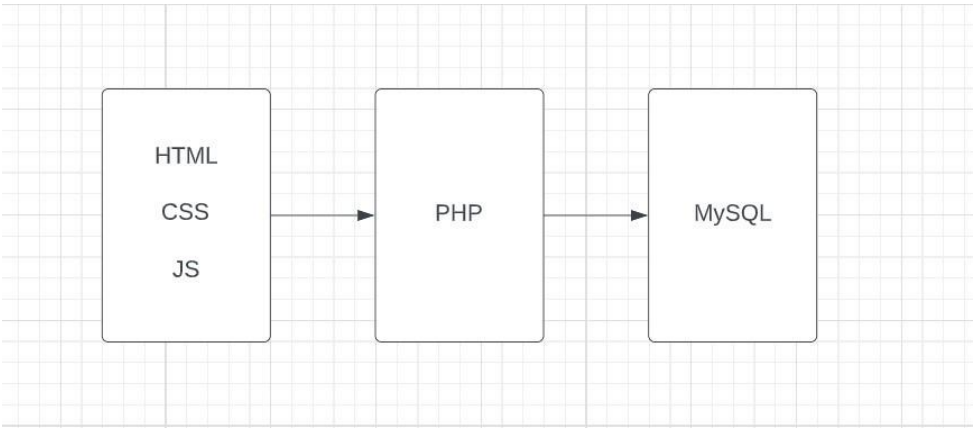


Figure 4.2: Technical Architecture

4.2 UML diagrams

UML Design

Unified Modeling Language (UML) is a general purpose modeling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite similar to blueprints used in other fields of engineering. UML is not a programming language; it is rather a visual language. We use UML diagrams to portray the behavior and structure of a system, UML helps software engineers, businessmen and system architects with modeling, design and analysis. The Object Management Group (OMG) adopted Unified Modeling Language as a standard in 1997. It's been managed by OMG ever since. International Organization for Standardization (ISO) published UML as an approved standard in 2005. UML has been revised over the years and is reviewed periodically.

Do we really need UML?

- Complex applications need collaboration and planning from multiple teams and hence require a clear and concise way to communicate amongst them.
- Businessmen do not understand code. So UML becomes essential to communicate with non-programmer's essential requirements, functionalities and processes of the system.
- A lot of time is saved down the line when teams are able to visualize processes, user interactions and static structure of the system.
- UML is linked with object oriented design and analysis. UML makes the use of elements and forms associations between them to form diagrams. Diagrams in UML can be broadly classified as:

The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frame- works, patterns and components.
- Integrate best practices.

Types of UML Diagrams:

Structural Diagrams: Capture static aspects or structure of a system. Structural Diagrams include: Component Diagrams, Object Diagrams, Class Diagrams and Deployment Diagrams.

Behavior Diagrams: Capture dynamic aspects or behavior of the system. Behavior diagrams include: Use Case Diagrams, State Diagrams, Activity Diagrams and Interaction Diagrams.

4.2.1 Object diagram:

In Unified Modeling Language (UML), an object diagram is a structural diagram that represents a snapshot of objects and their relationships at a specific point in time within a system or a specific scenario. It provides a visual representation of instances of classes and the associations between them. Object diagrams are used to illustrate the runtime instances of classes and their interactions, allowing developers to better understand the relationships and structure of objects within a system.

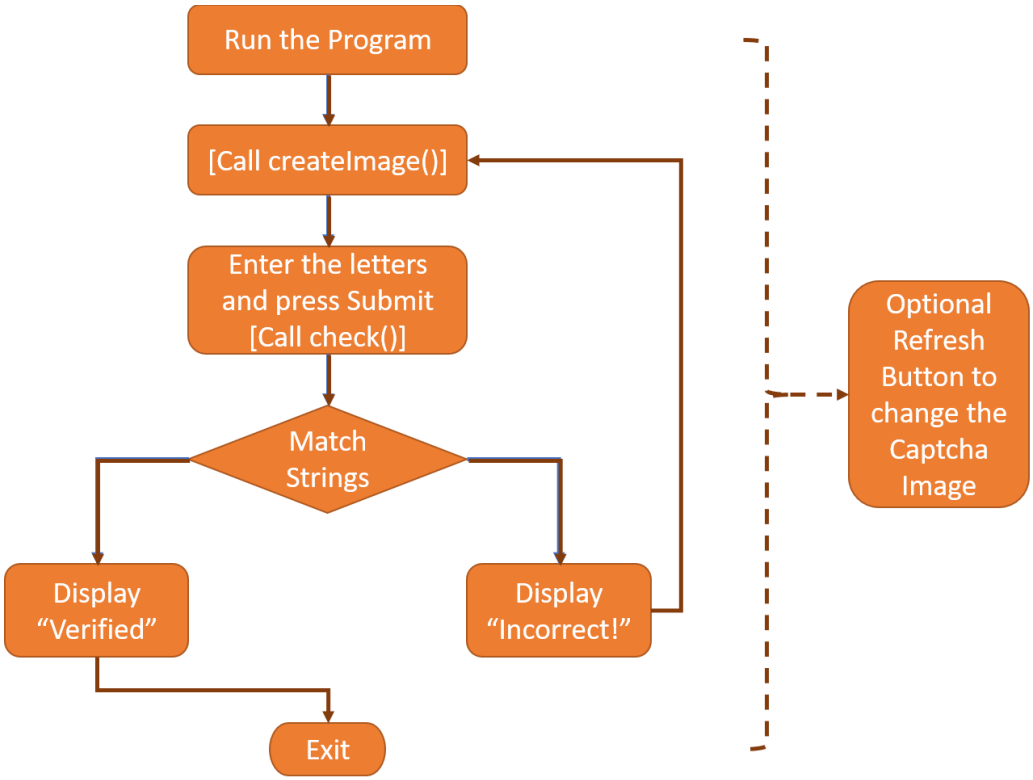


Figure 4.3: Object diagram

4.2.2 Sequence diagram

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

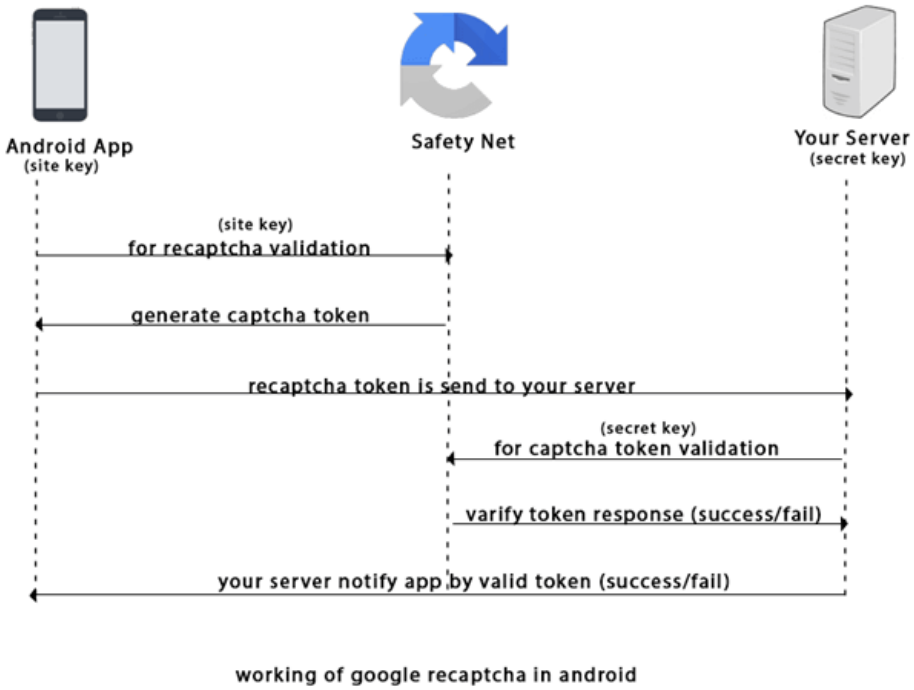


Figure 4.4: Sequence diagram

4.3 Flow Chart

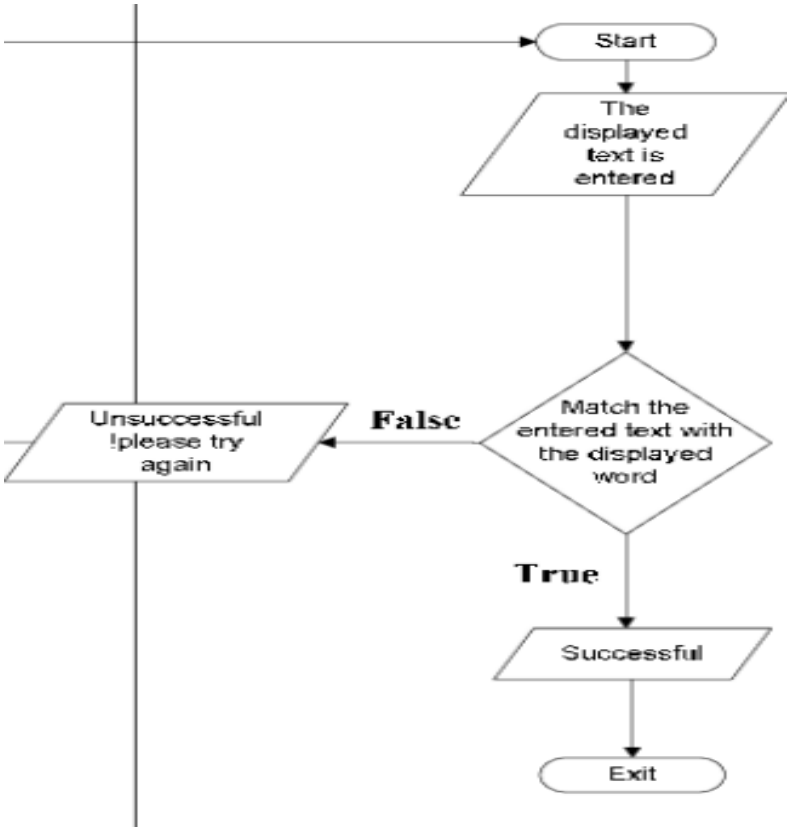


Figure 4.5: Flowchart

CHAPTER 5

Implementation

5.1 Introduction

A reusable CAPTCHA security engine is a system designed to provide an effective and efficient method of distinguishing between human users and automated bots on online platforms. CAPTCHA stands for "Completely Automated Public Turing test to tell Computers and Humans Apart." The primary purpose of a CAPTCHA is to prevent malicious bots from accessing or interfering with online services, such as spamming, data scraping, or unauthorized account creation.

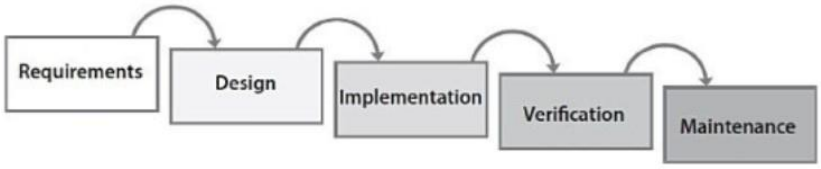


Figure 5.1: Process of Implementation

Requirement Analysis: In this initial phase, the requirements for the CAPTCHA generation system are gathered and analyzed. This involves understanding the purpose, goals, and desired features of the CAPTCHA system. Additionally, any specific security requirements or constraints are identified.

Design: In the design phase, the architecture and components of the CAPTCHA generation system are planned. This includes deciding on the types of CAPTCHA challenges to be implemented (such as image-based, audio-based, or text-based) and defining the algorithms and techniques for generating CAPTCHA challenges. The design should also consider factors such as scalability, adaptability, and compatibility with different platforms.

Implementation: In this phase, the actual coding and development of the CAPTCHA generation system take place. The chosen design and algorithms are implemented using programming languages and frameworks that best suit the requirements. The code should be well-structured, modular, and follow best practices for security and performance.

Testing: The testing phase involves verifying the functionality and security of the CAPTCHA generation system. This includes conducting unit tests to ensure individual components work correctly, integration testing to verify the interaction between different modules, and security testing to identify vulnerabilities or weaknesses in the CAPTCHA challenges. Robust testing helps ensure that the CAPTCHA system performs as expected and is resistant to bot attacks.

Deployment: Once the CAPTCHA generation system has passed rigorous testing, it is deployed in a production environment. This involves setting up the necessary infrastructure, configuring the system, and integrating it with the target application or platform. Thorough documentation and guidelines should be provided to aid in the deployment process.

Maintenance and Updates: After deployment, the CAPTCHA generation system requires ongoing maintenance and updates. This includes monitoring the system's performance, addressing any reported issues or vulnerabilities, and regularly updating the CAPTCHA generation algorithms to stay ahead of emerging threats. Continuous improvement and maintenance are essential to ensure the effectiveness and reliability of the CAPTCHA system over time.

5.2 Method of Implementation

In today's digital landscape, where automated bots pose a significant threat to online security and user privacy, the need for effective CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) systems has become paramount. CAPTCHAs serve as a crucial line of defense, preventing automated bots from accessing sensitive information, conducting malicious activities, or spamming online platforms. However, traditional CAPTCHA systems often present challenges that are difficult for users and may lead to frustration and poor user experiences.

We have developed a website where it contains all the information about CAPTCHA. We have different sections related to CAPTCHA .We also included two python files where we demonstrate the usage of CAPTCHA. We included two buttons which relate to the different python files. First one is the demonstration of text based CAPTCHA. And the second one in the demonstration of image based CAPTCHA.

Product Development

In stage two, product development, everyone stops talking and starts working.

Text-based CAPTCHA:

While clicking the button, the user is prompted to another web page which shows “Python File Executed!” and a popup is displayed to verify the CAPTCHA. We have to enter the given text based CAPTCHA in the place provided to enter the CAPTCHA. If the user enters the correct CAPTCHA it shows “Verified” in green color, else it displays “Incorrect” in red color.

Image-based CAPTCHA:

While clicking the button, the user is shown a random image with two rectangles, two triangles and two circles. You have to enter the output shown in the command prompt. Circle is denoted by ‘C’, triangle is denoted by ‘T’, and rectangle by ‘R’. If the user enters the correct output it displays “Captcha Verification Successful”, else is displays “Captcha Verification Failed”.

Alpha/Beta Test

In stage three:

Alpha Testing: Alpha testing is conducted by the developers or a select group of internal testers within the development organization. In the context of CAPTCHA verification, alpha testing involves testing the CAPTCHA system within a controlled environment before releasing it to a wider audience. Here's how alpha testing can be applied to CAPTCHA verification:

1. Test Environment Setup: Create a controlled testing environment that closely resembles the actual deployment environment where the CAPTCHA system will be used. This may include simulating user interactions, generating test cases, and configuring the necessary infrastructure.

2. Test Coverage: Develop a comprehensive set of test cases that cover various aspects of the CAPTCHA system, such as challenge generation, user response validation, security measures, and compatibility with different platforms and browsers.

3. Test Execution: Execute the test cases systematically, focusing on verifying the functionality, security, and user experience of the CAPTCHA verification process. This involves submitting different types of CAPTCHA challenges, evaluating the response validation mechanism, and ensuring that the system behaves as expected.

Beta Testing: Beta testing involves deploying the CAPTCHA system to a limited group of external users or a specific target audience. The purpose of beta testing is to gather feedback from real users and assess the system's performance in a real-world setting. Here's how beta testing can be applied to CAPTCHA verification:

1. Recruitment of Beta Testers: Select a group of external users, such as volunteers or customers, who will actively participate in testing the CAPTCHA system. Provide them with clear instructions on how to use and evaluate the system.

2. User Feedback Collection: Encourage beta testers to provide feedback on their experience with the CAPTCHA verification process. This can include their perception of the challenge difficulty, the accuracy of the validation mechanism, and any usability issues they encounter.

3. Performance Monitoring: Monitor the performance of the CAPTCHA system during the beta testing phase. Gather data on metrics such as response times, server load, and the effectiveness of the CAPTCHA challenges in deterring automated bots.

4. Bug Reporting and Iterative Improvements: Encourage beta testers to report any bugs, usability issues, or suggestions for improvement. Use their feedback to iterate on the CAPTCHA system, addressing identified issues, and making enhancements to the user experience and security.

5.3 Output Screens

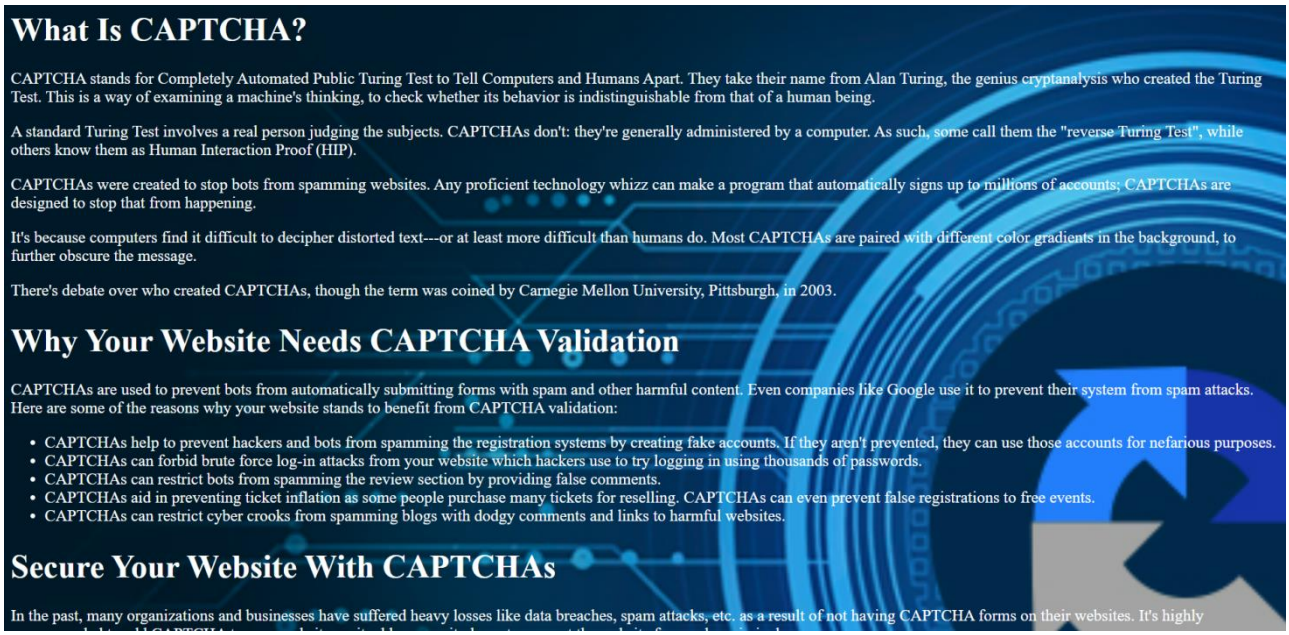


Figure 5.2.1: Home Page

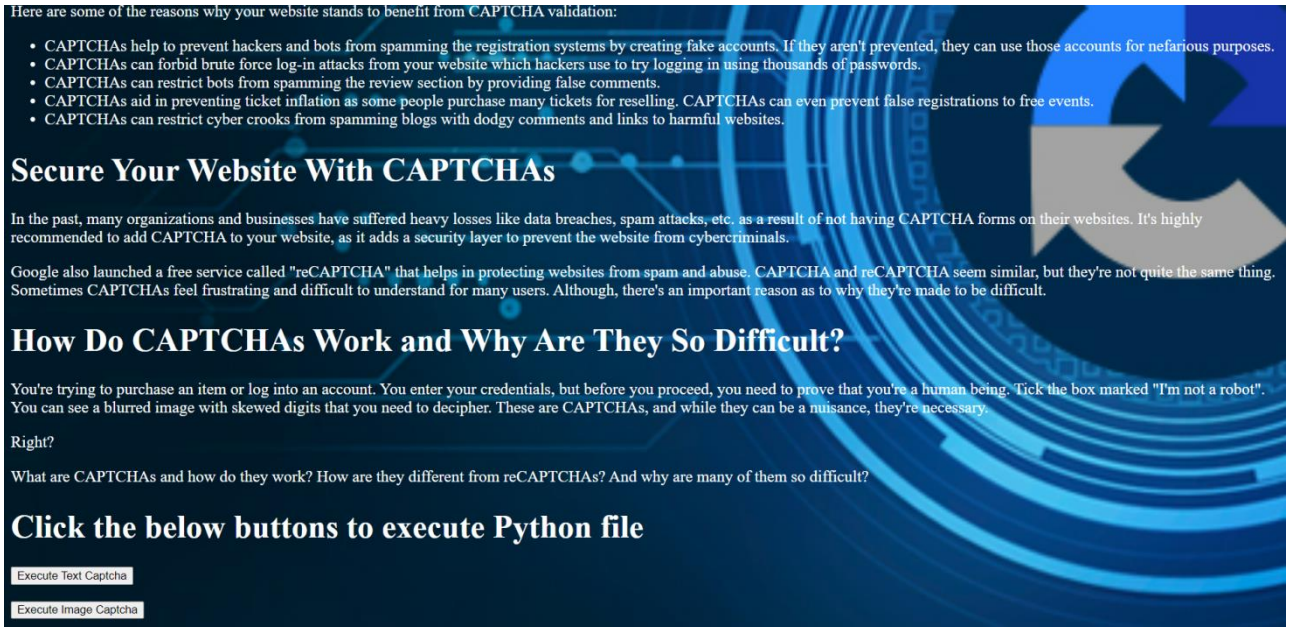


Figure 5.2.2: Home Page

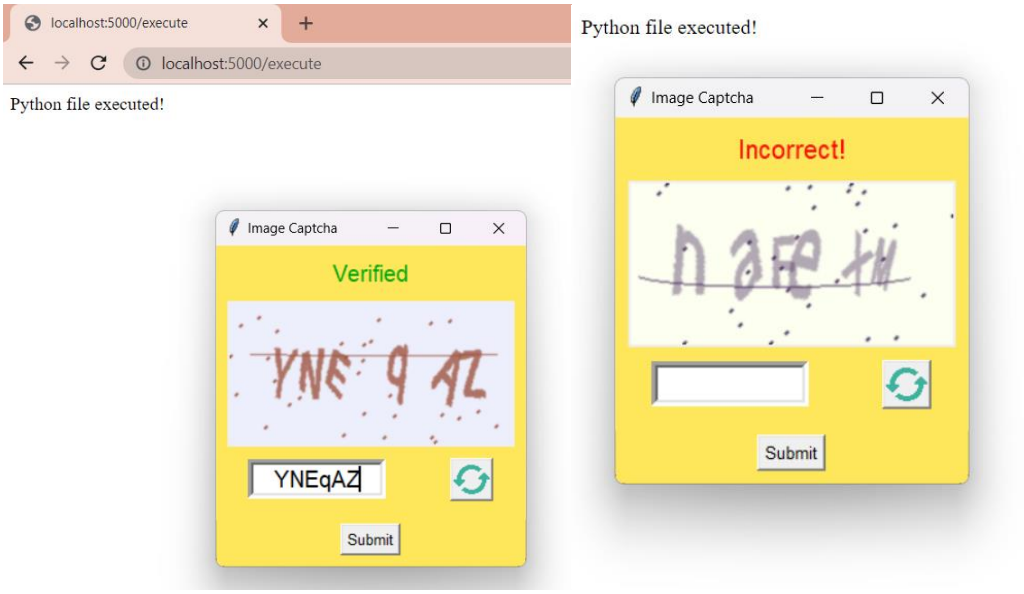


Figure 5.3: Text-based CAPTCHA

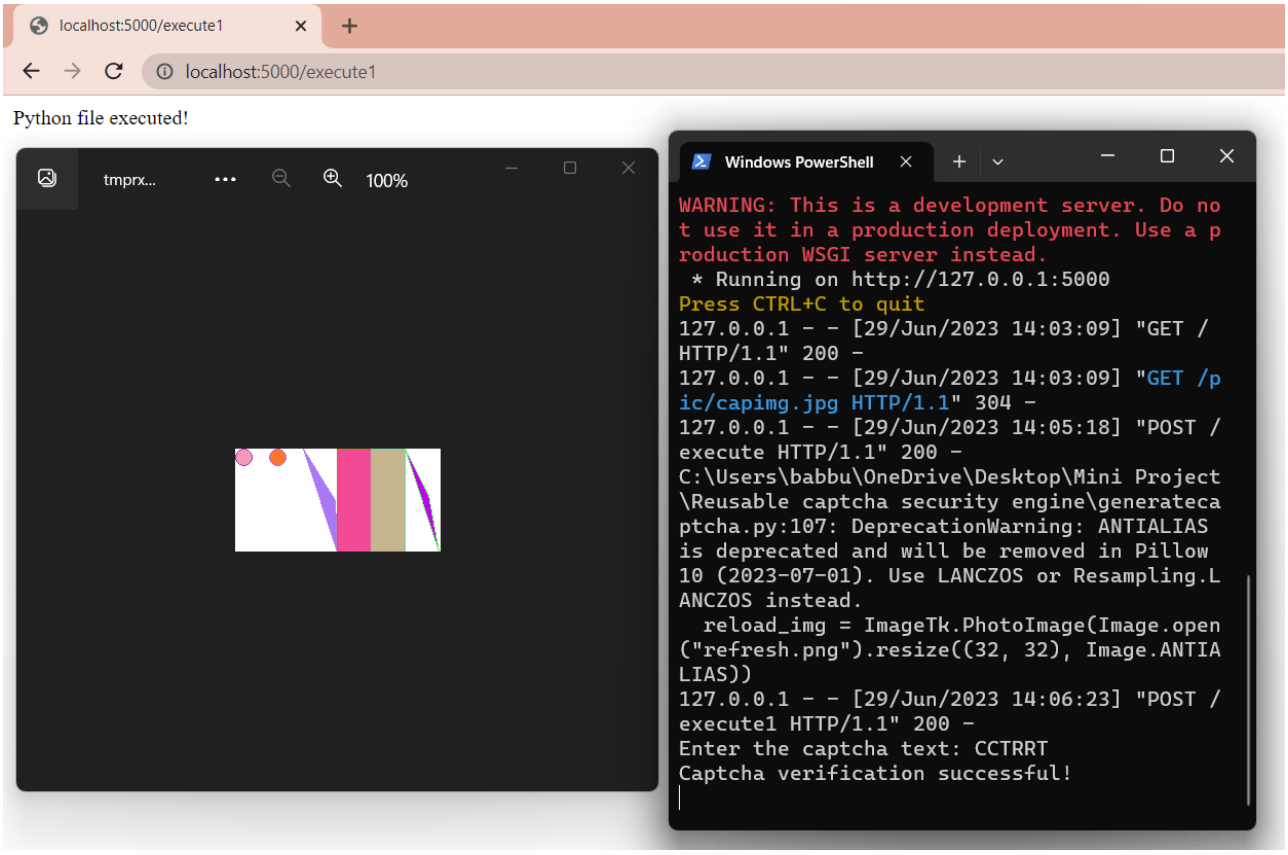


Figure 5.4.1: Image-based CAPTCHA

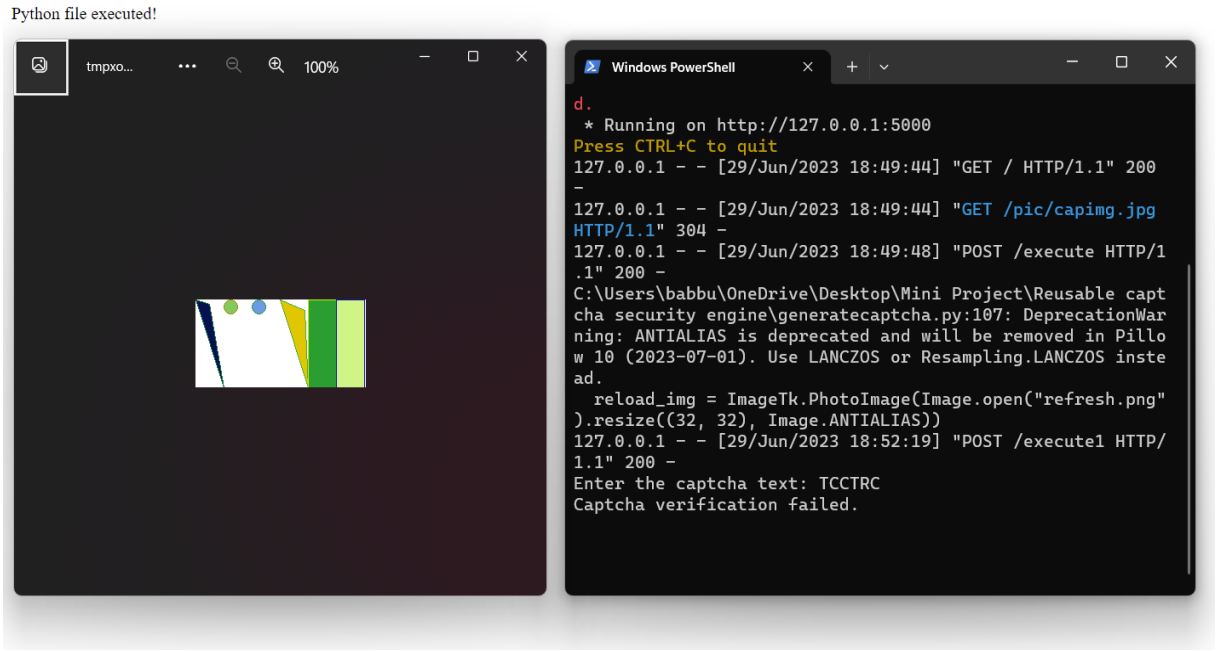


Figure 5.4.2: Image-based CAPTCHA

CHAPTER 6

Testing Results

6.1 Overview of testing

Testing:

An estimate says that 50 percent of whole software development process should be tested. Errors may ruin the software from critical level to its own removal. Software testing is done while coding by the developers and thorough testing is conducted by testing experts at various levels of code such as module testing, program testing, product testing, in-house testing and testing the product at user's end. Early discovery of errors and their remedy is the key to reliable software.

6.2 Dimensions of testing

Unit testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.3 Test cases

Security:

a. Verify that the CAPTCHA system is resistant to automated bot attacks, such as OCR (Optical Character Recognition) or audio recognition attacks.

b. Test the effectiveness of the CAPTCHA challenges by attempting to bypass or circumvent them using automated scripts or tools.

c. Validate that the CAPTCHA system has countermeasures against common attack techniques, such as brute-force attacks or CAPTCHA farms.

Accessibility:

a. Test the accessibility features of the CAPTCHA system, such as support for screen readers, alternative text-based challenges, or audio-based challenges for visually impaired users.

b. Validate that the CAPTCHA challenges are perceivable and understandable by users with different abilities or disabilities.

c. Test the compatibility of the CAPTCHA system with accessibility standards and guidelines.

Performance:

a. Test the response time of the CAPTCHA system to ensure it provides a seamless user experience without significant delays.

b. Verify the scalability of the CAPTCHA system by simulating a high volume of simultaneous CAPTCHA requests and monitoring its performance under load.

Integration:

a. Test the integration of the CAPTCHA system with the target application or platform, such as login pages, registration forms, or comment sections.

b. Validate that the CAPTCHA system seamlessly integrates and functions correctly within the intended context.

Error Handling:

a. Test the CAPTCHA system's behavior when encountering errors or exceptions, such as network connectivity issues or server failures.

b. Validate that appropriate error messages or instructions are displayed to users when there are failures or incorrect submissions.

CHAPTER 7

Conclusions and Future Scope

Hence, we have developed a website where it contains all the information about CAPTCHA. We have different sections related to CAPTCHA .We also included two python files where we demonstrate the usage of CAPTCHA. We included two buttons which relate to the different python files. First one is the demonstration of text based CAPTCHA. And the second one in the demonstration of image based CAPTCHA.

Improved Image Recognition: As technology advances, image recognition algorithms can become more sophisticated and accurate, leading to more robust CAPTCHA systems.

Biometric CAPTCHAs: CAPTCHAs could incorporate biometric data, such as fingerprint or facial recognition, to further enhance security.

Behavior-based CAPTCHAs: Instead of relying solely on static tests, future CAPTCHA systems could analyze user behavior, such as mouse movements, typing patterns, or browsing habits, to determine whether the user is human or a bot.

REFERENCES

[1]von Ahn, L., Maurer, B., McMillen, C., Abraham, D., & Blum, M. (2008). reCAPTCHA: Human-Based Character Recognition via Web Security Measures. Science, 321(5895), 1465-1468.

[2]Elson, J., Douceur, J. R., Howell, J., & Saul, J. (2007). Asirra: A CAPTCHA that Exploits Interest-Aligned Manual Image Categorization. In Proceedings of ACM Conference on Computer and Communications Security (CCS).

[3]Yan, J., & El Ahmad, A. S. (2008). A Low-Cost Attack on a Microsoft CAPTCHA. In Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS).

[4]Bursztein, E., Bethard, S., Fabry, C., & Mitchell, J. C. (2014). The End is Nigh: Generic Solving of Text-Based CAPTCHAs. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS).

[5] Gao, X., Liu, F., & Yu, S. (2016). Machine Learning-Based CAPTCHA Recognition Techniques. ACM Computing Surveys, 49(2), Article No. 34.

[6] Singh, V., Gupta, P., & Tyagi, V. (2017). An Overview on CAPTCHA: Alternatives and Current State of CAPTCHA Security. Journal of Network and Computer Applications, 88, 99-117.

[7] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. IEEE Access, 6:14410–14430, 2018.

[8] Abdalnaser Algwil, Dan Ciresan, Beibei Liu, and Jeff Yan. A security analysis of automated Chinese turing tests. In Proceedings of the 32nd Annual Conference on Computer Security Applications, pages 520–532, 2016.

[9] Qiujie Li. A computer vision attack on the ARTiFACIAL CAPTCHA. Multimedia Tools and Applications, 74(13):4583–4597, 2015.

[10] Joseph Redmon and Ali Farhadi. YOLOv3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018.