Benjamin's Grooper

CSCD01

Anya Tafliovich
David Gao, Simon Ngo, Andrei Grumazescu, Raunak Mathur, Tony Ng

Winter 2019

Table of Contents

Meet Benjamin's Grooper

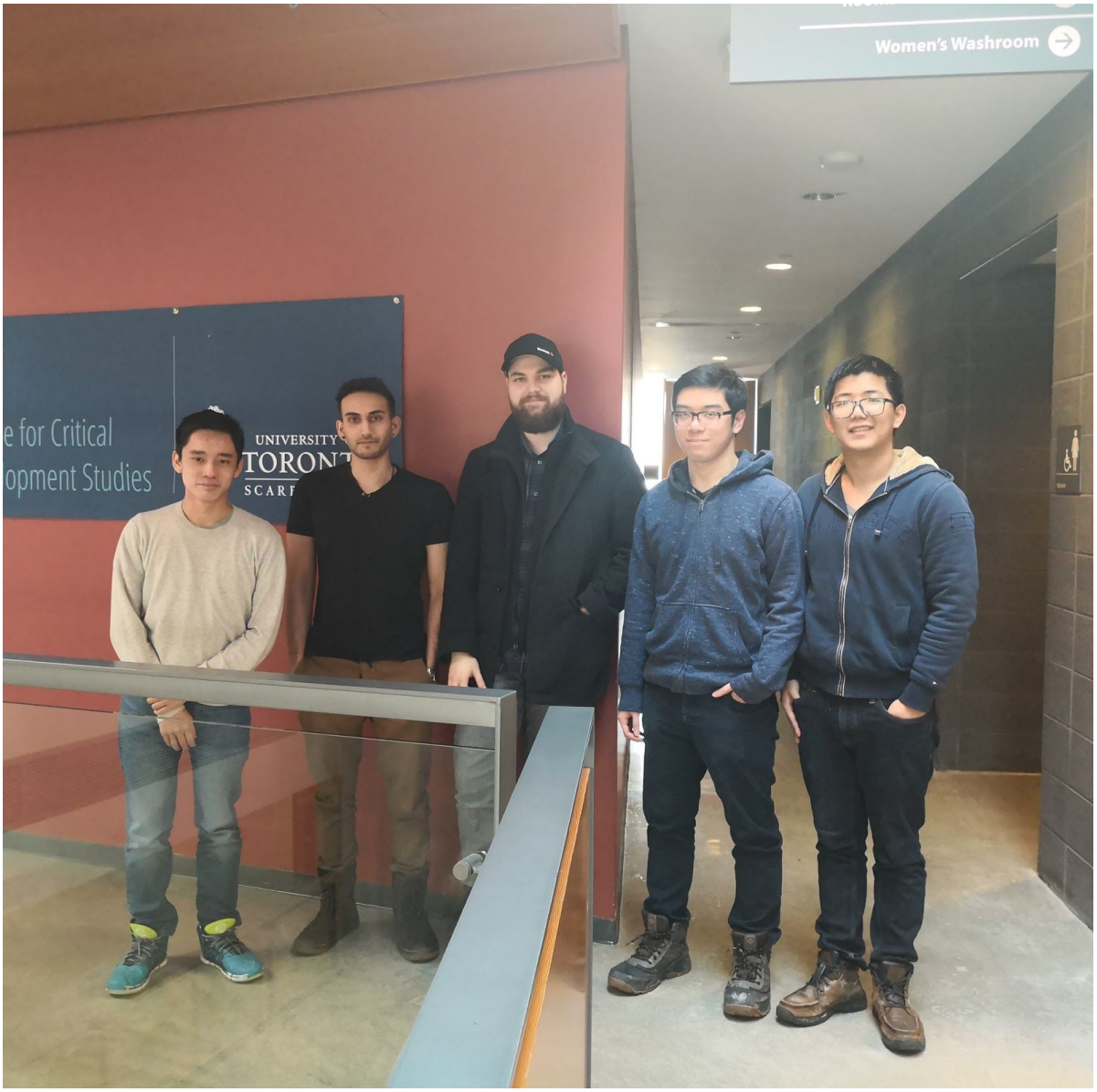
Hello! We are Benjamin's Grooper, a team of 5 UTSC students supported by our thoughtful TA Benjamin, and we will be working together this semester to improve matplotlib by adding bug fixes and new functionalities.

Our goals:

- Stay on track of sprints and communicate any issues along the way
- Ensure an effective final product that satisfies user requirements

Our teams strengths:

- Our team is multifaceted (experience with python and databases)
- Our team is willing to reach compromise (for example, on feature implementations, choice of language, etc.) for sake of project quality as well as team integrity
- Our team is eager to work, and willing to go the extra mile to make sure the product is feature complete
- All experienced with using git with other members

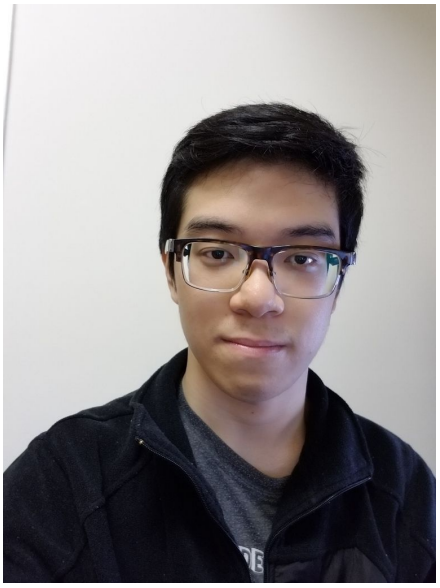


Team Members

David Gao

Xu Sheng (David) Gao is a UTSC student currently in his 4th year of Computer Science Co-op Program - Software Engineering Stream. Being in his fourth year of computer science at the University of Toronto Scarborough, he is gaining extensive experience in writing, documenting, testing, implementing code, logical analyses and debugging both academically and in extracurricular hobbies. He also has a strong theoretical background in programming languages and web applications, such as Java, C, Python, Angular, HTML and CSS. Not only are his programming skills an asset, but he also enjoys building or modifying hardware, such as fixing broken gadgets, assembling computers and building robots. Working with multiple talented people on interesting projects allowed him to become a self-sufficient individual and team player, he has the analyzing, interpretation and communication skills to fulfill the any role suited for my team. He is looking forward to overcome challenges and conclude this project with fresh experience alongside his teammates.

Simon Ngo



Simon is a talented and hard working 3rd year computer science student. Since high school, Simon has been an advocate for computer science by organizing the Hour of Code for 9th grade students and participating in team coding competitions. Having held previous positions at RBC, Rogers Media, and CAMH, he possesses a diverse portfolio of high demand skills such as test automation, web development, and database management. In his spare time, Simon likes to read textbooks about emerging computer science fields. Currently, he is learning more about the scikit-learn library for python and is trying to understand a classification algorithm for the MNIST dataset. Simon is studying at the University of Toronto where he one day hopes to use his education to disrupt the workforce. Simon strives to be a competitive worker and is always ready to take on whatever is sent his direction!

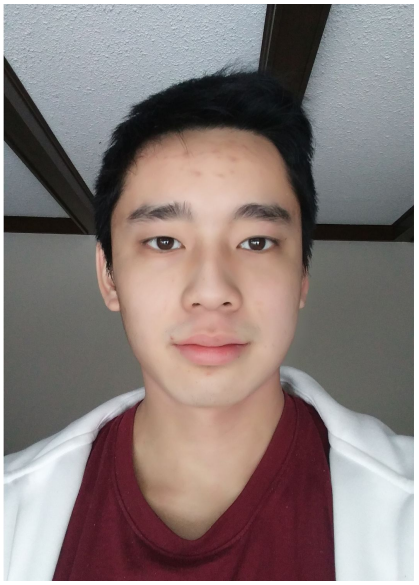
Andrei Grumazescu

Andrei Grumazescu is a third year student at UTSC, pursuing a specialist in Software Engineering. Andrei is proficient in Java, C and C++, and Python. He has experience working in an agile development cycle as a Scrum master, as well as creating Android applications, and solving real world problems in embedded systems. He has a natural tendency to want to lead groups, and acts to ensures no other team members have conflicts, or confusion about tasks at hand. Andrei makes certain he and his team members are clear about the team's goals, and coordinates meetings to ensure development is progressing smoothly. Andrei is a believer in the UNIX philosophy. In development, Andrei uses consistent design rules to prevent bogged down and confusing code, or allows for easy future additions. Ultimately, Andrei communicates effectively to both clients and team members, resulting in a friendly, stable, and efficient environment where development is bolstered.

Raunak Mathur

Raunak Mathur is a fifth year student at University of Toronto Scarborough double majoring in Computer Science and Mathematics who enjoys problem solving and is very driven and logical when it comes to working on projects. He has gained a passion for programming through several high school and university projects including an Android application using Java and games using a GUI in Python. He also enjoys solving mathematical/computer programming problems from leetcode during his spare time. Through past experience of working on an android application as a group, he has shown to be capable of bringing excellent work ethic, communication and interpersonal skills and can be a valued member of any team. Raunak is proficient in C#, Java, Python and shell programming languages. He has worked with Windows, Linux and Android operating systems. Raunak hopes to apply his problem solving and teamwork skills to more projects in the future.

Tony Ng



Tony Ng is currently in his fourth year at the University Of Toronto Scarborough studying computer science. He enjoys the challenges and problems that comes with programming. Through his academics, work experiences and personal projects he has gained a wide exposure in many fields but he is always eager to expand on his sea of knowledge. He believes that strong communication is the key to building a strong team in order to create greater projects. Currently he is interested in furthering his experience with web programming and taking his first steps in machine learning. He hopes to create a positive influence within this open source project and to create a learning experience along with the team.

Team Agreement

Methods of communication - Messenger, Discord and Slack

We chose Messenger because it is a tool that all members use regularly and is convenient. Discord allows for voice and video communication. Additionally Discord has text channels which can be used to sort more technical discussions. We will also use Slack for communicating with Ben whenever necessary.

Communication Response Time - Check intermittently (3 - 5 times) throughout the day, maximum one day response rate.

We decided that checking our communication channels a few times a day is not obtrusive and it will allow for issues to be solved in a reasonable time. The maximum limit of one day is to prevent work from becoming blocked too heavily but also allows for personal business.

Meeting attendance -

TA Meeting: 1-2PM monday (in person and mandatory but can be excused if reason is valid and provided ahead of time, and the individual is responsible for letting the TA know ahead of time)

Team meeting:
11PM - 12PM Monday (in person, mandatory but can be excused for valid reasons)
12PM - 1PM Thursday (discord, mandatory but can be excused for valid reasons)

We chose an in-person meeting to plan the weekly sprint and another Discord meeting halfway through the sprint to stay updated and discuss issues. Meetings are mandatory due to the importance of these plannings. Any additional meetings must be notified to the team at least a day ahead to give time to planning.

Running Meetings - During the Monday meeting we plan to determine the sprint backlog but we also leave a bit of time to discuss other issues. We also plan on doing a short and casual code review for keeping everyone up to speed. Notes will be taken by one member, the member will be chosen based on rotation.

Taking notes on rotation start with Raunak on February 4th, order is (Raunak -> Tony -> Simon -> Andrei -> David)

Meeting Preparation - For the TA meeting we will have a quick discussion beforehand about what work is done, and some of the stuff we want to work on the next week. We will also discuss what kinds of questions we need to ask the TA. We will run the code and make sure there’s no unexpected work.

Tools -

Github Kanban - for visual representation of tasks and convenience of using the same program
Travis-ci - for CI/CD when testing our code against Matplotlib's testing suite
Discord - for communication on specific channels for specific tasks
Messenger - for a convenient and alternative method of communication
Slack - for communicating with Ben whenever necessary

Version control

Separate features into branches where name of branch is the feature.
Log messages should describe what work is done, but doesn't have to be very specific.
Use pull request to merge branches. Have a description of what was changed.
Don't commit:
- code that is plagiarized. First commit your code to branch and review before merging.
- Private files
- Undocumented code
- Broken code (to Master)
- Inappropriate content (inappropriate language, verbal abuse, etc)
- Unorganized code (entire code written on one line)

Log message content:
- Bug detail (if any) + fix implemented
- New features include a short description
- For rollbacks, state specifically why rollback was implemented
- Branch per feature before implementing

Division of work

Decide during Monday planning meeting
Democratically divide work.
Everyone decides for themselves which tasks they will do based on their strengths and preferences.
Everyone is responsible for the tests of their own code.

Submitting deliverables

Have sprint content for deliverable finished 24 hours before deliverable due time.
Submit deliverable 5 hours before due time.
Everyone will review the submission.
Deliverables made in docs - delivered in PDF.

Contingency planning

If team member drops then split that members work between two other members
If a team member is consistently missing meetings then we will first try and reason with them to make sure they can join us. If there is no success then we will discuss this issue with our TA.
If a team member is unsure if an action academically dishonest, first they should consult the team, and if they are still unsure consult the TA.
If a team member commits code which is academically dishonest, they have to explain to the TA and prof.

Agile Features:

User Stories: We have chosen to modify user stories to use tasks instead because the user itself is not relevant for most of the work. Instead we will use tasks to breakdown bug fixes and features we plan to implement.

Burndown Charts: We have chosen not to make burndown charts because we think the statistical purpose they serve is not relevant for our purposes.

Sprint Plans and Sprint Execution: Despite not using burndown charts we have decided that using Sprint Plans and Sprint Execution charts are helpful tools to keep track of progress and sprint velocity.

Planning Poker: We will be having planning poker to assign story points to tasks.

0 - trivial, might already be finished, just needs some fine touching
1 - easy, can be finished in one day
3 - medium, can be finished in three days
5 - large, can be finished in less than a five days
8 - needs research, time investment, task might be broken down

Continuous Integration: Travis-ci may be used to keep track of bug testing on different branches.

Scrum Master: We will have this role rotated weekly, to ensure our developers are on track, not blocked on any work and to resolve conflicts if necessary.

The signatures below indicate that the members of Benjamin's Grooper have read, understood, and intend to follow the guidelines stated above.

We accept these guidelines and intend to fulfill them (sign below):