PROJECT CHORUS

Software Design Description

METU CENG350 2017-2018 Spring

Name	**Student ID**
Adil Kaan Akan	2171155
Azad Baykara	2171320

TABLE OF CONTENTS

1.Introduction.....5
1.1. Purpose ……..……5
1.2. Scope ……..……...5
1.3. Stakeholders and their concerns ……5
2. References.....6
3. Glossary 7
4. Architectural Views.....7
4.1. Context View ……...7
4.2. Composition View ……..19
4.3. Information View ……....22
4.3.1. Service Interfaces ……...23
4.3.2. CRUD Operations ……...28
4.4. Interface View ……30
4.4.1. Internal Interfaces ……...30
4.4.2. External Interfaces ……..33
4.4.2.1. User Interfaces ……33
4.4.2.2. System Interfaces ……37

FIGURES

Figure 1: Context Diagram…….....8
Figure 2: Use Case Diagram…….....…..9
Figure 3: Component Diagram.....19
Figure 4: Deployment Diagram.…….....….....….21
Figure 5: Service Interfaces Class Diagram….…….....…..….23
Figure 6: Database Class Diagram.…….....….....…….28
Figure 7: Sign in Sequence Diagram.…….....….....….32
Figure 8: Interface of end users ……34
Figure 9: Interface of crowd workers ……35
Figure 10: Register sequence diagram……...37
Figure 11: Provide Chat Environment sequence diagram ……..39

TABLES

Table 1: Glossary……7
Table 2: View error logs……9
Table 3: Send message……10
Table 4: Give feedback……11
Table 5: View data……11
Table 6: Add results of experiments……12
Table 7: Block malicious workers……12
Table 8: Fix bugs……13
Table 9: See the feedbacks……13
Table 10: Provide chat environment……14
Table 11: Register……15
Table 12: Sign in……16
Table 13: Report malicious workers……16
Table 14: Send data to memory……17
Table 15: Store data……17
Table 16: Vote to responses……18
Table 17: Service interfaces operation descriptions……23
Table 18: Service interfaces operation design ……25
Table 19: CRUD Operations……29

1.Introduction

1.1. Purpose

The purpose of this document is to provide a baseline for the project on which the system can be implemented. This document specifies the details of components to be built according to these details. The descriptions in this document identifies how the components which are subsystems of the Chorus be built clearly and definitely.

1.2. Scope

This documents describes the system at architectural level, with use cases and their descriptions, the interface classes of the system and how the components of the system interacts with the other components, the components and subsystems of the system with the hardware these components should be working on, and a complete detailed design of the database with how to create, build, update, read and delete operations are performed on the database.

1.3 Stakeholders and their concerns

End Users: End users are people from all over the world who want their problems be solved by the Chorus system. One of their concerns is to get their problems be solved completely and get informed about all aspects of the their problems. They also want that process take not much time.

Researchers: Researchers are scientists who are doing their who are doing their researches on the system. One of their concerns is to access the data easily and fast in a simple interface. They also want to add their results of their experiments to system.

Admins: Admins are people who will be maintaining the system and handling technical issues after the platform is put to work. Their concern is to have a simple interface for an admin panel that contains all functionality needed for them to view errors, to view the feedbacks that are given by end users, to block the malicious workers, and to make necessary changes on the system in order to resolve these errors and make system efficient and easily maintainable.

Crowd Workers: Crowd workers are people from Amazon Mechanical Turk service who are trying to answer the questions

that are asked by the end users. Their concern is to have a simple interface in order to send message, vote to responds, report the malicious worker to admins, send data to memory and view data from the memory.

Google Hangouts: Google Hangouts is a Google's messaging service and it is providing the chat environment for the end users. Their concern is to the process of the transmission of the messages between the Hangouts system and Chorus system be fast and encrypted.

System Developer: System developers are the people who will develop the software that will be used in the Chorus system. Their concern is having zero ambiguity with requirements and design specifications of the system. They want the software they develop to be clearly explained by the SRS and SDD.

2. References

This document is written with respect to the specifications of the document below:

IEEE standard for information technology--systems design--software design descriptions. (2009). New York, NY: Institute of Electrical and Electronics Engineers.

Other Sources:

Chorus: A Crowd-Powered Conversational Assistant

W.S. Lasecki, R. Wesley, J. Nichols, A. Kulkarni, J.F. Allen, J.P. Bigham. (2013). Retrieved from

https://web.eecs.umich.edu/~wlasecki/pubs/chorus.pdf

3. Glossary

Term	Definition
End user	People from world that send message to the Chorus
Crowd workers	People working for the Chorus system trying to answer to questions
Admins	People trying to maintain the Chorus system
Researchers	People doing their researches on the Chorus system
DBMS	Database Management System
HDD	Hard Disk Drive
SATA	Serial ATA, a computer bus interface
Database file	.db file
AUTHDATA	Authorization data
RAID	Redundant Array of Independent Disks
URL	Uniform Resource Locator, web address

Table 1: Glossary

4. Architectural Views

4.1. Context View

In this viewpoint all use cases of the system ares specified with detailed step-by-step descriptions of the basic flow, along with an alternative flow and/or an error flow if applicable. These description tables describe in detail how the system should work in the different scenarios and they can be guideline for the system developers who will develop that functions. The Context Diagram below shows the general exchanges between actors and the system, whereas the Use Case Diagram shows how different actors, namely different types of users or external systems interact with Chorus System through use cases.

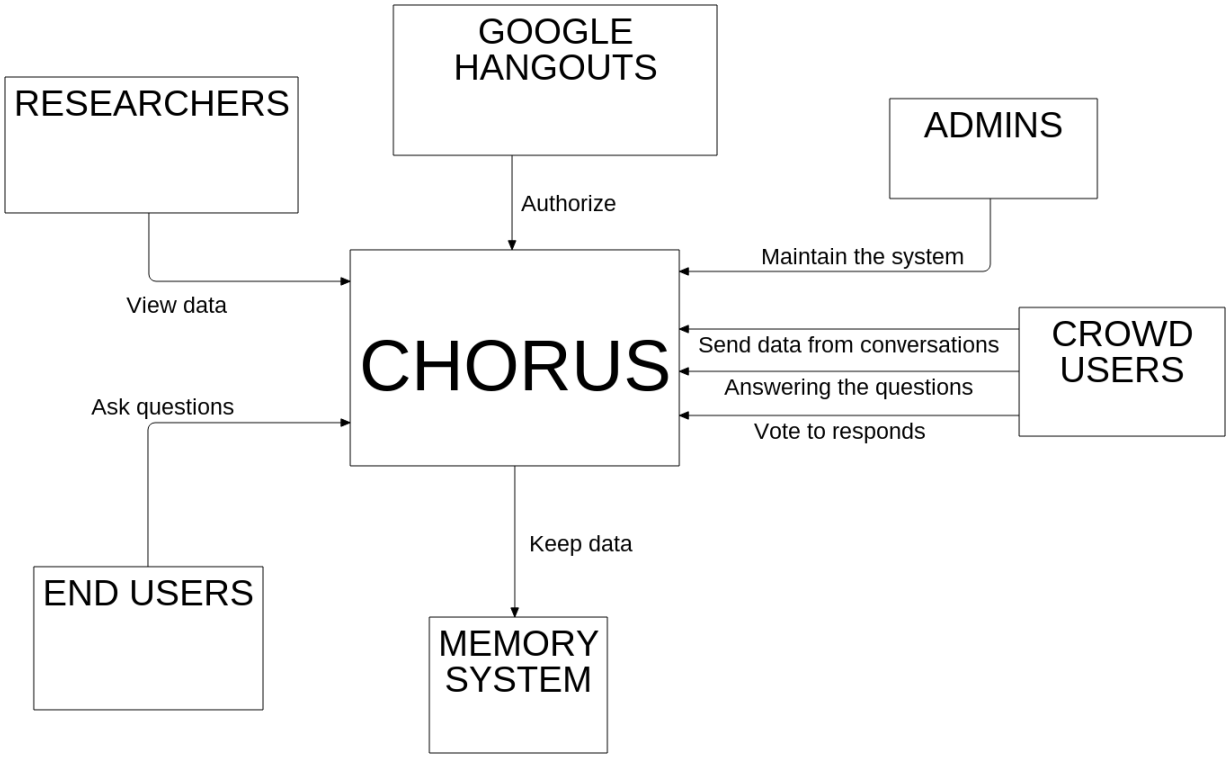

Figure 1: Context Diagram

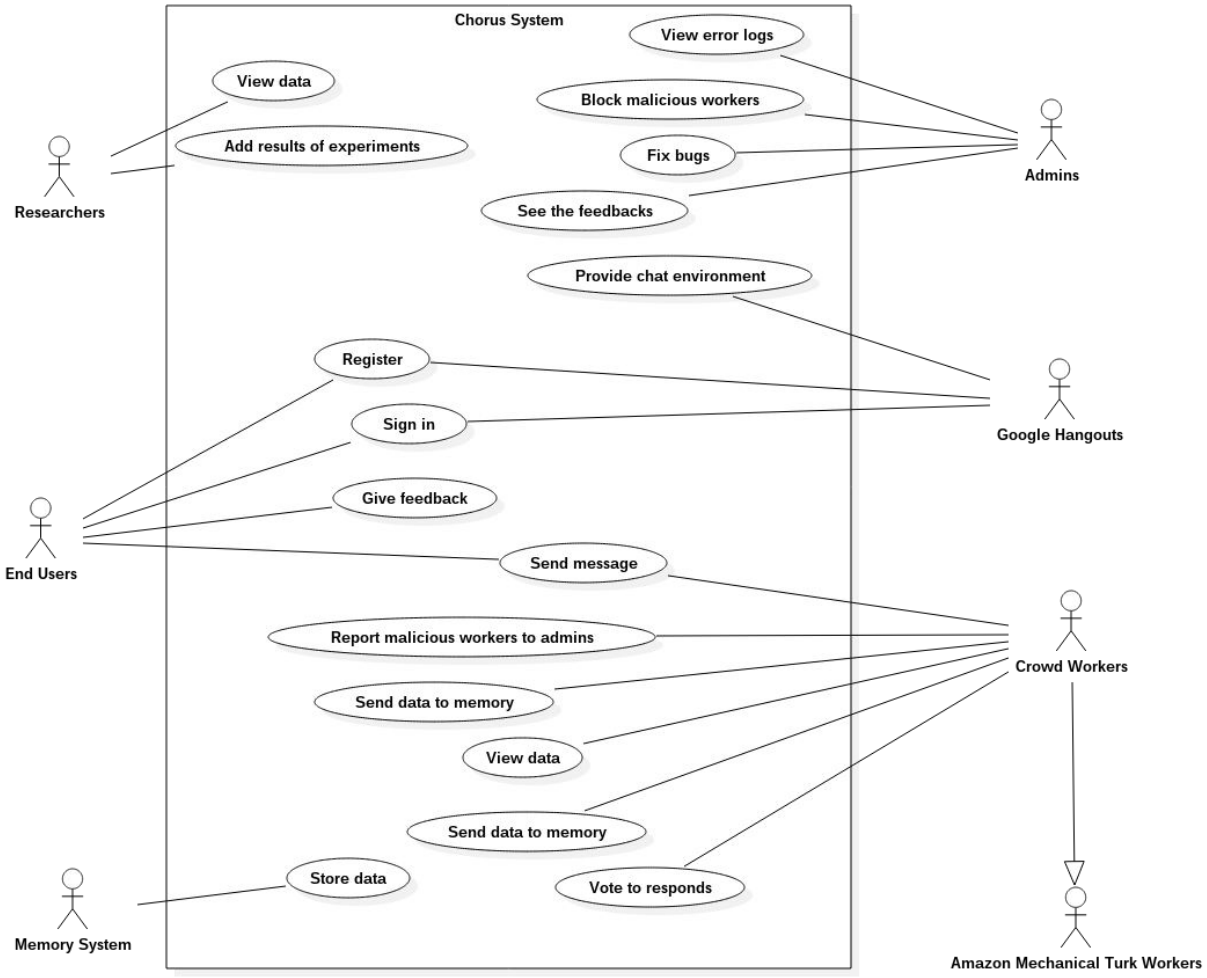

Figure 2: Use Case Diagram

Use case name	View error logs
Actors	Admins
Description	Admins can view the errors logs happened in the Chorus system.
Data	Error logs
Preconditions	The person who views must be an admin.
Stimulus	Admin presses “View error logs” button
Basic Flow	Step 1 - Request is received by Chorus system. Step 2 - Error logs are shown to the Admin on a special screen.

Table 2: View error logs

Use case name	Send message
Actors	End user, Crowd worker
Description	End user sends messages by using the hangouts interface. Crowd workers send their messages and the best message, which have the most upvotes, will be shown in the interface.
Data	Message that end user and crowd workers send, how many upvotes the messages that crowd workers send
Preconditions	Message shall not be empty, sign in to Google Hangouts
Stimulus	End user and crowd workers presses enter key after they write their messages
Basic Flow	Step1- After seeing the end user's message, crowd workers give their responses. Step2- The most upvoted message will be sent to the end user
Alternative Flow	Step 2 - If messages that crowd workers send have equal upvotes, then all messages that have equal upvotes will be send to the end user.
Exception Flow	-
Postconditions	The message is shown in the interface.

Table 3: Send message

Use case name	Give feedback
Actors	End user

Description	After completing the conversation, there will be shown a survey that evaluates the performance of the chorus. Users give their feedbacks about the conversation.
Data	Survey message, answer of the survey
Preconditions	The conversation must be closed by crowd worker
Stimulus	User presses a number 1-5 and send it.
Basic Flow	Step 1 - User writes a number 1-5 and send it. Step 2 – Feedback is sent to admins.
Alternative Flow	-
Exception Flow	-
Postconditions	The feedback is sent to admins.

Table 4: Give feedback

Use case name	View data
Actors	Researchers, crowd workers
Description	Researchers and crowd workers can view the data which crowd workers provide from the conversation they make with users.
Data	Data obtained from the conversations which crowd workers enter
Preconditions	The person should be a researcher in order to view data.
Stimulus	They press “View data” button.
Basic Flow	Step 1 - They send a request to view the data. Step 2 - The data is available on the screen.
Alternative Flow	-
Exception Flow	-
Postconditions	The requested data is shown to the researcher or crowd worker.

Table 5: View data

Use case name	Add results of experiments
Actors	Researchers
Description	Researchers examine the data entered by crowd workers and obtain some results which they will add afterwards.
Data	Results obtained from the experiments.
Preconditions	The user who adds the results of experiments must be a researcher.
Stimulus	The researcher presses “Save results” button after writing the results.
Basic Flow	Step1- Researchers send a request to add the results they prepare. Step2- The results sent by researchers are saved to the database.
Alternative Flow	-
Exception Flow	-
Postconditions	The results which the researchers send are kept in the database.

Table 6: Add results of experiments

Use case name	Block malicious workers
Actors	Admins
Description	Admins can block the crowd workers who abuse their job.
Data	Crowd worker who has been blocked
Preconditions	The person who is going to block a worker has to be an admin.
Stimulus	Admin presses “Block” button.

Basic Flow	Step 1 – Blocking a crowd worker request is received by Chorus system. Step 2 – The crowd worker abusing his/her job is blocked from the system.
Alternative Flow	-
Exception Flow	If the person who will be blocked is already blocked by another admin, a message box is shown saying the user is already blocked.
Postconditions	A success message is shown on the screen.

Table 7: Block malicious workers

Use case name	Fix bugs
Actors	Admins
Description	Admins fix the parts which is not functioning properly.
Data	-
Preconditions	Only admins have the right to fix the bugs.
Stimulus	Admin presses “Fix” button.
Basic Flow	Step 1 – Admin clicks “Fix” button. Step 2 – Admin is redirected to the panel where he/she edits the system. Step 3 – Admin makes the necessary changes before saving the system.
Alternative Flow	-
Exception Flow	-
Postconditions	Edited system is shown on the screen.

Table 8: Fix bugs

Use case name	See the feedbacks
Actors	Admins

Description	Admins can see the feedbacks given by the users about the conversation they made with crowd workers.
Data	Feedbacks, crowd workers who are given feedback
Preconditions	The user must be an admin to view the feedbacks.
Stimulus	Admin presses “See feedbacks” button.
Basic Flow	Step 1 – The system receives the request sent by the admin. Step 2 – Data about the feedbacks is received from the database. Step 3 – Data is available for the admins on the screen.
Alternative Flow	-
Exception Flow	If there is no new feedbacks given by the users, a message box is displayed saying no feedbacks available.
Postconditions	Feedbacks are shown to the admin.

Table 9: See the feedbacks

Use case name	Provide chat environment
Actors	Google Hangouts
Description	Google Hangouts enables users to interact with the Chorus system.
Data	Feedbacks, crowd workers who are given feedback
Preconditions	The user must be an admin to view the feedbacks.
Stimulus	Admin presses “See feedbacks” button.
Basic Flow	Step 1 – The system receives the request sent by the admin. Step 2 – Data about the feedbacks is received from the database. Step 3 – Data is available for the admins on the screen.

Alternative Flow	-
Exception Flow	If there is no new feedbacks given by the users, a message box is displayed saying no feedbacks available.
Postconditions	Feedbacks are shown to the admin.

Table 10: Provide chat environment

Use case name	Register
Actors	End users, Google Hangouts
Description	Ends users have to register to Google by creating a gmail account.
Data	First and last name, username, password, birthday, gender
Preconditions	-
Stimulus	The user presses “Create an account” button.
Basic Flow	Step 1 – The user is redirected to the registration page. Step 2 – Information boxes are filled by the user. Step 3 – A new account is created for the user.
Alternative Flow	-
Exception Flow	If any of the mandatory informations is left empty, user is informed to fill them. The user is also notified if any information he/she entered is unavailable.
Postconditions	The user is registered to Google.

Table 11: Register

Use case name	Sign in
Actors	End users, Google Hangouts
Description	End users should be signed in to start a chat with Chorus.
Data	Email, password

Preconditions	User must have an account already.
Stimulus	User presses “Sign in” button.
Basic Flow	Step 1 – The user is redirected to the sign in page. Step 2 – The user enters email and password. Step 3 – The user is signed in to Google.
Alternative Flow	-
Exception Flow	If the user enters either email or password wrong, then the user is informed that his/her email or password is not correct.
Postconditions	The user is signed in to Google.

Table 12: Sign in

Use case name	Report malicious workers to admins
Actors	Crowd workers
Description	Crowd workers can report other workers who abuse their job.
Data	Crowd worker who has been reported
Preconditions	Crowd workers can report other workers.
Stimulus	Crowd workers press “Report” button.
Basic Flow	Step 1 – Reporting a crowd worker request is received by Chorus system. Step 2 – Report is directed to the admins who will consider blocking the crowd worker.
Alternative Flow	-
Exception Flow	-
Postconditions	A message box indicating the crowd worker is successfully reported to the admins.

Table 13: Report malicious workers to admins

Use case name	Send data to memory

Actors	Crowd workers
Description	Crowd workers send data which they obtain from the conversations made with end users.
Data	Data obtained from conversations
Preconditions	The person who is going to send the data must be a crowd worker.
Stimulus	Crowd worker presses “Send data” button.
Basic Flow	Step 1 – The request is received by the system. Step 2 – Data is redirected to the memory. Step 3 – Sent data is taken by memory.
Alternative Flow	-
Exception Flow	-
Postconditions	Data is sent to the memory.

Table 14: Send data to memory

Use case name	Store data
Actors	Memory system
Description	Memory stores the coming data.
Data	The data sent by the crowd workers.
Preconditions	There should be empty space in the memory.
Stimulus	-
Basic Flow	Step 1 – Request to store data is received by the memory. Step 2 – Sent data is stored in the memory.
Alternative Flow	-
Exception Flow	If the current data is already present in memory, it is not accepted by the memory.
Postconditions	Data is stored in the memory.

Table 15: Store data

Use case name	Vote to responses
Actors	Crowd workers
Description	Crowd workers vote for the responses they send to choose the best one for the end user.
Data	All the responses crowd workers sent, number of upvotes each response got
Preconditions	-
Stimulus	Crowd worker presses “Vote” button for the response they want to vote.
Basic Flow	Step 1 – Available responses are shown to the crowd workers. Step 2 – Crowd workers choose the response they think is the best. Step 3 – Votes of crowd workers are received by the system.
Alternative Flow	-
Exception Flow	-
Postconditions	All crowd workers have submitted their votes.

Table 16: Vote to responses

4.2. Composition View

In this viewpoint, components of the system and their functionalities will be shown from top-level point of view. Design entities of the components, their interactions and the hardware that these entities are working on can be seen in the composition and deployment diagrams below.

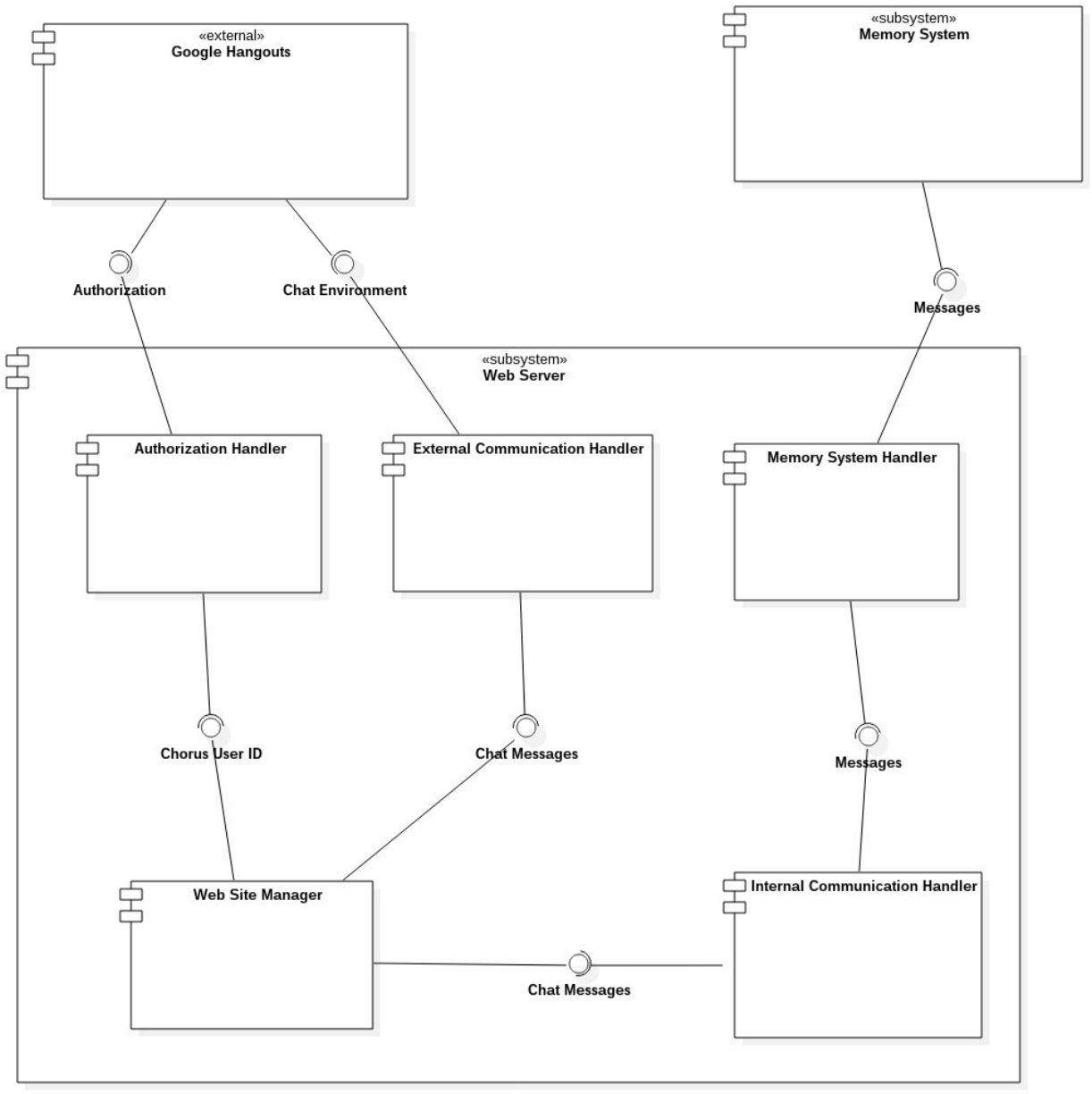

Figure 3: Component Diagram

Design Rationale:

- The component named Website Manager handles all the Chorus' system functionalities such as querying data from the memory system in order to analyze it to improve the Chorus system, sending the data obtained from the conversations to the memory system, or controlling authorization issues. Since both user interface and crowd worker

interface have similar functionalities, one component is enough to handle them.

- There are several components in the Web Server subsystem, some of which interact with the Google Hangouts external system.
- After the quest of sending data is made, Web Server transfers this data to the Memory system handler through the Internal Communication Handler. Then, then the data is handed in to the Memory System in a suitable format to be stored.
- End users are able to use Chorus as long as they are logged in to the system. In order to do so, Authorization Handler component provides an interface to the Google Hangouts to send required data for authorization.
- Since Chorus is available for use through Google Hangouts platform, end users are provided a chat environment via External Communication Handler component when they request to start a conversation with the system.

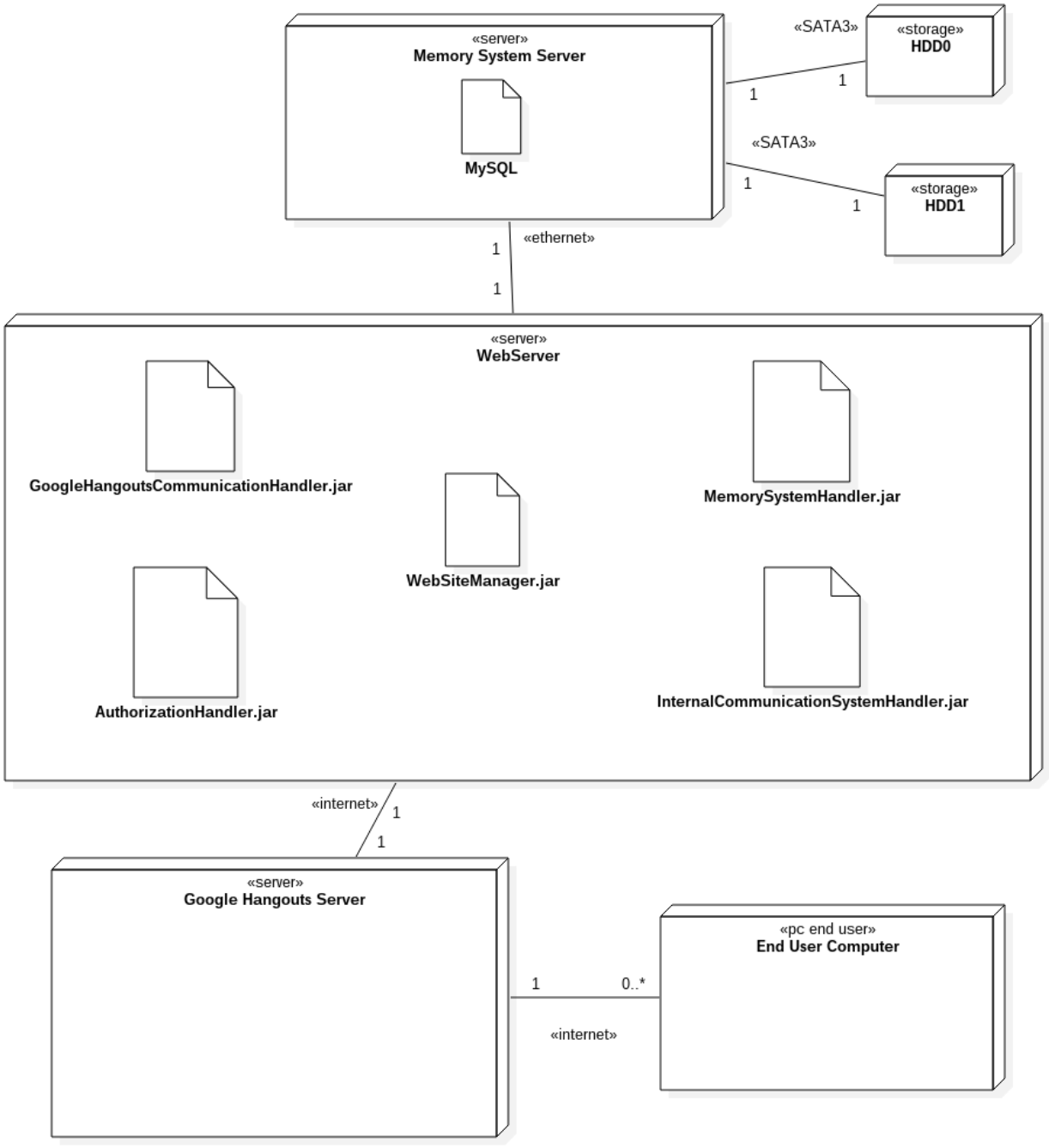

Figure 4: Deployment Diagram

Design Rationale:

- There will be some changes in the Chorus system based on the data in the Memory System Server. The system, therefore, should be open to change. Having separate Memory System Server and Web Server makes it easier as the only change needed is about the former.
- The Memory System Server has 2 hard drives, which increases the system safety. That is, whatever gets written to the first drive also stored in the second one. In this way, if something happens to one of the drives, the data in it will be still available from the other one.
- As a database management system, MySQL is preferred because it is a stable, reliable and powerful solution with advanced features such as data security and high performance.
- The connection between Web Server and Google Hangouts Server uses the Internet, but there is no security problem since it uses the HTTPS connection. Also, connection between client computer and Google Hangouts Server uses the HTTPS connection to prevent any security problems.

4.3. Information View

In this view, how information is stored in the database with the use of classes and the relationship among these defined classes are demonstrated in class diagrams. Database components and classes have also create, read, update and delete functionalities.

4.3.1. Service Interfaces

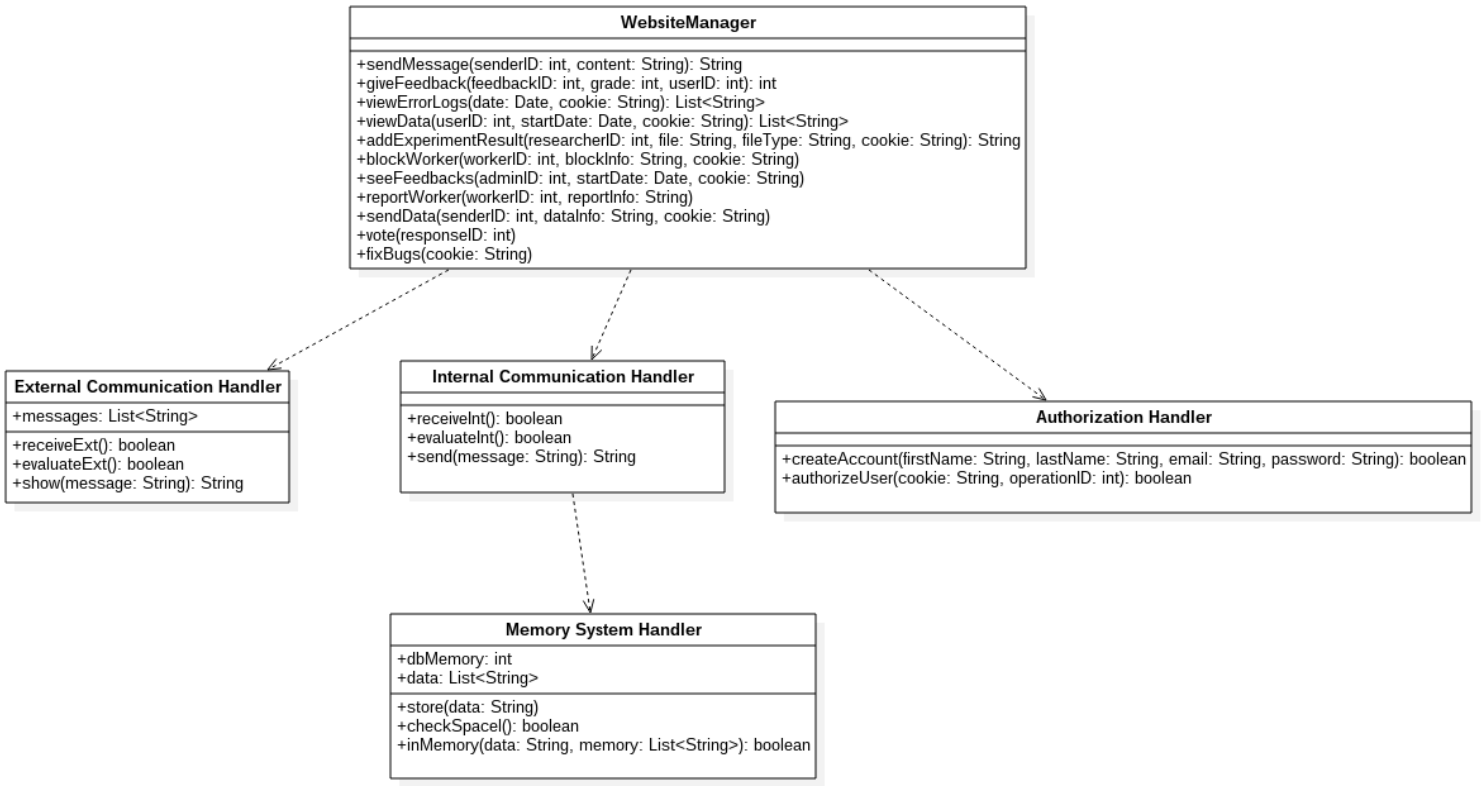

Figure 5: Service Interfaces Class Diagram

Operation	Description
sendMessage	End users and crowd workers are able to send message to each other in order to communicate.
giveFeedback	End users have a chance of grading each conversation at the end by giving points between 1 to 5.
viewErrorLogs	Admins are able to view the error logs in the system in order to take the necessary actions.
viewData	When researchers and crowd workers request to view the data in the memory, this operation queries the database and shows the data to them.
addExperimentResult	Researchers can add the results which they obtain from the

	conversations.
blockWorker	Admins are able to block the crowd workers abusing their jobs.
seeFeedbacks	Admins can view the feedbacks of each conversation between the end users and the crowd workers.
reportWorker	Crowd workers are able to report other workers to the admins.
sendData	Crowd workers can make a request to send the data that they obtain from the conversations they made to the memory .
vote	Crowd workers can vote to the messages that. other crowd workers come up with
fixBugs	Admins are able to fix the bugs in order to make the system run properly.
receiveExt	Requests which are sent from Web Site Manager component and about Google Hangouts are received by this operation.
evaluateExt	This operation processes the requests related to Google Hangouts.
show	This operation shows the messages in the chat environment provided by Google Hangouts.
receiveInt	Requests which are sent from Web Site Manager component and about Memory System are received by this operation.
evaluateInt	This operation processes the requests related to Memory System.
send	This operation transmits the data to the Memory System Handler in a suitable format.
store	If no error occurs during the lifetime of this operation, present data is stored in the memory system.
checkSpace	This operations checks whether there is enough space for the data requested to be stored , and if not informs the admins accordingly.
inMemory	This operation checks if the data given is already in the database or not.

createAccount	When this operation is used, the user is redirected to the Google's registration page.
login	As end users need to login to the system in order to start a conversation, this operation redirects them to the Google's login site.

Table 17: Service interfaces operation descriptions

Operation	**Inputs**	**Outputs**	**Exceptions**
sendMessage	senderID content	Message	Google Hangouts server not available
giveFeedback	feedbackID grade userID	Grade	Google Hangouts server not available
viewErrorLogs	Authorization cookie date	Error List	Authorization failed Database server not available
viewData	Authorization userID startDate	Resultant data at given date after querying the database	Authorization failed Database server not available
addExperimentResult	Authorization researcherID file fileType	Message operation OK or failed	Authorization failed Database server not available
blockWorker	Authorization workerID blockInfo	Message operation OK or failed	Authorization failed Worker not found
seeFeedbacks	Authorization adminID startDate	Feedback information	Authorization failed Database server not available
reportWorker	workerID reportInfo	Message operation OK or failed	Google Hangouts down Worker not found
sendData	Authorization senderID dataInfo	Message operation OK or failed	Authorization failed Database server not available

vote	responseID	Message operation OK or failed	Google Hangouts server not available
fixBugs	Authorization	Message operation OK or failed	Authorization failed
receiveExt	-	-	Data corrupted
evaluateExt	-	Message operation OK or failed	Invalid data
show	message	Message content	Google Hangouts server not available
receiveInt	-	-	Data corrupted
evaluateInt	-	Message operation OK or failed	Invalid data
send	message	-	Database server not available
store	data	-	Database server not available
checkSpace	-	Memory status	Database server not available
inMemory	data memory	-	Database server not available
createAccount	firstName lastName email password	-	Google server not available
authorizeUser	cookie operationID	Authorization status	Google server not available

Table 18: Service interfaces operation design

Design Rationale:

- There is not a direct way of dealing with users via the given operations. A handler class is, therefore, defined to establish a communication between Chorus and Google Hangouts external system.
- Instead of having many classes each of which directly interacts the database, interaction with the database takes place through only the Website Manager, which makes the design simpler.
- Each time a user makes a request that needs authorization, Authorization Handler is asked to find out whether the user has right to do the action or not. The reason is that the Chorus system does not hold the users' authorization status, so this type of operations takes an authorization cookie as parameter.

Both External Communication Handler and Internal Communication Handler have operations just to receive the request and to evaluate them accordingly, so they do not have any input.

4.3.2. CRUD Operations

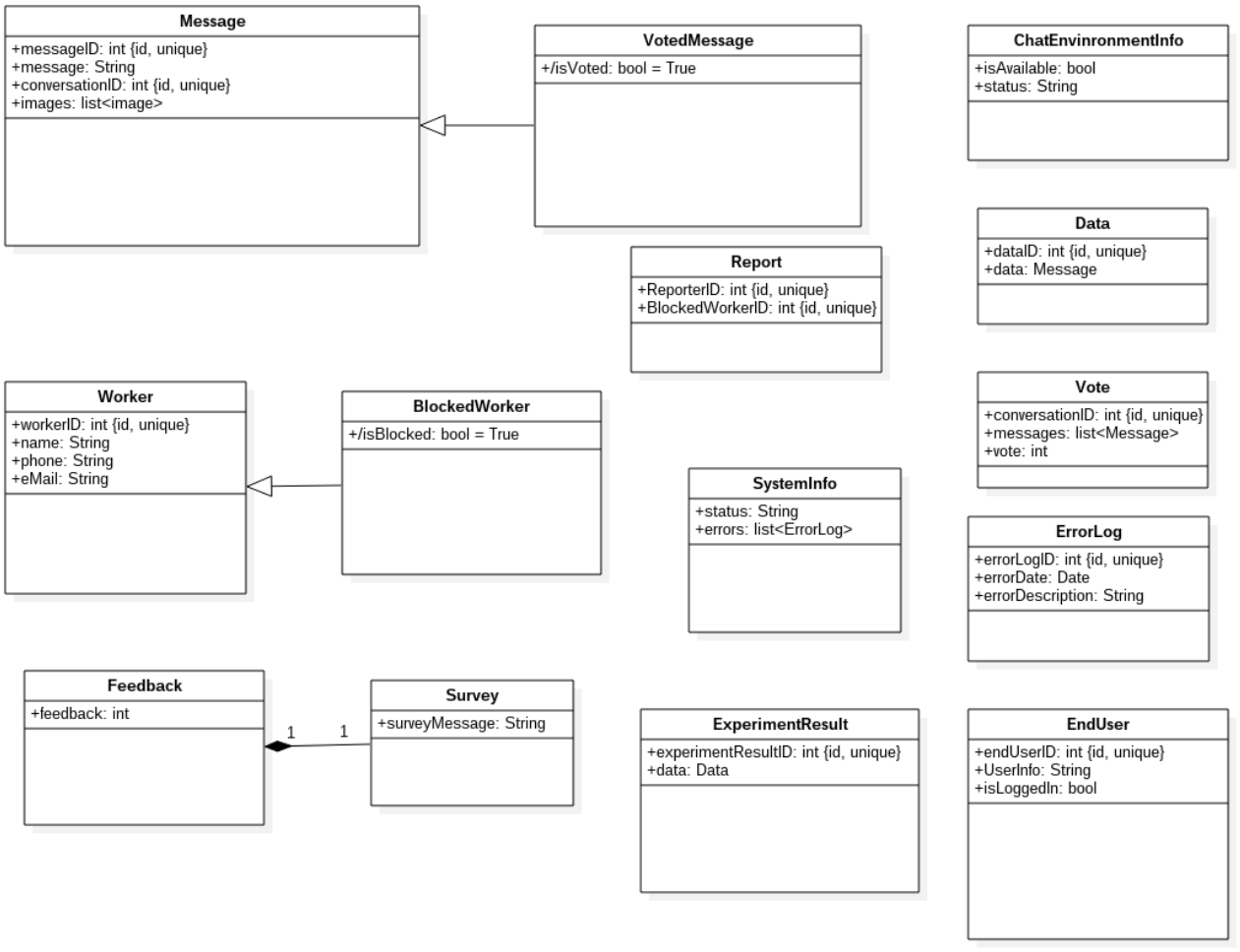

Figure 6: Database Class Diagram

Operation	**CRUD Operations**
ViewErrorLogs	CREATE - READ – ErrorLog UPDATE - DELETE -
SendMessage	CREATE - Message READ - Message UPDATE - DELETE -
GiveFeedback	CREATE - Feedback READ - Survey UPDATE - DELETE -
ViewData	CREATE - READ - Data UPDATE - DELETE -
AddResultsofExperiments	CREATE – ExperimentResult READ - UPDATE - ExperimentResult DELETE -
BlockMaliciousWorker	CREATE - BlockedWorker READ - UPDATE - DELETE -
FixBugs	CREATE - ErrorLog READ – SystemInfo

	UPDATE - SystemInfo DELETE - ErrorLog
SeeFeedbacks	CREATE - READ - Feedback UPDATE - DELETE -
ProvideChatEnvinronment	CREATE - READ - ChatEnvinronmentInfo UPDATE – ChatEnvinronmentInfo DELETE -
Register	CREATE - EndUserID READ - UserInfo UPDATE - DELETE -
SignIn	CREATE - LoggedInUser READ - EndUserID UPDATE - EndUser DELETE -
ReportMaliciousWorker	CREATE - Report READ - WorkerID UPDATE - Report DELETE -
SendData	CREATE - Data READ - Message UPDATE - Data DELETE -
StoreData	CREATE - READ - Data UPDATE - DataList DELETE -
VotetoResponds	CREATE - Vote READ - Message UPDATE - VotedMessage DELETE -

Table 19: CRUD Operations

Design rationale:

- MySQL should be chosen as a relational DBMS in order to make sure that all integrity constraints will be satisfied.
- Since all data is kept in the own database of Chorus system, there will be no confusion in terms of data.

4.4. Interface View

4.4.1. Internal Interfaces

The Interface Between the Database Server and the Internal Communication Handler: The Website Manager queries the Database Server for every operation in the system which is related to the Database Server. The queries from Internal Communication Handler to Database Server is handled by SQL. The queries which came from Internal Communication Handler to Database Server is handled by MySQL DBMS. If the operation is successful, then the result of the query is sent to the Internal Communication Handler. Otherwise, if there is an error in SQL, the error message is sent to the Internal Communication Handler.

Design Rationale:

- Since the database is important for the Chorus system, all data kept in and researchers and crowd workers are using it, there is a need for an interface for connections between Internal Communication Handler and Database Server.
- The operation which Internal Communication Handler tries to do generate the SQL query which contains the data which will be sent to the Database Server. Since the DBMS is MySQL, SQL query is selected to handled the queries.

The Interface Between the Authorization Handler and the Website Manager: Every time a end user try to do some action which is related to Chorus system, user's authorization level is asked to Google Hangouts in order to authorize the end user. That operation is handled by the Authorization Handler. Website Manager uses the Authorization Handler's interface to send the Cookie which came from the end user and the ID of the operation which the end user tries to do. The authorization information, whether the end user's authorization level valid or not , is sent back to the Website Manager.

Design Rationale:

- Since the Chorus system does not keep any information about the end users, the system has to ask to Google Hangouts for any information about the end user. Since for every operation that must be done, there is a need for a separate component, authorization handler.

- Only the Cookie, which is the initially given to end user by Google Hangouts, is sent from the Website Manager to Google Hangouts.

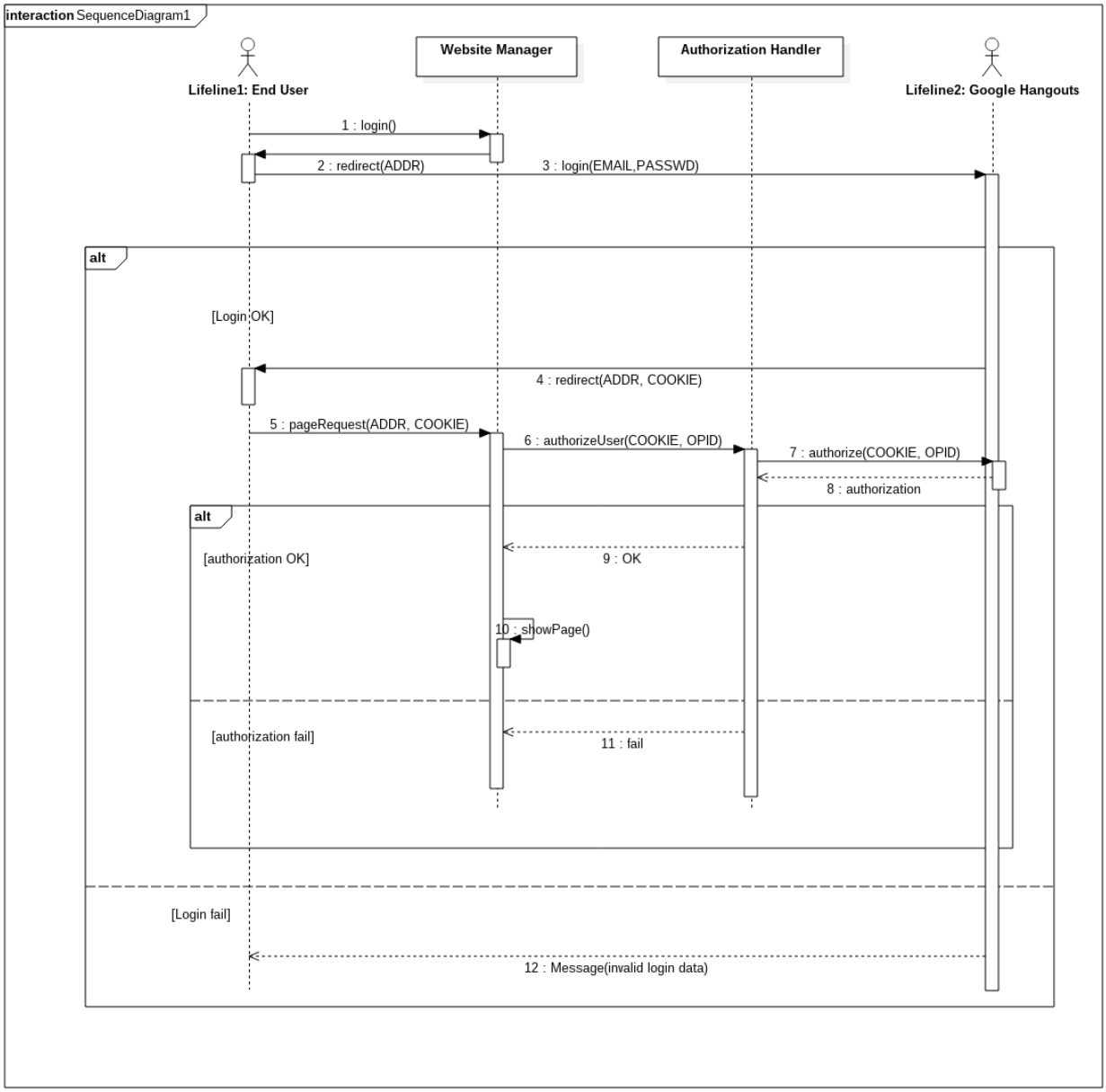

Figure 7: Sign in Sequence Diagram

The Interface Between the External Communication Handler and Website Manager: Every time a message is received from the end user to Chorus system, that message is sent to the Website Manager. Moreover, every time the crowd workers

choose the best respond to the received message, is sent from Website Manager to External Communication Handler.

Design Rationale:

- The only data that is sent or received is message from the end user or from the crowd workers. The system is separate from the Website manager because of simplicity of the system.

The Interface Between the Internal Communication Handler and Website Manager: Every time a message received by the Website Manager, the Website Manager sends the message to Internal Communication Handler. Moreover, the Internal Communication Handler provide a chat interface for the crowd workers. By this way, crowd workers can write their responses and vote the best response. When the best respond is chosen, it is sent to the Website Manager.

Design Rationale:

- There is a need for an interface since the system sends the best respond to end user.

4.4.2. External Interfaces

4.4.2.1. User Interfaces

There are various types of user interfaces and the way they are different from each other depends on their functionalities served to the users. They consist of the user interface for end users who login to the system via Google Hangouts and crowd workers who are provided extra functionalities than end users. They also includes the user interface for researchers and admins who can access the Data Access and Analysis Interface and the Admin Panel respectively. In order for users to access either of these two interfaces, first they need to sign in with their staff IDs. There are more detailed information about these user interfaces in the following pages.

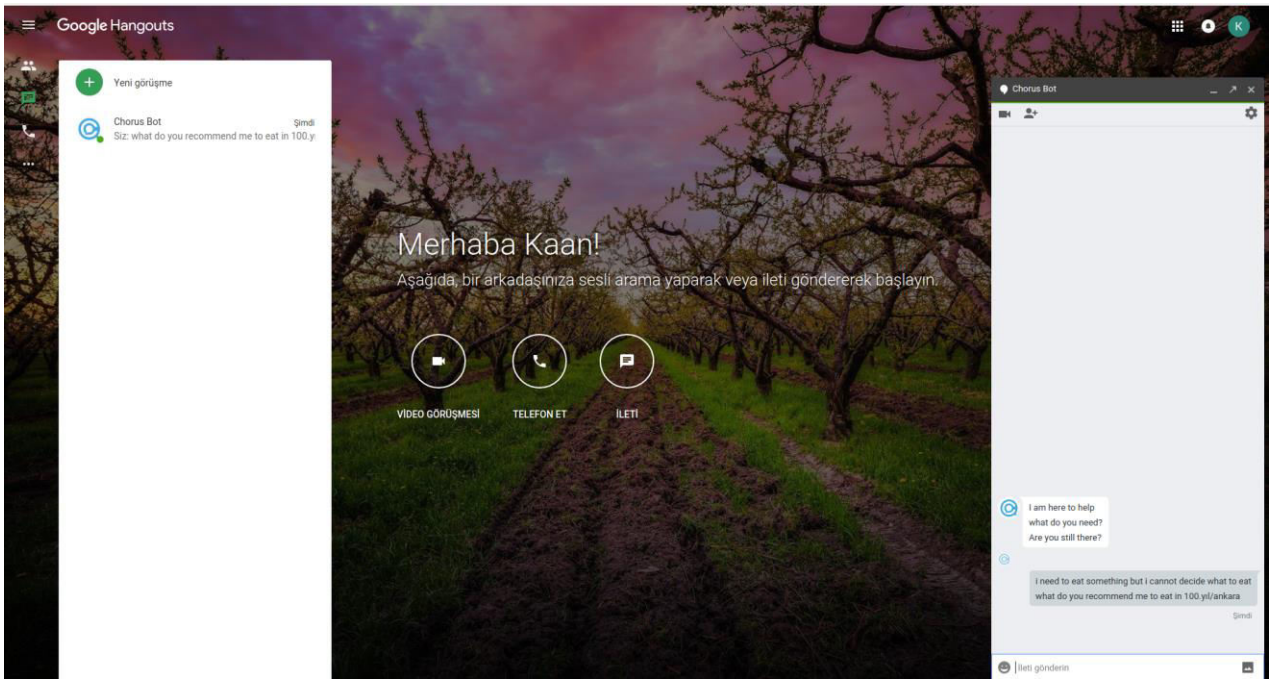

Figure 8: Interface of end users

End user Interface: This interface is provided by Google Hangouts which is responsible for the communication between end users and crowd workers. Therefore, it's no different chatting with Chorus than chatting with somebody else and end users are able to use all functionalities of Google Hangouts. Before end users start chatting with Chorus, they have to either create a new account or simply login with their current account. Afterwards, they can send messages and at the end of the conversations they may grade the conversation by giving feedback. This is implemented in "Register", "Sign in", "Send Message" and "Give Feedback" use cases.

Design Rationale:

- Since Chorus system does not constitute to the end user interface and it is all handled by Google Hangouts, end users just use the Google Hangouts functionalities as always.
- "Sign in" button is at the top right corner of the page and handled by Google since end users have to login to the Google Hangouts not Chorus system .
- There is no visible "Send Message" button, instead end users press enter button to send their messages.

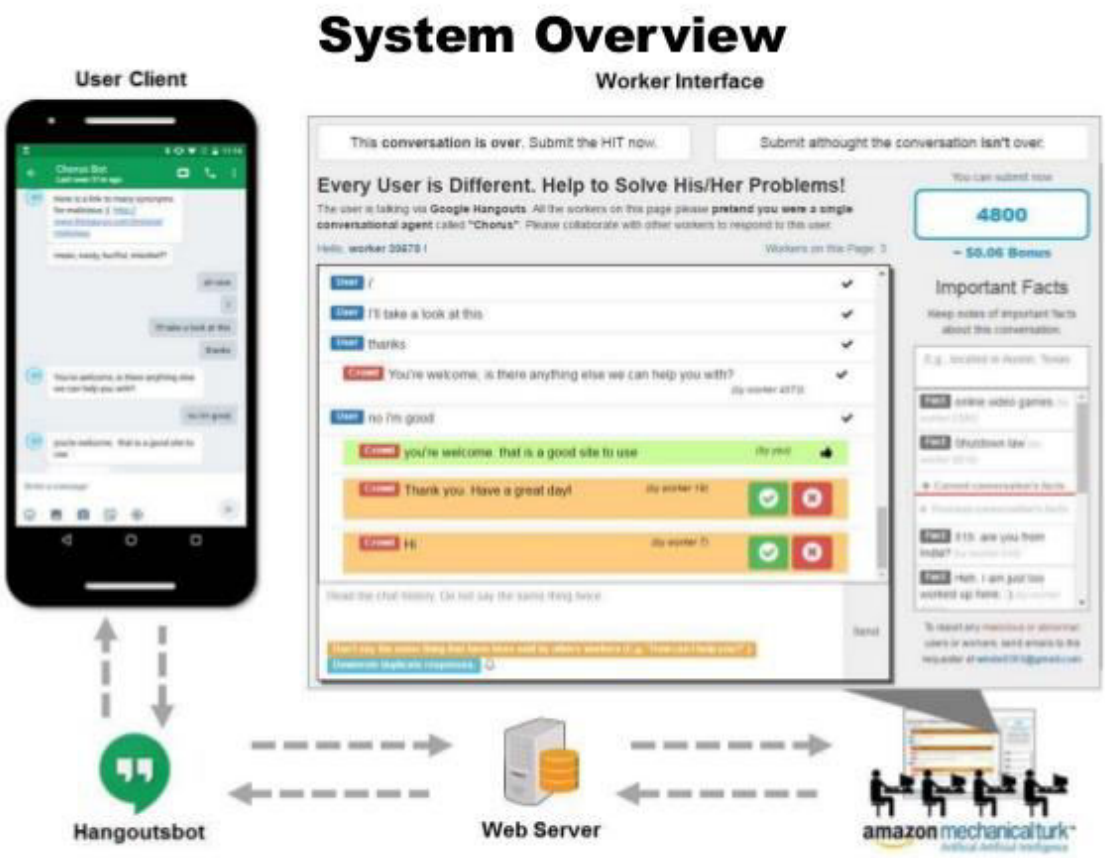

Figure 9: Interface of crowd workers

Crowd Worker Interface: This interface is only available to crowd workers who are Amazon Mechanical Turk employees. In order to access this interface, however, first they need to sign in with their staff ID. Through this interface, crowd workers can send messages to end users, report malicious workers to the admins, send data to memory, view data and vote to other workers' response. These are handled by "Send message", "Report malicious workers", "Send data to memory", "View data" and "Vote to responses" use cases.

Design Rationale:

- Since this interface has more functionalities than the end users', crowd workers' interface is a bit more complicated.
- Chat window comprises the essential part of the interface as crowd workers mostly deal with end users by answering their questions, so it's a large part of it.

- There is a important facts part at the right of the page for crowd workers to help them give their responses.
- There are two tabs at the top of the page for crowd workers to submit the HIT (Human Intelligence Task) which is is a question that needs an answer, and represents a single, self-contained task that a Worker can work on, submit an answer, and collect a reward for completing. The first tab is for the ended conversation while the second tab is for the ones that are still in progress.

Researcher Panel Interface: This interface is only accessible to users who signed in as a researcher with their staff ID. Through this interface, researchers are able to access the data that is stored in the memory. In this way, they can examine the data in order to improve the Chorus system. They can also add their own data which they obtain from the conversations. This functionalities are handled by “View data” and “Add results of experiments” use cases.

Design Rationale:

- This interface has very few functionalities as researchers only need to access the data in the memory and add what they obtained from conversations. The design of the system , therefore, based on these two operations.
- Since this interface is not accessible to end users and only available to researchers, there is not any aesthetic concerns.
- A graphical user interface is utilized while designing the interface instead of a text based user interface. In this way, researchers are able to find the data they are looking for easily.

Admin Panel Interface: A user has to sign in as an admin to the system in order to access admin panel interface. Through this interface, admins can see the error logs, a list of reported workers to block and feedbacks given by end users. They are also able to make changes on the system to fix the parts not running correctly. These functionalities are implemented by “View error logs”, “Block malicious workers”, “See the feedbacks” and “Fix bugs” use cases.

Design Rationale:

- This interface is only accessible by admins, so visual concerns aren’t of importance as in the case of researcher panel interface.

- Database related operations are also done graphically like the researcher panel interface. By doing so, the system is made easier to use for admins. Moreover, a direct access to database is prevented in order to avoid possible data corruption.

4.4.2.2. System Interfaces

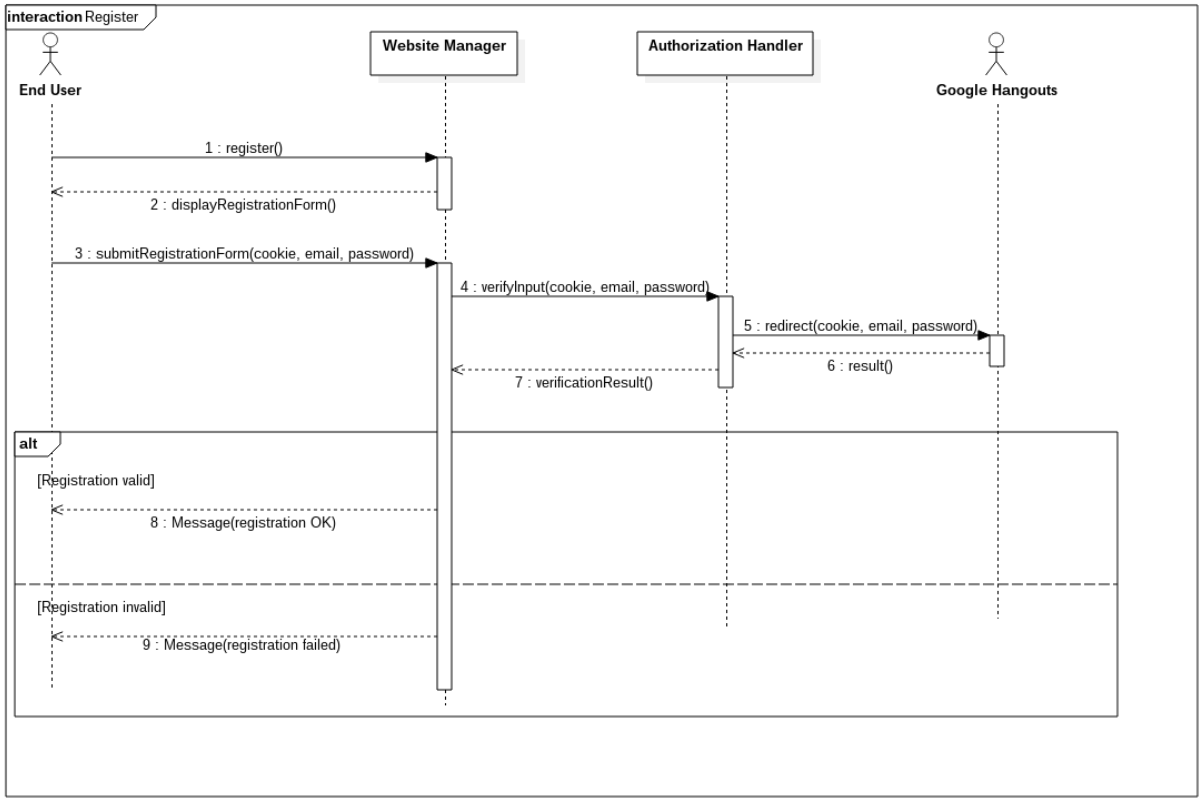

Figure 10: Register sequence diagram showing the interfaces between Authorization Handler and the external system Google Hangouts

The Interface Between the Authorization Handler and Google Hangouts: After an end user requests to create an account, this request is carried from the Website Manager to the Authorization Handler. The Authorization Handler, then, contacts with Google Hangouts through this interface. The Authorization Handler sends the information received from the Website Manager with a cookie to the Google Hangouts. Google Hangouts sends back the result back for the Authorization Handler to inform back the Website Manager. This is implemented in “Register” use case.

Design Rationale:

- Chorus system does not hold end users’ data. For this reason, Google Hangouts is sent the necessary information of end users through this interface in order end users to create an account to start a conversation with Chorus.

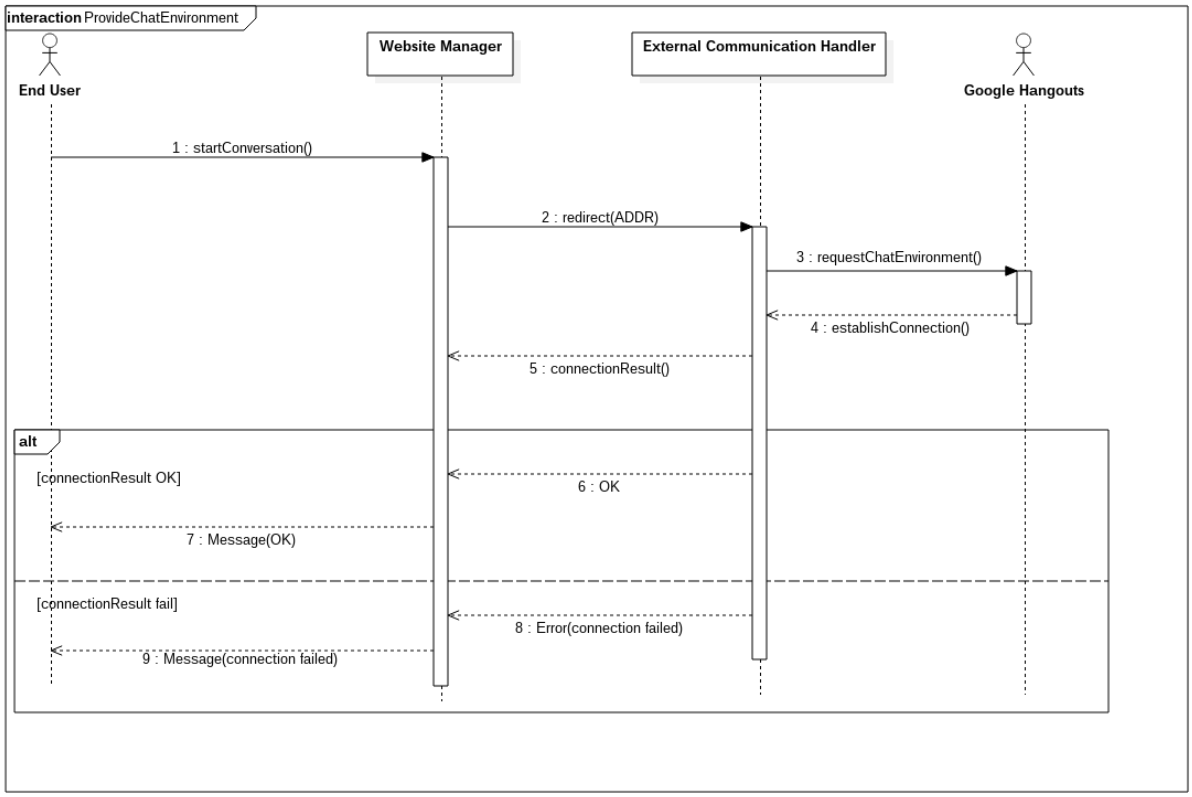

Figure 11: Provide Chat Environment sequence diagram showing the interfaces between External Communication Handler and the external system Google Hangouts

The Interface Between the External Communication Handler and Google Hangouts: The External Communication Handler component of the Chorus system communicates with Google Hangouts through this interface**.** When an end user wants to start a conversation box, this request is first taken by the Website Manager and then carried to the External Communication Handler. Afterwards, External Communication Handler contacts with Google Hangouts to provide a chat environment between end users and crowd workers. If no error occurs during the process, an available connection is established between them. Otherwise, the External Communication Handler informs the admins accordingly. The related use case is “Provide a chat environment”.

Design Rationale:

- Since the Chorus system does not have a chat environment to enable end users and crowd workers to communicate with each other, a conversation box is provided through this interface.