DOI: https://doi.org/ 10.70656/ijcse.v3i02.892

Design of an Integrated Meeting Management Information System for Enhancing Team Coordination Effectiveness and Meeting Information Management at PT Ascon Inovasi Data

1st Zahra Aulia Fidelita
Information System Study Program
Department of Engineering
University of Bengkulu
Bengkulu, Indonesia
zahraauliafidelita758@gmail.com

2nd Yusran Panca Putra
Information System Study Program
Department of Engineering
University of Bengkulu
Bengkulu, Indonesia
yusranpanca@unib.ac.id

3rd Robbie Shugara
PT Ascon Inovasi Data
Bengkulu, Indonesia
akachopa@gmail.com

Abstract*— The advancement of information technology has driven digital transformation in organizational coordination processes, particularly in meeting management. However, the use of multiple unintegrated platforms often leads to issues such as scheduling conflicts, data redundancy, and lack of structured documentation. This study aims to design and develop a web-based integrated meeting management information system at PT Ascon Inovasi Data to improve business process efficiency and team coordination. The system is developed using the prototype model, which enables iterative engagement with users. The proposed system implements real API integration with Zoom and Google Calendar services using OAuth 2.0 authentication—fully functional in a local development environment—and provides features such as meeting scheduling, room reservation, digital minutes, and automated notifications. Usability evaluation using the System Usability Scale (SUS) with eight representative employees (44% of the workforce) yielded a score of 77.5, indicating good perceived usability. Although the system has not yet been deployed to a public domain, the functional prototype demonstrates provides a functional foundation for future production deployment into a comprehensive production solution for organizational meeting management.

Keywords- Information Systems, Meeting Management, System Integration, Application Programming Interface (API), OAuth 2.0, Zoom API, Google Calendar API, Prototype, Team Coordination, Digital Transformation

I. INTRODUCTION

In the current era of digital transformation, organizations increasingly rely on coordinated teamwork to achieve strategic objectives. Meetings remain a fundamental mechanism for aligning goals, distributing tasks, and monitoring progress. However, many organizations still struggle with fragmented meeting management practices, including unstructured scheduling, dispersed documentation, and poor follow-up on meeting outcomes. Mohamad Johari and Arbain showed that digitizing manual meeting processes can significantly improve efficiency and reduce coordination errors. [1]. Blanchard and Allen emphasize that meetings function not only as a medium for information exchange but also as a critical moment for building team cohesion and identity, particularly in distributed work settings [2]. Nevertheless, these benefits are often hindered by poor integration among commonly used tools such as Google Calendar, Zoom, and Google Meet, resulting in information fragmentation and unnecessary administrative burden [3]. The novelty of this study lies in the integration of three core capabilities—physical room booking with conflict prevention, real API synchronization with Zoom and Google Calendar using OAuth 2.0, and structured digital minutes—within a single web-based platform designed specifically for a small organization with limited technical resources. Unlike prior works that focus on isolated features, this system uniquely combines these functionalities without requiring third-party middleware or expensive subscriptions.

PT Ascon Inovasi Data, a small-sized company operating in data and technology innovation, faces similar challenges. Preliminary observations revealed that meeting management is largely manual and fragmented: employees rely on spreadsheets or instant messaging to book rooms and invite participants. Scheduling conflicts are common, meeting minutes are often lost or inconsistently documented, and there is no automated mechanism to synchronize meeting schedules with personal calendars. Moreover, the absence of integration with third-party services forces users to manually create meeting links via Zoom or Google Meet and distribute them separately.

Prior studies have demonstrated the potential of integrating calendar APIs and using artificial intelligence for minutes generation [4]. However, these approaches have not fully addressed the integration of physical room booking and simultaneous integration with Zoom and Google Calendar within a unified platform. Others have developed prototype-based meeting management systems focusing on user and document management [5]. However, these approaches have not fully addressed the combined requirements of physical room booking, online meeting link generation, and calendar synchronization within a single, unified platform – especially for a small organization with limited technical resources.

To position the contribution of this study relative to existing work, Table 1 presents a comparative analysis of key features across representative meeting management systems and the proposed prototype.

Table 1. Comparison of Key Features with Existing Systems

Feature	Mohamad Johari & Arbain [1]	Rachmawati [3]	Arthatia & Mubassiran[4]	Sari[5]	This Study (Prototype)
Room Booking (offline)	×	✓	×	✓	✓
Zoom API Integration	×	×	×	×	✓
Google Calendar API Integration	×	×	×	×	✓
OAuth 2.0 Authentication	×	×	×	×	✓
Digital Minutes (Rich Text + Images + PDF)	Partial	✓	✓	Partial	✓
Internal Notifications	×	✓		✓	✓
Deployment Status	Not Specified	Not Specified	Not Specified	Not Specified	✓

As shown in Table 1, while previous studies have developed meeting management systems with some digital features, none have simultaneously integrated physical room booking, real Zoom meeting creation, and automatic Google Calendar synchronization within a unified prototype. This study addresses that gap.

To address these gaps, this study aims to design and develop a web-based prototype of an integrated meeting management information system for internal use at PT Ascon Inovasi Data. The prototype is built using Laravel 12 and MySQL, running on a local development environment. The system implements real API integration with Zoom and Google Calendar using OAuth 2.0 authentication. Meeting links are generated via Zoom's API, and events are automatically created in participants' Google Calendars. All API calls are functional when the system runs on localhost. Key features include meeting scheduling, room booking with conflict prevention, digital minutes (rich text, images, PDF uploads), internal notifications, and separate dashboards for users and administrators.

The scope of this study is limited to prototype design and functional validation in a local development environment. The system has not yet been deployed to a public domain due to infrastructure constraints. External meetings may still be held via shared links without requiring external user accounts. The development method follows the prototyping model by Pressman, which allows iterative feedback—although only one iteration was completed within this study due to time constraints. Usability is evaluated using the System Usability Scale (SUS) with a representative sample of employees.

II. RESEARCH METHOD

A. Prototype Development Stages

The prototyping model is a method used to develop a system by creating an initial version of the product, allowing users to obtain a detailed and specific understanding of how the system will function [6]. The prototyping approach serves as a model to simulate the prospective software application intended for the client, thereby facilitating their comprehension and ability to select a solution that fulfills their specific requirements. The utilization of this model effectively resolves client misinterpretations concerning technical complexities, ensuring that the specification requirements articulated by the client are comprehensively understood and elucidated by the software developers [7]. The prototype method applied in this study adopts five main phases based on the model proposed by Roger S. Pressman, namely: (1) communication, (2) quick planning, (3) quick design modeling, (4) prototype construction, and (5) deployment and feedback. The implementation of each phase in the context of system development at PT Ascon Inovasi Data is described as follows. The steps of the prototyping method are shown in Figure 1.

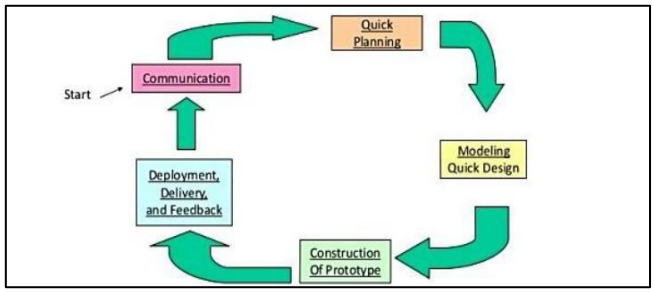


Fig. 1. Stages of the Prototyping Method [8]

1) Communication

The preliminary stage of requirement elicitation between the developer and the client entails a collaborative definition of overarching project goals, identified prerequisites, and an outline of future required components. This process facilitates the discovery of more granular and in-depth requirement specifications [7]. Initial requirements were gathered through interviews and observations with key stakeholders at PT Ascon Inovasi Data, including the operations manager and two team leads. Existing meeting practices (scheduling, room booking, minutes recording) were documented. This process involved interviews and observations of the existing meeting management practices [9].

2) Quick Planning

Based on the requirements, a set of core features was defined: meeting scheduling (online/offline), room booking, minutes taking (with text, image, PDF support), internal notifications, and administrative management of users, rooms, divisions/teams, and company profiles. In addition, system actors were defined, consisting of admin and user roles, along with their respective access rights [10].

3) Modeling (Quick Design)

At this stage, a visual representation of the system was designed prior to the coding phase. The quick design modeling consists of three main aspects:

- User Interface Design: Designed using Figma to visualize system workflows and interactions;
- Database Design: Designed using MySQL with 13 tables (e.g., users, meetings, rooms, summaries, notifications);
- System Architecture: A client-server architecture with server-side logic implemented in Laravel 12 and a MySQL database managed via phpMyAdmin. The system runs on a local development environment (not deployed to a public domain).

4) Construction of Prototype

Integration with third-party services (Zoom and Google Calendar). The system implements real API integration using OAuth 2.0. For Zoom, the system calls the Zoom API endpoint (https://api.zoom.us/v2/users/me/meetings) to create a new meeting, generating a valid join URL. For Google Calendar, the system uses the Google Calendar API to create events and add them to each participant's calendar. Both integrations are fully functional when the application runs on a local development environment (localhost). API credentials (client ID, client secret, access tokens) are stored in the application's .env file. Token refresh mechanisms are implemented to maintain continuous access. The only limitation is that the system has not been deployed to a public domain, which prevents full testing of webhooks and push notifications. Production deployment is planned for future work.

5) Deployment and Feedback

The developed prototype is then tested by users to obtain feedback[11]. The developed prototype was demonstrated to 8 employees of PT Ascon Inovasi Data (representing 44% of the workforce). Due to time constraints, only one complete iteration of the prototyping cycle was conducted. The feedback collection process proceeded as follows:

- Feedback format: Semi-structured interviews combined with task-based observation. Each participant performed three key tasks (schedule meeting with room booking, invite participants, create minutes with attachment) while being observed by the researcher.
- Types of feedback collected: Usability issues (e.g., "the room selection dropdown is not clearly labeled"), feature requests (e.g., "add PDF export for minutes"), and API performance observations (e.g., "Zoom link generation took approximately 3 seconds").
- Incorporation status: Due to the one-iteration constraint, feedback was documented and prioritized but not yet implemented in the prototype. The most critical issues (e.g., clarifying UI labels) are scheduled for the next iteration.
- API testing scenario on localhost: All API integrations (Zoom and Google Calendar) were tested using Oauth 2.0 authorization code flow. Testing involved:
    - Creating 10 test meetings with varying participant counts (2–10 participants)
    - Verifying that Zoom meeting URLs were generated and accessible
    - Confirming that Google Calendar events appeared in each participant's calendar (checked via Google Calendar web interface)
    - Testing token refresh by manually expiring tokens and observing automatic refresh
- Localhost limitation: Because the system runs on localhost (not a public domain), Google OAuth redirect URIs were configured to http://localhost:8000/callback. This works for development but prevents testing of webhooks and push notifications that require public endpoints. Production deployment would resolve this limitation.

B. Requirements Analysis

Based on the interview results obtained during the communication phase, a total of 12 functional requirements were identified for the system. These requirements are summarized in the form of use cases in Table 2, which categorizes system actors into two main roles: User (general employees of PT Ascon Inovasi Data) and Admin (users with the highest level of access responsible for overall system management). Each functional requirement is labeled with an identification code (F1 to F12) to facilitate tracking and verification during the testing phase.

Functional requirements delineate the specific processes and operational services that a system is obligated to deliver, encompassing its expected reactions to given inputs and its predefined behavior under specific conditions. These represent the fundamental, mandatory capabilities of the system [12]. In contrast, non-functional requirements constitute auxiliary specifications that operate independently of direct input-process-output mechanisms [13].

In addition to functional requirements, several non-functional requirements were also identified, including: (1) the system must be responsive and accessible across various screen sizes, (2) the main page load time must be under three seconds, (3access to minutes and attachments shall be restricted using role-based access control, and (4) the system must be usable by individuals with moderate technological literacy without requiring intensive training.

Table 2. Functional Requirements

ID	Functional Requirement	Actor
F1	User authentication through login/logout with role-based verification (Admin/User)	Admin, User
F2	Creation of new meetings (online/offline) with integration to Zoom/Google Meet via API	Admin, User
F3	Automatic synchronization of meeting schedules to each participant’s Google Calendar	Admin, User
F4	Reservation of meeting rooms for offline meetings, including time availability validation	Admin, User
F5	Creation and editing of meeting minutes with support for rich text, image uploads, and PDF attachments	Admin, User

F6	Display of all company meetings within a shared calendar interface	Admin, User
F7	Reception of notifications (simulated) for invitations, schedule changes, and reminders	Admin, User
F8	Management of user data (create, update, delete, search) by the administrator	Admin
F9	Management of room data (create, update, delete, status modification) by the administrator	Admin
F10	Management of organizational structure, including divisions, teams, and member assignments by the administrator	Admin
F11	Management of company profile information (name, logo, working hours) by the administrator	Admin
F12	Monitoring of all meeting-related activities through a centralized logging system by the administrator	Admin

C. Architecture System

The system follows a client–server architecture. The server side is implemented using Laravel 12 (PHP) with a MySQL database, running on a local development environment (not deployed to a public domain). All business logic, authentication, and data persistence are handled on the server. The client side consists of standard web technologies (HTML, CSS, JavaScript, Bootstrap) accessible through a web browser.

For third-party services (Zoom and Google Calendar), real API integration is implemented. The system performs actual HTTP requests to Zoom and Google Calendar APIs using OAuth 2.0 authentication. When a user schedules an online meeting, the system creates a real Zoom meeting via the Zoom API and adds the event to each participant's Google Calendar via the Google Calendar API. All API calls are functional on localhost. Deployment to a public domain is required for full production readiness but is outside the scope of this study.

D. Database Design

A database constitutes a logically coherent collection of data and its corresponding metadata, engineered to fulfill the informational needs of an organization. It serves as a large-scale data repository facilitating concurrent access by multiple users, encapsulating both the organizational operational data and the inherent descriptive data [14]. The database design was implemented using MySQL as the DBMS and phpMyAdmin as the administration tool.

Table 3. Databases Design

No	Table Name	Description
1	Agendas	Transaction
2	Bookings	Transaction
3	Companies	Master
4	Divisions	Master
5	Integrations	Transaction
6	Meetings	Transaction
7	Meeting_participants	Transaction
8	Notifications	Transaction
9	Positions	Master
10	Rooms	Master
11	summaries	Transaction
12	Team_user	Transaction
13	users	Master

E. Usability Evaluation using System Usability Scale (SUS)

To evaluate the usability of the proposed system, this study employs the System Usability Scale (SUS), a widely used and reliable method for measuring user perceptions of system usability. SUS is recognized for its simplicity, effectiveness, and ability to provide quick usability assessments even with a relatively small number of respondents [15]. SUS consists of 10 standardized statements with a five-point Likert scale ranging from “Strongly Disagree” to “Strongly Agree.” The questionnaire alternates between positive and negative statements to reduce response bias and ensure balanced evaluation. This method has been extensively applied in various information system studies due to its validity and reliability in capturing user experience [16].

Participants. PT Ascon Inovasi Data has a total of 18 employees across five divisions. Eight respondents (44% of the workforce) were selected purposively to represent each division: Engineering (2), Sales (2), Operations (2), Finance (1), and Human Resources (1). All were active employees who regularly participate in meetings.

Justification of sample size and representativeness. Although a sample of eight respondents (n=8) is relatively small for statistical generalization, it is considered acceptable for exploratory usability studies in small organizations. According to Sauro and Lewis [18], a sample of 8–12 users can detect approximately 80–85% of usability problems. Furthermore, the eight respondents represent 44% of PT Ascon Inovasi Data's total workforce (18 employees), and the purposive sampling strategy ensured coverage of all five divisions: Engineering (2), Sales (2), Operations (2), Finance (1), and Human Resources (1). This distribution mirrors the actual proportional composition of the company. However, the small sample size limits the ability to perform subgroup analysis (e.g., comparing usability perceptions across divisions) and reduces the precision of the SUS score confidence interval. Therefore, the SUS score should be interpreted as an indicative rather than definitive measure of usability.

Procedure. Each respondent performed three key tasks on the prototype: (1) schedule a new meeting (online) with room booking, (2) invite participants from another division, and (3) create meeting minutes with an image attachment. After completing the tasks, they filled out the SUS questionnaire. No time constraints were imposed, and assistance was not provided.

Scoring. SUS scores were calculated using the standard formula: for odd-numbered items, score = response – 1; for even-numbered items, score = 5 – response. The sum of adjusted scores was multiplied by 2.5 to obtain a final score from 0 to 100. An average score above 68 is considered above-average usability.

Interpretation of the SUS score. According to Bangor, Kortum, and Miller [19], SUS scores can be categorized as follows: 0–25 (worst imaginable), 26–39 (poor), 40–52 (ok), 53–73 (good), 74–85 (excellent), and 86–100 (best imaginable). An average score of 77.5 (as will be presented in the Results section) falls into the "excellent" category using Bangor's scale, or "good" using the more conservative scale proposed by Sauro [20] (where 68–80 is considered good).

This indicates that users perceive the system as easy to use and well-integrated. However, the score does not guarantee improved productivity or meeting coordination—these require objective measurement in a production environment.

Because the system is a prototype and has not been deployed in production, formal User Acceptance Testing (UAT) was not performed. Claims about improved meeting efficiency are not made; the SUS results only reflect users' perceptions of usability.

Limitations of SUS in this study. While SUS is a reliable instrument for measuring perceived usability, it has several limitations in the current context. First, SUS captures subjective user perceptions after a single, supervised interaction with the prototype; it does not measure long-term usability or learning effects over repeated use. Second, SUS does not provide diagnostic information about specific usability problems (e.g., which interface elements caused confusion); such insights were obtained separately through observation and interviews. Third, because the prototype was demonstrated in a controlled environment rather than deployed in daily work, participants may have exhibited a Hawthorne effect (i.e., modified behavior due to awareness of being observed). Finally, SUS does not measure objective efficiency metrics such as task completion time or error rate, which are recommended for future work.

III. RESULT

A. Implementation Outcomes

The prototype was successfully built using Laravel 12 and MySQL, running on a local development environment. All 12 functional requirements (F1–F12) were implemented. Key features include:

1) API Integration (fully functional on localhost)

The system successfully integrates with Zoom and Google Calendar APIs using OAuth 2.0. Zoom meetings are created via the Zoom API endpoint, generating valid join URLs that can be used for real online meetings. Google Calendar events are automatically created in each participant's calendar, as verified during local testing. Figure 2 shows a screenshot of a successfully created Google Calendar event from the system. The integration code is complete and functional; the only missing piece is deployment to a public domain to enable external callbacks and webhooks.

2) Meeting and Calendar Management

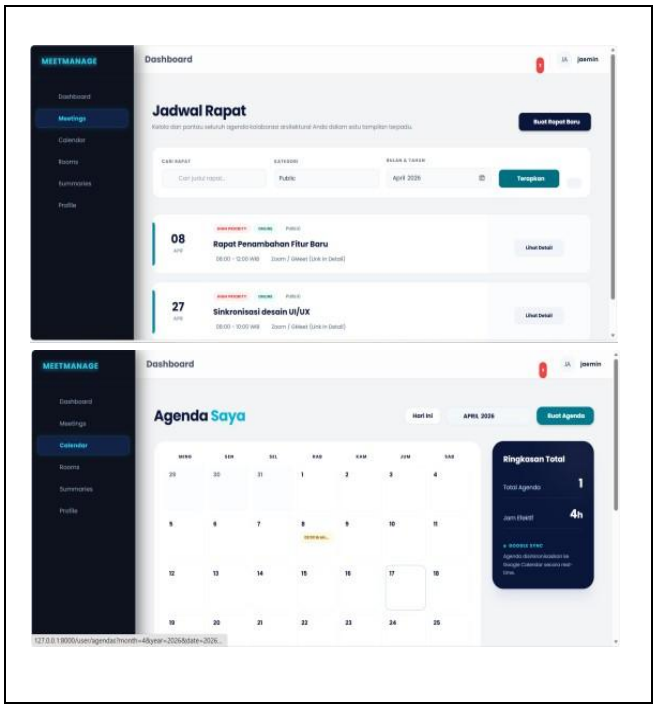


Fig. 2. User Interface Meeting and Calendar Management

The prototype provides two calendar views to support scheduling. The My Schedule view displays meetings in which the logged-in user participates, implemented using an Eloquent many-to-many relationship between the users and meetings tables via the meeting_participants pivot table. Meanwhile, the Company-Wide Schedule view presents all meetings across divisions and teams without filtering, thereby improving schedule transparency and reducing potential time conflicts.

3) Room Booking System

The room booking feature for offline meetings is implemented with a double-booking prevention mechanism. When users select a date and time within the meeting creation form, the system simultaneously queries the rooms and meetings tables to determine availability. Only rooms with an "active" status and no existing meetings scheduled within the same time range are presented as selectable options. In addition, server-side validation is enforced at the controller level prior to data persistence to prevent potential data manipulation from the client side.

4) Minutes

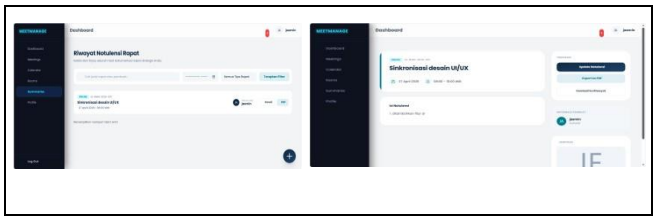


Fig. 3. User Interface Minutes

The minutes-taking feature is designed to comprehensively document meeting outcomes by supporting three types of content. First, formatted text is facilitated through a WYSIWYG editor (integrated TinyMCE), which enables rich text formatting such as bold, italics, bullet lists, and simple tables. Second, image uploads are supported, allowing users to upload files in .jpg or .png formats; these

images are stored in the storage/app/public directory and displayed inline within the minutes content. Third, PDF attachments can be uploaded and presented as downloadable links accessible to other meeting participants.

5) Division and Team Structure Management

Administrators are provided with the capability to manage the company's organizational structure hierarchically through a dedicated interface. This functionality includes the creation, modification, and deletion of divisions (e.g., Engineering Division, Sales Division), as well as the management of teams within each division (e.g., Frontend Team, Backend Team, DevOps Team under the Engineering Division). In addition, the system supports the assignment of users to specific teams through a multi-select form mechanism.

Furthermore, when creating a new meeting, users are able to invite participants not only on an individual basis but also collectively by selecting entire teams or divisions. This feature significantly reduces administrative effort and ensures that all relevant members are included in the meeting invitation without omission.

6) Internal Notification System

The prototype incorporates an internal notification mechanism designed to capture and communicate key events related to meetings. These events include new meeting invitations, schedule changes (such as rescheduling or cancellations), reminders issued 15 minutes prior to the start of a meeting, and updates to meeting minutes, whether newly added or edited. At the prototype stage, the notification system is implemented through two primary components. First, notification data are stored in a dedicated notifications table within the MySQL database. Second, notifications are presented in the user interface via a bell icon, which is periodically refreshed using AJAX polling at 30-second intervals. This approach ensures that users receive near real-time updates while maintaining a relatively simple implementation suitable for a development environment.

7) Administrator Dashboard

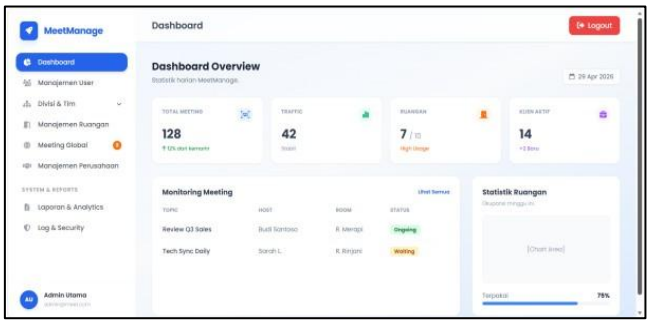


Fig. 4. Administrator Dashboard

The administrator dashboard is designed to provide full control over system configuration and activity monitoring. It comprises five main modules. The Meeting Activity module presents a comprehensive log of all meetings that have been created, including information on the creator, creation time, number of participants, status, and change history; the log can be filtered by division, team, or time range. The User Management module offers a full CRUD (Create, Read, Update, Delete) interface for managing user data, including password reset functionality and role assignment (admin/user). The Room Management module enables administrators to manage room data through CRUD operations, covering attributes such as room name, capacity, location, and active status. The Division and Team Management module provides a hierarchical tree interface to manage organizational structure and member assignments. Finally, the Company Profile module allows administrators to update organizational information, including the company name, logo upload, and the configuration of default working hours, which serve as a reference when scheduling new meetings.

8) System Evaluation Results

To quantitatively assess the usability of the proposed system, the System Usability Scale (SUS) method was employed. A total of 8 participants (represents 44% of the company's workforce.) were involved in the evaluation process, where each participant interacted with the system and completed the SUS questionnaire.

The SUS score is calculated using the following formula:

$$SUS = (\sum_{I=1}^{10} S_i \times 25)$$

where:

- S_i represents the adjusted score for each question
- For odd-numbered questions: score = response − 1
- For even-numbered questions: score = 5 − response

The final SUS score ranges from 0 to 100. Detailed individual scores are omitted for brevity but were used in the calculation of the overall SUS score. The average SUS score obtained from all respondents is 77.5, indicating that the system demonstrates an above-average level of usability. Based on the evaluation results using the System Usability Scale (SUS), the system achieved an average score of 77.5, as illustrated in Figure 5. This value is above the standard threshold of 68 and falls into the "Good" category, indicating that the system has a high level of usability and can be used effectively without requiring intensive training.

Users were able to understand and perform key features such as meeting scheduling, room booking, and meeting minutes management effectively. The integration of multiple functions within a single platform also contributes to improved work efficiency and reduced operational complexity compared to conventional methods.

However, some challenges were identified during the initial use, particularly related to understanding certain interface elements and system responsiveness. Overall, the system demonstrates strong potential for implementation in supporting integrated meeting management processes.

The SUS evaluation results are further presented in a bar chart to provide a clear visualization of the distribution of usability scores across all respondents.

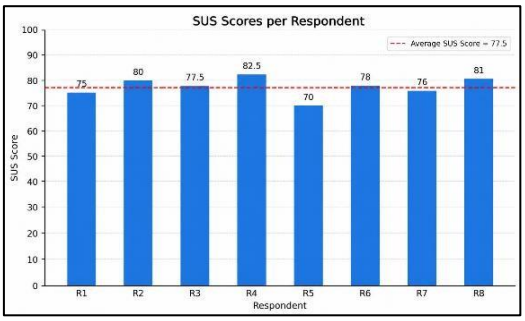


Fig. 5. Distribution of Respondents' SUS Scores

IV. DISCUSSION

A. Comparison with Conventional Practices

Current practices at PT Ascon Inovasi Data rely on spreadsheets and instant messages, leading to scheduling conflicts and lost minutes. The proposed prototype offers a unified platform that combines scheduling, room booking, minutes, and notifications. Compared to using separate tools (Google Calendar + Zoom + chat), the prototype reduces context switching and centralizes meeting records. However, because no objective time-motion study was conducted, the magnitude of improvement remains unquantified.

B. Challenges for Real Implementation

Transitioning from the current prototype to a full production system will require addressing several technical challenges. First, real integration with Zoom and Google Calendar APIs demands a robust OAuth 2.0 mechanism, including secure token management, automatic refresh of expired tokens, rate limiting handling, and comprehensive error recovery. Second, deployment infrastructure must be established, such as procuring a public domain, enabling HTTPS with a valid TLS certificate, and provisioning a production server (e.g., AWS or DigitalOcean). Third, security measures need to be enhanced: implementing encryption at rest (e.g., AES-256 for meeting minutes and attachments), ensuring HTTPS with TLS 1.3 for data in transit, and maintaining protection against SQL injection (although Laravel's Eloquent ORM already mitigates this risk). Fourth, while the company is small, a queue system (e.g., Redis combined with Laravel's built-in queue worker) may still be necessary to handle email and notification delivery without degrading user experience. Finally, compliance with applicable regulations, particularly Indonesia's Personal Data Protection Law (UU PDP), must be ensured, as meeting minutes may contain sensitive organizational information.

1) Potential for Increasing Team Coordination Effectiveness

Existing literature suggests that digitalized meeting management can reduce preparation time and improve attendance through automated notifications [1,2]. The shared calendar feature (Company-Wide Schedule) implemented in this prototype is analogous to features that have been associated with improved coordination in prior work. However, because no objective measurements were conducted in this study, these potential benefits remain theoretical. Future research should empirically test whether the prototype actually improves coordination metrics such as meeting preparation time, attendance rates, or action item completion. Therefore, the system does not 'demonstrate' enhanced effectiveness; rather, it incorporates design elements that prior studies have found promising.

2) Limitations of the Current Study

This study has several limitations, which are primarily related to the deployment status rather than the functional completeness of the system. First, the system has not been publicly deployed and is still running in a local development environment (localhost). As a result, real-world testing involving HTTPS, OAuth callbacks, webhooks, and concurrent user access under production conditions could not be conducted. Second, the sample size used in this study is relatively small (n=8). Although this represents 44% of the company's workforce, it is still insufficient for broader statistical generalization. Third, the evaluation relies solely on the System Usability Scale (SUS), which measures perceived usability; objective efficiency metrics such as time savings per meeting, reduction in errors, or improvement in attendance were not measured. Additionally, formal User Acceptance Testing (UAT) was not performed due to the lack of system deployment, making it infeasible to involve all employees.

From a security perspective, the system currently implements only role-based access control without encryption at rest. The implementation of data encryption, such as AES-256 for stored meeting minutes and attachments, is planned for future production deployment. Furthermore, several production-level features are still missing in the prototype, including PDF/DOCX export functionality, mobile-responsive design, Single Sign-On (SSO), and advanced reporting capabilities. Lastly, the development process only completed one iteration of the prototyping model, meaning that user feedback has not yet been fully incorporated into subsequent refinements. Importantly, these limitations do not affect the core objective of the study, which is to evaluate usability and demonstrate the feasibility of the proposed system design.

3) Security Considerations for Production Deployment

While the prototype successfully implements OAuth 2.0 for Zoom and Google Calendar integration, several security enhancements are required before public deployment. First, token storage: Currently, OAuth tokens (access and refresh tokens) are stored in the database without encryption. In production, tokens must be encrypted using AES-256 to prevent exposure in the event of a database breach. Second, HTTPS enforcement: The prototype runs on HTTP (localhost). Public deployment requires TLS 1.3 with valid certificates (e.g., Let's Encrypt) to protect OAuth authorization codes and access tokens during transmission. Third, token refresh security: The automatic refresh mechanism should be protected against race conditions and should invalidate old tokens immediately upon refresh. Fourth, OAuth redirect URI validation: The system must strictly validate redirect URIs to prevent open redirect attacks. Fifth, rate limiting: API calls to Zoom and Google Calendar should be rate-limited to avoid exceeding quota limits (Zoom allows 100 requests per second; Google allows 1,000,000 queries per day). Sixth, sensitive data in minutes: Meeting minutes may contain confidential business information; therefore, attachments should be encrypted at rest (AES-256)

and access logged for audit purposes. A summary of security readiness is provided in Table 4.

Table 4: C Security Readiness Assessment for Production Deployment

Security Aspect	Current Status (Prototype)	Required for Production	Priority
OAuth token encryption at rest	×	✓	High
HTTPS/TLS	×	✓	High
OAuth redirect URI validation	✓	✓	Medium
Token refresh security	×	✓	Medium
Rate limiting for external APIs	×	✓	Medium
Encryption at rest for attachments	×	✓	Medium
Audit logging for minutes access	×	✓	High
SQL injection protection	✓	✓	Low
XSS protection	Partial (TinyMCE Sanitizaion)	✓	Medium

Production Readiness Assessment. Based on Table 2, the prototype is functionally complete but not yet production-ready from a security perspective. The core OAuth 2.0 integration works correctly on localhost, but the absence of encryption at rest and HTTPS makes it unsuitable for deployment. Approximately 4–6 weeks of additional development (including security hardening, public domain setup, and SSL certificate configuration) would be required before the system could be safely deployed for organizational use.

V. CONCLUSION

This study successfully designed and developed a functional web-based integrated meeting management information system for PT Ascon Inovasi Data using Laravel 12 and MySQL. All 12 functional requirements were implemented, including meeting scheduling with real Zoom API integration, room booking with conflict prevention, multimedia minutes, internal notifications, and an administrative dashboard. The system also integrates with Google Calendar API via OAuth 2.0, automatically creating events in participants' calendars. All API integrations are fully functional on the local development environment. Usability testing using the System Usability Scale (SUS) with eight representative employees (44% of the workforce) yielded an average score of 77.5, indicating "Good" perceived usability.

The main limitation is that the system has not yet been deployed to a public domain. This prevents full production testing but does not affect the functional integrity of the API integrations. Future work will focus on public deployment (with HTTPS), formal UAT, objective efficiency measurements (e.g., pre-post comparison of meeting preparation time), responsive design, PDF export, encryption at rest, and SSO integration. Once deployed, the system is ready to support meeting management at PT Ascon Inovasi Data.

REFERENCES

[1] M. A. F. Mohamad Johari And A. F. Arbain, "Transforming Workplace Efficiency: Using Agile Development Methodologies For Developing Meeting Management System," *International Journal Of Innovative Computing*, Vol. 14, No. 2, Pp. 117–127, Nov. 2024, Doi: 10.11113/Ijic.V14n2.505.

[2] C. Blankenberg, B. Gebel-Sauer, And P. Schubert, "Using A Graph Database For The Ontology-Based Information Integration Of Business Objects From Heterogenous Business Information Systems," In *Procedia Computer Science*, Elsevier B.V., 2021, Pp. 314–323. Doi: 10.1016/J.Procs.2021.12.019.

[3] S. H. Rachmawati, I. Arwani, And W. Purnomo, "Pengembangan Sistem Informasi Manajemen Rapat Berbasis Web (Studi Kasus : Pt Sea Mobile Indonesia)," 2022. [Online]. Available: Http://J-Ptiik.Ub.Ac.Id

[4] Shofa Zahra Arthatia And Mubassiran, "Perancangan Sistem Informasi Rapat Pengelolaan Kegiatan Pilar (Studi Kasus: Divisi Synergy Business Alliance Dan Tanggung Jawab Sosial Lingkungan Pt Pos Indonesia Persero)," 2023.

[5] I. Sari, N. Hadinata, P. Studi Sistem Informasi Fakultas Ilmu Komputer, J. A. Jenderal Yani No, And S. Selatan, "Pengembangan Sistem Informasi Agenda Rapat Kecamatan (Siarke) Menggunakan Metode Prototype (Studi Kasus Kecamatan Tulung Selapan)," 2023.

[6] I. Faizan, E. A. R. Kasim, M. J. Salam, Lm. F. Aksara, And Lm. B. Aksara, "Rancang Bangun Sistem Informasi Notulensi Rapat Menggunakan Metode Prototype," *Semantik : Teknik Informasi*, Vol. 9, No. 2, P. 83, Dec. 2023, Doi: 10.55679/Semantik.V9i2.44149.

[7] S. Dhina Pohan *Et Al.*, "Pengembangan Sistem Informasi Akademik Berbasis Web Menggunakan Metode Prototype Pada Sekolah Menengah Pertama (Smp) Advent Kotamobagu," 2024. [Online]. Available: Http://Ejournal.Stmik-Time.Ac.Id

[8] M. Alda, "Pengembangan Aplikasi Pengolahan Data Siswa Berbasis Android Menggunakan Metode Prototyping," *Jurnal Manajemen Informatika (Jamika)*, Vol. 13, No. 1, Pp. 11–23, Jan. 2023, Doi: 10.34010/Jamika.V13i1.8216.

[9] Ragel Trisudarmo, "Penerapan Metode Prototype Dalam Sistem E-Government Pada Pelayanan Administrasi Kependudukan," 2022.

[10] I. P. Sari, O. K. Sulaiman, A.-K. Al-Khowarizmi, And M. Azhari, "Perancangan Sistem Informasi Pelayanan Masyarakat Pada Kelurahan Sipagimbar Dengan Metode Prototype Berbasis Web," *Blend Sains Jurnal Teknik*, Vol. 2, No. 2, Pp. 125–134, Aug. 2023, Doi: 10.56211/Blendsains.V2i2.288.

[11] M. Syarif And D. Risdiansyah, "Pemanfaatan Metode Prototype Dalam Perancangan Sistem Informasi Penjualan Berbasis Website," 2024.

[12] K. Surya Ningsih, N. Jamilah Aruan, And A. Taufik Al Afkari Siahaan, "Aplikasi Buku Tamu Menggunakan Fitur Kamera Dan Ajax Berbasis Website Pada Kantor Dispora Kota Medan," 2022.

[13] F. Elfaladonna And I. Mareta, "Aplikasi Arsip Digital (E-Arsip) Data Pegawai (Studi Kasus: Pt. Bukit Asam Kreatif)," 2023.

[14] C. Efendi *Et Al.*, "Manajemen Database Sistem Organisasi Dakwah," 2023.

[15] R. Yani, I. Nazhifah, And M. I. Pradika, "System Usability Scale In Information System Application Development Using Systematic Mapping Study," Vol. 2, No. 2, Pp. 130–140, 2025, Doi: 10.57152/Ijatis.V2i2.2275.

[16] M. Nasyiah, B. Kelana, And A. Riskinato, "System Usability Scale For Measuring Usability Of Social Network Applications From User Perspectives," In *E3s Web Of Conferences*, Edp Sciences, Jan. 2024. Doi: 10.1051/E3sconf/202448303010.