

DAWBIO M12

PROYECTO DE FINAL DE CURSO



Submission date
World With Clear Ari
W.W.C.A

Albert Casany
Iván de Montserrat
Roger Puga Ruiz

Table of Contents

RA1. Introduction and Context.....	5
1. Bioinformatics Market.....	5
2. Business opportunities.....	5
3. Project description.....	5
Introduction.....	5
Bibliography.....	5
Problem you want to address.....	5
Data sources.....	5
Data structure.....	5
Actors.....	7
Goals.....	7
Technologies (Java, PHP, MariaDB, Angular, Biopython, R, etc.).....	7
Mockup of the main page.....	8
4 Occupational Safety and Health.....	8
RA2. Use Cases, Requirements, Mockups, Resources, Cost Estimation.....	9
1. Use Cases, Functional Requirements, Non-Functional Requirements.....	9
2. Mockups and Diagrams.....	13
3. Resources and Cost Estimation.....	20
RA3. Planning.....	21
1. Project Planning.....	21
2. Risk Mitigation Planning.....	22
RA4. Tests.....	23
1. Unit/Integration testing.....	23
2. Usability/Accessibility testing.....	23

Project Guide (v10)

This template is intended as a guide for writing the documentation of the final project.

1. About this template:

- It is a LibreOffice Writer document that you can use directly or convert to another format.
- However you use it, you **MUST** follow the structure in this document.
- The Table of Contents must be exactly the same and in the same order.
- Keep the same format (fonts, sizes, styles). No changes allowed.
- You must remove all text other than the headings and the table of contents.
- If you have difficulties applying this template to your project, discuss it in class.

2. Evaluation:

Your project will be evaluated according to the following table.

	Docs	Code	Total
RA1: Intro	5		5
RA2: FRs	15		15
RA3: Plan	5		5
RA4: Tests	5		5
RA5: Code		70	70
Total	30	70	100

2.1 Documentation:

- The documentation (“Docs” column) refers to this document.
- Be sure to include your group name and full name of the members.
- This document must be written in English.
- The documentation is evaluated globally. All team members get the same grade.
- RA2 distributes the work among the team members. It is essential that all team members agree to it as soon as possible.
- You must submit the distribution of the functional requirements in a separate .ods file, together with this document. Refer to “dawbio2-m12-prj-requirements.ods” for details.

2.2 Code:

- The code (“Code” column) is evaluated individually for each team member (RA5).
 - Different members can get different grades.
 - Different members may have to do different workloads to obtain the same grade.
 - The workload is decided in section RA2.
 - If your RA5 grade is less than 5 you automatically fail your project.
 - Additional considerations:
 - It must include docstrings (for functions and at the top of each file).
 - It must be formatted according to a standard convention.
 - Variables and functions must have sensible names.
 - It must include tests.
 - You have to submit your git repository with the full history.
 - It has to be deployed and run from the ProvenApp server.
- If not possible, discuss it in advance. (Dual, etc.)

3. Presentation:

The presentation helps reviewers to better understand the submitted documentation and code. Presentations will be split in two parts:

1. Video presentation: Each member must submit a short video explaining his/her code.
2. Questions and Answers session: Each member will answer questions from the reviewers.

Each member must submit a video and attend the Q&A session.

Failing to do so means automatically failing the project.

3.1 Video:

Each team member must make a video explaining his/her part of the project.

- Video length: Between 10 and 15 minutes long.
- Each member must upload it to his own Google/Microsoft/etc. account.
- Each member must generate a shareable url and paste it in a text file called "video.txt".

You do not need to prepare any powerpoint slide.

Use only your documentation, your application and its source code.

The video must follow this structure:

1. Introduction: (15 secs)

Say your name, DNI, team and project title.

2. Use cases and Functional Requirements (FRs): (15 secs)

Show the FRs you implemented and say what use cases they relate to.

Use the diagram and tables you put in your documentation.

3. Demo: (Between 4 and 7 minutes long)

For each use case in the previous section, do a demonstration of your program working.

Show that your program validates wrong inputs and is robust against errors.

4. Code: (The remaining time. Remember that total time must be between 10 and 15 minutes)

For each FR you implemented:

- Do a tour of the code in the same order as it is executed.
- Comment only the most important parts of the code.
- Explain what decisions you made when writing the code.
- Introduce briefly any external library you use.
- Skip any code that is not yours.

Say explicitly where to find the code you are commenting.

You can highlight parts of the code with the mouse as you comment them.

Make sure that fonts are big and readable in the video.

3.2 Q&A Session:

- It will be done using Google Meet.
- Each team member must participate.
- Each team member must have his/her program ready to be run and modified upon request.
- The schedule and Google Meet links will be uploaded to moodle after the submission.

4. What to submit:

1. This document as an .odt file.
2. This document as a .pdf file.
3. The list of requirements as an .ods file.
4. A .zip file containing: all data, the final code and the git repository with the whole history.
5. A text file named “video.txt” containing the urls for downloading your video presentations.

All these files must be submitted to the moodle task.

In total, they cannot exceed 100 Mbs. That is the maximum allowed size by moodle.

Videos can exceed 100 Mbs, but you are encouraged to keep them small. If videos are small enough to fit in the moodle task, you can upload them directly.

5. Submission:

- If you miss the submission deadline, you will lose one point in each RA for every hour you are late.
- If you forget to submit any of the above files, you will be penalized accordingly.
You may fail your project automatically depending on the severity.

ALWAYS delete all explanations in this guide before submitting. Good luck!

RA1. Introduction and Context

1. Bioinformatics Market

Our project focuses on the pollution of planet earth, it is a current issue and related to life sciences. In this sector we find many world organizations such as Europe, the United States or the UN, which provide a lot of current data in real time and facilitate the creation of APIs aimed at controlling emissions

2. Business opportunities

We have detected that there is a lot of data on pollution but they are not clear, and they are not grouped either and make it difficult to use them. Our business opportunity is to be one of the few companies that uses this data to clearly show a trend and its consequences.

3. Project description

Introduction

This project is based on a web that will dynamically display a map with the respective pollution, we intend to clearly show the difference in pollution over time and health problems.

We also propose to create a communication area between researchers and science enthusiasts who want to contribute their knowledge.

Bibliography

Problem you want to address

Currently there is a lot of information on pollution and we intend to use official data from public organizations to visually show what they represent.

Data sources

<https://discomap.eea.europa.eu/App/AirQualityStatistics/index.html>

Data structure

Columns DataSet

- Country (String) - Nombre del país o territorio.
- Air Quality Network (String) - Identificador de la red de medición de la calidad del aire, proporcionado por el proveedor de datos.

- Air Quality Network Name (String) - Nombre de la red de medición de la calidad del aire, proporcionado por el proveedor de datos.
- Air Quality Station EoI Code (String) - Código EoI de la estación de medición de la calidad del aire (como en AirBase).
- Air Quality Station Name (String) - Nombre de la estación de medición de la calidad del aire (como en AirBase), proporcionado por el proveedor de datos.
- Sampling Point Id (String) - Identificador del punto de muestreo, proporcionado por el proveedor de datos.
- Air Pollutant (String) - Sustancia que contamina el aire, cuyo nivel se mide y notifica a la AEMA (consulte la notación en el Diccionario de datos: <http://dd.eionet.europa.eu/vocabulary/aq/pollutant>).
- Air Pollutant Description (String) - Descripción de la sustancia que contamina el aire.
- Data Aggregation Process Id (String) - Identificador de información sobre el proceso de agregación de datos en valores que no sean por hora (consulte el Diccionario de datos: <http://dd.eionet.europa.eu/vocabulary/aq/aggregationprocess>).
- Data Aggregation Process (String) - Información sobre el proceso de agregación de datos en valores que no sean por hora (consulte el Diccionario de datos: <http://dd.eionet.europa.eu/vocabulary/aq/aggregationprocess>).
- Year (Numeric) - Año para el cual se informaron los datos primarios (y se calcularon las estadísticas anuales).
- Air Pollution Level (Numeric) - Concentración o nivel de sustancia contaminante del aire, aquí dada como una agregación de valores de concentración de contaminantes del aire de la serie de tiempo de observación primaria.
- Unit Of Air Pollution Level (String) - Unidad de concentración o nivel de sustancia contaminante del aire (ver en Diccionario de datos: <http://dd.eionet.europa.eu/vocabulary/uom/concentration>).
- Data Coverage (Numeric) - Proporción de medición válida incluida en el proceso de agregación dentro del período de promediación, expresada como porcentaje. Si la Cobertura de datos < 75 % durante un período promedio de un año, las estadísticas anuales no deben incluirse en las evaluaciones de la calidad del aire; si la Cobertura de datos < 85 % (en un año), las estadísticas anuales no deben incluirse en las verificaciones de cumplimiento.
- Verification (Numeric) - Información basada en indicadores de verificación encontrados en series temporales notificadas (ver en Diccionario de datos: <http://dd.eionet.europa.eu/vocabulary/aq/observationverification>).
- Air Quality Station Type (String) - Tipo de estación de medición de la calidad del aire: información sobre si está midiendo la contaminación del aire de fondo, industrial o relacionada con el tráfico (consulte el Diccionario de datos: <http://dd.eionet.europa.eu/vocabulary/aq/stationclassification>).
- Air Quality Station Area (String) - Área de la estación de medición de la calidad del aire: información sobre si está midiendo la contaminación del aire en un entorno urbano, suburbano, rural (etc.) (consulte el Diccionario de datos: <http://dd.eionet.europa.eu/vocabulary/aq/areaclassification>).
- Longitude (Numeric) - Longitud de la estación de medición de la calidad del aire [grados decimales].
- Latitude (Numeric) - Latitud de la estación de medición de la calidad del aire [grados decimales].
- Altitude (Numeric) - Altitud de la estación de medición de la calidad del aire [m s.n.m.].
- City (String) - Nombre de la ciudad.
- City Code (String) - Identificador de la ciudad.
- City Population (Numeric) - Población de la ciudad.
- Source Of Data Flow (String) - Fuente de la información.
- Calculation Time (String) - Tiempo de calculo.

- Link to raw data (only E1a/validated data from AQ e-Reporting) (String) - Link de la información, solo para casos restringidos.

Actors

Users: login and logout












Admin: admin the page

Investigator: create news

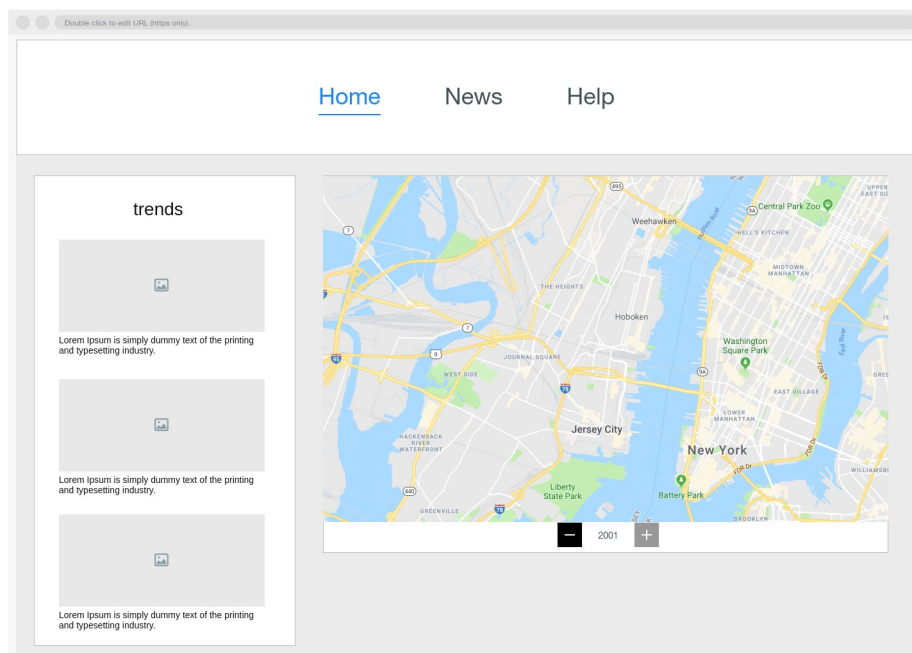
Goals

Create a large community to get as many people as possible to solve pollution problems

Technologies (Java, PHP, MariaDB, Angular, Biopython, R, etc.)

- Java (spring-boot)  
- Typescript (angular)  
- Javascript (d3) 
- Python (flask)  
- Json 
- JWT 
- CSV 
- MySQL 

Mockup of the main page



4 Occupational Safety and Health

1. Commuting accident: Do not exceed the speed limit and try to be attentive.
2. falls: signal that a floor is slippery and not run through the corridors.
3. falling objects: do not leave heavy objects at a great height
4. electrical contacts: keep wires unpeeled and out of contact.
5. cuts and punctures: Avoid sharp areas or objects
6. Mental fatigue: rest your eyes and mind every 30 minutes it is advisable to have a window where you can see the horizon
7. Postural fatigue: avoid having a position for a long time and alternate between sitting and standing, use a chair that adapts to the contour of the back.
8. Bumps or bumps against stationary objects: Avoid objects in the corridor and if it is not possible to properly signal
9. Fire: Inform about the measures to be taken in case of fire and know the emergency exits avoiding panic.
10. Overexertion: Knowing your limits and not wanting to cover more than necessary, it is important to properly manage the work.

RA2. Use Cases, Requirements, Mockups, Resources, Cost Estimation

RA2 is the most important among all RAs. It helps reviewers understand the project and will be the main reference when evaluating each member's code.

You must follow this order:

1. **Use cases:** Gather the Use Cases from the point of view of the users. Make diagrams and explain briefly each use case with a few lines.
2. **Functional Requirements:** List the functionality of your application from the point of view of the implementation. The functionality must cover all use cases. E.g: The application can create, read, update and delete sequences. The application can align two sequences. The application can authenticate users and grant permissions according to their roles. Etc.
3. **Non-Functional Requirements:** Very brief list of the non-functional requirements that your program may have.
4. **Mockups:** Paste here the mockups of your application. You can substitute them for the real views once you have implemented them. They must cover all the use cases.
5. **Entity-Relationship Diagram:** Draw the Entity-Relationship (ER) model of your application based on the use cases. Focus on all the data that will be stored in your application. The ER Diagram must cover all Use Cases. This will be the design of your database. In case of not having a database, describe in detail the data sources and format.
6. **Class Diagram:** Draw the Class Diagram of your application based on the functional requirements. Focus on where to place the functions of your application. The Class Diagram must cover all Functional Requirements. This will be the design of your code.

Do not spare details here. Write as many pages as you need.

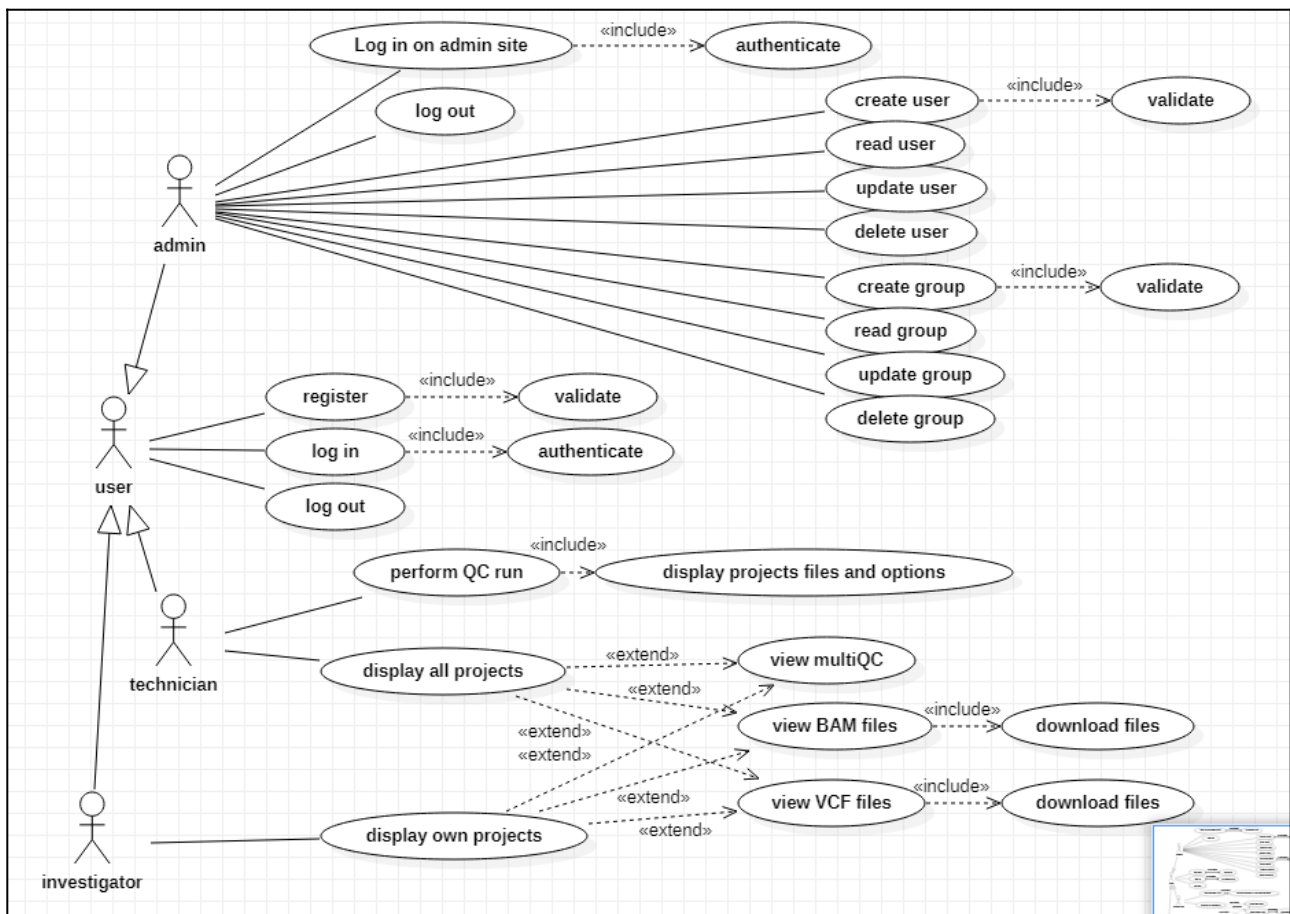
1. Use Cases, Functional Requirements, Non-Functional Requirements

Make a list of use cases for the application as explained in “M05 Entorns de desenvolupament”. Based on that, write functional and non-functional requirements.

Functional requirements must be distributed among team members. Each student will have its own list of functional requirements that he/she will implement. Requirements cannot be shared among team members.

Each team member must assign a score to each functional requirement summarizing its workload. The sum of all scores must be 10. This will be used to evaluate the code of each student individually.

Example: Use Cases from David Medel's M12 Project.



This is a detailed example.

To the left there are the actors (admin, user, technician, investigator) and to the right their use cases. Some use cases are repeated among different actors (registration, log in/out), but here are shown in a compact form.

Once you have gathered all use cases, you must rewrite this information as a list of functional requirements. The list of functional requirements must be non-overlapping and will be used to distribute the work among the team members. See the example in the next page.

Additionally, you can have a list of non-functional requirements that can be important for the project as a whole.

Example: Functional Requirements

Student X (66 hours)		
ID	Functional Requirements	Workload
FR1	The frontend automatically validates customer inputs	2
FR2	The frontend generates graphs programmatically from the json the backend sends	3
FR3	The software system must be integrated with the Google Maps API	5
...		
...		
...		
...		
...		
Total:		10

Student Y (245 hours)		
ID	Functional Requirements	Workload
FR7	The backend automatically validates customers against the database	1,5
FR8	The warning system sends emails automatically to patients with a high risk of disease	2
FR9	Only doctors can see the patient's medical history	1,5
FR10	A script updates every day the local database from the NCBI automatically	3
FR11	The software allows users to change the taxonomy in the configuration settings	2
...		
...		
...		
Total:		10

Make the above tables in a separate spreadsheet (.ods file), paste them in this document and submit the .ods together with this file.

Beware that different team members may have to work more or less hours depending on the type of project they are doing.

These are the three kind of projects and their expected lengths:

- FCT: 66 hours
- Dual: 132 hours
- Integrated: 245 hours

Team members who have to do more hours will have a longer list of functional requirements. Still, the scores must sum 10 points.

Instructions:

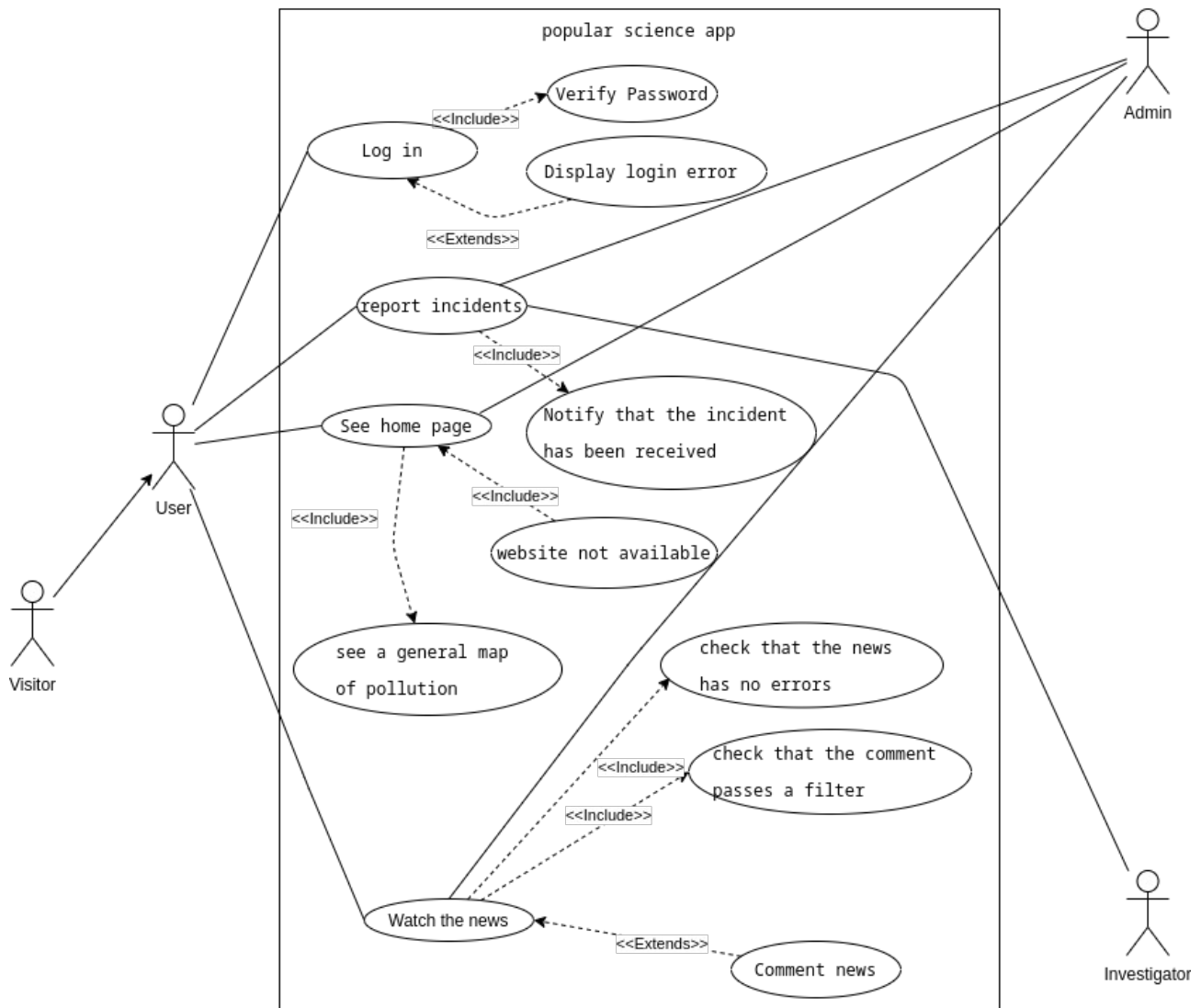
1. Write your team name, full names and project hours instead of «Student X (xx hours)».
2. Write as many functional requirements as you see fit.
3. Keep the current format.
4. Keep it all in a single spreadsheet.
5. Paste these lists in the main document (.odt)
6. Submit this file (.ods) together with the main document (.odt)

Team name: World With Clear Ari (W.W.C.A)

Roger Puga Ruiz		
ID	Functional Requirements	Workload
FR1	See the historical pollution map	3
FR2	Login, Logout and register	2
FR3	Show News	2
FR4	See relevants news	1
FR5	Validate User	1
FR6	Authenticate User	1
...		
...		
Total:		10

Iván de Montserrat		
ID	Functional Requirements	Workload
FR7	Comment News	1.5
FR8	Report wrong news	2
FR9	Read and write message	3
FR10	Create News	1.5
FR11	Comment News	2
...		
...		
...		
Total:		10

Albert Casany		
ID	Functional Requirements	Workload
FR12	Admin can create new user	1
FR13	Admin can Read users	1.5
FR14	Admin can Update users	1.5
FR15	Admin can Delete users	1.5
FR16	Admin can Delete news	1.5
FR17	Admin can Change the configuration page	3
...		
...		
Total:		10



Administrators and researchers are a type of user, administrators can manage the inner workings of the page and make sure it works properly. Researchers have to create new news related to pollution. Finally, users can register and log in to see the content of the news, see the pollution map, read and write messages and report incidents.

2. Mockups and Diagrams

Paste here the following:

1. Mockups of the views.
2. Entity-Relationship diagram.
3. Class diagram.

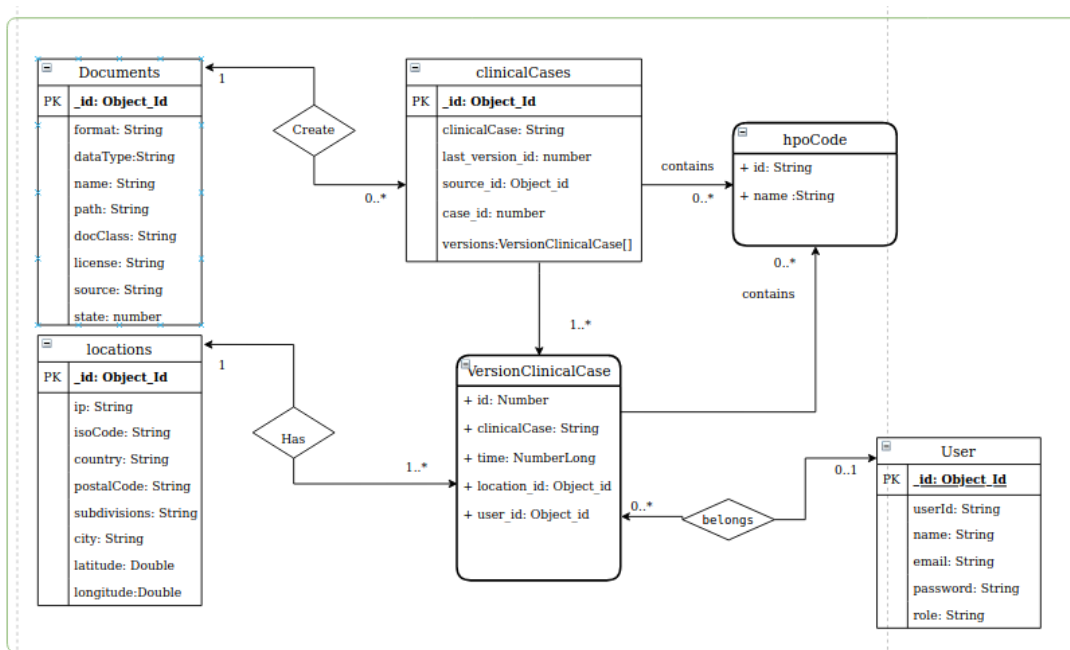
Each mockup and diagram must be followed by a short description explaining the design.
If any of the above points does not apply to your project, explain why (e.g. No database, etc.)

Example: Mockup from David Medel's M12 Project. (a wireframe drawing is ok).

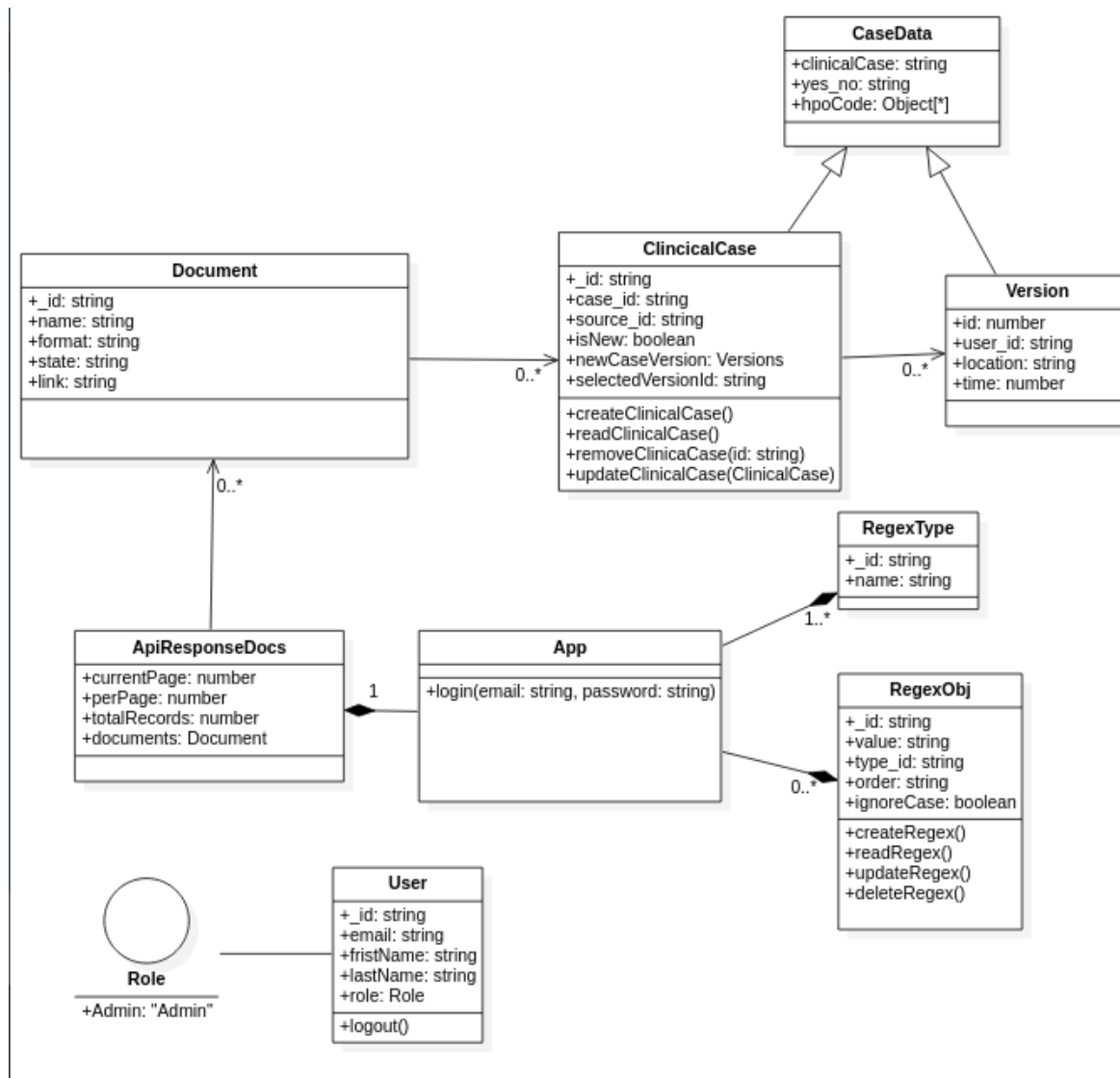
The mockup shows a web browser window with the address bar displaying '127.0.0.1:8000/usermanagem...'. The main content is a 'REGISTER ACCOUNT' form. The form includes the following elements:

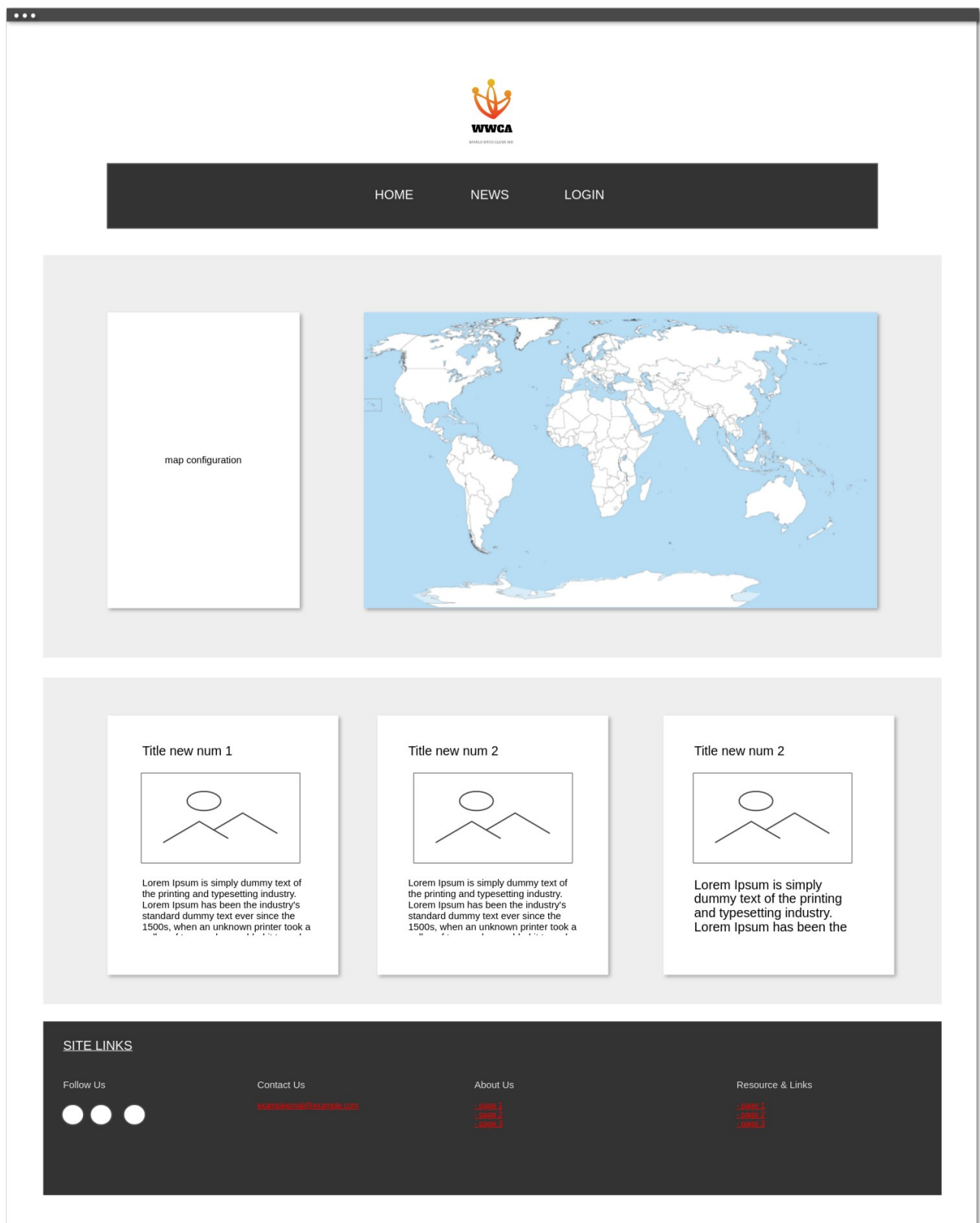
- A title 'REGISTER ACCOUNT' at the top.
- Input fields with orange icons: 'Username...' (person icon), 'First name...' (person icon), 'Last name...' (person icon), 'Email...' (envelope icon), 'Enter password...' (key icon), and 'Re-enter Password...' (key icon).
- A blue 'Register Account' button.
- A link 'Already have an account? Login' at the bottom.

Example: Entity-Relationship diagram from Ankush Rana's M12 Project.



Example: Incomplete class diagram from Ankush Rana's M12 Project.

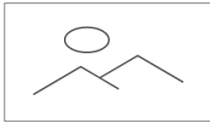




main view or home, we can see the pollution map and some relevant news.

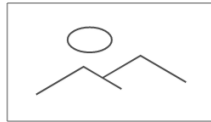
[HOME](#)[NEWS](#)[LOGIN](#)

Title new num 1



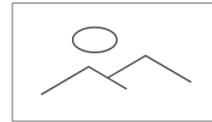
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a

Title new num 2



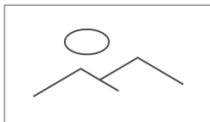
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a

Title new num 2



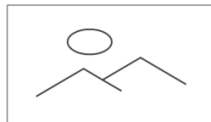
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the

Title new num 1



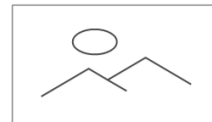
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a

Title new num 2



Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a

Title new num 2



Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the

SITE LINKS

Follow Us



Contact Us

example@example.com

About Us

[0000.1](#)
[0000.2](#)
[0000.3](#)

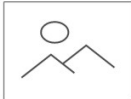
Resource & Links

[0000.1](#)
[0000.2](#)
[0000.3](#)

News view, where we can see the list of all news

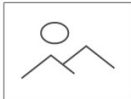
[HOME](#)[NEWS](#)[LOGIN](#)

Title new num 1



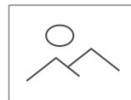
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and

Title new num 1



Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and

Title new num 1

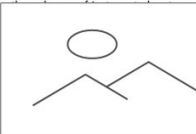


Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and

Title new num 1

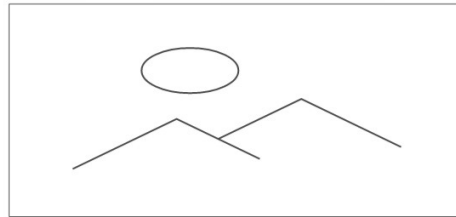
SubTitle new num 1

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with



Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially

unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.




Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

SITE LINKS

[Follow Us](#)[Contact Us](#)www@wwca.com[About Us](#)[0800 1 0800 2 0800 3](#)[Resource & Links](#)[0800 1 0800 2 0800 3](#)

View for a selected news, where we can see the selected news and three other news.



WWCA
WORLD WIDE COUNCIL OF ARTISTS

HOME NEWS LOGIN




LOGIN

NAME

PASSWORD

SITE LINKS

Follow Us



Contact Us

example@sample.com

About Us

[+0998-1](#)
[+0998-2](#)
[+0998-3](#)

Resource & Links

[+0998-1](#)
[+0998-2](#)
[+0998-3](#)

Login view, where to log in

3. Resources and Cost Estimation

List all the resources needed to complete your project and their cost.

Calculate the total cost of your project considering one month of work for the whole team.

- For the salaries of the team members, you can look at <https://www.linkedin.com/jobs/> .
- For the cost of computers, have a look at <https://www.pccomponentes.com/> .

A short text description and a link to an assembled model is enough.

Do not put photos or long descriptions of components.

RA3. Planning

There is no limit of pages for RA3, but be concise. No need to be extremely detailed.

1. Project Planning

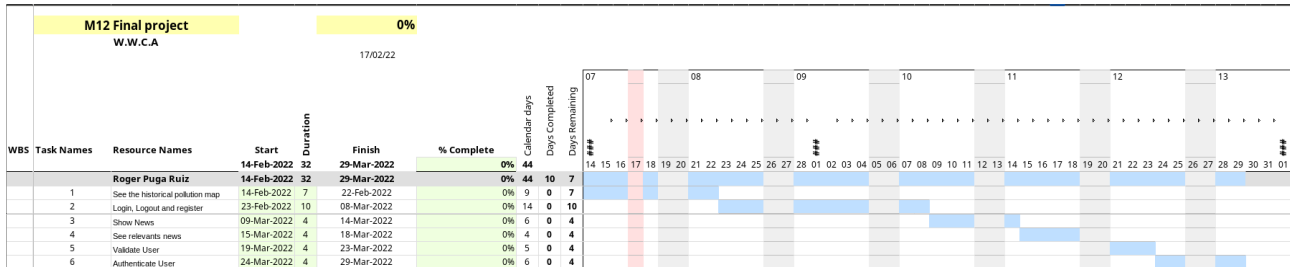
Pick a development methodology, plan and document it accordingly, as explained in “M05 Entorns de desenvolupament”.

- Each team member must have its own plan.
- Include the initial planning and the actual development until finishing the project.
- In the end, for each plan, do a “Planned vs Actual development” comparison highlighting what went right and what went wrong. (Two paragraphs approximately)

Example: Classical waterfall development planning in Esther Vendrell’s M12 Project.

			MAIG 2020																																																	
		total h.	missing h.	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
1- Create main interface to list workflows		4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
2 - Create the Datatable to list all the workflows	Esther	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Esther	10	0	0	5	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
3 - Design the pseudo API	Esther	10	0	0	0	0	5	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Esther	10	0				5	5																																												
4 - Create the button reload and reload only the table		8	0	0	0	0	0	0	0	0	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Esther	8	0								4	4																																								
5 - Enable a select in the status of workflows to change it		11	0	0	0	0	0	0	0	0	0	0	5	5	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Esther	11	0										5	5	1																																					
6 - Create main interface to create new workflow		4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Esther	4	0															4																																		
7 - Create JSON Schema		20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Esther	20	0																5	5	5					5																										
8 - Implement JSON Schema in interface		7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Esther	7	0																								5	2																								
9 - Create enum list with ontologies		20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Esther	20	0																									5	5			5	5																			
10 - Autocomplete JSON Schema automatically (internally)		3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
	Esther	3	0																																																	
11 - Insert the new workflow into MongoDB validating all the fields		10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Esther	10	0																																																	
12 - Create nextflow		25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Esther	25	0																																																	

We have thought of using scrum to do the planning, in scrum teamwork, workflow and therefore productivity are improved. The problem is that it requires very good planning and a good understanding of the difficulty of the project. To make things easier for us, we will use a waterfall planning.



2. Risk Mitigation Planning

Describe the main risks of your project and what to do if trouble strikes.

The main risk is almost always unexpected delays due to bad analysis or bad planning. In that case, explain briefly what functional requirements to prioritize and what to discard.

Another risk to consider is what happens if one team member has to quit the project. Distribute tasks so that other members can still show their work in the final demo.

RA4. Tests

There is no limit of pages for RA4, but be concise.

1. Unit/Integration testing

Teachers will evaluate if there is a reasonable list of tests for the core functionality. Write the functional requirement (FR) ID for each test. When evaluating the code (RA5), reviewers will be looking for the actual tests. Write separate lists for each team member.

Example:

Let us suppose that we have a function called `create_doctor(dni)`. You could write the following test.

FR2: <code>create_doctor(String: dni)</code> → <code>Doctor</code>
1. Must throw an exception if DNI does not match pattern "8 digits + 1 letter".
2. Must throw an exception if DNI is already in database.
3. Must return a <code>Doctor</code> object with the same dni upon success.
4. After creating the doctor, <code>read_doctor(dni)</code> must return a doctor object with the same dni.

This example is in the .ods file. You can write your tests there and paste them here, if you want

Write tests only for the core/most important functions. Each team member should write 5 tests, approximately. Each test may have several cases, like in `create_doctor(dni)`. List all corner cases and test them (empty strings, zero-length arrays, wrong values, etc.)

Then, once you have written the tests, implement them using a unit testing framework:

- For Python you can use `pytest`.
- For Java you can use `JUnit`.
- For PHP you can use `PHPUnit`.

Or you can use other libraries if you prefer.

Ideally, you always write your tests first using a testing framework and then the code. If you are tight on time, at least write them on this document. Doing so will make your code less error-prone and easier to debug.

If your tests modify the database, remember to do it in a temporary database, not the main one. Most often you will need to clean up after the tests (E.g.: Delete the doctor you just inserted).

2. Usability/Accessibility testing

Describe here any usability or accessibility tests that you may have performed.

A well-thought usable/accessible interface usually implies more complex code. This difficulty will be reflected in higher scores in related functional requirements.

RA5. Code

There must be NO RA5 section in this document.
Delete this page before your final submission!

RA5 will be evaluated entirely from your code.

In addition to the instructions in section *About this template*, your code must include the following in order to help teachers evaluate you.

1. README.md

Place this file in the root directory of your project.

It must be written in markdown and contain the following:

1. A description of each directory in the source code.
2. The location of the main() function in your program.
3. How to compile your program.
4. How to install it.

If your project includes code from other people, write in detail what code is yours in this file.

This file is very important. Do not forget it.

2. HTML manual for administrators

Write an html file with a brief manual for each use case of the administrator. Include screenshots. Link it from the administrator's main view.

3. HTML manual for users

Write an html file with a brief manual for each use case of each user. Include screenshots. Link it from each user's main view.