TPM Vol. 32, No. S8, 2025
ISSN: 1972-6325
https://www.tpmap.org/

Open Access

OPTIMIZING THE AUTOMATION PROCESS WITH N8N

ISARA NAKAVISUTE[1*], TITAPORN SINCHAROONSAK[2]

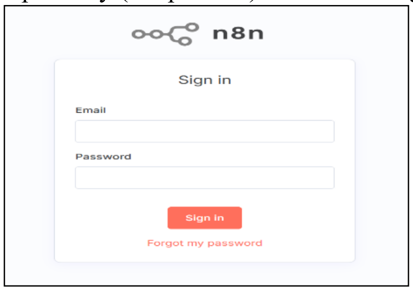
[1]DEPARTMENT OF COMPUTER SCIENCE, RAJAMANGALA UNIVERSITY, BANGKOK, THAILAND
[2]DEPARMENT OF COMPUER SCIENCE, SRIPATUM UNIVERSITY, BANGKOK, THAILAND EMAILS;
[1*]isara.nak@rmutr.ac.th, [2]titaporn.si@spu.ac.th

Abstract - Automation Is The Process Of Converting Work Process, Procedure And Workflow Into An Automatic Process Where Human Intervention Is Kept To A Minimum Level. There Are Many Automation Tools In The Market Today. For Example - There Are Zapier, Make.Com, N8N And Many Other Automation Tools. This Paper Shall Explain Deeply Into The Methodology Of Automation Tools And Extend The Explanation To N8N Automation. It Will Discuss The Installation Of N8N Through Docker Containers And Will Provide A General Layout For Docker Service And Installation. It Will Then Explain How N8N Can Help With The Automation Process And It Can Reduce The Human Intervention Within The Organization. It Will Explain The Relevant Statistical Research Data In Comparing The Organizations With N8N And The Organizations Without It. Automation Does Not Simply Transfer Human Functions To Machines, But Involves A Deep Reorganization Of The Work Process, During Which Both The Human And The Machine Functions Are Redefined.

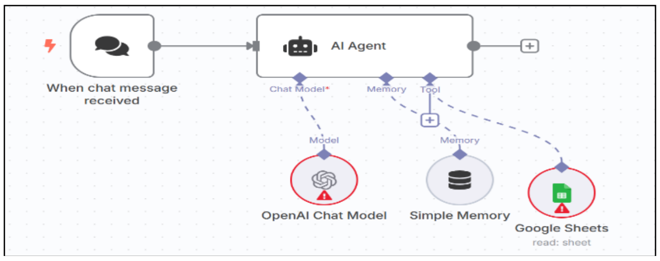
Keywords: automation, work process, procedure workflow, human intervention, N8N, Zapier

INTRODUCTION

Automation is a basis of today's software deployment, especially in the working area of DevOps where software development and software operation have converged to produce the concentrated and focused software delivery pipelines. Software deployment explains how to deliver and install the software application or the software update into a target environment, making it available for use by end users (Yah 420). The environment can be development or production. This flow can include the tasks such as setting up the server, copying important files, setting up databases, running scripts and performing validating tests to ensure the software functions correctly. Deployment can be either manual or automated with the latest practices often involving the good usage of fully pre-written Continuous Integration/Continuous Delivery pipelines to release updates quickly, reliably and repeatedly (Scoper 255). The following illustrates the n8n screen.

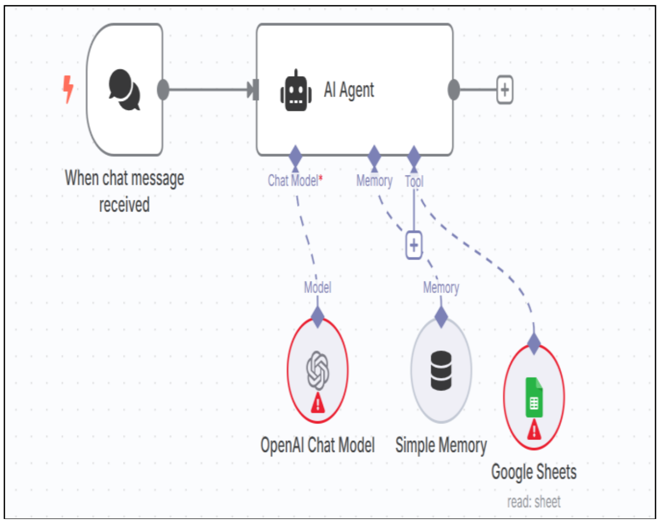


The effective deployment process ensures that software is delivered efficiently by using the very minimal downtime and errors. This is to maintain a consistent user experience across production environment. It focuses the deployment process by scheduling many tasks such as code compilation, testing, configuration, installation, reducing the risk of human error, fixing the error and eventually deploying. Continuous Integration/Continuous Delivery enables teams to deliver new features to provide bug fixes and to update consistently and frequently in order to ensure that applications remain up-to-date for maintaining the high-quality software with minimal downtime (Doyce 12). The following illustrates n8n workflow diagram.



DevOps is a set of routines that unite software development to deliver applications faster and more reliably. Each run will produce a build and validation process to detect errors to ensure that the codebase remains stable and can run without any errors (Torter 141). This will accelerate the development cycle. If the version control and

automated testing are combined, Continuous Integration does form a backbone for faster, more dependable software delivery and supports the great communication among the full development teams (Selkin 40).
Continuous Delivery (CD) is a software development practice that builds upon Continuous Integration. After code passes automated tests and quality checks, it is packaged and staged so that it can be deployed to production at any time with minimal manual effort (Sanning 21). Continuous Delivery ensures faster, safer, and more predictable releases, reduces deployment risk and allows teams to respond quickly to user feedback or business needs. At the same time, it will maintain the high software quality (Toallier 12). Using tools and utilities like infrastructure as code, automated testing and real-time monitoring, DevOps does break down traditional silos between developers and people who handle deployment (Selzer 21).
Infrastructure as Code (IaC) is a practice in which servers, networks and storage are provisioned and then managed by using the readable configuration files or scripts and not by using manual processes. Infrastructure as Code allows developers and operations teams to define the desired state of their infrastructure in code, enabling automation, version control and reproducibility. This illustrates the AI Agent implementation of n8n.



This approach reduces human errors and it ensures consistency across environments (Dox 278). This will make it easier to scale up the infrastructure. Tools like Terraform, Ansible and CloudFormation are commonly used to implement Infrastructure as a service (Selzer 78). It supports efficient and reliable deployment in cloud and on-premises environments. The result is quicker releases and higher-quality software. It provides the ability to respond rapidly to changes or issues in production environments. It is the final stage in the software development lifecycle. It follows development, testing, staging environments and is designed to run reliably and publicly. The production environment handles actual user traffic, processes real data. It has to maintain high availability, performance and security. If there are changes to be made to this environment, they must be carefully managed. This will fully be through automated deployment pipelines and monitoring tools to minimize downtime and ensure a stable, seamless user experience (Oskan 11). This will ensure that the new code changes are properly tested, built and deployed correctly (Salton 11). A number of cloud-based tools have emerged to meet this demand. The examples are Github Actions, Gitlab CI/CD, Jenkins and CircleCI.
GitHub is a cloud-based platform for hosting (DcTill 101). It manages code which uses the Git version control system to track changes. It allows programmers to share the changes on software projects by sharing repositories. Programmers can review code through pull requests and can manage issues or feature requests (Trady 21). This makes it a central hub for open-source projects and private enterprise development. On the other hand, GitLab is an all-in-one DevOps platform that provides Git-based source code management. It does support the built-in commands for Continuous Integration / Continuous Deployment (CI/CD) (Terbslep 347). It also enables teams to share code through repositories, merge requests and issue tracking. It will also support automating testing, building and deployment within a single application. It is also available as both a cloud service and a self-hosted solution. GitLab is very popular for its complete feature set and flexibility. This makes it suitable for open-source projects, enterprises. It will also enable organizations which seek an end-to-end software development and delivery workflow. Jenkins helps teams automate tasks like building, testing and deploying applications. It will also ensure code changes are correctly integrated and released quickly and reliably. It is also very flexible and provide support for both simple and complex workflows. This makes Jenkins a centralized tool for streamlining the software delivery process.
CircleCI is actually a cloud-based Continuous Integration and Continuous Delivery (CI/CD) platform. It automates the process of building, testing and deploying software. It does integrate with version control systems like GitHub, Gitlab and Bitbucket to run automated workflows when code changes are pushed (Wilkon 330). This is to ensure that bugs are caught early and releases are consistent. It also allows teams to scale up appropriately and release software faster. It will also maintain high quality and reliability. These platforms absorb much of the complexity involved in building automation pipelines. It will also offer integration, fully pre-configured production environments and fully scalable execution infrastructure. There are some concerns around data

privacy, cost at scale. It has the inability to fully customized execution environments which often arise in enterprise (Sarley 113). It will also arise in compliance-heavy context. For example, companies which operate in industry sectors such as finance, healthcare or defense may require more thorough control over their data than public platforms. Typically cloud-based CI/CD software impose limits on usage. It will also restrict the functionality behind the price plans. This will create scalability issues for growing teams. It will also create the problems with resource intensive projects. These challenges cause the need for lightweight, self-managed alternatives that allow the development teams to maintain full ownership of their automation stack. n8n is an emerging automation software designed to run the pipelines and orchestrate workflows in environments which are self-hosted.

A self-hosted environment is defined as an infrastructure environment where an organization deploys, manages and maintains its software or services on its own servers or data centers. It does not rely on cloud providers. n8n has a full control over hardware, security, configurations and data privacy. This allows for customized setups to meet specific business or compliance requirements. A self-hosting does require dedicated resources for maintenance, updates, backups and scaling. It is commonly used for applications which require the very strict control over data, regulatory compliance or very specialized configurations that cloud environments may not fully support.

n8n supports Docker based deployment, restful API and GitOps-style integration. This makes it flexible and easy to adopt. It also reduces conflicts between dependencies and simplifies scaling and management. By actually adopting a container-first mindset, it fully encourages teams to use the container orchestration tools like Kubernetes. This will integrate seamlessly with CI/CD pipelines and build cloud-native applications that are more resilient, efficient and easy to deploy across diverse infrastructures. This is not like monolithic solutions such as Jenkins which require full extensive configuration and plugin management, n8n offers a completely simplified, modular approach that caters to smaller teams or developers seeking an efficient and minimal CI/CD engine.

This study seeks to evaluate the viability of using n8n as a self-hosted automation solution for the real world. A self-hosted automation solution is a hosting system where an organization installs and manages automation software on its own servers or infrastructure rather than using a cloud-based service. This setup allows the complete control over data, security, configuration and integrations. This makes it ideal for sensitive or compliance-heavy environments. This improves efficiency. At the same time, it will fully keep resources and execution fully within the organization’s control. It completely requires the dedicated maintenance, updates and monitoring to ensure reliability and scalability. Specifically, the research addresses the question of whether or not self-hosting n8n can fully provide an effective, secure and resource-efficient as a good alternative to cloud-based CI/CD platforms? The research hypothesizes that deploying n8n in a controlled server environment enables faster task execution, reduced latency and increased configurability. At the same time it will maintain reasonable system resource usage. To investigate this point, the research sets up n8n on virtual server using Docker. It then configures it with NGINX and PostgreSQL. It then executed representative build-test-deploy workflows. The results of this research is to guide developers and researchers in designing CI/CD solutions that align with security, performance.

II. METHODOLOGY

n8n automation tool was downloaded from its public Github repository(https://github.com/n8n/n8n). n8n has been an workflow automation tool which users can connect different applications and services in order to automate tasks without extensive coding. It does allow the creation of complex workflows using a web-based visual interface. The actions can be triggers, actions and conditional logic in order to move data, publish data, synchronize google documents or perform repetitive processes iteratively. n8n does support cloud-hosted and self-hosted deployment in order to give users flexibility and to have the full control of their data. Its integration capabilities, ability for extension and user-friendly user interface make it popular for streamlining business processes, improving efficiency, and reducing manual work. The tool is distributed under MIT License and publicly available.

A virtual private server (VPS) with Ubuntu 24.04 LTS was provisioned from AWS T2 Micro equipped with 2 vCPUs, 4GB RAM and 30 GB SSD storage. A Virtual Private Server is a VM that mimics a dedicated physical server by using a publicly shared hosting environment. It provides users with dedicated resources such as CPU, RAM and storage. If it runs on a larger physical server, it will offer greater control, flexibility and isolation. VPS does allow users to install any legal software because it is a blank server. It can configure the operating system and manage security settings. It will be making it suitable for hosting websites, applications, databases and development environments.

Ubuntu is a distribution of Linux operating system and it is based on Debian. In fact, Debian is known for its user-friendliness, full stability and strong community support. It provides a complete operating system with a wide range of pre-installed software. This includes development tools, productivity applications and system utilities. Ubuntu is the operating system which is widely used for desktops, servers and cloud environments due to its ease of installation, regular updates, and robust security features. It is compatible with a vast collection of software and support for containerization, cloud computing and DevOps tools. This makes it a preferred choice for both beginners and enterprise users.

Amazon Web Services is actually a comprehensive cloud computing platform that provides on-demand services such as computing power, storage, databases, networking, analytics, machine learning, and more. Organizations can quickly scale applications and infrastructure using AWS’s pay-as-you-go model in order to reduce upfront

costs and increasing flexibility. AWS's pay-as-you-go model is a pricing approach where users only pay for the cloud resources and services they actually consume. This reduces upfront costs. It does not have long-term commitments either. AWS covers computing power, storage, databases, networking and other services
AWS helps businesses reduce wasted resources, control costs and maintain financial flexibility. In the meantime, it takes the full advantage of scalable and on-demand cloud infrastructure. AWS offers EC2 and object storage S3 to enable businesses of all sizes to build, deploy, and manage applications efficiently in the cloud.
Amazon Elastic Compute Cloud is a core AWS service that provides resizable virtual server. This is called an instance. It allows users to quickly configurate the operating system type, the security protocol in order to quickly launch, configure, and scale computing capacity without investing in physical hardware. EC2 supports a variety of operating systems, instance types. This flexibility allows developers and businesses to deploy the product from small websites to large-scale enterprise applications and pay only paying for the compute resources they use.
Docker is an open-source platform that fully simplifies building, packaging, and running applications by using lightweight, portable containers. These containers bundle an application with all its dependencies such as libraries and configuration files. It runs consistently across different environments whether on a developer's laptop, a test server or in the cloud. Docker improves efficiency by isolating applications, reducing conflicts between software versions and enabling rapid deployment and scaling. Its environment includes Docker Engine for running containers, Docker Hub for sharing images and tools that integrate with full CI/CD pipelines and full orchestration platforms like Kubernetes.
Docker Hub is a cloud-based repository service for storing, sharing and managing Docker container images. Docker Hub is authenticated by username and password. It allows developers to push their container images and pull images which are created by others. Docker Hub simplifies the resource sharing by enabling the entire team to distribute standardized containers across development, testing and production and development environment to ensure the consistency and portability. Docker Hub also supports automated builds, webhooks and access control in order to make it a full central hub for containerized application workflows.
Kubernetes is an open-source platform designed to automate the deployment, scaling and management of containerized applications. It groups containers into logical units. This is called Pods and it makes it easier to distribute workloads across a cluster of servers for high availability and efficient resource use. In Kubernetes, a pod is the smallest deployable unit that represents a single instance of a running application. A pod can contain one or more tightly coupled containers that share the same network namespace, storage volumes and configuration. Pods are temporary by nature. They can be created, scaled, and destroyed dynamically based on the application's needs.
Kubernetes manages pods through controllers like Deployments and self-healing. This makes them the fully fundamental building blocks of containerized applications in a full Kubernetes cluster. Kubernetes handles tasks such as load balancing, self-healing, rolling updates and service discovery to allow developers to have a focus on application code rather than infrastructure. Widely adopted in cloud and on-premises environments, Kubernetes is a key tool for managing complex, large-scale container-based systems.
PostgreSQL was used to store the persistent data storage, installed as Docker container from the official PostgreSQL image. NGINX was deployed as a fully reverse proxy with SSL secured through Let's Encrypt (https://letsencrypt.org).
Let's Encrypt is a free, automated, and open certificate authority (CA) that provides SSL/TLS certificates to enable secure HTTPS connections for websites. It does simplify the process of obtaining, installing and renewing all certificates through automation. This removes the need for manual configuration and reducing barriers to web security. By offering trusted certificates at no cost, Let's Encrypt promotes widespread encryption across the internet and helps the website owners protect data, ensure privacy and gain user trust. In addition, it supports modern security standards.
NGINX is a high-performance web server and reverse proxy server that is widely used to serve web content, distribute traffic and improve application scalability. A web server is a software or hardware system that delivers web content. The examples are HTML pages, images, videos. This is shown to users over the internet through HTTP, HTTPS protocols. It processes incoming requests from clients. This is through typically web browser. It retrieves the requested resources and sends them back as responses. This enables users to access websites and applications. Web servers can also handle additional functions such as load balancing, caching, logging and security through SSL/TLS encryption. Popular web server software includes Apache, NGINX and Microsoft IIS. These are widely used to do the hosting of websites and web applications efficiently and reliably. It efficiently handles simultaneous connections by using an event-driven architecture. This makes it ideal for high-traffic websites and APIs.
Event-driven architecture is a software design approach in which systems communicate and react to events. It senses the changes in state or significant occurrences, instead of relying on direct request-response interactions. In this architecture, components produce, detect and respond to events asynchronously. This enables real-time processing, loose coupling and scalability. Event-Driven Architecture is commonly used in applications that require responsiveness and flexibility. The examples are IoT platforms, financial systems and microservices to allow systems to efficiently handle high volumes of data and dynamically trigger workflows or actions based on events as they occur. Beyond serving static content, NGINX can act as a load balancer, SSL/TLS terminator and HTTP cache to help with optimizing performance and reliability. Its flexibility, speed and lightweight design have made it a popular choice for both small projects and large enterprise applications. Secure Sockets Layer is a cryptographic protocol that secures communication between a web browser and a server by encrypting data

transmitted over the internet. It ensures that sensitive information such as passwords, credit card numbers and personal data cannot be intercepted or tampered with by attackers. Websites that use SSL display "HTTPS" in the location bar. By providing authentication, data integrity, encryption and SSL are a fundamental technology for protecting online transactions and building trust with users. HyperText Transfer Protocol Secure is an extension of HTTP that uses SSL/TLS encryption to secure communication between a web browser and a server. It ensures that data exchange such as passwords, personal information or payment detail is encrypted. This protects it from eavesdropping and tampering. HTTPS also provides authentication, verifying that users are connecting to the intended website. This helps prevent man-in-the-middle attacks. Websites using HTTPS display a lock icon in the browser's location bar. This is to to signal to users that the internet connection is secure and trustworthy.
Transport Layer Security is a cryptographic protocol that ensures secure communication over computer networks by encrypting data transmitted between clients and servers. It does provide confidentiality, integrity and authentication to protect sensitive information such as passwords, payment detail and personal data from eavesdropping or tampering. Transport Layer Security is widely used in web browsing, email, instant messaging and other online services. The above components form the good basis for HTTPS connections. By verifying the identity of servers and clients, TLS helps establish trust when it maintains the privacy and security of data in transit at the same time. Github was used as the version control system. Webhook integration was manually configured to trigger workflows automatically.
Version control is a system that records changes to files. This is typically the source code. Over time developers can track, manage and revert to previous versions when it is needed. It enables multiple team members to share resources on a project simultaneously. They can merge changes to avoiding conflicts. Version control systems maintain a complete history of edits. It supports branching and merging for experimentation and provide accountability through tracking people who made each change. This makes development more organized, reduces errors and it ensures that projects can evolve safely and efficiently. All components have been installed and configured manually through shell scripting to ensure replicability.
Shell scripting is the practice of writing scripts like text files which contain a sequence of commands in order to automate tasks in a command-line environment. This typically use a Unix/Linux shell like Bash. These scripts can fully perform repetitive operations such as file manipulation, program execution, system monitoring and software installation to save time and reducing human error. Shell scripting allows for conditional logic, loops and functions to enable complex workflows to be executed automatically. It is widely used by system administrators, developers and DevOps engineers in order to streamline routine tasks and manage systems efficiently. The n8n container was launched using Docker Compose with environment variables defined in an external .env file. An .env file is a plain text file used to store environment variables for an application such as configuration settings, API keys, database credentials or secret tokens. By keeping these values outside the source code, .env files help improve security and simplify global configuration across different environments and make applications more portable. Programs can read the .env file at runtime to access the necessary variables, allowing developers to change settings without modifying the code. This practice is common in modern development workflows especially in the fully functioning web applications and containerized environments. A volume was mounted to persist logs and configurations.
PostgreSQL was linked to the n8n service to manage job states and workflow metadata. NGINX was configured to route external HTTPs traffic to the internal n8n service port, using TLS certificates issued by Certbot.
Certbot is actually an open-source tool developed by the Electronic Frontier Foundation (EFF) that automates the process of obtaining, installing and renewing SSL/TLS certificates from Let's Encrypt. It simplifies securing websites with HTTPS by automatically configuring web servers, validating domain ownership and managing certificate renewals. This will be without manual intervention or human intevention. Certbot supports a wide range of server software including Apache and NGINX. This makes it accessible for both beginners and experienced administrators. By streamlining certificate management,
Certbot helps website owners maintain secure, encrypted connections and ensures ongoing compliance with modern web security standards. Transport Layer Security certificate is a digital certificate that authenticates the identity of a website or server and enables encrypted communication between clients and servers. A digital certificate is an electronic credential issued by a trusted Certificate Authority that verifies the identity of an individual, organization or website. This enables secure communication over the internet. It contains information such as the certificate holder's public key, the issuer's details and a validity period. It is used to establish encrypted connections through protocols like TLS/SSL. Digital certificates provide authentication, data integrity and encryption. This ensues that users can trust the identity of the server or entity which they are communicating with. The transmitted data remains confidential and unaltered during the entire transit process. It ensures that data transmitted over the internet such as passwords, payment details or personal information is secure from eavesdropping or tampering. Transport Layer Security certificates are issued by trusted Certificate Authorities. They are a key component of HTTPS and provide both encryption and trust signals. For example, there will be the padlock icon in browsers in order to assure users that they are connecting to a legitimate and secure website. The server firewall was managed through UFW to restrict access to only necessary ports. Examples are port 80 and port 443. Secure Shell is a cryptographic network protocol used to securely access and manage remote computers over an unsecured network. A server firewall is a security system that monitors and controls incoming and outgoing network traffic to protect a server from unauthorized access, attacks and malicious activity. It acts as a barrier between the server and external networks. It allowing only traffic that meets defined security rules while blocking potentially harmful connections. Firewalls can be hardware-based, software-based or a

combination of both. This often includes features such as packet filtering, intrusion detection and logging. By regulating access and filtering threats, a server firewall helps maintain the integrity, confidentiality and availability of the server and its hosted applications. It fully provides the encrypted communication and ensures that commands, files and credentials transmitted between a client and a server are protected from eavesdropping or tampering. Encrypted communication is the process of securing data transmitted between devices or systems by converting it into a coded format that can only be read by authorized parties with the correct decryption key. This ensures that sensitive information such as passwords, financial data or personal messages remains confidential. It will be protected from eavesdropping, tampering or interception during the entire transit process. Encrypted communication is widely used in technologies like HTTPS, email encryption, messaging apps and virtual private networks.

Secured Shell is commonly used by the system administrators and developers to log into remote servers and execute commands. It can also be used to transfer files securely and manage infrastructure efficiently. Its combination of security, authentication and flexibility makes it a useful fundamental tool for remote server management. Two representative software projects were created: one in Python (Flask-based web app) and one in Node.js (Express-based API). Python is a high-level, interpreted programming language known for its readability, simplicity and versatility. It supports multiple programming paradigms, including procedural, object-oriented and functional programming. This makes it suitable for a wide range of applications such as web development, data analysis, artificial intelligence, automation and scientific computing.

Python has a large standard library and a vast collection of third-party packages. This allows developers to quickly implement complex functionality. Its clear syntax and strong community support make it an ideal language.

Node.js is an open-source, cross-platform runtime environment that allows developers to run JavaScript on the server side. This is built on Chrome's V8 engine. It is designed for building scalable and high-performance applications particularly programs that handle many simultaneous connections such as web servers and real-time applications.

Chrome's V8 engine is an open-source JavaScript engine developed by Google that powers the Chrome browser and Node.js runtime by compiling JavaScript code directly into high-performance machine code. It uses advanced techniques such as just-in-time compilation, efficient memory management and garbage collection to execute JavaScript quickly and optimize performance for complex web applications. V8 enables fast execution of both client-side and server-side JavaScript. It fully supports modern ECMAScript standards and serves as the full backbone for the deployed JavaScript-based tools and frameworks. This makes it a key component of today's web and cloud application ecosystems. Node.js uses an event-driven, non-blocking I/O model. This makes it efficient and lightweight. A non-blocking I/O model is a programming approach where operations such as reading from or writing to a file, network or database do not pause the execution of a program. Instead, the program can continue executing other tasks and handle the I/O result asynchronously through callbacks, promises or event loops. This model improves efficiency and scalability especially in applications that handle many simultaneous connections, such as web servers and real-time systems. This is done by reducing idle time and making optimal use of system resources. With its rich ecosystem of packages available through Node Package Manager, Node.js enables rapid development of server-side applications using JavaScript. This allows developers to use a single language across both front-end and back-end development.

Node Package Manager is fully the default package manager for Node.js. This is used to install, manage and share reusable JavaScript libraries and tools. It allows developers to easily several incorporate third-party packages into their projects. It will manage dependencies and maintain version control for consistent builds. npm also provides a vast online registry where developers can publish and access millions of open-source packages. It will also facilitate rapid development and collaboration. With commands for installing, updating and removing packages, npm streamlines project setup, dependency management and workflow automation in Node.js applications. For each project, a CI/CD pipeline was defined using n8n's declarative YAML syntax. YAML is a human-readable data serialization format commonly used for CI/CD configuration files and data exchange between applications. Its syntax is designed to be simple and intuitive by using indentation to represent hierarchy instead of brackets or tags. This makes it easy to read and write.

YAML supports complex data structures such as lists, dictionaries and nested objects. This is widely used in DevOps, cloud configurations and tools like Docker Compose, Kubernetes and CI/CD pipelines. Its readability and flexibility make it a popular choice for managing application and infrastructure settings. The pipeline included build, test and deploy stages. Each step executed in an isolated Docker containers to ensure clean environments. Webhooks were configured to trigger the pipeline upon push events on the main branch. A webhook is a way for one application to send real-time data or notifications to another application over the web when a specific event occurs. A webhook allows a server to automatically "push" information such as a new user signup and payment confirmation or code commit to a predefined URL endpoint.

Webhooks are widely and fully used in APIs, DevOps workflows and integrations between services. This will enable automation, event-driven processes and seamless communication between many different systems without manual intervention. Logging was also enabled for all jobs and stored in mounted volumes for analysis. Each workflow was executed 20 times to account for variability in performance and to simulate real world usage. The start and end times of each stage were recorded, along with the status, resource consumption and any errors encountered. This research adopted a comparative benchmarking approach. The comparative benchmarking approach is a method of evaluating the performance, efficiency, or quality of a system, process or product by comparing it against established standards, competitors or best practices. Instead of measuring performance in isolation, this approach identifies gaps, strengths and areas for improvement by using reference points or industry benchmarks. Comparative benchmarking is widely used in business, software development and technology to guide decision-making, optimize processes, and drive continuous improvement by learning from the successes and shortcomings of others. Workflow latency, system load and recovery time were measured for n8n and compared against Github Actions, using identical repositories and scripts. The primary goal was to assess the effectiveness of n8n in terms of speed, reliability and resource usage in a self-hosted configuration. For Github Actions, the free tier was used with default runners located in the U.S. East region. Basic statistical analysis was applied to a compare workflow performance between n8n and Github Actions. Mean execution times, standard deviations and 95% confidence intervals were calculated. Paired t-tests were used to assess the significance of differences in task duration and resource usage across platforms. A paired t-test is a statistical method used to compare the means of two related groups to determine whether there is a significant difference between them. It is commonly applied when measurements are taken from the same subjects under two different conditions, such as before-and-after studies or matched samples. By analyzing the differences within each pair rather than the groups independently, the paired t-test accounts for individual variability and provides a more precise assessment of the effect. This test assumes that the differences are normally distributed and is widely used in fields like medicine, psychology and social sciences to evaluate interventions or treatments. Data was visualized using Python's matplotlib and pandas libraries to assist in interpretation. Pandas is a powerful open-source Python library designed for data manipulation and analysis. It provides flexible, high-performance data structures like Data Frames and Series, which allow users to easily handle, clean, and transform structured data such as spreadsheets, CSV files or SQL tables. Pandas offers a wide range of functions for filtering, grouping, aggregating and merging datasets as well as time-series analysis and handling of missing data. Its intuitive syntax and seamless integration with other Python libraries like NumPy and Matplotlib make Pandas an essential tool for data analysis, scientific computing and machine learning workflows. Matplotlib is a widely used Python library for creating static, interactive and animated visualizations of data. It provides a flexible framework for generating a wide range of plots including line charts, bar graphs, histograms, scatter plots and more. This is to allow users to customize colors, labels, scales and layouts. Matplotlib enables data scientists, analysts and developers to explore, analyze and present data visually. This makes complex datasets easier to understand and interpret. Its versatility and extensive functionality make it a foundational tool for data visualization in Python. All experiments were conducted within a controlled network environment and no back processes were allowed to interfere with server performance during test runs.

III. RESULT AND DISCUSSION

A total of 40 workflow executions were carried out for both the self-hosted n8n environment and the Github Actions platform. The research data has included the average execution latency, resource usage and recovery time in case that failure happens. Standard deviation (SD) and p-values from paired t-tests are used for the analysis. The research showed a lower average latency (M = 9.4 s, SD = 0.51 s) compared to Github Actions (M = 16.8 s, SD = 0.97 s). A paired t-test revealed that this difference was statistically significant. CPU and memory usage were tracked during all test runs using Docker analysis. CPU usage averaged 45% and memory usage averaged 545 MB (SD = 44 MB). No significant resource peaking were noticed. Recovery time was measured from the moment which the failure happens to the successful restart of a pipeline. The n8n environment showed a significantly faster recovery rate (M = 14.1 s, SD = 1.7 s) in comparison to GitHub Actions (M = 29.4 s, SD = 3.1 s). The difference in recovery times was also statistically important. This research analyzed the performance, performance efficiency and operational analysis of the n8n automation tool deployed in a production environment. The objective was to determine whether an independently managed instance of n8n could serve as a viable alternative to the globally branded cloud-based CI/CD services. The hypothesis assumed that self-hosting n8n would offer a better solution, reduced latency and more cost-reduction without losing reliability. Latency is the amount of time it takes for data to travel from its source to its destination in a network or system. It is typically measured in milliseconds. It reflects delays that can occur due to transmission, processing or queuing in networks, servers or applications. Low latency is crucial for real-time applications such as online gaming where small delays can affect performance and user experience. Factors influencing latency include network distance, bandwidth, hardware performance and the number of intermediary devices. The results presented in the previous section support this assumption. The first objective was to evaluate the execution performance of n8n compared to the branded Github Actions. The data showed that workflows executed in the self-hosted n8n environment consistently ran smoother and faster, with an average latency reduction of over 37%. This improvement can be attributed to reduced network overhead and full control over pipeline concurrency. These results align with prior studies suggesting that on-premise or local automation solutions can outperform cloud-based options in controlled environments. The second objective was to assess system resource utilization. The n8n deployment maintained stable CPU and memory usage during all workflow executions, averaging 45% CPU load and 545 MB RAM usage. This efficiency highlights the performance nature of n8n and its suitability for smaller VMs. In comparison, cloud-based systems abstract resource consumption, making them harder to predict and optimize. The predictability in n8n's performance adds value for development teams. The third objective was to measure reliability. Self-hosted n8n demonstrated faster failure recovery, taking less than half the time required by Github Actions. The ability to restart jobs locally without dependency or intervention. Despite its strengths, several limitations must be referenced. First, setting up and maintaining a self-hosted solution requires a moderate level of system administration privilege. Misconfiguration or weak security practices may expose the system to a very serious security incident. In addition, the research did not look into the distributed workload conditions. In comparison to existing literature, few open-source CI/CD systems are as modular and minimal as n8n. Tools like Jenkins or CircleCI introduce complexity due to plugin management and maintenance. Drone CI offers a similar containerized model. By providing research performance data, this research contributes practical insights into the operational significance of self-hosted automation.

IV. CONCLUSION

This research has looked into the self-hosting the n8n automation platform as a good solution for managing CI/CD pipelines. By deploying the tool on a virtual server and comparing it against the globally branded git repository such as GitHub Actions, the results revealed significant improvements in workflow latency, resource predictability. It also looks at the statistics on recovery performance. The experiments support the assumption that self-hosting n8n can serve as a cost-effective and controllable solution to the branded cloud native platforms. These are Github, Gitlab and other brands. This is looked especially in contexts where security and implementations are priorities. The major consideration points highlight n8n's system overhead, quick turn-around time and deterministic performance. While some operational work is still required for setup and configuration, the trade-off in control and transparency is considerable. In addition, the ability to fully define and monitor automation in separated environments makes it well-suited for research institutions, startups and regulated industries. This research also contributes to the area of DevOps automation by providing important benchmarks and deployment guidance for an open-source tool. It encourages further exploration into other globally branded CI/CD models and leads to the future innovation beyond the branded cloud service. It is important to note that n8n is the future technology which is going to make a strong impact on the technology sector. n8n is not the only sole product in the market. It should be noted that Zapier and make.com are also among the competitors of n8n.

V. REFERENCES

[1] I. L. Yah, Software Assurance, 105 (5) (2020) 415-423.
[2] L. Scoper, "Beyond the software quality," Inf. Process. Manag., vol. 20, no. 1, pp. 253-258, 2020.
[3] F. Torter, "A quality algorithm," Program, vol. 15, no. 1, pp. 140-145, 2021
[4] L. Doyce, Managing software services, Technical Papers, 2020.
[5] T.L. Sanning, Information Security, Online. Cambridge, England: Cambridge University Press, 2020
[6] S. Selkin, "Informaton Communication" Communication of the Information, vol.23, no. 10, pp. 39-48, Dec-2020.
[7] T. Selzer, Quality Improvement Process, Process: Improvement, 1 (2) (2020) 187-220.
[8] F. Toallier, ISO 9001 Implementation, IEEE Software, 11 (1) (2021) 198-200.
[9] T. Selzer, Software Quality Control, Software Quality Journal, 7 (1) (2020) 75-84.
[10] D. Dox, "Information retrieval" Process. Management., vol. 20, no. 1, pp. 277-287, 2021.
[11] G. Salton, Automatic text processing. Reading, Massachusetts: Addison-Wesley, 2020.
[12] D. Oskan, An Approach to Quality Software, Prentice Hall, New Jersey, USA, 2020.
[13] L. Trady, Process Improvement, Prentice Hall, 2021.
[14] T. DcTill, Introduction to modern information retrieval. New York: McGraw-Hill, 2023.
[15] D. Talton, "Vector space model for automatic processing," Common. Acm,no. 18, 2020.
[16] T. Terbsleb, ICSE 2020, pp. 345-355, 198. [15] P. 2 (3) (2020) 11-15.
[17] D. Wilkon, Process Improvement, Software Process: Improvement and Practice, 10 (1) (2024) 327-334.
[18] n8n GitHub Repository. (2024). n8n automation tool. GitHub. https://github.com/n8n/n8n
[19] J. Sarley, D. (2020). Software Automation: Reliable Software Assuance Automation. Addison-Wesley.
[20] Docker. (2024). Docker Engine and Docker Compose. https://docs.docker.com
[21] NGINX. (2023). NGINX reverse proxy configuration guide. https://nginx.org/en/docs/
[22] GitOps Working Group. (2022). GitOps principles and best practices (CNCF Whitepaper).Cloud Native Computing Foundation.