# Camelot Progress Presentation

## Software Engineering

Ian Howell, Hunter Mathews, Illya Starikov, William Thurman, Zachary Wileman (*Server Team #1*)

March 24th, 2017

Missouri University of Science and Technology

# About the Camelot Server

- Camelot is an asynchronous *Python3* server.
- The server is housed on a Raspberry Pi (as provided by Ian Howell) so that the client team will be able to interact and test their code on it (assuming the server is up and running) and also so that interaction with the server can be done remotely.
- The database is being tackled specifically by a module in Python called *psycopg2* which uses *PostgreSQL*.
- All communication between the server and client applications is done via *JSON* encoded data.

# Camelot Inspiration

**Figure 1:** “On second thought, let’s not go to Camelot. ’Tis a silly place.” — Arthur, Monty Python

# What Has Worked Well

Below are some of the things that have worked well for us up to this point.

**Teamwork** Task delegation and team work has made us able to accomplish more, in less time. Having teammates with a wide array of complementing skills and backgrounds.

# What Has Worked Well

Below are some of the things that have worked well for us up to this point.

**Teamwork** Task delegation and team work has made us able to accomplish more, in less time. Having teammates with a wide array of complementing skills and backgrounds.

**Meetings** Our meetings have been productive, where topics such as documentation, implementation details, and how to change the users this time are discussed.

# What Has Worked Well

Below are some of the things that have worked well for us up to this point.

**Teamwork** Task delegation and team work has made us able to accomplish more, in less time. Having teammates with a wide array of complementing skills and backgrounds.

**Meetings** Our meetings have been productive, where topics such as documentation, implementation details, and how to change the users this time are discussed.

**Changes** Sometimes when you innovate, you make mistakes (i.e. choices for users. . .). It is best to admit them quickly, and get on with improving your other innovations.

# What Has Worked Well

Below are some of the things that have worked well for us up to this point.

**Teamwork** Task delegation and team work has made us able to accomplish more, in less time. Having teammates with a wide array of complementing skills and backgrounds.

**Meetings** Our meetings have been productive, where topics such as documentation, implementation details, and how to change the users this time are discussed.

**Changes** Sometimes when you innovate, you make mistakes (i.e. choices for users. . .). It is best to admit them quickly, and get on with improving your other innovations.

**Tools** By picking some of the best tools (Github, LaTeX

# Past Changes

Below are past decisions that if we could change, we would.

# Past Changes

Below are past decisions that if we could change, we would.

- 
- 
- 
- 
- 
- 
- 
- 
- 
- 
- 

# Past Changes

Below are past decisions that if we could change, we would.

# Tools I

A summary of some of the tools we're using/enjoying.

- Server code is written in Python3.
  - Documentation is written in Doxygen.
- All development is on Github.
  - This includes meeting notes, documentation, and all production code (i.e. database and server).
  - Github issues/milestones for task management.
  - Contributions to make sure even workload.
- All documentation/presentation is written in LaTeX.
- Using ProgreSQL for the database.
- Using Discord for team/client chat – by any memes necessary.

# Documentation Demo

# Tools II

## Python enables easy server setup.

```
    def main():
        server = ThreadedTCPServer(("0.0.0.0", 9009), ThreadedTCPRequestHandler)
        ip, port = server.server_address
        server.socket.listen(10)

        server_thread = threading.Thread(target=server.serve_forever)
        server_thread.dameon = True
        server_thread.start()
        print("Server loop running in thread:", server_thread.name)

        try:
            # Loop forever
            while True:
                pass
        except KeyboardInterrupt:
            print("Cleaning up server..")
            server.shutdown()
            server.server_close()
            print("Done! Goodbye")


```

# *Server Demo*

# *Github Demo*

# Challenges

We have encountered three big challenges so far.

1. Communication between the client and server teams.

### Description & Solution

Even though we have set up a system for communicating between the client and server teams (Discord), there is little to no talk between the teams. *Solution: Communicate more..?*

# Challenges

We have encountered three big challenges so far.

1. Communication between the client and server teams.
2. Direction and vision of the product.

### Description & Solution

Although there is an established, core vision (IRC clone), working out the details has proven to be difficult. *Solution: Break it down to basic principles and work with teammates.*

# Challenges

We have encountered three big challenges so far.

1. Communication between the client and server teams.
2. Direction and vision of the product.
3. Learning curve of tools, teammates, and project.

### Description & Solution

Because we're relatively new to each other, and are unaware of each other's skill sets, assigning tasks becomes tricky. Also learning how to implement new things can be difficult. *Always be sure teammates are comfortable with the tasks they're given.*

# Extras

- Encryption.
  - Still debating if we'll be using ROT13 (znlor vs jr tvg thq, jr'yy hfr EBG26.) or an in-house, post-quantum cryptographic hash function using a multivariate-quadratic public-key signature system.
- Open source

# Extras

- Encryption.
  - Still debating if we'll be using ROT13 (znlor vs jr tvg thq, jr'yy hfr EBG26.) or an in-house, post-quantum cryptographic hash function using a multivariate-quadratic public-key signature system.
- Open source at the end of the semester.
- Full documentation guide courtesy of Doxygen.

# Current and Future Plans

1. Consolidate the client and server teams to get unified protocol and systems specification document.
2. Finish coding the actual chatroom.
    - Server
    - Database
    - Script to automate a chatroom environment
3. Finish documenting the chatroom (possibly API guide).

# Current and Future Plans

1. Consolidate the client and server teams to get unified protocol and systems specification document.
2. Finish coding the actual chatroom.
    - Server
    - Database
    - Script to automate a chatroom environment
3. Finish documenting the chatroom (possibly API guide).
4. . . .
5. Profit

# In Closing

Spam Illya with all questions, comments, and insults.


- @IllyaStarikov
- starikov@mst.com


Special thanks to our awesome team.

- Ian Howell
- Hunter Mathews
- Illya Starikov
- William Thurman
- Zachary Wileman