

# 1. INTRODUCTION

## 1.1 Introduction

The substantial and automated access to Web resources through robots has made it essential for Web service providers to make some anticipation about whether "user" is a human or a robot. A Human Interaction Proof (HIP) like Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) offers a way to make such a distinction. Captcha is a reverse Turing test used by Web service providers to secure human interaction with assumed services from Web bots. Several Web services that include but are not limited to free email accounts, submission of e-mail, online polls, chat rooms, search engines, blogs, password systems, etc. use Captcha as a defense mechanism against automated Web bots. This paper presents various aspects of Captcha methods that include its types, generation methods, robustness against attacks and various usability aspects. It presents a review of existing Captcha schemes besides relative merits of text and image based on them. It illustrates the working of Captcha and general methods used for their generation besides security and usability issues of Captcha methods. It provides guidelines to improve security control of Captcha methods against various possible attacks and guidelines to improve their usability. Also study the text based Captcha solving by deep learning techniques. The attackers easily retrieve the text using OCR technique from text-based Captcha. The proposed a new image-based Captcha technique known as Style Area Captcha (SACaptcha) that is based on the neural style transfer technique. To pass the test, users are required to click foreground style-transferred regions in an image based on a brief description. Unlike earlier image-based Captchas, SACaptcha relies on human understanding of semantic information and pixel-level segmentation, which seems to be more difficult for machines to solve.

## **1.1 Problem Overview**

1. Captcha robustness is to prevent automated attacks from achieving a success rate higher than 1%.
2. Increases the security of the image-based Captcha using SACaptcha.
3. Provides evidence that deep learning is a double-edged sword used to detect and attack Captchas and improve the security of Captchas.

## **1.2 Problem Statement**

To design and implement an image-based Captcha known as Style Area Captcha (SACaptcha) that is based on the neural style transfer techniques that are user friendly, require less server processing and offer improved security control against bots.

## **1.3 Objectives**

- The primary objective of image-based CAPTCHAs is to distinguish between human users and automated bots or scripts.
- To authenticate human users, ensuring that the user interacting with a website or service is indeed a real person and not a computer program.
- To improve security measures against advanced bot attacks.
- To create CAPTCHAs that are effective against bots while remaining user-friendly and ensuring that legitimate users can easily pass through without undue hassle or frustration.
- To maintain security, verify human presence, prevent abuse, and continually adapt to counter evolving threats from automated scripts and bots.

## 2. LITERATURE SURVEY

A main feature [1] of such hollow CAPTCHAs is to use contour lines to form connected characters with the aim of improving security and usability simultaneously, as it is hard for state-of-the-art character recognition programs to segment and recognize such connected characters, which are however easy to human eyes. The analysis provides a set of guidelines for designing hollow CAPTCHAs, and a method from comparing security of different schemes. Advantages are: It improves usability. It finds a segmentation resistant mechanism that is secure and user-friendly simultaneously. Disadvantages are: Need to design better for getting better security.

In [2] paper, systematically analyzed the security of the two-layer Captcha. A novel two-dimensional segmentation approach is proposed to separate a Captcha image along both vertical and horizontal directions, which helps create many single characters and is unlike traditional segmentation techniques. Advantages are: It is a simple and effective method to attack the two-layer Captcha deployed by Microsoft, and achieves a success rate of 44.6%. It decreases time spent on data preparation and reduces manual labor. Disadvantages are: Need to design better two-layer or multi-layer Captchas with higher security levels than their predecessors.

The paper [3] introduces a novel approach to solving captchas in a single step that uses machine learning to attack the segmentation and the recognition problems simultaneously. The algorithm to exploit information and context that is not available when they are done sequentially. Advantages are: The breadth of distortions proposed algorithm is able to solve shows that it is a general solution for automatically solving captchas. It removes the need for any hand-crafted component, making given approach generalize to new captcha schemes. Disadvantages are: Need to perform reverse Turing tests.

The paper [4] presents a fast, fully parameterizable GPU implementation of Convolutional Neural Network variants. All structural CNN parameters such as input image size, number of hidden layers, number of maps per layer, kernel sizes, skipping factors and connection tables are adaptable to any particular application. We applied our networks to benchmark datasets for digit recognition (MNIST), 3D object recognition (NORB), and natural images (CIFAR10). Advantages are: The best adaptive image recognizers. No unsupervised pre training is required. The implementation is 10 to 60 times faster than a compiler optimized CPU version. Disadvantages are: It requires more computing power.

A novel approach for automatic segmentation and recognition of CAPTCHAs with variable orientation and random collapse of overlapped characters is presented in [5] paper. The main purpose of this paper is to reduce vulnerability of CAPTCHAs from frauds and to protect users against cyber-criminal activities as well as to introduce a novel approach for recognizing either handwritten or damaged texts in ancient books, manuscripts and newspapers. Advantages are: It provides better segmentation of reCAPTCHAs. The required time for reCAPTCHA word breaking using extended approach is four times less than in approach of version 2011. Robust technique. Disadvantages are: Need to protect users against cyber-criminal activities and Internet threats.

### **3. SYSYEM ARCHITECTURE AND METHODOLOGY**

#### **3.1 Introduction**

##### **3.1.1 Proposed System Scope**

A CAPTCHA is a means of automatically generating new challenges which:

1. Current software is unable to solve accurately.
2. Most humans can solve.
3. Does not rely on the type of CAPTCHA being new to the attacker.

SACaptcha is used to improve the security of Captchas by utilizing deep learning techniques.

##### **3.1.2 User Classes and Characteristics**

###### **❖ User**

- User first register account in web application.
- User fills the username and password details and solves the captcha, after he/she should login to the system.
- After user logout.

###### **❖ Admin:**

- Admin login to system.
- Admin authenticate the user.
- Admin select one input image and give input number.

- Admin selects random images like a given number.
- Admin generates style transferred images and crops one of the objects (shape).
- After, it will synthesize the basic input image with the objects.
- Admin generates the SACaptcha.
- Admin will store SACaptcha.
- Admin logout.

### **3.1.3 Assumptions and Dependencies**

- Assumption: Here we assume the images are colorful.
- Dependencies: The SACaptcha System is dependent on the large image dataset.

## **3.2 FUNCTIONAL REQUIREMENTS**

### **3.2.1 System Feature 1 – (Functional Requirement)**

1. Propose a system that generates a Style Area Captcha to increase security against bots.

#### **Performance**

The performance of the system will give results within time. It increases the security against bots and easily solves the problem.

#### **Capacity**

Capacity of a project according to data depends on the dataset of the project.

#### **Availability**

Users are allowed to login after activation of the user's account. Users can login to the system after solving captcha.

### **Reliability**

System is reliable for providing security and resistance from the attacks.

### **Security**

The system is secure because the SACaptcha is designed.

## **3.2.2 System Feature 2 – (Functional Requirement) Facilitating other Documentation**

The SRS forms the basis for a load of other important documents such as the Software Design Specification.

### **Product Validation**

It basically helps in validating with the client that the product which is being delivered, meets what they asked for.

## **Characteristics of a Software Requirement Specification**

### **Accuracy**

This is the first and foremost requirement. The development team will get nowhere if the SRS which will be the basis of the process of software development, is not accurate.

### **Completeness**

The software requirement specification should not be missing any of the requirements stated in the business requirements documentation that the user specified.

### **Prioritization of Requirements**

Software Requirement Specification should not simply be a wish list. The requirements should follow the order of priority and preference

### **3.3EXTERNAL INTERFACE REQUIREMENTS**

#### **3.3.1 User Interfaces**

##### **User**

Registration();

Login();

SolveCaptcha();

Validate();

Logout();

##### **Admin**

- login();
- Authorize\_user();
- getContentImage();
- getStyleImages();
- transfer();
- crop();
- synthesize();
- generate();
- store();



### **3.3.2 Hardware Interfaces**

- Processor - Intel i3 core
- Speed - 1.1 GHz
- RAM - 2GB
- Hard Disk - 50 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

### **3.3.3 Software Interfaces**

Operating system: Windows

7/8/10 Coding Language: Java

1.7

Database: MySQL

5.0 IDE: Eclipse Luna

### **3.3.4 Communication Interfaces**

Firefox, Chrome Browser.

## **3.4 NONFUNCTIONAL REQUIREMENTS**

### **3.4.1 Performance Requirements**

Performance will depend on entered query available data.

### **3.4.2 Safety Requirements**

Validation should be properly provided.

### **3.4.3 Security Requirements**

Account password should follow some standard rules.

### **3.4.4 Software Quality Attributes**

#### **Usability:**

The system should be user friendly and self-explanatory. Proposed system is Flexible, Robust, and easily Testable.

#### **Accuracy**

The system will give results related to the query.

#### **Openness**

The system should be extensible to guarantee that it is useful for community systems.

#### **Usability**

The proposed system will be helpful in Google, Microsoft etc. applications where users can solve the captcha not bots.

#### **Reliability**

The system should have accurate results and fast responses to users.

## **3.5 SYSTEM REQUIREMENTS**

### **3.5.1 Database Requirements**

Database: MySQL 5.0

### **3.5.2 Software Requirements**

Operating system: Windows 7.

Coding Language: Java 1.7 IDE:

Eclipse Luna

### 3.6 ANALYSIS MODELS: SDLC MODEL TO BE APPLIED

The Waterfall Model is sequential design process, often used in Software development processes; where progress is seen as flowing steadily down through the phase of conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance. There are 5 Phase of waterfall model:

#### **Requirement Gathering and analysis:**

Here requirements are gathered which means which kind of dataset we have to use. Then what are the functional requirements of the system. Document is prepared that use cases are designed.

#### **System Design:**

In this stage we decide which hardware and software we require to design the system. It uses:

#### **Hardware Requirements:**

Processor	- Intel i3 core
Speed	- 1.1 GHz
RAM	- 256 MB(min)
Hard Disk	- 20 GB
Key Board	- Standard Windows Keyboard
Mouse	- Two or Three Button Mouse
Monitor	- SVGA

#### **Software Requirements:**

Operating System	: Windows7
Application Server	: Apache Tomcat
7.0 Coding language	: JAVA (JDK 7.0)
Database	: MySQL 5.0 IDE

**Verification:** This stage all developed software is installed and they are tested in different ways against system requirements.

**Maintenance:** According to the software's new version and use, they need to update. In our system some browsers are not supportive to our web pages because we need to use only specific browser.

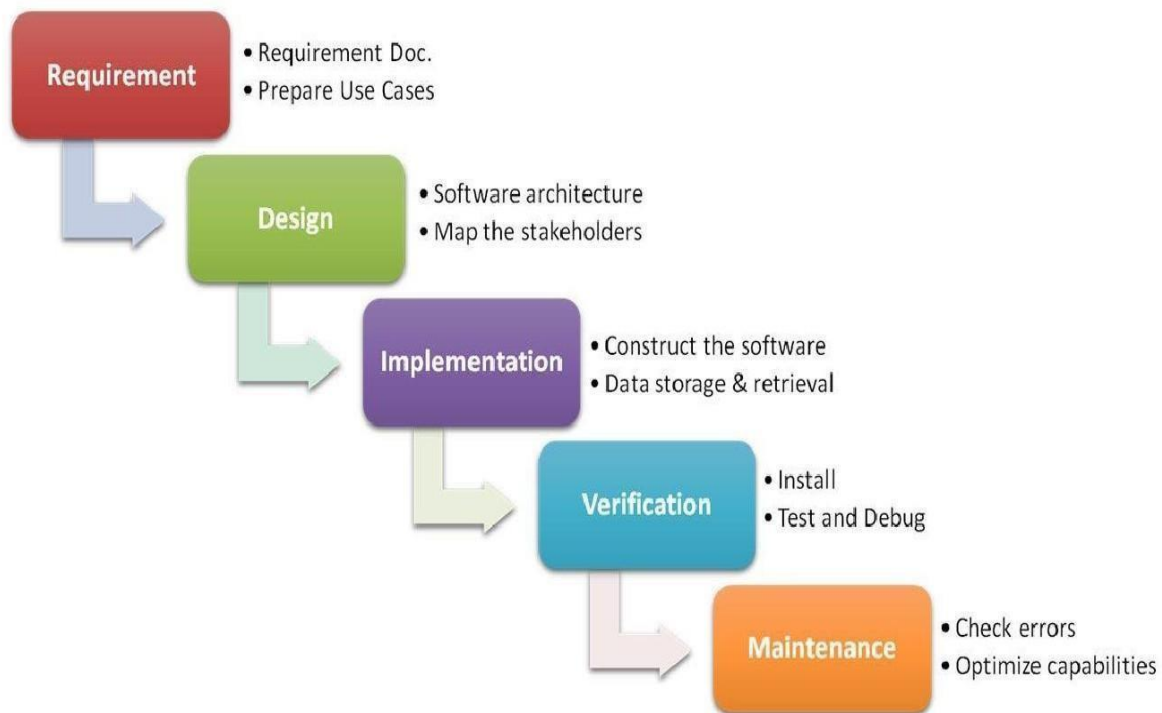


Fig. 1: Waterfall model

## 4. SOFTWARE AND SYSTEM DESIGN

### 4.1 SYSTEM Design

Previous image-based Captchas have describes various issues: some schemes require humans to manually select source images or add labels to images; some are based on a database, and if the database is compromised, they become vulnerable; some schemes incur a high transmission cost; and, most importantly, almost all of them have been proven to be insecure. To overcome these issues, proposes a novel image-based Captcha named Style Area Captcha (SACaptcha), which is based on semantic information understanding, pixel-level segmentation and deep learning techniques.

The proposed system is described in given fig.1. In this work, select a single input content image. After creating the number of foreground style-transferred regions in each Captcha image to 4 to 7 the style transfers images. The shape of each region is randomly selected: it can be a rectangle, a triangle, a circle or other irregular shapes such as a heart, a leaf, a moon and so on. One of these style-transferred images is synthesized with the original image. Randomly crop regions with different shapes from other style-transferred images and relocate them in the synthetic background to generate a Captcha. After, generate a brief description to guide users on how to pass the test. The output is the generated SACaptcha.

Advantages:

1. SACaptcha are easy for humans to solve but remain difficult for computers.
2. SACaptcha is easy to generate and evaluate.
3. Improves the security of Captchas by utilizing deep learning techniques.

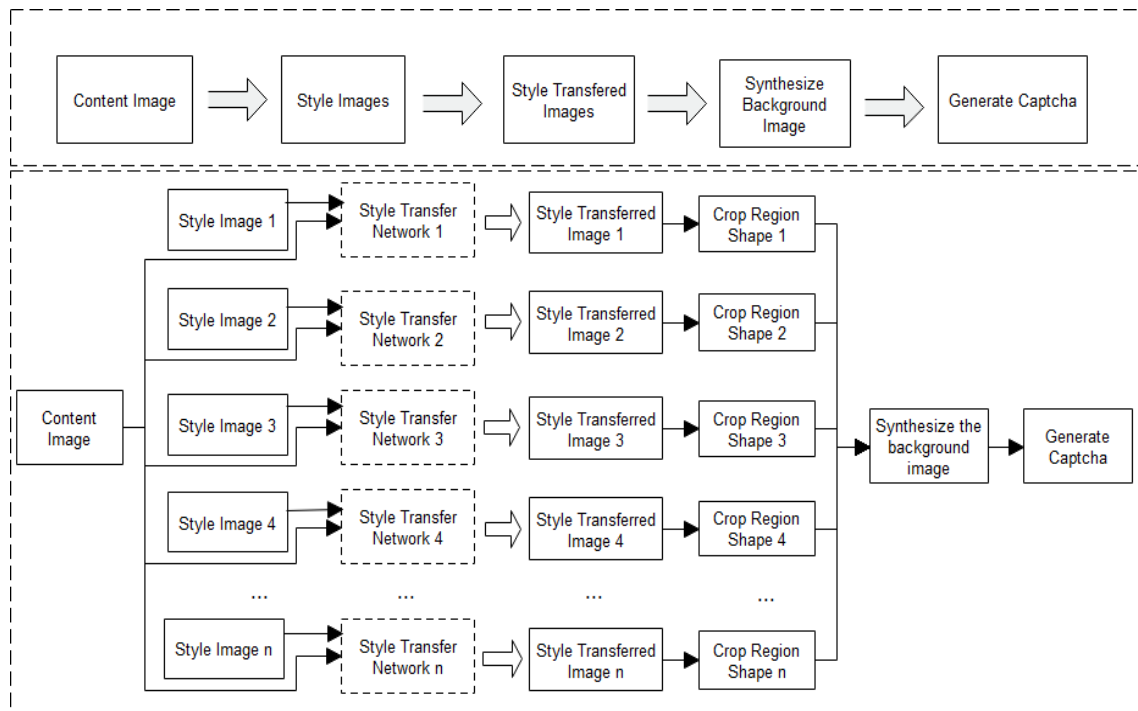


Fig 1: Proposed System Architecture

## 4.2 DATA FLOW DIAGRAMS:

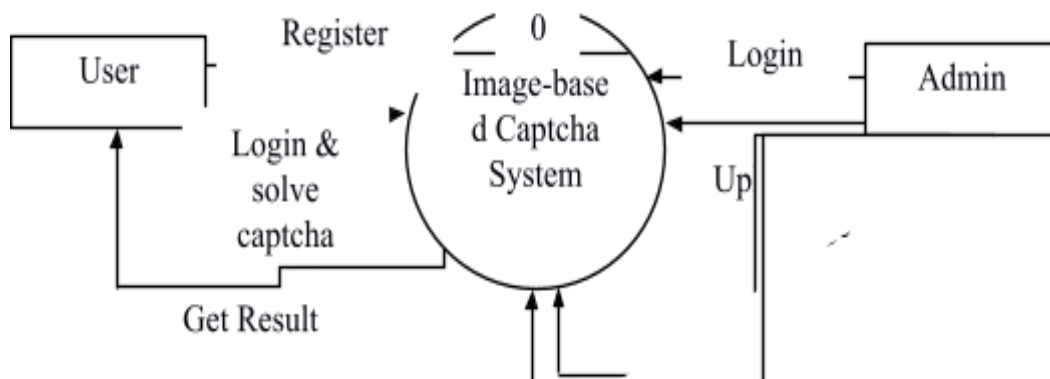


Fig.2. DFD Level 0

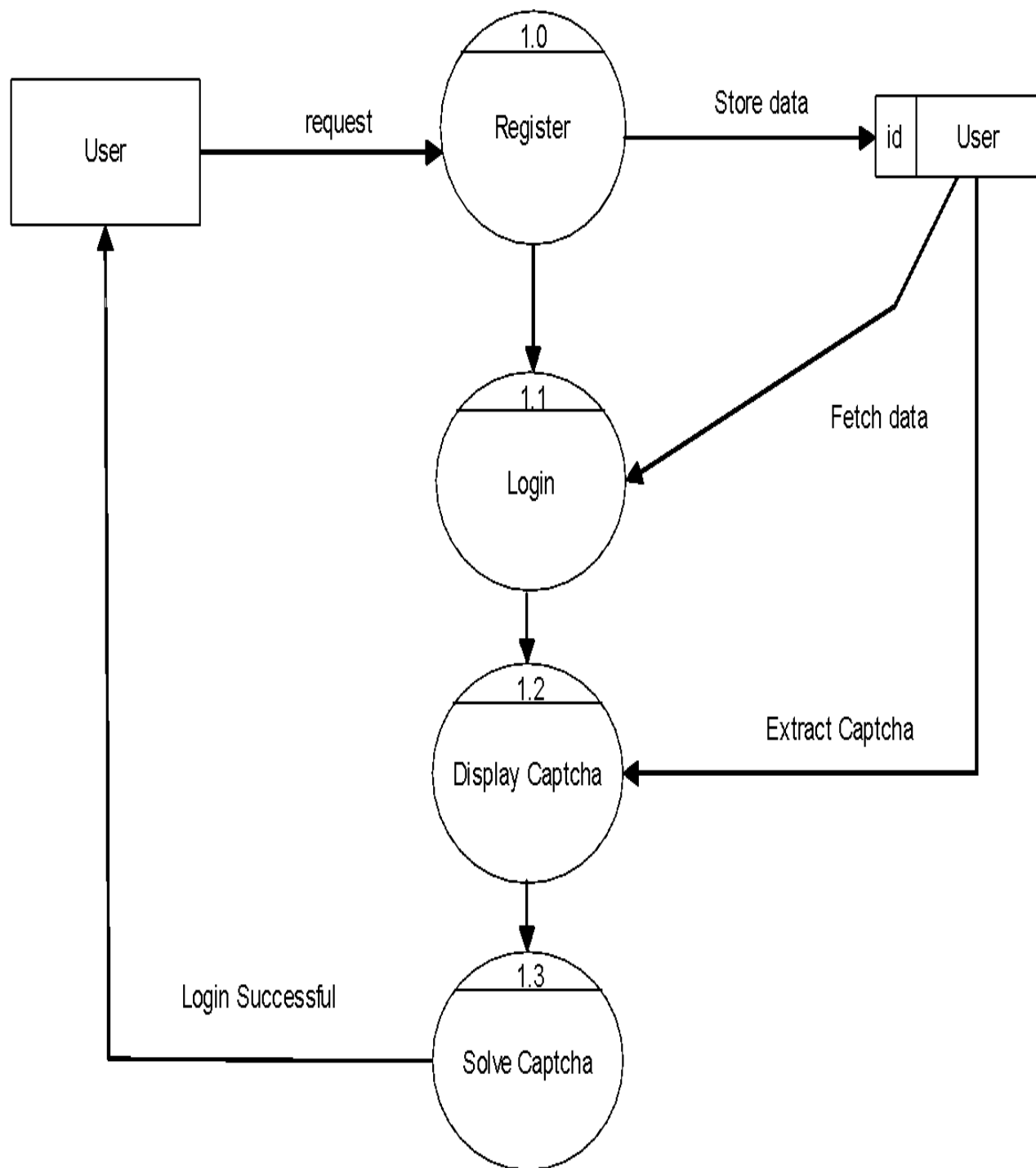


Fig.3. DFD Level 1

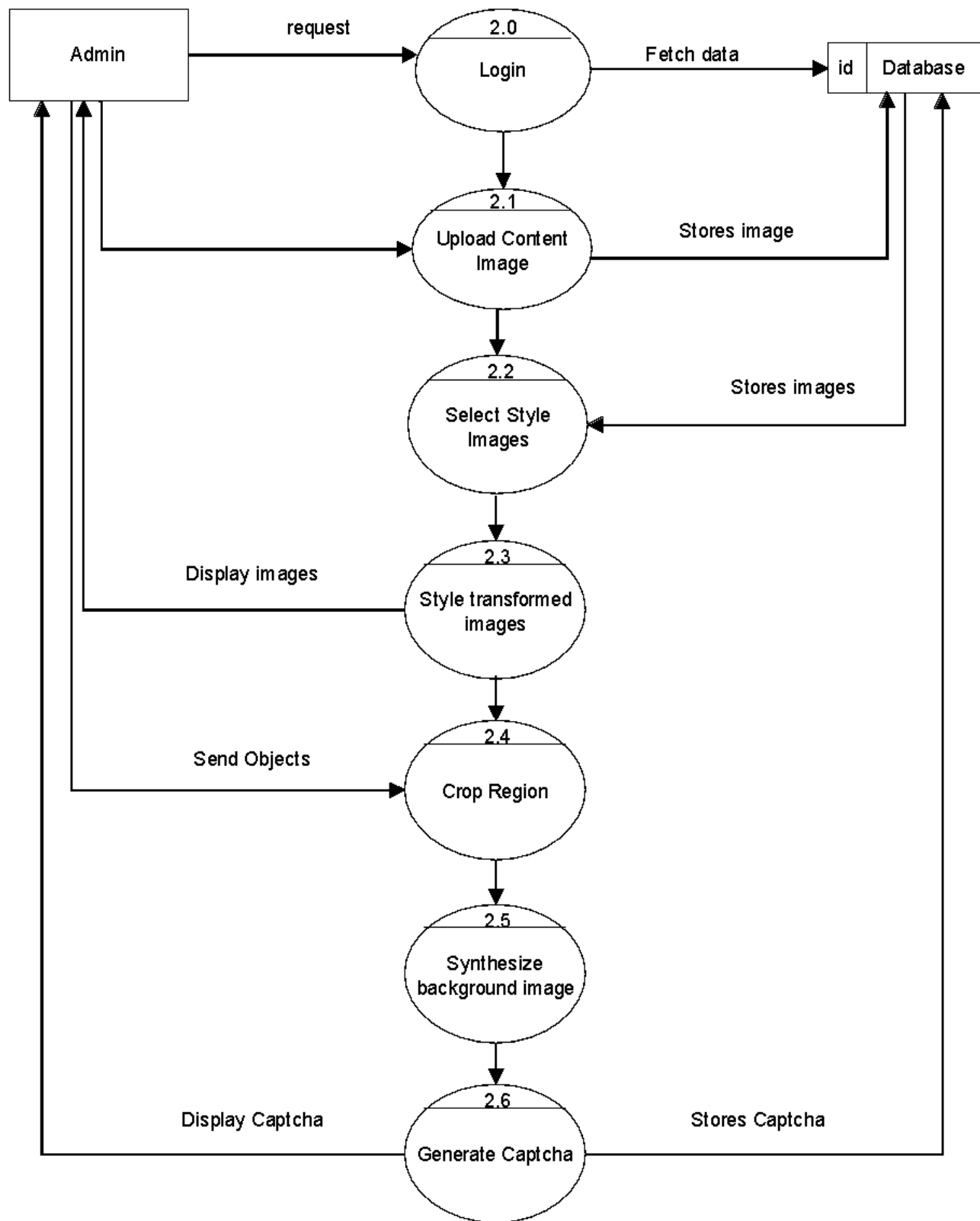


Fig.4. DFD Level 2



## 4.3 UML DIAGRAMS

### 4.3.1 USE CASE DIAGRAM:



Fig.5. Use Case Diagram

### 4.3.2 Sequence Diagram:

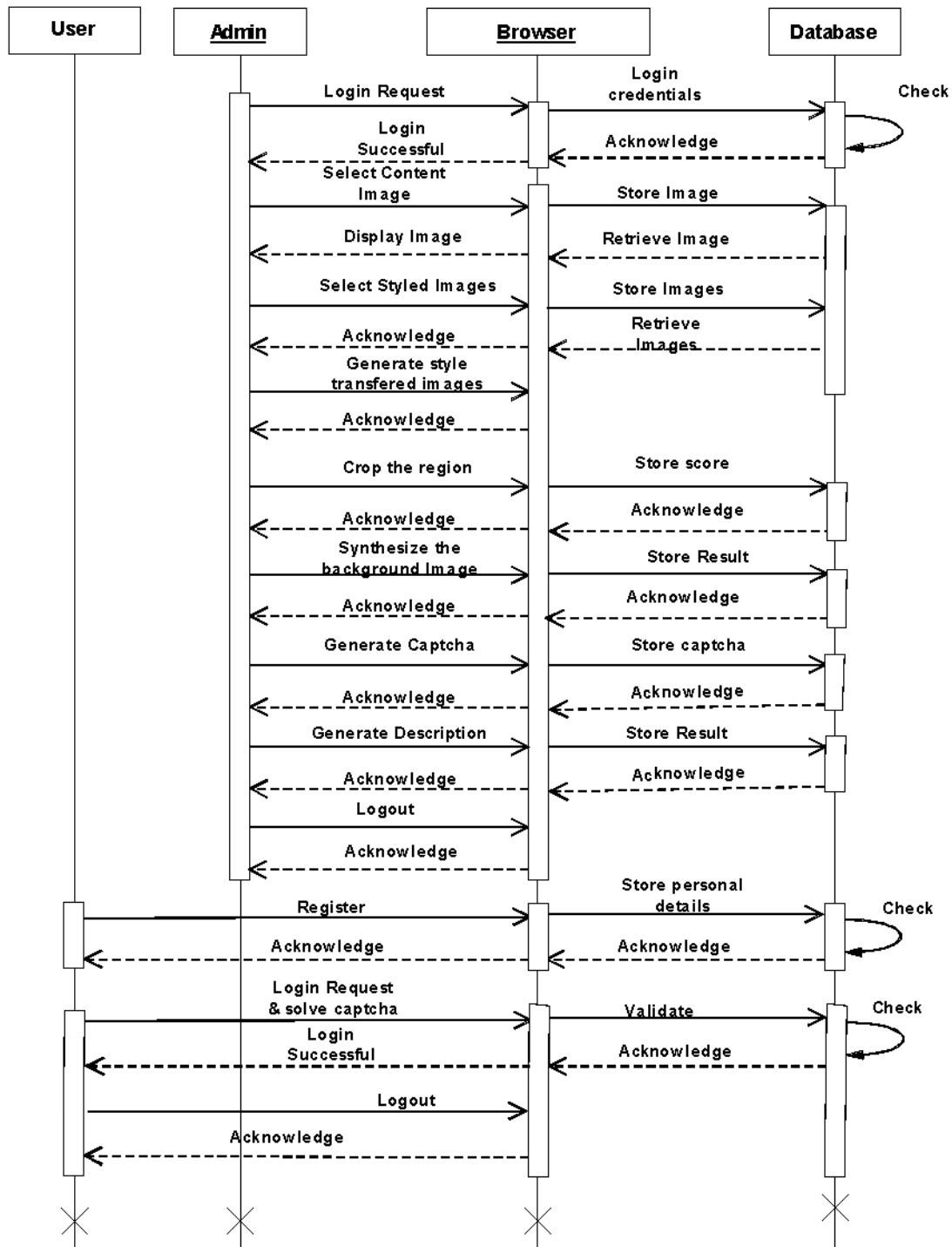


Fig.6. Sequence Diagram

### 4.3.3 Class Diagram:

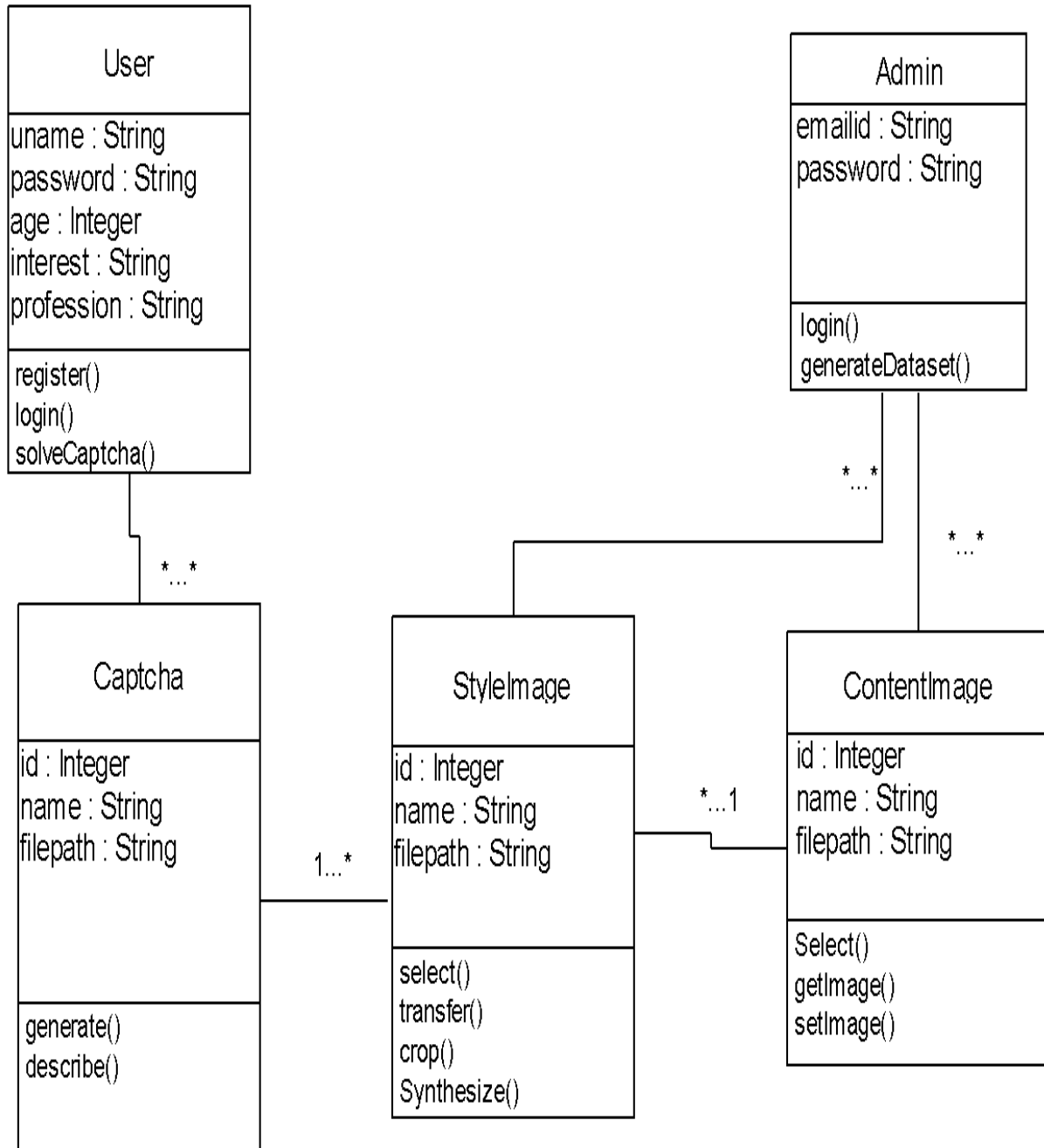


Fig.7. Class Diagram

#### 4.3.4. Component Diagram:

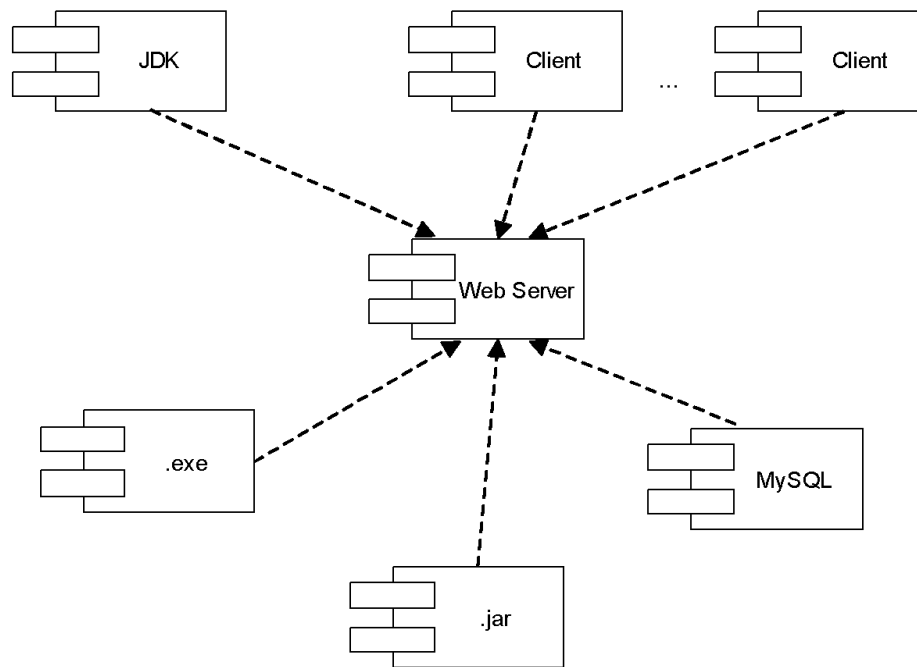


Fig.8. Component Diagram

#### 4.3.5. Deployment Diagram:

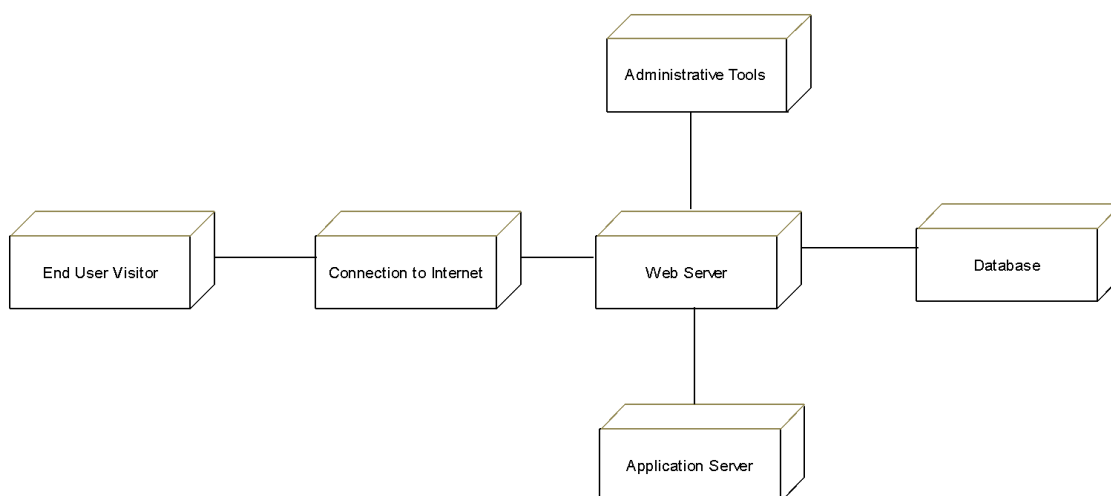


Fig.9. Deployment Diagram

## 5. HARDWARE AND SYSTEM DESIGN

### Hardware Requirements:

- |              |                             |
|--------------|-----------------------------|
| 1. Processor | - Intel i3/i5/i7            |
| 2. Speed     | - 3.1 GHz                   |
| 3. RAM       | - 4 GB(min)                 |
| 4. Hard Disk | - 40 GB                     |
| 5. Key Board | - Standard Windows Keyboard |
| 6. Mouse     | - Two or Three Button Mouse |
| 7. Monitor   | - SVGA                      |

## **6. RESULTS**

### **6.1 Testing:**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

### **TYPES OF TESTS**

#### **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.

Unit testing in an adversarial CAPTCHA project using JSP and CNN is critical for ensuring the robustness and security of the system. The unit tests should focus on the

CAPTCHA generation and verification process, including the integrity and performance of the CNN model used to generate CAPTCHA images and evaluate user responses. Tests should verify that the CAPTCHA images are generated with varying degrees of complexity to withstand automated attacks while remaining usable for legitimate users. The tests should also validate the CNN's ability to accurately classify human response and reject automated ones. Additionally, edge cases such as malformed inputs and network issues should be tested to ensure the system remains resilient under different conditions.

### **Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

In an adversarial CAPTCHA project using JSP and CNN, integration testing plays a key role in validating how different components interact to provide a seamless and secure user experience. The focus is on the entire login process, from generating and displaying CAPTCHA images to handling user input and verifying responses using the CNN model. Integration tests should verify that the user interface (UI) correctly displays the CAPTCHA image with shapes in various orientations, colors, and sizes, ensuring diversity and resistance to automated attacks.

When the user clicks on a shape, tests should confirm that the click is accurately registered and passed to the server for verification. The integration testing should ensure that the CNN model properly evaluates the user's selection against the expected answer, allowing login only if the user correctly identifies the target shape.

Tests should account for different scenarios such as varying screen sizes, device types, and network speeds to guarantee consistent behavior across environments. Additionally, tests should verify error handling, such as displaying an appropriate message to the user if the CAPTCHA is incorrect or if there are system errors. Integration tests should also assess how the system behaves under different loads and user concurrency, ensuring that the system remains responsive and performs well even under heavy usage. Finally, potential security vulnerabilities, such as bypassing the CAPTCHA or exploiting the login process, should be tested to maintain the integrity and security of the system.



## Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted. Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identifying Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

## System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

## **White Box Testing**

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It has a purpose. It is used to test areas that cannot be reached from a black box level.

In an adversarial CAPTCHA project using JSP and CNN, white box testing involves inspecting the internal code to ensure the logic for generating CAPTCHA images.

handling user input is correct. This includes verifying the CNN model's integration, ensuring accurate shape generation, and confirming proper user input handling and verification against expected answers. It also involves checking error handling and performance aspects, particularly under high-load scenarios, to maintain the system's efficiency.

## **Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document.

It is a test in which the software under test is treated as a black box, you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

This includes testing the user interface to confirm CAPTCHA images display correctly with diverse shapes and the user can interact intuitively. The process ensures the login functionality works as expected when the correct shape is clicked and appropriate error messages are shown for invalid inputs. Additionally, security and compatibility across different devices, browsers, and network conditions should be tested to confirm the system's robustness and accessibility.

## **Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

### **Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

### **Test objectives**

- All field entries must work properly
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

### **Features to be tested**

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

## **Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

## **Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

### **Test cases:**

Testing of project problem statements using generated test data (using mathematical models, GUI, Function testing principles, if any) selection and appropriate use of testing tools, testing of UML diagram's reliability.

Module-ID:-01

Modules to be tested:- Registration

1. Enter the case insensitive Username click on Submit button. Expected: It should display errors.
2. Enter the case sensitive Username click on the Submit button. Expected: It should be accepted.
3. Enter the case insensitive Password click on Submit button. Expected: It should display errors.
4. Enter the case sensitive Password click on the Submit button. Expected: It should be accepted.
5. Enter the case insensitive Mobile Number click on Submit button. Expected: It should display errors.
6. Enter the case sensitive Mobile Number click on the Submit button. Expected: It should be accepted.
7. Enter the wrong address and click on the Submit button. Expected: It should display errors.
8. Enter the correct address and click on the Submit button. Expected: It should be accepted.

**Table 1:**

Test Case_ID	Description	Test case I/P	Actual Result	Expected result	Test case criteria (P/F)
101	Enter the case insensitive Username click on Submit button.	Username	Error comes	Error Should come	P
102	Enter the case sensitive Username click on Submit button.	Username	Accept	Accept Username	P
201	Enter the case insensitive Password click on Submit button.	Password	Error comes	Error Should come	P
202	Enter the case sensitive Password click on Submit button	Password	Accept	Accept	P
301	Enter the case insensitive Mobile Number	Mobile Number	Error comes	Error Should come	P

	click on Submit button				
302	Enter the case sensitive Mobile Number click on Submit button.	Mobile Number	Accept	Accept	P

Table1: Test Cases

Module-ID:-2

Modules to be tested:- Login

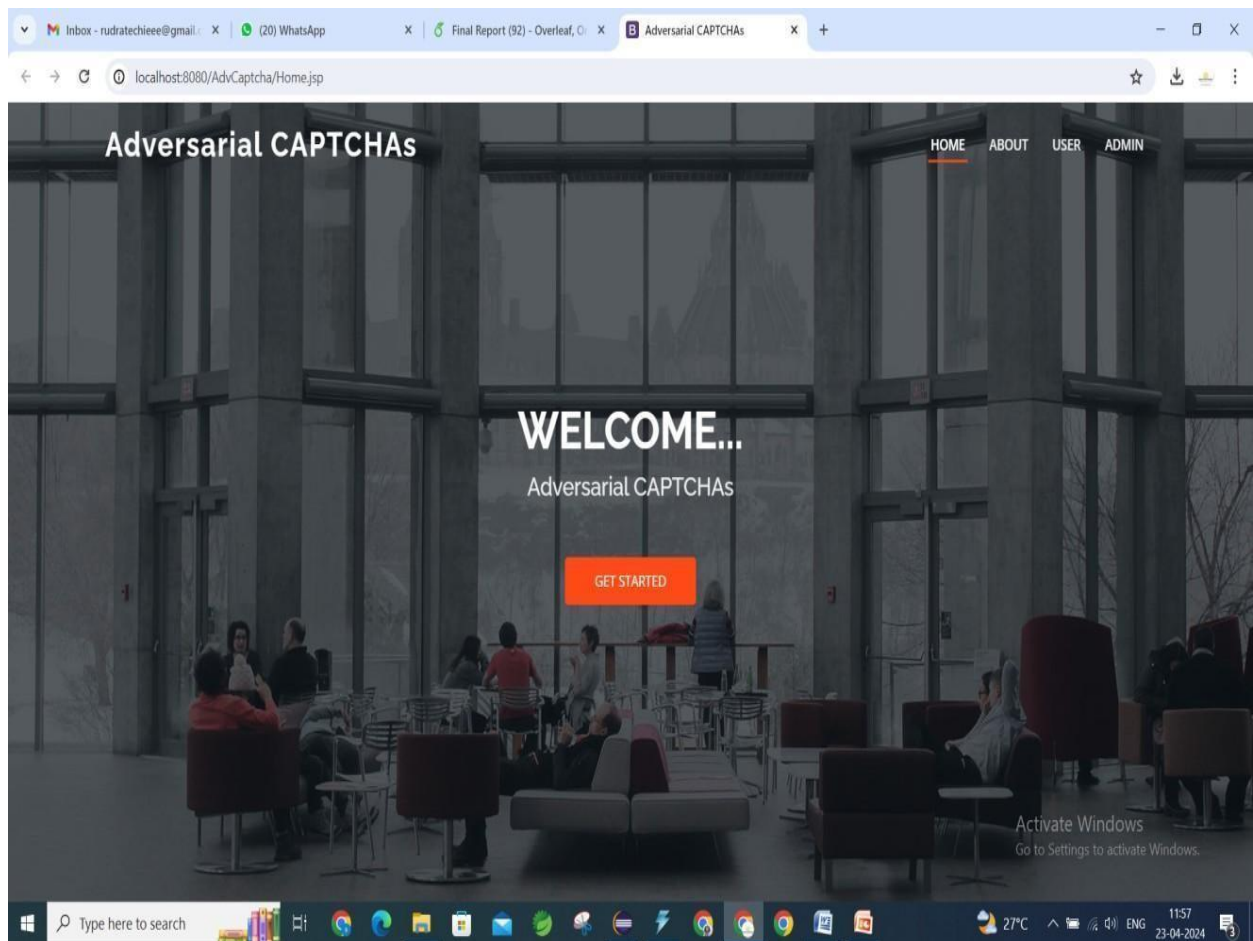
1. Enter the correct username and wrong password click on the Submit button.  
Expected: It should display errors.
2. Enter the wrong username and correct password and click on the Submit button.  
Expected: It should display errors.
3. Enter the correct username and password and click on the Login button.  
Expected: It should display a welcome page.
4. After login with valid credentials click on the back button.  
Expected: The page should be expired.
5. After login with valid credentials, copy the URL and paste in another browser.  
Expected: It should not display the user's welcome page.
6. Check the password with lowercase and uppercase.  
Expected: Password should be case sensitive.

Test Case_ID	Description	Test case I/P	Actual Result	Expected result	Test case criteria (P/F)
001	Enter the correct username and wrong password click onLogin button.	Username Password	Error comes	Error Should come	P
002	Enter the wrong username and correct password click onLogin button,	Username Password	Error comes	Error Should come	P
003	Enter the correct username and password and click on Login button.	Username Password	Accept	Accept	P

Table 2: Test Cases



## 6.1 output Screens:



Inbox - rudratechieee@gmail.com X (20) WhatsApp X Final Report (92) - Overleaf, O X Adversarial CAPTCHAs X +

localhost:8080/AdvCaptcha/StudentRegistration.jsp

## Adversarial CAPTCHAs

[HOME](#) [ABOUT](#) [USER](#) [ADMIN](#)

REGISTER HERE

Your name

Your address

dd-mm-yyyy


Your Mobile Number

Your Email

Your password

REGISTER

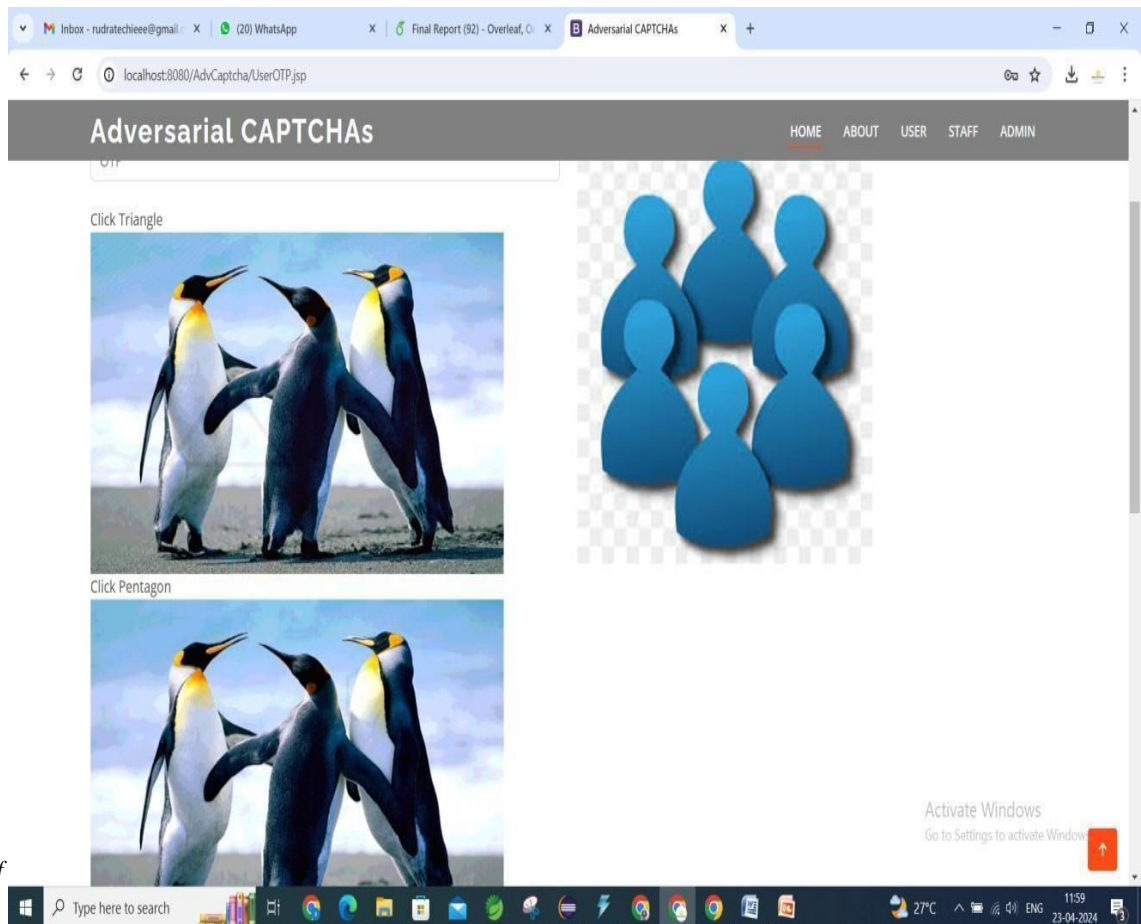
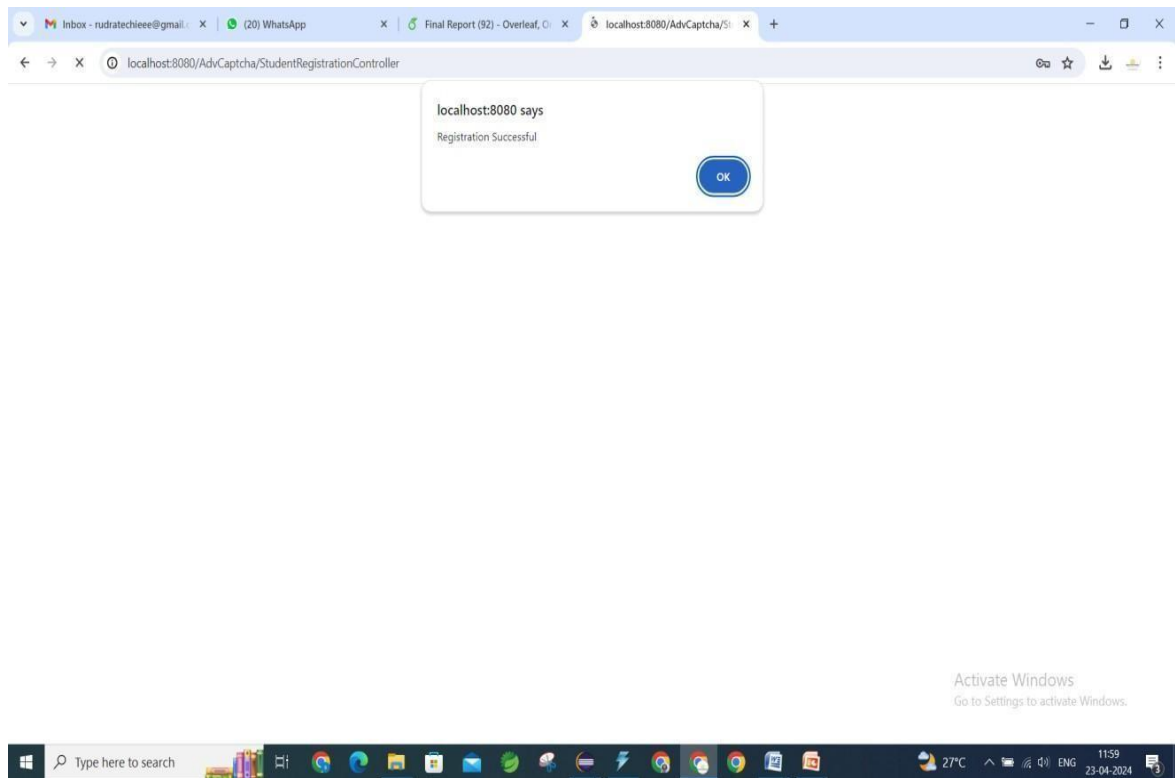
Login here!!!

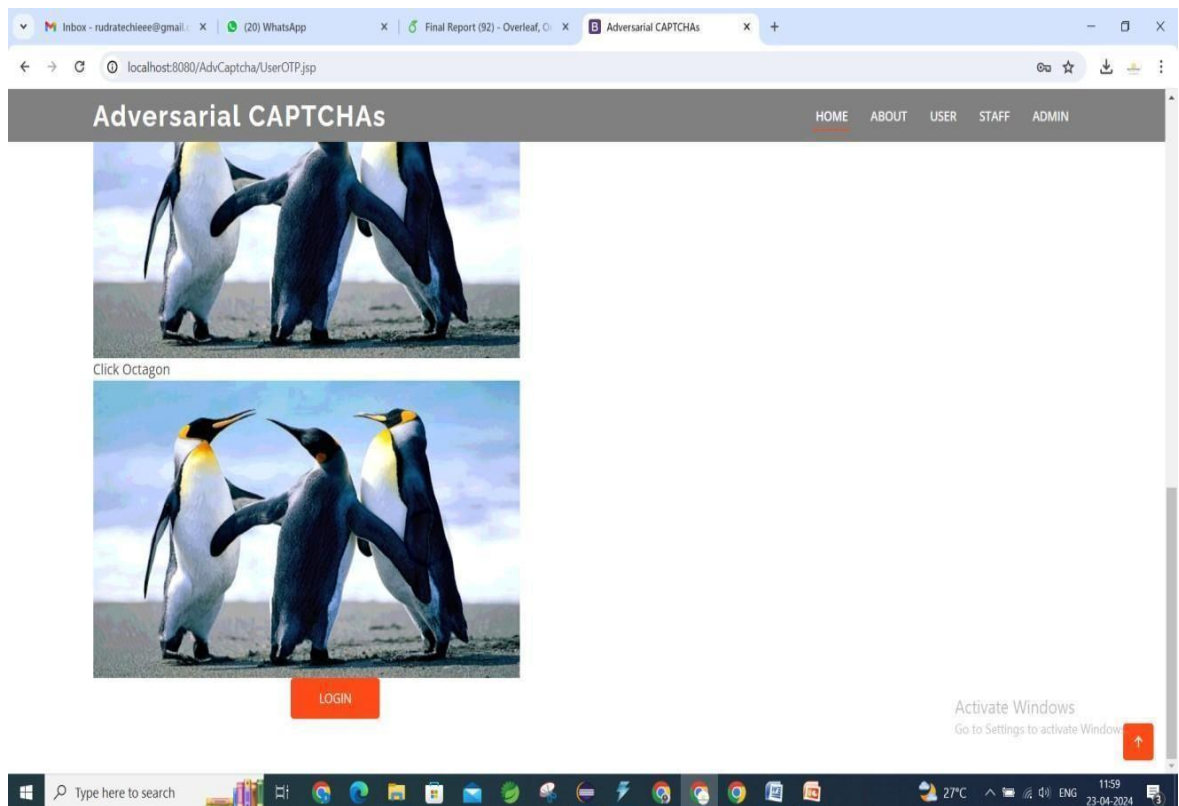
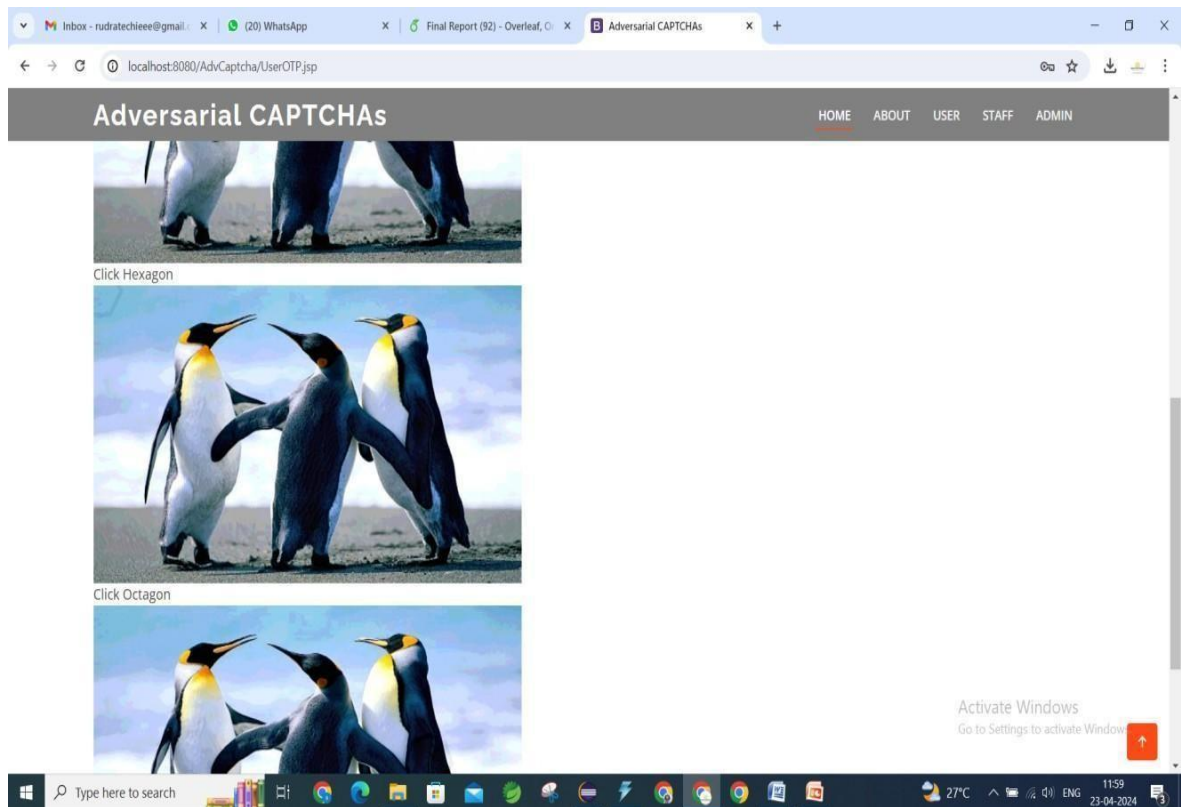


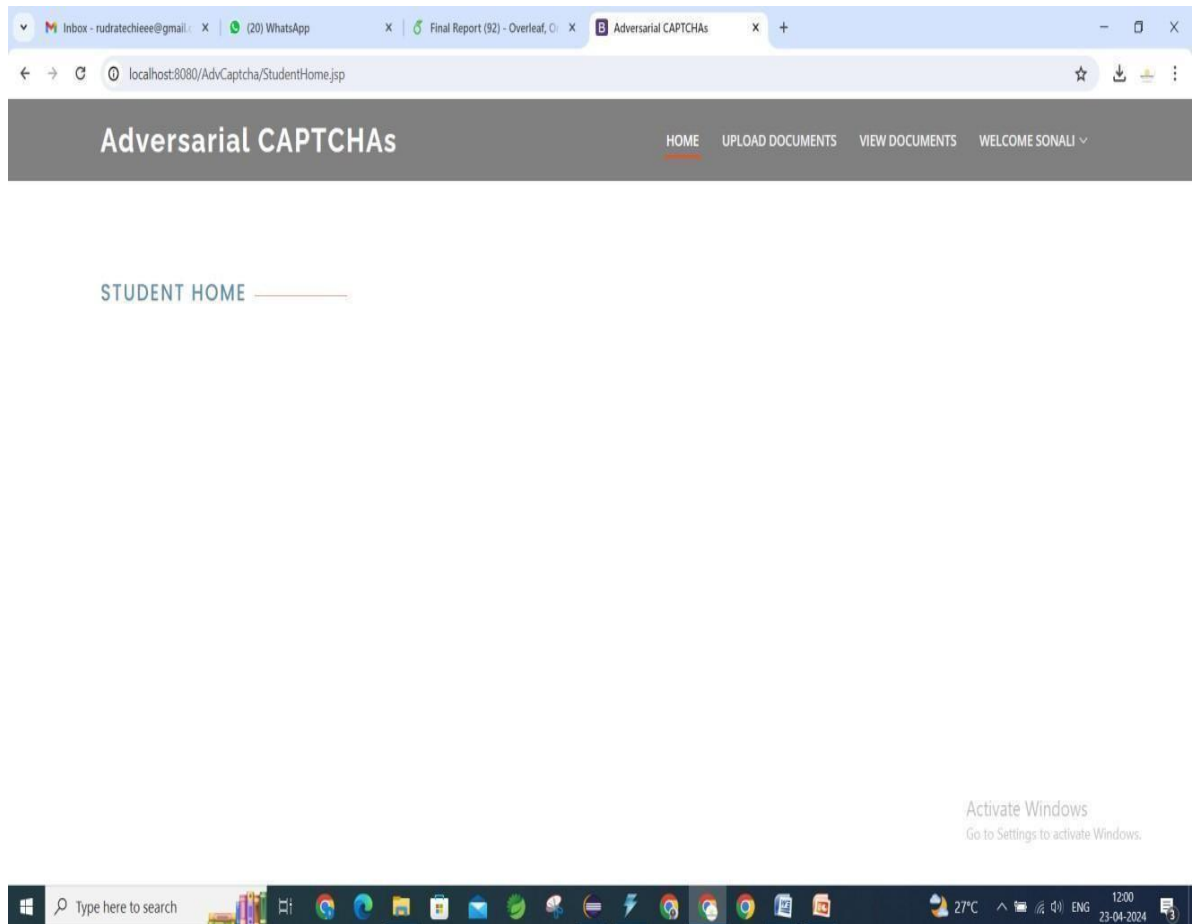
Activate Windows  
Go to Settings to activate Windows.

Type here to search

27°C 11:58 23-04-2024







## 7. ADVANTAGES AND DISADVANTAGES

### 7.1. Advantages:

- **High Security:** CNN-based CAPTCHA systems are often more secure than traditional text-based CAPTCHAs, as they can effectively differentiate between humans and bots by analyzing complex visual patterns.
- **Resistance to OCR:** Convolutional Neural Networks (CNNs) are proficient at recognizing patterns and features within images, making it difficult for Optical Character Recognition (OCR) software to bypass the CAPTCHA.
- **Customization:** With Java, you have the flexibility to customize and fine-tune the CAPTCHA system according to your specific security requirements and the aesthetics of your website.
- **Scalability:** Once implemented, CNN-based CAPTCHA systems can scale well with the growth of your website, efficiently handling increased traffic without compromising security.
- **Enhanced User Experience:** Compared to traditional text-based CAPTCHAs, image-based CAPTCHAs can provide a more user-friendly experience, as they often involve simpler interactions like selecting images or identifying objects.

### 7.2. Disadvantages:

- **Resource Intensive:** Training and deploying CNN models can be computationally intensive, especially if you're dealing with a large number of images or complex neural network architectures. This could potentially increase the server load and response times.
- **False Positives/Negatives:** While CNNs are generally effective at distinguishing between humans and bots, there's still a possibility of false positives (humans being incorrectly identified as bots) or false negatives (bots successfully bypassing the CAPTCHA).
- **Accessibility Concerns:** Image-based CAPTCHAs might pose challenges for users with visual impairments or those using assistive technologies, potentially excluding certain segments of the user population.

- **Development Complexity:** Implementing a CNN-based CAPTCHA system requires expertise in both machine learning (for designing and training the model) and web development (for integrating it into the website), which could increase the complexity of the project.
- **Maintenance Overhead:** Keeping the CAPTCHA system up-to-date with evolving security threats and maintaining the trained CNN model might require ongoing effort and resources.

Overall, while a CNN-based image CAPTCHA system in a Java web development project can offer enhanced security and user experience, it's essential to carefully consider the potential challenges and invest appropriately in development, maintenance, and user accessibility.

## 8. APPLICATIONS

- **Website Security:** Implementing the CAPTCHA system can enhance the security of websites by preventing automated bots from performing malicious activities such as spamming forms, creating fake accounts, or conducting brute-force attacks on login pages.
- **User Authentication:** Integrate the CAPTCHA system as part of the user authentication process to verify that the entity attempting to log in is a human user rather than a bot. This adds an extra layer of security to protect user accounts from unauthorized access.
- **Form Submission Verification:** Require users to complete the CAPTCHA challenge before submitting sensitive forms, such as contact forms, registration forms, or payment forms. This helps ensure that only genuine human users are interacting with the website.
- **Content Access Control:** Use the CAPTCHA system to control access to certain content or features on the website. For example, users may need to solve a CAPTCHA puzzle before accessing premium content, downloading files, or performing specific actions.
- **Transaction Security:** Implement CAPTCHA challenges during online transactions, such as making purchases or transferring funds. This helps prevent automated bots from exploiting vulnerabilities in the payment process or performing fraudulent transactions.
- **Bot Mitigation:** Employ the CAPTCHA system as part of a broader bot mitigation strategy to identify and block malicious bots attempting to scrape website content, manipulate rankings, or carry out denial-of-service attacks.
- **Account Registration:** Require users to solve a CAPTCHA challenge during the registration process to verify that they are real individuals and not automated scripts attempting to create multiple fake accounts.
- **Comment Spam Prevention:** Use the CAPTCHA system to prevent automated bots from posting spam comments on blogs, forums, or discussion boards. Requiring users to solve a CAPTCHA challenge before submitting a comment helps maintain the quality and integrity of the discussion platform.
- **Data Protection Compliance:** Implementing CAPTCHA challenges can assist in achieving compliance with data protection regulations by adding an additional layer of security to sensitive data processing activities, such as user authentication or form submissions.
- **Brand Protection:** Protect the reputation and brand integrity of the website by deterring automated bots from engaging in activities that could tarnish the website's image, such as posting inappropriate content, spreading malware, or engaging in fraudulent activities.



## **9. CONCLUSIONS & FUTURE WORK**

The proposed a novel image-based Captcha named SACaptcha using neural style transfer techniques. Most early image-based Captchas are based on the problem of image classification, whereas SACaptcha relies on problems of semantic information understanding and pixel-level segmentation. This is a positive attempt to improve the security of Captchas by utilizing deep learning techniques. The future work will promote more-effective ways to enhance the security of text Captchas.

## 11.REFERENCES

- [1] H. Gao, W. Wang, J. Qi, X. Wang, X. Liu, and J. Yan, "The robustness of hollow captchas," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 1075–1086.
- [2] H. Gao, M. Tang, Y. Liu, P. Zhang, and X. Liu, "Research on the security of microsoft's two-layer captcha," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 7, pp. 1671–1685, 2017.
- [3] E. Bursztein, J. Aigrain, A. Moscicki, and J. C. Mitchell, "The end is nigh: Generic solving of text-based captchas." in *WOOT*, 2014.
- [4] D. C. Ciresan, U. Meier, J. Masci, L. Maria Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, vol. 22, no. 1. Barcelona, Spain, 2011, p. 1237.
- [5] O. Starostenko, C. Cruz-Perez, F. Uceda-Ponga, and V. Alarcon-Aquino, "Breaking text-based captchas with variable word and character orientation," *Pattern Recognition*, vol. 48, no. 4, pp. 1101–1112, 2015.
- [6.] Hitaj, Dorjan & Hitaj, Briland & Jajodia, Sushil & Mancini, Luigi. (2020). Capture the Bot: Using Adversarial Examples to Improve CAPTCHA Robustness to Bot Attacks. 10.1109/MIS.2020.3036156
- [7.] G., DHANALAKSHMI & Devadarshini, S & Sri, R & Srinidhi, R. (2021). CAPTCHA SECURITY ENGINE. 8. 216-21
- [8.] S. Awasthi, A. P. Srivastava, S. Srivastava and V. Narayan, "A Comparative Study of Various CAPTCHA Methods for Securing Web Pages," 2019 International Conference on Automation, Computational and Technology Management (ICACTM), London, UK,

2019, pp. 217-223, doi: 10.1109/ICACTM.2019.8776832.

- [9.] Osadchy, Margarita & Hernandez-Castro, Julio & Gibson, Stuart & Dunkelman, Orr & Perez-Cabo, Daniel. (2017). No Bot Expects the DeepCAPTCHA! Introducing Immutable Adversarial Examples, with Applications to CAPTCHA Generation. IEEE Transactions on Information Forensics and Security. PP. 1-1. 10.1109/TIFS.2017.2718479.
  
- [10.] Shao, Rulin & Shi, Zhouxing & Yi, Jinfeng & Chen, Pin-Yu & Hsieh, Cho-Jui. (2022). Robust Text CAPTCHAs Using Adversarial Examples. 1495-1504. 10.1109/BigData55660.2022.10021100. CAPTCHA forVisually Impaired Users. 10.13140/RG.2.2.25923.22560.