

System Analysis

1. Project Overview

This project is an AI-powered assessment platform that automates both formative and summative assessment, provides personalized feedback, and generates performance analytics.

It helps teachers, students, and educational administrators save time, reduce grading errors, and gain actionable insights into learning outcomes.

--

2. Problem Statement

The problem is that exam grading is currently manual and time-consuming, which leads to inconsistent evaluation, delayed or missing feedback, and limited insights into student performance. Universities often rely on MCQs for faster grading, even when they are insufficient for proper assessment, especially in credit-based systems where each term affects future course registration.

--

3. Stakeholders (Actors)

Actor	Description
Student	Takes exams, receives grades, and gets feedback
Instructor / Teaching Staff	Uploads exams and configures grading and evaluation settings
University / College	Receives aggregated performance reports and insights for academic decision-making
Administrator	Manages users, exams, permissions, and system configuration
ExamAI System	AI-powered backend that performs grading, feedback generation, and analytics
External OCR Service	Extracts text from scanned exam documents

--

4. User Stories

Student

- As a student, I want to take exams online or submit scanned exams, so that my answers can be evaluated automatically.
- As a student, I want to receive grades and feedback immediately, so that I can understand my mistakes and learn without waiting for weeks.
- As a student, I want the exam to adapt to my level, so that it accurately measures my understanding and true performance.
- As a student, I want to practice on the platform outside of exams, so that I can improve my skills continuously.
- As a student, I want the exam to allow diverse formats, including writing on paper and uploading it, so that I can express my solutions and reasoning better, not just multiple-choice answers.

Instructor / Teaching Staff

- As an instructor, I want to upload exams and answer keys, so that students can be evaluated automatically.
- As an instructor, I want to configure grading and evaluation settings, so that the assessment matches my teaching objectives.
- As an instructor, I want the exam to adapt to each student's level, so that I can measure their true understanding and provide meaningful feedback.
- As an instructor, I want a detailed profile for each student, so that I can track their progress and strengths accurately.
- As an instructor, I want to correct exams quickly, so that I can save time on routine grading.
- As an instructor, I want to create practice exams that adapt to student levels, so that students can train and reach higher learning outcomes.

Administrator

- As an administrator, I want to manage users and permissions, so that the system is secure and well-organized.
- As an administrator, I want the system to operate correctly under all conditions, so that educational processes are smooth and reliable.
- As an administrator, I want to ensure no user has permissions beyond their role, so that data integrity and security are maintained.

University / College

- As a university/college, I want to monitor student progress and learning outcomes, so that we are aware of academic performance across programs.
- As a university/college, I want to receive aggregated exam results for each term quickly, so that we can proceed with credit-based registrations without delay.
- As a university/college, I want the system to adapt exams to student levels, so that performance metrics reflect actual learning rather than only standardized scores.

--

5. Acceptance Criteria

Acceptance criteria define **when a user story is considered successful**.

Each case follows the **Given – When – Then** format for clarity and testability.

Student

- **Case 1**
 Given the student submits an exam or practice attempt
 When the system receives the submission
 Then the answers are graded automatically and feedback is returned immediately.
- **Case 2**
 Given the student is taking an adaptive exam
 When the system evaluates the student's answers
 Then the exam difficulty adjusts dynamically to match the student's level.
- **Case 3**
 Given the student uploads answers in a paper-based or free-form format (e.g., handwritten sheets)
 When the system processes the uploaded content
 Then the answers are recognized, evaluated, and detailed feedback is provided.

- **Case 4**

Given the student uses the platform for training and practice

When they attempt practice exams

Then their progress is tracked and learning gaps are clearly identified.

Instructor / Teaching Staff

- **Case 1**

Given the instructor uploads an exam along with its official answer key

When the exam is published

Then it becomes available for students and ready for automated grading.

- **Case 2**

Given the instructor uploads student answers manually (e.g., scanned sheets or externally collected responses)

When the system processes these submissions

Then the answers are graded automatically using the provided answer key.

- **Case 3**

Given the instructor defines grading rules, rubrics, or feedback depth

When student answers are evaluated

Then grading and feedback strictly follow the instructor's configuration.

- **Case 4**

Given the instructor reviews student performance

When accessing analytics and reports

Then a complete performance profile for each student is available.

- **Case 5**

Given the instructor creates adaptive practice exams

When students attempt them

Then the exam difficulty adapts per student and learning progress is recorded.

Administrator

- **Case 1**

Given different system users and roles exist

When the administrator manages permissions and access levels

Then each user only has the permissions required for their role.

- **Case 2**

Given the system operates under normal or peak usage

When exams, grading, and analytics are running

Then the platform remains secure, stable, and reliable.

University / College

- **Case 1**

Given exams for a term are completed

When the university or college requests exam results

Then results are generated accurately and quickly to support the credit system workflow.

- **Case 2**

Given student performance data is collected across courses

When institutional analytics are applied

Then the university gains clear insights into student progress and learning outcomes.

6. Functional Requirements

User & Access Management

- The system shall authenticate users and assign roles (student, instructor, administrator, institution).
- The system shall enforce role-based access control to ensure users only access authorized functionalities.
- The system shall allow administrators to manage users, roles, and permissions.

Exam Creation & Management

- The system shall allow instructors to create and upload exams in digital formats.
- The system shall allow instructors to define question types, grading rubrics, evaluation rules, and feedback depth.
- The system shall allow instructors to publish exams and control exam availability windows.
- The system shall support adaptive exams with dynamically adjusted difficulty per student.

Exam Submission

- The system shall allow students to take exams online.
- The system shall allow students to upload scanned or paper-based exam answers.
- The system shall validate exam submissions for completeness and correctness.

OCR & Exam Ingestion

- The system shall ingest scanned exam documents and images.

- The system shall apply OCR to extract questions, answers, and metadata.
- The system shall preprocess extracted text through cleaning and formatting correction.
- The system shall divide written exam papers into individual questions, sub-questions, and corresponding answer regions.
- **The system shall detect and correct language and spelling mistakes in student answers, with configurable correction levels set by instructors.**
- The system shall structure extracted data into questions, sub-questions, and answers.

Automated Grading

- The system shall automatically grade objective questions using predefined answer keys.
- The system shall evaluate open-ended, essay, and reasoning-based answers using AI models.
- The system shall apply rubric-based grading aligned with instructor-defined criteria.
- The system shall dynamically adjust grading depth based on question complexity and student performance history.

Feedback Generation

- The system shall generate personalized feedback for each student submission.
- The system shall highlight strengths, weaknesses, and improvement suggestions per question.
- The system shall adapt feedback style and depth based on student performance history.

Student Profiling & RAG

- The system shall store historical student performance data securely.

- The system shall retrieve relevant past performance data using RAG-based mechanisms.
- The system shall use retrieved data to personalize grading and feedback decisions.

Analytics & Reporting

- The system shall track student progress, learning gaps, and performance trends over time.
- The system shall generate detailed performance profiles for individual students.
- The system shall generate aggregated analytics for instructors and institutions.
- The system shall generate term-level and institutional performance reports.

Dashboards & Visualization

- The system shall provide role-based dashboards for students, instructors, administrators, and institutions.
- The system shall visualize grades, feedback summaries, and performance trends.

System Reliability & Scalability

- The system shall support concurrent grading of large exam volumes.
- The system shall maintain stability and reliability under peak usage conditions.
- The system shall log system operations, grading decisions, and errors for monitoring and auditing.

Deployment & MLOps

- The system shall deploy grading, feedback, and analytics models as scalable APIs.
- The system shall track model versions, experiments, and grading performance metrics.
- The system shall support model updates and rollback without service disruption.

7. Non-Functional Requirements

Performance

- The system shall provide near real-time responses to exam submissions and grading requests.
- The system shall support concurrent users without noticeable performance degradation.
- Automated grading shall complete within an acceptable processing time.
- The system shall efficiently utilize computational resources.

Reliability & Availability

- The system shall maintain stable operation during examination sessions.
- The system shall implement backup mechanisms to prevent data loss.
- The system shall log system operations and errors for monitoring.

Security

- The system shall protect user data using encryption.
- The system shall enforce secure authentication and authorization.
- The system shall prevent unauthorized access to sensitive data.

Usability

- The system shall provide an intuitive interface.
- The system shall display clear feedback and error messages.

Maintainability

- The system shall use a modular architecture.
- The system shall include clear documentation.

Compatibility

- The system shall operate across modern web browsers and devices.

Data Integrity

- The system shall ensure data consistency and accuracy.
- The system shall protect grading results from unauthorized modification

8. System Components

This section describes the main components of the ExamAI platform and how they collaborate to deliver automated grading, feedback, and analytics.

Client

The client represents all interfaces used to interact with the system.

It provides access to exams, submissions, results, feedback, and dashboards.

The client may include:

- Web browser interfaces for students, instructors, administrators, and institutions
- Mobile or tablet applications for exam access and result viewing
- Scripts or system-to-system clients used by institutions for bulk uploads or data integration

The client communicates exclusively with the backend through secure APIs.

Backend (API Layer)

The backend is the core orchestration layer of the system.

It exposes APIs that handle all business logic, coordination between agents, and data flow.

Responsibilities include:

- User authentication and role-based authorization
- Exam creation, publishing, and submission management

- Coordination of OCR, grading, feedback, and analytics agents
- Configuration of grading rules, rubrics, and feedback depth
- Serving dashboards, reports, and results to different user roles
- Logging, monitoring, and auditing system operations

The backend acts as the control plane for all AI-driven components.

Database

The database stores all structured system data required for operation and analysis.

Stored data includes:

- User accounts, roles, and permissions
- Exams, questions, rubrics, and configurations
- Student submissions and grading results
- Feedback outputs and analytics summaries
- System logs and operational metadata

The database ensures data integrity, consistency, and secure access across all components.

Machine Learning & AI Components

The system includes multiple AI-driven components deployed as services and agents.

These components include:

- OCR and document understanding models for scanned exams
- Automated grading models for objective, open-ended, and reasoning-based questions
- Agentic decision-making logic for dynamic grading depth and evaluation strategy
- RAG-based retrieval mechanisms for accessing historical student performance
- Feedback generation models for personalized and contextual responses
- Analytics models for performance trends and institutional insights

These models are versioned, monitored, and deployed through MLOps pipelines.

External Services

The system may integrate with external services to enhance functionality and scalability.

Examples include:

- OCR engines or cloud-based document processing services
- Email or notification services for result delivery and alerts
- Identity or authentication providers
- Cloud storage and compute services
- Monitoring and logging platforms

External services are integrated through well-defined and secure interfaces.

9. Data Description

This section describes the types of data used in the ExamAI platform, their sources, formats, and how they are utilized throughout the system.

Data Types

The system processes multiple data types to support automated grading, feedback generation, and analytics.

The main data types include:

- **Text data:** exam questions, student answers, rubrics, feedback, comments, and reports
- **Numerical data:** grades, scores, rubrics weights, performance metrics, analytics indicators
- **Image data:** scanned exam sheets, handwritten answers, uploaded documents

- **Metadata:** timestamps, exam identifiers, question identifiers, user roles, and configuration flags

Data Sources

Data enters the system from different sources depending on the workflow.

Primary data sources include:

- **Users:** students, instructors, administrators, and institutions through client interfaces
- **Uploaded files:** scanned exam papers, PDFs, images, and digital exam documents
- **Internal system components:** grading agents, feedback agents, analytics agents
- **External services:** OCR engines, authentication services, and notification providers
- **APIs:** institutional systems or scripts performing bulk operations

Data Formats

The system uses standardized data formats to ensure interoperability and scalability.

Common formats include:

- **JSON** for API communication, structured exam data, grading results, and feedback
- **CSV** for analytics exports, reports, and aggregated institutional data
- **PDF / Image formats** for exam uploads and scanned submissions
- **Vector representations** for embedding-based retrieval in RAG components

Example Data Structures

Below is an example of a structured student exam submission after OCR and preprocessing:

```
{  
  "exam_id": "EXAM_2026_01",
```

```

"student_id": "STU_1024",
"submission_type": "scanned",
"questions": [
  {
    "question_id": "Q1",
    "question_type": "open_ended",
    "student_answer": "The algorithm minimizes the loss function
using gradient descent.",
    "max_score": 10
  },
  {
    "question_id": "Q2",
    "question_type": "mcq",
    "student_answer": "B",
    "max_score": 5
  }
],
"submission_timestamp": "2026-02-01T14:35:22"
}

```

Example of grading and feedback output generated by the system:

```

{
  "student_id": "STU_1024",
  "exam_id": "EXAM_2026_01",
  "total_score": 13,
  "feedback": [
    {
      "question_id": "Q1",
      "score": 8,
      "comments": "Good explanation, but missing details about
convergence conditions."
    },
    {
      "question_id": "Q2",
      "score": 5,
      "comments": "Correct answer."
    }
  ],
}

```

```
"graded_at": "2026-02-01T14:36:10"  
}
```

Data Usage Across the System

- OCR and preprocessing components transform raw image data into structured text
- Grading agents consume structured answers and rubrics to generate scores
- Feedback agents use grading outputs and historical data for personalization
- Analytics agents aggregate numerical and textual data to produce insights and reports
- RAG components retrieve historical embeddings to support context-aware decisions

10. System Flow (High-Level)

This section describes the system flow for each user role and its main usage scenarios.

10.1 Student Flow

(Student – Exam Mode & Practice Mode)

Step 1: Student Sends a Request

The student interacts with the platform to:

- Take an official exam
- Upload scanned or handwritten exam answers
- Attempt adaptive practice exams
- Request grades, feedback, or progress reports

Step 2: System Validates the Request

The system:

- Authenticates the student
- Verifies exam or practice availability
- Validates submission format and completeness

Step 3: System Processes the Request

The system:

- Applies OCR and preprocessing if answers are scanned
- Structures questions and student responses
- Invokes grading agents
- Applies adaptive difficulty logic
- Retrieves historical performance using RAG
- Generates personalized feedback

Step 4: System Stores Results

The system stores:

- Exam or practice results
- Feedback and comments
- Updated student learning profile

Step 5: System Responds to the Student

The student receives:

- Grades and detailed feedback
- Practice insights and learning gaps
- Progress tracking information

10.2 Instructor / Teaching Staff Flow

(Exam Design, Grading Control, Student Monitoring)

Step 1: Instructor Sends a Request

The instructor uses the system to:

- Create and upload exams
- Upload answer keys or reference solutions
- Configure grading rules, rubrics, and feedback depth
- Create adaptive practice exams
- View student profiles and analytics

Step 2: System Validates the Request

The system:

- Authenticates the instructor
- Verifies exam and grading permissions
- Validates exam structure and configuration

Step 3: System Processes the Request

The system:

- Prepares exams for delivery
- Configures grading and feedback agents
- Processes student submissions automatically
- Aggregates student performance data

Step 4: System Stores and Updates Data

The system updates:

- Exam definitions and grading configurations
- Student grades and profiles
- Analytics summaries

Step 5: System Responds to the Instructor

The instructor receives:

- Confirmation of actions
- Grading results
- Detailed analytics and student performance insights

10.3 Administrator Flow

(User Management & System Reliability)

Step 1: Administrator Sends a Request

The administrator interacts with the system to:

- Manage users, roles, and permissions
- Configure system-level settings
- Monitor system health and operations

Step 2: System Validates the Request

The system:

- Authenticates the administrator
- Verifies administrative privileges

Step 3: System Processes the Request

The system:

- Applies access control rules
- Updates system configurations
- Collects logs and monitoring data

Step 4: System Stores and Logs Data

The system stores:

- Configuration changes
- Audit logs
- Operational metrics

Step 5: System Responds to the Administrator

The administrator receives:

- Confirmation messages
- System status and reliability indicators

10.4 University / College Flow

(Institutional Monitoring & Decision Support)

Step 1: Institution Sends a Request

The institution requests:

- Aggregated student performance data
- Term-level exam results
- Learning outcome and progression reports

Step 2: System Validates the Request

The system:

- Authenticates the institutional user
- Verifies access scope and permissions

Step 3: System Processes the Request

The system:

- Aggregates grading and analytics data
- Applies institutional-level analytics
- Evaluates learning trends and outcomes

Step 4: System Stores Generated Reports

The system stores:

- Institutional analytics
- Term-level reports

Step 5: System Responds to the Institution

The institution receives:

- Aggregated reports
- Insights supporting academic and credit-based decisions

11. Error Handling & Edge Cases

This section describes how ExamAI handles errors and exceptional situations.

Input Validation Errors

- Invalid or incomplete requests → rejected with clear error message
- Logs the error for monitoring

Empty or Incomplete Data

- Empty submissions, missing answer keys, unreadable scans → request rejected
- User notified to resubmit

Authentication & Authorization Errors

- Unauthorized actions → blocked, error returned, logged

OCR & Data Extraction Failures

- Low-quality scans or unsupported formats → flagged for re-upload or review
- Prevents unreliable data entering grading pipeline

AI Grading Exceptions

- Ambiguous or low-confidence answers → fallback grading or flagged for instructor review
- Model confidence logged

External Service Failures

- OCR or notification service fails → retries or fallback used
- Temporary service error returned if needed

System & Infrastructure Failures

- Backend crashes, DB issues, high load → standardized server error
- Data consistency maintained, issue logged

Data Consistency Issues

- Conflicting updates → rejected, last valid state preserved
- Administrators alerted

12. Assumptions & Constraints

This section lists the main assumptions and limitations for ExamAI.

Assumptions

- Users have internet access to use online exams and platform features.
- Students and instructors submit exams in supported formats (digital or scanned).
- The system supports **English-only content for OCR-based exam processing**, while **digitally submitted exams (non-OCR) may be in English or Arabic**.
- The system evaluates student answers based on reference solutions and instructor-defined grading rubrics, without performing continuous model training during exam evaluation.
- Required system resources (CPU/GPU, memory) are available for processing.
- Users follow proper authentication and role-based access procedures.

Constraints

- OCR performance depends on scan quality and clarity of **English text only**.
- The system may initially operate in a CPU-only environment, limiting grading speed.
- Adaptive evaluation relies on historical student performance; limited history may reduce personalization.
- Variations in handwriting styles may affect OCR accuracy and extraction quality.
- The platform must comply with security and privacy regulations, which may limit data sharing.