This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n: 761999

EasyTV: Easing the access of Europeans with disabilities to converging media and content.

D5.6 – Final report on the set up and implementation of the EasyTV crowdsourcing sign language platform and repository

EasyTV Project

H2020. ICT-19-2017 Media and content convergence. – IA Innovation action.

Grant Agreement n°: 761999

Start date of project: 1 Oct. 2017

Duration: 30 months

Document ref.: Deliverable 5.6

Disclaimer

The EasyTV Consortium consists of the following partners:

	Partner Name	Short name	Country
1	Universidad Politécnica de Madrid	UPM	ES
2	Engineering Ingegneria Informatica S.P.A.	ENG	IT
3	Centre for Research and Technology Hellas/Information Technologies Institute	CERTH	GR
4	Mediavoice SRL	MV	IT
5	Universitat Autònoma Barcelona	UAB	ES
6	Corporació Catalana de Mitjans Audiovisuals SA	CCMA	ES
7	ARX.NET SA	ARX	GR
8	Fundación Confederación Nacional Sordos España para la supresión de barreras de comunicación	FCNSE	ES
9	Unione Italiana dei Ciechi e degli Ipovedenti	UICI	IT

PROGRAMME NAME:	**??**
PROJECT NUMBER:	761999
PROJECT TITLE:	EASYTV
RESPONSIBLE UNIT:	CERTH
INVOLVED UNITS:	CERTH
DOCUMENT NUMBER:	D5.6
DOCUMENT TITLE:	Final report on the set up and implementation of the EasyTV crowdsourcing sign language platform and repository
WORK-PACKAGE:	WP5
DELIVERABLE TYPE:	R
CONTRACTUAL DATE OF DELIVERY:	30-11-2019
LAST UPDATE:	29-11-2019
DISTRIBUTION LEVEL:	PU

Distribution level:

PU = *Public,*

RE = *Restricted to a group of the specified Consortium,*

PP = *Restricted to other program participants (including Commission Services),*

CO = *Confidential, only for members of the LASIE Consortium (including the Commission Services)*

Document History

VERSION	DATE	STATUS	AUTHORS, REVIEWER	DESCRIPTION
V0.1	11/11/2019	Draft	Thanasis Kalvourtzis, Kosmas Dimitropoulos, (CERTH)	T.o.C
V0.2	22/11/2019	Draft	Thanasis Kalvourtzis, Kosmas Dimitropoulos, (CERTH)	Content update
V0.3	25/11/2019	Ready for internal review	Thanasis Kalvourtzis, Kosmas Dimitropoulos, (CERTH)	Final version ready for internal review
V0.4	29/11/2019	Final	Thanasis Kalvourtzis, Kosmas Dimitropoulos, (CERTH)	Final version ready for submission

Definitions, Acronyms and Abbreviations

ACRONYMS / ABBREVIATIONS	DESCRIPTION
API	Application programming interface
CSS	Cascading Style Sheet
FAQ	Frequently Asked Questions
HTML5	HyperText Markup Language
UI	User Interface
HTTP	HyperText Transfer Protocol
Mocap	Motion Capture
CLI	Command-line interface
SPM	Subtitle Production Module

CONTENTS

List of Figures

EXECUTIVE SUMMARY

This deliverable (D5.6) is the final report concerning the set up and implementation of the EasyTV Crowdsourcing Platform which is a web infrastructure for the Sign Language Production and Repository and the Subtitle Production workflow. The goal of this deliverable is to present the work outcomes of Task 5.2, detailing the current state of the platform as of month 26 with reference to its preliminary version as presented in document D5.2. More specifically, the main points of difference with the preliminary version outlined in this deliverable are the following:

- Multi-lingual User-Interface support of 5 languages (Spanish, Catalan, English, Greek, Italian).
- Implementation of more fine-grained user roles and management based on newly defined workflows.
- Extension of the API with public endpoints for the integration with Subtitle Production Module, the Multilingual Ontology Service and the 3D Avatar Service.
- Implementation and integration of new interfaces and frontend apps to support the crowdsourced tasks of the Multilingual Subtitle and Sign Language Production modules.

1. INTRODUCTION

In the context of component-based EasyTV platform, the Crowdsourcing Platform serves as a web infrastructure, allowing users to participate in tasks and contribute with their knowledge-based skills. It should be noted that the current version of the platform's setup supports crowdsourcing tasks with two-fold scope:

- Sign Language tasks to facilitate the creation of a multilingual sign language repository and the enrichment of a certain multilingual ontology.
- Subtitle jobs to facilitate the Multilingual Subtitle Production by integrating them with the collaboration of the broadcaster's part of the system.

Throughout this document, the methodologies, tools and workflows designed and developed for the crowdsourcing module during the second year of the EasyTV project are described. The objective is to present the current deployment of the Crowdsourcing Platform. This deliverable is the work outcome of the continuous development and refinement of the EasyTV crowdsourcing module that is presented in deliverable D5.2 "Mid-term report on the set up and implementation of the EasyTV crowdsourcing sign language platform and repository" [9].

The present document is organized as follows:

Chapter 2 provides an overview of the crowdsourcing concept, describes the way other systems took advantage of it and how the EasyTV platform aims to leverage it.

Chapter 3 describes the application's architecture and provides details about the software components, the user roles and requirements and the software stack.

Chapter 4 focuses on the technical description of the software stack and supportive services that realize the current deployment of the Crowdsourcing Platform.

Chapter 5 provides a detailed account of the development and outcome of the platform's integration with other EasyTV components and services.

Chapter 6 consists of demonstrations of the user interfaces and outlines how they relate to the crowdsourcing workflows.

Chapter 7 briefly summarizes the work presented in this document and discusses about the forthcoming steps related to the platform's utilization and exploitation.

2. CROWDSOURCING SYSTEMS OVERVIEW

In this section, we initially review existing crowdsourcing frameworks and present the definitions of crowdsourcing terminology used in the rest of the document. Subsequently, we present the general context of the Sign Language crowdsourcing platform and its interconnections with the other modules/components of the EasyTV system.

2.1. Definitions

To facilitate reading, in this sub-section, we provide the definitions of some of the most typical terms used in crowdsourcing systems.

***Task*:** A number of well-defined steps to be performed as constituents of a project.

Project: A collection of tasks.

***Moderator*:** The creator and manager of tasks.

***Worker*:** A simple user of the platform. It is assumed that a user contributes to projects by employing his/her knowledge and skills.

***Review*:** Content is open to review according to the policies of the corresponding project.

***Validation*:** Users with specific roles provide the validation of submitted data. These users are usually expert at manipulating and handling these particular types of data.

2.2. Examples of crowdsourcing applications

Crowdsourcing is an online, distributed problem-solving and production model that has emerged in recent years [1]. According to Howe [2], who first introduced this term, crowdsourcing is a sourcing model, in which organizations can develop applications that employ advanced internet technologies in order to harness the efforts and resources of a group of people, i.e. crowd[1] to perform specific tasks [3]. In contrast to traditional recruiting processes, where dedicated employees are selected and assigned to tasks by an employer, in crowdsourcing, the employer submits the task as an open call to a large anonymous crowd of workers. The workers can then freely decide which available task they want to work on [4].There are several advantages in using crowdsourcing models, such as improved costs, speed, quality, flexibility, scalability and diversity [5]. However, there are also significant challenges that arise by employing crowdsourcing systems. The most important of these challenges can be categorized as task related, i.e. design, routing, coordination and aggregation of tasks and crowd-related, i.e. motivation problems, incentive systems and quality control of work results [5]. Crowdsourcing is not bounded by specific constraints on the areas that it can be applied and, as a result, there are several successful implementations of crowdsourcing systems in several areas that are presented in the literature. A few important areas, in which crowdsourcing has been successfully applied are in journalism [6], policy making [7] and health [8].

Typical applications of the crowdsourcing concept are:

- Dataset collection and enhancement
- Dataset annotation
- Dataset categorization
- Sentiment analysis
- Other human-driven tasks

Apart from strictly task-based models, other web-based models of voluntary and collaborative knowledge synthesis and production has been established. Such models are proven successful

[1] https://searchcio.techtarget.com/definition/crowdsourcing

based on simple yet fundamental innovations, i.e., open-source the publishing model and allow the review work after publishing. Examples of such models are the wiki-like platforms that are based on knowledge creation to achieve a great deal in the empowering of the global Information Society. It has been observed that interested users are actively involved in the work of creating and synthesizing content, as well as the meta-work of reviewing, correcting, and organizing.

Finally, we should note the existence of modern web platforms that aim to crowd-source the experience of learning through interactive concepts. Probably the most successful example is that of Memrise [9], an online community that attempts to teach people foreign languages through user-generated content. The users participate in the platform by creating foreign-language lessons, enriching them with media content like memes and translated videos. The platform lets other users to customize the memes, in case they come up with alternative ideas. This open contribution model is able to lead to the creation of user-generated content of high quality. The high quality of the content is achieved by continuous user feedback that constantly improves the content.

2.3. Crowdsourcing application scope in the context of the EasyTV component-based system

The development of the EasyTV Crowdsourcing Platform is based upon a micro-tasking framework that has been specifically designed for providing infrastructure for crowdsourcing applications. It is a web platform that allows the launching and management of projects from the content owners through a web Graphical User Interface (GUI) or an Application Programming Interface (API). Additionally, volunteers-workers can contribute from their web browsers towards the completion of the projects. The EasyTV Crowdsourcing Platform is designed in a way that aims to provide an intuitive User Interface (UI) for the creation, distribution and assessment of crowdsourcing tasks.

More specifically the Crowdsourcing Platform aims to provide a number of functionalities to facilitate the sign language and subtitle production procedures with the collaboration of crowd workers. The platform includes functionalities for task definition, distribution and validation, and allows the creation and management of professional user profiles and the access of users in the sign language and subtitle production procedures through web GUIs. The interconnections of the Crowdsourcing Platform with the other EasyTV modules are depicted in Figure 1.

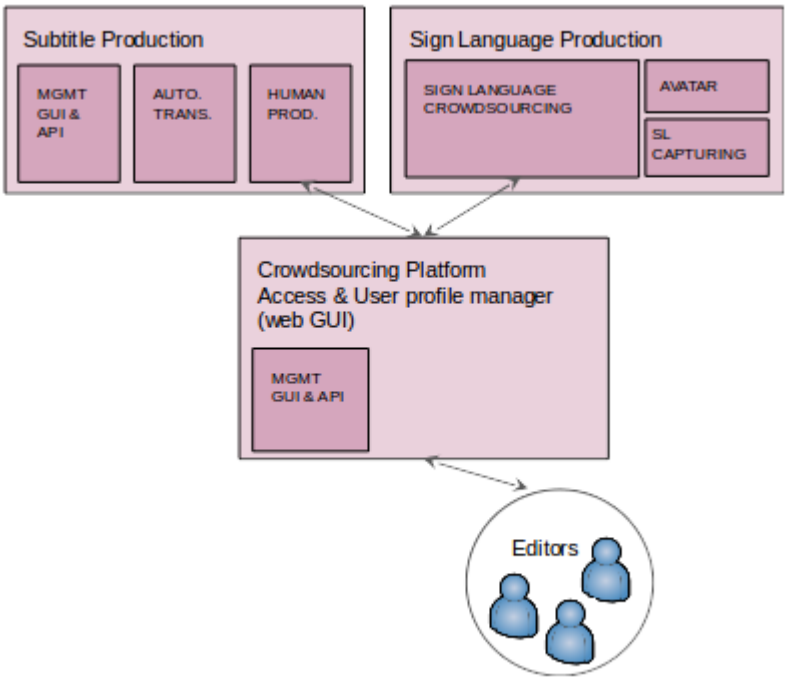


Figure 1: Crowdsourcing component positioning [11]

We should also note that one of the main requirements of the platform is the establishment of a two-way communication with the EasyTV ontology module *(as a service)*, which provides software infrastructure to process annotated sign language videos with Natural Language Processing (NLP) techniques. The ultimate goal is the creation of a multilingual knowledge base through the combination of annotated sign language videos that are created from users and stored in a repository with multilingual ontologies. Signs in different languages are being associated with their meaning in natural language in order to enrich the BabelNet – a multilingual encyclopedic dictionary and semantic network.

3. PLATFORM ARCHITECTURE

This chapter focuses on the description of the architecture of the EasyTV Crowdsourcing Platform. Initially, we define a high-level view of the main components of the platform. Afterwards, the user roles and scenarios in form of flowcharts are presented. These descriptions draw upon work from previous deliverables, in which the system requirements of the EasyTV platform, the user requirements and the EasyTV components and services were defined [11][12][13]. Moreover, in this chapter, we go into providing more details about the updated data models, which form the platform's main database entities.

3.1. Architecture components overview

The analysis of the user and system requirements leads to the creation of the first architecture of the EasyTV Crowdsourcing Platform that is outlined in Figure 2 with the help of a component diagram. The component diagram comprises a static implementation view of the system. It should be noted that these components and interfaces form a group of functionalities and services that is meant as a guideline to support the implementation perspective and does not represent a strict definition of isolated software modules.

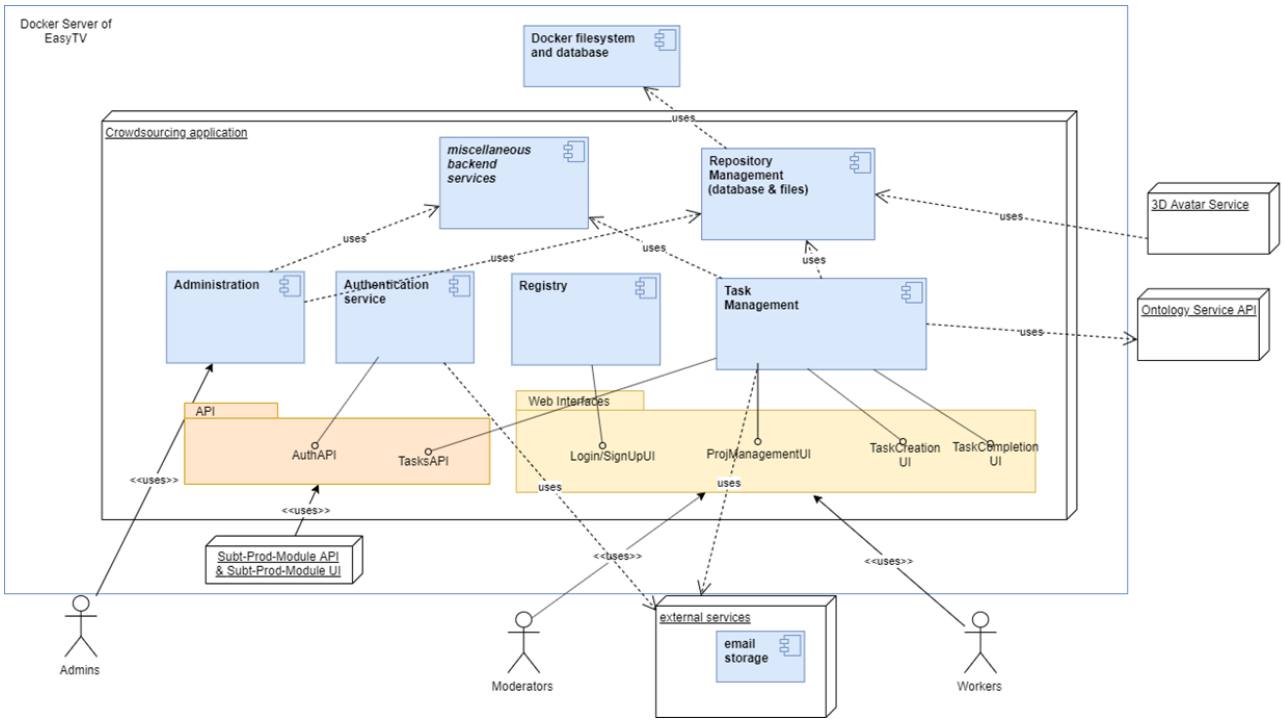


Figure 2: System Component Diagram

Administration component:

This component facilitates the administration functionalities, which are mainly concerned with the technical control and monitoring of the platform. The functionalities of this component are meant to be exposed via web UIs, like dashboard interfaces and settings to the administrators of the EasyTV platform.

Registry component:

The registry component is responsible for handling the registration of users or apps in the EasyTV Crowdsourcing platform. UIs are constructed and deployed for user registration and app registration through token generation and authentication.

Authentication component:

The authentication component has two main purposes: it handles user authentication and enforces the defined access policies for the different types of users. The implementation of this component incorporates well-known standards of user authentication via web token authentication.

Project management component

This component provides a number of functionalities related to project management. These functionalities are provided to end-users through user interfaces (or an API for external modules or services) in order to enable these users to define and manage projects.

Task creator/management component

This component provides a number of functionalities that are related to task creation and completion procedures. Examples of such procedures are content submission, validation of previously submitted content. Different front-end apps have been designed and developed to facilitate task participation, while an API is provided for external apps.

Files management component

The files management component is responsible for handling operations that involve the creation of file streams for the storage and upload of content to the EasyTV Crowdsourcing Platform.

Miscellaneous services component

The miscellaneous services component includes a number of back-end helper services. An example of such services is the aggregation of statistics about task completion rates or user involvement in tasks. These services are not directly exposed through API endpoints.

API component

The API component is responsible for managing the endpoints for the communication of the EasyTV Crowdsourcing Platform with other modules.

External services

External services refer to a group of external components that offer services that are vital for the smooth operation of the platform, such as off-platform communication or data and backup storage requirements.

3.2. Users and scenarios

In the context of a web application, such as a crowdsourcing and data-repository platform, the definition of users, their roles and their functionalities constitute a crucial part of the platform's development. Before any technical implementation is made, the users' roles and functionalities have been under heavy discussion from the EasyTV consortium. Furthermore, the deliverable D1.1 [12] gave us useful insights about the functionalities that users need in order to fulfill their role. Based on the discussions and the user requirements, the following user roles have been proposed and defined:

3.2.1. Administrator

The role of administrator is defined as the technical manager of the platform. In the first place, an administrator should be supported by an appropriate dashboard, which provides an overview of the current status of the platform regarding the statistics of platform's usage and resources. The administrator undertakes responsibilities related to technical aspects of the platform's operation.

For example, an administrator may configure the settings of the EasyTV Crowdsourcing Platform in order to enable the notification of users about upcoming upgrades of the platform.

3.2.2. **Workers**

3.2.2.1 Signer worker

These users constitute the main pool of contributors for the creation of a sign language database. Their participation in the platform is decided according to their language background that is defined by themselves during the registration procedure. The user should initially sign up to the platform by providing a minimum amount of information in order to be able to work in a project listed in the crowdsourcing platform. During the crowdsourced tasks, the platform prompts the user to follow a number of steps in order to fulfill a task by uploading content in the appropriate format. It also gives them rights to edit and review their own submissions or others' according to policies.

3.2.2.2 Subtitle worker

According to the design specifications of the Subtitling Production Tool of the EasyTV platform, the crowdsourcing platform is a constituent part of the pipeline of the *Human subtitling production* module by fulfilling the role of a profile-management portal. For this purpose, the profiles of potential collaborative users are going to be kept and managed through an appropriate interface, which is meant to provide a broadcaster with access granting options. The user/ subtitle worker will come into contact with the content owner and will be tasked with performing translation requests by the content owner.

3.2.3. **Evaluators**

3.2.3.1 Evaluator of Sign Language Tasks

The moderator of sign language tasks (SLTs) constitutes an advanced user of the EasyTV crowdsourcing platform. A moderator is most frequently the user that requests the creation and completion of crowdsourcing-based tasks. Through the platform's infrastructure, this user creates and manages Sign Language projects and is responsible for reviewing the progress of the tasks. The platform offers specific tools and content access for this purpose, like a dashboard.

3.2.3.2 Evaluator of Subtitle Jobs

This evaluator worker of content owner is a type professional user of the EasyTV Crowdsourcing Platform that can be assigned by a broadcaster/content owner in order to be responsible for the assignment of reviewer and professional profiles to the corresponding users.

3.2.4. **User Scenarios**

In this section, we describe user scenarios that are designed and implemented for the EasyTV Crowdsourcing Platform in the framework of the EasyTV project. These scenarios are backed up by flowcharts that visualize the user roles and functionalities. In the following flowcharts, the involved shapes, along with the corresponding semantic information they carry, are presented in Appendix 1. It should be noted that these charts demonstrate a flow of asynchronous events as they involve human initiated actions.

It is a prerequisite that all categories of users have provided a minimum of personal information during the sign-up process. This information is necessary and is taken into account during processes like user-profiling or task-distribution.

Sign Language Task scenario:

A user that has a sign language background can log in the platform as a signer worker. User's UI

displays a panel of active projects and their descriptions according to the logged user's background information. The user selects a project and the system responds with a view that lists the active tasks. The user proceeds to the task completion guided through steps by the GUI. At this stage, the system simply confirms that the submitted files have the correct format and meet the requirements of the task. After the successful completion of the task, the submissions of the user are stored in the system and labeled with status "under review". A moderator, who may be the creator of the project or assigned as reviewer of the project, gets notified about any progress related to tasks of his/her responsibility. In the event that a moderator validates the input of the signer worker for a task, the status of the submission of the task changes as accepted and is included in the repository. Finally, a user subscribed to a project should get automated notifications, when updates occur in the corresponding project.

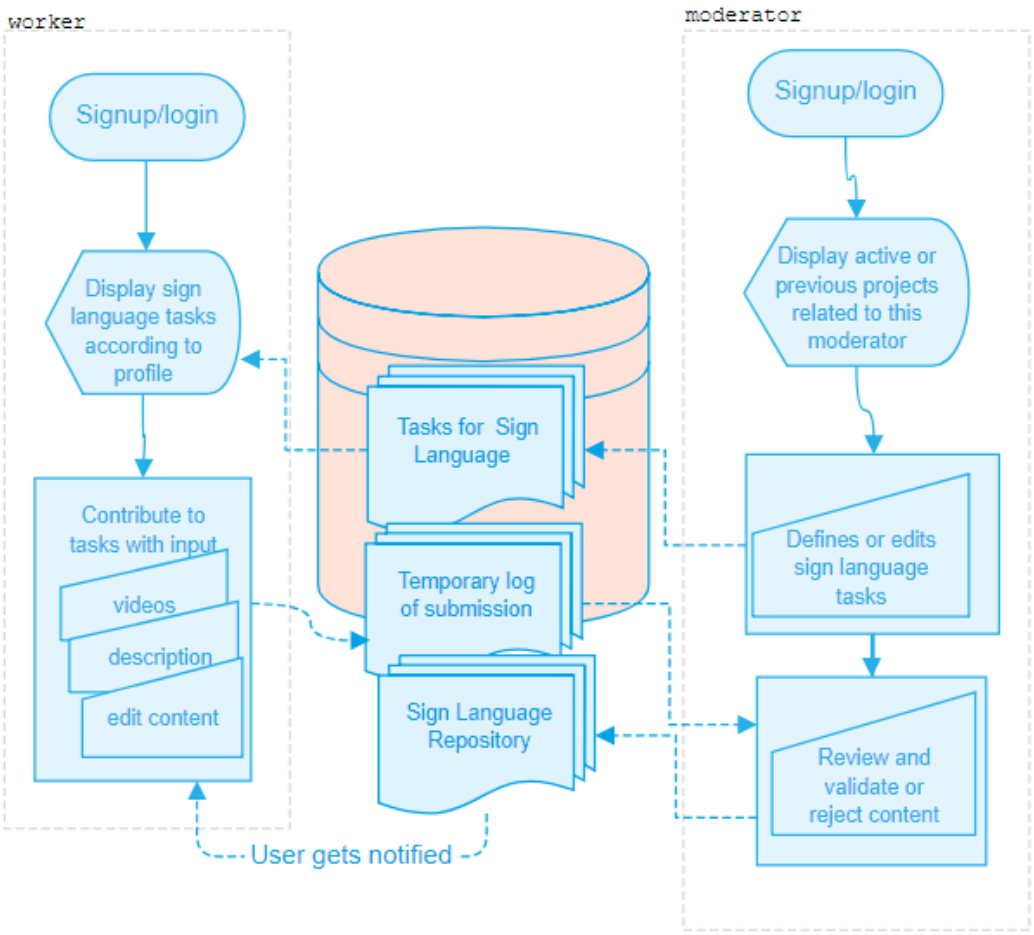


Figure 3: Scenario - Task workflow

Human subtitling module scenario:

Users are registered in the platform as potential collaborators and have also provided information about their language background and their competence. The broadcaster user manages different user's trust levels through a web GUI and also manages content access rights and quality assigning trust levels. Users proceed to contribute to a task and the platform assists them in accordance with the trust level of the author user of the translation. The broadcaster manages access to contents with subtitles in its original language. Once the translated material is ready, the

broadcaster is able to download it. Neither the original nor the translated content is to be made publicly available directly from the EasyTV Crowdsourcing Platform due to rights management. The publication of this content will be performed exclusively by the broadcaster over its own publication platform. Finally, the EasyTV Crowdsourcing Platform provides alert messages to the users, when additional content that requires translation is uploaded and to the broadcaster, when a translation task is finished.

Non-User Scenarios

The following description constitutes a non-user scenario, in the sense that it describes interaction between EasyTV software modules. The flow presented in this chart is related to the scope and functionality of the EasyTV Ontology module, as described in D3.2.1 [14]. In that deliverable, the objective of the enrichment of the multilingual ontology is described in relation to the EasyTV Sign Language repository.

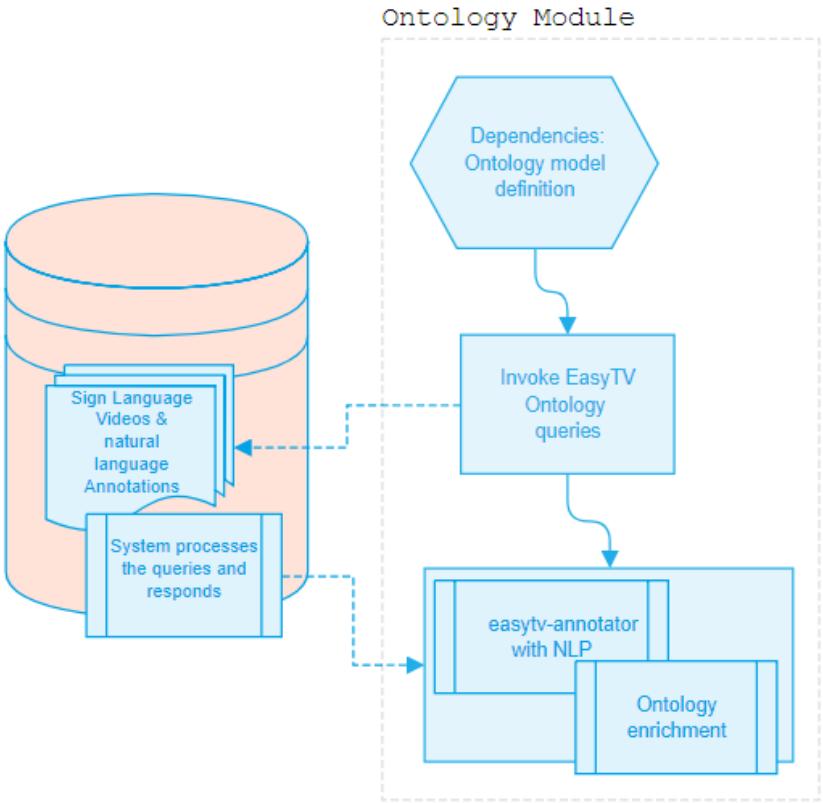


Figure **4**: Scenario – Crowdsourcing communicates with Ontology web service

3.3. Data models

In this section, we describe with natural language the data models that are employed from the EasyTV crowdsourcing platform. The scope of these data models is to define the main database entities that will support the creation, management and distribution of the crowdsourcing tasks. The current description of the data models is an updated version that takes into account the integration development that has been made with other collaborating EasyTV services.

A user entity consists of the following entities:

- Full name

- Email address
- User's languages
- User's notification settings
- Other languages that the user is proficient at
- Statistics of user's platform activity projects (number of projects, completion status, etc.)

A project entity consists of the following data:

- The name of the project
- An informative description
- A project may include many tasks
- A project may have many participants
- A project may have many subscribers

A task entity is comprised of the following entities:

- The name of the task
- The type of the task (i.e., sign language or subtitle production)
- An informative description
- A task has submissions
- A task may have a required confidence level

A submission entity consists of the following information:

- It may point to physical files, stored in the EasyTV repository
- It refers to one task
- It is submitted by a user
- It may be reviewed by many users

An organization entity consists of the following information:

- It is administered by certain admin users
- It has settings for email notifications
- It has settings for register-requirements

These data models are of paramount importance for the smooth operation of a crowdsourcing platform as they not only define important information that flows among the components of a crowdsourcing platform, but also define relationships among users, components, entities and functionalities of the platform.

4. THE DEVELOPMENT STACK

4.1. Frameworks

The survey for potential development frameworks was performed according to a number of criteria related to the challenges of the growing complexity that exist in modern web development. For this purpose popular community-backed projects which are built atop of other open-source and well-tested modules. These frameworks incorporate and encourage the use of standardized software methodologies and depend on a number of community-backed tools that are continuously updated. For our purpose, we consider that an interactive platform even in prototype stage can benefit from such practices in a number of aspects like user experience, security updates etc.

In our case it has been important for the chosen framework to be compatible with the concept of Hybrid Web Application – an application that combines a JSON API with server-rendered views, that is, in addition to an API, this type of application can serve dynamic HTML pages. Hybrid web apps can optionally employ a front end framework but not necessarily. For the scope of the EasyTV crowdsourcing platform, the chosen framework is expected to not only serve web interfaces for the human end users, but also expose APIs to facilitate the communication with other components of the EasyTV platform.

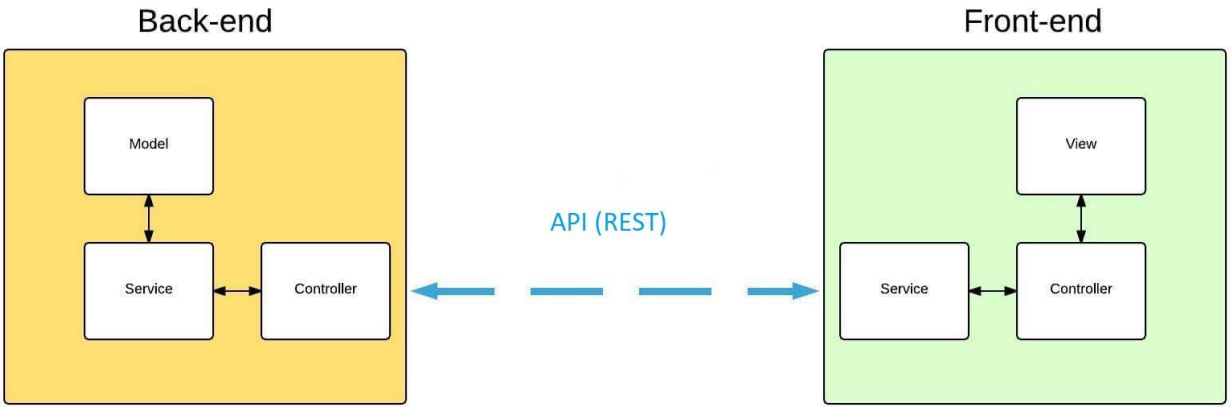


Figure 5. High level overview of Back-end – API – Front-end concepts

The block diagram in Figure 5 depicts the concept of *Back-end and Front-end decoupling* and provides a high-level abstraction about how data flows between the front-end and back-end.

In order to meet our needs the Sails.js framework – a popular and actively developed micro-framework built on Node.js and its standard server, Express.js – was chosen because it offers the following advantages:

- is written entirely on the Javascript programming language this means that time spent on context-shifting is minimized and it is significantly easier for the developer to maintain the work's codebase
- is supported by CLI tools for code scaffolding
- follows the MVC pattern
- supports multiple databases with abstraction layer
- its data models get automatically a REST-API with CRUD operation
- provides basic security and role-based access control by default
- is agnostic to front end choices
- is generally extendable and pluggable structure

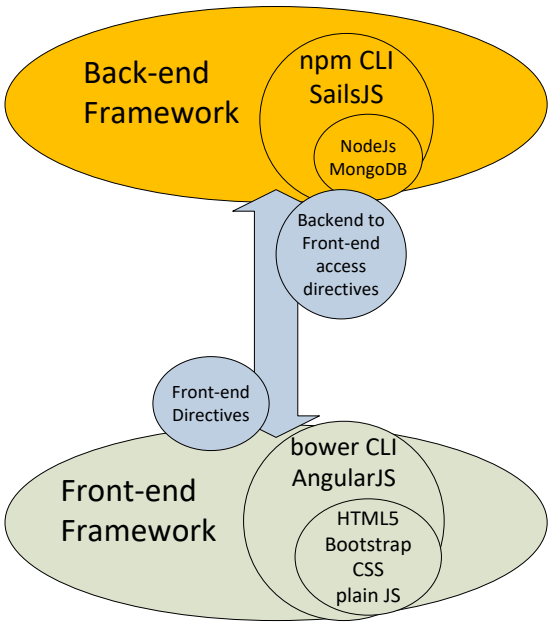


Figure 6. Development stack along development tools

In Figure 6 we present an overview of the main tools involved in platform's development. These separate frameworks apart from including automation tools for code scaffolding, they also encourage a number of guidelines for the development process.

As a front-end design solution the platform is based on Bootstrap the most popular framework bootstrap. This framework is installed in the form two development libraries, CSS and JavaScript, and was open-sourced by Twitter in 2011. Bootstrap stands as a very popular choice for prototyping and further development as it offers a big collection of ready-to-use front-end components which are responsive by default. During last years the most recent versions of the library offer mobile-ready components respecting the total share of mobile browsing. The components are used in a 'grid system' and importantly they are easily customizable through CSS programming.

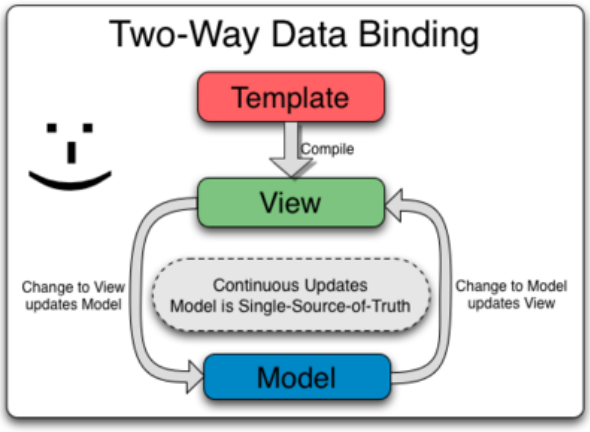


Figure 7. The concept of two-way data binding in Angular.js

Moreover, in order to meet the need for dynamic web applications, Angular.js[2] framework was included in the front-end development. This framework provides a toolset that its capabilities can be leveraged to create web applications with interactive UIs - dynamic views - instead of static

[2] https://angularjs.org

server-rendered HTML. This capability is mainly possible through the Angular.js realization of the concept of two-way data binding which allows the user-interfaces' components to be continuously synchronized with the updates of the actual data model.

It is important to note that through Angular.js community the developer has access to an ecosystem of front-end reusable and extensible components that are written in plain JavaScript and are supported by all modern browsers.

4.2. Mongo database and cloud

MongoDB[3] is a No-SQL database where data are stored in JSON-like hierarchical document with key-value pair content. The employment of such a database for this phase of the development is justified by a range of reasons: its No-SQL structure and dynamic schema fits for fast prototyping and allows keeping the design simple as the developer can modify the document's schema according to the data and not the data according a strict schema. Furthermore, a convenient trait related to our previous choice, is the fact that MongoDB engine is well supported within Sails.js framework database adapter. It provides an abstraction layer on top of the underlying database and this allows the development codebase to remain focused on the application's logic and avoid writing of complex query language.

4.2.1. Utilization of Databases-as-a-Service

Considering the deployment and lifecycle of a web application, it is worth mentioning that an important feature of MongoDB is that it can be conveniently migrated to cloud infrastructure as various cloud services exist that support it[4]. During our development iterations this has proven essential part of the integration as it allowed depending on easily deployable and accessible temporary development databases by free-tier cloud-based services.

4.3. Application's structure overview

4.3.1. Outlining back end structure

In the following tables, the functionality of our back-end by presenting the basic endpoints grouped with under the main resource is summarized. A summary of the codebase structure and an outline of how various functionalities are grouped into the platform's structure are also presented.

The selected URIs should follow a predictable, hierarchical structure to enhance understandability and, therefore, usability: predictable in the sense that they are consistent, hierarchical in the sense that data has structure—relationships.

Basic resource	**user/**	
Methods	GET	user/home
	GET	user/index
	GET	user/show?{id}
	GET	user/register
	PUT	user/signup?body:{name,email,language}
	POST	user/updateinfo?body:{name,email,language}

[3] https://www.mongodb.com/cloud/stitch/

[4] https://www.mongodb.com

	POST user/notify?{id}
Description	Methods for managing user profiles

4.3.2. Defining access policies

The following description about access policy implementation is based on the corresponding policy system employed by the development stack. The JSON-like script of Figure 8 presents an excerpt of access policies in the format that should be defined in the back-end Sails.js framework. The names of the policies for each endpoint represent middleware functions in the sense that they form a layer of functions that is always executed before the actual controller is activated. These functions are not only restricted to provide access-related logic, an example is the 'localize' middleware function that monitors any change in the user's preferred language and applies it in the content of next response.

```
project: {
  'index': ['sessionAuth', 'moderator','localize'],
  'create': ['sessionAuth', 'moderator','localize'],
  'show': ['sessionAuth', 'moderator','localize'],
  'edit': ['sessionAuth', 'moderator', 'isModerator','localize'],
  'addmoderator': ['sessionAuth','moderator', 'isModerator', 'localize'],
  'destroy': ['sessionAuth', 'admin', 'localize'],
  'progress': ['sessionAuth', 'moderator','localize'],
  'new': ['sessionAuth', 'moderator','localize'],
},

settings: {
  '*': ['sessionAuth', 'admin', 'localize']
},
```

Figure 8. Policies and a corresponding policy controller defined in Sails.js

The flexibility in the definition of the middleware functions is depicted in Figure 9 where the ones presented are applied when a user requests access to edit a project entity's information. The first one on the left simply performs check on the access rights of the logged user which was specified upon his sign-in request. The second function on the right is querying the 'Project' model in the database in order to confirm that the moderator belongs to the assigned moderators of the requested project.

```
module.exports = function(req, res, next) {

  // User is allowed, proceed to controller
  if (req.session.User && req.session.User.access === "moderator")
    return next();
  else if (req.session.User && req.session.User.access === "admin")
    return next();

  // User is not allowed
  else {
    FlashService.error(req, 'You must be a moderator.');

    req.session.returnTo = req.path;
    res.redirect('/');
    return;
  }
};
```

```
module.exports = function(req, res, next) {

  projId = req.param('id')
  // Moderator is owner, proceed to controller
  if (req.session.User && req.session.User.access === "moderator")
  {
    Project.findOne(projId)
    .populate('moderators')
    .exec( function(err, project) {
      if (req.session.User.id in project.moderators)
      return next();
    });
  }
  else if (req.session.User && req.session.User.access === "admin")
    return next();

    FlashService.error(req, 'You must belong to the moderators');

    req.session.returnTo = req.path;
    res.redirect('/');
    return;
  }
};
```

Figure 9. Middleware functions

4.4. File repository

A major competence of the crowdsourcing platform is the handling, managing and storing of the crowdsourcing tasks user-generated input. These data include many files that are saved in the file-system of the production server in a way that allow their association with their corresponding tasks. In the database level this files-tasks structure is also tagged with appropriate concepts in order for them to be accessed consistently through API requests.

4.5. Testing

The application of software testing tools is of high importance especially when a project is under continuous development and is being integrated with other modules.

The Sails.js framework neither ships with a testing framework[5] nor explicitly suggests one. For the purposes of this development phase, we follow the community-proposed `Mocha` testing framework which targets the JavaScript language. The developer defines unit-test scenarios written in .js scripts which may include API requests or database queries in order to test the expected answers and the data integrity after the executed back-end operations. Depending on testing frameworks like Mocha is essential for handling the asynchronous nature of the back-end responses. For testing HTTP servers the framework can be integrated with tools like `supertest`[6].

```
var request = require('supertest');
should = require('should');

describe('UserController', function() {

  var Nusers;
  describe('#find_N_users()', function()
  {
    it('DB find function', function(done) {
      User.find().exec(function(err, users) {
        Nusers = users.length;
          done();
        })
    });
  });
  describe('#register user',function ()
  {
      var test_user = {
        firstName:'testUser',
        lastName:'testUser',
        email: "testmail@mail.com",
        email2: "testmail@mail.com",
        password: "testest",
        confirmation: "testest",
        userLanguage: 'English'
      };
      it('registers a new user',function (done) {
          request(sails.hooks.http.app)
            .post('/user/signup')
            .send(test_user)
            .expect(302, done);
      });
  });
  describe('#find_N+1_users()', function()
  {
    it('should equal N+1 users', function(done) {
      User.find().exec(function(err, users) {
          users.length.should.be.eql(Nusers+1);
          done();
        })
    });
  });

  describe('#login()', function() {
    it('should redirect to /', function(done) {
      request(sails.hooks.http.app)
        .post('/session/create')
        .send({
          username: "testmail@mail.com",
          password: 'testest'
        })
        .expect(302, done);
    });
  });
});

```

Figure 10. Mocha testing

[5] https://sailsjs.com/documentation/concepts/testing

[6] https://github.com/visionmedia/supertest

A typical testing of the system includes the definition of an initialization procedure (launch the server and connect to database) and the execution of a number of HTTP requests that test the responses after each single step and inspect the integrity of the database. In Figure 10, a test file containing a flow of steps, which test both the controllers and the database's state is presented. The steps are provided with a short description in natural language. Initially the database is queried and a variable is set. The `supertest` library is used to execute a POST request with the appropriate arguments and the database is queried again to confirm the operation.

.

5. INTEGRATION WITH OTHER MODULES

In general the crowdsourcing platform offers multiple functionalities to the end users that depend on other EasyTV components and services. In this chapter we begin by focusing on the Sign Language Production part of the platform, which supports crowdsourcing tasks with the purpose of generating content that is used for the enrichment of the multilingual ontology and for the creation and visualization of the users' signs. The full realization of the production pipeline is based on the integration of various EasyTV components, which take part in the sign language production interacting as presented in **Error! Reference source not found.**.

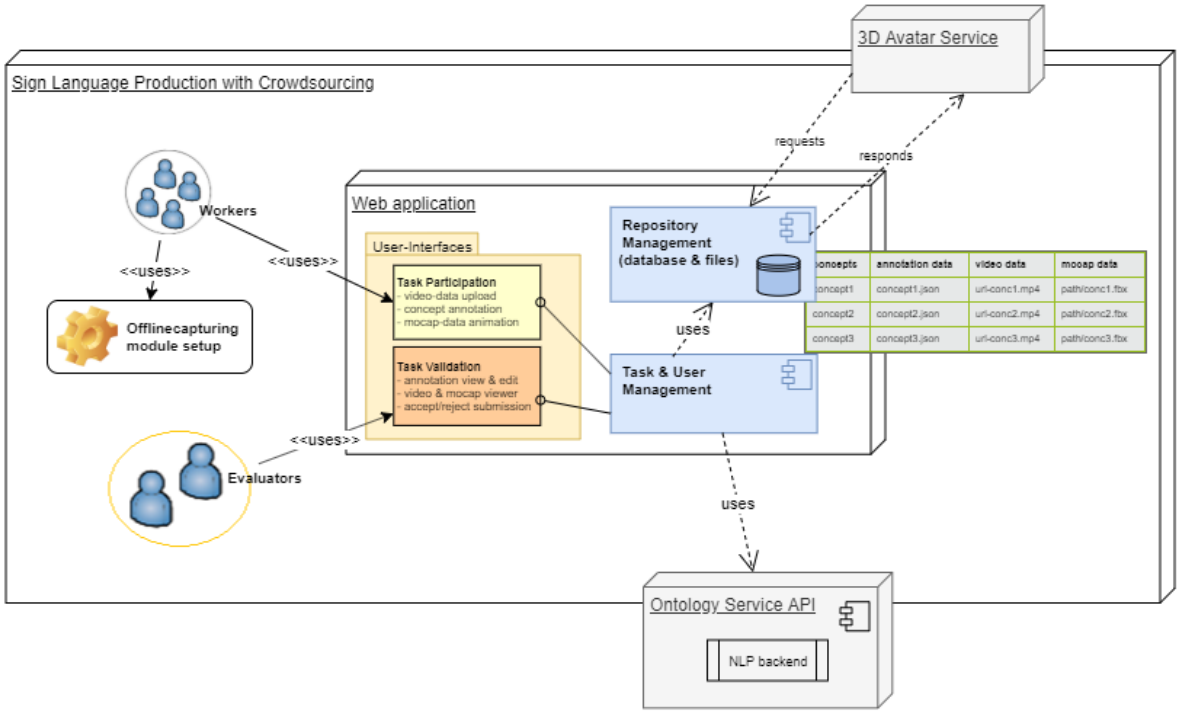


Figure 11. Crowdsourcing positioned in the integrated system

5.1. EasyTV capturing module

The utilization of the latest version of the **EasyTV capturing module**, as presented in D3.7 [15], is a prerequisite for the proper completion of the Sign Language Tasks. Its product is a set of files containing both motion-capture 3D data and RGB video-files of the recorded signs. These files enable the crowd-workers to participate into the crowdsourcing platform by completing a requested task, and the platform stores these files into repositories temporary in order to be validated and eventually become available for other modules or end-users. Prior to the submission's validation, the Crowdsourcing platform enables the Evaluator users to visualize the recorded RGB video along with the user's annotations and also to animate the AI-generated motion data. This poses an integration requirement as the files should be properly exported in a playable FBX format by the pipeline of the capturing module. The inspection and validation step implies that the platform has already accepted the format of the submitted files during the task completion steps.

5.2. Integration with the Sign Language Ontology Web Service

The Sign Language repository that is being created by the crowdsourcing tasks is connected with the multilingual Sign Language Ontology web service infrastructure. To this end, the backend of the platform has been extended to communicate with the Ontology Module service and exchange certain user-generated data to the ontology's online database.

According to the discussion in [14] and [15], in the preliminary version of the modules a JSON file

format is being used with a predefined structure. The EasyTV annotator library (Figure 12) receives from the crowdsourcing platform a POST request with a certain JSON format, containing the temporary stored user-generated submissions. These include the URL of Sign Language video along with its natural language transcription sentence and other video-related metadata.

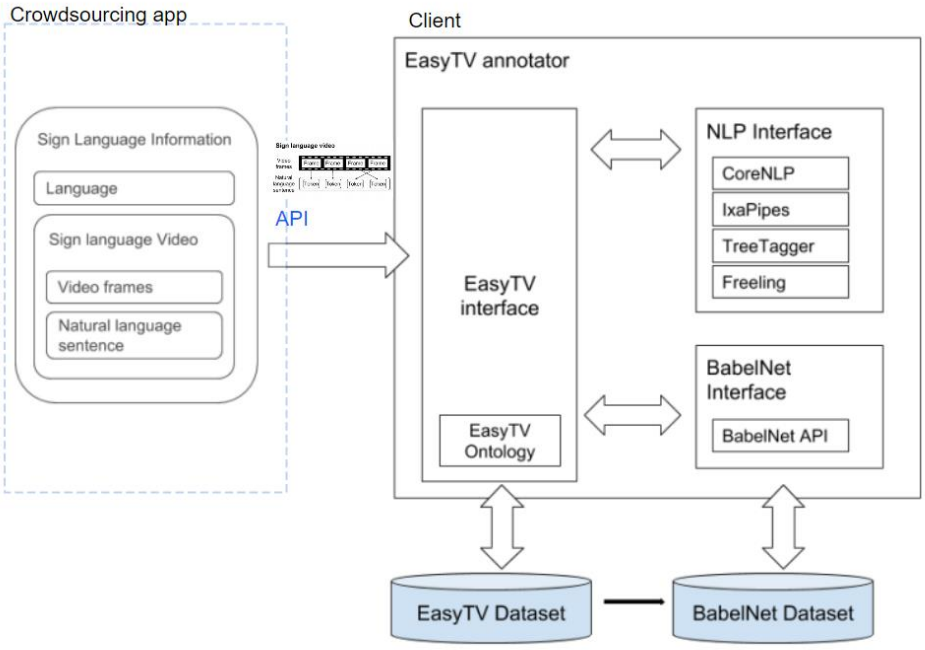


Figure 12. Crowdsourcing's input to Multilingual Ontology Service

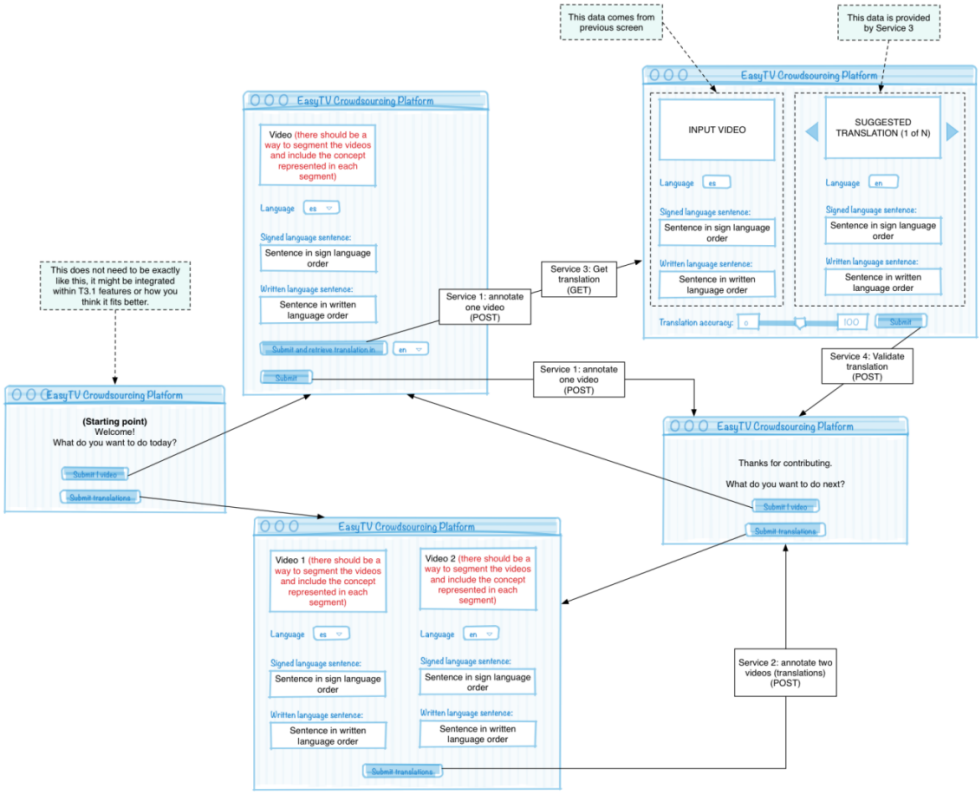


Figure 13. Wireframes by OEG[7] designing the integration with the Multilingual Ontology Service

[7] http://www.oeg-upm.net/

For this purpose, a number of API endpoints have been exposed in order for the Multilingual Ontology to be accessible and connected to the crowdsourcing Sign Language tasks. The platform includes a dedicated set of methods that processes user's inputs and communicate them to the Multilingual Ontology Web Service.

In Figure 13 a wireframe designed by the Ontology Engineering Group during the preliminary design of the integration of the crowdsourcing tasks' workflow with the multilingual Ontology web service API is depicted. The functionalities offered through these interfaces are supported in the Crowdsourcing frontend stack as lightweight single page applications. These frontend apps were built on top of an open-source video annotation tool[8] which provides for an extendable and adaptable solution.

5.3. Signer 3D-Avatar Service

The design and development work related to the creation of the realistic Sign Language avatar was initially described in the deliverable D2.1 [16] that lays the details about the tools and frameworks which are currently employed. The avatar development pipeline depends on the motion data acquired through the completion of the crowdsourcing tasks. A target of the EasyTV platform is to make these avatars available providing an EasyTV signer 3D-Avatar service to the end-user.

During the preliminary development phase, the selected platform of the signer avatar creation team is Unity [17], a cross-platform development environment. Unity powers the main program for editing the avatar, connecting with the Crowdsourcing Platform and translating FBX motion files into realistic graphics. The EasyTV 3D-avatar module acts as a web-client application (Unity engine supports platforms Android and WebGL) that is granted access to the crowdsourcing platform from which it can request the required motion-capture files using standard HTTP protocol. The querying of the Sign Language Repository is performed through the Crowdsourcing's API endpoints.

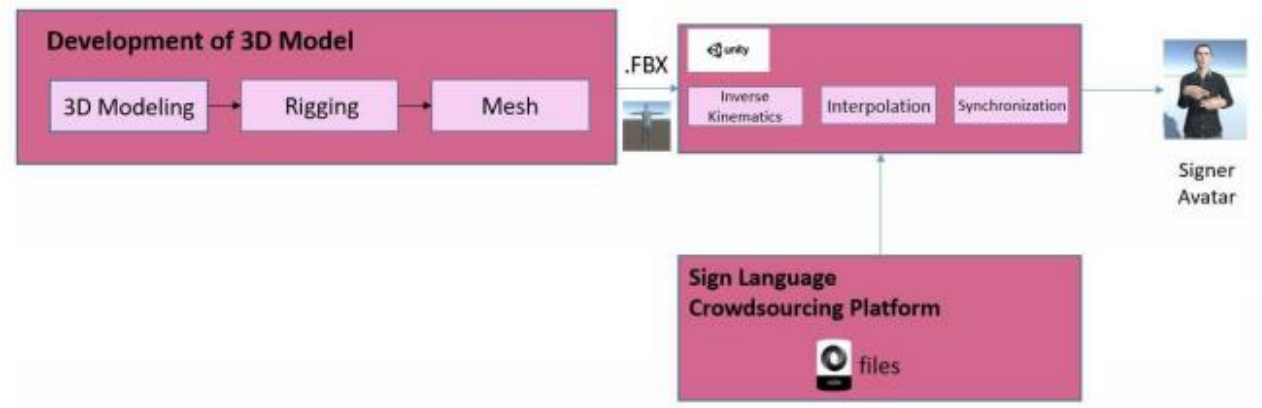


Figure 14. Overview of the relation of the 3D-Avatar service with Crowdsourcing platform

It should be noted that an important functional detail not demonstrated in the diagram of Figure 14 is the inclusion of users' time segmentation of the input video, collected during the crowdsourcing tasks. As explained in section 5.2 the video-annotation process requests from the signer crowd-worker to access an online tool and annotate the submitted RGB video with text in natural language and sign language order. This time-segmented information can also be connected to the motion-data files as metadata, in order to be used by the playback service as a subtitling asset.

[8] https://github.com/contently/videojs-annotation-comments

5.4. Multilingual Subtitle Production Module with Crowdsourcing

5.4.1. Concept overview

This section focuses on describing the integration design and development that was undertaken to implement the concept of the *Human Subtitle Production Module based on Crowdsourcing tasks.* In the context of the EasyTV project, the main idea of the subtitle production pipeline was to build upon the Crowdsourcing infrastructure as defined in D5.2 and elaborate new workflows that can take advantage of crowdsourced tasks to complete *subtitle jobs* by accessing content published and managed by the collaborative broadcaster.

The crowdsourcing user roles and their main intended activities, localized for the Subtitle Production pipeline's context, are summarized as follows:

- **Reviewer**: this type of user is a crowd-worker that is enabled to choose and get assigned an available subtitle job on which to work. The platform redirects the Reviewers to an appropriate reviewer-tool, accessed as Broadcaster's application, and enables them to assess the accuracy of the automatically generated subtitles and fix the potential errors by content editing.
- **Evaluator**: this type of user inspects and validates the work submitted by the Reviewers. Her/his actions are performed through a similar online tool like the one accessed by the Reviewers where the previous work is accessed, assessed and possibly edited. Reviewer's main competence is to accept or reject the translation according to its quality and even more to perform a judgment of the Reviewer's ability.
- **Administrator**: this type of user is the main representative of the broadcaster and is enabled to perform general management tasks, while she/he does not directly work on the subtitle-content production.

Next to the supported functionalities for the management of the crowd-workers, the platform is responsible for handling the visibility of the working content. Based on the broadcaster requirements the content's visibility is decided by a set of confidence level-based filtering of the users from the perspective of the collaborative broadcaster.

5.4.2. Development of API integration

The following list of steps consist a scenario of user-system interaction that involves different modules of the EasyTV platform. Such scenarios were used during the definition of the workflows and the development of the integration APIs.

1. Initialization – Subtitles Production Module (SPM) registers a new translation task on the
2. Initialization - SPM to enable the translation service on the Service Manager (SM)
3. A content owner admin logins to the EasyTV Platform and checks the available services
4. Creates a new job to the EasyTV Platform through the SM, which consists of the translation of the original Catalan subtitles of a given asset into English. The SM receives all the required data including the access a video version of the content in low quality, and the Catalan subtitle file.
5. The SM orchestrates the different tasks required to achieve the jobs. The first step starts the translation job on SPM for an English version of the subtitle file. Inside the module, the Automatic Translation Module stores the result data of the generated file of English subtitles.
6. The SPM posts a new job in the Crowdsourcing Platform (CP) that requests the translation of the subtitles of this asset from Catalan to English language.
   *The activity in the CP continues: all subscribed users receive an e-mail, containing a new-job notification, and then an Evaluator assigns the job to a collaborative user (a reviewer). The collaborative user logins in the CP, to assign and begin with the job, then will be

forwarded to the Subtitle Reviewer Interface to accept or edit the automatic translation.

7. The CP generates a one-time token that is valid for a unique user and contains information related to the user-id and the job-id assigned. It is provided to the SPM
8. Once the SPM has validated the *jobid-userid-token* rule policy, the user is redirected to the **Subtitle Reviewer Tool**, adding as query parameter the access triplet to the URL
9. The Subtitle Reviewer Tool, that is part of EasyTV SPM, recovers the triplet information and do login on Subtitles Production API with the triplet credential, and get a session cookie to continue working (1 week). Nobody else should reuse the triplet credentials. In any case, the assigned user could enter the CP; get another one-access token, allowing to be paired again.
10. The Subtitle-Reviewer-Tool recovers job information, and prepares the environment for working.
11. Once the collaborative user ends the job, the module stores the resulting content of the subtitles translation in the filesystem controlled by the SPM and updates the status of the job to indicate that it is pending of validation.
12. The SPM informs the CP, that there is a job pending to be reviewed. The platform notifies the proper reviewer/s, indicating that there is a job pending. At some point, there will be a reviewer assigned that wants to start their job and access to the editor/validator tool.

Based on narrative scenarios like the ones presented in the previous list, the integrating partners compiled activity diagrams which led to design the first draft of the integration API specification and the involved data models which served as the basis for their current functional deployment [9,10].

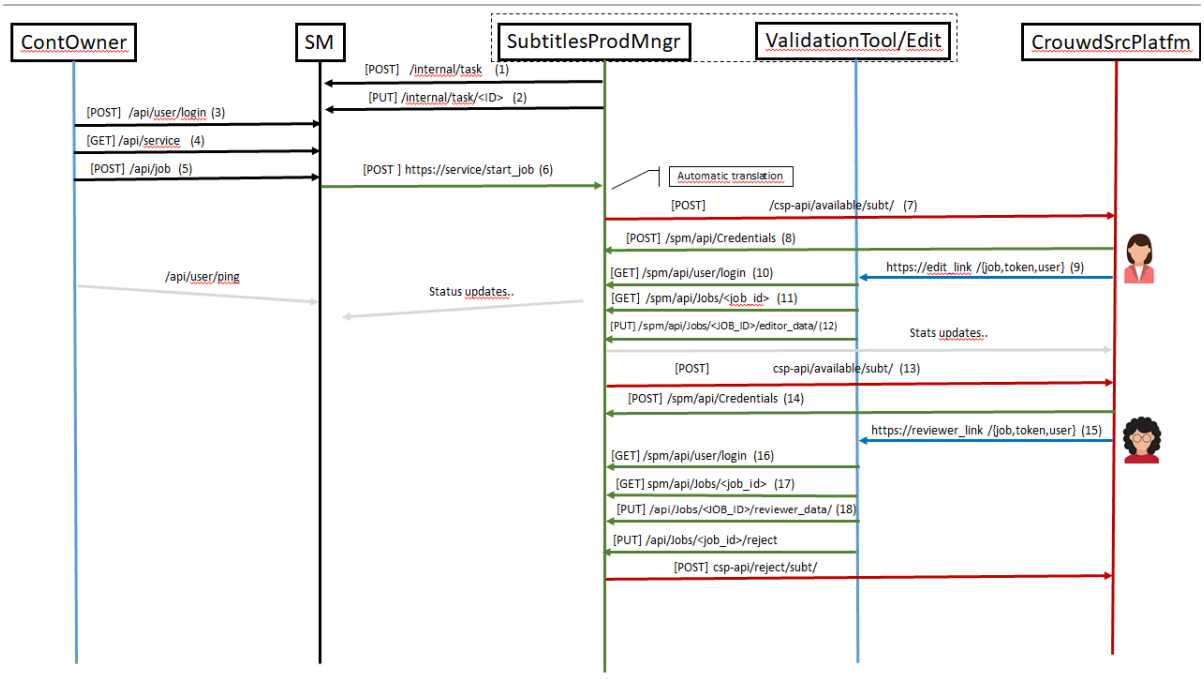


Figure 15. Activity diagram of Crowdsourcing with SPM

5.4.3. Volunteer evaluation sub-workflow

Apart from the standard subtitling workflow, the broadcaster partner has compiled a volunteer

[9] https://cp.easytv.eng.it/api-docs/

[10] http://spm-api.easytv.eng.it/index.html/

recruitment plan that has been developed as part of the Crowdsourcing recruitment system. The aim of this sub-workflow is to serve as a tool to apply the broadcaster's qualification standards during the conducting of the open pilot with users.

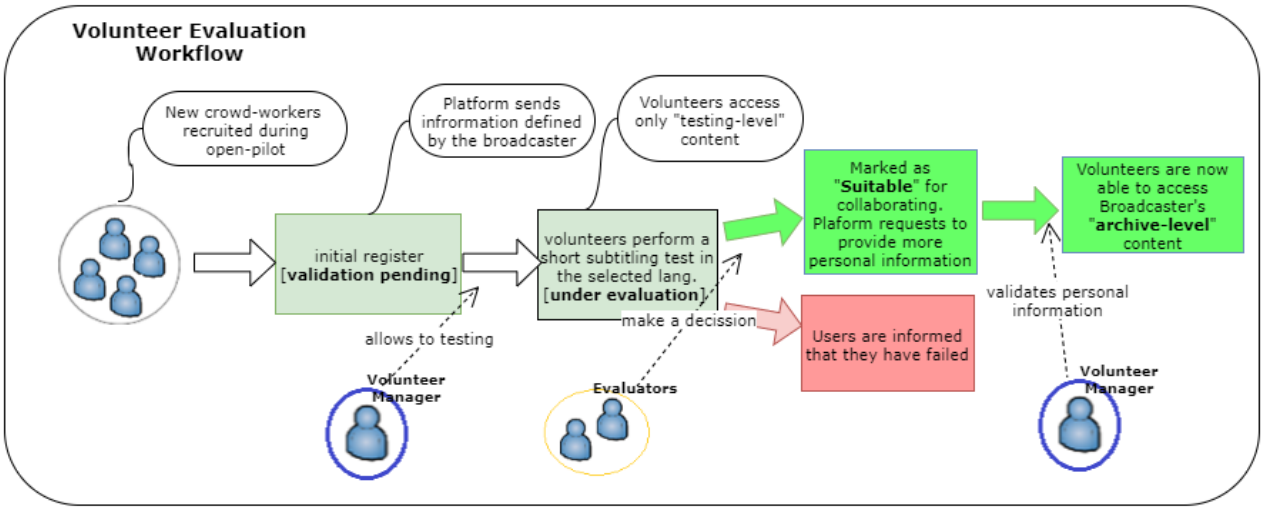


Figure 16. Overview of the volunteer evaluation workflow

The volunteer-evaluation scenario is implemented as a Crowdsourcing sub-workflow being part of the Subtitling Pipeline. An overview of this workflow is depicted in the diagram of Figure 16. Initially the new recruitments are requested to pass a separate subtitling test for each language they are interested to work with. In this stage, the platform restricts the access to the new volunteers only to testing-level content, as provided by the broadcaster. The submitted answers are passed to collaborating users of evaluator role, who make the final assessment for the new recruit based on the qualitative results.

It should be noted that during all the stages of the workflow, the crowdsourcing platform facilitates the communication of the involved users with a dedicated email-notification service. The platform allows these emails' content to be customized for the needs of the volunteer workflow by the broadcaster.

5.4.4. **Multi-provider scenario**

In the context of the EasyTV project, the partners contemplate a multi-provider scenario of the crowdsourcing platform for an open multilingual subtitles pilot, conducted by the CCMA. The multi-provider scenario will seek to put into test the platform's ability of handling the case that several content owners publish crowdsourcing tasks with their content to the EasyTV platform expecting to retrieve their results once completed.

The CCMA as the EasyTV partner being the **main content provider** orchestrates the plan with the purpose to take advantage of the integrated system. Therefore, CCMA's provided content will be divided into different accounts (organization entities) in order to test this feature and validate its management. Besides that, also few pieces of content from UAB EasyTV partner will be used to test the workflow, under a new content owner account. For that reason, each content owner account will employ its own platform administrator, reviewers and evaluator meaning that for example a reviewer user working with UAB should be enabled to contribute to UAB content but should not have visibility to access CCMA content.

Considering that the crowdsourcing platform implements certain access-policy strategies, in the case that a broadcaster like CCMA wants to go further and offer content that is sensitive in terms of rights and confidentiality, the access-policy should be customizable by the collaborating

organization. This entails confidentiality agreements with users and the preparation of the platform to support more demanding visibility restrictions. As result, this restrictive scenario will require from the platform to apply exclusive visibility conditions over the contents.

6. PLATFORM'S GRAPHICAL USER INTERFACES

This chapter demonstrates different functionalities of the EasyTV crowdsourcing platform by providing descriptions and illustrations through the use of screenshots from the user interfaces encountered by different types of users.

6.1. User Signup, Registration and Settings

Generally, any user is able to sign-up to the EasyTV crowdsourcing platform by providing a valid e-mail address and the corresponding correct password as shown in Figure 17. Furthermore, a user that is not currently member of the platform can easily register by providing his/her first and last names, a valid e-mail address and any other required information, as shown in Figure 18.

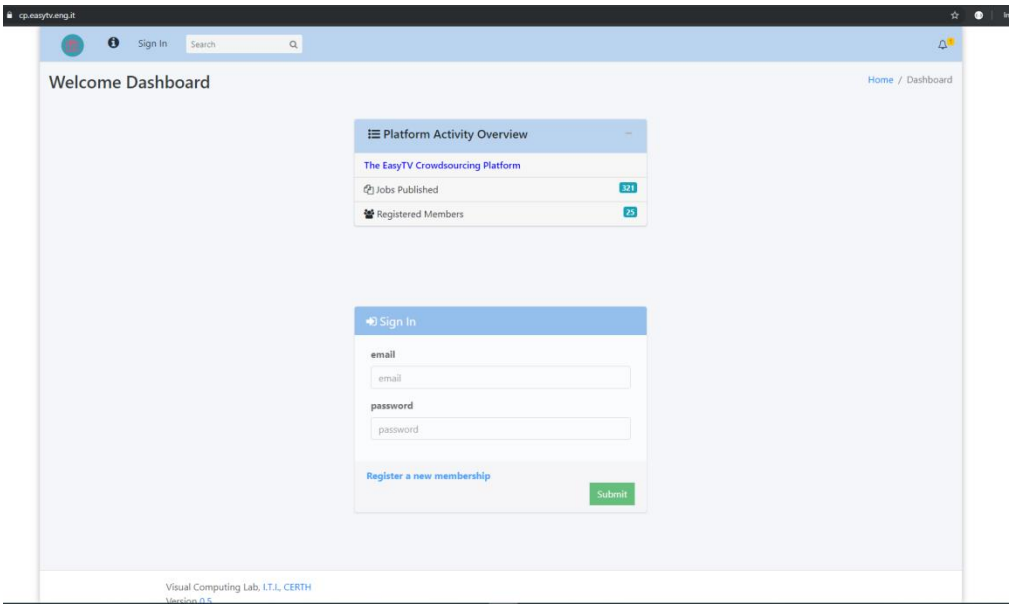


Figure 17. The login screen appears whenever a user wants to login with the EasyTV crowdsourcing platform.

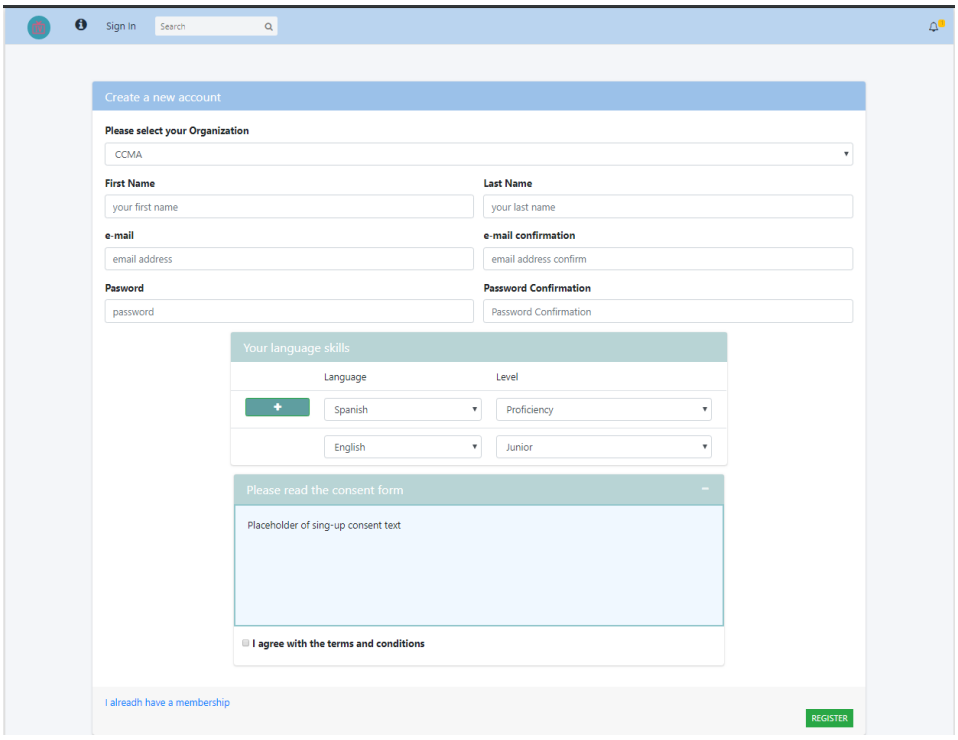


Figure 18. An instance of the user-register page requesting certain information.

The user-register forms page may be customized per organization entity, so it may request additional info according to the settings enabled by the Organization's administrator user. For example, in the presented instance of user sign-up page, the platform requests the registering user to declare her/his knowledge level for certain languages and also to accept a consent form (see data models in 3.3). The sign-up page's appearance is managed by the administrator of the organization through a settings panel where customization options are available presented in Figure 19.

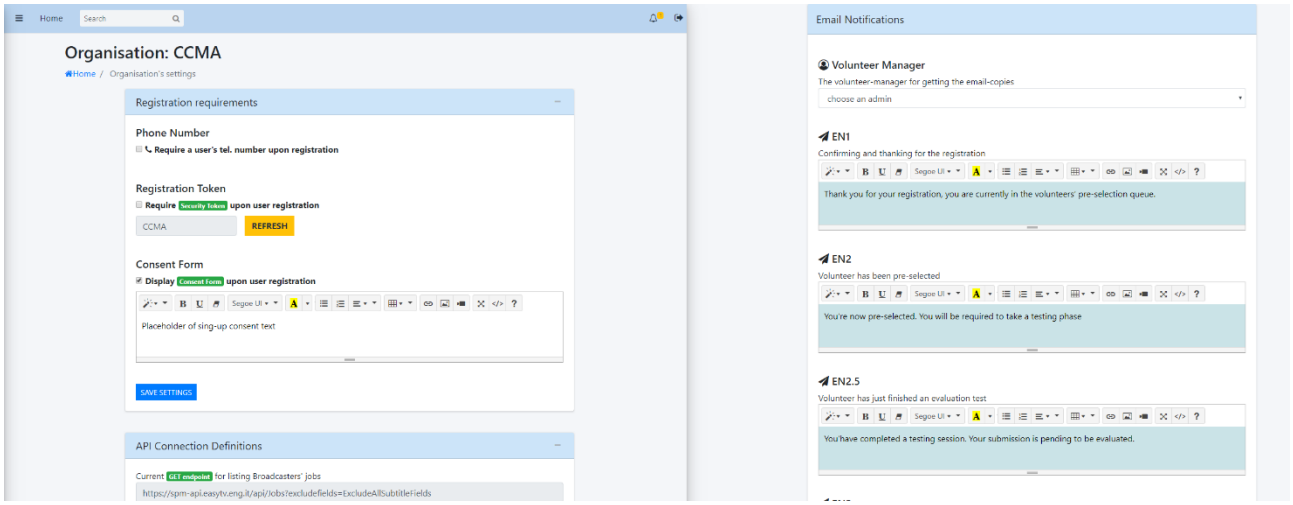


Figure 19. Supported organization settings

6.2. Interfaces involved in the Subtitle Production Pipeline

6.2.1. Overview

As discussed in the description of Subtitle Production Module the crowdsourcing platform acts as gateway for tasks publication, content access and progress tracking. For this purpose the platform needs to present to the collaborative crowd-workers up-to-date listing interfaces which are integrated to the content owner's API. This set of interface has been developed iteratively to meet broadcaster's requirements.

6.2.2. Reviewer's Dashboard and Subtitle Application

In Figure 20, a reviewer user navigates in her/his main dashboard. On top of the dashboard the user is informed about her/his current confidence level status per declared language, which is used as the main criteria for implementation of content-visibility policy. In this instance the working Reviewer user is notified by the crowdsourcing task-management policy that she/he is currently restricted to work only on his assigned subtitle-job, which implies that she/he should either finish it or get unassigned from it before platform's policy enables her/him to access any other published and pending job. In Figure 21, there is an instance of a Reviewer user working with broadcaster's content by accessing the CCMA's front-end application **Subtitle Reviewer Tool** in an iframe window. According to the description of the workflow in paragraph 5.4.2, the crowdsourcing platform implements certain access policies conforming to the broadcaster requirements. In the presented case, the reviewer is assigned a unique token shared between the platform and broadcaster, and passes it as a parameter in the iframe window of the application. Subsequently the user can contribute to the subtitling production process by following the subtitle-flow in the broadcaster's application and save his work. This action redirects them back to the crowdsourcing dashboard, which is at the same time being informed by the broadcasters' API about the latest progress in the assigned subtitling-job.

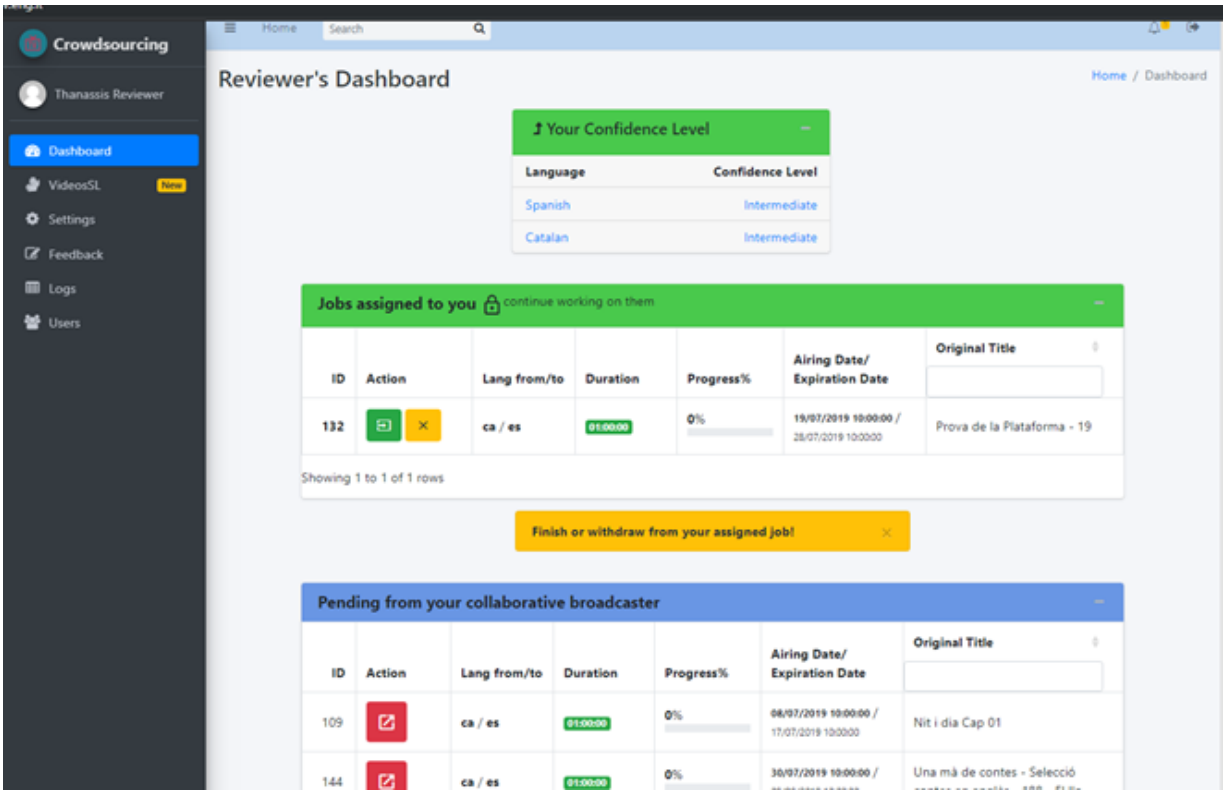


Figure 20. A Reviewer user is presented with the available subtitles jobs in his/her main dashboard

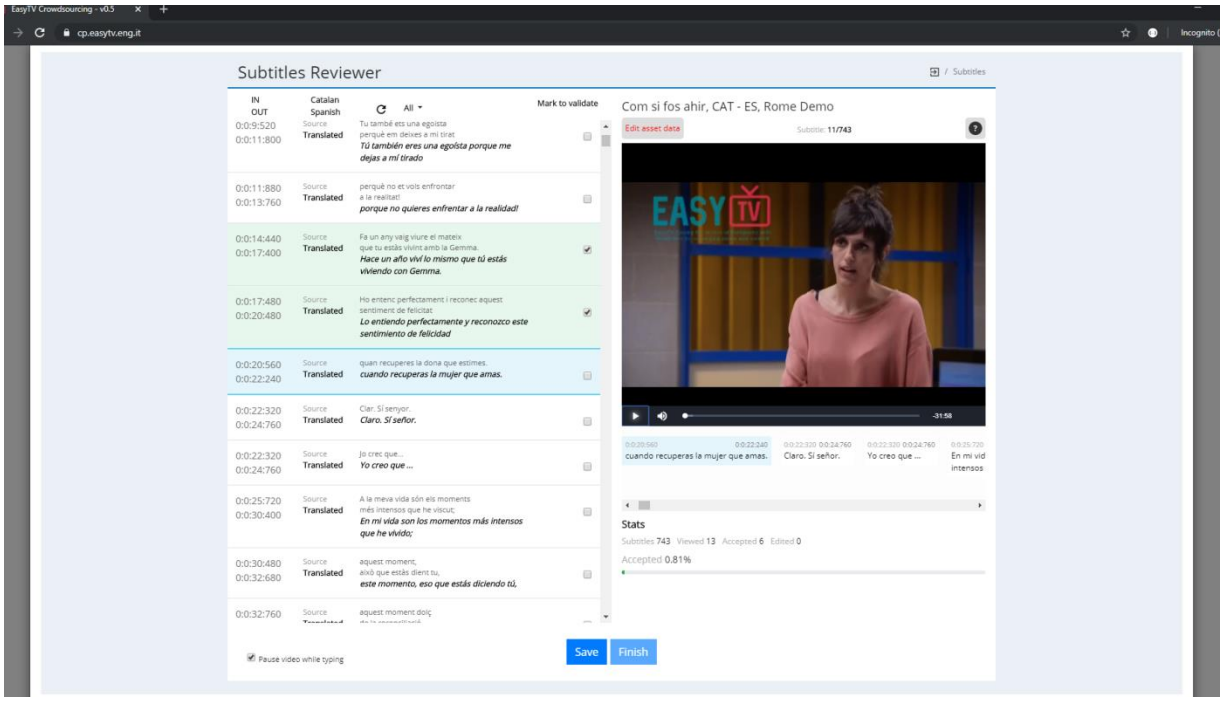


Figure 21. The Subtitle-Reviewer-Tool accessed as an iframe from the Crowdsourcing Platform

6.2.3. Evaluator's Dashboard and Subtitle Application

From the broadcaster's perspective the evaluators are regarded as professional collaborative users who have an elevated role related to content visibility and users' evaluation. Essentially, the platform's interfaces for this type of user enable them to access the same content as the reviewer

users but with a different final scope. In their main dashboard they are informed about previously reviewed subtitles. They get assigned jobs in the same manner as the reviewer users, and subsequently they access the Subtitle-Evaluator-Application application on which they evaluate the submitted work of reviewers. Their assessment has two different effects: on one hand their judgement is the final assessment about the quality of the produced subtitles' translation and on the other hand their evaluation is communicated to the crowdsourcing platform and aggregated to the reviewer user's performance statistics.

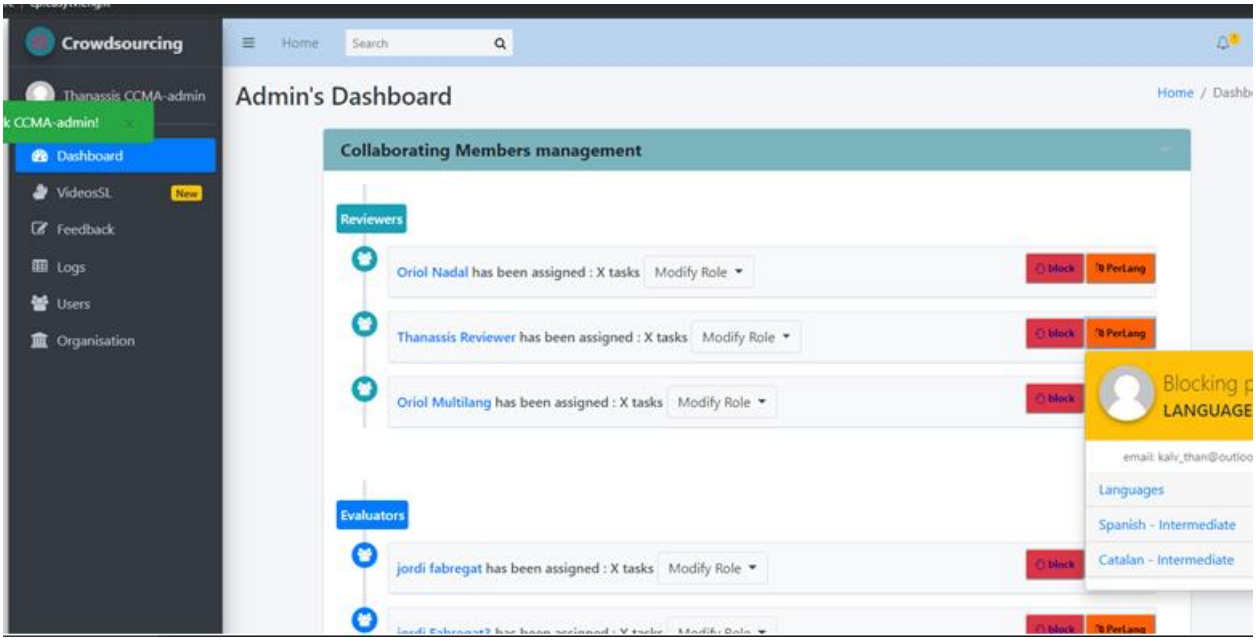


Figure 22. Broadcaster's admin can manage users' role and content visibility

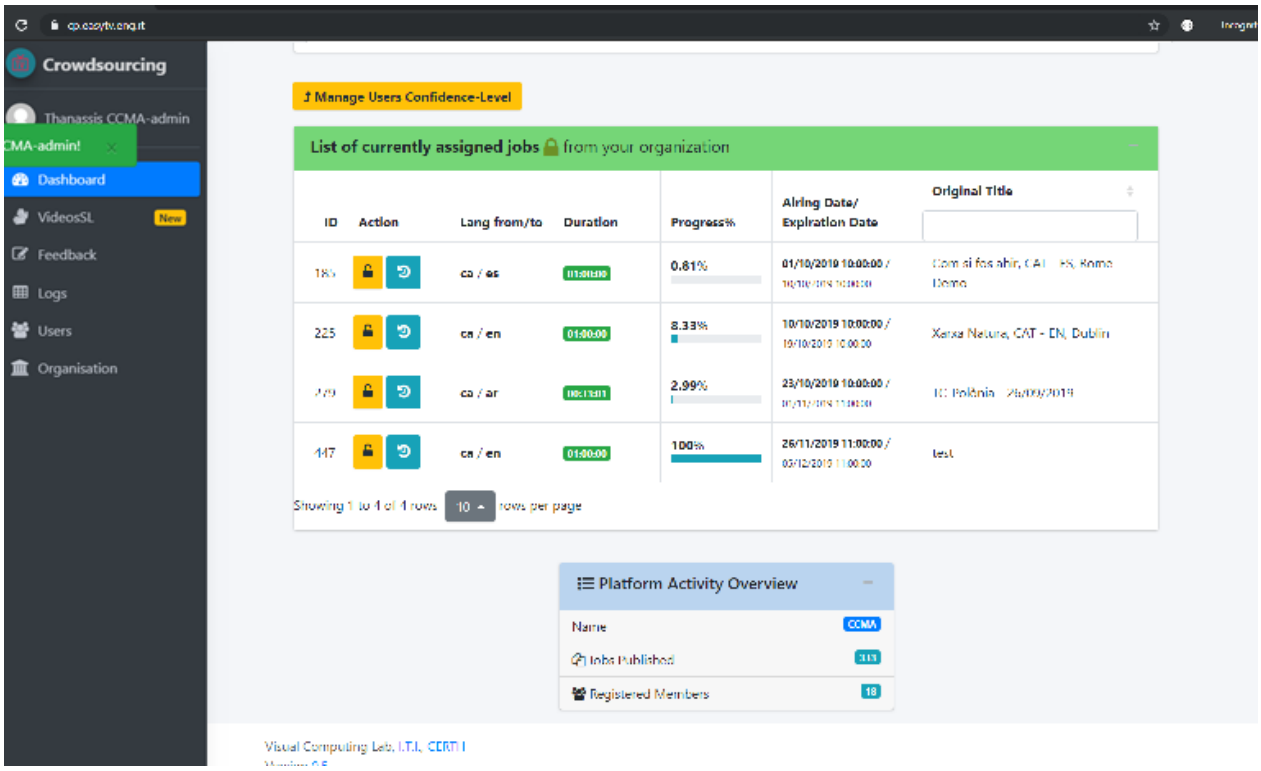


Figure 23. Admin can review the history and unlock pending jobs

6.2.4. Administrator's Dashboard

The broadcaster organization assigns to their administrator user the tasks of configuring his/her institution setting on the platform and also the general managing of its users. For this purpose, the main dashboard of the administrator contains an overview of the registered users (Figure 22), where the admin is granted the ability to elevate the role of the currently collaborating members or block them. Additionally the administrator is responsible for unlocking broadcaster's pending jobs that lack the expected progress. The platform enables the administrator to access a history of the progress and decide if the job should be un-assigned from the current user (Figure 23). Finally, the administrator panel presented in Figure 24 is related to the discussion of paragraph 5.4.3. On the volunteer evaluation workflow the administrator is responsible to take certain actions related to the volunteers' acceptance as collaborative users. The interface for managing volunteers is separated per language and enables the administrator to take care of the users' progression in the involved stages.

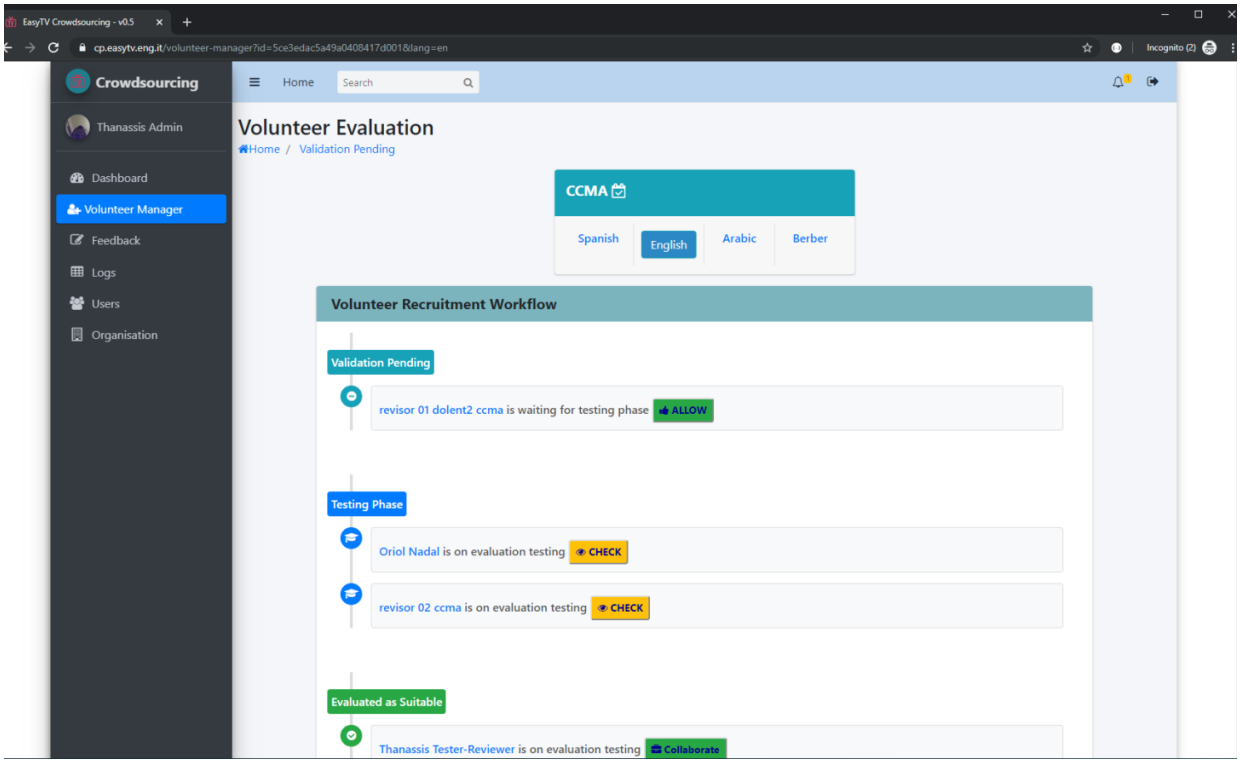


Figure 24. Administrator manages the volunteer workflow

6.3. Interfaces related to Sign Language Tasks

In this section, we describe the functionality of the crowdsourcing platform that enables the creation of new projects and the creation and distribution of tasks for the Sign Language Production. Initially, we revisit the definition of a general task entity, as well as the required fields that need to be completed before we describe in detail the functionality of project and task creation.

Crowdsourcing Task– includes the sequential steps of a job

i) Language options
ii) Expected number of contributions and users (profiles' requirement)
iii) Verification method

6.3.1. Creating new tasks

The scope of the *task-creator* interface is to provide users with the ability to create and edit

platform-related content through intuitive forms. This interface was part of the development for the first prototype of the EasyTV crowdsourcing platform.

A task workflow can be summarized in four main steps:

- Access a simple tool for defining tasks
- Communicate (broadcast) the project tasks to a targeted pool of users
- Accept and process user's input in the system
- Validate the content and update the task status

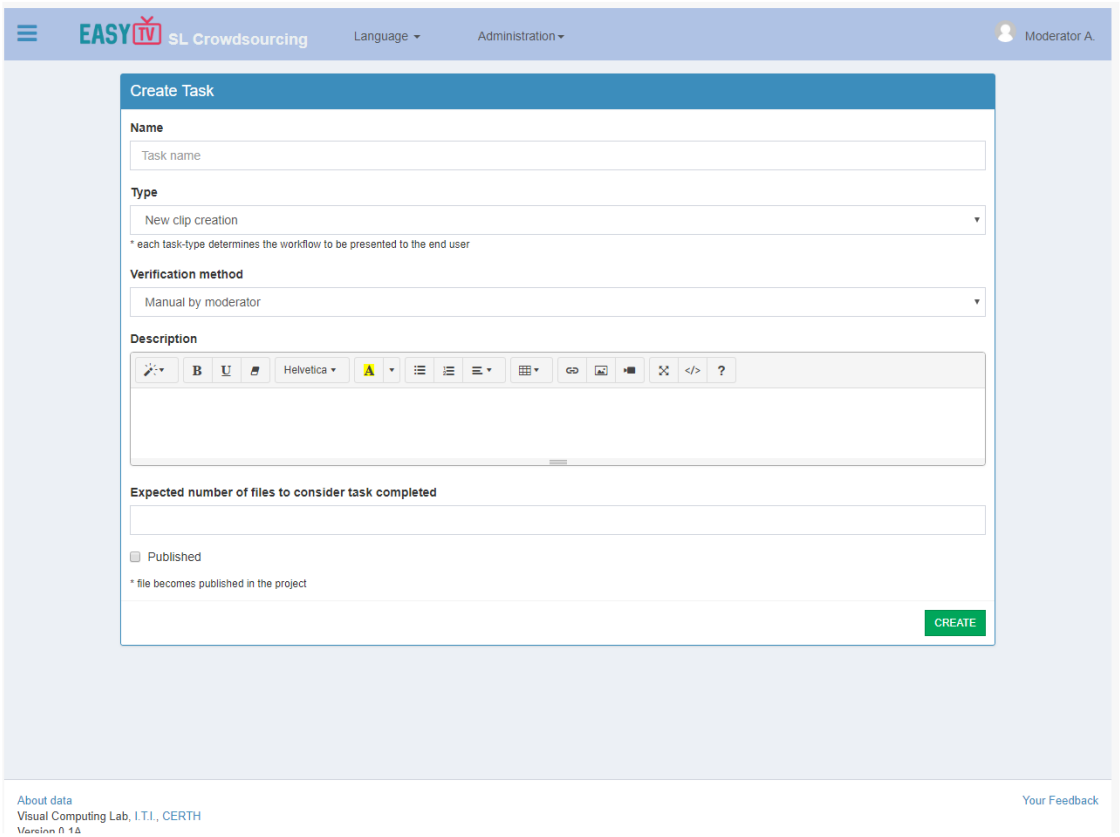


Figure 25. Definition of a new crowdsourcing task

As far as the task creation is concerned, a moderator can define a new task for a certain concept by filling the form, shown in Figure 25 with all the necessary information that include the name and type of the task implying the stage of the task. The method used for the verification of the task completion and important information that can assist the interested users in successfully completing the task is also provided.

Furthermore, the moderator is given the option to notify specific users about the presence of a new task (see Figure 26). This notification is processed as an offsite communication method and has the purpose to disseminate faster the news about the presence of new tasks and, as a result, achieve a possibly faster response and task completion time.

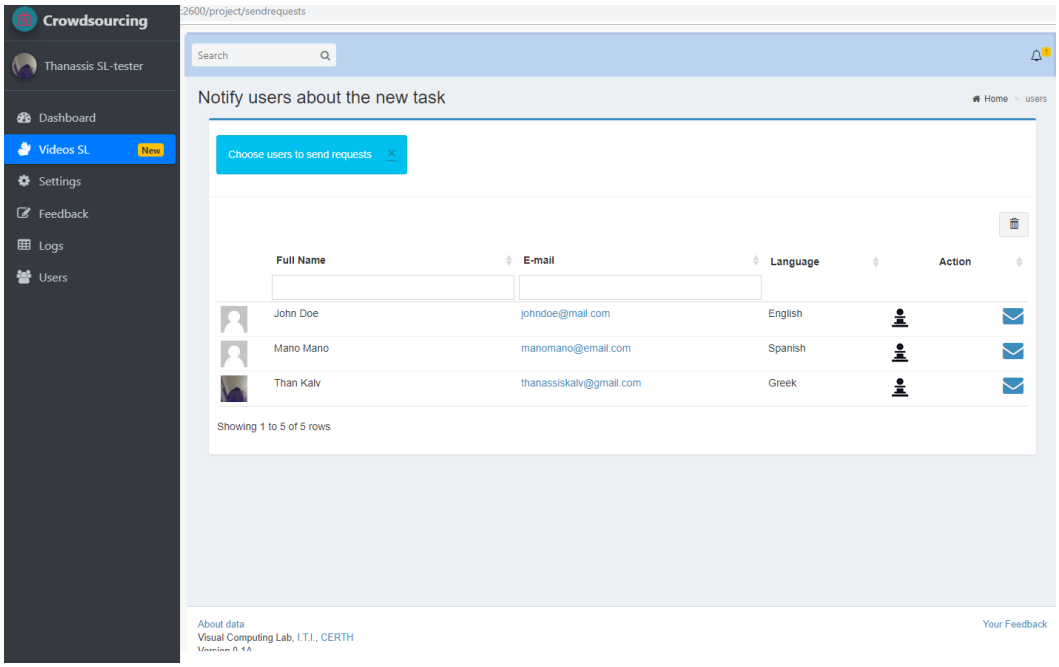


Figure 26. A moderator notifies available users for a new task

6.3.2. Completing a Sign Language Task

This section describes the procedure that a signer crowd-worker should follow in order to complete a Sign Language task. The procedure for the task completion is presented as a workflow in the EasyTV crowdsourcing platform and it consists of a number of **steps interfaces,** which prompt the user to provide the necessary inputs in order to successfully complete the task. It is crucial to allow for a simple and easy navigation between the steps in order to facilitate the tasks of a crowd-worker.

The steps required for the task completion by a crowd-worker are presented in Figure 27, with illustrations of the UI elements of the crowdsourcing platform. Initially, the user is presented with the concept of the sign language task and can see the required steps to complete the submission. The first step prompts the user to upload the necessary video.

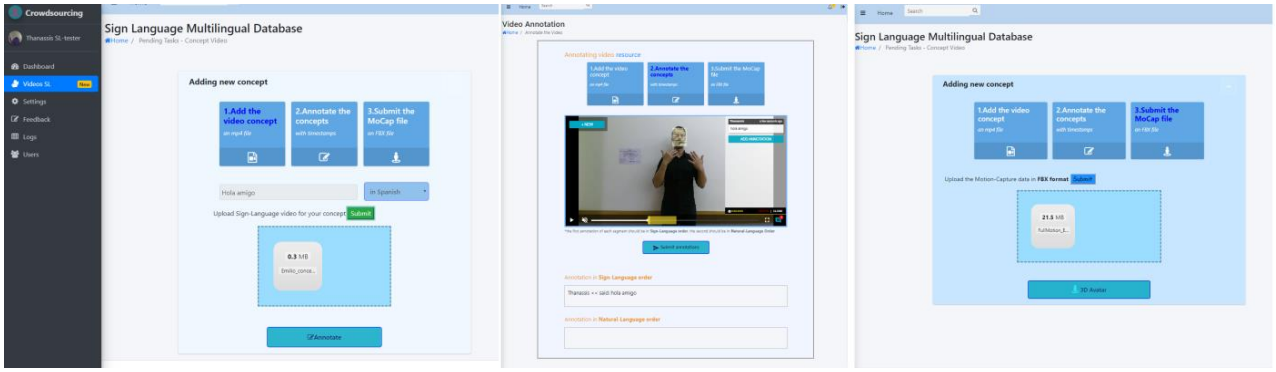


Figure 27. User performing sub-tasks steps which prompt her/him to upload and annotate and visualize the submitted data

Afterwards, the user is prompted to provide the video annotations by writing them in video-segmented intervals that include a single word or set of words accompanied with the corresponding natural language order of signing in the uploaded video file. In the third step, the task requests from the user the motion-capture data that have been generated by employing **the Sign Language Capturing** module that was described in detail in deliverable D3.7 [15], which

enable the user to playback an online 3D-Avatar and inspect the quality of the work. Finally the crowdsourcing platform informs the user about the successful upload of all the required files accepts the submission and informs the corresponding evaluators about the presence of a new submission, which receives the status of “under review”.

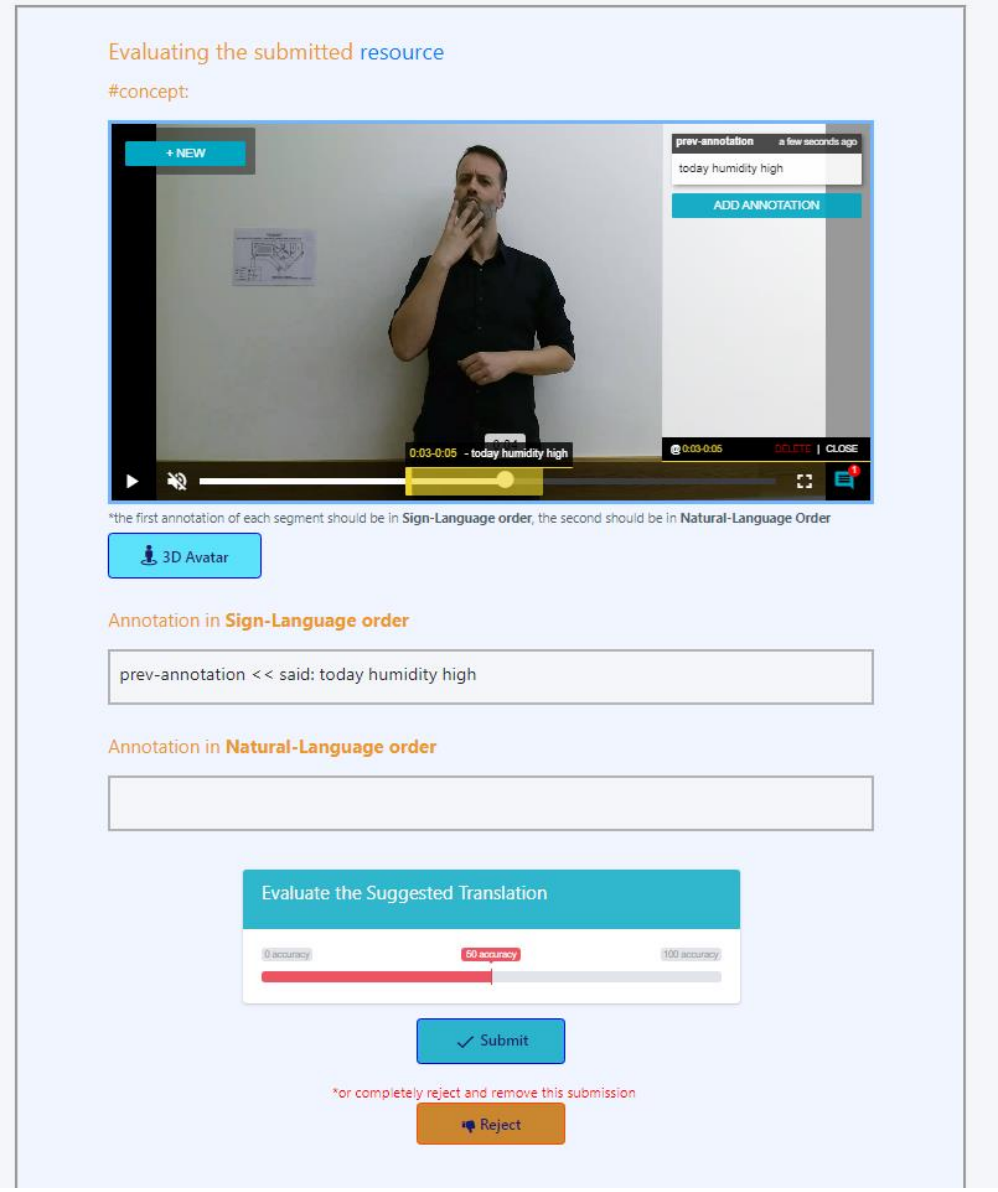


Figure 28. Content Evaluation by using the annotation-tool

6.3.3. Evaluating submissions to Sign Language Tasks

In this paragraph, we visit the interfaces accessed by the evaluator users of Sign Language that enable them to accept or reject submitted content. Once an evaluator is informed about the presence of a new submission, he/she is able to access the submission and view the submitted material. Afterwards, the expert user should be enabled to inspect and validate the material based on his/her knowledge and make a decision. The worker user is finally notified about the decision of the evaluator. In the case of acceptance, a final version of the submission is stored on a data repository and is made available in the platform’s filesystem. The crowdsourcing platform also enables the evaluator users to edit previously submitted time-segmented annotations using the annotation tool. This means that the evaluator user is allowed to edit submitted annotations in case that he/she considers that the submitted content can be accepted with certain corrections instead of being rejected completely. An instance of this interface in action is presented in Figure 28 where

most of the supported functionality is within sight.

7. CONCLUSIONS

Initially we revisited the technical details included in the development stack that was selected and analyzed how they serve the development process of the crowdsourcing web application. In the rest of the discussion the focus has been on presenting the development upgrades since the preliminary version of the Crowdsourcing Platform. The main components of the web application as presented in the Mid-Term deliverable have been used as the developing infrastructure and the application's codebase has been expanded to support communication and functionalities.

The major development drivers have been the integration requirements of the platform with Subtitle Production Module API and the Sign Language Ontology Web-Service API and their use-case scenarios. In order to meet these requirements the platform infrastructure supports an upgraded user management system and has incorporated new interfaces with enriched features.

The platform's current deployment[11] in the EasyTV Docker server allows continuous testing activities to be performed in order to ensure technical robustness and smooth cooperation with collaborative EasyTV services. This is a prerequisite before the EasyTV final test sessions with end users and the launch of a successful campaign of volunteers' registration for the Subtitle Production Module by CCMA as broadcaster.

[11] https://cp.easytv.eng.it

8. REFERENCES

[1] Brabham, D. C., “Crowdsourcing as a model for problem solving: An introduction and cases,” *Convergence*, *14*(1), pp. 75-90, 2008.

[2] Howe, J., “The Rise of Crowdsourcing,” *Wired Magazine*, June 2006.

[3] Saxton, G.D., Oh, O. and Kishore, R., “Rules of Crowdsourcing: Models, Issues, and Systems of Control,” *Inform Syst Manage*, vol. 30, no. 1, pp. 2–20, 2013.

[4] Hoßfeld, T. et al. “Survey of web-based crowdsourcing frameworks for subjective quality assessment,” *IEEE 16th International Workshop on Multimedia Signal Processing (MMSP)*, Jakarta, pp. 1-6, 2014.

[5] Buettner, R. “A Systematic Literature Review of Crowdsourcing Research from a Human Resource Management Perspective,” *48th Annual Hawaii International Conference on System Sciences*, Kauai, Hawaii: IEEE. pp. 4609–4618, 2015.

[6] Aitamurto, T., “Crowdsourcing as a Knowledge-Search Method in Digital Journalism: Ruptured Ideals and Blended Responsibility,” *Digital Journalism*, vol. 4, pp. 280–297, 2016.

[7] Taeihagh, A., “Crowdsourcing: a new tool for policy-making?,” *Policy Sciences*, 50 (4), pp. 629–647, 2017.

[8] Créquit, P., “Mapping of Crowdsourcing in Health: Systematic Review”, *Journal of Medical Internet Research*, 20 (5), 2018.

[9] Memrise – Learning, made joyful: https://www.memrise.com/.

[10] D5.2: “Mid-term report on the set up and implementation of the EasyTV crowdsourcing sign language platform and repository”

[11] D1.4: “Final release of the EasyTV system architecture.”

[12] D1.1: “User scenario and requirements definition.”

[13] D1.2: “EasyTV system requirements specification.”

[14] D3.2.1: “Enriched multilingual ontology with signs in different languages preliminary version.”

[15] D3.7: “Sign language capturing technology final version”

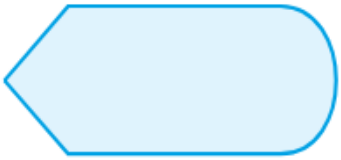
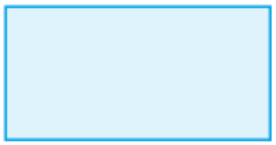
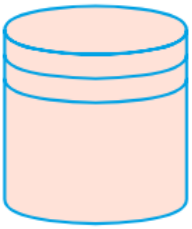
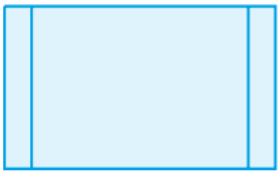
[16] D2.1 Sign language animation preliminary development and production

[17] Unity3D,“Interpolation.”[Online].Available:https://docs.unity3d.com/ScriptReference/Rigidbody-interpolation.html. [Accessed: 30-Sep2018].

APPENDICES

1. Flowchart shapes for user scenarios

In this section, we present the links between flowchart shapes and their semantic relations. These shapes are employed in order to clearly illustrate the user roles and functionalities in the developed EasyTV Crowdsourcing Platform presented in the current deliverable.

 **Preparation**	 **Display**	 **Process**
 **Database**	 **Manual User Input**	 **Database documents**
 **Subprocess**	 **Workflow direction**	 **Communication**