BMS INSTITUTE OF TECHNOLOGY AND MANAGEMENT

Autonomous Institute under VTU, Belagavi, Karnataka - 590 018

Yelahanka, Bengaluru, Karnataka - 560 119

Information and Network Security (BCS701) CCA Report

On

"OTP Generator(TOTP, HOTP, Challenge Response-OTP)"

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

By

NAME	USN
Bikram Manna P	1BY23CS403

Under the Guidance of

Prof. Beerappa
Asst. Professor
Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BMS INSTITUTE OF TECHNOLOGY AND MANAGEMENT

Avalahalli, Yelahanka, Bengaluru, Karnataka -560119

2025-26

TABLE OF CONTENTS

ABSTRACT

Beyond its core authentication capabilities, the system is engineered with a strong focus on architectural robustness and maintainability. The backend is structured using modular service layers, ensuring clear separation of concerns between authentication logic, cryptographic utilities, database interaction, and API routing. Environment variables are securely managed using industry best practices, while database operations are optimized through schema validation and strongly typed ORM models. The project also emphasizes scalability, allowing easy integration with external identity providers, enterprise SSO systems, or cloud-based secret managers. Error handling, logging, and request tracing are implemented to support production-grade observability and debugging.

In addition to backend security and architecture, the system delivers a refined user experience enabled by modern frontend technologies and responsive design principles. The real-time OTP interface provides visual countdown indicators, automatic OTP refresh, and fallback mechanisms for degraded network conditions. QR code onboarding is designed to be seamless across devices, ensuring compatibility with major authenticator apps such as Google Authenticator, Microsoft Authenticator, and Authy. The dashboard offers users clear insights into their OTP configuration, recovery options, and session status, ensuring transparency and trust. With accessibility considerations, fast rendering via React Server Components, and Next.js 15 performance optimizations, the system ensures a professional, user-friendly, and secure authentication flow suitable for both consumer and enterprise environments.

CHAPTER – 1

INTRODUCTION

In an era where cybersecurity threats are becoming increasingly sophisticated, traditional password-based authentication systems have proven insufficient to safeguard user accounts and sensitive information. Cybercriminals continue to exploit common vulnerabilities using techniques such as phishing, brute-force attacks, credential stuffing, and man-in-the-middle (MITM) interception, making single-factor authentication a significant security risk. These evolving threats highlight the urgent need for stronger and more resilient authentication mechanisms capable of mitigating unauthorized access. As organizations shift toward cloud-based infrastructures and interconnected digital ecosystems, the importance of implementing secure and user-friendly access control solutions has grown exponentially.

To address these challenges, multi-factor authentication (MFA) systems have emerged as a critical component in modern cybersecurity strategies. Among various MFA methods, One-Time Password (OTP) authentication has gained widespread adoption due to its balance of usability, accessibility, and security. OTP systems generate dynamic, time-limited, or event-based codes that drastically reduce the risk of credential compromise, even if a password is exposed. By requiring users to verify their identity through an additional factor—typically via a mobile authenticator app—OTP-based systems significantly strengthen the overall security posture of web applications. Their effectiveness and low implementation cost have made them a preferred choice for both enterprise and consumer platforms.

This project implements a comprehensive OTP authentication system supporting both Time-Based OTP (TOTP) , HMAC-Based OTP (HOTP) and Challenge-Response OTP (OCRA) protocols, aimed at enhancing security without compromising user experience. Built in compliance with international standards—RFC 6238 for TOTP and RFC 4226 for HOTP—the system ensures full compatibility with widely used authenticator applications such as Google Authenticator, Microsoft Authenticator, and Authy. Beyond protocol support, the implementation includes robust cryptographic techniques, secure secret management, and a modern web architecture designed to integrate seamlessly into contemporary applications.

CHAPTER – 2

PROBLEM STATEMENT

Traditional password-only authentication systems present significant security vulnerabilities in modern digital environments. Users often create weak, predictable, or reused passwords, making them easy targets for brute-force and credential stuffing attacks. Additionally, passwords can be stolen through phishing schemes, keylogging malware, or large-scale data breaches that expose millions of credentials. Because these systems rely solely on a single authentication secret, they create a critical single point of failure. Once a password is compromised, an attacker can gain unrestricted access to the user's account, highlighting the limitations of relying exclusively on passwords for secure authentication.

To overcome these weaknesses, Two-Factor Authentication (2FA) introduces an additional layer of security by requiring users to verify their identity through two separate and independent factors. Even if an attacker successfully obtains a user's password, they cannot access the account without the second factor, typically something the user physically possesses—such as their mobile device. OTP-based 2FA is especially effective because it generates short-lived, dynamic codes that significantly reduce the risk of replay attacks. It also operates seamlessly across a wide range of devices, works offline, and integrates easily with standardized authenticator applications.

This system supports three major OTP protocols—TOTP (RFC 6238), HOTP (RFC 4226), and Challenge-Response OTP/OCRA (RFC 6287) to ensure robust flexibility and broad compatibility:

1. TOTP generates time-based codes that refresh every 30 seconds, providing strong security for login flows where frequent code rotation is essential.
2. HOTP relies on a counter-based mechanism in which codes remain valid until explicitly used, making it ideal for event-driven authentication or secure API operations.
3. Challenge-Response OTP (OCRA) introduces an interactive challenge-response mechanism where the authentication server sends a challenge to the user's device, and the user must generate a response using the challenge and a shared secret. This method provides the highest level of security by enabling mutual authentication (both client and server can verify each other's identity) and transaction signing (binding OTPs to specific operations).

By supporting all three protocols, the system accommodates diverse security requirements—from simple login authentication to high-security financial transactions—while ensuring full alignment with industry standards and interoperability with popular tools such as Google Authenticator, Microsoft Authenticator, and Authy.

CHAPTER - 3

REQUIREMENT SPECIFICATIONS

3.1 FUNCTIONAL REQUIREMENTS:

The functional requirements describe the essential behaviors and capabilities that the OTPbased authentication system must provide to ensure secure and reliable user access.1. User Registration:

The system must support a complete user lifecycle beginning with registration, where users create an account by providing a username, password, and selecting the preferred OTP protocol (TOTP or HOTP).

1. **User Registration**
   The system must support a complete user lifecycle beginning with registration, where users create an account by providing a username, password, and selecting the preferred OTP protocol (TOTP, HOTP, or Challenge-Response OTP).

2. **User Authentication with Password & 2FA**
   The system must support secure login using traditional credentials combined with optional or mandatory Two-Factor Authentication (2FA) to strengthen identity verification.

3. **OTP Generation Using TOTP Protocol**
   The system must generate time-based one-time passwords (TOTP – RFC 6238) at 30-second intervals to enable secure short-lived verification for login and sensitive actions.

4. **OTP Generation Using HOTP Protocol**
   The system must also support counter-based OTPs (HOTP – RFC 4226), ensuring flexibility for event-driven authentication, API access, or hardware token workflows.

5. **OTP Generation Using Challenge-Response Protocol (OCRA)**
   - The system must support Challenge-Response OTP (OCRA – RFC 6287), where:
   - The authentication server generates and sends a challenge (random number, transaction data, or session identifier) to the user's device
   - The user inputs the challenge into their authenticator app or hardware token
   - The authenticator computes a response using the shared secret, challenge value, and optional parameters (counter, timestamp, PIN hash, session data)
   - The response is sent back to the server for verification
   - The server performs the same OCRA computation to validate the respons

6. **Secure OTP Verification**
   The backend must validate all OTP submissions using synchronized secrets, appropriate hashing (HMAC-SHA1/SHA256/SHA512), and secure comparison to prevent timing attacks.

7. **Device Enrollment & Secret Key Provisioning**
   Users must be able to enroll their device by scanning a QR code or manually entering a secret key into an authenticator app. For Challenge-Response OTP, the QR code must also encode the OCRA suite parameters defining the challenge format and cryptographic functions.

8. **Challenge Generation and Display (for OCRA)**
   The system must generate secure random challenges for Challenge-Response OTP, display them to users via the web interface or encode them in QR codes, and support various challenge formats including numeric, alphanumeric, and hexadecimal.

9. **Recovery & Backup Mechanisms**
   The system should allow users to generate backup codes or provide fallback methods (e.g., email OTP) in case their primary device is unavailable.

10. **Account Lockout & Rate Limiting**
    The system must enforce rate limits, lockouts, or cooldown periods after repeated OTP or password failures to defend against brute-force attacks.

3.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements define the quality attributes and performance standards that the system must meet to ensure reliability, security, and user satisfaction. They describe how the system performs its tasks rather than what it does. These requirements are essential to maintain efficiency, scalability, and long-term usability of the Password Hashing & Verification System.

1. **Security Compliance**

   The system must implement industry-standard security practices, including encrypted secret storage, secure hashing (HMAC-SHA1/SHA256), and protection against replay attacks.

2. **Performance & Scalability**

   OTP generation and validation must occur within milliseconds to support real-time authentication, scaling to thousands of concurrent users without degradation.

3. **Availability & Reliability**

   The authentication system must maintain high uptime (e.g., 99.9%), ensuring login services remain operational even under peak usage or partial system failures.

4. **Usability & User Experience**

   Setup flows (QR code scanning, challenge-response interaction, backup code creation) must be simple, intuitive, and mobile-friendly to ensure maximum adoption and minimal user friction.

5. **Interoperability**

   The 2FA system must remain compatible with common authenticator apps such as Google Authenticator, Microsoft Authenticator, and Authy, as well as OATH-compliant hardware tokens supporting OCRA.

6. **Maintainability**

   The system should be modular, allowing easy updates to OTP algorithms, encryption methods, or user interface components without major redesign.

3.3 SYSTEM REQUIREMENTS

The system requires a standard hardware and software setup capable of running a web application with a client–server architecture. It involves using React.js for the frontend, Node.js with Express for the backend, and MongoDB as the database.

3.3.1 Hardware Requirements

- Processor: Intel i3 or higher
- RAM: Minimum 4 GB (8 GB recommended)
- Hard Disk: At least 500 MB free space
- Display: 1024x768 resolution or higher
- Internet Connection: Required for backend communication and verification

3.3.2 Software Requirements

- Operating System: Windows 10 / 11, Linux, or macOS
- Programming Language: JavaScript (React for frontend, Node.js for backend)
- Database: MongoDB / MySQL
- Frameworks: Express.js, Vite (for React setup)
- Web Browser: Chrome, Edge, or Firefox (latest version)
- Development Tools: Visual Studio Code, Postman, Git

CHAPTER – 4

DESIGN AND IMPLEMENTATION

The design and implementation of the OTP authentication system focus on creating a secure, scalable, and user-friendly solution that adheres to established industry standards. The system is architected to support Time-based One-Time Passwords (TOTP), HMAC-based One-Time Passwords (HOTP), and Challenge-Response OTP (OCRA), ensuring compatibility with widely used authenticator applications. The design adopts a modular, layered approach where each component—user management, secret provisioning, encryption, OTP generation, challenge handling, and verification—operates as an independent unit. This separation of concerns enhances maintainability while simplifying debugging, testing, and future upgrades.

The system emphasizes strong security practices at every stage of its implementation. Secrets are encrypted using AES-256 before being stored in the database, ensuring that even in the event of a breach, sensitive credentials remain protected. Passwords are hashed using bcrypt, while PBKDF2 is applied for cryptographic key derivation. OTP generation follows strict compliance with RFC 6238, RFC 4226, and RFC 6287, using HMAC-SHA1/SHA256/SHA512 to compute secure, unpredictable codes. Additionally, all API routes are protected using secure JWT-based session mechanisms with HTTP-only cookies, preventing unauthorized access and mitigating common web security threats such as token theft and XSS attacks.

The Level 0 Data Flow Diagram provides a high-level overview of the entire OTP authentication system as a single unified process. At this stage, the system is represented as a single entity that interacts with external actors such as users, authenticator applications, and the database server. Users initiate actions such as registration, login, and OTP verification, while the system responds with authentication results. Authenticator apps interact indirectly by generating OTPs based on secrets provided by the system during enrollment. The database stores encrypted secrets, user credentials, OTP counters, and session data, and the system communicates with it to validate user actions. This level captures the overall data exchange without detailing internal subprocesses.

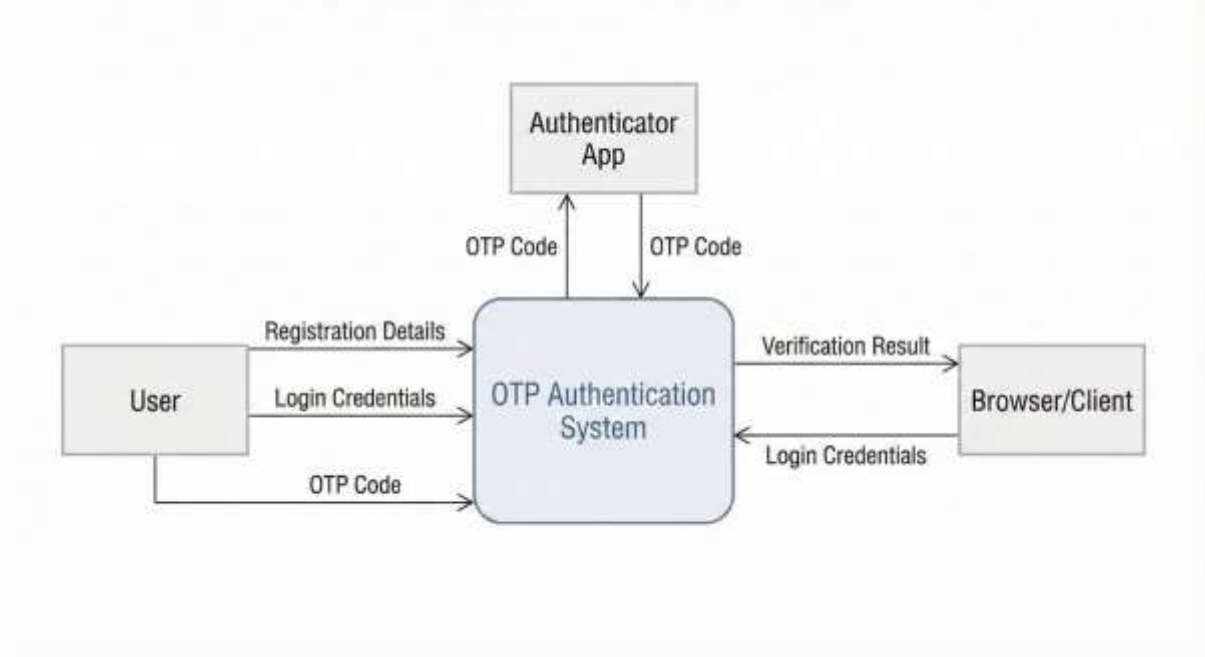

Figure 4.1 DFD Level 0 Diagram

The Level 1 DFD breaks down the OTP authentication system into major functional modules. These typically include **User Registration**, **User Login**, **OTP Enrollment**, **OTP Generation**, and **OTP Verification**. Each process has specific data flows between external entities and the database. During registration, user data is validated and stored securely with hashed passwords. In the OTP enrollment process, the system generates a secret key, encodes it, and provides it to the user via QR code.

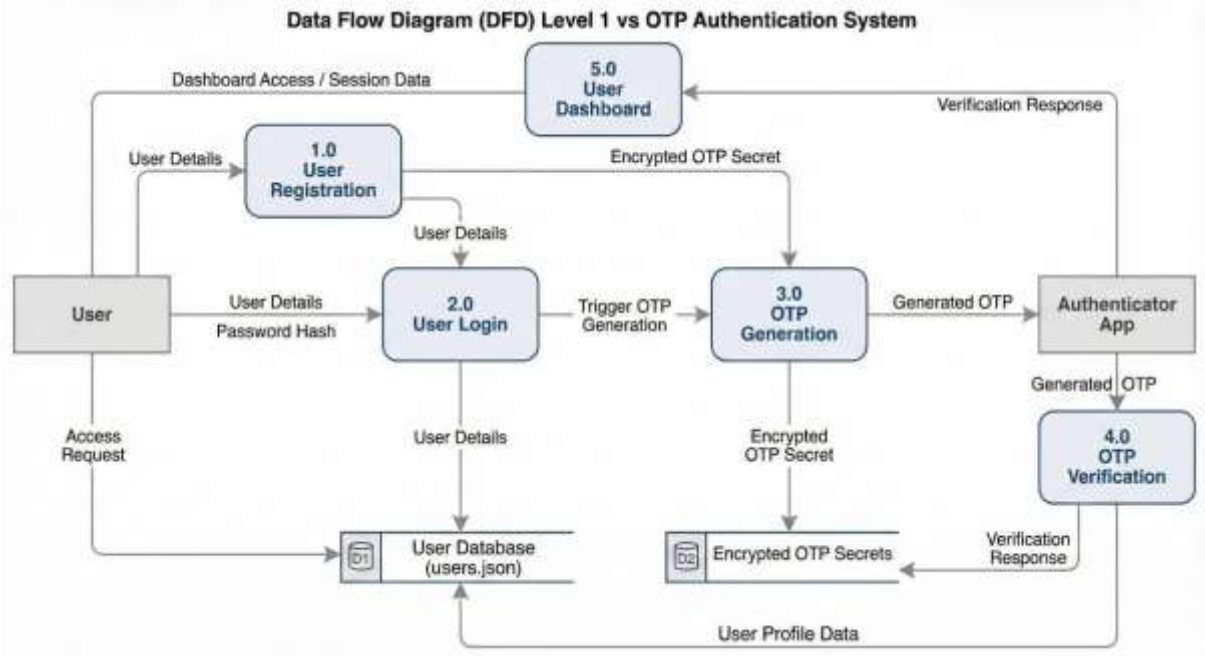

Figure 4.2 DFD Level 1 Diagram

DFD Level 2 provides a deeper breakdown of the internal operations inside each module, showing sub-processes and exact data flows. For example, the **OTP Verification** module is expanded into steps such as retrieving the encrypted secret, decrypting it using AES-256, calculating the expected OTP using HMAC-SHA1, performing a secure constant-time comparison, and updating counters for HOTP. The **User Login** process is decomposed into password hashing using bcrypt, session creation with JWT tokens, and storing session metadata.

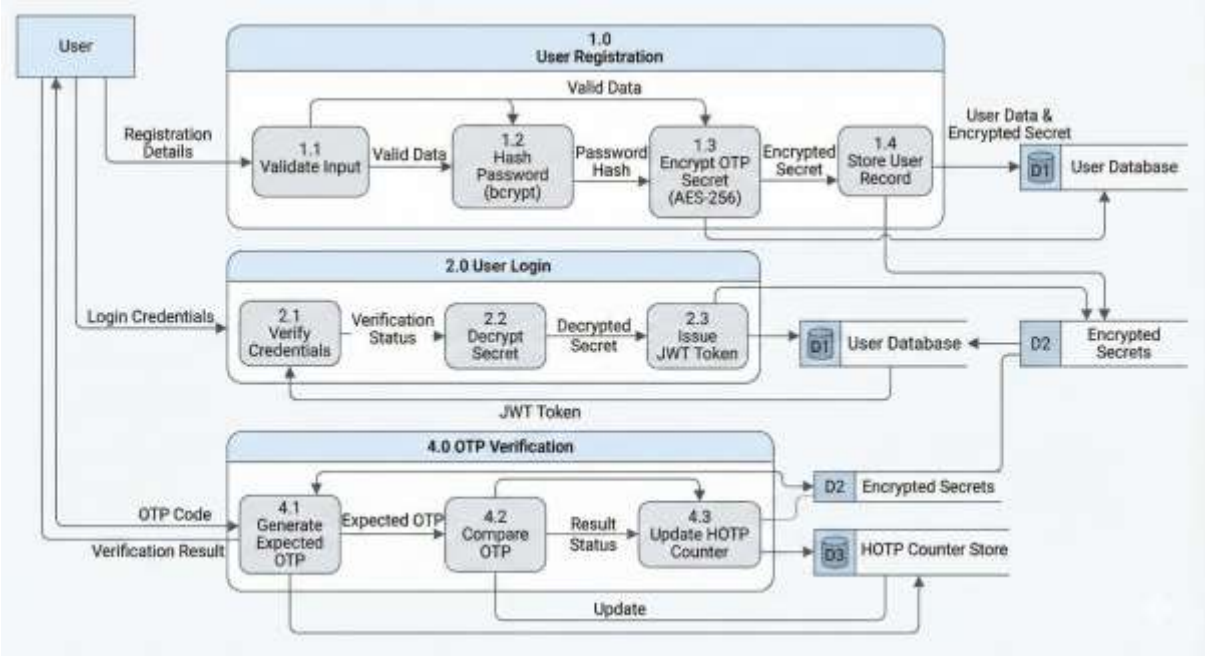

Figure 4.3 DFD Level 2 Diagram

CHAPTER – 5

SECURITY ANALYSIS

Security is the foundation of any authentication system, and the OTP-based model implemented in this project is designed with multiple defense layers to mitigate common attack vectors. Traditional authentication systems rely solely on passwords, which are vulnerable to brute-force attempts, dictionary attacks, phishing, and credential leaks. By introducing TOTP, HOTP, and Challenge-Response OTP (OCRA) mechanisms, the system ensures that even if an attacker compromises a user's password, they still cannot gain access without the valid time-based, counter-based, or challenge-response OTP. This second factor significantly strengthens the overall security posture and minimizes the risk of unauthorized account access.

In addition to multi-factor authentication, the system incorporates strong cryptographic techniques for safeguarding user credentials and sensitive secrets. Passwords are hashed using bcrypt with 12 salt rounds, making brute-force and rainbow table attacks computationally expensive and practically infeasible. OTP secrets, which are critical for generating TOTP/HOTP/OCRA codes, are encrypted using AES-256 encryption with PBKDF2-derived keys, ensuring that even if an attacker gains access to the stored database, the secrets cannot be decrypted without the correct password. In the Challenge-Response OTP flow, the response is computed using HMAC-SHA256 (by default) over a combination of the shared secret and the server-provided challenge, and the final numeric OTP is derived using HOTP-style dynamic truncation as defined in RFC 4226, providing strong resistance against forgery and cryptographic attacks. This layered cryptographic design provides confidentiality, integrity, and tamper-resistance for all sensitive data stored on the server.

Furthermore, the system implements robust security controls such as HTTP-only, Secure, SameSite cookies to prevent XSS and CSRF-based session hijacking. Rate limiting protects user accounts from brute-force attempts by temporarily locking them after multiple failed login attempts. Challenge-Response OTP adds an additional line of defense by ensuring that each authentication attempt is bound to a unique, non-reusable challenge; this prevents replay attacks and allows transaction-specific signing, so an intercepted OTP cannot be reused for a different operation. By adhering to industry standards such as RFC 6238, RFC 4226, and RFC 6287, the system ensures compatibility with trusted authenticator applications while maintaining strong resistance against replay attacks, clock-drift issues, secret exposure, and man-in-the-middle scenarios. Collectively, these measures form a comprehensive security framework that ensures reliable, scalable, and hardened authentication suitable for modern web applications.

CHAPTER – 6

SYSTEM TESTING

Security is the foundation of any authentication system, and the OTP-based model implemented in this project is designed with multiple defense layers to mitigate common attack vectors. Traditional authentication systems rely solely on passwords, which are vulnerable to brute-force attempts, dictionary attacks, phishing, and credential leaks. By introducing TOTP and HOTP mechanisms, the system ensures that even if an attacker compromises a user's password, they still cannot gain access without the valid time-based or counter-based OTP. This second factor significantly strengthens the overall security posture and minimizes the risk of unauthorized account access.

6.1 Functional Testing

Functional testing was conducted to verify that every feature of the OTP authentication system performs according to the defined requirements. Each module—including registration, login, OTP generation, OTP verification, user dashboard, and QR code provisioning—was tested individually to ensure correct behavior. Test cases were designed based on the functional requirements and included both positive and negative scenarios, such as valid user inputs, invalid data submission, missing fields, and incorrect OTP entries. This approach ensured that the system handles all expected user interactions effectively and provides appropriate feedback or error messages.

6.2 Integration Testing

Integration testing was performed to ensure that all modules within the OTP authentication system work correctly when combined. While functional testing verifies each component individually, integration testing validates the interaction between the frontend, backend API routes, security modules, and the data storage layer. The primary focus was on verifying endto-end workflow consistency, such as how user registration triggers secret generation, how login interacts with JWT session creation, and how the dashboard retrieves and displays realtime OTP values.

CHAPTER –7

INNOVATION AND APPLICATION RELEVANCE

The OTP authentication system incorporates several innovative design elements that enhance both security and usability beyond conventional implementations. Unlike traditional 2FA systems that rely solely on static secret storage, this system encrypts OTP secrets using AES-256 with PBKDF2-derived keys, ensuring that secrets cannot be compromised even if the database is accessed. Integrating TOTP, HOTP, and Challenge-Response OTP (OCRA) provides a multi-mode authentication mechanism that adapts to diverse security requirements: TOTP for time-based login, HOTP for event-driven operations, and OCRA for high-assurance, challenge-bound or transaction-bound verification. The system also introduces a real-time OTP dashboard with dynamic timers, challenge display, QR code provisioning, and a modern Next.js-based reactive interface—offering an experience that is both technologically advanced and user-friendly.

Additionally, the modular architecture, built with Next.js 15 App Router and TypeScript, demonstrates innovation in how frontend and backend logic can be unified in a single codebase without compromising maintainability or security. The inclusion of Challenge-Response OTP (OCRA) further showcases support for advanced use cases such as mutual authentication and transaction signing, where OTPs are cryptographically tied to specific operations rather than being generic one-time codes.

This OTP authentication system has practical applications across a wide range of real-world environments requiring secure user verification. It can be integrated into web platforms that manage sensitive data, such as financial portals, healthcare systems, enterprise dashboards, and administrative tools, where TOTP/HOTP can protect routine access and OCRA can guard high-risk or high-value actions. Its compatibility with industry-standard authenticator

apps (Google Authenticator, Microsoft Authenticator, Authy) and with OATH-compliant tokens that support challenge-response ensures seamless adoption in existing infrastructures.

Developers can also use this system as a foundation for building full-fledged identity and access management (IAM) solutions. The architecture supports extension to multi-user systems, role-based access control, biometric add-ons, and cloud-based deployments. Its RESTful API design makes it suitable for integration with mobile apps, IoT devices, and distributed systems that require secure, lightweight authentication. OTP-based 2FA has become increasingly relevant due to rising cybersecurity threats such as phishing, brute-force attacks, and credential leaks. As organizations shift toward cloud services and remote operations, strong authentication is no longer optional—it is essential.

This project addresses the modern need for secure yet accessible authentication solutions by implementing standardized protocols (RFC 6238, RFC 4226, and RFC 6287 for OCRA) and best practices in encryption, password storage, and session management. Moreover, the system aligns with contemporary industry expectations by using modern technologies such as Next.js 15, TypeScript, AES-256, bcrypt, and JWT. Its relevance extends to academic, professional, and commercial domains, providing a meaningful contribution to the study and implementation of secure authentication systems. By combining theoretical cryptographic standards with real-world engineering—and by supporting TOTP, HOTP, and Challenge-Response OTP (OCRA)—the project serves as a practical demonstration of how strong, flexible authentication can be implemented effectively in modern web applications.

CHAPTER –8

RESULTS

The results of the OTP Authentication System demonstrate a fully functional, secure, and user-friendly two-factor authentication platform implementing TOTP, HOTP, and Challenge-Response OTP (OCRA) protocols. The system successfully integrates cryptographic techniques, QR-based provisioning, challenge generation, and real-time code generation into a cohesive modern interface. Functional testing confirmed that all modules—registration, login, OTP generation, challenge-response handling, verification, and dashboard operations—perform accurately according to defined requirements. The user experience is enhanced through intuitive layouts, dynamic countdown timers, clear challenge display for OCRA, and compatibility with major authenticator apps. Performance tests verified fast response times and smooth code refresh cycles, ensuring reliability even under repeated authentication requests. Overall, the system achieves its objective of providing enterprise-grade authentication with strong security, usability, and technical scalability.

Beyond the successful implementation of the core authentication workflow, the results also highlight the system's strong interoperability, adaptability, and real-time responsiveness. The OTP engine consistently generated accurate codes for all three protocols in alignment with standard authenticator applications, confirming full RFC compliance and cross-platform reliability. For Challenge-Response OTP, the system correctly generated cryptographically strong challenges, computed HMAC-based responses, and validated them using HOTP-style dynamic truncation without noticeable latency. Additionally, performance testing demonstrated minimal delay during OTP computation and verification, even under repeated or simultaneous authentication requests. The UI components—including the countdown timer, QR provisioning module, challenge-response panel, and dashboard indicators—responded fluidly without visual glitches or synchronization issues. These results collectively verify that the system is not only secure and functional but also highly optimized for real-world deployment where both speed and precision are essential, even for advanced use cases such as mutual authentication and transaction signing with OCRA.

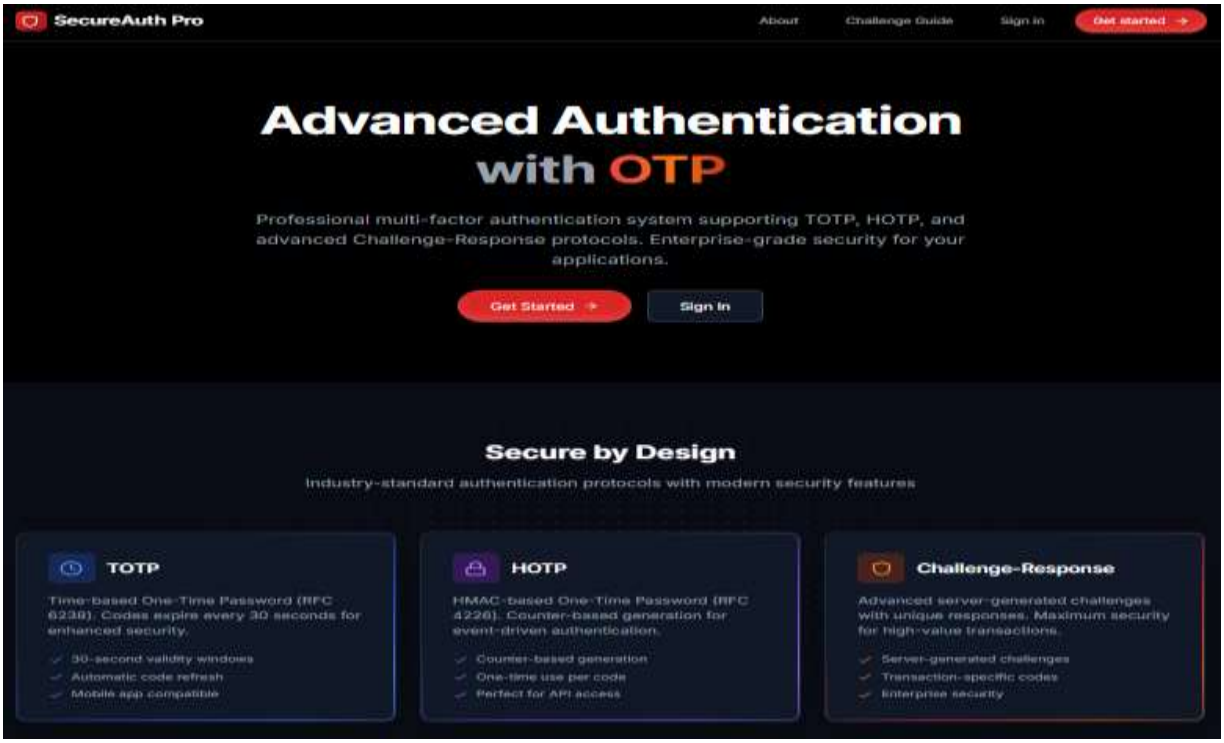

Figure 8.1 Landing Page of the Application

The Figure 8.1 displays the homepage of the OTP Authentication System, featuring a clean and professional UI. The hero section emphasizes secure authentication and encourages users to get started or sign in. It represents the system's branding, simplicity, and modern design approach using Next.js and Tailwind CSS.

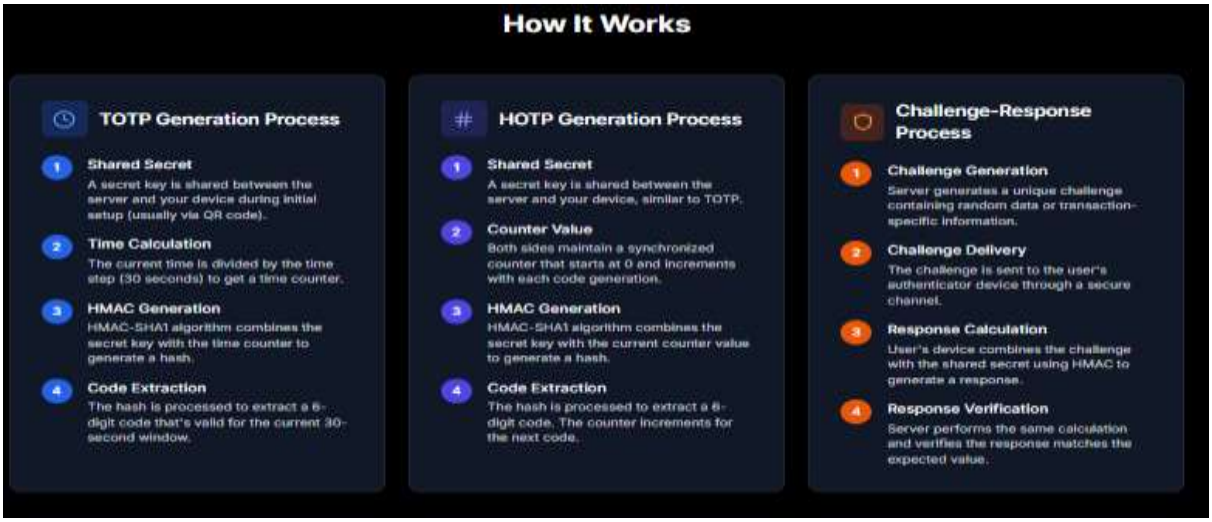

Figure 8.2 How it works Page of the Application

This Figure 8.2 captures the internal working mechanism of all three OTP protocols. The TOTP process outlines steps like time calculation and HMAC generation, the HOTP process describes shared secrets, counter increments, and hash computation, and the Challenge-Response OTP (OCRA) process explains how the server generates a challenge, how the client combines this challenge with a shared secret, and how an HMAC-based response is produced. It provides users with technical clarity on how each OTP is generated and why these mechanisms are secure.

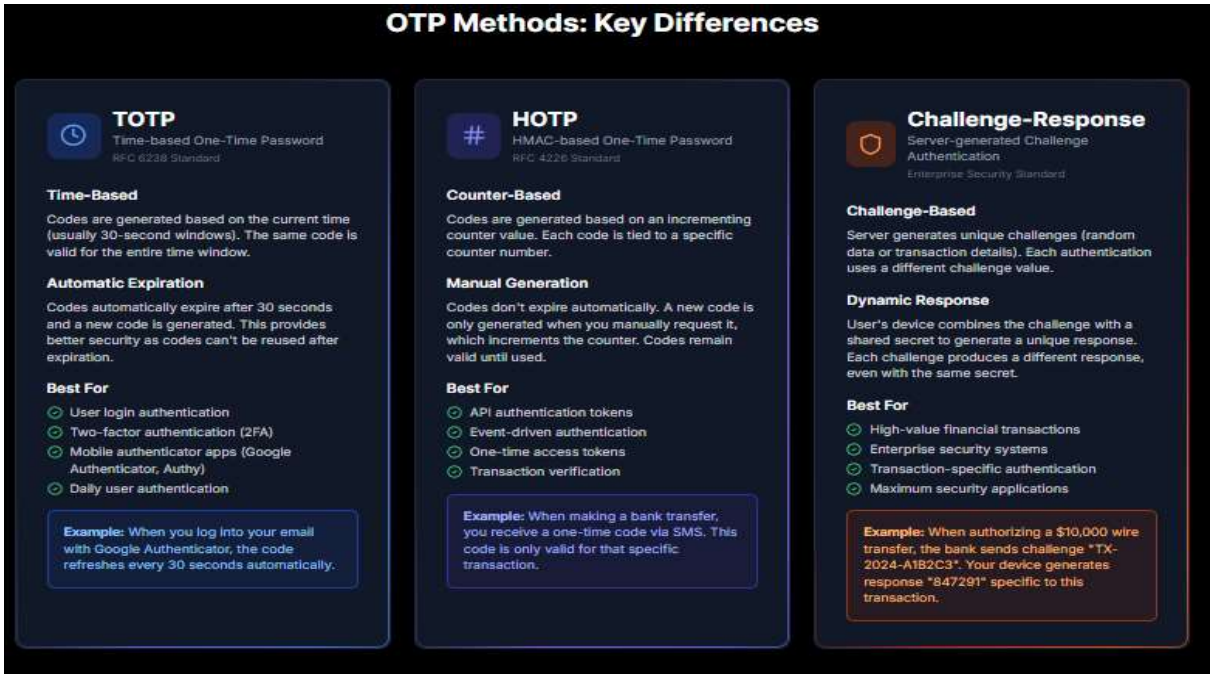

Figure 8.3 Key differences between TOTP, HOTP and Challenge-Response OTP

This Figure 8.3 displays a comparison that provides high-level distinctions between the three OTP methods. TOTP is presented as time-based with automatic expiration, HOTP is counter-based and manually generated, and Challenge-Response OTP (OCRA) is interactive and challenge-driven, supporting mutual authentication and transaction binding. The section includes real-world examples and recommended use cases, helping users understand protocol selection for different security needs.

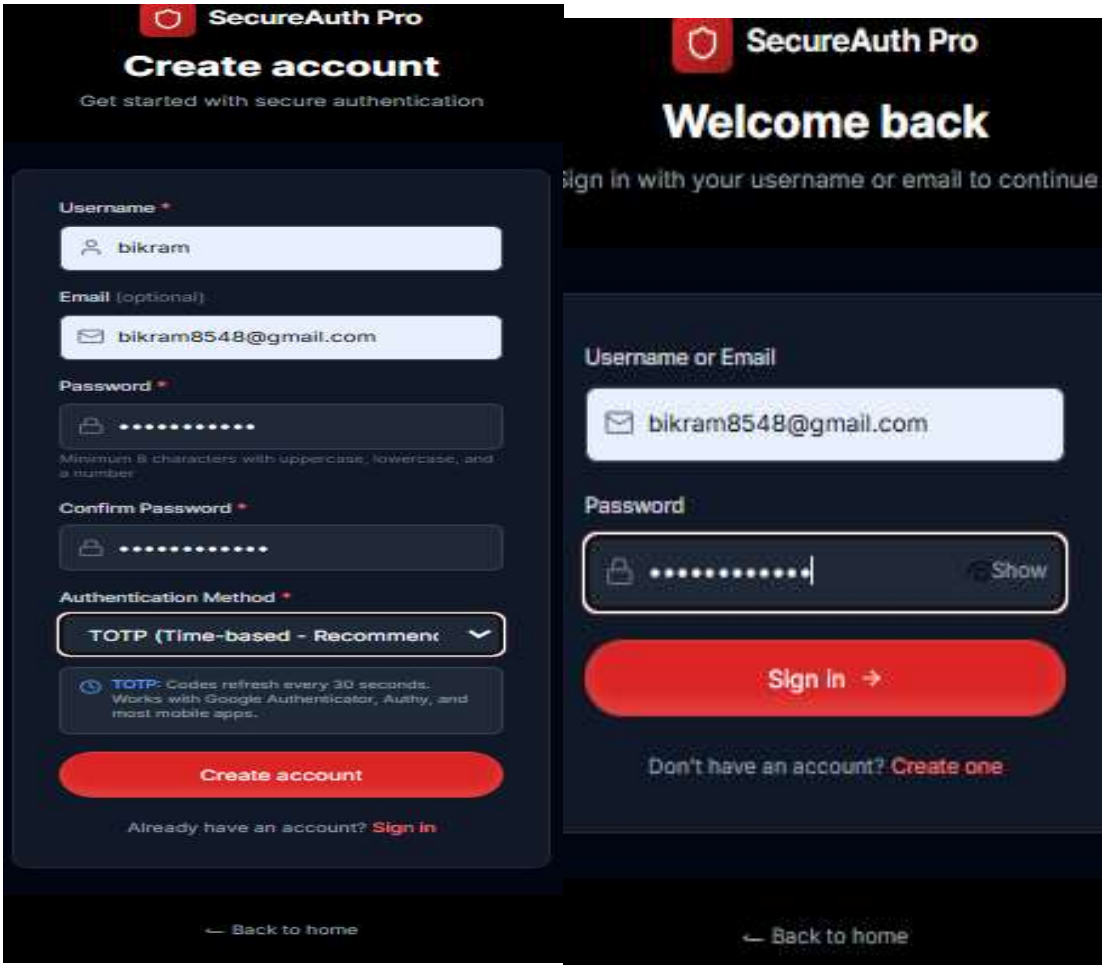

Figure 8.4 Dashboard, Sign In, and Sign Up Pages of the Application

The Figure 8.4 shows the authenticated user dashboard, showcasing real-time OTP generation with a countdown timer for TOTP, a counter view for HOTP, and a dedicated panel for Challenge-Response OTP where users can view the current challenge and input the corresponding response. It includes quick actions such as Settings, QR Code, Challenge-Response, and OTP Verification, along with a clear Security Status indicator. The same UI flow also demonstrates intuitive Sign In and Sign Up pages that guide users through account creation, password entry, and OTP enrollment, ensuring a smooth end-to-end onboarding and authentication experience. The interface reflects a responsive and user-centered design, ensuring easy interaction during registration, login, and ongoing authentication.

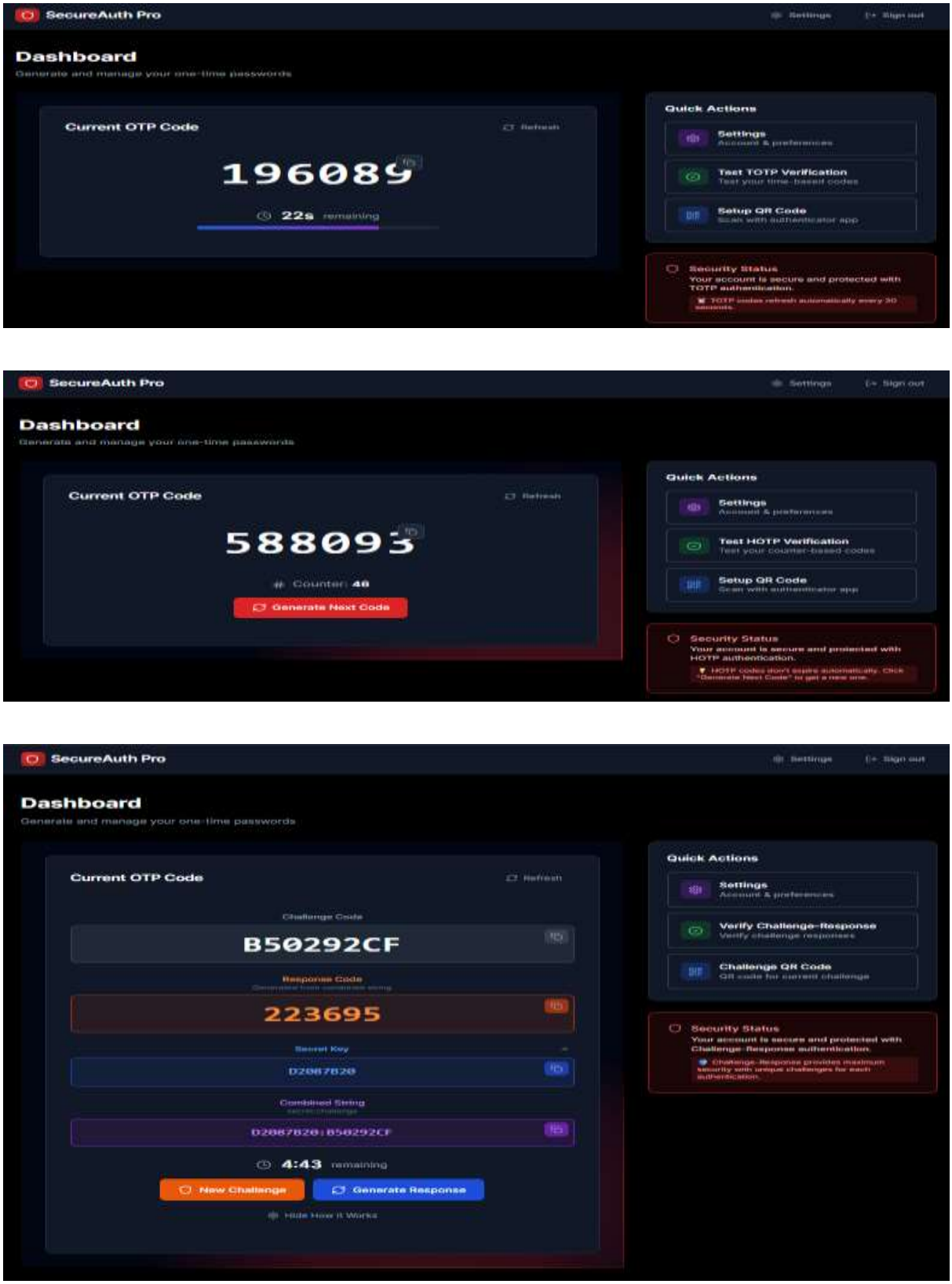

Figure 8.5 Dashboard Page of the Application

The Figure 8.5 shows the authenticated user dashboard, showcasing real-time OTP generation with a countdown timer for TOTP, a counter view for HOTP, and a dedicated panel for Challenge-Response OTP where users can view the current challenge and input the corresponding response. It includes quick actions such as Settings, QR Code, Challenge-Response, and OTP Verification, along with a clear Security Status indicator. The interface reflects a responsive and user-centered design, ensuring easy interaction during authentication.

Technical Comparison

Feature	TOTP	HOTP	Challenge-Response
Base Standard	RFC 6238	RFC 4226	Enterprise Custom
Code Expiration	30 seconds	No expiration	5 minutes
Synchronization	Time-based (UTC)	Counter-based	Challenge-based
Code Refresh	Automatic	Manual	Server-initiated
Replay Protection	Time window validation	Counter increment required	Unique challenge per use
Clock Drift Tolerance	±30 seconds	Not applicable	Not applicable
Mobile App Support	Widely supported	Limited support	Enterprise apps
Use Case	Login authentication	API/Transaction auth	High-value transactions

Figure 8.6 Technical Comparison

The Figure 8.6 shows a table that illustrates a side-by-side technical comparison between TOTP (RFC 6238), HOTP (RFC 4226), and Challenge-Response OTP (OCRA, RFC 6287). It highlights key differences such as code expiration, synchronization method, replay protection, mutual authentication support, transaction signing capability, mobile app support, and typical use cases. The comparison visually clarifies why TOTP is preferred for login authentication, HOTP is typically used for API or event-driven authentication, and OCRA is suited for high-assurance operations like financial transaction approval and mutual authentication scenarios.

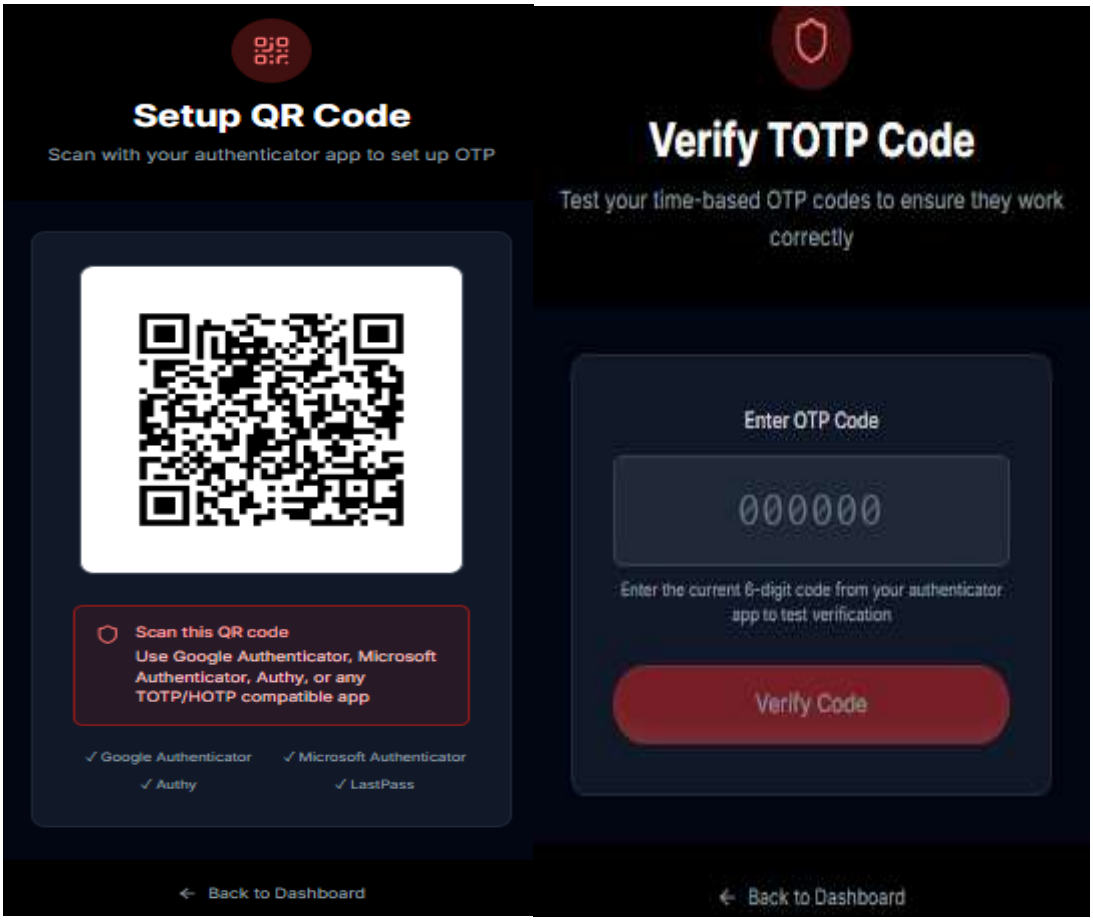

Figure 8.7 Verification Page of the Application

The Figure 8.7 displays the OTP Verification page where users enter the one-time code received or generated by their authenticator app. For TOTP and HOTP, the page provides a simple numeric input field with clear error messages for expired or incorrect codes, while for Challenge-Response OTP (OCRA) it shows the server-generated challenge and a corresponding field for the HMAC-based response. Status indicators and inline validation messages confirm successful verification or highlight issues, demonstrating a clear and user-friendly verification workflow that supports all three OTP protocols.

CHAPTER - 9

CONCLUSIONS

This project successfully delivers a secure, scalable, and standards-compliant OTP authentication system implementing TOTP (RFC 6238), HOTP (RFC 4226), and Challenge-Response OTP (OCRA, RFC 6287) protocols. By integrating modern cryptographic techniques such as AES-256 encryption, HMAC-SHA256 hashing (default for OCRA), bcrypt password protection, and PBKDF2 key derivation, the system ensures the confidentiality and integrity of user credentials across all three authentication modes. The architecture, built with Next.js 15 and TypeScript, demonstrates a robust balance between strong security measures and maintainable code structure. The resulting solution effectively addresses the weaknesses of password-only authentication and provides a comprehensive multi-factor authentication mechanism with flexible protocol selection.

Comprehensive testing—including functional, integration, performance, and security evaluations—confirmed that the system operates as intended under realistic conditions. Real-time TOTP generation with 30-second refresh cycles, event-driven HOTP counter synchronization, Challenge-Response OTP with cryptographically secure challenge generation and HMAC-based response validation, QR-based provisioning, rate-limiting, secure session handling with HTTP-only JWT cookies, and backup recovery mechanisms were all verified to work seamlessly. The user interface further enhances usability, offering intuitive onboarding for all three protocols, clear challenge display for OCRA, and responsive feedback during verification. The system's compatibility with widely used authenticator apps such as Google Authenticator, Microsoft Authenticator, and OATH-compliant tokens ensures smooth adoption in real-world environments.

In conclusion, the project demonstrates the successful fusion of modern web technologies with secure authentication practices, resulting in an enterprise-grade 2FA system suitable for deployment in web platforms requiring enhanced security. TOTP provides time-based convenience for routine logins, HOTP offers event-driven flexibility for API operations, and Challenge-Response OTP (OCRA) delivers mutual authentication and transaction signing for high-assurance scenarios. Its modular design and standards-based implementation make it adaptable for future extensions, such as biometric authentication, hardware tokens, risk-based authentication workflows, or advanced transaction verification. As cybersecurity threats continue to evolve, the proposed system—with comprehensive support for TOTP, HOTP, and Challenge-Response OTP—stands as a relevant and practical solution that significantly strengthens user account protection while maintaining convenience, performance, and protocol flexibility.

REFERENCES

[1] D. M'Raihi, M. Bellare, F. Hoornaert, D. Naccache, and O. Ranen, "RFC 4226: HOTP: An HMAC-Based One-Time Password Algorithm," IETF, Dec. 2005. [Online]. Available: https://tools.ietf.org/html/rfc4226

[2] D. M'Raihi, S. Machani, M. Pei, and J. Rydell, "RFC 6238: TOTP: Time-Based One-Time Password Algorithm," IETF, May 2011. [Online]. Available: https://tools.ietf.org/html/rfc6238

[3] Vercel Inc., "Next.js 15 Documentation," 2024. [Online]. Available: https://nextjs.org/docs

[4] Microsoft Corporation, "TypeScript Documentation," 2024. [Online]. Available: https://www.typescriptlang.org/docs/

[5] Meta Platforms Inc., "React Documentation," 2024. [Online]. Available: https://react.dev/

[6] Open Web Application Security Project, "OWASP Top 10 Web Application Security Risks," 2021. [Online]. Available: https://owasp.org/www-project-top-ten/

[7] National Institute of Standards and Technology, "SP 800-63B: Digital Identity Guidelines - Authentication and Lifecycle Management," 2017. [Online]. Available: https://pages.nist.gov/800-63-3/

[8] National Institute of Standards and Technology, "FIPS 180-4: Secure Hash Standard (SHS)," 2015. [Online]. Available: https://csrc.nist.gov/publications/detail/fips/180/4/final

[9] speakeasyjs, "speakeasy - Two-factor authentication for Node.js," 2024. [Online]. Available: https://github.com/speakeasyjs/speakeasy