**CS673S16 Software Engineering** 

**Team 4 - Project Name**

**Software Design Document**

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Role(s) | Signature | Date |
| Daniel Budris | Team Lead, Configuration Lead |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Change** |
|  |  |  |  |
|  |  |  |  |

[Introduction](#_87t9hln2vjz0)

[Software Architecture](#_buttcq9i221r)

[Design Patterns](#_x18fj36s1121)

[Key Algorithms](#_mtfbusfb0eq3)

[Classes and Methods](#_7ucksmkf6rzx)

[References](#_15tmymhipvdv)

[Glossary](#_8n34lvocupub)

# Introduction

In this section, give an overview of this document, and also address the design goals of your software system.

# Software Architecture

In this section, you will describe the decomposition of your software system, which include each component (which may be in terms of package or folder) and the relationship between components. You shall have a diagram to show the whole architecture, and class diagram for each component. The interface of each component and dependency between components should also be described. If any framework is used, it shall be defined here too. Database design should also be described if used.

**Communications tool chat features class diagram:**

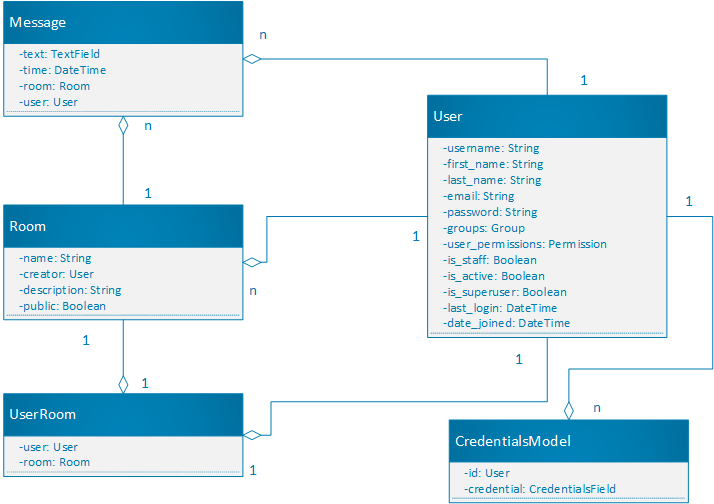


Figure 1.1 is class diagram represents the current state of the chat feature in the communication tool. It represents the state of the application architecture at the start of this project before modifications were made. As we start implementing more features and make significant updates to the system we will be updating this class diagram

**Communications tool video chat class diagram:**

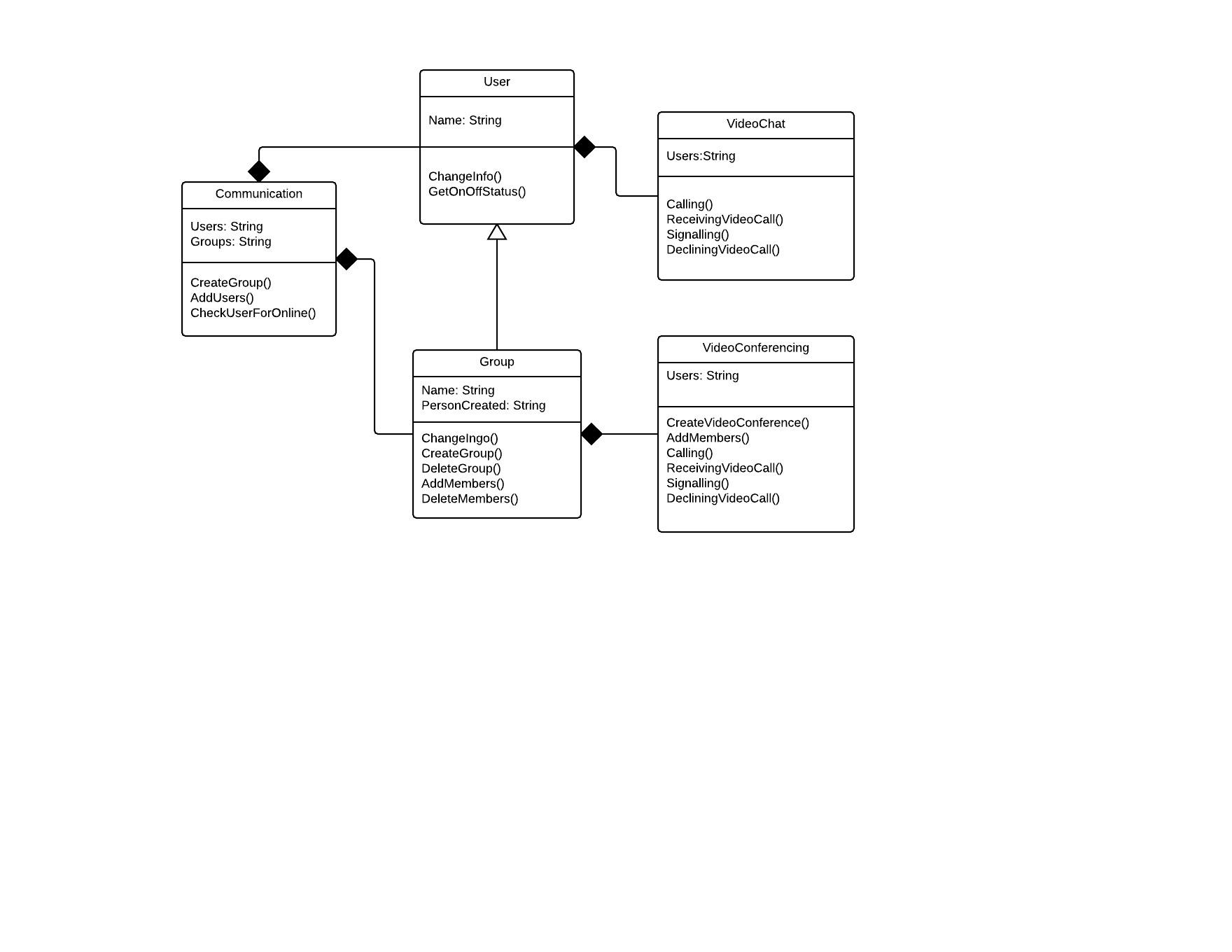


Figure 1.2 is a class diagram of the video chat application, which is being integrated into the existing communications tool. It displays the relationships between the users, groups and video chat entities and how they are composed.

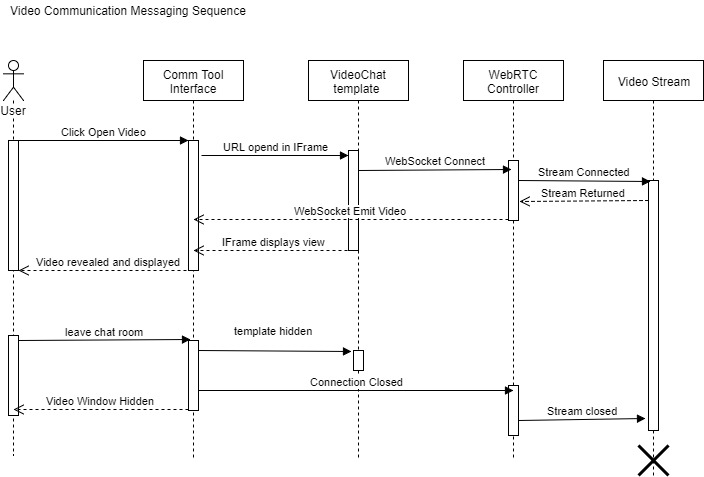
Frameworks:

Django (Python web framework)

Node.js (Javascript backend platform)

Express.js (Node JS framework for building simplified RESTful APIs and servers)

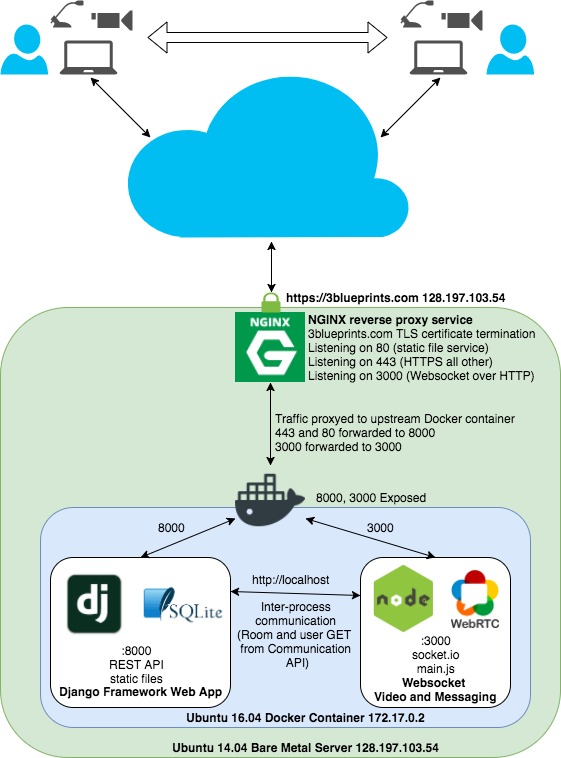
**The video conferencing application** functions across 4 layers, the communication tool interface, video chat template, webrtc controller and the video stream itself. Below is sequence diagram which lays out how a message travels across these layers from the end user, leading to the establishment and maintenance of a video steam inside of the 3 blueprints application.



**Production System Architecture:**

The production system is currently based on a docker container running the latest production release of the 3blueprints application.

When an merge is made into the main branch, a docker build is started on docker hub. If this build is successful, it is considered a candidate for production release. It can then be released onto the server and made publically available.



# Design Patterns

In this section, you shall describe any design patterns used in your software system.

# Classes and Methods

Automated class definition tools are still under analysis and review.

# References

# Glossary