Scenario 1
An integration between 2 systems using REST APIs and webhooks

Contents

1. Feature Document

1.1. Problem / Goal

The client wants to integrate System A (Document Automation and E-Signature System) with System B (Document Management System) to save documents generated on System A in System B

1.2. Solution (Proposal)

The proposed solution is to create an API integration between System A and System B and trigger API calls from System A to System B to create folders, upload documents, and receive webhooks to confirm successful actions.

1.2.1. Triggers and consequent actions table

Trigger	Consequent Action
The document is manually generated on System A	The document is saved to System B as the first draft
A new version of the document is saved on System A	A new version of the document is saved onto System B
The document is e-signed on System A	The execution version of the document is saved as a .pdf file onto System B
The execution version of the document is saved onto System B	Document on System A is deleted

1.2.2. API Endpoint Reference Table

Endpoint Name	Description
Get Drives	Retrieves the drives and folder structure of a user in System B
Create Folder	Creates a new folder in System B
Upload File	Uploads a file to System B
Rename File	Renames a file in System B

1.3. Platform

System A and System B both have APIs that can be used to integrate them. System A will trigger API calls to System B, and System B will send webhooks to System A

1.4. Success metrics and Indicators

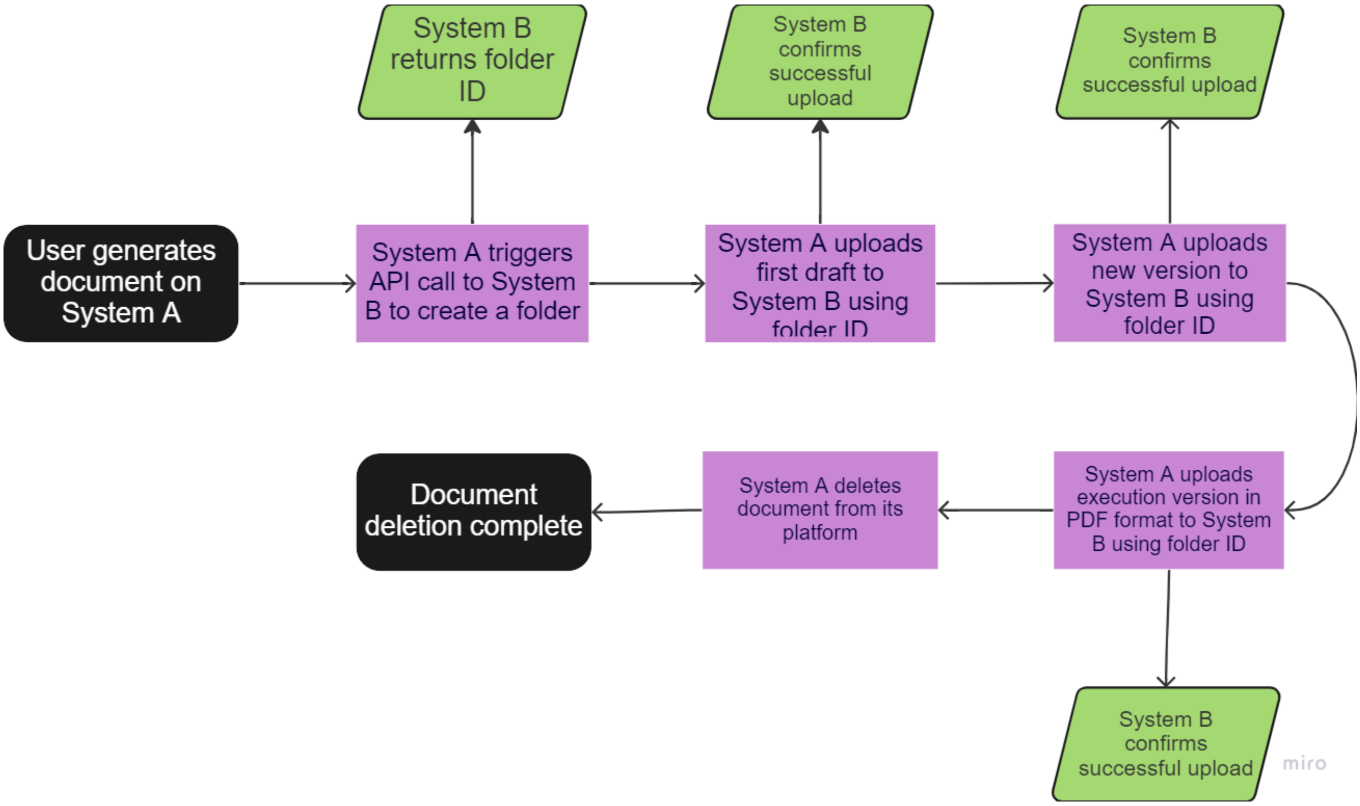
Success will be measured by the successful integration of the two systems, where documents generated on System A are saved in System B, and the user flow described by the client is achieved.

1.5. Potential Challenges

One potential challenge could be authentication and security, as System B requires a bearer token for API access. Another challenge could be ensuring that the webhooks from System B are received and processed correctly by System A.

1.6. User Flow

The user flow for the integration is as follows:


System B returns folder ID
System B confirms successful upload
System B confirms successful upload
User generates document on System A
System A triggers API call to System B to create a folder
System A uploads first draft to System B using folder ID
System A uploads new version to System B using folder ID
Document deletion complete
System A deletes document from its platform
System A uploads execution version in PDF format to System B using folder ID
System B confirms successful upload
miro

1.7. Rollout Strategy

The rollout strategy for the integration can be as follows:

1. Develop and test the integration in a development environment.
2. Perform user acceptance testing with a small group of users.
3. Roll out the integration to a larger group of users.
4. Monitor usage and gather feedback.
5. Make improvements and updates based on feedback.

2. Development Cycle

Requirements Gathering	Design and Prototyping	Development	Testing	Deployment	Monitoring and Maintenance
The first stage of the development cycle is to gather requirements from stakeholders, end-users, and technical teams from both System A and System B.	Based on the requirements gathered, a high-level design of the integration is created and a prototype is developed to demonstrate the functionality.	Once the prototype has been approved, the development team begins the process of developing the integration. This involves coding, testing, and debugging.	The integration is thoroughly tested to ensure it meets all the requirements and performs as expected. This includes functional, integration, performance, and security testing.	The integration is deployed into the production environment and made available to end-users.	The integration is monitored to ensure it continues to perform as expected, and any issues are addressed in a timely manner.

3. Deliverables

Integration Documentation	API Endpoint Reference Table	User Flow Diagram	Code and Deployment Scripts	Test Cases and Test Results
Detailed documentation of the integration, including the design, architecture, and implementation details.	A table that lists the API endpoints available in System B and a brief description of what each endpoint does.	A visual depiction of the flow of the user experience, starting from document generation on System A and ending with document deletion.	All the code and deployment scripts required to implement the integration, along with detailed instructions on how to deploy the integration into a production environment.	A comprehensive set of test cases and test results to demonstrate the integration meets all requirements and performs as expected.

Scenario 2 - Feature request

Contents

Feature Document: Automated Reminder Emails

1.1. Problem / Goal

The client wants to be able to schedule reminder emails that would go out to predetermined recipients based on the value of the document generated on System A

1.2. Solution (Proposal)

A new workflow action will be added to System A's Document Automation and Negotiation platform to allow the scheduling of reminder emails. The workflow will be triggered when a document with a value greater than 10000000 is generated on System A. The client will be able to customise the content of the email and the schedule for sending the reminder email. The schedule set-up will include the ability to select the number of days or business days from the triggering of the workflow and the time and time zone of the day the reminder email will be sent out.

1.2.1. Schedule Set-Up Table

Days/Business Days	Time	Time Zone
5	9:00 AM	PST
7	11:00 AM	EST
10	2:00 PM	GMT

This table represents a sample schedule set-up for the reminder email, where the user can select the number of days or business days from the triggering of the workflow, the time, and the time zone. The table can be easily updated and customised based on the user's needs.

1.2.2. Triggers and consequent actions table

Trigger	Action	Conditions	Recipients	Email Content	Schedule
Document Value > 10000000	Send reminder email	Value of document	approver@client.com	Customisable	Customisable (number of days/business days from triggering, time, time zone)

This table provides a concise and easy-to-understand overview of the automated workflow and how it meets the client's business requirements. This table clearly outlines the relationship between the trigger (document value greater than 10000000) and the action taken (sending a reminder email to "approver@client.com"). The conditions for the action to be taken, the recipients of the reminder email, the customisable content of the email, and the customisable schedule for the reminder email are also clearly specified.

1.2.3. API Endpoint Reference Table

Endpoint	Description
/schedule_reminder	Endpoint for scheduling reminder emails based on the specified conditions
/customise_content	Endpoint for customising the content of the reminder email

/customise_schedule	Endpoint for customising the schedule (number of days/business days, time, time zone) for sending the reminder email

This table lists the different API endpoints available for scheduling the reminder emails and customising the email content and schedule. Each endpoint is described briefly to give an understanding of its purpose and function. This table can be a reference for the development team and any future updates or modifications to the workflow.

1.3. Platform

System A Document Automation and Negotiation platform.

1.4. Success metrics and Indicators

Success Metrics and Indicators:

- Increased efficiency in communication and follow-up with clients
- Improved document value tracking
- Increased client satisfaction with the platform.

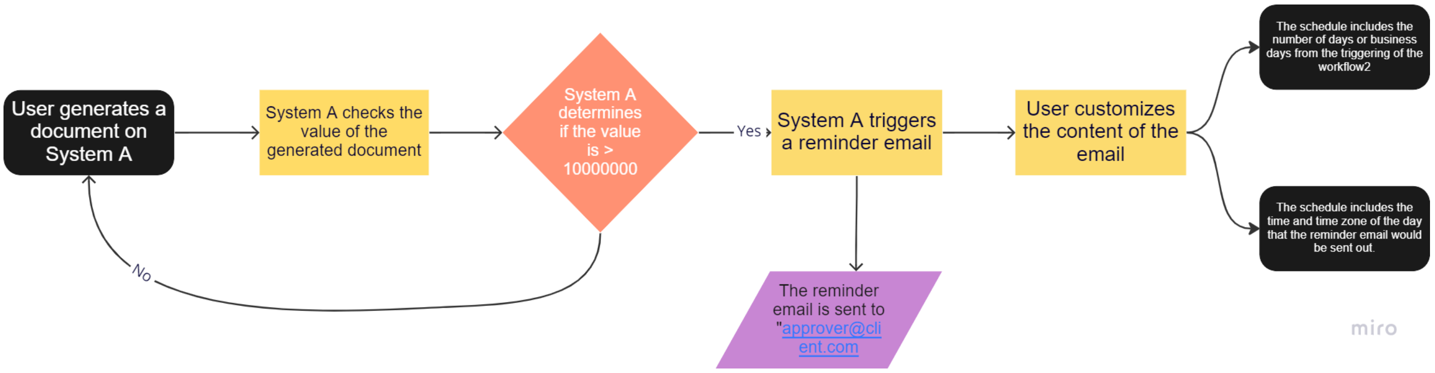
1.5. Potential Challenges

Potential Challenges:

- Integration with the client's email system
- User adoption and usage of the new feature
- Data accuracy and consistency.

1.6. User Flow

The user flow for the integration is as follows:

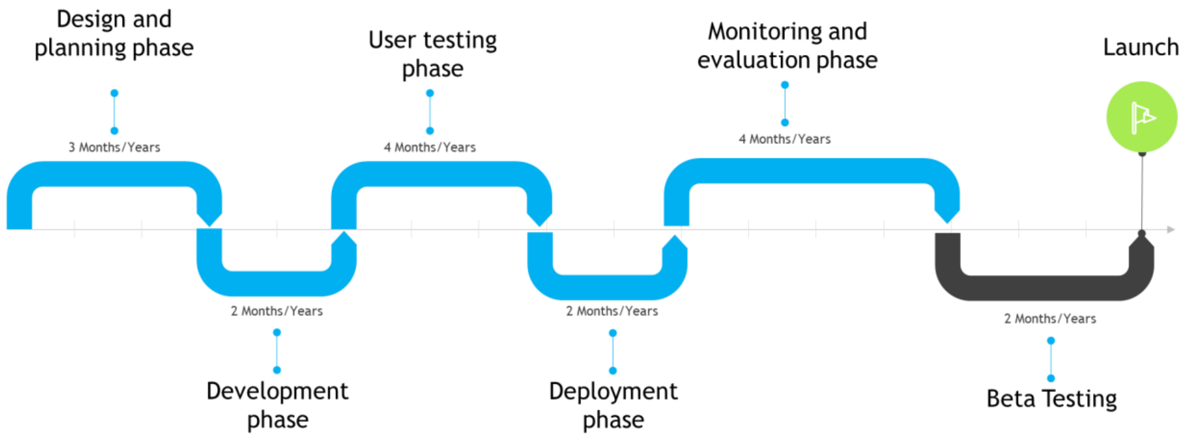

User generates a document on System A
System A checks the value of the generated document
System A determines if the value is > 10000000
Yes
No
System A triggers a reminder email
User customizes the content of the email
The schedule includes the number of days or business days from the triggering of the workflow2
The schedule includes the time and time zone of the day that the reminder email would be sent out.
The reminder email is sent to "approver@client.com
miro

1.7. Rollout Strategy

Rollout Strategy:

1. Internal testing of the new feature.
2. Communication and training for the client on the new feature.
3. Roll out to a small group of clients for a trial period.
4. Expansion to all clients based on feedback and results from the trial period.

Rollout Timeline Chart



2. Development Cycle

Requirements Gathering and analysis	Design and development	Testing and quality assurance	Deployment to production environment	Maintenance and support
• Understanding the client's business requirements and use case. • Identifying potential challenges and constraints. • Defining the scope of the project and the required resources. • Developing a clear and comprehensive set of requirements for the development team. • Reviewing the existing set-up page for the Add Approver action for reference.	• Designing the user interface and user experience for the feature, including the scheduling set-up and customisation options. • Developing the automated reminder email feature, including integration with the client's email system. • Implementing the customisation options for the content of the email and the scheduling of the reminder email. • Testing the feature to ensure its functionality and performance. • Debugging any issues that arise during development.	• Conducting internal and user acceptance testing to validate the feature's functionality and performance. • Verifying that the customisation options for the email content and scheduling are functioning as intended. • Addressing and fixing any bugs or issues that arise during testing. • Ensuring the feature meets System A's standards for security and data privacy.	• Deploying the feature to the production environment for the client to use. • Monitoring the feature's performance and usage to ensure its stability and reliability. • Implementing any necessary security measures to protect the data and systems involved.	• Providing ongoing support and maintenance to ensure the feature continues to function as intended. • Monitoring and addressing any issues or bugs that may arise in the future. • Updating the feature to address changing business requirements or to take advantage of new technologies.

3. Deliverables

User interface and user experience design for the feature

- A detailed design document that outlines the user interface and user experience for the Automated Reminder Emails feature.
- Wireframes, mockups, and prototypes that demonstrate the design and functionality of the feature.

Automated reminder email feature, integrated with the client's email system

- A fully functional automated reminder email feature, integrated with the client's email system.
- Customisation options for the content of the email and the scheduling of the reminder email.
- A secure and reliable system for storing and managing the data involved in the feature.

User documentation and training materials

- User documentation that explains how to use the Automated Reminder Emails feature, including instructions for customising the email content and scheduling.
- Training materials that assist the client in getting started with the feature.
- Support and assistance during the rollout and adoption of the feature.

Support and assistance during the rollout and adoption of the feature

- On-site support and assistance during the initial rollout of the feature to the client.
- Ongoing remote support and assistance as needed.
- A plan for addressing any issues or bugs that may arise during the rollout and adoption of the feature.