Abstract

The Online One-Time Registration (OTR) Verification System is designed to address the inefficiencies and vulnerabilities associated with traditional manual verification processes in user registration systems. In many organizations, the process of verifying user-submitted documents such as identity proofs, academic certificates, and other essential records can be time-consuming and prone to human error. The proposed system utilizes advanced technologies like Optical Character Recognition (OCR) and secure cloud storage to automate and streamline this process.

The core objective of this system is to significantly reduce the time and effort required for document verification while ensuring a high level of security and accuracy. OCR technology is employed to automatically extract and validate key information from scanned documents, reducing the reliance on manual checking. This information is cross-verified with databases to ensure authenticity, and any discrepancies are flagged for further review by administrators.

The Online OTR Verification System also integrates a real-time notification feature, keeping users informed about the status of their document verification process. This transparency ensures a smoother and more reliable experience for users, eliminating the uncertainty associated with manual verification. Additionally, the system employs cloud-based storage to securely store all documents and verification data, ensuring data integrity and preventing unauthorized access.

Acknowledgment

We extend our sincere gratitude to Dr. Kiran Jyoti, Head of the Computer Science and Engineering Department at GNDEC, Ludhiana, for her constant guidance, encouragement, and invaluable insights throughout the development of our sholarship portal. Her unwavering support has been instrumental in shaping this project. We are deeply indebted to our project guide, Dr. Amandeep Kaur Sohal, for her expert supervision, patient mentorship, and constructive feedback at every stage of this work. Her technical expertise and dedication played a pivotal role in overcoming challenges and ensuring the successful completion of this project. We also express our heartfelt appreciation to all the faculty members of the Computer Science and Engineering Department, GNDEC, for their intellectual support, encouragement, and valuable suggestions. Their guidance helped refine the functionality and usability of the portal. Lastly, we acknowledge the contributions of everyone friends, family, and peers who supported us directly or indirectly during this journey. Their motivation and cooperation were vital in bringing this project to fruition.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Introduction to Project

In the era of digital transformation, the need for efficient and secure identity management systems has become paramount. One Time Registration (OTR) systems offer a convenient solution by allowing users to submit their details and documents once, which can then be used for multiple processes such as admissions, job applications, or government schemes.[3]

Despite the convenience, conventional OTR systems suffer from significant drawbacks. Most systems still rely heavily on manual verification of documents, which is time-consuming, error-prone, and resource-intensive. Additionally, users are often left unaware of their application status due to a lack of automated feedback mechanisms.

To address these challenges, the **Online OTR Verification System** has been developed. It focuses specifically on automating the verification of the OTR document — a single, comprehensive form submitted by users — while allowing the upload of other supporting documents for reference. The system leverages modern technologies such as Optical Character Recognition (OCR) for OTR validation and ensures the safe storage of all user-uploaded files in a secure cloud environment. It also includes administrative dashboards and notification services to streamline the verification workflow.[3]

This targeted approach not only reduces human workload and verification errors but also enhances the user experience by providing timely updates and a centralized portal for document submission and tracking.

1.2 Project Category

- **Category:** Application-Based Project

- **Type:** Web Application
- **Domain:** Online Verification Systems

1.3 Objectives

The primary objectives of the Online OTR Verification System are as follows:

1. Design and implementation of a web-based platform for scholarship form submission and verification.
2. Facilitate the college scholarship department to check the status of Student's scholarship application.
3. Auto-generate and send confirmation messages to the user's email upon successful verification and submission.

1.4 Problem Formulation

Traditional verification systems that rely on manual processing of user-submitted documents introduce several inefficiencies and risks:

1. **Delays:** Manual checking leads to slower turnaround times.
2. **Human Error:** Verification accuracy varies based on personnel expertise.
3. **Data Management:** Physical or unstructured digital files are difficult to organize and retrieve.
4. **Security Concerns:** Sensitive user documents are often stored without adequate protection.

The Online OTR Verification System addresses these issues by digitizing the OTR verification process and securely storing other uploaded documents for future reference.

1.5 Identification/Recognition of Need

With increasing application volumes in educational institutions, government bodies, and corporations, there is an urgent need for:

1. A system that automates repetitive and labor-intensive document verification.
2. A scalable platform capable of handling high-volume document submissions.
3. Real-time feedback and transparency for users submitting important credentials.
4. Secure document storage with easy retrieval and administrative oversight.

The Online OTR Verification System fulfills these needs by automating OTR validation while providing secure document handling infrastructure.

1.6 Existing System

In the current approach adopted by many organizations:

- Applicants upload scanned documents, including the OTR and additional certificates.
- Every document must be manually verified by administrative personnel.
- There is no mechanism for automatic status updates or alerts to users.
- Administrators often have to search through disorganized files and folders.
- Security and privacy protocols are inconsistently enforced.

Limitations:

- Time-consuming manual processes.
- Lack of systematized document handling and tracking.
- No audit trail for tracking verification steps.

- Poor communication between applicants and verifiers.

1.7 Proposed System

The proposed system aims to create an automated and secure web-based solution for OTR verification. The key characteristics include:

1. **Focused OTR Verification:** Only the OTR document is automatically processed using OCR and backend validation rules (e.g., name matching, format checks).

2. **Document Upload Portal:** Users can upload additional documents such as ID proof or academic records, which are stored securely but not automatically verified.

3. **Notification System:** Users receive real-time updates via email after OTR verification is completed.

4. **Administrator Control:** An admin dashboard allows oversight of all submissions, with features for manual intervention, status updates, and audit logging.

5. **Scalable and Secure Infrastructure:** The system is hosted on a secure cloud platform with encrypted file storage, ensuring data privacy and scalability.

Major Modules:

1. **User Login and Authentication:** Secure login and access control for students or applicants.

2. **OTR Document Upload and Verification:** OCR-based verification flow for submitted OTR documents.

3. **Additional Document Upload:** Upload mechanism for storing other documents in the database or cloud storage.

4. **Email Notification System:** Automated alerts sent to users upon status changes.

1.8 Unique Features of the Proposed System

The Online OTR Verification System offers several innovative features:

- **Selective Verification:** Only the OTR document undergoes automated validation, reducing unnecessary processing overhead.
- **Secure Cloud Storage:** Uploaded files are stored in encrypted cloud environments, protecting user privacy.
- **Real-Time User Notifications:** Status updates are sent via email to keep users informed about verification outcomes.
- **Role-Based Access:** Differentiated access levels for users and administrators to ensure data integrity.
- **Responsive Interface:** The user interface is mobile-friendly and accessible across various devices.
- **Manual Admin Review Capability:** While automation is applied to OTRs, admins can manually review other documents when needed.
- **Scalability:** Architecture supports expansion to handle thousands of users and submissions.

Chapter 2

Requirement Analysis and System Specification

2.1 Feasibility Study (Technical, Economic, Operational)

2.1 Technical Feasibility

The technical feasibility analysis examines the tools and technologies used in the system. The core technical feature of the Online OTR Verification System is the automated verification of OTR documents using Optical Character Recognition (OCR). OCR is applied to extract textual information from uploaded OTR documents and validate it against the user's registration data stored in the system. Other documents such as identity proof, academic certificates, or supporting files can be uploaded by users, but they are not subject to automated verification; they are stored securely for administrative access or reference.[2]

The system utilizes widely available and well-supported technologies, including cloud platforms for storage and hosting, and proven OCR libraries like Tesseract. Since the application is built with a modern JavaScript stack (React.js, Node.js, MongoDB), the development team possesses the necessary skills and expertise to successfully implement and maintain the system. Overall, the required technology is readily available and technically feasible for deployment.[2]

2.1 Economic Feasibility

Economic feasibility assesses whether the anticipated benefits of the system justify the costs involved in its development, deployment, and maintenance. Since the system focuses primarily on automating the OTR verification process, cost savings are expected in terms of reduced manual effort and improved accuracy.[1]

(a) **Software Development Costs:**

- **Human Resources:** Includes compensation for developers, testers, and project coordinators working on building the OTR verification system and document upload module.
- **Development Tools:** Use of open-source frameworks (e.g., React, Node.js, MongoDB) reduces software licensing costs. Any additional cost may come from third-party APIs or IDE subscriptions if applicable.
- **Time Investment:** Measured in terms of developer hours invested in frontend development, backend logic, OCR integration, and cloud setup.

(b) **Third-Party Services:**

- **OCR Processing:** Costs may arise from using OCR APIs or computational resources required to process document images.
- **Notifications and Integrations:** Optional expenses may include email services or notification APIs for user updates.

(c) **Maintenance and Support:**

- **Regular Updates:** Time and effort required to improve the verification accuracy or enhance the user experience.
- **Support Services:** Providing user support, handling technical queries, and ensuring high uptime.

Since the system automates the most critical and frequently used process (OTR verification), it yields significant time and cost savings. Other document uploads serve a supporting role and do not require costly validation steps, thus keeping operational expenses low.

2.1 Operational Feasibility

The operational feasibility focuses on whether the system will work effectively in real-world scenarios and integrate smoothly into existing organizational workflows.

(a) **Core Functionality Focus:** The system is primarily designed to automate the verification of OTR documents. Other documents, such as ID proofs or certificates, are accepted for upload and stored securely, but they are not automatically verified. This targeted approach ensures that critical verification is handled efficiently, while still offering a platform for centralized document submission.[2]

(b) **Integration with Existing Infrastructure:** Since the system is web-based, it runs on standard hardware and browsers, requiring no additional investment in IT infrastructure. It can be accessed securely by both users and verifiers, and integrates well with existing organizational workflows.[3]

(c) **Improved Efficiency:** Automating the OTR verification process significantly reduces the time and effort involved in manual verification. With real-time status updates and centralized storage, document handling becomes more streamlined, and data retrieval is faster.[3]

(d) **User Adoption and Training:** The system features a simple, intuitive interface. Since only the OTR verification process is automated, the training required for users is minimal. Uploading other documents follows standard file upload procedures familiar to most users.[2]

(e) **Monitoring and Maintenance:** Logging and monitoring tools are integrated to track document uploads, verification outcomes, and user activity. Regular backups ensure data protection, and any issues can be addressed by the support team promptly.[1]

(f) **Benefits to Operations:** The system reduces dependency on paper documents and physical

records, eliminates human errors in OTR validation, and speeds up verification turnaround time. Secure access to uploaded files improves document management and compliance with data retention policies.[2]

2.2 Software Requirement Specification (SRS)

2.2.1 Data Requirements

The data requirements define the types of data that will be collected, processed, and stored throughout the lifecycle of the OTR verification system. These requirements ensure that the system performs its intended verification tasks while maintaining data integrity and privacy.[3]

(a) **Input Data:**

- **OTR Number:** Users are required to enter their One Time Registration number for verification.
- **OTR Document/Image:** Users must upload an image or PDF of the OTR document. This is used for OCR-based extraction and comparison with the entered OTR number.
- **Other Documents (Optional Uploads):** Users may upload additional supporting documents such as identity proofs or certificates. These documents are stored but not subjected to any automated verification.

Output Data:

(a) **Verification Status:** The system generates a result indicating whether the entered OTR number matches the one extracted from the uploaded document/image. Possible statuses include “Verified” or “Mismatch.”

(b) **Notifications:** Users are notified of the verification result via the system interface or email/SMS if integrated.

(c) **Activity Logs:** All user actions, including uploads and verification outcomes, are logged with timestamps for auditing purposes.

Data Flow:

1. **Submission:** The user submits the OTR number and uploads the OTR document image.
2. **OCR Extraction:** The system uses OCR to extract the OTR number from the image.
3. **Validation:** The extracted number is compared to the manually entered number.
4. **Response:** Based on the comparison, the system returns a verification result.
5. **Storage:** All submitted data, including optional documents, are securely stored.

2.2.2. Functional Requirements

The system supports the following core functionalities necessary for OTR number verification, user interaction, and administrative oversight:[2]

1. **OTR Verification:**
   (a) Accept user input of the OTR number and upload of the OTR document.
   (b) Perform OCR on the uploaded document to extract the OTR number.
   (c) Compare the extracted OTR number with the manually entered number.
   (d) Return a verification result based on this comparison.
2. **User Registration and Login:**
   (a) Allow users to create accounts by submitting basic personal information.
   (b) Use OTP or email-based verification for account activation.
   (c) Enable users to log in and access their document submission and verification history.

3. **Document Upload (Unverified):**

    (a) Allow users to upload additional documents (e.g., ID proofs, certificates).

    (b) Store uploaded files securely for future use or reference.

    (c) Clearly indicate that these documents are not verified by the system.

4. **Faculty Dashboard:**

    (a) View all user submissions, including OTR verification status.

    (b) Perform manual review or re-verification if needed.

    (c) Access logs and system usage statistics.

5. **Notification System:**

    (a) Send the message regarding successful OTR verification to Student.

2.2.3 Performance Requirements

The system must adhere to the following performance benchmarks to ensure it remains efficient, responsive, and dependable, especially under varying user loads.[1]

1. **Response Time:** The system should be able to process and verify OTR document/image within a maximum of 5 minutes under normal load conditions. This includes:

    (a) Upload the OTR document/image.

    (b) OCR processing time for data extraction.

    (c) Verification time for comparing extracted data with user-input OTR number.

    (d) Delivery of verification result to the user.

Efficient backend handling, such as asynchronous job queues and optimized OCR processing, should be implemented to meet this response time.

2. **Scalability:** The system must support growth in the number of users without degradation in performance. Specifically:

   (a) Support thousands of concurrent users submitting OTRs.

   (b) Enable dynamic scaling of cloud resources based on demand.

   (c) Use of load balancing and distributed processing for verification tasks.

   (d) Efficient handling of non-verified document uploads to avoid bottlenecks.

2.2 Dependability Requirements

The system should be dependable and ensure the trustworthiness of the OTR verification process.[3]

(a) **Availability:** The OTR verification service must be available 24/7, especially during deadlines.

(b) **Data Integrity:** All uploaded documents (OTR and others) must be securely stored and protected from unauthorized changes.

(c) **Security:** User data must be secured through encryption, access control, and regular security audits.

(d) **Maintainability:** The system must allow easy updates to the verification engine and support future document handling extensions without affecting core functionality.

2.2 Maintainability Requirements

To ensure the system remains robust and future-proof, the following maintainability principles must be adhered to:

1. **Ease of Updates:** New document formats or OCR models for OTR should be easily integrable without impacting the rest of the system.
2. **Modularity:** The system should separate OTR verification logic from optional document upload handling.
3. **Code Readability and Documentation:** Code should follow clean coding standards with sufficient documentation for both backend and frontend components.
4. **Scalability and Flexibility:** Future integrations like admin feedback on uploaded documents (non-verification based) should be easily supported.
5. **Error Logging and Monitoring:** The system must track verification errors, upload issues, and system health in real time.

2.2 Security Requirements

Given that personal and sensitive documents are uploaded, even if not all are verified, strong security measures are essential:

1. **Data Encryption:** All uploaded documents (verified or not) must be encrypted both in transit (SSL/TLS) and at rest.
2. **Authentication and Authorization:** Only authenticated users can upload/view their documents. Admin access is role-based and restricted.
3. **Audit Trails:** All user and admin actions, including uploads and verifications, must be logged with timestamp and IP address.

4. **Input Validation and Threat Prevention:** All file uploads and inputs must be sanitized to prevent XSS, CSRF, and injection attacks.

5. **Regular Security Updates:** The system should be regularly updated to fix vulnerabilities in dependencies and platforms.

2.2 Look and Feel Requirements

The user interface must be designed to promote usability and clarity:

1. **Intuitive and Clean:** Users should clearly see where to input OTR number, upload the document, and get the verification result.

2. **Responsive:** The UI must work seamlessly on desktops, tablets, and mobile phones.

3. **Accessible:** The system must follow accessibility guidelines, including screen reader support and high-contrast modes.

2.3 SDLC Model to Be Used

The system will be developed using the **Agile SDLC model**, ensuring flexibility and continuous feedback throughout development.

1. **Planning:** Define requirements specific to OTR verification and optional document upload.

2. **Design:** Architect a modular system that separates core verification logic from general uploads.

3. **Development:** Build features in short sprints, starting with OTR verification, followed by upload handling and user dashboard.

4. **Testing:** Each module will undergo unit, integration, and user testing.

5. **Deployment:** Roll out to staging and production in phases.

6. **Maintenance:** Monitor, debug, and upgrade the system post-deployment.

Chapter 3

System Design

3.1 Design Approach (Function Oriented or Object Oriented)

The system follows an **Object Oriented Design Approach**. Object-oriented design promotes modularity and reusability, which is suitable for a system that involves various modules such as user registration, document upload, verification, notification, and administrative actions. Each module is treated as an object with its own attributes and methods, promoting encapsulation and maintainability.[2]

3.2 Detail Design

TThe system design is presented using structured analysis and design tools that clearly illustrate the interaction, data flow, and workflow among various system components. These tools—including Data Flow Diagrams (DFDs), Entity-Relationship (ER) Diagrams, Use Case Diagrams, and Activity Diagrams—enable a visual and logical understanding of how the system functions in different scenarios. By breaking down the system into subsystems and components, the design highlights how user inputs, data processing, document verification, and administrative actions are coordinated to achieve seamless functionality. Entity-Relationship (ER) Diagrams serve as blueprints for the database design, visually representing the relationships between different entities, such as users, data, and processes. [1]

(a) Flow Chart / Block Diagram

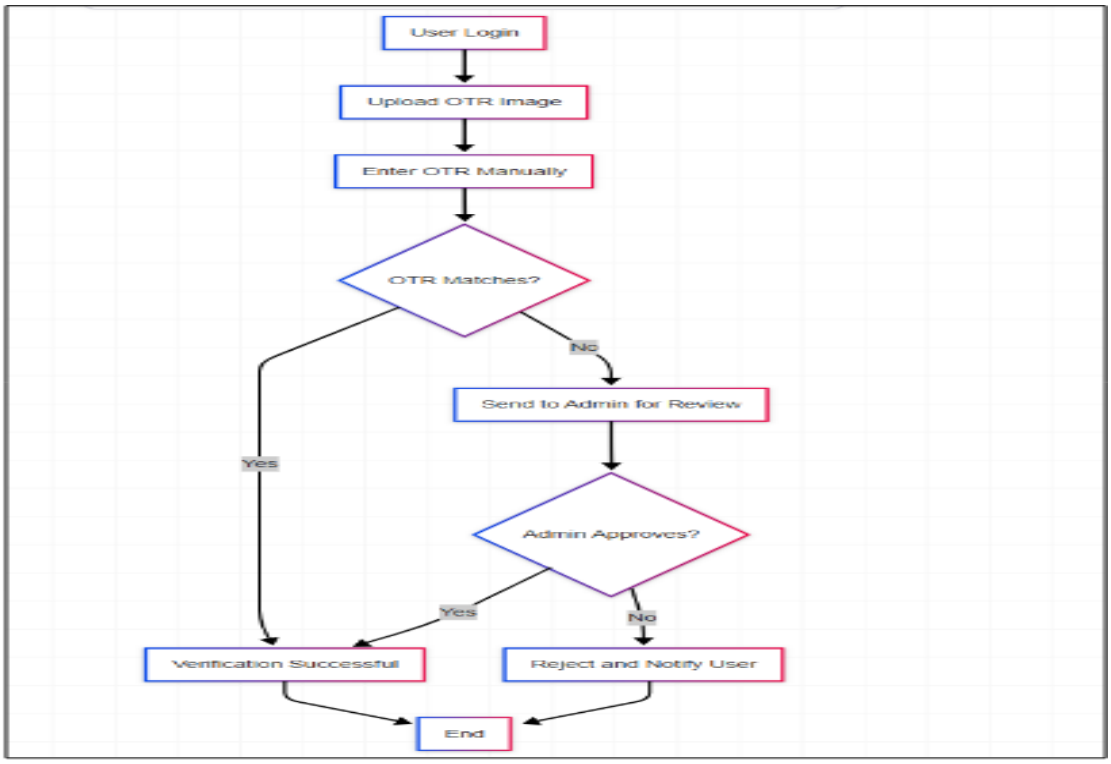

Figure 3.1: System Flowchart

(b) UML Diagrams

- **Use Case Diagram**

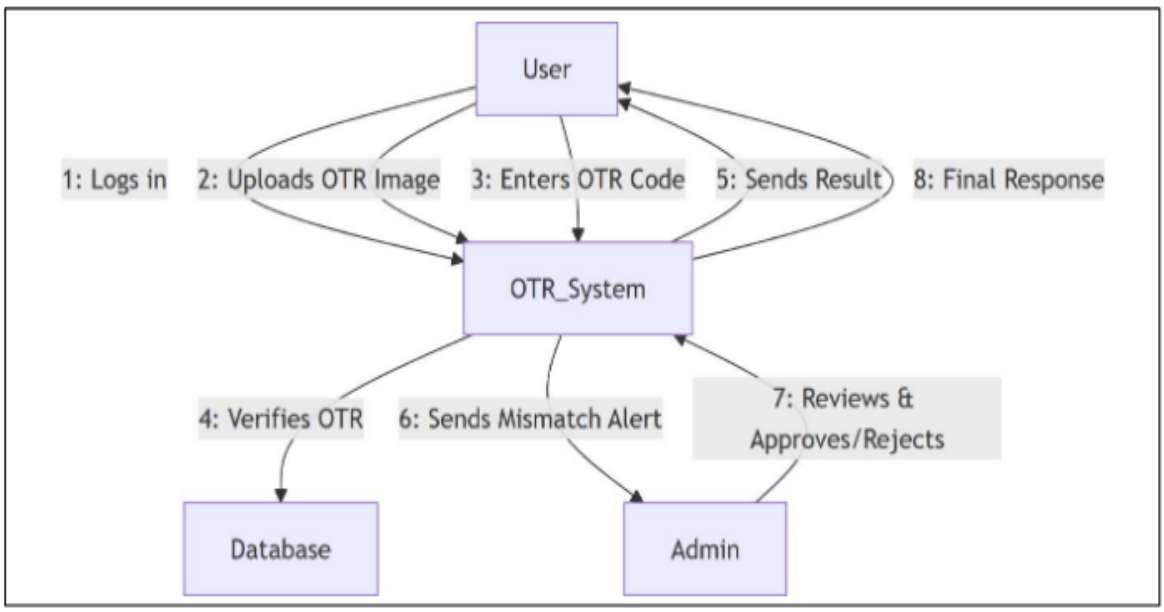

Figure 3.2: Use Case Diagram for User and Admin

- **Sequence Diagram**

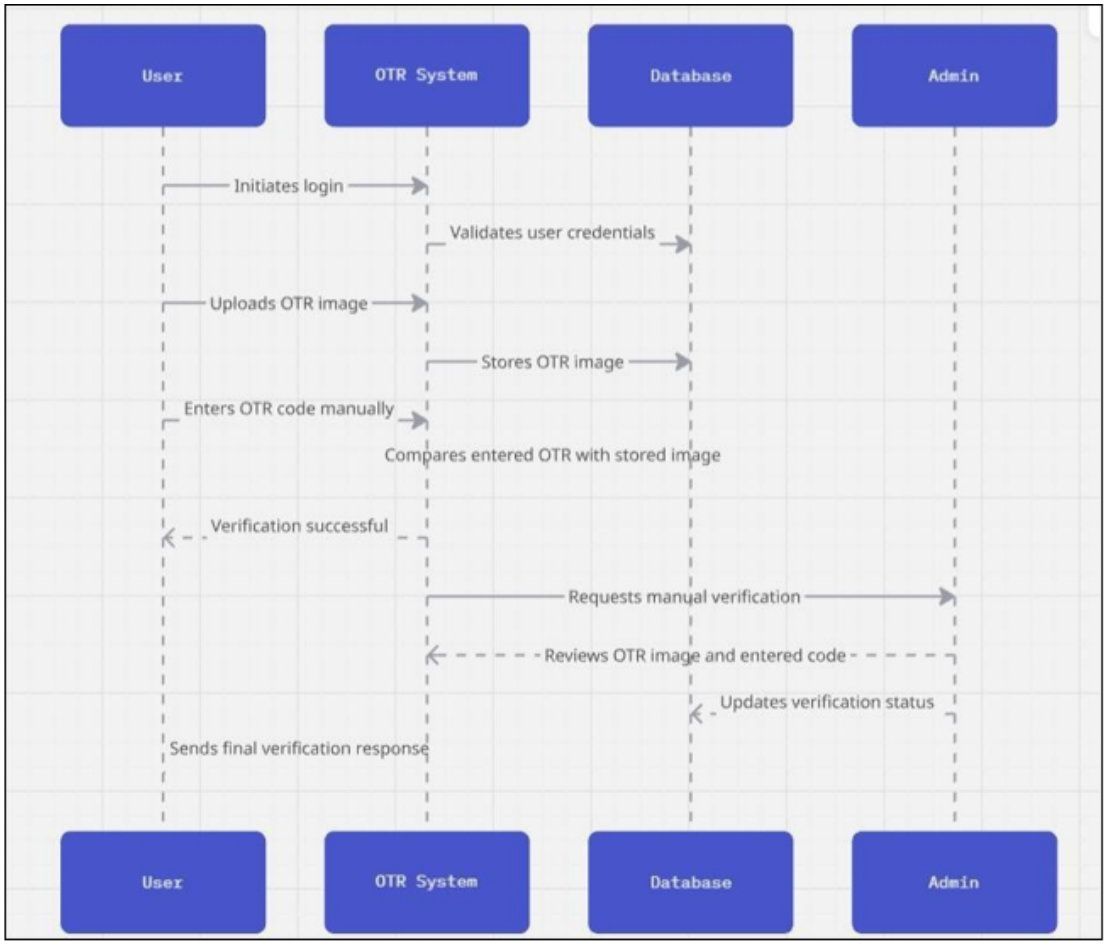

Figure 3.3: Sequence Diagram of Document Verification

- **Class Diagram**

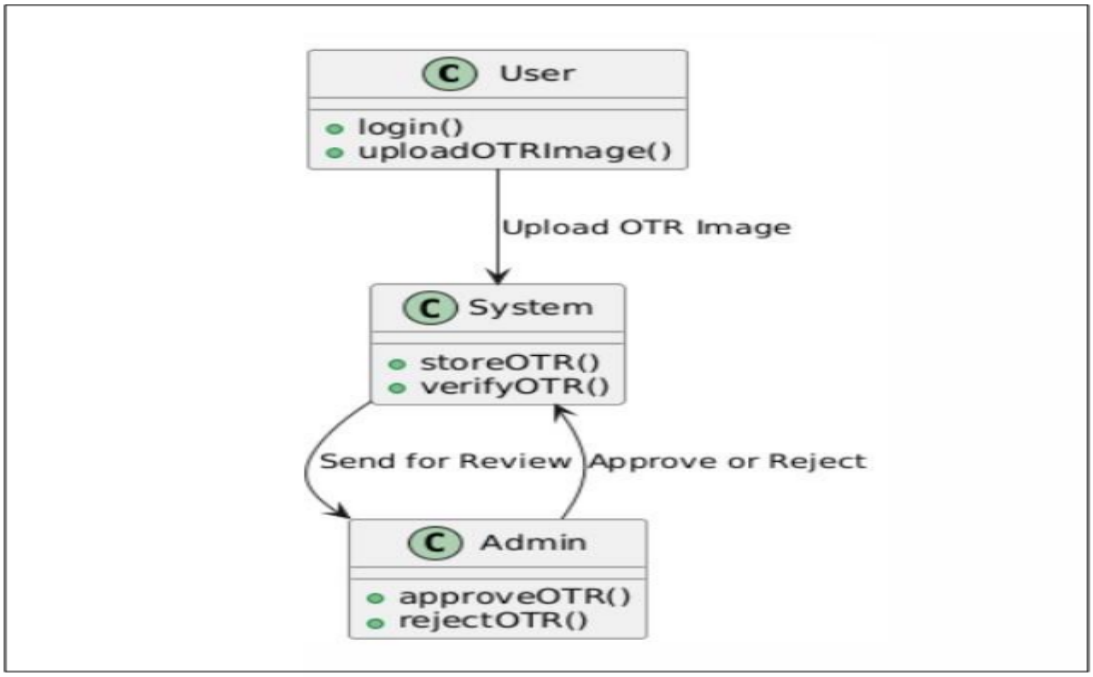

Figure 3.4: Class Diagram of the System

(c) DFD (Data Flow Diagrams)

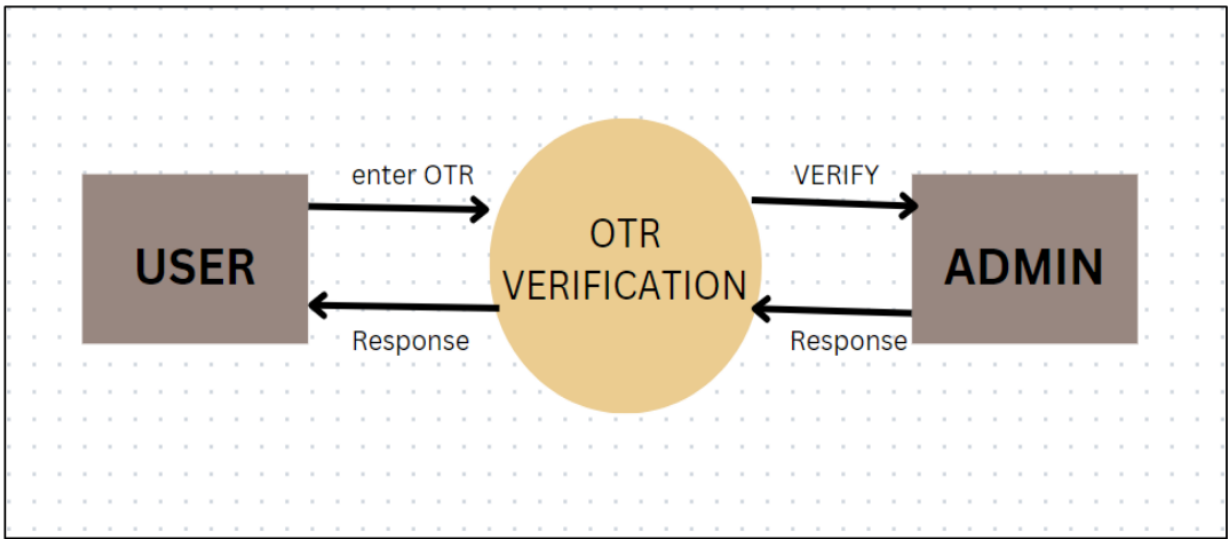

Figure 3.5: DFD Level 0 – Context Diagram

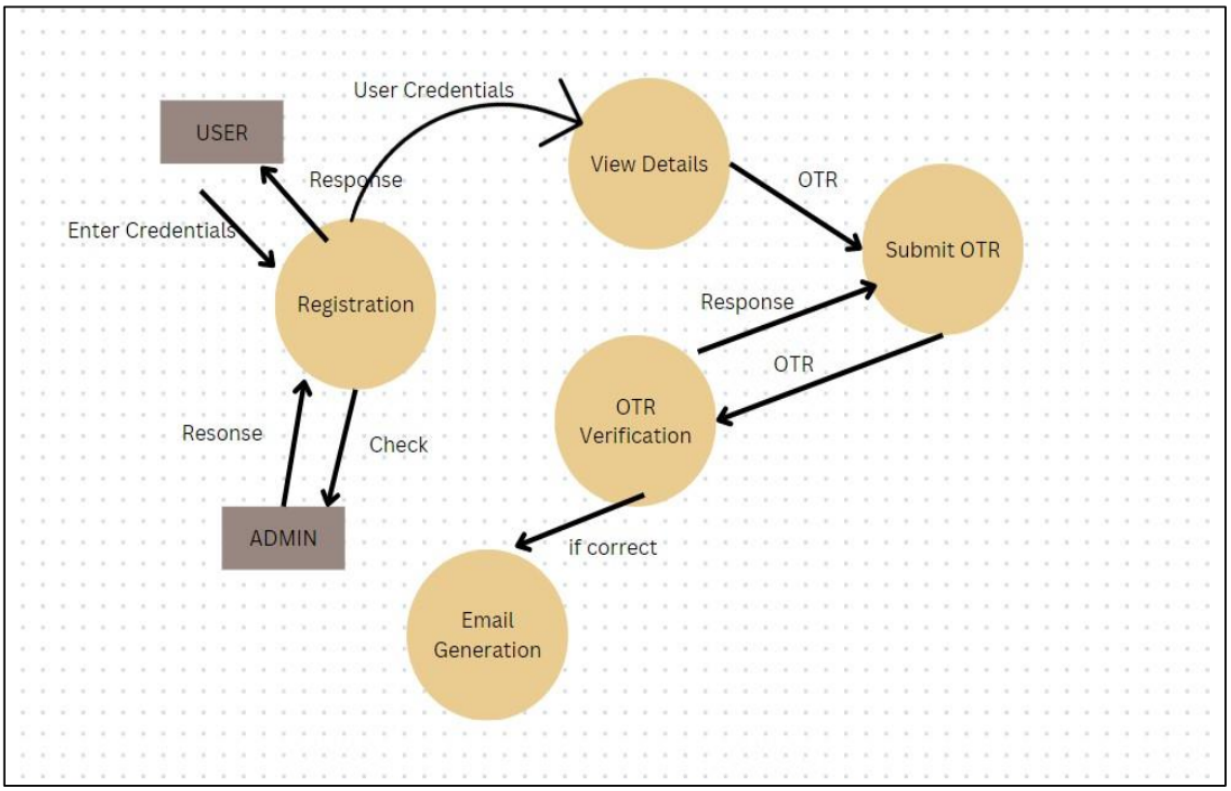

Figure 3.6: DFD Level 1 – Major Modules of the System

Database Design (E-R Diagram)

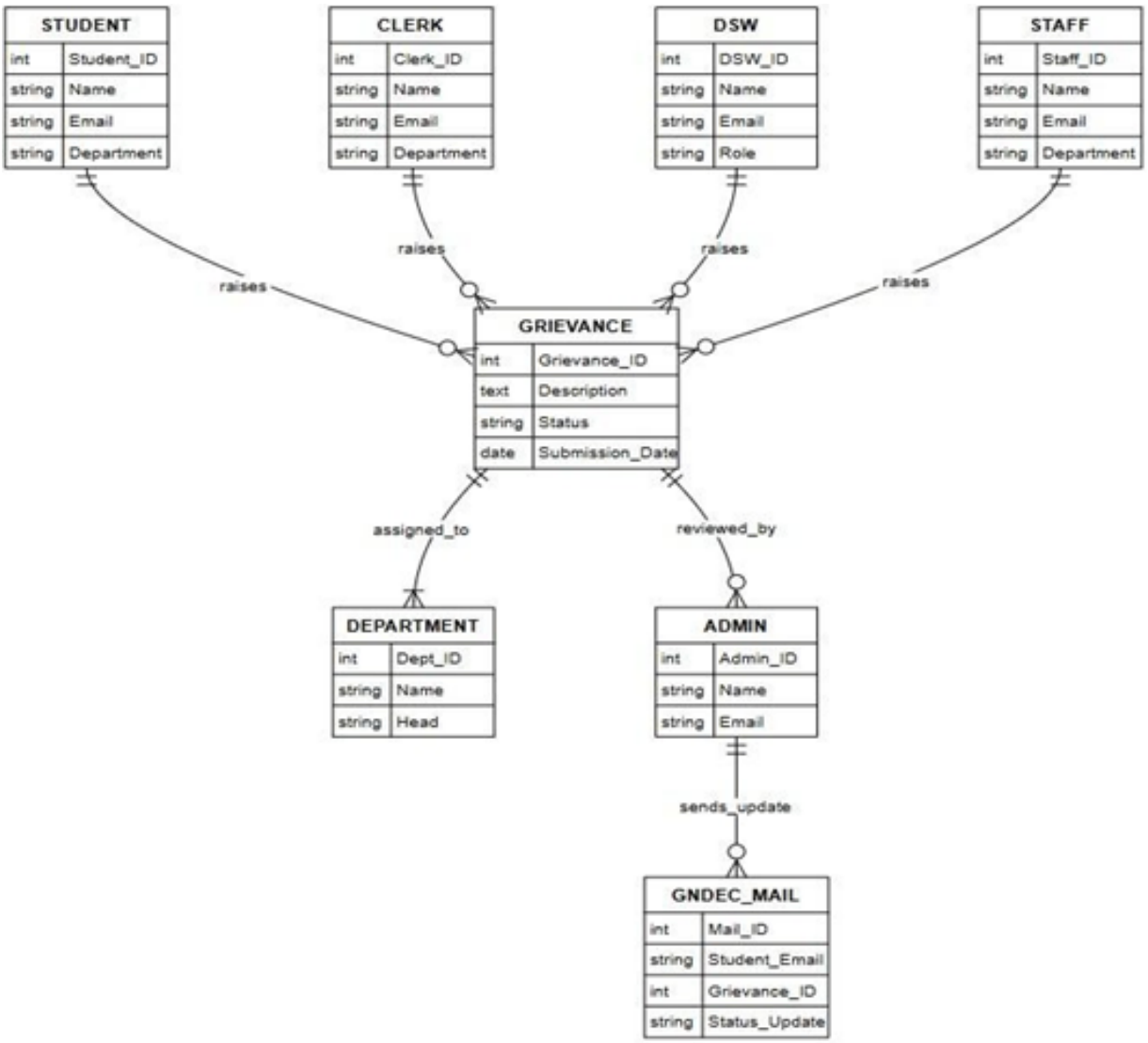

Figure 3.7: E-R Diagram for Database Design

3.3 User Interface Design

The user interface is designed with a clean and responsive layout, ensuring a smooth experience across different devices. The following principles are applied:

1. **Responsiveness:** Adapts to different screen sizes (mobile, tablet, desktop).
2. **Clarity:** Clear forms for user registration/login and document upload.
3. **Feedback:** Real-time notifications and messages upon successful actions.
4. **Simplicity:** Easy navigation with an intuitive layout.

5. **Security Prompts:** Proper alerts and validations for secure login and document submission.

3.4 Methodology

1. **Input Submission**

   (a) The user enters the **OTR Number** manually in a form.

   (b) The user also uploads an image containing the OTR (e.g., a screenshot or scanned copy).

2. **OCR-Based Extraction**

   (a) Optical Character Recognition (OCR) is applied to the image to extract the **OTR number**.

   (b) OCR library `Tesseract` is used for text detection.

3. **Verification**

   (a) The system compares the **OCR-extracted OTR number** with the **user-entered OTR number**.

   (b) If both match:

   - The OTR is considered **verified successfully**.

   (c) If there is a mismatch:

   - The system returns a **verification failure message**.

4. **Audit Logging**

(a) The system logs each verification attempt including:

TimestampEntered OTR number Extracted OTR number Match result (Success or Failure)

(b) These logs are maintained for auditing and security purposes.

Chapter 4

Implementation and Testing

4.1 Introduction to Languages, IDEs, Tools and Technologies Used

The development of the Online OTR Verification System involved a modern full-stack JavaScript framework with a modular and scalable architecture. The following technologies were used in different layers of the system:

1. **React.js:** A popular JavaScript library for building dynamic user interfaces. It allows for the creation of reusable UI components and supports efficient rendering using a virtual DOM. React was used to develop the frontend of the system, providing a responsive and interactive user experience.[2]

2. **Tailwind CSS:** A utility-first CSS framework used to design sleek and responsive interfaces without writing custom CSS from scratch. It enables rapid prototyping and enforces a consistent design system throughout the application.[2]

3. **Node.js:** An open-source, cross-platform JavaScript runtime environment used to build the server-side of the application. Node.js allows for non-blocking, event-driven architecture, making it efficient for handling real-time data and I/O-intensive operations.[1]

4. **Express.js:** A lightweight and flexible web application framework built on top of Node.js. Express was used to handle HTTP requests, define API routes, and manage middleware functionality in the backend.[1]

5. **MongoDB:** A NoSQL database that stores data in flexible, JSON-like documents. MongoDB is schema-less, making it suitable for dynamic applications like this one, where the structure of stored data can evolve over time.[1]

6. **Visual Studio Code (VS Code):** A powerful, open-source code editor that provides support for JavaScript/TypeScript, debugging tools, version control integration, and extensions to speed up development.

7. **Git and GitHub:** Git was used for version control to track code changes and collaborate efficiently. GitHub was used as the remote repository for source code hosting and management.

8. **npm (Node Package Manager):** Used to install, manage, and configure various libraries and dependencies required for both frontend and backend development.[1]

This technology stack was chosen for its high performance, scalability, and widespread community support, ensuring a robust and maintainable application.[2]

4.2 Algorithm/Pseudocode Used

Pseudocode for OTR Document Verification

```
1. Start
2. Accept OTR document/image from user upload
3. Apply OCR library (Tesseract) to extract text from document
4. Parse extracted data for required fields (e.g., OTR number)
5. Match extracted data with user registration data in MongoDB
6. If match found:
     a. Mark document as verified
   Else:
     b. Mark document as not verified
7. Update verification status in the database
8. Notify user about the verification status through E-mail
9. End
```

4.3 Testing Techniques

To ensure the robustness, reliability, and performance of the Online OTR Verification System, a comprehensive set of software testing techniques was employed. The goal was to detect bugs early, verify functionality, and validate that the system meets both technical specifications and user expectations.[2]

1. **Unit Testing:** Each component or function, such as the user login module, OTR document upload function, and OCR verification logic, was tested in isolation. Unit testing helped ensure that individual pieces of code function correctly before integration with other modules.[3]

2. **Integration Testing:** After unit testing, integration testing was conducted to ensure that modules interact with each other correctly. For example, testing whether the OTR upload module passes correct data to the OCR service, and whether the backend API processes and stores the results in the database.[2]

3. **System Testing:** This type of testing involved validating the complete and integrated application to check whether it fulfills all the specified requirements. The system was tested in an environment similar to the production setup to detect issues related to overall functionality, usability, and stability.[1]

4. **Black Box Testing:** In this approach, the internal structure of the application was not considered. Instead, inputs and expected outputs were tested. For example, inputs such as a valid user email and password were tested for expected redirection, while invalid inputs were checked for proper error messages.[3]

5. **User Acceptance Testing (UAT):** This was the final stage of testing, performed from an end-user perspective. Real users tested the system to validate if it met their requirements

and expectations. Feedback gathered during UAT helped make final refinements to the interface and workflow.[1]

6. **Regression Testing:** After updates or bug fixes were applied, regression testing was performed to ensure that new changes did not affect the existing functionalities.[2]

7. **Performance Testing:** The system was tested under different loads, especially for simultaneous OTR document uploads, to evaluate response time, throughput, and stability.[1]

8. **Security Testing:** Special attention was given to input validation, authentication mechanisms, and database access to prevent vulnerabilities such as injection attacks, unauthorized access, and data leakage.[1]

Only OTR documents undergo automated verification using OCR and database matching. Other uploaded documents are stored securely for reference but are not verified by the system.[2]

4.4 Test Cases Designed for the Project

Table 4.1: Sample Test Cases for Online OTR Verification System

Test Case	Input	Expected Output	Result
Login - Valid Credentials	Valid email, correct password	Redirect to dashboard	Pass
Login - Invalid Password	Valid email, wrong password	Show error message	Pass
Upload OTR Document	Scanned OTR document image (clear)	Verified status stored in MongoDB	Pass
Upload Non-OTR Document	PDF/PNG/JPG (e.g., marksheet, ID)	Stored but not verified	Pass
OTR Verification - Valid OTR	Clear scanned OTR document	Marked as Verified	Pass
OTR Verification - Unclear OTR	Blurry or unreadable image	Flagged for manual review or not verified	Pass
Invalid Document Format	Unsupported file type (e.g., TXT)	Show error message	Pass

Chapter 5

Results and Discussions

5.1 User Interface Representation

The Online OTR Verification System provides an intuitive and responsive user interface built using React and styled with Tailwind CSS. The design focuses on clarity, usability, and accessibility, ensuring a seamless experience for both users and administrators.[3]

5.1 Brief Description of Various Modules of the System

1. **User Login :** Allows new users to login by submitting personal details and uploading required documents.

2. **Login Module:** Authenticates users and redirects them to their respective dashboards based on roles (user/admin).

3. **Document Upload and Verification Module:** Enables users to upload scanned copies of documents which are then automatically verified using OCR.

4. **Admin Dashboard:** Provides administrators with tools to review flagged documents, approve or reject user verifications, and monitor system activities.

5. **Notification System:** Notifies users about verification statuses and any required follow-up actions via email or alerts.

5.2 Snapshots of System with Brief Detail of Each and Discussion

Figure 5.1: User Interface

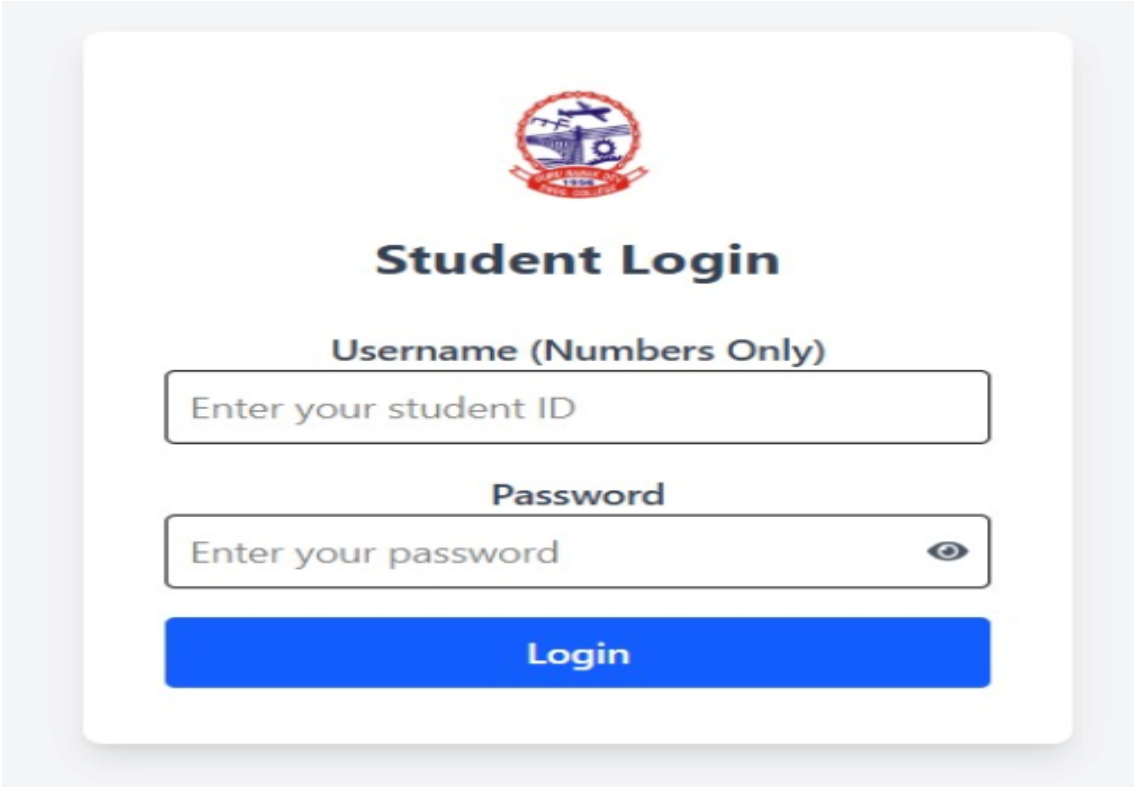

Figure 5.2: Login Page – Allows students and faculties to access their respective dashboards.

Figure 5.3: User Dashboard – Displays upload status, document verification updates, and profile management.

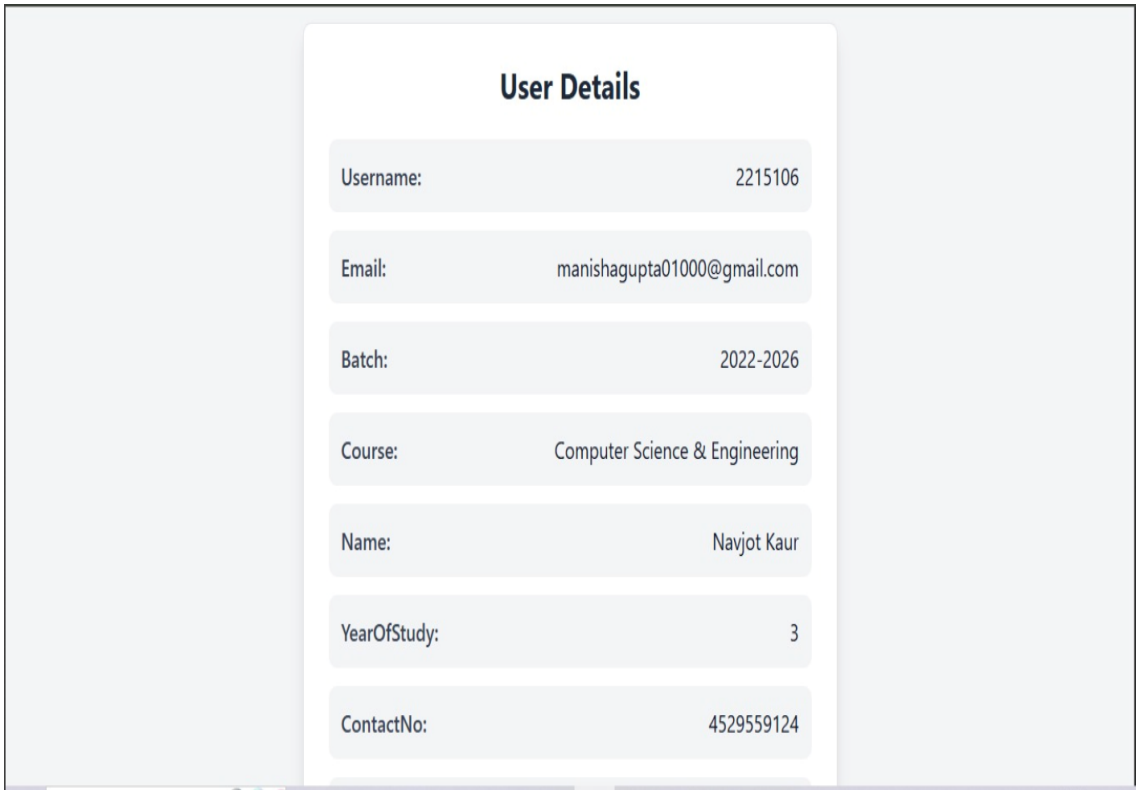

Figure 5.4: Student Details – Shows student details

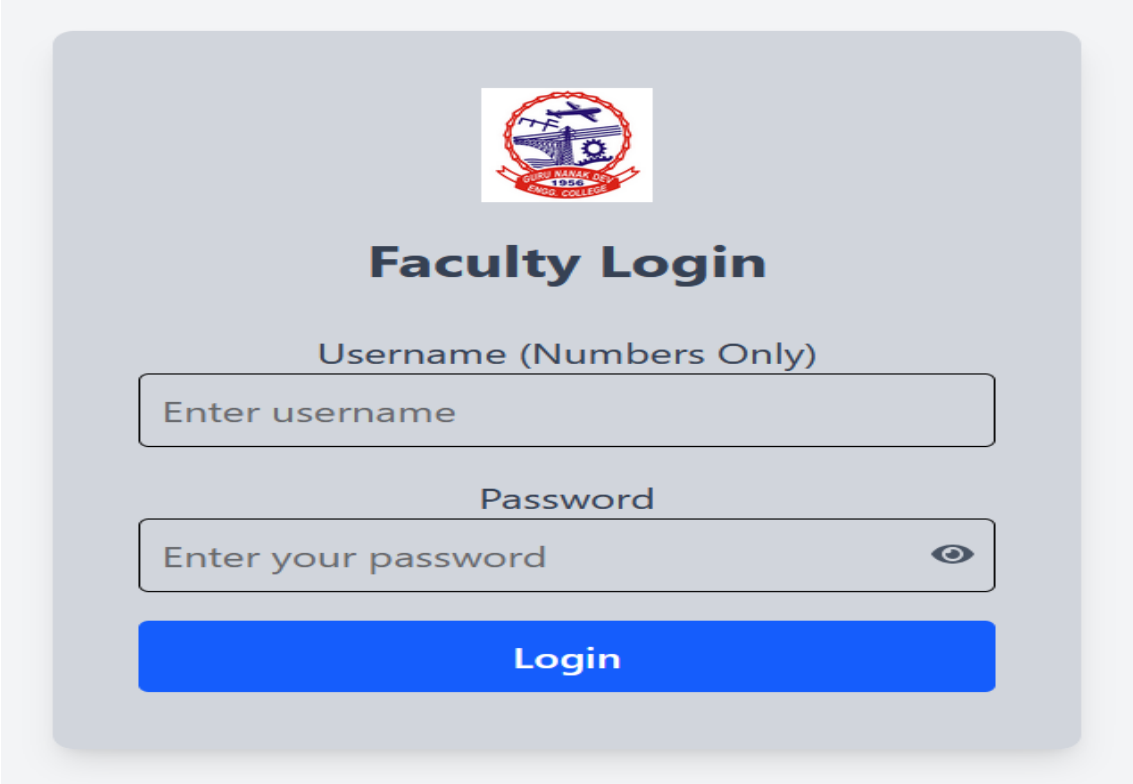

Figure 5.5: Faculty Login: Here faculty will login

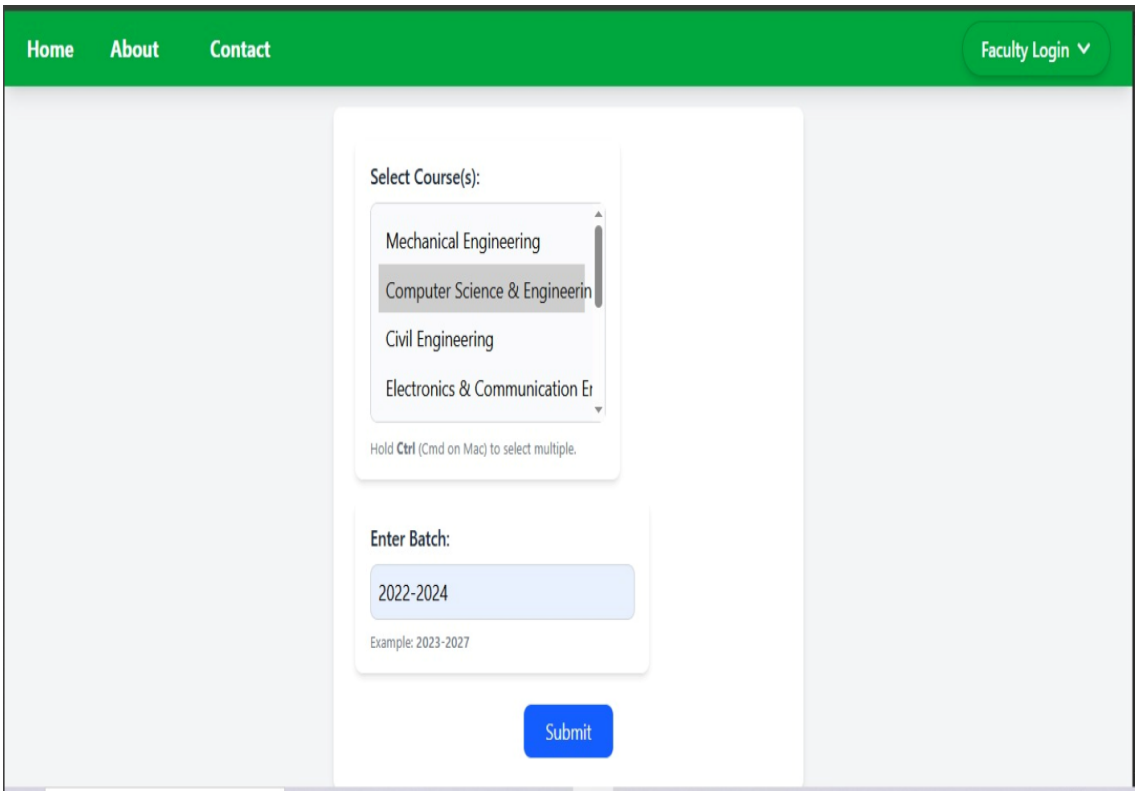

Figure 5.6: Fetching of Students – here the faculty can see the students details

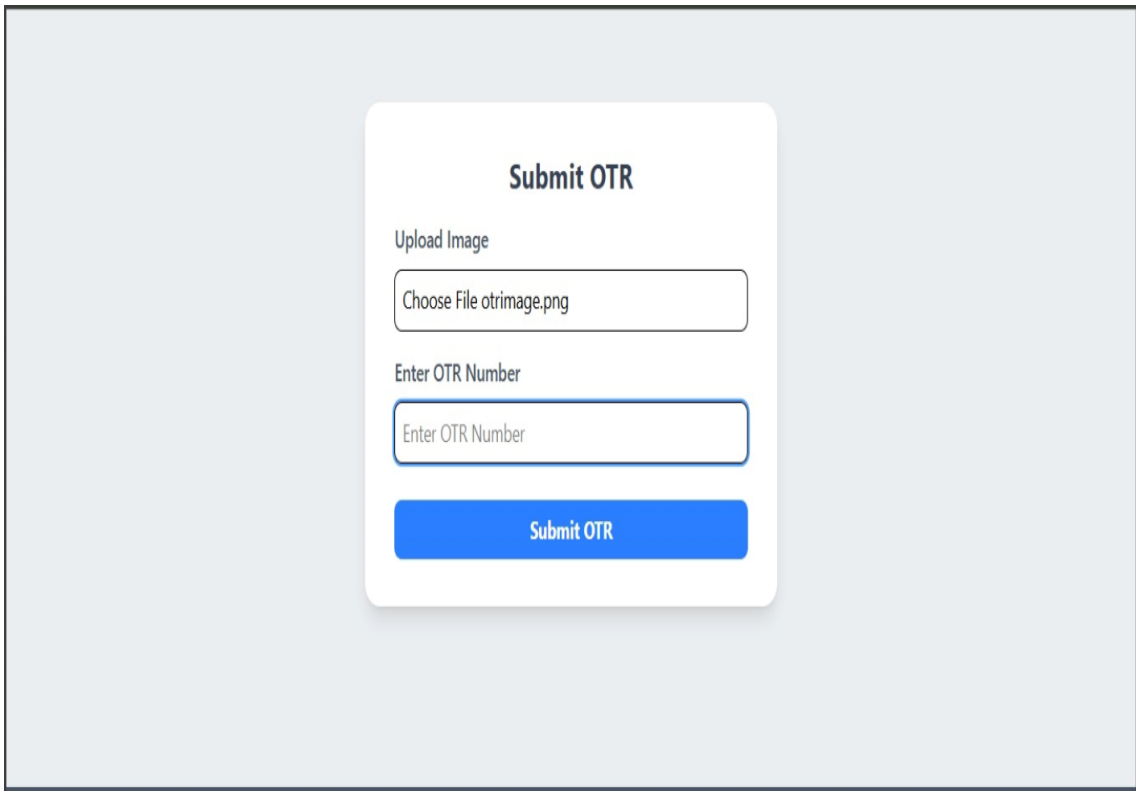

Figure 5.7: Document Upload – Users can upload PDF, JPG, or PNG documents for verification.

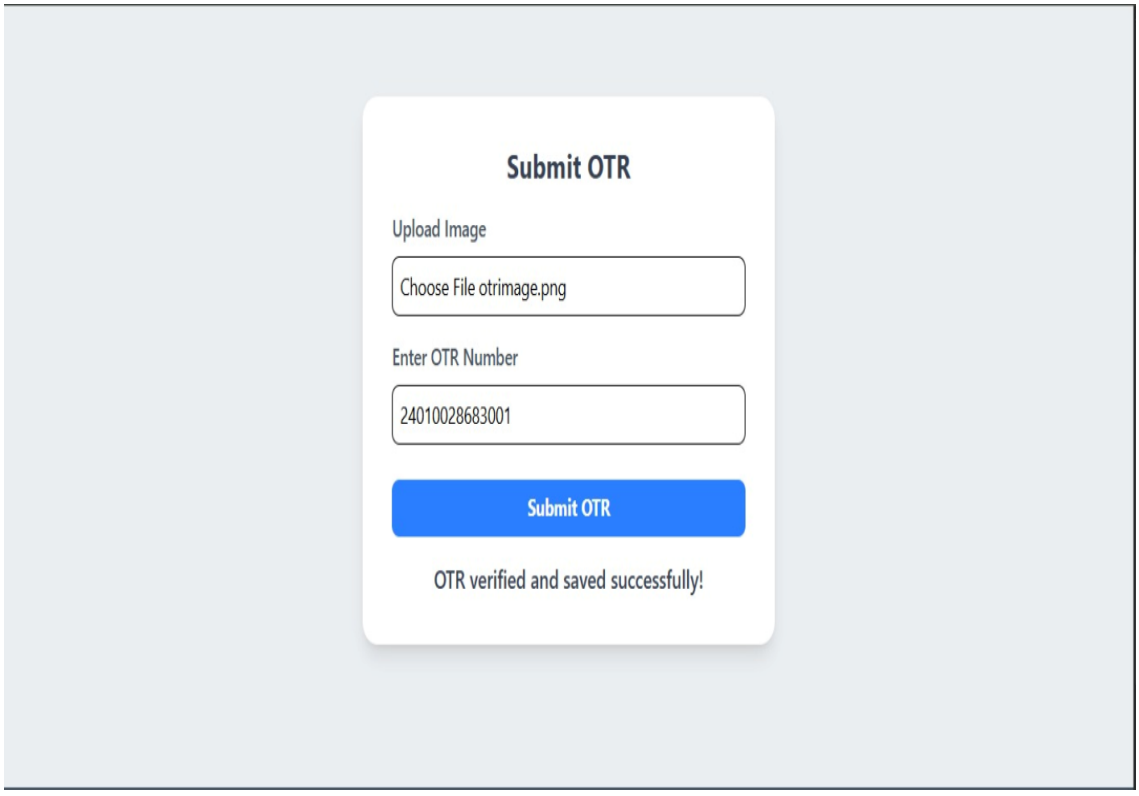

Figure 5.8: OTR Verification: OTR Verfied and saved successfully

Figure 5.9: Send Email: Sent email after OTR was verified to the student

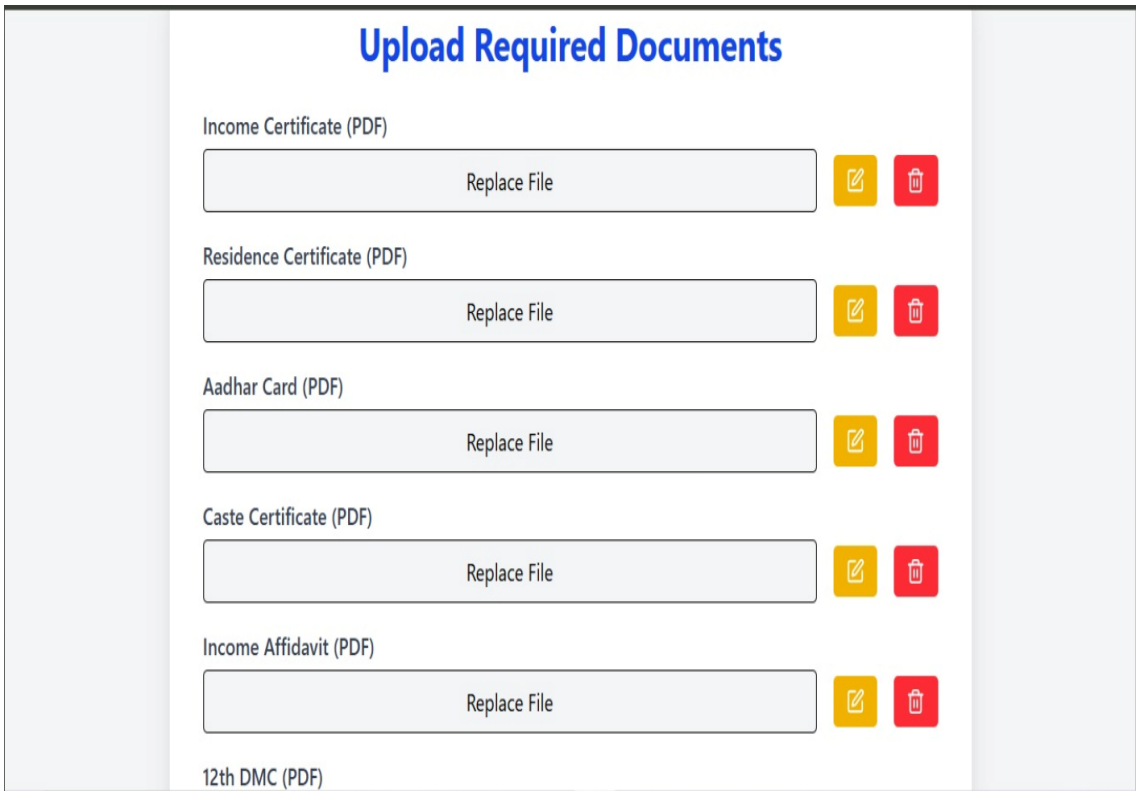

Figure 5.10: Upload Document: uploads other scholarships regarding documents

5.3 Back-End Representation (If Database Has Been Used)

The backend of the system is implemented using Node.js and Express.js, with MongoDB as the primary database. The database stores user profiles, document verification status, and logs of all system activities.[2]

5.3 Snapshots of Database Tables with Brief Description

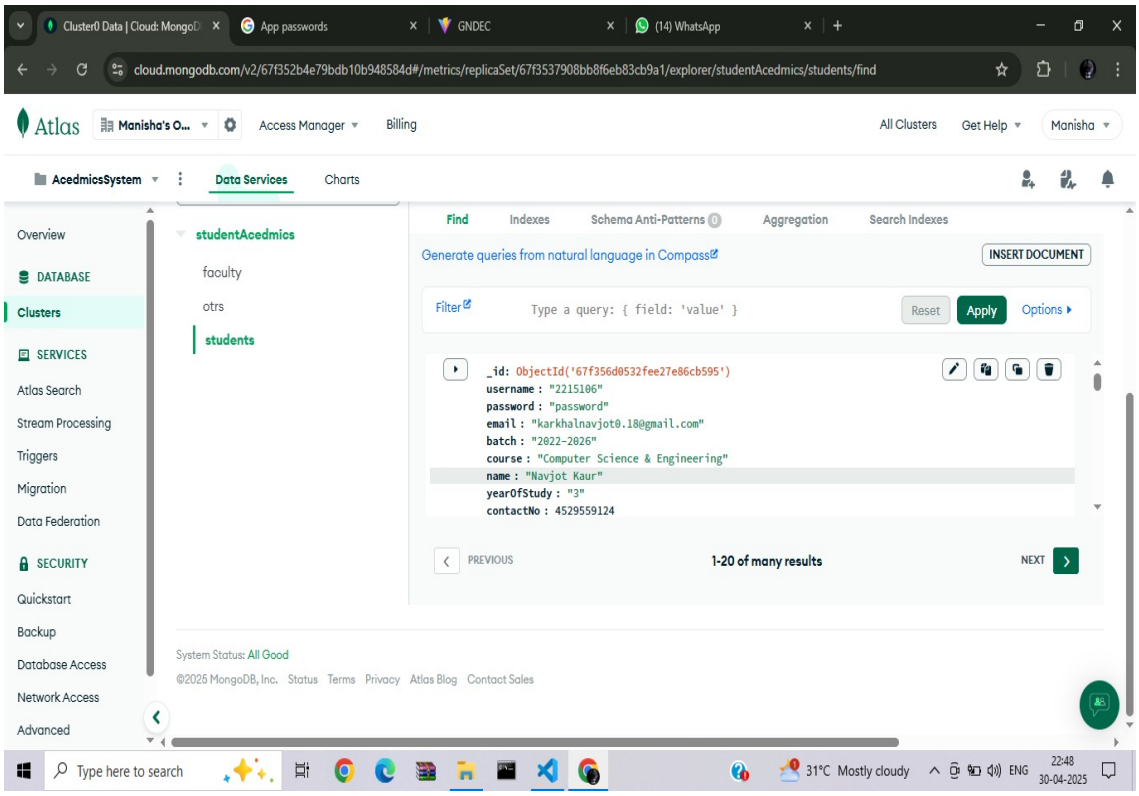

Figure 5.11: Users Collection – Stores information such as username, email, role, and hashed passwords.

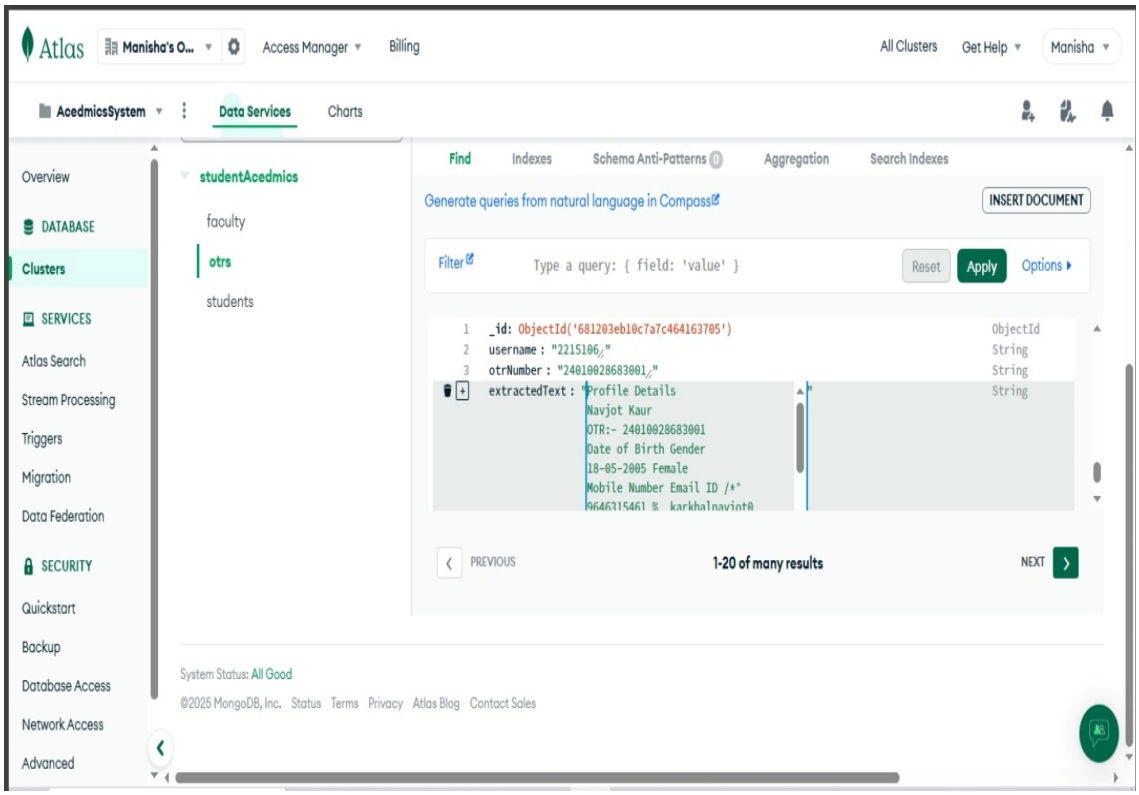

Figure 5.12: Documents Collection – Maintains uploaded document details, extracted OCR data, and verification status.

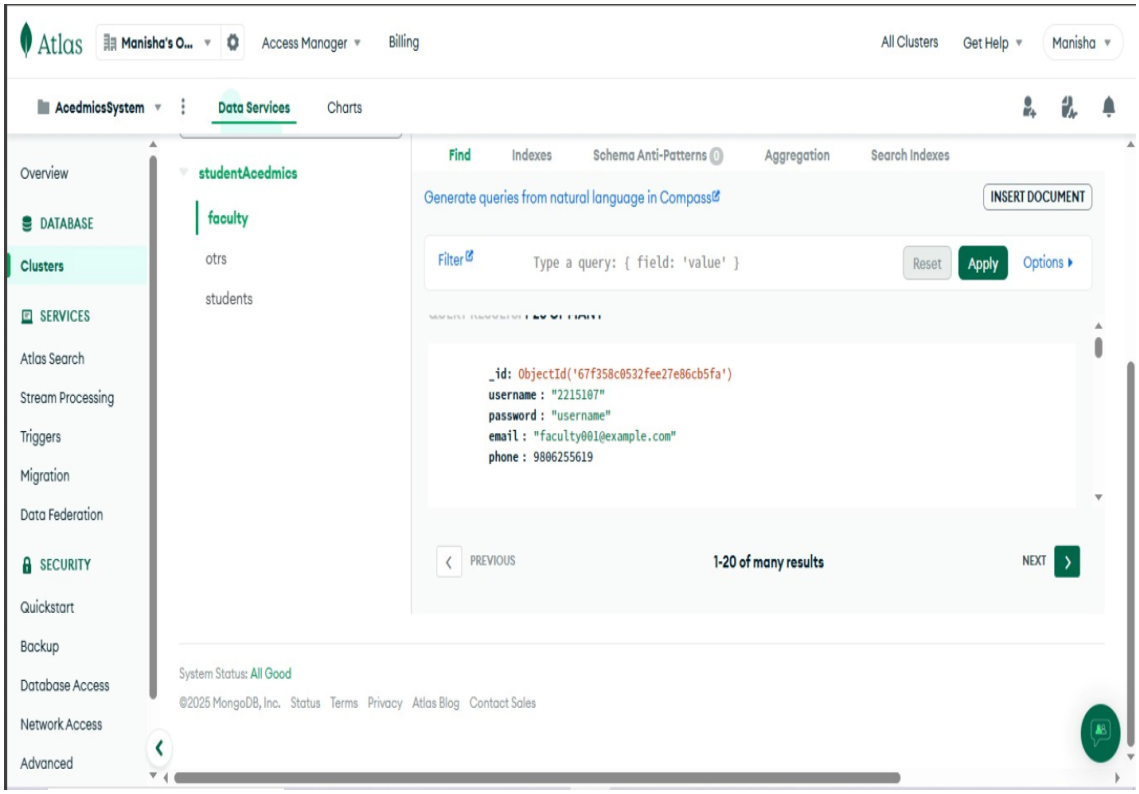

Figure 5.13: Faculty Collection – Description of the Faculty Collection, such as the data it contains, purpose, and any other relevant information.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

The Online OTR Verification System was successfully developed and implemented using modern web technologies such as React, Tailwind CSS, Node.js, Express, and MongoDB. The system automates the traditionally manual process of verifying documents, making it faster, more secure, and less prone to human error.

Through modules such as user registration, document upload, automated OCR-based verification, and administrative oversight, the platform achieves its primary objectives of improving efficiency and reducing verification delays. Furthermore, the system ensures data integrity, confidentiality, and role-based access, making it robust and scalable.

The testing phase confirmed the system's reliability and usability, with successful outcomes across various test cases. Feedback from users during the UAT (User Acceptance Testing) phase also reflected satisfaction with the system's performance and interface.

6.2 Future Scope

The One Time Registration (OTR) Verification System, which extracts and verifies OTR numbers from uploaded images, has significant potential for future development and improvement. The following points outline its future scope:

- **Update Announcements:** Admin can post the latest notices regarding scholarships.
- **Student Details Filtering:** Faculty can view student details based on various filters such as document submission status, batch, year of study, branch, or view all student data in a single frame.

- **Student Query Panel:** If students face any issues not covered in the FAQs, they can directly send an email to the Scholarship Management Authority.

These enhancements will significantly expand the system's usability, accessibility, and trustworthiness in real-world deployment scenarios.

References

1. Abhishek Chamoli. *Convert Images to Text in Node.js with Tesseract.js: A Step-by-Step Guide*. Medium. Available at: https://medium.com/@abhishekchamoli007/convert-images-to-text-in-node-js-with-tesseract-js-a-step-by-step-guide

2. Smashing Magazine. *Image to text conversion with React and Tesseract.js*. Available at: https://www.smashingmagazine.com/2021/06/image-text-conversion-react-tesseract-js-ocr/

3. International Journal of Research Publication and Reviews. *Volume 4, Issue 4*. Available at: https://ijrpr.com/uploads/V4ISSUE4/IJRPR11943.pdf