1 Introduction

1.1 Purpose

Purpose of this document is to describe Chat Service System. This paper will explain the key features of the system, as well as set of interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli, by presenting the general structure and a set of specific use cases. This document is intended for both the stakeholders and the developers of the system.

1.2 Scope of project

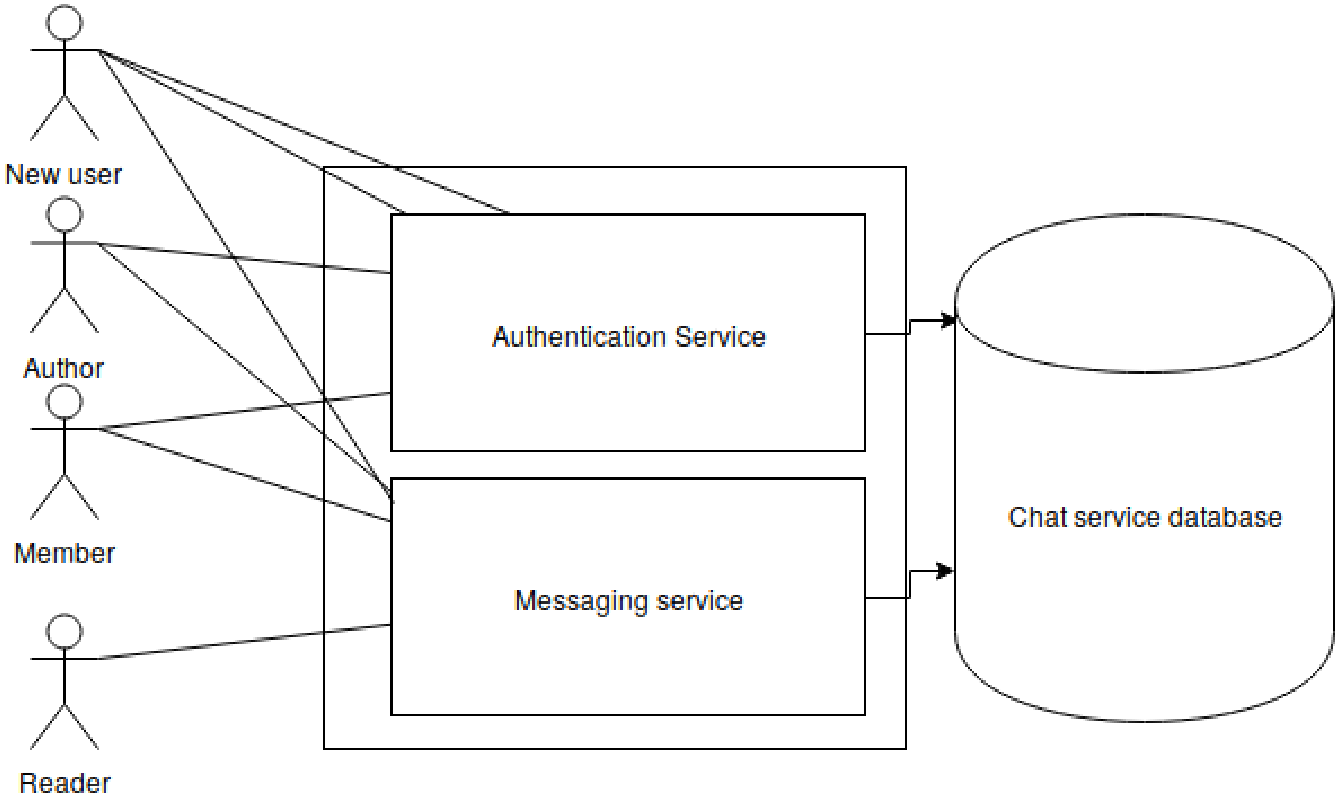
This system is a simple messaging that allows users to register a nickname and post a message that's bundled with such identifier. The system consists of a RESTFul service, a PostgreSQL database and a web-browser Single-Page Application. Database contains messages, users, and active sessions. User's session is defined by a pair of the authentication token and nickname stored in data base, that user is provided as a result of authentication process. User's nickname is bundled to the message through the process of verification of authentication token, that retrieves the respective nickname.

1.3 Glossary

Active Article	The document that is tracked by the system; it is a narrative that is planned to be posted to the public website.
Author	Authenticated user with an intent to post a prepared message to the board
Database	Collection of all the information monitored by this system.
Field	A cell within a form.
Historical Society Database	The existing membership database (also HS database).
Member	A member of the Chat service listed in the CSD database.
Reader	Anyone visiting the site to read messages.
Software Requirements Specification	A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. For example, this document.
Stakeholder	Any person with an interest in the project who is not a developer.
User	Author or Reader.

2 Overall description

2.1 System enviroment

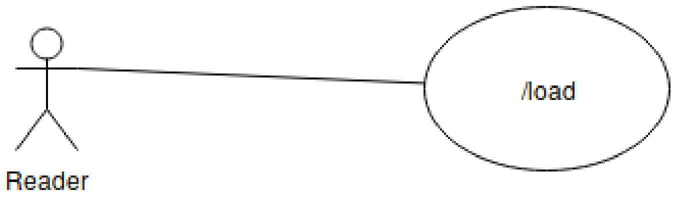


RESTFul service consists of two controllers — Authentication that hands out tokens, and Messaging that puts messages in DB and loads last 50. All users communicate through HTTP requests to specific parts of the server application, for example, /load to load messages

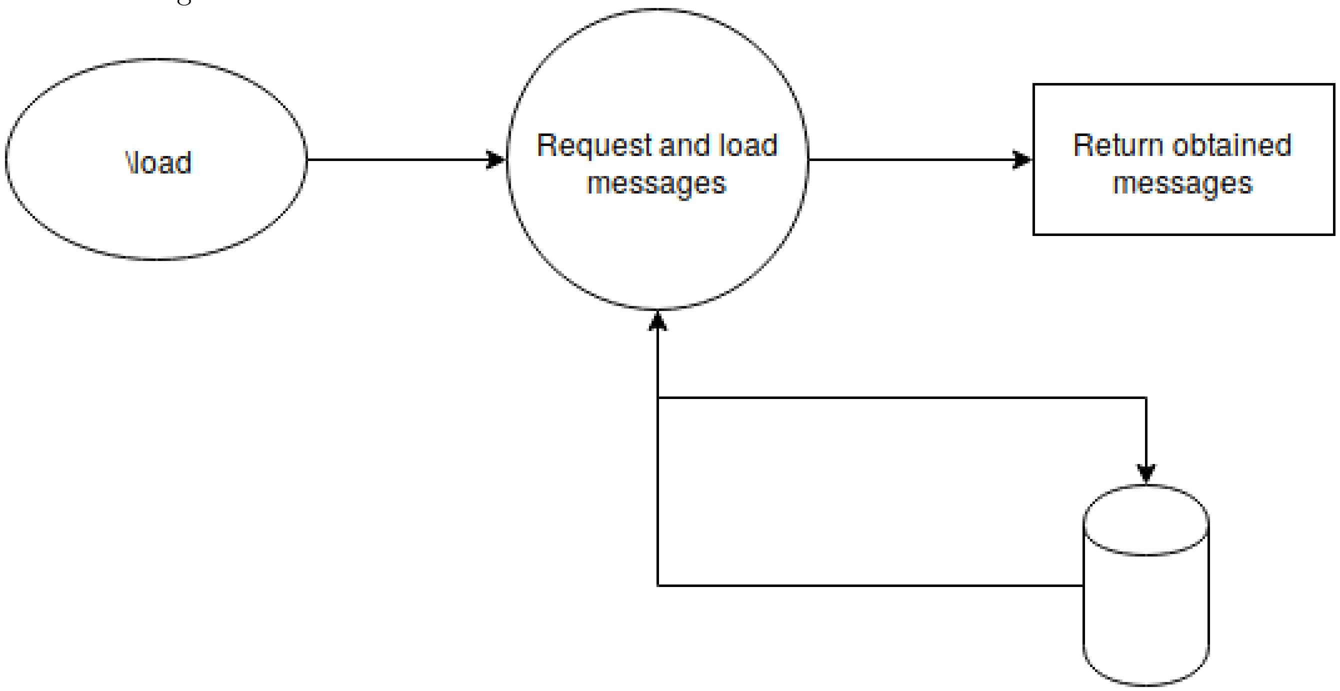
2.2 Functional requirement specification

This section outlines the use cases for each of the active readers separately. The reader has one use case, the author and the member have three and new user has to go through the process of registration, which is his fourth use case. This paper will present a chart of each user's interaction with requests to REST, and a diagram showcasing the process of each request

2.2.1 /load

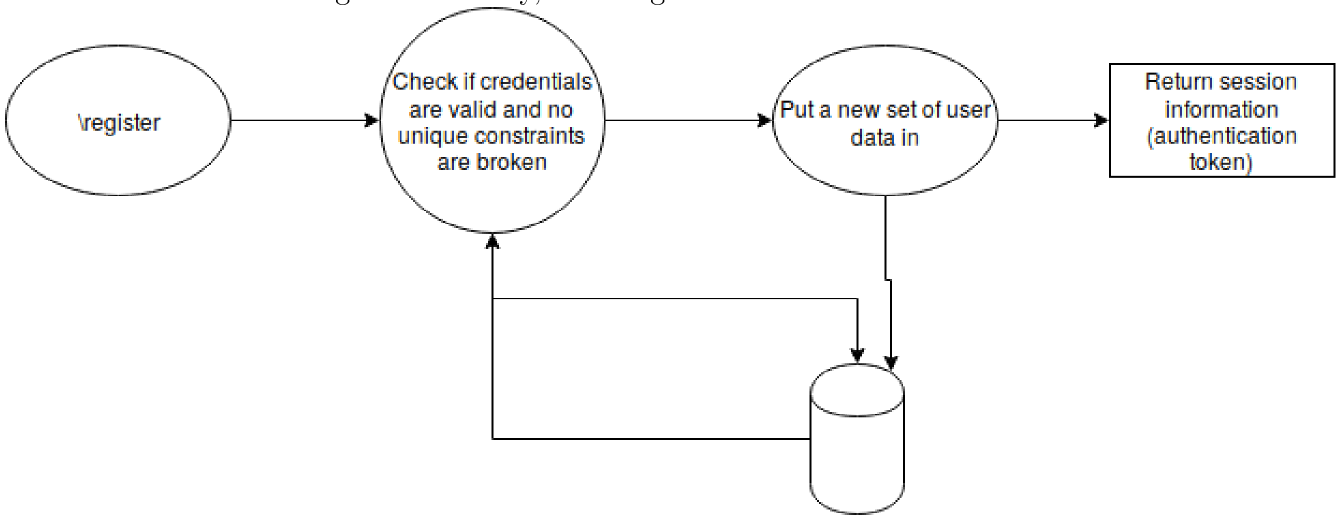


Loads messages from database.



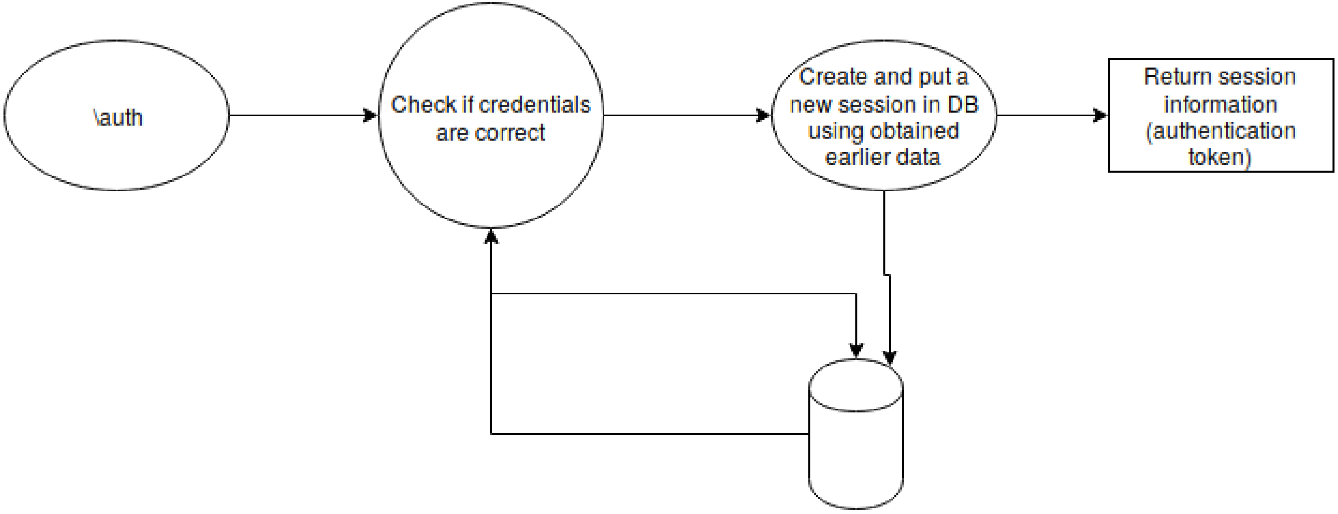
2.2.2 /register

Checks nickname and login availability, then registers user's desired credentials


\register
Check if credentials are valid and no unique constraints are broken
Put a new set of user data in
Return session information (authentication token)

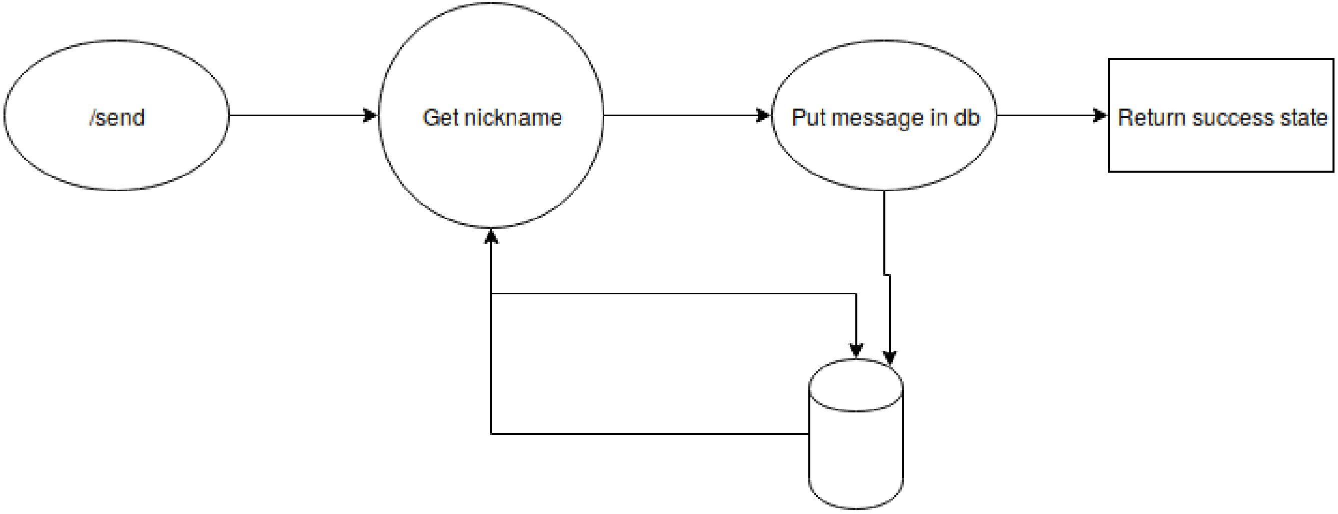
2.2.3 /auth

Authenticates the user and generates authentication token that's associated with user's nickname.

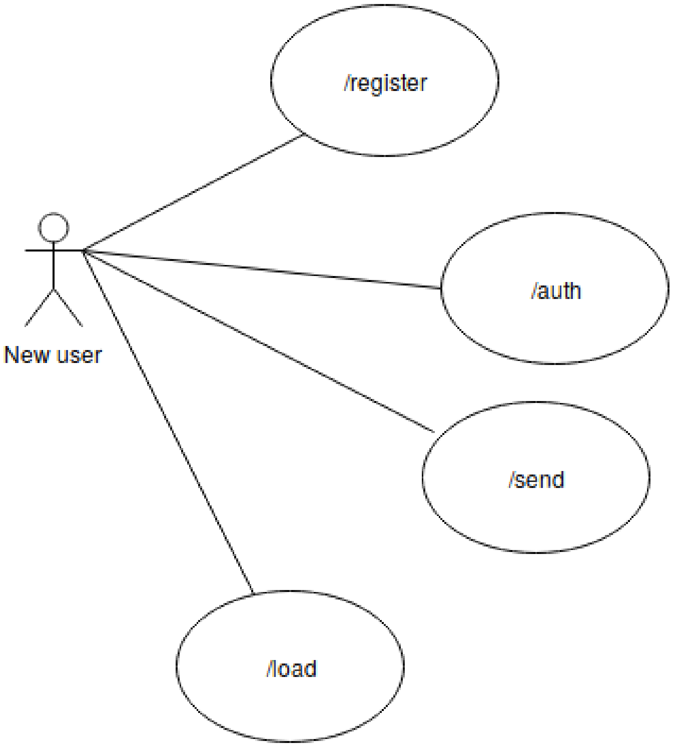

\auth
Check if credentials are correct
Create and put a new session in DB using obtained earlier data
Return session information (authentication token)

2.2.4 /send

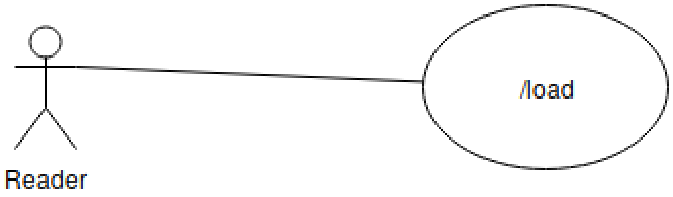
Retrieves nickname associated with user's token, and bundles it to the message before posting it to database.



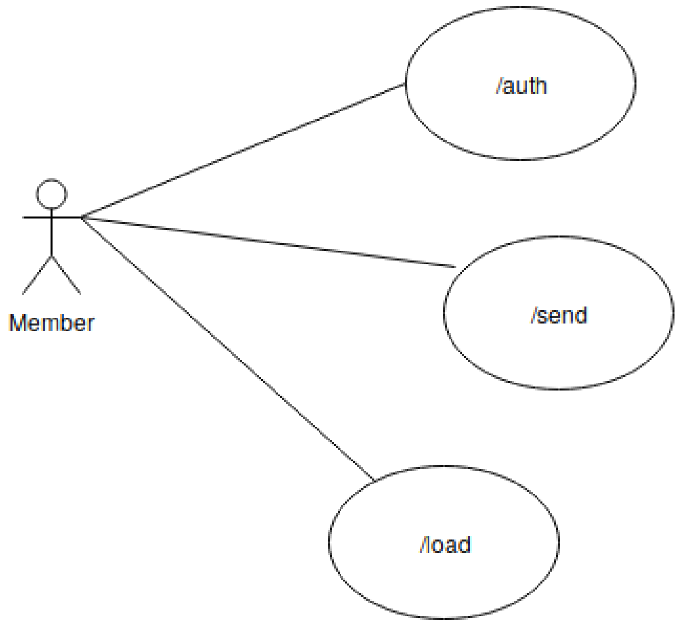
2.2.5 New user



2.2.6 Reader



2.2.7 Member/Author



2.3 Non-Functional requirements

The chat service can be installed on a server via on-Premise deployment, given a postgresql database and server are configured beforehand.