International Journal of Trend in Scientific Research and Development (IJTSRD)
Volume 10 Issue 3, May-Jun 2026 Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

Built using the TypeScript and Node.js Collaborative Management Platform

Zhang Ying[1], Ji Yutao[2], Ma Mingrui[3]

[1]School of Accounting, Beijing Wuzi University, Beijing, China
[2,3]School of Computer Science and Artificial Intelligence, Beijing Wuzi University, Beijing, China

ABSTRACT

To address common management challenges in traditional university innovation and entrepreneurship projects-including chaotic task allocation, difficult progress tracking, unquantifiable member contributions, and lack of process traceability-this study proposes Tecline, a comprehensive intelligent supervision and collaboration management platform for such projects. Centered around a chain-based supervision mechanism and leveraging artificial intelligence and big data technologies, the platform integrates the fundamental frameworks and advantages of TypeScript and Node.js languages to demonstrate their critical role in collaborative management systems. By comparing them with traditional backend languages, the study highlights how TypeScript and Node.js enhance Tecline's functionality. The paper further outlines the platform's architecture built on these languages and details its application-layer features and key functionalities in practical use scenarios.

KEYWORDS: *Tecline: A comprehensive intelligent monitoring and collaborative management platform; supported programming languages: TypeScript and Node.js; Platform architecture.*

How to cite this paper: Zhang Ying | Ji Yutao | Ma Mingrui "Built using the TypeScript and Node.js Collaborative Management Platform" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-10 | Issue-3, June 2026, pp.1052-1059, URL: www.ijtsrd.com/papers/ijtsrd133312.pdf

IJTSRD133312

1. INTRODUCTION

The nation continues to promote the in-depth development of mass entrepreneurship and innovation, explicitly emphasizing the need to bridge the "last mile" of campus innovation initiatives and foster integrated development across education, technology, and talent resources. Within this context, our team has observed that mainstream team collaboration tools and general-purpose innovation platforms currently available only support basic task assignment and communication functions, lacking dedicated chain-based supervision mechanisms tailored for university innovation environments. These platforms fail to enable comprehensive task decomposition, dynamic performance evaluation, or outcome traceability throughout the entire process. Furthermore, some university-developed platforms offer limited functionality, failing to integrate artificial intelligence and big data technologies for intelligent matching and monitoring, nor do they facilitate collaboration with enterprises and research institutions to establish physical support networks. Consequently, these limitations make it difficult to fundamentally address challenges such as difficulties in implementing student entrepreneurship projects, insufficient resources, and inadequate regulatory oversight.

The project aims to address gaps in the campus innovation and entrepreneurship ecosystem, optimize student team management models, reduce startup trial-and-error costs, break down barriers in university-industry collaboration, help enterprises identify innovative talents and high-quality projects more effectively, and promote deep integration of innovation chains, industrial chains, and talent networks. Built using TypeScript and Node.js languages, it establishes the Tecline comprehensive intelligent supervision and collaborative management platform.

2. Overview of TypeScript and Node.js

2.1. Overview of TypeScript

TypeScript (abbreviated as TS) is a programming language developed by Microsoft, designed for large-scale project development, code reuse across front-

end and back-end environments, and enhanced LDE development. It adheres to the ECMAScript standard and builds upon it, encompassing all JavaScript functionalities as a superset while introducing additional structural features and security checks [1][2]. TS significantly improves development efficiency and code quality in large-scale projects requiring close team collaboration, such as collaborative management platforms. Its key advantages include:

2.1.1. Enhanced Type Safety

It leverages the strengths of Java and C#, employing static type annotations to enable developers to identify potential type errors during coding and resolve issues at their early stages. This significantly reduces low-level errors introduced in large collaborative projects due to members' unfamiliarity with the codebase, substantially lowering debugging costs.

2.1.2. Promote full-stack collaboration

It provides JavaScript with comprehensive object-oriented programming features, enabling consistent and rigorous architectural patterns (such as MVC or MVVM) across both front-end and back-end JavaScript/TS development. This facilitates easier collaboration between backend developers and front-end developers alike, fostering seamless full-stack teamwork.

2.1.3. Improve development efficiency

It integrates seamlessly with modern editors such as Visual Studio Code, offering highly intelligent code suggestions, auto-completion, real-time error detection, and code refactoring capabilities [1]. This significantly enhances coding efficiency and code consistency in collaborative development, especially when team members need to quickly familiarize themselves with others' code.

2.2. Overview of Node.js

Node.js (abbreviated as Node) was created by Ryan Dahl in 2021. It is an open-source, cross-platform JavaScript runtime built on the Chrome V8 engine, enabling JavaScript code-originally designed for browsers-to run on computers and servers [3]. Widely used in backend development, real-time applications, and full-stack development, Node leverages Chrome's V8 engine to execute JavaScript code compliant with the ECMAScript standard (the official specification for the JavaScript language), delivering robust performance for high-concurrency collaborative scenarios. Node offers the following advantages:

2.2.1. Unified Technology Stack

It enables teams to handle all tasks-from front-end interfaces to backend APIs-using a single language (JavaScript/TS). This eliminates technical barriers, makes full-stack development possible, significantly accelerates development iteration cycles, and simplifies team communication and collaboration.

2.2.2. Reduce additional resource costs

Its single-threaded event-driven model is ideally suited for handling I/O-intensive tasks (such as database queries and file operations), delivering exceptional performance in high-concurrency environments while processing large volumes of user requests at low hardware cost. This represents a significant performance advantage for applications like collaborative management platforms that require frequent data interactions and file uploads/downloads.

2.2.3. Unified Development Standards

It has given rise to the world's largest package management ecosystem, npm, as well as powerful build tools like webpack and Rollup. These tools are not only used in front-end development but also widely employed for automated building, testing, and deployment in back-end projects, making the entire development process more standardized and efficient-particularly in collaborative projects where they ensure all team members adhere to consistent development standards.

2.3. The core advantages of the collaborative development management platform built with TypeScript and Node.js (abbreviated as TS+Node):

2.3.1. Lower collaboration barriers

With a unified technology stack and robust type system, team collaboration becomes smoother and new members can learn faster.

2.3.2. Improve development efficiency

Full-stack development, modern development tools, and a rich ecosystem significantly shorten the product development cycle [2].

3. The necessity of Tecline adopting the TS+Node collaborative creation and management platform

3.1. Designed for long-term iteration and collaborative development by multiple teams

The Tecline platform is a long-term operational system: it undergoes annual updates to competition rules, support policies, application templates, and mentorship frameworks, while also managing staff transitions and onboarding new members.

TS comes with built-in code annotation-based documentation that clearly defines interface parameters, database entities, and business models, enabling new developers to quickly grasp the system architecture. When refactoring or modifying code, the editor automatically detects related errors, ensuring that changing one part won't trigger cascading issues

and supporting continuous iterative updates over time.

3.2. Facilitates subsequent expansion to multiple endpoints

For future development of entrepreneurs' mini-programs, H5 application interfaces, and mentor mobile applications, the front-end will continue to utilize the TS technology stack, while server-side code can leverage reusable models and utility functions. This unified technical framework supports multiple platforms without requiring backend infrastructure refactoring.

3.3. Supporting the architecture of a large-scale modular management platform

The platform features a comprehensive suite of modules: User Module (for entrepreneurs/mentors administrators), Competition Module, Application Materials Module, Evaluation and Scoring Module, Financial Subsidy Module, Notification System, and Data Dashboard.

TS supports generics, classes, decorators, namespaces, and dependency injection. When combined with Nest.js (the native TS framework), it enables strict layering: separation of controllers, services, data layers, and DTOs for object transmission, resulting in highly cohesive yet loosely coupled code that prevents the codebase from becoming chaotic or difficult to maintain in later stages.

3.4. Flexible expansion of supporting functions

Quickly scalable supporting tools: automatically generate application PDFs, batch-export review reports, schedule regular review reminder emails and SMS messages, and integrate with park OA systems, student registration systems, and bank subsidy payment interfaces. Node.js's asynchronous capabilities deliver significant performance advantages for batch tasks, while TS ensures parameter compliance and error-free integration with third-party APIs.

4. Support for the Tecline Collaborative Management Platform by TS+Node Language

4.1. Comparison with Other Programming Languages

There is considerable diversity among programming languages, and this study takes the commonly used languages Java, Python, and TS+Node as examples.

Java exhibits strict and rigorous syntax with strong constraints, high memory consumption, stringent hardware requirements, and a fragmented architecture that necessitates separate front-end and back-end development roles. New developers face significant learning challenges due to its demanding skill requirements; modification and refactoring processes are cumbersome, compilation and packaging are time-consuming, JVM configuration is complex with numerous parameters, and production deployment/ debugging requires extensive effort-all resulting in substantial overall investment costs. Python is limited by GIL global locks, lacks robust multithreaded performance, experiences frequent bottlenecks under high-concurrency scenarios, demonstrates low batch request throughput, lacks robust backend management tools or high-concurrency support solutions, often suffers from code formatting inconsistencies and hidden bugs during long-term collaboration, offers inadequate native transaction support, is prone to parameter type vulnerabilities with dynamic types, entails high maintenance costs, presents challenges for scaling main components, requires prolonged stability testing, and has slow deployment speeds. In contrast, TS+Node's architectural design allows flexible parameter flexibility adjustments, delivers excellent throughput in I/O-intensive scenarios with minimal memory overhead, features a unified full-stack ecosystem, supports mature shared microservices across front-end/back-end interfaces, achieves optimal end-to-end integration with moderate learning curve, effectively avoids most parameter type errors, enables short development cycles [2], maintains stable operation on low-spec servers, offers rapid compilation speeds, simple startup commands, easy horizontal scaling, and is ideal for incremental expansion in small-to-medium platforms. Table 1 provides a comprehensive comparison of these three programming languages' support capabilities for the Tecline collaborative management platform:

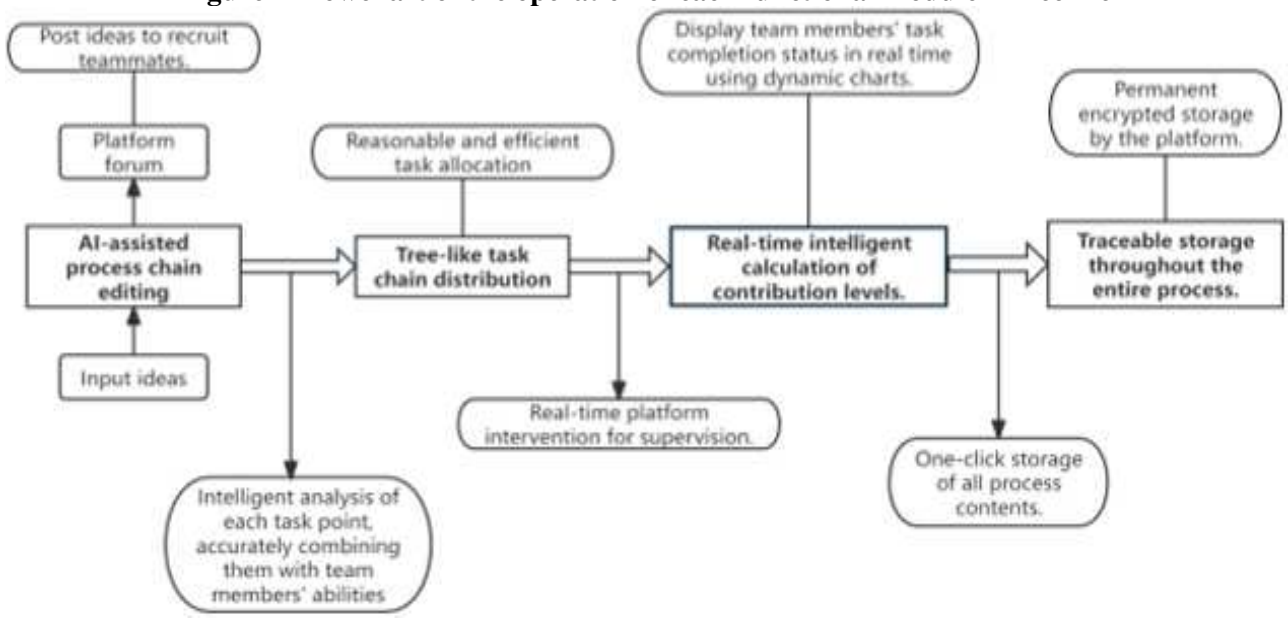
Table 1 Comparison chart of three different programming languages

	Java	Python	TS+Node
language foundation	The grammar is strictly standardized and highly restrictive.	Missing mandatory code constraints	Static strong type; type rules can be adjusted flexibly as needed
Performance and concurrency capabilities	High memory consumption and stringent hardware requirements	EasyCardon has low throughput for batch requests	Low memory overhead with excellent throughput
Architecture Ecosystem	one-stop	The backend management and tools supporting high-concurrency services are inadequate.	The lightweight backend tools are comprehensive, and microservices support is mature.
Team Collaboration and Development Efficiency	It takes time to get started, and modifying or restructuring is complex.	Easy to learn, but long-term collaboration among multiple users may lead to formatting issues and hidden bugs.	The level of proficiency is moderate; it's most efficient when one person handles multiple tasks.
data security	Highest security	Most insecure	Medium security
Construction and maintenance costs	Slow deployment process with high overall investment in both software and hardware	High maintenance costs in the long term, with continuous increases in expansion expenses.	Lightweight and low memory requirements; can run on basic servers
Scalability and Multi-device Compatibility	Best expansion capability	Not suitable for system expansion	Suitable for small to medium-sized platforms to expand and iterate annually.
Deployment and Launch Time	The JVM environment configuration is complex, with numerous cumbersome parameters, and production deployment debugging takes a long time.	The production environment requires high compatibility and stability; debugging takes a considerable amount of time.	Fast compilation speed, concise startup commands, and rapid deployment time
Information Update	The modification cycle is long, and updates are slow.	It is difficult to update information.	Information updates efficiently.
Data Format	The structure is arbitrary and prone to errors	The data structure is rigorous.	Strictly define structured data
Project Management Approach Project Management Approach	The separation between front-end and back-end languages	High version conflicts occur due to reliance on pip management	*A single technology stack that integrates both the front-end and back-end components.*

4.2. Support and facilitate the operation of all functional modules on the Tecline platform

The Tecline platform is a digital-intelligent integrated service platform for comprehensive lifecycle monitoring and management of college students 'innovation and entrepreneurship activities. The synergistic operation of its various functional modules plays a crucial role in the platform's functionality. It comprises six core components: an AI-assisted workflow editor, a dedicated forum, built-in team chat rooms, a tree-based intelligent task allocation system, a comprehensive contribution evaluation model, and a traceable project repository. The AI workflow editor supports drag-and-drop customization of workflows tailored to diverse project requirements; the tree-based task allocation mechanism optimally matches tasks based on team members' expertise, availability, and performance metrics; the contribution assessment model generates multidimensional competency profiles and dynamic scores using 27 types of behavioral data including code submissions, documentation contributions, and meeting attendance; while the end-to-end outcome traceability system records key process nodes via blockchain to ensure authentic execution and clear accountability. As illustrated in Figure 1.

Figure 1 Flowchart of the operation of each functional module in Tecline



4.2.1. Tree-shaped task chain allocation and chain-based supervision and management

One of the platform's core features. The platform utilizes artificial intelligence technology to develop an AI workflow chain editing system, combined with a tree-based task chain allocation architecture, to break down project tasks layer by layer.

The system intelligently matches personnel with specific tasks by leveraging team members 'professional expertise and skill profiles, ensuring precise task allocation. Meanwhile, the platform backend generates real-time visual tree-shaped task chain diagrams, employing a chain-based supervision approach to track each task throughout its entire lifecycle and dynamically display members' task completion status. This methodology enhances the alignment between personnel and tasks, addressing issues such as inefficient role distribution and sluggish task progression.

4.2.2. Quantitative Management of Intelligent Contribution Levels

The platform conducts comparative analysis and assigns weights across multiple dimensions, including task difficulty, technical complexity, implementation effectiveness, and collaboration coordination.

The platform automatically collects data such as task completion records and interaction logs for each member, calculates their contribution percentage in real time, and visualizes individual task performance and contributions using dynamic charts. This approach enhances process oversight, addresses the challenge of quantifying team members' contributions, and ensures efficient team operations.

4.2.3. Full-process outcome traceability

Leveraging blockchain-based evidence storage technology, the platform assigns blockchain traceability labels to data throughout the project lifecycle and to interim deliverables. This feature ensures that all project data is tamper-proof and fully traceable, providing credible certification for students' innovation achievements and practical competencies. This approach addresses the issues of low recognition of innovative outcomes and the lack of verifiable proof of capabilities.

4.3. Application of the Digital Intelligent Task Allocation and Supervision Platform

To validate the practical effectiveness of the digital intelligent task allocation and monitoring management platform, this paper takes a real-world implementation project as a case study, focusing on two key features-tree-based task chain allocation and chain-based supervision management, along with intelligent contribution quantification management-to demonstrate system implementation and evaluation of application outcomes.

4.3.1. Tree-shaped task chain allocation and chain-based supervision and management

During the project's initial phase, the project concept is first entered into the platform, which utilizes AI to optimize the workflow editing process for implementation. The platform then conducts precise analysis of how each task within the workflow aligns with team members' capabilities, subsequently assigning tasks efficiently to individuals. Meanwhile, a tree-based task dashboard in the backend continuously displays task assignments and completion statuses. This intelligent task allocation and monitoring system enhances team members' competency

alignment with tasks, improves task completion efficiency, reduces redundant workloads, and significantly boosts overall task management effectiveness.

4.3.2. Quantitative Management of Intelligent Contribution Levels

This project conducts comprehensive comparisons across various aspects-including task technology, implementation, and design-to analyze the varying time requirements of different tasks for team members and accurately assess individual contributions. It dynamically visualizes members' task completion rates and personal contribution percentages in real-time using interactive charts, with continuous updates throughout the process to enable precise evaluation of team performance [4]. This intelligent contribution quantification management approach enhances process oversight, improves team efficiency, and ensures effective project execution.

The Tecline collaborative management platform, built using the TS+Node language, provides users with a user-friendly communication solution. It enables efficient project workflow execution, precise task allocation, transparent tracking of task completion and contribution levels, thereby facilitating effective team management, ensuring high-quality project deliverables, and reducing organizational redundancy and resource waste. Additionally, it offers Tecline a robust backend infrastructure characterized by stability, speed, cost-effectiveness, and easy scalability, while delivering enhanced development experience and improved security for developers.

5. Building a Collaborative Management Platform Based on TS + Node Language Tools

5.1. Basic Architecture of the Collaborative Management Platform Based on TS+Node Programming Language Tools

The fundamental architecture of the TS+Node programming language-based collaborative management platform consists of three components: core technology stack, overall hierarchy, and operation/deployment mechanisms, as shown in Table 2.

Table 2 shows the basic architecture of the TS+Node programming language tool collaboration management platform.

The Three Fundamental Architectures	Core Technology Stack	Development Environment: Node.js; Development Language: TypeScript; Primary Framework: NestJS; Database: MySQL; Interface Specifications: RESTful API + WebSocket (real-time messaging); ORM Framework: TypeORM (type mapping, table management); Cache: Redis (for session persistence, hot data, and API rate limiting); File Storage: Cloud OSS (for attachments, images, and certificates); Operations Tools: PM2, logging component.	
	Overall Layering	Access Layer	Request Entry: Processes all client request entries, including cross-domain handling and request routing distribution; maintains global logs, implements API rate limiting, and performs IP interception; provides unified exception capture and response format encapsulation.
		operating floor,key-course	Responsible for receiving and responding; does not write complex business logic
		Business Layer	Implement core business processes for users, projects, events, resources, and messages; include permission validation, data rules, decoupled interfaces for business transaction processing and data operations, and reusable logic.
		Data Access Layer（Repository / ORM）	Map database entities using TypeORM to perform insert, delete, update, and query operations, as well as join queries; enforce type constraints on database fields to prevent data anomalies; and centrally manage database interaction logic.
		Common Foundation Layer	Tool functions, data encryption, format conversion; custom enumerations, generic types, constant configuration

		Data Storage Layer	Primary database MySQL: Stores core business data such as users, projects, events, and resources; Redis cache: Stores session logs, frequently queried data, and API rate limits; Object storage OSS: Stores images, documents, project attachments, and digital certificates
	Run and Deploy	exploitation environment, development environment	TS Node hot-starts with code taking effect in real time and local debugging
		Compile and Package	Compile TypeScript into native JavaScript to generate production code
		Online Deployment	Linux server with PM2 process daemon for process monitoring and automatic restart upon anomalies
		Extension Architecture	Supports multi-instance deployment and load balancing, enabling smooth migration to a microservices architecture as business grows

5.2. Core Application Functions of Tecline Collaborative Management Platform's Backend Implementation Capabilities Tecline

The core functionalities of the collaboration management platform's backend implementation primarily consist of six modules: User Experience, Innovation and Entrepreneurship Projects, Event Management, Resource Matching, Messaging & Notifications, and General Administration Panel.

5.2.1. User System Module

The platform supports account creation for multiple roles-including makers, students, mentors, judges, enterprises, and administrators-covering diverse needs. It offers essential features such as registration, login, JWT authentication, password management, and personal information maintenance, providing users with secure and reliable authentic information support.

5.2.2. Innovation and Entrepreneurship Project Module

The platform enables rapid and precise management of team project submissions, edits, rejections, and status updates. It features dynamic form validation, text/image/attachment upload and storage, and real-time project information queries, version history, and categorized search for efficient data retrieval.

5.2.3. Event Management Module

The platform promptly and efficiently captures various competition events nationwide, publishing all relevant information through its internal forum to assist participating teams in completing online registration swiftly, thereby significantly reducing the risk of missing opportunities. It also integrates with the judging website to provide real-time updates on online evaluations, score entry, result consolidation, and offers a one-stop solution for generating, downloading, and exporting electronic certificates, eliminating the need to navigate multiple web pages.

5.2.4. Resource Matching Module

The platform integrates management information on data from numerous mentors, enterprises, and investment resources, effectively addressing the challenge of finding specialized advisors for innovation and entrepreneurship projects. It also features resource search, intelligent matching, and consultation messaging functions to help innovation teams efficiently identify qualified mentors and accelerate project implementation.

5.2.5. Messages and Notifications Module

The platform features a comprehensive messaging network, startup forum, and community forum, enabling timely delivery of in-site messages, SMS, and emails, real-time notifications via WebSocket, and online interaction to efficiently address both internal and external communication needs.

5.2.6. General Backend Module

The platform's backend continuously aggregates all team data, supporting report generation, Excel file import/export, global logging, anomaly detection, and API monitoring. All relevant files are centrally stored with integrated image and document processing capabilities, complemented by a visual dashboard for multidimensional analysis and permission-based management, ensuring system security, controllability, and full content traceability.

6. Conclusions

This paper addresses the current state of innovation and entrepreneurship development in universities and the challenges faced by existing platforms, delivering a comprehensive design for the Tecline collaborative

management platform using TypeScript and Node.js. Through technical comparisons and requirement analysis, the advantages of this technology stack in development efficiency, system maintenance, and concurrent processing are demonstrated. The platform features a complete architecture and multiple business modules, enabling end-to-end collaboration, oversight, and resource integration for innovation and entrepreneurship projects.

Current research has only completed the conceptual design phase, and the system's actual performance remains to be validated through practical application. Moving forward, efforts will focus on deploying and testing the platform, optimizing intelligent features and system performance, expanding application scenarios, and continuously enhancing its service capabilities to provide actionable references for digital management of innovation and entrepreneurship in universities.

References

[1] Fan Dingwei. Design and Research of a TypeScript-Based Front-end MVVM Framework [D]. Beijing University of Posts and Telecommunications, 2021. DOI:10.26969/d.cnki.gbydu.2021.001833.

[2] Chen Guang. Implementing a Tree Directory Using TypeScript [J]. Fujian Computer, 2018,34(6):153–155. DOI:10.16707/j.cnki.fjpc. 2018.06.07.

[3] Zhu Jun, Zhao Zitong. Design and Implementation of a Node.js-Based Platform for Visualizing Tourism Resources [J]. Electronic Technology and Software Engineering, 2022, (3):62–66. DOI:10.20109/j.cnki.etse. 2022.03.01.

[4] Liu Jinyu. Development and Application of Digital Governance Platform for Smart Cities [J]. Smart China, 2025, (11):110-111.