
Bilkent University

Department of Computer Engineering

# Senior Design Project

*Project short-name: YouTalkWeSign*

# Analysis Report

Abdurrezak Efe, Yasin Erdoğdu, Enes Kavak, Cihangir Mercan

Supervisor: Hamdi Dibeklioğlu
Jury Members: Varol Akman, Mustafa Özdal
Innovation Expert: Mustafa Sakalsız

Progress Report
November 6, 2017

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

# Contents

# 1 Introduction

YouTalkWeSign will be a web application which is designed to serve people with hearing impairments or deafness. In today's world, there are many people who suffers from hearing loss and deafness. According to the last records of World Health Organization (WHO), there are 360 million people worldwide, corresponds over %5 of the world population who have disabling hearing loss. Moreover, they also claim that 1.1 billion young people, who have age between 12 and 35 years, are at the risk of hearing loss because of the exposure to the noise in recreational settings. The general reasons of the hearing loss are genetic causes, complications at birth, chronic ear infections, the use of certain drugs and growing old. People who have a significant hearing loss which also called deaf frequently use sign language to communicate with other people.

Sign language is a language which primarily uses manual communication to have a meaning, as opposed to spoken language. It is basically combination of simultaneous hand shapes, orientation, facial expressions and movement of the hands, arms or body to express ideas and communicate with other people. Sign Language helps to build bridge between people who can hear and who cannot. In addition to this, Sign language is not only used by deaf people, but it also can be used at some circumstances among people who have suffering from hearing and talking to each other such as very crowded or very quiet areas. Despite the significant role and

widespread usage of the sign language in human lives, it is not cared sufficiently and there are no enough resources.

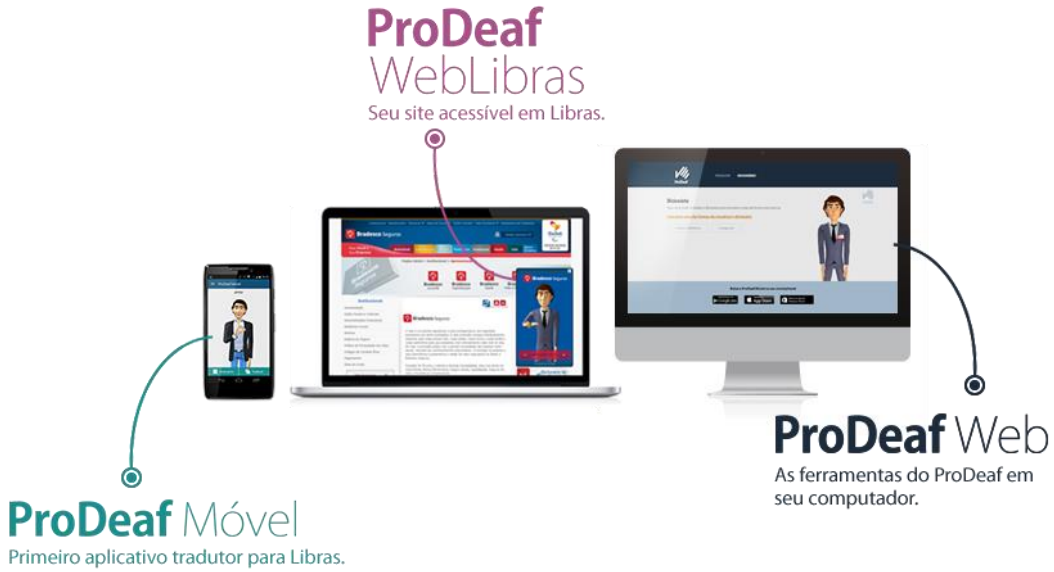
With YouTalkWeSign, we aim to provide easily accessible source to create sign language of a video from YouTube. Thanks to the YouTalkWeSign, we are hoping that people who have disabling hearing loss can watch videos they like with understanding what kind of conversations and sounds are in the videos. It would also be a good resource for other people who do not have a hearing loss, but want to learn sign language.

## 2 Current system

There are many different applications about sign language at the internet. However, the idea of converting sound or text to the sign language is not that popular. We have found three important applications for this purpose in which two of them are Brazilian applications.

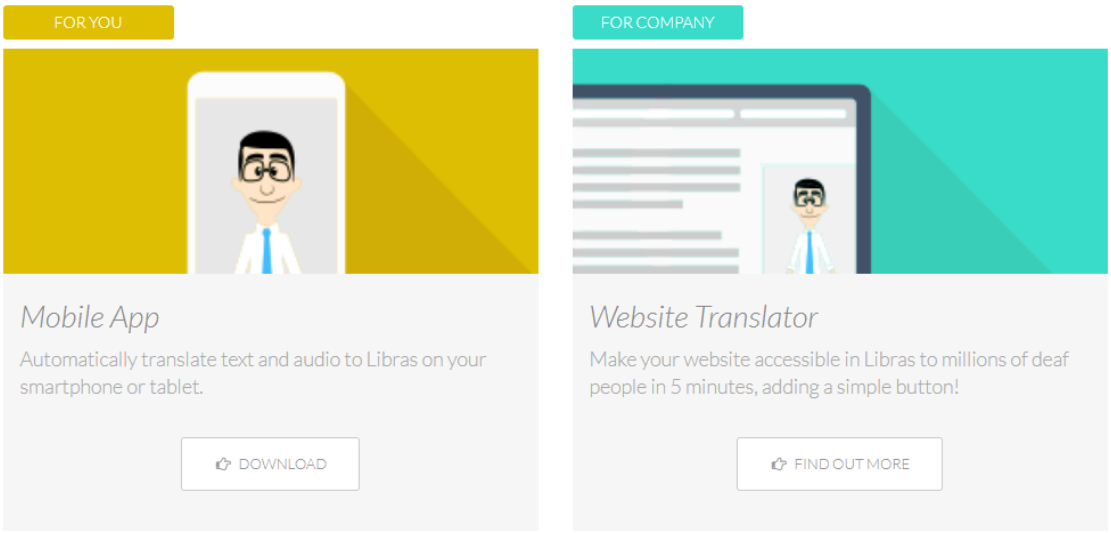
### 2.1 ProDeaf

ProDeaf is a set of software that translates text and voice to Portuguese Libras - Brazilian Sign Language - to allow communication between the deaf and hearing. Their solutions are designed to allow businesses to promote accessibility and social inclusion to their customers and employees (http://prodeaf.net/) [1].




## 2.2 Hand Talk

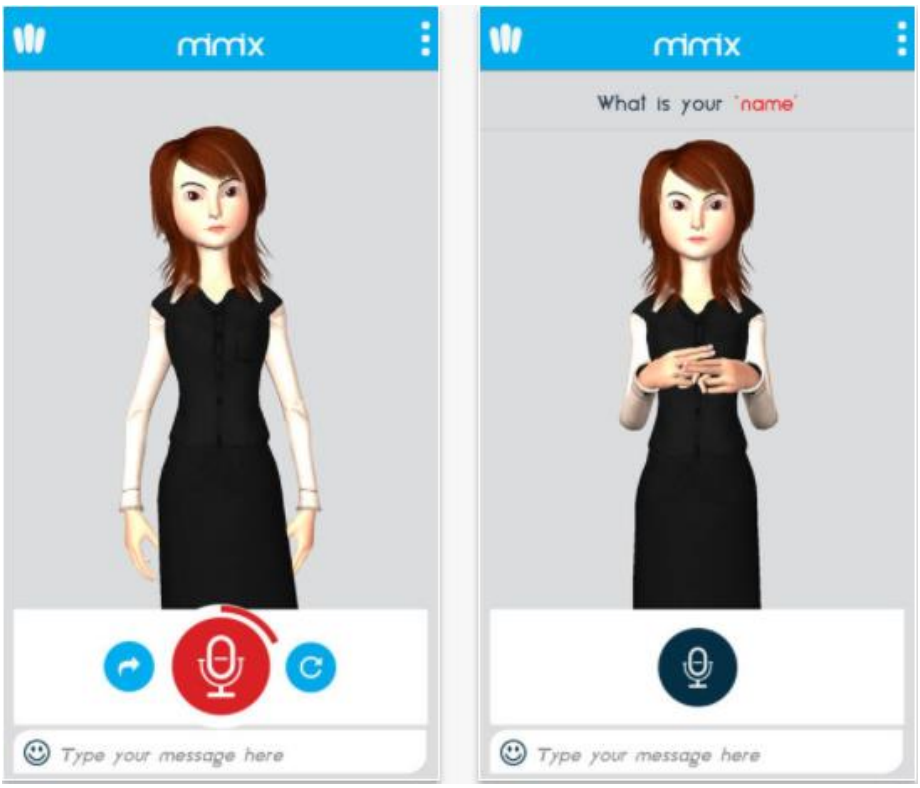
They offer digital translation for Brazilian Sign Language. They use Hugo, avatar, their virtual interpreter, to make deaf communication more accessible (https://handtalk.me/) [2].




## 2.3 Mimix3D

Mimix3D Sign Language Translator is a mobile app that interprets spoken and written English into sign language using a 3D friendly avatar. New signs are added every week to their ios application

(https://itunes.apple.com/tr/app/mimix3d-sign-language-translator/id1156035569?l=tr&mt=8) [3].




After searching very deep through the internet, we have found these three applications. However, these three applications are not really related to our idea. They do not have anything about video to sign language translation. Therefore, since all three of them are similar in terms of functionality, there does not really exist a project about our idea. Also, the YouTube integration makes our application unique.

On the other hand, projects that convert sign language to the text is available, but our idea is the exact opposite of this idea.

# 3 Proposed system

## 3.1 Overview

YouTalkWeSign will be a web application which is designed to serve people with hearing impairments or deafness. Through an on-screen avatar, the app will translate the spoken words in YouTube videos into sign language. The app can be run by replacing the "youtube" word in the video address with "youtalkwesign". Therefore, our website is http://www.youtalkwesign.com/.

For most of the videos, YouTube has speech-to-text subtitles functionality. For these videos, we will simply get the speech text from the YouTube. Then, we will translate the text to the sign language. For some of the videos, there is no speech-to-text subtitles support. For example, live videos do not have it. For these videos, we will first convert speech to text. Then, we will translate the text to the sign language.

While we are doing this conversion, we will be using the state of the art solutions of deep learning like Convolutional Neural Networks, Recurrent Neural Networks and maybe even GANs. The training data will be collected from a TV channel’s morning news that also includes sign language explanation along itself. The model is to recognize speech and generate corresponding text, then from the text train the sign language correspondence.

Therefore, for example, when user makes this request to our website: https://www.youtalkwesign.com/watch?v=oxi7HcgWga0,

we will generate the sign language from the text transcript of the video. Then, we will display it on the right side of the screen near the video. Hence, the video and the sign language avatar will be played simultaneously.

## 3.2 Functional requirements

- Users can watch any video from YouTube with sign language translation by replacing the "youtube" part before the ".com" with "youtalkwesign".
- Users can switch to a theatre mode while watching the video. That is, video takes %80 of the screen and avatar will take up to other %20.
- Users can change video settings such as quality, sound and full-screen in application website just like they do at the YouTube.
- Users can pause the video. When it is paused, the avatar will also be paused. They play and pause simultaneously.
- Users will be able to see trending videos regarding view counts at our website.
- Users will be able to search for a video with keywords just like they do at the YouTube.
- Users can register to our website to benefit from keeping track of their watch history and hearted videos by giving us a username, a valid email address and a password.

- After a successful registration, the system will redirect user to the home page as logged in.
- Registered users can log in to the website with their username and password at any time and become authenticated.
- Authenticated users can see their watch history.
- Authenticated users can heart a video.
- Authenticated users can see their hearted videos.
- Authenticated users can remove heart from a hearted video.
- Authenticated users can log out.

### 3.3 Non-functional requirements

- YouTalkWeSign will have a good-looking and attractive user interface. It will be user friendly and easy to use.
- YouTalkWeSign will be suitable for extensions. For example, we can increase the number of languages supported. Also, the other platforms like Twitch and Dailymotion can also be integrated.
- A user focused system may be implemented to enhance the quality of the usage- via recommendations etc.
- We are aiming to make the conversion to the sign language as much as fast because we do not want our users to wait a lot. For live videos, they may wait longer than the normal videos. However, we will try to decrease the delay as much as possible.

- The system will be reliable such that we will retrieve the video from the YouTube without making any changes on the original version.

- YouTalkWeSign website will have a responsive design. That is, its front-end will be appropriate for a variety of devices and window or screen sizes.

### 3.4 Pseudo requirements

- For front-end of the web application, we will use:
  Thymeleaf (Java template engine)
  Bootstrap v4 beta (for a responsive and a good-looking design)
  jQuery v3.2.1 (for being a one-page app with ajax calls)

- For back-end of the web application, we will use Java technologies:
  Spring Boot (auto configured version of Spring MVC)
  Spring Security (JDBC authentication and authorization)
  Spring Data JPA (database operations with MySQL [login, register, history, hearted and trending videos])

- For the implementation of the web application, we will use:
  Eclipse IDE Oxygen Version
  Apache Maven (for a dependency management)

- Our server will be in a Digital Ocean droplet. Apache Tomcat 8 will be our web application manager where we will deploy our application.

- The back-end of the deep learning model will be implemented in Python 3.5 with the support of Tensorflow and Keras. The model will consist of Convolutional Neural Networks, Recurrent Neural Networks and supposedly Long Short-Term Memory.

- The training will be done by using videos of the same person presenting things via sign language parallel to spoken English language. The trained model will be stored in a cloud to be used simultaneously by multiple users.

## 3.5 System model

### 3.5.1 Use case scenarios

Use-Case 1

***Watch Video***

**Primary Actor:** Anonymous User

***Main Success Scenario***

1. User changes the URL of a youtube video (https://www.youtalkwesign.com/watch?v=oxi7HcgWga0) and presses enter.
2. Video starts with an avatar translator near it.
3. User switches to theatre mode to watch it at a bigger screen size.
4. User finishes watching.

***Alternate Flow***

-

Use-Case 2

***Watch Video from Trending***

**Primary Actor:** Anonymous User

***Main Success Scenario***

1. User enters the homepage of our website (http://youtalkwesign.com).
2. "About Us" page opens.
2. User sees the "Trending" link at the sidebar and clicks.
3. User sees the trending videos that are watched at our website.
4. User opens the first one.
5. Video starts.

***Alternate Flow***

4a. Trending videos do not interest the user.

4a1. User exits the website.

Use-Case 3

***Register***

**Primary Actor:** Anonymous User

***Main Success Scenario***

1. User changes the URL of a youtube video (https://www.youtalkwesign.com/watch?v=oxi7HcgWga0) and presses enter.
2. Video starts with an avatar translator near it.
3. User likes the video and decides to put a heart on it.
4. Application says, "This requires a login".
5. User sees the "Register" button.
6. Register form comes under the video while video is continuing to play.
7. User enters a username, a valid email address and a password.
8. The system succeeds registration and redirects user to the where he came from with automatically making him login.

***Alternate Flow***

5a. User does not care and does exit the website.

7a. User enters a username that is already taken.

7a1. The system redirects user to the step 6 with an appropriate info.

7b. User enters a not valid email address.

7b1. The system redirects user to the step 6 with an appropriate info.

Use-Case 4

***Login***

**Primary Actor:** Registered User

***Main Success Scenario***

1. User enters the homepage of our website.
2. User wants to watch the video that he has watched yesterday.
3. User clicks the "History" link at the sidebar.
4. Application says, "This requires a login".
5. User realizes that he is in the Incognito Mode at the browser.
6. User sees the login form at the top navigation bar.
7. User enters his username and password.
8. The system succeeds authentication.
9. User sees his history now.
10. User clicks to the video that he has watched yesterday.
11. Video starts.

***Alternate Flow***

7a. User forgets his credentials and exits the website.

8a. User enters bad credentials and fails to login.

8a1. The system redirects user to the step 7 with an appropriate info.

Use-Case 5

***Search for a Video***

**Primary Actor:** Anonymous or Authenticated User

***Main Success Scenario***

1. User enters the homepage of our website.
2. User sees the search form at the top navigation bar.
3. User searches for a “cnn news”.
4. Search results come as a list.
5. User opens the first result.
6. Video starts.

***Alternate Flow***

-

Use-Case 6

***Heart and Unheart a Video***

**Primary Actor:** Authenticated User

***Main Success Scenario***

1. User watches the videos that he has found from search results
2. User likes the video and wants to note it.
3. For this purpose, he puts a heart on the video.
4. The system saves the video to the hearted list of the user.
5. User watches the video.
7. User exits the website.
8. User goes outside.

***Alternate Flow***

8a. User thinks that the video is not that good, so he opens the website again.

- 8a1. He clicks the "Hearted" from sidebar.
- 8a2. His hearted videos come as a list.
- 8a3. User opens his last hearted video.
- 8a4. User removes the heart from the video.

Use-Case 7

***Video Settings***

**Primary Actor:** Anonymous or Authenticated User

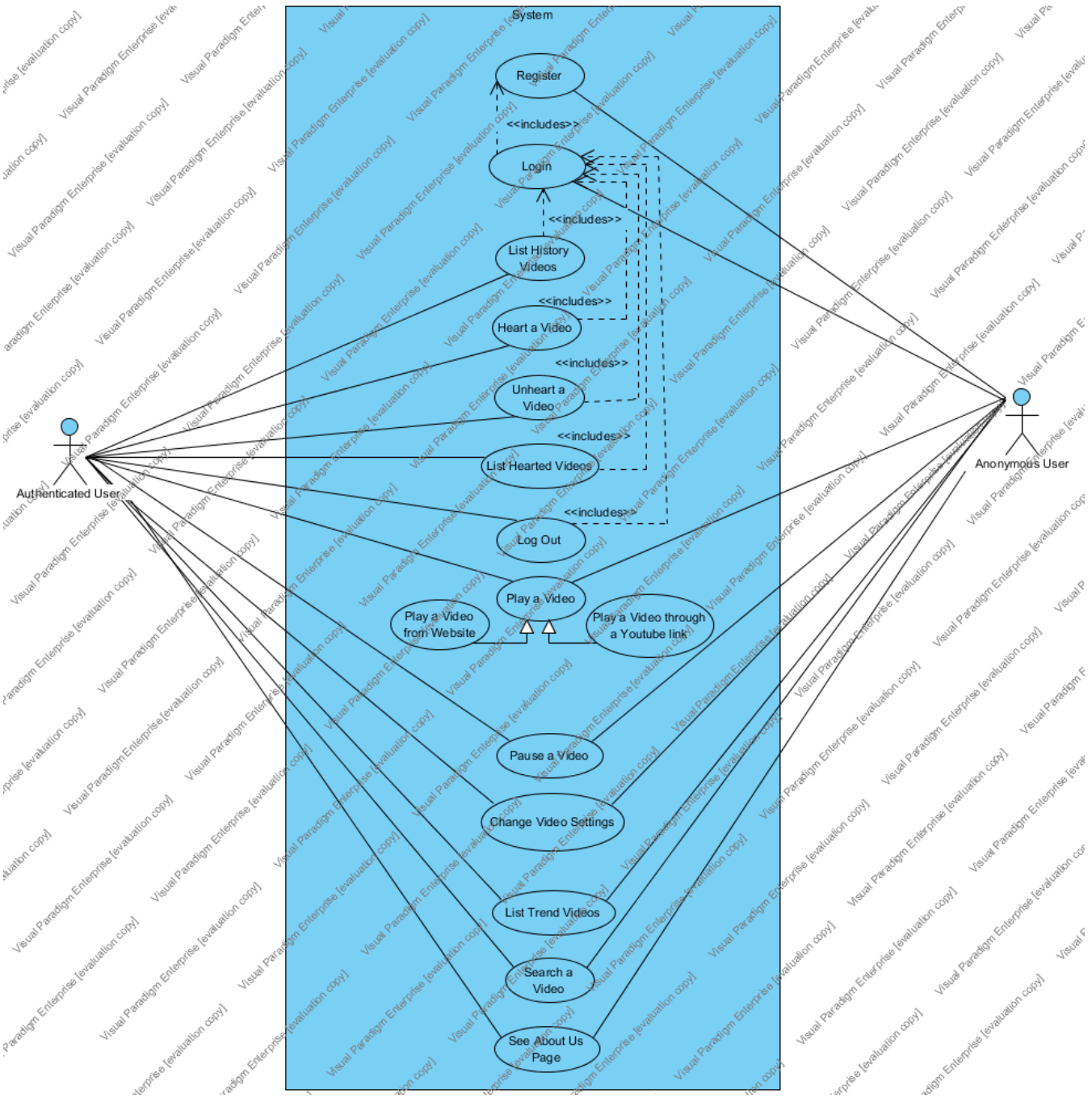
***Main Success Scenario***

1. User changes the URL of a youtube video (https://www.youtalkwesign.com/watch?v=oxi7HcgWga0) and presses enter.
2. Video starts.
3. User pauses the video, so the avatar also stops translating.
4. User changes the video settings. That is, he changes the quality of the video from auto to 720p.
5. User continues watching.
6. User finishes watching and logs out.

***Alternate Flow***

-

### 3.5.2 Use case model




**Play a Video:** Users can play a video from the website or they can play a video through youtube link by replacing "youtube" with "youtalkwesign" in the video address.

**Pause a Video:** Users can pause a video.

**Change Video Settings:** Users can change video settings such as video quality, full-screen mode and sound.

**List Trend Videos:** Users can what videos are trending at our website.

**Search a Video:** Users can search for a video by entering a keyword.

**See About Us Page:** Users can see information about YouTalkWeSign and contact information about developers.

**Register:** Anonymous users can create a new account by entering a username, an email and a password.

**Log in:** Anonymous users can sign in to their accounts by entering their username and password if they are registered before.

**List History Videos:** Authenticated users can see their history videos.

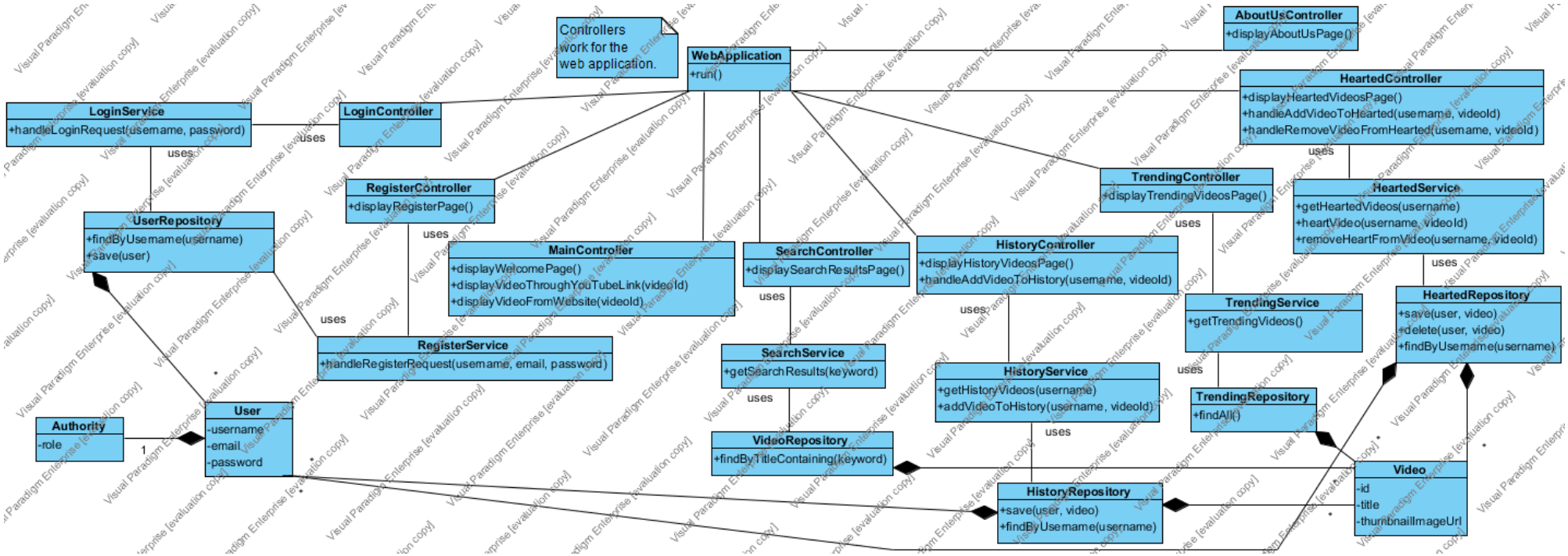
**Heart a Video:** Authenticated users can heart a video.

**Unheart a Video:** Authenticated users can remove a heart from a hearted video.

**List Hearted Videos:** Authenticated users can see their hearted videos.

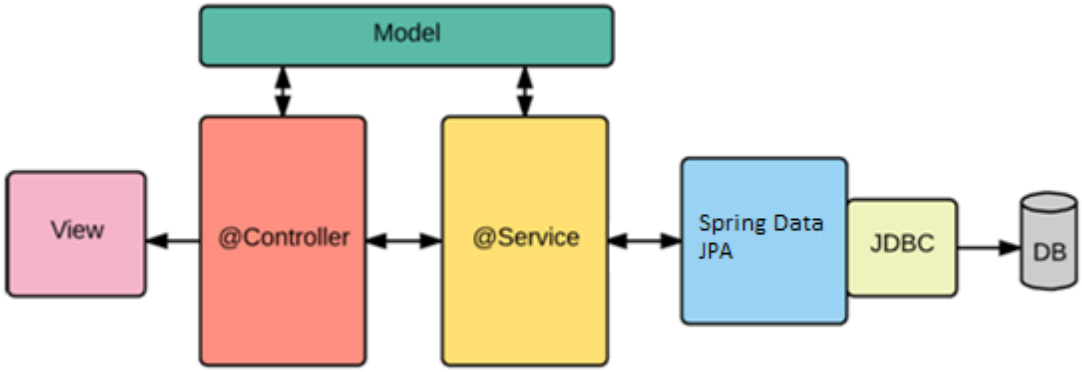
**Log Out:** Authenticated users can log out from their accounts.

### 3.5.3 Object and class model



Our application logic follows the idea of Spring MVC. Controllers catch the requests. Then, controllers use service classes to work for them. Service classes do their works on the database repository.

For example, when user wants to heart a video, by clicking heart icon, he sends a POST request to HeartedController. HeartedController catches this and calls HeartedService to make it insert necessary data to the database. After successful insertion to the database, Controller warns view to do the necessary changes. For example, heart icon's color changes.




- Our model classes are User and Video.
- Our controller classes are MainController, LoginController, RegisterController, SearchController, HistoryController, TrendingController, HeartedController and AboutUsController where each of them has corresponding service classes.

### 3.5.4 Dynamic models

#### 3.5.4.1 Sequence diagrams

#1

***Play video through a YouTube link, then see the trending videos***

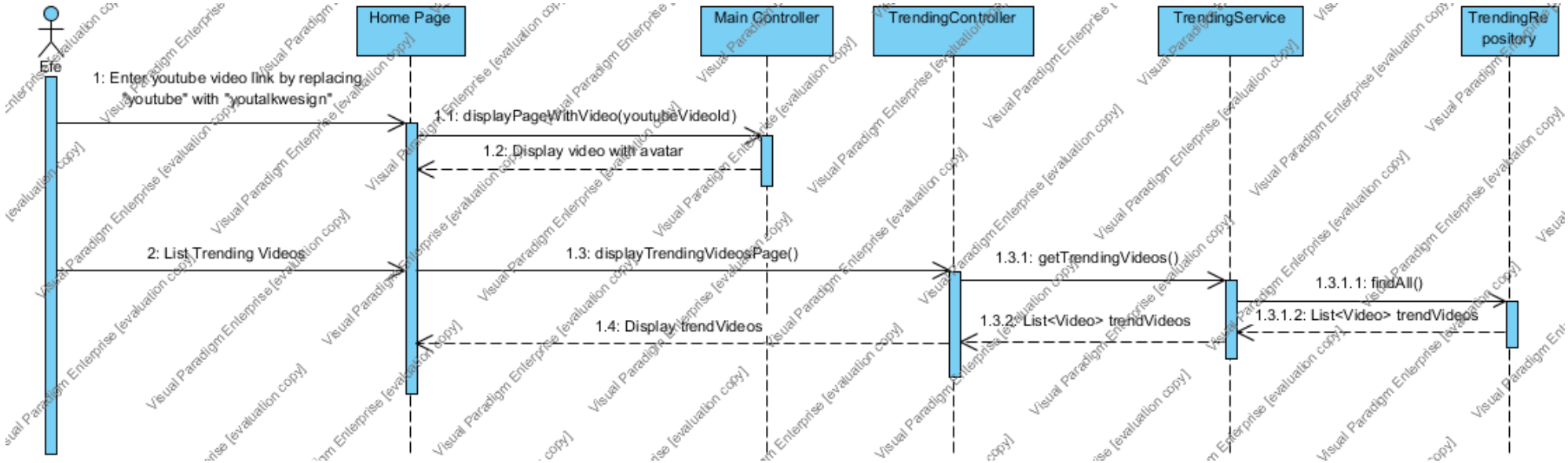
**Primary Actor:** Efe

***Scenario***

Efe changes the URL of a youtube video (https://www.youtalkwesign.com/watch?v=oxi7HcgWga0) and presses enter. He watches it. Then, he sees "Trending" link at the sidebar. He clicks and sees the trending videos.

***Description***

After Efe presses enter, our MainController catches this GET request. It takes the "v" parameter from the URL which is YouTube video id. MainController requests a sign language translation of the spoken words at the video as an avatar video from Python server-side. It gets the avatar video link from there. Then, it displays the video to the user with avatar translation near it. After Efe finishes watching, he presses "Trending" link. Now, TrendingController catches this request. It makes a call to the TrendingService to get the list of trending videos. TrendingService gets them from the database by using findAll() built-in method from Spring Data JPA library. After TrendingController gets the videos, it sets the view to open a list of videos. The videos will be decided by the list that is granted by TrendingController.


Efe
Home Page
Main Controller
TrendingController
TrendingService
TrendingRepository
1: Enter youtube video link by replacing "youtube" with "youtalkwesign"
1.1: displayPageWithVideo(youtubeVideoId)
1.2: Display video with avatar
2: List Trending Videos
1.3: displayTrendingVideosPage()
1.3.1: getTrendingVideos()
1.3.1.1: findAll()
1.3.1.2: List<Video> trendVideos
1.3.2: List<Video> trendVideos
1.4: Display trendVideos

#2

***Play video through a YouTube link, then see the trending videos***

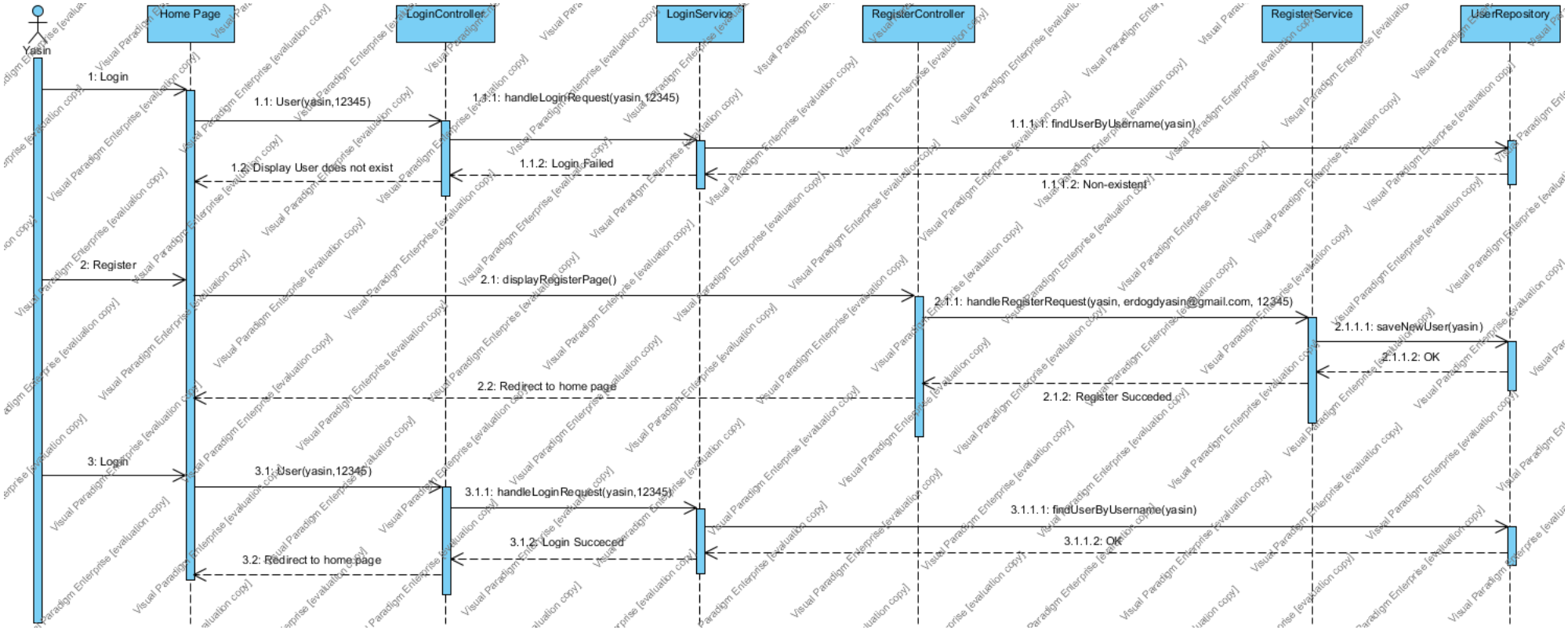
**Primary Actor:** Yasin

***Scenario***

Yasin enters the homepage of the website (http://youtalkwesign.com). He tries to log in to website through login form at the top navigation bar. Nobody tries to log in to a website that he has not registered before but Yasin does. Not surprisingly, login fails. Then, he sees the register button at the top navigation bar. He clicks and sees the register form. He enters a username, a valid email address and a password. Register succeds. He is redirected to the home page. Then, the system automatically logins Yasin. However, Yasin does not like this. He opens a new incognito window at the browser. He logins manually.

***Description***

When Yasin sends his credentials for login, LoginController catches this POST request. It calls LoginService to check if that account exists. LoginService calls built-in method from repository. Hence, it does not exist, and login fails. After Yasin clicks "Register", RegisterController catches this GET request and sets the view. Then Yasin fills the form and sends. RegisterController catches this POST request. First, it checks for validations. For example, is email valid? Is password null? After all validations, it calls RegisterService to save the user to the database. After all, it redirects Yasin to homepage because Yasin was at home page before the register request. Then, Yasin makes login call again. This time, procedure succeeds.


Yasin
Home Page
LoginController
LoginService
RegisterController
RegisterService
UserRepository
1: Login
1.1: User(yasin,12345)
1.1.1: handleLoginRequest(yasin,12345)
1.1.1.1: findUserByUsername(yasin)
1.1.2: Login Failed
1.2: Display User does not exist
1.1.1.2: Non-existent
2: Register
2.1: displayRegisterPage()
2.1.1: handleRegisterRequest(yasin, erdogdyasin@gmail.com, 12345)
2.1.1.1: saveNewUser(yasin)
2.1.1.2: OK
2.1.2: Register Succeded
2.2: Redirect to home page
3: Login
3.1: User(yasin,12345)
3.1.1: handleLoginRequest(yasin,12345)
3.1.1.1: findUserByUsername(yasin)
3.1.1.2: OK
3.1.2: Login Succeed
3.2: Redirect to home page

#3

***Get history videos, then, search for a video***

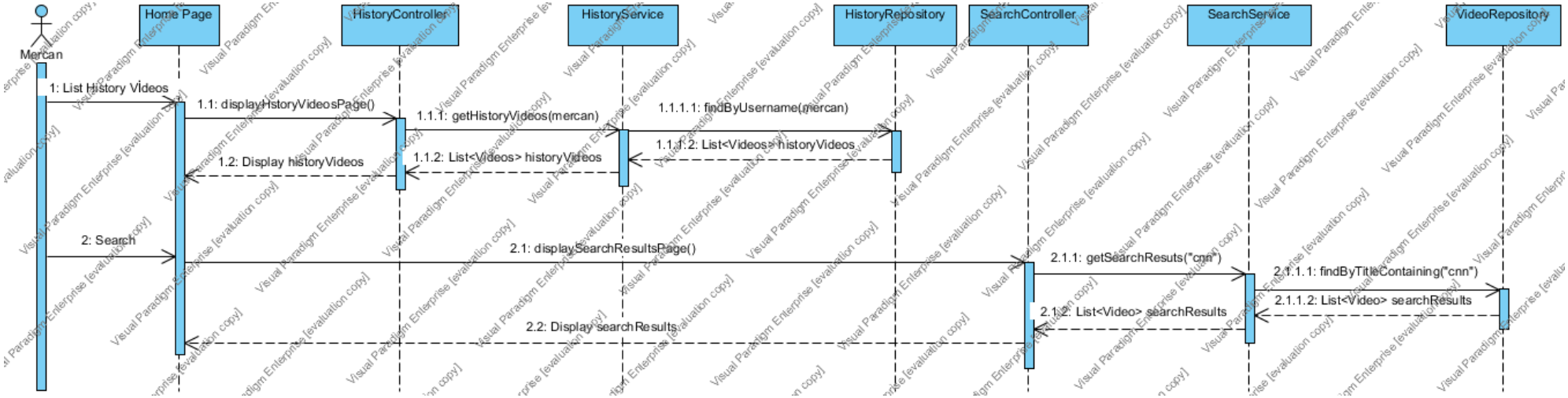
**Primary Actor:** Mercan

***Scenario***

Mercan thinks he has watched a video of CNN news before. Hence, he clicks the "History" link at the side bar. He realizes that he did not watch. Then, he searches for it. He sees the results as a list of videos afterward.

***Description***

When Mercan clicks "History" link, HistoryController catches this request. It first calls the HistoryService to get the list of videos that Mercan (findByUsername('mercan')) has watched before. Then, after videos have returned, it sets the view with these videos. Then, Mercan makes a search request. SearchController catches this POST request. It calls the SearchService. SearchService uses findByTitleContaining('cnn') built-in method. This method runs the query that includes %LIKE% at the back-end. Thanks to the Spring Data JPA framework, we will not deal with MySQL queries. After results return, SearchController sets the view to display the list of videos.


Mercan
Home Page
HistoryController
HistoryService
HistoryRepository
SearchController
SearchService
VideoRepository
1: List History Videos
1.1: displayHistoryVideosPage()
1.1.1: getHistoryVideos(mercan)
1.1.1.1: findByUsername(mercan)
1.1.1.2: List<Videos> historyVideos
1.1.2: List<Videos> historyVideos
1.2: Display historyVideos
2: Search
2.1: displaySearchResultsPage()
2.1.1: getSearchResuts("cnn")
2.1.1.1: findByTitleContaining("cnn")
2.1.1.2: List<Video> searchResults
2.1.2: List<Video> searchResults
2.2: Display searchResults

#4

***Get history videos, then, search for a video***

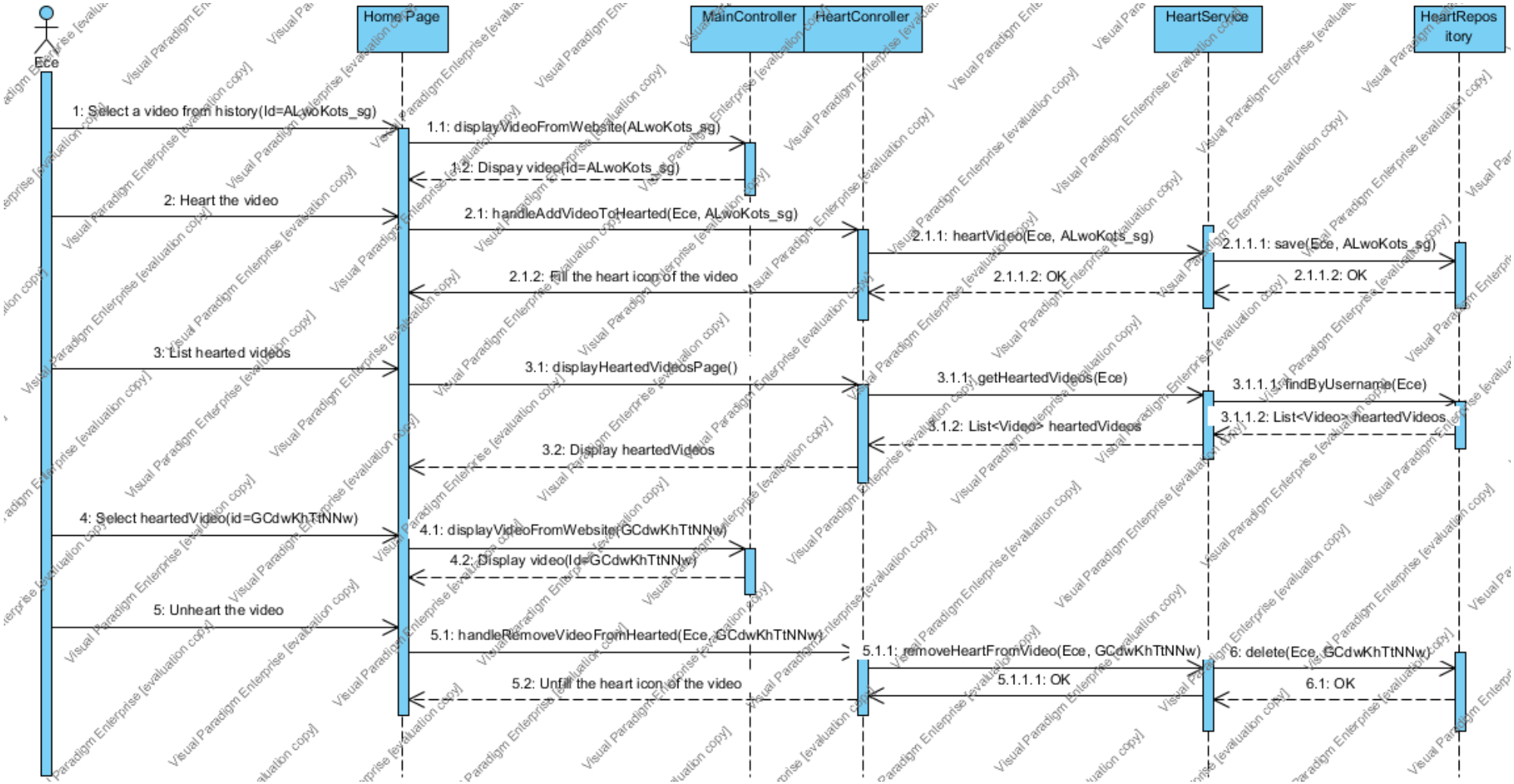
**Primary Actor:** Ece

***Scenario***

Ece opens a random video from history. Then, she decides she has liked it and puts a heart. Then, she wants to see her hearted list. She opens a video from there but she could not remember why she liked that video so she removes the heart from that video.

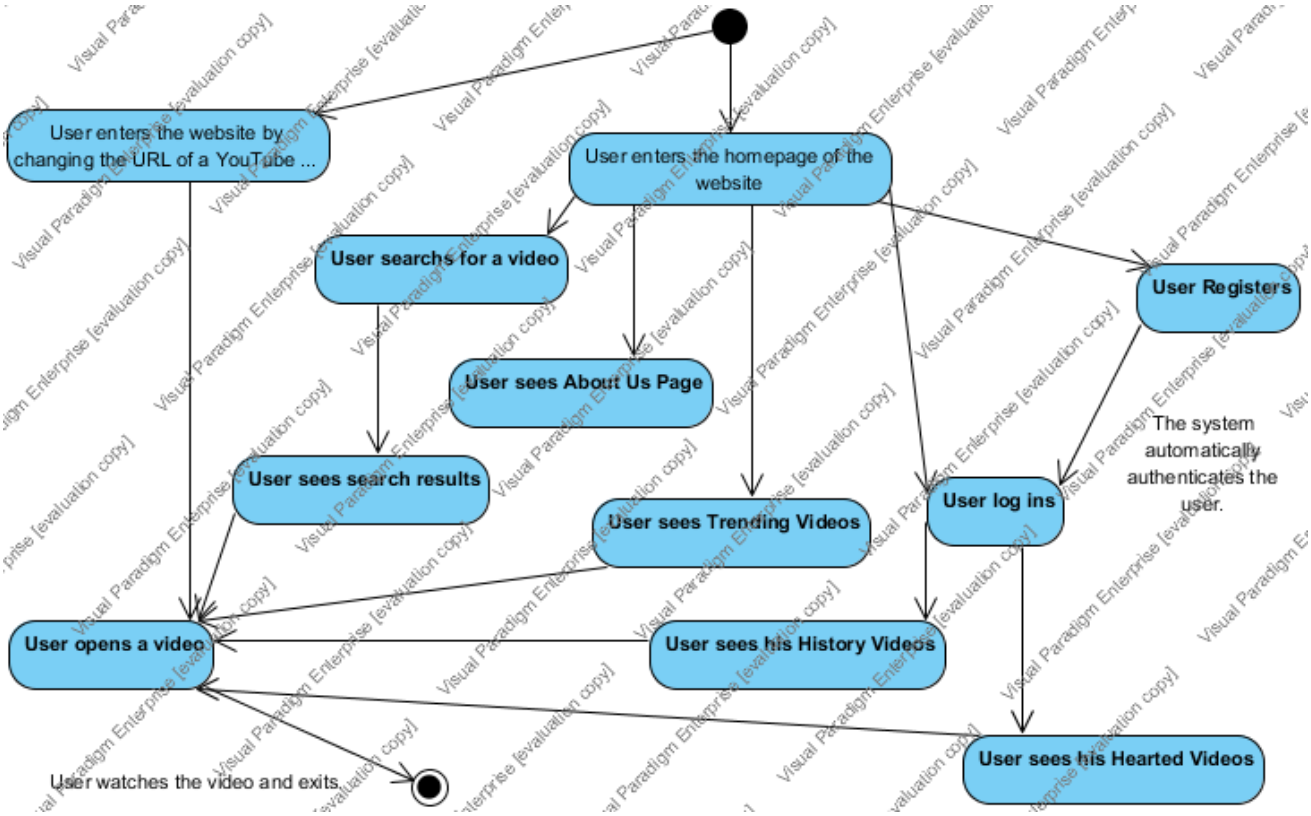
***Description***

When Ece puts a heart on the video, HeartedController catches this POST request. It handles the request by calling HeartedService. HeartedService, by calling repository, adds the video to the table which hearted videos are kept. After all, HeartedController gives appropriate info to the view because it wants Ece to see that operation has succeeded. Then, Ece makes a request to see her hearted videos. HeartedController catches this GET request. It calls its' slave again. HeartedService gives him the list of videos that Ece has hearted. Then, HeartedController sets the view to display this list of videos.


Ece
Home Page
MainController
HeartConroller
HeartService
HeartRepos itory
1: Select a video from history(Id=ALwoKots_sg)
1.1: displayVideoFromWebsite(ALwoKots_sg)
1.2: Dispay video(Id=ALwoKots_sg)
2: Heart the video
2.1: handleAddVideoToHearted(Ece, ALwoKots_sg)
2.1.1: heartVideo(Ece, ALwoKots_sg)
2.1.1.1: save(Ece, ALwoKots_sg)
2.1.1.2: OK
2.1.2: Fill the heart icon of the video
3: List hearted videos
3.1: displayHeartedVideosPage()
3.1.1: getHeartedVideos(Ece)
3.1.1.1: findByUsername(Ece)
3.1.1.2: List<Video> heartedVideos
3.2: Display heartedVideos
4: Select heartedVideo(id=GCdwKhTtNNw)
4.1: displayVideoFromWebsite(GCdwKhTtNNw)
4.2: Display video(Id=GCdwKhTtNNw)
5: Unheart the video
5.1: handleRemoveVideoFromHearted(Ece, GCdwKhTtNNw)
5.1.1: removeHeartFromVideo(Ece, GCdwKhTtNNw)
6: delete(Ece, GCdwKhTtNNw)
5.1.1.1: OK
6.1: OK
5.2: Unfill the heart icon of the video

### 3.5.4.2 Activity diagram

Users can start their activities in two ways. They either come to our page by changing YouTube link or come to our home page directly. If they come through YouTube, the video will start. Otherwise, if they are not logged in, they can do these: They can see about us page, they can search for a video and see the results, they can see the trending videos. From these lists, they can open any video they want. On the other hand, if they are not logged in, they cannot see their history videos or hearted videos. To be able to make them, they must register. After successful registration, they system will automatically authenticate them. Then, they will be able to use these features too.

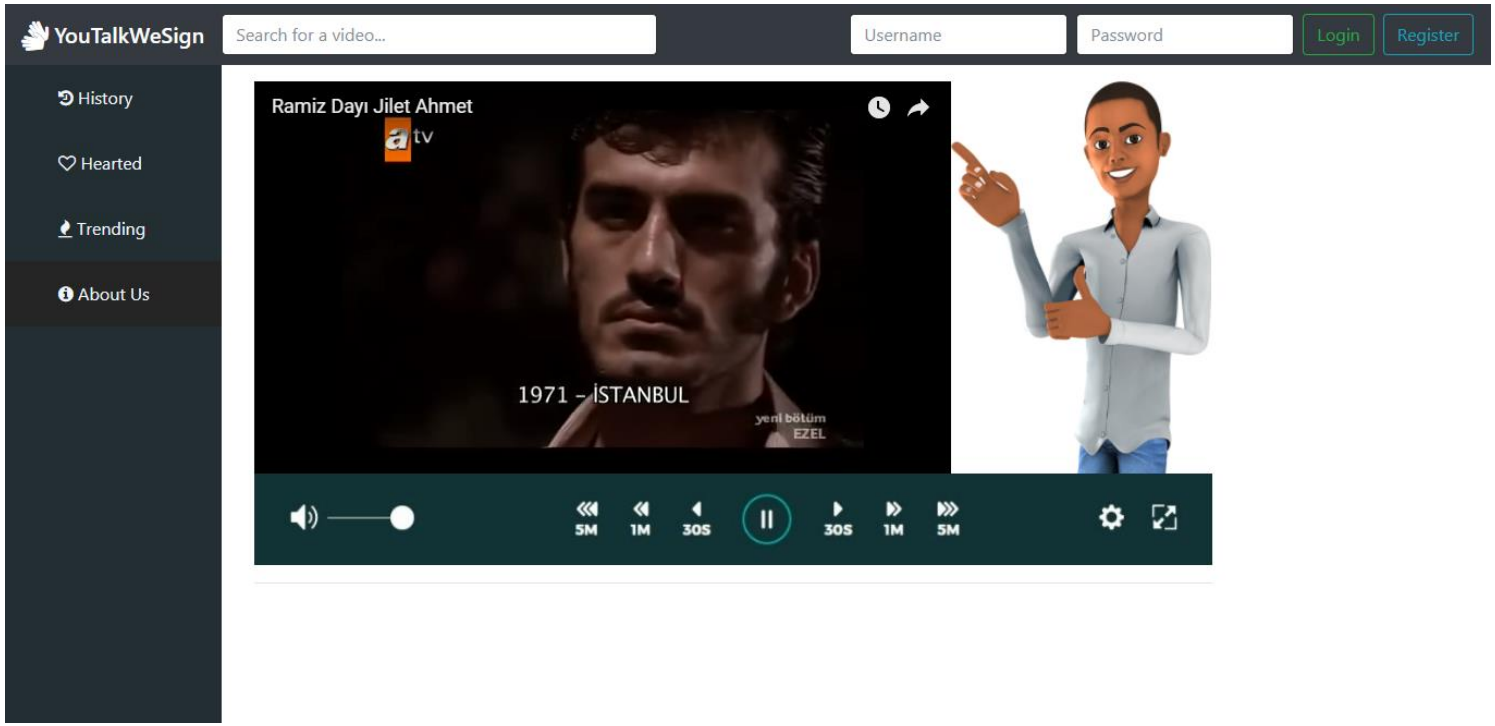


### 3.5.5 User interface

**Open a video by changing the link of the YouTube video**

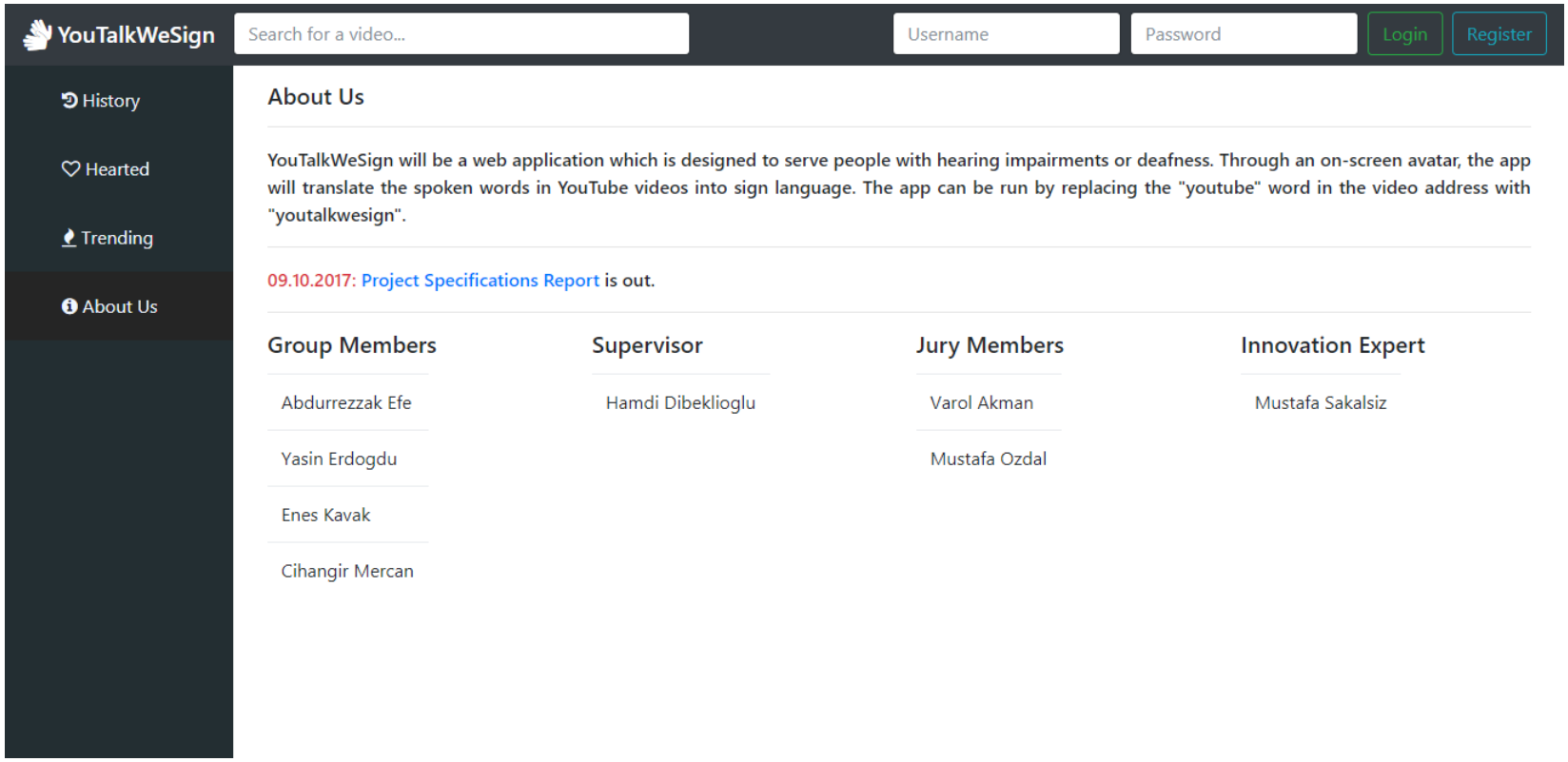
When we change this link: https://www.youtube.com/watch?v=qXkX52n3FJo to this link:

https://www.youtalkwesign.com/watch?v=qXkX52n3FJo, the website will open. As we are not a user that logged in, the screen shows the options for us to register or log in.



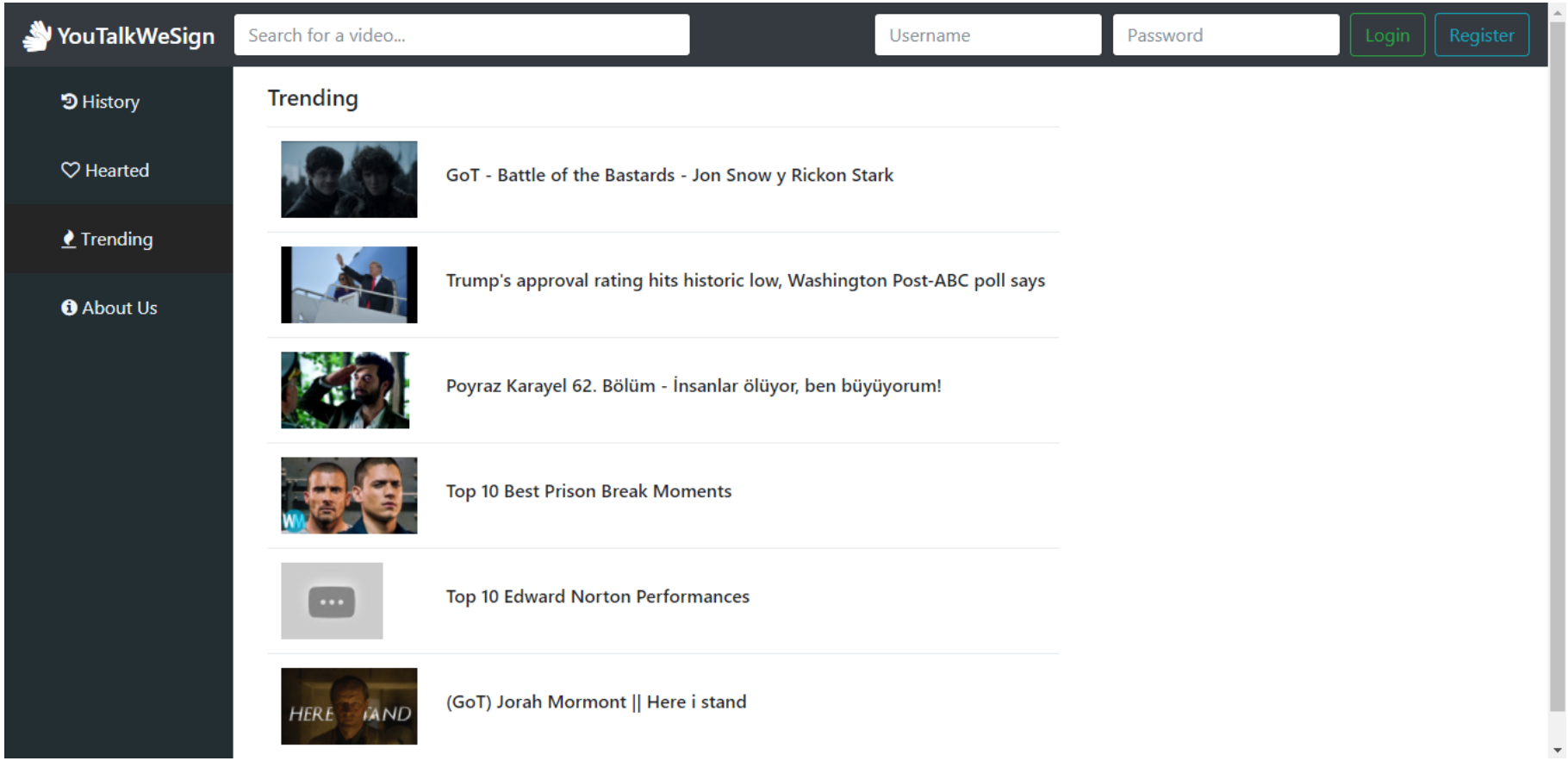
**Home page**

When we visit the homepage, following screen (About Us page) greets us. As we are not a user that logged in, the screen shows the options for us to register or log in.



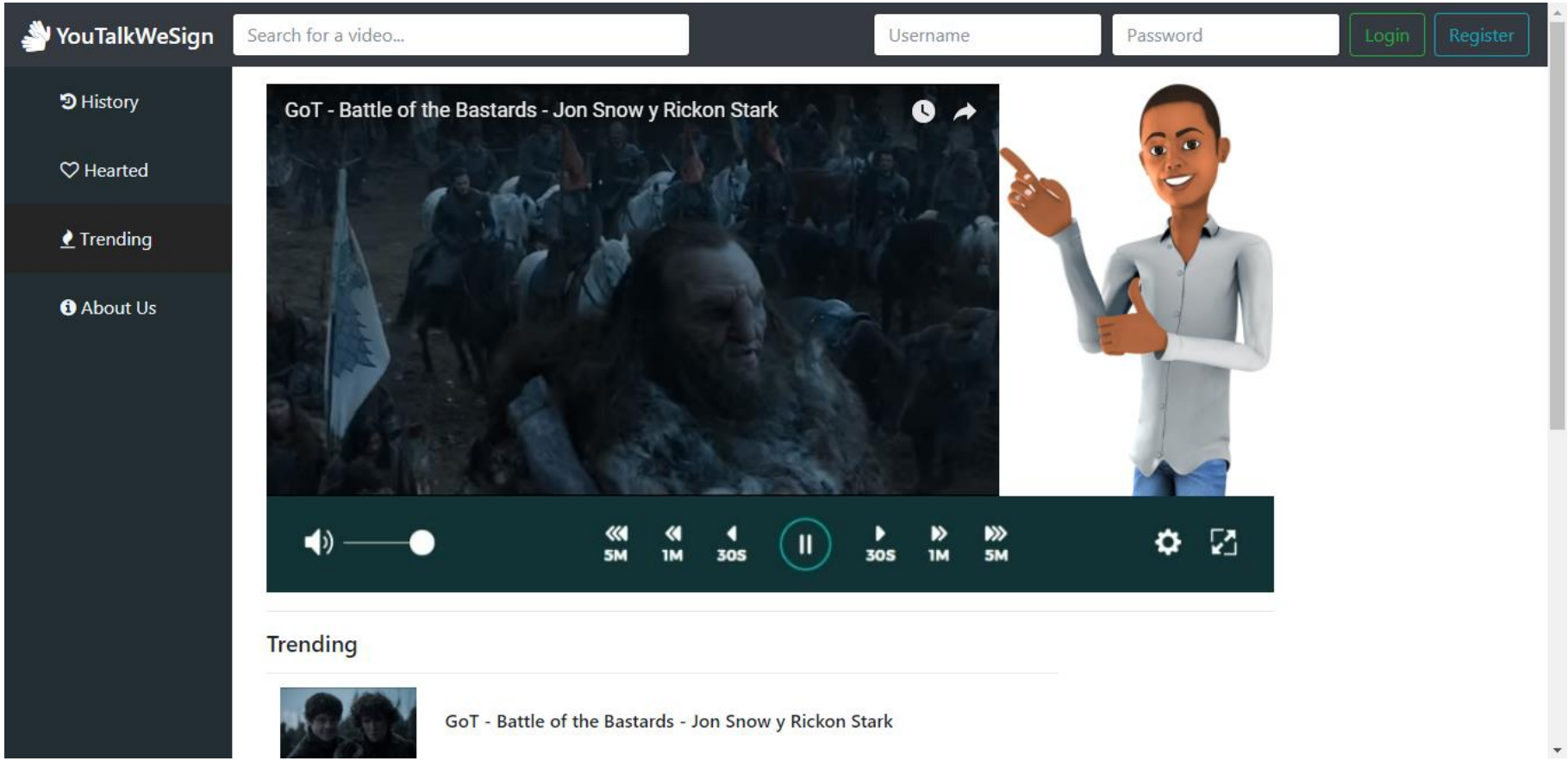
**Display trending videos**

Any user/visitor can access the videos that are on trend among other users by simply clicking Trending button on the left sidebar.



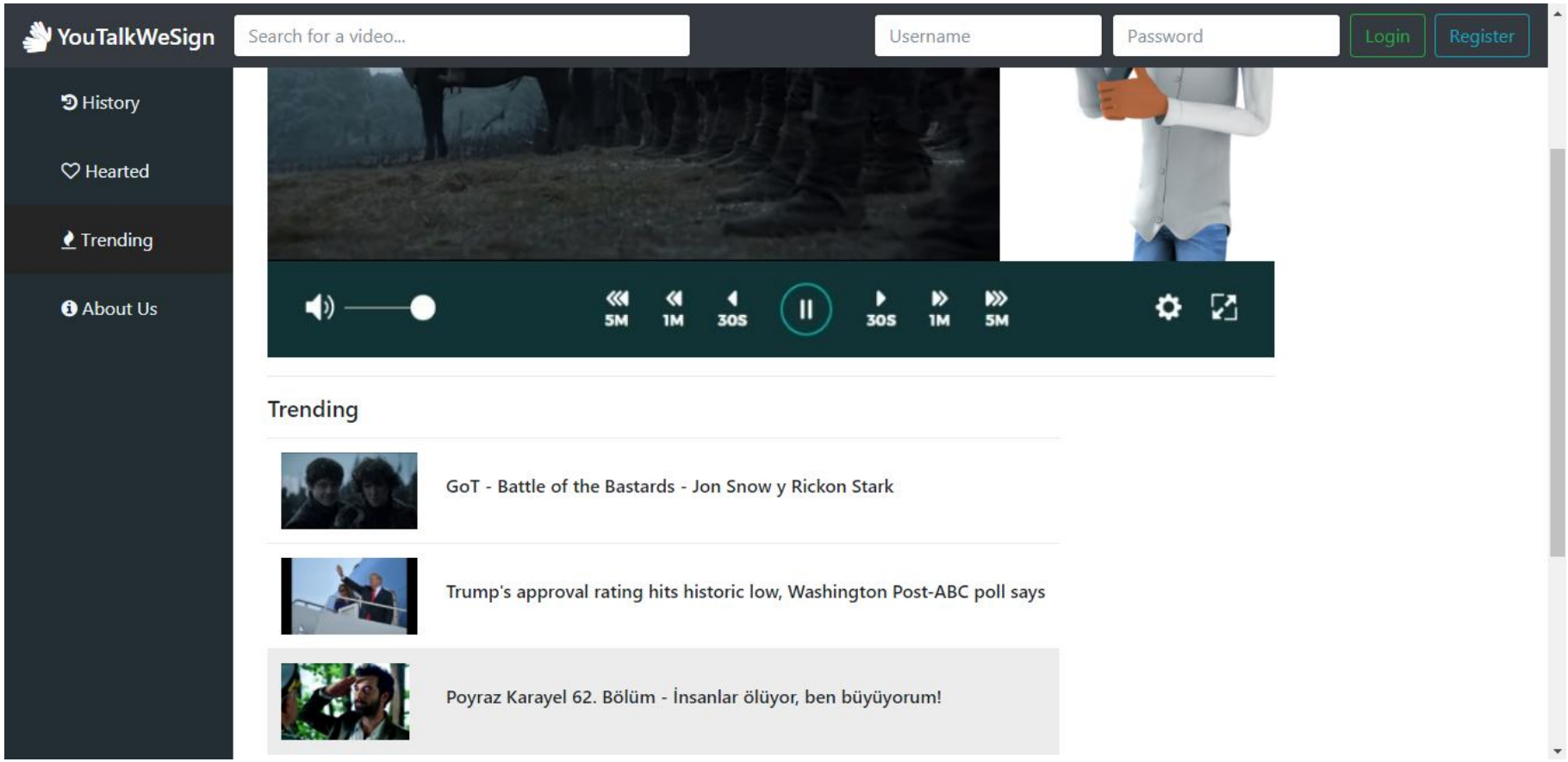
**Open a video from trending videos**

We can open a video by clicking to it. Then, the screen will stay same, but the video will start at the top of the trending videos list. YouTalkWeSign is a one-page web application.



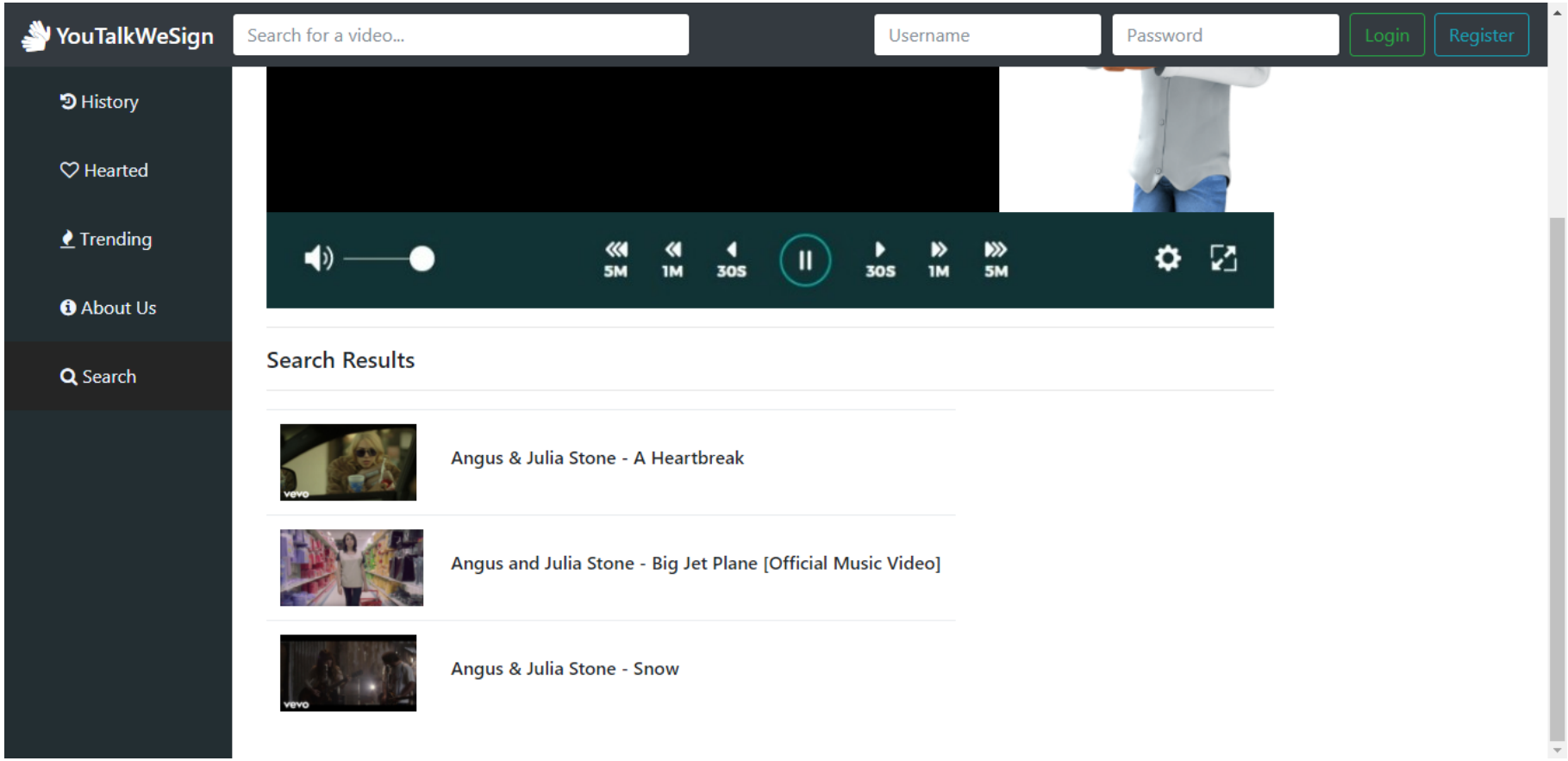
**Change the video playing**

We open another trending video at the same page.



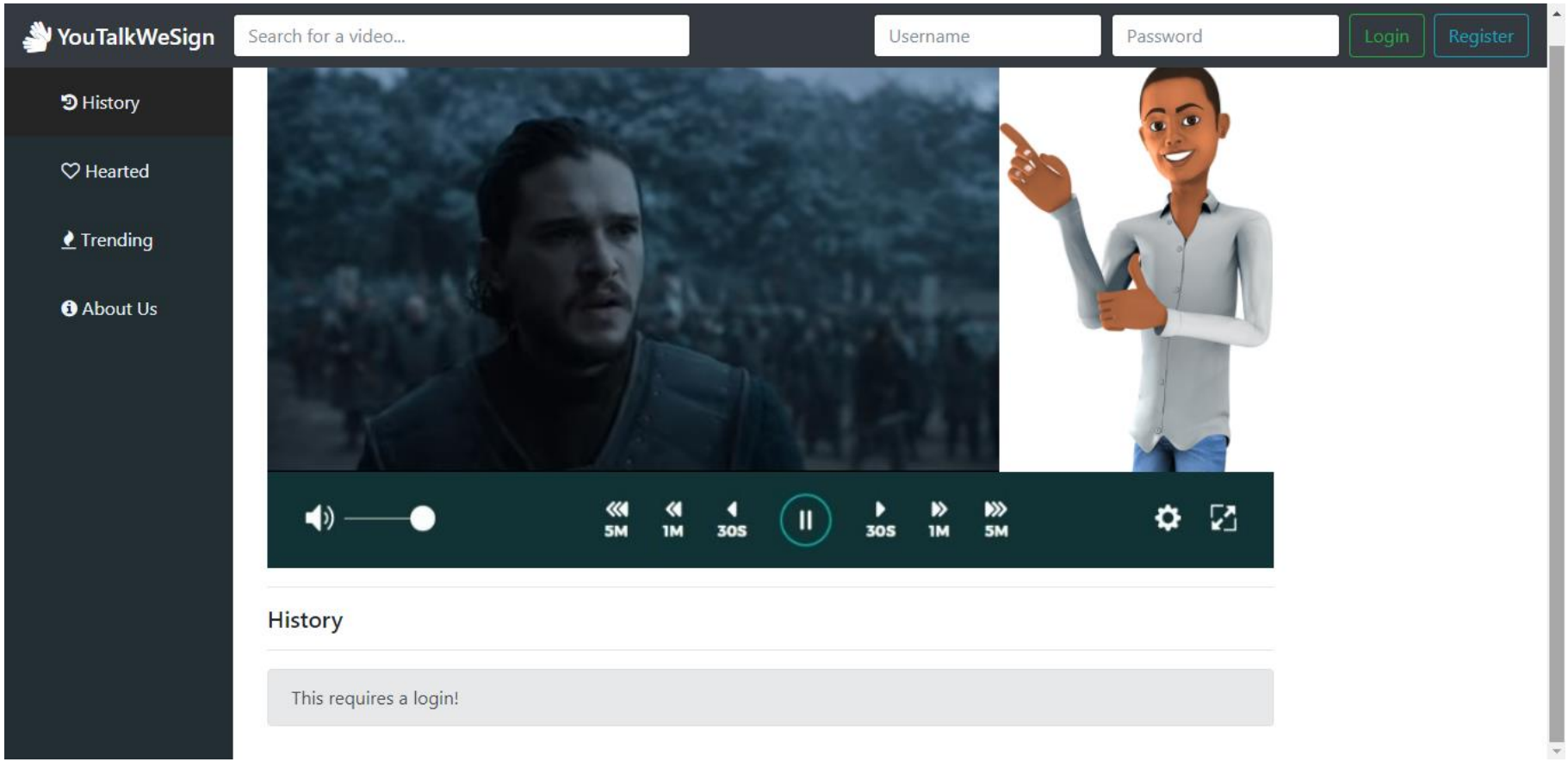
**Search for a video**

We search for a “angus and julia stone” while the video is still playing. Then, the results will come under the video. Also, as it is seen, the new “Search” link is also added to the sidebar.



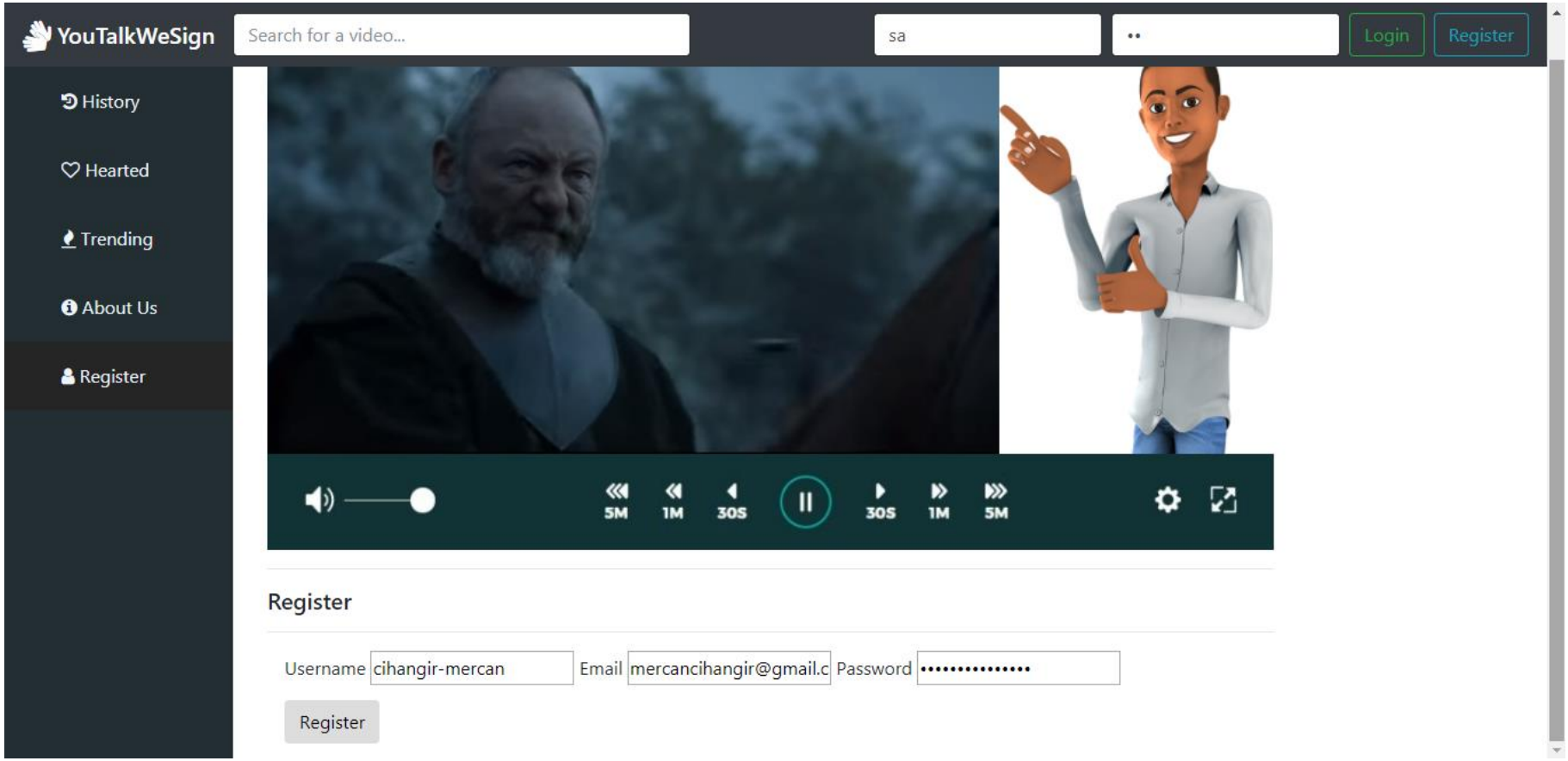
**See your watching history**

Only authenticated users can do this!



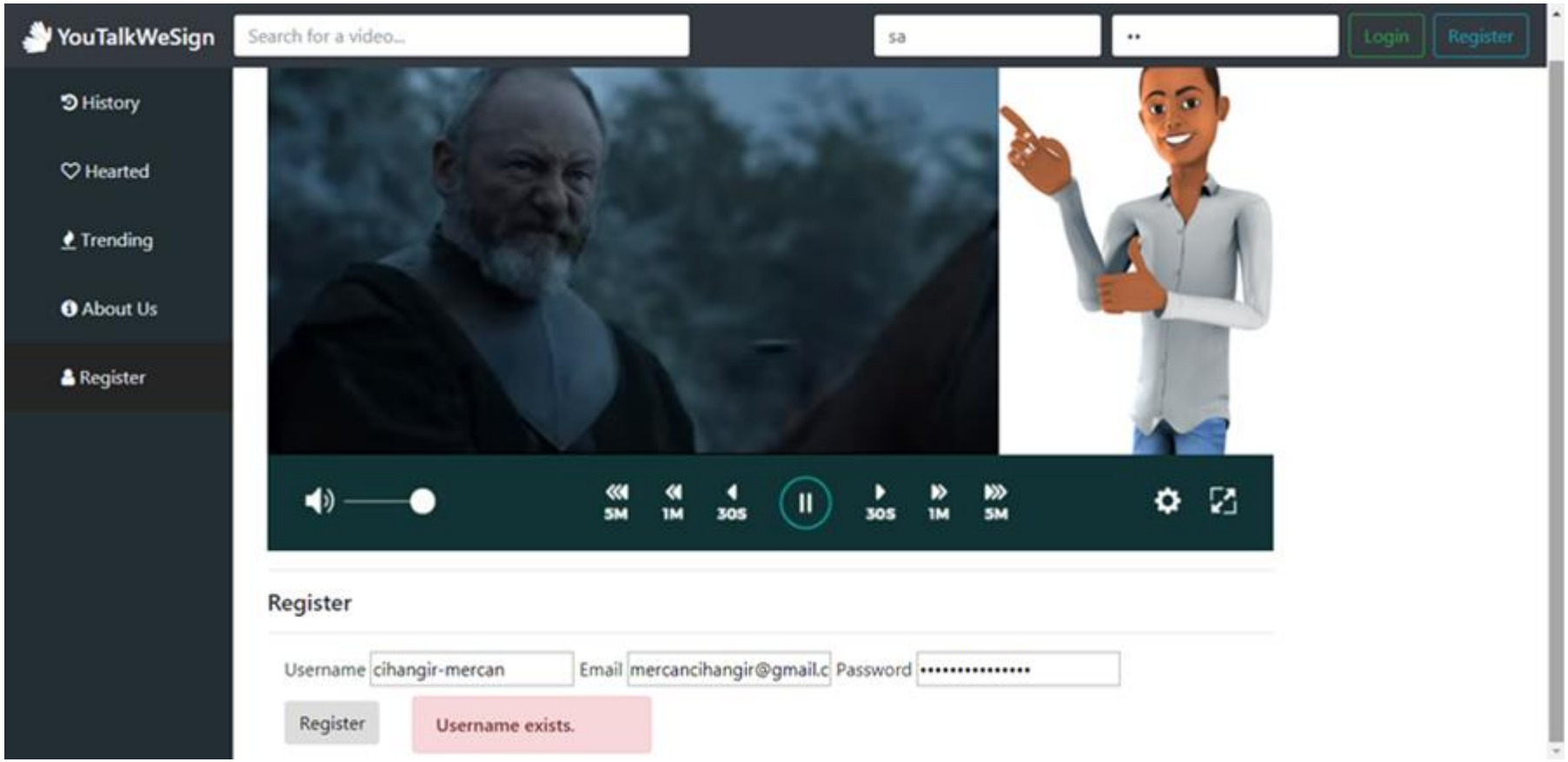
**Register**

We click the register button while the video is still playing, and we fill the form.



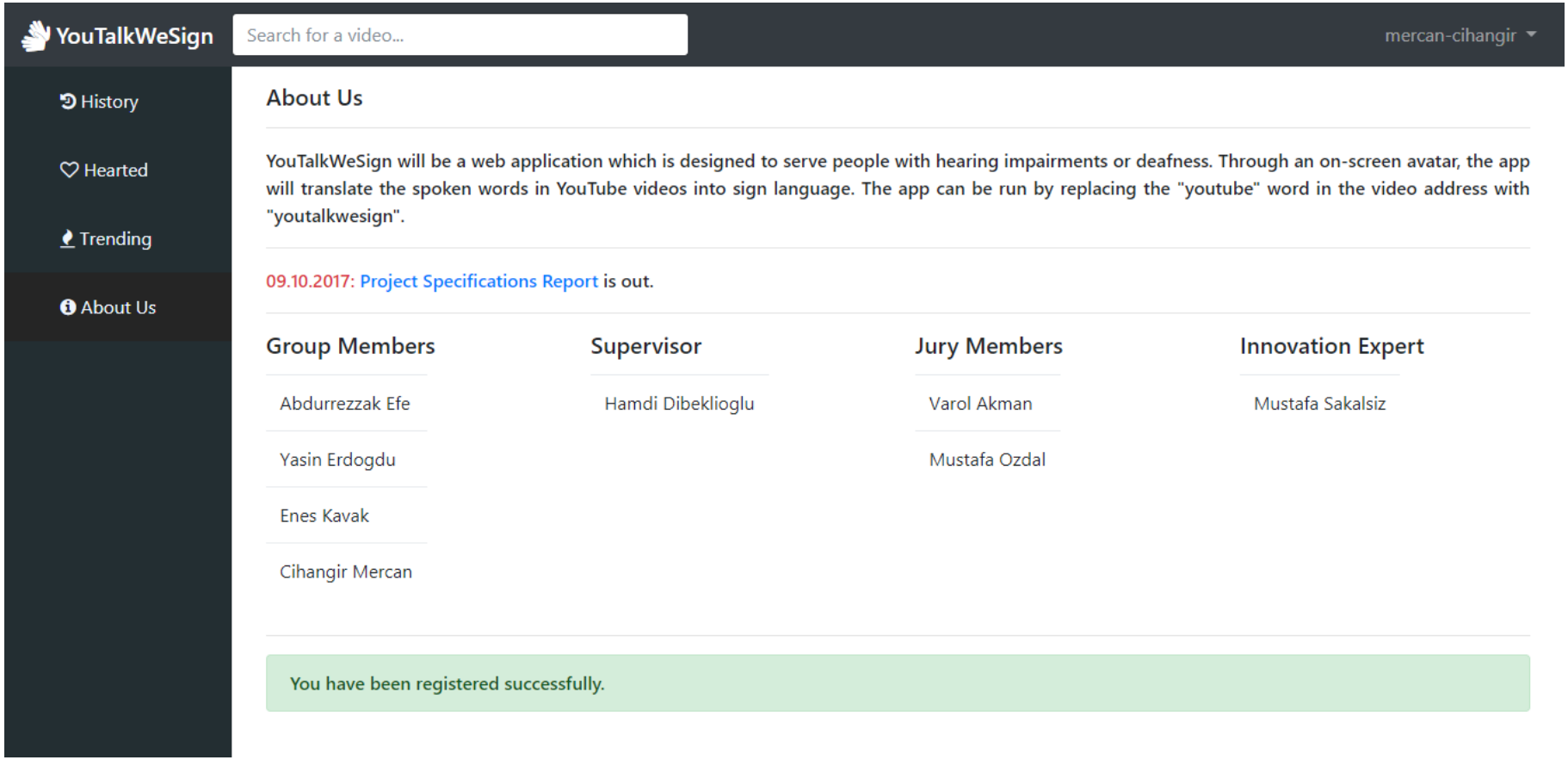
**Register Fails**

Username exists (alert-danger at the bottom).



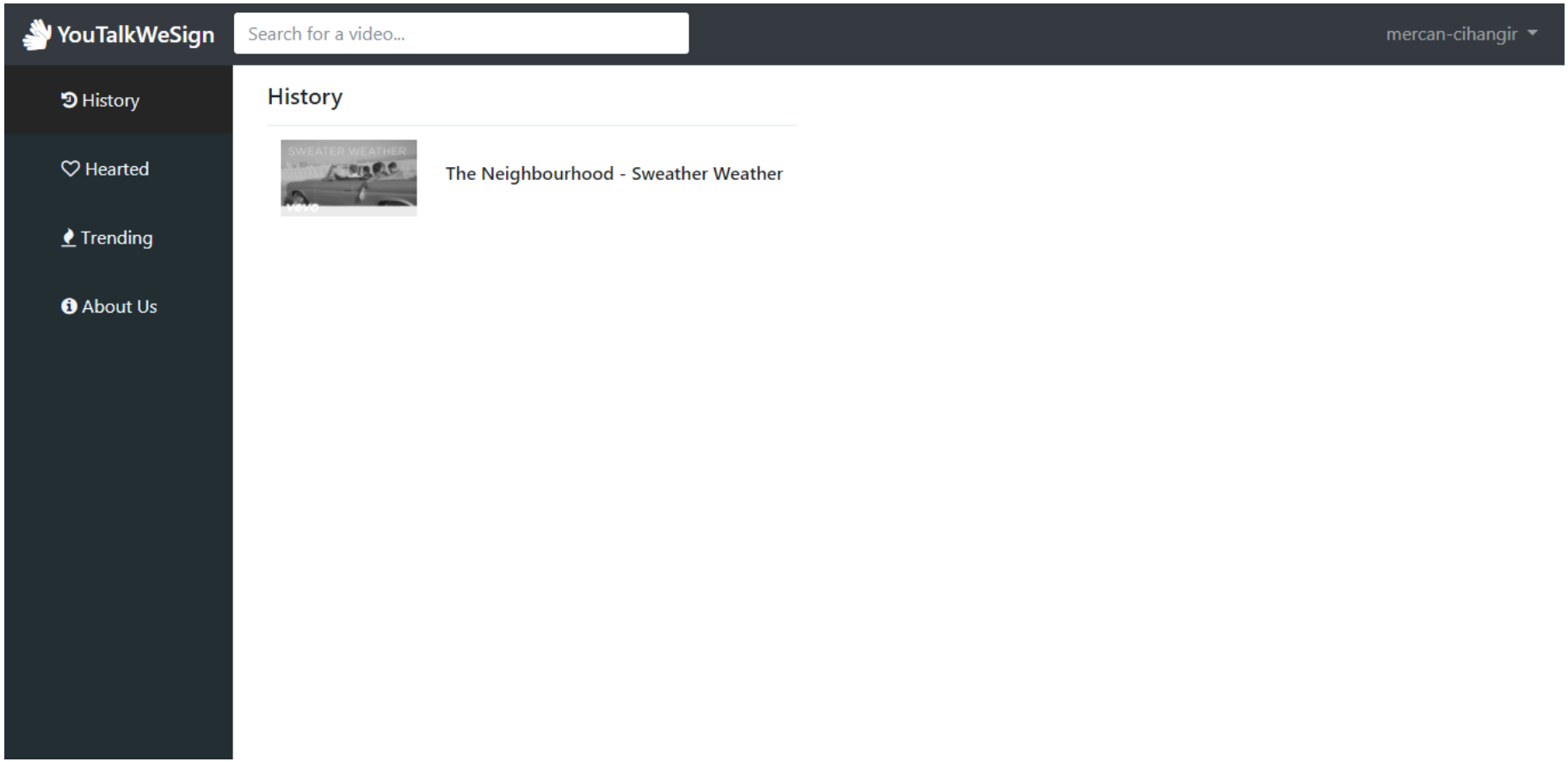
**Register Succeeds**

We try again with mercan-cihangir username. We succeed. The system automatically authenticates us as it is seen there is no more login form at the top navigation bar.



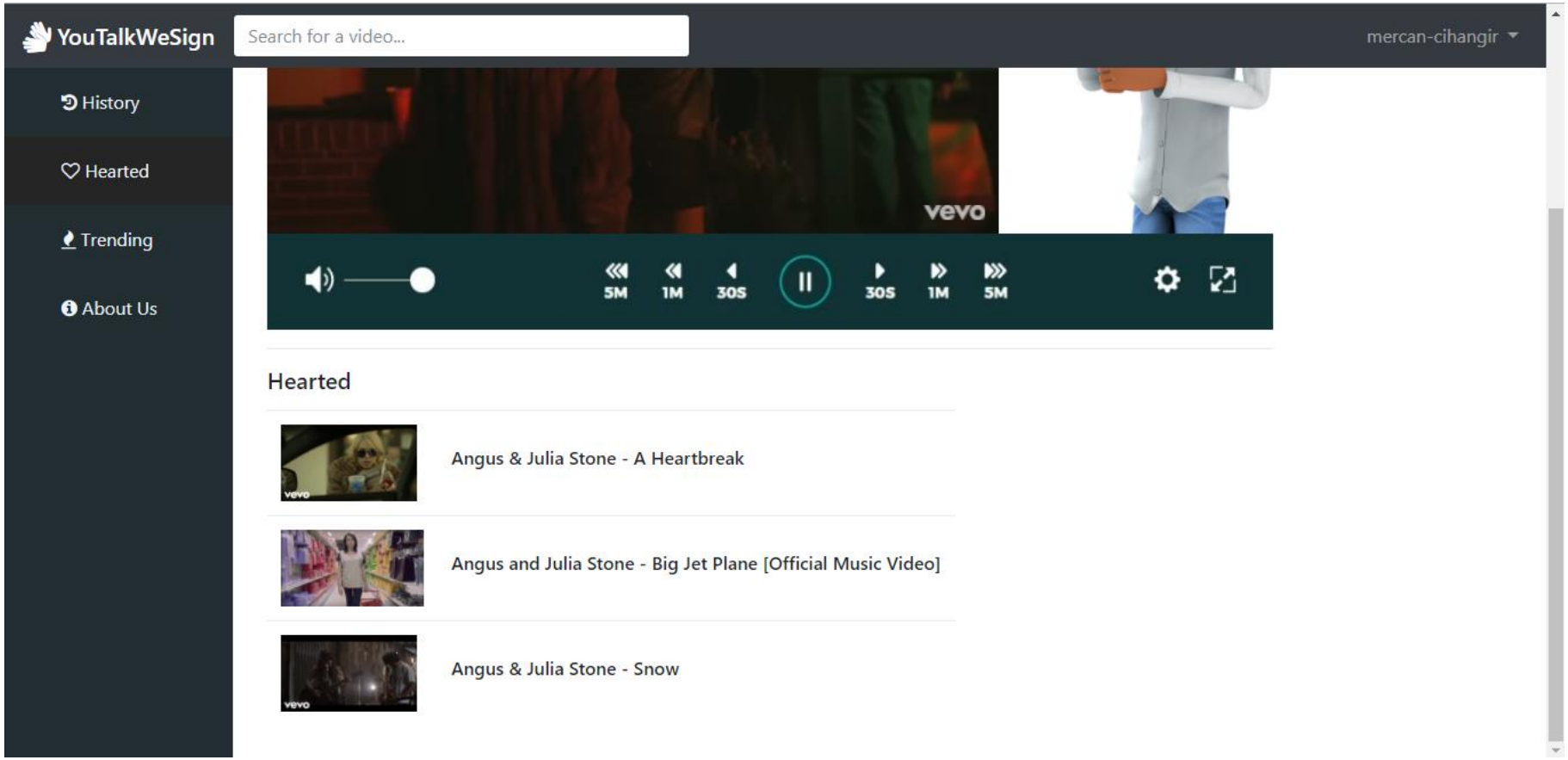
**See watching history**

We watch a video and close the website. Then, we open the website again and check our history videos.



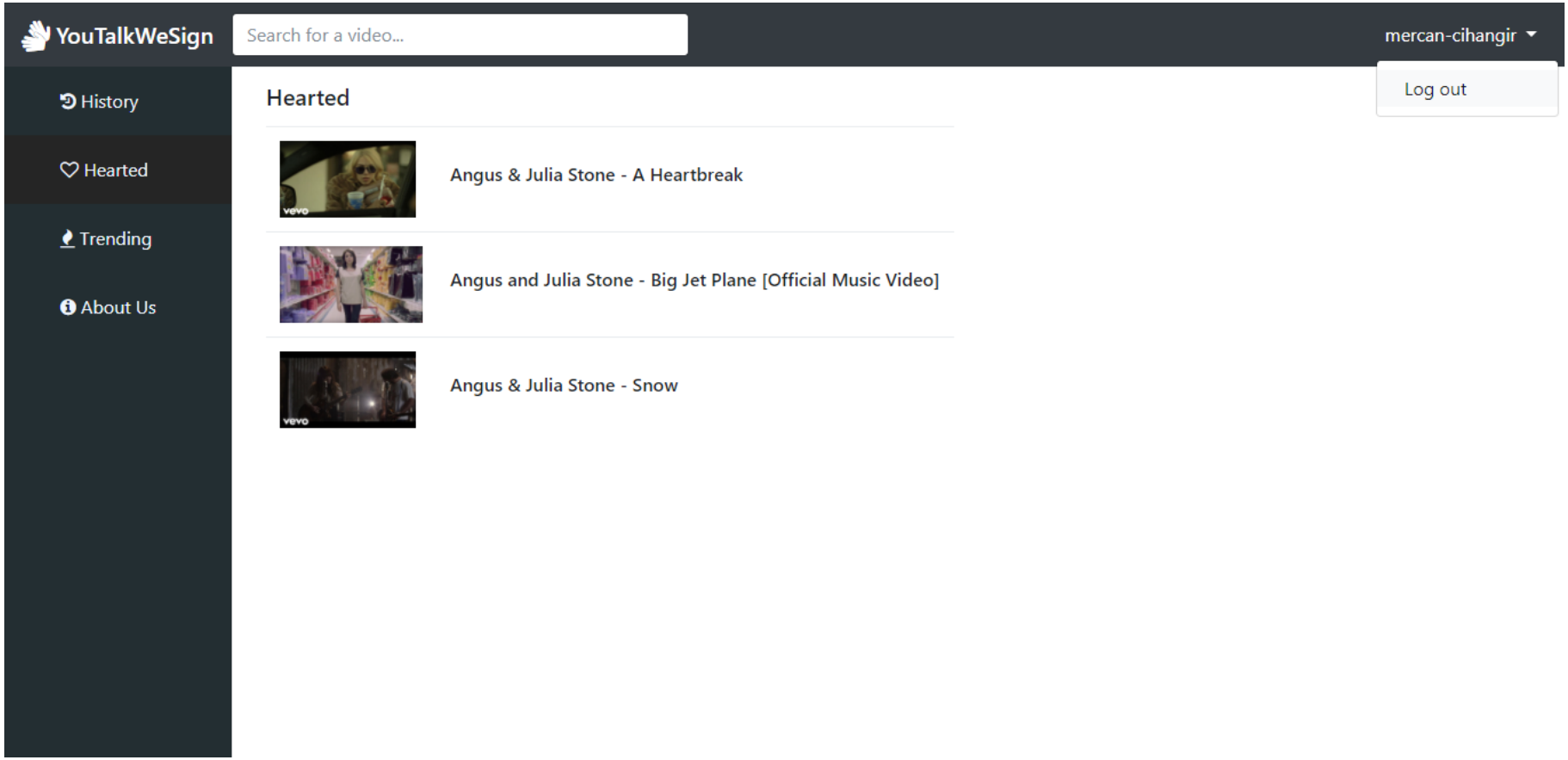
**Heart a video and see our hearted videos**

We open a video by changing the URL of the YouTube video. Then we like it. We decide to put a heart on it. There is not a heart icon yet at the controls of the video but there will. Then we want to see what we hearted until now.



### Log out

We log out from the drop-down menu at the top navigation bar.



## 4 References

[1] Pro Deaf. http://prodeaf.net/ [Accessed: Oct 6, 2017].

[2] Hand Talk. https://handtalk.me/ [Accessed: Oct 6, 2017].

[3] Mimix3D. https://itunes.apple.com/tr/app/mimix3d-sign-language-translator/id1156035569?l=tr&mt=8 [Accessed: Oct 6, 2017].