Senior Design 2023-24

This document provides guidelines for your senior design project– design, mentoring, and criteria. In addition to the project design, the course will also cover oral and written communication skills. Starting Fall 2022, the senior design projects are no longer limited to "startup" projects/products. However, all types of projects must meet the criteria discussed later in this document. Projects can be coarsely mapped to three categories:

- Research projects supervised by faculty: these projects will emphasize solving a research problem and evaluating the solution together with a faculty research advisor. These projects could be continuation of prior research but must have new problems (systems) that meet the project criteria defined later in this document.
    - If you are considering a research project, then you need to contact the faculty as soon as possible and get their approval to serve as your research mentor.
- Startup Projects: these emphasize building an innovative idea (with a need in the market) and showing it is marketable. The project should include customer research and some marketing plans.
- Development projects: these emphasize solving a customer need and integrate software (and/or hardware) components. These could provide a capability or set of features that are useful and interesting, despite not always being "marketable". These development projects could be proposed by the student or by a faculty or industry mentor.

Project Proposals – will be due first week of classes.

Each team must propose *at least 2* and a *maximum of 3* projects, each of which must meet the project criteria defined at the end of this document. A requirement of the senior design course is the iteration of your project ideas to finally arrive at the project and the timeline schedule for the project (including deliverables at each of the project demo stages).

SD Mentors (Tentative)

Each project *may* be assigned mentors – these could be faculty research groups (for research projects and/or faculty proposed product development projects), industry mentors, or alumni mentors from industry. The assignment will be made based on the topic (technical area) of your project and after you specify your project (or at least the technical area of your project). The mentors will be meeting with the project teams regularly to help support the teams and provide technical guidance. There will *be required bi-weekly check-ins* with the mentors (and status reports will be documented for assessment).

SD Teams

One of the objectives of the SD course is team software development. Therefore, all projects must be team projects. The size of the team must be 3 persons per team. Exceptions, for a team size of two, *may* be granted only in the case of research projects. When defining the project each team member must be assigned specific components of the project – meeting the criteria described in the next section. There will be regular (bi-weekly) check-ins and team assessments to ensure the equitable and productive teamwork.

Timeline: You can start forming your teams at any time before the semester.

SD Project Criteria

The typical senior design project will build a significant software (and/or hardware) system for an application (or product) and must have technical and algorithmic components. In a nutshell, an SD project needs to have core computer science as its "meat", something that's innovative, something that will challenge you and take you out of your comfort zone. We know that some projects involve a lot of work, but a lot of programming is not a substitute for innovative core CS content. Your project should clearly include both **technical challenges** and an **algorithmic component.**

To illustrate, here are some projects that would NOT be acceptable, no matter how ambitious the programming effort:

1. Any project where a database-backed website is the core contribution. You already know how to build websites - there is no new technical or algorithmic achievement in merely building a more feature-rich website. Websites can definitely be part of a project, but not the core contribution. As a thumb rule, anything that resembles your Database course project (CS 2541) in complexity will not be acceptable.
2. Any project that is mostly a direct application of sophisticated APIs. A good example of such a case (which would not meet the criteria) is populating Google maps with data from some other source. For example, showing restaurants and their ratings on maps. Another example (of a project that does not meet criteria), is providing a front end to submit a query and then using the ChatGPT API to output the answer. Again, we know you can use APIs and piece together an innovative application.
3. Any project whose primary contribution is game design. Even if the game is dazzling and innovative, if the effort is primarily in game design and graphics-package mastery, it won't suffice for a project unless it has a challenging algorithmic component and a key problem/application that it addresses.

Now, there are exceptions. For example:

1. A website can be part of a project whose core includes some algorithmic or systems content. Or your website can include some key algorithmic component. For example, you can show school bus routes on a map and also provide a sophisticated scheduling algorithm, demonstrating why it's effective.
2. Similarly, you can do a game if your primary contribution is something core in graphics or computational geometry.

However, to be granted an exception, you will have to make a strong case for an exception.

Here are some criteria to use in looking for projects:

1. Does the project have an algorithmic challenge and a compelling application of the algorithm? Again, it is not enough to merely code up an algorithm to a problem, as you might in a research project or an algorithms course. *The algorithm needs to be part of a substantive application*.
2. Are there novel CS abstractions/mechanisms and a compelling application?
3. What libraries/technical components do you need to integrate – we expect your project architecture will have several interacting components. For example, a Database backend, Web scraping tool, React.js front end, a mobile app, and SciKit-learn.
4. Will your project build or significantly augment a software tool (e.g., language, compiler, Operating System) that is useful and innovative in some way?
5. Does your project make use of interesting hardware where there's a challenge in controlling the hardware?

A few more points about the project:

- If the success of your project depends on a lot of data, then you'll need to explain how you're going to get the data. For example, if you were going to create a searchable image database (with a new kind of search algorithm), to show that it's effective, you'll need lots of images. Where are these going to come from? How large? Why are the datasets compelling/useful ?
- Your project needs to be innovative - not a copy of some previous project. If it builds on an existing product then what innovative technology are you building into the product. We urge you to think out of the box. For startup projects, will your project excite a patent lawyer (for its technical contribution)?
- You also need to think about what you will demo and you will make your demo compelling. This includes User interface, Usability, the “wow” factor which can include key research contributions, etc. Thus, while much of your project is about the technical meat, you should think about devoting time and effort to these other aspects. Of course, we understand that it's harder to demonstrate “wow” with an OS project than, say, a robotics project.