

# Due on October 30/31 [Worth 10 points]

You should write this proposal in a way that convinces your TA that this project is something interesting and worth pursuing. Your TA will base your final project on this proposal so do not deviate too much from it. However, you are allowed to make changes to your schedule each week depending on the progress you have made. It is your responsibility to e-mail the changes each week during discussion section (this prevents you from doing the work at the last moment and just modifying the schedule to suit what you have done).

Strive to create a project that you will actually use. Treat your TA as the customer for the project and your fellow students as the reviewers. This will help you focus on what to incorporate as part of your project.

## Project name

*Give a suitable title for the project.*

## Project member(s)

*If you want to work in pairs, put your partner's name here. Pair projects should require more effort than individual projects. You do not need to pair program if you do not want to. Groups of 3 are allowed but you must have a very compelling reason for doing so and this is subjected to your TA's approval.*

## Language(s)

*List the languages that you think you will be using. You can make this project multi-lingual.*

## Libraries, frameworks, platform

*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, or all of them.*

## Description

*Give a couple of sentence (4 sentences maximum) describing this project and what it does. If your project solves a specific problem, describe the problem here.*

## Motivation

*Describe why you are doing this project. If you are not passionate about this project, you probably should not be doing it. For a game, that is going to be easy: "Because it is fun!"*

## Risks/Challenges

*Enumerate the challenges that you have for this project. Example challenges are: learning a new language, using a new framework, getting the framework to install, etc. A real serious risk is version compatibility. If you are using a framework, I suggest that you do not switch over to the latest version half way between your project.*

## Schedule

*Give a four week schedule. You should allocate work that is approximately equal to what we have each week. Each weekly evaluation is going to be worth around 40 points spread across the Technical, Style and Participation criteria.*

*Order your schedule by priority. The first three weeks should address the "must-have" features. For the last week, you should plan on getting the "nice-to-have" features implemented.*

*If you cannot find 4 week's worth of **real** work, then the project is probably too simple and you need to expand on it or choose a different project.*

Your TA will decide if your proposal is really doable in 4 weeks. Also, your TA will tell you if your weekly plans are insufficient.

### **Week 1 [demo on Nov. 6/7]**

*Week 1's schedule has to be more detailed since this the first thing you will be demonstrating.*

*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>. That way you actually have something to deliver. And your TA can actually tell that you have accomplished something.*

### **Week 2 [demo on Nov. 13/14]**

*Try your best to make this detailed. For instance, for a web application, you should have: set up the users database table and enable logging in.*

### **Week 3 [demo on Nov. 27/28]**

*Try your best to make this detailed. Think of what your customer would like to see done by this time and make sure that you have it done already.*

### **Week 4 [demo on Dec. 4/5]**

*This is probably where you have almost everything working and you are just making it better. You should include a list of features that will really complement your project and make it like a real working application. For instance, adding a proper icon, creating a help file, etc. All the small things that make an application great.*

---

**If you have questions, ask early to save yourself from anxiety.**

---