# Milestone Two

## Primary Objectives:

1. Inception phase activities for class project
2. Refine team project idea(s)
3. Practice Git workflow

## Overall Requirements:

- ☐ Class Project:
    - Lists of needs & features
    - Lists of requirements, including non-functional
    - Output of initial modeling
    - Initial architecture
    - User stories
    - Full vision statement
- ☐ Team project idea Q/A
- ☐ Git pull request/branch/commit evidence

Grade sheets (ods)

## Introduction

This week we begin with one week of Inception on a class project. It's a dummy or throw-away project that we can all go through together to make sure we're doing things right. After that your team will do a second run-through on your 3 project ideas. Finally you'll do some practice with Git on the workflow you'll use this term. I say finally, but you should do this first and use this workflow for all documents you produce this week (and here on out).

## Tasks:

1. Gather with your team for a good amount of time and go through an abbreviated inception phase for the following class project idea. You should be able to do this in a 1-2 hour meeting, but only because this is an abbreviated class project that we will only pursue for 2 short sprints. Follow the example that we learned last term. Remember that the inception phase is not long and does not produce numerous or lengthy documents; however, there are some very clear objectives and things you should do. So do them. Refer to the DAD book when in question. All your documents should go in your team's Git repository in a Milestone 2 folder and preferably in Markdown format (using the correct workflow -- below). I expect that you will produce at least:
    - A refined list of primary needs and features. I'll give you some to start with.
    - A refined list of requirements, including any non-functional requirements. Again, just the top-level, most important ones, or the ones that go with your most important needs/features.
    - Identify Epics and Features and break them down into user stories. Then refine/write at least 10 user stories for your main features. Use this format.

> As a *type of user*, I want *some goal* so that *some reason*.

- Modeling outputs. I would like to see some informal Use Case diagrams for your top features, sequence or flow diagrams and a preliminary database design or ER diagram. Identify Entities and Actions. Please do these on a whiteboard (preferred) or hand drawn on paper -- so you can erase and redraw many times. Afterwards, take a photo or scan to put in your repository.
- Architecture drawing. What hardware/software/network is going to be required to make this work?
- Refine the given rough Vision discussion into a solid Vision Statement 1.0 using the format we saw in the video (in the Software Engineering slides from last term).
- Remember that you don't just go through all these in a linear fashion and check off what's done. Use a cyclical approach where you use the tool that will help you accomplish what you're trying to accomplish: figure out the requirements and potential solution space of the project.

Here's what you should start with: Class_Project_Definition_201920.md Pretend that this document was the output of the initial meeting between your team and the stakeholders/customers. Your job is to take it to the next step and start working on it. You'll have questions for the stakeholders so just ask yourselves (better yet, interview each other or choose 2 members as stakeholders and the other 1 or 2 as the dev team. Role play and act it out). You can personalize this class project and move it to wherever you'd like.

Please go ahead and use this document/format for your team, just personalize, refine, add to it and include it in your Milestone 2 docs. When you create a modeling output (diagram), you should then add a link to it in this document in the appropriate section. This way this document is the starting point and all other outputs can be reached from it. It'll be placed in your repository so you can use links to other documents in that same repository. **Everything, always, goes in your repository!** (except binaries, dlls, etc. of course we all know that by now)

2. Team Project Ideas

   Develop each of your ideas a little more. Do some background research and look for existing solutions. Evaluate your topics thoroughly and then choose TWO to continue with (or your advisor will have discarded one or more already). Write up a short explanation of each topic, that describes it more than you did for Milestone 1. Additionally, answer these questions for each topic:

   - What is new/original about this idea? What are related websites/apps? (Be able to answer the question: isn't somebody already doing this?)
   - Why is this idea worth doing? Why is it useful and not boring?
   - What are a few major features?
   - What resources will be required for you to complete this project that are not already included in the class. i.e. you already have the Microsoft stack, server, database so what else would you need? Additional API's, frameworks or platforms you'll need to use.
   - What *algorithmic content* is there in this project? i.e. what algorithm(s) will you have to develop or implement in order to do something central to your project idea? (Remember, this isn't just a software engineering course, it is your CS degree capstone course!)
   - Rate the topic with a difficulty rating of 1-10. One being supremely easy to implement (not necessarily short though). Ten would require the best CS students using lots of what they learned in their CS degree, plus additional independent learning, to complete successfully.

3. Practice the class Git workflow with all the material from this milestone (feel free to practice with (small) files in a temp folder first).

We will go over this at length in class, so here is just the barest of reminders. The Team lead created the official repo under their account. There are two branches: `master` and `develop` . Everyone will start with a *fork* of the “official” repository (the one that is continuously published). All team members (including the owner of the official repo -- just for practice) fork this repository. They now have a remote repository that is identical, and has built-in connections, to the official repo. Team members create a feature branch off of `develop` for each feature or user story they are working on, work locally and push to their own remote. When a feature is finished they first make sure it will merge into `develop` without conflicts so they first update their version of `develop` in their remote and local, then (and this is key) merge `develop` into their feature branch. Then fix any merge conflicts and commit. Finally once they know their feature can be merged easily into the main repo, push their feature branch to their remote, then issue a *Pull Request* to the owner of the official repo, who reviews it, and if OK approves it and merges it into the offical repo in the `develop` branch. This merge should now be easy and probably a fast forward. After it is merged then all team members can pull it in order to update their `develop` with the latest changes from the official repo. When the Sprint is over, the repo owner merges `develop` into `master` for the official release, which everyone can then also integrate into their forks to synchronize everything.

See these pages for additional information. We're using the GitFlow Workflow with the Forking Workflow. Here's some info on Pull Requests, and here's a general one for forking and pull requests How to Github: Fork, Branch, Track, Squash and Pull Request.

For this task, print out evidence that shows every member of your group is following this workflow. Every member needs at least one accepted (and therefore merged) pull request, and it must be accepted/merged before the meeting each week with your advisor.

4. Lastly, print and assemble your work into the workbook under Milestone 2, including the grading sheet. Yes, everything is in your repository but we want it in the book too so it is easier to see at the meeting and we can write/draw on it for examples.