iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IX **Month of publication:** September 2026

DOI: https://doi.org/10.22214/ijraset.2026.84995

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IX Sep 2026- Available at www.ijraset.com

An Evidence Bounded Workflow Architecture for Integrated Goal, Study, and Venture Execution in ForgeanOS: A Context-Aware Multi-Module AI Platform for Academic and Career Support

Ms. Shraddha Raut[1], Shravya Bansod[2], Neil Sewak[3], Pranav Kohad[4]
Author 1: Project guide | Authors 2–4: Student researchers
B.Tech. Computer Science & Engineering (AI), School of Engineering & Technology, G H Raisoni Skill Tech University, Nagpur, Maharashtra, India
Corresponding author: Shravya Bansod

Abstract: ***Fragmented goal, academic, career, and venture tools can leave planned work disconnected from recorded outcomes. This paper presents a source-based case study of ForgeanOS, a Next.js application that links goal structures to study and founder execution while separating deterministic recommendations from optional generative AI. The architecture uses authenticated route handlers, owner-scoped PostgreSQL records, transactional mutations, version checks, and metered provider routing. A repository guide reports 621 passing tests and a local production build on 27 September 2026, plus disposable-account FounderPilot checks; the underlying test logs and deployed runtime were not supplied for independent replication. No user study, comparative benchmark, or production performance measurement was available. The contribution is an explicit workflow and evidence model, with its implementation boundaries and validation gaps stated rather than inferred benefits.***
Keywords: Goal management, self-regulated learning, workflow integration, software architecture, evidence tracking, generative AI.

I. INTRODUCTION

Self-regulated learning involves planning, performance monitoring, and reflection [1]. Workflow research likewise distinguishes process definitions from executed work and records [2]. These ideas motivate a software question: how can a personal system connect a chosen goal to concrete study or venture actions while preserving the difference between a recommendation, a recorded action, and an observed result?

ForgeanOS combines GoalGraph, Career Guide, StudyFlow, StudyAI, FounderPilot, and supporting workspaces in one authenticated application. This paper analyzes the documented source architecture and its explicit state transitions. It does not infer educational or commercial effectiveness from the existence of code. Sections II–VII describe the gap, design, mechanisms, and available verification; Sections VIII–X discuss limitations and future evaluation.

II. PROBLEM AND RELATED WORK

Zimmerman describes learning as a cycle of forethought, performance, and self-reflection [1], but that educational framework does not specify persistence, authorization, or cross-domain application links. The Workflow Management Coalition reference model describes interoperable workflow components [2], while Fielding analyzes architectural constraints for network-based applications [3]. These works provide concepts rather than evidence that the particular combination implemented here improves outcomes.

The engineering gap addressed in this case is the representation of goals, academic sessions, and venture actions as linked but distinct records, with measurements kept separate from generated advice. This is a design gap within the presented system scope, not a claim that comparable commercial systems do not exist. A systematic product comparison and controlled user experiment remain necessary for a broader novelty or effectiveness claim.

©IJRASET: All Rights are Reserved

III. OBJECTIVES AND METHOD

The design objectives are to (i) connect planning structures to execution records through explicit links, (ii) produce interpretable scheduling and founder recommendations from entered data, (iii) require deliberate recording of learning or business observations, and (iv) constrain access and AI use at the server. The method is descriptive source analysis of three supplied platform guides, whose current research guide reports its basis as a 288-file Repomix snapshot. The repository itself and raw test outputs were not among the four attachments; source file paths below are provenance reported by that guide, not independently inspected code. [4]–[6].

A. Originality and research contribution

The original contribution of this case study is a specific system-level design: goal milestones can connect to independently persisted study and venture actions; deterministic planners use explicit capacity, prerequisite, revision, and measurement records; model-generated advice is routed through separately metered services; and progress is updated only by defined user or system mutations. The resulting distinction between intended, performed, and observed states is a design property that can be inspected and tested. The contribution is an integration and state-model account, not a new optimization algorithm, learning theory, or demonstrated improvement in outcomes [4]–[6]. The cited self-regulated learning framework explains a learner cycle [1], the workflow reference model describes process-system interfaces [2], and network-architecture work provides design constraints [3]. None of these references evaluates this implementation. Combining their concerns in ForgeanOS is an original framing of the supplied system, but neither the guides nor the present literature review establishes global novelty over existing products. Originality of authorship and absence of prior publication must be confirmed by the named authors; a similarity screening against external corpora is outside this source analysis.

IV. SYSTEM ARCHITECTURE

The browser renders Next.js 16 App Router pages with React 19. Authenticated route handlers validate requests and entitlements before calling domain logic. Supabase Auth identifies the user; PostgreSQL with row-level security holds owned records. Server-only transactions handle privileged mutations. Optional model requests pass through a metered gateway to configured Gemini, Groq, or NVIDIA providers; Razorpay handles subscription creation and webhook updates when configured. This is a logical architecture, not a verified deployment diagram. [4].

1) Architecture summary. Logical architecture from supplied platform guide.
2) Client: Dashboard forms, workspaces, study timer and rendering
3) API and domain: Session and plan checks; validated commands; study, goal and founder calculations
4) Persistence: PostgreSQL schema, owner-scoped RLS, transactions and migrations
5) External: Supabase Auth, optional AI providers, conditional Razorpay billing

A typical mutation flows from browser form to authenticated route, validation and entitlement checks, domain command, owner-scoped transaction, persisted version, and refreshed snapshot. AI generation has an additional quota reservation and provider-routing step. A generated answer alone does not advance a goal, establish mastery, or create observed venture metrics.

V. DESIGN AND IMPLEMENTATION

A. Goal and career records

GoalGraph stores goals, milestones, tasks, prerequisites, and tracked states; calculated progress and suggested actions derive from these records. Career Guide searches by profession and geography, presents attributed and dated claims, compares up to three professions, and can activate a selected path as goal records in a transaction. Research availability depends on a configured search-capable Gemini model. Exact-match merging reduces duplicate requirements but similar wording may require manual review.

B. Academic execution

StudyFlow holds subjects, topics, assignments or projects, exams, availability, revision records, sessions, and progress events. A GET returns an owner-scoped snapshot; validated POST commands use a server-only transaction, a snapshot version, and a short retry ledger. The documented rule allows one active or paused session per user. Its priority combines importance, mastery gap, deadline proximity, revision due, unstarted work, and focus compatibility. [4], [5].

Study debt is a bounded heuristic; momentum weights seven-day completion (50%), study-day consistency (30%), and recorded productive outcomes (20%). Readiness weights coverage (30%), measured practice mastery (40%), practiced topics (20%), and revised topics (10%), and is withheld without practice measurements.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

These are specified calculations rather than calibrated predictions. StudySim allocates work to calendar capacity up to 365 days, shows collisions, and adds sessions only after confirmation; its undo removes only new sessions that have not started. StudyAI can use an owned topic as context; a separate explicit outcome submission records time, confidence, or practice accuracy.

C. Venture execution

The current Idea Forge route opens FounderPilot, replacing an older blueprint-generator description. Its normalized record classes are ventures, tasks, metrics, budget, skills, experiments, and decisions. Entered constraints and prerequisites lead to explained NOW, NEXT, LATER, and DO NOT BUILD YET buckets. Actual spending, measured metrics, and observed experiment results remain separate from estimated task costs, completion counts, and hypotheses. Remaining cash uses initial capital plus funding minus expenses; runway is left undefined when burn is zero. Versioned, serialized owner-scoped mutations persist records and can link a founder action to GoalGraph or create a StudyFlow learning action. [4], [6].

Workflow summary. Selected documented workflows and evidence boundaries.

StudySim: Scenario input → capacity allocation → preview; Confirmed sessions; started sessions not undoable by scenario undo

StudyAI: Question with optional owned topic → metered model call; Answer; progress changes only on submitted outcome

FounderPilot: Record update → constraints and prerequisites recalculated; Explained board; task completion does not imply revenue

Career activation: Selected researched path → transactional goal creation; Goal records; retrieved claims still need source review

VI. DATA, SECURITY, AND RELIABILITY

Owner-scoped RLS and authenticated sessions restrict personal records. StudyFlow and FounderPilot mutations are server-only, validated, version checked, and transactionally serialized; a retry ledger and webhook idempotency limit duplicate effects. Central entitlements and atomic monthly credit reservations constrain model calls. Input sizes are bounded and rendered Markdown is sanitized. Resume extraction is temporary unless source retention is explicitly selected; a retained original is accessed through an expiring link. These are controls described by the supplied source guide, not an independent penetration test or certification.

The persistent data are structured goals, sessions, venture records, profiles, and saved work rather than every uploaded original. The snapshot does not establish deployed storage policy, backups, operational access, complete erasure, or actual third-party data-handling terms. These require deployment-specific review.

VII. TESTING AND RESULTS

The supplied research guide cites a 27 September 2026 launch audit reporting 621 passing tests, a successful local production build, and live disposable-account FounderPilot checks. The original test report, commands, environment, coverage, and repository snapshot were not attached here; consequently these are attributed audit reports, not independently reproduced findings. The guide explicitly leaves authenticated visual review, live payment, production deployment, email delivery, and production-domain configuration unverified. No response-time, security, usability, learning-outcome, or venture-outcome measurement is documented.

The evidence supports a source-described implementation and a reported local verification milestone. It cannot support comparative superiority, reliability under production load, better grades, improved goal completion, or business impact. The deterministic and model-assisted boundaries are identifiable in the documented flows, but their practical comprehensibility to users has not been measured.

VIII. LIMITATIONS AND THREATS TO VALIDITY

This analysis relies on guides summarizing a code snapshot rather than direct inspection of source and raw test artifacts. External integrations depend on migrations, credentials, configuration, and live services. Heuristic weights are design choices; self-reported practice, competency, and financial entries can be incomplete. AI answers may be inaccurate despite attribution or fallback. Historical files may describe retired functions: the current guide states that heavy ingestion endpoints return HTTP 410 and should not be presented as active processing.

IX. FUTURE EVALUATION

A reproducible evaluation should publish a versioned build and test protocol, verify owner isolation and retry behavior, inspect mobile and accessible interfaces, and exercise payment and deployment paths with controlled accounts. A consent-based study could compare linked workflows against separate tools on time to a feasible plan, action completion, correction burden, and understanding of recommendation reasons. AI answers should be rated against human-reviewed factuality and attribution rubrics; failures and uncertainty should be reported with model versions and sample sizes. None of these studies has yet been reported.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IX Sep 2026- Available at www.ijraset.com

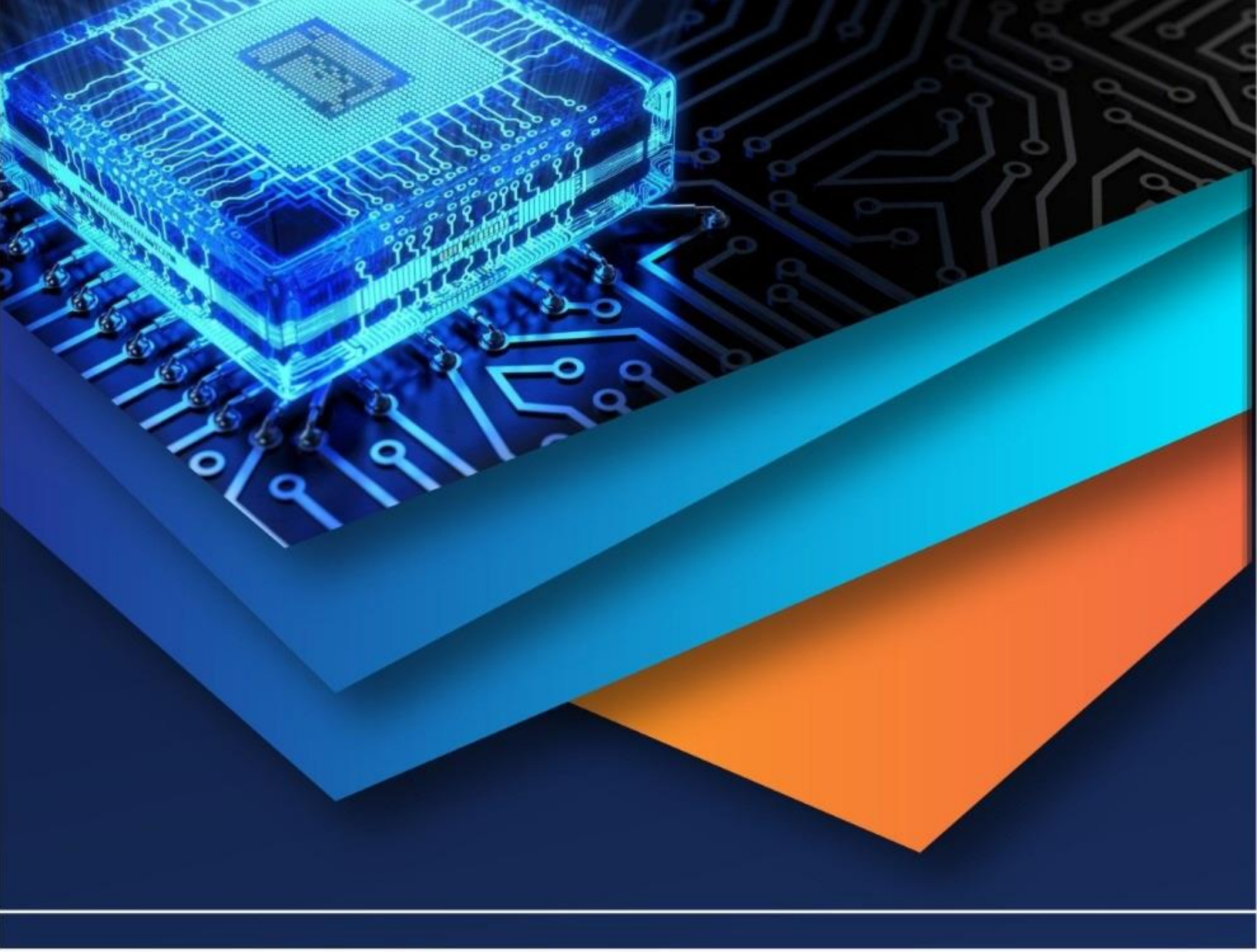
X. CONCLUSION

ForgeanOS is documented as an integrated application that connects goals, academic sessions, and venture execution through explicit records while treating AI output as advisory. Its defensible contribution is the architecture and the separation of planned, performed, and observed states. A guide reports local tests and a build, but user benefit and production properties remain open empirical questions.

REFERENCES

[1] B. J. Zimmerman, “Becoming a self-regulated learner: An overview,” Theory Into Practice, vol. 41, no. 2, pp. 64–70, 2002, doi: 10.1207/s15430421tip4102_2.
[2] D. Hollingsworth, The Workflow Reference Model, Workflow Management Coalition, TC00-1003, version 1.1, 1994. Available: https://www.aiai.ed.ac.uk/project/wfmc/ARCHIVE/
[3] R. T. Fielding, Architectural Styles and the Design of Network-based Software Architectures, Ph.D. dissertation, University of California, Irvine, 2000. Available: https://ics.uci.edu/~fielding/pubs/dissertation/top.htm
[4] ForgeanOS Platform Technical and Functional Guide, supplied author documentation, 29 September 2026.
[5] ForgeanOS Complete User Guide, supplied author documentation, 29 September 2026.
[6] ForgeanOS Complete User Guide 20 Pages, supplied author documentation, 29 September 2026.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)