PROJECT PLAN

Madrid, 23/04/2021

Reference: TEAMTEAM-PP45 / PP45 / v4.5

GENERAL INFORMATION:

Document Type:	Project Plan
Authors:	Jorge Blanco Rey Ángel Casanova Bienzobas Rodrigo Juez Hernández Pablo Ernesto Soëtard García
Last revised by:	Angel Casanova Bienzobas
Last revision date:	23/04/2021
Approved by:	Rodrigo Juez Hernández
Approval date:	23/04/2021
Team:	Runner Solutions
Status:	DELIVERED

CHANGE LOG:

Version	Date	Main modifications	Purpose
0.5	28/03/2021	Section1	Finish the first section, leaving the summary for the end
0.7	30/03/2021	Project description	Finish project description to be able to start with requirements
1.5	06/04/2021	Functional requirements	Correct functional requirements and modify them, so they have input, process, and output
2.0	09/03/2021	Non-Functional requirements	Finish requirements to be able to start with the function points
2.5	13/04/2021	Starting section 4, with some unadjusted function points	Start with unadjusted function points
3.0	17/04/2021	Finish function points	Finish part of function points to be able to start with MS Project
3.5	18/04/2021	Start with sections 4 and 5	Start with sections 4 and 5, which we could finish once we have MS Project completed
4.0	21/04/2021	Finish sections 4 and 5	Finish versions 4 and 5. Pending the revision of the document
4.5	22/04/2021	Revision of the document and correct format	Finish all the details of the document to be able to deliver

SUMMARY

The Escuela Politécnica Superior (EPS), one of the schools of the Universidad Autónoma de Madrid (UAM), has decided to contact Runner Solution S.L for implementing the TEAMTEAM application. This system is pretended to be the solution for the students that are attending online classes during this tough lockdown times.

The main objective of this application is to keep all the functionality need to attend a class under the same roof, that is, to avoid the necessity of having other software's opened while holding a meeting such a text editor or a repository.

We will integrate our system with Moodle in order to maintain the actual way that the student have to identify themselves. So, through the Moodle page any user shall be able to sign in TEAMTEAM and then it will be launched in a different window.

This document shows the development process which will be carried out in each of the five subsystem that we have defined and also the details about the integration of the whole project as well as the probing method that will ensure the quality of the final product.

As this is a complex project, we have reckoned that is better to share the whole project in three increments. In the first increment we will develop the Team Management and the Meeting Schedule subsystems, then in the second one, we will complete the Meeting Making and the Meeting Management subsystem, finally in the last increment we will end up with the Statistic Management subsystem.

The real aim of this document is to become the initial agreement in terms of the requisites of the project, stablish an estimate for the customer and close the planning for the whole project, that is, make a Gantt diagram where all the dates and the personnel is distributed in a proper way to maximize the performance of the developing team.

We have made an estimation of the amplitude of the project through the function points method, after applying this methodology, we have concluded that we will spend 328 working days in the project, and we will need an inversion of 188.988€.

TABLE OF CONTENTS

Pages

ANNEXES

LIST OF FIGURES

Figure 23: Hardware Resources and Development Software Costs

1. INTRODUCTION

1.1 PURPOSE

TEAMTEAM is an application that will allow EPS students to do anything that they should need while doing their practices or attending to lessons through virtual meetings.

The purpose of this document is to provide a detailed view of the functionalities based on the given requirements in order to calculate the total cost of the project, so our client would have an explanation of the cost estimation.

In this document all the key subsystems and features that compose the application will be explained.

To estimate the size of the system, a Function Point analysis will be performed and using the MS Project tool a Gantt diagram will be created for the client to see the estimated time of the project, moreover, he/she will be able to see the time each task is going to take. The life cycle chosen for this project to be developed on will be an incremental one, which will be descried deeply in section 3.3.

Based on the Function Points and time estimate mentioned above, a budget for the project will be calculated.

To ensure the quality of the products produced during the development of the project, a rigorous monitoring and control will be performed, as well as all the documentation needed will be generated during the process and handed to the client if requested.

1.2 SCOPE

The scope of this document is to provide a complete overview of the requirements of the application TEAMTEAM and specify some guidelines for:

- Subsystem based software structure.
- Specification of the requirements of the application.
- Application deploy environment, including software and hardware.
- Methodologies used during the development of the application.
- Gantt chart gathering all the tasks and due dates of them needed to accomplish the creation of the application.

- Task division among the members of the development team.
- Estimate by function points of the time and monetary cost of the development of the project.
- Milestone monitoring to ensure reliability and quality of the project.

This document is not intended to be a detailed software design document nor provide any GUI or code POC of the application.

1.3 RESPONSIBILITIES

PROJECT LEADER	Ángel Casanova Bienzobas
PROJECT HEAD	Rodrigo Juez Hernández
PROJECT QUALITY MANAGER	Jorge Blanco Rey
PROJECT DOCUMENTATION MANAGER	Pablo Soëtard García
CUSTOMER REPRESENTATIVE	María Elena Gómez Martínez
OTHER	N/A

Figure 1: Responsibilities

1.4 DEFINITIONS

REFERENCE	TITLE
FR	Functional Requirement
NFR	Non-Functional Requirement.
GUI	Graphical User Interface
POC	Proof of Concept
UAM	Universidad Autónoma de Madrid
EPS	Escuela Politécnica Superior
API	Application Programming Interface
ILF	Internal Logic Files
EIF	External Interface Files
EI	External Inputs
EO	External Outputs
EQ	External Inquiries
TM	Team Management
MS	Meeting Scheduling
MManag	Meeting Management
MMaking	Meeting Making
SM	Statistics Management

Figure 2: Definitions

1.5 REFERENCED DOCUMENTATION

REFERENCE	TITLE
1	Adobe Systems
2	Microsoft Teams
3	Discord
4	IdeaFlip
5	Google For Education
6	Esendex gateway
7	Microsoft Teams API
8	Risk Management Plan

Figure 3: Referenced Documentation

2. PROJECT OVERVIEW

We are currently in a global pandemic where students are already very stressed, and they have to deal with messy apps like the ones that are being used right now to accomplish all the work they are asked to carry out.

2.1 PROJECT DESCRIPTION

We are currently in a global pandemic where students are already very stressed, and they have to deal with messy apps like the ones that are being used right now to accomplish all the work they are asked to carry out.

A survey made by Adobe Systems [1], found that many people spend more than 2 hours every day checking emails and back-and-forth conversations, organizing their calendar, and planning their daily workflow.

Students should not have to spend time switching from apps to apps or from environment to environment, that is why TEAMTEAM is necessary, an application that is straight-forward for EPS students, tightly integrated with their existing Moodle ecosystem and where all the calendars and teams will be synchronised automatically.

We offer a unified solution that includes under the same roof different functionalities such as the ones implemented in other already existing frameworks like Microsoft Teams [2], Discord [3], IdeaFlip [4], Google For Education [5].

Systems [1], found that many people spend more than 2 hours every day checking emails and back-and-forth conversations, organizing their calendar, and planning their daily workflow.

Students should not have to spend time switching from apps to apps or from environment to environment, that is why TEAMTEAM is necessary, an application that is straight-forward for EPS students, tightly integrated with their existing Moodle ecosystem and where all the calendars and teams will be synchronised automatically.

We offer a unified solution that includes under the same roof different functionalities such as the ones implemented in other already existing frameworks like Microsoft Teams [2], Discord [3], IdeaFlip [4], Google For Education [5].

2.1.1 APLICATION SCOPE

The main goal of this project is to integrate several already existing systems under the same roof, but not reinventing the wheel, therefore already existing code bases may be reused to implement some of the features exposed in this document.

Also, despite being an open-source project the system may use proprietary software from external providers, like the one of the video call system, and the system is not going to include an API to integrate more subsystems into the app in the future, these are out of the scope of this project.

The system will not store information of students such as their credentials, assignments, teams, calendar, meetings... This information will be retrieved by the system from Moodle.

2.1.2 RELATION WITH OTHER SYSTEMS

The TEAMTEAM application will make use of other external applications that will help to load student's data, have video calls, and receive SMS notifications.

UAM database
This database contains all the information of the university personnel. TEAMTEAM will read the following data from students and professors: login credentials, email, subjects enrolled and calendar. The application will also store on the database teams, meetings, assignments, and notifications. The information of the repository will be stored in the application database.

SMS gateway
Users will be able to select how they want to receive the notifications, and one of the options is by SMS. For this purpose, TEAMTEAM uses the Esendex gateway [6], and the cost for this service will be handled by the UAM.

Microsoft Teams API
Users will be able to have meetings by voice, as well as video calls, for this purpose we will make use of the Microsoft Teams API [7]. This API supports the number of team members in a meeting room, and they will be able to communicate easily between them and have calls with good quality.

2.1.3 SUBSYSTEMS

We have developed the four subsystems of the previous practical assignment, but we also have added a new one:

The first subsystem that the user had to deal with, is Team Management Subsystem since the first step is to create a team. From that subsystem, the user will be able to access the Statistics of the meeting through the Statistic Management Subsystem, also he could Schedule a meeting in the Meeting Scheduling Subsystem, or he could create a new one accessing the Meeting-Making Subsystem.

Once the Meeting is created the user may manage the different options in the Meeting Management Subsystem and finally during the meeting call the user may use the tools that have been provided in the same subsystem. Also, it is possible to access the statistic from the Meeting-Making Subsystem from here.

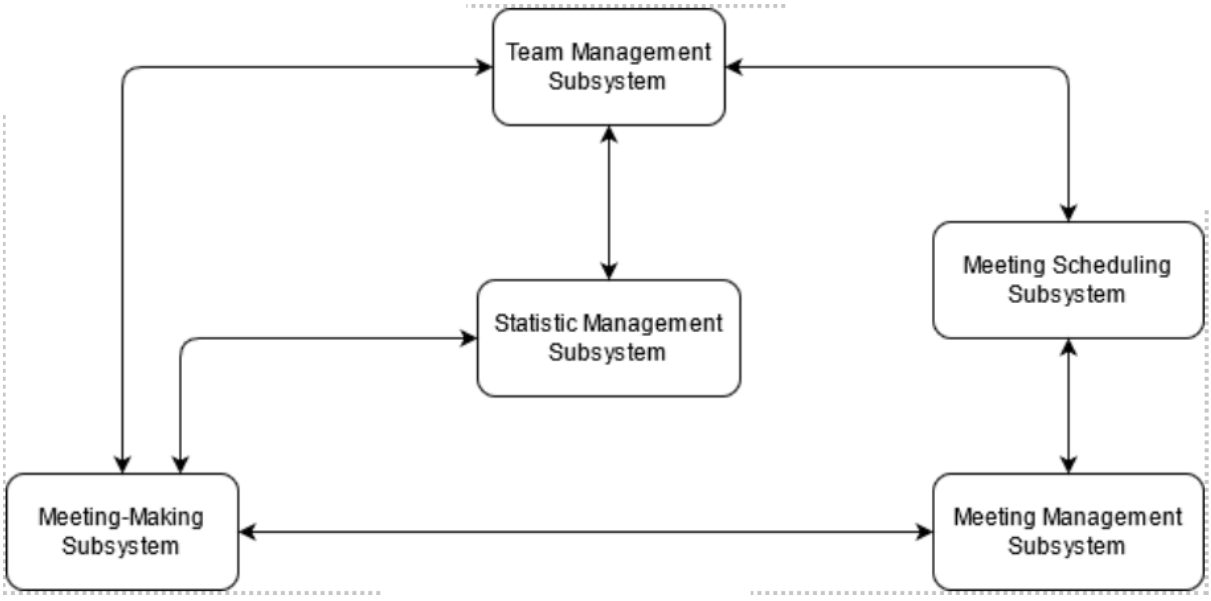


Figure 4: Subsystems Relation

1.- Team Management

The Team Management Subsystem is in charge of managing the teams among the application, it should offer the following functionalities:

- Create the team for practical assignments.
- Attach the members of each practical assignment to its corresponding team.
- Make notifications.
- Select preferences from the teams.
- Select view information from the teams.

2.- Meeting Scheduling

The Meeting Scheduling Subsystem allows students to see pending or completed teamwork meetings.

3.- Meeting Management

The Meeting Management Subsystem, which should provide the following functionalities:

- Create the meeting.
- Meeting date agreement mechanism.
- Send reminders.
- Manage the rooms that make up the meeting infrastructure.
- Consult information on meeting development.
- Cancel the meeting.

4.- Meeting-making

The Meeting-making subsystem that should implement the following functionalities:

- Manage the automatic delivery of practical assignments to Moodle.
- Provides an in-call toolbox for students, that will include tools like a whiteboard, live chat, screen sharing, repository, notebook, or a video call system.

The Meeting Making Subsystem also provides the user with a set of collaborative tools that he may use during the call and that will provide our application with a wide range of options to develop his work and will eliminate the necessity of using other software while attending to a meeting call. Those tools are:

- Shared whiteboard.
- Personal notebook.

- Live chat.
- Screen sharing.
- Shared repository.

5.- Statistics Management Subsystem

The Statistics Management Subsystem will provide professors textual and graphical reports. These reports must contain enough information for teachers to evaluate the scope and complexity of the practical assignments.

The subsystem must track:

- Both meetings and the work of the teams.
- Participation in meetings.

The forms will be made for:

- One assignment and all the class.
- One team and all assignments.

And must handle the data to:

- Visualize different statistics and the work of the team.
- Percentage of meetings held and cancelled.
- Average duration of meetings.
- Average meetings held.
- Number meetings attended or excused by all members.
- Frequency of meetings during development.

2.2 REQUIREMENTS

2.2.1 FUNCTIONAL REQUIREMENTS

2.2.1.1 TEAM MANAGEMENT SUBSYSTEM

FR1. Creation of a Team.

The user can create a team with the required information.

> Input: The user shall specify the subject name to which they are related, team name, maximum or minimum number of students per team, name and email per student, and these students must be members of the same team.
>
> Process: The application shall automatically fill out the user's schedule, the users preselected shall receive an invitation to join the team, which they may accept or decline.
>
> Output: A team is created, with the information provided, and the users that are invited.

FR2. Broadcasting

Teams may broadcast an invite to all users that are not included in any team for that subject.

> Input: Team to broadcast.
>
> Process: The application shall gather all the users compatible with the schedules of the team.
>
> Output: All these users receive an invitation to join the Team.

FR3. Joining a team not full

The user may join a team at any time if it is not full.

> Input: The team to join and the user that joins.
>
> Process: The application shall check if the user is compatible with the team's schedule.
>
> Output: The user joins the team.

FR4. Asking to join a team.

The user may ask to join a team instead of being invited.

> Input: The user that asks to join and the team.
>
> Process: The application checks if the user can join and then sends a request to the members of the team.

Output: The user is accepted or rejected.

FR5. Leaving a team

The user shall not be able to leave a team once they join, as well as not be able to dissolve it until all practical assignments are done.

Input: None

Process: When all assignments are done the application enables the process to leave the team.

Output: The user does not leave the team during assignments.

FR6. Notifications

Notifications for cancelling or creating meetings.

Input: Creation or cancelling.

Process: The application gathers all the members of the meeting cancelled or created.

Output: When cancelling a meeting a notification shall be automatically sent to the members and when a meeting is created, the app automatically sends a notification to its user without them needing to configure anything.

FR7. Signing In

Signing in must be independent from Moodle, and it is required for doing anything related with teams.

Input: Credentials from user.

Process: The application will check if the credentials are correct and will load all the information required for the user actions.

Output: All functions related with teams and meetings will be enabled.

2.2.1.2MEETING SCHEDULING SUBSYSTEM

FR8. **Schedule Access**
The registered user may access its meeting schedule where he will find the completed or pending meetings. This meeting schedule shall be integrated into the Moodle calendar, apart from being accessible from the TEAMTEAM app.

> Input: The user selects the calendar from the Moodle page or TEAMTEAM app.
>
> Process: The system redirects the user to the calendar page, loading all the information about the past and pending meetings of all the teams that he has (or has been) for practical assignments.
>
> Output: The schedule with all the past and pending meetings.

FR9. **Check the details of a past meeting.**
From the meeting schedule the user may see the details of a meeting that has already taken place.

> Input: The user has accessed the schedule and clicks on the title of a meeting that has already taken place.
>
> Process: The system will load all the information that is going to show to the user related to that meeting.
>
> Output: The user will see a small pop-up with the meeting name, team, subject, date, time, duration and how he was notified in case he had the notifications activated. Also, the attendee's names, punctuality of each attendee, remaining time in the meeting and meeting report (goals achieved and objectives to be met for the next meeting).

FR10. **Check the details of a pending meeting.**
From the meeting schedule the user may see the details of a future meeting.

> Input: The user has accessed the schedule and clicks on the title of a meeting that has not taken place yet.
>
> Process: The system will load all the information that is going to show to the user related to that meeting.

Output: The user will see a small pop-up with the meeting name, team, subject, date, time, duration and how he shall be notified in case he has the notifications activated.

2.2.1.3 MEETINGS MANAGEMENT SUBSYSTEM

FR11. **Notification management.**

Users should select the type of notifications they wish to receive, they can choose from email, push notifications to desktop or mobile or message from a bot inside the app.

Input: A student creates/deletes a meeting.

Process: The system checks that all the data that is needed in order to create/delete the meeting is correct.

Output: All the participants of the meeting get notified of the creation/deletion of the meeting.

FR12. **Schedule a meeting.**

Any member of a team can schedule a meeting (videoconference and joint working session) for practical assignments, any of its members can propose the call for a meeting for their team.

Input: The student that wants to schedule the meeting needs to provide the title, its date, time and duration, the objectives of the meeting and/or the topics to be discussed.

Process: Before scheduling the meeting a mechanism to reach agreement among team members on the date, time and duration of the meeting is run. The system checks the calendar gaps of each member of the team and selects the ones that fit the duration requirement of the proposed meeting to be voted by the members.

Output: Once all members have voted, the system reserves both matches and for the first option creates the meeting in each member's calendar (generating the meeting link too) and notifies all team members that the meeting has been created with all the information of the call. The second match is left on the waiting list in case any unforeseen events arise, and the meeting has to be rescheduled.

FR13. **Cancelling a Meeting.**

Any participant of the meeting should cancel a scheduled meeting, providing a reason why it is being cancelled.

Input: A meeting participant wants to cancel an already scheduled meeting, he/she provides the reason to do so.

Process: The system retrieves the information given by the user.

Output: The meeting record is deleted from the calendar of all the participants of the meeting cancelled.

FR14. **Meeting Development.**

The user may associate meetings with practical assignments and may be able to create, close, open and leave a room.

Input: Once the meeting has been scheduled, the system will open the room for the meeting 5 minutes before it starts.

Process: While the room is opened, members may join the room by clicking on its link on their calendar. The calendar must have a link in each tile, so by clicking in one tile the system shall redirect the student to the meeting.

Output: The system shall close the room 5 minutes after the established time if all the members have left the room. When a meeting ends the system asks the members to check the objectives that they have fulfilled during the call.

FR15. **Room access.**

Only allowed students should enter the room, and they can do so as many times as needed.

Input: A student tries to join the room.

Process: The system checks whether the student is allowed to enter that room.

Output: If the user is allowed, then he/she successfully enters the room.

2.2.1.4 MEETINGS MAKING SUBSYSTEM

FR16. **Task-board**

During a meeting, students will be able to access a task-board with which they are going to distribute the work to do by each participant.

Input: The user needs to be inside a meeting.

Process: Once the user is in a meeting.

Output: A task-board for the meeting is created where students can check the tasks. The tasks that have not been done appear in red, and those completed appear crossed out.

FR17. **Collaborative tools**

Registered users have access to chat, screen sharing, notebook, collaborative whiteboard, and a shared repository during the meetings.

Input: The user must be inside a meeting.

Process: Once the user is inside a meeting, he would be able to toggle among any of the tools by just clicking a button.

Output: The main screen will toggle and will make visible the functionality that the user has selected.

FR18. **Roster of attendees**

The system will keep the total minutes that each user has spent in the class.

Input: The user should be an attendee of a meeting call.

Process: At the time a room is opened, the system will generate a dictionary whose keys are the name of the registered users and the information associated is a list that must keep the total minutes that each user has spent in the class.

Output: The dictionary is completed and the user that has created the meeting should access it.

FR19. **Recording of the classes**

The system will record the classes for the users to access them afterwards as well as whether the meeting objectives have been met.

Input: Nothing.

Process: The system will start recording the meeting when it begins and will store the recording in the folder of the team for any team member to review the call at any time he wants to. Also, the system keeps the track of the objectives that have been proposed before the beginning of the meeting in order to generate a report when the meeting ends.

Output: The system will generate a recording of the class in the share repository of the team.

FR20. **Automatic delivery**

Everything under the "Practical Assignment Submission" folder of the shared repository will be compressed and automatically delivered into Moodle by the system.

Input: Nothing.

Process: When the deadline for a task is over the system will compress and automatically deliver the task that is placed in the shared repository folder with the name of the deliverable.

Output: The task will be automatically submitted.

FR21. **Leaving and joining the room.**

Users may leave and join the room as long as it remains opened.

Input: The user must be in a meeting call.

Process: If the user decides to leave the call for any reason, he will be able to re-enter in it if the meeting remains open.

Output: The user will be inside the meeting call again.

FR22. **How to manage the information.**

The application loads (and dumps) the team, meeting, and calendar information from Moodle´s database so it is safe to assume that the device must be connected to the network.

Input: The user makes use of the TEAMTEAM application for looking in the calendar.

Process: The system will connect to Moodle and will send a query for looking at the desired calendar (the database management system of Moodle is transparent for our application) and will get proper data. Also, it can dispatch some data to Moodle in order to insert it into the database.

Output: The user will see the desired data in the TEAMTEAM application directly.

2.2.1.5 STATISTICS MANAGEMENT SUBSYSTEM

FR23. **Practical assignment and group report**

The teacher shall be able to ask for a report with the statistics by practical assignment and group.

Input: The teacher will enter the group and practical assignment numbers of which she wants the report.

Process: The system records the numbers of meetings that have been proposed and calculates the percentage ones that have taken place and the ones that have been cancelled. It also calculates the average meetings that each team has had and the average hour in which the meetings were held.

Output: The teacher will receive a report with the percentage of meetings held and cancelled, average duration of meetings, average of the meetings held by all groups, number of meetings attended by all members, average hour in which the meetings were held.

FR24. **Practical assignment and group report for the whole class**

The system will keep track of the meetings and the work of the team at each practical assignment, so the software system will allow to generate and visualize different statistics of the meetings.

Input: The teacher will enter the group, practical assignment and the team selected of which she wants the report.

Process: The system records the numbers of meetings that have been proposed, calculates the ones that have been held, cancelled and their average duration. It also records the participants of each meeting to calculate the percentage of meetings attended by each member of the team, percentage of meetings in which each member entered and left, frequency and distribution of the meetings during the assignments.

Output: percentage of meetings held and cancelled, average duration of meetings, percentage of meetings attended and excused each member of the team, percentage of meetings in which each member of the team entered and left the meeting, frequency of meetings during the development of each practical assignment, in the middle or end and time period when the meetings were held.

2.2.2 NON-FUNCTIONAL REQUIREMENTS

Interface and Usability Requirements:

NFR1. **The application should run on a window different from Moodle.**

The process of entering the web application is separate from Moodle, so when opening the application, a new window pops out, where all the interaction with the user will be.

NFR2. The error messages will be displayed in red.

If any error is detected, be it by user error or application error, a message will be displayed to the user in red, the system however will try to disrupt the workflow as little as possible, that is, less than half of all errors produced must force a complete restart.

NFR3. The application must have an interface adequate to the needs of EPS students.

Students must spend less than five minutes on understanding how each subsystem works and is interacted with.

They must remember how it works on average, at least during a year, this way each semester they do not spend time getting started again.

NFR4. The application must be responsive.

That is between clicking and some kind of response no more than 0.5 seconds shall pass, it does not need to be the content or the action the user requested necessarily, rather an acknowledgment that the action is being processed, or an error message in case of failure.

NFR5. Uniform experience.

Switching between compatible browsers (specified later on) must not change the interface nor the experience, everything should work exactly the same. And the level of responsiveness must not vary more than 20% between the worst performer and the best.

Operational Requirements:

NFR6. System backup

Every day at 2 a.m. the application should make a backup.

Documentation Requirements:

NFR7. The system should include a user manual in English and Spanish.

A user manual both in English and Spanish should be provided, the manual will include a detailed guide of use of the web application, explanation of all the menus, functions, and features.

NFR8. The system should include a technical manual in English and Spanish.

A technical manual both in English and Spanish should be provided, the manual will include a detailed guide of the subsystems of the web application, explanation of APIs, functions, and architecture of the system.

NFR9. The system should include an online initial tutorial in English and Spanish for starting users.

An online initial tutorial should be included to guide starting users on how to interact with the system as well as show them what action they can make on the system.

NFR10. The system should include an online information centre in English and Spanish.

An online information centre where users can be assessed (in English and Spanish) on the problems they may experience while using the system, should be available.

Security Requirements:

NFR11. The system shall cipher the data in order to store it in the Moodle database, we will use the Openssl library.

User and system data stored in the Moodle database should be ciphered to secure it against possible data leaks produced by an attack to the database, for that purpose the open-source library Openssl should be used.

NFR12. The communication should **follow the TCP/IP protocols.**

The communication should follow the TCP/IP protocol in order to guarantee the flow and congestion control with the aim of avoiding the packet misses and provide a more reliable communication.

NFR13. Web Protocol Communication.

The application will implement a web-service over the HTTP protocol in order to pass any firewall without problems.

Maintenance and Portability Requirements:

NFR14. System stops for maintenance.

The application server should be stopped a maximum of 15 minutes per month for
maintenance purposes. During this period, the system will not be available, so the application will not let the users program meetings during the time it is planned. These system stops will always be programmed during hours of low use of the application and it will inform the users.

NFR15. The system compatibility

The application will run on Chromium v.89.0.4389.114, Firefox v.87 and Safari v.13 at least.

Resources Requirements:

NFR16. Secondary memory capacity

The system should have at least 4 TB of secondary memory in order to store all the user's data.

NFR17. Repository capacity

The system should allocate a maximum of 2 GB per user for their repository data.

Performance Requirements:

NFR18. Number of users.

The system should be able to support 2,000 concurrent users.

Availability Requirements:

NFR19. Available time.

The web application must be available more than 90% of the time, during daytime.

Verification and Reliability:

NFR20 The server should get back online automatically.

In case the server gets down for any reason, it has to load up again in less than 5 minutes, that is, the maximum reset time is set at 5 minutes.

NFR 21 The server shall keep a status if it gets down.

When the server gets back online, it reloads a previous copy of the web service that was running before the crash**.**

Legal Requirements:

NFR22. Third parts

Data stored by the system will not be shared with third parties, this company does not use the users' data with commercial purposes in any case.

NFR24. Deletion of personal data

The system will allow any user to ask for the deletion of his data on the app in order to let him decide his degree of privacy.

NFR25. Protection Regulation laws of the European Union

The system will comply with legal requirements enforced by the General Data Protection Regulation laws of the European Union.

2.3 DELIVERABLES

First the software company must accept the project plan, after that development will start. The development will start, using an incremental-iterative life cycle, with 3 increments, each consisting with phases for design, coding and unit testing which will be performed in parallel.

The Requirements Analysis will be common, as the Integration Testing, the increments should not advance from the Requirements Analysis until they have been revised, and Unit Testing only be done once the Implementation has finished.

Increment	Products	Date
1	PRODUCT 1	19/11/2021
2	PRODUCT 2	23/05/2022
3	FINAL PRODUCT	20/7/2022

Figure 5: Deliverables

3 SOLUTION TO EMPLOY

The system proposed as a solution should meet all the requirements (functional and non-functional) stated by the client and specified deeply on this document. Considering all the constraints and performance demand, the solution to employ should be the following.

3.1 ARCHITECTURE

The system will be based on a three-tier architecture, those tiers should be the client layer, the application layer, and the data layer.

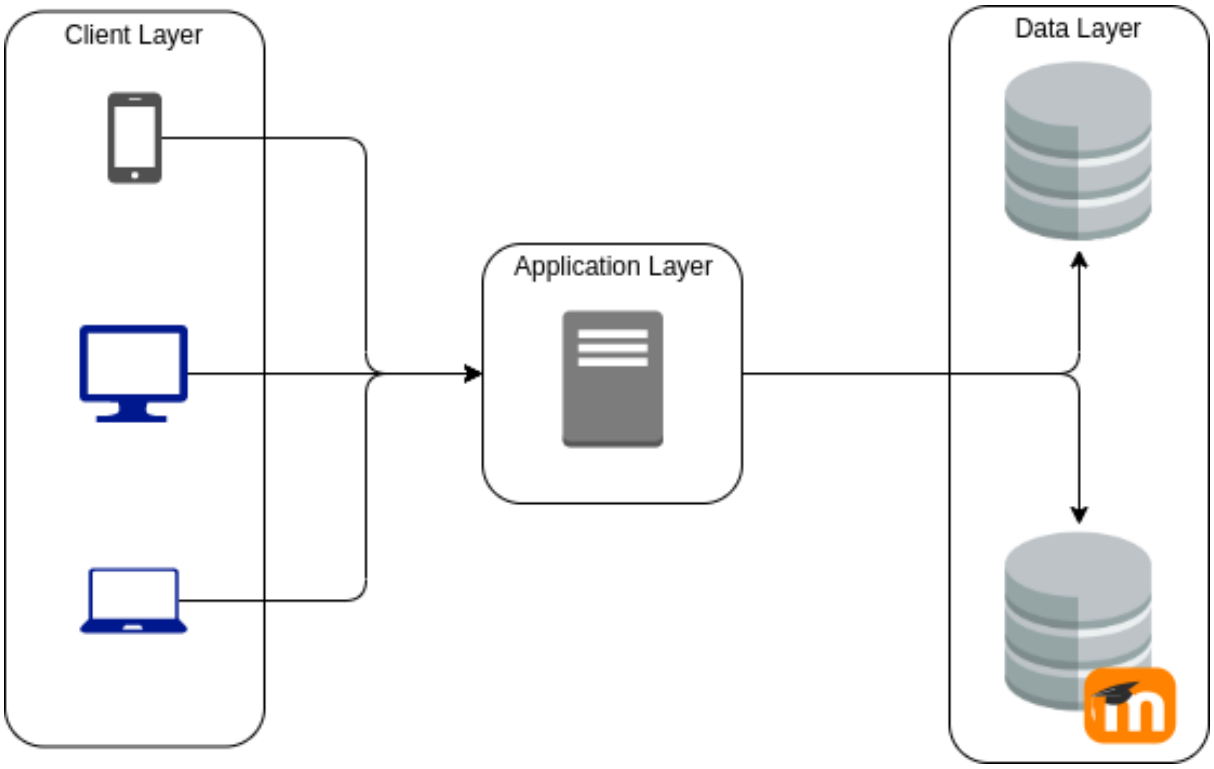


Figure 6: Architecture Design

As shown on the diagram, this architecture provides more flexibility to scale the system horizontally, for example if it is needed to add more servers to the application layer this can be done just adding a load balance system to distribute the client petitions among the available servers, this is very appropriate for this system since it should handle a medium-high size of client flow, and it should always stay high on performance and security. The machines of this tier may use a Linux based operating system that is suitable for a web server, they may also be

capable of handling up to 2000 users as stated on one of the non-functional requirements.

It can also be seen that the clients are heterogeneous, meaning that they are not all the same, having different software and hardware, the machines of this tier may use different web browsers to access the system, such as Firefox, Safary, Chromium and have different OS such as Android, Windows, MacOS, iOS.

Furthermore, on the right side we can see the two databases which the system needs in order to properly work, the bottom one is the Moodle's database where academic related data is retrieved and the top one is the system database where the repositories of the students are stored. These databases should store the data securely (encrypted), and will be SQL based databases, performing backups of their stored information each day at 2 a.m. using Amazon Services to ensure the integrity and protection of the information.

3.2 TECHNOLOGIES

For the development of this system an object-oriented based python framework will be used, due to its ease of use and flexibility that framework should be Django 3.1.3. The development team is experienced in python and in the Django framework, furthermore, python has ports of most of the libraries needed for the development of the application such as OpenSSL to manage security aspects of the application.

More intensive tasks such as calendar schedule or student data analysis for teachers reports may be implemented in a lower-level programming language such as C, which is more CPU efficient.

The communication with the clients and databases will be automatically handled by Django, developers may just need to specify the architecture of the back end and Django will take care of mapping the abstract models and communicate them to that specific back-end architecture.

Some of the external APIs that may be used to meet all the requirements of the application are the Esendex gateway and the one from Microsoft Teams, the first one will be in charge of sending notifications over SMS to the users (if they have

selected to be notified as so), and the second one will allow students to have meetings over voice and/or video.

For the internal database in charge of storing the students' repositories, a PostgreSQL database will be used, since the development team is familiar with that type of databases and knows how to manage and perform analysis on it.

3.3 LIFE CYCLE MODEL

The system should be developed according to a software life cycle model of "incremental and iterative" type, using the following phases at each increment: requirements analysis, design, coding, unit tests, integration tests and deployment. A schema of how this life cycle works can be seen in the figure below.

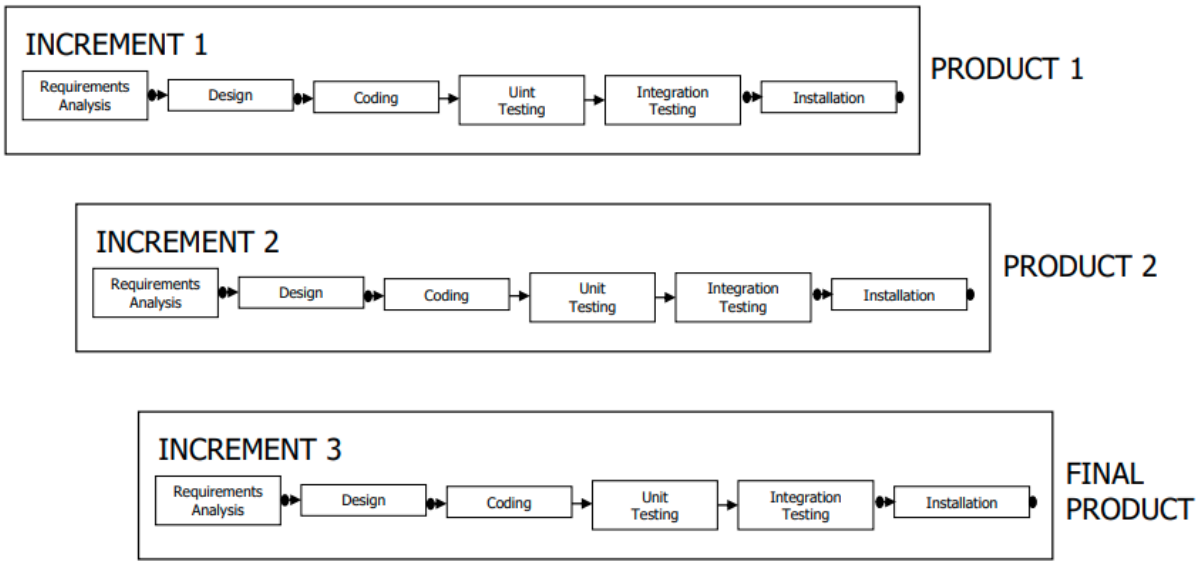


Figure 7: Life Cycle

During the development of the system, four milestones for each intermediate and final product should be met. One at the end of the requirements analysis, at the end of the design, at the end of integration tests and at the end of the installation.

When doing the increments, we must consider that the requirements analysis shall be performed in a unified way for the whole system/increment.

The phases of design, coding and unit testing can be performed in parallel for each of the possible subsystems considered at each increment.

About precedence, the phases of design, coding and unit testing only shall begin once the requirements analysis has been revised. Each subsystem implementation shall begin once the subsystem design has been revised; unit testing shall begin once implementation is finished.

For each increment, the integration testing phase shall begin when all the unit tests of the various subsystems are completed.

The installation phase shall start once the integration tests have been revised.

4 PROJECT MANAGEMENT

In this section we will discuss the estimation procedures and results obtained by analysing the size of the project. Given that results on time and costs estimates, an organization structure will be suggested, assigning responsibilities and timings to all the members of the development team.

4.1 ESTIMATIONS OF THE SOFTWARE SYSTEM

To perform the estimations of the software system we will first analyse each subsystem of the application using the Functions Points techniques.

After adjusting those estimations of complexity, we will be able to make an estimation of the cost and time to market of the project, as well as the people/month needed for each subsystem to be implemented.

4.1.1 UNADJUSTED FUNCTION POINTS

This technique is known as the Function Points technique, that considers all the software system requirements (with its inputs, processes, and outputs) and makes an estimate of the complexity of the software system based on them.

This method provides an objective way of measuring the complexity of a project, avoiding personal biases. In the following sections an estimate of each subsystem complexity using the Unadjusted Function Points will be provided.

4.1.1.1 TEAM MANAGEMENT

	COMPLEXITY						
	Simple	Average	Complex	Simple	Average	Complex	Unadjusted FP (UFP)
Data Functions	Frequency			Weighting			
Internal Logical File (ILF)	***0***	***0***	***0***	*7*	*10*	*15*	0
External Interface File (EIF)	***2***	***0***	***0***	*5*	*7*	*10*	10
Transaction Functions							
External Input (EI)	***7***	***0***	***0***	*3*	*4*	*6*	21

External Output (EO)	***5***	***2***	***0***	*4*	*5*	*7*	30
External Inquiry (EQ)	***5***	***1***	***0***	*3*	*4*	*6*	19
						TOTAL	**80,0**

Figure 8: UFP of Team Management Subsystem

4.1.1.2 MEETING SCHEDULING

	COMPLEXITY						
	Simple	**Average**	**Complex**	**Simple**	**Average**	**Complex**	**Unadjusted FP (UFP)**
Data Functions	**Frequency**			**Weighting**			
Internal Logical File (ILF)	***0***	***0***	***0***	*7*	*10*	*15*	0
External Interface File (EIF)	***1***	***0***	***0***	*5*	*7*	*10*	5
Transaction Functions							
External Input (EI)	***3***	***0***	***0***	*3*	*4*	*6*	9
External Output (EO)	***1***	***2***	***0***	*4*	*5*	*7*	14
External Inquiry (EQ)	***0***	***0***	***3***	*3*	*4*	*6*	18
						TOTAL	**46,0**

Figure 9: UFP of Meeting Scheduling Subsystem

4.1.1.3 MEETING MANAGEMENT

	COMPLEXITY						
	Simple	Average	Complex	Simple	Average	Complex	Unadjusted FP (UFP)
Data Functions	Frequency			Weighting			
Internal Logical File (ILF)	***1***	***0***	***0***	*7*	*10*	*15*	7
External Interface File (EIF)	***1***	***0***	***0***	*5*	*7*	*10*	5
Transaction Functions							
External Input (EI)	***5***	***0***	***0***	*3*	*4*	*6*	15
External Output (EO)	***4***	***1***	***0***	*4*	*5*	*7*	21
External Inquiry (EQ)	***1***	***1***	***2***	*3*	*4*	*6*	19
						TOTAL	**67,0**

Figure 10: UFP of Meeting Management Subsystem

4.1.1.4 MEETING-MAKING

	COMPLEXITY						
	Simple	Average	Complex	Simple	Average	Complex	Unadjusted FP (UFP)
Data Functions	Frequency			Weighting			
Internal Logical File (ILF)	***1***	***0***	***0***	*7*	*10*	*15*	7
External Interface File (EIF)	***1***	***0***	***0***	*5*	*7*	*10*	5
Transaction							

Functions							
External Input (EI)	***5***	***0***	***0***	*3*	*4*	*6*	15
External Output (EO)	***3***	***0***	***0***	*4*	*5*	*7*	12
External Inquiry (EQ)	***3***	***3***	***1***	*3*	*4*	*6*	27
						TOTAL	**66,0**

Figure 11: UFP of Meeting Making Subsystem

4.1.1.5 STATISTICS MANAGEMENT SUBSYSTEM

	COMPLEXITY						
	Simple	**Average**	**Complex**	**Simple**	**Average**	**Complex**	**Unadjusted FP (UFP)**
Data Functions	**Frequency**			**Weighting**			
Internal Logical File (ILF)	***0***	***0***	***0***	*7*	*10*	*15*	0
External Interface File (EIF)	***0***	***0***	***0***	*5*	*7*	*10*	0
Transaction Functions							
External Input (EI)	***2***	***0***	***0***	*3*	*4*	*6*	6
External Output (EO)	***2***	***0***	***0***	*4*	*5*	*7*	8
External Inquiry (EQ)	***0***	***0***	***2***	*3*	*4*	*6*	12
						TOTAL	**26,0**

Figure 12: UFP of Statistics Management Subsystem

4.1.2 ADJUSTMENT FACTOR

Once the UFP have been obtained from each subsystem we must set an Adjustment Factor, rating from 0 to 5 the different General Features of our specific project.

This method considers 14 characteristics that describe the system that we will develop:

Complexity Factors	TDI
Data Communications	3
Distributed Data Processing	4
Performance	3
Heavily Used Configuration	2
Transaction Rate	2
On-line Data Entry	5
End-User Efficiency	2
On-line Update	4
Complex Processing	1
Reusability	0
Installation Ease	0
Operational Ease	2
Multiple Sites	3
Facilitate Change	4
TOTAL	**35,0**

Figure 13: Complexity Factors

The justifications for these values are:

Data Communications 3:

Online data input to a query system.

The system is a web-based application with only one front-end and all the clients get to render the same one (with the aid of its browser). The system

The system should support at least 1 communication protocol, HTTP to interact with the clients and the Moodle database.

Distributed Data Processing 4:

The distributed process and the data transfer are online and in both directions.

It is in both directions, as the web application is independent from Moodle, so it has communicated with the modules of Moodle and call its functions to, for example, add a team, or add a meeting to the schedule, and it is both ways as it will return all the information related to students, subjects and more.

Performance 3:

Response time or process capacity are critical at all operation times, but they do not require any special design for the CPU use.

It is required that the system shall carry out high demand operations, such as creation of teams for practical assignments and/or meetings scheduling and creation in the shortest possible time to be defined by the team. This means that the system response time or process capacity are critical at all operation times.

Strongly Used Configurations 2:

There are some restrictions of security or time related.

Due to the system features, the population of concurrent users is currently expected to be of medium-high size; therefore, special considerations shall be required concerning performance and security of transactions.

Transaction Rate 2:

The weekly rush period is known.

The frequency rate is very distributed between weekdays, albeit weekends and Fridays surely have smaller rates, as there will not be any deliveries and most of the activity will be of meetings if any.

On-line Data Entry 5:

More than 30% of the inputs are interactive.

It is required that all data input shall be through online processes, this means that 100% of the inputs are interactive.

Efficiency of Final User 2:

4-5 functions.

To make easier the interaction between system and user, usability mechanisms such as feedback, simplified help and language selection shall be provided.

The application should run on a window different from Moodle.

On-line update 4:

Besides the previous conditions, the protection against data loss is essential and it has been especially designed and programmed in the system.

It is essential for the system to avoid plagiarism between the teams for practical assignments, because of that, it is important to make the biggest effort in terms of security. This specially affects repositories.

Complex Processes 1:

Especial controls (auditoria) and/or security applications and use of complex devices (multimedia, etc.).

To avoid plagiarism between the teams for practical assignments, it is important to make the biggest effort in terms of security.

The system should handle a repository with version control and a video call system, which both include interaction with complex devices.

Reusability 0:

Not reusable.

The application is tailor made for a specific purpose, hence there is no need for the system to be reusable, as the client has not demanded it.

Ease of Installation 0:

There are no special installation requirements, and no especial developments are needed.

The application requires no special considerations or developments for its installation.

Operational Ease 2:

No requirements specify the minimization of tapes nor paper, but the recovery and boot must be efficient as specified by the requirements the system shall not spend more than a certain amount of time recovering and booting.

Multiple Sites 3:

The application must be designed to be used in multiple locations, but the hardware and software environments will be different.

The machines of the client layer shall be all types of devices (PCs, tablets, mobile devices, etc.) so the hardware will vary all across the board, moreover a responsive design is expected to work in many types of browsers (as specified in the NFR) so we must support many different software.

Ease of Change 5:

Easy to perform reports or queries of medium complexity, such as using logical AND/OR operators over more than one internal logical file (value: 2).

Like in the previous case, but changes will be effective immediately (value: 2).

The system should only need to execute and/or operators over the Moodle database and the changes on data should be available for all the users of the system immediately.

4.1.3 ADJUSTMENT FUNCTION POINTS

Once we have the function points unadjusted and owning the adjustment factor, we can compute the adjusted function point which is given by the formula:

$$FP = UFP * AF$$

Then, we obtain the following result.

Subsystem	Unadjusted Function Points (UFP)	Adjusted Function Points (AFP)
Team Management	83	83
Meeting Scheduling	48	48

Meetings Management	64	64
Meetings Making	68	68
Statistics Management	26	26
Total	289	289

Figure 14: Adjusted Function Points

The total Adjusted Function Points of the system is 289 and the Unadjusted Function Points is 289, we can see that they are the same numbers, this is due to the AF being equal to (TDI*0.01) + 0.65, and having in this case TDI = 35, therefore AF = 1.

Increment	Subsystems	Effort		Percentage (%)	
1	Team Management	83	131	63.36	45.33
	Meeting Scheduling	48		36.64	
2	Meetings Management	64	132	48.48	45.67
	Meeting-Making	68		51.52	
3	Statistics Management	26	26	9	

Figure 15: Effort Percentage per Subsystem

On the figure above, figure 15, we can see the distribution of effort between each increment and subsystem and the percentages of each ones too.

4.2 ORGANIZATIONAL STRUCTURE

In this project we will have 5 people working, the assigned personal for this task is:

ROLE	NAME	RESPONSABILITIES	SALARY

Project Manager	Samuel de Luque	Project Management	400
System Analyst	Samuel de Luque	Requirements Analysis, Design, Integration Tests, Installation	400
Senior Designer	Alejandro Bravo	Design, Coding and Unit Tests, Integration Tests	350
Junior Designer	Ibai Llanos	Design, Coding and Unit Tests, Integration Tests	200
Junior Designer	Raúl Álvarez	Design, Coding and Unit Tests, Integration Tests	200
Systems Technician	Borja Pavón	Installation	300

Figure 15: Team roles

In this case the Project Manager will be the same as the System Analyst and will oversee handling the project and talking to the clients, (in this case UAM).

Here is an organization chart of our project's personnel.

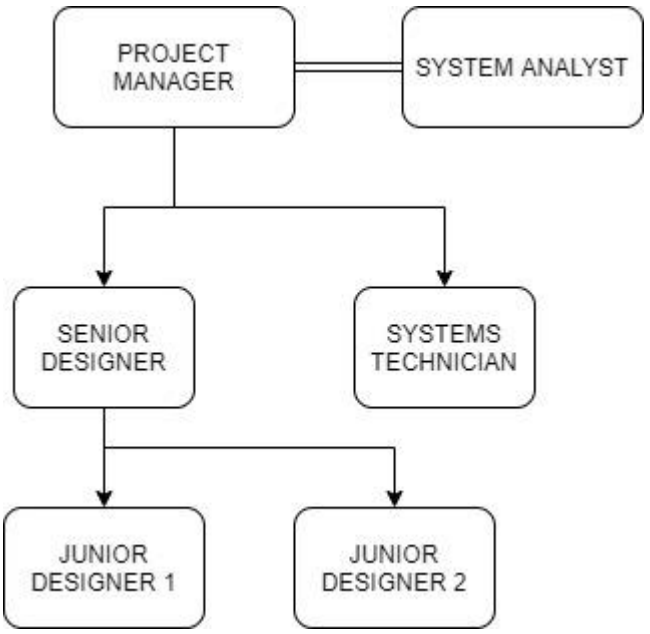


Figure 16: Team roles

4.3 ASSIGNED QUALIFIED PERSONNEL

As it has been explained in section 3, our project is going to be spited into 3 increments, each of these periods has been divided in five activities: Requirements Analysis, Design, Coding and Unit Testing, Integration testing and Implementation. In the following table we are comparing the time that we spend doing each of these activities.

Nombre	Comienzo	Fin	Trabajo restante
System Analyst	lun 19/04/21	mié 20/07/22	676,37 horas
Senior Designer	lun 24/05/21	vie 01/07/22	896 horas
Junior Designer 1	jue 10/06/21	mié 13/07/22	1.448 horas
Junior Designer 2	jue 10/06/21	mié 13/07/22	1.448 horas
Systems Technician	lun 25/10/21	mié 20/07/22	368 horas
Equipment Cost	lun 19/04/21	mié 20/07/22	2.624 horas

Figure 17: Personnel Work Amount

4.4 TIME MANAGEMENT

We generated a GANTT diagram included in the ANNEX D and in the MS Project, file attached in the delivery, here is some information.

The time dates consist of increment 1 goes from 19/04/21 to 19/11/21 lasting 155 days and increment 2 goes from 22/11/21 to 23/05/22 lasting 131 days and increment 3 from 24/05/22 to 20/07/22 lasting 42 days. Overall, the whole project takes 328 days.

Inside each increment subsystems are parallelized in design, coding, and unit testing, that is, one subsystem can advance to the next phase even if the other from the same increment is not ready. But Integration Testing, Analysis and Installation is common for all the increment.

We divided each increment in 4 milestones, analysis, design, integration, and installation. Two of those milestones (analysis and design) are split in two, as we already mentioned we wanted to parallelize two subsystems per increment.

INREMENT 1	MILESTONE DATE
Analysis	09/06/21
Design TM.	13/07/21
Design MS.	18/06/21
Integration	22/10/21
Installation	19/11/21
INCREMENT 2	MILESTONE DATE
Analysis	14/01/22
Design Mmanag.	11/02/22
Design Mmaking.	28/01/22
Integration	22/04/22
Installation	23/05/22
INCREMENT 3	MILESTONE DATE
Analysis	03/06/22
Design SM	15/06/22
Integration	13/07/22
Installation	20/07/22

Figure 19: Time Management

Even though we tried to parallelize as much as possible there where some dependencies, here are those we considered:

- Analysis, Integration, and Installation subtasks are done sequentially.
- Even though each subsystem is parallelizing they still follow Analysis, Design, Coding, Unit Testing, Integration, and Installation.
- Integration does not start until both subsystems are done.
- To avoid overusing our resources we set dependencies between increments so that they are estimated sequentially.

We calculated the effort based on experience in the company:

- Requirements Analysis – 20%
- Design – 20%
- Coding and Unit Testing – 30%
- Integration Testing – 20%
- Installation – 10%

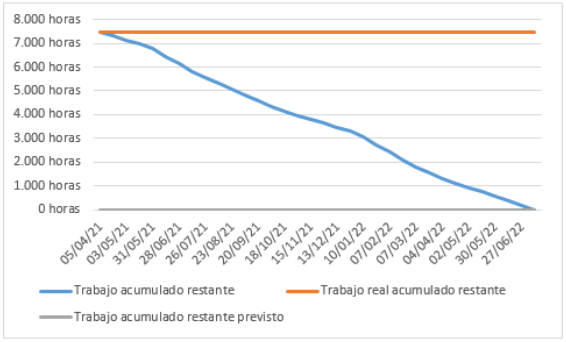
Applying these criteria to our subsystems we obtained the effort per subsystem and stage:

		Subsystems				
		TM	MS	MManag.	MMaking.	SM
Phases	Analysis	16,6	9,6	12,8	13,6	5,2
	Design	16,6	9,6	12,8	13,6	5,2
	Coding	12,45	7,2	9,6	10,2	3,9
	Unit Testing	12,45	7,2	9,6	10,2	3,9
	Installation	8,3	4,8	6,4	6,8	2,6

As you can see in this burndown diagram, we spread in time the tasks, this is to not have to rush some months and keep our workers busy in an efficient way during all the projects.
This is the estimation of the effort that our developing team have to accomplish for finishing the final product in the desired day.
In the first plot we have planned this comparative by looking to the total hours of remaining work and in the second one we have focus on the number of tasks that we need to finish yet.

lun 19/04/21 - mié 20/07/22

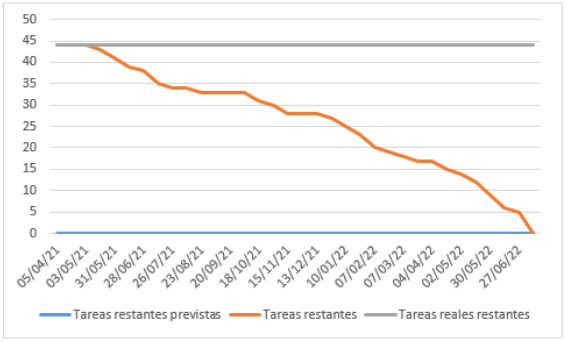
EVOLUCIÓN



EVOLUCIÓN DEL TRABAJO

Muestra la cantidad de trabajo completado y la cantidad que ha quedado sin completar. Si la línea del trabajo acumulado restante es pronunciada, puede que el proyecto esté atrasado. ¿La línea base es cero?

Intente establecer una línea base



EVOLUCIÓN DE LA TAREA

Muestra cuántas tareas se han completado y cuántas han quedado sin completar. Si la línea de las tareas restantes es pronunciada, puede que el proyecto esté atrasado.

Más información

Figure 20: Project Evolution

4.5 COSTS MANAGEMENT

We had made an analysis with the MS Project tool and we have seen the cost of the different personnel depending on the role that he has on the company.

DETALLES DE COSTOS

Detalles de costos de todos los recursos de trabajo.

Nombre	Trabajo real	Costo real	Tasa estándar
System Analyst	0 horas	0,00 €	400,00 €/día
Senior Designer	0 horas	0,00 €	350,00 €/día
Junior Designer 1	0 horas	0,00 €	200,00 €/día
Junior Designer 2	0 horas	0,00 €	200,00 €/día
Systems Technician	0 horas	0,00 €	300,00 €/día
Equipment Cost	0 horas	0,00 €	1.050,00 €/ms

Figure 21: General View of the Project

As we can see in this plot, the major cost of the project is the salary of our workers but eventually it is going to be worth because they have a long experience in the sector and will provide the better solution for the client necessities, so as we use to say, “you only pay it once” since it shall not be necessary to make modifications in the future.

Also, we must dedicate a part of the estimate to pay the undirect spends, such a proportional part of the office rent, office material and undirect workers. those things make the monthly add of 1,050€ per mount so:

Equipment Fees	1,050 €	1,050 €/month 17,220 € whole project

COST SUMMARY PLOT

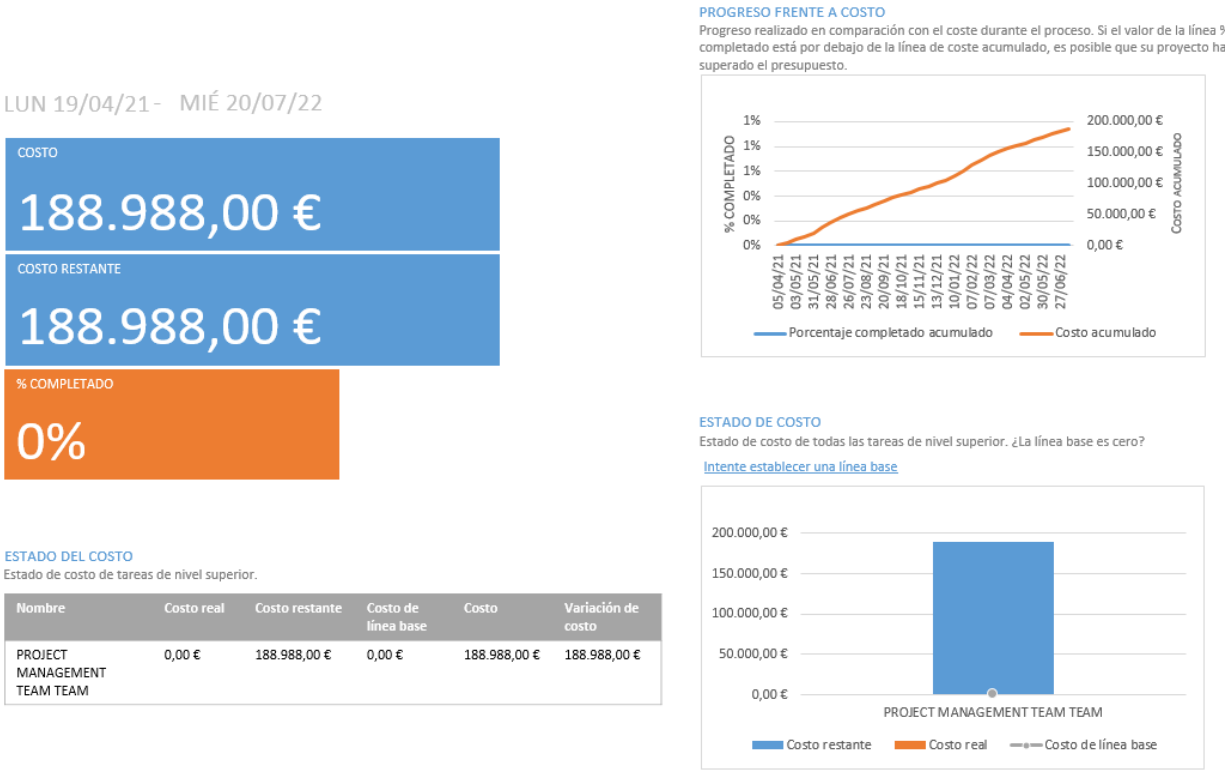


Figure 22: Cost Summary Plot

In this other plot we can obverse the part of estimate of the project that we have consume up to the date of the publication of this document. As we are now in the analysis phase and the project itself have not begun yet, the remaining cost is the total cost of the project, but we have included anyway to compare this plot with the future plots of the design phase, that is, in order to be consistent with the future documentation that we will send the customer in the case they rely on us to develop the project.

After defining the project plan throughout the previous sections, one of the aspects that was not considered inside the project cost section was the infrastructure and the non-developer staff cost.

We need to consider the cost of the Runner Solutions office rent as well as the cost of the maintenance staff.

4.6 QUALITY MANAGEMENT

During the development of the project, we will have revisions of the work done after the conclusion of almost all phases. These revisions will be useful to detect errors as soon as possible and to avoid spreading these errors to successive phases.

When we detect the errors, all the products involved should be revised and its versions modified. Also, the maintenance of the application will be done solving the minor problems during the date stablished with the client, and the serious ones will be solved once each month. Details about this can be found on section 5.

4.7 RISKS MANAGEMENT

See the Risk Management Plan [8] of the TEAMTEAM application.

4.8 ACQUISITIONS MANAGEMENT

For the development of the project, we need to consider the acquisition of three workstations with a cost of 1,650 € each one. For each workstation we need a development environment with a cost 1,100 € for each one.

Also, for testing, we need to invest in another workstation, with a cost of 3,200 €. And although it is not an acquisition, a cost that we should also consider is that for the necessary equipment, both hardware and software, we need to pay 1,050 €/month.

An important part of the cost's estimation is the hardware and software used. We need to include the hardware and software:

Equipment	Price per Unit	Total
Workstation (x3)	1,650 €	4,950 €
Workstation Performance Testing	3,200 €	3,200 €
Integrated Development Environment (x3)	1,100 €	3,300 €
Total		11,450 €

Figure 23: Hardware Resources and Development Software Costs

4.9 DOCUMENTATION MANAGEMENT

Each development phase will come along with its documentation, analysis, design, and code documentation will be created and classified by subsystems.

This documentation will be iterative, including the new features of each increment on the new version of the corresponding document, in order to ease the possible expansion of the software system in future releases.

The system will be provided along a user manual, which should help on the interaction of the user with the system. A technical manual will also be provided, to help on the maintenance of the software in the event it would be necessary.

5 MONITORING AND CONTROL

In this section the parts of the system that should be subject to monitoring and control throughout the project development may be specified. The different system quality checks, such as verifications, testing and monitoring will be explained.

5.1 CONFIGURATION MANAGEMENT

The configuration items of the project are explained in this document (software requirements, system requirements, database information, design documentation) and will be controlled by a version control software such as git, to manage and keep track of the different changes made to the configuration.

The need for a change on the system configuration could be activated via two ways, by the customer or by the contractor or sub-contractors.

No changes will be introduced into the project unless the project manager approves it, and the estimated costs of the change does not exceed 5% of the project cost estimated in this document for the project.

All change requests may be communicated by the customer, contractor, or sub-contractors to the project manager during the different phase revisions that will take place during the development of the project, those changes will be handed via a standard change request document that will be provided in future documents.

Once a change has been accepted, a review of the current project plan must be assessed in order to check whether the standards are followed and introduce the modifications needed to adjust the management of the project development to the new configuration.

5.2 PROGRESS MONITORIZATION

The principal milestone of the progress monitorization is to provide an objective view of the project process related to the schedule, in comparison with the initial estimations.

The monitorization will be based in the plot, the requirements and in critical items analysis.

We will focus specially in:

- Follow the datelines stablish in the grant diagram.
- Update the monitorization plots such as bar chart or burndown schemas.
- Ensure that all the requirements that had been listed in section 4 are successfully fulfilled without deviations.
- Our clients are up today in any relevant matter regarding the project, so any modification will be reported as soon as possible.

In order to do this, we will make several meetings with the developing team, and with the client, firstly two weeks before the project's design phase begins, we will hold the first meeting for presenting our ideas of the project. Then we will have another meeting each time a milestone is achieved, in such a way that our customers will see the evolution of the software step by step and they could share their impressions with the developing team.

Also, each member of the developing team that reach a milestone will revise that the software that he had done fits the requirement exposed in this document before doing the corresponding meeting.

In order to satisfy the quality requirements that we have exposed before we have developed a proving planning that is explained in detail in section 5.4.

Finally, we have stablished that in case something unexpected happens and we run out the dates that were marked in the grant diagram, the Project Leader will have the responsibility of remaking the organization diagrams and to communicate all the decision to his team.

5.3 VERIFICATIONS AT EACH PHASE

The project development is based on an incremental life cycle (explained in section 3.3), which consists of 6 different phases: Requirements Analysis, Design, Coding, Unit Testing, Integration Testing, and Installation.

At the end of each of the mentioned phase a meeting should be held to check whether the objectives proposed for that phase have been met, in order to consider a new milestone of the project development.

The verification of each phase should allow the traceability of all the requirements implemented, monitoring of the overall verification process during the project life cycle and flexible reporting,

Those datapoints should be reflected on the Verification Control Document, this document will be maintained throughout all the phases of the development of the project.

For the requirement analysis phase, the verification will consist of detecting requirements inconsistencies and ambiguities among them.

The system design obtained from the design phase will be assessed on its flexibility, modular cohesion, and coupling. The correctness of the design regarding the requirements stipulated by the client will also be verified.

When the coding phase ends, unit tests and integration tests will follow in order to ensure the correctness of the code and the system implemented, making sure the system features what the client demanded and nothing else.

At the end of the development of the project, the whole project will be checked, in order to verify whether it fulfils all the requirements stated by the client and the final documentation, user and technical manuals will be thoroughly analysed by the project manager, making sure they are clear and complete.

5.4 TESTING AND VALIDATION

For guaranteeing that all the requirement listed in section 4 are fulfilled we will develop a testing plan. This plan will contain dynamic proves such as black box and white box where we will see try to see as many errors as possible in the sake of correcting them before showing them to the client.

As we have a limited estimate, we cannot afford to do white box proves in every module we will define our own technique in order to see which ones have a higher likelihood to crash. So, we have decided to make white box proves only to the

modules that are related with the Meeting Scheduling and Meeting Statistics subsystems due to those subsystems have more connections than the others and are thinner as well. The rest of the modules will be evaluated with the black box technique at module level but anyway the team leader will check that each requirement is fulfilled.

Once we have declared the way in which we are going to make the test, we are going to define the frequency in which we are going to do the tests. Each time an updated version is closed, it is, we have achieved a milestone, we will pass those tests to each module that had been modified, and then an integration test will be made by the quality manager in the sake of checking the usability of the hole system.

At the end of each increment, we will make additional tests in order to see if the project follow the non-functional requirements, we have decided to develop extra test regarding performance, security, portability and documentation and after we run those tested, we will ask our client to do a usability test, that is, we are going to ask the client to use the graphical user interface of the application to avoid ambiguities between two the parts.

If the client accepts the last increment, the software product will be delivered and a new base line will be created, indicating that the proving phase have finished.

6 REMARKS

We reckon that is quite sensitive to talk about the actual situation that we are leaving due to the Covid-19 pandemic we must have in mind that it is possible that the situation goes wrong, and we could enter in a new lockdown.

So, we have to prepare a “plan B” in case everyone needs to work from home, so just in case, we will promote one member of the developing team, as Covid coordination, this member shall be responsible of keeping the track about the new laws and in case it is necessary, he must inform the team leader for making the modifications in the schedule that process with the situation.

The aim of this improvement is to minimize the chain effect that a sudden halt may have in the whole process of development.

7 CONCLUSIONS

In this document we have discussed the project plan that will be followed during the implementation of the TEAMTEAM system. Following the stated plan present on this document should ensure the quality of the system developed, obtaining successful results on a reasonable time and budget.

To analyse the size of the project we have used the Function Points technique, based on the requirements given for the project. After gathering the Function Points, a Gantt Diagram has been developed using the MS Project tools, from this analysis we have obtained a developing time of **328 days**, and a budget of **188.988 €**, that considers the resources, effort and staff estimated costs.

To ensure that the time a cost budget estimates are met, a monitoring plan has been designed, to ensure the quality and accomplishment of the system. Different milestones have been defined throughout the development process as well as a verification and validation plan to ensure the correctness of the system.

Performing the actions described above we can reassure the client that the project will be accomplished successfully and guarantee that all the requirements stated will be met, as he/she could review the documentation generated at each step of the development process to ensure that the system is being developed according to its needs.

ANNEXES

ANNEX A. FUNCTIONS POINTS

A.1 FUNCTIONAL REQUIREMENTS

In order to assign the parameters of all the requirements, we have made a Database with the following diagram:

In Team Team´s Database:

Tables:

Repository (Name, Subject, Last Update, Data, Size, Checksum).

In Moodle´s Database:

Tables:

- Subject (Name, Meetings, Assignments).
- Assignment (Name, Deadline).
- Team (Name, Maximum of students per group, Minimum of students per group, Team participants, Subject, Assignment).
- Student (Name, Email, Password, Subject, Teams).
- Teacher (Name, Email, Password, Subject).
- Notification (Subject, Type, Body, Sender, Receiver, Isread).
- Calendar (Owner, Last modification, Date).
- Meeting (Date, Duration, Log, plots, Name, Tasks, Proposer, Participant, Assignment).

In Call Provider´s Database:

Tables:

- Room (Allowed students, IsOpen).

Moodle and the Call Provider had shared their database diagrams because our application needs to query them in order to get and insert information.

We have used this information and the mock-up for assigning the RETs and DETs in the following annexes.

A.1.1 TEAM MANAGEMENT SUBSYSTEM

FR1. Creation of a Team.

The user can create a team with the required information.

EXTERNAL INPUT:

DET:

1. Subject Name
2. Team Name
3. Maximum of students per team
4. Minimum of students per team
5. Names of students
6. Create button
7. Broadcast button (We do not count the invite button, because it is used the broadcast or the invite, but not at the same time).

FTR:

1.

CONCLUSION:

Complexity is Low as we have 7 DET.

EXTERNAL OUTPUT:

DET:

1. Success story

CONCLUSION:

Complexity is Low as we have 1 DET.

FR2. Broadcasting

Teams may broadcast and invite to all users that are not included in any team for that subject.

EXTERNAL INPUT:

DET:

1. Team name
2. Subject's students
3. Broadcasting checkbox

CONCLUSION:

Complexity is Low as we have 3 DET.

EXTERNAL OUTPUT:

DET:

1. Success story

CONCLUSION:

Complexity is Low as we have 1 DET.

FR3. Joining a team not full

The user may join a team at any time if it is not full.

EXTERNAL INPUT:

DET:

1. Team name
2. OK Button

CONCLUSION:

Complexity is Low as we have 2 DET.

EXTERNAL OUTPUT:

DET:

1. Team participants
2. Maximum of students per team
3. Minimum of students per team
4. Subject name
5. Assignment name
6. Student teams
7. Student name
8. Calendar
9. Notifications (number of non-read)

FTR:

1. Team
2. Calendar
3. Notification

CONCLUSION:

Complexity is Medium as we have 9 DET and 3 FTR.

FR4. Asking to join a team.

The user may ask to join a team instead of being invited.

EXTERNAL INPUT:

DET:

1. Team name
2. Accept button

CONCLUSION:

Complexity is Low as we have 2 DET.

EXTERNAL OUTPUT:

DET:

1. Success story

CONCLUSION:

Complexity is Low as we have 1 DET.

FR5. Leaving a team

The user shall not be able to leave a team once they join, as well as not be able to dissolve it until all practical assignments are done.

EXTERNAL INPUT:

DET:

1. Team name
2. Leave button

CONCLUSION:

Complexity is Low as we have 2 DET.

EXTERNAL OUTPUT:

DET:

1. Team per subject (If the user is able to leave)
2. Student (If the user is able to leave)

3. Success or fail story

CONCLUSION:

Complexity is Low as we have 3 DET.

FR6. Notifications

Notifications for cancelling or creating meetings.

EXTERNAL INPUT:

DET:

1. Meeting name
2. Input info when creating a meeting
3. Cancelling reason
4. Cancel button

CONCLUSION:

Complexity is Low as we have 4 DET.

EXTERNAL OUTPUT:

DET:

1. Notification subject
2. Notification body

FTR:

1. Notification

CONCLUSION:

Complexity is Low as we have 2 DET and 1 FTR.

FR7. Signing In

Signing in must be independent from Moodle, and it is required for doing anything related with teams.

EXTERNAL INPUT:

DET:

1. User's email
2. Password
3. Sign in button

FTR:

1. User (student or teacher)

CONCLUSION:

Complexity is Low as we have 3 DET and 1 FTR.

EXTERNAL OUTPUT:

DET:

1. Calendar
2. Notifications (number of non-read)
3. Username
4. Subject's name

FTR:

1. Calendar
2. Notification
3. Student
4. Subject

CONCLUSION:

Complexity is Medium as we have 4 DET and 4 FTR.

A.1.2 MEETING SCHEDULING SUBSYSTEM

FR8. Schedule Access

The registered user may access its meeting schedule where he will find the completed or pending meetings. This meeting schedule shall be integrated into the Moodle calendar, apart from being accessible from the TEAMTEAM app.

EXTERNAL INPUT:

DET:

1. Schedule access button
2. Expand meeting button

CONCLUSION:

Complexity is Low as we have 2 DET.

EXTERNAL OUTPUT:

DET:

1. Calendar
2. Student subjects

FTR:

1. Calendar
2. Student

CONCLUSION:

Complexity is Low as we have 2 DET and 2 FTR.

FR9. Check the details of a past meeting.

From the meeting schedule the user may see the details of a meeting that has already taken place.

EXTERNAL INPUT:

DET:

1. Meeting name
2. Expand meeting button

CONCLUSION:

Complexity is Low as we have 1 DET.

EXTERNAL OUTPUT:

DET:

1. Meeting name
2. Subject
3. Team name
4. Date
5. Duration
6. Proposed by
7. Notification type
8. Attendee's name
9. Punctuality of each attendee
10. Remaining time

FTR:

1. Meeting

2. Notification

CONCLUSION:

Complexity is Medium as we have 10 DET and 2 FTR.

FR10. Check the details of a pending meeting.

From the meeting schedule the user may see the details of a future meeting.

EXTERNAL INPUT:

DET:

1. Meeting name
2. Expand meeting button

CONCLUSION:

Complexity is Low as we have 1 DET.

EXTERNAL OUTPUT:

DET:

1. Meeting name
2. Subject
3. Team name
4. Date
5. Duration
6. Proposed by
7. Notification type
8. Meeting link
9. Meeting objectives

FTR:

1. Meeting
2. Notification

CONCLUSION:

Complexity is Medium as we have 9 DET and 2 FTR.

A.1.3 MEETING's MANAGEMENT SUBSYSTEM

FR11. Notification management.

Users should select the type of notifications they wish to receive, they can choose from email, push notifications to desktop or mobile or message from a bot inside the app.

EXTERNAL INPUT:

DET:

1. SMS Checkbox
2. TEAMTEAM Checkbox
3. Email Checkbox

CONCLUSION:

Complexity is Low as we have 3 DETs.

EXTERNAL OUTPUT:

DET:

1. Notification subject.
2. Notification body.

FTR:

1. Notification

CONCLUSION:

Complexity is Low as we have 2 DET and 1 FTR.

FR12. Schedule a meeting.

Any member of a team can schedule a meeting (videoconference and joint working session) for practical assignments, any of its members can propose the call for a meeting for their team.

EXTERNAL INPUT:

DET:

1. Meeting name
2. Date
3. Time
4. Duration

5. Objectives
6. Assignment
7. Duration

CONCLUSION:

Complexity is Low as we have 7 DET.

EXTERNAL OUTPUT:

DET:

1. Meeting in calendar
2. Notification subject
3. Notification type

FTR:

1. Meeting
2. Subject
3. Calendar
4. Notification

CONCLUSION:

Complexity is Medium as we have 3 DET and 4 FTR.

FR13. Cancelling a meeting.

Any participant of the meeting should cancel a scheduled meeting, providing a reason why it is being cancelled.

EXTERNAL INPUT:

DET:

1. Reason for cancelling

CONCLUSION:

Complexity is Low as we have 1 DET.

EXTERNAL OUTPUT:

DET:

1. Success story

CONCLUSION:

Complexity is Low as we have 1 DET.

FR14. Meeting Development.

The user may associate meetings with practical assignments and may be able to create, close, open and leave a room.

EXTERNAL INPUT:

DET:

1. Click on meeting link
2. Click on leave room (if the system has not closed it yet)
3. Check the accomplished objectives

CONCLUSION:

Complexity is Low as we have 3 DET.

EXTERNAL OUTPUT:

DET:

1. Meeting window

CONCLUSION:

Complexity is Low as we have 1 DET.

FR15. Room access.

Only allowed students should enter the room, and they can do so as many times as needed.

EXTERNAL INPUT:

DET:

1. Meeting Link

CONCLUSION:

Complexity is Low as we have 1 DET.

EXTERNAL OUTPUT:

DET:

1. Meeting window

CONCLUSION:

Complexity is Low as we have 1 DET.

A.1.4 MEETINGS TOOL SUBSYSTEM

FR16. Task-board

During a meeting, students will be able to access a task-board with which they are going to distribute the work to do by each participant.

EXTERNAL INPUT:

DET:

1. Click on the task-board button

CONCLUSION:

Complexity is Low as we have 1 DET.

EXTERNAL OUTPUT:

DET:

1. Task board window

CONCLUSION:

Complexity is Low as we have 1 DET.

FR17. Collaborative tools

Registered users have access to chat, screen sharing, notebook, collaborative whiteboard, and a shared repository during the meetings.

EXTERNAL INPUT:

DET:

1. Notebook Button.
2. Repository Button.
3. Share Screen Button.
4. Live Chat Button.
5. Whiteboard button.

CONCLUSION:

Complexity is Low as we have 1 DET.

EXTERNAL OUTPUT:

DET:

1. Tool window selected

CONCLUSION:

Complexity is Low as we have 1 DET.

A.1.5 MEETINGS MAKING SUBSYSTEM

FR18. Roster of attendees

The system will keep the total minutes that each user has spent in the class.

EXTERNAL INPUT:

DET:

1. Click on a past meeting on the calendar

CONCLUSION:

Complexity is Low as we have 1 DET.

EXTERNAL OUTPUT:

DET:

1. Student name
2. Student minutes in the meeting

FTR:

1. Meeting

CONCLUSION:

Complexity is Low as we have 2 DET and 1 FTR.

FR20. Automatic delivery

Everything under the “Practical Assignment Submission” folder of the shared repository will be compressed and automatically delivered into Moodle by the system.

EXTERNAL INPUT:

DET:

1. Deadline
2. Assignment name

CONCLUSION:

Complexity is Low as we have 2 DET.

FR21. Leaving and joining the room.

Users may leave and join the room as long as it remains opened.

EXTERNAL INPUT:

DET:

1. Click on the leave button in a meeting

CONCLUSION:

Complexity is Low as we have 1 DET.

A.1.6 STATISTICS MANAGEMENT SUBSYSTEM

FR23. Practical assignment and group report

The teacher shall be able to ask for a report with the statistics by practical assignment and group.

EXTERNAL INPUT:

DET:

1. Group
2. Practical Assignment
3. Team
4. Make Report Button

CONCLUSION:

Complexity is Low as we have 3 DET.

EXTERNAL OUTPUT:

DET:

1. Percentage of meetings held.
2. Percentage of meetings cancelled.
3. Average duration of meetings
4. Percentage of meetings attended by each member of the team.

5. Percentage of meetings in which each member of the team entered and left.
6. Frequency of meetings during the development of each assignment
7. Time period when the meetings were held.

FTR:

1. Meeting

CONCLUSION:

Complexity is Low as we have 7 DET and 1 FTR.

FR24. Practical assignment and group report for the whole class

The system will keep track of the meetings and the work of the team at each practical

assignment, so the software system will allow to generate and visualize different statistics of

the meetings.

EXTERNAL INPUT:

DET:

1. Team
2. Practical Assignment.
3. Make Report button.

CONCLUSION:

Complexity is Low as we have 3 DET.

EXTERNAL OUTPUT:

DET:

1. Percentage of meetings held
2. Percentage of meetings cancelled
3. Average duration of meetings
4. Percentage of meetings attended by each member of the team

5. Percentage of meetings in which each member of the team entered and left
6. Frequency of meetings during the development of each assignment
7. Time period when the meetings were held

FTR:

1. Meeting

CONCLUSION:

Complexity is Low as we have 7 DET and 1 FTR.

A.2 DATA FUNCTION TYPE

A.2.1 SYSTEM DATABASE

NOTE: MOST ARE EIF BECAUSE IN OUR REQUIREMENTS ITS SPECIFIED THAT EVERYTHING IS STORED IN MOODLE'S DATABASE EXCEPT FOR THE REPOSITORY.

FR1. Creation of a Team.

The user can create a team with the required information.

EXTERNAL INTERFACE FILES (Moodle's database):

DET:

1. Subject Name
2. Team Name
3. Maximum of students per group
4. Minimum of students per group
5. Names of students
6. Emails of students

RET:

1. Teams per subject
2. Student

CONCLUSION:

Complexity is Low as we have 6 DET and 2 RET.

FR2. Broadcasting

Teams may broadcast an invite to all users that are not included in any team for that subject.

EXTERNAL INQUIRY:

DET:

1. Team name
2. Subject's students

RET:

1. Teams per student

CONCLUSION:

Complexity is Low as we have 2 DET and 1 RET.

FR3. Joining a team not full

The user may join a team at any time if it is not full.

EXTERNAL INQUIRY:

DET:

1. Team name

RET:

1. Teams per student

CONCLUSION:

Complexity is Low as we have 1 DET and 1 RET.

FR4. Asking to join a team.

The user may ask to join a team instead of being invited.

EXTERNAL INQUIRY:

DET:

1. Team name
2. Names of students
3. Subject's students

RET:

1. Teams per student
2. Student

CONCLUSION:

Complexity is Low as we have 3 DET and 2 RET.

FR5. Leaving a team

The user shall not be able to leave a team once they join, as well as not be able to dissolve it until all practical assignments are done.

EXTERNAL INQUIRY:

DET:

1. Team name
2. Subject Name
3. Names of students

4. Maximum of students per group
5. Minimum of students per group
6. Subject's students

RET:

1. Teams per student
2. Student
3. Subject data

CONCLUSION:

Complexity is Medium as we have 6 DET and 3 RET.

FR6. Notifications

Notifications for cancelling or creating meetings.

EXTERNAL INTERFACE FILES (Moodle's database):

DET:

1. Notification subject
2. Notification Body
3. Notification sender
4. Notification type
5. Notification Receiver

RET:

1. Notification

CONCLUSION:

Complexity is Low as we have 5 DET and 1 RET.

EXTERNAL INQUIRY:

DET:

1. Names of students

RET:

1. Students

CONCLUSION:

Complexity is Low as we have 1 DET and 1 RET.

FR7. Signing In

Signing in must be independent from Moodle, and it is required for doing anything related with teams.

EXTERNAL INQUIRY:

DET:

1. Student Name
2. Student Password
3. Student email

RET:

1. Student

CONCLUSION:

Complexity is Low as we have 3 DET and 1 RET.

FR8. Schedule access

The registered user may access its meeting schedule where he will find the completed or pending meetings. This meeting schedule shall be integrated into the Moodle calendar, apart from being accessible from the TEAMTEAM app.

EXTERNAL INQUIRY:

DET:

1. Meeting starts date.
2. Meeting end date
3. Meeting log
4. Meeting plots
5. Meeting name
6. Subject Name
7. Team Name
8. Names of students
9. Notification subject
10. Notification type

RET:

1. Teams per subject

2. Student
3. Subject
4. Notification
5. Meeting

CONCLUSION:

Complexity is High as we have 10 DET and 5 RET.

EXTERNAL INTERFACE FILES (Moodle's database):

DET:

1. Calendar owner
2. Calendar last modification
3. Calendar date

RET:

1. Calendar

CONCLUSION:

Complexity is Low as we have 3 DET and 1 RET.

FR9. Check the details of a past meeting.

From the meeting schedule the user may see the details of a meeting that has already taken place.

EXTERNAL INQUIRY:

DET:

1. Calendar owner
2. Calendar last modification
3. Calendar date
4. Meeting starts date.
5. Meeting end date
6. Meeting log
7. Meeting name
8. Notification subject
9. Notification type
10. Subject Name
11. Team Name

RET:

1. Meeting
2. Teams per subject
3. Student
4. Subject
5. Notification
6. Calendar

CONCLUSION:

Complexity is High as we have 11 DET and 6 RET.

FR10. Check the details of a pending meeting.
From the meeting schedule the user may see the details of a future meeting.

EXTERNAL INQUIRY:

DET:

1. Meeting starts date.
2. Meeting end date
3. Meeting log
4. Meeting name
5. Notification subject
6. Notification type
7. Subject Name
8. Team Name

RET:

1. Meeting
2. Teams per subject
3. Student
4. Subject
5. Notification

CONCLUSION:

Complexity is High as we have 8 DET and 5 RET.

FR11. Notification management.

Users should select the type of notifications they wish to receive, they can choose from email, push notifications to desktop or mobile or message from a bot inside the app.

EXTERNAL INQUIRY:

DET:

1. Names of students
2. Emails of students
3. Notification subject
4. Notification type
5. Notification Body
6. Notification sender
7. Notification Receiver

RET:

1. Meeting
2. Student
3. Notification

CONCLUSION:

Complexity is Medium as we have 7 DET and 3 RET.

FR12. Schedule a meeting.

Any member of a team can schedule a meeting (videoconference and joint working session) for practical assignments, any of its members can propose the call for a meeting for their team.

EXTERNAL INQUIRY:

DET:

1. Subject Name
2. Team Name
3. Maximum of students per group
4. Minimum of students per group
5. Names of students
6. Emails of students
7. Calendar owner
8. Calendar last modification
9. Calendar date

RET:

1. Calendar
2. Teams per subject
3. Student
4. Subject

CONCLUSION:

Complexity is Medium as we have 9 DET and 4 RET.

INTERNAL LOGICAL FILES:

DET:

1. Meeting date
2. Meeting Duration
3. Meeting log
4. Meeting plots
5. Meeting name
6. Meeting tasks
7. Meeting proposer

RET:

1. Meeting

CONCLUSION:

Complexity is Lowas we have 1 DET and 7 RET.

FR13. Cancelling a Meeting

Any participant of the meeting should cancel a scheduled meeting, providing a reason why it is being cancelled.

EXTERNAL INQUIRY:

DET:

1. Calendar owner
2. Calendar last modification
3. Calendar date
4. Meeting start date
5. Meeting end date
6. Meeting log
7. Meeting plots

8. Meeting name
9. Subject Name
10. Subject meetings
11. Student name

RET:

1. Student
2. Subject
3. Notification
4. Calendar
5. Meeting

CONCLUSION:

Complexity is Highs we have 11 DET and 5 RET.

FR14. Meeting Development

The user may associate meetings with practical assignments and may be able to create, close, open and leave a room.

EXTERNAL INQUIRY:

DET:

1. Calendar owner
2. Calendar date

RET:

1. Calendar

CONCLUSION:

Complexity is Low as we have 2 DET and 1 RET.

FR15. Room access.

Only allowed students should enter the room, and they can do so as many times as needed.

EXTERNAL INTERFACE FILES:

DET:

1. Allowed students
2. IsOpen

RET:

1. Room

CONCLUSION:

Complexity is Low as we have 2 DET and 1 RET.

FR16. Task-board

During a meeting student will be able to access a task-board with which they are going to distribute the work to do by each participant.

EXTERNAL INQUIRY:

DET:

1. Meeting date
2. Meeting duration
3. Meeting log
4. Meeting plots
5. Meeting name
6. Meeting tasks

RET:

1. Meeting

CONCLUSION:

Complexity is Low as we have 6 DET and 1 RET.

FR17. Collaborative tools

Registered users have access to chat, screen sharing, notebook, collaborative whiteboard, and a shared repository during the meetings.

EXTERNAL INQUIRY:

DET:

1. Meeting date
2. Meeting duration
3. Meeting log
4. Meeting plots
5. Meeting name
6. Meeting tasks
7. Allowed students

8. IsOpen

RET:

1. Meeting
2. Room

CONCLUSION:

Complexity is Medium as we have 8 DET and 2 RET.

FR18. Roster of attendees

The system will keep the total minutes that each user has spent in the class.

EXTERNAL INQUIRY:

DET:

1. Meeting date
2. Meeting duration
3. Meeting log
4. Meeting plots
5. Meeting name
6. Meeting tasks
7. Allowed students
8. IsOpen

RET:

1. Meeting
2. Room

CONCLUSION:

Complexity is Medium as we have 8 DET and 2 RET.

FR19. Recording of the classes

The system will record the classes for the users to access them afterwards as well as whether the meeting objectives have been met.

EXTERNAL INQUIRY:

DET:

1. Meeting date
2. Meeting duration
3. Meeting log
4. Meeting plots
5. Meeting name
6. Meeting tasks
7. Meeting proposer

RET:

1. Meeting

CONCLUSION:

Complexity is Low as we have 7 DET and 1 RET.

INTERNAL LOGICAL FILES (only the repository is stored in our system):

DET:

1. Repository name
2. Repository subject
3. Repository lastUpdate

RET:

1. Repository

CONCLUSION:

Complexity is Low as we have 3 DET and 1 RET.

FR20. Automatic delivery

Everything under the "Practical Assignment Submission" folder of the shared repository will be compressed and automatically delivered into Moodle by the system.

EXTERNAL INQUIRY:

DET:

1. Meeting date
2. Meeting Duration
3. Meeting log
4. Meeting plots

5. Meeting name
6. Meeting tasks
7. Meeting proposer
8. Repository name
9. Repository subject
10. Repository lastUpdate

RET:

1. Meeting
2. Repository

CONCLUSION:

Complexity is Medium as we have 10 DET and 2 RET.

EXTERNAL INTERFACE FILES (Moodle's database):

DET:

1. Assignment name
2. Assignment deadline

RET:

1. Assignment

CONCLUSION:

Complexity is Low as we have 2 DET and 1 RET.

FR21. Leaving and joining the room

Users may leave and join the room as long as it remains opened.

EXTERNAL INQUIRY:

DET:

1. Meeting date
2. Meeting Duration
3. Meeting log
4. Meeting plots
5. Meeting name
6. Meeting tasks
7. Meeting proposer

RET:

1. Meeting

CONCLUSION:

Complexity is Medium as we have 7 DET and 1 RET.

FR22. How to manage the information

The application loads (and dumps) the group, meeting, and calendar information from Moodle´s database so it is safe to assume that the device must be connected to the network.

EXTERNAL INQUIRY:

DET:

1. Meeting date
2. Meeting Duration
3. Meeting log
4. Meeting plots
5. Meeting name
6. Meeting tasks
7. Meeting proposer
8. Calendar last modification
9. Calendar date
10. Team Name
11. Maximum of students per group
12. Minimum of students per group
13. Team participants

RET:

1. User
2. Calendar
3. Team
4. Meeting

CONCLUSION:

Complexity is High as we have 13 DET and 4 RET.

FR23. Practical assignment and group report

The teacher shall be able to ask for a report with the statistics by practical assignment and group.

EXTERNAL INQUIRY:

DET:

1. Meeting date
2. Meeting Duration
3. Meeting log
4. Meeting plots
5. Meeting name
6. Meeting tasks
7. Meeting proposer
8. Team Name
9. Maximum of students per group
10. Minimum of students per group
11. Team participants
12. Subject Name
13. Subject meetings
14. Subjects Assignment
15. Assignment name
16. Assignment deadline

RET:

1. Team
2. Assignment
3. Meeting
4. Subject

CONCLUSION:

Complexity is High as we have 16 DET and 4 RET.

FR24. Practical assignment and group report for the whole class

The system will keep track of the meetings and the work of the team at each practical assignment, so the software system will allow to generate and visualize different statistics of the meetings.

EXTERNAL INQUIRY:

DET:

1. Meeting date
2. Meeting Duration
3. Meeting log
4. Meeting plots
5. Meeting name
6. Meeting tasks
7. Meeting proposer
8. Team Name
9. Maximum of students per group
10. Minimum of students per group
11. Team participants
12. Subject Name
13. Subject meetings
14. Subjects Assignment
15. Assignment name
16. Assignment deadline

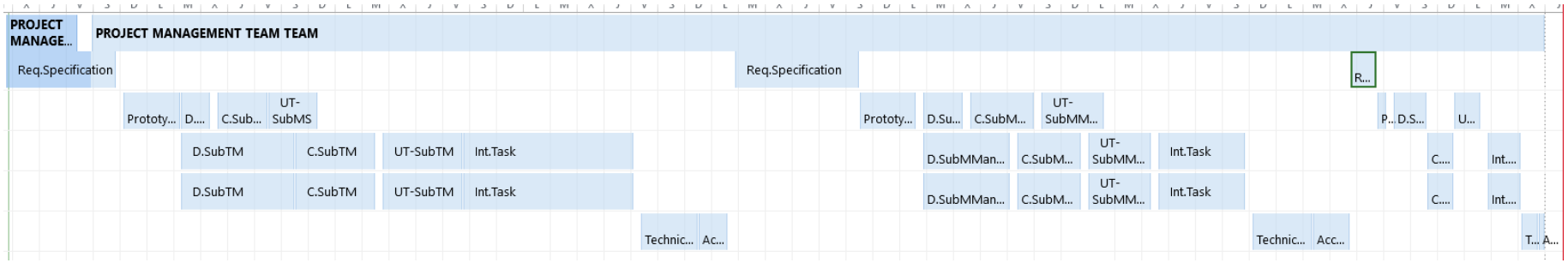
RET:

1. Team
2. Assignment
3. Meetings
4. Subject

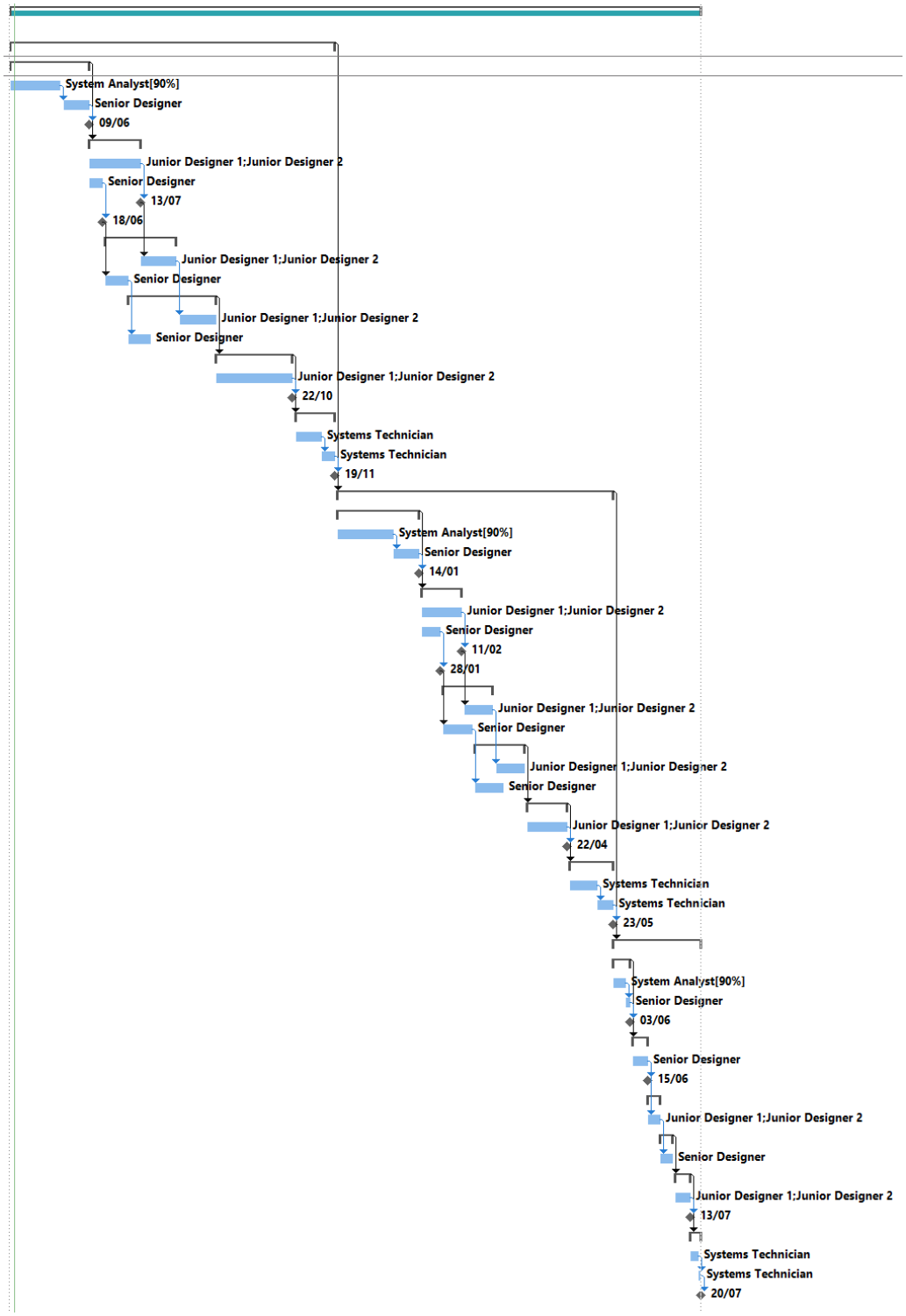
CONCLUSION:

Complexity is High as we have 16 DET and 4 RET.

ANNEX C. RESOURCES AND ACTIVITIES ASSIGNMENT



ANNEX D. GANTT DIAGRAM



ANNEX E. MEETING MINUTES

MEETING ANNOUNCEMENT

From: Rodrigo
To: The rest of the team

DATE AND TIME: 06/04/2021 at 16:00
PLACE: Microsoft Teams
DURATION: 2h

PURPOSE: Review the first sections and put together all the requirements

1. **AGENDA: Put together requirements and review first section**
2. **DECISION FOLLOW-UP:**
3. **DOCUMENTATION: Practical Assignment 1**

MEETING MINUTES

DATE AND TIME: 06/04/2021 at 16:30
PARTICIPANTS: ALL

1. **KEY POINTS DISCUSSED**

Put together the requirements with input and output and organize.

2. **DECISIONS MADE**

ACTIONS	RESPONSIBLE PERSON	DEADLINE
Requirements Subsystem 3 and 4	Angel	24/02/2021
Requirements Subsystem 5	Jorge	24/02/2021
Requirements Subsystem 1	Rodrigo	24/02/2021
Requirements Subsystem 2	Pablo	24/02/2021

MEETING ANNOUNCEMENT

From: Rodrigo
To: The rest of the team

DATE AND TIME: 13/04/2021 at 16:00
PLACE: Microsoft Teams
DURATION: 2h

PURPOSE: Unadjusted Function Points

4. **AGENDA: Put together Non-functional requirements and start with function points**
5. **DECISION FOLLOW-UP: Requirements are done, and first sections are done.**
6. **DOCUMENTATION: Practical Assignment 1**

MEETING MINUTES

DATE AND TIME: 13/04/2021 at 16:00

PARTICIPANTS: ALL

2. ## KEY POINTS DISCUSSED

Non-functional requirements revision and organization of function points and unadjusted function points.

3. ## DECISIONS MADE

ACTIONS	RESPONSIBLE PERSON	DEADLINE
Function Points	Angel	24/02/2021
Function Points	Jorge	24/02/2021
Function Points	Rodrigo	24/02/2021

Non-Functional Requirement revision.	Pablo	24/02/2021

MEETING ANNOUNCEMENT

From: Rodrigo
To: The rest of the team

DATE AND TIME: 21/04/2021 at 16:00
PLACE: Microsoft Teams
DURATION: 2h

PURPOSE: Finish sections 4 and 5 and review MS Project

7. **AGENDA: Write sections 4 and 5 and finish MS Project**
8. **DECISION FOLLOW-UP:**
9. **DOCUMENTATION: Practical Assignment 1**

MEETING MINUTES

DATE AND TIME: 06/04/2021 at 16:30
PARTICIPANTS: ALL

3. **KEY POINTS DISCUSSED**

How do we write sections 4 and 5 and where do we get the information, and how to fix MS Project errors Rodrigo was having?

4. **DECISIONS MADE**

ACTIONS	RESPONSIBLE PERSON	DEADLINE
Section 5	Angel	24/02/2021
Section 5	Jorge	24/02/2021
MS Project and Section 4	Rodrigo	24/02/2021
Section 5	Pablo	24/02/2021