International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 11 | Nov - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

SecureMark: Advanced Multimedia Watermarking and Authentication System for Images, Audio, and Video

Prof. Alsaba Naaz[1], Tanishq Nagrare[2], Tanmay Banode[3], Shivanshu Yadav[4], Aditya Gupta[5], Rishi Joshi[6]

Department of Computer Science and Engineering, Priyadarshini Bhagwati College of Engineering, Nagpur, Maharashtra, India

***Abstract*—**Images, audio recordings, and videos can be copied and shared again with very little effort, so creators often lose track of where their work ends up. This paper presents the design of SecureMark, a web-based system that embeds identifying information into image, audio, and video files and later extracts and verifies it. The image module replaces selected least significant bits of pixel values, the audio module does the same for 16-bit PCM WAV samples, and the video module embeds bits in selected frame pixels and rebuilds the processed frames as a video. A fixed end marker tells the extractor where the message stops. The system is planned with React and Vite, Node.js and Express, Python, and MongoDB, with Pillow, NumPy, OpenCV, FFmpeg, and ExifTool for media handling. The paper describes the requirements, architecture, workflows, threat assumptions, and a test plan. Least significant bit embedding is simple but fragile under compression, resizing, resampling, and re-encoding, and successful extraction shows only that a mark is present, not that ownership is legally proven. The supplied project material contains no measured results, so no performance figures are reported.

Keywords*—digital watermarking, least significant bit, image watermarking, audio watermarking, video watermarking, watermark verification*

I. INTRODUCTION

People use digital media every day for study, work, entertainment, and social contact. A photograph, a song, or a video can travel across the world in seconds, and the same speed lets others copy, alter, and repost it without asking the creator. After a file spreads, proving who first made it is difficult, and a visible copyright mark can be cropped out or painted over. Hiding identifying data inside the file itself is one response to this problem [1].

A watermark is a small piece of identifying data placed within the content of a file, for example in the pixels of a picture or the samples of a recording. It may hold a creator name, copyright details, or an identification message. Watermarking belongs to the wider field of information hiding, which also includes steganography [2]. Many available tools handle only one media type, so a creator who works with photographs, music, and video may need several separate programs.

The project, titled SecureMark, proposes one web application for invisible watermarking of images, audio, and videos using the least significant bit (LSB) technique. A user uploads a file, enters ownership information, receives a watermarked output, and can later extract the message for verification. The main contribution of this paper is a structured design of the project, covering its threat assumptions, embedding and extraction workflow, functional modules, and a reproducible evaluation plan. Since the supplied project material reports no completed measurements, the paper separates design expectations from demonstrated outcomes.

A. Problem Statement

Creative work is routinely copied and passed on without the owner's consent. Visible marks can be removed, and separate tools for image, audio, and video make marking inconvenient. A file that has been shared widely may also be changed by others. A workflow is therefore needed in which identifying information is embedded in each media type, can be extracted from the file, and can be compared with the expected ownership details, while its limits are stated honestly.

B. Aim and Objectives

The aim is to design a web-based watermarking system for three media types. The objectives are:

- Embed invisible identifying information into image pixels.
- Embed ownership information into 16-bit PCM audio samples.
- Embed identifying information into video through frame-based processing.
- Extract and verify the embedded message from supported watermarked files.
- Provide one interface for upload, processing, extraction, and verification.
- Define tests that can evaluate correctness, quality, and robustness without inventing results.

C. Scope and Assumptions

The system covers embedding, extraction, and verification for images, audio, and video, with user registration and login. It assumes that watermarked outputs are kept in lossless formats where possible, since lossy compression rewrites the values that carry the mark. It does not claim robustness against deliberate removal, strong cryptographic protection of the message, or proof of legal ownership. A recovered message shows that a mark is present and matches expected details; it does not show that the file has never been modified.

II. LITERATURE REVIEW

van Schyndel et al. were among the first to place a watermark in the least significant bit plane of an image, using pseudo-random sequences [3]. Changing the lowest bit moves a pixel by one grey level, which is why the mark is hard to see and also why it is easily lost. Chan and Cheng later reduced the distortion of plain LSB substitution with an optimal pixel adjustment step and judged quality

numerically [4], which supports reporting image quality with a measured value.

Transform-domain work places the mark where it is harder to discard. Cox et al. spread a watermark over significant frequency components and reported detection after compression, filtering, and cropping [5], and Kundur and Hatzinakos embedded marks in wavelet detail coefficients at several resolutions [6]. These methods are usually proposed as an upgrade over LSB embedding, at the cost of more computation.

For verification, Yeung and Mintzer described a key-based invisible mark that can locate altered regions of an image [7], and Lin and Chang separated harmless JPEG compression from malicious editing [8]. Together they show that verification can be treated as an integrity decision and that fragile and robust marks serve different purposes.

For audio, Bassia et al. embedded a signal-shaped mark in time-domain samples and reported resistance to filtering and compression [9], and Hua et al. reviewed twenty years of audio watermarking, including the attacks of resampling, noise, and compression [10]. For video, Hartung and Girod addressed both uncompressed and MPEG-2 compressed streams [11], and Hartung and Kutter surveyed watermarking across images, audio, and video [12]. Video is the hardest case for LSB methods because encoding rewrites the pixel values of each frame.

Petitcolas argued that watermarking claims need defined tests [13]. Chen and Wornell analysed the trade-off among embedding rate, distortion, and robustness through quantization index modulation [14]. Begum and Uddin reviewed image techniques by embedding domain [15], and Zhu et al. trained neural networks to hide data that survives distortions such as JPEG compression [16].

Taken together, these studies indicate that a watermarking design should address embedding, quality, robustness, and verification as related but separate concerns. Existing work usually concentrates on one media type or one technical aspect. The proposed system brings image, audio, and video into one workflow, and its present limitation is that the project material gives no implementation measurements. The evaluation plan in this paper is intended to close that gap during implementation.

III. METHODOLOGY

A. System Architecture

The system is a client–server web application, shown in Fig. 1. The React.js and Vite frontend provides the Home, Login, Register, Dashboard, Image, Audio, Video, and Verify pages. The Node.js and Express.js backend manages authentication, API routes, and file handling, and runs the matching Python script. Python performs embedding and extraction, with FFmpeg for audio conversion and video reconstruction [17], the wave module and NumPy for PCM samples [18], Pillow for image pixels [19], and OpenCV for video frames [20]. ExifTool reads and writes selected metadata fields [21]; metadata supplements the watermark and is not a substitute for it. MongoDB stores user and application records.

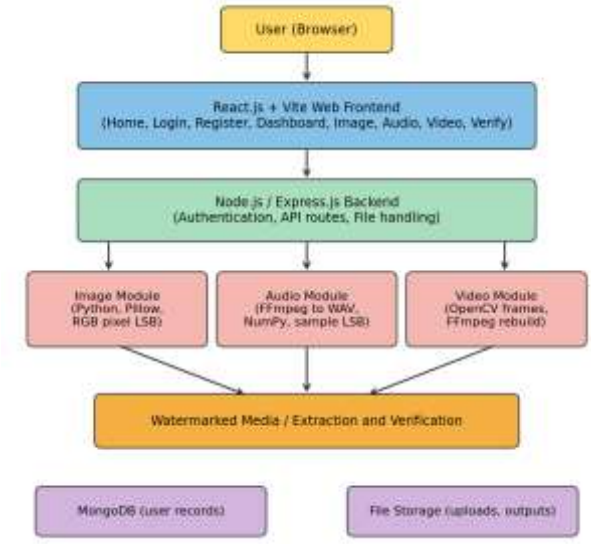


Fig. 1. Proposed logical architecture of the SecureMark system.

B. Functional Modules

1) *User interface:* provides the pages through which users register, log in, upload files, and run watermark operations.

2) *Image watermarking:* reads RGB pixel values, embeds the message by LSB substitution, and saves the result as PNG, because lossless storage preserves the modified pixel values.

3) *Audio watermarking:* converts supported formats to 16-bit PCM WAV with FFmpeg when required and embeds the message in sample LSBs.

4) *Video watermarking:* reads frames with OpenCV, embeds bits in selected frame pixel-channel LSBs, and rebuilds the video with FFmpeg, keeping a compatible audio stream where possible.

5) *Extraction:* reads pixel, sample, or frame data in the same order used for embedding and reconstructs the message.

6) *Verification:* displays the extracted text so it can be compared with the expected creator or copyright details.

7) *Database:* stores user account information and other application records.

C. Embedding and Extraction Workflow

The workflow is shown in Fig. 2. After login, the user selects a media type, uploads a file, and enters the watermark message. The system prepares the media (RGB conversion for images, PCM WAV for audio, frame reading for video), converts the message and the end marker #####END##### to bits, and writes them into selected pixel-channel, sample, or frame-pixel LSBs. The output is saved as PNG, WAV, or a rebuilt video.

For extraction, the watermarked file is read in the same order and positions used during embedding, the LSBs are collected and grouped into characters, and reading stops when the end marker is found. For verification, the recovered text is compared with the expected ownership details.



Fig. 2. Flowchart of embedding, extraction, and verification.

D. Security Requirements and Threat Model

The main goals are embedding that is not normally noticeable, correct recovery of the message from an unmodified output, and honest verification. The threat model includes someone who copies a file, recompresses or resizes it, or tries to read or overwrite the embedded bits. Because plain LSB embedding stores the message without encryption, anyone who knows the extraction procedure can read it and anyone who can edit the low-order bits can remove or change it. The design does not assume protection against a deliberate attacker who rewrites low-order bits, and it does not treat the watermark as a substitute for access control or encryption. Login, authorization of file operations, and secure handling of uploaded files remain application-level controls that must follow current practice.

E. Proposed Evaluation Plan

Because no implementation logs or measured results were supplied, evaluation is specified as a plan. Tests should be run on the implemented system with synthetic or non-sensitive media, following the principle of defined tests [13]:

- *Functional correctness:* embed, save, upload, extract, and verify for each media type.
- *Quality:* PSNR for images and video frames, and SNR for audio, comparing original and watermarked data.
- *Negative cases:* extract from an unmarked file and from a file with a modified watermark region, and confirm the system does not report a false message.
- *Robustness:* attempt extraction after JPEG compression, image resizing, audio resampling or format conversion, and video re-encoding, and record the outcome.
- *Cost:* record processing time and file-size change for several file sizes.

For images and video frames the PSNR is

$$\mathrm{PSNR} = 10 \log_{10}(\mathrm{MAX}^2 / \mathrm{MSE}) \quad (1)$$

where MAX is the largest possible pixel value and MSE is the mean squared error, and for audio the SNR is

$$\mathrm{SNR} = 10 \log_{10}(\Sigma x^2 / \Sigma(x - y)^2) \quad (2)$$

where x denotes original samples and y watermarked samples. The machine, software versions, file types, and file sizes should be recorded, each measurement repeated, and the number of runs stated. No numerical result should be included unless it is produced by the implemented system and can be reproduced.

IV. RESULTS

The supplied project material defines the intended components and workflow but does not provide measured quality values, processing times, test logs, or datasets. This paper therefore does not claim that embedding, extraction, or verification has been experimentally validated. The current deliverable is a design and evaluation specification, and Table I lists the validation tests to be completed before results are reported.

TABLE I. PLANNED VALIDATION TESTS

Test	Expected outcome	Status
Embed and extract (image)	Original message recovered	Not tested
Embed and extract (audio)	Original message recovered	Not tested
Embed and extract (video)	Original message recovered	Not tested
Unmarked file	No false message reported	Not tested
Lossy re-encoding	Outcome recorded, loss expected	Not tested
Image PSNR, audio SNR	Measured values recorded	Not tested
Time and file size	Measured values recorded	Not tested

V. DISCUSSION

The design is a reasonable baseline for a student project because it keeps image, audio, and video processing in separate modules behind one interface. Several choices need care before the system can be described as reliable. First, the end marker tells the extractor where to stop, but a message that happens to contain the marker, or a damaged file that never shows it, must be handled so extraction does not run on or return noise. Second, embedding in sequential positions is easy to read and easy to erase; using key-dependent positions or encrypting the message would raise the cost of casual tampering. Third, formats matter: PNG and 16-bit PCM WAV preserve low-order bits, whereas lossy image, audio, and video formats may not.

Video needs particular attention. Re-encoding changes pixel values in every frame, so a frame-based LSB mark is likely to be lost when a video is converted or compressed. The project material does not state which frames or positions are chosen, so frame selection, capacity, and the effect of the chosen output codec remain open design questions. ExifTool metadata can supplement identification, but metadata is easy to strip and should not be treated as a watermark.

Verification should be described carefully. Recovering the expected text shows that a mark is present in that file, but it does not prove who created the content or that the file is unaltered. Stronger claims would need a cryptographic hash or digital signature stored where an attacker cannot replace it, as integrity-oriented schemes suggest [7], [8]. Transform-domain embedding, message encryption, role-based access, and cloud deployment are reasonable extensions, but they should be reported as future work unless they are implemented and tested. The main limitation of this paper is the absence of implementation evidence; journal-ready claims about quality or robustness require repeatable tests.

VI. CONCLUSION

This paper presented a design for SecureMark, a web-based system that watermarks images, audio, and videos using LSB embedding and offers extraction and verification through one interface. The architecture separates the media-specific modules, defines the embedding and extraction workflow with an end marker, and states the threat assumptions and limits of the approach. The literature shows that LSB embedding is simple and has low visible impact but is fragile under compression and re-encoding, so the system is suitable for basic ownership identification and not for strong protection.

Future work includes implementation and testing, measurement of PSNR, SNR, and extraction accuracy, evaluation of DCT and DWT embedding, encryption of the watermark message, cryptographic identifiers for verification, better frame selection for video, and cloud deployment. Actual results must be added before the manuscript is represented as an experimentally evaluated system.

ACKNOWLEDGMENT

The authors acknowledge the guidance of Prof. Alsaba Naaz and the Department of Computer Science and Engineering, Priyadarshini Bhagwati College of Engineering, Nagpur, for support of the project.

REFERENCES

[1] I. J. Cox, M. L. Miller, J. A. Bloom, J. Fridrich, and T. Kalker, *Digital Watermarking and Steganography*, 2nd ed. Burlington, MA, USA: Morgan Kaufmann, 2008.

[2] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding—A survey," *Proc. IEEE*, vol. 87, no. 7, pp. 1062–1078, Jul. 1999.

[3] R. G. van Schyndel, A. Z. Tirkel, and C. F. Osborne, "A digital watermark," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, vol. 2, 1994, pp. 86–90.

[4] Y.-K. Chan and L.-M. Cheng, "Hiding data in images by simple LSB substitution," *Pattern Recognit.*, vol. 37, no. 3, pp. 469–474, Mar. 2004.

[5] I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Trans. Image Process.*, vol. 6, no. 12, pp. 1673–1687, Dec. 1997.

[6] D. Kundur and D. Hatzinakos, "Digital watermarking using multiresolution wavelet decomposition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 5, 1998, pp. 2969–2972.

[7] M. M. Yeung and F. Mintzer, "An invisible watermarking technique for image verification," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, vol. 2, 1997, pp. 680–683.

[8] C.-Y. Lin and S.-F. Chang, "A robust image authentication method distinguishing JPEG compression from malicious manipulation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 2, pp. 153–168, Feb. 2001.

[9] P. Bassia, I. Pitas, and N. Nikolaidis, "Robust audio watermarking in the time domain," *IEEE Trans. Multimedia*, vol. 3, no. 2, pp. 232–241, Jun. 2001.

[10] G. Hua, J. Huang, Y. Q. Shi, J. Goh, and V. L. L. Thing, "Twenty years of digital audio watermarking—A comprehensive review," *Signal Process.*, vol. 128, pp. 222–242, Nov. 2016.

[11] F. Hartung and B. Girod, "Watermarking of uncompressed and compressed video," *Signal Process.*, vol. 66, no. 3, pp. 283–301, May 1998.

[12] F. Hartung and M. Kutter, "Multimedia watermarking techniques," *Proc. IEEE*, vol. 87, no. 7, pp. 1079–1107, Jul. 1999.

[13] F. A. P. Petitcolas, "Watermarking schemes evaluation," *IEEE Signal Process. Mag.*, vol. 17, no. 5, pp. 58–64, Sep. 2000.

[14] B. Chen and G. W. Wornell, "Quantization index modulation: A class of provably good methods for digital watermarking and information embedding," *IEEE Trans. Inf. Theory*, vol. 47, no. 4, pp. 1423–1443, May 2001.

[15] S. Begum and M. S. Uddin, "Digital image watermarking techniques: A review," *Information*, vol. 11, no. 2, Art. no. 110, 2020.

[16] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "HiDDeN: Hiding data with deep networks," in *Computer Vision—ECCV 2018* (Lecture Notes in Computer Science, vol. 11219). Cham, Switzerland: Springer, 2018, pp. 657–672.

[17] FFmpeg Developers, "FFmpeg documentation." [Online]. Available: https://ffmpeg.org/documentation.html

[18] Python Software Foundation, "wave—Read and write WAV files." [Online]. Available: https://docs.python.org/3/library/wave.html

[19] Pillow Developers, "Pillow documentation." [Online]. Available: https://pillow.readthedocs.io/

[20] G. Bradski, "The OpenCV library," *Dr. Dobb's J. Softw. Tools*, 2000.

[21] P. Harvey, "ExifTool." [Online]. Available: https://exiftool.org/