A

SYNOPSIS

of

MINOR PROJECT

on

No Cheator Zone

तमसो मा ज्योतिर्गमय्

Darkness to Light

Submitted by

JAGRATI SHRIMALI (Roll No.-21EGICA015)

Project Guide
Ms. Monika Bhatt mam

Head of Department
Dr. Mayank Patel

Geetanjali Institute of Technical Studies, Dabok, Udaipur (Raj.)
Department of Computer Science and Engineering

October,2023

Problem Statement:

The "No Cheater Zone" project aims to develop a comprehensive system to uphold academic integrity by preventing and detecting cheating during examinations and assessments. This system is designed to ensure that students are evaluated fairly based on their knowledge and abilities, free from dishonest practices.

Brief Description:

The "No Cheater Zone" project aims to develop a secure and reliable system to maintain academic integrity by preventing and detecting cheating during examinations and assessments. This system combines user authentication, real-time proctoring, and plagiarism detection to ensure a fair and honest evaluation of students.

Key Features:

1. **User Authentication**:
   - Secure login system using usernames and passwords.
   - Multi-factor authentication to enhance security.
2. **Real-time Proctoring**:
   - Monitor users during tests to detect suspicious behaviour, such as switching away from the test window.
   - Use AI to analyze webcam feeds for unusual activities, like multiple faces or frequent looking away.
3. **Plagiarism Detection**:
   - Compare submitted work against a database of previous submissions to identify exact matches.
   - Use advanced algorithms to detect paraphrasing and other subtle forms of plagiarism.

Objective and Scope:

• Prevent unauthorized access to the system.
• Ensure the person taking the test is the intended test-taker.
• Detect and prevent cheating behaviours during tests.
• Verify the originality of submitted work.

Methodology:

The "No Cheater Zone" project follows a structured methodology to ensure the effective development and deployment of the system. The methodology is divided into several key phases: Requirements Gathering, Design, Development, Testing, Deployment, and Maintenance.

• **Research and Planning**:

- Gather requirements from stakeholders.
- Research existing solutions and technologies.

• **Design**:

- System architecture design.
- UI/UX design for the web application.

• **Development**:

- Develop authentication module.
- Develop proctoring module.
- Develop plagiarism detection module.

• **Testing**:

- Unit testing and integration testing.
- User acceptance testing with a pilot group.

• **Deployment**:

- Deploy the application on a secure server.
- Train users and administrators.

- **Evaluation and Feedback**:

  - Collect feedback from users and make necessary improvements.

Hardware and Software Requirements:

Hardware:

- **Processor**: Dual-core processor (e.g., Intel i3 or equivalent)
- **Memory**: Minimum 4 GB RAM
- **Storage**: At least 100 GB free space
- **Webcam**: HD webcam for proctoring
- **Microphone**: Built-in or external microphone
- **Internet**: Stable internet connection with at least 5 Mbps speed
- **Monitor**: Minimum resolution of 1280x720

Software:

- **Operating System**: Linux (Ubuntu, CentOS, or similar) or Windows Server
- **Web Server**: Apache or Nginx
- **Database**: PostgreSQL, MySQL, or MongoDB
- **Programming Language**: Python (for backend logic)
- **Framework**: Django or Flask for Python backend
- **AI/ML Libraries**: OpenCV, TensorFlow, or PyTorch (for proctoring module)
- **Plagiarism Detection API**: Integration with a third-party service like Turnitin or an open-source alternative

Technologies: Frontend Technologies

1. **HTML5**: Markup language for structuring and presenting content on the web.

CSS3: Styling language for designing the layout and appearance of the web application.

2. **JavaScript**: Programming language for adding interactivity and dynamic behaviour to web pages.
3. **React.js**: JavaScript library for building user interfaces, particularly for single-page applications.
4. **Bootstrap**: CSS framework for responsive design and pre-designed UI components.

Backend Technologies

1. **Python**: High-level programming language used for backend logic and server-side scripting.
2. **Django**: High-level Python web framework that encourages rapid development and clean, pragmatic design.
3. **Flask**: Lightweight Python web framework (alternative to Django) for smaller applications or microservices.
4. **Node.js**: JavaScript runtime for building fast and scalable server-side applications (optional, for specific use cases).

Database Technologies

1. **PostgreSQL**: Open-source relational database management system known for its robustness and performance.
2. **MySQL**: Widely used open-source relational database management system.
3. **MongoDB**: NoSQL database for handling large amounts of unstructured data.

Proctoring and AI Technologies

1. **OpenCV**: Open-source computer vision library for real-time image processing and analysis.

2. **TensorFlow**: Open-source machine learning framework for building and training AI models.
3. **PyTorch**: Open-source machine learning library for deep learning applications.

Testing Techniques:

• **Unit Testing**: Test individual components and modules for correctness.
• **Integration Testing**: Test the interaction between different modules to ensure they work together seamlessly.
• **System Testing**: Perform end-to-end testing of the entire system to ensure it meets the requirements.
• **User Acceptance Testing (UAT)**: Conduct testing with a group of end-users to gather feedback and make necessary improvements.

Project Contribution:

• **Requirement Gathering**:

- Collect input from stakeholders to define project requirements.
- Document functional and non-functional requirements.

• **Sprint Planning**:

- Break down the project into manageable tasks and sprints.
- Assign tasks to team members based on their roles and expertise.

• **Development**:

- Follow Agile methodology for iterative development.
- Conduct regular code reviews and pair programming sessions.

• **Testing**:

- Implement automated and manual testing processes.
- Continuously integrate and test new features.

- **Deployment**:
  - Use CI/CD pipelines for smooth and efficient deployment.
  - Monitor the live system and perform regular updates and maintenance.

- **Feedback and Improvement**:
  - Gather feedback from users and stakeholders.
  - Continuously improve the system based on feedback and new requirements.