

VISA Management System

Software Requirements Specification

Version 1.0
9/15/2025

Stakeholder: Diane Zeenny

Team Members: Maaz Ashfaq, Harsh Bhavsar, Gabriel Pagudar

Table of Contents

1.0 Introduction	3
1.1 Purpose	3
1.2 Intended Audience	3
1.3 Scope	3
1.4 Definitions	3
1.5 References	3
2.0 Overall Description	4
2.1 Description	4
2.2 User Classes	4
2.3 Operating Environment	4
2.4 Constraints	5
2.5 Assumptions	5
2.6 Dependencies	5
3.0 Business Rules	6
4.0 Functional Requirements	7
5.0 Use Cases	8
Use Case: View Employee Data & History	10
Use Case: Sync Data with Excel	12
6.0 Data Requirements	14
6.1 Logical Data Model	14
6.2 Data Dictionary	15
7.0 External Interface Requirements	17
7.1 User Interface	17
7.2 Software Interface	18
7.3 Communication Interface	18
8.0 Nonfunctional Requirements	19
8.1 Availability Requirements (Priority 1)	19
8.2 Compatibility Requirements (Priority 2)	19
8.3 Reliability Requirements (Priority 3)	20
8.4 Performance and Scalability Requirements (Priority 4)	20
8.5 Security Requirements (Priority 5)	21
8.6 Usability and Maintainability Requirements (Priority 6)	21
9.0 Reporting & Supplemental Information	22

1. Introduction

1.1 Purpose

The purpose of this system is to provide the Office of International Students and Scholars (OISS) at UMBC with a secure, centralized Visa Management System (VMS) that directly links to the office's existing Excel workbook. The system will enable staff to manage and update international employee records in real-time, with changes made in Excel automatically reflected in the software and edits made in the software written back into Excel. This ensures that OISS staff maintain their familiar workflow while gaining powerful new tools to monitor visa timelines, generate reports for third-party agencies (e.g., DHS, DOJ), and receive proactive alerts for expiring statuses.

1.2 Intended Audience

Primary Users: OISS staff (International Scholar Coordinator, Diane Zeenny)

Secondary Users: UMBC administrators who review compliance

Developers/QA Team: Responsible for building and maintaining the system

Reviewers: Course instructor and teaching assistant for CMSC 447

1.3 Scope

The VMS will:

- Provide a dashboard with a live count of active cases, visa type distributions, and upcoming expirations.
- Maintain a two-way sync with Excel, so that edits in either Excel or the VMS automatically update the other
- Allow staff to view and edit individual employee records, including personal details, organizational unit, and visa history.
- Highlight employees whose visas are expiring within configurable time windows (e.g., 7 months).
- Support in report generation for compliance purposes, filterable by visa type, expiration window, or organizational unit.

1.4 References

U.S Visa Info:

<https://travel.state.gov/content/travel/en/us-visas/immigrate/the-immigrant-visa-process/step-1-submit-a-petition.html>

OISS Info:

<https://isss.umbc.edu/>

Microsoft Graph API:

<https://learn.microsoft.com/en-us/graph/use-the-api>

2.0 Overall Description

2.1 Description

The Visa Management System (VMS) is a browser-based application that builds directly on the current Excel process used by the Office of International Students and Scholars (OISS). The system establishes a two-way synchronization between the Excel workbook and the software. When changes are made in Excel, the VMS automatically updates its records; when staff edit data in the VMS, those changes are written back to the Excel file. This approach preserves the existing workflow while enhancing it with dashboards, proactive alerts, reporting tools, and individual employee case views.

2.2 User Classes

- **Staff Users (Primary):** OISS coordinators and staff who interact with international employee records daily. They review timelines, check flagged expirations, update employee data, and generate reports. Staff require an intuitive interface and the ability to quickly search and filter data.
- **Admin Users (Secondary):** OISS managers and staff who oversee the system. They configure alert thresholds, manage user accounts, maintain the Excel link, and ensure data integrity. Admins require deeper access to configuration and audit features.

2.3 Operating Environment

- The Visa Management System (VMS) will operate as a web-based application accessible through modern browsers, including Google Chrome, Microsoft Edge, Mozilla Firefox, and Safari.
- The system will be hosted on an UMBC-approved server cloud platform
- It will connect to an SQLite database for local data storage and use the Microsoft Graph API to maintain synchronization with the OISS Excel workbook stored in OneDrive or SharePoint

- Authentication will be handled using Supabase Auth with UMBC Single Sign-On (SSO) credentials
- The front-end interface will be developed with React, HTML, and CSS, ensuring responsive display on both desktop and tablet screens
- The back-end will use Node.js and Express to handle API requests, user authentication, and synchronization operations
- The system will be designed to operate during regular OISS business hours, but remain accessible 24/7 for authorized staff

2.4 Constraints

- The Visa Management System depends on the Microsoft Graph API to communicate with the Excel workbook, which means network connectivity and valid API credentials are required for synchronization
- Every change made through the system must be recorded in an audit log with the user name, action, and timestamp
- Because the stakeholder's Excel workbook is the primary data source, any major change to its structure or location may temporarily interrupt synchronization until the configuration is updated
- System performance and data availability are limited by Excel API rate limits and internet connection stability
- All configurations, credentials, and database files must be stored securely and excluded from public repositories

2.5 Assumptions

- OISS staff will continue using Excel as the base file for all records
- Each employee record will have a unique identifier (e.g., Employee ID) for reliable mapping.
- Staff and admins will access the system using UMBC-issued credentials.
- The Excel file will remain available on a secure shared location

2.6 Dependencies

The Visa management system relies on several external tools and services to function properly. It depends on:

- Microsoft Graph API for reading and writing data to the Excel workbook stored in OneDrive or SharePoint.

- SQLite as the local database engine for storing employee records, visa data, notes, and sync logs
 - Supabase Auth, for verifying user sessions and user roles
 - React, HTML, and CSS libraries for the user interface
 - Node.js and Express for API requests and synchronization logic
-

3.0 Business Rules

- **BR-1 (Excel as Source):** The Excel workbook shall be the operational source of truth. The VMS must read from and write back to this file to keep records synchronized.
 - **BR-2 (Unique Identifier):** Each employee record shall contain a unique identifier (e.g., Employee ID) to ensure updates map to the correct row.
 - **BR-3 (Expiration Flag):** A case shall be flagged when the visa expiration date is within 213 days (≈ 7 months) of the current date.
 - **BR-4 (Active Cases):** The dashboard shall only display active employee cases; inactive or archived records must be excluded.
 - **BR-5 (Audit Trail):** Every change (whether made in Excel or the VMS) shall be logged, including the user, timestamp, and before/after values.
 - **BR-6 (Conflict Resolution):** If the Excel file and the VMS are updated simultaneously, the system shall apply a last-writer-wins policy, with the conflict recorded in the audit log.
-

4.0 Functional Requirements

FR-1 Authentication

- The system shall allow Staff and Admin users to log in using UMBC-issued credentials.

FR-2 Excel Data Sync (Read)

- The system shall automatically refresh data when changes are made to the linked Excel workbook.

FR-3 Excel Data Sync (Write)

- When a Staff or Admin user edits a record in the VMS, the system shall write the changes back to the correct row in the Excel workbook.

FR-4 Dashboard Overview

- The system shall display:
 - The total number of active employee cases.
 - Counts of cases by visa type.
 - Counts of cases by organizational unit (College → Department → Unit).
 - The number of new cases added within the past 30 days (configurable).

FR-5 Expiration Alerts

- The system shall flag cases where the visa expiration date is within 213 days (7 months) or other configured thresholds (90/60/30 days).

FR-6 Employee Search

- The system shall allow Staff users to search for employees by name, Employee ID, visa type, or organizational unit.

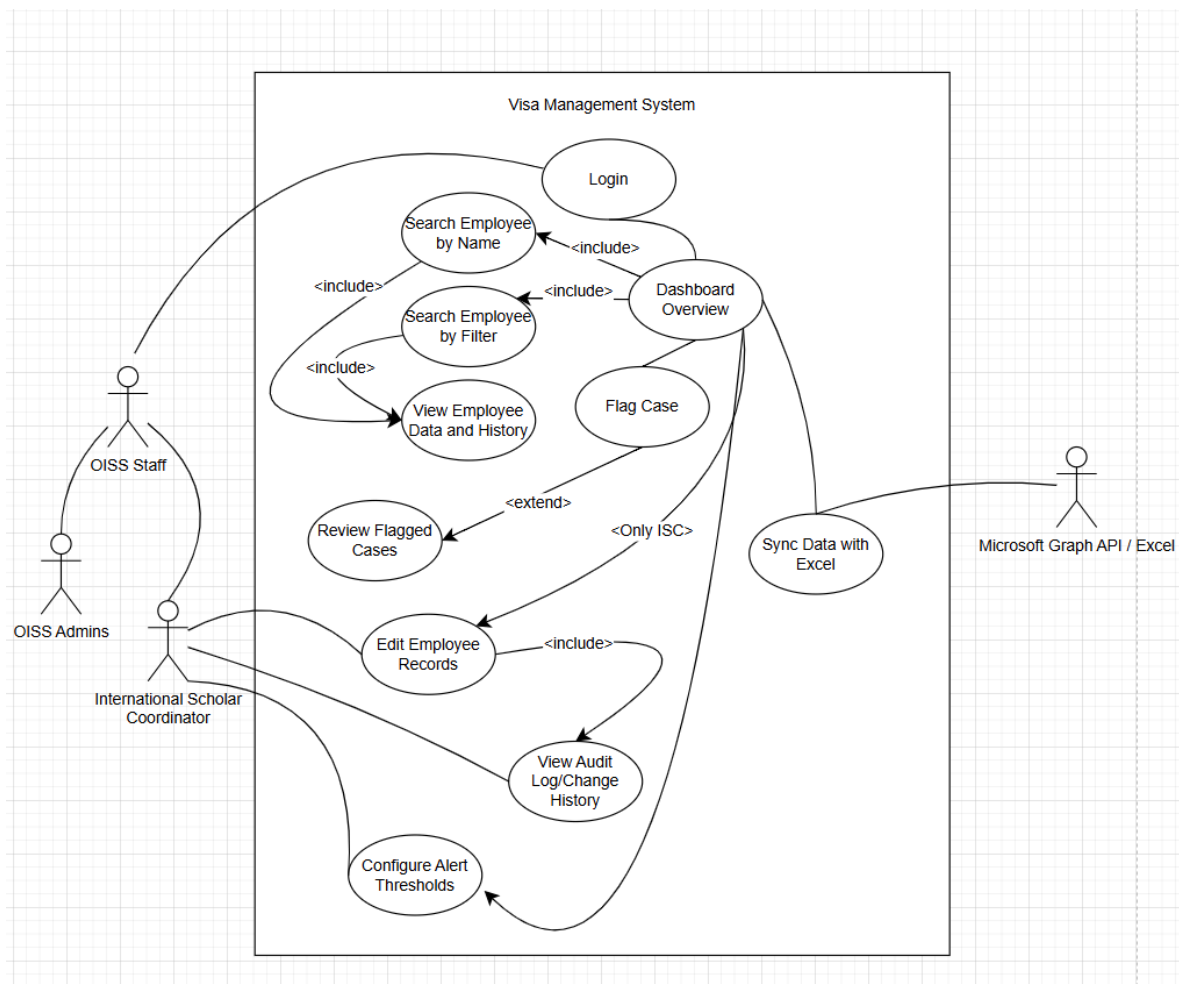
FR-7 Employee Record View

- The system shall display a detailed profile for each employee, including personal data, organizational affiliation, visa history, and expiration alerts.

FR-8 Audit Logging

- The system shall log all changes, including edits, imports, and exports, with user, timestamp, and before/after values.

5.0 Use Cases

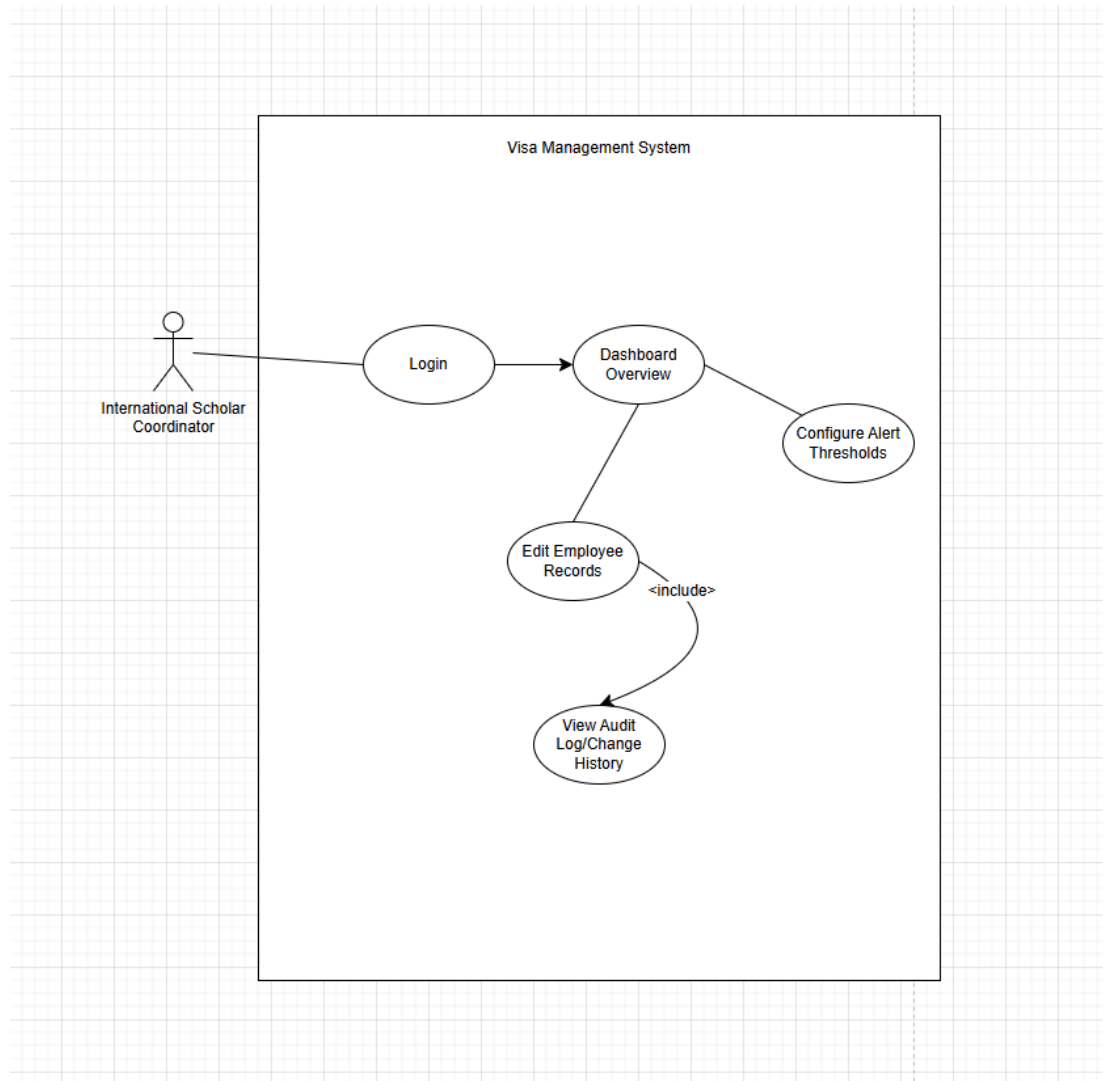


What it shows:

The big picture of how OISS roles interact with the VMS and the external Microsoft Graph API/Excel. From the Dashboard, users can search by name or other filters and then land on the "View Employee Data & History" page. Staff can flag a case that extends into the ISC's "Review Flagged Cases." Only the International Scholar Coordinator (ISC) can edit records, configure alert thresholds, sync with Excel, and view the audit log/change history. Microsoft Graph API/Excel appears as an external actor connected to sync. SQLite is internal (not shown as an actor).

- **Actors:** ISC; OISS Staff; OISS Admins (login/dashboard only); Microsoft Graph API/Excel (external).
- **Key relationships:**
 - Dashboard «include» → Search by Name; Search by Filter.
 - Search by Name/Filter «include» → View Employee Data & History.
 - Flag Case «extend» → Review Flagged Cases (handled by ISC).
 - Edit Employee Records «include» → View Audit Log/Change History.
- **Why this matters:** Captures system scope, permissions, and the one external integration at a glance.

Use Case: View Employee Data & History



What it shows:

This diagram shows how OISS roles, including Staff, Admins, and the International Scholar Coordinator (ISC), interact with the Visa Management System to view employee information. From the Dashboard Overview, users can search for employees either by name or by specific filters such as visa type or department. Both search actions lead to the View Employee Data & History page, where users can review detailed records including past visa information, status, and case history. If an issue is identified, users can flag a case, which later extends to the ISC's Review Flagged Cases use case.

Actors:

OISS Staff, OISS Admins, International Scholar Coordinator (ISC).

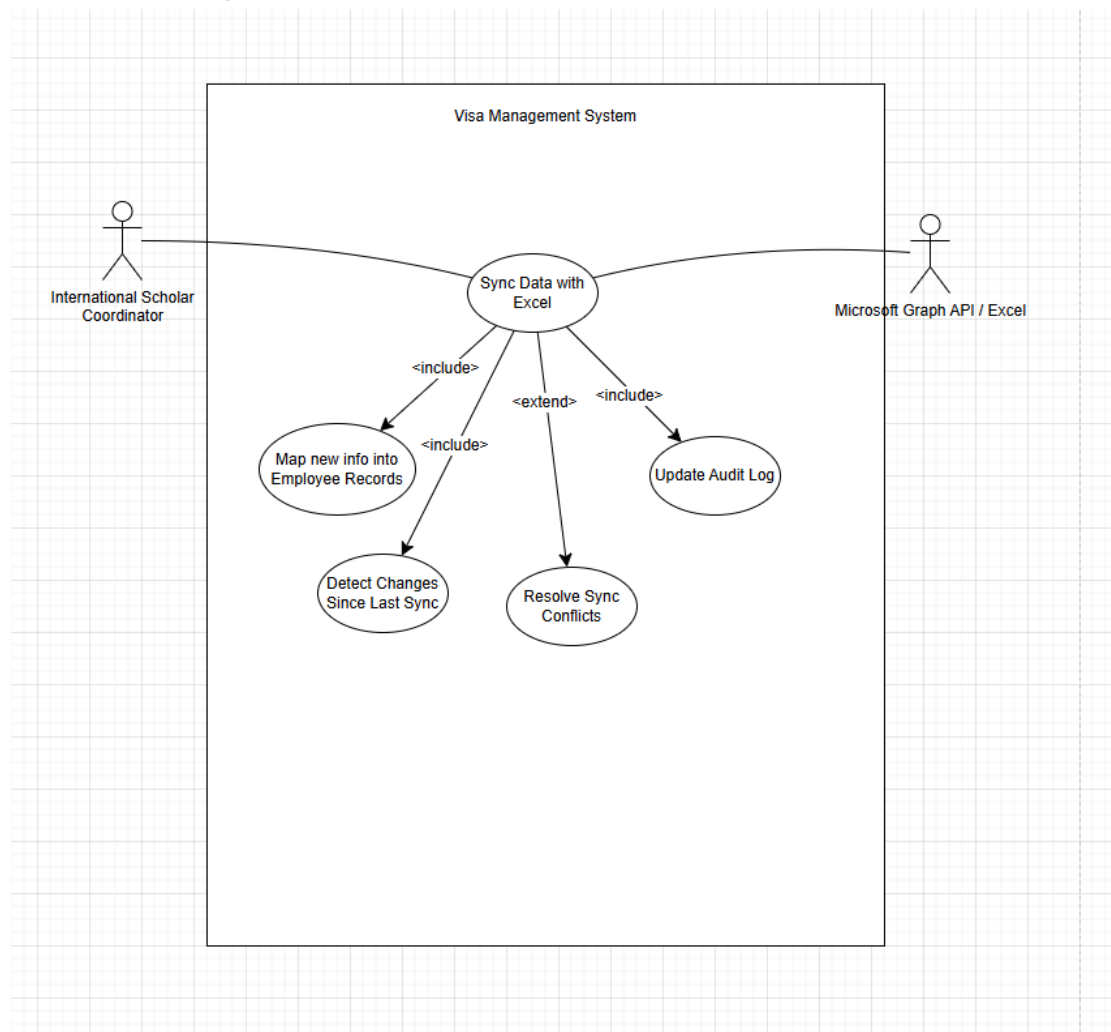
Key relationships:

- Dashboard Overview «include» → Search Employee by Name
- Dashboard Overview «include» → Search Employee by Filter
- Search Employee by Name/Filter «include» → View Employee Data & History
- Flag Case «extend» → Review Flagged Cases

Why this matters:

This use case represents the main workflow for users accessing employee data in the Visa Management System. It highlights how authorized users locate and review records efficiently while maintaining clear permissions between OISS roles. The extension to Review Flagged Cases ensures accountability and structured follow-up by the ISC.

Use Case: Sync Data with Excel



What it shows:

This diagram shows how the International Scholar Coordinator (ISC) interacts with the Visa Management System and the external Microsoft Graph API or Excel service to keep employee data up to date. The ISC can initiate a sync between the Visa Management System and the Excel sheet used by OISS. During this process, the system maps new information into employee records, detects any changes since the last synchronization, and updates the audit log. If differences or overlapping edits are found, the system can extend the process to resolve sync conflicts to ensure accuracy between both sources.

Actors:

International Scholar Coordinator (ISC), Microsoft Graph API or Excel (external system).

Key relationships:

- Sync Data with Excel «include» → Map New Info into Employee Records
- Sync Data with Excel «include» → Detect Changes Since Last Sync
- Sync Data with Excel «include» → Update Audit Log
- Sync Data with Excel «extend» → Resolve Sync Conflicts

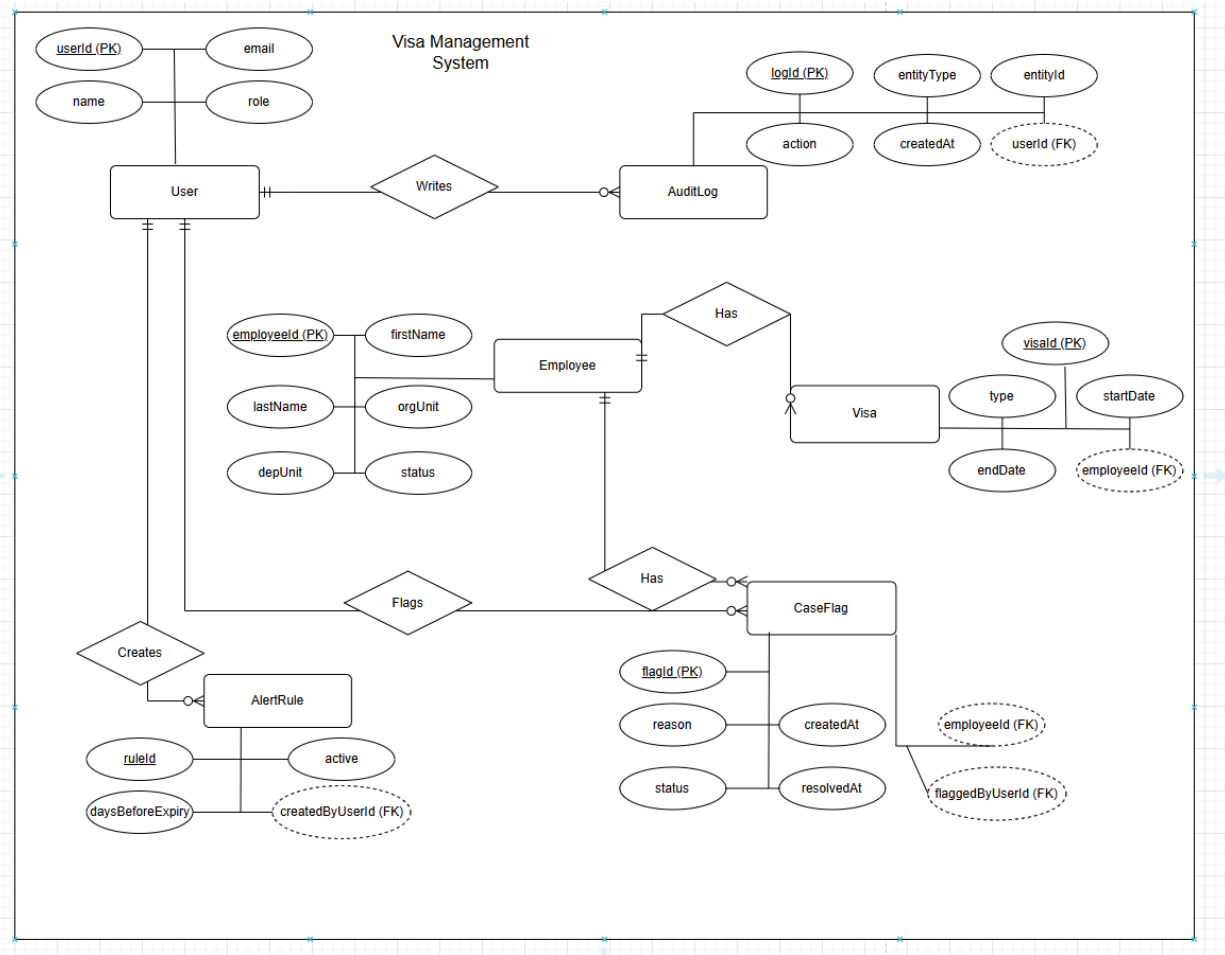
Why this matters:

This use case captures how the Visa Management System integrates with Excel through the Microsoft Graph API to support accurate and efficient data management. It ensures that employee information stays consistent between the web system and the official Excel database while maintaining an audit trail and resolving potential sync conflicts.

6.0 Data Requirements

6.1. Logical Data Model

6.1.1 Entity Relationship Diagram



6.2. Data Dictionary

Attribute	Entity	Description	Required	Type	Source Table	Length	Default Val	Comments
userId (PK)	User	Unique user identifier	Yes	UUID/INT	users			Primary key
name	User	Full name of the user	Yes	Text	users	100		
email	User	User email address	Yes	Text	users	254		Unique
role	User	User role in system	Yes	Enum/T	users	32	analyst	

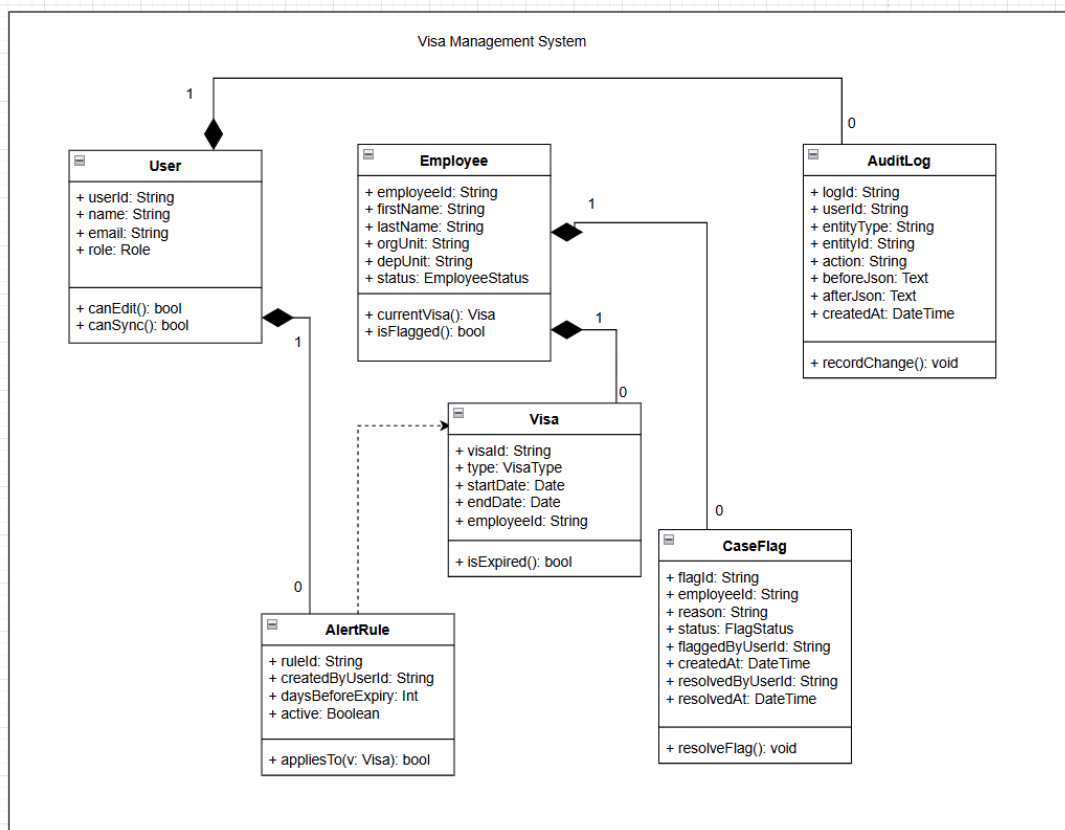
System Requirements Specification - Team 9

		(admin, analyst, etc.)		ext				
logId (PK)	AuditLog	Unique audit log identifier	Yes	UUID/INT	audit_logs			Primary key
action	AuditLog	Action performed (CREATE, UPDATE, DELETE)	Yes	Enum/Text	audit_logs	32		
entityType	AuditLog	Type of entity affected	Yes	Text	audit_logs	32		
entityId	AuditLog	ID of entity affected	Yes	UUID/INT	audit_logs			
createdAt	AuditLog	Timestamp when action occurred	Yes	Timestamp	audit_logs		CURRENT_TIMESTAMP	
userId (FK)	AuditLog	User who performed action	Yes	UUID/INT	audit_logs			FK → users.userId
employeeId (PK)	Employee	Unique employee identifier	Yes	UUID/INT	employees			Primary key
firstName	Employee	Employee first name	Yes	Text	employees	50		
lastName	Employee	Employee last name	Yes	Text	employees	50		
orgUnit	Employee	Organization unit	Yes	Text	employees	64		
depUnit	Employee	Department unit	No	Text	employees	64		
status	Employee	Employment status	Yes	Enum/Text	employees	16	Active	
visaId (PK)	Visa	Unique visa record	Yes	UUID/INT	visas			Primary key
type	Visa	Visa type (H1B, F1, OPT, etc.)	Yes	Enum/Text	visas	16		
startDate	Visa	Visa start date	Yes	Date	visas			
endDate	Visa	Visa end date	Yes	Date	visas			
employeeId (FK)	Visa	Employee linked to visa	Yes	UUID/INT	visas			FK → employees.employeeId
flagId (PK)	CaseFlag	Unique flag or case ID	Yes	UUID/INT	case_flags			Primary key
reason	CaseFlag	Reason for flag	Yes	Text	case_flags	255		

System Requirements Specification - Team 9

	ag							
status	CaseFlag	Case status (Open, Resolved)	Yes	Enum/Text	case_flags	16	Open	
createdAt	CaseFlag	When flag created	Yes	Timestamp	case_flags		CURRENT_TIMESTAMP	
resolvedAt	CaseFlag	When flag resolved	No	Timestamp	case_flags		NULL	
employeeId (FK)	CaseFlag	Employee associated	Yes	UUID/INT	case_flags			FK → employees.employeeId
flaggedByUserId (FK)	CaseFlag	User who flagged	Yes	UUID/INT	case_flags			FK → users.userId
ruleId (PK)	AlertRule	Unique alert rule ID	Yes	UUID/INT	alert_rules			Primary key
daysBeforeExpiry	AlertRule	Days before visa expiry for alert	Yes	Integer	alert_rules		30	
active	AlertRule	If rule is active	Yes	Boolean	alert_rules		TRUE	
createdByUserId (FK)	AlertRule	User who created rule	Yes	UUID/INT	alert_rules			FK → users.userId

System Requirements Specification - Team 9



7.0 External Interface Requirements

7.1 User interface

- 7.1.1. The system must provide a secure and responsive interface for OISS staff and administrators to view, search, and manage visa records.
- 7.1.2. The dashboard must visually display active cases, visa types, and upcoming expirations with filtering and sorting options for ease of access
- 7.1.3. Users must be able to add and edit employee or scholar details, view visa history, and flag cases requiring attention.
- 7.1.4. The interface should maintain a consistent layout and color scheme aligned with the university and accessibility and usability standards.

7.2 Software Interfaces

- 7.2.1. The system must integrate with Microsoft Graph API to enable two-way synchronization with the existing Excel workbook used by OISS.
- 7.2.2. The backend must connect to a SQLite database for local data storage, ensuring compatibility with RESTful APIs built using Node.js and Express.
- 7.2.3. Authentication and authorization must be managed through Supabase Auth for secure user access and session handling.

7.3 Communication Interfaces

- 7.3.1. The system must send automated notifications to OISS staff when visa expirations or compliance deadlines approach.
- 7.3.2. The application must log every user action (create, edit, delete) in an AuditLog to support internal reporting and compliance tracking.
- 7.3.3. Reports for agencies such as DOS and DHS must be exportable from the system in a standardized format (Excel or CSV).

Mockup Designs:

Dashboard

of live cases (not cumulative)
12

F-1 cases
5
OPT: 2 • STEM OPT: 1

J-1 cases
2

H-1B / PR
3 / 2

of upcoming cases in the next 3 months

2 case(s)

NAME	VISA	END DATE	DUE
Name 1	F-1 OPT	9/30/2025	4d
Name 9	F-1 OPT	12/4/2025	69d

New cases

NAME	VISA	START DATE	IN SYSTEM
Name 10	J-1 Intern	5/31/2024	483d
Name 5	PR EB-2	7/11/2024	442d
Name 3	H-1B CAP	9/30/2024	361d

Employees & Scholars

212
of live records

126
F-1 cases


48
J-1 cases

28 / 10
H-1B / PR

212 shown

Name	Visa	Position	College	Academic Dept.	Start	Status
Dr. A. Scholar	J-1	Professor	COEIT	Computer Science	08/15/2024	Full Time
Jane Doe	F-1 OPT	Teaching Assistant	CAHSS	Psychology	09/01/2024	Part Time
John Researcher	J-1	Postdoctoral Researcher	CNMS	Biology	07/10/2024	Full Time
Maria Student	F-1	Library Assistant	CAHSS	History	09/05/2024	Casual
Ali Worker	F-1 STEM OPT	Food Services	COEIT	Mechanical Eng.	10/01/2024	Contract
Sam Intern	J-1	Short-Term Scholar	CNMS	Chemistry	06/15/2024	On Leave

Employee History - Dr. A. Scholar



Dr. A. Scholar

Professor • COEIT • Computer Science

Visa & Employment History

Start Date	End Date	Visa	Position	Status
08/15/2024	Present	J-1	Professor	Full Time
07/01/2022	08/01/2024	J-1	Postdoctoral Researcher	Full Time
09/01/2020	06/30/2022	F-1 OPT	Teaching Assistant	Part Time

Notes

• 08/20/2024 - Scholar extended J-1 status by 2 years

• 07/15/2023 - Published research paper in AI journal

• Add new note...

8.0 Nonfunctional Requirements

8.1 Availability Requirements (Priority: 1)

The Visa Management System (VMS) shall maintain **99% uptime** excluding scheduled maintenance. Regular automated backups of the SQLite database and linked Excel workbook (via Microsoft Graph API) shall prevent data loss in the event of an outage. The system must recover gracefully from temporary sync interruptions by queuing pending write operations until connectivity is restored.

8.2 Compatibility Requirements (Priority: 2)

The VMS shall be compatible with all major browsers, including **Google Chrome, Microsoft Edge, Mozilla Firefox, and Safari**, and support operating systems such as **Windows, macOS, and Linux**. The system shall maintain responsive layouts on tablets and mobile devices.

Integration with **UMBC's Microsoft 365** ecosystem through the Graph API is required to support authentication and Excel synchronization.

8.3 Reliability Requirements (Priority: 3)

The system shall ensure **data integrity** during two-way synchronization. Each transaction will be logged with a unique identifier, timestamp, and user ID. In the event of conflicting updates between Excel and the VMS, the **last-writer-wins** rule will apply, with all conflicts recorded in the audit log. The system shall remain functional in read-only mode during connection interruptions to prevent data corruption.

8.4 Performance and Scalability Requirements (Priority: 4)

The system shall efficiently handle **200+ active records** with near real-time synchronization. Performance degradation must remain below 10% when scaling up to **1,000 records**. Common operations such as search queries, dashboard loading, and data synchronization shall execute within **two seconds** on average. The backend architecture (Node.js + Express) shall allow for easy migration from SQLite to PostgreSQL or Supabase for scalability.

8.5 Security Requirements (Priority: 5)

Access to the system shall be restricted to **UMBC-authenticated users** through Supabase Auth with JWT-based session management. Role-based access control (Admin, ISC, Staff) shall limit permissions to appropriate functions. All communication must use **TLS 1.2 or higher** for encryption in transit. Data shall be securely stored, and inactive sessions shall automatically time out after **30 minutes**. Audit logs must capture user actions for compliance tracking.

8.6 Usability and Maintainability Requirements (Priority: 6)

The VMS interface shall follow **UMBC web accessibility standards**, including adequate color contrast, ARIA labels, and keyboard navigation. The design shall be intuitive for OISS staff with minimal training required. Code shall be modular, well-commented, and follow standardized naming conventions. System errors and alerts must be clearly communicated to the user while maintaining professional tone and log entries for diagnostics.

9.0 Reporting & Supplemental Info

9.1 Reporting Requirements

1. The VMS shall support automated and manual report generation for the Office of International Students and Scholars (OISS).
2. Reports shall include:
 - Total active employee cases, grouped by visa type, organizational unit, and expiration window.
 - Summaries of upcoming expirations and flagged cases.
 - Audit trail summaries showing edits and responsible users.
3. Reports must be exportable in Excel (.xlsx) and PDF formats. Headers shall include the report title, stakeholder name, and generation date. Footers shall include page numbers and a timestamp.
4. The reporting interface shall support filtering by visa type, expiration date, and department, and allow users to save report presets for repeated use.
5. The system shall support scheduled report generation, allowing automated monthly summaries to be emailed to designated administrators.

9.2 Supplemental Information

9.2.1 Front-End

- Developed with React 18 using Bootstrap and Tailwind CSS for responsive and accessible design.
- Components include: Login, Dashboard, Employee Search, Employee Record View, and Report Export modules.
- Visual indicators (icons, color-coded alerts) will highlight expiring visas and synchronization status.

9.2.2 Back-End

- Implemented using Node.js + Express, with data stored in SQLite for initial deployment.
- Integrated with Microsoft Graph API for live Excel synchronization.
- Supabase Auth handles user authentication and authorization.
- RESTful API endpoints support data retrieval, editing, logging, and report generation.

9.2.3 Coding Practices

- Modular architecture: separate directories for frontend components, backend services, and utility functions.
- Version control: GitHub repository using branching workflow (main, develop, feature/*).
- Linting and formatting: enforced via ESLint and Prettier.
- Documentation: JSDoc and inline comments for maintainability and onboarding.
- Testing: manual verification via seeded demo data; integration testing planned in later sprints.

9.2.4 Additional Considerations

- Ensure compliance with ADA and Section 508 accessibility standards.
- System designed for future scalability — migration from SQLite to Supabase or PostgreSQL supported.
- AI integration (future feature): potential use of AI summarization for compliance reports.
- Deployment targets include Render or Heroku, with automated CI/CD pipelines for build and testing.
- Regular stakeholder reviews ensure continuous improvement and user-centered updates.