Deliverable 2: Solution design +

Functionality preview

**Create a GitHub Wiki entry (NOT an attachment) with the items requested in this template**

All notes containing explanations and examples for filling out the template should be deleted and replaced by project information.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# 1. Architecture & Data

### **1.1. Deployment Diagram**

[Create a **Deployment Diagram** that illustrates the integration of various system elements, including software, hardware, and networks, to achieve the necessary functionality. Construct the diagram using draw.io, GenMyModel, or similar, ensuring strict adherence to the graphical notation standards of UML. For each node, **detail the technologies to be used**, including names and, if applicable, versions of those technologies.

Guidelines and examples:

<https://www.youtube.com/watch?v=o5kOHVZ08o0>

<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-deployment-diagram/>

<https://www.ibm.com/support/knowledgecenter/SS8PJ7_9.5.0/com.ibm.xtools.modeler.doc/topics/cdepd.html>

<https://creately.com/blog/diagrams/deployment-diagram-tutorial/>

<https://www.tutorialspoint.com/uml/uml_deployment_diagram.htm>

<https://www.lucidchart.com/pages/uml-deployment-diagram>]

***1.2. Component Diagram***

[Create a **Component Diagram** that depicts the software modules and their dependency relationships. Enhance the diagram by **including a description for each module**. Keep in mind that the implementation view should highlight the project's integration aspects. Develop this diagram using draw.io, GenMyModel, or similar, while adhering to the graphical notation rules of UML.

Guidelines and examples:

<https://www.youtube.com/watch?v=o5kOHVZ08o0>

<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/>

<https://www.uml-diagrams.org/component-diagrams.html> <http://agilemodeling.com/artifacts/componentDiagram.htm>

<https://www.lucidchart.com/pages/uml-component-diagram>

<https://codeburst.io/software-architecture-the-difference-between-architecture-and-design-7936abdd5830>

<https://towardsdatascience.com/10-common-software-architectural-patterns-in-a-nutshell-a0b47a1e9013>

<https://www.oreilly.com/programming/free/files/software-architecture-patterns.pdf>]

### **1.3.** **Data Model**

[Make a **Relational** **Database Model (RDS)** as a refined version of the domain model. This diagram should be designed using draw.io, GenMyModel, or similar, and adhere to the graphical notation rules for RDSs.

Guidelines and examples:

<https://www.youtube.com/watch?v=6ey2CXVXpJ0&t=12s>]

# 2. Mockups

[Represent graphically (at wireframe or mockup level) the views or windows of the solution where the requirements of Sprint 2 are reflected, considering that the graphics should implicitly answer the following questions:

1. What? The main content groups.

2. Where? The structure of the information.

3. How? The basic description and visualization of the user-interface interaction.

Guidelines and examples:

<https://www.mockplus.com/blog/post/wireframe-mockup-prototype-selection-of-prototyping-tools>

<https://brainhub.eu/blog/difference-between-wireframe-mockup-prototype/>]

# 3. Repository

[Create a GitHub repository containing commits from each team member. Ensure that each commit includes descriptive and clear comments.

In the **requirements.txt** file, list all required libraries that need to be installed before executing the program

In the **README** file, include instructions for running the program, along with any necessary commands and their respective parameters.]

**4. Video**

Add the link to the video, ensuring access (YouTube, Vimeo).

All members must appear at some point in the video.

Length of the video: **00:04:00** to **00:05:00** minutes.

Video structure:

1. Pitch, presented by ONE of the members (Supporting video https://youtu.be/554Fwv7jQMI) [**00:01:00 minutes**].

* Team members
* Project name
* Problem
* Solution
* Where you are and where you want to go.

1. Explanation (but not reading) of the requirements addressed for this delivery [**00:02:00 minutes**].
2. Evidence of compliance with planned requirements for this iteration [**00:02:00 minutes**].

**5. Project management**

* Record **weekly meetings** (each student should answer: What did you do last week? What are you going to do this week? Are there any obstacles in the way?).
* Record Sprint **retrospective** (the team must answer: What should we continue to do (best practices)? What should we start doing (process improvements)? What should we stop doing (process problems and bottlenecks)?).
* Develop class **assignments** and add them to the wiki.
* **Requirements** with updated status, sprint, and responsible individuals.