**SAVITRIBAI PHULE PUNE UNIVERSITY**

**A PROJECT REPORT ON**

# Devnagari CAPTCHA Generator

SUBMITTED TOWARDS THE
PARTIAL FULFILLMENT OF THE REQUIREMENTS OF

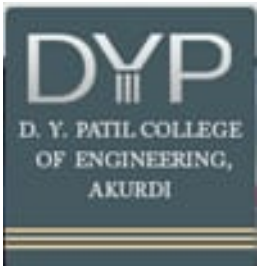
**BACHELOR OF ENGINEERING (Computer Engineering)**

BY


| | |
|---|---|
| BHAVANA THAKARE | 72018608M |
| MRUNAL SADAWARTE | 72018470D |
| SAKSHI GHONGE | 72018080F |
| ABHIRAM SHELKAR | 72017869L |

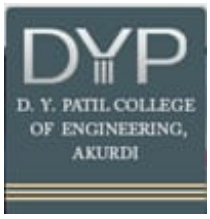

**Under The Guidance of**

Mrs. Yasmin Khan
Mrs. Soudamini Somvanshi





**DEPARTMENT OF COMPUTER ENGINEERING**
**D. Y. Patil College of Engineering,**
**Akurdi.**




**D. Y. Patil College of Engineering, Akurdi.**
**DEPARTMENT OF COMPUTER ENGINEERING**

# CERTIFICATE

This is to certify that
**Bhavana Thakare(72018608M), Mrunal Sadawarte(72018470D), Sakshi Ghonge (72018080F), Abhiram Shelkar(72017869L)** has satisfactorily completed the project work entitled "Devnagari CAPTCHA Generator" which is bonafide work carried out by him/ her under the supervision of Mrs. Soudamini Sowmvanshi and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of Bachelors of Engineering (Computer Engg.) for the academic year 2022-2023.

Mrs.Yasmin Khan
Co.Guide
Dept. of Computer Engg.

Mrs.Soudamini Somvanshi
Guide
Dept. of Computer Engg.

Dr. Mrs.Madhuri A. Potey
HOD
Dept. of Computer Engg.

Dr. Mrs P. Malathi
Principal
D. Y. Patil College Of Engineering

Signature of Internal Examiner

Signature of External Examiner

**PROJECT APPROVAL SHEET**

A Project

Devnagari CAPTCHA Generator

is successfully completed by


| | |
|---|---|
| BHAVANA THAKARE | 72018608M |
| MRUNAL SADAWARTE | 72018470D |
| SAKSHI GHONGE | 72018080F |
| ABHIRAM SHELKAR | 72017869L |


at


(D. Y. Patil College of Engineering, Akurdi.)

DEPARTMENT OF COMPUTER ENGINEERING


SAVITRIBAI PHULE PUNE UNIVERSITY,PUNE

ACADEMIC YEAR 2022-2023

Mrs. Yasmin Khan
Mrs. S. T. Somvanshi
Dept. of Computer Engg.

H.O.D
Dr. Mrs.M. A. Potey
Dept. of Computer Engg.

## Abstract

An alternative methodology for the generation of CAPTCHA which can be used as a language barrier for people who can only understand their native language. The proposed methodology involves a system of text-based CAPTCHA that is based on Devanagari script which is written in the form of Indian languages that is Hindi and Marathi.

The process of creating abstract texts that use the Devanagari script, which is an essential part of many Asian languages such as Hindi, Marathi, Nepali, etc. This report highlights the potential benefits of adding a Devanagari language in the generation of CAPTCHAs.

It is implemented for the better understanding of Indian people. This system will allow people to understand and use captchas in rural parts. Users may use this in any form of login resulting in the swift working of the process.

# Acknowledgments

It give us great pleasure in presenting the preliminary project report on **'Devnagari CAPTCHA Generator'.**

I would like to take this opportunity to thank my internal guide **Mrs. Soudamini Somvanshi** for giving me all the help and guidance I needed. I am really grateful to her for her kind support. Her valuable suggestions were very helpful.

I am also grateful to **Dr. Mrs. M. A. Potey**, Head of Computer Engineering Department, DYPCOE for his indispensable support, suggestion.

In the end our special thanks to **Mrs.Yasmin Khan** for providing various resources such as a laboratory with all needed software platform, continuous Internet connection, for our Project.

Bhavana Thakare
Mrunal Sadawarte
Sakshi Ghonge
Abhiram Shelkar
(B.E. Computer Engg.)

# Contents

# List of Figures

# List of Tables

# Synopsis

## 1.1 Project Title

Devnagari CAPTCHA Generator

## 1.2 Project Option

BE Final Year Project

## 1.3 Internal Guide

Mrs.Yasmin Khan and Mrs. Soudamini Somvanshi

## 1.4 Sponsorship and External Guide

Not Sponsored

## 1.5 Technical Keywords (As per ACM Keywords)

1. CAPTCHA Generation
2. CAPTCHA Verification
3. Challenge-response system
4. Distortion
5. Noise
6. Segmentation
7. Randomization
8. Security

## 1.6 Problem Statement

The goal of this project is to develop a Devanagari language-based CAPTCHA generator that effectively distinguishes between humans and automated bots.The CAPTCHA should present users with distorted Devanagari characters or words and require them to correctly identify or enter the corresponding text. The generator must ensure that the CAPTCHAs are challenging for bots to solve while remaining legible for human users.

## 1.7 Abstract

Devanagari Captcha Generation is a System that generates Captcha in Hindi/Marathi.It is a System of text-based CAPTCHA that is based on the Devanagari script .It is written in the form of Indian languages Hindi and Marathi.The objective of a Devnagari CAPTCHA System is to differentiate a human from a Bot. It can also be used as a language barrier for people who can only understand Hindi and Marathi.It is implemented for the better understanding of Indian people. This system will allow people to understand and use captchas in rural parts.Users may use this in any form of login resulting in the swift working of the process.This System provides guidelines to improve security control and storage of Captcha methods against various possible attacks and guidelines to improve their usability.

## 1.8 Goals and Objectives

The main goals and objectives of the system are as follows:

1. The primary purpose of using captchas is to prevent automated bots from accessing a website, and allowing legitimate users to gain access without difficulty.

2. Captchas help to ensure that online services remain secure and prevent unauthorized access, such as spamming, scraping, and other malicious activities.

## 1.9 Relevant mathematics associated with the Project

Begin:
Randomly pick a number n = Random();
// captcha module is used
adv(I, Cd , p) = I + {
The immutable adversarial example
}

Discard I;
Randomly select m  1 different indexes j1,..., jm1 from[1,..., n]
Choose the representative images [Ij1 ,..., Ijm1 ] of the corresponding classes
Output: a random permutation of m possible answers {I, Ij1 ,..., Ijm1 }.

## 1.10 Names of Conferences / Journals where papers can be published

- IEEE/ACM Conference/Journal
- Conferences/workshops in IITs
- Central Universities or SPPU Conferences
- IEEE/ACM Conference/Journal 2

## 1.11 Review of Conference/Journal Papers supporting Project idea

1. **Name:** Robust CAPTCHAs Towards Malicious OCR.
   **Standard:** IEEE
   **Author:** [1] Zhang, Jiaming and Sang, Jitao and Xu, Kaiyuan and Wu, Shangxi and Zhao, Xian and Sun, Yanfeng and Hu.
   **Year:** 2020
2. **Name:** A Security Analysis of Captchas With Large Character Sets.
   **Standard:** IEEE Xplore
   **Author:** [2] Ping Wang , Haichang Gao , Member, IEEE, Qingxun Rao ' , Sainan Luo, Zhongni Yuan , and Ziyu Shi.
   **Year:** 2019
3. **Name:** Vulnerabilities of Existing Design, and Countermeasure.
   **Standard:** IEEE
   **Author:** [3] Song Gao, Manar Mohamed,Nitesh Saxena and Chengcui Zhang.
   **Year:** 2018
4. **Name:** Simple and Easy: Transfer Learning-Based Attacks to text CAPTCHA
   **Standard:** IEEE
   **Author:** Wang,Haichang Gao , Ziyu Shi,Zhongni, Jiangping Hu
   **Year:** 2020

5. **Name:** An End-to-End Attack on Text CAPTCHAs.
   **Standard:** IEEE
   **Author:** Yang Zi, Haichang Gao , Member, IEEE, Zhouhang Cheng
   **Year:** 2017
6. **Name:** Selective Learning Confusion Class for Text-Based-CAPTCHA Recognition
   **Standard:** IEEE
   **Author:** Jun Chen, Xiangyang Luo,Yingying Liu, Jinwei Wang and Yuanyuan Ma
   **Year:** 2017

## 1.12 Plan of Project Execution

| Topic / Module | Module Head | Current Status | Plan of Completion |
|---|---|---|---|
| Character Recognition | Mrunal | Completed | August 2021 |
| CAPTCHA Generation | Mrunal | Completed | September 2021 |
| Image Creation | Bhavana | Completed | October 2021 |
| Audio Generation | Bhavana | Completed | October 2021 |
| CAPTCHA Verification | Abhiram | Completed | November 2021 |
| Database Creation | Abhiram | Completed | December 2021 |
| User Interface | Sakshi | Completed | March 2022 |
| Documentation | All Members | Completed | April 2022 |

Table 1.1: Project Planner

# Technical Keywords

## 2.1 Area of Project

CAPTCHA Generation Algorithm, CAPTCHA Security Analysis, Accessibility in Devanagari CAPTCHA, Multi-lingual CAPTCHA Generation, Real-time CAPTCHA Generation and Verification, CAPTCHA Security Enhancement, Human Solvability Evaluation.

## 2.2 Technical Keywords

1. CAPTCHA Generation
2. CAPTCHA Verification
3. Challenge-response system
4. Distortion
5. Noise
6. Segmentation
7. Randomization
8. Security

# Introduction

Devanagari CAPTCHA generation refers to the process of creating CAPTCHAs (Completely Automated Public Turing test to tell Computers and Humans Apart) using Devanagari script. Devanagari is an important script used for writing several Indian languages, including Hindi, Marathi, Nepali, and Sanskrit. CAPTCHAs are widely used on websites to differentiate between human users and automated bots.

The main objective of Devanagari CAPTCHA generation is to design a system that generates visually distorted and challenging Devanagari characters or words, which can be easily recognized by humans but are difficult for automated algorithms to decipher accurately. This helps ensure the security and integrity of online systems by preventing automated bots from spamming or performing malicious activities.

Devanagari CAPTCHAs typically involve the random selection and distortion of characters from the Devanagari script. The distortion techniques can include warping, rotation, noise addition, line interference, or other modifications to make it harder for automated character recognition algorithms to identify the characters. The generated CAPTCHAs are then presented to users, who are required to enter the correct characters or words to prove that they are human users section0

## 3.1 Project Idea

The goal of this project is to develop a Devanagari language-based CAPTCHA generator that effectively distinguishes between humans and automated bots.The CAPTCHA should present users with distorted Devanagari characters or words and require them to correctly identify or enter the corresponding text. The generator must ensure that the CAPTCHAs are challenging for bots to solve while remaining legible for human users.

## 3.2 Motivation of the Project

1. Language-specific protection: Devanagari captchas can be used to protect websites and online services that primarily cater to users who use the Devanagari script, such as websites in Hindi, Marathi, Nepali, and other languages that use the Devanagari script. By implementing Devanagari captchas, you can ensure that only users who are familiar with the script can access the services, adding an extra layer of security.

2. Accessibility: Devanagari captchas can contribute to making online services more accessible for individuals with visual impairments who use screen readers. By providing captchas in Devanagari, these users can rely on their screen readers to accurately vocalize the content, improving their ability to access and use online platforms.

3. Localized security: Using Devanagari captchas can be beneficial in regions where English may not be widely understood. By implementing captchas in the local script, you ensure that security measures are aligned with the local user base and minimize potential confusion or barriers caused by language differences

4. Cultural relevance: Devanagari captchas can help promote and preserve the cultural and linguistic identity of communities that use the Devanagari script. By incorporating captchas in their native script, websites can create a more inclusive and familiar user experience for users from those communities

5. Improved user experience: Captchas are often used to prevent automated bots and malicious activities, but they can sometimes be challenging for users to decipher. By using Devanagari captchas for Devanagari-speaking users, you can provide a more user-friendly experience, as the users will be more accustomed to reading and understanding the script, resulting in quicker and more accurate completion of the captchas.

## 3.3 Literature Survey

| Sr. No | Conference, and Publication year | Title | Description |
|---|---|---|---|
| 1 | IEEE 2020 | Robust CAPTCHAs Towards Malicious OCR | Due to the complexity of OCR model, compared with the common CNN model, standard adversarial training does not show the effectiveness as usual. |
| 2 | IEEE 2019 | A Security Analysis of Captchas With Large Character Sets | The rotation mechanism used in this Captcha scheme negatively impacts the recognition accuracy. A slight difference between synthetic samples and real data also affects the recognition accuracy. |
| 3 | IEEE 2018 | Vulnerabilities of Existing Design, and Countermeasure | Each captcha challenge loops continuously within a short video to limit the captcha file size and prevent collection of codeword contours. Since each frame is a binary image only black pixels are super imposed to form clearer shape. |

# Literature Survey

| Sr. No | Conference, and Publication year | Title | Description |
|---|---|---|---|
| 4 | IEEE 2019 | Simple and Easy: Transfer Learning-Based Attacks to text CAPTCHA | Using deep leaming techniques, they have tested 20 Roman character-based schemes and 5 Chinese schemes and achieved remarkably good success rates, ranging from 36.3 to 96.9 percent at an average attack speed 0.02 seconds per CAPTCHA. Most of the tested schemes achieved success rates of 0 percent. |
| 5 | IEEE 2017 | An End-to-End Attack on Text CAPTCHAs | The variation of character fonts increased the difficulty of the recognition process. This indicates that resisting recognition, rather than resisting segmentation, will be the likely development direction in the future |

# Literature Survey

| Sr. No | Conference, and Publication year | Title | Description |
|---|---|---|---|
| 6 | IEEE 2017 | Selective Learning Confusion Class for Text-Based-CAPTCHA Recognition | They Test all the class DCNNs with initial testing and then analyse the output by judging whether the prediction result of a testing sample is in the confusion class. In Bot Detect CAPTCHA, the number of recognized error classes is large and scattered |
| 7 | Springer 2016 | Research on Deep Learning Techniques in Breaking Text-Based Captchas and Designing Image-Based Captcha | They proposed the following methodology :- Generate style-transferred images Synthesize the background : Generate the Captcha: Generate a description. . |
| 8 | Solar 2016 | No Bot Expects the Deep CAPTCHA! Introducing Immutable Adversarial Examples, With Applications to CAPTCHA Generation | The proposed system can operate on the set of labels they have been trained for. Switching to an alternative set of labels is likely to reduce their effectiveness. |

# Literature Survey

| Sr. No | Conference, and Publication year | Title | Description |
|---|---|---|---|
| 9 | IEEE 2015 | Captcha as Graphical Passwords—A New Security Primitive Based on Hard AI Problems. | In entering a CAPTCHA, a user-clicked point is replaced by the grid-cell it lies in. If click errors are in it ,each user-clicked point falls into the same grid-cell as the original CAPTCHA . Therefore the sequence of grid-cells generated from user-clicked points is identical to the one that the authentication server. |
| 10 | ACM 2014 | Chinese Character CAPTCHA Recognition Based on Convolution Neural Network | In this paper they have studied the efficiency of model training, by conducting experiments to learn how to optimize the training time and to show the effects of different training strategies. Their attack provides a more promising strategy that not only reduces the attack complexity and manual-labeling cost but also preserves comparable accuracy. |

Table 3.1: Literature Survey

# Problem Definition and scope

## 4.1 Problem Statement

The goal of this project is to develop a Devanagari language-based CAPTCHA generator that effectively distinguishes between humans and automated bots.The CAPTCH- A should present users with distorted Devanagari characters or words and require them to correctly identify or enter the corresponding text. The generator must ensure that the CAPTCHAs are challenging for bots to solve while remaining legible for human users.

### 4.1.1 Goals and objectives

The main goals and objectives of the system are as follows:

1. The primary purpose of using captchas is to prevent automated bots from accessing a website, and allowing legitimate users to gain access without difficulty.
2. Captchas help to ensure that online services remain secure and prevent unauthorized access, such as spamming, scraping, and other malicious activities.

### 4.1.2 Statement of scope

The scope of this project encompasses the development and evaluation of a Devanagari Captcha generation system with the objective of enhancing security, user experience, and inclusivity for Devanagari-speaking users. The project will focus on the following key aspects:

1. Design and develop an efficient and secure algorithm for generating Devanagari Captchas
2. Research and analyze existing Devanagari Captcha generation techniques.
3. Ensure the generated captchas are visually clear, readable, and resistant to automated attacks

User Experience Evaluation:

1. Conduct user studies and feedback surveys to assess the usability and effectiveness of the Devanagari Captcha system.
2. Measure the accuracy and efficiency of users in completing Devanagari Captchas compared to other captcha types.
3. Gather feedback on the clarity and comprehensibility of the generated captchas.

## 4.2 Methodologies of Problem-solving and efficiency issues

**Methodologies of Problem-Solving:**

- Define the Problem: Clearly articulate and understand the problem statement related to Devanagari Captcha generation, user experience, accessibility, or integration. Identify the specific challenges and constraints involved.
- Research and Analysis: Conduct thorough research and analysis of existing techniques, algorithms, and best practices related to Devanagari Captcha generation, usability, accessibility, and integration. Study academic papers, industry standards, and relevant resources to gain insights into effective solutions.
- Brainstorming and Ideation: Engage in brainstorming sessions to generate innovative ideas and potential solutions to the identified problem. Encourage collaboration and diverse perspectives to explore different possibilities.
- Design and Prototyping: Develop a systematic and well-thought-out design for the Devanagari Captcha system. Create prototypes or mock-ups to visualize and refine the proposed solution. Iterate on the design based on feedback and evaluation.
- Implementation and Testing: Implement the designed solution, considering efficiency, security, and usability. Conduct rigorous testing to identify and resolve any issues or bugs. Test the system under various scenarios and validate its effectiveness.
- Evaluation and Feedback: Evaluate the implemented solution through user studies, feedback surveys, and performance metrics. Gather feedback from users, experts, and stakeholders to assess the efficiency, effectiveness, and user satisfaction of the Devanagari Captcha system.
- Iterative Improvement: Based on the evaluation results and feedback received, make iterative improvements to the system. Address any identified efficiency issues, security vulnerabilities, usability challenges, or accessibility concerns.
- Efficiency Issues: Algorithm Optimization: Ensure that the Devanagari Captcha generation algorithm is optimized for efficiency. Consider techniques like caching, precomputation, or parallelization to reduce computational overhead and improve response times.
- Resource Management: Efficiently manage system resources such as memory, processing power, and storage to minimize resource consumption. Avoid unnecessary redundancy or inefficient data structures that can impact performance.

- Scalability and Performance: Design the system to handle increased user load and scale horizontally or vertically as needed. Implement performance monitoring and profiling to identify bottlenecks and optimize critical components.
- Minimize Network Latency: Optimize network communication and reduce latency by employing techniques such as data compression, content delivery networks (CDNs), or efficient data transfer protocols.
- Caching and Response Caching: Utilize caching mechanisms to store and reuse frequently accessed data, such as generated captchas or validation results. This can significantly reduce the computational overhead and improve response times.
- Code Efficiency: Write clean and efficient code by following best practices, employing appropriate data structures and algorithms, and avoiding unnecessary computations or redundant operations.
- Continuous Monitoring and Optimization: Regularly monitor system performance, analyze metrics, and identify areas for optimization.

## 4.3 Outcome

- Developed Devanagari Captcha Generation System: The project will result in the development of an efficient and secure algorithm for generating Devanagari Captchas. The system will be capable of producing visually clear and readable captchas that are resistant to automated attacks.
- Improved User Experience: The Devanagari Captcha system will enhance the user experience for Devanagari-speaking individuals. Users will find the captchas more familiar and easier to read, leading to quicker and more accurate completion of the captchas.
- Enhanced Accessibility: The project will address accessibility concerns by ensuring compatibility with screen readers and other assistive technologies. Visually impaired users will be able to perceive and understand the Devanagari Captchas through alternative means, improving their accessibility to online platforms.
- Increased Security: The generated Devanagari Captchas will provide a robust layer of security against automated attacks, such as OCR or machine learning algorithms. This will help protect websites and online services that implement Devanagari Captchas from malicious activities and unauthorized access.

- Adoption and Integration: The project aims to promote the adoption of Devanagari Captchas in relevant online platforms and websites. Collaborative efforts with developers and platform owners will result in the integration of the Devanagari Captcha system, expanding its usage and impact.

- Cultural Relevance and Preservation: By incorporating captchas in the Devanagari script, the project will contribute to the cultural relevance and preservation of Devanagari-speaking communities. It will create a more inclusive and familiar online environment for users from those communities.

- Documentation and Resources: The project will provide documentation and resources for easy implementation and customization of the Devanagari Captcha system. This will support developers and platform owners in implementing the captchas effectively and efficiently

## 4.4 Applications

1. Websites and Online Portals: Devanagari Captchas can be implemented in websites and online portals that provide content, services, or transactions in Devanagari-based languages such as Hindi, Marathi, Nepali, etc. This can include e-commerce platforms, news websites, educational portals, government websites, and more.

2. User Registration and Login: Devanagari Captchas can be used as an additional security measure during user registration and login processes. By verifying that users can accurately decipher the Devanagari captchas, it helps prevent automated bots or unauthorized access to user accounts.

3. Online Forms and Surveys: Devanagari Captchas can be integrated into online forms and surveys to ensure that responses are generated by genuine human users. This helps prevent spam submissions and ensures data integrity.

4. Account Recovery and Verification: Devanagari Captchas can be employed in account recovery or verification processes, such as password resets or email verifications. By including Devanagari captchas, adds an extra layer of security to validate the identity of users.

5. Online Voting and Polls: Devanagari Captchas can be utilized in online voting systems and polls conducted in Devanagari-based languages. This helps ensure that the participants are genuine human users, preventing automated voting or manipulation.

6. Mobile Applications: Devanagari Captchas can be incorporated into mobile applications that are designed for Devanagari-speaking users. This can include various types of apps such as language learning apps, news apps, social media apps, and more.
7. Content Accessibility: Implementing Devanagari Captchas can contribute to making online content more accessible to individuals who are fluent in Devanagari-based languages. It ensures that access to content, features, or services is limited to users who can read and understand the script.

## 4.5 Hardware Resources Required

1. Server - (Intel i7, 16GB RAM, 1 TB HDD)
2. PC - (Intel Pentium4, 4GB RAM, 500 GB HDD)

## 4.6 Software Resources Required

1. Programming Language : Python
2. Integrated Development Environment (IDE): Visual Studio
3. Image Processing Libraries:
4. Web Frameworks: Flask , ReactJs
5. Version Control: Git
6. Database Management System: MySql
7. Documentation and Collaboration Tools

# Project Plan

The project is planned to work in a waterfall model.

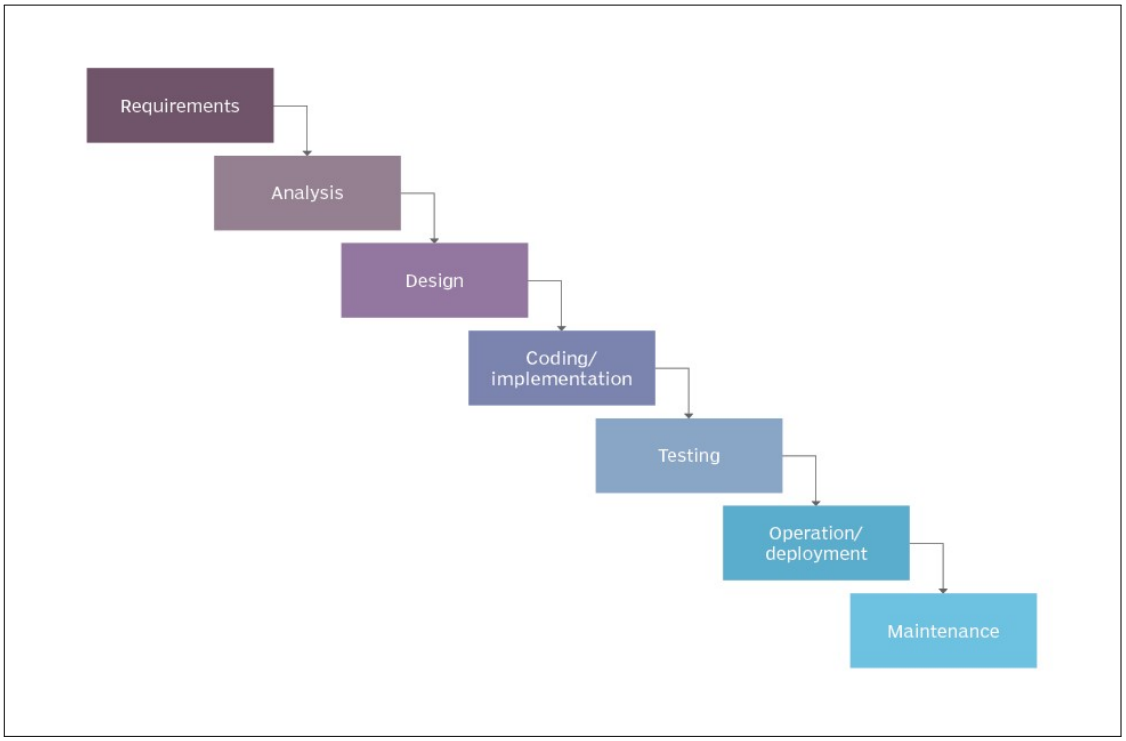



Figure 5.1: Waterfall Methodology

## 5.1 Project Estimates

### 5.1.1 Reconciled Estimates

**Cost Estimate**

Rs.2000 / Month (For EC2 VM)

**Time Estimates**

This project has been completed by April 2023.

### 5.1.2 Project Resources

#### 5.1.2.1 People:

1. Software Developer
2. Database Developer

3. UI Developer
4. Tester

#### 5.1.2.2 Hardware

1. CPU Speed (2GHz)
2. RAM 8 GB

#### 5.1.2.3 Software

1. Programming Language : Python
2. Integrated Development Environment (IDE): Visual Studio
3. Image Processing Libraries:
4. Web Frameworks: Flask , ReactJs
5. Version Control: Git
6. Database Management System: MySql
7. Documentation and Collaboration Tools

## 5.2 Risk Management with respect to NP-Hard analysis

**Technical Risks:**

- Risk: Inaccurate or inefficient captcha generation algorithm.
- Approach: Conduct thorough research and analysis of existing captcha generation algorithms. Implement and test multiple algorithms, evaluating their accuracy, efficiency, and suitability for Devanagari characters. Iterate and refine the algorithm based on performance evaluation.

**Security Risks:**

- Risk: Vulnerabilities in the captcha system exploited by automated bots
- Approach: Implement security measures such as character distortion, noise addition, or background variations to prevent automated recognition. Regularly monitor and update the captcha system to address emerging security threats. Conduct security audits and penetration testing to identify and rectify any vulnerabilities.

**Scalability Risks:**

- Risk: Inadequate scalability to handle increasing user demand.

- Approach: Design the system with scalability in mind from the beginning. Utilize scalable server architectures, distributed computing, and caching mechanisms. Monitor system performance and scalability regularly, and make necessary optimizations and infrastructure adjustments as the user load increases.

**Resource Risks:**

- Risk: Insufficient availability of development resources.

- Approach: Plan resource allocation carefully, considering the project timeline and required skill sets. Identify potential bottlenecks in resource availability and plan for backup resources or external support if necessary. Maintain clear communication and coordination among team members to optimize resource utilization.

### 5.2.1 Risk Identification

Risk identification is an essential step in project risk management. Here are some potential risks to consider when developing a Devanagari Captcha system:

1. **Technical Risks:** Inaccurate or inefficient captcha generation algorithm. Inadequate performance and scalability of the system. Compatibility issues with different browsers or devices. Inability to handle a large number of simultaneous users.

2. **Security Risks:** Vulnerabilities in the captcha system exploited by automated bots. Unauthorized access or breaches compromising user data. Inadequate encryption or data protection measures. Resource Risks: Insufficient availability of skilled development resources. Lack of necessary hardware or software infrastructure. Delays or disruptions due to external dependencies (e.g., third-party APIs, libraries).

3. **Integration Risks:** Challenges in integrating the captcha system into existing platforms or websites. Incompatibility with specific content management systems (CMS) or frameworks. Difficulties in aligning with the user experience and design of the target platform. Schedule Risks: Unrealistic project timeline or milestones. Dependencies on external factors or deliverables. Delays in requirements gathering or changes in project scope.

4. **Budget Risks:** Insufficient budget allocation for development, resources, and infrastructure. Unforeseen expenses arising during the project. Cost overruns due to changes in requirements or scope.

5. **User Acceptance Risks:** Poor usability or accessibility of the captcha system. Lack of user adoption or resistance to using captchas. Negative feedback or dissatisfaction from users.

6. **Legal and Compliance Risks:** Non-compliance with privacy regulations or data protection laws. Violation of intellectual property rights or copyright infringements. Failure to meet accessibility standards or regulations

### 5.2.2 Risk Analysis

| ID | Risk Discription | Probability | Impact |
|---|---|---|---|
| 1 | Security Risk | Low | High |
| 2 | Integration Risk | Low | Medium |

Table 5.1: Risk Analysis

## 5.3 Project Schedule

### 5.3.1 Project task set

1. Text Generation
2. Image Generation
3. Character Segmentation
4. Distortion and Noise Application
5. Accessibility Considerations
6. Randomization and Variation
7. Security Analysis
8. Usability and User Experience
9. Integration
10. Testing and Evaluation

### 5.3.2 Task network

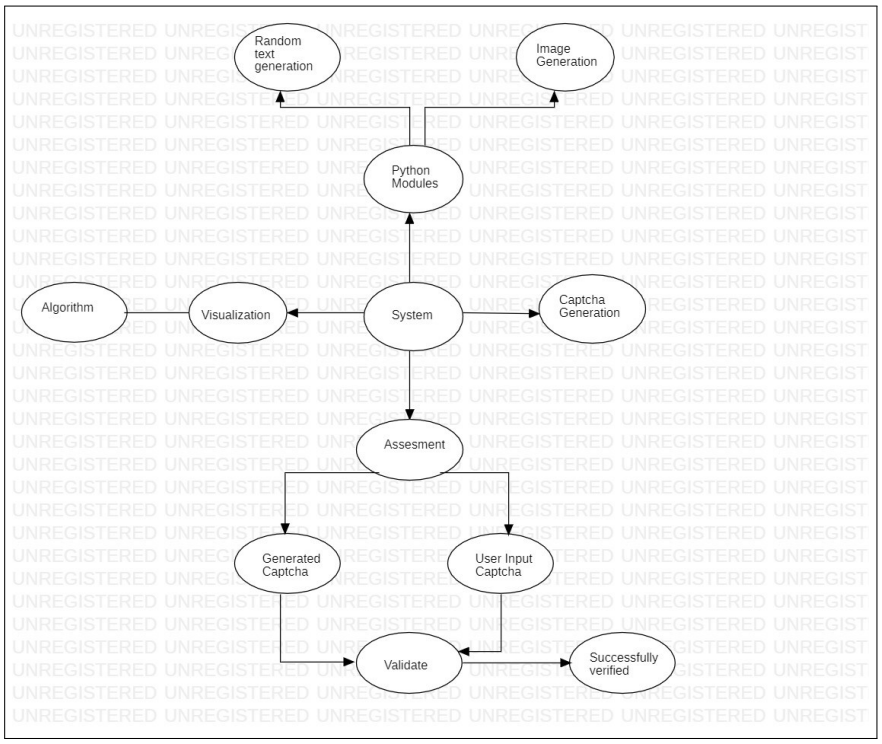



Figure 5.2: Task Network

### 5.3.3 Timeline Chart

| Topic / Module | Module Head | Current Status | Plan of Completion |
|---|---|---|---|
| Character Recognition | Mrunal | Completed | August 2021 |
| CAPTCHA Generation | Mrunal | Completed | September 2021 |
| Image Creation | Bhavana | Completed | October 2021 |
| Audio Generation | Bhavana | Completed | October 2021 |
| CAPTCHA Verification | Abhiram | Completed | November 2021 |
| Database Creation | Abhiram | Completed | December 2021 |
| User Interface | Sakshi | Completed | March 2022 |
| Documentation | All Members | Completed | April 2022 |

Table 5.2: Project Timeline

### 5.3.4 Team Organization

Project Guide: Mrs.S. T. Somvanshi
Project Co-Guide: Mrs.Yasmin Khan
Image and Audio Creation Head: Bhavana Thakare
Character and captcha generation Head : Mrunal Sadawarte
Captcha Verification Head:Abhiram Shelkar
UI/UX Head: Sakshi Ghonge

### 5.3.5 Team Structure

1. **Bhavana Thakare :** Responsible for Image recognition and Audio Generation.
2. **Mrunal Sadawarte :** Responsible for Character Recognition and CAPTCHA generation.
3. **Abhiram Shelkar :** Responsible for CAPTCHA verification and Database design.
4. **Sakshi Ghonge :** Responsible for User-Interface and Documentation.

### 5.3.6 Management reporting and communication

Dedicated Team of four developers working together and taking ownership of different actions to be performed with project progress is the attitude followed by following ways:

- Daily Scrums
- Inter team communications via virtual platforms
- Github Collaboration
- Guidance from Mentors on monthly basis

# Software requirement specification

## 6.1 Introduction

Devanagari CAPTCHA generation refers to the process of creating CAPTCHAs (Completely Automated Public Turing test to tell Computers and Humans Apart) using Devanagari script. Devanagari is an important script used for writing several Indian languages, including Hindi, Marathi, Nepali, and Sanskrit. CAPTCHAs are widely used on websites to differentiate between human users and automated bots.The main objective of Devanagari CAPTCHA generation is to design a system that generates visually distorted and challenging Devanagari characters or words, which can be easily recognized by humans but are difficult for automated algorithms to decipher accurately. This helps ensure the security and integrity of online systems by preventing automated bots from spamming or performing malicious activities.Devanagari CAPTCHAs typically involve the random selection and distortion of characters from the Devanagari script. The distortion techniques can include warping, rotation, noise addition, line interference, or other modifications to make it harder for automated character recognition algorithms to identify the characters. The generated CAPTCHAs are then presented to users, who are required to enter the correct characters or words to prove that they are human users

### 6.1.1 Purpose and Scope of Document

The purpose of Devanagari CAPTCHA is to differentiate between human users and automated bots or computer programs. It aims to ensure that certain actions or access to resources are performed only by humans, preventing malicious activities such as spamming, fraud, or unauthorized access.

- **User Verification:** Devanagari CAPTCHAs are used to verify that a user interacting with a system or website is human and not a bot.
- **Enhancing Security:** Devanagari CAPTCHAs contribute to the overall security of systems and applications by reducing the risk of unauthorized access or exploitation.
- **Accessibility Considerations:** Alternative text descriptions, audio-based challenges, or other accessible features can be integrated to accommodate users with visual impairments or other disabilities.
- **User Experience:** CAPTCHAs should be designed to be user-friendly, with clear instructions, legible characters, and an appropriate level of difficulty to minimize user frustration.

### 6.1.2 Overview of responsibilities of Developer

A Software Developer is a professional who is charged with designing and coding software for businesses and consumers alike. They work closely with clients to determine what they need, then use programming languages like Java or Python to create programs. They must have critical thinking skills, as well as strong problem-solving abilities.

1. Data which will be handled by the software is excessively great; therefore, developers should take care of efficient memory management.
2. It is very important for Developers to pay attention to memory leaks
3. Software product should be fast and use memory efficiently, as much as possible. Therefore, high-performance CPUs should be preferred and maximum effort should spend efficiency

## 6.2 Usage Scenario

- **Scenario 1: A successful login with valid credentials:**
    1. User needs to enter his login credentials.
    2. User enters his username.
    3. User enters his password.
    4. User enters devnagari captcha with the help of virtual keyboard.
    5. User logins successfully into the system if the credentials are correct otherwise error is shown on the screen.
- **Scenario 2: A successful sign-up :**
    1. User needs to enter his personal information.
    2. User enters his username.
    3. User enters his password.
    4. User confirms his password.
    5. User enters devnagari captcha with the help of virtual keyboard.
    6. User signs-up successfully into the system if the captcha is correct.

### 6.2.1 User profiles

**Actor 1: User**

- User can make use of website for account creation and login..
- While the process of sign up and login user can make use of virtual devnagari keyboard for captcha verification.

**Actor 2: Developer**

- A developer would be responsible for creating and integrating the virtual keyboards of different languages for entering captchas .
- A developer would be responsible for maintaining the website on regular basis.

### 6.2.2 Use-cases

All use cases for the software are presented. A description of all main Use cases using a use case template is to be provided.

| Sr. No. | Use Case | Actors | Assumptions |
|---|---|---|---|
| 1 | User Registration with Devanagari CAPTCHA | User : The individual who intends to create an account on the website. | The user has access to the website's registration page. The user has basic knowledge of Devanagari characters and can interpret them. |
| 2 | Use Case: Login Authentication with Devanagari CAPTCHA | User: The individual attempting to log into their account on the website. | The user has an existing account on the website.The user knows their login credentials (username/email and password). |

Table 6.1: Use Cases

### 6.2.3 Use Case View

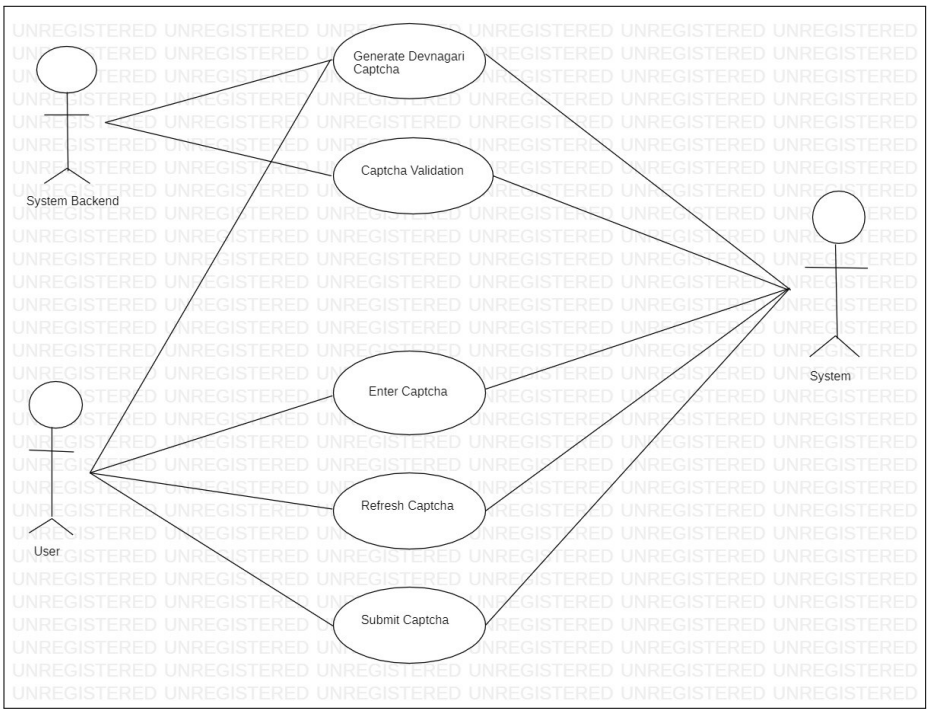



Figure 6.1: Use Case Diagram

## 6.3 Functional Model and Description

The class diagram depicts the relationships between all functions, modules, and data structures, as well as the ties between them, such as extends and aggregation. The dataset should include a collection of Devanagari characters. Devanagari is the script used for writing several languages, including Hindi, Nepali, and Sanskrit. The characters range from basic consonants and vowels to complex conjuncts. Ensure that the dataset covers a diverse range of Devanagari characters, including all the major consonants, vowels, and conjuncts.
The Captcha Generation generates a random string of a particular length and generates a image for the generated string. The functional model outlines the steps involved in generating a Devanagari CAPTCHA. The model takes inputs such as the desired length of the CAPTCHA, a dataset of Devanagari characters with corresponding images, and an image-character mapping. It also allows for optional parameters to customize the CAPTCHA appearance.

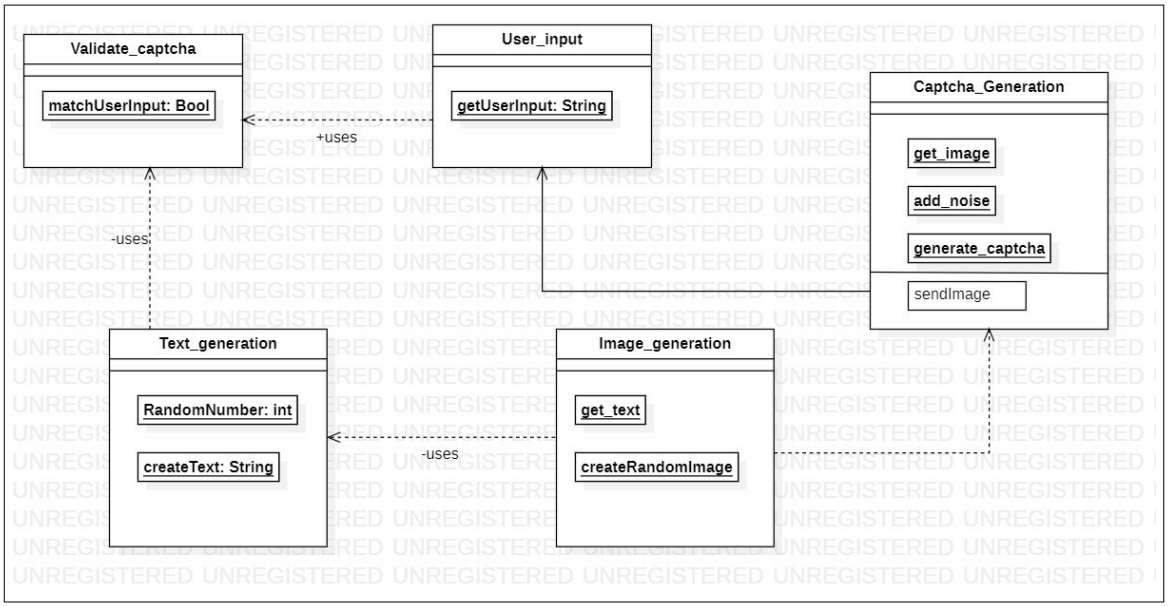



Figure 6.2: Class Diagram

## 6.3.1 Data Flow Diagram

### 6.3.1.1 Level 0 Data Flow Diagram

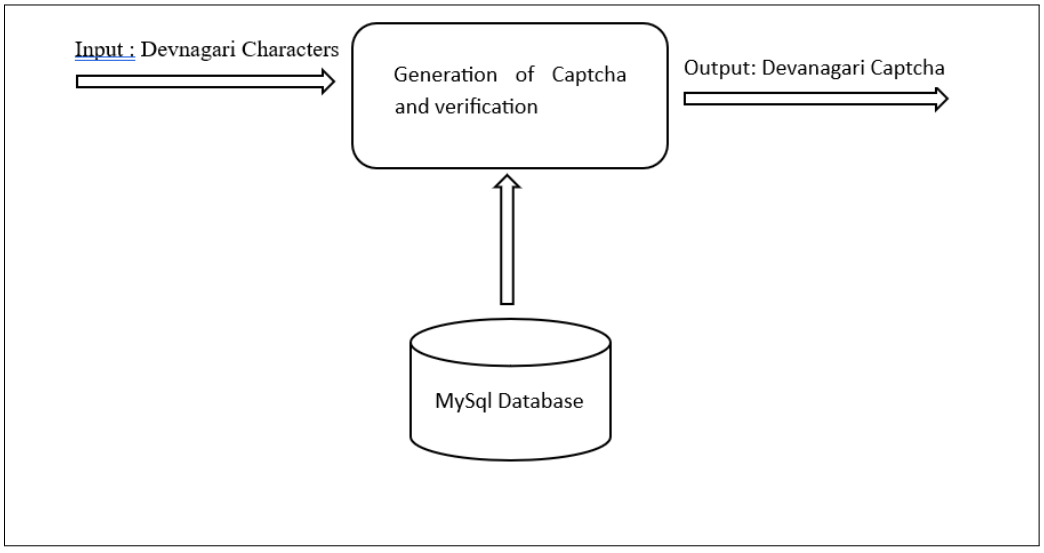



Figure 6.3: DFD Level 0

#### 6.3.1.2 Level 1 Data Flow Diagram

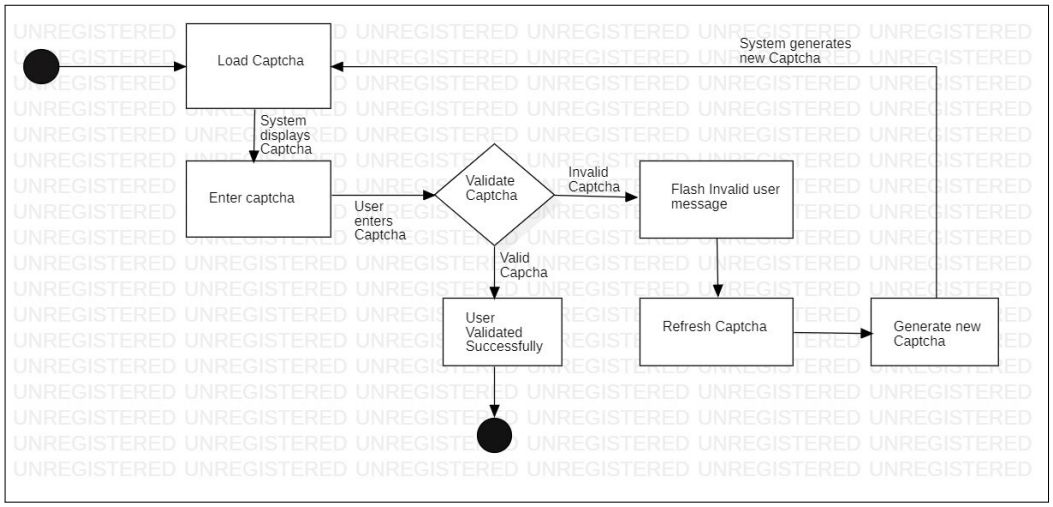



Figure 6.4: DFD Level 1

### 6.3.2 Activity Diagram:

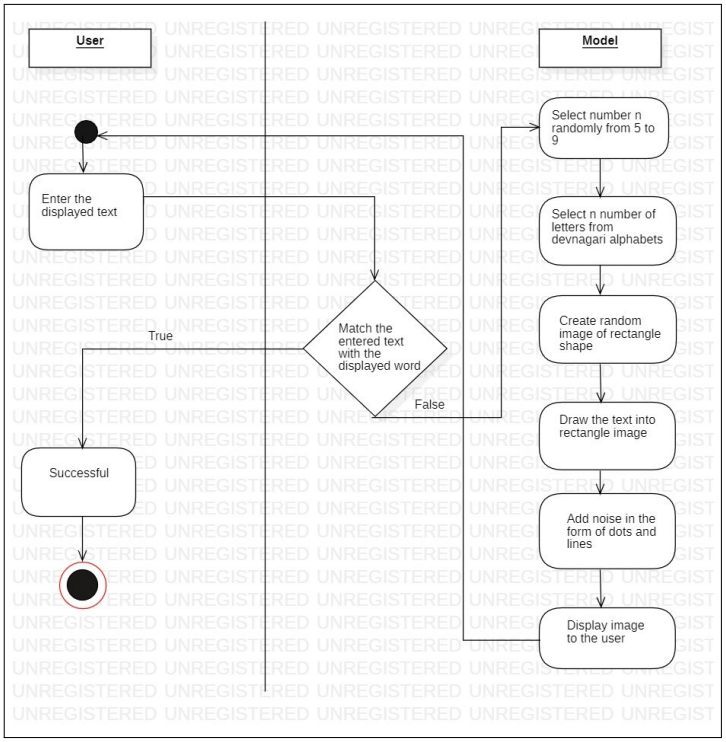



Figure 6.5: Activity Diagram

### 6.3.3 Non Functional Requirements:

**Performance:**

- **Efficiency:** The CAPTCHA generation process should be optimized to generate CAPTCHAs quickly, minimizing any delays in user interactions.

**Security:**

- Robustness: The generated CAPTCHAs should be robust against automated attacks, making it difficult for bots or computer algorithms to solve them.
- Randomness: The selection and arrangement of characters should be random and unpredictable to prevent pattern recognition or algorithmic exploitation.
- Resistance to OCR: The CAPTCHAs should be designed to be resistant to Optical Character Recognition (OCR) techniques used by automated bots.

**Usability:**

- Readability: The generated CAPTCHAs should be visually clear and legible, ensuring that human users can easily interpret and enter the characters.
- Accessibility: The CAPTCHAs should be accessible to users with visual impairments or other disabilities by providing alternative methods, such as audio-based CAPTCHAs or alternative text prompts.
- User Experience: The CAPTCHA generation process should be user-friendly, providing clear instructions and feedback to users during the interaction.

**Scalability:**

- Performance under Load: The CAPTCHA generation system should be able to handle a high volume of requests efficiently, ensuring smooth functioning during peak loads.
- Horizontal Scalability: The system should be designed to scale horizontally by adding additional resources or servers to handle increased demand.

**Maintainability:**

- Modularity: The CAPTCHA generation system should be designed in a modular and extensible manner, allowing for easy maintenance and future enhancements.
- Code Maintainability: The codebase for CAPTCHA generation should follow best practices, maintain readability, and be well-documented to facilitate ongoing maintenance and updates.

**Compatibility:**

- Cross-platform Compatibility: The CAPTCHA generation system should be compatible with different platforms and devices, ensuring consistent performance and functionality.

### 6.3.4 Design Constraints

**Character Set Limitations:**

1. Limited Character Set: The CAPTCHA generation system must work within the constraints of the available Devanagari character set. It should be designed to handle a specific set of characters and not rely on characters outside of this set.
2. Character Consistency: The generated CAPTCHAs should maintain consistency in terms of character style, font, and size to ensure a coherent visual representation.

**Image Generation Constraints:**

1. Image Size and Resolution: The CAPTCHA generation system should adhere to predefined size and resolution constraints to ensure consistency and compatibility across different devices and platforms.

**Security Constraints:**

1. Security Measures: The CAPTCHA generation system should implement appropriate security measures to protect against attacks, such as rate limiting, IP blocking, or detection of suspicious activity.
2. Avoiding Patterns: The system should be designed to prevent the formation of predictable patterns or repetitions within the generated CAPTCHAs, as this could potentially be exploited by automated algorithms.

**Performance Constraints:**

- Response Time: The CAPTCHA generation system should generate CAPTCHAs within an acceptable response time to provide a seamless user experience.
- Scalability: The system should be designed to handle a large volume of CAPTCHA requests simultaneously and efficiently, without compromising performance.

### 6.3.5 Software Interface Description

1. The user interface allows users to interact with the CAPTCHA generation system. It can include web-based forms, APIs, or command-line interfaces.
2. The user interface provides input fields or parameters for specifying the desired CAPTCHA length, font style, image size, or any other customization options.
3. It may also display the generated CAPTCHA image to users for visual verification or provide alternative options for accessibility, such as audio-based CAPTCHAs.

# Detailed Design Document

## 7.1 Introduction

**Devnagari Captcha Generation:**
The purpose of this design document is to outline the implementation details and specifications for generating Devanagari Captchas. Captchas (Completely Automated Public Turing tests to tell Computers and Humans Apart) are widely used in web applications to verify that a user is human and not a malicious bot. Devanagari, the script used for writing languages such as Hindi, Nepali, and Marathi, poses a unique challenge for captcha generation due to its complex and varied character forms.
This document aims to provide a comprehensive understanding of the design considerations, architecture, and algorithms involved in the generation of Devanagari Captchas. It will cover the key objectives, the target audience, and the high-level overview of the proposed solution

## 7.2 Data design

### 7.2.1 Internal software data structure

- Captcha Configuration: This data structure stores the configuration settings for generating Captchas. It includes attributes such as the length of the Captcha, the allowed character set, the complexity level, and any additional customization options.

- Captcha Image: This data structure represents the generated Captcha image. It typically includes attributes such as the image data, dimensions, file format, and any metadata associated with the image.

- Captcha Solution: This data structure stores the correct solution to a generated Captcha. It is used for validation and verification purposes when comparing user input with the original Captcha.

- User Response: This data structure holds the user's response to a presented Captcha. It includes the user's input, timestamps, and any metadata associated with the user's interaction.

- Session Data: If the Captcha generation system operates within a session-based environment, a data structure can be used to store session-specific information. This can include session IDs, session start and end times, and any other relevant session data.

- Logging Data: To facilitate monitoring and debugging, a data structure can be employed to capture logging information. This structure may include timestamps, log levels, error messages, and any contextual data necessary for troubleshooting.

### 7.2.2 Global Data Structures

- Captcha Data Structure: This data structure represents the generated Captcha and its associated attributes. It can include fields such as the Captcha image, solution, complexity level, timestamp, and any other relevant metadata. The Captcha data structure is accessible to components involved in Captcha generation, rendering, validation, and user interaction.
- User Data Structure: This data structure contains information related to the user interacting with the Captcha system. It may include fields like the user's ID, IP address, session details, user preferences, and any other relevant user-specific data. The User data structure is shared across components responsible for user management, session handling, and user interaction tracking.
- Logging Data Structure: The logging data structure stores logs and relevant information about system events, errors, and activities. It can include fields such as timestamps, log levels, event descriptions, and contextual data. This data structure is accessible to components responsible for logging, error handling, and system monitoring.
- Configuration Data Structure: The configuration data structure holds the system configuration settings and parameters. It includes fields such as Captcha length, character set, noise and distortion levels, and other customization options. The Configuration data structure is available to components involved in Captcha generation, rendering, and customization.

## 7.3 Architectural Design

The system architecture has been designed with a focus on four components Random Devnagari Character Generation,Image Rendering, Noise and Distortion, Validation and Verification.

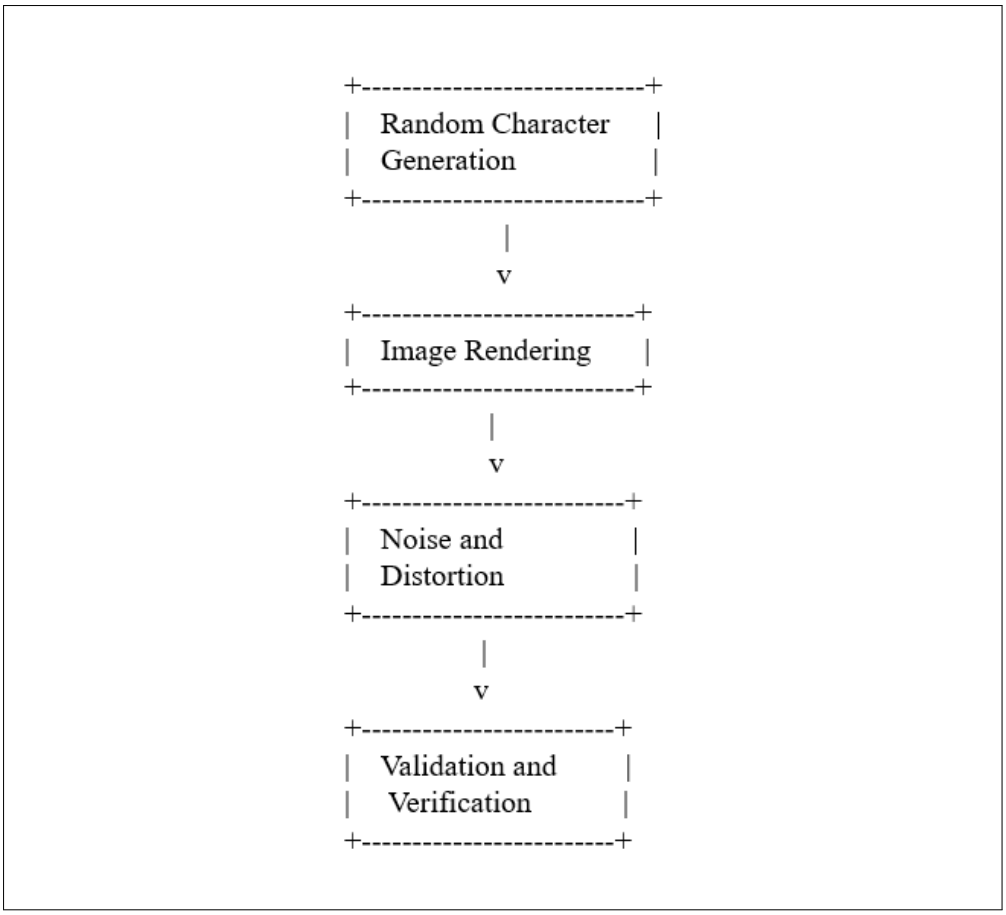



Figure 7.1: System Architecture

## 7.4 Database Description

MySQL Database is used in this application. It contains table containing columns such as name, username, email,mobile,etc.

## 7.5 Component Design

**Class Diagram**

A class diagram is a form of a static structural diagram that depicts a system's structure by displaying its classes, characteristics, operations, and object relationships.

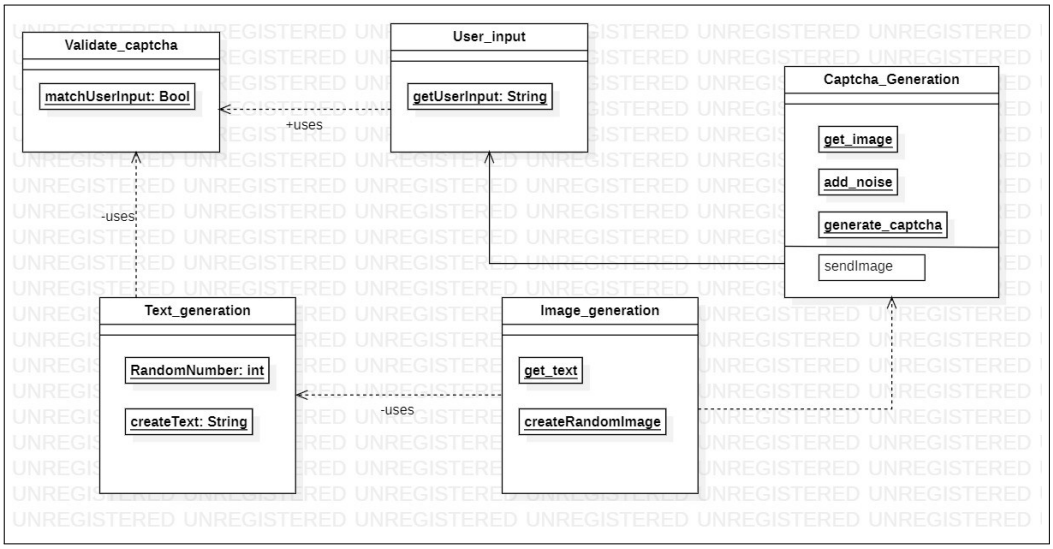



Figure 7.2: Class Diagram

The above diagram shows the relation between all the functions, modules, data structures, their attributes and operations of our project. It also shows "extends" "aggregation" features. The Captcha Generation modules are in aggregation with the user class.

# Project Implementation

## 8.1 Introduction

The implementation phase of the Devanagari CAPTCHA generation project is a crucial step that involves transforming the design specifications into a fully functional system. This phase focuses on coding, testing, and integrating the various components to create a robust and efficient Devanagari CAPTCHA generation system.

During implementation, the development team will utilize the design document and other reference materials to guide the coding process. The team will follow established software development practices, coding standards, and frameworks to ensure high-quality code and maintainable solutions.

## 8.2 Tools and Technologies Used

**Tools:**

1. Programming Language : Python
2. Integrated Development Environment (IDE): Visual Studio
3. Image Processing Libraries:
4. Web Frameworks: Flask , ReactJs
5. Version Control: Git
6. Database Management System: MySql
7. Documentation and Collaboration Tools

## 8.3 Methodologies/Algorithm Detail:

1. Character Distortion: To increase the security and prevent automated recognition, various distortion techniques can be applied to the character images. These may include geometric transformations, noise addition, warping, or other image processing operations.

2. Background Generation: The CAPTCHA generation algorithm may include the generation of complex and random backgrounds for the CAPTCHA images. This adds visual complexity and makes it harder for automated algorithms to isolate and recognize individual characters.

3. Font and Style Variation: The algorithm can incorporate font and style variations for each character, making the CAPTCHAs visually diverse and challenging for automated recognition.

4. Randomization: Randomization plays a crucial role in the CAPTCHA generation process. Random selection of characters, their positions, orientations, and other parameters ensures that each CAPTCHA is unique and difficult to crack using automated methods.

5. Security Measures: The algorithm may incorporate additional security measures to protect against attacks, such as rate limiting, IP blocking, or detection of suspicious behavior. These measures help prevent brute-force attacks or malicious activities.

# Software Testing

In order to check whether the product required and the product built is meeting the expected requirement, software testing has to be done. Software testing is an activity to check whether the actual result matches with the expected result and to make sure that the software is defect free. It helps to identify errors, gaps, or missing requirements contrary to the actual requirement. Software testing can be done either manually or by using automated tools.

## 9.1 Type of Testing Used

- **Unit Testing:** Every module i.e. prediction and activity recommendation modules has been tested for all possible bugs and errors that could have occurred during the execution of software.
- **Integration Testing:** After the compilation of all modules in the UI, the integration test is performed to check any dependency or resource-sharing issues that have been resolved before proceeding to system testing.
- **System Testing:** Our software has met all the functional requirements at adequate performance levels.
- **Acceptance Testing:** The final result is checked with the end users and is up to the satisfaction level including a clear understanding of it.
- **Functional Testing:** Functional testing is performed by focusing only on the output to check if it is as per the requirement or not.
- **Performance Testing:** To check whether the system meets the performance requirements.

## 9.2 Test Cases and Test Results

All use casess for the software are presented. A description of all main Use cases using use case template is to be provided.

- **Test Case 1 :** Verify successful login with valid credentials and correct CAPTCHA Steps:

  1. Enter a valid username and password.
  2. Enter the correct Devanagari CAPTCHA.
  3. Click the ”Login” button.

  Expected Result: The user should be successfully logged in.

- **Test Case 2 :** Verify error message for invalid username Steps:

  1. Enter an invalid username.
  2. Enter a valid password.
  3. Enter the correct Devanagari CAPTCHA.
  4. Click the ”Login” button.

  Expected Result: An error message should be displayed indicating that the username is invalid.

- **Test Case 3 :** Verify error message for incorrect password Steps:

  1. Enter a valid username.
  2. Enter an incorrect password.
  3. Enter the correct Devanagari CAPTCHA.
  4. Click the ”Login” button.

  Expected Result: An error message should be displayed indicating that the password is incorrect.

- **Test Case 4 :** Verify error message for incorrect CAPTCHA Steps:

  1. Enter a valid username.
  2. Enter a valid password.
  3. Enter an incorrect Devanagari CAPTCHA.
  4. Click the ”Login” button.

Expected Result: An error message should be displayed indicating that the CAPTCHA is incorrect.

- **Test Case 5:** Verify successful registration with valid credentials and correct CAPTCHA Steps:

  1. Enter a unique username.
  2. Enter a valid password.
  3. Confirm the password.
  4. Enter the correct Devanagari CAPTCHA.
  5. Click the ”Register” button.

  Expected Result: The user should be successfully registered.

- **Test Case 6 :** Verify error message for an existing username during registration Steps:

  1. Enter an existing username.
  2. Enter a valid password.
  3. Confirm the password.
  4. Enter the correct Devanagari CAPTCHA.
  5. Click the ”Register” button.

  Expected Result: An error message should be displayed indicating that the username already exists.

# Results

## 10.1 Screen shots

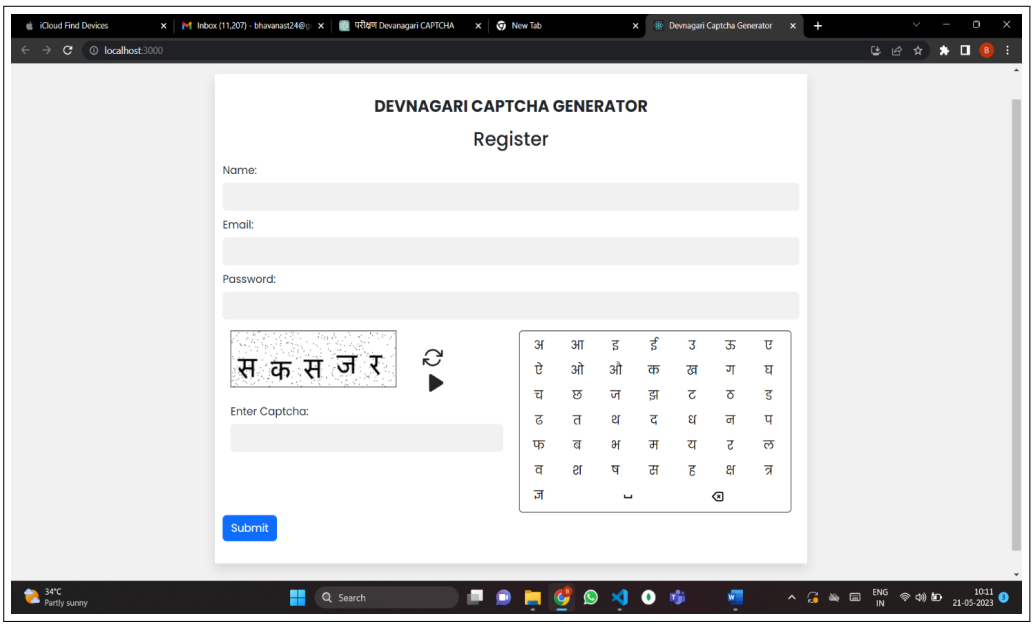



Figure 10.1: Sign Up Page

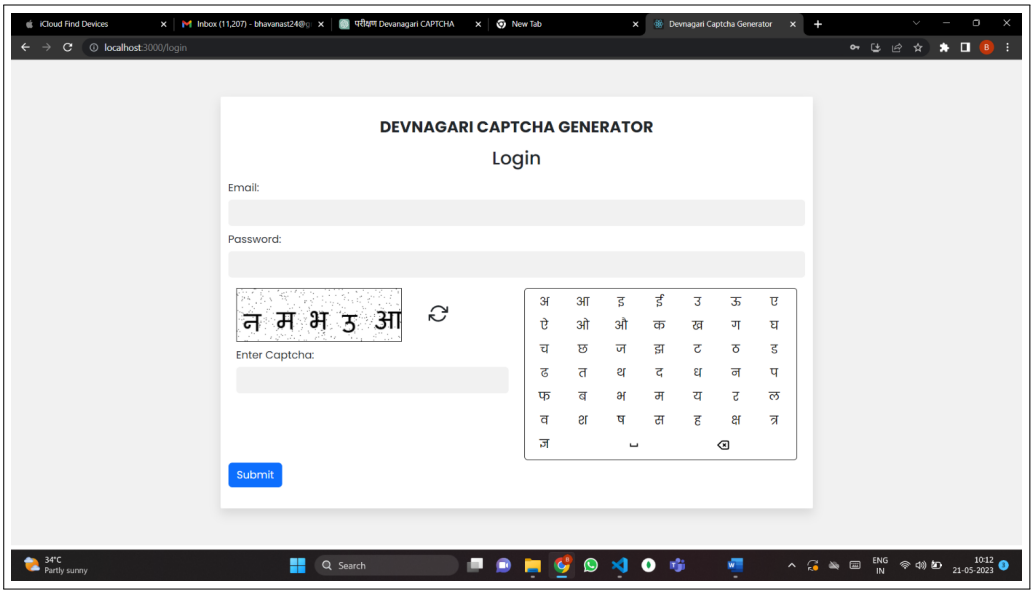



Figure 10.2: Login Page

## 10.2 Outputs

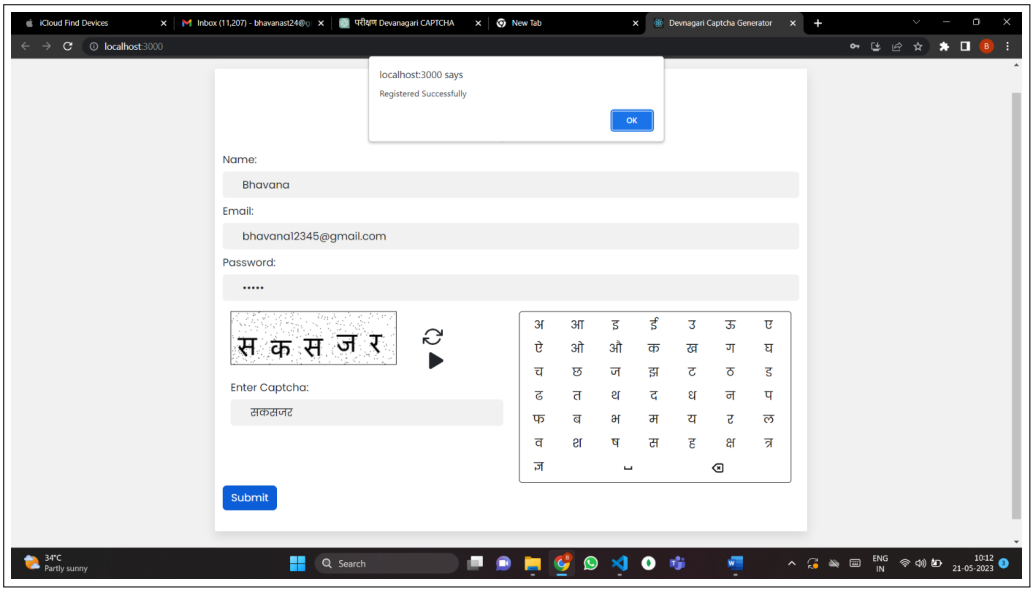



Figure 10.3: Sign Up Successful

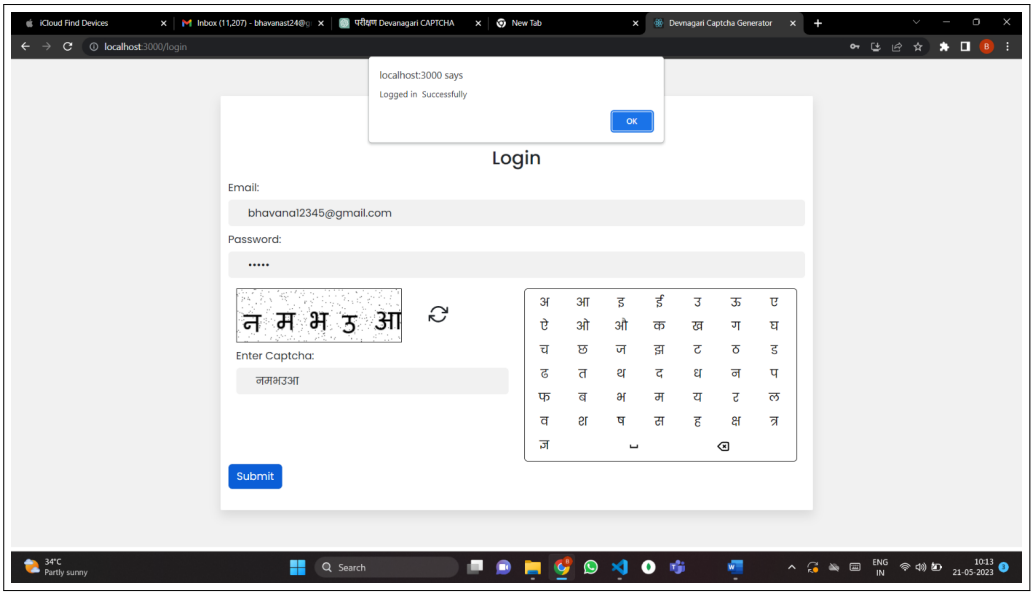



Figure 10.4: Login Successful

## 10.3 Installation and un-installation

Install the following modules using ‘pip install command’

1. captcha
2. Flask
3. Flask-MySQLdb
4. Flask-SQLAlchemy
5. itsdangerous
6. Jinja2
7. MarkupSafe
8. mysqlclient
9. numpy
10. opencv-python-headless
11. pi
12. python-dotenv
13. SQLAlchemy
14. Werkzeug

Run the ’python app.py runserver’ command to run the server.

# Conclusion and Future scope

Devanagari CAPTCHA is a valuable tool for enhancing security and preventing automated attacks in applications that utilize the Devanagari script. It provides a means of verifying human users by presenting them with a challenge that is typically difficult for automated scripts to solve.

**Multilingual CAPTCHA:** Expanding the capabilities of Devanagari CAPTCHA to support multiple languages within the same CAPTCHA challenge would be beneficial for applications catering to diverse language requirements.

**Mobile-friendly CAPTCHA:** With the growing use of mobile devices, optimizing Devanagari CAPTCHA for mobile screens and touch interfaces can improve user experience and ease of interaction.

**Adaptive CAPTCHA:** Developing adaptive CAPTCHA systems that dynamically adjust the difficulty level based on user behavior and risk analysis can enhance security while minimizing user frustration.

# Bibliography

[1] Y. Zi, H. Gao, Z. Cheng, and Y. Liu, "An end-to-end attack on text CAVrCHAs," IEEE Trans. Inf. Forensics Secur., voL 15, pp. 753-766, 2020.

[2] Y.-W. Chow, W. Susilo, and P. Thomcharoensri, "CAPTCHA design and security issues," in Proc. Adv. Cyber Secur.: Principles, Techn., Appl. , 2019, pp. 69-92.

[3] X. Ling et al., "Deepsec: A uniform platform for security analysis of deep learning model," in Proc. IEEE Symp. Secur. Privacy, 2019, pp. 673—690.

[4] J. Chen, X. Luo, Y. Liu, J. Wang, and Y. Ma, "Selective learning confusion class for text-based captcha recognition," IEEE Access, vol. 7, pp. 22246-22259, 2019.

[5] J. Li, S. Ji, T. Du, B. Li, and T. Wang, "Textbugger. Generating adversarial text against real-world applications," in Ptoc. Annu. Netw. Distrib. Syst. Secur. Symp., 2019, pp.

# References

[1] Y. Zi, H. Gao, Z. Cheng, and Y. Liu, "An end-to-end attack on text CAVrCHAs," IEEE Trans. Inf. Forensics Secur., voL 15, pp. 753-766, 2020.

[2] Y.-W. Chow, W. Susilo, and P. Thomcharoensri, "CAPTCHA design and security issues," in Proc. Adv. Cyber Secur.: Principles, Techn., Appl. , 2019, pp. 69-92.

[3] X. Ling et al., "Deepsec: A uniform platform for security analysis of deep learning model," in Proc. IEEE Symp. Secur. Privacy, 2019, pp. 673—690.

[4] J. Chen, X. Luo, Y. Liu, J. Wang, and Y. Ma, "Selective learning confusion class for text-based captcha recognition," IEEE Access, vol. 7, pp. 22246-22259, 2019.

[5] J. Li, S. Ji, T. Du, B. Li, and T. Wang, "Textbugger. Generating adversarial text against real-world applications," in Ptoc. Annu. Netw. Distrib. Syst. Secur. Symp., 2019, pp.

[6] H. Kwon, Y. Kim, H. Yoon, and D. Choi, 'CAPTCHA image generation systems using generative adversarial networks," IEICE Trans. Inf. Syst., vol. 101, no. 2, pp. 543-546, 2018.

[7] J. Chen, X. Luo, Y. Liu, J. Wang, and Y. Ma, "Selective learning confusion class for text-based captcha recognition," IEEE Access, vol. 7, pp. 22246-22259, 2019.

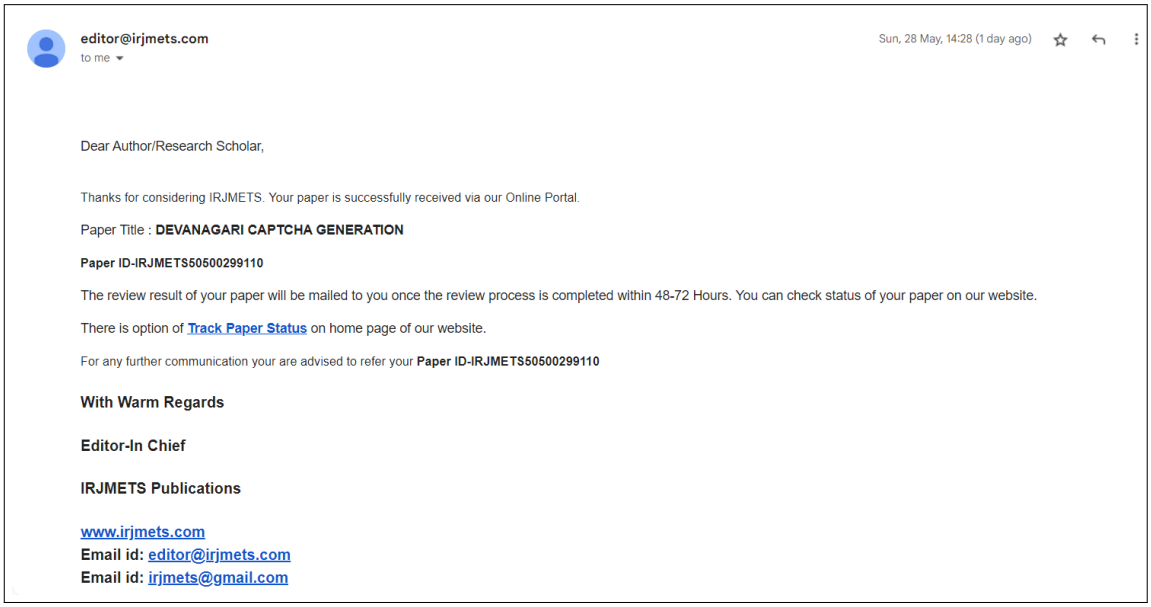
[8] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks, in Proc. Int. Conf. Learn. Repre- sentations, 2018, pp.

[9] W. Jonathan, Strong CAPTCHA Guidelines: v1.2. http://bitland.net/captcha.pdf, December.

[10] Y. Jeff and A. El Ahmad, "Usability of CAPTCHAs or Usability Issues in CAPTCHA Design," In SOUPS '08, pp. 44–52, New York, NY, USA,. ACM.

# Appendix : Proof of Paper Submission

1. Paper Title: Devnagari Captcha Generation
2. Name of the Conference/Journal where paper submitted : IRJMETS (INTERNATIONAL RESEARCH JOURNAL OF MODERNIZATION IN ENGINEERING TECHNOLOGY AND SCIENCE)
3. Paper accepted/rejected : Pending
4. Review comments by reviewer : Pending
5. Corrective actions if any : NA



editor@irjmets.com
to me

Sun, 28 May, 14:28 (1 day ago)

Dear Author/Research Scholar,

Thanks for considering IRJMETS. Your paper is successfully received via our Online Portal.

Paper Title : **DEVANAGARI CAPTCHA GENERATION**

**Paper ID-IRJMETS50500299110**

The review result of your paper will be mailed to you once the review process is completed within 48-72 Hours. You can check status of your paper on our website.

There is option of Track Paper Status on home page of our website.

For any further communication your are advised to refer your **Paper ID-IRJMETS50500299110**

**With Warm Regards**

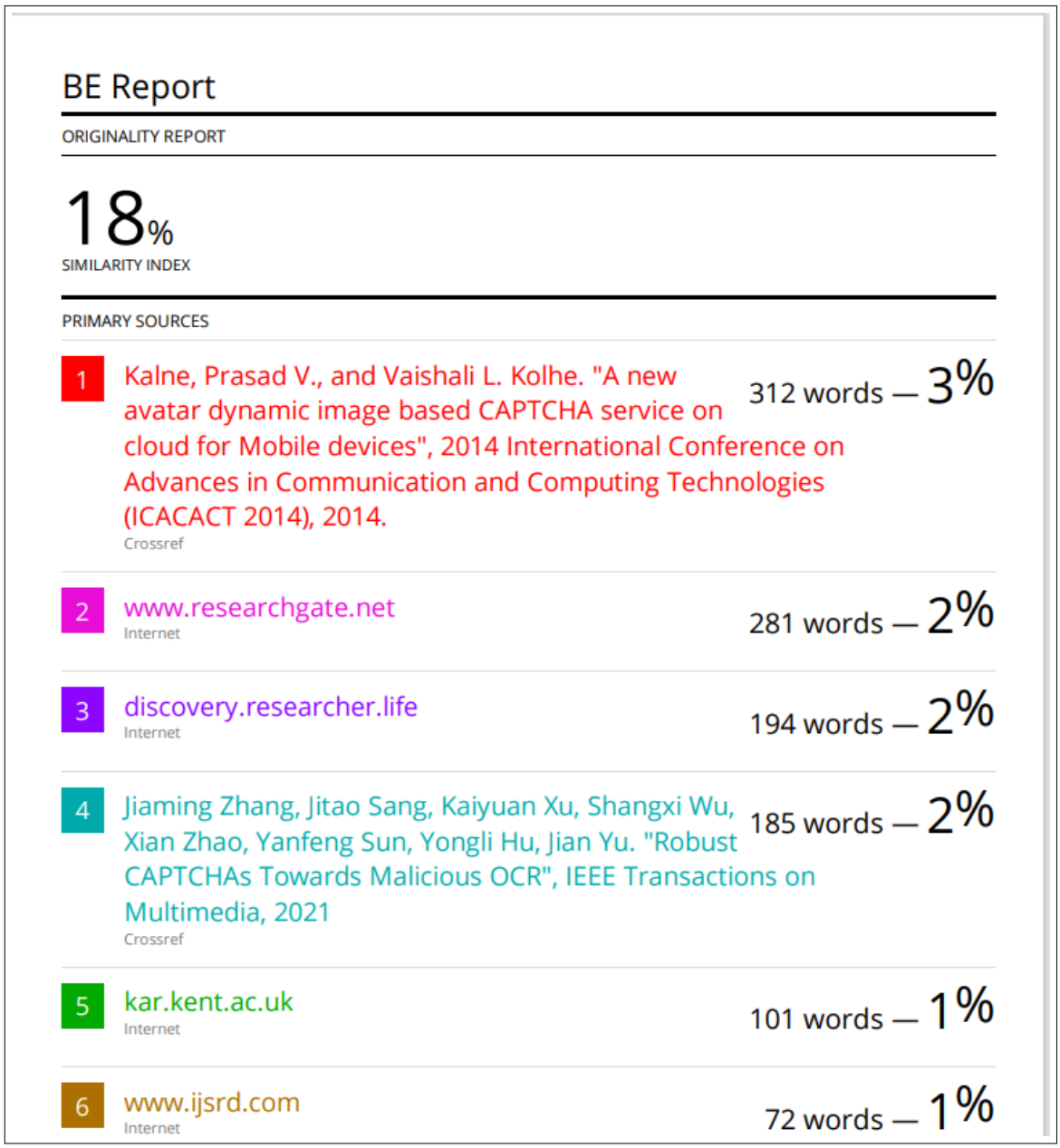
**Editor-In Chief**

**IRJMETS Publications**

www.irjmets.com
**Email id:** editor@irjmets.com
**Email id:** irjmets@gmail.com

Figure 1: Paper Submitted

# Plagarism Report



BE Report

ORIGINALITY REPORT

18%
SIMILARITY INDEX

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | Kalne, Prasad V., and Vaishali L. Kolhe. "A new avatar dynamic image based CAPTCHA service on cloud for Mobile devices", 2014 International Conference on Advances in Communication and Computing Technologies (ICACACT 2014), 2014. Crossref | 312 words — 3% |
| 2 | www.researchgate.net Internet | 281 words — 2% |
| 3 | discovery.researcher.life Internet | 194 words — 2% |
| 4 | Jiaming Zhang, Jitao Sang, Kaiyuan Xu, Shangxi Wu, Xian Zhao, Yanfeng Sun, Yongli Hu, Jian Yu. "Robust CAPTCHAs Towards Malicious OCR", IEEE Transactions on Multimedia, 2021 Crossref | 185 words — 2% |
| 5 | kar.kent.ac.uk Internet | 101 words — 1% |
| 6 | www.ijsrd.com Internet | 72 words — 1% |

Figure 2: Plagiarism Report

# Information of Project Group Members

**1. Bhavana Thakare**

- Date of birth: 02/04/2001
- Gender: Female
- Address: Lasalgaon, Nashik
- Email: bhavanast24@gmail.com
- Contact Number:8805472698

**2. Mrunal Sadawarte**

- Date of birth: 17/01/2001
- Gender: Female
- Address: Malkapur, Buldhana
- Email: sadawartemrunal@gmail.com
- Contact Number: 8669885980

**3. Sakshi Ghonge**

- Date of birth: 30/0/2001
- Gender: Female
- Address: Darwha , Yavatmal
- Email: sakshighonge3434@gmail.com
- Contact Number: 7350673897

**4. Abhiram Shelkar**

- Date of birth: 01/09/2001
- Gender: Male
- Address: Akola
- Email: abhiramshelkar5@gmail.com
- Contact Number: 9561224685