1. Introduction (About 3 pages)

   1.1 Motivation.

   Why is it important to do a project in this topic?

   1.2 Executive Summary.

   Write in 3-4 paragraphs what is your project about, including the main features / functional requirements provided.

   1.3 Contributions.

   Enumerate the main contributions achieved. In this section, follow a zoom-out approach, so as to talk from the perspective of a Computer Science graduate. In other words, imagine you were talking to a job panel, and you were decided to show them your computer science skills by enumerating how these are reflected in your project work.

   1.4 Structure of the document: Briefly describe the structure of this document, enumerating what does each chapter and section stands for.

2. Background. (About 10-20 pages)

   2.1 Position your topic within Computer Science.

   Imagine you were trying to position one city in the world by zooming-out from it using Google Maps. What would you see? First, the city. Then, its country. Finally, its continent. In this section the goal is similar. The idea is to zoom-out to as to distinguish three levels:

   1) What is the concrete topic your project is about?
      E.g., Mobile app for online voting.
   2) What concrete area(s) does the project fall under?
      E.g., Mobile applications, Social Networking, Service Providers.
   3) What main area(s) of Computer Science does the project fall under?
      Software Development, Cloud Computing.

   2.2. Project specifics: Minimal background knowledge.

   Imagine you wanted to explain the specifics of your project to a person that knows nothing about Computer Science.

   Obviously, you cannot talk about everything (as the idea is not to write a 500+ pages report). Thus, the key in this section is to identify the minimum set of ideas belonging to the main and concrete areas your project falls under (the areas listed as (3) and (2) in the paragraph above) that are basic so as to understand your project specifics.

   E.g., the minimum set of ideas about software development, cloud computing, mobile applications, social networking and service providers that are basic so as to understand the specifics of a project about a mobile app for online voting.

   2.3. Identify your community.

   Research on the main sources you should be aware of and pay attention regularly so as to strength your knowledge in the concrete topic you are working at.

   Here you should question yourself about your concrete topic, but also about the concrete areas the topic falls under.

More specifically, you should research on:

- Which are the 5 International Conferences and Journals most related to your topic. This is crucial to you, as it represents the main source for keeping you aware of the state-of-the-art on your topic.
    - In particular, it allows you to be aware of other projects related to yours (so that you can compare/position your project w.r.t. them).
    - Also, allows you to be aware of what new techniques are being developed, so that you can apply them to your project.

- Which are the 5 wiki/forums/blogs/Youtube channels most related to your topic. This is crucial to you, as it represents a more accessible, personal and informal way of communication with other people working/interested in your topic. This communication can be extremely helpful for improving your skills, solving technical doubts and increase the global interest/relevance of the topic itself.

- Which could be the 5 companies/organisations most interested in the product you are developing.
  This is crucial to you, as it forces you to not to think just from a programmer point of view, but also to consider the market, any potential stakeholder and niche where your product becomes useful.
  Moreover, Computer Science is a huge topic, and a as graduate you will have access to multiple roles. In this context, if you pick a project in an area you feel passionate about, plus you identify the market interests in this area, then you can drive your future professional career (from the very beginning) towards the path that makes you happier. I know that this does not sound as a very technical reason, but I suppose we all agree that this is probably the most important of all reasons.

2.4. What has already been done in your community w.r.t. your topic?

Once you have identified the top 5 conferences / journals, wiki / forums / blogs / Youtube channels and companies / organisations, the next step is to research in depth on them (spending at least a good week full-time work).

The aim here is to find the different trends followed by previous works done in your concrete topic (c.f. category 1 in section 2.1), and more in general in the concrete areas your project falls under (c.f. category 2 in section 2.1).

In particular, you must find at least 5 concrete works (ideally 10) belonging, or at least related enough, to your topic. You must describe these works and position your project w.r.t. them (i.e., clearly identify the similarities and differences between your project and each of these works).

3. Problem. *What do we want to do?* (About 6 pages)
   Rename this section with the title of your project.

3.1. Problem Definition.

Describe the problem you are trying to solve in your project.

3.2. Objectives.

Enumerate the objectives you want to achieve in your project.

3.3. Functional Requirements.

Enumerate the functional requirements you want your project to have.

Please, do not include here the use-cases. Even if you want to create a one-to-one mapping between each functional requirement and a use-case (which does not necessarily need to be the case), in which case please use this section to describe what do you want this functional requirement / use-case to do. In summary, in no case use this section to provide a description on how to implement the use-cases of your project.

Let me explain this with a bit more of detail. A common mistake is that people use to confuse the problem description with the solution approach. More generally, this is a common mistake of confusing the "*what*" with the "*how*". This section must be purely focused on the *what*: *"What is this project about? What are the objectives? What are the functional and non-functional requirements?"*

How are we going to do all these things is to be explained in next chapter. Provided a problem, an objective or a functional requirement, obviously there might be many ways of tackling it. That is, there might be many "*how to*". However, the definition of what do we want to achieve (i.e., the definition of the *what*) should be unique.

3.4. Non-functional Requirements.

Enumerate the non-functional requirements you want to achieve in your project.

4. Solution Approach. *How are we going to do what we want to do?* (About 15 pages).

4.1. Architecture of the Solution.

Describe the architecture of the solution that you have in mind, including the technologies being involved (e.g., frameworks, programming language) and the hardware needed to develop the project (and to further support it at deployment stage).

Provide a high level view of the class diagram you have in mind, describing each class package, its responsibilities and communication points to other class packages.

Provide a high level view of the database needs of your project, describing the different tables/documents and the relationships among them.

4.2. Use-Case Description.

Describe how you envision tackling the functional requirements of your project via a set of use-cases.

For each of them, provide a complete use-case flow-diagram, including the role of the different parts of the architecture of the solution, as well as any interaction among them.

4.3. Risk Assessment.

Identify any potential risk precluding you from successfully complete your project. Describe each risk in terms of its importance and possibilities to arise. Also, enumerate the decisions you are going to make so as to anticipate it and to palliate it (in case it happens).

4.4. Methodology.

Describe your approach to tackle the different parts of the project, including:

- How to tackle the needed research to fulfil the background chapter.
- How to reach any computer science skill needed to fulfil the project (e.g., describe your plan to learn any new technology involved on the project that you are not familiar with).
- What concrete project managing approach are you going to follow (e.g., Waterfall, Scrum).

4.5. Implementation Plan Schedule.

Come up with a schedule for the second semester, describing how do you envision to achieve the implementation of your project during these 12 weeks.

4.6. Evaluation.

Come up with an evaluation plan, allowing you to measure how much have you actually achieved of the initial goals you proposed when you started the project.

4.7. Prototype.

Although no implementation has done yet, come up with some drawings, snapshots or representation of how do you envision your product to look like once the implementation phase has been completed.

5. Research Conclusions. This chapter should comprise 2-3 pages and enumerate conclusions of this phase of work.

4.1. Discussion

A discussion of some of the issues you encountered during the project.

4.2. Conclusion

Enumerate the main conclusions you have got in terms of background, problem description and the solution approach you have come up with