CIS 4398 Project in Computer Science

Course:	CIS 4389
Course Title:	Project in Computer Science
Time:	WF 2:00 – 3:20 M 2:00 – 3:50
Place:	WF TL305A (class) W CC200 (lab)
Instructor:	Paul Wolfgang
Instructor Phone:	215-204-5155
Office Hours:	M 10:00 – 10:50 4:00 – 5:00, W 10:00 – 10:50, F 10:00 – 10:50
Course Web Page:	www.cis.temple.edu/~wolfgang
Prerequisites:	C- or better in: CIS 3238 Software Design and Senior Standing
Textbooks:	None assigned Recommended Reference: *Pro Git* available at http://git-scm.com/book
Course Description	Team-oriented design and implementation of a large programming project. Students will propose topics for review and acceptance early in the semester. Students will provide written documentation of their completed projects and will demonstrate the operation of their completed projects in an oral presentation.
Course Goals:	Students will be organized into project teams ranging in size from 3 to 6. The project will follow a modified version of the Rational Unified Process known as the Unified Process for Education. • Inception – defines the scope of the project • Elaboration – design product(s) and plans the project • Construction – builds the product(s) The end of each phase is a Milestone with specified documentation and other artifacts.
Grading:	The artifacts at the end of phase will be graded. Students will have an opportunity to revise draft documentation prior to final grading. THIS IS A WRITING INTENSIVE COURSE. 50% of the grade will be based on the technical content and quality of writing.
Exam Dates:	There are no exams. Final project presentations will be made in lieu of a final exam during the final exam week.
Attendance Policy:	Attendance is mandatory. Unexcused absence will result in reduction of final grade.

Schedule

There are 14 weeks in the semester.

Inception	3 weeks	September 16, 2013
Elaboration	5 weeks	October 21, 2013
Construction	6 weeks	November 25, 2013

Other Important Information

Disability disclosure:

Any student who has a need for accommodation based on the impact of a disability should contact me privately to discuss the specific situation as soon as possible. Contact Disability Resources and Services at 215-204-1280 in 100 Ritter Annex to coordinate reasonable accommodations for students with documented disabilities. (Temple University Policy and Procedures Manual)

Academic freedom:

Freedom to teach and freedom to learn are inseparable facets of academic freedom. The University has a policy on Student and Faculty and Academic Rights and Responsibilities (Policy #03.70.02) which can be accessed through the following link: *http://policies.temple.edu/getdoc.asp?policy_no=03.70.02*.

Academic Honesty

Academic cheating (such as plagiarism, copying during an exam, copying homework, stealing files and passwords, etc.) is strictly prohibited in this course. The penalty for the first offense will normally be an F in the course. A subsequent offense (in this or any other course) may also be referred to the University Disciplinary Committee.

No collusion what-so-ever during an exam will be tolerated. In particular not talking or other sharing of information (for example during open book exams) is permitted. Keep your eyes on YOUR paper.

IGNORANCE OF ACCEPTABLE GUIDELINES OF CONDUCT IS NO EXCUSE

http://policies.temple.edu/getdoc.asp?policy_no=03.70.12

Dates to Remember

First Day of Class:	Monday, August 26, 2013
Last Day to Drop:	Monday, September 9, 2013
Last Day to Withdraw[*]:	Tuesday, October 22, 2013
Last Day of Class:	Wednesday, December 4, 2013
Final Presentations	Monday, December 9, 2013 1:00 – 3:00

[*] Students may withdraw from a course only once, and repeat a course at most twice. (Taking a course for the third time requires the Dean's permission. See: http://policies.temple.edu/getdoc.asp?policy_no=02.10.12

Weekly Status Reports

Team Status Reports

Each team will prepare and present a weekly status report. This will be a Power Point presentation that contains the following slides:

- Planned Accomplishments for the Week
- Actual Accomplishments for the Week
- Planned Accomplishments for the following Week
- Earned-Value chart
- Issues/Discussion

Individual Status Reports

Each team member will submit a weekly status report to the team and the instructor. This will be a written report that addresses the following:

- Planned accomplishments for the week
- Actual accomplishments for the week
- Planned accomplishments for the following week
- Time spend
    - Time per task as identified on the project plan
    - Additional time not allocated to tasks on the project plan
- Issues/Discussion

Documentation Artifacts

Documentation by Phase

The documentation is expected to be developed over the life of the project, but certain documents are required to be initially delivered (I), finalized (F), or revised/updated (R) at the end of particular phases as shown in the following table:

	Phase		
Document	Inception	Elaboration	Construction
Project Proposal	F		
Software Development Plan	I	F	
Requirements Specification	I	F	R
Users Manual		I	F
Design Document (Part I)*		F	R
Design Document (Part II)*		I	F
Test Procedures		I	F
Test Report			F

* A unified design document may be submitted.

General Requirements

Each document will contain:

Title Page: showing the name of the document, authors, release date, and revision number.

Table of Contents

System overview (copy/adapt the Project Abstract)

Document overview: summarize the purpose and contents of the document

Body: the content of this document

References to other documents

Glossary if required

Each page will include a page number, document title, release date, and revision number in the header or footer.

Project Proposal

Purpose

The Project Proposal provides an initial description of the project's goals. It offers:

- a flavor of what the application will accomplish
- why the user should use this application
- why this application and approach are superior to the competition's efforts

Requirements

The Project Proposal will consist of the following.

Project Abstract

The Project Abstract will be at least ½ page, but not more than one page long. A page is defined as an 8½" × 11" with 1" margins containing 11-point single-spaced text. The Project Abstract will contain.

- A top-level description of the requirements.
- A conceptual design.
- Reference to similar products.

The abstract will **not** contain phrases such as:

- We are a team from Temple University …
- This application fulfills a class requirement …
- I am doing this because …
- Our team has decided to …

Examples can be found on software websites, advertising brochures, and in the introduction section of many manuals. Here is a short example:

> *TestItAll sets a new standard for automating software testing and development. A series of user defined menus will enable testers to create easily regression suites without the need to learn a proprietary programming language. And on, and on, and on, for at least half a page of text.*

Note that the Project Abstract will be placed on the project development web page/wiki, and repeated as the introduction to all other documents associated with the project.

Optional, but strongly suggested, a project logo no more than 32 pixels high and no more than 128 pixels wide.

Top Level Requirements

Describe the requirements – i.e., what the product does and how it does it from a user point of view – at a high level.

Conceptual Design
Describe the initial design concept: Hardware/software architecture, programming language, operating system, etc.

Background
The background will contain a more detailed description of the product and a comparison to existing similar projects/products. A literature search should be conducted and the results listed. Proper citation of sources is required. If there are similar open-source products, you should state whether existing source will be used and to what extent. If there are similar closed-source/proprietary products, you should state how the proposed product will be similar and different.

Required Resources
Discuss what you need to develop this project. This includes background information you will need to acquire, hardware resources, and software resources. If these are not part of the standard Computer Science Department lab resources, these must be identified early and discussed with the instructor.

Software Development Plan

Purpose

The Software Development Plan describes the activities and tasks to be performed to develop the software product.

Requirements

In addition to the general requirements the Software Development Plan will contain the following sections:

Activities	E.G. requirements gathering, top-level design, detailed design, test.
Tasks.	A task is the performance of an activity leading to a specific product. E.G. Design of unit x. Associated with each task is • predecessor tasks (what tasks must be complete before this task can start) • an estimated effort • estimated finish data • responsible individual • successor tasks (what tasks cannon start until this task is complete)
Schedule	A graphical layout of the tasks in the form of a Gantt Chart.
Development Environment	The required hardware and software to be used to develop the project. This includes the selected IDE, compilers, editors, test tools, etc.
Version Control	The selected version control tool and procedures for maintaining a defined master configuration.

Requirements Specification

Purpose

The Requirements Specification defines the functional and non-functional requirements for the product.

Requirements

In addition to the general requirements the Requirements Specification will include the following:

> Use-case diagram or some other diagram that identifies the external interfaces. External interfaces include the user and external hardware or software.
>
> Use-case descriptions:
>
> > For each use-case define the triggering event and the interactions between the actor and the system. Normal and alternate flows should be described. State Diagrams may be useful in some cases.
> >
> > As an alternative to use-cases, user stories may be used, provided they are accompanied by scenarios that can form the basis for acceptance testing.
>
> Non-functional requirements:
>
> > Any constraints on the system. E.G. minimum/maximum response time. Minimum/maximum memory, etc.

User’s Manual

Purpose

The User’s Manual describes how to use and maintain the system. It can be presented as two documents, one for the end-user and the other for the maintainer.

Requirements

In addition to the general documentation requirements the User’s Manual will contain

Quick Start Guide	For the experienced user.
Installation	Basic installation, network considerations, uninstalls. Installation should be automated and seamless. The system must be responsible for determining if minimum requirements are satisfied. For a client/server system this includes installation of both the client and server software. For a mobile application, it includes build and deployment instructions.
Configuration	Single user, multi-user, network, resources. Automation is again the key to successful configuration.
Security	Passwords.
Database	Installation and maintenance if required.
Application Functions	System functionality, screen shots. Step-by-step operating procedures.
Backup and Recovery	
Error Messages	Messages and actions. Don’t leave people hanging. When an error is reported, there must be a corrective action.
Troubleshooting	Troubleshoot the application – not the operating system or network.
Support	Contacts, contracts

Design Document (Part I)

Purpose

The Design Document (Part I) describes the software architecture and how the requirements are mapped into the design. This document will be a combination of diagrams and text that describes what the diagrams are showing.

Requirements

In addition to the general requirements the Design Document (Part I) will contain:

A description the different components and their interfaces. For example: client, server, database.

For each component provide class diagrams showing the classes to be developed (or used) and their relationship.

Sequence diagrams showing the message flow for each use-case (or user story).

State diagrams for classes that have states.

If there is a database:

Entity-relation diagram.

Table design.

Design Document (Part II)

Purpose

The Design Document gives the complete design of the application.

Requirements

In addition to the general documentation requirements the Design Document will contain

Updated copy of Part I

For each class define the data fields, methods.

- The purpose of the class.
- The purpose of each data field.
- The purpose of each method
  - Pre-conditions if any.
  - Post-conditions if any.
  - Parameters
  - Return value
  - Exceptions thrown*.

This information should be in structured comments (e.g. Javadoc) in the source files. A documentation generation tool (e.g. Javadoc) should be used to generate the document.

* At the top level, or where appropriate, all exceptions should be caught and a error message that is meaningful to the user generated. It is not OK to say ("xxxx has encountered a problem and will now close (OK?)" Error messages and recovery procedures should be documented in the User’s Manual.

Test Procedures

Purpose

The Test Procedures describe the test approach and the tests to be performed.

Requirements

In addition to the general documentation requirements this document will contain the procedures for the following tests:

Unit tests

For each method, one or more test cases.

A test case consists of input parameter values and expected results.

All external classes should be stubbed using mock objects.

Integration tests

Tests to demonstrate each use-case based on the use-case descriptions and the sequence diagrams. External input should be provided via mock objects and results verified via mock objects. Integration tests should not require manual entry of data nor require manual interpretation of results.

Acceptance test

Demonstration of all of the functional and non-functional requirements. This can be a combination of automated tests derived from the use-cases (user stories) and manual tests with recorded observation of the results.

Test Report

Purpose

The Test Report is a record that the tests were run and a documentation of their results.

Requirements

In addition to the general documentation requirements this document will contain:

Output from the unit test runs.

Output from the integration test runs.

Output from automated acceptance tests.

A copy of the manual acceptance test procedures with notations indicating that the test was performed and the observed results.

List of known problems: describe each failed test.