



Cooperative Systems

Course Project

EXCHANGE YOUR LANGUAGE WEBSITE

By

Awad Mukbil

TUC: 449652

LUH: 3167890

Suhair Ahmed

TUC: 449542

LUH: 3167910

Lin Wang

TUC: 471916

LUH: 3279750

Cooperation Systems

Winter Semester 2015/2016

Submitted in: 12 April 2016

*This document is for a mandatory project for **Cooperative Systems** course, winter semester 2015/2016. In this project, we have applied on of the ideas for **CSCL** (computer-supported collaborative learning) as a web application for exchanging languages, in a way that each user will have a partner and join a shared text editor and video/voice chat room to enable the communication, and enjoy exchanging their languages.*

1. Introduction

The Exchange Language website's main idea is to exchange languages between two users via social interaction using a computer through the Internet. The user has to sign in with his/her information and enters what is his/her mother tongue language and what is the desired language to learn. After that, he/she can search for his/her partner in the search bar offered in the website.

If the user is online, a notification is sent to the user and he/she can accept the invitation or not. If he/she has accepted it, both of them join one chat room and can easily start the learn session (texting and video/voice call), and exchange their language experiences with each other.

1.1. Team Management

Briefly, we are **three** students from ITIS Master program students, one of us lives in Goslar. So, we have used some useful tools in order to keep in touch and work collaboratively. The following table describe the Process Life Cycle of our group

Phase	Description	Useful Tools
Forming	Done easily in the lecture period, all of us are ITIS students	None
Storming	Two phases: 1. During lecture period (meeting) 2. At home using tools	- Stormboard : Useful tool enables anyone to put his idea in shared bulletin board - Skype .
Norming	No strict rules need to be followed	None
Performing	At the beginning each person had a separate task (cooperatively), then we	None

	shared the results and combined it to finalize the work (collaboratively)	
--	---	--

1.2. Technologies Used

Firstly, we have discussed about which technologies/tools we will use to develop the project, and we agreed on the following tools:

- 1) For Webpages design: **HTML, CSS and jQuery**
- 2) For Project Development: **JSP, MySQL and jQuery**
- 3) For implementing real-time shared space: **Firepad and WebRTC**

1.2.1. Firepad:

An open source, real-time, collaborative text editor. The main reason to include such a technique is to enable the collaborative text sharing, to give the users the ability to share texts.

1.2.2. WebRTC:

WebRTC is a free, open project that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities via simple APIs.



2. Project Analysis

Our website is **CSCL (Computer Supported Collaborative Learning)** which is an approach wherein learning takes place via social interaction using a computer or through the Internet. Collaboration comes in the sense that both users can text, chat and interact collaboratively in the same area that is given for them. It is classified in the **same time - different space** matrix, where the users must be available at the same time no matter where they are, i.e. they can be in different locations.

As Tim Koschman 1996 has categorized the CSCL applications, our project support **presentation** by means of Firepad text editor, and provide **communication** using WebRTC.

2.1. System Structure

We only consider about the clients. The clients can register, login and search for partners. Clients can also contact us by sending emails to:

languageexchangeauto@gmail.com

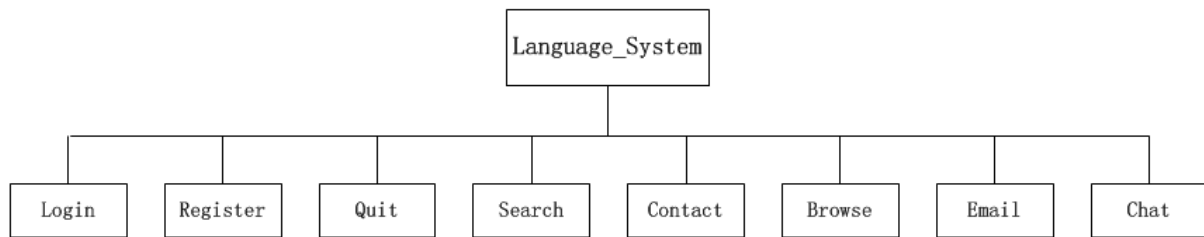


Figure 2.1 : System Actions

Our system uses **JavaEE** classical three-layer architecture: the presentation layer, business logical layer and data access layer(persistence layer).

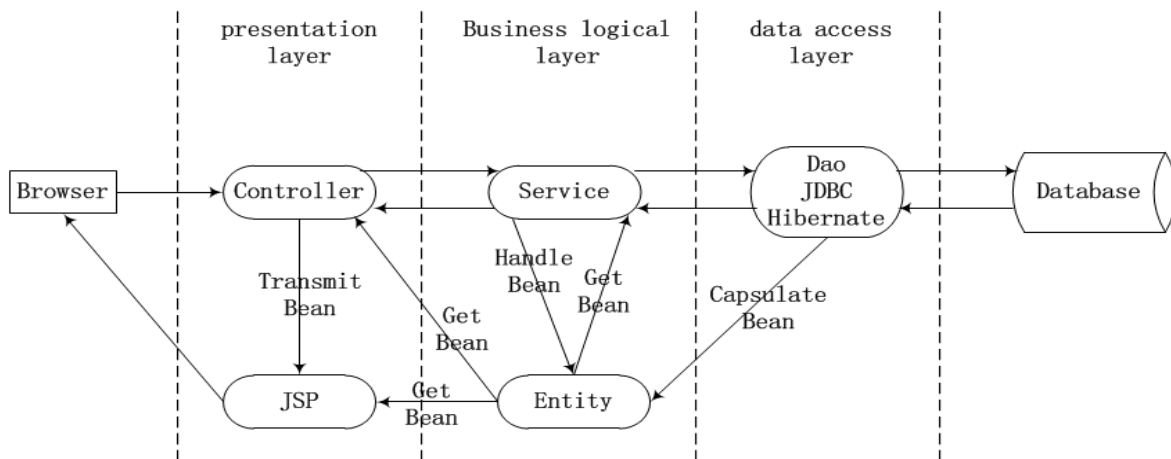


Figure 2.2: Architecture diagram

The presentation layer uses **MVC** (model-view-control) framework. Models are presented by objects (vo) and views are showed as .jsp files. We solve the requests and response by controller (action). The business logical layer is to handle the logical calculations (service). The data access layer was used to visit and operate to the data, put the data into the persistence layer (database) or just get them out. Furthermore, our project uses the Hibernate framework. It can use objects to encapsulate data and reflect objects into database (dao).



Figure 2.3: Example of a Layered Model

2.2. Database Tables

1. Contacts table is used to store contacts messages which is controlled by the button "submit request".

Column	Type
◇ contact_id	int(11)
◇ user_name	varchar(255)
◇ email	varchar(255)
◇ message	varchar(255)

2. Users table is used to store the user data who registered, and is used for searching function.

Column	Type
◇ user_id	int(11)
◇ user_password	varchar(45)
◇ email	varchar(45)
◇ online	int(11)
◇ nation	varchar(45)
◇ user_name	varchar(45)
◇ phone	varchar(45)
◇ city	varchar(45)
◇ country	varchar(255)
◇ known_language	varchar(255)
◇ desired_language	varchar(255)
◇ birth_day	varchar(255)
◇ birth_month	varchar(255)
◇ birth_year	varchar(255)
◇ gender	varchar(255)

3. Learner log table is used to store the shared room for the learning invitations.

Column	Type
◇ id	int(11)
◇ inviter_id	int(11)
◇ guest_id	int(11)
◇ room	varchar(45)

3. Project Development

3.1. Interaction Models

Our project follows the **Client/Server architecture** in most actions, except the voice/video chat phase, which performed as **Peer-to-Peer** communications (thanks to WebRTC). Most of the actions are synchronous with a function call/execution (like login and search), and few are asynchronous (like invite user or check username availability). Here are examples of actions.

Login Action (Synchronous):

- In login.jsp page, click login button and deliver data into action.
- Find method in service.
- Check in Dao, match data with database and response result to service, then to action, then reflects in jsp page.

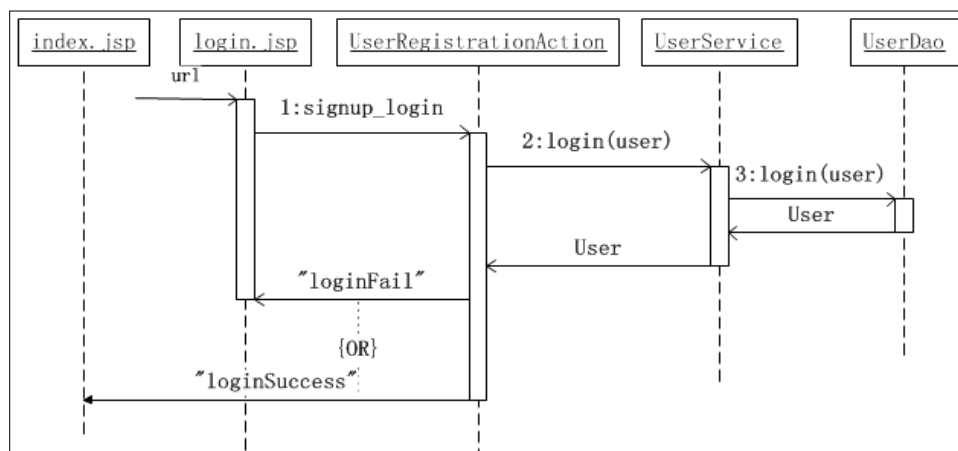


Figure 2.6: Communication Mechanism for Login

Search Action (Synchronous):

- In index.jsp choose languages (desired and native) and click button "get my partner", which will send data to action.
- If the user not exist, return to login.jsp.
- If the user exist, the action call a method findByLanguageLearn(), transfer data to UserService in order to get users match the opposite desired/native language.
- Check in Dao and match data in database, then return result to service, then action, and last to search.jsp which will show the result.

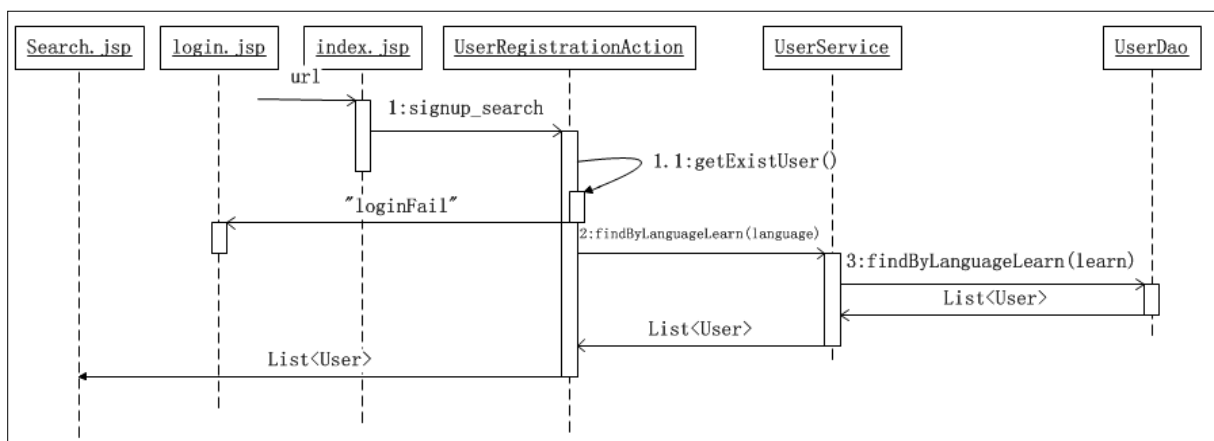


Figure 2.7: Communication Mechanism for Search

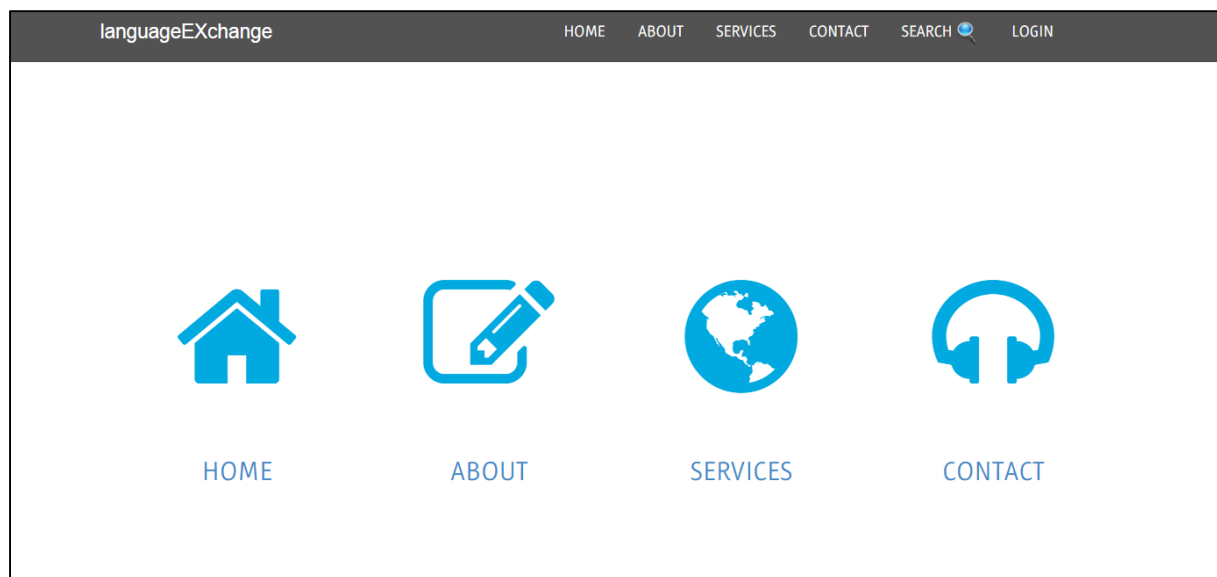
Invite Action (Asynchronous):

This is the interesting action of our project, which is the responsible for invitation of user to join learning and chatting:

- From search.jsp, the user choose one of the users appear as a result from search.
- When click invite, it redirects to the room.jsp which the shared room started with firepad text editor and waiting for the other user to come.
- Once room.jsp opened, invitation entry send to database (table learnlog)
- Once the other user login, check invitation request fired and a notification appeared if there's an invitation, including the shared room url.

3.2. Website Screenshots

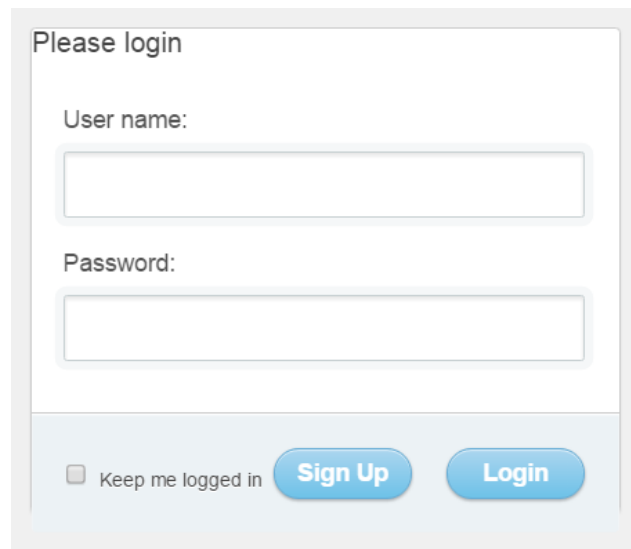
The home page with the menu: home, about us, contact us, search and login. Each of them will result to a specific section in the page. It is as follows.



The user must register in order to be able to use the services offered by the website, the registration form is as follows:

The image displays a 'Registration' form with a light grey background. The form is divided into three main sections: 'Account Details', 'Personal Details', and 'Further Information'. The 'Account Details' section includes fields for 'Email *', 'Repeat email *', 'Password*', and 'Repeat Password*', with a note '* obligatory fields' next to the password fields. The 'Personal Details' section includes fields for 'Name *', 'Phone *', 'City *', 'Country *' (a dropdown menu), 'Known Language' (a dropdown menu), and 'Desired Language' (a dropdown menu). The 'Further Information' section includes 'Gender *' with radio buttons for 'Male' and 'Female', 'Birthdate *' with a date picker showing '01' and 'January' and a text field for the year 'e.g 1976', and 'Nationality *' (a dropdown menu). At the bottom left, there is a 'Terms and Mailing' section with three checkboxes: '* I accept the Terms and Conditions', 'I want to receive personalized offers by your site', and 'Allow partners to send me personalized offers and related services'. At the bottom right, there are two buttons: 'Register' and 'Back'.

As soon as the user have registered an account, he can log in easily into the website from the following form:



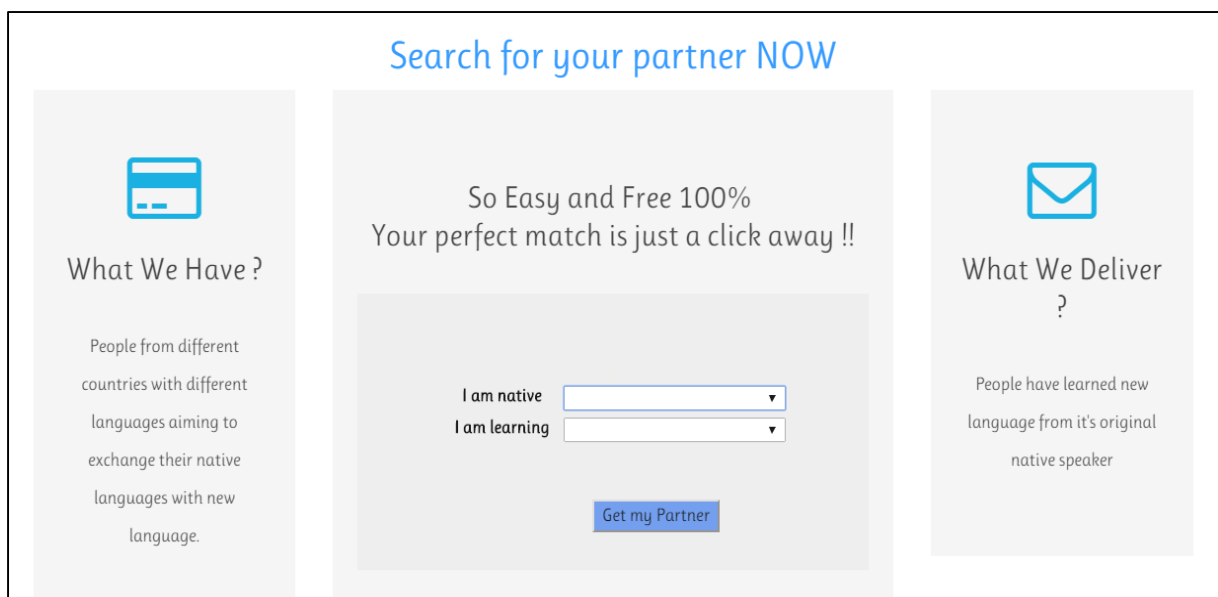
Please login

User name:


Password:

☐ Keep me logged in [Sign Up](#) [Login](#)

The user can now search for his language partner by choosing search icon in the home page. This following section of the page will appear in which it asks the user to choose his native language as well as the desired language, and press Get my Partner button:



Search for your partner NOW



What We Have?


People from different countries with different languages aiming to exchange their native languages with new language.

So Easy and Free 100%
Your perfect match is just a click away !!

I am native

I am learning

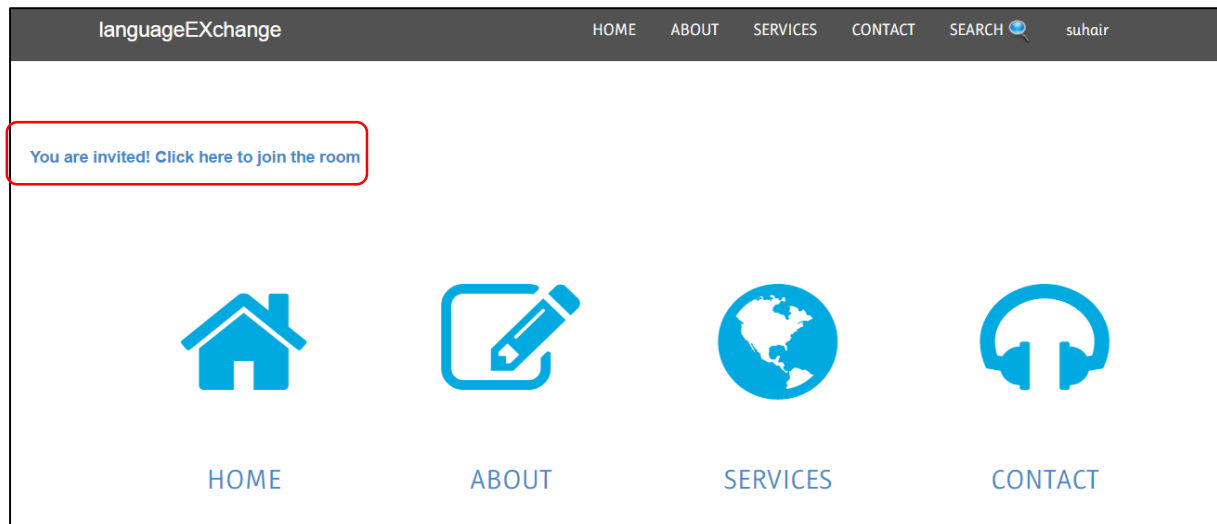
[Get my Partner](#)



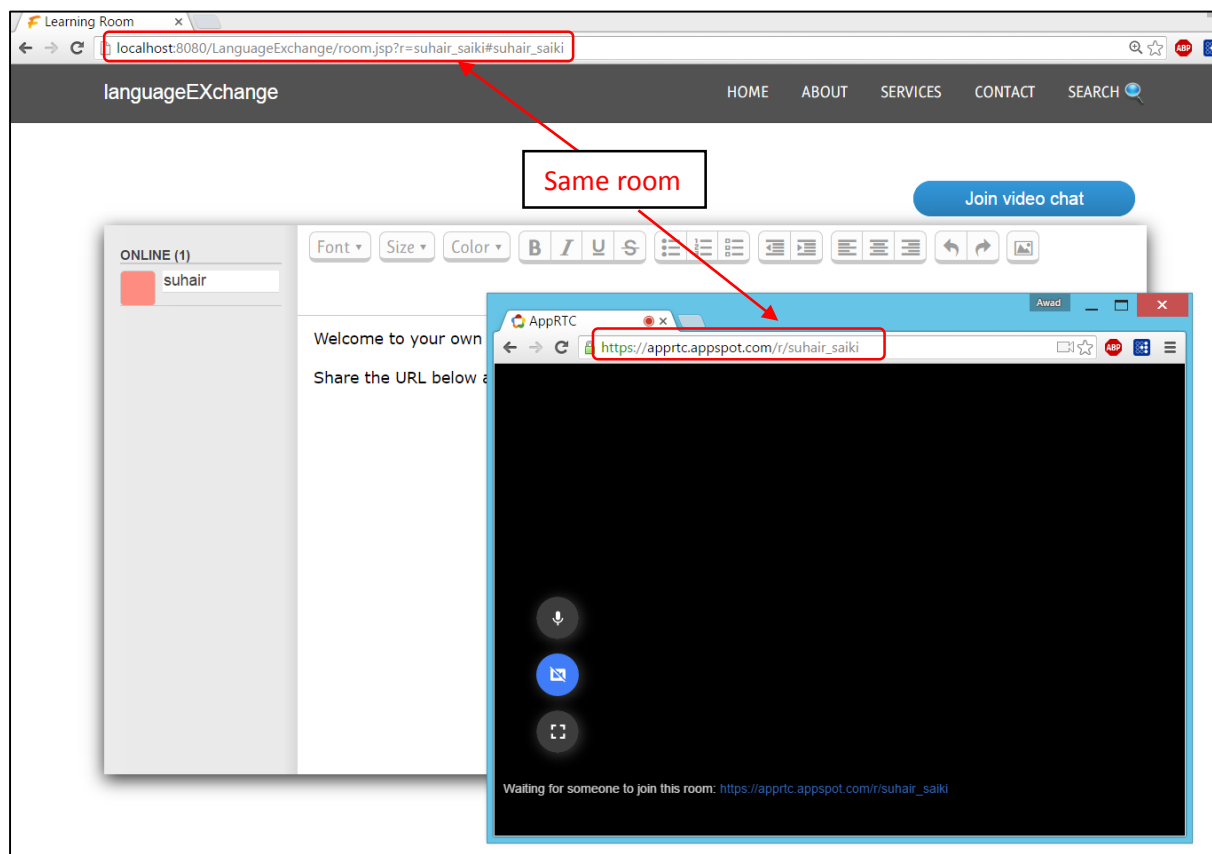
What We Deliver?

People have learned new language from it's original native speaker

If any user gets an invitation to start a chat, a notification is sent asking him to click the link that will allows him to join the chat room with his partner. The notification is as follows:



When the user clicks on the invitation link, he will be directed to the same chat room of the inviter, where both users can text messages and/or use video chat.



For a live demo, [click here](#) to watch a YouTube video.

4. Summary and Limitations

To sum up, this project implements an idea demonstrating the CSCL, allowing users to collaboratively interacting and exchanging languages knowledge. The idea mainly based on Firepad and WebRTC to share an environment and actively communicate.

There are some limitations due to the complexity of the technologies, such as implementing a stand-alone WebRTC communication, which needs instantiating an SIP server (Session Initiation Protocol) in order to generate keys needed for each individual peer-to-peer communication.