Weekend Saver

Simplifying and automating weekend tasks to reduce stress and enhance work-life balance for project manager

Phase III

MIS 6308-SAPM-Fall 2024

Professor Srinivasan Raghunathan

Prepared By – Varmi Ashokkumar Sanghani

Contents

Executive Summary

Our group project focuses on designing a feature within project management apps that helps ease the tasks of project managers over the weekends. Our research on Gen AI engines indicates that many project managers work about 50 hours a week, with weekend work being quite common. This leads to disruptions in work-life balance, which is a sign of poor work culture and negatively impacts overall company performance eventually.

A major portion of a typical project manager's weekend tasks involves reviewing and updating project plans, including updating the project timeline, reviewing task priorities, and identifying potential risks related to the project. They also prepare weekly updates, summarizing the progress of the project, highlighting key accomplishments, outlining the next steps, and performing many similar tasks. Additionally, they plan the upcoming week of their project by identifying the next goals, outlining key tasks and deliverables for the week, and scheduling high-priority tasks. Besides these tasks, they set clear boundaries by allocating specific hours for tasks and ensuring time for personal activities and rest.

Recognizing the evolving needs of managers in today's fast-paced corporate world, we propose the Weekend Saver feature. This feature aims to automate major tasks for managers that usually fills up their weekends. These includes Task Prioritization and Delegation, Email organization, Intelligent scheduling and coordination, and Document generation and sharing. We aim to integrate this feature into MS Project to enable fast and seamless project management. Currently, most of these tasks are performed manually, leading to overwork and frustration. By automating these processes, our project aims to promote a healthier work-life balance for managers, reduce stress, and foster a more efficient and positive work culture within organizations.

Problem Statement

Problem

Project managers often find themselves spending a huge number of weekends on routine tasks such as prioritizing activities, scheduling meetings, organizing emails, and preparing reports and presentations. These are time-consuming and repetitive tasks that eventually lead to disruption in their work-life balance. Having an improper work-life balance eventually leads to increased stress and reduced productivity. Furthermore, current tools like MS Project and JIRA which are widely used by project managers are effective, but they lack automation capabilities. As a result, project managers tend to spend most of their time doing tedious activities on these unautomated applications. Our aim in this project is to introduce a feature for applications like MS Project to help project managers to automate most of their tedious and manual work. This will help them plan weekend activities well and maintain their work-life balance.

Objectives

1. Task Prioritization and Delegation: Develop a feature that helps project managers to automate the task prioritization process and automatedly help them to delegate the task based on team members' availability.

2. Email Organization: Implement a feature that helps the project manager to automatedly organize and prioritize emails. Furthermore, this feature will also sort the emails into categories enabling project managers to easily maintain and respond to their emails.

3. Intelligent Scheduling and Coordination: This feature will provide suggested meeting times to the project managers based on team members' availability, enabling them to easily coordinate with the other members for important project meetings.

4. Document Generation and Sharing: Develop a feature that utilizes the project details stored in the database to prepare auto-generated reports and presentations. This will save time for project managers and help them generate and share their project documents with the stakeholders easily.

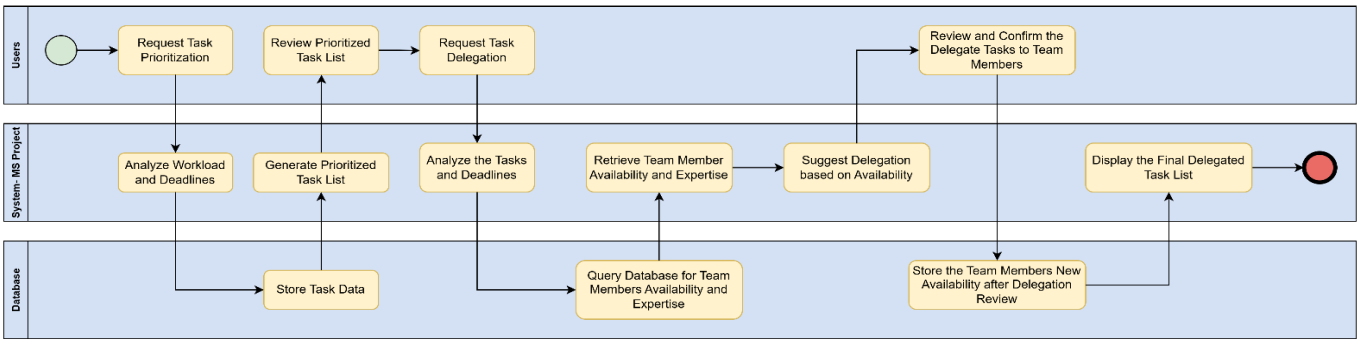
Scope

The scope of the proposed new feature in MS Project is as follow:

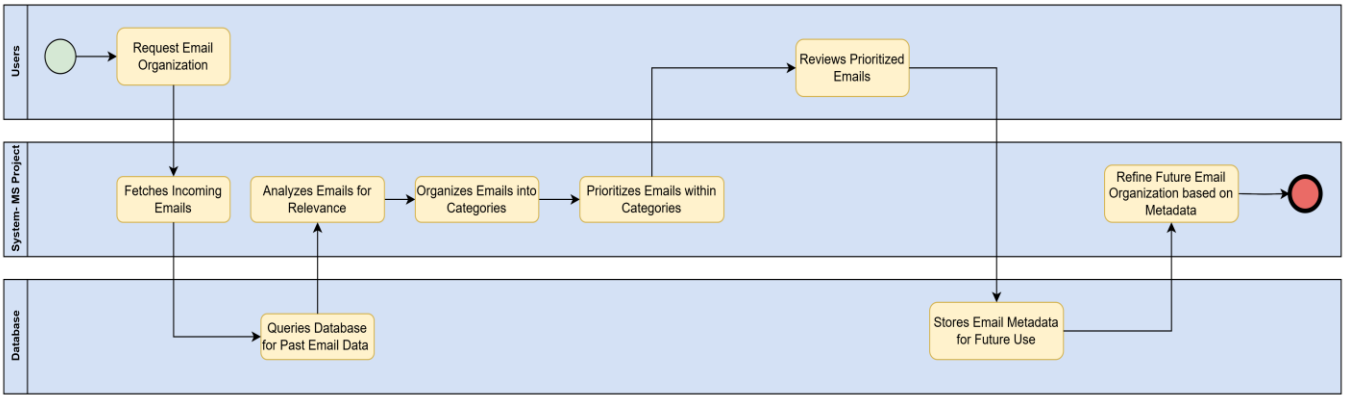
- The estimated cost for the development and integration of the feature is approximately $85,000.
- The development timeline for the project is estimated to be 8 months.
- The team will require developers, AI specialists, database administrators and project managers.
- The feature will rely on AI tools to automate tasks like email organization, task prioritization and such other tasks.
- The existing database will be enhanced to store metadata, task outputs and team members availability.

Business Process Model Notation (BPMN)

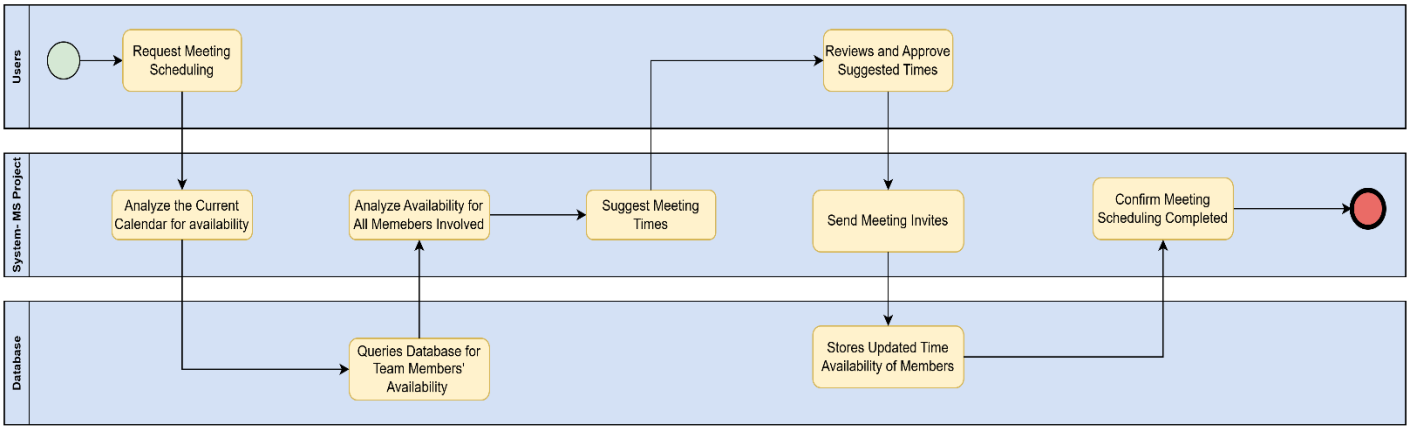
Task Prioritization and Delegation



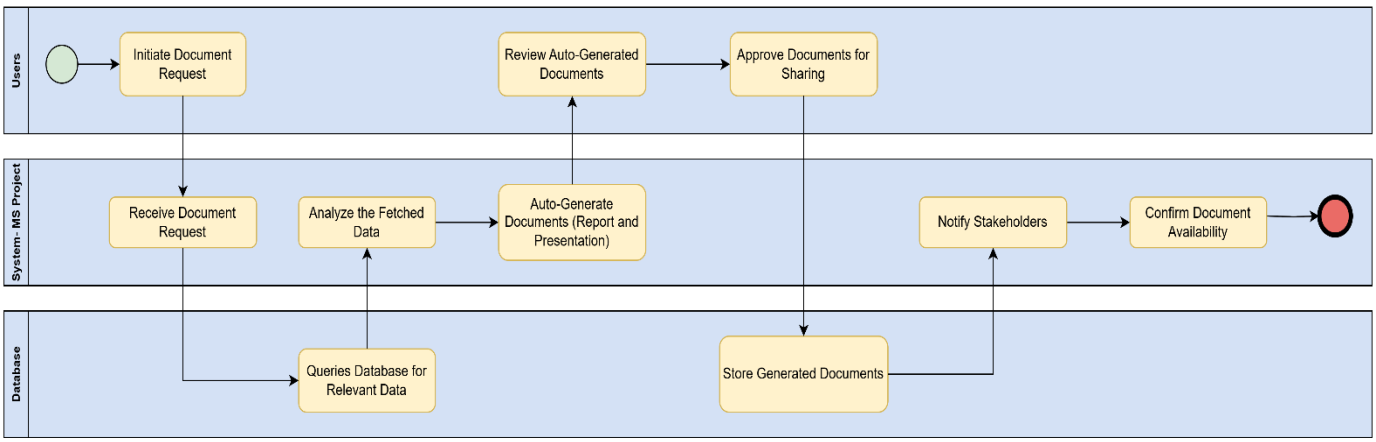
Email Organization



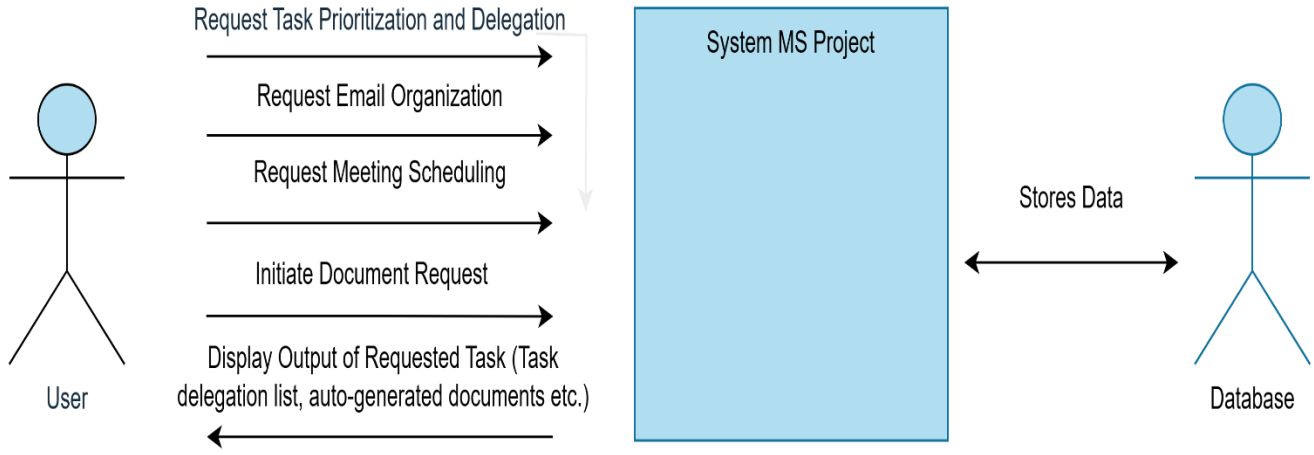
Intelligent Scheduling and Coordination



Document (Reports and Presentation) Generation and Sharing



Context Diagram



Use Case Diagram

Use Cases Descriptions

The description of use cases shown in the above diagram are given below:

Use Case Description #1:

Use Case Name: Access "Weekend Saver" Feature
Primary Actor: User
Stakeholders: MS Project, User (Project Manager), Organization
Brief Description: The user accesses the "Weekend Saver" feature to get access to automative tools.
Trigger: User selects the "Weekend Saver" feature from MS Project's toolbox.
Normal Flow of Events: 1. User navigates to and accesses the "Weekend Saver" feature in the toolbar. 2. MS Project displays the list of available automation features in the toolbox. 3. User select the specific automation feature based on their preference. 4. MS Project loads the feature and displays the related tools and options.
Exception Flow: - Feature Unavailable: This error will occur when the feature might be unsupported based on the version of MS Project used or the operating system being used.

Use Case Description #2:

Use Case Name: Query Database
Primary Actor: MS Project
Stakeholders: Database
Brief Description: MS Project Retrieved the data from the database based on the tasks requested by the user.
Trigger: MS Project initiates the query request when user requests to perform automated tasks under "Weekend Saver" feature.
Normal Flow of Events: 1. MS Project receives a data retrieval request based on the task selected by the user. 2. MS Project queries database using the provided request parameters based on the task selected. 3. The MS Project retrieves the requested data from the database and processes it. 4. The MS Project updates the project sheet and displays the retrieved data to the user.
Exception Flow: - Query Failure: MS Project displays an error message if a query fails to be executed and displays cached data if available.

Use Case Description #3:

Use Case Name: Prioritize and Delegate Tasks
Primary Actor: MS Project
Stakeholders: User (Project Manager), Organization, Database
Brief Description: MS Project automatedly prioritizes and delegates task on the project sheet when "Prioritize and Delegate Tasks feature is accessed.
Trigger: User selects the "Prioritize and Delegate Tasks" option from the toolbar.
Normal Flow of Events: 1. User navigates to "Prioritize and Delegate Tasks" option in the "Weekend Saver" feature. 2. The user requests to prioritize and delegate tasks. 3. MS project analyzes the task urgency, deadline and team members' availability availability and generates a prioritized task list. 4. The user reviews and confirms the prioritized task list. 5. MS Project stores the prioritized task list in the database. 6. The user requests to delegate tasks. 7. MS Project retrieves team members' availability and expertise data from the database. 8. MS Project recommends task delegation based on team members' availability and workload. 9. The user reviews and confirms the suggested task delegation.

10. The database stores the updated team members' availability. 11. MS Project displays the final prioritized and delegated task list.
Exception Flow: - Task prioritization and delegation error: MS Project will display an error if it is unable to prioritize tasks and delegate the tasks to team members. It will prompt the user to initiate task prioritization and delegation request again. - Task Rejection: The user rejects the task prioritization or task delegation suggestions, prompting the system to re-perform the task.

Use case Description #4:

Use Case Name: Organize and Sort Emails
Primary Actor: MS Project
Stakeholders: User (Project Manager), Organization, Database
Brief Description: Automated organizes and categorizes the emails based on their urgency.
Trigger: User selects the "Organize and Sort Emails" option from the toolbar.
Normal Flow of Events:

1. The user navigates and requests to "Organize and Sort Emails" from the “Weekend Saver” feature in the toolbar.
2. MS Project fetches a list of emails.
3. MS Project applies filters to sort the emails based on their priority and categorize them.
4. MS Project displays the organized and sorted emails.
5. The user reviews the organized and sorted emails and provides further suggestions if the emails need to be sorted in a more efficient way.
6. MS Project saves the email metadata for future email sorting tasks.

Exception Flow:

- Email Organization failure: MS Project displays an error message and shows the default unsorted list of emails.
- No Emails Found: The system displays an error message if there are no emails to organize and sort.

Use Case Description #5:

Use Case Name: Schedule and Coordinate Meetings
Primary Actor: MS Project
Stakeholders: User (Project Manager), Organization, Database

Brief Description: Automatedly schedules and coordinates meetings based on team availability and project deadlines.

Trigger: User selects the "Intelligent Scheduling and Coordination" option from the toolbar.

Normal Flow of Events:

1. The user selects the "Intelligent Scheduling and Coordination" option from the “Weekend Saver” feature in toolbar.
2. MS Project queries the database and retrieves information for team members’ availability.
3. MS Project analyzes the retrieved data and suggests meeting times based on task deadlines and team members’ availability.
4. The user reviews and approves one of options from the suggested meeting times.
5. MS Project sends the meeting invites to the participants’ email address.
6. MS Project stores the updated meeting schedule and revised team members’ availability in the database.

Exception Flow:

- Scheduling Failure: The feature may detect conflicting availability and might not be able to find a common meeting time when everyone is free before the teams’ “task” approaches the deadline. The feature will show the error message and prompt the user to request for meeting scheduling again.

Use Case Description #6:

Use Case Name: Generate and Share Documents
Primary Actor: MS Project
Stakeholders: User (Project Manager), Organization, Database
Brief Description: Automates the generation and sharing of project documents with other people.
Trigger: User selects the "Generate and Share Documents" option from the toolbar.
Normal Flow of Events: 1. The user selects the "Generate and Share Documents" options from the “Weekend Saver” feature in the toolbar. 2. The user initiates a request to create a document. 3. MS Project queries the database and retrieves relevant project data. 4. MS Project auto-generates the requested document based on the fetched data. 5. The user reviews the generated document for accuracy and completeness. 6. The user suggests or makes necessary changes if needed. 7. The user approves the final document for sharing. 8. MS Project stores the approved document in the database for future access. 9. MS Project sends the approved document to relevant stakeholders.

Exception Flow:

- Document Generation error: MS Project displays an error message and prompts the user to request for document generation again.
- Sharing error: MS Project displays an error message when the documents are not successfully shared with the stakeholders and prompts the user to retry sharing the documents.

Use Case Description #7:

Use Case Name: Review and Confirm Task Outputs
Primary Actor: User
Stakeholders: MS Project, User (Project Manager), Organization, Database
Brief Description: The user reviews and confirms the task outputs after a task of the "Weekend Saver" feature is completed. This feature allows the user to verify if all deliverables meet the expected quality and requirements.
Trigger: The user is prompted to review and confirm the completed tasks once a particular task is completed.
Normal Flow of Events: 1. The user selects a specific task to be performed from the "Weekend Saver" feature.

2. MS Project displays the completed task outputs (final deliverables, task duration, and status) for user's review.
3. The user reviews the task outputs to verify quality and completeness.
4. The user confirms the task outputs by approving them or sending feedback for revisions if necessary.
5. MS Project updates the task status to "Confirmed" or displays a new revised task output if revisions are needed.

Exception Flow:

- Review error: The user may not be able to review the task if the MS project displays an error and is unable to complete the task. The user will be prompted to initiate task request again.
- Confirmation error: MS Project may display an error if it is not able to confirm the final output and it will prompt the user the send confirm request again.

Use Case Description #8:

Use Case Name: Store Task Outputs and Metadata
Primary Actor: Database
Stakeholders: MS Project, User (Project Manager), Organization
Brief Description: The final outputs and confirmed tasks along with their metadata are stored in the database for future access.

Trigger: Database stores the final outputs of tasks and their metadata after user reviews and confirms the task.
Normal Flow of Events: 1. The user requests to perform specific task from the "Weekend Saver" feature. 2. MS Project displays the output after performing the task. 3. The user reviews and confirms the task output within the MS Project application. 4. The task output (final deliverables, task duration, and status) and its associated metadata (task ID, timestamps, and user notes) is stored in the database for future use and reference.
Exception Flow: - Storage Unavailable: The system displays this error when there is no storage left in the database.

Use Case Description 9#:

Use Case Name: Update Team Members' Availability
Primary Actor: Database
Stakeholders: MS Project, User (Project Manager), Team Members, Organization
Brief Description: The database updates the members' availability based on the roles assigned to members in various tasks of "Weekend Saver" feature.

Trigger: User confirms task delegation or confirms a meeting time.

Normal Flow of Events:

1. The user request to perform specific task from the “Weekend Saver” feature.
2. MS Project completes the requested tasks and displays the task output to the user.
3. The user confirms a task such as task delegation or meeting time on MS Project.
4. The MS Project saves the confirmed tasks in the system.
5. The database updates the team members' availability based on the confirmed and saved tasks in the MS Project for future task scheduling.

Exception Flow:

- Team Member Not Found: If a team member's details are not available or the system fails to retrieve the data, the user will be prompted to verify the member's information or contact support for troubleshooting.
- Database Error: In case of an error updating the database, the system will notify the user.

Data Dictionary

The data dictionary used is given below:

Use Case: Access “Weekend Saver” Feature

user = user id + email

automation features = feature name + 1{feature tool}n

toolbox = toolbox name + 1{tool}n

preference = preferred task

tools = tool name + tool category

tool category = ["Task Prioritization" | “Task Delegation” | "Meeting Scheduling" | "Email Organization" | "Report Generation" | "Powerpoint Generation" | "Document Sharing"]

options = 1{option}n

option = option name + (default setting) + (preference setting)

Use Case: Query Database

data retrieval request = task selection + request parameters

database = tables + records + relationships

request parameters = 1{parameter}n

requested data = requested query information + (associated resources) + (task progress data)

project sheet = sheet data + (updated retrieved data)

user = user id + email

Use Case: Prioritize and Delegate Tasks

user = user id + email

prioritize and delegate tasks = 1{prioritized task}n + 0{delegated team member}n

task urgency = [high | medium | low]

deadline = due date + (time)

team member's availability = team member's name + availability status

availability status = [availability | busy | tentative]

prioritized task list = 1{task}n

tasks = 0{task id + task name + task deadline + (assigned resources)}n

expertise data = worker's expertise name

database = tables + records + relationships

task delegation = task id + task name + 0{assigned team member}n

workload = task in progress + allocated hours

Use Case: Organize and Sort Emails

toolbar = 1{toolbox name}n

emails = sender + subject + content + timestamp + (priority status)

organized and sorted emails = email list + sorting criteria + categorization rules

email metadata = 1{email attribute}n + priority status + categorization rules

email sorting tasks = emails + priority status + (further suggestions) + categorization rules

priority status = [high | medium | low]

categorization rules = labels + folders

Use Case: Schedule and Coordinate Meetings

user = user id + email address

toolbar = 1{toolbox name}n

team member's availability = team member's name + availability status

availability status = [availability | busy | tentative]

retrieved data = requested query information + (associated resources)

meeting times = 0{suggested time slot}n+ priority status + (meeting agenda)

priority status = [high | medium | low]

task deadline = task name + task due date + priority status

meeting invites = meeting time + 1{participant names}n + 1{participant's email address}n

meeting schedule = meeting details + team member's availability

meeting details = meeting time + meeting agenda + 1{participant}n

Use Case: Generate and Share Documents

user = user id + email address

toolbar = 1{toolbox name}n

document = document type + project id + project name + content + (metadata)

document type = [Report | PowerPoint]

project data = project details + task details + (resource details)

project details = project name + project ID + project deadlines

task details = task ID + task name + task status + (priority status)

resource details = resource name + resource role + resource availability

generated document = document request + retrieved data + (suggested changes)

stakeholders = stakeholder name + email address + role

Use Case: Review and Confirm Task Outputs

user = user id + email address

task output = final deliverable + task duration + status

final deliverable = [Task prioritization | Task delegation | Generated Document | Organized and Sorted emails | Meeting confirmation] + (revised results)

task duration = time to complete task

task status = [pending | completed | in progress | confirmed | needs revision]

Use Case: Store Task Outputs and Metadata

user = user id + email address

task outputs = final deliverable + task duration + status

final deliverables = [Task prioritization | Task delegation | Generated Document | Organized and Sorted emails | Meeting confirmation] + (revised results)

task duration = time to complete task

task status = [pending | completed | in progress | confirmed | needs revision]

metadata = task id + timestamps + user notes

timestamps = creation date + last modified date

user notes = comment

Use Case: Update Team Members' Availability

tasks = 0{task id + task name + task deadline + (assigned resources)}n

task output = final deliverable + task duration + status

final deliverable = [Task prioritization | Task delegation | Generated Document | Organized and Sorted emails | Meeting confirmation] + (revised results)

task delegation = task id + task name + 0{assigned team member}n

meeting time = 0{suggested time slot}n+ priority status + (meeting agenda)

team member’s availability = team member’s name + availability status

availability status = [availability | busy | tentative]

user id = data element

email = data element

feature name = data element

feature tool = data element

toolbox name = data element

tool = data element

preferred task = data element

tool name = data element

tool category = data element

option = data element

option name = data element

sheet data = data element

prioritized task = data element

due date = data element

default setting = data element

preference setting = data element

task selection = data element

request parameters = data element

tables = data element

records = data element

relationships = data element

parameter = data element

requested query information = data element

associated resources = data element

task progress data = data element

updated retrieved data = data element

delegated team member = data element

time = data element

team member’s name = data element

task = aggregated data

task name = data element

assigned resources = data element

assigned team member = data element

allocated hours = data element

sender = data element

content = data element

priority status = data element

sorting criteria = data element

email attribute = data element

labels = data element

suggested time slot = data element

task due date = data element

participant name = data element

meeting details = aggregated data

document type = data element

project name = data element

availability status = data element

task id = data element

task deadline = data element

worker’s expertise name = data element

task in progress = data element

toolbox name = data element

subject = data element

timestamp = data element

email list = data element

categorization rules = aggregated data

further suggestions = data element

folders = data element

meeting agenda = data element

meeting time = data element

participant’s email = data element

participant = data element

project id = data element

content = data element

metadata = data element

task details = aggregated data

project deadline = data element

resource name = data element

resource role = data element

suggested changes = data element

role = data element

final deliverable = data element

status = data element

user notes = data element

last modified date = data element

project details = aggregated data

resource details = aggregated data

task status = data element

resource availability = data element

document request = data element

stakeholder name = data element

retrieved data = data element

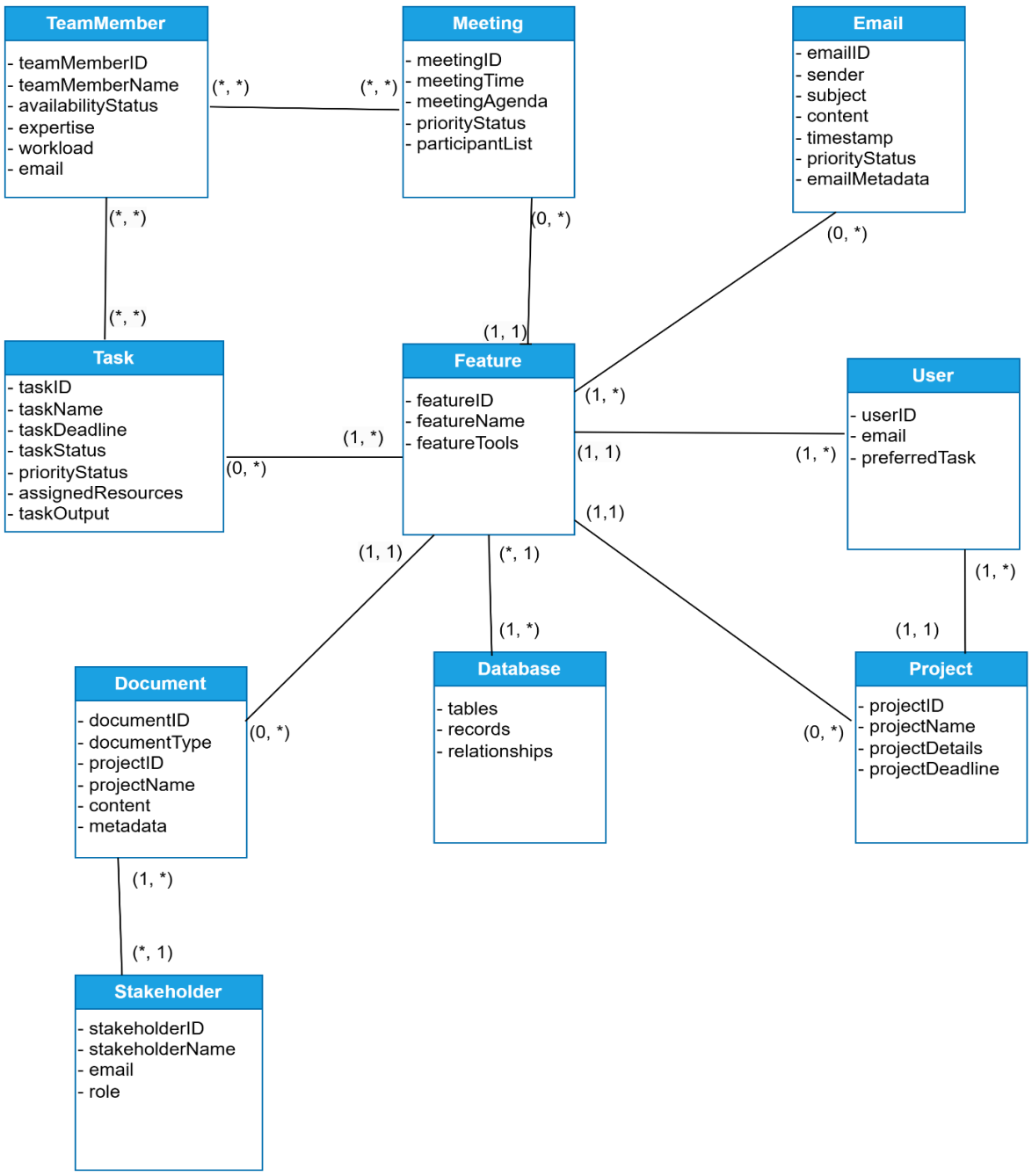
task duration = data element

time to complete task = data element

creation date = data element

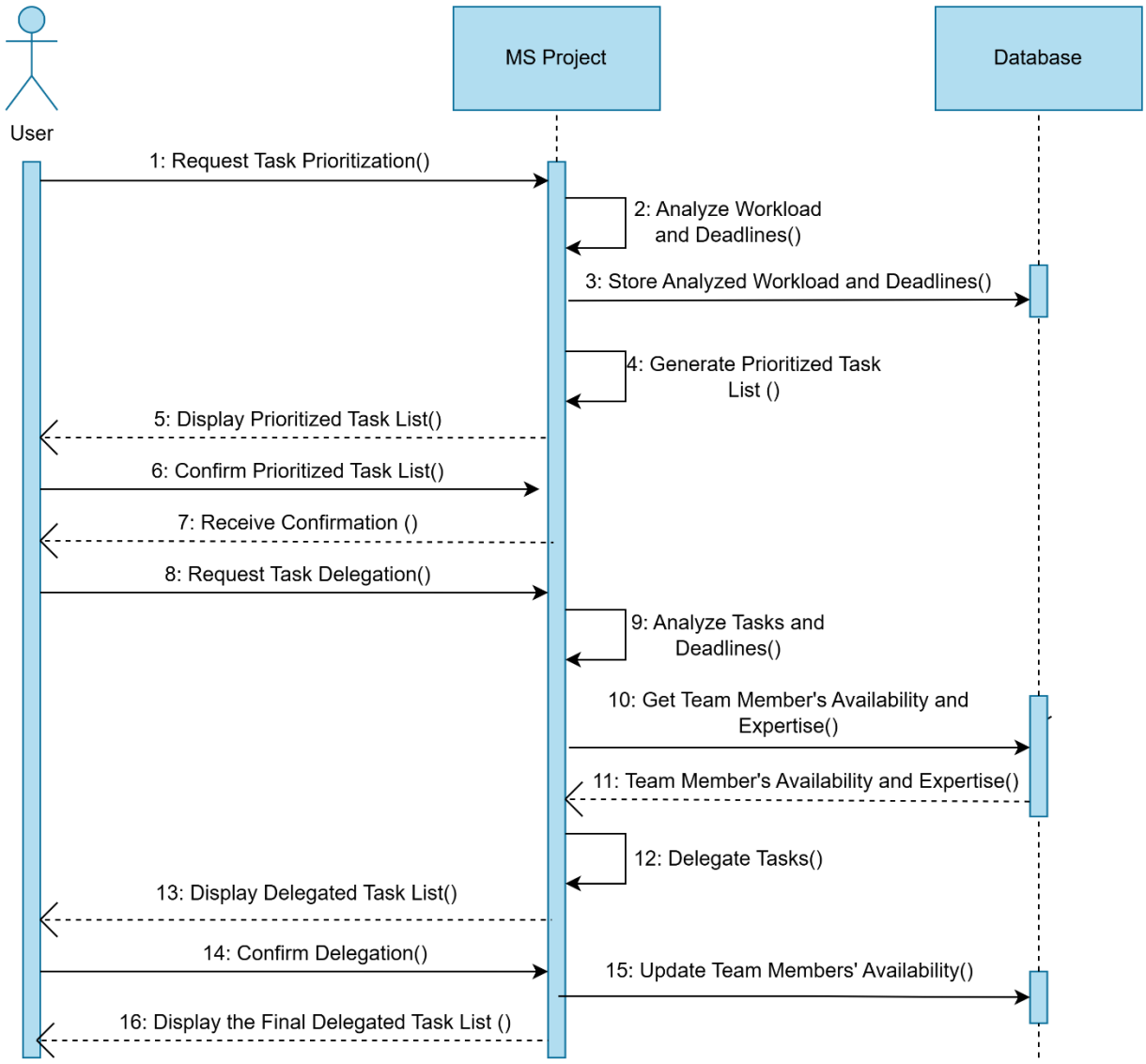
comment = data element

Class Diagram without Methods

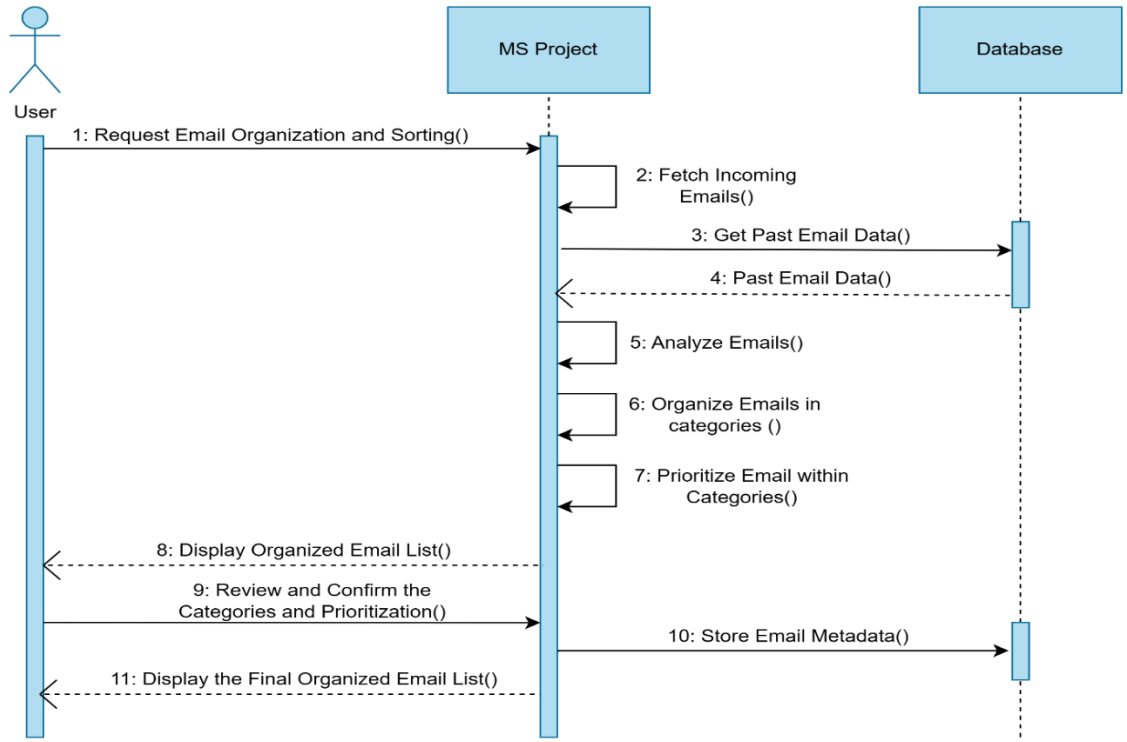


Sequence Diagram

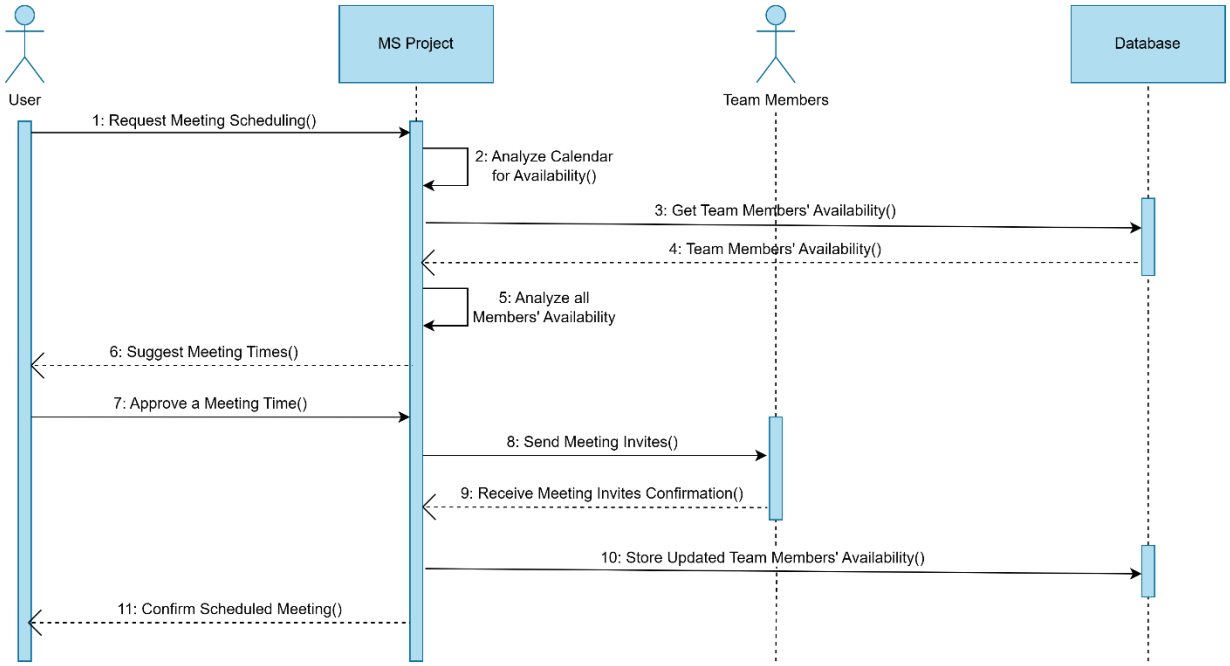
Task Prioritization and Delegation



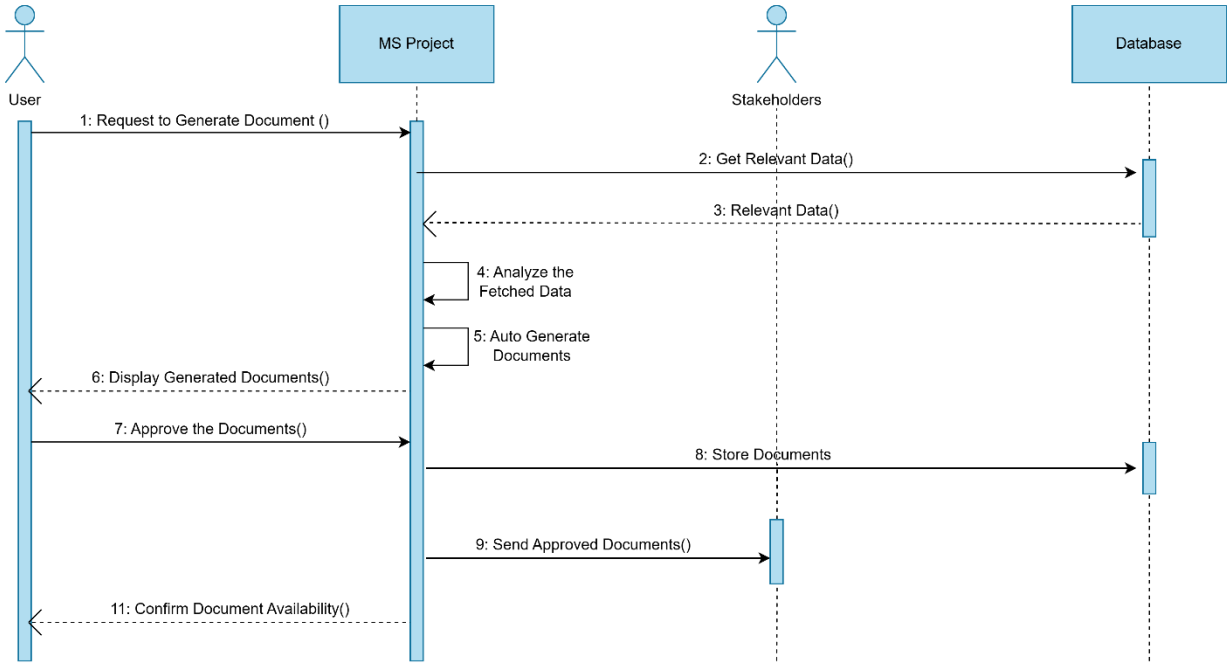
Email Organization



Intelligent Scheduling and Coordination



Document (Reports and Presentation) Generation and Sharing



Functional Specifications for the Proposed System

User Story 1: Access "Weekend Saver" Feature

- As a user, I want to access the "Weekend Saver" feature in the MS Project, so that I can use the project management automation tools to easily complete my tasks over the weekend.

User Story 2: Query Database

- As MS Project, I want to retrieve relevant data for the tasks performed by the automation tools, so that I can complete the tasks successfully and accurately.

User Story 3: Prioritize and Delegate Tasks

- As MS Project, I want to analyze task urgency, deadlines, and team members' availability, so that I can automatically prioritize and delegate tasks.

User Story 4: Organize and Sort Emails

- As MS Project, I want to organize and sort emails into categories and based on their priority, so that the user can streamline communication by easily focusing on high priority emails.

User Story 5: Schedule and Coordinate Meetings

- As MS Project, I want to access and analyze team members' availability and suggest meeting times, so that the project manager can organize meetings efficiently and ensure that project timelines are being met.

User Story 6: Generate and Share Documents

- As MS Project, I want to analyze the relevant data to generate and share the documents with the stakeholders, so that everyone involved in the project remains informed about its progress.

User Story 7: Review and Confirm Task Outputs

- As a user, I want to review the task outputs and confirm them, so that I can ensure that quality standards are being met.

User Story 8: Store Task Outputs and Metadata

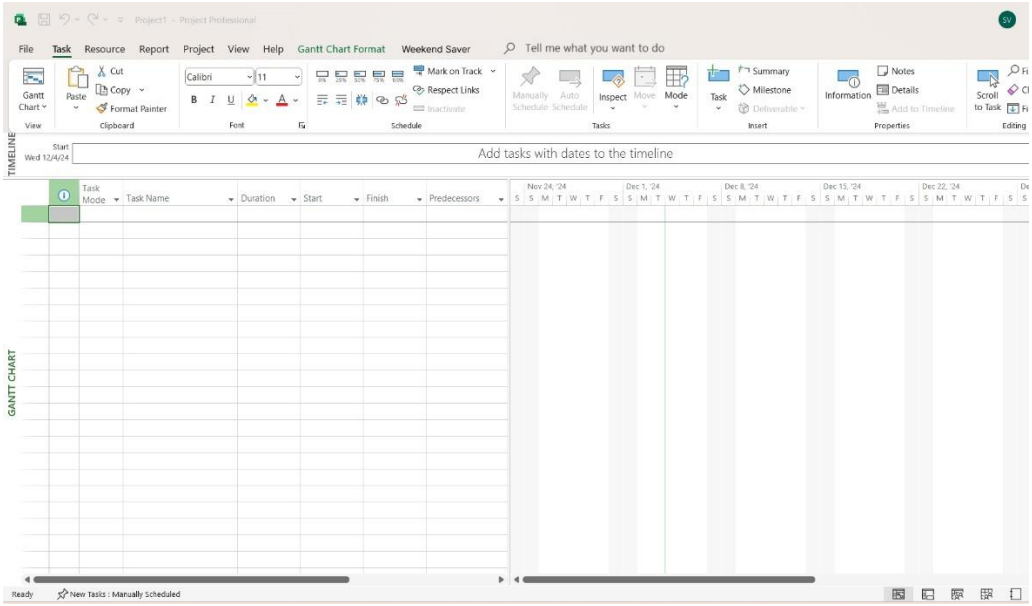
- As MS Project, I want to store the deliverables of tasks performed and their metadata in the database, so that future tasks are being performed based on the most recent information.

User Story 9: Update Team Member' Availability

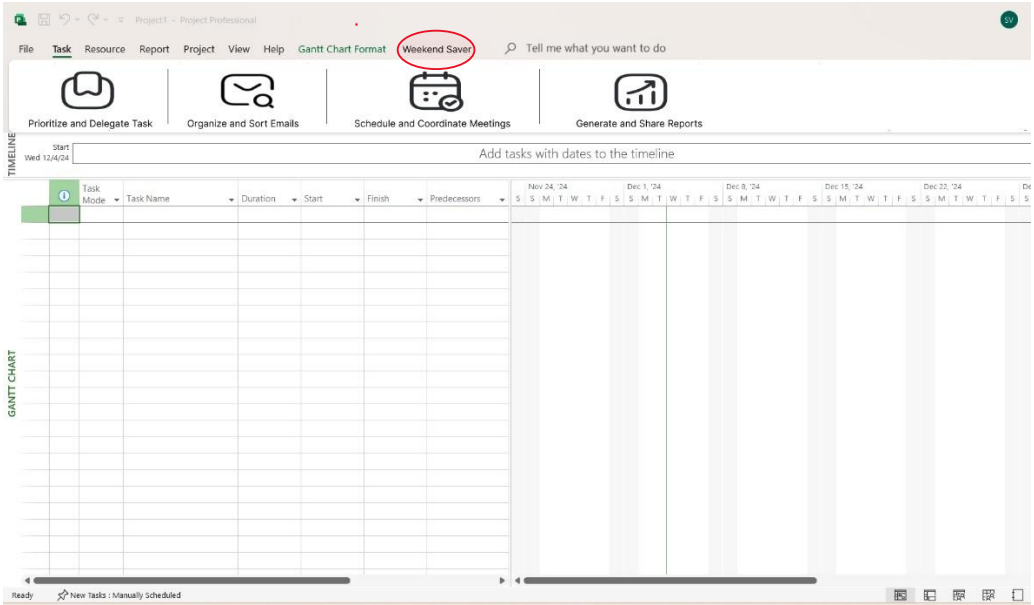
- As a database, I want to update the team members' availability based on confirmed tasks and meetings, so that future scheduling and delegation tasks can be optimized.

User Interface

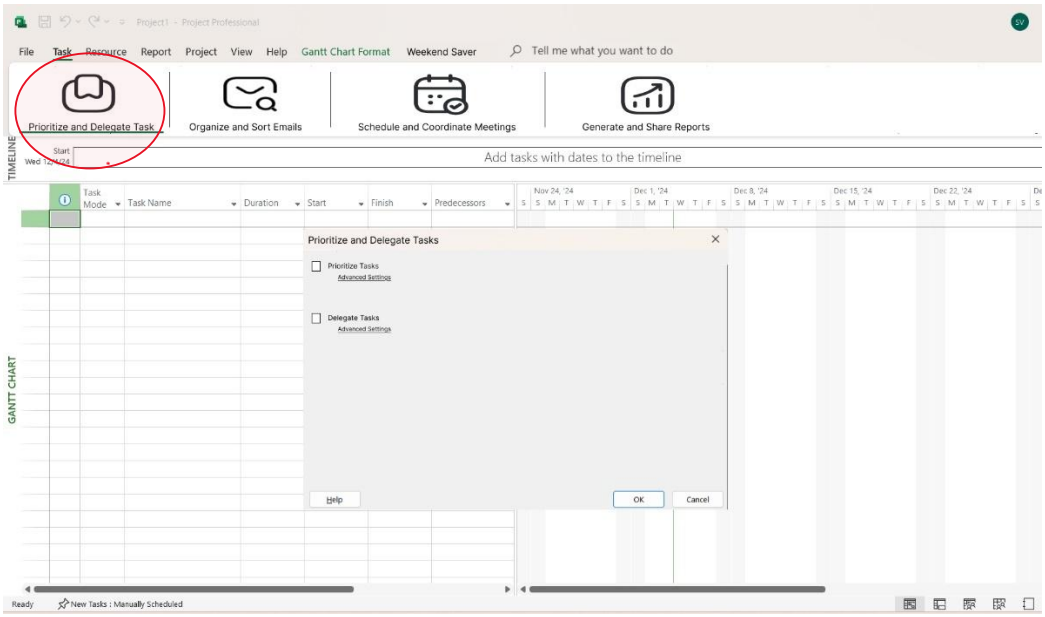
User Home Page



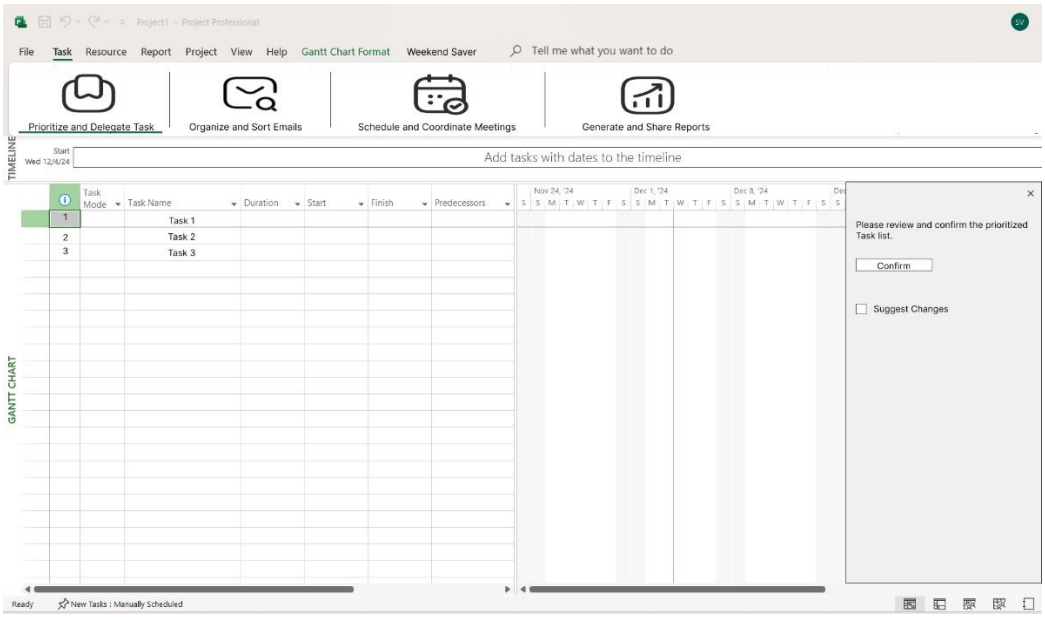
Click Weekend Saver Feature and Weekend Saver Tools Pop Up



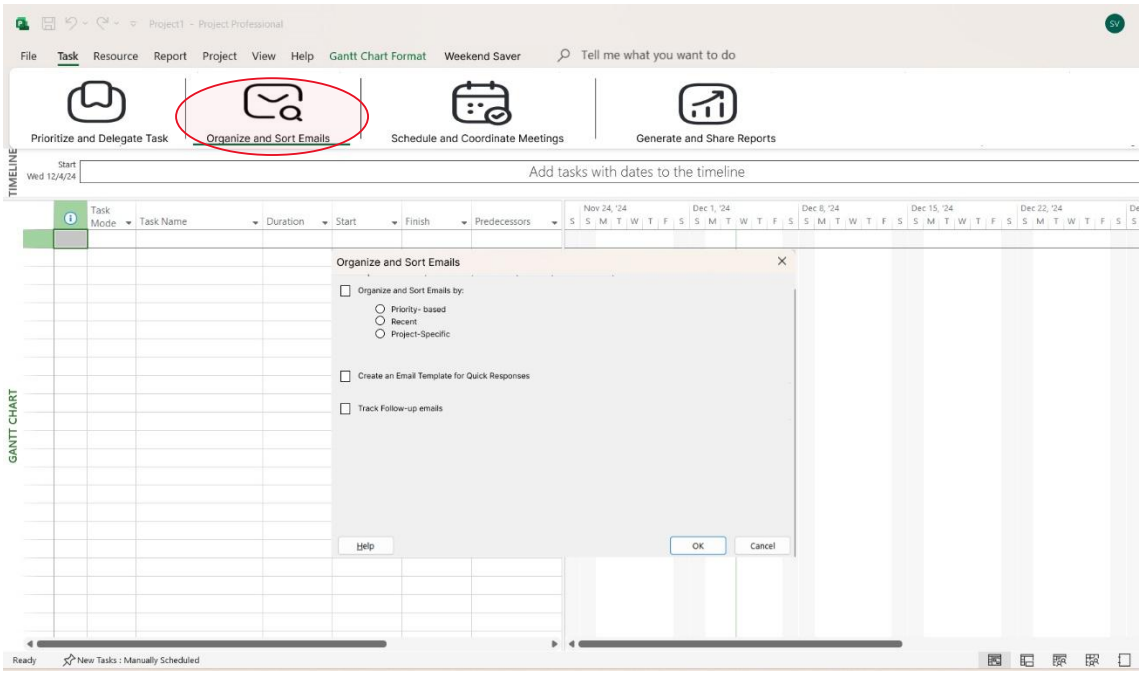
User Accesses Prioritize and Delegate Task Feature



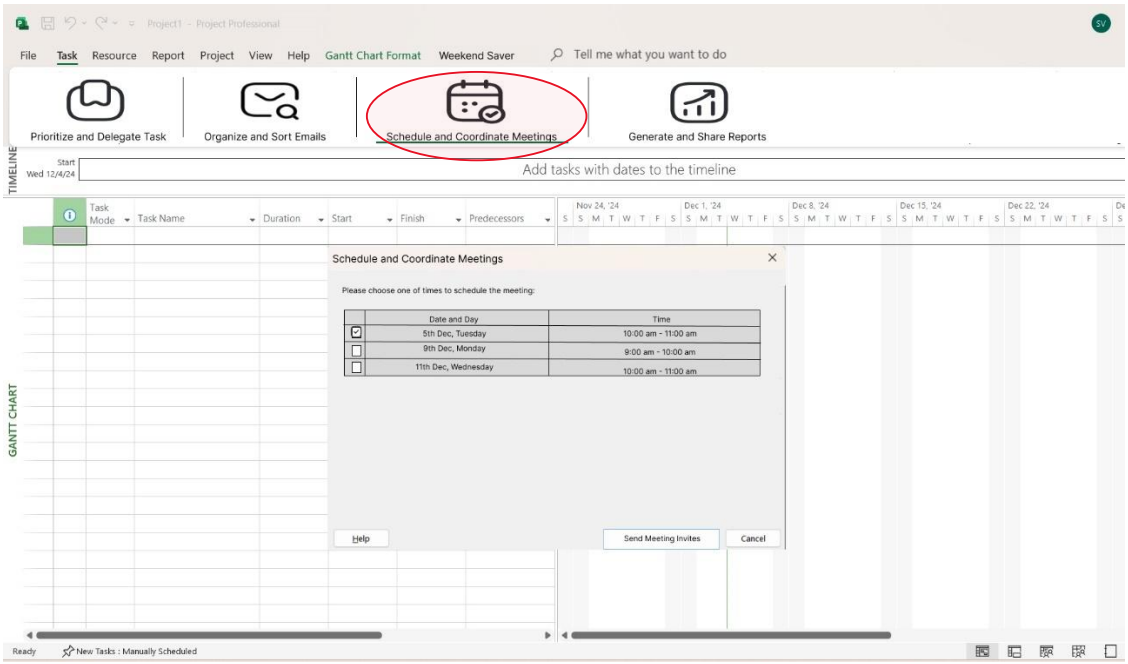
User Requests to Prioritize Task List and Weekend Saver Asks to Confirm the List or Make Changes



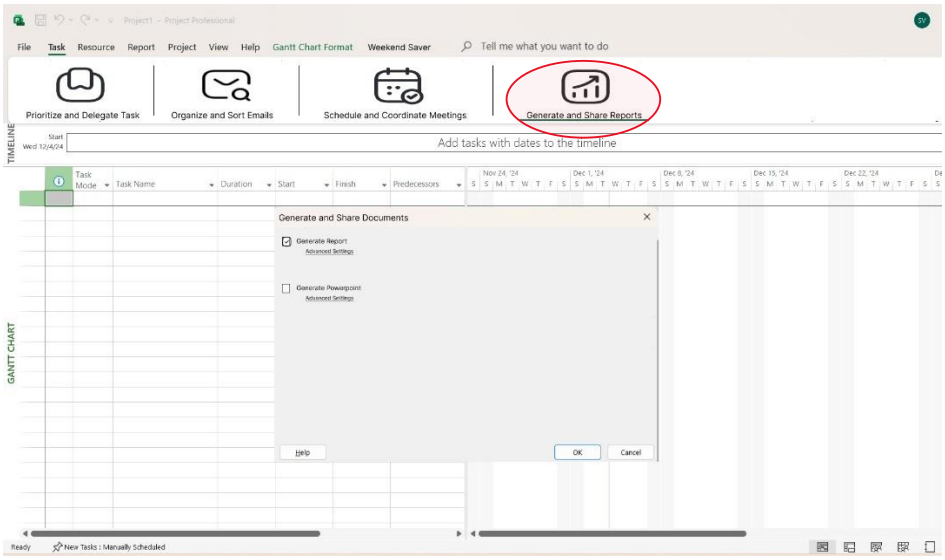
User Accesses Organize and Sort Email Feature



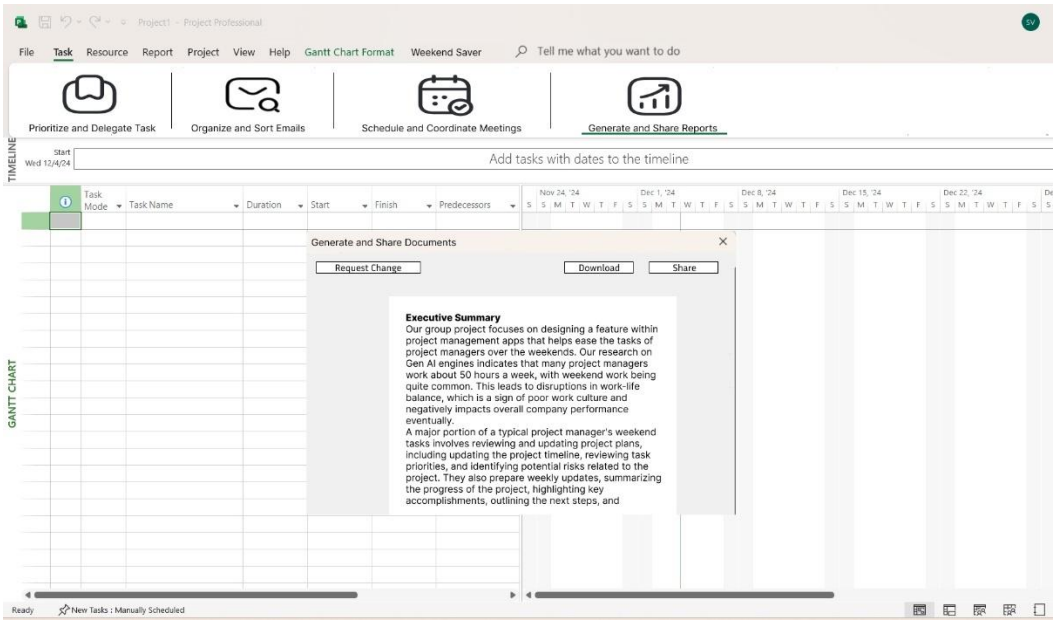
User Accesses Schedule and Coordinate Meetings Feature



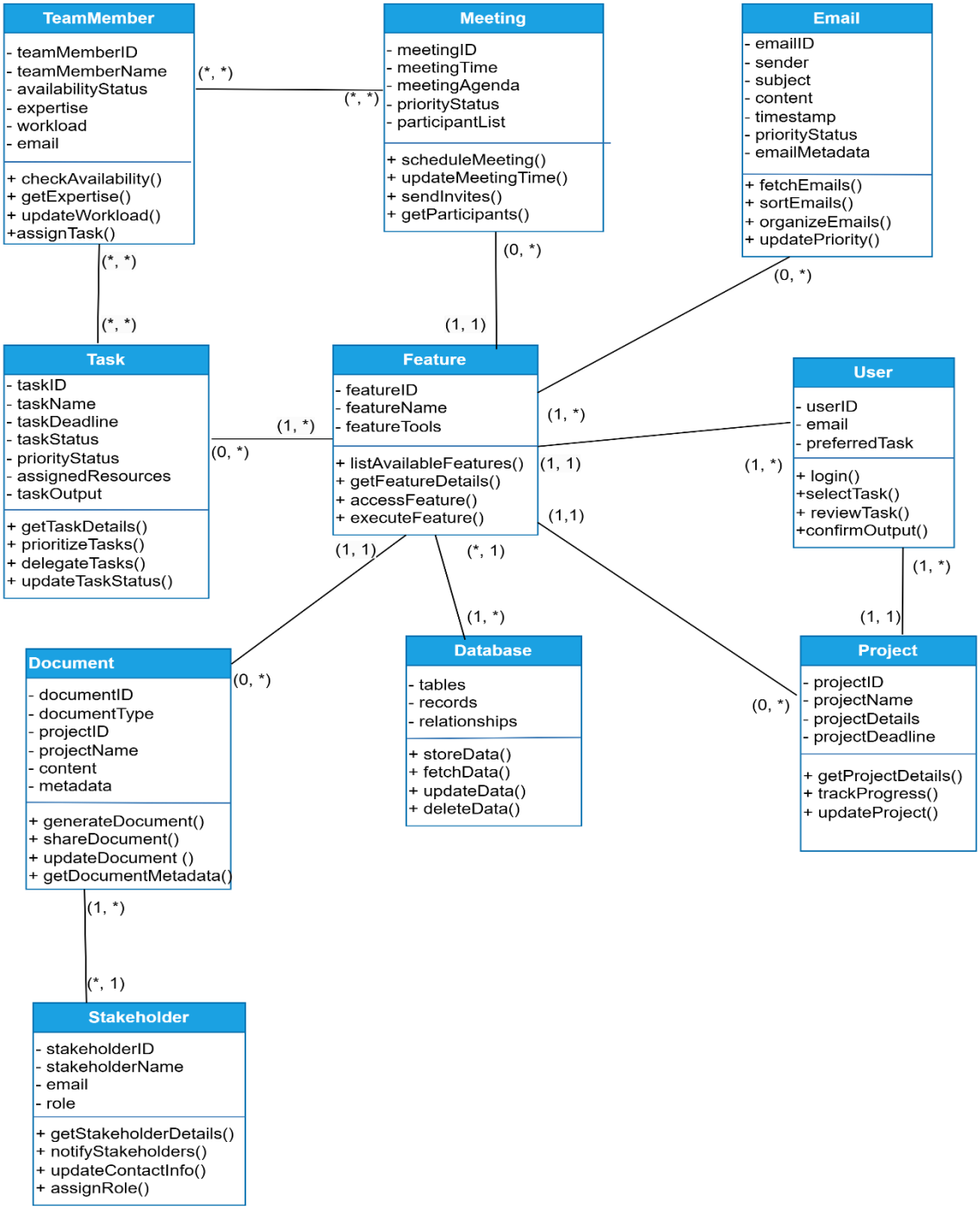
User Accesses Generate Document Feature and requests to Generate a Report



Feature Displays the Document and Prompts user to Share, Download or Request Changes to Document

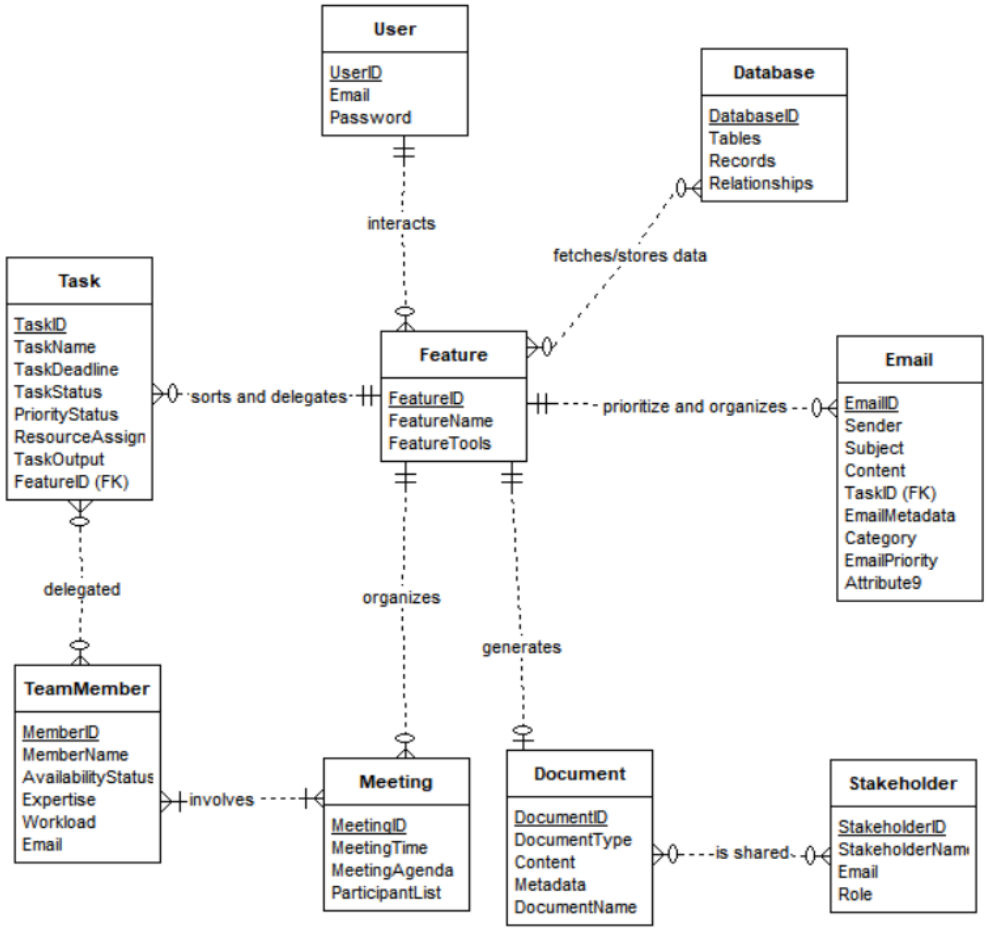


Class Diagram with Methods



Database Design

Entity Relationship Diagram



Database Constraints

User Table

- Primary Key(s): UserID
- Unique Constraint(s): UserID, Email
- Not Null Contraint(s): Email, Password

Feature Table

- Primary Key(s): FeatureID
- Foreign Key(s): UserID, references User(UserID)
- Not Null Contraint(s): FeatureName, FeatureTools

Task Table

- Primary Key(s): TaskID
- Foreign Key(s): FeatureID, references Feature(FeatureID)
- Not Null Contraint(s): TaskName, TaskDeadline, TaskStatus

TeamMember Table

- Primary Key(s): MemberID
- Not Null Contraint(s): MemberName, AvailabilityStatus, Expertise, Workload, Email

Email Table

- Primary Key(s): EmailID
- Foreign Key(s): TaskID, references Task(TaskID)
- Not Null Constraint(s): Sender, Subject

Document Table

- Primary Key(s): DocumentID
- Not Null Contraint(s): DocumentType, DocumentName, Content, Metadata

Meeting Table

- Primary Key(s): MeetingID
- Foreign Key(s): FeatureID, references Feature(FeatureID)
- Not Null Contraint(s): MeetingTime, MeetingAgenda

Stakeholder Table

- Primary Key(s): StakeholderID
- Not Null Contraint(s): StakeholderName, Email, Role

Database Table

- Primary Key(s): DatabaseID
- Not Null Contraint(s): Tables, Records, Relationships

Software Design

Method Name: Generate Prioritized Task List()
Class Name: Task
ID: Task Prioritization and Delegation #4
Clients (Consumers): User
Associated Use Case(s): Prioritize and Delegate Task
Description of Responsibilities: Analyzes tasks based on urgency, deadlines, and dependencies, and creates a prioritized task list.
Arguments Received: Task Name, Task Deadline, Task Status
Type of Value Returned: Prioritized Task List
Pre-Condition(s): Task details needed are available.
Post-Condition(s): A list of prioritized tasks is generated if all the data provided is in correct format.

```
Method GeneratePrioritizedTaskList( taskName, taskDeadline, taskStatus)
    Initialize an empty list called prioritizedTaskList
    FOR each task in taskDetails
        Calculate priority based on taskDeadline and taskStatus
        IF taskDeadline is approaching
            Assign higher priority to the task
        END IF
        Add task with calculated priority to prioritized TaskList
    END FOR
```

Sort prioritizedTaskList based on priority in descending order

Return prioritizedTaskList

End Method

Method Name: Organize Emails in categories()
Class Name: Email
ID: Email Organization #6
Clients (Consumers): User
Associated Use Case(s): Organize and Sort Emails
Description of Responsibilities: Organizes emails into categories based on metadata and priority.
Arguments Received: Email Metadata, Categorization Rules
Type of Value Returned: Categorized Email List
Pre-Condition(s): Emails are existing in the system. Metadata for emails is retrieved.
Post-Condition(s): Emails are organized into appropriate categories based on their metadata and categorization rules.

Method OrganizeEmailsInCategories(emailMetadata, categorizationRules)

Initialize an emplty list called categorizedEmailList

FOR each email in emailMetadata

Apply categoraizationRules to determine the category of the email

Assign the email to its determined category

Add categorized email to categorizedEmailList

END FOR

Return categorizedEmailList

End Method

Method Name: Suggest Meeting Times()
Class Name: Meeting
ID: Intelligent Scheduling and Coordination #6
Clients (Consumers): User
Associated Use Case(s): Schedule and Coordinate Meetings
Description of Responsibilities: Suggests potential meeting times based on team members' availability and preferences.
Arguments Received: Availability Status, Available Meeting Time
Type of Value Returned: Suggested Meeting Times
Pre-Condition(s): Team members' availability is retrieved.
Post-Condition(s): List of suggested meeting times is generated.

Method SuggestMeetingTimes(availabilityStatus, availableMeetingTime)

Initialize an empty list called suggestedMeetingTimes

Retrieve availabilityStatus of all team members

FOR each team member

Identify overlapping availableMeetingTime

Add overlapping times to suggestedMeetingTimes

END FOR

Rank suggestedMeetingTimes based on preferences or least conflicts

Return suggestedMeetingTimes

End Method

Method Name: Send Meeting Invites()
Class Name: Meeting
ID: Intelligent Scheduling and Coordination #8
Clients (Consumers): Team Members
Associated Use Case(s): Schedule and Coordinate Meetings
Description of Responsibilities: Sends meeting invite all participants for a scheduled meeting.
Arguments Received: Meeting Time, Meeting Agenda, Participant List
Type of Value Returned: Success/ Failure message
Pre-Condition(s): Meeting time is scheduled and confirmed by the user.
Post-Condition(s): Meeting invites are sent to all participants.

Method SendMeetingInvites(meetingTime, meetingAgenda, participantList)

FOR each participant in participantList

Create a meeting invite with meetingTime and meetingAgenda

Send the invite to the participant

Log the status of the sent invite (success or failure)

END FOR

IF all invites are sent successfully

Return "Success" message

ELSE

Return "Failure" message

End Method

Method Name: Send Approved Documents()
Class Name: Document
ID: Document Generation and Sharing #9
Clients (Consumers): Stakeholders
Associated Use Case(s): Document Generation and Sharing
Description of Responsibilities: Sends approved documents to stakeholders via email.
Arguments Received: Document Details, Stakeholder List
Type of Value Returned: Success/ Failure Message
Pre-Condition(s): Document is approved for sharing.
Post-Condition(s): Document is sent to stakeholders.

Method SendApprovedDocuments(documentDetails, stakeholderList)

FOR each stakeholder in stakeholderList

Retrieve stakeholder’s contact information

Attach documentDetails to an email

Send the document to the stakeholder

Log the status of the sent document (success or failure)

END FOR

IF all documents are sent successfully

Return "Success" message

ELSE

Return "Failure" message

End Method

Project Presentation

Presentation Recording:

https://drive.google.com/file/d/12qlHUxz489LtxdNNnqYeMATAxWH6L_7N/view?usp=sharing

Presentation PowerPoint: Weekend Saver.pptx

Weekly Project Timeline

Planned Due Date	Actual Completion Date	Tasks
Sep 14, 2024	Sep 14, 2024	Executive Summary
Sep 15, 2024	Sep 15, 2024	Problem Statement
Sep 20, 2024	Sep 20, 2024	Business Process Model
Sep 25, 2024	Sep 25, 2024	Context Diagram
Sep 27, 2024	Sep 27, 2024	Case Diagram
Sep 30, 2024	Sep 30, 2024	Case Descriptions
Oct 12, 2024	Oct 12, 2024	Data Dictionary
Oct 16, 2024	Oct 16, 2024	Class Diagram without Methods
Oct 25, 2024	Oct 25, 2024	Sequence Diagram
Oct 29, 2024	Oct 29, 2024	Functional Specifications Document
Nov 14, 2024	Nov 14, 2024	Interface Design
Nov 18, 2024	Nov 18, 2024	Class Diagram with Methods
Nov 25, 2024	Nov 25, 2024	Database Design
Nov 28, 2024	Nov 28, 2024	Software Design
Dec 4, 2024	Dec 4, 2024	Presentation
Dec 4, 2024	Dec 4, 2024	Project Report Due

Appendix

Prompts used to get information about the project manager's general activities on weekend

- How project managers usually perform their weekend activities?

- Are most of these tasks manual? And how much time do these tasks consume on weekends?