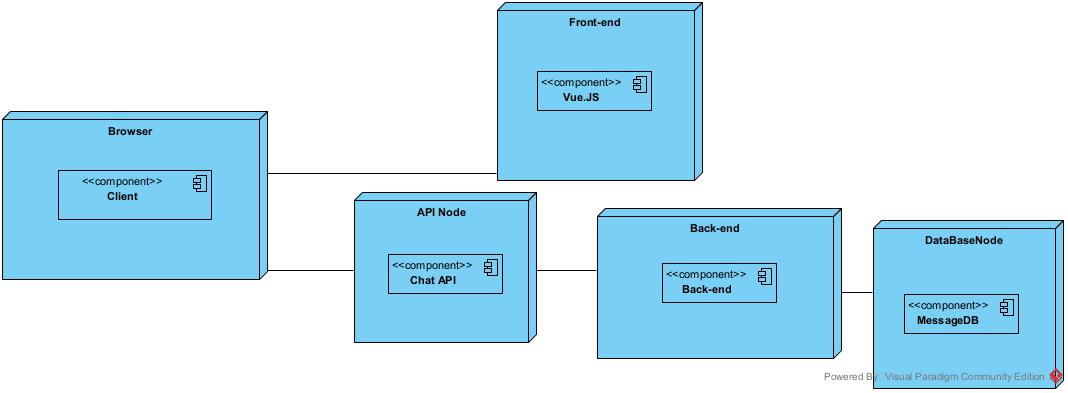
Distributed Software Architecture

## Context Diagram



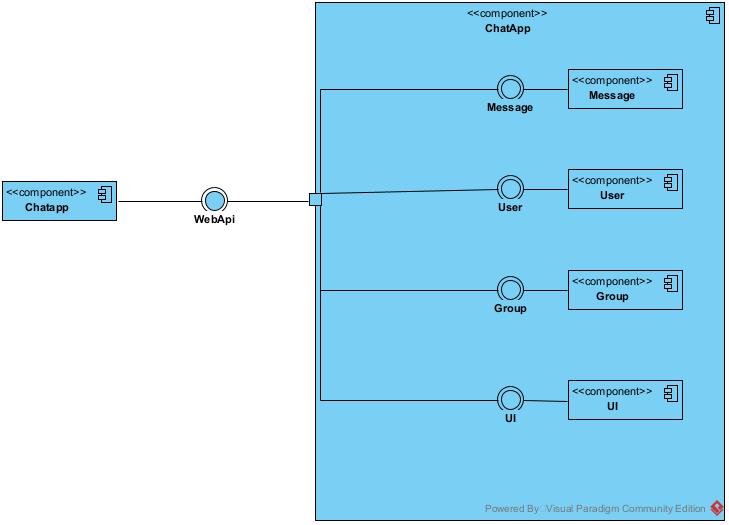
For my application there wil be users that use the system. There will be an admin who uses the system and the system will post and get data from a database.

## Deployment Diagram



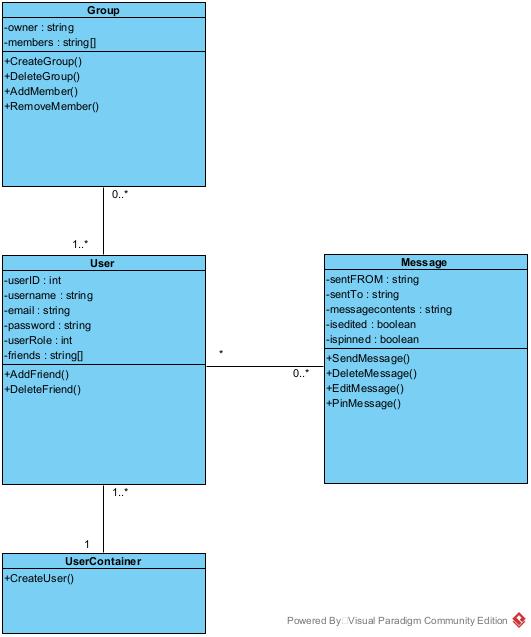
My application will be deployed in separate pieces. The Front and back-end will be separated. Behind the back end runs a database.

## Component Diagram



For my component diagram I have every component with its corresponding interface, each component is than linked to the gateway that goes to the client.

## Class Diagram



This is my class diagram, at the center is the user class. This class has a container that is used for creating a new user, this is done because a user can’t create itself. There are also classes for Messages and Groupchats.

What is a distributed software system?

A distributed software system means you have a system with multiple software components spread over multiple servers/computers but it is run on one single machine.

What is the difference between a monolith and a distributed software system?

A software system is monolith if the main components are interwoven(all build in the same program and running on the same system) instead of spread out over multiple micro servers.

Which advantages does a distributed software system offer?

A distributed software system is very reliable, for instance if one of the servers crashes it does not effect the other servers. It is also very scalable you can always add a new machine if it is necessary. And it is both runnable local and remotely.

Explain why your architecture is distributed

My architecture is distributed because that makes it safer against crashing. And it is spread over multiple platforms so if one part gets corrupted for instance the rest won’t be affected.

What are the important architectural principles and techniques when developing a distributed software system?

One of the vital skills of an architect is to be able to view the architecture from many **different standpoints**: each one of them individually might not be fully relevant, but combining them together gives a helicopter view of the product.