ISSN: 2581-3404 (Online) IF: 5.86 (SJIF)
IJIRTM, Volume-10, Special Issue-2, April-2026

Flow Sync: A Workflow Automation System

Shubhaam[1], Sanyam[2], Hardik[3], Satyam[4], Harkesh[5]
Department of Computer Science & Engineering (AI & DS), Panipat Institute of Engineering and Technology, Panipat, India [1,2,3,4,5]
dahiyashubhaam@gmail.com[1], sanyamdureja08@gmail.com[2], hardikantil43@gmail.com[3], satyammitrav@gmail.com[4]

Abstract. *Manual coordination of disconnected digital tools is repetitive and it takes time and introduces avoidable errors into day-to-day work. FlowSync is a workflow automation platform (SaaS) that is designed specifically to do this. It integrates Google Drive, Slack, Notion, and Discord with a webhook-based backend, allowing them to build event-driven pipelines by using a drag-and-drop interface, without a single line of code. When a file arrives in Google Drive, e.g., the system can automatically add a Slack notification and add an entry in Notion in seconds and without the human touch. The site is developed on Next.js, Node.js, Express.js, PostgreSQL and Prisma ORM, and Clerk to provide authentication. The experiments in various automation conditions proved the presence of reliable trigger detection and fast action execution. The current implementation does not include artificial intelligence, which is viewed as an extension of workflow suggestion in the future.*

Keywords: Workflow Automation, Trigger-Action Systems, SaaS Platform, Webhook Integration, OAuth 2.0, API Integration, Node.js, Drag-and-Drop Interface.

Introduction

Contemporary teams work using an ever-expanding set of disinter connected tools file storage, messaging apps, databases, and collaboration apps that run in parallel with very little coordination among them. The act of switching these services to perform repetitive and rule-based tasks consumes valuable time and errors that can be avoided in total through automation are introduced. A notification has to be sent out when a file is uploaded. A record of task is to be formed when a message comes. These are predetermined patterns which do not require human intervention.

Flow Sync is an automation workflow system that is designed to address exactly such patterns. Users can connect their accounts with Google Drive, Slack, Notion, and Discord and create pipelines with a visual canvas: a trigger node on the left, action nodes on the right, and the edge between them. The backend listens to trigger events via webhooks and triggers the actions that are configured using the corresponding platform APIs. This entire process occurs even when the user does not open any of the apps connected. Platforms like IFTTT and Zapier established commercial demand for this category of software, but each is a trade-off, constraining usefulness. IFTTT is not that complex and can not process multi-step logic. Zapier is expensive and only supports complex flows, but it is only familiar with API concepts and small teams. Flow Sync aims at the gap between them, a mid-tier solution that can be easily accessible to non-technical users and takes on multi-action workflows and integrations like Notion and Discord, which many of the competing products do not.

The current implementation does not have any artificial intelligence. The literature on the matter continues to demonstrate that AI adoption in automation platforms is raising legitimate concerns in the areas of transparency, reliability, and computational cost. It is possible to realize the main automation objectives here with well-designed rule-based event processing and none of those trade-offs. Intelligence workflow suggestion AI will be introduced as an improvement in the future after the utilization patterns have been collected based on actual deployments.

2. Related Work

Workflow automation research includes program synthesis, privacy engineering, usability, system integration. The following ten studies were reviewed and influenced the design choices made in Flow Sync. Dalal and Galbraith [1] experimented with Seq2Seq models that are used to automatically write if-then automation programs based on natural-language input. Transformer architecture performed better than LSTM and OpenNMT, but all models were not able to handle multi-step logic, and the quality of results was very much dependent on training data. This established that FlowSync would employ a visual builder as opposed to NLP to ensure it was not dependent on any dataset and still accessible by non-technical users. Rahmati et al. [3] compared IFTTT with Zapier in terms of usability and expressiveness. Their observation that no extant platform provided a suitable service to both easy and complex use cases at once was the direct inspiration behind the design objective of Flow Sync being to offer both low-code interface and multi-step workflow richness in a single platform. Chen et al. [2] considered the privacy of data in trigger-action systems through secure multi-party computation. Their eTAP system maintained privacy but was 3-5 times slower than normal pipelines. FlowSync addresses this trade-off by encrypting OAuth tokens at storage and validating webhook contents with HMAC signatures, offering significant security without the latency overhead of MPC. Davis [5] enumerated the challenges in integration with enterprise automation, which were inconsistent APIs, and authentication conflicts as the most common blockers. Flow Sync normalises OAuth 2.0 between all four of the interconnected service and normalises data schema with Prisma ORM, minimising the surface area of such failures. Lightweight state management libraries are more appropriate to mid-scale reactive applications than the heavier architecture of Redux, which prompted Brown [6] to select a React context provider as the state of the workflow editor in Flow Sync. Wilson [7], Robinson [9], and Harris [10] all found efficiency improvements of AI-enhanced automation of between 10% and 30% but also expressed worries about model transparency and training cost. These results justify the choice to make AI an addition but not a dependency in the present time. White [8] discovered that smart automation systems provided superior ROI on conditional logic flows, the latter being the basis of the future roadmap and not architecture of Flow Sync.

Table 1: Literature Survey Summary

Ref	Authors / Title	Year & Method	Finding & Relevance
[1]	Dalal & Galbraith – Seq2Seq Models for If-Then Program Synthesis	2020 | Seq2Seq, Transformer	Transformer outperformed LSTM for trigger-action synthesis; dataset quality is the bottleneck. FlowSync avoids NLP entirely, using a visual node builder.
[2]	Chen et al. – Data Privacy in	2021 | MPC /	Privacy preserved but 3-5x slower than standard

Ref	Authors / Title	Year & Method	Finding & Relevance
	Trigger-Action Systems	eTAP	pipelines. FlowSync uses OAuth 2.0 token encryption rather than MPC to keep latency acceptable.
[3]	Rahmati et al. – IFTTT vs. Zapier Comparative Study	2017 | Comparative	IFTTT is simple but limited; Zapier is powerful but complex. No platform bridges both. FlowSync targets this gap.
[4]	Smith – Low-Code Platforms for Workflow Automation	2020 | Benchmarking	Low-code tools deploy 40% faster but cap customisation. FlowSync uses a low-code interface with extensible JSON workflow schema for power users.
[5]	Davis – Integration Challenges in Automation Systems	2023 | Case studies	Inconsistent APIs and auth conflicts are the top integration blockers. FlowSync standardises OAuth 2.0 across all connectors.
[6]	Brown – Zustand vs Redux in Automation Apps	2023 | Performance eval	Zustand is better for mid-scale reactive apps. FlowSync's editor state uses a lightweight React context provider aligned with this finding.
[7]	Wilson – AI in SaaS Workflow Automation	2022 | AI case study	10% accuracy gain in task prioritisation; data privacy concerns remain. AI is planned as a future enhancement in FlowSync only.
[8]	White – Comparative Study: Zapier, Make, Power Automate	2021 | ROI analysis	Intelligent systems show better ROI; niche integrations underserved. FlowSync adds Discord and Notion, absent in most compared platforms.
[9]	Robinson – Workflow Automation and the Role of AI	2023 | AI decision nodes	30% cost reduction via AI routing, but accuracy depends on training data. FlowSync uses deterministic rule-based event processing currently.
[10]	Harris – AI-Driven Process Automation	2021 | AI monitoring	20% error reduction and 15% efficiency gain. FlowSync achieves comparable monitoring through webhook-based event detection without AI overhead.

3. System Architecture

A. Overall Design

Flow Sync is built on a typical three-tier web application: a Next.js application frontend, a Node.js/Express.js application backend, and a PostgreSQL database operated by Prisma ORM. The three

levels interact via RESTful APIs. Another fourth layer, the webhook ingestion engine, co-locates with the backend and receives real-time events published by integrated third-party services.

The basic operating cycle is simple. In the visual editor, a user creates a workflow and saves it. Upon the trigger event in a related service, the service will post an HTTP POST to the webhook endpoint of Flow Sync. The backend checks the payload, gets the corresponding workflow in the database, determines the action to be performed, calls the API of the target service and records the result. The result appears in the execution log of the user without their having communicated with either service byhand.

Fig. 1: FlowSync system flow from user onboarding to workflow execution

B. Frontend

React and Next.js are used to construct the frontend. Next.js is a server-side renderer on public pages, ensuring its initial loads are minimized, and its file-based router manages navigation between the Dashboard, Workflow Editor, Connections, Settings, and Billing pages without complete full-page loads. All of the styling is done using utility classes in tailwind CSS and the light/dark mode toggle is implemented using CSS variable theming.

The most complicated frontend component is the workflow editor. It is built with React Flow to display a node-based canvas allowing users to drag trigger and action nodes onto the canvas and connect them with directed edges. Every node opens up to a configuration panel that exposes the parameters of the service that a Slack node indicates channel selection; a Notion node indicates database field mappings.

C. Backend and Webhook Engine

Node.js is used on the backend to allow non-blocking I/O operations to process multiple concurrent webhooks and API calls in parallel without blocking threads. Routes are grouped into modules based on the context in which they're used - user management, workflow operations, integration token handling, webhook ingestion, and notification dispatch. Custom middleware layers provide CORS policy, JWT validation via Clerk, rate limiting, and verification of the signature of each webhook.

The webhook engine is the core of all operations. When a valid payload is received by the webhook engine, it: (1) Verifies HMAC-SHA256 signature against the stored webhook secret; (2) Identifies the event type and source service; (3) Retrieves all active workflows for which the trigger matches the event; (4) Resolves the action configuration from the stored JSON graph from each workflow; (5) Calls the

appropriate service SDK or API; (6) Creates a record in the execution log of what happened with the timestamp. The engine processes each action node in the graph in sequence for workflows with multiple actions.

D. Database

The PostgreSQL database contains four main entity types: Integrations, Execution Logs, Users and Workflows. Workflows are stored as serialized JSON node graphs, along with their associated metadata and a current status. Records of integrated services will include encrypted OAuth tokens, as well as an expiration timestamp for each token per user. Execution logs will record the type of triggering event, the Action that was performed, and the API response code for each workflow run, as well as the time at which the execution of the workflow occurred. Furthermore, developers will be able to utilize Prisma ORM's ability to make type-safe queries and to perform versioned schema migrations to provide guarantees of data integrity between the different new development and production environments where a given piece of data may reside.

E. Authentication and Integration Security

User authentication is managed utilizing Clerk, which has built-in features for both sign-in/sign-up processes by supporting both email/passwords as well as Google OAuth. Upon successful user authentication, Clerk generates signed JWTs for that user; every secured API end-point will also validate against the JWT using Clerk's Express middleware prior to executing the request on that end-point.

Connections to any third-party services are established via the standard OAuth 2.0 authorisation code flow. When the user consents to allow access from within the third-party service on their authorisation page, the callback route will be responsible for exchanging the code returned for the access/refresh tokens. The access/refresh token pairs returned from the service will be stored using AES encryption and are stored in the Integrations table. The refresh token for each connection will be refreshed automatically just prior to any API calls requiring the use of the refresh token. Each API call will also check to ensure that the HMAC signature included as part of the webhook payload has been generated using the stored webhook secret. If the HMAC signature does not match what was generated using the stored webhook secret, the request will be rejected.

4. Implementation

A. Technology Stack

The technologies used to create this implementation were chosen for their specific purposes as follows:

1) Next.js (React): A component-based frontend framework that supports server-side rendering (SSR) for fast initial page loads

2) Tailwind CSS: Utility-first CSS styling framework that provides built-in responsive breakpoints and theme switching

3) NodeJS + ExpressJS: A non-blocking, event-driven server architecture that is highly efficient and well-suited to the concurrent processing of webhooks and API calls.

4) PostgreSQL + Prisma ORM: Relational database that has the ability to run type-safe queries and perform schema migrations.

5) Clerk: A service that provides managed authentication using OAuth 2.0 (via Google login) and two-factor authentication.

6) Ngrok: Service that allows for secure local tunnelling of webhooks to test against live third-party services during development.
7) Vercel + GCP Cloud Run: Frontend is hosted on the Vercel CDN; Backend is a containerized application on Google Cloud Run with autoscaling.
8) Stripe: Service that is used to process credit card payments and provide webhook notifications for subscription events and payment events.

B. API Integrations

The table below shows the connection guidelines for the selected services. Each of our four connected services has its own integration module as follows:
- Google Drive uses the Drive API (v3) and has set up a push notification service for changes via the change's endpoint. This allows us to receive real-time notifications when you create or modify files in Google Drive.
- Slack uses the Slack Web API and bot token authentication to allow posting messages to specific channels.
- Notion uses the Notion API and workspace-level OAuth to create and update entries in a Notion database.
- Discord uses OAuth 2.0 with webhook-based message delivery to allow public access to channels on a Discord server.

When setting up any of these integrations in the Flow Sync User Interface, users will follow the same flow; they must first click 'Connect' on the Connections page, then complete the necessary authorisation on the Service's screen before returning to Flow Sync with an active integration record. Then, on the Connections page, the user can see their current status, token expiration date, and an option to disconnect from each service.

C. Workflow Builder

You will create your workflow using the Workflows page by entering your Workflow Name and Trigger Type into a dialog box; once these have been entered, you have created your Canvas with 1 existing trigger node. From here, you can add action nodes from the left side panel into the Canvas by dragging them and connecting them with a line to indicate the workflow path between nodes. When you connect two nodes, the workflow editor will validate the connection on an ongoing basis; therefore, it will not allow you to connect two triggers or connect action nodes with incompatible data types. Once you have made a connection between nodes, you will be able to see the configuration panel on each node that requires the appropriate information needed to carry out the actions associated with that node based on its service provider.

The Canvas will automatically save the draft to the server every 30 seconds; however, when you click on the Save button, then the position of each node and line (edge) connection and all parameter values will be converted into a string in JSON (JavaScript Object Notation) format and written to the Workflows table in the server database using an HTTP PUT request to the backend application.

D. Billing and Credit System

FlowSync functions with a pricing plan of credits using a supported billing system with Stripe as the product provider. Each workflow is executed by deducting a credit from the users account upon execution of the workflow. The React context for the Billing Provider will check to ensure there are enough credits before the workflow can be executed and will prevent the workflow from being executed if there are no

credits available. Users may purchase credits directly through the Stripe-hosted checkout pages or upgrade their Billing tier through those same pages. Payments are confirmed with webhooks from Stripe to the backend systems, which will then update the user's balance in the database.

5. Result and Testing

A. Test Scenarios

All of the end-to-end automation scenarios below test were completed successfully.

1. From Google Drive to Slack — A document was uploaded into a monitored folder on Google Drive, which triggered a formatted message to appear in a Slack channel about the document's existence within a time frame of 2 to 4 seconds from the time of upload.
2. From Google Drive to Notion — When a new document is added to Google Drive, it automatically creates a database record in Notion that includes the name of the file, type of file, and date/time of the upload.
3. From Slack to Discord — A new message sent to a monitored Slack channel is "mirrored" to a channel configured in Discord.
4. Multiple actions occurring from one action — A new document uploaded to Google Drive can cause sequential Slack notice and Notion creation to occur in the configured order.

Webhook latency from the time an API call is made until action takes place ranged from 1.5–4 seconds across all tests completed. Therefore, a majority of the total latency was due to external API response times from any involved third-party services. Completion time for database queries was measured to be approximately 20–80 ms during normal query load.

B. Interface Screens

The homepage shows what the platform is worth and how to sign up easily. Once you log in, the Dashboard shows how many workflows are running, the most recent ones, and the health of the integration. The canvas from Section IV-C is shown in the Workflow Editor.

C. Limitations

During testing, three limitations were found. First, Google Drive push notification channels stop working after seven days and need to be manually re-authorized. However, automated channel renewal is in the works. Second, the Notion API has rate limits (3 requests per second per integration), which means that when multiple workflows target the same workspace at the same time, there are short delays. Third, the visual editor doesn't yet let you use conditional branching in a workflow; all paths run one after the other. These are written down on the product roadmap and don't change the main automation features.

6. Conclusion and Future work

FlowSync shows that it's possible to automate workflows across Google Drive, Slack, Notion, and Discord using standard web technologies and a clean integration architecture. The event engine that runs on webhooks always finds trigger events and carries out the actions that are set up with low latency. The drag-and-drop visual editor makes this feature easy for people who don't know how to program to use. Testing in a variety of multi-service scenarios confirmed that the system consistently runs from start to finish, renders quickly on different devices, and performs well even when multiple workflows are running at the same time.

The implementation purposefully eschews artificial intelligence. Rule-based event processing meets all of today's automation needs while still being completely predictable and auditable. These are important features when a system acts on a user's behalf.

Acknowledgments
The authors would like to thank Mr. Jatin, an Associate Professor in the Department of Computer Science and Engineering – AI & DS at the Panipat Institute of Engineering and Technology, for his constant help with this project.

References:

[1] D. Dalal and B. V. Galbraith, "Evaluating sequence-to-sequence learning models for if-then program synthesis," arXiv:2002.03485, 2020.

[2] Y. Chen et al., "Data privacy in trigger-action systems," in Proc. IEEE Symposium on Security and Privacy, 2021.

[3] A. Rahmati et al., "IFTTT vs. Zapier: A comparative study of trigger-action programming frameworks," arXiv:1709.02788, 2017.

[4] J. Smith, "Leveraging low-code platforms for workflow automation: A comparative study," Journal of Software Engineering, vol. 12, no. 3, pp. 45–60, 2020.

[5] L. Davis, "Integration challenges in modern workflow automation systems," Journal of Business Processes, vol. 14, no. 1, pp. 33–47, 2023.

[6] A. Brown, "State management in workflow automation: An evaluation of Zustand vs Redux," Proc. Software Development Conference, vol. 8, no. 4, pp. 22–30, 2023.

[7] P. Wilson, "Enhancing SaaS workflow automation with AI: A case study on improving productivity," Journal of Cloud Computing, vol. 19, no. 5, pp. 88–95, 2022.

[8] K. White, "A comparative study of workflow automation systems: Challenges and opportunities," Journal of Information Systems, vol. 20, no. 3, pp. 120–135, 2021.

[9] J. Robinson, "Workflow automation and the role of AI in streamlining processes," Intl. Journal of Business Process Management, vol. 11, no. 2, pp. 50–68, 2023.

[10] M. Harris, "AI-driven process automation: Enhancing collaboration and reducing errors," Journal of Management Information Systems, vol. 25, no. 4, pp. 78–92, 2021.