GINA CODY
SCHOOL OF ENGINEERING
AND COMPUTER SCIENCE

Course number	Course Title	Term
SOEN 390	Software Engineering Team Design Project	Winter 2025

Course Instructor	Office	E-Mail	Office Hours
Nikolaos Tsantalis	ER 11.27	nikolaos.tsantalis@concordia.ca	

General information

1. Academic Integrity

Violation of the Academic Code of Conduct in any form will be severely dealt with. This includes copying (even with modifications) of program segments. You must demonstrate independent thought through your submitted work. The Academic Code of Conduct of Concordia University is available at: https://www.concordia.ca/conduct/academic-integrity.html

It is expected that during class discussions and in your written assignments you will communicate constructively and respectfully. Sexist, racist, homophobic, ageist, and ablest expressions will not be tolerated.

All students must read and sign the Expectations of Originality form and submit the signed copy to course instructor by September 14, 2020

2. Third-party software/website and personal information

Note that, as a part of this course, some or all of the lectures and/or other activities in this course may be recorded. Recordings will be focused on the instructor and will normally exclude students. It is possible, however, that your participation may be recorded. If you wish to ensure that your image is not recorded, speak to your instructor as soon as possible.
Also, please note that you may not share recordings of your classes and that the instructor will only share class recordings for the purpose of course delivery and development. Any other sharing may be in violation of the law and applicable University policies, and may be subject to penalties.

3. Third-party software/website usage for work submission

Students are advised that external software and/or websites will be used in the course and students may be asked to submit or consent to the submission of their work to an online service. Students are responsible for reading and deciding whether or not to agree to any applicable terms of use. Use of this software and service is voluntary. Students who do not consent to the use the software or service should identify themselves to the course instructor as soon as possible to discuss alternate modes of participation that do not require them to give copyright or the right to use their work to a third party.
By using the external software or websites, students agree to provide and share their work and certain personal information (where applicable) with the website/software provider. Students are advised that the University cannot guarantee the protection of intellectual property rights or personal information provided to any website or software company. Intellectual property and personal information held in foreign jurisdictions are subject to the laws of such jurisdictions.

4. **You must successfully complete "Safety Module 2: Risk Analysis Techniques in Safety" training available in your Moodle and receive a PASS grade.**

CLASS, LAB AND TUTORIAL SCHEDULE					
Section	**Day**	**Time**	**Location**	**Instructor**	**E-mail**
Tutorials					
Q QA	M	17:45-18:35			
Q QC	M	19:15-20:05			
R SA	M	17:45-18:35			
R SC	M	19:15-20:05			
Labs will take place as meetings with your Product Owner/Mentor after the completion of each Sprint. You can schedule the meeting time in agreement with your TA.					

COURSE CALENDAR DESCRIPTION
Students work in teams to design and implement a software project from requirements provided by the coordinator. Each team will demonstrate the software and the testing of the software, and prepare adequate documentation for it. In addition, each team will generate a report based on the process of development.

PREREQUISITE
The following courses must be completed previously or concurrently: SOEN 345 and SOEN 357.

Iterations	Evaluation criteria	Weight
Release 1 **Jan 13 - Feb 9** Sprint #1 Jan 13 - Jan 26 Sprint #2 Jan 27 - Feb 9	-**Repository setup and Continuous Integration** (You should setup a GitHub repository and GitHub actions that automate your test executions, test code coverage with CodeCov): **2** points -**User Stories Backlog** (You should create user stories for **all project requirements**. Before the beginning of each Sprint you will plan the user stories to be completed and estimate their user story points using planning poker): **3** points -**Agile Planning** (Each release consists of 2 Sprints and each Sprint is 2-weeks long. For each Sprint you will include the Sprint summary, burndown chart and retrospective. All actions in the retrospective should be justified based on the "lessons learned" from the previous sprints. You must use an Agile planning software, such as ZenHub, which provides a Board and Agile metrics, and you must provide access to the instructor): **2** points -**Software Architecture** (Domain Model, Component Diagram, ER Model if needed, analysis of external libraries that will be used with a justification about their pros and cons): **4** points -**UI Prototypes** (Mockups linked to each user story of the Sprint, all mockups will be annotated to explain the functionality of buttons/inputs/outputs, Persona descriptions, UI variations based on the personas, Show updates to the UI mockups after receiving feedback from the product owner): **5** points -**Unit/System Testing Report** (Unit test coverage report for the current release, Record automated System/UI tests as shown in https://github.com/RGPosadas/WayFinder/wiki/Acceptance-Tests-GIFs): **3** points -**Issue Tracker** (Create bug reports with replication steps, screenshots, links to the commits fixing the bug for traceability, links to commits updating tests to cover the reported bug. All issues related to bugs should be tagged with "bug" label, issues related to performance should be tagged with "performance" label): **2** points -**Pull Requests and Code reviewing** (All features will be contributed in the form of pull requests. Each pull request will have assigned code reviewers and discussions between the reviewers and authors about potential improvements related to code/design quality): **3** points -**Code Quality and Security Report** (Software metrics [size, duplication, documentation,	30%

	complexity, coupling, cohesion] evolution, Report the technical debt, security vulnerabilities, and code convention violations for the current release. List the commits and pull requests in which technical debt, security vulnerabilities, and code convention violations have been addressed. The commits and pull requests should explicitly mention/discuss/motivate the type(s) of technical debt addressed). You can use SonarQube for automatically collecting metrics and technical debt: **3** points -**Presentation to the Instructor** (the content of the presentation will be announced before the release): **3** points	
Suggested Plan for Sprint #1 1. Setup your GitHub repository and yml scripts for GitHub workflows 2. Create your project Backlog in ZenHub (make a user story for each project requirement) 3. Define the **personas** representing potential users and describe some of the tasks they will perform on the system. You should consider how to make your software accessible to more people. People with disabilities (color-blind, blind, deaf, people with hand injuries), illiteral people, or marginalized people. 4. Create UI mockups for the user stories you will implement in Sprint 2 5. Think about and document the design of your app (basic domain model, component diagram). Explore external libraries/frameworks that might be useful for your project and discuss their pros and cons. 6. Plan your Sprint 2 in ZenHub by adding some user stories from your backlog (add features than can be completed within 2 weeks) 7. Conduct and document your Sprint 1 Retrospective. Meet with your product owner and get feedback for your mockups and your backlog		
Suggested Plan for Sprint #2 1. Implement the planned features 2. Write unit tests for your code 3. Submit your features in Pull Requests. Assign team members for code review. There should be active discussions and suggestions for improvements. 4. Create UI mockups for the user stories you will implement in Sprint 3 5. Investigate how you can make automated System/UI tests. In the meantime, some team members will do manual system testing for the implemented features and report well documented bug reports on GitHub as issues. 6. Install and run SonarQube for your project. Summarize the issues detected by SonarQube and discuss how and when you will address them. 7. Conduct and document your Sprint 2 Retrospective. Meet with your product owner for acceptance testing and getting feedback 8. Prepare your presentation for the Instructor and practice it with all team members at least once. 9. Submit your Release 1 report on Slack 10. Plan your Sprint 3 in ZenHub by adding user stories from your backlog (add features than can be completed within 2 weeks)		
Release 2 **Feb 10 - Mar 9** Sprint #3 Feb 10 - Feb 23 Sprint #4 Feb 24 - Mar 9	-**Agile Planning**: **2** points -**Software Architecture** (Discuss about the source code implementation of your design patterns and how you extended these patterns over time as new requirements were implemented): **2** points -**UI Prototypes**: **3** points -**Unit/System Testing Report**: **3** points -**Usability/Performance testing** (Develop code in your app or use tools for collecting user session recordings. Analyze the collected data to find ways to improve the UI experience): **5** points -**Issue Tracker**: **2** points -**Refactoring activity** (List the commits and pull requests in which refactoring took place. The commits and pull requests should explicitly mention/discuss/motivate the type(s) of refactoring. The complete list of refactoring types is available at https://refactoring.com/catalog/: **2** points -**Code reviewing**: **4** points -**Code Quality and Security Report**: **3** points -**Presentation to the Instructor**: **4** points	30%
Suggested plan for Sprint #3 1. Implement the planned features 2. Write unit tests for your code 3. Submit your features in Pull Requests. Assign team members for code review. There should be active discussions and suggestions for improvements. These suggestions could also include refactorings. Document systematically your refactoring efforts by collecting commits and PRs with specific types of refactorings 4. Create UI mockups for the user stories you will implement in Sprint 4 5. At this point you should have automated System/UI tests for all implemented features so far. You will do manual system testing for the ongoing features and report well documented bug reports on GitHub as issues.		

6. Investigate potential design patterns that could be useful for your project. Justify their need in your project. 7. Investigate how you will conduct usability testing and create the required infrastructure (setup and try usability testing tools). 8. Plan your Sprint 4 in ZenHub by adding some user stories from your backlog (add features than can be completed within 2 weeks. Note that in the next Sprint you will conduct usability testing. Make a realistic plan) 9. Conduct and document your Sprint 3 Retrospective. Meet with your product owner for acceptance testing and getting feedback		
Suggested plan for Sprint #4 1. Implement the planned features following the same practices (PRs, code reviews, unit tests) 2. Introduce design patterns in your project. Document commits where your design pattern instances were introduced and evolved (adding new abstract methods, adding new subtypes) 3. Address technical debt and security warnings from SonarQube. Update your SonarQube (code quality) report 4. Create UI mockups for the user stories you will implement in Release 3 (Sprints 5 and 6). 5. Add automated System/UI tests for the newly implemented features. 6. Recruit users and conduct your first usability testing. Collect and analyze your usability test results. Make well documented issues for UI improvements and bugs found during testing. 7. Conduct and document your Sprint 4 Retrospective. Meet with your product owner for acceptance testing and getting feedback 8. Prepare your presentation for the Instructor and practice it with all team members at least once. 9. Submit your Release 2 report on Slack 10. Plan your Sprint 5 in ZenHub by adding some user stories from your backlog (add features than can be completed within 2 weeks)		
Release 3 **Mar 10 - Apr 6** Sprint #5 Mar 10 - Mar 23 Sprint #6 Mar 24 - Apr 6	-**Agile Planning**: **2** points -**Unit/System Testing Report** (At this point all user stories should be automated): **4** points -**Usability/Performance testing** (Implement the improvement opportunities you found in the previous release and rerun your usability/performance tests to measure the improvements quantitatively): **5** points -**Issue Tracker** (At this point all open bugs, performance issues should be addressed): **3** points -**Refactoring activity**: **4** points -**Code reviewing**: **4** points -**Code Quality and Security Report** (At this point all technical debt, security vulnerabilities, code convention violations, and code smells should be addressed. If not addressed, you should explain the reasons): **4** points -**Presentation to the Instructor**: **4** points	30%
Suggested plan for Sprint #5 1. Address all UI issues discovered in usability testing 2. Implement the remaining planned features 3. Write unit tests for the new features 4. Submit your features in Pull Requests. Assign team members for code review. There should be active discussions and suggestions for improvements. These suggestions could also include refactorings. Document systematically your refactoring efforts by collecting commits and PRs with specific types of refactorings 5. Add automated System/UI tests for the newly implemented features. 6. Address technical debt and security warnings from SonarQube. Update your SonarQube (code quality) report 7. Plan your Sprint 6 in ZenHub by adding any remaining user stories from your backlog. 8. Conduct and document your Sprint 5 Retrospective. Meet with your product owner for acceptance testing and getting feedback		
Suggested plan for Sprint #6 1. Conduct a second round of usability testing. Compare the current usability test results with the previous ones to show the improvement in user satisfaction or other usability metrics. 2. Add automated System/UI tests for all implemented features. 3. Address all remaining technical debt and security warnings from SonarQube. Update your SonarQube (code quality) report 4. Conduct and document your Sprint 6 Retrospective. Meet with your product owner for any remaining acceptance testing 5. Prepare your presentation for the Instructor and practice it with all team members at least once. 6. Submit your Release 3 report on Slack		
Acceptance tests	Percentage of the number of user stories signed-off by the product owner through acceptance tests over the total number of user stories in the backlog. **For each user story there should be an acceptance test in the issue tracker of the project. The test is considered as accepted if it is signed-off with the account of the product owner by making an accept comment on the corresponding issue.**	10%

GRADING POLICY

Evaluation Tool	Weight
Release 1	30%
Release 2	30%
Release 3	30%
Completed project requirements	10%
Total	100%

Passing Criteria:

- Students with a consistent lack of contributions in the project will fail the course, regardless of the team's performance. Lack of contribution can be determined with repository analytics tools, and/or from teammate peer evaluations.

GRADUATE ATTRIBUTES: SKILLS TO LEARN AND/OR UTILIZE

As part of either the Computer Science or Software Engineering program curriculum, the content of this course includes material and exercises related to the teaching and evaluation of graduate attributes. Graduate attributes are skills that have been identified by the Canadian Engineering Accreditation Board (CEAB) and the Canadian Information Processing Society (CIPS) as being central to the formation of Engineers, computer scientists and information technology professionals. As such, the accreditation criteria for the Software Engineering and Computer Science programs dictate that graduate attributes are taught and evaluated as part of the courses. This particular course aims at teaching and evaluating 10 graduate attributes. The following is the list of graduate attributes covered in this course, along with a description of how these attributes are incorporated in the course.

Knowledge-base for Engineering: Demonstration of application of most of the knowledge and skills learned in all courses in the program in a project specified by the instructor, developing, modelling and documenting analysis and design, implementation and testing.
This attribute will be evaluated based on the quality of the final software product and the delivered report (essentially, this graduate attribute is assessed from the final grade for the course).

Problem analysis: Analyze and document the provided project requirements.
This attribute will be evaluated based on your final Requirements Document and the quality of the User Story Backlog.

Design: Develop and document architectural and detailed design for the adopted solution for the project.
This attribute will be evaluated based on your final Software Architecture Document.

Use of Engineering tools: Use appropriate tools to manage, analyze, design and implement the project.
This attribute will be evaluated based on the degree of utilization of software engineering tools (e.g., GitHub, ZenHub, Prototyping tools, SonarQube).

Team/individual work: Work in teams to design and implement a software project from requirements provided by the coordinator.
This attribute will be evaluated based on your Sprint Planning reports and Sprint Retrospectives.

Communication: Demonstrate the software and prepare adequate documentation for it.
This attribute will be evaluated from the quality of your project presentations in all iterations.

Economics/management: Project cost/effort estimation and tracking.
This attribute will be evaluated based on your Risk Management Plan and Test/Quality Reports.

Professionalism: The project requires professional interactions with the stakeholders and within the team.
This attribute will be evaluated based on the peer-evaluation forms.

Ethics and equity: Analyze the possible ethical aspects of the problem or its solution.
This attribute is not always applicable and its applicability depends on the nature of the project.
If your project deals with ethical concerns, you should discuss them in the Requirements document.

Investigation: Investigate on different proposed solutions for the project. Design elaborated testing using elaborated tools and techniques to prove important system qualities.
This attribute will be assessed based on the Feasibility Analysis performed for the new feature requested by the project owner close to the end of the project.

COURSE LEARNING OUTCOMES (CLOS)
By the end of this course students will be able to:

- Collect and document the requirements for a mobile application from a client (product owner).
- Design and implement a fully functional software.
- Design the user interface for a mobile application and test its usability. Improve the usability based on client's feedback.
- Work in small teams following the agile software development process (6 sprints and 3 releases in total, team should deliver a functional software at the end of each release).
- Assess and manage the risks of the project.
- Plan the tasks to be completed before the beginning of each sprint.
- Assess the success of each sprint after its end, and take actions to improve (retrospective).
- Test the functionality of the software (unit, system, integration testing).
- Assess the design and code quality of the software and refactor the code to improve quality.
- Use software engineering tools for project management (e.g., ZenHub), source code management and bug tracking (e.g., GitHub, Bitbucket), testing (e.g., JUnit, Espresso), source code development and refactoring (e.g., Eclipse, IntelliJ IDEs), quality monitoring (e.g., SonarQube), UI prototyping (e.g., Moqups).

ON CAMPUS RESOURCES

HEALTH SERVICES An on-campus health clinic and health promotion center with nurses and doctors. SGW 514-848-2424 ext. 3565 LOY 514-848-2424 ext. 3575	**COUNSELLING AND PSYCHOLOGICAL SERVICES** Counsellors (licensed mental health professionals) work with students to address their mental health and wellbeing needs. SGW 514-848-2424 ext. 3545 LOY 514 848-2424 ext. 3555
ACCESS CENTRE FOR STUDENTS WITH DISABILITIES Supports students with a variety of disability conditions (including temporary disabilities arising from illness or injury). Students receive academic support for their educational experience at Concordia. acsdinfo@concordia.ca 514-848-2424 ext. 3525	**SEXUAL ASSAULT RESOURCE CENTRE** Provides confidential and non-judgemental support and services to students, staff and faculty of all genders and orientations affected by sexual violence and/or harassment. Jennifer Drummond, Coordinator jennifer.drummond@concordia.ca sarc@concordia.ca 514-848-2424 ext. 3353
STUDENT SUCCESS CENTRE Support network from first-year to graduation. You'll find one-on-one tutors, study groups, workshops as well as learning and career advisors 514-848-2424, ext. 3921	**DEAN OF STUDENTS** Supports students to enhance their Concordia experience by engaging in student life outside the classroom. Terry Kyle, Manager deanofstudents.office@concordia.ca SGW 514-848-2424 ext. 3517 LOY 514-848-2424 ext. 4239
ABORIGINAL STUDENT RESOURCE CENTRE An on-campus resource for First Nations, Métis and Inuit students that helps them make the most of the many resources available at the university. Orenda Konwawennotion Boucher-Curotte, Coordinator orenda.boucher@concordia.ca 514-848-2424 ext. 7327	**INTERNATIONAL STUDENTS OFFICE** Supporting international students with immigration documents, health insurance, social events, and workshops. iso@concordia.ca 514-848-2424 ext. 3515
STUDENT ADVOCACY OFFICE Advocating for students facing charges under the Academic Code of Conduct or the Code of Rights and Responsibilities. studentadvocates@concordia.ca 514-848-2424, ext. 3992	**MULTI-FAITH & SPIRITUALITY CENTRE** Provides a home for all those wishing to celebrate the human spirit in the widest sense of the word, through programs, events and a quiet space for reflection. Ellie Hummel, Coordinator mfsc@concordia.ca 514-848-2424, ext. 3593
CAMPUS SECURITY Ensures the safety of our members and campus property through prevention, surveillance, intervention, training, and education. Provides emergency medical services. security@concordia.ca 514-848-3717 (dial 1 for urgent situations; dial 2 for non-urgent situations)	**CONCORDIA UNIVERSITY STUDENT PARENTS CENTRE** An accessible space for student parents to study, share interests and develop a support network. Sumaiya Gangat, Coordinator cusp@concordia.ca 514-848-2424, ext. 2431

ACADEMIC HONESTY AND CODE OF CONDUCT

Violation of the Academic Code of Conduct in any form will be severely dealt with. This includes copying (even with modifications) of program segments. You must demonstrate independent thought through your submitted work. The Academic Code of Conduct of Concordia University is available at:
http://www.concordia.ca/students/academic-integrity/offences.html

It is expected that during class discussions and in your written assignments you will communicate constructively and respectfully. Sexist, racist, homophobic, ageist, and ablest expressions will not be tolerated.

ADDENDUM

ACADEMIC CONDUCT ISSUES THAT APPLY IN GENERAL
The basic ten rules that make you a good engineer

The B. Eng. program is set to satisfy most of the requirements for your education and prepares you for a professional engineering career that requires dedication and knowledge. What you learn, and how you learn, will be used extensively in your engineering profession for the next 30 to 40 years. Therefore, the four years spent in the engineering program are crucial towards your professional formation. The first step is for you to learn to "think like an engineer" which means:

- accept responsibility for your own learning
- follow up on lecture material and homework
- learn *problem-solving skills,* not just how to solve each specific homework problem
- build a body of knowledge integrated throughout your program
- behave responsibly, ethically and professionally

One of the mainstays of being a professional engineer is a professional code of conduct and as an engineering student this starts with the Academic Code of Conduct (Article 16.3.14 of the undergraduate calendar). However, you may encounter situations that fall outside the norm and in such cases, you use your common sense.

Further, the following issues should be given serious consideration:

1) Attendance at lectures and tutorials are major learning opportunities and should not be missed. The labs represent a unique opportunity for you to acquire practical knowledge that you will need in your career. Class and tutorial attendance is important for you to comprehend the discipline and make the connections between engineering skills. You are strongly encouraged to participate in the class, ask questions and answer the instructor's questions. Tutorials are just extensions of the classes in which application of the concepts presented during the lectures are presented and problems are practically solved.

2) The decision to write tests that are not mandatory is entirely yours. For example, midterm test are often stated in many courses as optional. However, one the objectives of midterms is to check on your

comprehension of the material and allow time for whatever action is necessary (from more study time to discontinuing a course). Plan to attend the class tests even if they are not mandatory. If you pay attention in the lectures, it will take you significantly shorter time to comprehend the material. **Note also** that if you are in the unfortunate position of being unable to write a final exam due to medical reasons and seek a deferral, this may not be possible if the instructor has no information indicating that you have been attending the course and assimilating the material (ie through midterms, quizzes, assignments etc).

3) Homework is usually mandatory and it has some weight in the final grade (such information is given in the course outline). Homework may also be conceived as training material for the class tests. Under all circumstances, it is highly recommended to carry out the home work on time and submit it on the prescribed date. Late submissions are not granted to individual cases regardless of the reason. This is part of the training for being in the workforce where deadlines have to be met. Please, plan your work such that you submit all the assignments and lab reports on time and in the correct place (not in the corridor or on the street!).

4) Office hours with tutors, lab instructors or class instructors are listed in the course outline/website/office doors. Please respect these office hours and in case you have a serious conflict, contact the instructor asking for a special time arrangement.

5) Class tests (midterms, quizzes) are returned to the student. The final exams are not. If you wish to see your exam paper, be aware that most instructors allow only a narrow window of time for that purpose. For the fall term, exams may usually be reviewed in January and May for the spring term.

6) When you see your marked work (assignments, midterms, final exam etc), be aware that you are supposed to review your material and see the type of errors you made and if marks have been added incorrectly. This is not an opportunity to try and "negotiate" a higher grade with the instructor. If you believe that your grade is not right, you may apply for a formal Course Reevaluation through the Birks Student Centre.

7) Writing tests and exams represents a major component of your course work. These tests and exams have rigorous requirements such as:

- **No cell phone or other communication enabling tool is allowed on the student** during the examination period.
- Only **specified faculty calculators** are allowed during tests and exams unless otherwise indicated by the instructor.
- Usually, **no materials** are allowed in the exam unless otherwise announced.

Get used to signing in and out of your exam. Make sure that you leave your exam papers with the invigilator. There are rules concerning general exam issues in the UG Calendar. These requirements are there to eliminate any possible misunderstanding and you are asked to **respect the rules**. Disciplinary measures are taken when the rules are not followed.

8) Respect your colleagues and those that you meet during the class: tutors, instructors, lab instructors, technical personnel, assistants, etc. Use appropriate communication means and language. Be considerate for all human beings. This includes small things such as turning off cell-phones before a class begins. Concordia University is a very diverse group of people and a very large multicultural community.

9) Communication is part of your future profession. Learn how to communicate effectively and efficiently in the shortest time possible. Write short but meaningful e-mails, make effective phone calls, etc. If your instructor accepts emails make sure that your request is clear with the course number and your name in the *Subject* line. Do not ask for special treatment as instructors have to treat all students equitably.

10) Respect all the above and you will get closer to your future profession.