Department of Electrical, Computer, and Software Engineering
University of Auckland
COMPSYS 302 S1 2019 – Python Project (Weighting 50%)

Background

We live in the information era. We proliferate information about ourselves online, sharing our thoughts, morals, goals, and moments with our family, friends, acquaintances and colleagues.
However, this increasing usage of digital connectivity comes with difficult compromises. In order to share our data, we must place trust in those services that promise to do this for us in a secure and private manner. In order for new connections to make contact with us, we must expose channels of communication which others could use to send unwanted information to us. And in an era of digital impersonation and *fake news*, we must place a certain level of trust in the data that we receive from our connections, trust which could be abused by malicious third parties.

Emboldened by your successful game development, your previous client returns to you with a new challenge. He's disappointed in the current industry offerings for digital social media. He wants a new service, one that is fundamentally different from those that have come before. Rather than a centralised service that we trust with our data, he wants a peer-to-peer social network to be created for himself, his friends, and his daughter and her friends.

Your job now is to create that system for him - well, a functional prototype of it anyway. A functional prototype means that we are mainly concerned about the *functionality* of the system, and not its visual design or aesthetics (although it would be nice to have them if possible).
The client has specified qualitative design requirements: he wants the system to feature *privacy* (how do I share my data with only those I want to share it with?), *trust* (how can I ensure that I can believe that incoming data is sent from whom it says and wasn't modified en route?), and *consent* (how can I opt out of receiving unpleasant communications?).

However, these requirements are not technical in nature: he isn't sure how they will map to the final implementation. As a result, the client would like to engage the class (via the lecturer and teaching assistants) to find out what you as developers will be able to offer.
This is an individual project, but the class will be developing the requirements and protocol together.

The system will be implemented in Python 3, with the help of CherryPy, a simple web framework written in object oriented Python. To make the program easier to use, a simple web interface will be constructed for the app. Each student in the class will develop their own implementation of the system, which will be able to interact with each other. Therefore, an interfacing protocol will need to be agreed upon between the developers.
The system will be used by the client on Ubuntu; therefore, the system will be developed and tested on Ubuntu.

Objective

The overall goal of this project is to design a simple peer-to-peer social media network. Basic objectives are presented here. Additional elements will be decided upon in discussion between the class and the client. During project discussion times, feel free to propose additions to the lecturer, who is acting as the client for the purposes of this project. On Monday of Week 11, the teaching staff will release a more complete requirements list developed in conjunction with the students.

At a bare minimum, the requirements are:

1. The application must run in **python3** on the Faculty Teaching Linux system without us having to download additional packages or configure anything manually.
    a. We will read the README,
    b. run the application,
    c. and test it against our test application and/or other student's applications.
2. The application must:
    a. Allows a user to log into the system
    b. Automatically find other users on other computers
3. Users must be able to:
    a. See if other users are online
    b. Send public broadcast (to everyone/anyone) messages
    c. Send private (to specific people) messages

The purpose of this project is to learn Networking, Web Development, Security, and Software Project Planning & Development. Languages learnt are Python, HTML/CSS, and Javascript (if desired).
The project is completed **individually.**

Versioning

The use of the Git versioning system via Github is compulsory and will be monitored. Steady progress from all groups is expected throughout the semester, and you should not just commit the day before everything is due. Students should create one private repository per group with the name "2019-Python-abcd123" (where abcd123 is your UPI).

Assessment

This project is completed and assessed individually. We do not want to see any identical projects. We will be checking for code plagiarism, and if any plagiarism is found, then we will not hesitate to fail you and report you to the University Senate. There are three assessment phases (more details of the specific requirements will be explained in lectures):

1. **Python Assignment (5%) – Due 8pm Thursday 9 May (Week 8)**

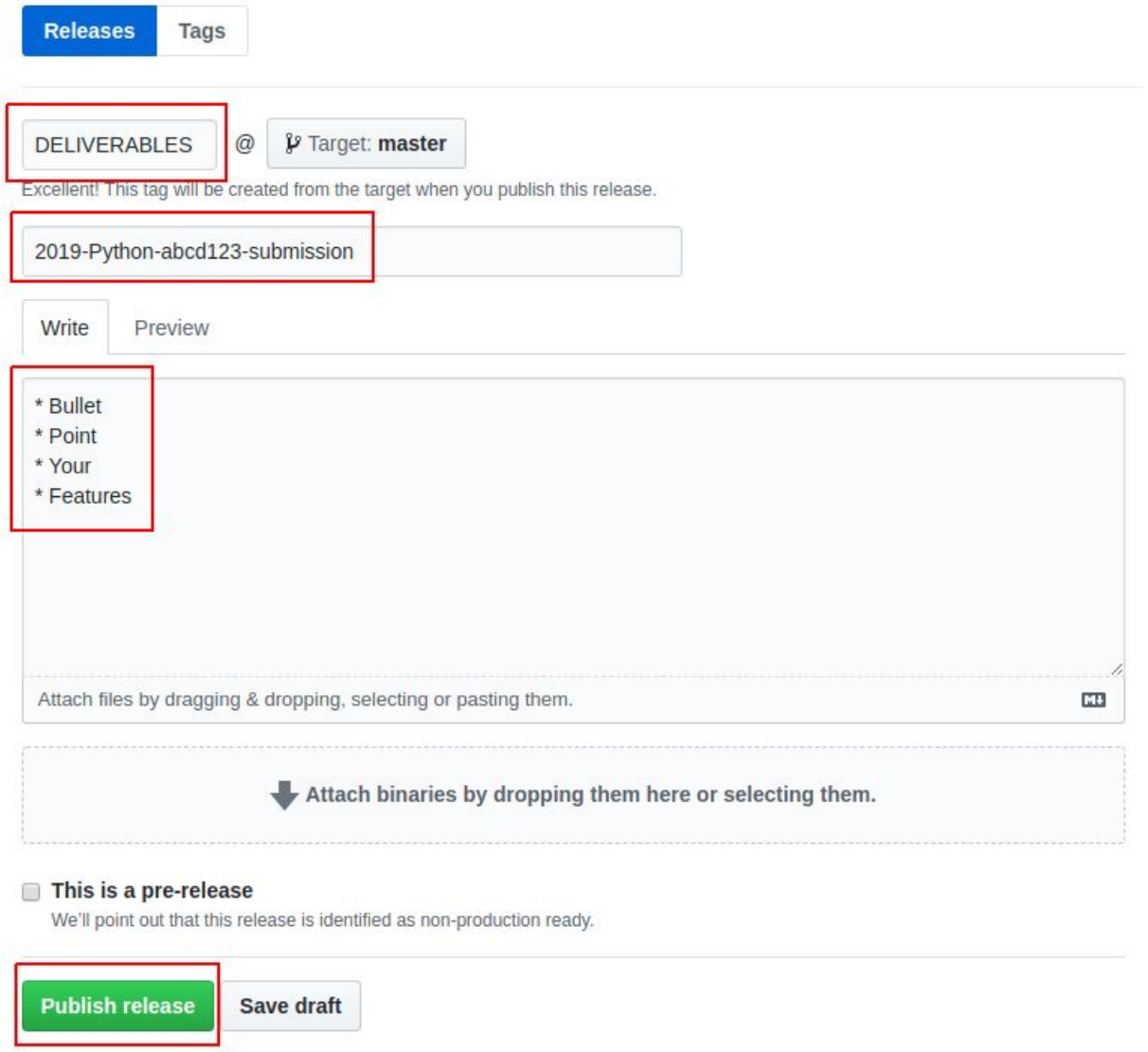
This is a simple programming assignment that helps you become familiar with Python syntax. It is administered via Coderunner - coderunner2.auckland.ac.nz and comprises five programming challenge questions, each worth 1%.

2. **Protocol Submission (5%) – Due 6pm Friday 17 May (Week 9)**

Students will be randomly assigned to groups of around 10 students at the beginning of Week 9, and will submit a protocol proposal by the end of Week 9. More details will be provided in class. The document should include a description of how the student proposes that the network should work (with regard to request and information flows), and a list of proposed APIs (including the name, description, inputs, and outputs). Submission is via Canvas.

3. **Final Project Code (30%) – Due 2pm Wednesday 5 June (Week 12)**

Submission is via Github – create a release tag on the appropriate commit with DELIVERABLES as the tag name, and 2019-Python-abcd123-submission as the title, where abcd123 is your UPI.



4. **Final Project Report (5%) – Due 2pm Wednesday 5 June (Week 12)**

Your design should be documented in a report which is in the form of a Github Wiki.

To make this, you will use the "Wiki" link on the Github project page, and then create the Wiki. This is a very intuitive process, so I won't go into much detail here. You should have (at least) one page for each of the following major bullet points, with around 500 words on each page:

- Landing/welcome/index page
    - Introduction to the project/system
    - Introduction to the developer
    - Links to the other pages
- (System features)
    - How the developed system has met the client's requirements
    - Features that improve functionality of the system
- (System implementation and architecture)
    - A technical, top-level view of how the system works (diagrams help)
    - A discussion of how the implemented system functions
- (Protocol discussion)
    - A brief discussion on peer-to-peer methods and its suitability for this application
    - A brief discussion on the protocol and its suitability for the system
- (General discussion)
    - A brief discussion on the process of developing the protocol and its suitability
    - A comment on the suitability of the suggested tools (e.g. Python) for the application
    - Suggested improvements for future development

5. **Final Project Demo (5%) – Demo in Lab on Friday 8 June (Week 12)**

On the final demo day (in addition to submitting the code), there will be **individual** interviews with each student, where each student individually presents their social media interface, and discusses the networking elements of the project.

Final Note

In this project you are expected to do your own planning and learning. You'll be responsible for finding out how to do things yourselves. There will be content delivered in lectures to help speed this process up at the beginning, but ultimately it is your responsibility to find out what you will need. If you get stuck and need help, please ask a TA or the lecturer. **Do not leave things to the last minute**. The easiest way to fail this project is to try to start it in the last week (or night) before it's due.

Do not mistake the minimum requirements as the only requirements; if you want to do well in this course, you will need to design your application carefully and manage your time well. Do

not hesitate to contact the lecturers and TAs if you are in doubt! They are available to help with concepts, troubleshooting, and debugging, but they will not write your project.

Poorly documented or implemented code, while otherwise functionally correct, may not get you full marks for the project overall. Someone else has to be able to work on your code later! Please develop consistent and good coding practices.

The Friday lab sessions of Weeks 8, and Tuesday lab sessions of week, 9, 10, 11, and 12 are **compulsory**, and attendance will be taken. WIthout prior arrangement with the teaching staff, penalties can be given to final marks due to non-attendance. Attendance at the other labs is also recommended.

Due to the nature of this project, a certain level of collaboration is required within the class. As a result, it is strongly recommended that you interact with other students in the labs. Additionally, TAs will be there and you can discuss requirements with them as well.

Academic Integrity Notice

The University of Auckland will not tolerate cheating, or assisting others to cheat, and views cheating in coursework as a serious offence. The work that a student submits for grading must be the student's own work, reflecting his or her learning. Where work from other sources is used, it must be properly acknowledged and referenced. This requirement also applies to sources on the world-wide web. **Do not copy code from other students or the internet** without attribution.