San José State University
Department of Computer Engineering

CMPE/SE 131
Software Engineering I

Section 3
Spring 2017

GREEN SHEET

CMPE/SE 131-03 TuTh 9:00 - 10:15 PM ENG 337

Instructor: Ron Mak
Office hours: Th: 2:30 – 4:30 PM
Office location: ENG 250
E-mail: ron.mak@sjsu.edu
Instructor web page: http://www.cs.sjsu.edu/~mak/
Class web page: http://www.cs.sjsu.edu/~mak/CMPE131/index.html

Course catalog description

"Why software engineering? What is software engineering? Software development lifecycle activities: project planning and management requirements analysis, requirement specification. Software design, software testing, verification, validation, and documentation. Software quality assurance and review techniques, software maintenance, team-based projects Prerequisite: For CMPE Major: CMPE126 with a grade of "C-" or better. Allowed Declared Majors: Any Engineering For SE Majors: CS 046B with a grade of "C-" or better." *3 units*

Please submit into Canvas a copy of your transcript with the prerequisite courses highlighted. "*Students who do not provide documentation of having satisfied the class prerequisite and co-requisite requirements (if any) by the second class meeting will be dropped from the class."*

Course goals

Become familiar with **industry standard processes and practices** to develop a software product.

The instructor will share decades of experience working as a software developer and project manager in industry, government, and scientific research institutions.

This is a challenging course that will demand much of your time and effort throughout the semester.

Course learning outcomes

CLO 1: **Software process:** Reason about and apply the entire software development process. Create a software project schedule and use project scheduling like Microsoft Project. Use version control tools like Git.
CLO 2: **Requirements engineering:** Solicit, elaborate, and validate software product specifications and generate meaningful use cases.
CLO 3: **Software design:** Understand what software design architectures are suitable for various software projects. Apply appropriate software designs to a team project. Explain and defend design decisions. Use appropriate software design tools.
CLO 4: **Software verification and validation (V&V):** Understand the software validation process and use issue-tracking tools. Create and execute test plans.

Acquire **critical job skills** that are immediately applicable in the software industry:

- Understand and apply software engineering tools and methodologies.
- Work together in a small programming team.
- Recognize people and team management issues.
- Solicit and analyze product requirements and generate use cases.
- Write functional specifications at a level of detail sufficient for software design.
- Make correct architectural and design choices.
- Develop and document a software design at a level of detail sufficient for implementation.
- Draw meaningful UML class and sequence diagrams.
- Do oral presentations to explain and defend design decisions.
- Generate a project schedule with achievable milestones.
- Create Gantt charts from a work breakdown structure.
- Identify task dependencies and critical paths.
- Perform resource management to balance workloads.
- Track issues and measure development progress.
- Use revision control software.
- Formulate and execute a test plan.
- Carry out formal code reviews in a workplace setting.
- Automate the build and deploy process for a software project.

You will work together on a team to **build and deploy a web application** using the full-stack **Ruby on Rails** framework, which supports important client-server concepts:

- Model-view-controller (MVC) architecture
- Representational State Transfer (REST) architecture
- Object-relational mapping (ORM) and the active record design pattern

This is a software engineering class, not a web programming class. The Ruby on Rails project is a way for you to immediately apply the software engineering tools and methodologies as you learn them. Your grade depends on how well you do software engineering. How well your web application turns out should be a result of how successfully you applied the tools and methodologies.

Required texts

Title: Author: Publisher: ISBN-13	**Rails Crash Course: A No-Nonsense Guide to Rails Development** Anthony Lewis No Starch Press, 2015 978-1593275723
Title: Author: Free online:	**Ruby on Rails Quick Start Guide** (Google doc) Melvin Ch'ng https://goo.gl/uRdNhF

Recommended texts

Title: Author: Publisher: ISBN-13: Free online:	**Ruby on Rails Tutorial, 4th edition** Michael Hartl Addison-Wesley Professional, 2016 978-0134598628 https://www.railstutorial.org/book/
Title: Author: Publisher: ISBN-13	**Beginning Software Engineering** Rod Stephens Wrox/Wiley, 2015 978-1118969144

The Hartl book (available for free online) contains much useful Rails code that you can use to start your project. However, your project must go beyond the tutorials in this book and not simply consist of code cut and pasted from the book.

Software to install

As described in **Rails Crash Course**, you will need to download and install on your Mac, Windows, or Linux platform the following software: Ruby, Rails, Git, Heroku.

The book's **Windows installation instructions** for Ruby and Rails are obsolete.
Use instead these instructions prepared by Melvin Ch'ng:
http://www.cs.sjsu.edu/~mak/CMPE131/RoR_Rev1.0.2_16102016.pdf

Download and install either of the following:

- SQLite Studio:
  http://sqlitestudio.pl/?act=download
- SQLite Manager (Firefox add-on):
  https://addons.mozilla.org/en-US/firefox/addon/sqlite-manager/

Download and install:

- GanttProject:
  http://www.ganttproject.biz/

There may be other software packages announced during the semester.

Course website

Course materials such as syllabus, handouts, notes, assignment instructions, etc. can be found on the class web page at http://www.cs.sjsu.edu/~mak/CMPE131/index.html and on the Canvas Learning Management System course website at http://sjsu.instructure.com. You are responsible for regularly checking these sites to learn of any updates.

Course requirements and assignments

You will form project teams of four students each. **Team membership is mandatory for this class.** During the semester, each team will experience a realistic industry-level software development project by participating in the various development activities and creating **project artifacts**, including:

- Requirements specification
- Functional specification
- Design document
- Test plan
- Project schedule

Each team will also give several **oral presentations** to the class in conjunction with the project artifacts, including:

- Product pitch
- Design review
- Code review
- Product demo

During an oral presentation, the members of the rest of the class will play the roles of project advisors and potential customers for the presenting team. *Class attendance is especially important during oral presentation days.*

Team members will assume various **development team member roles**, such as:

- Project lead
- Chief architect
- User interface developer
- Server logic developer
- Database developer/administrator
- Software quality assurance engineer
- Documentation writer

Remember that the goal of this course is not to produce a winning web product in one but to learn team-based software engineering tools and methodologies.

The assignments during the semester will be to create the various project artifacts. As described above, some of the assignments will include oral presentations. A project team turns in one copy of each artifact or gives each presentation, and *all members of the team will each receive the same score for each assignment*.

Each assignment will be worth up to 100 points. Late assignments will lose 20 points and an additional 20 points for each 24 hours after the due date.

Team projects

A major portion of the coursework will be the team projects. Each project will provide opportunities for team members to apply immediately the material taught in the lectures and in the textbooks. *Choose your teams wisely!* Once teams are formed, students may not move from one team to another.

Each team will choose its own web application to develop, but it must fit the architectural framework described above. Each team must use the assigned project management tools and track its own progress.

At the end of the semester, *all the members of a project team will each receive the same project score*. The project grade will be determined by the overall quality of the final version of the project team's artifacts and by how well the team achieved its goals to create a successful web application.

Postmortem report

At the end of the semester, each student must also turn in a short (1 or 2 pages) **individual postmortem report** that includes:

- A brief description of what you learned in the course.
- An assessment of your accomplishments for your project team on the assignments and the web application project.
- An assessment of each of your other project team members.

Only the instructor will see these reports.

Exams

The midterm and final examinations will be closed book. Instant messaging, e-mails, texting, tweeting, file sharing, or any other forms of communication with anyone else during the exams will be strictly forbidden.

There can be no make-up midterm examination unless there is a documented medical emergency. Make-up final examinations are available only under conditions dictated by University regulations.

Class grade

Your individual class grade will be weighted as follows:

30%	Assignments*
35%	Project*
15%	Midterm exam**
20%	Final exam**

* *project team scores*
** *individual scores*

During the semester, you can keep track of your progress in Canvas. Each assignment and exam will be scored (given points) but not assigned a letter grade. The average score can be seen in Canvas after each assignment and the midterm exam.

At the end of the semester, all the students will be ranked in the order of their weighted class scores. Students with the median score will be assigned the B grade. Higher and lower grades will then be assigned based on how the scores cluster above and below the median.

Your final class grade can be adjusted up or down depending on your level and quality of participation on your project team as determined by the project tracking tools and your team members' assessments of your performance.

Classroom protocol

It is very important for each student to attend classes and to participate. Cell phones in silent mode, please.

University policies

Per University Policy S16-9, university-wide policy information relevant to all courses, such as academic integrity, accommodations, etc. will be available on Office of Graduate and Undergraduate Programs' Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/"

Schedule

Subject to change with fair notice. Chapter readings are from **Rails Crash Course**.

Each project team will make several oral presentations to the rest of the class.
Brevity will be crucial!

Week	Dates	Topics and activities	Chapters
1	Jan 26	What is software engineering? The team projects Web application architecture *Form project teams*	
2	Jan 31 Feb 2	Ruby fundamentals Rails fundamentals The model-view-controller (MVC) architecture Object-relational mapping (ORM) • The active record design pattern	1, 2
3	Feb 7, 9	Models Controllers Views	3, 4, 5
4	Feb 14, 16	Version control with Git Application deployment with Heroku Document management Project phases Agile software development Requirements • Gathering • Functional • Non-functional Use cases Bank ATM example Functional specification	6
5	Feb 21, 23	Conceptual design Example conceptual design presentation UML diagrams • Class diagram • Sequence diagrams Software architecture Framework classes Coupling and cohesion	
6	Feb 28 Mar 2	*Conceptual design reviews and prototype demos*	

7	Mar 7, 9	Data independence Relational databases Conceptual data model • Entity-relationship (ER) diagrams Logical data model • Relational schemas Normalization	
8	Mar 14, 16	Structured Query Language (SQL) Queries Data access layer Connection pools ***Midterm exam Thursday, March 16***	
9	Mar 21, 23	Bug tracking Project management Estimations, estimates, targets, commitments Project schedules • Gantt chart • Critical path • Resource chart • PERT chart Application design review preparation	
SPRING BREAK			
10	Apr 4, 6	*Application design reviews*	
11	Apr 11, 13	Usability UI design for web applications Software quality assurance (SQA) Verification and validation (V&V) Testing • Types of testing • Test-driven development (TDD) Test plans	
12	Apr 18, 20	Code reviews Code review preparation Software metrics Project failures Software engineering code	
13	Apr 25, 27	*Code reviews*	
14	May 2, 4	Risk management Personnel management NASA project do's and don'ts Extreme Programming (XP)	

15	May 9, 11	The Rational Unified Process (RUP) The software profession Software ethics and moral responsibilities *Application demos*	
16	May 16	*Application demos*	
	May 17	***Final projects due Wednesday, May 17***	
Final	May 23 **Section 3**	***Final exam Tuesday, May 23*** 7:15 – 9:30 **AM**, ENG 337	