CSE 403

Zero Feature Release

The zero-feature release (ZFR) milestone consists of a skeletal implementation of your product and a document with instructions for accessing some of your development tools. The purpose of this milestone is to work out basic system organization, as well as making sure the major tools and subsystems of your project connect properly, such as your version control, bug tracking, deployment process, etc. Unless otherwise specified, no functionality needs to be working other than a "front page" (for web projects) or "initial welcome screen GUI" (for mobile app projects) as described below. Note that you are not forbidden from having more functionality; these are just minimum requirements.

1. **Featureless live product web site**
   Provide us with a URL to reach a front page for your product. If your project is a web app, this URL should be the front page of your actual web app hosted on cubist* (though it does not need to be functional). If your product has functionality, such as if you are getting started early on coding, that is fine; but such functionality will be ignored for grading this phase. If your project is a mobile app, this URL should be the address of a web page where we can download the app to install and run it. (The page does not need to be at all fancy; just a link to the file(s) to download is fine.) If you're doing a web app, at least one page you post should have some kind of active content on it, such as Ruby on Rails code that runs on the server and is then displayed in the user's browser. What the code does can be trivial, e.g. add 1+1. (* ask the instructor if you wish to host your project elsewhere than cubist.)

   Part of completing your ZFR is creating a database or other data source (cubist has some options). Therefore, your ZFR product must in some way connect to this source of data, fetch at least one piece of data from it, and display the data in some way on the screen or web page. If your data source is plain files, your product should open and display some content from at least one file. If your project involves several sources of data, several databases, etc., your ZFR should connect to each of them, though what it does with this connection may be trivial, such as fetching and displaying one row or record of data from each. (You do not have to have your entire database schema created at this time, just establish connections.)

2. **ZFR instructions document**
   Your group should also submit a short ZFR document describing the items below. Some documents describe processes the user or developer must perform. Part of your grade will be based on the simplicity of these processes, and how accurately your directions match what the user must actually do. You should also put the resources in place so that the graders can examine and test the processes described in the ZFR document promptly after your submission. You may want to test out a "dry run" of your own beforehand. Your ZFR document should address the following items:

   a. **GitHub general setup**
      Your group should create a repository for yourselves on GitHub (https://www.github.com). If your repository is not public you must grant full access for the instructor and TAs to your GitHub resources so that we can examine your work.

   b. **Bug-tracking setup**
      Your group will use GitHub for bug/feature tracking as well. This is where you will document bugs in existing code and list missing features and work to be done on your project in the future. For the ZFR, the bug tracking system should be set up and should have at least one bug or task filed for each active developer. These may be requests for features, such as "implement initial login feature," or can be bug reports, such as "make input form robust against invalid user input." If possible, the issues in the tracker should list a priority and a timeline to fix them. Beginning with the night of your ZFR turn-in, a grader will examine your bug-tracking system once a week. Part of your grade in later phases will reflect whether a significant number of accurate bugs are present that contain proper information such as severity and assignment to specific developers to fix them.

c. **Source control and build process instructions**
Your project's build process is the set of tools and commands to compile and "build" your system. Turn in a set of directions to find your build system, check out its files, and build them. Include a list of any extra software or packages that another developer would need to build and run your software. In grading, we will follow these directions. The instructions should contain the complete set of commands to extract your code from your repository on GitHub, compile the code, and start it running. The directions should be written in sufficient detail that an intelligent developer can follow them. If the system(s) require login information to access them, you should provide this information in your document. Part of your grade for this item relates to the number and complexity of commands the developer must use. Ideally your system will have a single command that does a "one-step build," that checks out all source code from your repository, builds all necessary binaries, packages them, and places them in a known location. Beginning with the night of your ZFR turn-in, the grader will log in to your version control system once a week to count the number of files and lines present in the system, as well as the number of lines modified, and log your group's development progress. Lack of significant weekly progress may impact your team's grades on later phases.

d. **Data access instructions**
Since your product must have a server-side data component, your ZFR should contain a set of instructions about where this data is stored and how to access it. Turn in a set of directions that tells the grader how to find your data, and how to briefly perform a trivial access of this data. For example, if your data is in a database, inform the grader how to connect to this data and perform one very simple query against it. In grading, we will follow these directions.

Submission and Grading:
Part of your grade comes from your web URL actually working: that is, from the grader being able to connect to it on the first attempt. The URL should remain "live" throughout the quarter while we are checking your work. Your data access instructions should be comprehensive and should list all steps necessary for us to find and access your data. For full credit, these directions must match the actual steps we need to perform to find and retrieve the resources.

Your build process instructions should be complete and correct. We should be able to successfully build and run your app on the first attempt, barring grader error. If any additional software or packages need to be installed for your software to run, this must be documented as well. The build process should not be overly complicated; streamline the build process so that a reasonably intelligent developer can follow it.

Your GitHub resources should actually exist when we go to examine it, and we should be able to log in and access them successfully on the first attempt.

One bug or task must be listed per developer with suitable details such as to whom the bug is assigned and its priority.

A small part of your grade comes from the looks or aesthetics of your documents. They do not need to be beautiful or excessively formatted, but your customers need to be able to read them and extract information from them. This means they should be clearly written, with proper spelling and grammar, clear wording, and formatted with enough organization to present your ideas clearly to the reader.

Your ZFR does not need to reflect "customer" interaction, but you can ask your customer (cse403 staff) to try testing out various aspects of your site before you officially submit your ZFR ("Are you able to connect to the following URL?", etc.). If you make your request a reasonable amount of time before the due date, the customer will do his/her best to try to accommodate such requests in a timely manner and give you feedback about the results.