Guidelines for Project Design Document for CS 188 Spring 2021

The project design document is due at 11:59 PM on Friday April 16. It must be submitted to Dr. Reiher via upload to the CCLE page in the form of a PDF. One design document per team should be submitted, with the names of all team members listed.

The design document should describe the design of the software your team has proposed to build. If you want to build something significantly different than what your proposal outlined, consult Dr. Reiher well before the deadline for the design document. Minor differences (such as changes of language to be used, dropping some of the secondary features of the software, etc.) are acceptable, but should be specifically pointed out in this document.

Lecture 3 (posted on the CCLE web site) provides an overview of what I am looking for in this design document. The document should provide a clear, fairly detailed description of:

- the purpose of the software
- the architecture of the software (its components, their interactions, their functionality)
- how each component will be achieved (using an existing software or hardware component with appropriate functionality, building a component from scratch, altering an existing component to meet the special needs of the project, etc.)
- hardware and software platforms to be used

The design should be described, as well, in one or more models that provide insight into its architecture. These could be component diagrams, data flow diagrams, UML descriptions of the structure of the software, flow charts, or any other fairly specific method of detailing how the software is intended to work. You are free to use whatever type of model you feel is most appropriate for specifying the architecture and behavior of your particular project, but at least one model of appropriate detail is required. The model should capture characteristics of the design that are not necessarily obvious from a simple description of the software.

Since this class is about secure software design and implementation, your design document should include a section in which you discuss the security requirements of your software, the threats you foresee to achieving the requirements, and general approaches you expect to use to mitigate those threats. You should develop the set of threats you expect to deal with using some more structured approach than just sitting around and talking about them. Possible approaches include those presented in class (attack trees, STRIDE/DREAD) and others not discussed in class (such as P.A.S.T.A and Trike). You can learn a bit more about options for threat modeling from this Wikipedia page:

https://en.wikipedia.org/wiki/Threat_model

You are not required to use any particular approach to threat modeling, but whatever you do use should be suited to the software you are building and your approach to building it. You do not need to provide all the details of your actual performance of threat modeling (such as the full set of attack trees you developed or a STRIDE specification of all of the threats associated with each component of your design), but should provide enough detail to give us an idea of how thoroughly you considered possible threats.

In general, your design document should be sufficiently detailed and clear that (in principle) you could hand it to another group and expect them to implement substantially the software that you had in mind. You will be graded down for designs that are too vague and lack critical details. For example, at this stage, if your design uses a database, you should be able to specify if you are using one database or many, where in your system the database(s) would reside, which other software components would interact directly with the database, what critical types of information would be stored in the database, and what the trust properties of the database are.

There is neither a length requirement nor limit for this design document, but you should remember that it is intended to be a detailed design suitable to serve as the starting point of an implementation, even by people who have not been involved in your project discussions or have any insight into your plans beyond what is embedded in the document. Very short project documents (of 4-5 pages, for example) are unlikely to contain sufficient detail, though the decision on adequacy will be based on my reading of the document, not on page count. Do not include a lot of irrelevant information that sheds no light on your proposed design, such as motherhood-and-apple-pie statements about security or lengthy descriptions of well-known existing software components (such as a MySQL database).

The design document should describe your intentions, but is not a contract. You are not expected to produce working software that matches the design in all details. However, if during the course of the project you find it necessary to alter the design or deviate significantly from it, you should discuss that with the professor and the TA. When the time comes for a final project report at the end of the quarter, one element of that report must discuss the degree to which you followed your original design. Again, the point is not that you are required to follow it precisely, but to give you experience in dealing with the inevitable changes that occur in designs during the course of their implementation.

You will not be graded on the quality of your prose, except to the extent that poor writing, grammar, and spelling obscure the meaning.

Submissions that come in after the deadline will receive a late penalty. Submissions up to 24 hours late will lose 10% of the grade. Submissions up to 48 hours late will lose 25% of the grade. Submissions later than 48 hours will not receive any credit, but must still be turned in, if the team intends to continue in the class. Later assignments will not be graded unless a design has been submitted first.