Project Proposals

If you already have some project ideas, or implemented some prototypes that's great. If not, start coming up with ideas right away.

The following candidate projects can be considered. You shall feel free to discuss with the course staff related to these projects. You are NOT limited to the candidate projects, and you are welcome to propose your own. We do not have strict restrictions on the project you propose, but every project are required to have some web services, e.g., some functionality of you application must communicate with your Azure server instance.

- Building a project from scratch:
  - Building a project from nothing provides the opportunities of following all procedures from brainstorming ideas, designing use cases, building prototypes, modifying requirements, quality assurance, and deploying. Examples are:
    - Building a web game where people can compete with each others online. A good example can be the Terminal (https://terminal.c1games.com/) or Auto-Chess (https://www.pcgamer.com/a-dota-2-custom-mode-may-be-2019s-most-popular-new-game/). Because this course focuses on the programming rather than game design, if you want to build a game, we suggest you to focus on the logic and service perspective of the game rather than the art elements (e.g., spending too much time on background music, character model, background story, etc.). Also note that building a real-time game can be very challenging.
    - Building a content-management system. A good example can be MediaWiki (https://www.mediawiki.org/wiki/MediaWiki) and WordPress (https://wordpress.com).
    - Building a mobile-phone application such as a budgeting tool such as Mint (https://www.mint.com/).
- Extending an existing open-source project:
  - In real life, we always need to read code from others and figure out how to use, modify, and extend upon it. Contributing to open-source project is a challenging but fruitful practice. Example is:
    - PrairieLearn (https://github.com/PrairieLearn/PrairieLearn) is an online problem-driving learning system developed by our university. You can make some contributions to the projects that can be potentially used by your peers and students in the future. One possible idea is to extend the PrairieLearn system to have better GUI interface to create quiz sets. Currently, PrairieLearn uses XML importing to generate questions but for instructors with no tech background, this can be hard to use. An interface where instructors can just type questions and answers on the web page and then have those automatically converted into quizzes, would be a good contribution to the project.

Writing a project proposal

None of the things you write in your project proposal is set in stone for the rest of the semester. You are allowed to make modifications later (if your project gets accepted and if students are interested in joining it). In particular, the specific user stories you do each iteration will almost certainly change. The purpose of writing them down now is to make sure that you think about how much work is needed.

Put as much information as you can into the proposal. This gives other students more information about the project, and will help them determine if they are interested in it. In addition, a fleshed-out proposal helps the staff decide if the project is feasible or not.

Your proposal should be posted on the wiki page you created. You must include the following sections in this order.
1. Description

Give a couple of sentences – **4 sentences maximum** – describing this project and what it does. If your project solves a specific problem, describe the problem here. This is the first thing that will be read, so make sure that it is clear, concise, and gives the gist of the project.

Reducing a project to four sentences can be hard. If you are having problems, you can write a four-sentence description as follows. The first sentence states the problem. The second states why the problem is a problem. The third is a startling sentence. The fourth states the implication of the startling sentence. As an example, a description for the Piazza website would be:

*Students need to communicate effectively with each other and with the instructor for active learning. They currently do so through emails which is cumbersome. Piazza is a website that offers an effective means for such communication. Piazza could greatly improve the learning environment.*

2. Motivation

Describe why you are doing this project. Describe why the project is interesting. If you are not passionate about this project, you probably should not be doing it.

3. Comparison with similar software

List similar software and describe how your proposed software would differ from those you have listed.

4. Programming language(s), libraries, frameworks, platforms

List the languages that you think you will be using and justify your choice. You can make this project multi-lingual. In addition, if there is a specific version of the language that you want to use, please state it: e.g. C# 7.0, Java 1.8, etc.

Also list the libraries, frameworks, etc. that you will be using. If you will be building upon some open-source project, list that here too. Put down the platform that your project will run on: Windows, Linux, OS X, web, Android, etc. or all/some of them.

5. Risks/Challenges

Enumerate the challenges in your project and ways to mitigate the risks.

General examples:

- learning a new language (e.g., Erlang is not for the faint of heart?)
- using a new framework, getting the framework to install, etc. (e.g., are you familiar with Ruby on Rails?)
- requiring specialized hardware (e.g., access to big parallel computer)
- cross-platform compatibility (e.g., Firefox and Edge?)
- new or untested ideas (e.g., research is fun, but risky)
- hard to meet requirements (e.g., is it possible to raytrace at 60 frames per second?)

6. User stories and iterations

Describe at least 4 iterations of the project. Each iteration should be 2 weeks. Make sure that each user story describes something that is well-defined and accomplishable.

If you are planning on learning a new language or framework, do not just put that down; that is not a sufficient description. Instead, put down goals such as learn Ruby by doing the Sudoku Solver puzzle on [http://www.rubyquiz.com/quiz43.html](http://www.rubyquiz.com/quiz43.html). That way you actually have something to deliver. And we can actually tell that you have accomplished something.

At this point, planning will mostly be guesswork, but being specific (despite the uncertainty) will help you better understand the project.

7. Meeting schedule

Describe how many hours you think this project requires for it to be completed by the end of the semester for a team of **about 8 students**. Make a schedule of when you plan to meet to fulfill those hours each week. A good rule of thumb is to spend at least 8 hours each week outside the lecture time. If you list more availability, you have a higher chance to attract other students to your project.

8. Skill sets

List the skills that you would like students who are interested in joining your project to have.

9. Process

Describe the process that you want to follow for developing your project. By default, your team will follow XP as in CS427. You can propose specific changes on XP that you would like to make.

10. Tools

List the tools you would like to use. You should plan for tools at least for these aspects:

- Version control and hosting platform. GitHub ([https://github.com](https://github.com)) and GitLab ([https://gitlab.com](https://gitlab.com)) are the most common platforms used. Git hosting is also provided from our college ([https://gitlab.engr.illinois.edu](https://gitlab.engr.illinois.edu)) and department ([https://github-dev.cs.illinois.edu](https://github-dev.cs.illinois.edu)). It is also OK for you to choose any other platform. Please be specific about: (1) what hosting platform you prefer to use,
whether you are OK with the project repository being hosted on a non-university platform, and (3) whether you are OK with your repository public or private.

- Continuous Integration (CI) / Continuous Delivery (CD) Platform (e.g., Jenkins, TravisCI, etc.)
- Project management (by default we will use this Wiki, but you can propose some issue-tracking system or task-management systems such as Trello)
- Team communication (e.g., Slack, Skype, Google Hangouts)

11. Related proposals, if any

If you are already considering a bigger team and writing related proposals, mention the NetIDs of other team members who are submitting related proposals. Note that each student has to submit his/her own proposal.

Grading

This assignment is worth 10 points. We will be grading you based on how well you address the issues above. Use your experiences from the past semester and reflect on them to help you determine the important issues that you need to address. What were some of the problems that you faced and how would you address them this time in your own projects? What were some tools that you really liked/hated and which would you choose this time around? What are some of your strong skills and what other skills would complement the ones that you have?

Even if your project is rejected, you are still able to get full points for the proposal based on how well you address the issues.

Submission

The staff will be looking through the project proposals right after the due time and will accept or reject each of them. If you are really interested in your project, make the proposal compelling and convincing so that it will be accepted.

Some reasons for rejection include:

- A project that is too complex for most of the students – so you will have trouble getting team members
- A project that is hard to write automated tests for
- A project that is too easy and does not require a team of about 8 students working for about 12 weeks
- A project that is almost identical to something that has already been implemented over and over again. Projects that are likely to be rejected include Address Book applications, Chat applications, Campus bar finder, etc.

Afterwards

The list of approved projects will be announced.

Students should contact the project leads (usually the students who proposed the idea) of a project which they are interested in. You can abandon your own proposal and join a different team as well. And if someone likes your idea, they can always become the lead for the project you initially proposed.