CS657/790 Cloud Computing

Special Interest Project Overview

This document provides general information about the special interest project to be completed by the end of the semester.

From the Syllabus…

Students often come into this class with an idea or topic that goes beyond what can be covered as introductory/baseline material. Here are some examples of the topics you might be interested in:

- Continuous Integration/Continuous Deployment (CI/CD) Pipelines
- DevSecOps Tooling and Team Processes
- Cloud Security
- Machine Learning Models and Applications
- Infrastructure as Code (IaC)
- Container Orchestration (e.g. Kubernetes)
- Internet of Things (IoT)
- Big Data and Cloud Analytics
- And more…

The special interest project is your opportunity to pursue an area you are interested in, learn it in more depth and collaborate with your classmates in doing so. We will organize the class into a set of teams who want to study the same topics. Your team will define the work you will do, do the work, and present the results twice: once in class at the end of the semester, and once in blog format on this course's blog site (uwm-cloudblog.net). I expect to meet with each team multiple times throughout the semester to coach and guide you through this process. The meetings will be conducted online using Zoom or another online meeting tool.

Required Steps

Here are the required steps for the special interest project:

- **Form a team and submit an idea.** Work with your team to come up with a general idea of the project you would like to do. The idea can come in the form of a few bullet points.
- **Write and submit a Concept Document.** The Concept Document describes what you are going to create with some key points about what it is. More importantly, it shows why your idea is good, and why a customer or user would want it.
- **Design your solution and communicate the design.** The design documentation provides technical details about what you are building. It helps you know how much work you still need to do to be done.

- **Demonstrate your progress to the teaching team.** This is a preliminary demo or walkthrough that assures both you and the teaching team that you are making good progress toward your final goal. If things are not going well, this is the time when we do a triage and revise goals to make them achievable during the semester.
- **Complete your solution.** Finish your development, test everything, and determine what you will be presenting.
- **Present your solution in class.** This is where you explain your work to your classmates and get some experience doing a live presentation.
- **Write a blog article.** This serves as a final written record of your work. It is evidence to the world at large of how good the Comp Sci program is at UWM and the level of talent that employers can expect to tap into when hiring a UWM graduate.

This all sounds like a lot, I'm sure. The key to making this process work is to carefully think through what your team can really achieve in the time you have available. Taking on a lot is heroic, but it can be quite painful if you fall short of your goals.

Concept Document

In the concept document, you identify your deliverable in broad terms and describe its key elements. You should provide your concept document as a PowerPoint presentation with bullet points and a few diagrams. The document should be easy for someone to skim and get the gist of your idea without parsing a lot of detail. Not much writing is required, but there may be quite a lot of thinking.

Here is a suggested sequence of slides you might use. Feel free to add or delete if these don't fit your situation – use your judgment.

- **Cover slide.** Give your product or deliverable a **short** name and provide a date. Generally, the shorter the name, the better.
- **What is _____?** Describe your product or deliverable in 30 seconds or less. This should be something you could say to a senior executive on an elevator if she asks you what you are working on.
- **Description.** Answer the most important of these questions: What form does your deliverable take? What does it do? What problem does it solve? Who could benefit from this, and how? What are the principal benefits?
- **Components, Features and Functionality.** What are the recognizable parts of your solution? What are the "selling points" you can offer to potential customers or users of your solution?
- **Differentiators.** What makes your solution distinct from other similar solutions? If you had to compete with other solutions, what would you say? Are there some reasons why your approach is better than the alternatives?

Design Document

You will need a way to divide up project work among your teammates while still having a common understanding of what you are building. Even if you are working by yourself, you still need a set of "blueprints" to lay out for yourself how the parts of your solution are supposed to work together, and how far along you are toward completion. Your design document will serve these purposes.

Design information takes a lot of forms and it's hard to say exactly what form would work best for your project. There are many techniques and tools available, and you need to decide which to use. The choices of which techniques and how much detail to create are very important. It is easy to spend a lot of time on design documentation that should be spent on code. It is even easier to jump right into code with an inadequate design only to discover that you have no clue how to make your product work.

Sometimes, design information never gets disseminated beyond the development team. It might just be a set of working papers that the team uses. For purposes of this course, I want you to share your designs with me early on so that I can help you trim project scope, select an easier approach, or point you in the direction of useful resources.

Here are a few ideas for documenting your design that you might find useful:

- **Block diagrams.** There are many ways to use boxes, lines, arrows, circles, etc. to communicate how your solution is put together. The rules for how to do these are not well defined, so you can invent pretty much any picture that your project team members can completely understand.
- **Class diagrams.** If you are doing object-oriented programming and you need to capture a domain model, a class diagram does a good job of documenting classes and the static relationships among them.
- **Entity-relationship diagrams**. If you are using a relational database, an entity-relationship diagram does a good job of representing the logical or physical database design. Class models and ERDs are often nearly interchangeable.
- **Flow charts.** If your project involves a sequence of steps or a workflow, old-school flowcharts or flowchart-like diagrams do a good job of capturing the details.
- **Data flow diagrams.** This is an old technique for doing a functional decomposition of your system while documenting process and data stores at a fairly high level.
- **Navigation hierarchy and wireframes**. If you are building a user interface, a navigation hierarchy chart with wireframes is useful to capture layout, page content and navigation among the parts of your user interface.
- **Code.** This is one of my favorites. If I can understand a design in detail right from the start, I will write up the design in the form of code that compiles and runs but doesn't do much. This can sometimes do the same thing that a class model does but without drawing a picture. It can also become skeleton code for the final product. Don't do this unless you are 100% sure that you have the right design.
- **Data stream mockups.** When you are designing APIs that take JSON, XML or other text-based message formats as inputs and outputs, you can create example messages using just a text editor. These are hugely helpful in specifying both back-end APIs and the clients that use them.

- **State transition diagrams.** If you have complex logic, need to maintain a complex object throughout multiple states, and write logic that depends on those states, a state transition diagram does a good job of defining the solution to this challenge.
- **Simple tables.** Here we mean a plain old table like you might create in Microsoft Word. Tables can contain just about any kind of design information you want and act as to-do checklists you can share with the team.

In the time you have for this course, you will only be able to complete a few of these. Choose the ones that are the most helpful to you in organizing your work and understanding what needs to be done.

You can find out more about each of the above with an appropriate Google search.

Demonstration

As you develop software or some other technical solution, you should be accumulating parts of the complete solution that can be demonstrated to others. In this step, you will do a demonstration of what is working so far to the teaching team. Ideally this will show the main content of your application or solution, leaving just the "polishing" for later. If not much is working yet that might be OK as well, but coming out of this, your team and the teaching team should all feel positive about your chances for success.

We will conduct these demonstrations as Zoom calls with the project team, the professor and potentially one of the TAs in attendance.

In-Class Demonstration

You will have 10-12 minutes to present the results of your effort to your classmates during the last two or three class periods. In your demonstration you should put forward the most positive aspects of your work. Try to get your classmates interested in "buying" your solution. Here, "buying" means that they might read your blog article, learn some things from it, and think that your team created something cool.

Blog Article

This is a written presentation of your project, complete with appropriate diagrams and illustrations. We will make a WordPress article out of your material and publish it on the course blog, uwm-cloudblog.net. Your article should be concise (short) and clear, and provide readers with a way to follow up on your work, such as a download, the ability to access your site and try it themselves, or links to other resources. As you get closer to having your presentation materials, the teaching team will provide guidance on how to put the article together on WordPress.