CEN 4010 Principles of Software Engineering

Milestone 3: More Detailed Requirements, Architecture and a Vertical Software Prototype

Florida Atlantic University

Dr. Shihong Huang

shihong@fau.edu

# Instruction:

Milestone 3 consists of two parts:

1. Milestone 3 document – an expanded version of Milestone 1
2. A vertical software porotype

**Part 1: Milestone 3 document**:

Milestone 3 has to be reasonably consistent with Milestone 1 and instructors’ feedback but it can also differ from Milestone 1 based on what you discover and develop in your design process in spirit of iterative software engineering process and based on the feedback you get.

The difference between M1 and M3 DO NOT need to be edited in Milestone 1 document which remains frozen. You should start with Milestone 3 only after you have incorporated instructors’ feedback on Milestone 1. Milestone 3 document is a separate document from Milestone 1.

**Part2: Vertical software prototype**

In addition to the Milestone 3 document, the team will create a “vertical software prototype” to test the infrastructure and chosen frameworks and to jumpstart the coding effort. The vertical prototype is the code that exercises full deployment stack from browser, via middleware, to DB and back-end, including your chosen framework. It has to be deployed from team account, in the same way that the final product will be deployed. For example, it shall allow one to enter a search term in the browser, then get a response form the DB and render it back on the browser. GUI for this can be simple one field entry and DB can have only a few items. The items in your DB shall be encoded with full schema as it is defined by now. The purpose of vertical prototype is to early and quickly test basic software components and deployment infrastructure and frameworks as well as the key architecture patterns and thus to serve as a basic “scaffolding” for final product. It also serves as “teaching and training” tool to bring the rest of the team up to speed on development, frameworks etc. We recommend that back-end team be assigned the task of constructing this vertical prototype.

Milestone 2 Document must be in the following order:

# Title page

See Milestone 1 instruction

# Executive Summary

Modify based on Milestone 1. Add or change as you see necessary.

# Competitive analysis

Modify based on Milestone 1. Add or change as you see necessary.

# Data definition

This should be reasonably consistent with Milestone 1 but should be expanded as needed and refined as per feedback. Major data items that comprise of sub-data items have to be defined in full (list all its sub-data items, and for images/video list formats, max size etc.). You must use all the data definitions and names consistently in all documents, including GUI text. Focus on data items unique and important to your application and avoid explaining obvious things like Internet,, Browser, Cloud, etc. Be sure to cover ALL items critical to your project and especially those providing a competitive advantage. At this stage data describing user privileges, registration info and main info (raw data, metadata, supporting data) have to be fully defined (as much as it is possible at this stage)

# Overview, scenarios and use cases

Modify based on Milestone 1. Add or change as you see necessary.

# ~~Initial list of~~ High-level functional requirements

Expand functional requirements from Milestone 1 into Milestone 3, with more details as necessary. Keep the same reference numbers with respect to Milestone 1 (i.e. if high level requirement was number 3 in Milestone 1, then in Milestone 3 more detailed requirements are 3.1, 3.2 etc.). Be sure to cover ALL and especially unique features of your product. OK to add new or delete previous functional requirements from Milestone 1, if you can justify it.

Prioritize each requirement/spec with 1, 2, 3. (1-must have; 2 – desired; 3 – opportunistic as defined in the class). To develop these priorities think of the user, use cases, and making your application complete from usability, marketing and business aspects. Base this also on your skills, resources and schedules. Instructors will check final priorities. The priorities you set later in Milestone 4 will constitute your commitment (especially priorities of 1), so be very careful.

# List of non-functional requirements

Reference to your final high-level functional requirements, modify based on Milestone 1. Add or change as you see necessary.

# High-level system architecture and database organization

Modify M1 accordingly, and add the following:

1. High level Architecture of the code must be consistent with UML class diagram (see below).
2. DB organization: Describe the main database schema/organization (high level), e.g. list main DB tables and items in each DB table
3. Media storage: Decide if images and video/audio will be kept in file systems or in DB. Describe any other special data format requirements like for video/audio/GPS etc.
4. Search/filter architecture and implementation: what will be the algorithm for search; what DB terms will be searched, how it will be coded and organized in the DB. Similarly, say what DB items will be filtered/sorted
5. Your own APIs: Describe and define at high level any major APIs that you will create
6. Describe any significant non-trivial algorithm or process (like rating, ranking, automatic prioritizing of items etc.)

# High-Level UML diagrams

Familiarize yourself with Unified Modeling Language (UML). Find your favorite UML tutorials from the Internet. One good one is <http://edn.embarcadero.com/article/31863>

At minimum provide:

1. High-level UML class diagrams for implementation classes of core functionality, i.e. functionality with provided interfaces. Focus on main high-level classes only (one or at most two levels deep). This must reflect an OO approach to implementing your site.
2. UML Component and deployment diagrams

Use data terms and names consistently with Glossary/Data Dictionary.

# Identify actual key risks for your project at this time

Identify only actual and specific risks in your current work such as (list those that apply:

1. Skills risks (do you have the right skills),
2. Schedule risks (can you make it given what you committed and the resources),
3. Technical risks (any technical unknowns to solve),
4. Teamwork risks (any issues related to teamwork);
5. Legal/content risks (can you obtain content/SW you need legally with proper licensing, copyright).

Tell us how do you plan to resolve risks? The key is to resolve risks as soon as possible. Categorizing risk as above helps a lot in managing them. Be brief: identify the risk and explain (2-3 lines), list how will you address these issues (2-3 lines)

# Submission

Store the modified Milestone 3 in your GitHub repo.

Each team submits one single word document with all the above required sections to Canvas by the due date. Must have a title page to your document.

# Grading criteria

Your document needs to be well-written, well-organized (formatted) and reads well. Grading is based on cohesiveness and completeness.

1. Executive Summary 10 points
2. Competitive analysis 10 points
3. Data definition 10 points
4. Overview, scenarios and use cases 10 points
5. High-level functional requirements 10 points
6. List of non-functional requirements 10 points
7. High-level system architecture (UML) 10 points
8. Identify risk and actions 10 points
9. Working with GitHub 10 points
10. Vertical demo 10 points