Course Overview

SWEN-261
Introduction to Software Engineering

Department of Software Engineering
Rochester Institute of Technology

There are a number of administrative tasks that we have to take care of first.

- Who is your instructor?
- Who are you?
- You each have an SE account which you will need to use to access the lab and teamroom computers.
- Can you login? To log into the Lab machines you should now be able to do this **using your RIT account**. If you previously had an SE account you should have your information accessible. Any problems email gccisit@rit.edu.

Information about this course is available in a number of locations.

Course Website

http://www.se.rit.edu/~swen-261

Make sure to pick your section!

- **Syllabus**
- **Schedule**
- **Topic Pages**
  - References
  - Exercise requirements
- **Project**
  - Requirements
  - Resources
  - Sprint release requirements
- **General Resources**

myCourses

- **Assignments**
  - Exercise and project submissions
- **"Quizzes"**
  - Course topic exercises
- **Discussion forum topics**
  - Project clarifications
  - Misc. topic discussions
  - Lecture requests
  - Extra credit!
- **Section-specific Content**
  - Peer evaluations
  - Exercise content

Final Grading Breakdown

Component	Weight of 100%
Exam 1	10%
Final Exam	25%
Term Project and Team Activities	43%*
Individual Exercises [including possible Take-home(s)]	22%
After 1st Unexcused Absence	-2%
After 1st Late Arrival per late arrival	-1%

Note: Your instructor may modify the above and/or make individual adjustments to the term project grade in either direction

* Some team activities are submitted separately from the term project.

The exercises prepare you for class discussions and the exams test your deeper understanding.

- Exercises
  - ***Generally 0 or 1 with minimal partial credit.***
  - ***Some multi-point exercises for more involved activities.***
  - ***Assess basic understanding of core concepts/terminology.***

- Exams
  - ***Assess deeper understanding and ability to apply course concepts.***
  - ***Generally short answer and application activities (diagrams, examples, etc.)***
  - ***NOT about memorization.***

There will be a term project that runs through the entire term.

- The same team of 4+/- students will work on this project for the term.
- There will be several deliverables through the term.
- You will receive a term project grade adjusted based on your **INDIVIDUAL** contributions to the project work.
  - ***Artifact submission and quality***
  - ***Instructor observation***
  - ***Peer evaluations***

*** Regarding Peer Feedback**

The integrity and quality of this course relies on open, honest peer feedback.

This means that students should not, under any circumstances, try to influence their peers into changing feedback.

Furthermore, students should not discuss the feedback that their peers gave them except to ask for suggestions about how to improve.

Doing otherwise is considered a violation of academic honesty (an attempt to influence your own grade through means other than your own work) and will be handled as such.

The term-long project will be a web-based software system.

- Angular – component-based web application framework
- Spring – framework for supporting REST API services
- VSCode – multi-platform integrated development environment
- Tutorials and exercises prior to starting project work
  - ***Typescript development for Angular UI components***
    - A little HTML/CSS
  - ***Server-side Java development for REST API***
  - ***No database use***
- Project work delivered in five sprints

There are rubrics that define the grading for all major elements of the term project work.

Exceptional (100)	“unusually excellent; superior”
Competent (88)	“having suitable or sufficient skill, knowledge, experience, etc., for some purpose; properly qualified”
Acceptable (75)	“capable or worthy of being accepted”
Developing (50)	“undergoing development; growing; evolving”
Unacceptable (0)	*Antonym of Acceptable*

*Definitions from http://www.dictionary.com

What takes place in each class will vary by topic.

- Classes will be made up of
  - ***Lectures***
  - ***Exercises – individual, project team, transient group***
  - ***Project time***
  - ***Demonstrations and presentations***
  - ***Exams*** 😕

Your team ***will*** **need** to spend significant time on the term project outside of class.

You should consult the course schedule every day to stay on top of the work and studying you have.

- Many course topics will have work that you will have to do outside of class.
  - ***Pre-class exercise***
    - Individual study
    - Verified with a quiz due at the start of the class session when the topic will be covered.
  - ***After-class***
    - Individual or project team
    - Introduced in a class session before it is due
    - Due at the start of a subsequent class session
  - ***Project work***
    - Most of the project work will be done outside of class time

It is up to *you* to remember what artifacts are due at the start of class.

Most days we will immediately launch into a discussion based on the before-class exercises due at the start of class.

There is simply not enough time in the schedule to offer individual extensions on before-class exercises and/or make the class wait while you rush to finish at the start of class.

You will do many class exercises in this course with the first QUIZ right now (5 mins)

- *What is software engineering?*
  - ***Take the quiz in myCourses***
  - ***What is software engineering?***
  - ***What do you expect to learn from this Introduction to Software Engineering course?***

Definitions of software engineering refer to systematic, develop/implement, scale and quality.

Software engineering is the application of engineering to the development of software in a systematic method. – Wikipedia

The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. – IEEE

Software engineering is concerned with developing and maintaining software systems that behave reliably and efficiently, are affordable to develop and maintain, and satisfy all the requirements that customers have defined for them. – ACM

A branch of computer science that deals with the design, implementation, and maintenance of complex computer programs – Merriam-Webster

This course is not a broad, thin overview of all things software engineering.

- Content was chosen by answering the question:
  What is most important for a student to learn in a first (and for many students only) software engineering course?
- Guidelines used for course content
  - ***Cover a limited set of topics in more depth***
  - ***Make sure that there is an obvious connection between the lecture and project***
  - ***Use up-to-date tools and techniques***

This course is about the principles and practices in the daily activities of a software engineer.

- Software Design **35%**
  - *Object-oriented design above the class level*
  - *Class interaction/behavioral design*
  - *Software architecture*
  - *Introduction to web application design*
- Software Process **35%**
  - *Requirements*
  - *Product backlog workflow*
  - *Code repository workflow*
  - *Planning, retrospectives*
  - *Software quality*
- Teamwork **15%**
  - *Personality types*
  - *Team formation*
  - *Task assignment*
  - *Team mentoring*
- Communication **15%**
  - *Code level*
  - *Design documentation*
  - *Demos and presentations*
  - *Tool support: Trello, Slack, GitHub*