Project: Implementation Stage

**Due date:** Thursday, March 20.

## Overview

This document provides instructions for the implementation stage of the course project, and specifies the expected deliverables, as well as the delivery process.

## Deliverables

The project will have the following deliverables:

- Final database design document. You need only submit the simplified version of the database design document, which includes:
  1. The final E-R model of the database.
  2. The final relational model of the database.
  3. Changelog. Include all changes in design undertaken *after* **Lab 4** submission.

  Submit hardcopy to the instructor at or before the demo.

- Code. All code developed for the project. The code must be submitted to the project wiki. Each team shall create a separate sub-repository of the form Color-Name and store the code archive within the sub-repository.

  Include all developed code. Red Cohort teams shall include the entire snapshot of the web site developed by each team. Also, Red Cohort teams shall put a link to their sites on the project wiki.

  Submit softcopy to the wiki. Notify instructor via email when submission is made.

- Documentation. Brief documentation outlining how to install the software developed by each team shall be included with the code submission.

  Submit softcopy with the code. Submit hard copy to the instructor at or before the demo.

- Data. Each team shall ensure that there is a way to populate the database of their application with data. Use the NUTRITION dataset as the sample database. Provide instructions for how the sample database can be loaded.

  Generally speaking, this can be achieved via one of the following ways:

  – You can embedd "load sample database" functionality into the front end of your software. (for example, accessbile only to the program administrator accounts).
  – You can create a standalone script which creates the database and populates it with data.
  – You can include SQL scripts for creating the database and for populating it with data.

  Submit softcopy of all materials with the code. Submit hardcopy of the database setup instructions to the instructor at or before the demo.

- Status Report. Each team shall prepare an assessment of the team's progress towards the full completion of the software. In particular, the status report shall indicate which functionality is completely implemented, which functionality is partially implemented, and which functionality is not implemented at all.

  Submit hardcopy to the instructor on or before the demo.

  ***Please note, that you are not expected (although you are encouraged) to implement the software application in 100% correspondence with the specification.*** See below for specific comments on project completeness.

- Experience Report. I would like to know about your team's experience with working on this project. What have you learned? What was easy? What was hard? What did you like? What you did not like?

  This part of the submission will not affect the final grade for the project or any project component. If you do not feel comfortable submitting your team's experience report together with the rest of the deliverables, you are welcome to submit it **after you learn your final grade**. The information from the experience report will be used to prepare and administer CSC 366 projects better and more efficiently in the future incarnations of the course.

  Submit hardcopy to the instructor at the demo, or submit softcopy to the instructor after receiving the course grade.

## Project Demonstration

The final deliverable for the project is the demonstation. Each team will have 30 minutes to demonstrate the operation of their database application. The demos will take place on Thursday, March 20 and Friday, March 21.

The demo sign-up sheet will be passed in class. At least one student from a team must be available to give project demonstration at the designated time. I encourage participation of everyone in the demo, however, it is not required: individual grades are not affected by whether a student was able to make it to the demo or not (only by whether a representative from the team was available).

We will try to hold as many demos as possible in 14-302. Should 14-302 be unavailable for some time slots, we will hold the demos in the 14-235 open lab.

**Demo instructions.** Make sure you know *where* the demo will be held. Arrive at least 10 minutes prior to the scheduled time. Log in, set up your demo. Make certain your program (or web site) runs. Load (if necessary) the database. The demo should concentrate on working with already existing database.

Based on their availability, Dr. Edwards and/or Stephanie Yarbro may participate in the demo as observers.

## Project Code

It is expected that each team shall make a concerted effort to establish a significant code base for the project. At the same time, due to the time frame of the project, I do not expect completed software.

The grading criterion for each project will be *establishment of proof of concept* for the functionality of the group's project. Specific comment for each cohort follow.

### Green Cohort Projects: Day-to-day management of the nutrition program.

Green cohort teams are tasked with building software for supporting day-to-day operations of the horse unit and the animal nutirition program. It is expected that the software completed by the teams will have adequate functionality to support most such day-to-day operations.

Operations of particular interest include:

- Data entry. Functionality for adding information about a new specimen, new location, new ingredient, new diet is expected to be available. Similarly, functionality for modifying existing information is also expected.

- Animal nutrition. Functionality for assigning diets to animals is expected to be available.
- Animal management. Functionality for assigning animals to locations is expected to be available.
- Inventory management. Functionality for inventory withdrawls and inventory additions is expected to be available.
- Information. Basic queries are expected to be implemented. Basic queries include information about animals, possibly restricted by animal features (e.g., breed, age, gender, etc.), informtation about available diets, locations, ingredients, information about location and diet history of animals and information about ingredient inventory and withdrawl/addition history.

The front-end design of the software shall target its use for day-to-day operations.

### Blue Cohort Projects: Analytical software.

Blue cohort teams are tasked with building software for supporting nutrition program analytics. Two main analytical tasks supported by the software are inventory reconciliation and resource use projection. It is expected that the software completed by the teams will have functionality addressing this task.

Operations of particular interest include:

- Diet management. Functionality for adding a new diet and assigning diets to animals is expected to be available (as this functionality is important for some of the main analytical tasks).
- Inventory reconciliation. Functionality for inventory reconciliation is expected to be available. This functionality includes: (a) ability to enter inventory information; (b) cross-checking entered inventory information against expected inventory based on (i) withdrawls and/or (ii) animal diets; (c) ability to enter reconciling inventory transactions (such as shrinkage, or inventory transfer).
- Inventory use projection. Functionality supporting *some form* of inventory projection is expected to be available.

The front-end design of the software shall target its use for analytical tasks. While the software may need to implement other tasks (e.g., some queries/reports, some extra data management), the stress of the UI shall be on the inventory reconciliation and inventory use projection tasks.

**Red Cohort Projects: Web-based software.**

Red cohort teams are tasked with building web-based application front-end to the nutrition database. The web-based application front-end stresses delivery of information to a wide range of user categories, including unique (among cohorts) user categories of animal owners and guests. As such, it is expected that the functionality of the software will include the ability of the software to work with **all six** user categories and deliver authorized information to them.

The main stress of the web-based application is on queries and reports. Operations of particular interest include:

- Guest browsing. It is expected that guest browsing functionality (i.e. access to all parts of the database open to guests) is available.[1]
- Owner-specific functionality. It is expected that the application supports owner-specific functionality. In particular, owners are expected to be able to view their animals, and any relevant information (diet, diet history, notes, location, location history) associated with their animals. Implementation of diet revision requests and other similar forms communication of the owners with the nutritionists is encouraged.
- Data access for managers. It is expected that the application provides sufficiently rich level of functionality for information access for managers and employees of the nutrition program (within the framework of explicitly state access privileges). This includes browsing information about individual specimens, access to their location and diet history. This also includes access to current inventory information and withdrawls/addition history.

The UI design has to concentrate on allowing efficient information needs specification and on clear presentation of results.

## Notes on Grading

Your final submission will be graded based on

1. Successful delivery of all required materials.
2. Overall correctness of design (you **should** have sound design by the implementation time, anyway).
3. Success of the proof-of-concept of your team's implementation.

Additionally, points may be added or deducted for any features that enhance or hamper the usability of your software.

[1] This functionality, of course, shall be available to all user categories.

**Note:** Bugs in code leading to run-time issues (run time exceptions in java, server hang-ups etc...) revealed during demo, or during any subsequent test runs of your code by instructor will result in score deductions. The only exception is any runtime issues associated with improper input. Your project will be tested only on proper input.