Team 8

Design specification

2017312982 류재현

2016312445 허재승

2016314568 전형준

2016310886 김민준

2020002318 Sesar Hersisson

[1. Preface 7](#_Toc41255670)

[a. Objective 7](#_Toc41255671)

[b. Readership 7](#_Toc41255672)

[c. Document Structure 7](#_Toc41255673)

[i. Preface 7](#_Toc41255674)

[ii. Introduction 8](#_Toc41255675)

[iii. System Architecture 8](#_Toc41255676)

[iv. User Management System 8](#_Toc41255677)

[v. Video Communication System 9](#_Toc41255678)

[vi. Audio Conversion System 9](#_Toc41255679)

[vii. Protocol Design 9](#_Toc41255680)

[viii. Database Design 10](#_Toc41255681)

[ix. Testing Plan 10](#_Toc41255682)

[x. Development Environment 10](#_Toc41255683)

[xi. Development Plan 11](#_Toc41255684)

[d. Version of Document 11](#_Toc41255685)

[2. Introduction 11](#_Toc41255686)

[a. Objective 11](#_Toc41255687)

[b. Applied Diagram 12](#_Toc41255688)

[i. Sequence Diagram 12](#_Toc41255689)

[ii. Usecase Diagram 13](#_Toc41255690)

[iii. ER Diagram 13](#_Toc41255691)

[c. Applied Tool 14](#_Toc41255692)

[i. Creately 14](#_Toc41255693)

[d. Project Scope 15](#_Toc41255694)

[3. System Architecture 15](#_Toc41255695)

[a. Objective 15](#_Toc41255696)

[b. System Organization 16](#_Toc41255697)

[i. User Management System 18](#_Toc41255698)

[ii. Video Communication System 20](#_Toc41255700)

[iii. Audio Conversion System 22](#_Toc41255701)

[4. User Management System 24](#_Toc41255702)

[a. Objective 24](#_Toc41255703)

[b. Class Diagram 24](#_Toc41255704)

[i. Login 24](#_Toc41255705)

[1. Attributes 24](#_Toc41255706)

[2. Methods 25](#_Toc41255707)

[ii. Register 25](#_Toc41255708)

[1. Methods 25](#_Toc41255709)

[c. Sequence Diagram 26](#_Toc41255710)

[d. State Diagram 26](#_Toc41255711)

[5. Video Communication System 27](#_Toc41255712)

[a. Objective 27](#_Toc41255713)

[b. Class Diagram 27](#_Toc41255714)

[i. Video Streaming 27](#_Toc41255715)

[1. Attributes 27](#_Toc41255716)

[2. Methods 27](#_Toc41255717)

[ii. User 28](#_Toc41255718)

[1. Attributes 28](#_Toc41255719)

[2. Methods 28](#_Toc41255720)

[iii. Video Saving 28](#_Toc41255721)

[1. Attributes 28](#_Toc41255722)

[c. Sequence Diagram 29](#_Toc41255723)

[d. State Diagram 30](#_Toc41255724)

[6. Audio Conversion System 31](#_Toc41255725)

[a. Objective 31](#_Toc41255726)

[b. Class Diagram 32](#_Toc41255727)

[i. Video stream 32](#_Toc41255728)

[1. Attributes 32](#_Toc41255729)

[2. Methods 32](#_Toc41255730)

[ii. Audio stream 32](#_Toc41255731)

[1. Attributes 32](#_Toc41255732)

[iii. Text 32](#_Toc41255733)

[1. Attributes 33](#_Toc41255734)

[2. Methods 33](#_Toc41255735)

[iv. User 33](#_Toc41255736)

[1. Attributes 33](#_Toc41255737)

[2. Methods 33](#_Toc41255738)

[c. Sequence Diagram 34](#_Toc41255739)

[d. State Diagram 35](#_Toc41255740)

[7. Protocol Design 36](#_Toc41255741)

[a. Objective 36](#_Toc41255742)

[b. JSON format 37](#_Toc41255743)

[c. Protocol Description 38](#_Toc41255744)

[8. Database Design 39](#_Toc41255745)

[a. Objective 39](#_Toc41255746)

[b. ER diagram 39](#_Toc41255747)

[i. User 40](#_Toc41255748)

[ii. Storage 40](#_Toc41255749)

[iii. Video 41](#_Toc41255750)

[iv. Text 42](#_Toc41255751)

[v. Streaming 42](#_Toc41255752)

[c. Relational Schema 43](#_Toc41255753)

[9. Testing Plan 44](#_Toc41255754)

[a. Objective 44](#_Toc41255755)

[b. Testing Policy 44](#_Toc41255756)

[c. Test Cases 45](#_Toc41255757)

[10. Development Environment 46](#_Toc41255758)

[a. Objective 46](#_Toc41255759)

[b. Technologies 46](#_Toc41255760)

[i. Pytorch 46](#_Toc41255761)

[ii. NodeJS 47](#_Toc41255762)

[iii. MySQL 47](#_Toc41255763)

[c. Environment 47](#_Toc41255764)

[11. Development Plan 48](#_Toc41255765)

[a. Objective 48](#_Toc41255766)

[b. Approach 48](#_Toc41255767)

[c. Plan 49](#_Toc41255768)

# Preface

## Objective

The expected readership of the document is defined, along with the description of the overall structure of this document. The overall structure is described with a brief explanation of each chapters, and the objectives of each chapter. The version history of the document and the changes made are also summarized.

## Readership

This document is expected to be read by all members participating in development and maintenance of this project, ATlantis, including all software engineers and system developers.

## Document Structure

### Preface

The readership of this document is defined and the overall structure of the document with brief explanation of each section is described.

### Introduction

The diagrams and tools applied and used in designing the system are explained, together with the scope of this project.

### System Architecture

An overview of the whole system is described. The overall structure and the relationships between sub-systems are described by using a block diagram, and each will be further specified with a class diagram, a sequence diagram, or a state diagram where needed in further chapters.

### User Management System

Interoperating with the database, user management system manages the overall functionalities of managing the users of the system, such as login, user registration. Diagrams such as class diagram, sequence diagram, and state diagram are used to visualize the structure of the system and its interactions with the database.

### Video Communication System

This system enables video streaming and communication among users. Users can stream and share a video to the people whom they are interacting with, and record the communication for other purposes at the same time. The structure of the corresponding system is to be further explained via class diagram, sequence diagram and state diagram.

### Audio Conversion System

This system serves the function of converting an audio or a video file to a text form file. It can either be a form of a saved file, or it can also be a real-time video communication on-going to convert into a text format file. The structure of the corresponding system is to be further explained via class diagram, sequence diagram and state diagram.

### 

### Protocol Design

The protocol to be used for communication and interaction between sub-systems is defined. The protocol must be followed so that communications between sub-systems are done smoothly. The attributes are to be explained with its format, meanings and its use.

### Database Design

The database is designed to serve essential functions of the system, based on the requirements specification. The ER diagram of the database is described along with a relational schema.

### Testing Plan

In order to make sure the system operates as intended, as described in the requirements specifications, and to discover any flaws that exist throughout the system, the developed system is to be tested. Testing is planned in advance to developing, when designing the system. Testing policy and test cases are described here. In testing policy, the approach and the resources to be tested are explained, and test cases show how the system should respond in each case.

### 

### Development Environment

The environment of which the development is based on is defined, such as the programming language to be used, or the IDE (Integrated Development Environment) used.

### Development Plan

Overall plan and schedule on development is described.

## Version of Document

Version 1.0 (2020.05.24): First version of the entire document completed.

# Introduction

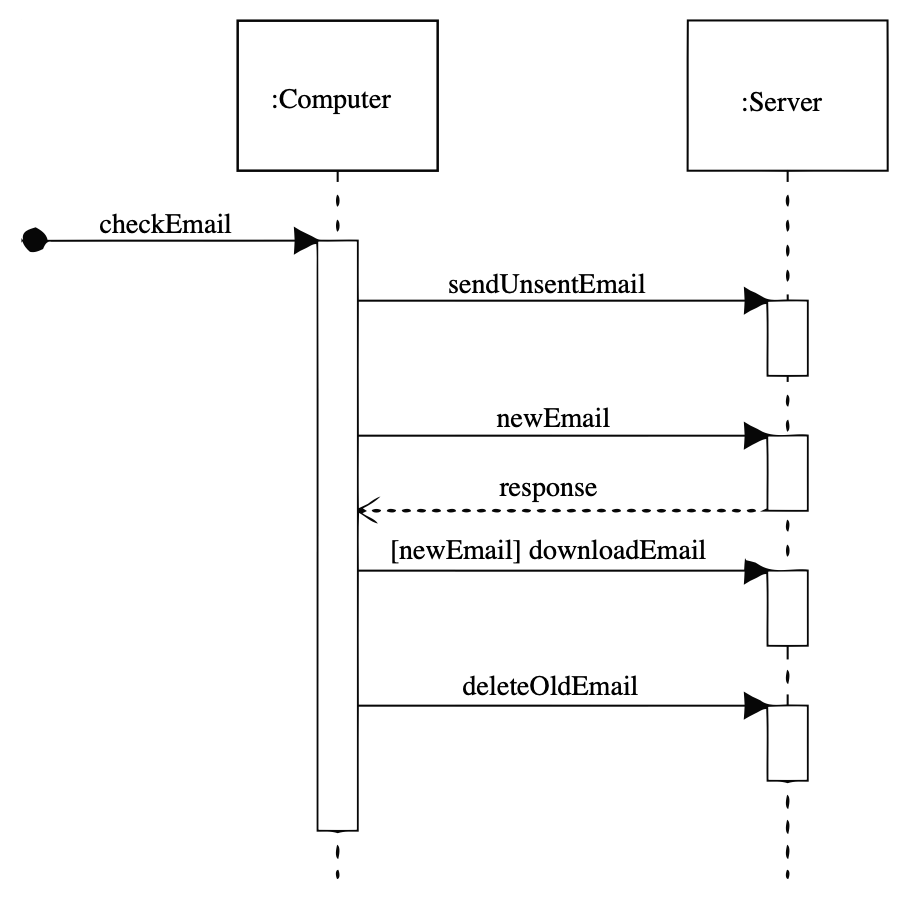
## Objective

The diagrams and tools applied and used in designing the system are explained, together with the scope of this project.

## Applied Diagram

In this document, we use below diagrams.

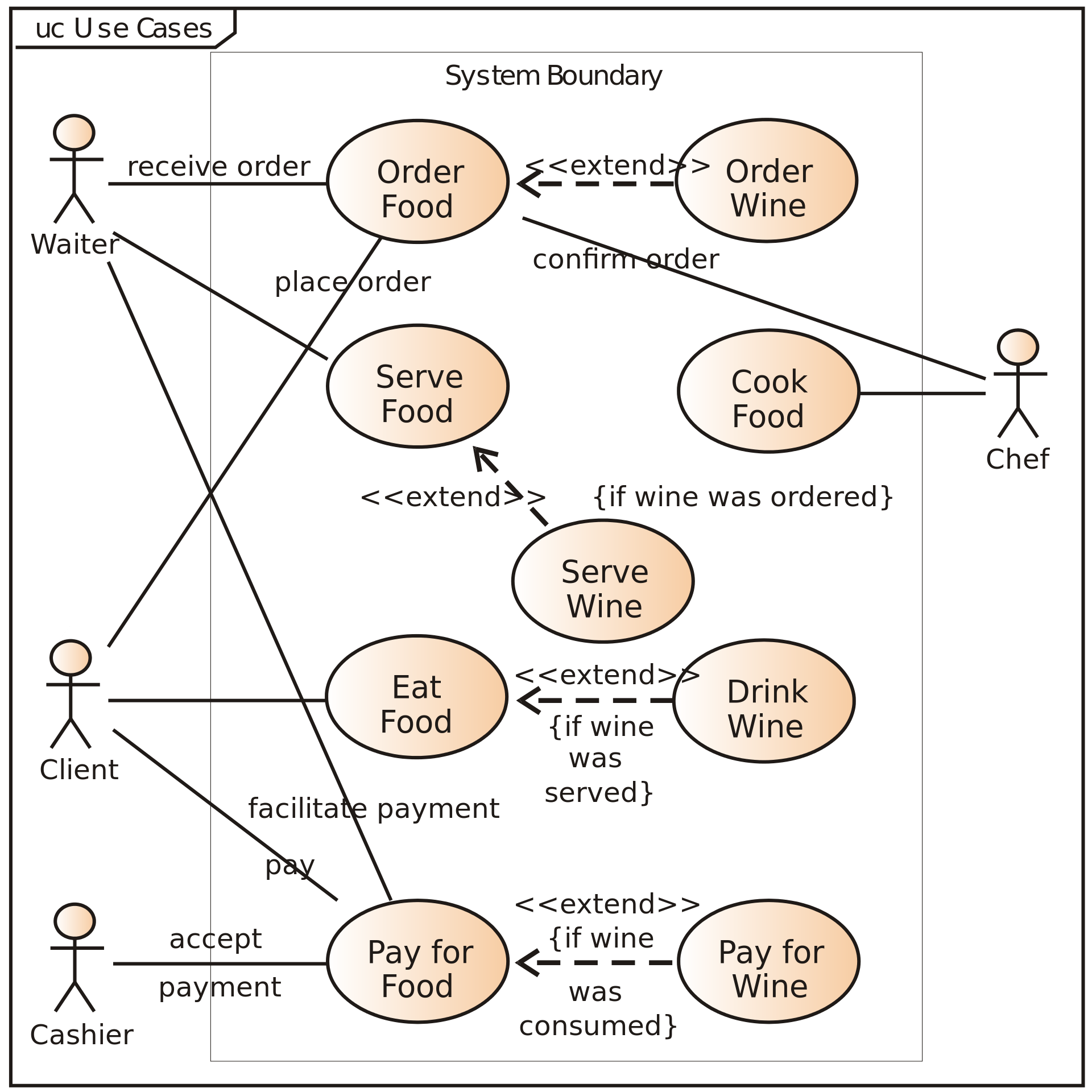
### Sequence Diagram



**Figure 1. Example of Sequence Diagram**

We use sequence diagrams for letting readers know how the streaming systems work, how to access databases and store and query data, and system architecture.

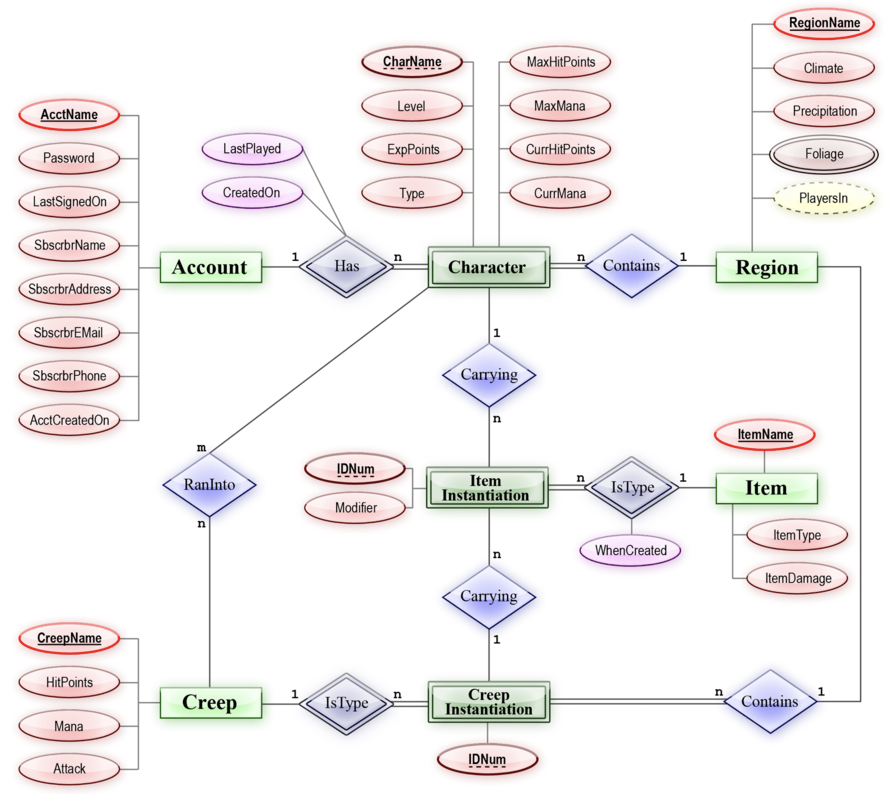
### Usecase Diagram



**Figure 2. Example of User Diagram**

For showing the interaction between the system and the users, we use usecase diagram in user management system section.

### ER Diagram



**Figure 3. Example of ER Diagram**

For database design, and since we use a relational database management system, we use ER diagrams.

## Applied Tool

### Creately

Creately is an online diagram creation tool which we used in making the diagrams.

**Figure 4. Creatley logo**

## Project Scope

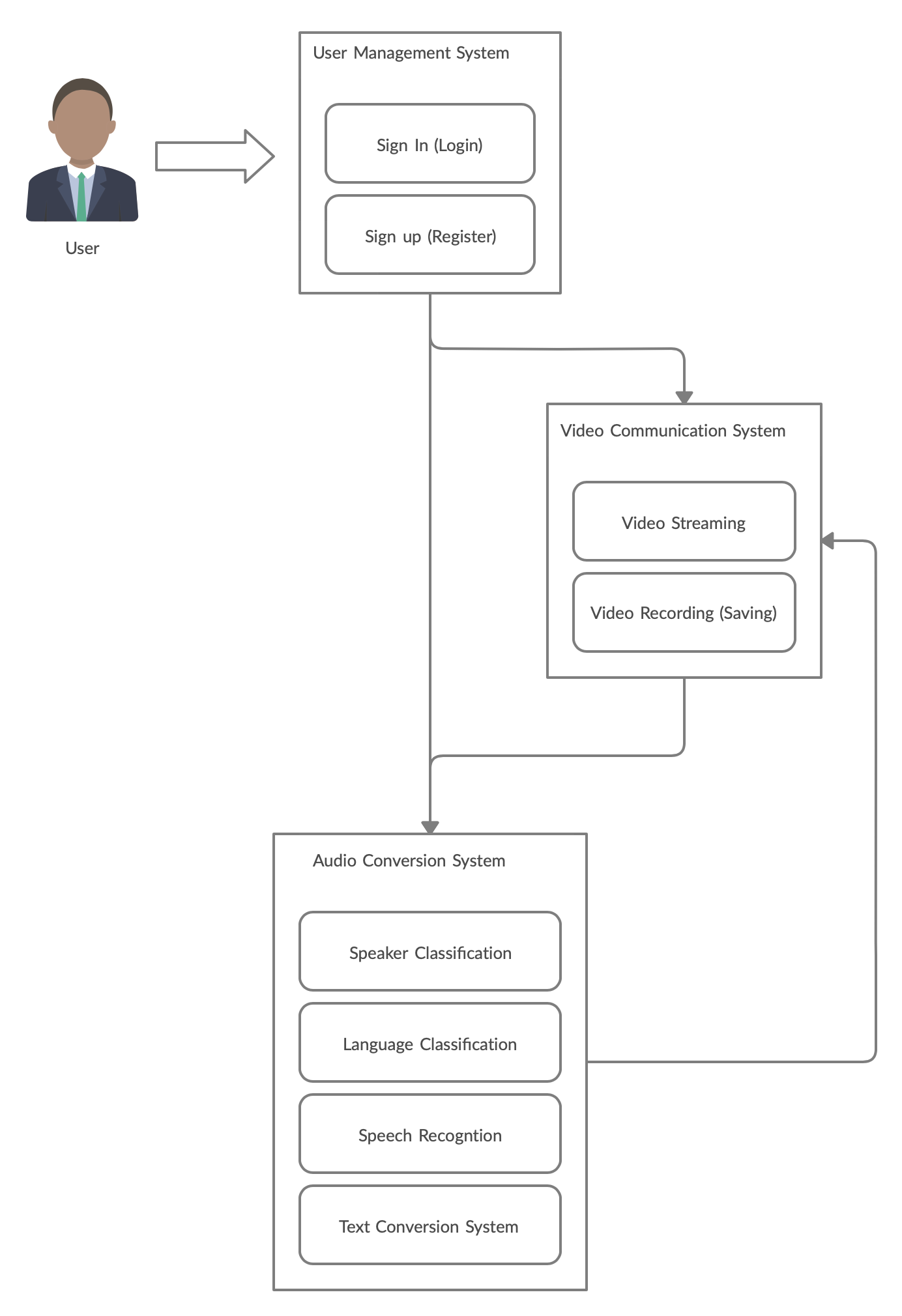
Our project is based on web development, so our main goal is to develop a web server that supports live streaming. Also, we need storage and DB for storing our text and video formats which in case clients want them to be recorded. On top of this, we need a processing engine, in our case, natural language processing. This is the machine learning area.

# System Architecture

## Objective

An overview of the whole system is described. The overall structure and the relationships between subsystems are described by using a block diagram, and each will be further specified with a class diagram, a sequence diagram, or a state diagram where needed in further chapters.

## System Organization



**Figure 5. System organization of ATlantis**

ATlantis is mainly consisted of three sub-systems, the user management system, video communication system, and the audio conversion system.

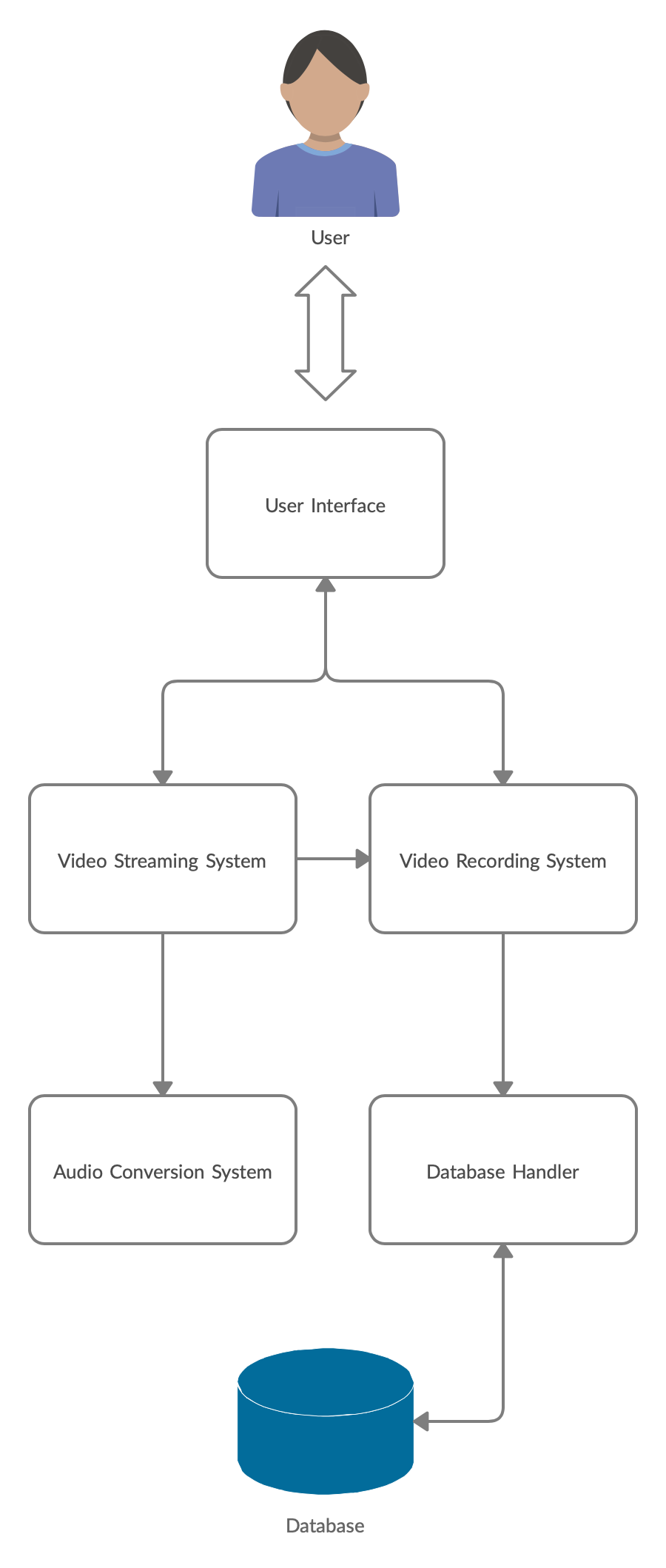
### User Management System

### 

**Figure 6. User Management System of ATlantis**

User management system is the very first system the user interacts with. A new user is lead to sign up and register for the service, with a pair of a new ID and a password. The system interacts with the database to check if there are any IDs with duplicates, and if there are any, it rejects to save it, and requires for a different ID. In case of a user already registered, they may simply enter their ID and corresponding password to sign in, and can take advantage of the system's service, such as video communication or audio conversion by using the video communication system or the audio conversion system. In this process, the system interacts with the database to figure out if there is an ID of the ID that the user has input, and the password the user entered matches the one that corresponds to the ID. If either the ID does not exist, or the password does not match, login is rejected, and notifies the user to try it again.

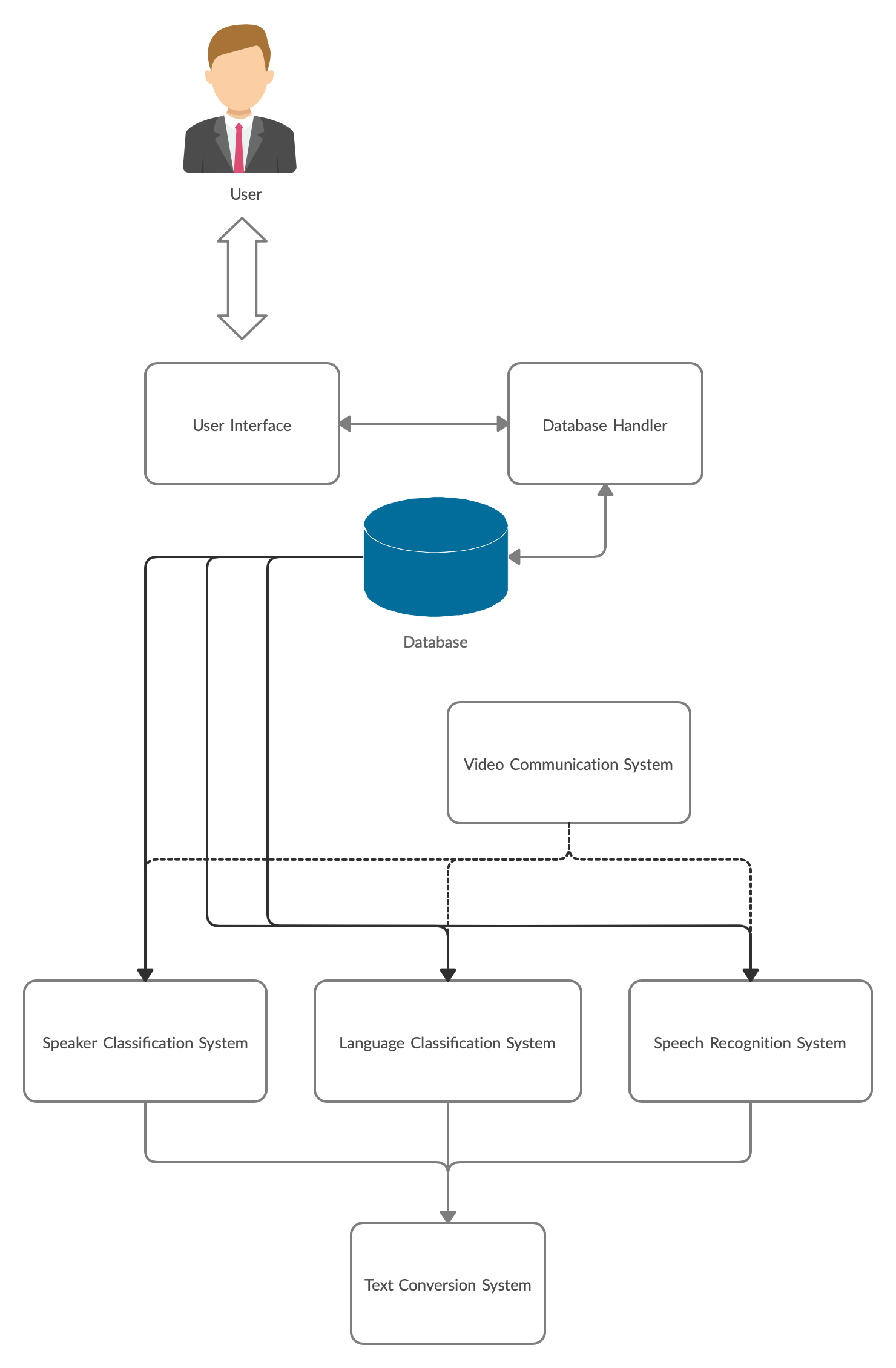
### Video Communication System



**Figure 7. Video Communication System of ATlantis**

Video communication system is a system to stream videos and have video communications online. Users can show their videos to other participants, and talk to each other face to face. A video of the entire meeting may be recorded if a user wishes to, by clicking the record button. The recorded video will be stored in a cloud database, and can be downloaded whenever a user wishes to. If a user wishes to have it scripted, audio conversion system may be activated and have the entire meeting scripted.

### Audio Conversion System



**Figure 8. Audio Conversion System of ATlantis**

Audio conversion system convert a video file or an audio file to a text format file. It first classifies the speaker, then the language of the speaker, and the actual speech of the speaker. This system can be used for either a real-time streaming video, or a file of a recorded video or an audio. For a recorded file, just by uploading the file to the system, the system will run a machine-learned speech recognition algorithm to identify the speech. For a streaming video, parts of the video streamed will be forwarded through the speech recognition algorithm sequentially.

# User Management System

## Objective

A system manages overall functionalities about users such as login, registration, checking Video or Text files with DB. It will be visualized by class, sequence, state diagrams

## Class Diagram

### Login

#### Attributes

* + - * 1. user\_name : Indicates user’s name
        2. user\_pw : Indicates user’s password

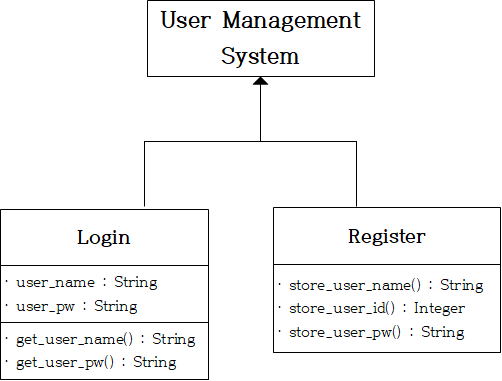
#### Methods

* + - * 1. get\_user\_name() : search user’s name in DB
        2. get\_user\_pw() : search user’s pw in DB

### Register

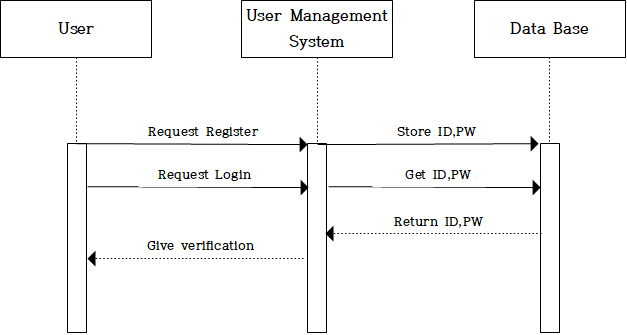
#### Methods

* + - * 1. store\_user\_name() : store new user’s name in DB
        2. store\_user\_id() : store new user’s id in DB
        3. store\_user\_pw() : store new user’s pw in DB



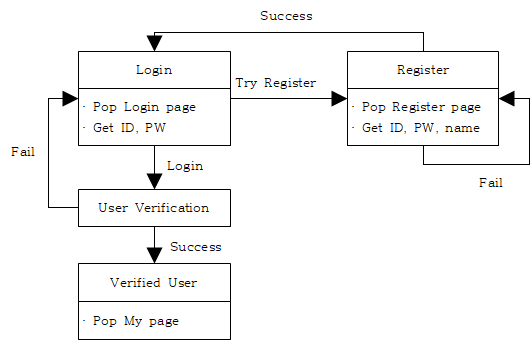
**Figure 9. Class Diagram - User Management System**

## Sequence Diagram



**Figure 10. Sequence Diagram - User Management System**

## State Diagram



**Figure 11. State Diagram - User Management System**

# Video Communication System

## Objective

A system that is the basic framework of ATlantis, which is responsible for conversations between users. The Class Diagram, Sequence Diagram and State Diagram of this system will be described.

## Class Diagram

### Video Streaming

#### Attributes

* + - * 1. host: Indicate the host of streaming.
        2. participating\_user: Indicate the users who participated.

#### Methods

* + - * 1. stop\_streaming: Stop streaming.
        2. handover\_host: hand over host to another person

### User

#### Attributes

* + - * 1. user\_id: Indicate the user's unique ID.
        2. user\_name: Indicate the user’s name.

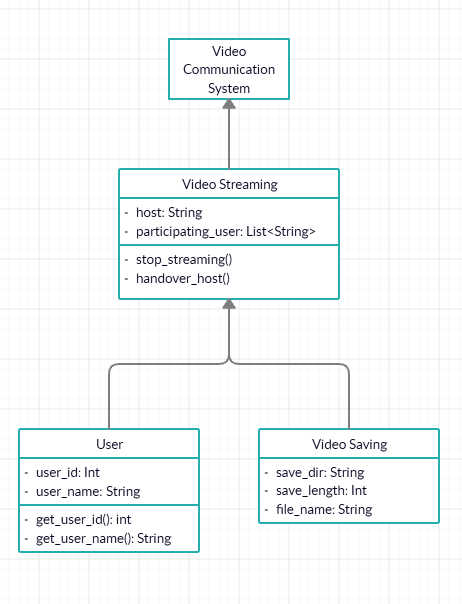
#### Methods

* + - * 1. get\_user\_id: Look up the user's ID.
        2. get\_user\_name: Look up the user's name.

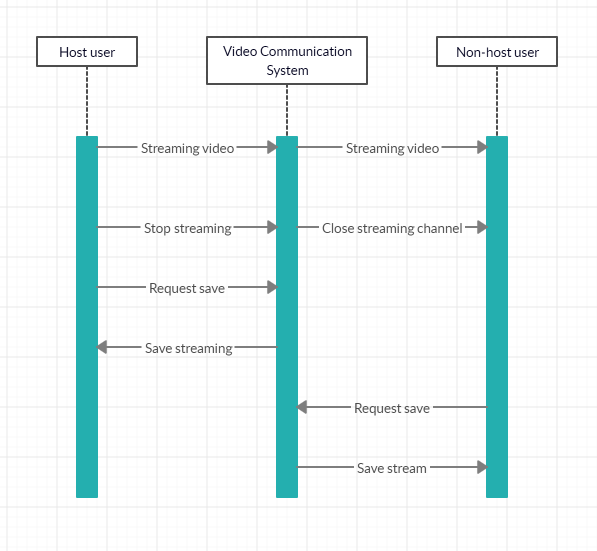
### Video Saving

#### Attributes

* + - * 1. save\_dir: Indicate the location to store.
        2. save\_length: Indicate the time to save.
        3. file\_name: Indicate the name of the file to save.

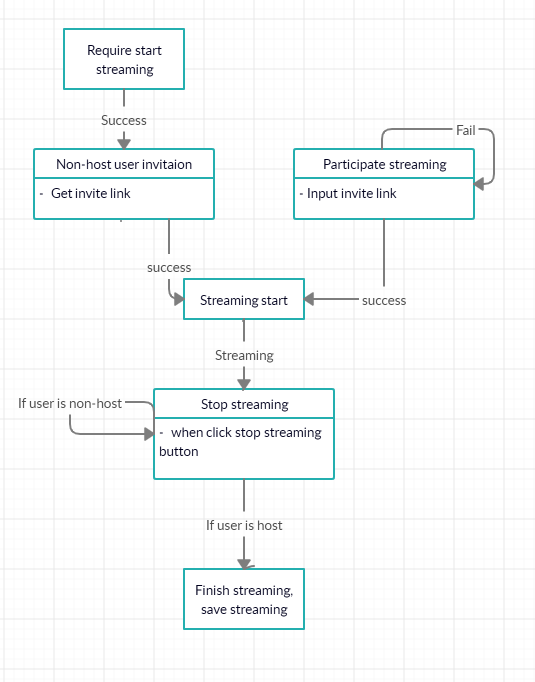
**Figure 12. Class Diagram - Video Communication System**

## Sequence Diagram



**Figure 13. Sequence Diagram - Video Communication System**

## State Diagram



**Figure 14. State Diagram - Video Communication System**

# Audio Conversion System

## Objective

A system that serves as a core function of ATlantis, which plays a role in converting Voice to Text. The Class Diagram, Sequence Diagram and State Diagram of this system will be described.

## Class Diagram

### Video stream

#### Attributes

* + - * 1. video\_file\_name: Indicate saved video file name.
        2. video\_length: Indicate video length.

#### Methods

* + - * 1. seperate\_video\_stream: Detach the video stream into an audio stream.

### Audio stream

#### Attributes

* + - * 1. audio\_file\_name: Indicate saved audio file name.
        2. audio\_length: Indicate audio file length

### Text

#### Attributes

* + - * 1. text\_content: Indicate each text content.
        2. text\_user: Indicate users who send a text.

#### Methods

* + - * 1. get\_text\_user: Get user name who send a text.

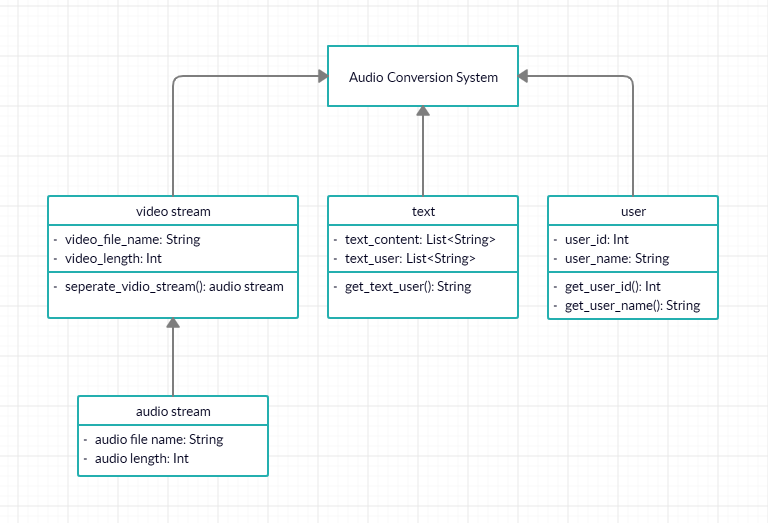
### User

#### Attributes

* + - * 1. user\_id: Indicates the user's unique ID.
        2. user\_name: Indicates the user’s name.

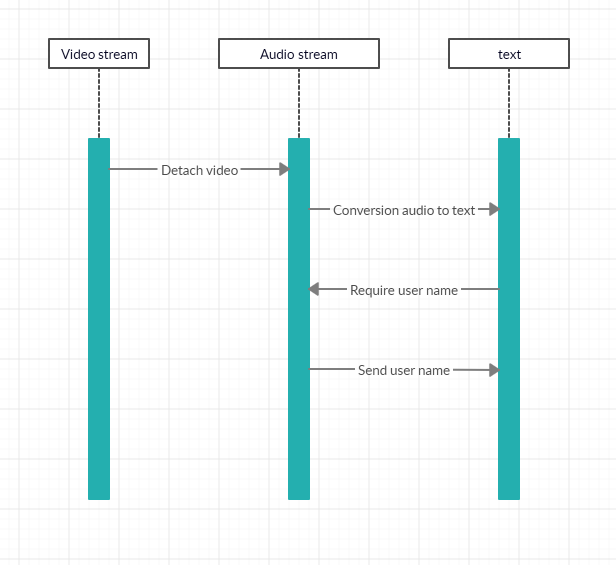
#### Methods

* + - * 1. get\_user\_id: Look up the user's ID.
        2. get\_user\_name: Look up the user's name.



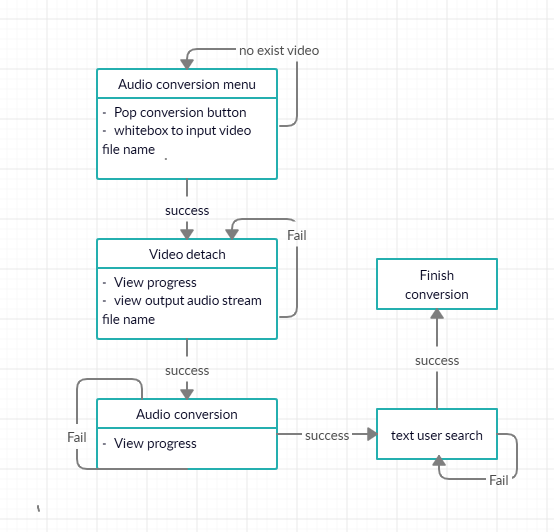
**Figure 15. Class Diagram - Audio Conversion System**

## Sequence Diagram



**Figure 16. Sequence Diagram - Audio Conversion System**

## State Diagram



**Figure 17. State Diagram - Audio Conversion System**

# Protocol Design

## Objective

In this section, we show the protocol we are going to use, JSON format, and how to use this.

## JSON format

|  |
| --- |
| { "account\_id": "id",  "sources":  [ { "src": "name.mp4", "container": "MP4" }, { "type": "application/x-mpegURL", "src": "webserverURL" }],  "id": "id",  "duration": (sec) } |

“account\_id” : set an ID of this account. Our system automatically gives it.

“sources” : it specifies the name of file, type of file(in our case, mp4)

“id” : similar with account\_id but in this case, it is used for distinguishing among the JSON packets.

“duration” : contains the information of this mp4 file’s play time.

## Protocol Description

Since we will develop web, JSON is widely used when sending data from client to server. Remember that our system is slave(client) - master(server) model. Below is the example of the JSON format.

{ } represents the class and nested class declaration is allowed. This format is compatible with JavaScript so it can be interpreted easily on a web-browser.

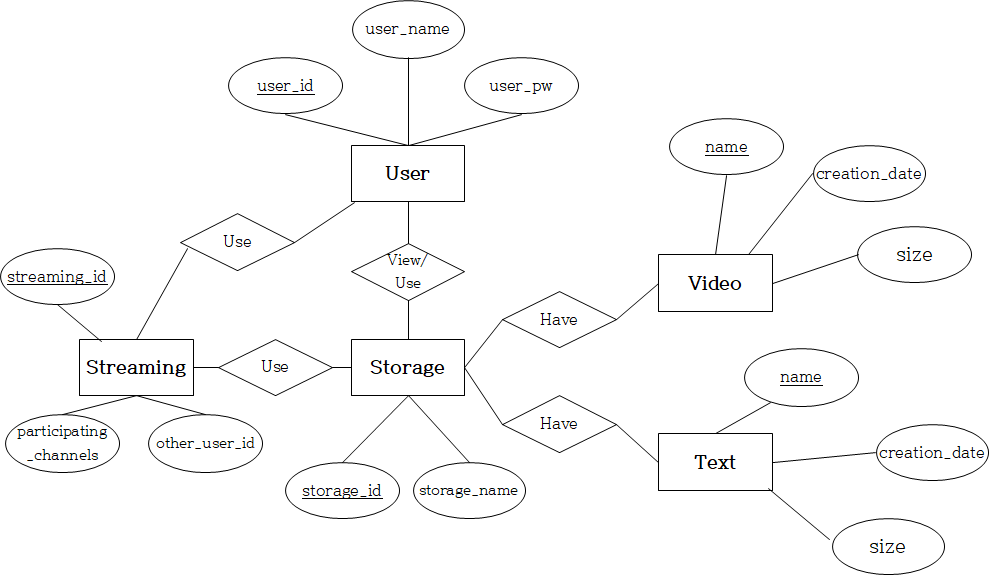
|  |
| --- |
| {  "Users": [  {  "Name": "John",  "Language": [  "English",  "Korean"  ]  },  {  "Name": "Smith",  "Language": [  "Korean",  "Japanese"  ]  }  ] } |

# Database Design

## Objective

This is designing Database for our functions and visualized by ER diagram and Relational Schema

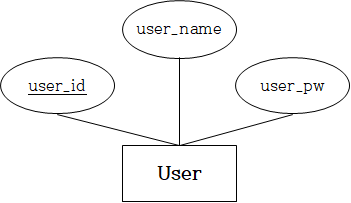
## ER diagram



**Figure 18. ER diagram - Database Design**

## Entities

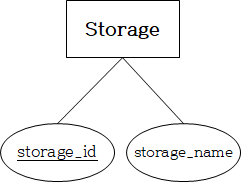
### User



**Figure 19. User Entity - ER diagram**

User Entity expresses user’s information, user\_id attribute is primary key, and other attributes are user\_name, user\_pw

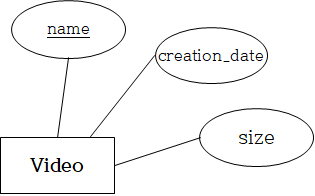
### Storage



**Figure 20. Storage Entity - ER diagram**

Storage Entity expresses the storages which save Video and Text, its primary key is storage\_id and other attribute is storage\_name

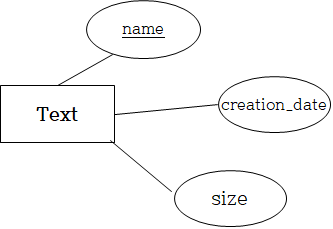
### Video



**Figure 21. Video Entity - ER diagram**

Video Entity expresses information of stored Video, name attribute is primary key and other attributes are creation\_date, size

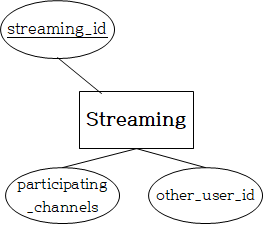
### Text



**Figure 22. Text Entity - ER diagram**

Text Entity expresses information of converted and stored Text, name attribute is primary key and other attributes are creation\_date, size

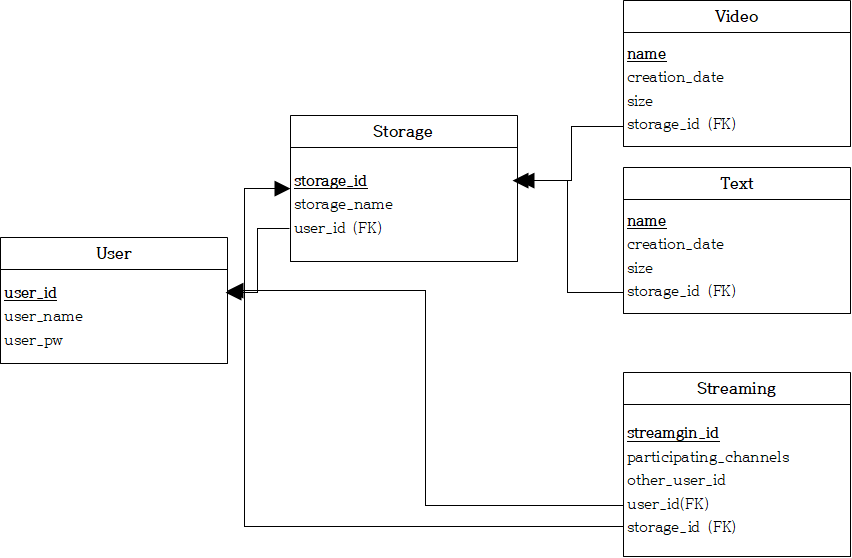
### Streaming



**Figure 23. Streaming Entity - ER diagram**

Streaming Entity stores informations in Streaming, streaming\_id is primary key and other attributes are participation\_channels, other\_user\_id

## Relational Schema



**Figure 24. Relational Schema - Database Design**

# Testing Plan

## Objective

Testing a software is a crucial part in its development process, as it demonstrates that the system can do what it is intended to do and to discover defects before it is deployed. This chapter will expound on how the testing process will transpire.

## Testing Policy

Through testing we wish to find and fix defects during the development process using unit testing where each component is tested as it is integrated. All operations of an object need to be tested and the unit test cases designed accordingly. The test cases should show that the component does what it is supposed to and reveal defects.

We will also conduct user testing where users, such as classmates or friends, will use the application and give feedback through beta testing.

After the system has been completely integrated we can test for performance and reliability with performance testing. The whole system will be tested under its intended load to see how it handles it.

## Test Cases

Unit testing will make sure each component is functioning as intended when it is integrated in the system.

During user tests the main objective is to test the user interface and how the user responds to it. The UI should be intuitive and easy to use.

The streaming service should run without major hiccups or rupture and the application should run smoothly even under traffic.

# Development Environment

## Objective

This section will cover the environment in which the system will be developed and the technologies used.

## Technologies

### Pytorch

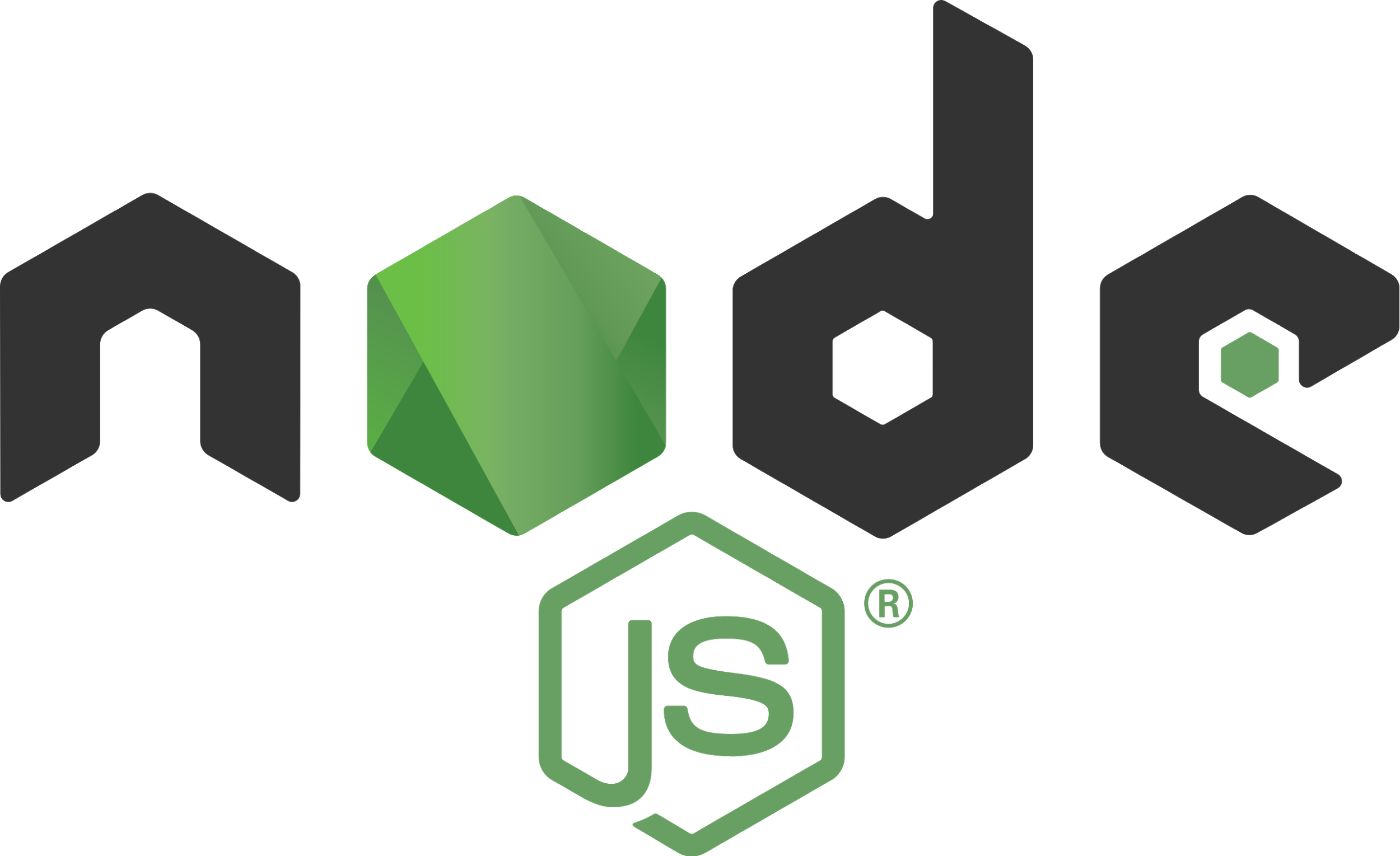
PyTorch is an open source machine learning library based on the Torch library, which is widely used for natural language processing, primarily developed by Facebook's AI Research lab. Pytorch will be used in recognizing keywords and speech recognition.



**Figure 25. PyTorch**

### NodeJS

NodeJS is an open source JavaScript runtime environment, which is widely used in a real-time web based system. Since our system has to support real-time streaming service.



**Figure 26. NodeJS**

### MySQL

MySQL is probably one of the most popular open source relational database management system (RDBMS), and we are going to use MySQL as our DBMS.



**Figure 27. MySQL**

## Environment

Each team member will develop the system from wherever they want and deploy changes through Github. Since all changes are deployed through Github the development environment, that is the IDE, that each member uses is up to their preference.

The application will be written mostly in javascript, aside from the machine learning which is written in python, and each member should use syntaxing and error catching inside their IDE for faster development.

# Development Plan

## Objective

This section will go over the overall development plan, how the system will be developed.

## Approach

We will utilize an agile approach in the development of the system. Team members will work in groups or on their own on a subsystem which will be developed incrementally. Subsystems will be updated frequently through their development process.

## Plan

Development of the system will start on May 25th and will be finished before june 7th. There is no rigid plan as to when each component or feature should be finished as every component except the UI will be developed in parallel. After the main component, the streaming service, is finished it can be integrated in the UI and therefore the UI development can start.