STM JOURNALS

Journal of
Software Engineering Tools & Technology Trends

ISSN: 2394-7292
Volume 12, Issue 3, 2025
September–December
DOI (Journal): 10.37591/JoSETTT

https://journals.stmjournals.com/josettt

Review JoSETTT

Text to Flowchart Convertor

Shagun Kumari[1,*], Pooja[2]

Abstract

With the advancement artificial intelligence tools and software, we can now transform process text into a proper formatted flowchart. This research work mainly focuses on how advancement of technologies has enabled to save time and energy by creating a structured understandable flowchart instead of merely presenting text, thereby converting process text into flowchart. Use of natural language processing (NLP) and libraries of JavaScript has led to easy transformation of text to flowcharts. Earlier it was not easy to convert text directly into flowcharts as it was difficult to convert source code of any programming language into a systematic flowchart. While, at present, with the help of NLP, it has become easier to extract relevant data from a process and align them into systematic order. Further, when the steps are generated into a form of a source code or a step-by-step algorithm with the help of mermaid.js libraries, we can convert the steps into a proper flowchart. Hence, allowing the user to make changes according to their needs and exporting or saving the generated output.

Keywords: Flowchart, NLP, AI tools, JavaScript, Mermaid.js, diagram generator

INTRODUCTION

Text-to-flowchart is an innovative AI-driven approach that focuses on automatically transforming unstructured textual information into a structured flowchart representation. In simple terms, the idea is to take raw text data, whether it is descriptive, instructional, or process-based, and convert it into a step-by-step diagram that clearly shows the logical flow of actions or decisions [1–3]. This technique provides a way to bridge the gap between written descriptions and visual understanding, as flowcharts are universally recognized tools for simplifying complex concepts. By processing the input text, extracting essential elements, and arranging them into a systematic order, the tool generates a clear, easy-to-follow flowchart that can be understood by individuals from both technical and non-technical backgrounds [3–5].

The underlying process relies heavily on the capabilities of natural language processing (NLP). NLP techniques allow the system to break down the given text into smaller, manageable components through tokenization, which involves identifying parts of speech, keywords, and important phrases. Once tokenization is complete, text parsing techniques are applied to interpret the structure of the sentences and establish the sequence of actions or events being described [6–8]. This enables the system to form a logical progression of steps, which can later be visualized as an algorithm representing the original text. After the sequence is created, the steps are translated into flowcharts using visualization libraries such as mermaid.js, which is particularly effective for generating flow diagrams in a web-based or digital environment.

***Author for Correspondence**
Shagun Kumari
E-mail: shagunsingh7522@gmail.com

[1]Student, Department of Computer Science and Engineering, Echelon Institute of Technology, Faridabad, Haryana, India
[2]Assistant Professor, Department of Computer Science and Engineering, Echelon Institute of Technology, Faridabad, Haryana, India

Received Date: June 20, 2025
Accepted Date: July 29, 2025
Published Date: September 15, 2025

Citation: Shagun Kumari, Pooja. Text to Flowchart Convertor. Journal of Software Engineering Tools & Technology Trends. 2025; 12(3): 18–25p.

The advantages of this approach are extensive. A large volume of text can often overwhelm readers, but presenting the same information as a flowchart helps in immediate comprehension [9–11]. Users are able to see the relationships, dependencies, and order of operations at a glance, making it much easier to analyze processes. For example, in software development, a text-to-flowchart tool can assist programmers and system designers in quickly modeling workflows or system logic from documentation. In project management, it can help teams visualize project stages and dependencies without manually designing charts. In the education sector, teachers can use it to simplify difficult concepts for students by converting lessons into process diagrams. Similarly, in business analysis, professionals can easily map out organizational processes, identify inefficiencies, and communicate ideas effectively to stakeholders [12–14].

In summary, the text-to-flowchart concept combines artificial intelligence with natural language processing to convert raw, often complex textual information into a visual format that is easier to interpret. By automating this transformation, the tool significantly reduces the time and effort required to create flowcharts manually, while at the same time enhancing clarity, productivity, and decision-making across various domains [15].

LITERATURE REVIEW

Some of related works on text to flowchart convertor are:

- *NoteGPT AI Diagram Generator:* Free browser-based application relying on AI technology to translate text into multiple different diagrams, like flowcharts, tree diagrams, network diagrams, and Venn diagrams. Paste text, and the application does the work to create and export diagrams, eliminating the need to manually visualize and organize complex data for educators, students, and professionals.
- *NoteGPT Main Platform:* A one-stop AI learning assistant tool that provides content summarization, idea visualization, script creation, note organization, among others. It is capable of batch video summarization, AI Q&A, and quick presentation making, making it highly productive for students, researchers, academics, and professionals.
- *NoteGPT Blog: Gamma AI Presentation:* A blog entry explaining how NoteGPT uses AI to transform text into professional presentation slides, just like Gamma AI Presentation. It showcases the platform's ability to automate the generation of well-structured and visually appealing presentations from raw text.
- *NoteGPT AI Notes Generator:* A utility to help users create well-organized notes from copied text with AI. It is meant for speedy and effective note-taking, helpful for students and professionals who require organizing information in a hurry.
- *Jeda.ai Multi-LLM Generative AI Flowcharts and Diagrams:* An expert visual AI system that employs a combination of multiple large language models (LLMs) to conceptualize and produce flowchart blocks and diagrams. Jeda.ai provides a multimodal workspace environment for real-time collaboration, configuration, and implementation of AI-facilitated diagramming as part of business intelligence and project planning processes.
- *Miro AI Flowchart Generator:* A co-editing whiteboarding tool with an AI-driven flowchart generator. Miro AI enables quick flowchart generation from text input in seconds to facilitate rapid kick-offs for projects, brainstorming, and team input. It also provides summarizing and cross-platform sharing features.
- *MyMap.AI Free AI Flowchart Maker:* Free AI tool for automatic creation of business flowcharts. MyMap.AI features versatile input (file or text upload), team collaboration, and simple export/sharing. It is an application for beginners with no designing capabilities and caters to several types of flowcharts and even intricate processes.
- *NoteGPT Workplace Beginner's Guide:* An instructional video detailing the use of NoteGPT's workplace functionalities, such as AI summarizing YouTube videos, translation, and summarization of PDFs. It explains the platform's emphasis on accelerating learning efficiency through AI-managed content management.

- *YouTube:* Create Flowcharts Quickly and Easily with AI: A video tutorial displaying the utilization of AI tools in creating flowcharts quickly and simply. It reveals useful features and workflows, rendering it perfect for professionals who are looking for effective and understandable diagram creation.

RESEARCH PROBLEM

While creation of this tool various challenges were faced such as text parsing of the right content and further identification of what text to be placed in which logical component of flowchart such as whether the generated text needs to be placed in process box, decision box, where the problem statement ends and much more. Maintenance of quality of diagram when lengthy textual inputs are provided. Transformation of textual data into a series of steps or either in form of code which is understood by mermaid.js so that it will convert data efficiently into flowchart.

NEED FOR RESEARCH

The creation of Flowcharts manually is often a slow process that requires one to possess graphic design expertise or at least know how to operate various features in software. Users of converters give a review of improvement in productivity, and time savings as text to flowchart converters replace text-based data with structured flowcharts. It provides a user-friendly interface as the one who needs to convert any textual data to flowchart need not know much about the software or any programming language. They just need to put textual data and this tool automatically generates data to flowchart by first text parsing using NLP and further generate steps to flowchart. It is still very difficult to manually generate a correct order sequence wise flow of steps into visuals but this system enables conversion of complex data easily into an ordered flowchart.

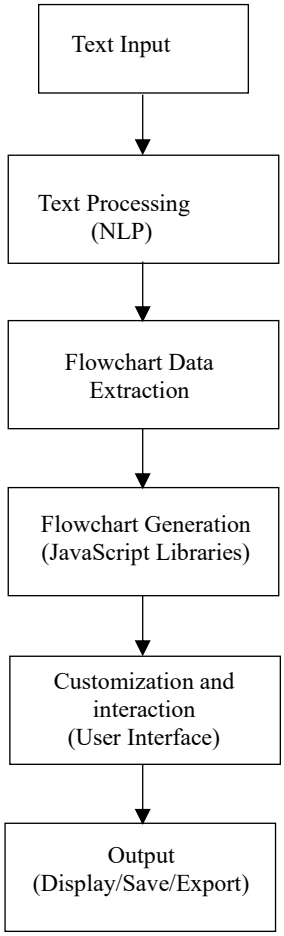

Figure 1. Generating flowchart structure.

PROPOSED METHODOLOGY

1. *Text Input:* In this step any process text or structured text is taken as input (Figure 1).
2. *Text parsing:* We apply NLP method in order to detect the main relevant content that needs to be generated in the form of steps, which even recognizes actions, decision, start and end point.
3. *Flowchart Data Extraction:* Each action or decision is assigned to a particular node type (process, decision, input/output, etc.). Sequence the nodes according to logic derived from the text.
4. *Flowchart Generation:* With the help of JavaScript libraries, flowchart is generated. JavaScript libraries such as mermaid.js helps in visual representation of steps generated with the help of natural language processing (NLP).
5. *Customization and interaction:* In this we can customize our flowchart accordingly as per our outcomes which enables enhancement of the flowchart as per user needs.
6. *Output:* This displays resultant flowchart which can be easily displayed, saved and exported.

Below is the pseudocode for conversion of text to flowchart:

```
function textToFlowchart (inputText):
 Step 1: Preprocessing
        cleanedText = preprocess(inputText)
        sentences = tokenize(cleanedText)

 Step 2: Parsing & Understanding Structure
        flowNodes = []
        for sentence in sentences:
        intent,entities=extractIntentsEntities(sentence)
        node =mapToFlowchartElement(intent,
        entities)
        flowNodes.append(node)

 Step 3: Generating Flowchart Structure
        flowStructure=buildFlowchartStructure
        (flowNodes)

 Step 4: Building Graph
        flowGraph = buildGraph(flowStructure)

 Step 5: Rendering Flowchart
        flowchartImage = renderFlowchart(flowGraph)

 Step 6: Output
        display(flowchartImage)
        return flowchartImage

# Helper Functions

function preprocess(text):
        Clean noise, correct casing, remove unwanted
        characters.
        return cleanedText

function tokenize(cleanedText):
        Split into sentences or logical process steps.
        return sentenceList
```



```
function extractIntentsEntities(sentence):
        NLP extraction: Find actions, decisions,
        branches, start/end points.
        return (intent, entities)

function mapToFlowchartElement(intent, entities):
        Map NLP findings to flowchart part (e.g.,
        Process, Decision, Start, End).
        return flowchartNode

function buildFlowchartStructure(nodes):
        Establish logical AND visual connections:
        what's next, what branches
        return flowchartStructure

function buildGraph(structure):
        Create graph data representing the flowchart.
        return graph

function renderFlowchart(graph):
       Visualize using a library (e.g., Graphviz,
        mermaid.js).
        return flowchartImage

function display(image):
        Show or export the image.
        pass
```

CASE STUDY

Designing a Text-to-Flowchart Converter for Novice Programmers

Learning programming is often described as learning how to "think like a computer". For many beginners, however, the most difficult part is not memorizing the syntax of a language, but rather understanding the logic behind solving a given problem. New programmers frequently struggle to convert a word problem or a description of a task into a structured solution. While experienced developers may already have a mental framework for designing algorithms, beginners are often left confused, unable to bridge the gap between problem statements and working code.

This is where flowcharts play a significant role. A flowchart provides a visual roadmap that outlines the logical steps required to solve a problem. It breaks down the solution into smaller, more manageable units, which can then be translated into code. The advantage of a flowchart is that it communicates logic in a universally understandable way; symbols and arrows make the process far less abstract than lines of code. For a novice, being able to see the flow of execution can make a tremendous difference in both comprehension and confidence.

Traditionally, students or developers create flowcharts manually. This requires analyzing the text of the problem, identifying key operations, decisions, and loops, and then representing them using flowchart symbols. While this method is effective, it is also time-consuming. For complex problems, the process of building a flowchart manually may take hours or even days. In some cases, beginners lose motivation during this lengthy stage and abandon the task altogether.

To address this difficulty, the idea of a Text-to-Flowchart Converter emerges as a practical solution. Such a tool would automatically convert problem statements or algorithm descriptions into

corresponding flowcharts using Natural Language Processing (NLP) and JavaScript-based visualization libraries. Instead of manually mapping every step, a user could simply input descriptive text, and within seconds, the system would generate a logical flowchart. This not only saves valuable time but also reduces the cognitive load on beginners, allowing them to focus on learning the logic itself rather than struggling with its representation.

The core functionality of the Text-to-Flowchart Converter lies in NLP. Natural Language Processing techniques can analyze a given text, identify verbs, conditions, and logical connectors, and map them to respective flowchart components such as processes, decisions, and arrows. JavaScript libraries for flowchart generation can then visually render these components in real time. For instance, keywords like "if", "else", and "loop" can directly correspond to decision diamonds or loop structures, while action verbs may translate into process blocks.

From an educational perspective, this innovation has profound benefits. It provides immediate feedback to learners by showing them how their textual understanding of a problem translates into logical steps. This encourages experimentation; like students can rephrase or modify their problem statements and instantly observe how the flowchart changes. Over time, they begin to recognize common logical structures and patterns, making the transition to coding much smoother.

In practice, the Text-to-Flowchart Converter not only helps students but can also support teachers and trainers. Instructors can quickly generate visual aids for classroom explanations, reducing preparation time. Furthermore, developers working in professional environments could use the tool for rapid prototyping of system workflows.

In conclusion, designing a Text-to-Flowchart Converter bridges a critical gap between problem description and solution design, particularly for novice programmers. By combining the analytical power of NLP with the visualization strength of JavaScript libraries, the tool makes programming more approachable, reduces frustration, and accelerates the learning curve. What once took hours of manual effort can now be achieved in seconds, ultimately making logic-building accessible to everyone (Figure 2).

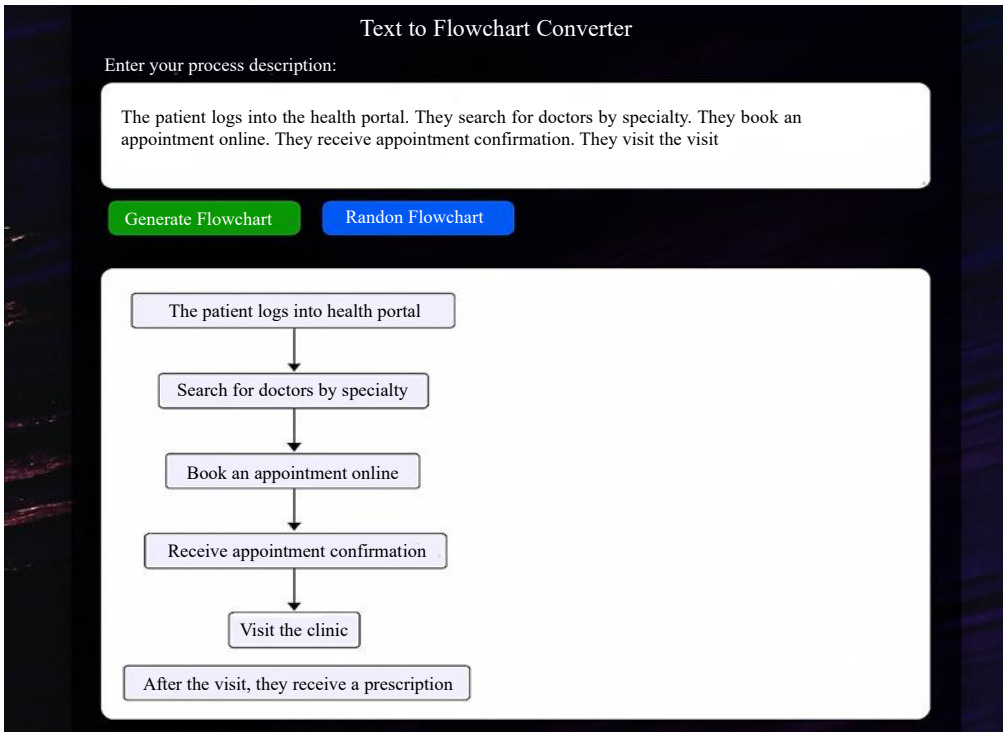

Figure 2. Text to flowchart converter.

CONCLUSION

The Text to Flowchart Converter effectively converts formal textual descriptions into graphical flowcharts, making it easier to comprehend and analyze logic or processes. Through automation of this process, the tool saves time, minimizes the chances of human error, and presents a clear visualization of workflows.

This solution is especially useful in software development, education, and business process, where clarity and efficiency are paramount. Future development can focus on enabling more natural language variations, integration with collaboration platforms, and more precise flowchart generation.

Future Scope

Teams might turn meeting notes into formal flowcharts immediately. Provide improvement suggestions to flow (e.g., removing redundant steps, merging similar branches). Provide industry best practices for process optimization suggestions. Assist students in visualizing algorithms, processes, or story structures.

Automatically generate quiz questions or explanations from flowchart structure. Accept voice dictation, hand-written notes, or multi-language text. Create flowcharts for code, business processes, education, etc., in different languages. Utilize sophisticated NLP models to interpret complex, ambiguous, or domain-specific instructions. Auto-correction of ambiguous steps or logical gaps within the input text.

REFERENCES

1. Chopade A, Shingde V, Chavare A, Bhagwat T. Code Insight-Flowchart Generator. In 2024 IEEE 2nd International Conference on Computer, Communication and Control (IC4). 2024 Feb 8; 1–6.
2. Mhatre M, Pandey A, Rane H, Sahu S. A novel approach for creating flowcharts using generative ai. In 2024 IEEE Asia Pacific Conference on Innovation in Technology (APCIT). 2024 Jul 26; 1–7.
3. de Almeida A, Zumstein D. Artificial Intelligence in the Generation of Product Description on the Conversion-Rate. In International Scientific-Practical Conference. Cham: Springer Nature Switzerland; 2024 Oct 9; 412–426.
4. Allmendinger L. Diagrams and design tools in context. ACM SIGDOC Asterisk J Comput Doc. 1994 Nov 1; 18(4): 25–41.
5. Fogel K. Producing open source software: How to run a successful free software project. O'Reilly Media, Inc.; 2005 Oct 7.
6. Tonis RB, Beteringhe A. Application of the Fibonacci Series in Natural Language Processing. Didactica Danubiensis. 2023 Oct 31; 3(1): 59–71.
7. Morell JÁ, Camero A, Alba E. Jsdoop and tensorflow. js: Volunteer distributed web browser-based neural network training. IEEE Access. 2019 Oct 30; 7: 158671–84.
8. Smilkov D, Thorat N, Assogba Y, Nicholson C, Kreeger N, Yu P, Cai S, Nielsen E, Soegel D, Bileschi S, Terry M. Tensorflow. js: Machine learning for the web and beyond. Proceedings of Machine Learning and Systems. 2019 Apr 15; 1: 309–21.
9. Parasuraman B. Mastering Spring AI: The Java Developer's Guide for Large Language Models and Generative AI. Springer Nature; Germany: Apress, 2024 Dec 1.
10. Humphrys M. Bringing AI APIs into the classroom with a JavaScript coding site. Int J Inf Educ Technol. 2025; 15(2): 272–279.
11. Jain P, Aggarwal PK, Makar K, Shrivastava V, Maitrey S. Machine learning for web development: A fusion. In Artificial Intelligence and Speech Technology. CRC Press; Florida, United States. 2021 Jun 29; 45–52.
12. Kulkarni A, Shah H, D'Mello L, Shah K. Flowchart generation and mind map creation using extracted summarized text. In 2023 IEEE International Conference on Recent Advances in Science and Engineering Technology (ICRASET). 2023 Nov 23; 1–6.

13. Shafeek N, Karunarathne DD. _toFlowchart: A prototype compiler to convert source-code to flowchart. In 2018 IEEE 18th International Conference on Advances in ICT for Emerging Regions (ICTer). 2018 Sep 26; 157–167.
14. Ellson J, Gansner E, Koutsofios L, North SC, Woodhull G. Graphviz—open source graph drawing tools. In International symposium on graph drawing. Berlin, Heidelberg: Springer Berlin Heidelberg; 2001 Sep 23; 483–484.
15. Hooshyar D, Ahmad RB, Nasir MH. A framework for automatic text-to-flowchart conversion: A novel teaching aid for novice programmers. In 2014 IEEE International Conference on Computer, Control, Informatics and Its Applications (IC3INA). 2014 Oct 21; 7–12.

