ECE 297 – Software Design and Communication

Course Syllabus, January 2022

Lecturers and Office Hours:

	Software Design	**Communication**
Professor	Vaughn Betz	Ken Tallman
Office Location	EA 311	Sanford Fleming, B670 (Engineering Communication Program Offices)
Email	vaughn@eecg.utoronto.ca	k.tallman@utoronto.ca
Online Coffee / discuss anything	Thursday noon – 1 pm on zoom at https://zoom.us/j/92243117936 Passcode: ECE297	Monday noon – 1pm on zoom at https://utoronto.zoom.us/j/8730195834 Passcode: ECE297
Additional Meetings	Email for appointment	

"Design is not just what it looks like and feels like. Design is how it works."
- Steve Jobs

"The art of communication is the language of leadership."
- James Humes (Presidential speechwriter & author of Apollo 11 plaque left on the moon)

Course Overview:

This course involves designing and completing a large software project in a team, and communicating effectively with both technical supervisors and less technical project managers. The course focuses on practical skills in all these areas -- design, software development, oral and written communication, project management and teamwork – as these skills are essential to having a successful engineering career. The evaluation is similarly practical: your grade will be based on the quality of your software design, the calibre of your oral and written communication, and your project management.

The information concerning the course project and evaluation is subject to change if circumstances warrant; such changes will be communicated to students as early as possible if they are necessary.

Learning Outcomes:

By the end of this course, students will be able to:

- Write clear documents and incorporate relevant background research.
- Create memorable and informative oral presentations.
- Work effectively in a team and communicate status succinctly.
- Use agile software development techniques to complete a project.
- Effectively use modern software development tools, including revision control, debuggers, code verifiers, and unit tests.
- Choose appropriate structures for efficiency and implement them using the C++ STL library.
- Develop graphical user interface and visualization software.

- Employ graph data structures and basic graph algorithms to solve optimization problems.

Required Text: **Made to Stick: Why Some Ideas Survive and Others Die**, Chip and Dan Heath, Random House, 2007. Available in the U of Toronto bookstore, or you can buy it for ~$20 from Amazon.ca.

Prerequisites: ECE 244 (Programming Fundamentals) or equivalent. The course project is completed in C++ so you must be familiar with this programming language.

Covid-19: Lectures, tutorials and labs will all be online using zoom until January 31; lectures and tutorials will also be recorded so you can view them asynchronously if you prefer. After that, hopefully we will be able to return to in-person learning, but of course the situation is unpredictable.

Lectures: Monday, 5 – 6 pm, MY 150
Wednesday, 5 – 6 pm, MY 150

Zoom at https://zoom.us/j/92243117936?pwd=NUNVN1ViVUpXdWxGbHZoM0tUdkFHQT09
Meeting ID: 922 4311 7936
Passcode: ECE297

The lectures notes (slides) will be posted on quercus (generally before the lecture), and links to videos of the lectures will be posted on quercus after the lecture.

Key information on communication, programming, and algorithms will be presented during lectures; you will need this information to complete the course project.

Tutorials: **Tutorials start right away**: the week of January 10. Both the tutorial slides and videos of the tutorials will be posted on Quercus, and zoom links are posted on Quercus as well.

Tutorial Sections	Day & Time	Rooms	Taught By
101 & 102	Wed., 10 am – noon	MY 380 & MY 330	Vaughn Betz
103 & 104	Fri., 1 – 3 pm	MY 380 & MY 330	Andrew Boutros
105 & 106	Thurs., 9 – 11 am	MY 330	Andrew Boutros
107	Mon., 9 – 11 am	MY 360	Vaughn Betz

From January 10 to 21, (the first two weeks) the tutorials teach important programming information on how to use the C++ Standard Template Library (STL) that will be essential to completing your project. Accordingly, you should consider the first two weeks of tutorials *mandatory* and attend the entire two-hour tutorial time slot for your section (or view the video).

The week of Jan. 24 you will meet your Communication Instructor (CI) and choose a weekly meeting time (which might be during your tutorial time slot, or at some other time). From Jan. 31 onwards your team of three will meet with your Communication Instructor during that agreed-upon time.

During some weeks from Jan. 24 onwards, there will be *optional* coding tutorials that give additional guidance on good software architecture. The topic of each of these tutorials will

be announced in advance, and the slides and videos will again be posted. These optional tutorials will typically be about an hour long; any remaining time will be for Q & A.

Labs: **Labs start right away**, the week of January 10. There are three time slots; depending on your section you will attend one time slot. Zoom links for each TA will be posted on quercus.

Lab Sections	Day & Time	Rooms
101 & 102	Mon., 10 am – noon	SF2102, SF2204, GB 251 & GB243
103 & 104	Thurs., 4 - 6 pm	SF2102, SF2204, GB 251 & GB243
105, 106 & 107	Wed., 9 – 11 am	SF2102, SF2204, GB 251 & GB243

For the January 10 – January 21 lab periods (2 labs), you will work individually to learn the software development infrastructure of the course, and you must meet with a TA (on zoom) to demonstrate your knowledge of the tools to him/her to complete this lab. From January 24 onwards, you will work in teams of three in the lab and will be assigned a specific TA to mentor & grade your team; you must meet weekly with this TA.

You should attend your lab period to meet with your TA and teammates each week and work on your project. This time will not be sufficient to complete the project, so you should also expect to do some work outside the lab. At least until January 31 you will meet with your TA on zoom and access the labs remotely. There are two ways to work remotely: using the VNC windowing software to log into the UG computers, or using a virtual machine we provide that replicates the lab machine setup on your home computer. Details are in the VNC and virtual machine quick start guides on quercus.

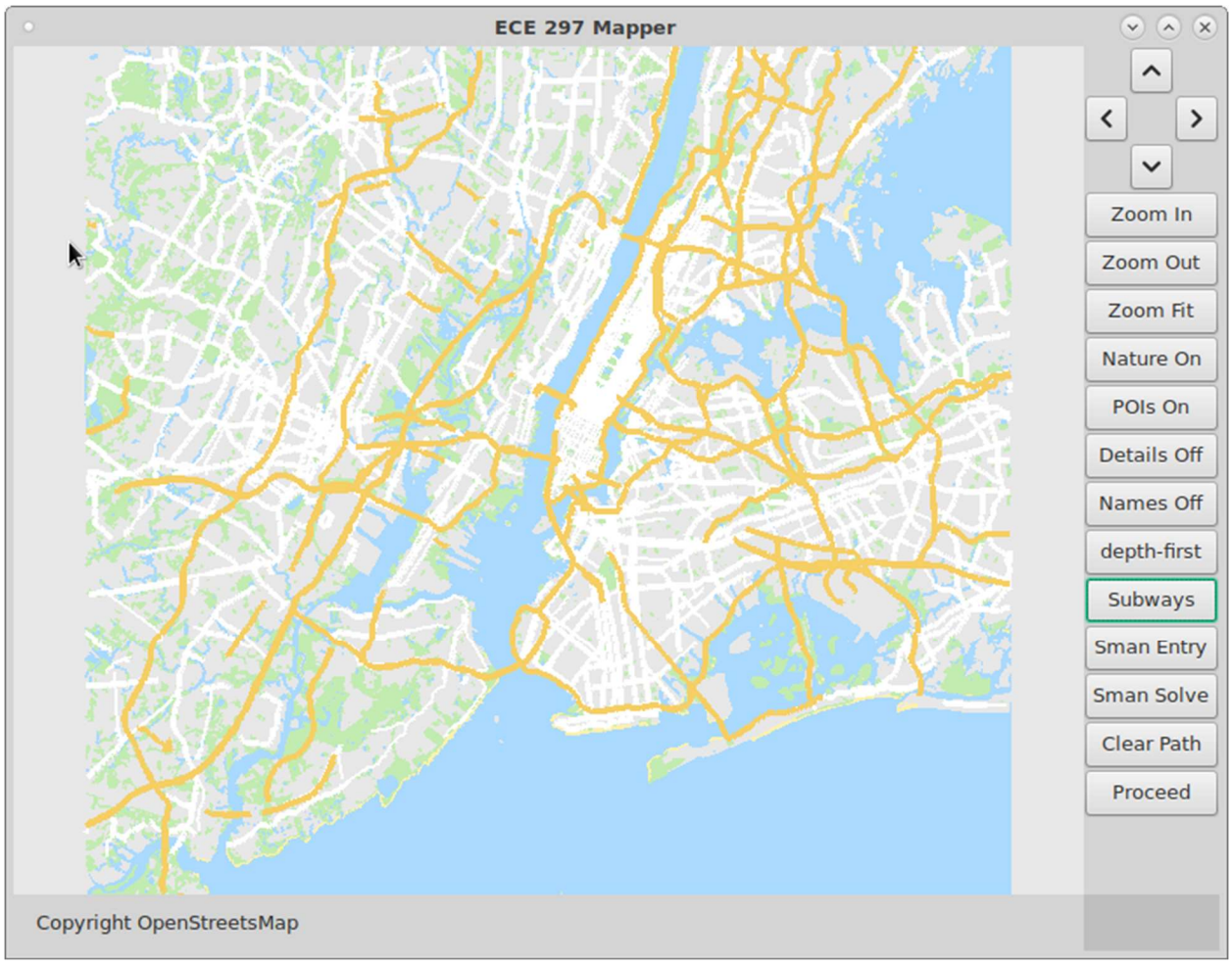
Head TAs: Mohamed Elgammal (mohamed.elgammal@mail.utoronto.ca)
Sameh Attia (sameh.attia@mail.utoronto.ca)

Tutorial TA: Andrew Boutros (andrew.boutros@mail.utoronto.ca)

Project and Organization:

In this course you will build a Geographic Information System (GIS) software program that will let you visualize and solve travel and optimization problems in maps of any city of the world. By the end of the course your program will be able to

- Read in a database of all the intersections and streets and other geographic features in a city and organize it into appropriate data structures.
- Draw the resulting map nicely and let the user interact (pan, zoom, highlight, search for locations, etc.) with it.
- Find good travel routes between intersections in the city and give directions to a user.
- Find a good order of deliveries and a good driving path for a courier company driver to complete his/her list of daily deliveries.



You will be evaluated by two people in this course.

1. A **Teaching Assistant (TA)**, who is a graduate student in engineering and will mentor you on design and software development, and evaluate the quality of your software project.

2. A **Communication Instructor (CI)**, who is trained in effective communication and will mentor you on oral and written communication, and who will evaluate your written documents and the majority of your oral presentations.

You will **work in teams of three students, and you will choose your teammates**. *All three team members must be in the same time slot for labs and for tutorials,* to make scheduling meetings with the teaching assistants and communication instructors feasible. You must choose your team members by the third week of the course and enter your team on Quercus; if you can't find team members we will assign you to a team in the 3rd week of the course. Each team will then be assigned one CI and one TA, who will mentor you and grade you for the duration of the course.

From January 24 onward, you will meet with your TA during your lab period to update him/her on your status and sometimes to demonstrate your progress toward a project milestone. You will also meet with your CI in another meeting to update him/her on your progress in creating oral presentations and written documents and to get feedback on these deliverables and potentially on the user interface of your software design. Your team will also have a wiki page (an easy-to-edit web page), and you should always update your wiki page before your status meetings to highlight what you have achieved each week, any challenges you have encountered, and what the tasks for each team member for the next week are.

Detailed Deliverables and Evaluation:

Your mark will be determined by the quality of your design and software, your oral and written communication, and how well you manage your project and how professionally you communicate your status throughout the course. The detailed list of deliverables and the mark distribution are given below. Full details on the requirements for each software or communication deliverable will be given in separate documents distributed when that deliverable is assigned later in the course.

Deliverable	Marks	Due Date	Marked by
Milestone 0 - basic SW tools & debugging (individual)	1	Friday, Jan. 21	**TA and automarker.** You will debug a program we give you, submit the fixed code & demo that you can use source code control and debugging tools to the TA.
Milestone 1 – create & load data structures, answer simple queries	10	Friday, Feb. 11	**TA and automarker.** Submit code. The code will be automatically graded for correctness, and your TA will also grade your project management (including your wiki page), code style and understanding.
Milestone 2 - graphics: draw map	15	Friday, March 11	**TA.** Submit code & demo to TA. Your mark will be based on the quality of your map visualization, code style and your project management.
Milestone 3 – finding travel routes between points	16	Friday, April 1	**TA and automarker.** Submit code & demo to TA. You will be marked on the performance of your path-finding algorithm, the usability of your travel directions and user interface, code style and project management.
Milestone 4 – find route for many courier deliveries to minimize travel time	12	Thursday, April 14	**Automarker**. Submit code. You will be marked on the performance of your route-finding algorithm; there are bonus marks and small prizes for the top solutions in the class.
CI meeting preparedness, professionalism and participation	2	All Semester	**CI**. Arrive at meeting with CI with clear status on wiki and well prepared communication deliverables. Behave professionally as a team member. Bonus marks are possible for students who excel at helping others on Piazza, or who make particularly strong contributions to lectures or tutorials.
Written Document 0: Presentation Analysis Using Made to Stick (Individual)	5	Sunday, Jan. 30	**CI**. A brief written analysis of the video of a technical presentation, highlighting how the presentation used Made to Stick principles and areas where it could be further improved.
Written Document 1: Graphics Proposal	10	Friday, March 4	**CI**. A written document summarizing your background research into how Geographic Information Services (GIS) applications are made responsive and easy-to-use, and a plan for how you will make your map visualization program usable.

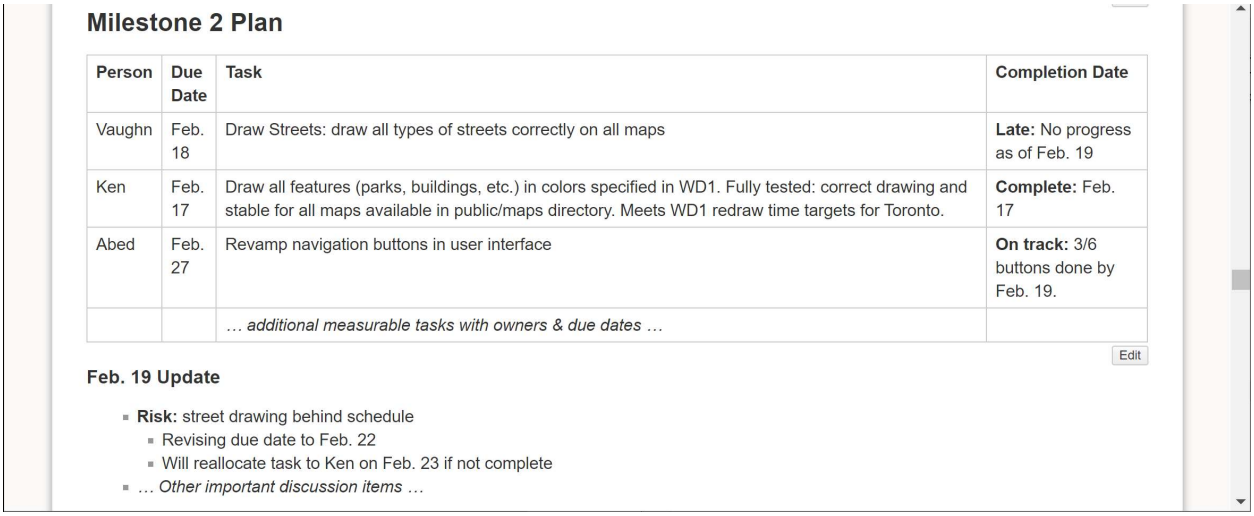
Oral Presentation 1: graphics functionality / usability	10	March 21 – 25, in your tutorial timeslot	**CI**. Oral progress presentation that gives an overview of your city visualization implementation and highlights unique features and how you made it responsive and easy-to-use.
Oral Presentation 2: project summary & pitch	19	During Exam Period (Likely Tuesday, April 19)	**CI and TA**. Final oral presentation summarizing what your program achieves; it is intended for an engineering management audience. You should describe the interesting features of your program, including screenshots or a video, and you should give a "pitch" for a future feature(s) you could add to the program to address some business or social opportunity.

All deliverables except those shaded and marked as "individual" will be completed in a team of three students. Note that while you will complete most deliverables as a team, the marks of individual team members can and will be different in some cases, based on the relative contribution of each team member and the quality and scope of the portion of the design / document / presentation created by each team member.

Submission deadlines will be at **11:59 pm** of the date in question. When a demo is required it will be done during your next lab period after the submission deadline. *The milestone (software code) submission deadlines are firm as completing a project on time is a key part of project management*. It is also important to realize that you can't achieve a perfect solution on a project; you are seeking a "good enough" solution in a reasonable time. The electronic submission procedure is disabled after the due date and late submissions will be accepted only in exceptional circumstances (contact the instructors if you feel you have such a circumstance). You should plan to complete your software well before the due date to avoid last minute submission problems. You can submit as many times as you like; the last submission will be the one graded. If multiple team members submit, the submission that achieves the highest grade will be used.

You are expected to meet each week with both your TA and CI, and have a written (on your wiki page) status update. If you cannot be present some week for some reason, it is up to you to make alternative arrangements with your TA and/or CI in advance. The screen shot below gives an idea of what a short status report might look like. Your status reports should list not just what the team has done and will do next, but should also give specific information on what *each team member* did or did not achieve in the past week, and what *each team member* commits to complete in the coming week. A table can be helpful to organize this information, as shown below.

Documenting your plan and your commitments to each other on the wiki is an important part of project management. Part of your milestone 1, 2 and 3 grades will be assigned by your TA based on your project management, and your CI will take the clarity of your weekly updates (on the wiki and in your meeting) into account in assessing the portion your mark for preparedness, professionalism and participation.



Milestone 2 Plan

Person	Due Date	Task	Completion Date
Vaughn	Feb. 18	Draw Streets: draw all types of streets correctly on all maps	**Late:** No progress as of Feb. 19
Ken	Feb. 17	Draw all features (parks, buildings, etc.) in colors specified in WD1. Fully tested: correct drawing and stable for all maps available in public/maps directory. Meets WD1 redraw time targets for Toronto.	**Complete:** Feb. 17
Abed	Feb. 27	Revamp navigation buttons in user interface	**On track:** 3/6 buttons done by Feb. 19.
		… additional measurable tasks with owners & due dates …	

Edit

Feb. 19 Update

- **Risk:** street drawing behind schedule
  - Revising due date to Feb. 22
  - Will reallocate task to Ken on Feb. 23 if not complete
- *… Other important discussion items …*

Suggestions and Workload:

To spread the workload of this course, get started on Milestone 0 and Written Document 0 right away! Your software deliverables will depend on knowing the tools that you are taught in milestone 0, so make sure you go through all the tutorials and complete Milestone 0 well. To complete Written Document 0 you will need to read the first six chapters of **Made to Stick**. You will also be expected to apply principles from **Made to Stick** in your written documents and oral presentations and be able to answer questions on how you applied the principles. For all these reasons you should read **Made to Stick** as early as possible, and certainly within the first three weeks of the course.

Later software milestones will depend on the code you write for earlier milestones, so it is important you do not fall behind on the early milestones, and that you create maintainable software so you can keep extending it through the course.

By Sunday, January 23 you should have formed a group of three, with all three teammates being in the same time slot (for both tutorial and labs). Get started on finding teammates right away, and be strategic in your choices. You want to choose teammates that:

- Have similar work ethics and goals for project quality / grades;
- Have complementary skill sets (e.g. an excellent writer plus two strong coders);
- Trust each other and work together well on a personal level.

You will enter your team composition in quercus. Balancing work across the three team members (and having a solid contribution from each) is key to doing well in this course with a reasonable workload.

Lecture Material:

The lecture material will help you complete the course milestones, written documents and oral presentations. Additionally, you will be asked questions after your presentations, and any material covered in the lectures and any material in the first six chapters of **Made to Stick** will be assumed background knowledge on which you should be able to answer questions.

The material presented in lectures and tutorials will include:

Lectures:

- Course goals and organization
- Source code control and git (Milestone 0)
- Made to Stick and effective writing overview (Written Document 0)
- Team dynamics and project management basics
- Graphs and data structures to represent them (Milestone 1)
- Good software style
- Testing, unit tests and software profiling and coverage tools
- Introduction to computer graphics (Milestone 2)
- Background research and writing a clear summary and plan (Written Document 1)
- Effective oral communication (Oral Presentation 1)
- Algorithms: Finding paths/routes in graphs (Milestone 3)
- Algorithms: The traveling salesman problem (Milestone 4)
- User-focused communication
- Handling questions well
- Body language and voice for impactful presentations
- How to pitch a technical business idea (Oral Presentation 2)

First two weeks of tutorials (mandatory):

- C++ templates
- The C++ Standard Template Library classes, including vectors, lists (linked lists), maps (binary search trees), unordered maps (hash tables) and iterators.

Later week tutorials (optional, agenda will be posted on piazza in advance):

- Getting started and structuring milestone 1
- Debugging techniques and walkthrough
- Good coding style
- Getting started with graphics and milestone 2
- CPU time measurement & profiling software speed
- Parallel code and multithreading to exploit multiple cores

Laboratory Computers and Working from Home:

Your lab assignments will be developed, tested, and submitted from the UG computers. You can access the UG machines (e.g. ug150.eecg.utoronto.ca) from your home through your Internet Service Provider (ISP). To do so, you must connect to ECE using a secure shell (**ssh**) and if you want to display graphics, through the Virtual Network Computing (**VNC**) program. For more information on using the ECE lab computers remotely, please consult the **VNC Quick Start Guide** on the course website.

You can also download a virtual machine that allows you to run a setup very similar to the UG machines directly on your home computer. See the **Virtual Machine Quick Start Guide** on the course website for details. A virtual machine allows you to complete most software development without being as dependent on the internet, but you will still have to do your final testing and submission on the UG machines.

All programs will be demonstrated and marked on the lab ECE computers. **A program that does not work correctly on ECE will be marked as incorrect!** Plan ahead, and make sure you leave time to integrate your code with your teammates, that it is committed into git and has been tested on the UG machines.

Getting Help via the Discussion Board:

If you have a question outside of your regular lab (TA) or tutorial (CI) time, you can post it to the course discussion board; we will use Piazza for the discussion board and there is a link to it on the course website. The TAs and course instructors regularly check this board to answer questions and the answers become available to all students. Thus, it pays to check the board before posting a question to make sure that the question has not been posted and answered earlier.

You are encouraged to use the discussion board, and to engage in discussions about the assignments with fellow students on the board. **However, do not post significant code as a public post on the discussion board** or give detailed solutions or algorithms on the discussion board. Doing so will be treated as an academic offense (see below). If you think your post needs to share significant code, make it a private post so only instructors can see it. Also, if the message is to be directed to a specific TA, CI or instructor consider email instead of the discussion board.

Other Help and Guidance:

When completing a major project like this there will be challenges, and that is part of the learning -- challenges and setbacks will happen on the real projects you undertake in your career, so gaining experience in managing them is important. The teaching team is here to mentor and advise you in any way we can. When you're looking for help or guidance on any part of the course (from questions about the course material, through teamwork or project management challenges), or have broader concerns about your workload or any other part of your academic experience, please reach out to the teaching team. Some options are:

- Discuss with your TA or CI in your weekly meeting
- Send a private email to your CI or TA, and/or set up a private one-on-one meeting
- Discuss with the tutorial TA (Andrew Boutros) in the coding tutorials
- Discuss with Dr. Tallman or Dr. Betz in their weekly zoom coffee hours, or after lecture
- Email Dr. Betz or Dr. Tallman to ask a private question, or to set up a one-on-one meeting

Independent Work and Academic Integrity:

For the university's academic integrity policy, see https://www.academicintegrity.utoronto.ca/.

Students from different teams are encouraged to discuss with one another issues and problems that arise in the course of completing the design project. Such discussions often provide interesting contrasts in design choices and can be very valuable learning experiences.

However, **work submitted for credit must be the student/team's own work**. It is one thing to discuss and compare, but quite another to rely on some other team's work to obtain credit for an assignment. It is also an offence to knowingly allow a copy of your work to be submitted by another person/team for credit. It is also an offense not to put in place protections to prevent your code from being copied without your knowledge. Use of any code written for this course by a student who took the course in a prior year is not permitted, whether you obtain that code directly from the prior year student or find it on an open website.

A reasonable rule of thumb to follow during a discussion that crosses teams is that you don't show the other team your code, and nobody leaves the discussion with written notes of what was said. It is unlikely

that two teams who have discussed only high-level approaches to a problem with no written notes will write highly similar programs.

All programs submitted for credit in this course will be compared pair-wise to identify cases of copying, excessive collaboration between teams, and similar offenses. A sophisticated program that is capable of detecting similar programs even if considerable effort has been taken to conceal their similarity does the comparison. We compare not only submissions from this year, but all prior years.

Any code you use in your program must either be written by your team or come from an open-source library. You can use the STL and boost libraries with no reference, but if you use source code from other open-source libraries you should add a reference to the library in your comments so it is clear where it came from. Use of significant amounts of code (outside of open-source libraries) you find from others (e.g. on the web) to implement a milestone is not allowed, and in no circumstances is any code written by another team (present or prior years) for this course permitted to be used in your project. If in doubt about whether you can use code available in a library or on the web, ask your TA or post a question to the instructors on Piazza.

All written reports and oral presentations must similarly be the work of a single team (or a single person, in the case of WD0). Reports and presentation files are submitted through Quercus, which is integrated with automatic plagiarism detection software. Normally, students will be required to submit their course written work to the University's plagiarism detection tool for a review of textual similarity and detection of possible plagiarism. In doing so, students will allow their written work to be included as source documents in the tool's reference database, where they will be used solely for the purpose of detecting plagiarism. The terms that apply to the University's use of this tool are described on the Centre for Teaching Support & Innovation web site (https://uoft.me/pdt-faq).

Any work submitted for credit that is not the work of the person submitting it will be treated as an offense under the Code of Academic Discipline of the University. Similarly, aiding anyone in the submission of work that is not the work of the submitter will also be treated as an offense under the Code of Academic Discipline of the University. Penalties can range from grade penalties in the course to suspension from the University. The Dean, as advised by the instructor and the Departmental Chair, determines the penalty for each case.

Notice of video recording and sharing (Download permissible; re-use prohibited):

At times during this course, some interactions including your participation, may be recorded on video and will be available to students in the course for viewing remotely and after each session.

Course videos and materials belong to the instructors, the University, and/or other source depending on the specific facts of each situation, and are protected by copyright. In this course, you are permitted to download session videos and materials for your own academic use, but you should not copy, share, or use them for any other purpose without the explicit permission of the instructor. For questions about recording and use of videos in which you appear please contact your instructor.

Mental Health:

As a university student, you may experience a range of health and/or mental health issues that may result in significant barriers to achieving your personal and academic goals. Moreover, this year the isolation of working during a pandemic may lead to extra stress for many students. The University of Toronto offers a

wide range of free and confidential services and programs that may be able to assist you. We encourage you to seek out these resources early and often.

- Health & Wellness Resources: undergrad.engineering.utoronto.ca/advising-and-wellness/health-wellness/
- U of T Health & Wellness Website: studentlife.utoronto.ca/hwc
- If, at some point during the year, you find yourself feeling distressed and in need of more immediate support, visit the Feeling **Distressed Webpage**: www.studentlife.utoronto.ca/feeling-distressed, for more campus resources.
- Off campus, immediate help is available 24/7 through **Good2Talk**, a post-secondary student helpline at 1-866-925-5454.
- All students in the Faculty of Engineering have an Academic Advisor who can advise on academic and personal matters. You can find your department's Academic Advisor here: uoft.me/engadvising

After the Course:

Hopefully you will find the course project memorable and will want to add it to your professional list of achievements. Many students keep a video of their final map and a copy of their source code so they can show them to potential future employers, and the teaching team strongly supports that.

Posting a video of your project publicly is fine, but posting the source code from this project on a public website encourages plagiarism by future students, and reduces their learning by giving them a template for how to attack the project. Such public posting violates the copyright on the code provided to you by the teaching team and also places you at risk of being drawn into an academic integrity violation if a future student copies your work. Therefore, while having a private copy or website of your project source code is fine (and encouraged), you **should not post your project source code on a public web site**.