**Software Engineering Project Demo**

**Demo #1 -- Iteration 1(c) --  (**[**date listed here**](http://www.ece.rutgers.edu/~marsic/Teaching/SE/syllabus.html#DELIVERABLES)**)**

1. [**Product Brochure**](http://www.ece.rutgers.edu/~marsic/Teaching/SE/demo1.html#FLYER)
2. [**Demo Format**](http://www.ece.rutgers.edu/~marsic/Teaching/SE/demo1.html#FORMAT)
3. [**Demo Schedule**](http://www.ece.rutgers.edu/~marsic/Teaching/SE/demo1.html#SCHED)
4. [**Effort Breakdown and Grading**](http://www.ece.rutgers.edu/~marsic/Teaching/SE/demo1.html#GRADING)
5. [**Demo Documents Submission**](http://www.ece.rutgers.edu/~marsic/Teaching/SE/demo1.html#SUBMIT)

This demo should show your interim accomplishments. The main criteria in grading the demo will be the **complexity and quality** of the software product. It should implement non-trivial behavior and run without or with minimum crashing.

**After this demo, you should pause and reconsider your project objectives.** Do you believe that you can accomplish all the goals you set out to accomplish by the end of the semester? Are you falling behind the schedule? Should you scale down the project? Should you simplify or entirely remove some use cases? Conversely, if you feel that things are going smoothly, perhaps you should expand the system features? Perhaps currently your project is not ambitious enough?

**1.   Product Brochure**

Each team should bring to the demo a one-to-two-pages advertising leaflet providing basic descriptive information about the software product they developed. No more than two (2) pages are allowed! At the least, the brochure should include the group number and members’ names, project title, project webpage URL, the list of features and short description of each feature, system requirements to run the system, etc. A block diagram of the system and some screen snapshots could be useful, too. Please keep in mind that this is a marketing brochure, not a technical document.

**2.   Demo Format**

The key part of the demo should be spent in demoing the software product. Do not waste too much time on non-essential features of your product, such as login, user authentication, and creating new user accounts (we see this all the time and it’s boring to see it anew). Create some user accounts before the demo, so you can use the existing accounts to demonstrate the system functionality. Explaining how login works is like advertising that a car has a steering wheel. Imagine that you are buying a new car and the salesman starts telling you how it has a functioning lock, working brakes, and even the steering wheel is functioning! I don’t think that you would find this even funny! Of course, such features are critical, but they are **assumed** and *never mentioned*by serious customers and salesmen. If you need to login during your demo, just do it as quickly as possible, without mentioning it. You must show the **essential and unique functions** of your product, not those that are *assumed by default*.

The demo format is as follows:

1. Use up to five (5) *slides* to give a short overview of the product—who and what it is for
2. Use *live demonstration* to show the features/functions that you implemented so far:
   1. Briefly describe what the function does
   2. Demonstrate a scenario of how the user uses it
   3. During the demonstration, highlight important aspects, such as user interface design, any data processing at the application’s back-end, etc.
3. Briefly outline your plan for the rest of the semester—what new features you are planning to have ready by the final demo. Use one (1) slide only.

Please keep in mind that this should be a **technical demonstration**, not a marketing pitch. If your application uses a database, prepare several example records *before* the actual demo, so you can focus on essential features during the demo. We don’t want to see how you create another user profile or manage user accounts.  
Likewise, if your system offers different services to different user types, create profiles for different user types *before* the actual demo. During the demo, focus on the services offered to these users.

You want to impress the audience that:

* You are addressing an *important problem*
* The problem presents *technical challenges*, which you are successfully tackling
* Your solution employs *sophisticated software engineering techniques*
* This product will be easy to understand and operate/use

You must *guide* the audience through your slide content, help them focus on the important text (foreground) vs. stuffing/supporting text (background).  
For example, when presenting a text-based slide, you need to break monotony, so use various highlighting of keywords or key concepts (boldface, colored font, animated box around the keywords, etc.)  
Similarly, when you just show a table of data, or a chart, you are actually asking the audience to come up with ideas of what this bunch of numbers or lines actually means. Different people come up with different ideas, and they are unsure if they are seeing that table or chart the way you want them to see it.   
If you want to avoid totally incoherent interpretations, *you must tell your audience what they are supposed to see in your data or charts.*

Plan well in advance, so you can avoid disaster scenarios:

* You’re programming the night before the demo and a team member accidentally overwrites the latest code with an old version.
* Everything is running fine on your home computer, but you never tried it on the ECE server, so the demo fails.
* Your software is demonstrated from a notebook/laptop that for some reason is not compatible with the data projector in the room, so the demo fails.
* A key team member, the only one who knows how a particular module works, cannot make it to the demo so the others are stuck. Every team member must be familiar with the system so to be able to conduct the demo on their own (and to answer questions).

**2.1   Demo Attendance**

*All* group members are **required** to be present for their demo. It is *not* required that each member take part in the demo. However, you must email us your individual [contributions breakdown](http://www.ece.rutgers.edu/~marsic/Teaching/SE/demo1.html#GRADING) to facilitate fair grading.   
If someone cannot make the demo, they should provide doctor’s notice or other justified reason for absence.

All students are *encouraged* to attend other groups demo so they can see how other students present their work and how their progress compares to the rest of the class!

**3.   Demo Schedule**

**The Demo Room is at D-110 (**[**EIT lab**](http://ecs.rutgers.edu/eitlab/)**), Engineering Building D-Wing.**   ([EIT reservation schedule](http://ecs.rutgers.edu/eitsched.php))  
Any food or drink is strictly forbidden in this lab. There is a large projection screen to show your work to everyone. The operating systems are Windows and Linux. (Check details about the available software on the [EIT lab home page](http://ecs.rutgers.edu/eitlab/).)

The demo should last **no more than 20 minutes** (not including the setup and cleanup times). Make sure your demo is ready **before it starts**; as you can see, the schedule is tight and it is unfair to the groups that follow your presentation if your delay messes up their schedule. See the TA **before** the demo if you need help with installing and trying it on the lab machines.   
Students are required to test the computers and any other equipment **before the actual demo**. In the past there were issues with software versions and hardware compatibility. You may use your own laptop/notebook and connect to the projectors. If you’re using your own laptop to project the slides and run your software, you must ensure that your computer works properly with the projector in the lab **before the actual demo**.

The current demo reservation schedule is maintained by the TA. If you have preferences for the demo slot, email the TA immediately requesting a time slot for you to give the demo (the slots will be assigned on a first come first served basis). Otherwise we will assign you randomly to one of the remaining slots. Do not choose time-slots immediately before your other classes, because it is difficult to control exact timing of all presentations, and some of the presentations before yours may last longer than planned.

When contacting the TA, put the request title in the subject line and indicate the group number for which the reservation is being made so to speed up the work. Indicate your 4 preferred slots and periods during which you have other classes.  
Slots will be assigned based on first-come-first-serve. The earlier you respond, the greater are the chances that you will get your preferred slot. If your preferred slot is not available, you will be assigned another slot during which you don’t have classes.

We will try to have the demo without much gap in-between two presentations. If there is a long gap, we may need to reschedule some presentations.

**4.   Grading**

See also the [grading policy](http://www.ece.rutgers.edu/~marsic/Teaching/SE/projects.html#GRADING) for the assigning the overall team grade vs. grades for the individual members.

**4.1   Individual Demo Grading**

To make possible individual demo grading, you must email us the [breakdown of individual contributions](http://www.ece.rutgers.edu/~marsic/Teaching/SE/projects.html#CONTRIB) immediately after the demo. It is very important that you email your contribution breakdown for demo presentations. Naturally, the person who presents leaves impression of knowing the best and contributing the most. If other team members did less visible parts (coding, debugging, slide preparation, etc.), they must inform us! We cannot know who did what unless you tell us!  
It is preferable that the specific contributions of each team member are explicitly listed, but if all teammates agree, it is acceptable to simply declare “equal contribution”.

The individual contributions must be split along the following items:

1. program code writing—list explicitly which use cases or user interface screens were programmed by individual team members
2. [unit testing](http://en.wikipedia.org/wiki/Unit_testing)‡ (includes writing the program code to run unit tests and analyzing their coverage)—breakdown by individual team members, applies to all remaining items
3. webpage design (HTML, [XML](http://en.wikipedia.org/wiki/XML), if applicable)
4. integration and integration testing, cross-platform issues, if you are developing for desktop and smartphone‡
5. testing algorithms that implement mathematical models, non-functional requirements, or user interface requirements stated in your Report #1‡
6. debugging
7. program documentation, including code comments, [technical documentation](http://en.wikipedia.org/wiki/Software_documentation#Technical_documentation), [user documentation](http://en.wikipedia.org/wiki/Software_documentation#User_documentation), etc…‡
8. designing database tables and maintaining the database (if applicable)
9. data collection (if applicable)‡
   * writing scripts or programs for data collection (if applicable)‡
   * Other: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
10. brochure/flyer preparation—informativeness and general appearance of the product brochure
11. slides preparation—informativeness and general appearance of the slides
12. project management, including
    * opening accounts on public websites for the project
    * organizing meetings
    * coordinating activities
    * system evaluation with external users (if applicable)
    * installing and maintaining the developer’s resources

([IDE](http://en.wikipedia.org/wiki/Integrated_development_environment), relational database, Web server, public archive, [version control](http://en.wikipedia.org/wiki/Version_control), ...)

* + combining all of the contributions into the correct files and formats
  + Other: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (include everything that you believe contributed towards making your demo successful)

**NOTE**: Items marked with(‡) will be assigned points *only if evidence is provided that they were actually done*.  
If you programmed and ran **unit tests** for your project, you may include the test diagrams (based on previous sequence diagrams) that show the tested components and mock object (drivers and stubs).

**IMPORTANT NOTE:** This breakdown should contain only information about your software implementation and testing activities. You should **not** report any activities related to requirements analysis and software design.  
Such activities should be reported as part of your [Report #1](http://www.ece.rutgers.edu/~marsic/Teaching/SE/report1.html#GRADING) and [Report #2](http://www.ece.rutgers.edu/~marsic/Teaching/SE/report2.html#GRADING).  
The team members who focused on requirements and design, and contributed less to the implementation and testing should not ask that the individual contributions be artificially tweaked so that their efforts are properly reflected. Instead, they should claim greater contribution to the project reports.

All demos will be graded independently of each other, as follows:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Aspect | Coding | Unit testing | Web design | Integr ation | Debug ging | Doc's | Data coll'ct | dBase | Flyer | Slides | Project mgmt |
| Points | 30 | 10 | \*5 | 10 | 5 | 10 | \*8 | \*5 | 2 | 3 | 12 |
| Total points | 100 | | | | | | | | | | |
| \* Point numbers preceded with an asterisk will be graded only if applicable. Otherwise, these      points will be assigned to the coding and testing aspects of the work. Also see the above note about the items marked with (‡). | | | | | | | | | | | |

**4.2   Cross-Comparison of Demos and Star Ratings**

All demos will receive a “[star rating](http://en.wikipedia.org/wiki/Star_rating)”, from 1 to 10 stars. Keep in mind that the ratings are assigned *competitively*, i.e., one or more groups that are perceived as the best receive 10 stars and the others are scaled relatively to them.

These are key ingredients of a *competitive* demo:

* *Functionality / Task complexity*:  Emphasize the key functions of your system, what are they useful for. Show that this is a *non-trivial project* that required a team of smart students working for a full semester. (Functions such as *login* and *user registration* do not count!)
* *Novelty / Usefulness*:  Demonstrate that you have invented at least something, some part of your product, rather than plainly copying an existing idea.
* *Programming,****not****webpage design*:  You must convince us that *programming* has been *central* for your project development and webpage design has only been *ancillary*. In addition to regular programming languages (C, C++, Java, C#), programming includes dynamic websites and webpages using PHP, JSP, JavaScript, AJAX, etc.
* *Software reliability*:  “Software reliability” means that your software runs without crashing and, generally, it does not display unintended behavior or bugs.
* *Presentation quality*:  Impress the audience that you have developed a useful, non-trivial product, that only a highly trained software engineer can do.
* *User Interface / Ease of use*:  “Ease-of-use of the user interface” should not be confused with a flashy interface. On the contrary, you should avoid flashy user interfaces. “Ease-of-use” means that interface is easy to understand and operate and allows the user to accomplish the task with a minimum effort (see here about [estimating user effort](http://www.ece.rutgers.edu/~marsic/Teaching/SE/report1-appA.html)). A minimal user interface that is well organized should be sufficient.

After this step, the student grades will be calculated as detailed [here](http://www.ece.rutgers.edu/~marsic/Teaching/SE/projects.html#GRADING).

**5.   Demo Documents Submission**

Using a [file sharing website](http://www.ece.rutgers.edu/~marsic/Teaching/SE/projects.html#RESOURCES), share the folders as below. If any item is not applicable to your project, state and explain in individual\_contributions.pdf below. Also propose how the points from non-applicable items should be allocated to other items.

The main folder should contain the following subfolders (see more details about each item [**above**](http://www.ece.rutgers.edu/~marsic/Teaching/SE/demo1.html#PARTS)):

1. 1\_code  
   includes all your code with comments and README1.txt on how to run the code (see about [README](http://en.wikipedia.org/wiki/README) on Wikipedia)
2. 2\_unit\_testing  
   program code to run unit tests and README2.txt on how to run the unit tests
3. 3\_integration\_testing  
   program code to run unit tests and README3.txt on how to run the integration tests
4. 4\_data\_collection  
   scripts or programs for data collection and README4.txt on how to run the data collection scripts
5. 5\_documentation  
   includes the following 5 separate documents with cover page indicating what document it is, project title, group number, group members. Include the table of contents wherever appropriate (**PDF only!**):
   1. technical\_documentation.pdf
   2. user\_documentation.pdf
   3. brochure/flyer.pdf
   4. presentation\_slides.pdf     (acceptable formats: .pdf or .ppt)
   5. individual\_contributions.pdf     (this document should include the project management information, item 12 in Section 4.1 of this webpage)

Upload your files on a file sharing website such as [GitHub](https://github.com/) or [Dropbox](https://www.dropbox.com/) because sometimes the mail server removes code file attachments. Make them public, so that we should not need to login to access your files or search for them.

Submission deadline: The documentation for the demo is due **no later than two (2) days after your demo presentation**.

**BACK TO:**

* [Course home page](http://www.ece.rutgers.edu/~marsic/Teaching/SE/index.html)
* [Course syllabus page](http://www.ece.rutgers.edu/~marsic/Teaching/SE/syllabus.html)

[*Ivan Marsic*](http://www.ece.rutgers.edu/~marsic/) *Thu Feb 16 19:20:48 EST 2012*