**Project Summary**

Development of an integration framework that enables seamless connectivity between the AuditCue platform and any third-party business applications, similar to Zapier's functionality. This framework should allow users to create automated workflows between AuditCue and external systems without requiring extensive programming knowledge.

**Project Duration**

Recommended timeline: 10 weeks (standard internship duration)

**Functional Requirements**

**Core Integration Capabilities**

1. Authentication Management
   * Support for OAuth 2.0 authentication flow
   * Secure credential storage system
   * Token refresh mechanism
   * Support for API key authentication
2. Connection Management
   * Interface for adding new third-party connections
   * Connection status monitoring
   * Connection testing functionality
   * Ability to revoke/delete connections
3. Workflow Builder
   * Visual workflow designer interface
   * Trigger selection mechanism
   * Action configuration system
   * Data mapping interface
   * Error handling configuration
4. Data Transformation
   * Field mapping capabilities
   * Data type conversion
   * Basic transformation functions (string manipulation, mathematical operations)
   * Custom transformation script support

**Initial Integration Targets**

Priority integrations to implement:

1. Google Workspace (Gmail, Google Drive, Google Calendar)
2. Microsoft 365 (Outlook, SharePoint, Teams)
3. Slack
4. Jira
5. ServiceNow

**Technical Requirements**

**Backend Development**

1. API Development
   * RESTful API design
   * OpenAPI/Swagger documentation
   * Rate limiting implementation
   * Request/response validation
   * Error handling middleware
2. Database Design
   * Schema design for:
     + User connections
     + Workflow definitions
     + Execution logs
     + Authentication tokens
   * Database migration system
   * Data backup mechanism
3. Job Processing
   * Asynchronous job queue system
   * Job status tracking
   * Retry mechanism for failed jobs
   * Job logging system

**Frontend Development**

1. User Interface
   * Workflow builder interface
   * Connection management dashboard
   * Execution history viewer
   * Error monitoring interface
2. Component Development
   * Reusable UI components
   * Form validation
   * Error handling
   * Loading states
   * Responsive design

**Security Requirements**

1. Authentication & Authorization
   * User authentication system
   * Role-based access control
   * API authentication
   * Session management
2. Data Security
   * Encryption at rest
   * Encryption in transit
   * Secure credential storage
   * PII data handling

**Technical Stack**

**Recommended Technologies**

1. Backend
   * Language: Go Programming
   * Framework: REST Framework
   * Database: SQLite
   * Queue System: Redis or its equivalent service in AWS
   * API Documentation: OpenAPI/Swagger
2. Frontend
   * Framework: React
   * State Management: Redux or React Query
   * UI Components: Material-UI or Tailwind CSS
3. Infrastructure
   * Containerization: Docker
   * Hosting: AWS

**Development Phases**

**Phase 1: Foundation (Week 1)**

* Project setup and infrastructure
* Basic API structure
* Database schema design
* Authentication system implementation

**Phase 2: Core Framework (Weeks 2-4)**

* Connection management system
* Basic workflow engine
* Data transformation system
* Initial third-party integration (Google Workspace, Pick any from the above integration list)

**Phase 3: UI Development (Weeks 4-6)**

* Workflow builder interface
* Connection management dashboard
* Basic monitoring and logging interface

**Phase 4: Integration Expansion (Weeks 6-8)**

* Additional third-party integrations
* Enhanced error handling
* Performance optimization
* Documentation

**Phase 5: Testing and Polish (Weeks 8-10)**

* Comprehensive testing
* Bug fixes
* Performance optimization
* Documentation completion

**Deliverables from the project**

**Documentation**

1. API Documentation
   * Complete OpenAPI/Swagger documentation
   * Integration guides for supported platforms
   * Authentication flow documentation
2. Technical Documentation
   * System architecture documentation
   * Database schema documentation
   * Deployment guide
   * Development setup guide
3. User Documentation
   * User manual
   * Integration setup guides
   * Troubleshooting guide

**Code**

1. Source Code
   * Well-documented code with comments
   * Unit tests
   * Integration tests
   * Example implementations
2. Infrastructure Code
   * Docker configurations
   * Development environment setup scripts