FOSYS: AN AI-BASED WORKFLOW AUTOMATION SYSTEM FOR SCRUM AND SOFTWARE DEVELOPMENT TEAMS

Project Reference No.: *49S_BE_4803*

College : *AMC Engineering College, Bengaluru*
Branch : *Department of Artificial Intelligence and Machine Learning*
Guide(s) : *Prof. C Subhashri*
Student(s) : *Ms. Chandana Rao K*
Mr. Nagesh
Ms. Naveen Singh Divya Singh
Ms. Trisha G

Keywords:
AI-Based Workflow Automation, SCRUM Intelligence, Task Extraction using LLM, GitHub PR Integration, Role-Based Dashboard Systems

Introduction:

Modern software development teams rely on Agile and SCRUM methodologies, yet a critical gap persists between communication, planning, and execution. SCRUM meetings generate valuable insights, but these discussions often remain unstructured, requiring manual effort to convert them into actionable tasks. Additionally, development progress is tracked separately through platforms like GitHub, leading to fragmented workflows, reduced transparency, and limited accountability.

To address these challenges, the proposed project **FOSYS** introduces an AI-powered workflow automation system that transforms SCRUM discussions into structured tasks and synchronizes them with real-time GitHub pull request activities. By leveraging Large Language Models (LLMs), webhook integrations, and role-based dashboards, the system creates a unified pipeline connecting communication, planning, and development. The system aims to reduce manual effort and improve coordination in software development workflows.

Objectives:

- To design and develop an AI-powered system that converts SCRUM meeting transcripts into structured, actionable tasks using Large Language Models.
- To implement real-time synchronization between GitHub pull request events and task status for automated workflow tracking.
- To develop a role-based access control (RBAC) system with adaptive dashboards for developers, managers, and administrators.
- To integrate communication, task management, and GitHub pull request based task tracking into a unified workflow automation pipeline.
- To reduce manual effort, improve task traceability, and enhance coordination in Agile software development environments.

Methodology:
The methodology defines the workflow that transforms SCRUM transcripts and GitHub activity into synchronized, actionable tasks. The system operates through four key stages:

1. Transcript Acquisition and Input Handling
SCRUM meeting transcripts exported from Microsoft Teams (.vtt format) are used as input. These transcripts are pre-generated with diarized speaker segments and timestamps, providing structured conversational data for processing.

2. LLM-Based Task Extraction
A Gemini-based Large Language Model (LLM) processes the transcript using structured prompts to extract discussion points, action items, owners, deadlines, and priorities. The output is generated in JSON format, including task details and pending items. LLMs are used due to their ability to handle unstructured conversational data and capture contextual relationships more effectively than rule-based methods.

3. Task Structuring and PR-Based Workflow Tracking
Extracted tasks are stored in a relational database (Supabase) with unique identifiers. GitHub webhook events (pull request creation, updates, merges) are processed and mapped to tasks. Task states are automatically updated (Pending → In Progress → Completed). This ensures real-time synchronization between development activity and task progress.

4. Role-Based Access Control and Dashboard Visualization Role-Based Access Control (RBAC) manages system access for Developers, Managers and Administrators. Dynamic dashboards provide role-specific views of tasks, pull request status. This ensures secure access and improved coordination.

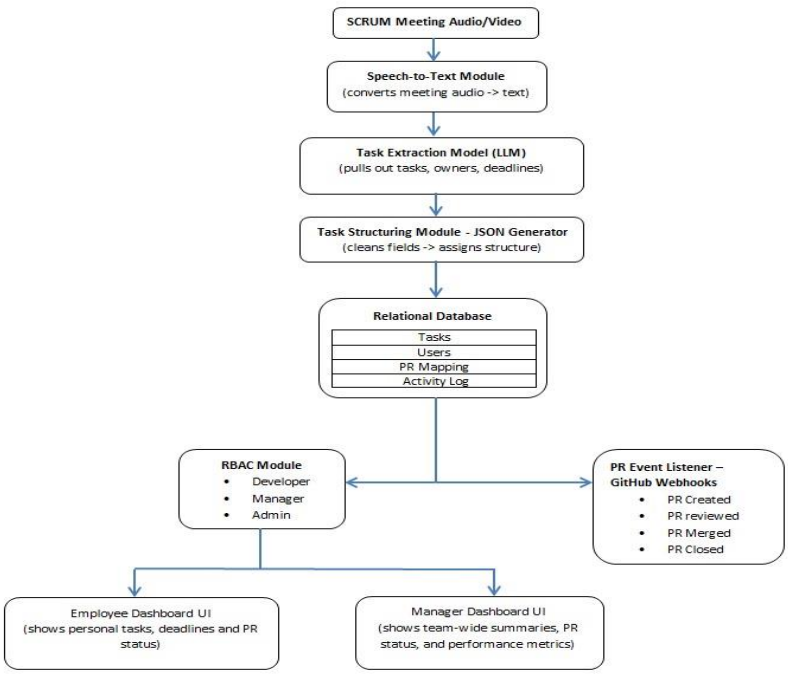

Fig. 1. System architecture showing transcript processing, LLM extraction, task structuring, PR event mapping, and RBAC driven dashboards.

The flowchart illustrates the end-to-end workflow of the proposed system, starting from SCRUM meeting transcripts obtained from Microsoft Teams to task automation.

Result and Conclusion:

1. The system was evaluated using real SCRUM meeting transcripts along with simulated GitHub pull request events.
2. The LLM-based task extraction module demonstrated high accuracy in identifying actionable tasks, assigning responsibilities, and capturing key discussion points.
3. The pull request synchronization module enabled automatic task status updates with near real-time tracking of development progress.
4. The role-based dashboard system provided stable performance, role-specific views, and improved clarity with reduced information overload.
5. The system effectively integrates AI-driven task extraction with automated workflow tracking, improving coordination, task visibility, and overall efficiency in Agile software development environments.

Project Outcome & Industry Relevance:

1. The proposed system results in a workflow automation platform that bridges the gap between SCRUM communication, task management, and development activity.
2. It converts meeting discussions into structured tasks and synchronizes them with GitHub pull requests, improving traceability.
3. Role-based dashboards provide clear and filtered visibility for developers, managers, and administrators, enabling better decision-making and progress monitoring.
4. The solution is highly relevant to software development companies, technology startups, and distributed Agile teams using SCRUM methodologies.
5. It addresses industry challenges such as fragmented workflows, communication gaps, and inconsistent task tracking.

Working Model vs. Simulation/Study:

1. The project is developed as a functional software prototype integrating LLM-based task extraction, GitHub webhook processing, and role-based dashboards.
2. Real SCRUM meeting transcripts are used as input for task extraction.
3. GitHub pull request events are simulated for controlled testing of task synchronization.
4. The system demonstrates end-to-end workflow execution in a prototype environment.
5. The project represents a working model with partial simulation, where core functionalities are implemented and validated under controlled conditions.

Project Outcomes and Learnings:

1. Understanding application of LLMs for extracting structured data from conversations
2. Designing end-to-end workflow pipelines
3. Implementing GitHub webhook-based real-time updates
4. Developing secure systems using RBAC
5. Gaining experience in backend architecture and system integration

References:

[1] X. Zhang, *et al.*, “MeetingBank: A Benchmark Dataset for Multi-Topic Meeting Summarization,” in *Proc. ACL*, 2023.

[2] M. Zhong, *et al.*, “QMSum: A Query-Based Multi-Domain Meeting Summarization Dataset,” in *Proc. NAACL*, 2021.

[3] P. Liu, *et al.*, “Extracting Action Items from Noisy Transcripts Using LLM-Based Semantic Parsing,” in *Proc. EMNLP*, 2022.

[4] R. Souza, *et al.*, “AI-Driven Automation of SCRUM Ceremonies for Distributed Teams,” *Journal of Systems and Software*, 2024.

[5] M. Wessel, *et al.*, “The Impact of GitHub Actions on the Pull Request Process,” in *Proc. ICSE*, 2023.

[6] R. Sandhu, *et al.*, “Role-Based Access Control Models,” *IEEE Computer*, 2020.

Future Scope:

1. Integration of analytics to forecast task delays, identify workflow bottlenecks, and enable proactive decision-making in Agile environments.
2. Enhancement of pull request monitoring by incorporating code quality metrics, test coverage analysis, and automated validation mechanisms for improved reliability.
3. Development of intelligent workflow automation features such as reminders and integration with scalable microservices architecture to support larger teams and multi-project environments.