**Project Chorus**

Software Requirements Specification

|  |  |
| --- | --- |
| **Name** | **Student ID** |
| Furkan Doğan | 2098937 |
| Anxhela Ngjeliu | 2174753 |

**TABLE OF CONTENTS**

1. Introduction 4
   1. Purpose of the System 4
   2. Scope 4
   3. [System Overview 4](#_bookmark3)
      1. [System perspective 4](#_bookmark4)
         1. [System interfaces 5](#_bookmark5)
         2. User interfaces 6
         3. Hardware interfaces 8
         4. Software interfaces 8
         5. Communications interfaces 9
         6. Memory constraints 9
         7. Operations 9
      2. System functions 10
      3. User characteristics 10
      4. Limitations 11
   4. Definitions 12
2. References 13
3. Specific Requirements 13
   1. External interfaces 13
   2. Functions 16
   3. Usability requirements 21
   4. Performance requirements 21
   5. Logical database requirements 21
   6. Design constraints 22
   7. Software system attributes 22
   8. Supporting information 23

**FIGURES**

Figure 1: Context model… 5

Figure 2: Chorus Bot Interface … 6

Figure 3: Chorus’s Worker Interface … 8

Figure 4: External Interfaces Class Diagram… 13

Figure 5: Use case diagram… 16

Figure 6: Send Message Sequence Diagram… 18

Figure 7: Select Best Response Sequence Diagram… 19

Figure 8: Logical Database Class Diagram… 21

# TABLES

Table 1: Definitions 12

Table 2: Send Message Use Case Table 17

Table 3: Select Best Response Use Case Table 18

Table 4: Use cases descriptions 20

# Introduction

# Purpose of the System

Chorus is a crowd-powered conversational assistant powered by Crowdsourcing which allows users to receive assistance on any online task by maintaining a natural-language conversation. The purpose of the system is to establish a conversational interaction between crowd workers and users where a group of workers will work together to give users the best response about different tasks that may be accomplished over the web using information that the user is comfortable sharing.

# Scope

# The system contains an interface for workers where they can propose answers, vote and also take notes for future reference.

# The system uses a word matching tool to detect whether workers are using abusive language.

# The system uses a retainer pool to recruit workers in near real-time.

# The system uses a threshold mechanism to make sure that the necessary number of workers needed will arrive at each conversation task simultaneously.

# The system uses an explicit voting method where crowd workers can vote on each other’s responses.

# The system uses a multi-tiered reward scheme that rewards users according to the contribution they give such as proposing answers or voting answers of other workers.

# The system uses a working memory which provides information about previous conversations and where workers can change the importance of each piece of information by voting. The lines of information are sorted in descending order starting form the most important one.

# System Overview

# System perspective

# Chorus is not an element of a larger system but there are other systems it interacts with. One of them is Google Hangouts which provides the opportunity for crowd workers and users to communicate with each other. Another system is Amazon Mechanical Turk which helps Chorus to recruit workers for short periods of time.

# 

Figure 1: Context model

# System interfaces

**Google Hangouts API:** Chorus system uses Google Hangouts as a tool for communication between crowd workers and users. So, it is necessary to first sign in through Google Hangouts and then you will be able to communicate with the crowd agents.

**Amazon Mechanical Turk:** Chorus finds semi-anonymous workers recruited for short periods of time through Amazon Mechanical Turk. Workers should log in through MTurk system such that they will be able to enter the Chorus crowd workers interface.

**Account Authorization Interface:** To be able to use the Conversational Interface of Chorus you are required to fill a form and only after submitting this form you are able to start a conversation with the crowd agents.

**Web server interface:** This is used to make the connection between messages sent by users through Google Hangouts and then delivered to the crowd agents who use Amazon Mechanical Turk to read and respond to these messages.

# User interfaces

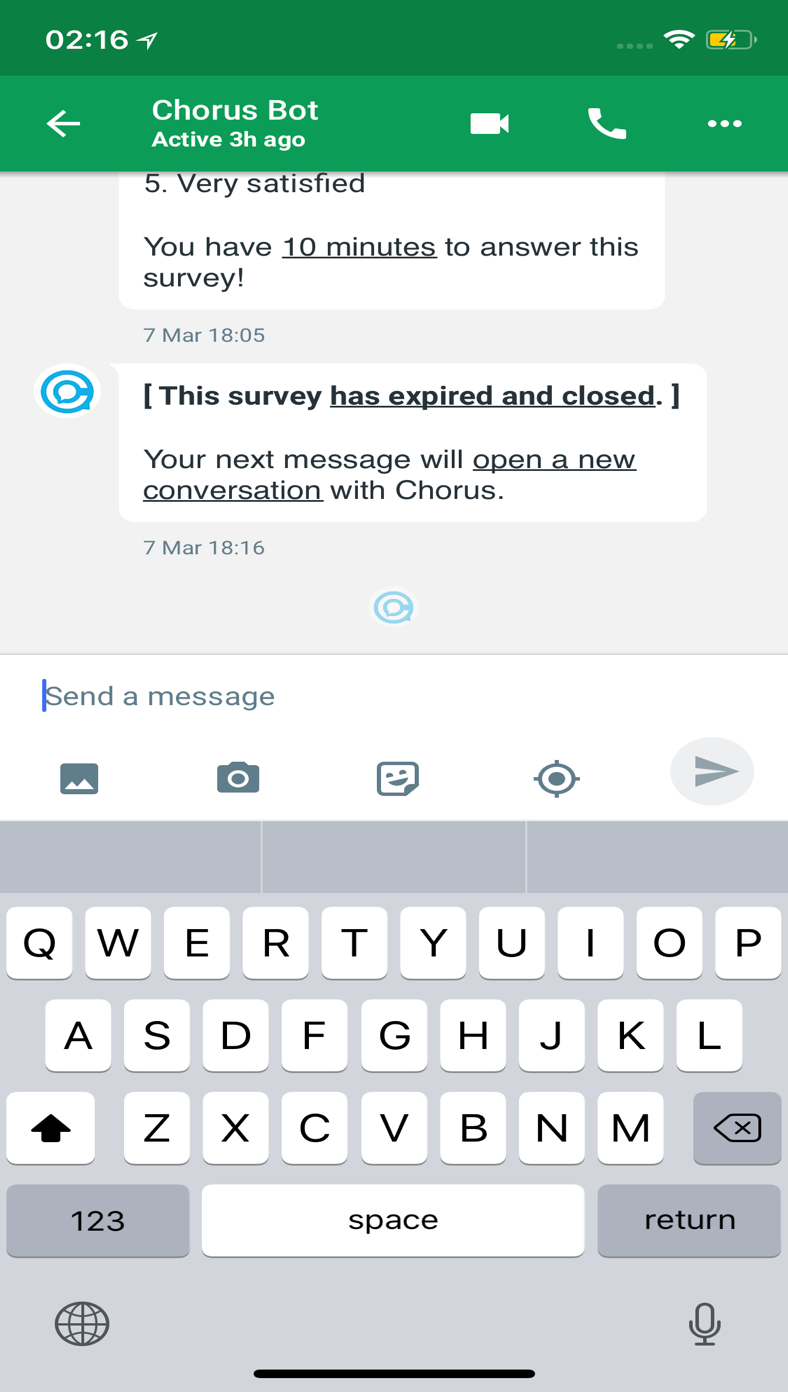
**Chorus bot:** Users are able to communicate with the crowd agents through the Chorus bot in Google Hangouts. First the user makes a request which is forwarded to the crowd and after the crowd has chosen the best response, only that answer is provided to the user and not all the responses that were suggested by the crowd agents.

Figure 2: Chorus Bot Interface

**Voting Interface:** Crowd agents are able to propose responses and also select reasonable responses by voting on other workers responses to be able to provide the best answer for the user. The system shows all the suggested responses to the worker and gives the worker the opportunity to vote on the response the seam to be the best. Worker is also able to retract the vote or change it. Also, if a worker disconnects their votes are removed.

**Working Memory Interface**: A memory space allows crowd agents to save important portions of conversations that helps in maintaining continuity even when a new worker has joined the conversation. Worker are able to vote on lines of information to make the most valuable lines appear first such that workers will be able to find important information easily.

**Scoring Interface:** Workers are rewarded based on their contribution of suggesting responses and also voting on responses of others. The workers are paid a small amount for each interaction with the interface, a medium amount for voting on an answer that it will later be chosen by the crows to be forwarded to the end user and a large reward when the worker suggests an answer that the crowd will eventually chose. To prevent the rewards from being abused workers are limited to only 3 contributions per user message.

**Admin Panel Interface:** This interface is used by the IT staff and researchers where they can keep maintenance of the system and also work on errors that the system may give.

# 

Figure 3: Chorus’s Worker Interface

# Hardware interfaces

Chorus runs on COTS devices, so it does not require any special hardware devices.

# Software interfaces

**Database Management System:** Chorus system uses a database to store data about workers to keep track about their contribution such that the system will be able to give them the right reward. Also, data about important information from previous conversations is kept such that this information will be sorted in descending order starting from the most voted information.

# Communications interface

Chorus System doesn’t interact directly with users and workers, but it interacts through Google Hangouts and Amazon Mechanical Turk. Both of them need Internet connection to allow users and workers to use Chorus system and both of them use HTTPS protocol.

# Memory constraints

Memory constraints are not a big issue for Chorus project. A memory size to keep track of some important information and to execute some basic database and web operation is sufficient.

# Operations

Operations provided by Chorus can be categorized as the following:

**Crowd workers operations:**

* Propose answers
* Select best response by voting
* Save key facts of conversations
* Vote on important information

**Users Operations:**

* Sign up
* Start the conversation
* Ask for information
* Score the conversation

**It staff:**

* Access data
* Check for errors
* Keep maintenance of the system

Operations will be explained further in Functions section (3.2).

# System functions

Users can use Chorus system by submitting the sign-up form and then they are able to start a conversation. Users can ask about assistance on different tasks by sending a message using the Chorus bot. The message is then delivered to the crowd workers who propose they answers and also vote on answers that seam more suitable to be delivered to the end user. According to these contributions, crowd workers are also rewarded by the system. Then the answer that has the greatest number of votes is delivered to the user. Also, workers save lines of information from the conversation that seams important for future reference. They also vote to decide which piece of information is the most important one so that the information will then be sorted showing the most important one first.

These functions will be explained further in Functions section (3.2).

# User characteristics

Chorus users can be categorized as the following:

**End users** who are ordinary people asking for assistance on different tasks. They have to be English speakers since they can only communicate in English with the crowd workers. Also, they have to know how to use Google Hangouts which is mandatory for having a conversation.

**Crowd workers** who are semi-anonymous people recruited for short periods of time from Amazon’s Mechanical Turk. They work online so they have to have some basic computer skills such that they will be able to search the web for tasks that they will be asked to give assistance for.

**Administrators** who can be divided into IT staff and researchers. They work to keep the maintenance of the system, keep the database up-to-date and also work on errors that the system can give.

# Limitations

1. **Regulatory policies:** Users identity such as user name or email address are kept hidden from the crowd workers.
2. **Hardware limitations:** Since Chorus runs on COTS devices, it does not have any hardware limitations.
3. **Interfaces to other applications:** The system must be compatible with other systems such as Google Hangouts and Amazon Mechanical Turk.
4. **Parallel operation:** Since operations are all basic and can all be executed on a single core machine, there is no need for parallel operations.
5. **Audit functions and Control functions:** There are no audit and control functions since they are handled by Amazon Mechanical Turk.
6. **Higher-order language requirements:** There are no higher-order language requirements.
7. **Signal handshake protocols:** Chorus system doesn’t directly interact with users or workers. It uses Google Hangouts to interact with users where https protocol is used, and it also uses Amazon Mechanical Turk to interact with workers where https protocol is also used.
8. **Quality requirements:** Priority of the system is to provide users the best responses that’s why using the voting system between workers and keeping them with rewards is very important for the maintenance of the system.
9. **Criticality of the application:** The system is not very critical since even if it crashes it will not be able to respond to the end user immediately, but it will be able to respond later.
10. **Safety and security considerations:** Security of workers accounts are covered by Amazon Mechanical Turk. Other than that, the database must be secure to keep the safety of data.
11. **Physical/mental considerations:** Anyone that has basic computer skills and be able to search the web can be a member user of Chorus.

# Definitions

|  |  |
| --- | --- |
| **End user** | Ordinary people that ask Chorus for assistance. |
| **Crowd worker** | Paid users of Chorus who propose answers for the end user. |
| **Crowd-powered** | Systems that combine computation with human intelligence, coordinated from large group of people who work online. |
| **Crowdsourcing** | The process getting work usually online from a crowd of people. |
| **Retainer pool** | Finding crowd workers who are paid to do some task in real time. |
| **Database** | A structured set of data. |
| **Multi-tiered** | Any application developed and distributed among more than one layer. |
| **Mturk** | Amazon Mechanical Turk |
| **Chatbot** | A computer program that attempts to simulate the conversation or "chatter" of a human being via text or voice interactions. |
| **API** | Application Programming Interface |
| **COTS** | Commercial Off-The-Shelf |

Table 1: Definitions

# References

**Chorus: A Crowd-Powered Conversational Assistant**

W.S. Lasecki, R. Wesley, J. Nichols, A. Kulkarni, J.F. Allen, J.P. Bigham.

In Proceedings of the ACM Symposium on User Interface Software and Technology (**UIST 2013**). St Andrews, UK. P151-162.

**"Is there anything else I can help you with?": Challenges in Deploying an On-Demand Crowd-Powered Conversational Agent**

Ting-Hao K. Huang, Walter S. Lasecki, Amos Azaria, Jeffrey P. Bigham.

In Proceedings of Conference on Human Computation & Crowdsourcing (**HCOMP 2016**), 2016, Austin, TX, USA.

# Specific Requirements

# External interfaces

Figure 4: External Interfaces Class Diagram

# Google Hangouts API

# req01: User shall be able to send message.

# req02: User shall be able to get message.

# req03: When user add Chorus, it should make connection between user and crowd workers.

# req04: User shall be able to manage his/her account.

# Amazon Mechanical Turk

# req05: Workers shall be able to get message.

# req06: Workers shall be able to post message.

# req07: While there is new conservation, it should assign enough workers.

# Account Authorization Interface

# req08: When the users submit their information, authorization interface shall send them info e-mail.

# Web server interface

# req09: Web server interface shall be to make connection between Google Hangouts API and Amazon Mechanical Turk.

# Chorus bot

# req10: User shall be able to send message to the crowd.

# req11: Worker shall be able to get message from user.

# req12: Chorus bot shall be able to send automatic message to user.

# Voting Interface

# req11: Workers shall be able to vote each other’s message.

# req12: Most voted message shall be chosen.

# req13: If vote numbers are equal one the message shall be chosen randomly.

# Working Memory Interface

# req14: Workers shall be able to save important messages.

# req15: Workers shall be able to access that messages.

# Scoring Interface

# req16: Scoring interface shall be able to grade the users.

# req17: Workers should be chosen by their scores.

# Admin Panel Interface

# req18: IT staff shall be able to access data.

# req19: IT staff shall be able to communication information of other staff members.

# req20: IT staff shall be able to see all logs.

# 

# Functions

# 

Figure 5: Use Case Diagram

|  |  |
| --- | --- |
| **Use case name** | Send message |
| **Actors** | User, Google Hangouts |
| **Description** | Users write his/her message and send it to the Chorus with Google Hangouts |
| **Data** | Chat history |
| **Pre-conditions** | User should sign up Chorus and sign in Google Hangouts to send message. |
| **Stimulus** | User write his/her message on Google Hangouts and click send button. |
| **Basic Flow** | Step 1 – User sign up Chorus.  Step 2 – User sign in via Google Hangouts.  Step 3 – User send his/her message to Chorus. |
| **Alternative Flow** | - |
| **Exception Flow** | If there is no internet connection user can not send message to Chorus. |
| **Post-conditions** | User waits for the response. |

Table 2: Send message Use Case Table

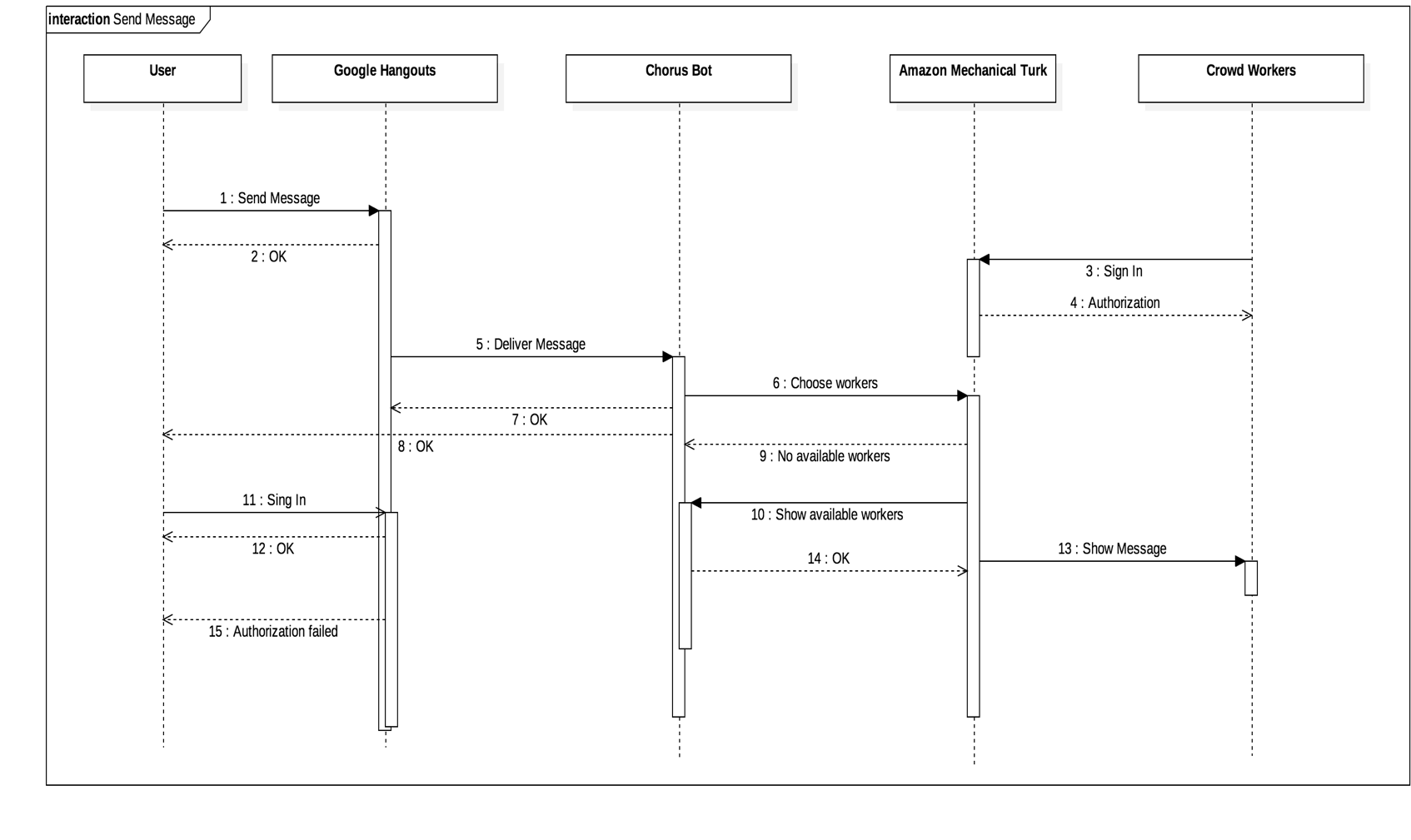


Figure 6: Send Message sequence diagram

|  |  |
| --- | --- |
| **Use case name** | Select best response |
| **Actors** | Crowd Workers, Collaborative Reasoning System |
| **Description** | Users ask for information and crowd workers try to  provide the best information. Crowd workers choose best  information with the help of collaborative reasoning  system. They vote each other’s responses and system choose best one. |
| **Data** | Chat history |
| **Pre-conditions** | Workers should sign in with Amazon Mechanical  Turk |
| **Stimulus** | User open Google Hangout and after sign in they just click Chorus Bot and they write their message and click send button. |
| **Basic Flow** | Step 1 – Crowd workers propose reasonable answers.  Step 2 – Workers vote for the best answer.  Step 3 – Collaborative Reasoning System select best response |
| **Alternative Flow** | - |
| **Exception Flow** | If number of votes of two responses is same, then system choose one of them randomly. |

Table 3: Select best response Use Case Table

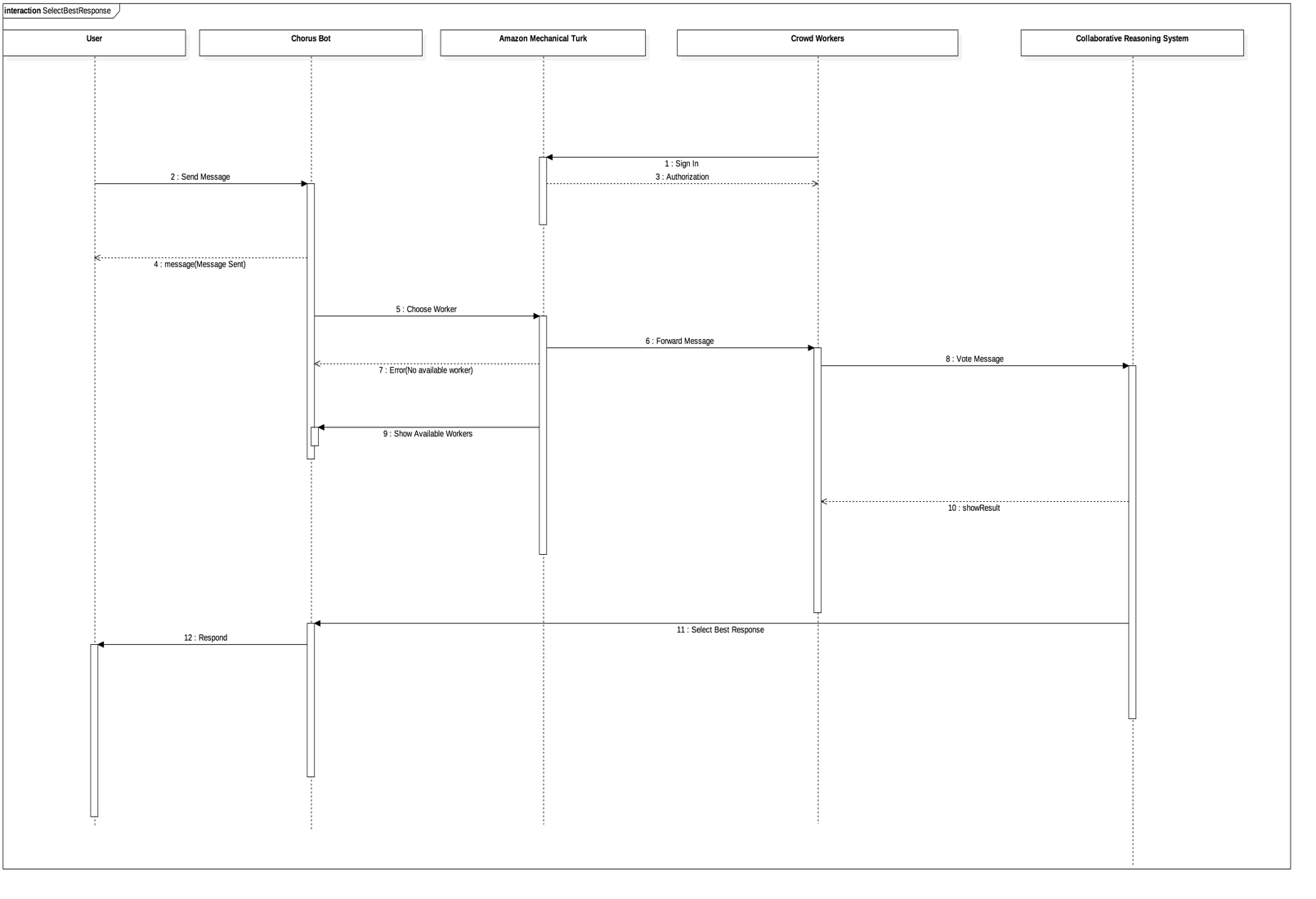


Figure 7: Select Best Response sequence diagram

|  |  |
| --- | --- |
| **Use case name** | **Description** |
| Score the conversation | User score the conversation via Google Hangout when conversation is ended. |
| Sign up | User sign up Chorus on Chorus’s website after filling sign up form. |
| Send automatic message | When user starts chat with Chorus and when the conversation ends system send automatic welcome and good-bye messages. |
| User sign in | User sign in to Chorus with adding Chorus on Google Hangouts. |
| Get response | After user send message, workers write their responses and system choose best one and send it to the user. |
| Receive message | After user send message, crowd workers receive that message on Amazon Mechanical Turk |
| Respond | Crowd workers write their responses for voting. |
| Worker sign in | Workers sign in to the Chorus on Amazon Mechanical Turk with their username and passwords. |
| Add summaries of key facts | Crowd workers add summaries of key facts and important statement with the help of Curated Memory System to keep crowd workers up-to-date with details of current or past conversation that may be relevant for providing future responses. |
| Keep workers up to date | Curated Memory System keep workers up-to-date with the help of summaries of key facts. |
| Reward workers | Dynamic Scoring System reward workers for interactions they made with the user. |
| Check system errors | IT staff can see the error logs happened in the Chorus system. |
| Access data | IT staff can access all the data in the Chorus system to improve the system. |

Table 4: Use cases descriptions

# Usability requirements

# req21: User shall be able to search old messages.

# req22: User shall be able to finish the conversation.

# req23: Conversation shall start when user send first message.

# req24: Workers shall be able to access other sources.

# Performance requirements

# req25: System shall respond at least 5000 users at the same time.

# req26: System shall transfer the message at most 3ms.

# req27: If number of users is there should be at least 2x crowd workers

# Logical database requirements

Figure 8: Logical Database Class Diagram

# req28: Conversation table shall be used while scoring workers.

# req29: WorkingMemory table shall be used while adding summaries of key facts.

# req30: Users shall be able to search for any key in the messages.

# req31: All responses shall be filtered, and user shall be able to saw just best response.

# req32: Conversation table shall store all the conservation between workers and users.

# req33: Database backup shall be done every day.

# req34: The information about staff members shall store in the StaffMember table.

# req35: The information about users shall store in the User table.

# req36: All error logs shall be added to ErrorLog table.

# req37: Only IT staff members shall be able to access ErrorLog table and filter them.

# req38: All statistical data shall be added StatisticalData table simultaneously.

# Design constraints

# req39: All coding should be done in standard C.

# req40: All conversation shall be stored for legal purposes.

# Software system attributes

* + 1. **Reliability**

**req41:** Maximum bug rate shall be at most 1bug/KLOC.

**req42:** Maximum time to repair the system shall be at most 20 minutes.

**req43:** Project shall be documented well for new members to understand everything clearly.

# Availability

# req44: The system shall be accessible all the time.

# req45: System should restart itself at most in 20 minutes if failure happens.

# Security

# req46: Database shall be able to defend itself from SQL injection.

# req47: While connection providing between user and crowd workers, connection shall be safe.

# req48: While making authentication system shall use HTTPS to ensure security.

# Supporting information

Chorus project is a project that helps people by assuring them online assistance for different tasks that might be accomplished through the web using information that the user chooses to share. The identity of the user though is kept hidden from the workers. The system uses Google Hangouts to be able to chat with users. It also has a worker’s interface where workers can suggest their responses, vote on responses of other workers and also take notes about important information. The connection between Chorus system and the workers is established by Amazon Mechanical Turk which makes possible the crowdsourcing of the system.