

TECHNICAL REPORT



INITIAL SOFTWARE REQUIREMENTS SPECIFICATION FOR A SOFTWARE PROJECT

Version 1.0

Jorge Blanco Rey
Ángel Casanova Bienzobas
Rodrigo Juez Hernández
Pablo Soëtard García

Group: 2391
Team: 2

10/03/2021

CHANGE LOG TABLE

Version	Date	Contents or main modifications	Purpose
v0.4	15/02/2021	Functional and non-functional requirements.	Get a draft of the document.
v0.6	27/02/2021	Correction of the main errors detected in the meeting of the 19/02/2021.	Spare the system into subsystems.
v0.8	01/03/2021	Get the majority of the document.	Finish the document.
v1.0	07/03/2021	Correction of little mistakes.	Future software test.

SUMMARY

The purpose of this document is to give a detailed overview of our project proposal for the tenth edition of the competition for IT Innovation Projects organized by the Autonomous University of Madrid (UAM).

In this year's edition, the motivation of the IT Innovation Project is to improve the current way to create online teams for performing practical assignments, due to the exceptional situation of COVID-19 pandemic. The main objective of this competition, named My E-Assignments, within the academic context of EPS-UAM Moodle, is to develop an open-source software project to specify the requirements that a software with these features targeted to students enrolled in different subjects offered by UAM-EPS.

To develop this requested solution, we will first perform a competitive analysis in order to have a look at the current available tools for students.

After getting inspired by the already existing tools we will gather the best features among them, also adding some interesting extra ones and integrate them all under the same roof, providing to the EPS students an all-in-one tool to satisfy all their work from home student needs.

INDEX

1.	Introduction	1
1.1.	Purpose	1
1.2.	Document conventions	1
1.3.	Intended audience and reading suggestions	1
1.4.	Project scope	2
1.6.	Structure	2
2.	Project definition	4
2.1.	Goals	4
2.2.	Functionality	5
2.2.1.	Summary of the Functionalities	5
2.2.2.	Infrastructure required	5
2.3.	Initial Catalogue Requirements	5
2.3.1.	Functional Requirements	5
2.3.2.	Non-Functional Requirements	8
3.	System Interaction. Conceptual and Visual Design	10
3.1.	Application login	10
3.2.	Landing page	10
3.3.	Team page	11
3.4.	Team creation	11
3.5.	Meeting creation	12
3.6.	Repository	12
3.7.	Calendar page	13
3.8.	Meeting page	14
3.9.	Notifications	15
3.10.	Configuration	16
4.	Conclusions	17
4.1.	Global summary	17
4.2.	Discussion of the proposed system	17
4.3.	Discussion of the contributions of the proposed system	17
4.4.	Benefits to be obtained	18
4.5.	Limitations of the proposal	18
4.6.	Future work	18

Glossary.....	19
References	20
Appendices.....	21
Appendix A. Competitive Analysis	21
Appendix B. Brainstorming	25

1. Introduction

The main concern of this document is to gather the SRS for the open-source software project for online EPS student's teamwork.

This project has been theoretically proven to work and that is why we propose it as a valid solution for this tenth edition of the IT Innovation Project contest.

1.1. Purpose

The purpose of this document is to provide a detailed view of the functionalities based on the given requirements, the features and competitive advantage of our proposal project for the IT Innovation Project upon all the other available software in the market.

In this document all the key subsystems and features that compose the application will be examined and explained in detail, as well as a preliminary GUI for the software will be given.

On the other hand, the motivation of building such a system is to improve the current way to create online teams for performing practical assignments, due to the exceptional situation of COVID-19 pandemic.

1.2. Document conventions

This document is a technical document, hence it may contain some acronyms or technical slang that readers may be unfamiliar with. For that reason, we provide a Glossary at the end of this document where most of those terms will be explained.

1.3. Intended audience and reading suggestions

This document is intended for the people in charge of the IT Service Department and for the IT Innovation Project jury, in order to give them all a detailed overview of our proposal for the contest.

Other readers that may not have the expertise to read such a technical document, may feel unfamiliar with the structure and contents of it, that is why a Glossary and a Reference section is included on it.

Readers who are only interested in learning on what is the project about may read sections 1 (Introduction) and 2 (Project definition). Otherwise, for a full overview, they may read the whole document.

1.4. Project 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.

There are four subsystems that are going to be implemented in this project:

The Team Management Subsystem which manages the creation of teams for practical assignments and attach to its members, make notifications, select preferences, and view information from the teams.

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

The Meeting Management Subsystem, which is in charge of creating the meeting, send reminders, manage the rooms that make up the meeting infrastructure, consult information on meeting development and cancel the meeting.

The Meeting-making subsystem that implements the automatic delivery of practical assignments to Moodle and provides an in-call toolbox for students, that will include tools like a whiteboard, live chat, screen sharing, repository, notebook or a videocall system.

Note that despite being an opensource project the system may use proprietary software from external providers, like the one of the videocall 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.

1.5. Methodology and background

We received the user's requirements via a document made available by the IT Innovation Project organizers, the first methodology used was a Competitive Analysis which helped us to decide which features of similar platforms were useful and which were not. After having a clear view of all the tools available we did a brainstorming session, where each of us proposed new features, and after that we exchanged ideas building on top of the original ones.

We finally divided the requirements in subsystems and non-functional and finally we did an elevator pitch under 8 minutes to put across the main points of our system.

1.6. Structure

This document is composed by four main sections a Glossary, a Reference table and two appendices. The first section is the Introduction where a general overview of the document is given, in the second section the project is defined more in depth, in section three a preliminary GUI with its mock-ups is described and in the last section, section four, a conclusion of the document is given.

In the Glossary and Reference table readers can find acronyms' meanings and external references links, respectively.

At the end of this document three appendices can be found, Appendix A containing a Competitive Analysis, Appendix B containing a Brainstorm performed by our team and Appendix C with all the team meetings.

2. Project definition

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 app or from environment to environment, that is why we introduce to you TEAM TEAM, 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].

So, in this section, we are going to sum up the main goals of our application and list all the functionalities.

2.1. Goals

These following points are our main goals.

Fast access interface

We have designed an interface based in symbolic buttons in order to save time to the user, that is, every tool that we provided is located at sight to the final user, there is not any functionality hidden to him.

All-in-one application

We reckon a work environment where the user could have all the tools that he needed in the same layout, in this way, the user will save time too, because he does not have to exit from the application to search for another functionality.

Keep the Moodle Database

Anyone likes to register twice, so we are going to use the Moodle database to log in and also to store (or update) the information of the students, such as the telephone number, etc.

Enhance the practices experiences

This application is made by students for students, that is, this application has been designed for making the user's life easier. We concentrate all the teaching tools in the TEAM TEAM application to improve the user experience with his practical assignments.

2.2. Functionality

2.2.1. Summary of the Functionalities

In this section we are going to briefly describe the main functionalities that the TEAM TEAM application develops.

[1] Team Management Subsystem

- The registered users can create new teams.
- The users may access a detailed view of the calendar and a detailed view.
- The user cannot leave a team once he is in.

[2] Meeting Scheduling Subsystem

- From the meeting schedule the user may see the details of each meeting.
- The schedule of the meeting shall be integrated into the Moodle calendar.

[3] Meeting Management Subsystem

- The user may be able to create, close, open and leave a room.
- The system will check that all the team members can attend to that meeting.
- The user must be able to enter in the meetings through the calendar.
- Notifications may be configured for meetings in advanced.

[4] Meeting-making subsystems

- Registered users have access to the toolbox during the call.
- The submission folder of the shared repository will be compressed and automatically delivered into Moodle by the system.

2.2.2. Infrastructure required

Our application does not require a specific infrastructure apart from a web server, the performance needed by the server will be proportional to the number of users.

2.3. Initial Catalogue Requirements

2.3.1. Functional Requirements

[1] Team Management Subsystem (create the team for practical assignments and attach to its members, make notifications, select preferences, and view information from the teams)

[1.1] Creation of Teams

- FR1. The user may create a team.
- FR2. Any creation of a team must comply with the requirements per group: Subject name, Team name, Maximum or minimum number of students per group or team, Name and email per student, Membership of the students to the same group.
- FR3. The application shall automatically fill out the user's schedule.

- FR4. When creating a team, the user may preselect users, but they shall not be directly added, the invited user receives an invitation which they may accept or decline.

[1.2] Team Information

- FR5. The user may access a detailed view of the calendar.
- FR6. The user may check the teams which they belong, not only in Moodle (where the information is stored) but in the application itself.
- FR7. The teams shall be related with practical assignments by subject.

[1.3] Managing members

- FR8. Teams may broadcast an invite to all users that are not included in any team for that subject.
- FR9. The user may join a team at any time if it is not full.
- FR10. The user may ask to join a team instead of being invited.
- FR11. The user shall not be able to leave a team once he joins, as well as not be able to dissolve it until all practical assignments are done.

[1.4] Notifications

- FR12. When a meeting is created, the app automatically sends a notification to its participants without them needing to configure anything.
- FR13. When cancelling a meeting a notification shall be automatically sent to the members.

[2] Meeting Scheduling Subsystem (see pending or completed teamwork meetings)

- FR14. The schedule of the meeting shall be integrated into the Moodle calendar.
- FR15. The calendar must be updated every time a meeting is created or cancelled.
- FR16. The registered user may access its meeting schedule where he will find the completed or pending meetings.
- FR17. The registered user may see all its teams that he has (or has been) for practical assignments.
- FR18. From the meeting schedule the student may see the following details of each meeting: meeting name, team and subject, date and time, duration and how the user shall be notified in case he has the notifications activated.

Moreover, if it is a pending meeting, it should contain also: a link to the room where the meeting shall be held, a list with objectives of the meeting and/or the topics to be discussed and a radio button indicating whether click to notify them when the meeting starts.

Conversely if it is a completed meeting, in addition to the base listed above, it should contain: the attendees name, punctuality of each attendee, time remaining in the meeting and meeting report (goals achieved and objectives to be met for the next meeting).

[3] Meeting Management Subsystem.

[3.1] Meeting Creation

- FR19. The user may create videoconferences and joint working sessions.
- FR20. The user may be able to create, close, open and leave a room.
- FR21. The user may schedule meetings for their practical assignments.
- FR22. When creating a meeting it is required: Name, Day, Time, Duration, Objectives, Topics to discuss.
- FR23. The system will check that all the team members can attend that meeting.
- FR24. The system creates an entrance in the user's calendar for the meeting.
- FR25. The system sends a notification to each team member.

[3.2] Meeting Development

- FR26. The user may associate meetings with practical assignments.
- FR27. When a meeting ends the system asks the members to check the objectives that they have fulfilled during the call.
- FR28. While the room is opened, members may join the room by clicking on its link on their calendar.
- FR29. The user may set the application to notify them whenever a meeting starts.
- FR30. The user must be able to enter in the meetings through the calendar, the calendar must have a links in each tile, so by clicking in one tile the system shall redirect the student to the meeting.
- FR31. Notifications may be configured for meetings in advanced.
- FR32. The system will open the room for the meeting 5 minutes before it starts, then the system shall close the room 5 minutes after the established time if all the members have left the room.

[4] Meeting-making subsystems (available tools provided by both the room and the system itself to hold the meeting and work on the internship team, deliver practical assignments)

[4.1] Meeting Tools

- FR33. Registered users have access to chat, screen sharing, notebook, collaborative whiteboard, and a shared repository during the meetings.
- FR34. Registered users shall get information for queries and statistics.
- FR35. 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.

FR36. The system will record the classes for the users to access them afterwards.

FR37. Everything under the “Practical Assignment Submission” folder of the shared repository will be compressed and automatically delivered into Moodle by the system.

2.3.2. Non-Functional Requirements

Security:

NFR1. To do anything related with teams the user must be signed in the application.

NFR2. Independent sign-in from Moodle.

NFR3. System must check that only allowed student may enter in the room.

NFR4. System allows the students to enter in the room as many times as needed.

Usability:

NFR5. Application runs on a different window from Moodle.

NFR6. The meeting schedule is integrated into the Moodle calendar.

Operational:

NFR7. The application loads group, meeting, and calendar information from Moodle, it does not store any of it.

NFR8. The application dumps information into Moodle, new groups, the calendar.

Availability

NFR9. The app retrieves its information from Moodle so it is safe to assume that the device must be connected to the network.

Behaviour

NFR10. Notifications may be in the form of an email, push notifications to desktop or mobile or messages from a bot inside the app.

NFR11. Meetings are accessed through a link which is automatically generated when a meeting is scheduled.

NFR12. The system checks whether there are overlaps in the calendar of each of the meeting participants. The system registers the meeting in each member’s calendar and notifies them and adds the link to their calendar too.

NFR13. Five minutes before starting each meeting the system automatically opens the room.

NFR14. Users may leave and join the room as long as it remains opened.

NFR15. The system closes the room five minutes after the finish scheduled time.

NFR16. Clicking on the meeting link in calendar redirects the user to the meeting.

NFR17. The system records the announced meetings, the ones done, cancelled and whether the meeting objectives have been met.

NFR18. The system shall notify the meeting participants in the event of the meeting being cancelled and why it was cancelled.

NFR19. The system must record the classes, the system stores the recording in the repository system when the call ends.

NFR20. Notifications may be in the form of an email, push notifications to desktop or mobile or messages from a bot inside the app.

Reliability

NFR21. Connection shall be automatically restored when internet is down for the student.

Resources

NFR22. The application must have memory to store, whiteboards, notebooks, and repositories, generated during meetings, these is not dumped to Moodle.

3. System Interaction. Conceptual and Visual Design

Now we will show the visual design of our application. We will explain by different mock-ups how the application works, that is how the user can move from one window to another and the different options he has on each one.

3.1. Application login

When we enter the application, we will need to do, is to log in with his Moodle credentials, as we can see in Figure 1.

As we can observe, we have the “Forgot your password?” option in case that is necessary. When it is clicked, it redirects us to the UAM page that manages the change of password process.

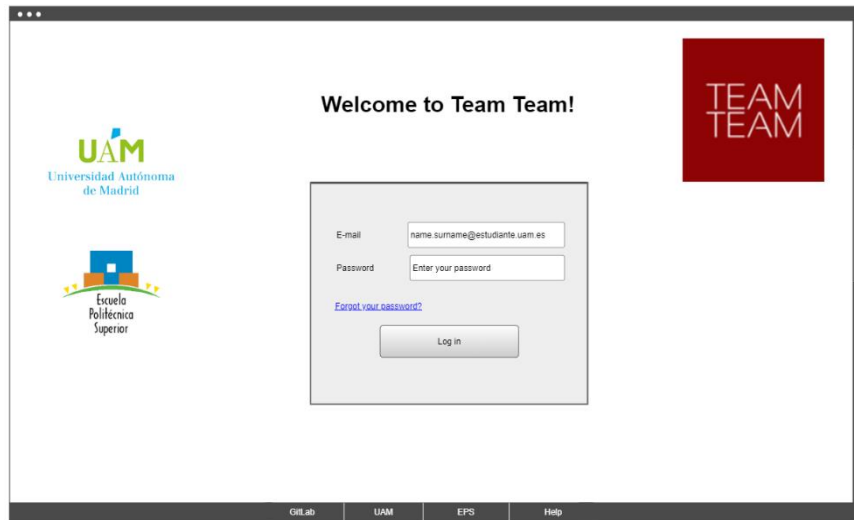


Figure 1

We must remark as well that this login page runs on a completely different window from the Moodle one.

Also, on the bottom of the screen we can see different options that will come with us during all the application. We have the “GitLab” option that will take us to the EPS GitLab; the “UAM” and “EPS” options that will take us to the university and school websites, respectively; and finally, a “Help” option that will show us a small manual of how to use the application.

3.2. Landing page

After logging in into the application, we will arrive at the landing page (Figure 2).

Here, we can access the unified calendar for all our meetings, the teams we are in, where we will also have the option of creating new ones as we are going to see later.

In this page, we also have access to the subjects, that will redirect us to the Moodle

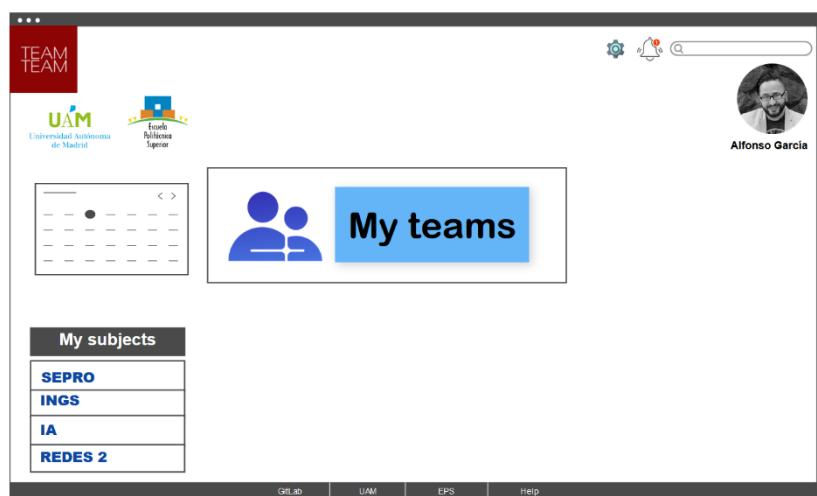


Figure 2



course, the notifications, application configuration and a search bar, where we can look for teams or meetings, for example.

Finally, it is important to mention that whenever in the application we click on the TEAM TEAM logo that is on the upper-left corner, we will arrive to this page.

3.3. Team page

As we have said before, if we click in “My teams” from the landing page (Figure 2), we will go to this page, Figure 3.

From this view, we can select a team we are in and it will tell us the team subject and the assignment in which we are.

We can search for other groups in the team search bar and if we click join at any time, a petition will be sent to the team participants.

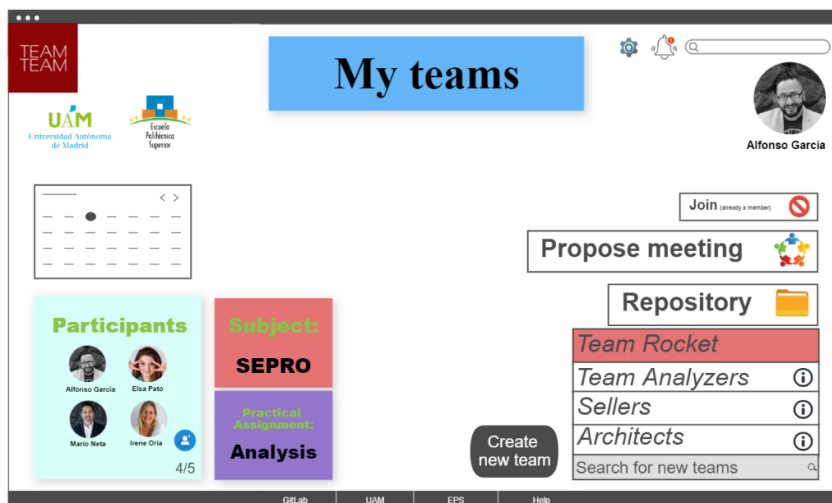


Figure 3

Also, we access to the team repository, where we can store assignment files, and the ones placed on “Practical Assignment Submission” folder, will be automatically delivered when the deadline arrives.

Finally, we can propose a meeting, which we will see in a minute, by clicking on “Propose meeting” or create a new team by clicking on the “Create new team” button, that will redirect us to the page we are going to see now.

3.4. Team creation

When we click on the “Create new team” button of the team page (Figure 3), we will arrive to this window (Figure 4).

Here, we can create new teams by fulfilling a short form in the page. We need to write the name of the team, the subject for which the team is going to be for and the colleagues we want to invite, we can search them with the search bar that is under “Invite

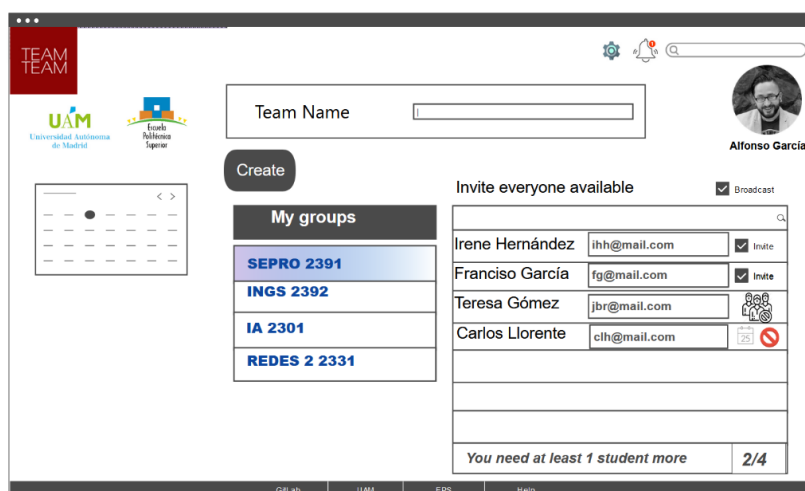


Figure 4



everyone available". Once we have chosen some, we can quickly invite all of them that are available with the "Broadcast", that is on the right of "Invite everyone available", we have also the option of selecting which ones we prefer.

As we have already mentioned, the application automatically blocks students with incompatible calendars (like Teresa Gómez) and groups (like Carlos Llorente).

Finally, we can observe on the bottom of the page with "You need at least 1 student more", that teams respect constraints set by the teacher such as minimum or maximum number of students.

3.5. Meeting creation

When we click on the "Propose meeting" button of the team page (Figure 3), we will arrive to this window (Figure 5).

Here, we can create meetings and joint working sessions for the teams we belong to.

We will have to provide the following information: Name of the meeting, duration, date, objectives and topics, and the associated practical assignment to which the meeting was created for.

Figure 5

When we create it, all participants will be notified (except the creator, obviously) and a link to that meeting will be added to their personal calendar.

3.6. Repository

When we click on the "Repository" button of the team page (Figure 3), we will arrive to this window (Figure 6), the repository.

Here each member of the team can place their documents, that will be saved on a cloud.

It is also important to remark the "Practical Assignment Submission" folder, that as we have said

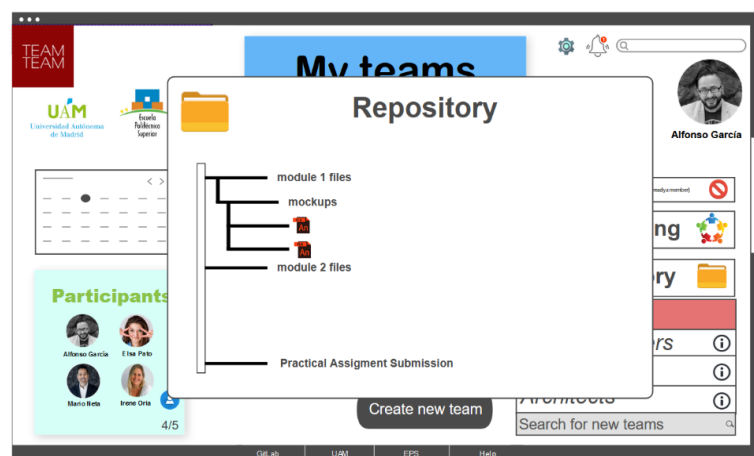


Figure 6

before, all the files that are in it will be automatically delivered when the deadline of the assignment arrives.

3.7. Calendar page

When we click on the calendar that is present in almost all pages, we will arrive to the window we can see on Figure 7.

Here, we will have our calendar where we could see all the past and future meetings as well as their information. On the left, there is a dropdown menu, where we can change the view of the calendar, choosing between day, month, or year.

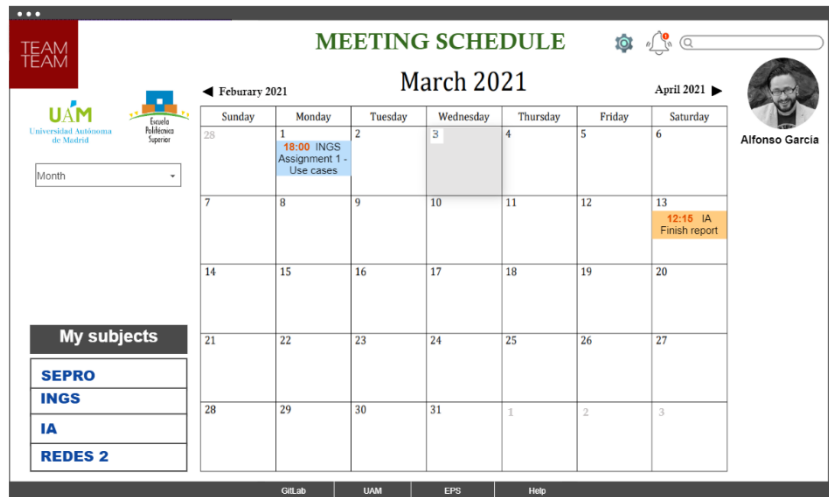


Figure 7

If we click on a pending meeting, we will see page similar to Figure 8.

There, we could see the main information of the meeting, like the title, which in this case is "Finish report", the subject, team, date, duration, the person who proposed it, and how it is going to be notified.

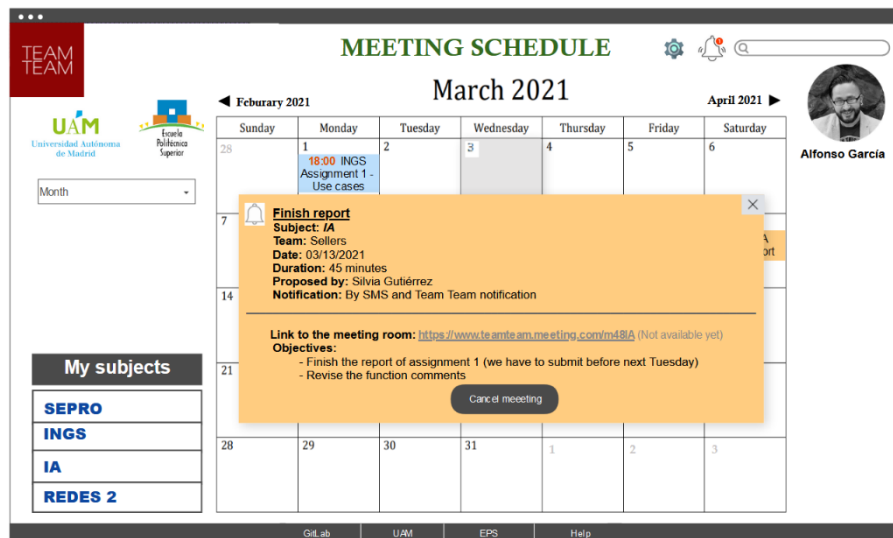


Figure 8

We could see a link to the meeting, which in case it is already running, we will be able to click it and enter in the room. Also, we could see the objectives of the meeting and the button with which we have the option to cancel it and will ask us to give a reason for the cancellation.

In terms of notifications, we have the possibility of disabling them for each particular meeting, by clicking on the bell in case it is activated (if it is not crossed). In case of Figure 8, the notification is activated.

If in Figure 7 we click on a past meeting, we will see page similar to Figure 9.

There we could see the main information of the meeting, like the title, which in this case is “Assignment 1 – Use cases”, the subject, team, date, duration, the person who proposed it, and how it was notified.

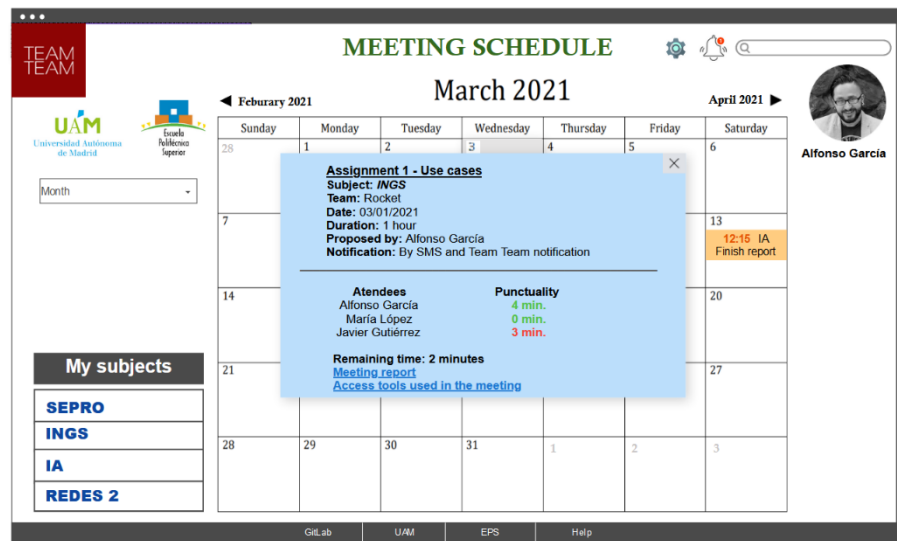


Figure 9

Also, we could check the punctuality of each participant, the ones marked in green means that they have entered that minutes before the start time and the ones in red that they have entered that minutes after the start time. In addition, we could check the tools used, where we can find the meeting recording as well, and of course, a meeting report with the accomplished objectives and statistics, like the minutes spoken by each participant or time-sharing screen.

3.8. Meeting page

When we click on the link (when it is available) placed at “Link to the meeting room” of Figure 8, we will enter the meeting, where we would have the tools, we can see on Figure 10.

These tools are a digital whiteboard, a notebook and a shared repository, each tool has a direct access clicking on each of them. The share screen and live chat tools are implemented by the meeting platform.

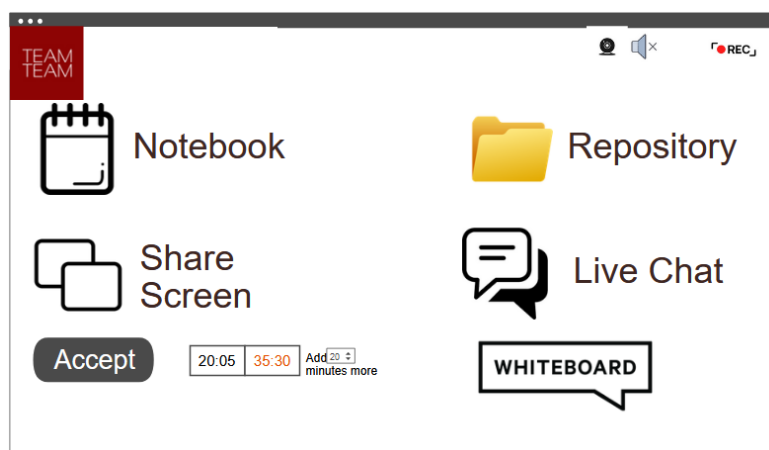


Figure 10

Also, as we can see on Figure 10, we can manage when the meeting ends. As the duration has been set before the meeting start, we can see on black colour the time it has been running (in this case, 20:05 min.) and on red, the remaining time (in this case, 35:30 min.). And since the meeting will be closed automatically by the application five minutes after the preestablished time, if we need more time, we can add more minutes on the box where it says, “Add X minutes more” and click the accept button.

When the meeting finishes a popup similar to the one, we can see on Figure 11 will appear.

Here, we will be able to select which objectives from the ones we wrote when we created the meeting (Figure 5), we have fulfilled, so the meeting report can be generated. We just need to click on the white box of the objective we have completed and when we are done, click on the “Send objectives” button, in the lower-right corner of the popup.

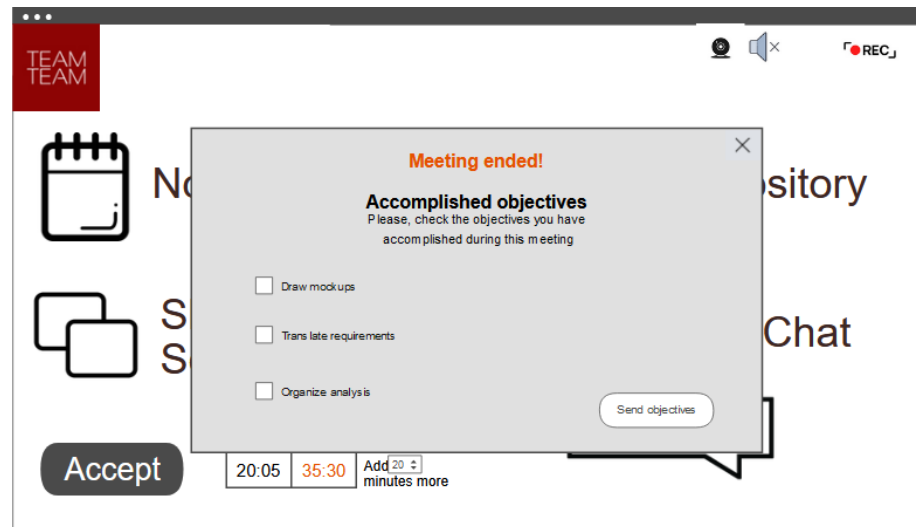


Figure 11

3.9. Notifications

In most of the previous pages, like Figure 2, we can see that the bell on the left of the search bar has a 1, which means that we have one new notification.

If we click on the bell, we will see something similar to Figure 12.

When we click on the bell, we would see the new notification messages, as well as past ones. In this case, is the first ever notification for Alfonso in the application, but notifications that we have not already read will appear with the red dot as we can see, and the past ones without it.

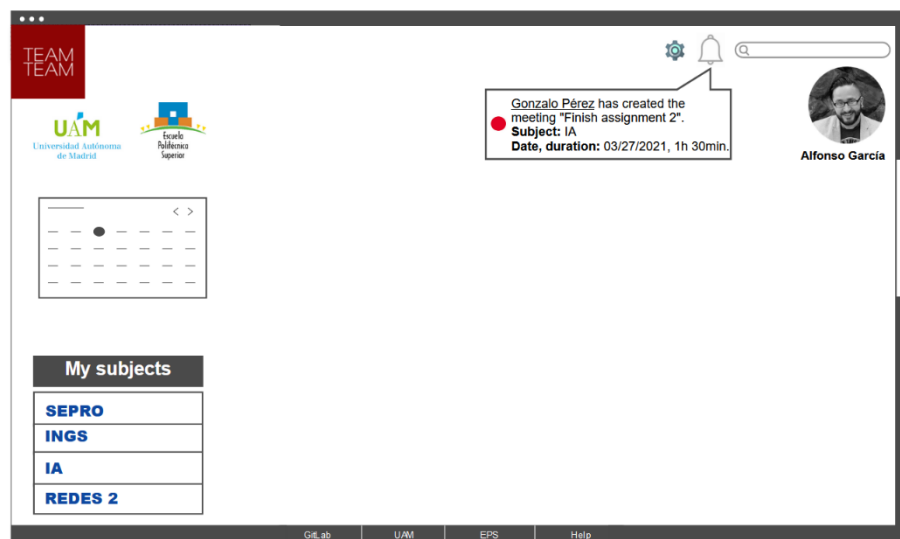


Figure 12

3.10. Configuration

If we click on the configuration button that is on the left of the notification's bell, we will have several options for configuration, and one of them will be notifications.

When we click on it, we will arrive to the page we can see on Figure 13.

Here, we have the "Enable all" button to activate all the options, and the also the possibility of enabling just the ones we prefer.

The options we have, is that the application notifies us when a meeting is created or cancelled and we can select the minutes before the meeting start, that we want to get notified. Also, we can select the way in which the notifications are going to be sent, choosing between email, SMS, or Team Team notification, this last one is the notifications that appear on the bell, as we have seen on Figure 12. In case we select SMS, we will need to enter a mobile number.

Finally, if we want to register the changes, we have to click on the "Save configuration" button on the lower-right corner, we can also go backward to the configuration menu by clicking on "Configuration" on the upper-left corner or just on the configuration button on the left of the bell.

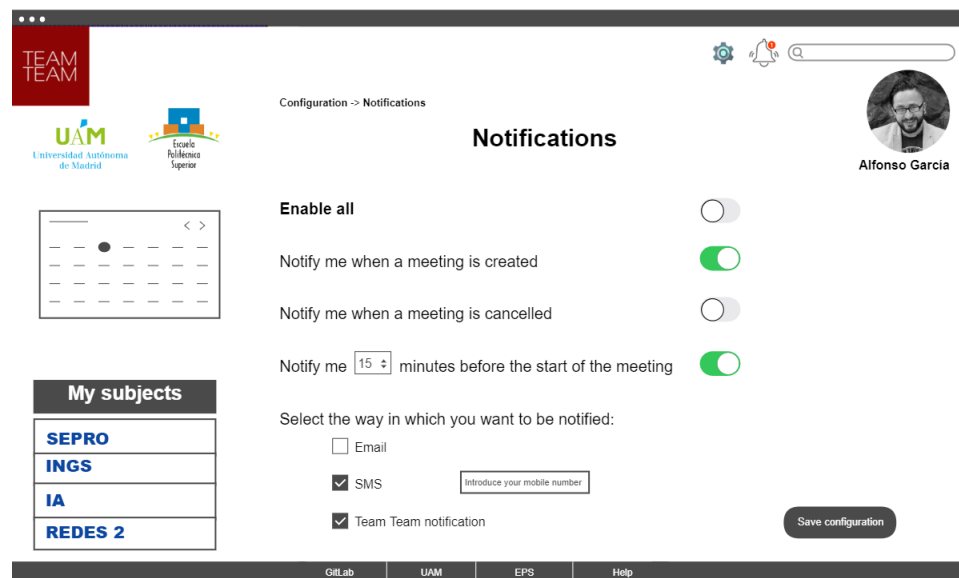


Figure 13

4. Conclusions

4.1. Global summary

The purpose of this Software Requirements Specifications document was to provide a detailed description of all the functionalities, features and requirements that are part of our proposal project for the IT Innovation Project upon all the other available software's in the market.

We designed an application that met all the requirements set by UAM and added some features that we observed in the competition, our goal was to alleviate the pressure students have right now, by simplifying and streamlining their workflow.

A brief summary of the features in our application includes the synchronization between all the subsystems, which would be: **Team Management, Meeting Creation, Meeting Scheduling, Meeting Making**, which you can see more in detail in section 2.3.

4.2. Discussion of the proposed system

Our system delivers on all the requirements, we divided them in subsystems, easy to differentiate, and linked them in a way it was seamless.

Thanks to the approach of designing a base app and then re-reading the requirements to try and understand how to best build each one of them into the experience we ensured that there will not be any requirement missing and of course that the experience feels unified.

Our system is guaranteed to work because of our experience with other similar platforms as students, this has given us a view to what kind of workflows are useful and which are not. Regarding reliability, our system is web-based, that is, the interface and navigation will be very familiar to most users and will be quite fail proof, while the back end will not be very heavy as most of the storage is done by Moodle, we only need to implement the logic.

4.3. Discussion of the contributions of the proposed system

Our contributions revolve around taking what is being already done by many teams but simplifying it and leaving only what students need, for example the repository is always easily accessible, unlike the competition which hides the tab where you store the file, another example would be the calendar, always accessible and with a quick glance you have most of the information you need in your daily routine, when a meeting ends a popup in the centre of the screen appears so you check all the objectives, instead of being hidden away where you need to click a button for it to appear.

All these examples are given to put across our main point, the user must not worry about anything regarding his/her organization, the application will handle everything.

4.4. Benefits to be obtained

Currently all these tools are in separate applications and places, the benefits are clear, a much-streamlined workflow than the competition, drastically reducing the time spent switching between apps and organizing with colleagues.

4.5. Limitations of the proposal

A limitation of the application is that its separate from Moodle, even though all the information is synchronized and stored there, it is a separate window, it would be better if everything were integrated under the same roof, following the objective of this application.

Another limitation is that the application is currently mainly tailored toward EPS students, so it would be great if it were a more general app for all kind of students.

4.6. Future work

Maintenance is key, we must ensure that the application lasts in time and that user information is safely stored in Moodle.

More features can be added to the application in the future such as:

- Personal repositories, so that students can store files that are not related with any practical assignment.
- Dark mode, as this is a web-based app the UI can be easily changed and customized.
- A wall where students can post information related to that team, like remainders or useful links.

Glossary

API – Application Programming Interface

EPS – Escuela Politécnica Superior

GUI - Graphic User Interface

Repository - a digital online based environment where archives may be stored.

SRS – Software Requirements Specification


UAM – Universidad Autónoma de Madrid



References



- [1] <https://gravityflow.io/articles/lost-time-seriously-affects-profitability/>
- [2] <https://www.microsoft.com/es-es/microsoft-teams/group-chat-software>
- [3] <https://discord.com/>
- [4] <https://ideaflip.com/>
- [5] <https://edu.google.com/intl/es-419/>



Appendices

Appendix A. Competitive Analysis

System, URL	Positive Aspects	Negative Aspects	Ideas for our Project
<p>Microsoft Teams https://teams.microsoft.com/</p> 	<ul style="list-style-type: none"> Meetings can be recorded Participants can “raise their hand” when they want to contribute Files used in the channel can be accessed and edited synchronously by all the participants. Integrated real-time content creation with Office apps like Word, Excel, PowerPoint, and OneNote. It has a chat for each team that it is also available during the meetings During a meeting you can search other participants and invite them Participants can share individual and total screens in good quality Polls can be done in the teams The organizer can mute other participants When someone talks it can be seen who is and it goes to the forefront A participant can be attached Participants can activate their cameras It has an integrated whiteboard that can be also edited in real time by other participants The organizer can download the list of participants 	<ul style="list-style-type: none"> It is possible to “become deaf” during a meeting, but it does not have easy access It is not very accessible to all OS It is very slow For the free version meetings can only be up to 60 minutes. Only one time up to 24 hours. 	<ul style="list-style-type: none"> Record meetings Participants can “raise their hand” Files used in the channel can be accessed and edited synchronously by all the participants Chat also during meetings Easy way to invite participants to the meeting Integrated real-time content creation Share individual and total screens in good quality Possibility of doing polls Mute other participants See who talks Attach participant Meetings of voice and image Integrated whiteboard The organizer is able to download the list of participants Make that users can “become deaf” easily

<p>Zoom https://zoom.us/</p> 	<ul style="list-style-type: none"> ▪ Meetings can be recorded ▪ It has a chat during the meeting ▪ Participants can share individual and total screens ▪ Polls can be done during the meetings ▪ If someone talks it is marked with a yellow border ▪ A participant can be attached ▪ It is possible to use voice and image ▪ A link to the meeting can be created and the organizer can control who enters the meeting ▪ It is easy to invite participants to the meeting ▪ Participants can share reactions during the meetings ▪ You can ask for unmute to other participants 	<ul style="list-style-type: none"> ▪ Meetings are limited to 30 min for the free version ▪ Participants can be unmuted without asking for "permission" 	<ul style="list-style-type: none"> ▪ Option to record the meeting ▪ Have always a chat, not only during meetings ▪ Be able to share individual and total screens ▪ Be able to do polls at any time, not only during meetings ▪ Mark who is talking and make it appear on the principal page ▪ Be able to use voice and image ▪ Be able to create a link to the meeting ▪ Be able to invite participants in an easy way ▪ The organizer can control the people who access the meeting ▪ Do not have the possibility of unmuting other participants, or in that case, ask for their permission
<p>Notion https://www.notion.so/</p> 	<ul style="list-style-type: none"> ▪ Is user-friendly in the desktop app. ▪ Notion is an "all-in-one" software, everything is placed under one roof which saves a lot of time. ▪ It works with a lot of common apps for companies. 	<ul style="list-style-type: none"> ▪ It takes so long to set up the application ▪ The phone app is not intuitive ▪ Notion notification is not reliable, the notifications do not go through. 	<ul style="list-style-type: none"> ▪ The free version is limited but it is enough for normal use. ▪ Notion empowers its users by allowing them to take notes and manage their to-do lists. It allows them to track their bills, dues, and details in a database found in their personal finance page.

<p>Asana https://asana.com/es</p> 	<ul style="list-style-type: none"> ▪ Modern UI ▪ Many types of workflows ▪ Suitable for everyone, from small businesses to big engineering projects. ▪ It is collaborative, several people can modify the schemes and agendas in real time. ▪ You can set individual goals apart from the projects in common. 	<ul style="list-style-type: none"> ▪ Although it has many types of projects, you cannot specify exactly your project, you must adapt to the templates available. ▪ The functionality to upload excels to import data does not work very well. ▪ Is web based, so it is a little bit slow on old hardware and notifications do not work very well. ▪ Learning curves is quite difficult, especially for new users on this type of platform. ▪ There is no free trial. 	<ul style="list-style-type: none"> ▪ Powerful tool to import projects from other platforms. ▪ Being able to create your own templates. ▪ Instead of only web-based, also create a specific app so that notifications work better, and it is faster.
<p>IdeaFlip https://ideafli.com/</p> 	<ul style="list-style-type: none"> ▪ Simple way to perform brainstorm activities ▪ Simple and attractive UI ▪ Sticker Flexibility 	<ul style="list-style-type: none"> ▪ Does not offer voice chats ▪ Does not offer team analytics ▪ Limited access for free users 	<ul style="list-style-type: none"> ▪ Simple attractive UI ▪ Team records analytics
<p>Google Meet https://meet.google.com/</p> 	<ul style="list-style-type: none"> ▪ It allows to transfer files without weight limitations. ▪ It has a phone application. ▪ It provides a caption in real time ▪ All communications are encrypted. ▪ It allows to share screens with high quality. ▪ It allows to record the meetings 	<ul style="list-style-type: none"> ▪ Does not allow more than 250 people to attend the same meeting. ▪ The phone app has not support for old version of android, you must have a modern phone in order to use it. ▪ It has a cost of 5-20€ per month 	<ul style="list-style-type: none"> ▪ It was made for the business environment. ▪ It is relatively new, this app was launched in 2017 ▪ There are more than 10M registered users.
<p>Google For Education https://edu.google.com/intl/es-</p>	<ul style="list-style-type: none"> ▪ Interactive way of teaching ▪ Easy to use ▪ Everything in the cloud (portable) 	<ul style="list-style-type: none"> ▪ Very costly installation and hardware (over 5000\$) 	<ul style="list-style-type: none"> ▪ Interactive environment among teachers and students

419/products/jamboard/ 	<ul style="list-style-type: none"> ▪ Accessible from any internet connected device 	<ul style="list-style-type: none"> ▪ Face to face driven for collaborative use 	<ul style="list-style-type: none"> ▪ Accessible from any internet connected device
Discord https://discord.com/ 	<ul style="list-style-type: none"> ▪ Intuitive interface ▪ Not only individual calls, but you also have servers that are easy to use and very customizable with roles, channels, hide some channels to certain people, etc. ▪ You can set up bots in servers to manage anything you want, from music to timetables, participation, etc. ▪ You can deafen yourself without exiting the voice channel. ▪ You can change the volume of each participant individually. ▪ Best-in-class voice filtering and quality. ▪ You can change the bitrate of the channels in a server in case some people have worse internet connection. ▪ Unlimited people in servers, and you can limit how many people in each channel. 	<ul style="list-style-type: none"> ▪ Sharing screen has very low quality in the free version (only 720p30fps) ▪ The app is quite buggy in all platforms and specially on Linux. ▪ Sometimes in older-hardware can be quite slow, as its web-based. ▪ Mobile app is broken, very slow, notifications do not work, and when opening it takes a long time. 	<ul style="list-style-type: none"> ▪ Offer better video in the base version, specifically 1080p. ▪ Implement a better-quality control and fix bugs in Linux too. ▪ Making an API so that people can interact with the app. ▪ In voice channel apply filtering, Deep Learning based is good but volume level based is fine too. ▪ Being able to change the quality of media so that slow connections work too. ▪ Unlimited people.

Appendix B. Brainstorming

Idea 1	Idea 2	Idea 3
The software for having classes must let the organizer have a total control of the “room”, like managing participants, mute them....	When a participant asks for a meeting filling up all the required fields, a notification must be sent to all students of the group, so they get informed, instead of having to enter the app.	The software should display all the free students (students that does not have team yet), so any other student can select him to send an invitation.
Each room may have different profile settings, such as, for a conference room all participants are allowed to open their mics and for an exposition room only the how is exposing is allowed to open his mic.	Only “instant” notifications are sent, no email ones, because they are annoying.	The application may include a student’s status which can be set by the student.
Having a “ride” feature that the organizer of the call can use to move all the people and himself to another meeting without the participants needing to do anything.	Advanced filtering for the notifications, only allowing certain words or from certain groups.	Automatically check if both student’s timetables are compatible, so that they can attend the same practice group.
To have a system that interprets the meetings like files so each student is an identifier that can be moved from one file to another.	An auto filling system that enrolls the student in all his subjects just introducing his course number.	Does not show the students that already have a pair in the selection area
The application must have a file sharing system to share files among students and teachers.	It must have an intuitive and modern UI with as little bugs as possible.	It may add a statistics/analytics section where the teacher could see the performance of each student

The file sharing system must have the possibility that students can edit the document at the same time. It has to have also control version, so they can undo the changes of the document up to one point.	The notification to the meetings, the participant can select if he wants them to be sent by email, SMS, or other option.	Statistics should be detailed, showing the time each participant has been in the meeting, time with the micro opened, for example...
You can do branches to the document, like gits.	Allowing the user, a more advanced UI or simpler, depending on how knowledgeable he is.	The gamification of those statistics gives points for more participation, or more efficiency.
You can send draft deliverables to keep it in the app but change the delivery after that. (Always before the deadline)	Use some little gifs or hover elements to lead the user in the app.	The student could generate reports of his current marks.
The application should have a high-resolution video, because otherwise it would be impossible to read a shared screen.	It has to take care of the delay in order to provide a face-to-face conversation feeling.	It should be user friendly, we should provide an agile implementation for saving time to the users.
It needs a variable bitrate, so that it has good voice quality and video when internet is good and can work on slower connections too.	It should use some kind of voice filtering.	Advanced search.
Allow input video from other sources than the camera, i.e.: other screens or mobile phones	For more informal meetings voice filters such as movie character voices may be funny	Anti-cheat mechanism to avoid copies.

After each meeting, teams will have the option to fill up a template where they can write which objectives must they accomplish before the next meeting. The points discussed that was already entered when the meeting was created, will already be filled in the template and it could be modified. The minutes of the meeting and participants is also filled automatically. This information could be also downloaded in case it is needed to deliver it.	Students should have the option to put some filter on the background when they activate their cameras during meetings.	When creating a meeting, if someone in the team is occupied (it already has a meeting), the application will tell the student who is trying to create the new meeting that X student is not available on that date, and that he/she should talk with him/her or try another date. (Students can ask for changing the date or deleting of a meeting in case they cannot attend or at the end it is not necessary. All students must confirm for the first case)
Make available an API so that students can practice programming and automate tasks.	Version control on deliveries so that when you upload an assignment and then correct things the teacher can see your progress.	Being able to control other people's computers during calls and meetings.
Make a Bot script that gives feedback about the status of the uploads.	Add a repository system to the page, so you can edit your deliveries anywhere.	Make an VPN in order to share content among students in the easiest way possible.
In code assignments it could be a progress bar that can tell which percentage of the work has been already done. It could be filled by having each point of the statement in a checklist, for example. (example: Integrated Trello)	Whiteboards used during a meeting are saved automatically and can be checked already after finishing the call.	If the pre-established minutes for the meeting are about to finish and more time is required, this could be set (specifying the

		additional minutes) as many times as needed.
An auto checker script could be implemented by the teachers to allow the student to run their code against it and retrieve the number of use cases that their code has successfully passed, with or without knowing which cases were those.	Integrated “post-it” technology to allow students to plan their work with software development methodologies such as SCRUM (integrated idea Flip)	Create a virtual machine environment where students can run test and code simultaneously.