CSE 403
Autumn, 2013

Alpha Release Deliverable, Phase 3a

The "Alpha Release" deliverable includes a first implementation of your product with a partial incomplete feature set (both a committed GitHub repo version identified as Alpha and a deployment of that codebase) and some additional documentation (both [A] management-oriented and [B] user-oriented).

We are looking to see a significant effort to implement a testable, usable (yet incomplete) product that implements an adequate amount of functionality to get a feel for the system and how it will work. Our intention is to have you implement a non-trivial amount of functionality in this phase, but still substantially less than the overall end product.

If necessary, you can implement "stub" versions of any use case functionality, such as a fake user account that can be used during a scenario, or fake data already populating the database that major features will manipulate. (Recall that earlier phases asked you to consider the dependencies between parts of the system so that you could prioritize and schedule when you would implement them and get "enough" of various parts built so that you could code other parts against them, and proceed with simultaneous development of several pieces if possible. Your team may have already made some "dummy" parts or test data. Your team may also have put together a "skeleton" or "framework" of dummy pieces to be later fully implemented.)

It is permitted for some functionality to be missing, but the system should already be useful, even if in a limited way. The Alpha Release management-oriented documentation on your wiki (see below) must clearly indicate which commands or other features are working, which are partially implemented, and which are still to be done. User-oriented documentation at this point should not only explain to the user how to use the software, but should also spell out to the grader exact scenarios of how to demonstrate and test the functionality of everything you have operational thus far (both fully and partially). (If you have not deviated much from original features, some or all of the use case descriptions from the requirements document may be a starting point for user documentation.)

Your product need not necessarily be 100% bug-free to receive full credit. However, any known bugs should be documented in your issue-tracking system, and a user testing the system against your documented scenarios should not encounter a significant number of bugs that are not listed in your issue-tracking system. Your product should be robust: errors should be gracefully handled as much as possible (for example, data validation in an HTML form). **If you have not had time to implement some error handling features, these omissions should be noted in the issue-tracking system so they do not come as a surprise to the user testing the system.**

[A] Alpha Release Documentation (Management-Oriented):

1) High-level summary of what is included in this release and what is not

On your group's GitHub wiki, create a separate page for your Alpha release containing a brief summary of what functionality exists in the Alpha version, including to what extent your major features and use cases from your requirements are implemented and which extensions to the use cases are supported. Likewise, indicate what features or parts of use cases are only partially implemented, or not yet implemented.

This information gives everyone a “snapshot” of the project’s progress thus far.

2) Individual Contributions to the Team Effort

Include on the wiki a brief list of team members and their contributions to this phase of the project as a high-level summary of who did what.

Additionally, if an individual team member's contribution is not obvious from the repository (such as if a pair worked together), please make this clear, such as by indicating it in the commit messages when checking in that code to the repo.

Also if team members are working on other project tasks besides coding, please indicate that via the issue-tracking system (they should clearly have “issues” or “tasks” they are working on and demonstrating progress or completion of such).

This information gives everyone an idea of how the workload is being distributed, and if adjustments need to be made. Your team should have **both** the high-level summary and the more low-level attributions of commit messages and entries in the issue-tracker.

3) Tentative schedule for the implementing the rest of the functionality after the Alpha Release – aim for all core features (and perhaps the “extra features” if time) implemented by the Beta Release, but not necessarily 100% bug-free

Provide a tentative schedule for how future implementation is planned (what is the order in which items will be developed, when will they be started, when will they be complete, is there a person or sub-team assigned to that functionality yet). **Accuracy in scheduling is very difficult to accomplish, so don’t worry about predicting the future perfectly; just be reasonable in planning what you think you can get done in the allotted time.** A schedule can be presented as a table, a spreadsheet, a Gantt chart, a graph, a calendar, or whatever makes sense to your team and can also be easily understood by the cse403 staff. You can also combine scheduling information with part (1) above if it is clearly presented.

This information gives everyone a “plan” to know their responsibilities and deadlines, to manage their own time, and to see how their work impacts or is impacted by everyone else.

4) Issue- and Bug-tracking

Your Alpha Release does not need to be 100% free of bugs. It is expected that your product may have some small issues that come up during testing or usage of your implemented functionality. We do, however, expect a reasonable effort at producing high quality software that does not show excessive or "show-stopping" bugs that stop the customer from being able to perform normal usage and evaluate your work.

Any known bugs or issues in your product should be documented using your group's issue tracker on GitHub. If we encounter significant bugs during our own testing that are not represented in the bug tracker or documented on the wiki as features that are currently unimplemented, you may lose points.

[B] Usage Instructions (User-Oriented):

Briefly describe how the user can download and/or execute your product to experience and use the functionality implemented in the Alpha Release codebase. (This documentation is targeted towards the end user of your product, a customer. The ZFR documentation, rather, was targeted towards either a new developer joining your team or a system admin who must deploy your product from scratch after you have graduated and are no longer available.)

For web-deployed projects, download and installation is probably not applicable, but basic user documentation must be more than "go to this website". How should a user actually use the product? **If you are planning to build-in "help", tutorials, demos, FAQs, or other user instructions as part of your website or app, then you may begin so now rather than having the information separate on the wiki, but the wiki entries for the Alpha Release must clearly indicate how the user can access the documentation from within the product. If you eventually will implement built-in help but do not have time to implement it in this release, then put the instructions on the wiki for now.**

The grader will go through your user documentation to test-drive the functionality of your product, so make sure it is clearly written and documents all the features you have implemented or partially implemented thus far. If the grader needs to know about "fake" user accounts or "fake" data in order to test-drive the product, make sure you clearly provide all information they need.

A "scenario" approach similar to your use cases would probably make the most sense at this phase to guide the grader through a demonstration of your implemented features. Later phases can augment user documentation with (for example) FAQs of exceptional cases, additional tutorials, demos, high-level feature documentation that doubles as "marketing", etc.