

# CSC301

---

Introduction to Software Engineering



This is David.

David thought he knew software engineering.

David failed at his first job painfully!

This course is about why.

Don't be like David.

## Instructor (David Jorjani)

- Didn't take Software Engineering in undergrad!
- Excited to learn web programming in Php
  - Who is excited to learn Php today?
  - Learn the concept. Forget the language

# Professional Experience

- Former Founding Engineer/Team Lead at Kooltra
  - FinTech startup with \$8.9M raised in capital
  - Hard life for 4.5 years
- Led product & development at Ideal.com
  - AI startup screening millions of resumes monthly
  - Acquired by Ceridian in 2021 (\$\$)
- Leading product and engineering at Sayge
- Teaching CSC301 since 2017

# Personal Fun Facts

- International student
  - Learned English as my 5th language!
  - Learning Spanish now
    - Still struggle with the "1" in Salsa. Anyone has tips?
- Quiet student at the back in my first-year
  - No proper education

# So What's My Point?

- You'll have challenges
- Learning takes time
- The point is to
  - Learn from others' mistakes and refine your technique
  - Practice in a safe environment
  - Learn now so you have it easier later
    - My 20s could have been so much better

# Who are you?

- Computer Science students
  - Internship or PEY experience
  - No industry experience
  - Taking the course as a program requirement
- Who else?

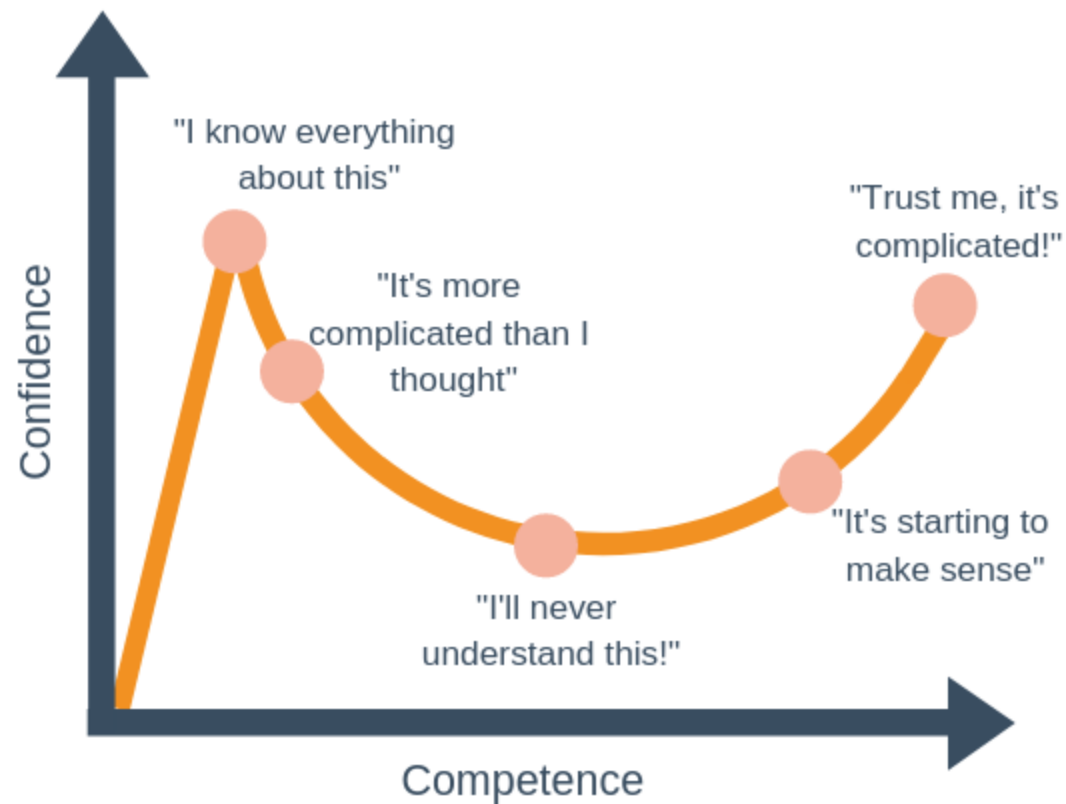
Already have an internship?







# Dunning-Kruger Effect



# Poll

1. Why did you take this course? In what areas do you want to improve?
  - Technical (specific language)
  - Specialty (cloud)
  - Soft skills
    - Team work
    - Communication
    - Leadership
  - Project experience
    - Resume & internship
  - Program requirement (just a good passing grade)



**What made you take this course?**  
**Learning technical skills, Specialty (e.g., cloud), soft skills, team work, project experience, resume & internship, program requirement**

# Course Goals

- Learn how to work **in a unique team** to build a **product**
  - Communication with your team and your customers
- Introduce **software development life-cycle** (SDLC)
  - Tasks, tools, practices and conventions used by software professionals
  - Standard software development framework used in industry
- Get you to think as pragmatic *professionals*
  - Identify **common problems** and apply **thoughtful solutions**
- Learn some new **tech skills**
- Improve programming skills in the language of **your choice**

# Topics

- Version control & Collaboration ([Git](#) & [GitHub](#))
- Product management
  - Articulating *what* we're building, *who* we're building it for and *why* it is useful/valuable
- Software processes & teamwork
  - focusing on [Agile](#) methodologies
- Software design & architecture
  - [Design patterns](#)
  - Crafting Code
- Automated Testing & DevOps
- Application Programming Interfaces (APIs)
- Teamwork

# Course Philosophy



Just the way driving is not about the stick shifts, software engineering is not about syntax

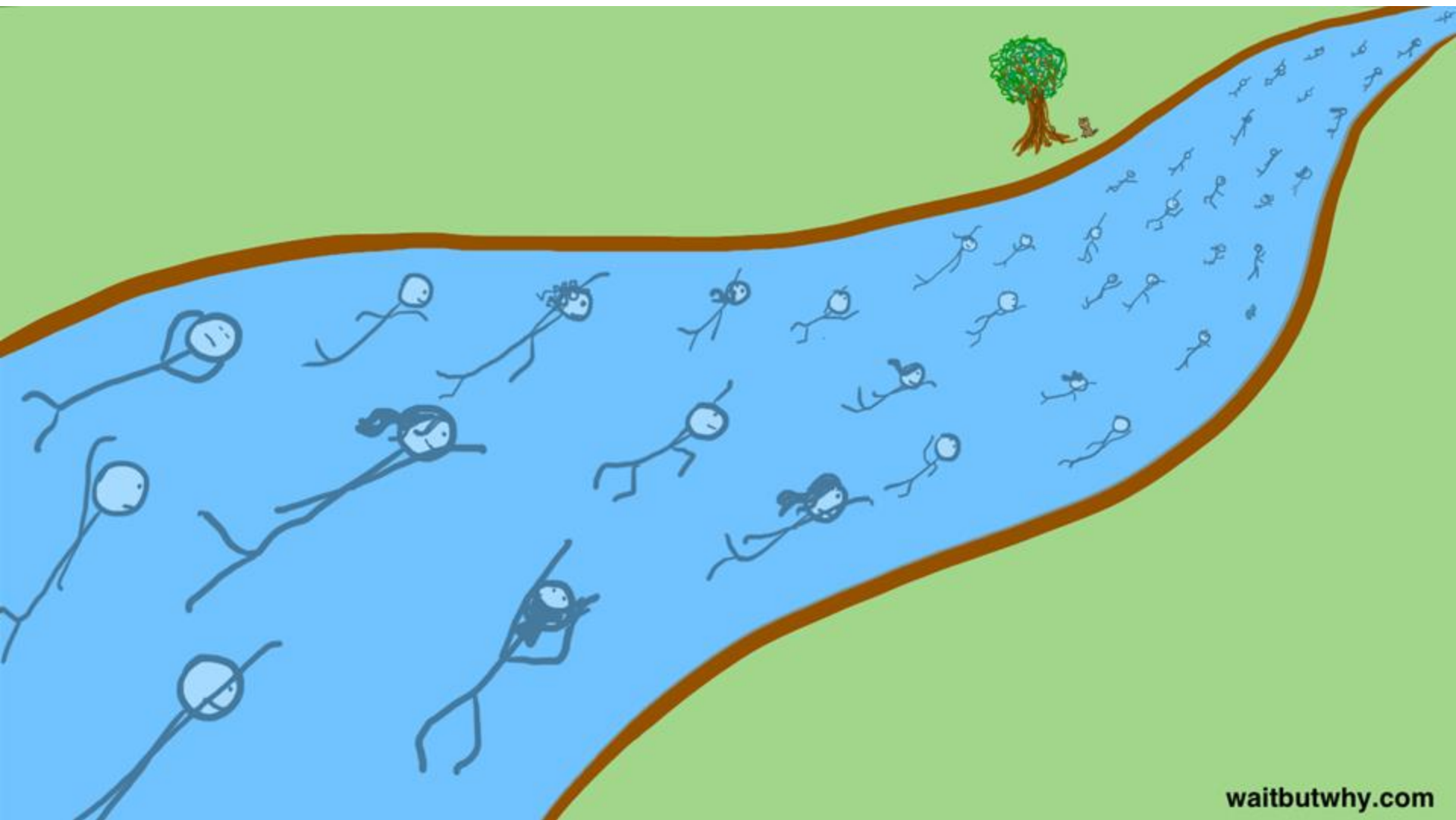




# Important Note(s)

- **You learn how to learn.** We DO NOT teach specific languages.
  - If you **just** want to learn a specific language (e.g., JS, Kotlin, etc.), take a course on Udemy or do a bootcamp
  - You will learn to learn a new language or improve your skills as part of this course
- **Team project** is the main component of the course.
  - If you want to work solo, you will disappoint everyone, including yourself. I recommend dropping the course.

# Academia



# Industry (and life)



# Warning

Welcome to the adult world. It sucks! This means: responsibilities, excuses cost you more than others, you take ownership for your life. Failure or success is on you. 1.1 lecture

# Logistics

Let's review the syllabus



The logo for the Wizarding World franchise, featuring a stylized sunburst or wand tip icon above the words "WIZARDING WORLD" in a serif font.

WIZARDING  
WORLD

CLIP

# Your feedback shapes the course

"Wingardium Leviosa and make A2 fly away, but if that doesn't work I'd narrow down the choice of tech stacks for A2 to better accomodate proper help"  
-Fall 2022

"Shorten the mandatory report for A1. We were forced to bs so much and instead could have summarized our actual decision in 5 sentences. I also didn't learn any more than I would have had I just spent 30 minutes researching. " -Fall 2021 student feedback

Led to simplification of A2 & removal of A3 for Winter 2022 onwards

Other noteworthy changes:

1. Tutorial restructuring
2. Peer evaluation adjustment
3. Lecture structure revisit



# Lectures & Tutorials

- Shortening the lectures & tutorials

It was pretty good, just a bit tiring since each lecture block was 3 hours

# Communication Policy

- General Emails
- Assignment Emails
- Project Emails
  - Begin email subject lines with “[CSC301]”
  - If your question is of general interest to the class, please post it on the discussion board (Piazza) first
    - You will likely get a faster response too
- Always include CSC301-A1/A2/D1/D2/D3, etc. in the subject
- Always include your utorid and specific details (e.g., repo link) with your request
- Office hours: **Thursdays 20:00-21:00** (tell me before lecture)

# Prerequisites

- Official

- [CSC209](#) - Software Tools and Systems Programming
  - Implicit prerequisite, [CSC207](#) - Software Design
  - Basic Object-Oriented Design
- [CSC263/CSC265](#) - (Enriched) Data Structures and Analysis

- Unofficial

- Python (basics)
- Git basics
- Unit testing
- Prior team experience

**Time to take a break**



# Team Project

- ~11 weeks long
- 6-7 students per team
- One TA per team, acting as a “mentor”
- Focus:
  - Defining a product considering users and need
  - Building a prototype/MVP
  - Organizing and working with a team
  - Working in a traceable manner
  - Presenting your work

# Team Project - What is it?

- Identify a problem, who cares, and how you intend to solve it
- Build a proof-of-concept or “Minimum Viable Product (MVP)”
- Evaluating your ability to use a software process to build a **usable product**
- Facilitated by the TAs during tutorials
- Two options
  - Working with a partner
  - Working on your own projects

# Team Project

- Three written deliverables
  - Concise deliverables presenting your work
  - Meant to be useful, not to add extra work
  - Evaluated by the TAs
  - Consistent individual contribution is **very important**
    - You are expected to contribute valuable work (i.e., code) regularly
    - Your grade will also depend on this!
    - **Peer evaluation on team deliverables**
- 20% - Demo
  - During the last week(s) of the term
  - Evaluated by the instructor
- Note: We will have peer evaluations! (Details to follow)

# Team Project - Option 1 - Partnership

- Partner option (*mini-internship*):
  - Product defined for a **good cause or startups**
  - More **realistic** projects with **real clients** and users
  - Products can be used after the term
- Proposals are shared with you (Check Quercus)

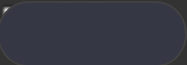


“my brother had recently got a great PEY job, and when I asked him, "what course helped you the most for the job search", he said Intro to software engineering because it allowed him to work in a team to build a quality app for a real company. This app looked great on resumes and all the interviewers were impressed with it” - 301 student

“This partnership program is extremely meaningful for both nonprofit/charitable organizations and students alike. All learning that occurs at the student and organizational level begets progress for both parties, which ultimately, moves the needle on solution oriented innovations that are needed to better the lives of vulnerable people and communities. “ -Feeding Canadian Kids

# What's the biggest secret tool to learn faster?

- "They didn't **ask questions** to clarify what we had communicated. "
- The students were very transparent and communicative. They should feel more open to reaching out to partners as mentors and asking questions.
- They didn't ask questions to clarify what we had communicated.

Hi 

Servers don't grow on trees.

I don't have time to find a long-lasting free server for you that works with docker.

Thanks,

# Team Project - Option 2

- Self-directed option:
  - Related to United Nations Sustainable Development Goals
  - Product defined by you
    - You can request to pitch your ideas in the tutorial and on Piazza

- “Working with a team that I am more comfortable in and can depend on more. I kind of regret working with my friends. ”

# Warning!

- By the end of the term, **your team** needs to know
  - Web or mobile development (CSC309)
  - Teamwork
  - DevOps (CSC209)
  - Databases (CSC343)

**If you are not good at any of them, you will have a lot to learn. I strongly recommend you choose a project with Python (Django/Flask) or another programming language that you are familiar with**

I want you to **support each other** and help each other get what you need. **CS is an individual program but life is a team sport.** Without teams, you'll fall behind.

# TEAM

- Learn from each other
- A teaches web programming to B. B teaches Databases to A
  - You both win
    - You learn a new topic
    - You learn to teach
    - You learn deeper
- Have sub-teams
  - Frontend, backend, data, API, etc.



# Team Formation Advice

- Find projects you are interested in rather than teaming with people you know
- Identify technology/roles you have experience or are comfortable learning
  - Identify missing skills (e.g., CSC343 (databases), CSC309 (Web programming))
  - Choose your tutorial time
- Find team members with similar interests and complementary skill sets
  - Develop areas of responsibility and specialty
- Aim to have sub-teams
  - Frontend
  - Backend
  - DevOps
  - Data
- Talk about your goal (Aiming to pass or aiming for 90+?)
- Do a team building activity (games, bubble tea, etc.)

# Team Project - Next steps and tips

- Review Proposals
- Find other interested students
- Do Assignment 1 as soon as you can

# Activity 1: Get to know each other

1. Join the breakout room with the project # that interests you the most
2. In the breakout rooms, choose a facilitator
3. Then each person takes 30 seconds to answer the following questions:
  - a. Why did you take this course? In what areas do you want to improve?
    - Technical (specific language)
    - Specialty (cloud)
    - Soft skills
      - Team work
      - Communication
      - Leadership
    - Project experience

# Fair grading exercise

Given a team mark of 80, what do you think is a fair grade for each student?

- A) All equal
- B) 80+ for the first two and ~70 for the last two
- C) All varied by their contributions
- D) Other



# Tentative Plan (in weeks)

1. Introduction
2. Product management (i.e., what problem are we solving?)
3. Data models (i.e., what data do we keep and how?)
4. Agile methodologies (how can we work better together?)
5. APIs (i.e., How do software systems/modules talk to each other?)
6. Teamwork (how do we become a high-performing team?)
7. Learning about Git/GitHub/Automation
8. Crafting software & professionalism (how to be a reliable engineer?)
9. Automated tests (how do we make sure everything always works?)
10. Industry panel (let's learn from those before us)
11. Presentations (let's learn from each other)
12. Accessibility (How to design software for everyone?) + design patterns

# Cheating

- Plagiarism can have **severe implications!**
- OK to discuss ideas, but don't share code or use others' code
- You can use available resources with proper references
  - Citing Sources
  - Academic Integrity in writing code
  - Citing software
- When in doubt, cite

# Getting Help

- If you don't understand something, you should ask
  - During office hours, on the discussion board or in class
- University-wide resources available to learn how to **write better**
  - Explore [English Language Learning](#) or [FAS English Language Learning](#)
  - [Arts and science writing centre](#)
  - [Writing advice](#)

# Personal Challenges

- If you are not confident in your English
  - Know that it's ok to make mistakes and not be fluent. You'll learn
    - My first English-speaking experience was at 21!
  - Find a friendly team member to get help from
  - Use resources shared earlier
- If you have personal, family, mental health challenges
  - You are not alone!
  - We are here to support you
  - Sometimes it's ok to not be ok. Communicate what you need
  - Seek more help from the university



# FAQ

- Is CSC301 worth it?
- I have already done an internship. I already know everything here, right?
  - Have you built and owned an application from start to finish?
  - Have you learned to lead?
  - Have you worked with all kinds of people out there?
  - Have you mastered all tech stacks?
- I can learn all these things in my internship or PEY. Do I really need 301?
- I don't know my team. Will I have a bad experience?
- Is it a good idea to work with my friends?
- I don't know web programming or databases. Should I still take the course?
- Should I just go to [bootcamp name] and learn [technology name] instead?
- Others?

Please keep in mind that CSC301 is a hands-on course!

In other words - a lot of fun! But also a lot of work.

**Stay in the course only if you are ready to do the work**

# It's ok to feel a bit stressed

- Find good teammates
- Set clear expectations
- Communicate early and often
- ASK For Help when you need!
  - Specially when things are not working

*"Change the course code to csc350 or something since having it named csc301 kinda makes it seem like its easy compared to other third year cs courses which is not the case"*

# I Need Your Help!

1. A few volunteer students to give feedback on assignments and course. Why?
  - a. More responsibility and more visibility
  - b. Better experience for everyone
  - c. A little sneak peek for you on what's coming
  - d. Help with your participation marks
2. What can I do to make you more comfortable with sharing your thoughts and feedback?

- “If you don't know to which port you are sailing, no wind is favorable.” - Seneca



# Inclusion Statement

We embrace differences. If we were all the same, a team would have no meaning. CSC301 supports and celebrates student diversity and we are committed to providing a fair experience regardless of gender, race, color, religion, sex, sexual orientation, disability or any other aspects of identity or background.

Discriminatory actions or statements will not be tolerated and we will do our best to foster an inclusive environments. Students are encouraged to report any discriminatory behaviour.



UNIVERSITY OF TORONTO  
FACULTY OF ARTS & SCIENCE

# Lead a Recognized Study Group (RSG)

SIDNEY SMITH COMMONS



**Sign up now to be an RSG Leader!**

Develop your leadership and facilitation skills while learning material for this course.

**Join an RSG starting January 16**

Study with your classmates, make new friends and stay focused while studying together.

Sign up now: [uoft.me/RSG](https://uoft.me/RSG)