IThink UPC

telecos
BCN

UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

UPC Estudiants APP

Final Report

REVISION HISTORY AND APPROVAL RECORD

Revision	**Date**	**Purpose**
0	21/05/2021	Document creation
1	01/06/2021	Document writing
2	03/06/2021	Document revision 1
3	07/06/2021	Document revision 2
4	11/06/2021	Document revision 3

DOCUMENT DISTRIBUTION LIST

Name	**E-mail**
Adrián Soria Montoya	adrian.soria@estudiantat.upc.edu
Blanca Ruiz Díaz	blanca.ruiz@estudiantat.upc.edu
Héctor Pérez Díaz	hector.perez.diaz@estudiantat.upc.edu
Alba Talaya Vidal	alba.talaya@estudiantat.upc.edu
Youssef El Houas Ghouddana	youssef.el.houas@estudiantat.upc.edu

WRITTEN BY:		REVIEWED AND APPROVED BY:	
Date	21/05/2021	Date	11/06/2021
Name	Youssef El Houas Ghouddana	Name	Héctor Pérez Díaz
Position	Project Leader	Position	Secretary

0. CONTENTS

1. DOCUMENT SCOPE

In this document you will find all the information needed to understand the module developed.

In the following pages you will find how things actually went, starting with a summary of the project with all the details of the functions implemented, and what went right and wrong with the project. You would see that on the time plan and the comparison with the previous plan and how the team adapted in order to develop the project.

Then you can find the description of the system implementation with all the schematics, diagram software blocks that will make you understand how the module works and communicate with the different parts and how it should work, and the final software blocks with all the requests that the application will make to work properly. Finally, a short demonstration of the performance and a short discussion of why the team thinks it is the best implementation taking into account the resources available.

An analysis of the risks are also in this document. This can be found in the previous document, but these are an update with some risks that the team found when developing the module, so they are more complete than the previous ones.

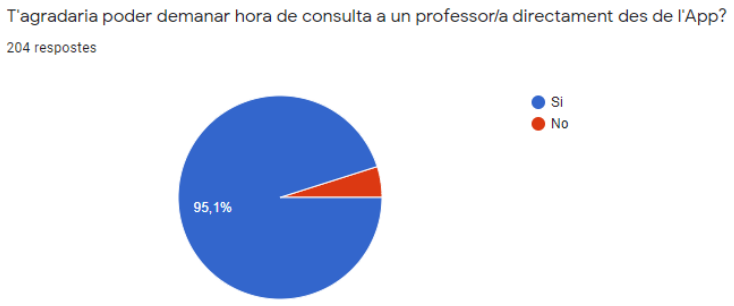
Following those, are the cost analysis, with the components list cost, the main tasks cost and the estimated equipment used if we were not students, and we have to pay for all the facilities used. In this point we make an estimation of how much it will cost to the client if he/she uses the same services as us, but iThinkUPC have their own servers. Also, there is the business model canvas of our project with a comment of the key points of it, and the sustainability analysis with all the costs and consequences that our project have, emphasizing that we are mainly a full software project.

Finally, you will find the conclusion of our project and a reflection of how things went as a team, what we have done right and what we will improve if we have to do it again.

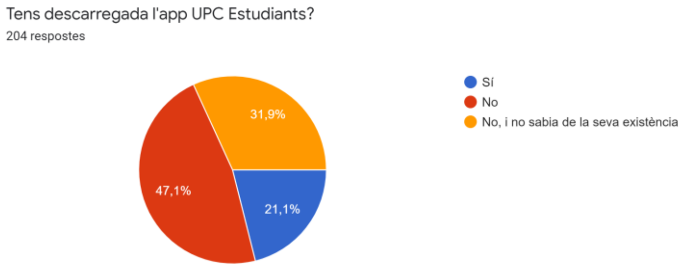
2. PROJECT SUMMARY

The aim of our project was to think about what functionalities add to the existing Estudiants UPC application.

These functionalities had to be interesting for students in order to cover the needs of the users and make the application more widely used. So we carried out a survey to more than two hundred students that included the different ideas we had thought of.


T'agradaria poder demanar hora de consulta a un professor/a directament des de l'App?

204 respostes

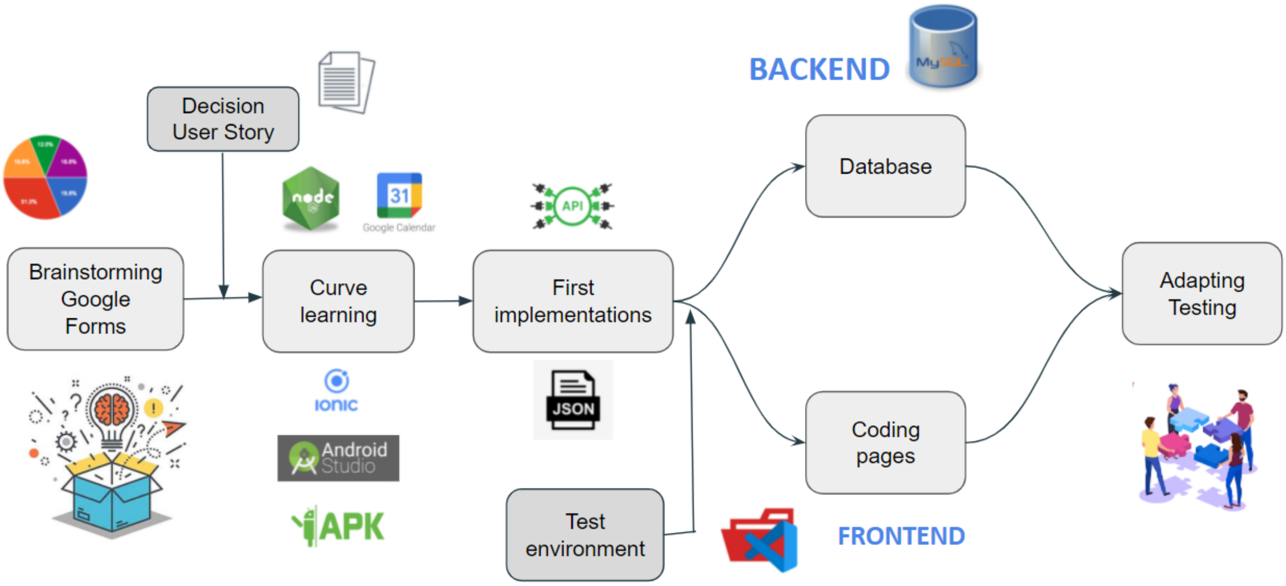

Tens descarregada l'app UPC Estudiants?

204 respostes

As you can see in the graphs, most of the students did not use the app, but they would find it useful to be able to ask their teachers for advice directly from the app.

So we decided to implement this functionality in the Estudiants app, creating a simple and intuitive way to request a consultation time, which will appear as an event in the google calendar of the student and the teacher.

Once we knew the needs of the product, we could start to make our project, and we summarized in this diagram all the phases we have made to get our implementation.



First we were in the phase of definition, where we defined which idea to carry out, and then we started the work in 2 branches, the frontend part and the backend part.

We started to do research and a curve learning of the types of programming we used to develop our project and make the first implementations of our idea until we waited for the test environment given by IThinkUPC. We have needed it to implement and make the testing of our improvement in the App.

Once we had it, the Backend did a DataBase to save the necessary information and provide it to the App, and the Frontend made the coding views of the different pages where the user will be able to see this information. Finally, we mixed the two parts and we did the necessary changes and adapted the code to obtain the final project successfully.

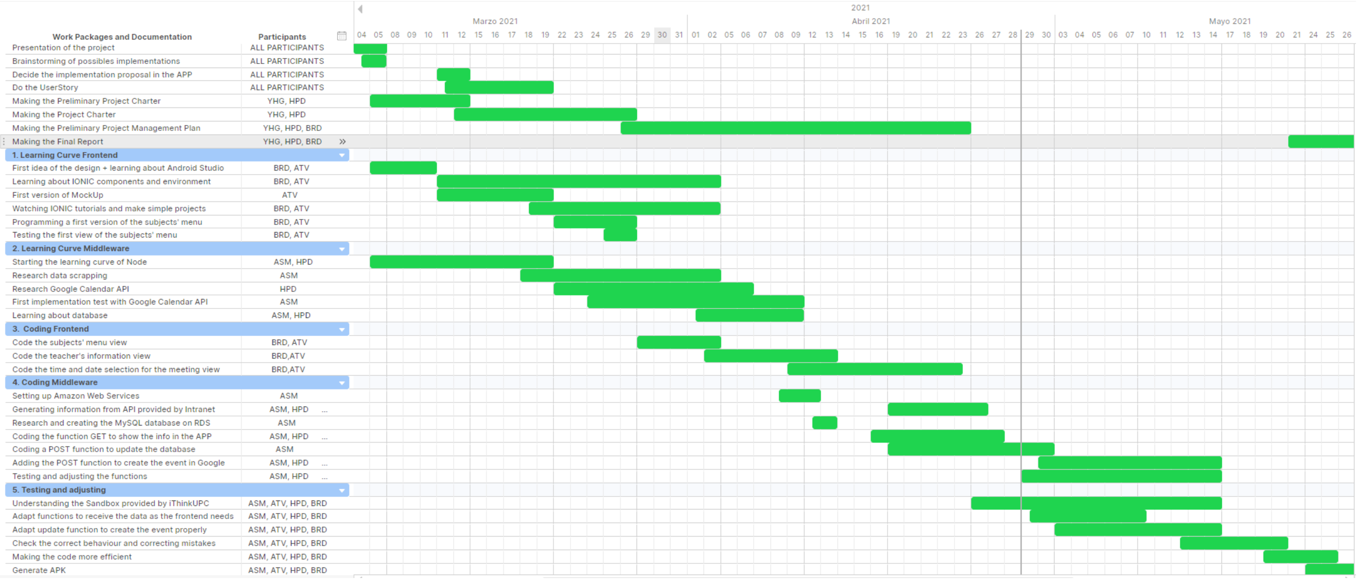
The following sections of the document go into more detail on the whole development process and what assumptions we had to make in order to achieve a functional version in the time we had available, also this version is shown in this document.

3. TIME PLAN UPDATED

In this section we have included our final Time Plan. We have used the Tom's Planner template to do it, and we have updated our work weekly. We have added tasks in the work packages previously defined in the Management Plan, where we have included all the activities we have done until our last day of work, the 28th of May, when we finished our final project. Also, you can see the workers of the team and what kind of activities have been done during these weeks, although we have helped each other in all the tasks.

Regarding the delays, we have delivered the documents on time, and we have managed to finish the task that we were planning. But we had to make changes in our Time Plan when we obtained the test environment given by iThinkUPC on the 22nd of April. That's why the Work Package 5 is completely different from the Management Plan. We had to spend a lot of time to understand the SandBox and adapt our implementations in it. So we dedicated a lot of hours in that part because of the testing and checking the correct behaviour of our improvement in that environment.

Finally, we include in this section the Gantt Diagram with all the activities we have done during the project and the time we spent making them. We want to add that we have started to complete the Final Report the 21st of May, and we finished on the 11th of June, when we delivered this document with the 2 min video requested by the professors.



Work Packages and Documentation	Participants
Presentation of the project	ALL PARTICIPANTS
Brainstorming of possibles implementations	ALL PARTICIPANTS
Decide the implementation proposal in the APP	ALL PARTICIPANTS
Do the UserStory	ALL PARTICIPANTS
Making the Preliminary Project Charter	YHG, HPD
Making the Project Charter	YHG, HPD
Making the Preliminary Project Management Plan	YHG, HPD, BRD
Making the Final Report	YHG, HPD, BRD
1. Learning Curve Frontend	
First idea of the design + learning about Android Studio	BRD, ATV
Learning about IONIC components and environment	BRD, ATV
First version of MockUp	ATV
Watching IONIC tutorials and make simple projects	BRD, ATV
Programming a first version of the subjects' menu	BRD, ATV
Testing the first view of the subjects' menu	BRD, ATV
2. Learning Curve Middleware	
Starting the learning curve of Node	ASM, HPD
Research data scrapping	ASM
Research Google Calendar API	HPD
First implementation test with Google Calendar API	ASM
Learning about database	ASM, HPD
3. Coding Frontend	
Code the subjects' menu view	BRD, ATV
Code the teacher's information view	BRD,ATV
Code the time and date selection for the meeting view	BRD,ATV
4. Coding Middleware	
Setting up Amazon Web Services	ASM
Generating information from API provided by Intranet	ASM, HPD ...
Research and creating the MySQL database on RDS	ASM
Coding the function GET to show the info in the APP	ASM, HPD ...
Coding a POST function to update the database	ASM
Adding the POST function to create the event in Google	ASM, HPD ...
Testing and adjusting the functions	ASM, HPD ...
5. Testing and adjusting	
Understanding the Sandbox provided by iThinkUPC	ASM, ATV, HPD, BRD
Adapt functions to receive the data as the frontend needs	ASM, ATV, HPD, BRD
Adapt update function to create the event properly	ASM, ATV, HPD, BRD
Check the correct behaviour and correcting mistakes	ASM, ATV, HPD, BRD
Making the code more efficient	ASM, ATV, HPD, BRD
Generate APK	ASM, ATV, HPD, BRD

4. SYSTEM DESIGN DOCUMENTATION

Assumptions

There will be some assumptions on which our project will work around, these were made before and during the project design:

- The first assumption we made was that if the teacher doesn't accept the meeting it means that he/she is not free at that time so we will delete that date.
- The second one was that the information about the professors will be updated every half year, so this is when we are going to update the information database, coinciding with the beginning of each course.
- The third one is that the servers, Google services and the APP will be working properly and won't have failures or problems.
- The next one is that the students will check the teacher's answer to the arrangement on their own and every user will be responsible for putting the reminders on Google Calendar. The problem is that with an external mail administrator we cannot send reminders to the users, so they have to enable the reminders in their own Google Calendar configuration.
- Another one is the fact that the information about all the teachers will be in the database and the meeting slots will last 30 minutes.
- And the last one is that the implementation we made is only for ETSETB because we could get the information we needed from the faculty. We asked FIB, Camins and Terrassa how they had the information about the teacher's consultation hour, and their responses were very similar. They don't have this information unified in one platform. So, we made the supposition that the other faculties work as our faculty does. It would be nice to be able to implement it to all the UPC.

The assumptions we made are the suppositions our project is based on and are the justification about how we worked and made the implementation we proposed.

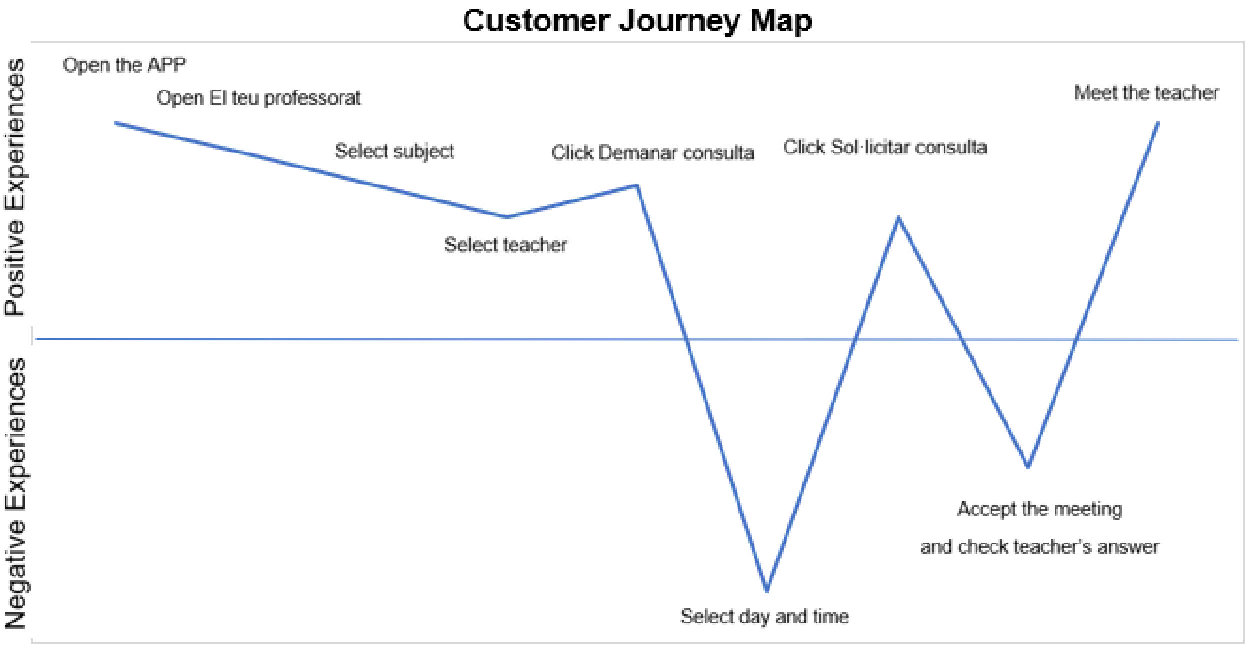
User Story

To explain to the IThinkUPC manager our idea to improve the UPC Estudiants App, we did a User Story. This is a guide that includes the designs of how you would like it to be displayed in the App and all the steps to follow to make the consulting correctly. In this way, we had a visual and well explained material so the manager understood us well and could have more feedback from him. This kind of document is really useful and helps us to organize our ideas and to know the next steps to reach our improvement.

We decided to do it because our supervisor told us about it. We could only have one meeting with the IThinkUPC company so it was the way to make the most of that time.

User Journey Map

To know the experience that the customers have when they are using our product we made a user journey map. In this map we can see the satisfaction level of the user during the different phases and what he/she can see or cannot see.

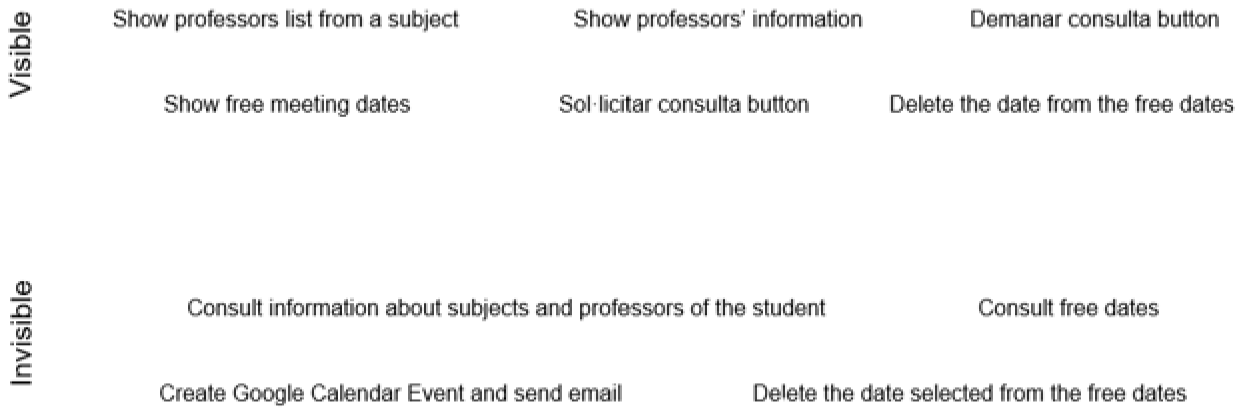


When the user starts the process of requesting a consultation hour he/she is in a good mood. Since then the mood is going down and getting worse until all the process is done. The first critical moment the user passes is when he/she has to select a date to have the meeting, this is because the user has to select a day that he/she has available and sometimes this can be a problem.

Another critical moment is when the user has to accept the meeting and has to check the teacher's answer too. This is caused because there are other applications involved in the process that the user has to open. Sometimes the teacher will not accept the meeting and this moment will make the user have a bad experience.

The best moment of the process is when the teacher accepts the meeting and the student assists to the meeting to solve questions. This is the final moment of all the process and is the best moment for the user because this is the reason why our product is developed.

Interaction line


Visible
Show professors list from a subject
Show professors' information
Demanar consulta button
Show free meeting dates
Sol·licitar consulta button
Delete the date from the free dates
Invisible
Consult information about subjects and professors of the student
Consult free dates
Create Google Calendar Event and send email
Delete the date selected from the free dates

While the user is advancing in the steps, there are some things involved that he/she can and cannot see. The components that the users can see in the application during the trip is what we call the Frontend and it is related to the application design, such as the buttons, the menus or the date selector , and the components that the users cannot see during their interaction with the application is what we call Backend or Middleware, such as the database consultations, the calendar event creation or the deletion of the dates selected.

Persona Canvas

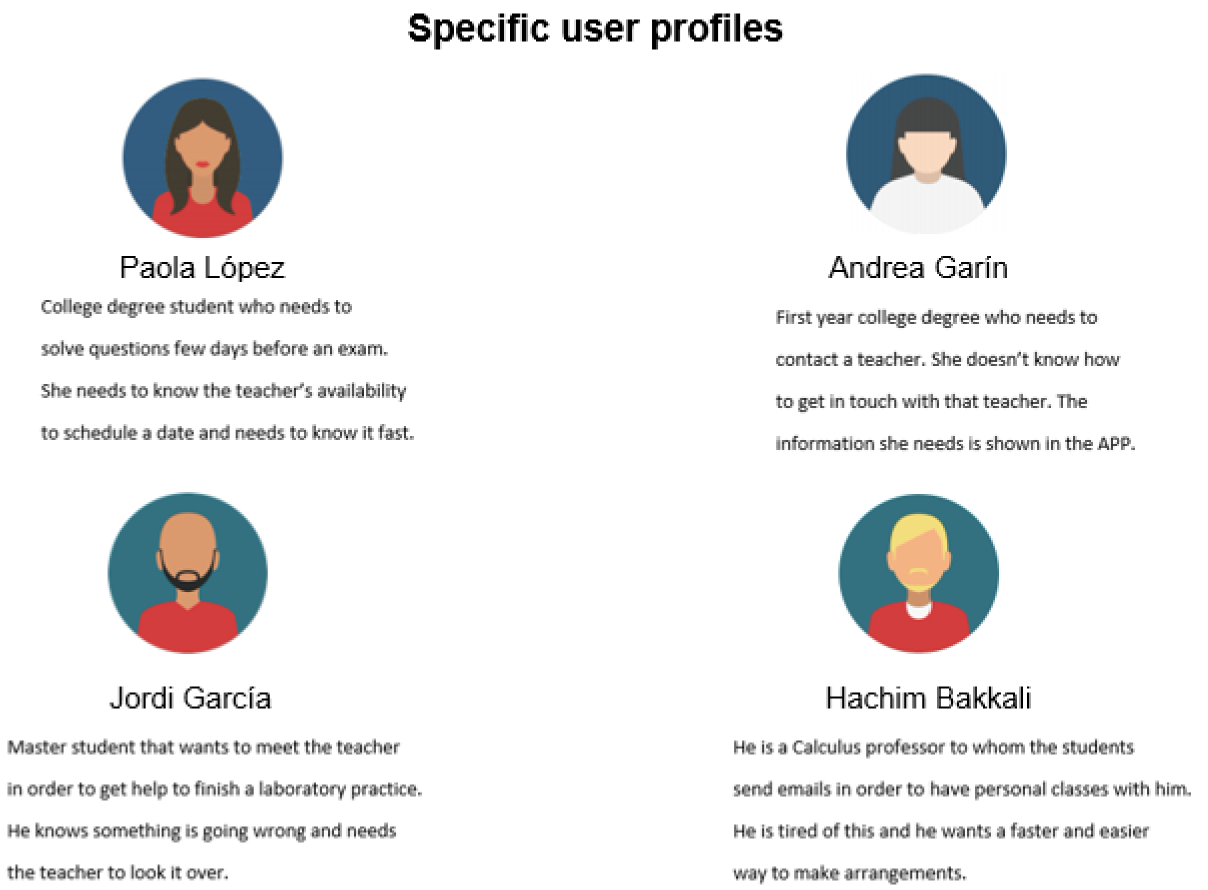
To know more about the users who are going to use the application we invented different types of user profiles. We separated the customers in four different profiles:

· The first one is a degree student. This type of student is going to use the application to have a consultation with the teacher to solve questions for example some days before an exam. There are a lot of students in her class so she will use the app in order to schedule a date fast and she will be able to know when the teacher is available for the meeting.

· The second one is a new degree student that wants to ask some questions to the teacher and she doesn't know how to get the information she needs to contact the teacher. This information is shown in the application. She doesn't know where this information is placed so she will use the app in order to get it.

· The third one is a master's degree student. This kind of user is going to use the application to solve laboratory practice questions.This type of user is not going to use this implementation a lot because he knows where he can find the information he needs to contact the teacher, there are less students in his class so the relationship with the professor will be closer than the degree students and sending an email will be as fast as using the application in this case. His teacher will have more free dates available because of the reduced number of students in the class and maybe he is going to adapt his consultation hours to the students.

· The last one is a teacher who is tired of receiving emails to schedule an hour to make a meeting with a student. The app will make him save sending a lot of emails to agree on a date.



Specific user profiles

Paola López

College degree student who needs to solve questions few days before an exam. She needs to know the teacher's availability to schedule a date and needs to know it fast.

Andrea Garín

First year college degree who needs to contact a teacher. She doesn't know how to get in touch with that teacher. The information she needs is shown in the APP.

Jordi García

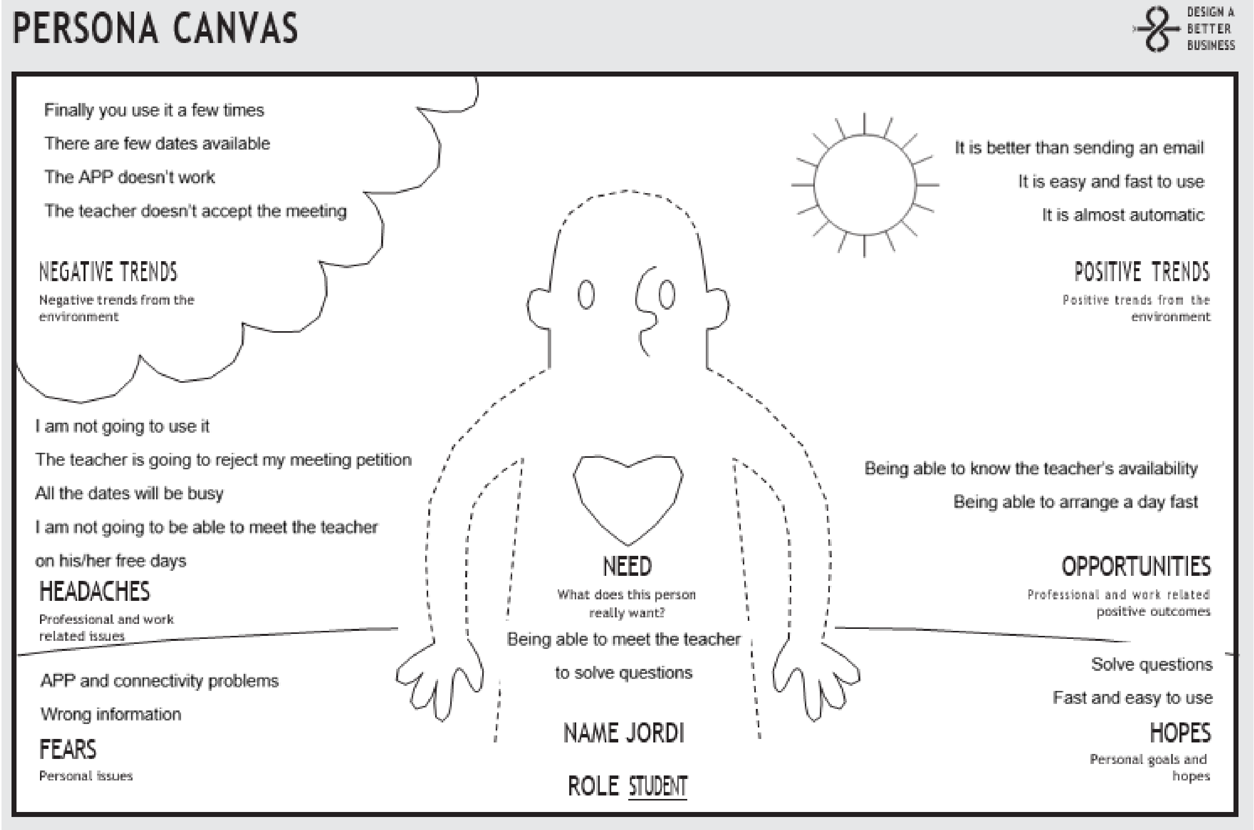
Master student that wants to meet the teacher in order to get help to finish a laboratory practice. He knows something is going wrong and needs the teacher to look it over.

Hachim Bakkali

He is a Calculus professor to whom the students send emails in order to have personal classes with him. He is tired of this and he wants a faster and easier way to make arrangements.

Our implementation to the APP UPC Estudiants will make the process of scheduling a date to have a meeting with a teacher easier to the students and professor. Every user has different needs to be satisfied and with this objective they will use the application.

We don't discard the possibility of all kinds of students using the application in order to have a meeting with the teacher, but we know that there are some student profiles that will use it more than others.



PERSONA CANVAS

DESIGN A BETTER BUSINESS

Finally you use it a few times

There are few dates available

The APP doesn't work

The teacher doesn't accept the meeting

NEGATIVE TRENDS

Negative trends from the environment

It is better than sending an email

It is easy and fast to use

It is almost automatic

POSITIVE TRENDS

Positive trends from the environment

I am not going to use it

The teacher is going to reject my meeting petition

All the dates will be busy

I am not going to be able to meet the teacher on his/her free days

HEADACHES

Professional and work related issues

NEED

What does this person really want?

Being able to meet the teacher to solve questions

Being able to know the teacher's availability

Being able to arrange a day fast

OPPORTUNITIES

Professional and work related positive outcomes

APP and connectivity problems

Wrong information

FEARS

Personal issues

NAME JORDI

ROLE STUDENT

Solve questions

Fast and easy to use

HOPES

Personal goals and hopes

To know more about the users of the application we made a persona canvas. The objective is to know what the user is going to think when he/she is using the app. We tried to dig deeply into the user's thoughts.

The main need that has to be satisfied is the fact of meeting the teacher to solve questions. This need is going to be accomplished with the app implementation we made.

The persona canvas helps us to balance the good and bad thoughts that the user is going to have while using the app, and makes it possible to see the product as the customers would see it.

To sum up the persona canvas, we can see in the image that Jordi has the NEED of having a meeting with a teacher. He has the FEAR that the app can have connectivity problems and is not going to work properly, or the HEADACHE that all the days are going to be booked and he is not going to find a date to meet the teacher. But in spite of that, he is going to give it a chance, because the App gives him the OPPORTUNITY to know the teacher's availability, so he HOPES it's a fast and easy process.

Frontend:

The initial designs of the frontend, are those that can be seen on the User Story previously shown.

We will add the option “El teu professorat” to the burger menu, and from then you can access all the teachers’ names from each subject. If you click on one, their information will appear visible and you will be able to select a day and time to book a meeting with them.

Our first idea was that if you clicked on the “Demanar consulta”, you would automatically be redirected to your default mail app, and you would have an already written message stating that you want to have a meeting with that teacher, and you could manually write the day and time.

We quickly realized that it was not an optimal solution, because you were not able to see which time slots the professor was already busy, so we decided on doing the current solution of selecting one from the options we display.

If we compare the initial designs with the final product, we can see that not a lot has changed, only two main things:

- The buttons design have been adapted to look like the ones already existing on the app.
- The professor’s photo is not shown anymore. This is due to the fact that we ended up taking all the information from the Intranet, and they did not give us access to the photos.

Backend:

Initially there was the idea to delegate the management of the meetings to each professor, so if a teacher is busy he or she would have to decline the meeting and do it for every student. This idea was later discarded and the MySQL database was implemented.

So the initial schematic was very simple. IONIC will make the petition directly to the Google Calendar API and also get all the information from the API REST from intranet and by doing scrapping from the directori UPC which let us implement these for all the university. This scrapping was quickly discarded because we have all the information in the API REST, and we make the assumption that all the schools have this information similarly, and it only added more libraries to the project.

This implementation has a lot of problems, and it is easy to upgrade, so we made new implementations in order to make it easier and more comfortable to use. For instance, we took control of the booked hours and the hours that the professor used for the meetings.

5. SYSTEM IMPLEMENTATION DOCUMENTATION

Frontend:

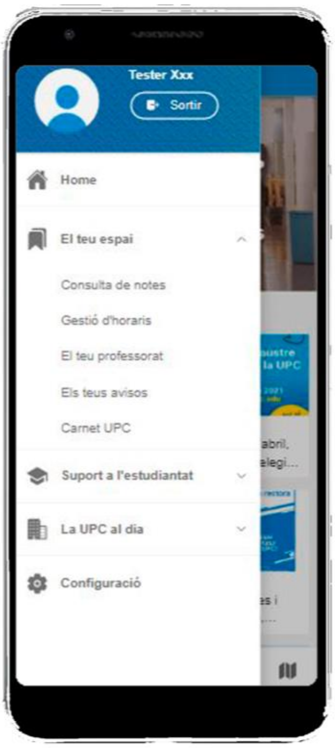
In this section we will show and explain the screenshots of the final project.

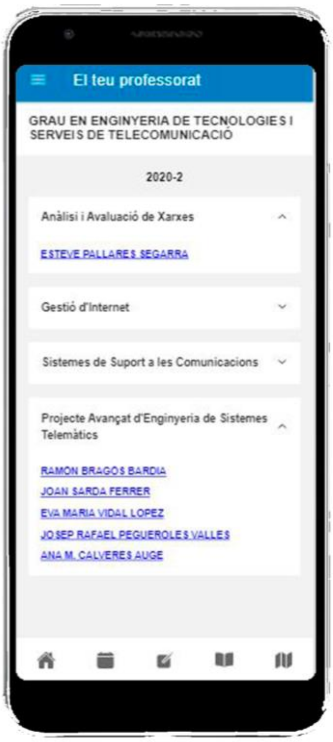
We started building our part as a modular app, and once we were done, we included it in a Sandbox that IThink UPC sent us so that we could work with a version of the whole app.

The first screenshot is from the real app, so we added the option to access the directory as it is shown on the second one. You have to go to the Burger Menu→ El teu Espai→ El teu professorat. This will bring you to the next page.

As you can see, the first view is a list of all the subjects you are currently coursing, and if you click each of them, it expands so the names of all the teachers appear beneath.

All this information is read from a json that we received with the Sandbox, called get-matricula.It initially came with a tester user studying Architecture, but we changed it so we could display our own subjects and teachers. If you want to see the information of someone specific, you can click on their name and you will be redirected to a new page.





GRAU EN ENGINYERIA DE TECNOLOGIES I SERVEIS DE TELECOMUNICACIÓ

2020-2

Anàlisi i Avaluació de Xarxes

ESTEVE PALLARES SEGARRA

Gestió d'Internet

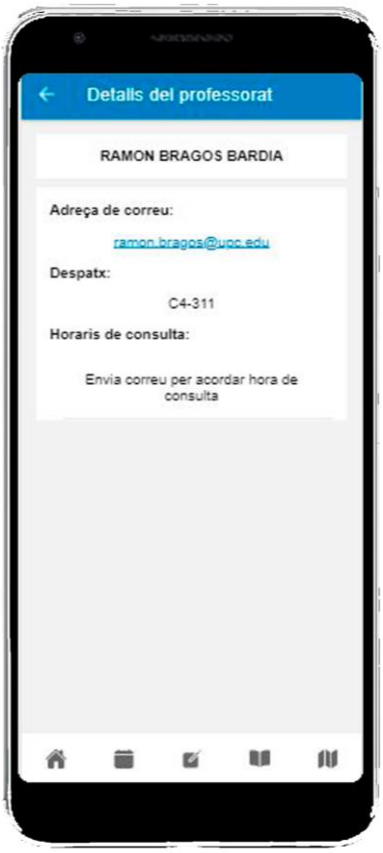
Sistemes de Suport a les Comunicacions

Projecte Avançat d'Enginyeria de Sistemes Telemàtics

RAMON BRAGOS BARDIA
JOAN SARDA FERRER
EVA MARIA VIDAL LOPEZ
JOSEP RAFAEL PEGUEROLES VALLES
ANA M. CALVERES AUGE

Then you can see the two cases you can find when you click on the professor's name. In both cases you will be able to see the information about their office and their emails. But as we can see in the first screenshot, Ramon doesn't have meeting hours registered. So if you want to have a meeting with him, you would have to click on the email, which is linked with your default mail account, and write to him to book a meeting day.

In the case of Josep, he has the meeting hours registered. So you will be able to see them and do the consulting from the App clicking on "Demanar Consulta".



Detalls del professorat

RAMON BRAGOS BARDIA

Adreça de correu:

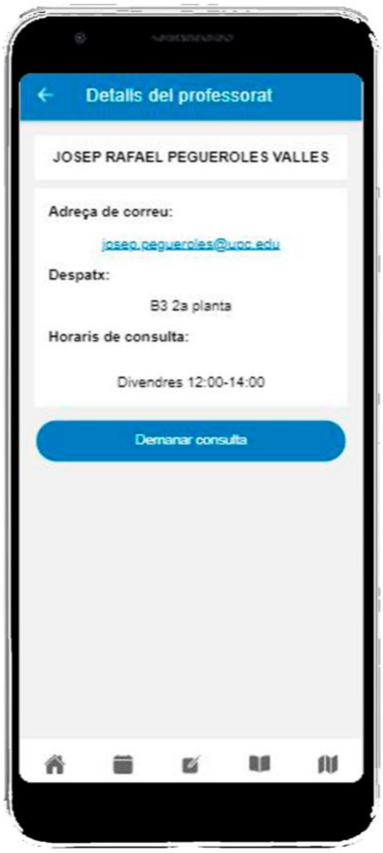
ramon.bragos@upc.edu

Despatx:

C4-311

Horaris de consulta:

Envia correu per acordar hora de consulta



Detalls del professorat

JOSEP RAFAEL PEGUEROLES VALLES

Adreça de correu:

josep.pegueroles@upc.edu

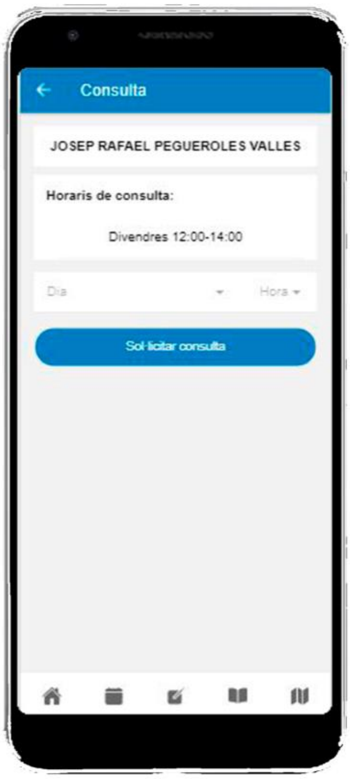
Despatx:

B3 2a planta

Horaris de consulta:

Divendres 12:00-14:00

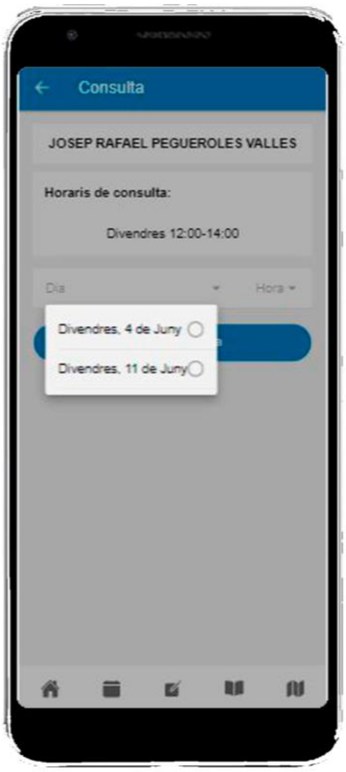
Then we can find the process you have to follow in order to book a meeting. First of all, you will see the first screenshot with the meeting hours and the available days and hours. The next step would be to select the day and the hour you want to book, and finally click on "Sol·licitar consulta".



JOSEP RAFAEL PEGUEROLES VALLES

Horaris de consulta:

Divendres 12:00-14:00

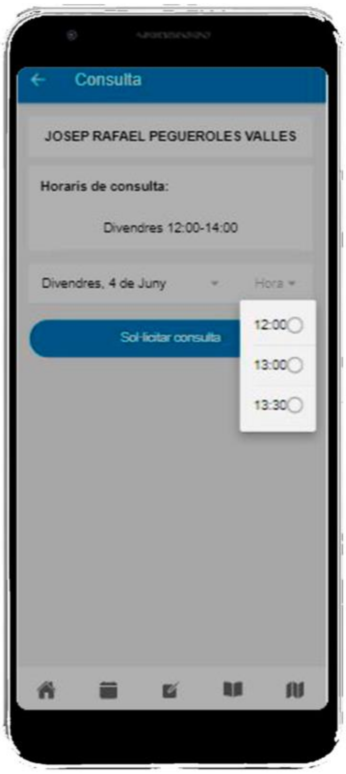


JOSEP RAFAEL PEGUEROLES VALLES

Horaris de consulta:

Divendres 12:00-14:00

Divendres, 4 de Juny ○
Divendres, 11 de Juny ○



JOSEP RAFAEL PEGUEROLES VALLES

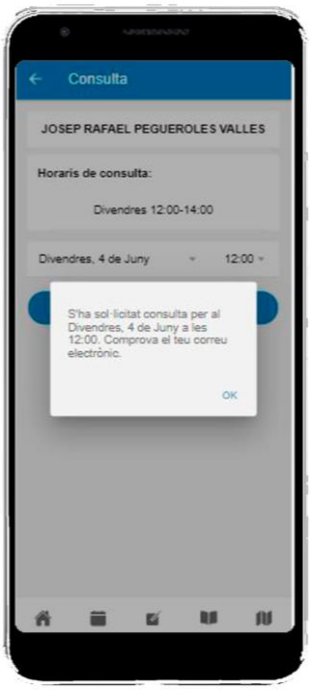
Horaris de consulta:

Divendres 12:00-14:00

Divendres, 4 de Juny

12:00 ○
13:00 ○
13:30 ○

Then, the App shows a popover and informs you that the meeting has been requested and you should check your email, because the event will be created in the Google Calendar with all the information and you should have received a notification.



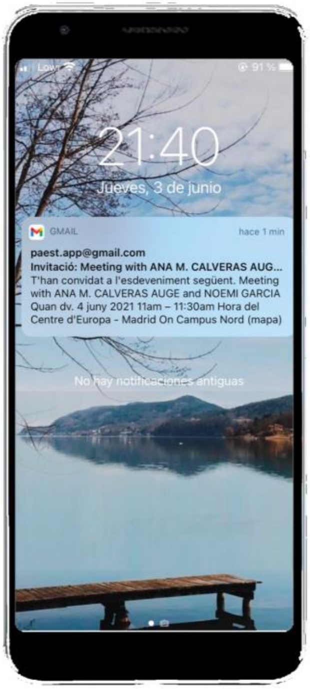
JOSEP RAFAEL PEGUEROLES VALLES

Horaris de consulta:

Divendres 12:00-14:00

Divendres, 4 de Juny 12:00

S'ha sol·licitat consulta per al Divendres, 4 de Juny a les 12:00. Comprova el teu correu electrònic.

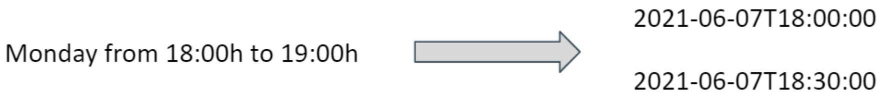


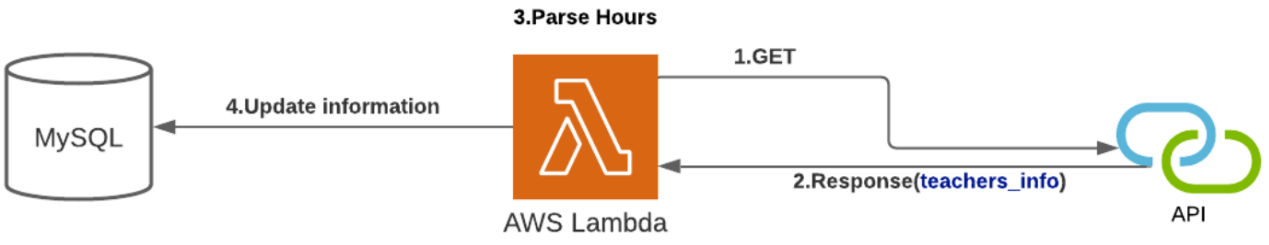
paest.app@gmail.com
Invitació: Meeting with ANA M. CALVERAS AUG...
T'han convidat a l'esdeveniment següent. Meeting with ANA M. CALVERAS AUGE and NOEMI GARCIA Quan dv. 4 juny 2021 11am – 11:30am Hora del Centre d'Europa - Madrid On Campus Nord (mapa)

Backend:

The proposal from the previous point is a very simple one. To do so, we are going to create our own database and server to execute all the scripts, using the services of the Amazon Web Services (AWS). The scrapping idea is discarded since we can obtain all the information from the intranet.

Firstly, we need to create the database, we choose MySQL because it is an easy and popular language. The MySQL database has two tables, one with the information of all the teachers, the identification (ID), their email, office, reserved hours and their full name. The second table has the same ID, and the meeting hours parsed as the system (the Lambda Server and the Google API) will need it.

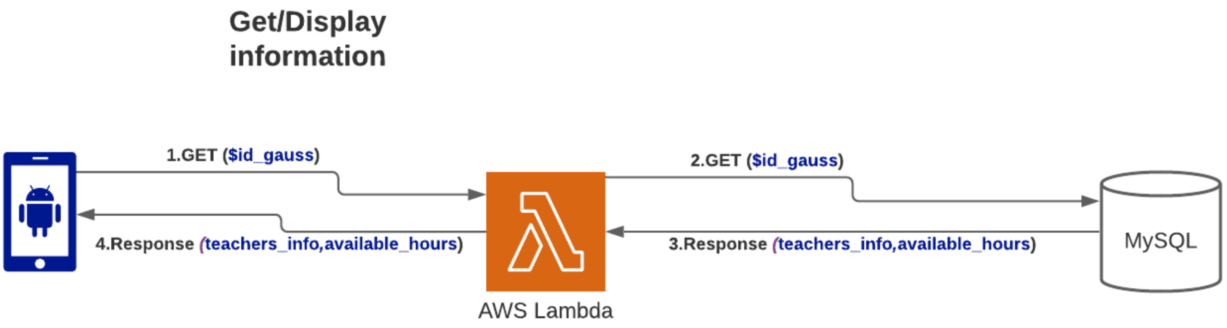




In the image we can see how this function works. Every certain time, ideally at the beginning of every quarter because it is when the information related to a teacher is susceptible to change, this function is executed. It updates the information to the last version in the intranet database. For this project we did this manually because we are in a testing environment, and we do not want to overcharge the API.

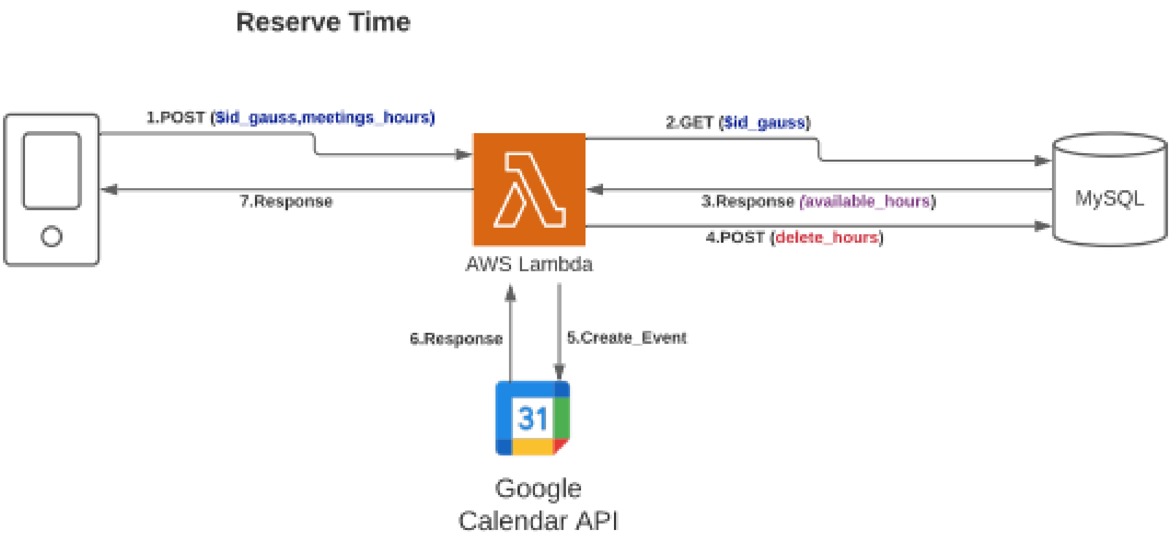
Secondly, we created an AWS Lambda, a server that runs our scripts in Javascript. Here we have two functions, one that manages to send all the information to the frontend, and another one that updates and schedules the meeting.

GET function:


Get/Display information
1.GET ($id_gauss)
4.Response (teachers_info,available_hours)
2.GET ($id_gauss)
3.Response (teachers_info,available_hours)
AWS Lambda
MySQL

This function receives the information of the professor and displays it in the app. First, the app makes a request to the lambda server by passing the ID of the professor, which is stored in the frontend. The lambda server gets the information from the MySQL server and makes a unique response. This response is an array of two elements. The first one is a JSON format with all the information that is required to show in the app. The second one is an array with all the hours available, filtering only the ones that are in the future. This element is used in case that the user wants to make an appointment. We have decided to send this information in order to reduce the amount of petitions that the frontend makes.

POST function & Google Calendar API:



In order to schedule a meeting, the lambda function requires the ID and the hour that the student selected to be booked. Lambda checks that the date is in the database searching the teacher by ID, and deletes it before creating the Google calendar event.

Finally, the Google Calendar API creates the event and sends the response that the meeting has been created. This API is made in NodeJS and uses an administrator email to create the events in the calendar. When a user selects a date to have a meeting, the API creates a new event with the information about the teacher and the student such as the name, the subject and the email. When the event is created, the administrator of the calendar sends an email to the participants to know if they are going to attend the meeting.

This API is implemented using Amazon Web Services and Google Cloud. We decided to use an administrator email because to have access to an email we need some private OAuth credentials, and it would be difficult to get the permissions from all the student emails. Another reason is that these credentials are temporary, and sometimes they can be deleted when the lifetime expires. We don’t have this problem because there is an auto refresh button that validates the credentials we are using.

6. SYSTEM CHARACTERIZATION

The project is properly implemented for the teachers of the ETSETB that have registered their meeting hours, so it can be easily scalable if all the teachers put their information in the database of the intranet.

We used this structure with the Amazon Web Services because we had to implement the control of the schedules, and we had to be in a testing environment. The AWS is always available, and it can be accessed from anywhere, furthermore they are widely used in companies, and it is a first good approximation to a real world implementation

.

1. When we designed the structure, we had always in mind to make the minimum number of communications. This made us send all the hours available to book, even if the user only wants to see the teacher information.

Although this can produce an error when scheduling a meeting, if two users download the same hours when checking the information of the professor, and then, both decide to schedule the same date at the same time, it can schedule the hours for the two users. We have made a control in the POST function, but if it is done simultaneously it can happen but it is very unlikely.

Github Repositories of the project → https://github.com/PAEST-APP?tab=repositories

In order to follow our project better, we provide the link to all the documents produced

during the development of this project. →

https://drive.google.com/drive/folders/1A_aGv_pjEh48taH0p-OV7-XmMeo9L0WN?usp=sharing

7. RISKS

	Impact	1	2	3	4	5
Probability						
5		🟩	🟨	🟧	🟥	🟥
4		🟩	🟨	🟨 Inaccurate Estimations	🟧	🟥
3		🟩 Team member's illness	🟩	🟨	🟨	🟧 Delay getting resources
2		🟩	🟩	🟩 Technologies never used by the team before.	🟩	🟨 Scalability
1		🟩	🟩	🟩	🟩 User's provoke not expected behaviour	🟩 Servers are down

As we have developed a software project, we do not have problems like lack of components or the risk to break a component or get injured, as other projects, instead we have more risks related to infrastructure on our module works.

Inaccurate Estimations: Since we are a small group and not having the knowledge and due to the scope variations, maybe we make a misestimation of the deadlines of the project. We will use tools such as Slack and Trello to mitigate the lack of organization and optimize our productivity.

Delay getting resources: Our module gets information from different sources and some are private, so we need APIs in order to collect all the data. So, we depend on external sources that can respond to us too late.

Servers are down: GitHub, Google services, Atenea and other servers can be, so we can have some problems at certain moments. We have to accept that because those services are external services and do not depend on us.

Technologies never used by the team before: Some tools that we use in this project are new to us, so as part of the learning process there is the risk to have some deprecated versions or to not have enough documentation to self-learn. We are going to mitigate this by doing some previous research before the sandbox is provided.

Team member's illness: Due to the Sars-CoViD-19 pandemic, any member of the team can get ill. As we are a young team, and we can work from home, this won’t have almost any impact on the project. Nowadays, online meetings and online working is very common, so we are just going to do most part of the project online.

Scalability: The application is developed to the “telecos” faculty, so one of the risks is that not all faculties has this information in a database, or they don’t have the consulting hours. For this problem we have decided to prevent it so if we don’t have the availability of the professor, we show a message to encourage the student to send an email.

User’s not expected behaviour: The module has a minimum chance to schedule a meeting with one professor in the same hour at the same time if the reservation is made at the same time. Despite that is a design risk that we can assume because it is very unlikely that this occurs in a real scenario. We are going to accept that risk.

8. COSTS

Components list:

As our project is basically software, we haven’t needed any kind of components to do it.

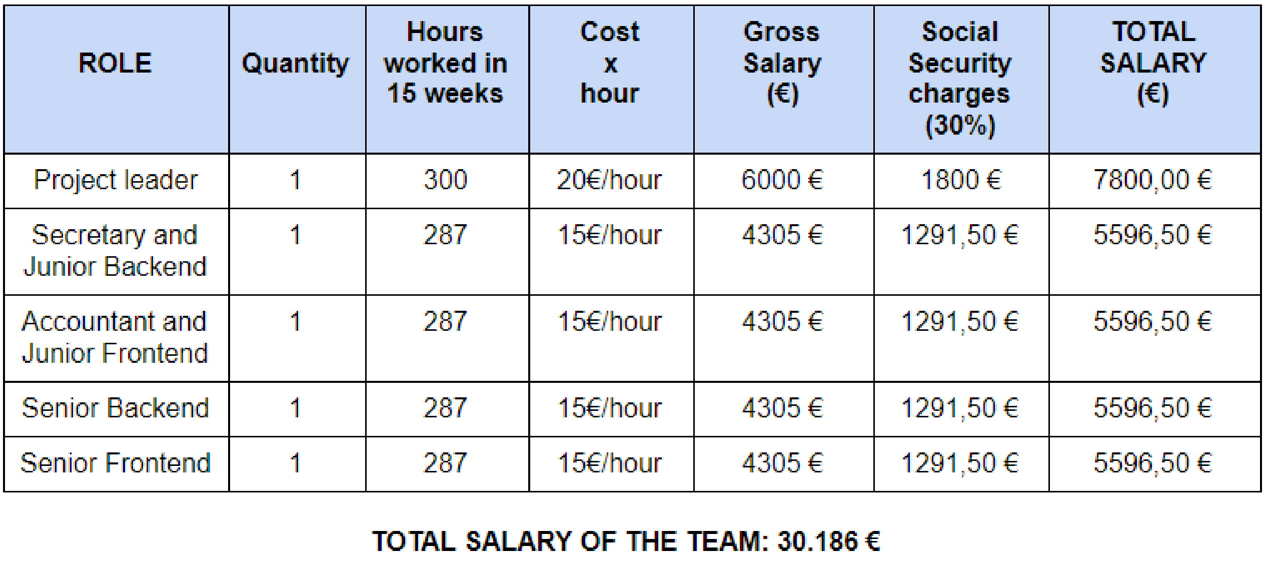
Design and prototyping:

In this section, we are going to include two tables to define the costs of the workers during these 15 weeks of work.

This first table will include the general salaries. There are 5 workers in the team, where the project leader will receive 20€/hour and the rest of the team will receive 19€/hour.

We can see how the project leader has worked more hours than the others. With this information, we have obtained the gross salary of each worker.

Apart from this, we need to take into account the 30% for Social Security paid by the enterprise to obtain the total salary of the team:



ROLE	Quantity	Hours worked in 15 weeks	Cost x hour	Gross Salary (€)	Social Security charges (30%)	TOTAL SALARY (€)
Project leader	1	300	20€/hour	6000 €	1800 €	7800,00 €
Secretary and Junior Backend	1	287	15€/hour	4305 €	1291,50 €	5596,50 €
Accountant and Junior Frontend	1	287	15€/hour	4305 €	1291,50 €	5596,50 €
Senior Backend	1	287	15€/hour	4305 €	1291,50 €	5596,50 €
Senior Frontend	1	287	15€/hour	4305 €	1291,50 €	5596,50 €

TOTAL SALARY OF THE TEAM: 30.186 €

In the second table, you will be able to find the same information but separated by main tasks, and the cost x hour that the workers have received for each task. We have defined the work packages as in the Gantt, with all the tasks done and the all the workers that have participated in them:

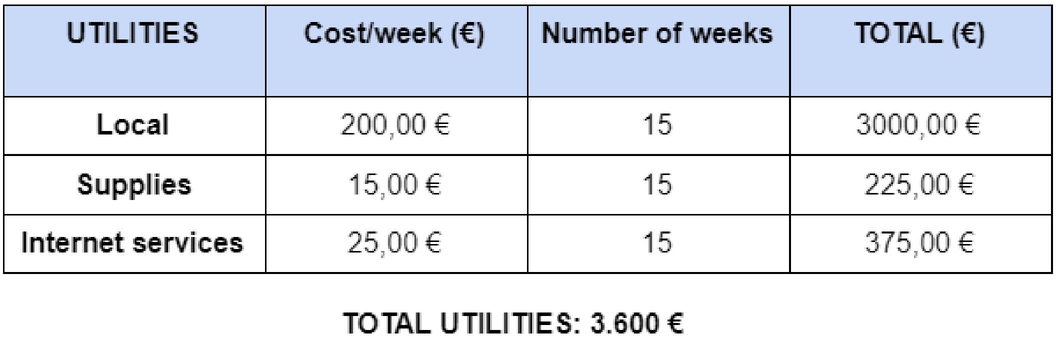
WP#	Tasks#	Title of the task	Participants	Hours x cost
0	1	Presentation of the project, brainstorming and final decision of the new implementation	ALL	YH: 10 hours x 20€ = 200€ AT,AS: 10 hours x 15€= 150€ BR,HP: 10 hours x 12€ = 120€
	2	Contacting with the IThinkUPC, Directori, Intranet and Atenea managers, via email or meet	YH	35 hours x 20€ = 700€
	3	Weekly meetings with the supervisor	YH	50 hours x 20€ =1000€
	4	Weekly meeting with the complete team	ALL	YH: 45 hours x 20€ =900€ AT,AS: 45 hours x 15€=675€ BR,HP: 45 hours x 12€ =540€
	5	Project Charter	YH, HP	YH: 30 hours x 20€ = 600€ HP: 6 hours x 12€ =72€
	6	Project Management Plan	YH, HP, BR	YH: 50 hours x 20€=1000€ HP: 10 hours x 12€ =120€ BR: 6 hours x 12€ = 72€
	7	Final Report	YH, HP, BR	YH: 80 hours x 20€ = 1600€ HP: 15 hours x 12€=180€ BR: 12 hours x 12€ =144€
1	1	Learning about Android Studio	BR, AT	AT: 13 hours x 15€=195€ BR: 13 hours x 12€=156€
	2	Learning about IONIC	BR, AT	AT: 4 hours x 15€=60€ BR: 4 hours x 12€=48€
	3	MockUp, first visual design	AT	AT: 18 hours x 15€=270€
	4	IONIC tutorials, making simple projects to learn	BR, AT	AT: 20 hours x 15€=300€ BR: 20 hours x 12€=240€
	5	Programming first version of the subject's menu	BR, AT	AT: 13 hours x 15€=195€ BR: 13 hours x 12€=156€

	6	Testing this first version	BR, AT	AT: 2 hours x 15€=30€ BR: 2 hours x 12€=24€
2	**1**	Starting the learning curve of Node	HP, AS	AS: 6 hours x 15€ =90€ HP: 6 hours x 12€ =72€
	2	Coding scrapping from directori UPC and adjusting with the format needed	AS	5 hours x 15€ =75€
	3	Research about Google Calendar API	HP	11 hours x 12€ =132€
	4	First implementation of Google Calendar API	AS	12 hours x 15€ =180€
3	**1**	Coding the subject's menu view	BR, AT	AT: 6 hours x 15€=90€ BR: 6 hours x 12€=72€
	2	Coding the teacher's information view	BR, AT	AT: 18 hours x 15€=270€ BR: 18 hours x 12€=216€
	3	Coding the date selection for the meeting view	BR, AT	AT: 25 hours x 15€=375€ BR: 25 hours x 12€=300€
4	**1**	Setting up Amazon Web Services	AS	3 hours x 15€ = 45€
	2	Generating information from API provided by intranet	AS, HP	AS: 11 hours x 15€=165€ HP: 11 hours x 12€=132€
	3	Research and creating the MySQL database on RDS	AS	3 hours x 15€ = 45€
	4	Coding the function GET to show all the information in the APP	AS, HP	AS: 20 hours x 15€=300€ HP: 20 hours x 12€=240€
	5	Coding a POST function to update the database with the meeting scheduled	AS, HP	AS: 22 hours x 15€=330€ HP: 22 hours x 12€=264€
	6	Adding to the POST function to also create the event in Google Calendar	AS	19 hours x 15€=285€
	7	Testing and adjusting the functions	AS, HP	AS: 18 hours x 15€=270€ HP: 18 hours x 12€=216€
5	**1**	Understanding and adapting the sandbox provided by iThinkUPC	AS,HP,BR, AT	AS, AT: 37 hours x 15€ = 555€ BR, HP: 37 hours x 12€ =444€

	2	Adapt GET function to receive the data as the frontend needs	AS,HP,BR, AT	AS, AT: 15 hours x 15€ =225€ BR, HP: 15 hours x 12€ = 180€
	3	Adapt update function to create the event properly	AS,HP,BR, AT	AS, AT: 23 hours x 15€ = 345€ BR, HP: 23 hours x 12€ =276€
	4	Check the correct behaviour and correcting mistakes	AS,HP,BR, AT	AS, AT: 13 hours x 15€ = 195€ BR, HP: 13 hours x 12€ =156€
	5	Making the code more efficient	AS,HP,BR, AT	AS, AT: 12 hours x 15€ = 180€ BR, HP: 12 hours x 12€ =144€
	6	Generate APK and checking the correct behaviour in the mobile app	AS,HP,BR, AT	AS, AT: 13 hours x 15€ = 195€ BR, HP: 13 hours x 12€ =156€

Facilities, equipment and licenses:

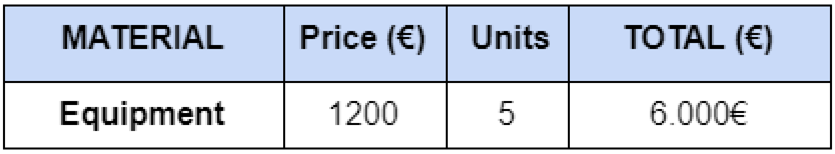
- **Facilities:** We are for rent in a local with a coworking space and adapted with all our needs such as tables, chairs and blackboards. We have done an estimated cost of that part, and also for the supplies (water, energy) and Internet services costs during these weeks of work:



UTILITIES	Cost/week (€)	Number of weeks	TOTAL (€)
Local	200,00 €	15	3000,00 €
Supplies	15,00 €	15	225,00 €
Internet services	25,00 €	15	375,00 €

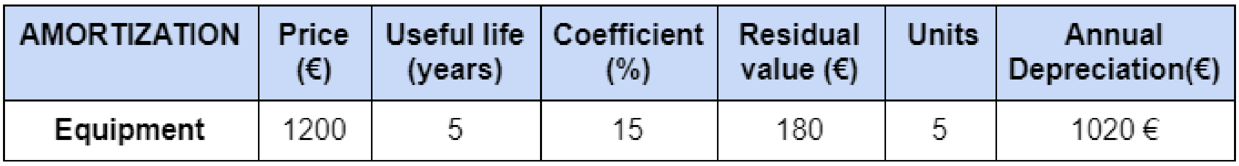
TOTAL UTILITIES: 3.600 €

- **Equipment:** We've only needed 5 powerful computers, one for each worker, to be autonomous in our work and to be able to work correctly:



MATERIAL	Price (€)	Units	TOTAL (€)
Equipment	1200	5	6.000€

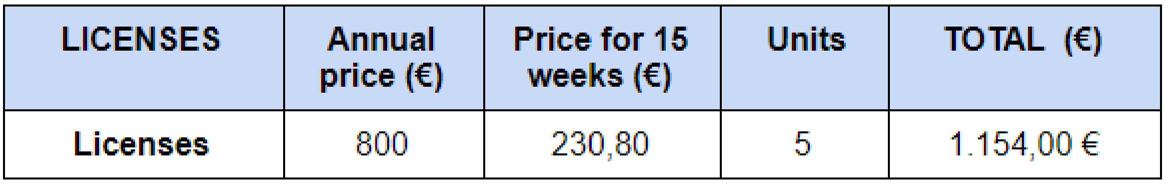
This will be an investment for the project, so we need to take into account the amortization of this material, because it will determine the cost of our investment during the project. We only have to calculate the amortization of our computers during the project, with the 15% of the amortization coefficient and 5 years of useful life. So once we make this calculation, we have obtained the total depreciation of the material used:



AMORTIZATION	Price (€)	Useful life (years)	Coefficient (%)	Residual value (€)	Units	Annual Depreciation(€)
Equipment	1200	5	15	180	5	1020 €

TOTAL AMORTIZATION FOR 15 WEEKS: 294,23 €

- **<u>Licenses:</u>** Another cost will be the licenses that we need to carry out our project, and we need 5 of them, one for each computer. So in the table below you can see the cost of this during only 15 weeks of work:



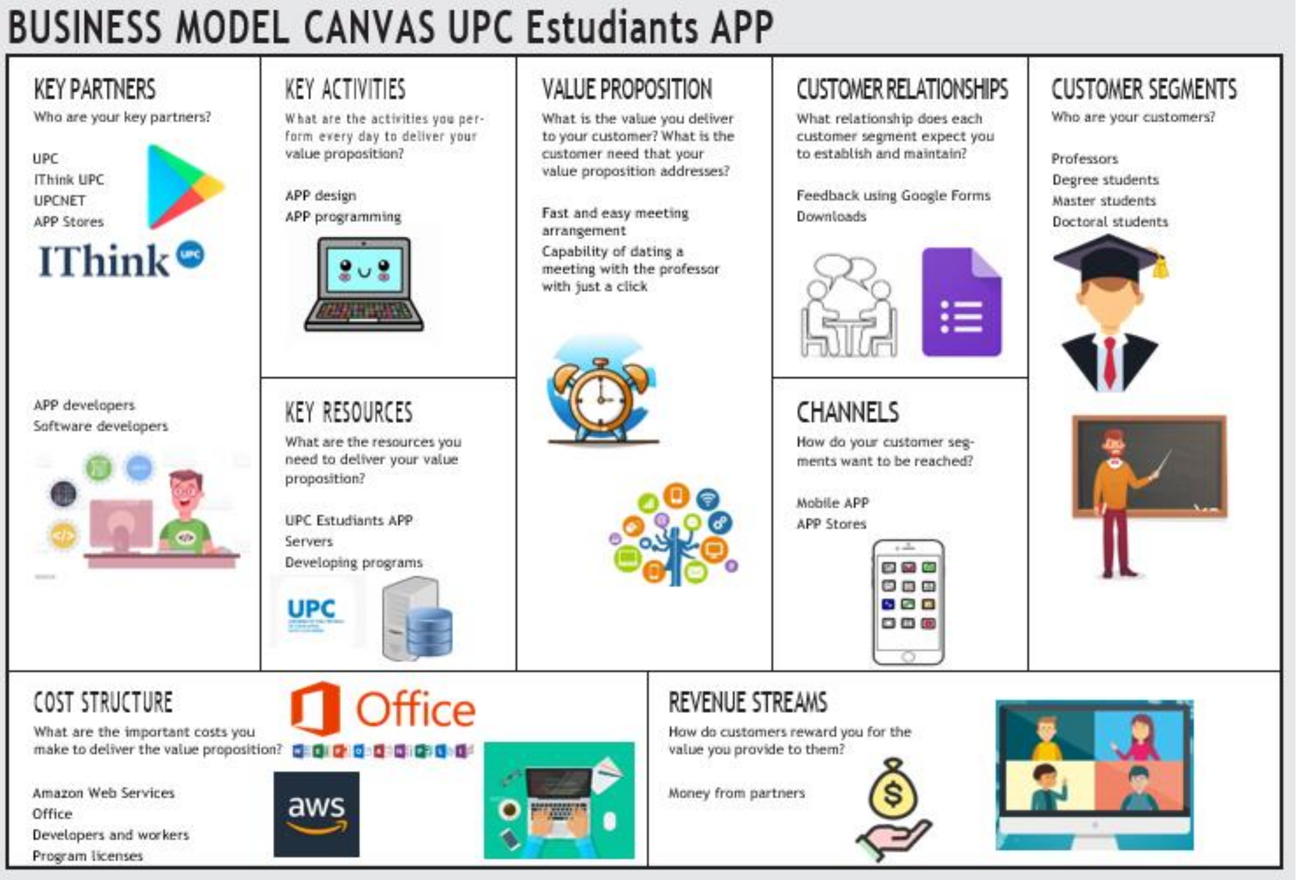
LICENSES	Annual price (€)	Price for 15 weeks (€)	Units	TOTAL (€)
Licenses	800	230,80	5	1.154,00 €

To sum up, we have included a table with all the costs explained before, and if we add them, we obtained the total costs of the project:

SALARY OF THE TEAM	30.186,00 €
LICENSES	1.154,00 €
AMORTIZATION	294,23 €
UTILITIES	3.600,00 €
TOTAL COST	**35.234,23 €**

TOTAL COST OF THE PROJECT: 35.234,23 €

9. BUSINESS MODEL CANVAS



BUSINESS MODEL CANVAS UPC Estudiants APP

KEY PARTNERS
Who are your key partners?

UPC
IThink UPC
UPCNET
APP Stores

APP developers
Software developers

KEY ACTIVITIES
What are the activities you perform every day to deliver your value proposition?

APP design
APP programming

KEY RESOURCES
What are the resources you need to deliver your value proposition?

UPC Estudiants APP
Servers
Developing programs

VALUE PROPOSITION
What is the value you deliver to your customer? What is the customer need that your value proposition addresses?

Fast and easy meeting arrangement
Capability of dating a meeting with the professor with just a click

CUSTOMER RELATIONSHIPS
What relationship does each customer segment expect you to establish and maintain?

Feedback using Google Forms
Downloads

CHANNELS
How do your customer segments want to be reached?

Mobile APP
APP Stores

CUSTOMER SEGMENTS
Who are your customers?

Professors
Degree students
Master students
Doctoral students

COST STRUCTURE
What are the important costs you make to deliver the value proposition?

Amazon Web Services
Office
Developers and workers
Program licenses

REVENUE STREAMS
How do customers reward you for the value you provide to them?

Money from partners

The value proposition of our product is to satisfy the need of scheduling an arrangement with a professor to solve questions. We are making a faster and easier way to agree on a date to make a meeting to solve questions about the subject.

Our customers are the teachers and the students from the Universitat Politècnica de Catalunya. The students can be studying a degree, a master's degree or a doctoral degree.

The main way of interaction with the customers is using the different APP stores to download the APP Estudiants UPC. We interacted with the customers before the project started by making a Google Form to know what the needs and the different implementations we could add. It would be good to make another form a few weeks after the implementation launch just to know what the customers think and what could be improved.

The resources we use are the APP UPC Estudiants where we made the implementation, the servers where we placed the database and the APIs, and the programs we used to design and develop our project. To make this new implementation we used Google Cloud, Google Calendar and Gmail too.

In order to deliver our value proposition, we designed and programmed the new module we wanted to add. Although it is finished, there are some improvements we could make .
Our partners are UPC, IThink UPC and UPCNET. The product is only for the UPC, so the competitors will be few, principally other companies involved in software and APP development.

The costs will fall back mainly in the servers from the Amazon Web Services, the licenses for the computers such as Microsoft Office, the office including the material and the infrastructure, and the developers and workers involved in the project.

Our product is only for UPC students and it is free to use, so we won't get money from the users. The money we get will come from IThink UPC, which is our main partner.

10. SUSTAINABILITY ANALYSIS

Like on every IT project, we have performed a sustainability report, considering the environmental, social and economic impact of our project.

In this section we will try to estimate our environmental impact by calculating our consumption during the design and useful life. It is difficult to be 100% exact on the measurements, but using different reference sources, we can approximate this impact.

This analysis is made knowing that it is impossible to be 100% sustainable, given the fact that in order to carry out any project, you will have to consume environmental and economic resources.

Furthermore, being our project an app targeted to students and teachers, we also need to take into account the social impact it may have during its life.

ENVIRONMENTAL IMPACT STUDY

- *CONSUMPTION:*

For the environmental impact study, we will start by measuring the overall consumption during the design project and the useful life of our app.

Design:

Computer usage → The computers are 35W each. We include 10W for networks and data centers = 45W* 19 hours /week* 15 weeks*5 computers/1000= 64,125 kWh

Electricity usage→ Electricity use on the class (lights)= 28W/lightbulb*19 hours/week*15 weeks*6 light bulbs/1000= 47,880kWh

Amazon Web Service→ Our backend is currently on Amazon Web Service, so we need to keep this into account. A server on average is of 427= 427W*168 hours/week*7 weeks/1000= 502,152 kWh

Commuting→ All the members of the group use public transportation and spend an average of 2h a day commuting. We assume the train uses 1050W. Energy= 1050W/passenger *10 hours/week*15 weeks*5 passengers/1000= 787,5kWh

Activity	Consumption
Computer usage	64,125 kWh
Electricity usage	47,88 kWh
Amazon Web Service	502,152 kWh
Commuting	787,5 kWh
Total	**1.401,657 kWh**

Useful life:

Once the design process of our project is over, the consumption reduces significantly. Since it is an app, the use will be 100% from mobile phones and no computers will be used. We will no longer have electricity usage of the local we were using since the working will be over, but we will still have the Amazon Web Service.

Phone usage→ We will assume the phones are an average of 3W.. We include 10W for networks and data centers. We speculate our implementation will be working for 2 years, and then it will be updated. We expect having 2000 users who use the app 1 hour/week = 13W* 1 hour /week* 104 weeks*2000 of users/1000= 2704 kWh

Amazon Web Service→ 427W*168 hours/week*104 weeks/1000= 7460,544kWh

Activity	Consumption
Phone usage	2704 kWh
Amazon Web Service	7460,544 kWh
Total	**10.164,544 kWh**

As we have seen, the total consumption of our project will be of 642,407kWh+10.164,544kWh= **10.806,951 kWh**.

- *SOLUTIONS TO IMPROVE:*

Since our project is a mobile app, it will not consume as much as a "physical" project, because we do not use any materials outside of the Computers.

In order to have our backend running more efficiently, we decided to use the Amazon Web Service. One of the reasons we kept in mind when deciding it, is that they are committed to running the business the most environmentally friendly way possible, and they already have large surfaces of solar and wind farms around the globe, so more than 50% of the energy used comes from renewable sources.

Another thing we could do in order to be more environmentally friendly, is once the computers no longer work as we need them to, we could donate them to Labdoo, this way we will be contributing to a circular economy, avoiding to send those computer to landfills when they are still usable, and giving them to people in need but without any other means to get one.

Finally, in the long run, we could slowly introduce the option to work from home. This will allow us to cut back on the consumption of having to commute everyday from home to the local.

- *ENVIRONMENTAL RISKS:*

Since all the consumption from the useful life of the app comes from user usage and the Amazon Web Service, not many situations can occur that could increase the carbon footprint of our project. This would only happen if any of this scenarios happens:

- More students than average want a meeting with a professor so they use the app to schedule it. This could happen, for example, if another lockdown occurs, where all the classes are performed online and the students find it more difficult to follow and understand. We also have to keep in mind that the number of meetings scheduled through the app is going to increase the closer we are to the exams.
- Something happens to the solar or wind farms of AWS so we won't be able to use that percentage of renewable energy.
- If Amazon Web Services changes its environmental policy, we would have to look for alternatives.
- If our computers have a lower useful life than expected, we would need to buy new ones.

ECONOMIC IMPACT STUDY

- *COSTS:*

While calculating the costs, we have kept in mind each worker's wage, with all the corresponding social charges, and all the material resources used to achieve our project. In our case, those have been computers, electricity supplies and local. We have also kept in mind the use of AWS and other licenses we have needed during the process.

- *ECONOMIC SOLUTIONS:*

After a thorough analysis of the Estudiant UPC app, we realized that it did not provide any implementation related to the one we wanted to work on, so it would be a completely new addition.

As for the economic part, we depend on IThink UPC, but for our section, we are using our own backend system, currently running on AWS. By using that system, we are able to significantly reduce the costs of having your own servers and machines running independently.

One of the most common ways of making money with an application is by monetising it, including advertising or micropayments. This solution could make the app more economically sustainable, but we would have to find a way of doing so, without compromising the final user experience.

Some options could be to include a recommendations section that introduces advertisements of events, restaurants and places of interests for students. This way we could receive extra capital from these advertisers, but then we would need to follow a strict policy of which ads to include and which not, so as to keep the whole app aligned with our beliefs and values.

- *USEFUL LIFE REPAIRS:*

In the future costs, we will keep in mind all kinds of updates and modifications that will be added to the app during its useful life. We will also consider future expenses of software licenses and additional tools that may require a monetary outlay.

We also expect this project to be picked up by the Estudiants UPC developers and introduced to the preexisting app.

- *RISKS AND ADDITIONAL COSTS:*

Being a software project, we are very unlikely to find unexpected additional costs down the road.If we got to the point of dismantling, it would not incur any more costs, due to the fact that we would only need to unsubscribe from AWS. As for the material resources, we could always try to sell the remaining computers.

One economic risk that could affect the viability of the implementation, would be if IThink UPC does not pick up this project and sponsor it. Being the app free of charge, we depend on the part of capital that would be assigned to us in the budget for the development of the app.

SOCIAL IMPACT STUDY

- *PERSONAL IMPACT:*

The personal impact that this project has had on the members of the group, are not regarding the users of the app, but the people who constantly work as App Developers.

We have realized how tedious the whole process of a simple implementation can be, so we cannot imagine how it would be to do an entire functioning app.

At the beginning of the project we could not have known how something as easy to explain as three clicks, is not as easy to actually code. It is not only having the idea of what to do, but also how to do it. How we separate each function from the backend, middleware and frontend, and how we can make them communicate between them.

- *SOCIAL IMPACT:*

The group of people affected by our project are mainly going to be UPC students and teachers. They are the ones that will be able to schedule a meeting through the app. A much faster and efficient method. But at the same time, they will have to get used to it and change their actual method of just sending emails asking for a meeting time.

- *INCLUSIVITY:*

Our app can easily be used by most people with special needs. Those who currently are not able to correctly use our app are people with visual impairment. In order for the app to be accessible to them, there are a few options we could add:

- Sound description of what is displayed on the screen.
- Sound effects and/or vibration of the phone when an action is performed (change screen, a button is pressed, a popover appears…)

We have also kept a neutral gender through the whole of our implementation so as not to use the masculine as a general (estudiantat, professorat...).

The privacy aspect of the user is not something we have had to manage, because all that data is given to us by the already existing app backend. However, we would need to keep the compromise of confidentiality with all the data we may get, and not giving this information to third-parties or lucrative actions.

As we have already said, the only way our project could exacerbate social inequality is because currently it is not adapted for the use of people with visual impairment. They can still ask for a meeting through mail and learn the teacher’s contact information through the “UPC Directori”, but they will not be able to do it through the app.

Even though, as we have already said before, Amazon Web Service is a company who is trying to slowly become more sustainable, over the years, it has been accused multiple times of exploiting its workers and creating a monopoly in the sector. By using their services, we are indirectly supporting this behaviour, which is one of the factors we need to improve if we want to become 100% socially sustainable.

The main risk that could be detrimental to the users of the app, is going to be if the Amazon Servers are down. In this case, our backend/middleware will not be reachable from the frontend, and the users will not be able to see any information on the screen of the app.

CONCLUSIONS

Even though our project is not 100% sustainable, we are on the right track trying to make it as sustainable as possible.

When looking at the 17 Sustainable Development goals (SDG), we can see that a few of them align with the ones we have for our project, but the most importants are:

- #5 Gender equality→ Two out of the 5 members of the group are females. During the whole process there have been equal opportunities for all of us while trying to keep an empowering environment.

- #8 Promote sustained, inclusive and sustainable economic growth, full and productive employment and decent work for all→ We have worked for everyone to have the opportunity to learn and grow during the whole process.

- #9 Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation→ We have had to be creative, especially at the beginning. We had to think of different possible implementations to add to the already existing app, and from all of those, choose which one was the more adequate, and how we could do it. So innovation has played a significant part.

- #12 Ensure sustainable consumption and production patterns→ As we have already said, even though we are not 100% sustainable, we are working on more ways to improve our carbon footprint.

11. CONCLUSIONS

In conclusion of this PAE, we can say that it has been an interesting project which has made us learn about technologies that we had never worked with before, besides being a project very applicable to a real working environment. We have used project management tools such as Trello and Slack, which have made our work more professional.

On the technical side, we have been able to get a fully functional version which could be added to the main app if IThink UPC would consider it, but in our version, the backend part is different from what they would use. Thanks to this project, the team members have been able to learn about Ionic, Node, MySQL and other programming languages and also, how to structure a client server application with calls to an API REST created with AWS.

On the business side, we have carried out a market study to find out what new functionality to include, taking into account user feedback. In addition, we have focused the new implementation on the different types of users that are going to use it, taking into account all their needs and a vision of our business idea through the canvas. Finally, a real simulation of all the costs necessary for the development of our work, which we consider to be very close to the reality of the costs of the development of a real software project.

The sustainability part has been useful to know the environmental, social and economic impact of an IT project. It is difficult to estimate the real consumption of electricity because there are no exact metrics of the use of the different technological equipment used, such as computers and the implementation of an external server, but even so we have extracted some consumption values that could serve as a real estimate. We have also determined what consequences and benefits to society our project can have, as well as the business idea and the necessary capital for it.

Concerning the scope, we consider that despite the difficulties encountered during the project, we have managed to meet the main objective, which was to achieve a new functionality useful for students. In the end, no more risks have appeared than those we had considered in the previous analysis.

12. REFLECTION DOCUMENT

As a final reflection, looking at the work we have done as a team, we are very proud of it. Taking into account we were such a small group, we were able to overcome all the challenges that we met. There are not many things to improve, but being a bit self-critical, there have been moments when the communication between the frontend and the backend has not been clear enough, creating situations of confusion, which we solved immediately.

Apart from that, we think we always have worked in a good environment, we have been constant during these weeks, and we learnt a lot of new things, and we improved anothers. We are glad to spend all these weeks working together.

Regarding the relationship between us and the enterprise, we would also like to add that we have felt a bit lonely during the project. We have not had much feedback from the company, and not much help to carry it out. We believe that our experience and learning would have been even greater if we had had more contact and motivation from IThinkUPC since the beginning, and they had more interest in finding an effective improvement for the App they work with.

During all the months of development we only had one meeting with a person in charge of IThink UPC, making it difficult at certain moments to work due to the lack of the necessary support. In the only meeting we had with the person in charge of IThink UPC we were told that they would send us a test version of the app for us to work with, but it took longer than expected. This meant that all the code of the directory frontend that we had already been working on, had to be modified in order to merge correctly with the rest of the app a few weeks before the end of the course.

Another thing that could have helped us a lot is knowing how the SandBox provided by the company worked. We would have liked to receive some kind of documentation or comment on what it contains and how everything is organized. Without it, we had to spend a lot of time trying to figure it out and trying to understand the code ourselves.

We believe that with increased participation, we could have worked together to develop a version of our project that could be added to the existing application immediately. But instead, we get the feeling that once we got the Sandbox, that was it. It was our turn to overcome all the complications that we have faced along the way, and we hope that everything will work out in the end.

We also do not know if our contribution to this project can finally be added in the application, or if they would like to work more on it in the future. We hope our contribution will not be forgotten, because it can make a difference and help students use the app and make it more useful for everyone.

On a more technical side, as future improvements we could:

- create scripts that run every certain period of time to update the information used
- create the event directly from the email accounts of the people involved without having to use an administration account. This would allow the student to receive a notification once the professor has confirmed the attendance.
- display the photo of the teachers to make the functionality a little more visual.
- add a message section to make the student able to write the reasons he/she has to schedule the meeting.

Finally, we want to thank all the support provided by our supervisors, Josep Pegueroles and Saúl García, who have helped us to achieve a good final result of the project and those who have accompanied us during all these weeks, giving us advice and from which we have learned a lot.