D5.2 Technical Specifications and System Architecture Definition

Partner Responsible	UTH-ECE
Other Contributors	AthenaRC
Document Reference	**D5.2**
Dissemination Level	Public
Version	1.0 (Final)
Due Date	February 2021 (**M13**)
Date of Preparation	February 2021

Contract No.: HFRI-FM17-2456

Editor

Gerasimos Potamianos (**UTH-ECE**)

Contributors

UTH-ECE: Gerasimos Potamianos, Katerina Papadimitriou

AthenaRC: Eleni Efthimiou, Stavroula-Evita Fotinea, Theodor Goulas, Christina Flouda, Gkioulan Ametoglou

SL-ReDu Principal Investigator:

Assoc. Prof. Gerasimos Potamianos
University of Thessaly, Electrical and Computer Engineering Department (**UTH-ECE**)
Volos, Greece 38221
email: gpotamianos@uth.gr (gpotam@ieee.org)

Executive Summary

The SL-ReDu project aims to advance the state-of-the-art in the automatic recognition of Greek Sign Language (GSL) from videos, while focusing on the education use-case of standardized teaching of GSL as a second language. In this deliverable (D5.2), we present the SL-ReDu platform technical specifications and define the adopted system architecture that, together with the design of the human-computer interface (which will be presented in Deliverable D4.1), will guide the implementation of the first version of the SL-ReDu system. Specifically, after a brief overview of the functional requirements of the platform, we proceed to describe the solution proposed for its technical implementation. This adopts a hybrid architecture, employing a server-hosted web-based application that communicates appropriately with the learner-side device, allowing simultaneous access by multiple learners. The learner device should meet several technical specifications to allow recording SL production videos of sufficient quality by the learner and, subsequently, the on-device automatic recognition of such videos. The deliverable is part of the second project milestone (MS2), and it will lead to the first version of the system implementation (Deliverable D5.3) and its subsequent user evaluation (Deliverable D5.4).

Table of Contents

1 Introduction

The design and implementation of interactive language learning systems represents a relatively recent trend in the sign language (SL) literature [1-3]. Compared to traditional spoken languages, the task faces additional methodological and technical challenges in the case of SLs [4-7]. For example, self-monitoring and testing in SL as L2 exploit video data streams to build interfaces for both passive and productive language exercising. In particular, while the creation of exercises to test comprehension of language elements is a relatively well-implemented task, active learner participation in SL production introduces a completely innovative aspect. Indeed, to accommodate this feature, the learners need to: (a) create their own videos of linguistic performance, thus requiring the presence of suitable recording equipment at the learner end; (b) upload the recorded video to a suitable device, thus necessitating speedy data transfer; (c) have the video processed by an SL recognition engine, which is a computationally demanding process due to the predominantly deep-learning based algorithms employed [8-18] that require graphics processing unit (GPU) acceleration [19]; and (d) obtain the SL recognition results within reasonable latency. Such pipeline could possibly be undertaken by multiple learners simultaneously, during their repetitive process of self-monitoring (learning), or during their objective evaluation by participating in exams, in both cases covering extensive GSL educational material, in line with an accredited SL curriculum, such as suggested by the Common European Framework for Languages [20] (see also Deliverable D3.2 [21]). In the meantime, the instructor / tutor should be enabled to access the learners' results, as well as to set up the educational material for their self-monitoring and exam taking. The above naturally lead to multiple functional and technical requirements for the SL-ReDu prototype system that must be accommodated through a carefully designed, flexible, robust, and well-performing system architecture. The design of such architecture constitutes the focus of this deliverable.

In more detail, in D5.2 we first overview the functional specifications of the SL-ReDu platform that the designed architecture should serve (Section 2). Next, in Section 3, we present the adopted architecture. Its nature is hybrid, employing a server-hosted web-based application that communicates appropriately with the learner-side device, while also allowing simultaneous access by multiple learners (each with their own device). Such devices should meet several technical specifications that we enumerate, so as to allow recording of SL production videos of sufficient quality and, subsequently, their on-device automatic recognition, requiring high-end computational capability. Further, in Section 4, we provide details on a number of aspects in the adopted architecture, namely: (a) the data representation of the educational material for self-monitoring and evaluation that is set up in the platform (Section 4.1); (b) the communication protocol and information exchange between the web-based application and the learner device (Section 4.2); and (c) the adopted SL recognition pipeline (Section 4.3), following recent Deliverable D2.1 [22]. We finally conclude D5.2 with a short summary (Section 5).

2 Functional Specifications

The SL-ReDu prototype system design aims to address the inherent obstacles of traditional practicing and testing methods in GSL as L2 learning, by providing two modules for self-monitoring and objective assessment of the learner. In its design, educational considerations should play a crucial role, since interaction options have to take into account all aspects of GSL linguistic systems from the GSL semantic to the morpho-syntactic phenomena in both GSL perception and production. For this purpose, the design should allow for direct implementation of teaching methods and material, entailing different SL practice drills, which cover the phenomena of GSL from sign formation to complex syntactic and semantic utterance productions. Exercise types may include typical multiple-choice questions that exploit image, video, and text to trigger user response (see Figure 1), as well as user answers via video recording of GSL production (see Figure 2). SL-ReDu is unique in this respect, allowing the GSL learner to actively sign, in order to enhance and solidify new knowledge, as well as to be assessed for the ability to correctly produce utterance signing.

The system should be accessible to registered users only, i.e. both learners and the instructor. Its two modules, for self-monitoring and objective assessment, will incorporate different types of components, including the system database, front-end user interface, back-end interface, images, and video files, and should provide a user interface (front-end) for the learner and a content management system (back-end) that is used by the instructor to create learners' assessment tests and allow performance record-keeping over time. The database will provide the material hosted in both user interface and content management sub-systems.

As far as user roles are concerned, both the learners and the instructor should first login to the platform with their credentials. The instructor could then create tests to administer to the learners, retrieving exercises from a topic-bank with all available exercise types and content. The instructor should also be able to keep a record of learners' performance, with the platform also supplying helpful tools to the instructor to adjust assessment parameters such as time limit and allowable attempts by the learners. On the other hand, if logged in as a learner, the user should be able to access all educational material covering the different topics of the curriculum, as incorporated in the various exercise types of the system, in order to consolidate and improve SL skills. When taking a test, the score of each learner response is saved after the answer submission, the overall test score gets updated, and finally it gets displayed at the end of the exam session.

In summary, the SL-ReDu platform should provide support for:

- Self-monitoring by learners, allowing multiple passes and non-linear navigation through the educational material at a leisurely pace.
- Objective evaluation of learners, either as a dry-run simulated exam or while undertaking an actual exam designed by the instructor, enforcing linear navigation through the material within limited time and number of iterations / attempts, as well as scoring the learner answers.
- "Passive", multiple-choice, single-answer type questions on GSL comprehension topics employing various combinations of text, image, icon, SL video, and signing avatar based stimuli.
- "Active" GSL production by the learner, with video recording and automatic recognition of the produced articulation, also providing feedback to the learner for correct positioning with respect to the recording equipment.
- Multiple learners that may simultaneously access the system.
- Educational material organization by the GSL instructor, covering the accredited GSL material.
- Access to the learners' evaluation records by the GSL instructor.

The architecture of the SL-ReDu system should accommodate the above functional requirements through a carefully designed, flexible, robust, and well-performing approach. The adopted architecture for this purpose is analysed in the following sections.

Figure 1: Mockup of an avatar video-driven, multiple-choice, single-answer question concerning GSL lexicon comprehension, shown to the learner during self-monitoring or objective evaluation, employing the SL-ReDu prototype system.

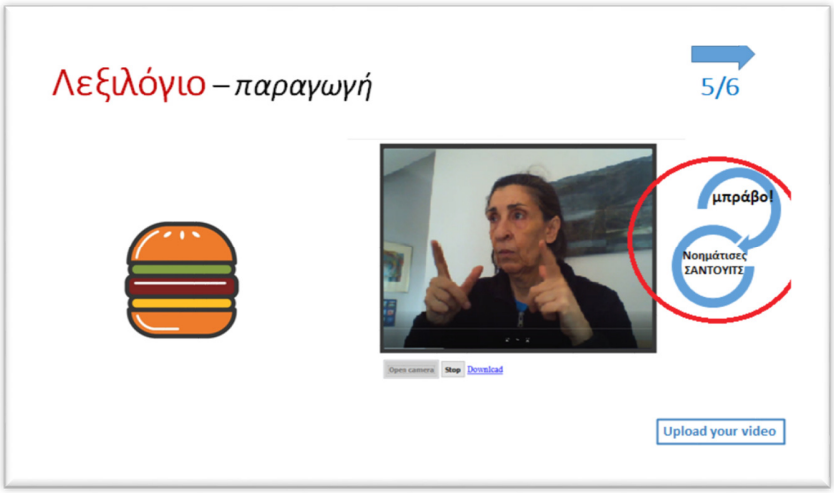

Figure 2: Mockup of GSL lexicon production by a learner with feedback from the GSL recognizer during self-monitoring or objective evaluation, employing the SL-ReDu prototype system.

3 Adopted Architecture and Technical Specifications

Based on the above functional requirements, it was deemed appropriate to develop the architecture of the SL-ReDu prototype system as a web-based application running on a web server that will be responsible for the end-user interaction with the web platform. The dynamic web platform will be developed in the PHP programming language [23], in combination with HTML5, CSS3, and JavaScript [24-26]. This represents a natural choice, since PHP is a popular general-purpose scripting language that is commonly used for web development, supporting communication with different types of databases, such as MySQL, SQL Server, and PostgreSQL [27-29]. Concerning the relational database used for the development of the web application, including the storage of the presented educational material and the performance of platform users, it was decided that this will be MySQL, since it constitutes the world's most popular open-source database [27]. The web application will be hosted on an Apache Web Server [30]. The overall approach adopted in the SL-ReDu project is depicted in Figure 3.

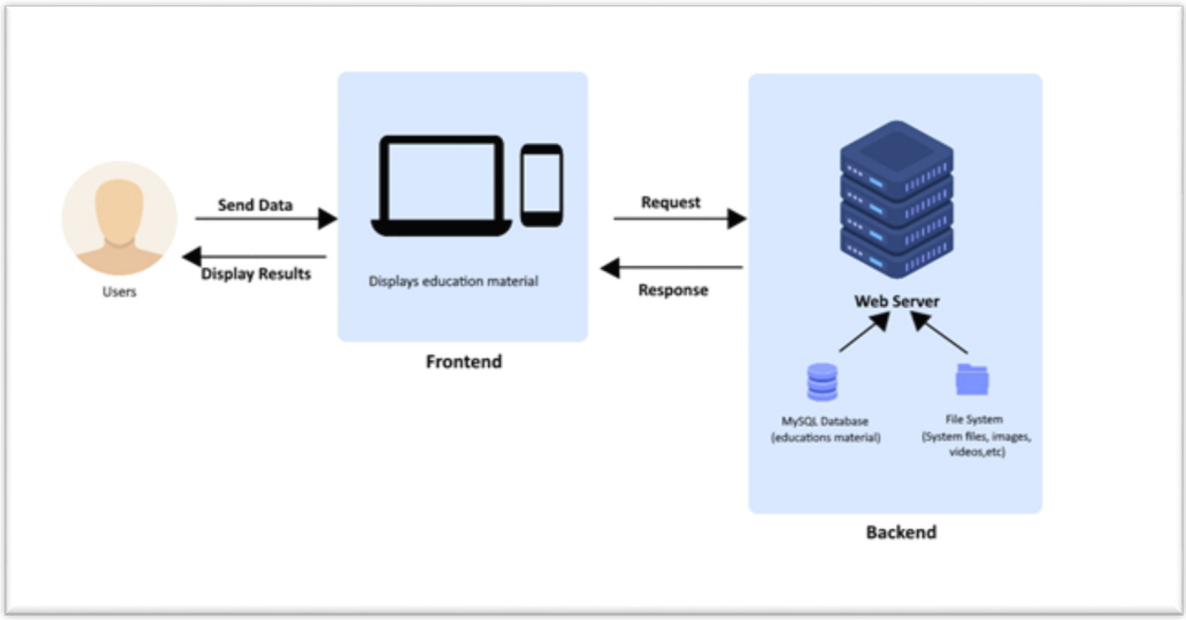

Figure 3: Schematic of the web-based architecture, adopted for the SL-ReDu prototype system.

To meet the functional specifications of Section 2, the web application will include a login page, where the user is encouraged to either sign up or login if already registered with the system, as well as a main page where the material will be presented. The application will support three types of user roles: student (learner), instructor (tutor), and system administrator. Based on the provided user credentials, the system will be able to recognize if the user is a learner or an instructor, and the appropriate web environment will be presented to accommodate the corresponding role.

In the learner case, the user will gain access to the self-monitoring and the test (objective evaluation) execution environments. In the former case, the learner will be granted access to all educational material in the self-monitoring environment covering the different topics of the GSL curriculum (see also Deliverable D3.2 [21]), as incorporated in the various exercise types of the system, in order to consolidate and improve GSL skills. The web application will allow multiple passes and non-linear navigation through the educational material at a leisurely pace. On the other hand, in the objective evaluation environment, the learner will be undertaking an exam that has been created by the instructor either as a

dry-run simulated exam or an actual one. In both scenarios, the web application will enforce linear navigation through the material within predefined time constraints and number of iterations / attempts, as well as score the learner answers, providing the overall grade at the end.

In the instructor role, the user will be able to create the tests to be administered to the learners. To do so, the instructor can retrieve exercises from a database with all available exercise types and content. Further, the instructor can keep a record of learners' performance, while the platform will supply helpful tools to adjust assessment parameters such as the exam time limit.

The main page of the application will be structured taking into consideration the web usability of the environment. A responsive template will yield the appropriate presentation of the educational material in various browsers. A main menu will provide the user the opportunity to navigate easily to the sections of the web platform. The material of the content will be presented in a tree-like structure, while a breadcrumb path will allow browsing to the various levels available.

The aforementioned approach suffices in the case of "passive", multiple-choice type drills on GSL comprehension topics, employing various combinations of text, image, icon, SL video, and signing avatar based stimuli. However, as discussed in Section 2, the application should also be able to accommodate GSL production exercises for both self-monitoring and objective evaluation. Such, naturally involve the video recording of the learners' productions and the automatic recognition of the resulting videos. This is a complex process, requiring: (a) the learners to create their own videos of linguistic performance, thus necessitating the presence of suitable recording equipment at the learner end; (b) uploading the recorded videos to the GSL recognition engine; (c) processing the produced videos by the GSL recognizer, which is a computationally demanding process; and (d) returning the GSL recognition results to the web application within reasonable latency. Since the above steps should be supported for multiple learner users, possibly performing GSL productions simultaneously, it was deemed appropriate to run the process of video recording and GSL recognition at the learners' side, employing their individual devices. This guards against overloading the SL-ReDu platform web server, but on the other hand complicates communication between the learner device and the server and also requires the learner device to support memory- and computationally-intensive GSL recognition algorithms.

To enable this approach, in the case of GSL production exercises where a learner is asked to perform signing in front of his/her device camera, a camera module will be developed with the use of the Web Real Time Communication (WebRTC) Application Programming Interface (API) [31]. WebRTC is a technology that enables web applications and sites to capture and optionally stream audio and/or video media, as well as to exchange arbitrary data between browsers without requiring an intermediary. The set of standards that comprise WebRTC makes it possible to share data and perform teleconferencing peer-to-peer, without requiring the user to install plug-ins or any other third-party software. This enables video data recording at the learner's device having a specific video file name that contains information concerning the learner user id, the specific exercise (e.g., the nature of the GSL recognition task), and turn. The learner will be allowed to preview the recorded file, before submitting it to the recognition engine. Such action will then trigger the GSL recognizer, based on a local application that will be running on the learner device. The recognition module will first employ computer vision algorithms (2D pose estimation of the learner [32]) to inspect the relative position of the learner with respect to the field-of-view of the device camera, providing instructions for rectification in case of wrong positioning (e.g., occluded manual articulators). If no problems are detected, the video file will then be subjected to GSL recognition by an appropriate visual feature extraction pipeline and statistical recognition models that correspond to the specified GSL task (see also Section 4.3). Finally, the web application will obtain the recognition result to provide learner feedback on the linguistic production, by employing the communication protocol explained in Section 4.2. This necessitates to upload such result to an intermediate web server (playing the role of a "communication repository"), where it will be fetched by the web application that will be pinging the specified server for the expected result.

The adopted architecture results in a number of technical requirements for the web server and the learner device. Indicatively, the web server should have:

- A high-speed internet connection (100 Mbps).
- Sufficient disk space (over 200 GB).
- Window Server 2012 R2 operating system.
- Apache Web Server version 2.4.10 software [30] installed.

In addition, the device available at the learner site should indicatively have:

- A high-quality RGB camera with a frame rate of at least 25 frames per second and a 640×480-pixel frame resolution.
- A sufficient fast CPU (e.g., Intel i7-6700HQ, 2.60 GHz processor) to allow fast computations.
- A sufficiently large SSD hard drive (e.g., 256GB) to allow fast data access and storage.
- An Nvidia GPU with at least 4 GB of memory and 750 CUDA cores (e.g., GeForce GTX 1050 Ti or better), used in accelerating deep-learning based algorithms for visual detection, feature extraction, and GSL recognition.
- A sufficiently fast internet connection (minimum download speed 1 Mbps, minimum upload speed 0.01 Mbps).
- An up-to-date Linux operating system installed (e.g., Ubuntu 20.04 LTS).
- The PyTorch 1.7.1 [33] packages installed.
- The CUDA tookit 10.1 [34] installed.

The above technical requirements of the learner device can be easily met by high-end laptop gaming computers. Such are preferable to desktop ones of similar specifications, as they facilitate portability and sharing among learners.

4 Additional Architecture Details

Having defined the SL-ReDu platform architectural approach, we next provide details on a number of its aspects. Specifically, in Section 4.1 we discuss the educational material representation and corresponding database structure, in Section 4.2 we present the communication protocol and information exchange between the web-based application and the learner device, and, finally, in Section 4.3 we overview the adopted SL recognition pipeline.

4.1 Educational Material Representation and Database Structure

As discussed in Section 3, the educational material of the platform (see also Deliverable D3.2 [21]) needs to be properly encoded in a suitable database for use in the self-monitoring environment and the composition and encoding of tests in the objective evaluation environment. We provide such details next.

In order to display educational material in the self-monitoring module, educational content entries will be structured around a 5-level classification scheme:

- The first level (`level1`) corresponds to the main educational chapters to appear in the table of contents. In this case, `level1=1` refers to the first chapter to be presented in the list, `level1=2` refers to the second, and so on.
- The second level (`level2`) corresponds to the sub-chapters to be included within each chapter of `level1`.
- The third level (`level3`) corresponds to the individual learning units entailed within each item of `level2`.
- In the fourth level (`level4`), content presentation is organized within each individual learning unit, taking three possible values that correspond to: (a) Theory Presentation (`level4=1`), (b) Consolidation (`level4=2`), and (c) Exercises (`level4=3`).
- The fifth level (`level5`) encodings are consecutive and refer to the number of web pages that are to be presented under the Theory Presentation, Consolidation, and Exercises options of `level4`.

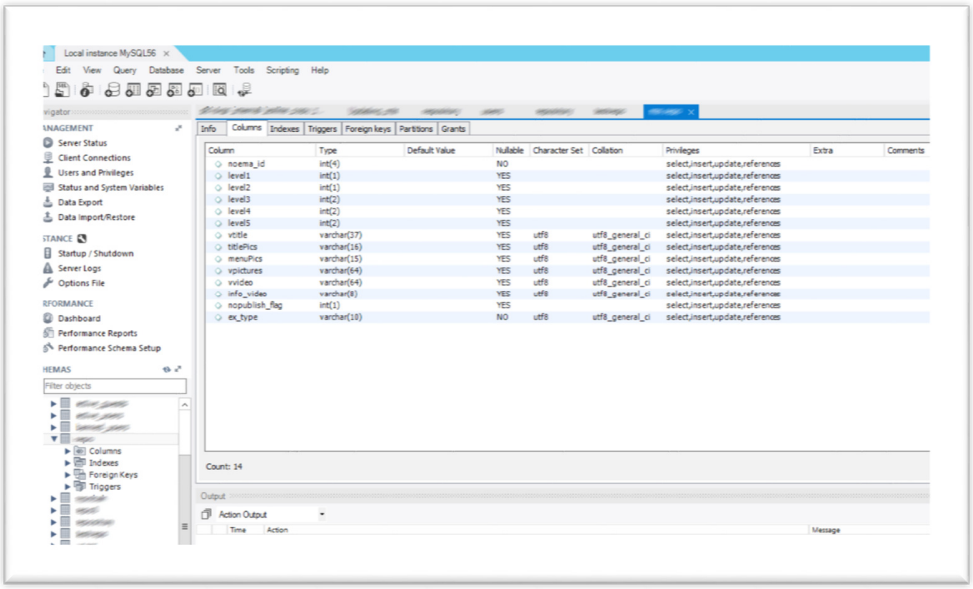

Figure 4: Structure of the test exercises in the objective evaluation module of the SL-ReDu platform.

Table 1 provides a detailed description of the fields used by the web application, when structuring both educational content and the exercises that are presented to the user in the self-monitoring module. The same exercise types may provide the content for composing a test in the objective evaluation module (see also Figure 4).

In Figure 5, we depict a snapshot of the database that accommodates the structure of exercise types (shown in Figure 4) to be exploited for test creation by the instructor in the objective evaluation module. For demonstration purposes, we consider an example of the entry with id = 17, representing a recognition exercise. As depicted in Figure 6 below, if the learner chooses the "slow" over the "fast" execution option (corresponding to test-duration in seconds and defined by values within fields `secs_slow` and `secs_fast` of Table 1, accordingly), he/she is offered the longest defined time to use the camera to record the SL production. In addition, the learner is offered the possibility to submit the recorded video twice, while the score that will be earned by the correct execution of the exercise will be 1.

Field Name	Type	Description
`noema_id`	`Integer`	A unique number that identifies the data entry
`level1`	`Integer`	Corresponds to the main educational chapters in the Table of Contents (e.g., value=1 refers to the first chapter in the list).
`level2`	`Integer`	Corresponds to the sub-chapters to be unfolded under each chapter (second level content).
`level3`	`Integer`	Corresponds to the content sections to be found under each sub-chapter of `level2`.
`level4`	`Integer`	Takes values from 1 to 3, reflecting content organization within each learning unit in the self-monitoring module as follows: Theory presentation (`level4 = 1`), Consolidation (`level4 = 2`), Exercises (`level4 = 3`).
`level5`	`Integer`	Corresponds to the encodings that indicate the number of web pages to be presented under each section of `level4`.
`vtitle`	`Varchar`	A text value that can be used as a title. Within the objective evaluation module, i.e. in a test, this field holds the value of the test title, e.g., "Test in fingerspelling".
`titlePics`	`Varchar`	A text value that can be used as a title of a picture content. It is of no use in the objective evaluation module, thus it remains unset (empty) in this case.
`menuPics`	`Varchar`	A text value that can be used as a title in a menu. It is of no use in the objective evaluation module, thus it remains unset (empty) in this case.
`vpictures`	`Varchar`	This field holds a series of text values that are used by the application in a multiple-choice exercise type. The correct value corresponds to the value of the `vvideo` field.
`vvideo`	`Varchar`	This field holds a text value for the video presented to the user.
`instructions`	`Text`	A text value, containing instructions to the user, to be used in the SL recognition type of exercises.
`EL_expected`	`Varchar`	A text value with the name of the correct item in Greek, expected to be signed by the learner, e.g., "ΜΠΑΛΑ".
`EN_expected`	`Varchar`	The same name as in `EL_expected`, but this time in English.
`feedback_video`	`Varchar`	This field holds a text value of the expected correct answer (the video that should be signed by the user).
`info_video`	`Varchar`	A text value used by the application to accommodate possible explanatory videos with respect to the different content structure levels.
`type_video`	`Varchar`	A text value that represents the type of the video presented to the user, i.e. the video of a human or an avatar, according to the exercise content needs.
`ex_type`	`Integer`	Values that are used only in the case of recognition exercises. This field takes two-digit values, with the first digit representing the type of the exercise (1: active, 2: passive) and the second digit representing the recognition task (1: isolated lemma, 2: fingerspelling...). These values are used internally by the application to serve the video file naming conventions of user input for communication with the recognition module.
`secs_slow`	`Integer`	Integer value representing the total duration of the test in seconds in a "slow-pace" execution of the test. It restricts conducting the test within the defined period.
`secs_fast`	`Integer`	Integer value representing the total duration of the test in seconds in a "fast-pace" execution of the test. It restricts conducting the test within the defined period.
`grading`	`Integer`	A value that holds the points that the user will earn in case of correct answer to the exercise.
`submit`	`Integer`	A value representing the number of times users can submit their answers.

Table 1: Description of the fields used by the web application to encode educational material exercises.

noema_id	level1	level2	level3	level4	level5	vtitle	titlePics	menuPics	vpictures	vvideo	instructions	EL_expected	EN_expected	feedback_video	info_video	type_video	ex_type	secs_slow	secs_fast	grading	submit
13	5	1	1	3	12				giota.gif|gama.gif|"delta.gif"	Delta.mp4								NULL	NULL	1	1
14	5	1	1	3	13				ZISE.png|"EZISA.png"|EZISE.png	EZHSA.mp4						avatar		NULL	NULL	1	1
15	5	1	1	3	14				"lamda.gif"|ni.gif|mi.gif	Lamda.mp4								NULL	NULL	1	1
16	5	1	1	3	15				xi.gif|psi.gif|"ksi.gif"	KSI.mp4								NULL	NULL	1	NULL
17	5	1	1	3	16				(Κάμερα)	ΣΑΣΑ	Νοημάτισε ...	ΣΑΣΑ		SASA.mp4			12	5	4	1	2
18	5	1	1	3	17				"fi.gif"|giota.gif|delta.gif	Phi.mp4								NULL	NULL	1	1
19	5	1	1	3	18				"mi.gif"|taf.gif|lamda.gif	Mi.mp4								NULL	NULL	1	1
20	5	1	1	3	19				vita.gif|"kapa.gif"|ro.gif	Kappa.mp4								NULL	NULL	1	NULL
21	5	1	1	3	20				(Κάμερα)	ΟΦΙΣ	Νοημάτισε ...	ΟΦΙΣ		OFIS.mp4			12	5	4	1	2
22	5	1	1	3	21				alfa.gif|omikron.gif|"sigma.gif"	Sigma.mp4								NULL	NULL	1	1
23	5	1	1	3	22				kapa.gif|psilon.gif|"omega.gif"	Omega.mp4								NULL	NULL	1	NULL
24	5	1	1	3	23				(Κάμερα)	ΑΦΙΣΕΣ	Νοημάτισε ...	ΑΦΙΣΕΣ		APISES.mp4			12	5	4	1	2
25	5	1	1	3	24				(Κάμερα)	ΜΠΑΛΑ	Νοημάτισε ...	ΜΠΑΛΑ		MPALA.mp4			12	5	4	1	2
26	5	1	1	3	25				GIDA.png|"OFIS.png"|FIDI.png	OFIS.mp4						avatar		NULL	NULL	1	1
27	5	1	1	3	26				"XANTHOS.png"|ANTHOS.png|XANTHI.png	XANTHOS.mp4						avatar		NULL	NULL	1	NULL
28	5	1	1	3	27				(Κάμερα)	ΚΑΦΕΣ	Νοημάτισε ...	ΚΑΦΕΣ		KAFES.mp4			12	5	4	1	2
29	5	1	1	3	28				SAS.png|"SASA.png"|ASSO.png	SASA.mp4						avatar		NULL	NULL	1	1
30	5	1	1	3	29				omega.gif|"vita.gif"|psilon.gif	Beta.mp4								NULL	NULL	1	NULL
31	5	1	1	3	30				(Κάμερα)	ΑΚΡΟ	Νοημάτισε ...	ΑΚΡΟ		AKRO.mp4			12	5	4	1	2
32	5	1	1	3	31				OFIS.png|GIDA.png|"FIDI.png"	FIDI.mp4						avatar		NULL	NULL	1	1
33	5	1	1	3	32				(Κάμερα)	ΜΟΝΑΞΙΑ	Νοημάτισε ...	ΜΟΝΑΞΙΑ		MONAXIA.mp4			12	5	4	1	2
34	5	1	1	3	33				(Κάμερα)	Σ	Νοημάτισε ...	Σ	S	Sigma.mp4			12	4	3	1	2
35	5	1	1	3	34				(Κάμερα)	Θ	Νοημάτισε ...	Θ	TH	Theta.mp4			12	4	3	1	2
36	5	1	1	3	35				KOVO.png|KAΒO.png|"KAFES.png"	KAFES.mp4						avatar		NULL	NULL	1	NULL
37	5	1	1	3	36				(Κάμερα)	ΠΑΘΟΣ	Νοημάτισε ...	ΠΑΘΟΣ		PATHOS.mp4			12	5	4	1	2

Figure 5: Sample encoding of objective evaluation exercises in a test.

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

noema_id	level1	level2	level3	level4	level5	vtitle	titlePics	menuPics	vpictures	vvideo	instructions	EL_expected	EN_expected	feedback_video	info_video	type_video	ex_type	secs_slow	secs_fast	grading	submit
13	5	1	1	3	12				giota.gif|gama.gif|"delta.gif"	Delta.mp4								NULL	NULL	1	1
14	5	1	1	3	13				ZISE.png|"EZISA.png"|EZISE.png	EZHSA.mp4						avatar		NULL	NULL	1	1
15	5	1	1	3	14				"lamda.gif"|ni.gif|mi.gif	Lamda.mp4								NULL	NULL	1	1
16	5	1	1	3	15				xi.gif|psi.gif|"ksi.gif"	KSI.mp4								NULL	NULL	1	NULL
17	5	1	1	3	16				(Κάμερα)	ΣΑΣΑ	Νοημάτισε ...	ΣΑΣΑ		SASA.mp4			12	5	4	1	2
18	5	1	1	3	17				"fi.gif"|giota.gif|delta.gif	Phi.mp4								NULL	NULL	1	1

Figure 6: Sample encoding of the objective evaluation exercise with id = 17.

4.2 Communication and Information Exchange

As already mentioned in Section 3, in the case of linguistic production exercises, the adopted architecture necessitates appropriate communication between the server running the web application and the device at the learner-side that is used to record the learner's GSL production and recognize it. Such communication is decided to take place as depicted in Figure 7, among the server, the learner's device, and an intermediate server (website) that operates as a "communication repository" to facilitate it.

As shown in the figure, the designed communication protocol commences with the recording and submission of the learner's video that contains the signed content. The video filename is predetermined by the web application and includes specific parameters, such as user id, assignment id, assignment iteration, and SL task (fingerspelling, isolated, or continuous). After submission, the signing video is saved at the learner's device, where the GSL recognition module runs locally. The module repeatedly pings the download folder to eventually receive the recorded video for processing and, at the same time, a website path is cleared at the intermediate server. The recognition module employs at a first stage appropriate computer vision algorithms (2D pose estimation of the learner [32]) to inspect the relative position of the learner with respect to the field-of-view of the device camera, providing instructions for placement rectification in case of wrong positioning (e.g., when the manual articulators are occluded). If no problems are detected, the video file is further processed and subjected to GSL recognition by an appropriate visual feature extraction pipeline and statistical recognition models that correspond to the specified GSL task (see also Section 4.3). In both cases, the results of this module are uploaded in the form of a JSON file [35] to the intermediate server, carrying information concerning both positioning and

the recognition result. The corresponding filename is predefined to reflect the processed video file name, thus the web application can access the results by pinging the intermediate server, as discussed later.

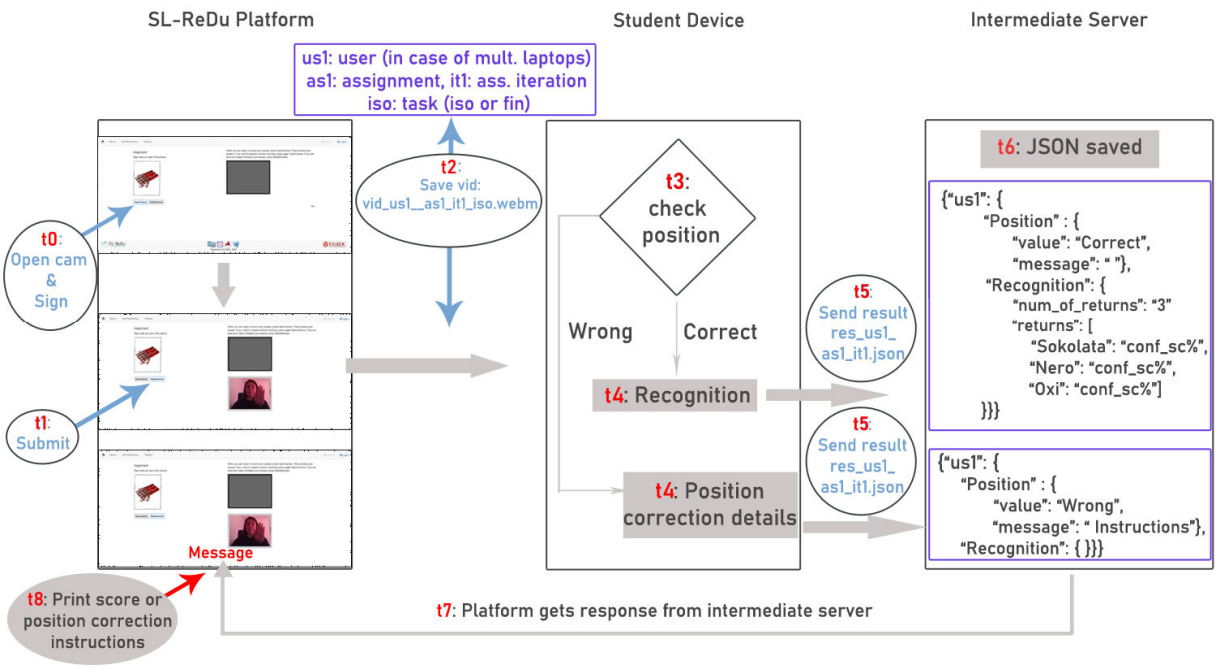

Figure 7: Architecture diagram showcasing the adopted SL-ReDu communication between the web application, learner device, and intermediate server to allow active SL production exercises by the learner.

Concerning the results file format, in case the learner's position with respect to the camera was problematic, it will be as follows:

> `Wrong` (text string, here indicating incorrect positioning)
>
> `$Instructions` (text string, containing instructions to the learner on how to fix his/her position, for example by moving backwards, left, or right, and whether the hands and/or face have been fully visible in the video)
>
> `NULL` (empty text string, signifying that no recognition results are returned due to incorrect learner positioning).

In case, though, the learner's position was deemed acceptable, there will be no positioning correction instructions, but instead the lower part of the file will be populated with an *N*-best list of GSL recognition results and their corresponding confidence scores, according to the following format:

> `Correct` (text string, here indicating correct positioning)
>
> `NULL` (empty text string, since no positioning correction message needs to be displayed)
>
> $\$N$ (integer, signifying the number of recognition results returned in the form of an *N*-best list)
>
> $\{\ \$\mathrm{Rec}_n\ \ \$\mathrm{Conf}_n\ \}_{n=1,\ldots,\$N}$ ($\$N$ pairs of text strings that represent the nth recognition result and floats that provide the corresponding recognition confidence score).

In the first version of the implemented system, only a single recognition result (1-best) will be returned with a default unit confidence, so the results file will be simplified to contain the following information (in the case of correct learner positioning):

```
Correct
NULL
1
$Rec 1.0
```
(where `$Rec` is a text string containing the recognized GSL gloss(es)).

Finally, in Figure 8 we schematically detail the timing of the aforementioned communication among the web-based platform, the learner device, and the intermediate server. As can be observed, the platform becomes aware of the recorded video, when the learner presses the submit button (at time instant t_1), whereas the learner device a bit later (at t_2), once the video is saved. This initiates the recognition process that returns results at time instant $t_5 > t_2$ and immediately uploads the results to the intermediate server. Thus, the results are available at the intermediate server at time $t_6 > t_5$. In the meantime, the web application initiates pinging of the intermediate server after a specific time interval (that is equal to the submitted video duration) and repeats pinging at regular intervals of t_s (t_s: 4 secs) until the JSON file is detected, in which case the file information is received by the web application for parsing and further processing (at time t_7). Note that the adopted file naming convention guards against accessing invalid files from the intermediate server, which are in any case cleared periodically. Specifically, the process that runs in the student device clears the intermediate server's path at time instant t_2 from any files corresponding to the user submitting the video file.

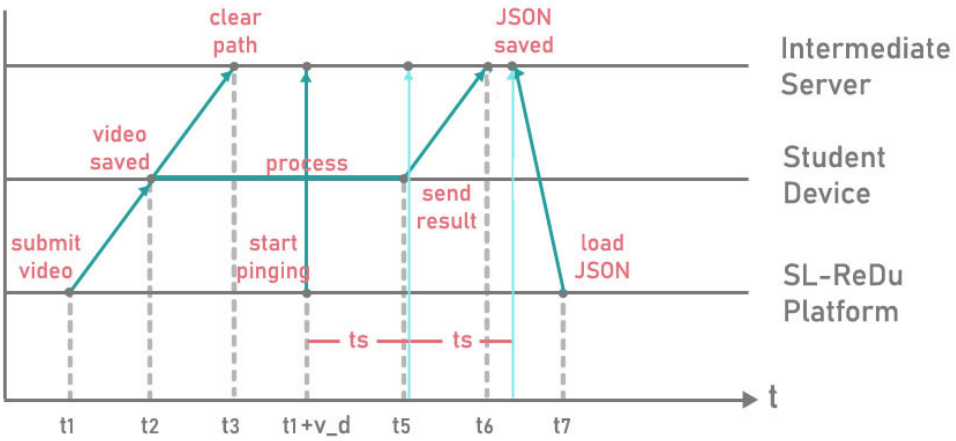

Figure 8: Timing diagram of the communication protocol adopted in the SL-ReDu system. Parameter v_d denotes the submitted video duration, and ts is the pinging interval.

4.3 GSL Recognition Pipeline

A variety of GSL recognition algorithms can be accommodated by the adopted architecture. Given the algorithmic analysis in recently completed Deliverable D2.1 [22] concerning the performance and computational complexity (parameter size) of various models and visual feature streams investigated there, we will proceed with integrating to the SL-ReDu system the bidirectional long short-term memory (BiLSTM) attentional encoder-decoder model, operating on the concatenation of a number of visual feature streams as discussed below and schematically depicted in Figure 9.

In more detail, the recognizer will commence with the detection of the learner's body skeleton via the OpenPose framework [32], which is a deep learning-based human joint detector of the body pose, hands, and face. Specifically, OpenPose provides the 2D location of 137 skeleton joints of the signer in the video frame, namely 25 body-pose joints, 21 keypoints for each hand, as well as 70 facial joints, some of which

can be seen in Figure 9. Since the most dominant SL information involves the hands, we segment the hand regions based on the corresponding skeletal coordinates returned by OpenPose, thus obtaining the two hand regions-of-interest (ROIs), each normalized to 224×224 pixels. We then feed each of the two hand ROIs to a ResNet-18 [36] image feature learner, generating 512-dimensional appearance features. Next, we concatenate those together with the 15 2D normalized human skeletal joints (30-dim skeletal features) of the upper body (excluding the facial and hand points), thus getting 1054-dimensional features that we subsequently feed to the encoder-decoder module for the recognition task.

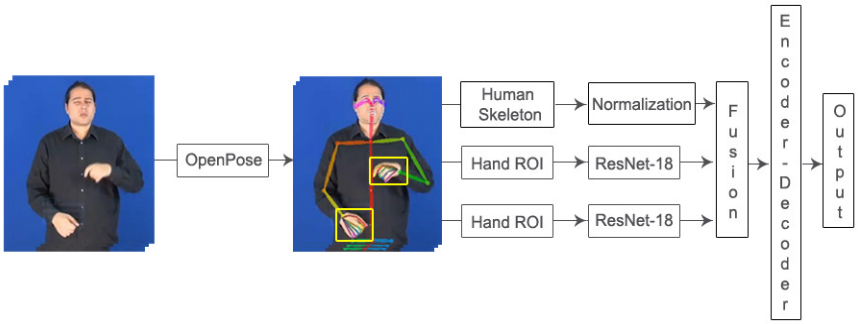

Figure 9: Schematic of the GSL recognizer algorithmic pipeline.

For this purpose, we exploit an attention-based encoder-decoder scheme for the prediction task, motivated by recent work in ASR and machine translation [37, 38]. The model involves an attention-based recurrent neural network (RNN) that relies on the BiLSTM model [39], equipped with an input feeding scheme. In its general form, the RNN encoder-decoder module comprises two processes: encoding and decoding (see also Figure 10). In particular, our RNN encoder is fed with latent representations derived from the first GSL recognizer component generating hidden state representations, but instead of predicting the current state based on the past context as in the case of LSTMs [40], we apply BiLSTMs, where in addition to the previous observations there is also access to the future ones. Namely, one LSTM processes the input sequence in a forward fashion (left-to-right), while the other processes it backwards (right-to-left), and both relate to the same output. These two calculations are concatenated generating hidden state representations. During decoding, the hidden state sequence is processed by the LSTM decoder [40] producing the elements of the output sequence, one by one. Further, encoder-decoder models equipped with attention are based on the alignment between input and output, driven by the "context" vector that expresses the likelihood of each chunk of the source sequence being related to the current output. More precisely, our model comprises a one-layer BiLSTM encoder and a one-layer LSTM decoder, both with hidden dimensionality fixed to 128. Training is carried out via the Adam optimizer [41] with initial learning rate of 0.001 decreased by a factor of 0.3. The alignment scores are computed through the function proposed in [37]. More details can be found in Deliverable D2.1 [22].

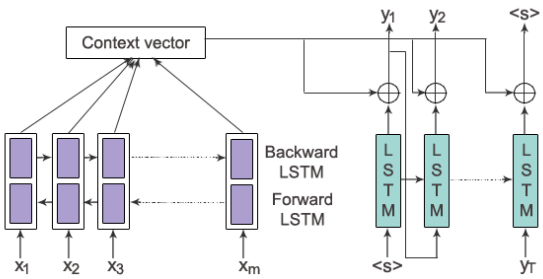

Figure 10: Schematic of the attention-based RNN encoder-decoder used for GSL recognition. The model employs BiLSTMs at the encoder and plain LSTMs at the decoder, with x's denoting the latent representations derived from the feature extraction stage and y's the predicted output sequence (glosses).

5 Conclusions

In this deliverable, we presented the SL-ReDu platform architecture that, together with the design of the human-computer interface (to be unveiled in Deliverable D4.1), will guide the implementation of the first version of the SL-ReDu system. Specifically, after a brief overview of the functional requirements of the SL-ReDu platform, we described a suitable hybrid architecture solution proposed for the technical implementation of the system, which enables meeting these functional requirements, and enumerated indicative technical specifications accompanying the proposed solution. Finally, we provided details of the various aspects of the proposed architecture. The deliverable constitutes part of the second project milestone (MS2), and it will lead to the first version of the system implementation (Deliverable D5.3), as well as its subsequent user evaluation (Deliverable D5.4).

References

[1] B. Berrett, "Using computer-assisted language learning in an American Sign Language course," *Innovation in Language Learning and Teaching*, vol. 6, no. 1, pp. 29–43, 2012.

[2] S.-E. Fotinea, E. Efthimiou, and A.-L. Dimou, "Sign language computer-aided education: Exploiting GSL resources and technologies for web deaf communication." In K. Miesenberger, A. Karshmer, P. Penaz, and W. Zagler (eds.), *Computers Helping People with Special Needs – ICCHP*, Part II, vol. LNCS-7383, pp. 237–244, 2012.

[3] T. Haug, "Web-based sign language assessment: Challenges and innovations", in *Proc. ALTE International Conference: Learning and Assessment - Making the Connection,* 2017.

[4] R. Campbell, P. Martin, and T. White, "Forced choice recognition of sign in novice learners of British Sign Language," *Applied Linguistics*, vol. 13, no. 2, pp. 185–201, 1992.

[5] D. Chen Pichler and H. Koulidobrova, "Acquisition of sign language as a second language," In M. Marschark and P.E. Spencer (eds.), *The Oxford Handbook of Deaf Studies in Language*. Oxford University Press, ch. 14, pp. 218–229, 2015.

[6] E. Efthimiou, S.-E. Fotinea, A.-L. Dimou, T. Goulas, P. Karioris, K. Vasilaki, A. Vacalopoulou, M. Pissaris, and D. Korakakis, "From a sign lexical database to an SL golden corpus – the POLYTROPON SL resource", in *Proc. Workshop on the Representation and Processing of Sign Languages: Corpus Mining (Satellite to the International Conference on Language Resources and Evaluation – LREC)*, pp. 63–68, 2016.

[7] M. Kemp, "Why is learning American Sign language a challenge?," *American Annals of the Deaf*, vol. 143, pp. 255–259, 1998.

[8] L. Pigou, S. Dieleman, P. Kindermans, and B. Schrauwen, "Sign language recognition using convolutional neural networks," in *Proc. European Conference on Computer Vision*, pp. 572–578, 2015.

[9] D. Konstantinidis, K. Dimitropoulos, and P. Daras, "A deep learning approach for analyzing video and skeletal features in sign language recognition," in *Proc. IEEE International Conference on Imaging Systems and Techniques*, 2018.

[10] S. Ko, J. Son, and H. Jung, "Sign language recognition with recurrent neural network using human keypoint detection," in *Proc. Conference on Research in Adaptive and Convergent Systems*, pp. 326–328, 2018.

[11] B. Shi, A. M. D. Rio, J. Keane, J. Michaux, D. Brentari, G. Shakhnarovich, and K. Livescu, "American sign language fingerspelling recognition in the wild," in *Proc. IEEE Spoken Language Technology Workshop*, pp. 145–152, 2018.

[12] B. Shi, A. M. D. Rio, J. Keane, D. Brentari, G. Shakhnarovich, and K. Livescu, "Fingerspelling recognition in the wild with iterative visual attention," in *Proc. IEEE International Conference on Computer Vision*, pp. 5399–5408, 2019.

[13] F. Nugraha and E. C. Djamal, "Video recognition of American sign language using two-stream convolution neural networks," in *Proc. International Conference on Electrical Engineering and Informatics*, pp. 400–405, 2019.

[14] Z. Yang, Z. Shi, X. Shen, and Y. Tai, "SF-Net: Structured feature network for continuous sign language recognition," *Computing Research Repository*, arXiv:1908.01341, 2019.

[15] G. Potamianos, K. Papadimitriou, E. Efthimiou, S.-E. Fotinea, G. Sapountzaki, and P. Maragos, "SL-ReDu: Greek sign language recognition for educational applications. Project description and early results," in *Proc. ACM International Conference on PErvarsive Technologies Related to Assistive Environments,* 2020.

[16] M. Parelli, K. Papadimitriou, G. Potamianos**,** G. Pavlakos, and P. Maragos, "Exploiting 3D hand pose estimation in deep learning-based sign language recognition from RGB videos," *Computer Vision – ECCV 2020 Workshops Proceedings, Part II,* A. Bartoli and A. Fusiello (Eds.), pp. 249–263, LNCS/LNIP vol. 12536, 2020.

[17] K. Papadimitriou and G. Potamianos, "Multimodal sign language recognition via temporal deformable convolutional sequence learning," in *Proc. Conference of the International Speech Communication Association*, pp. 2752–2756, 2020.

[18] N. M. Adaloglou, T. Chatzis, I. Papastratis, A. Stergioulas, G. Th. Papadopoulos, V. Zacharopoulou, G. Xydopoulos, K. Antzakas, D. Papazachariou, and P. Daras, "A comprehensive study on deep learning-based methods for sign language recognition," *IEEE Transactions on Multimedia* (Early Access), 2021.

[19] L. Baischer, M. Wess, and N. Taherinejad, "Learning on hardware: A tutorial on neural network accelerators and co-processors,'' *Computing Research Repository*, arXiv:2104.09252, 2021.

[20] *Common European Framework of References for Languages. Learning, Teaching, Assessment – Companion Volume*, Language Policy Programme, Education Policy Division, Education Department, Council of Europe, 2020 [Online]: https://www.coe.int/en/web/common-european-framework-reference-languages

[21] G. Sapountzaki, E. Efthimiou, and S.-E. Fotinea, "D3.2: Evaluation language material organization," *Tech. Report, SL-ReDu Project Deliverable*, Volos, Greece, 2021.

[22] K. Papadimitriou, G. Potamianos, E. Efthimiou, S.-E. Fotinea, and P. Maragos, "D2.1: First version of GSL recognizer," *Tech. Report, SL-ReDu Project Deliverable*, Volos, Greece, 2021.

[23] *PHP: Hypertext Preprocessor* [Online]: https://www.php.net

[24] *HTML 5.2: W3C Recommendation*, 2017 [Online]: https://www.w3.org/TR/html52

[25] *Descriptions of all CSS Specifications* [Online]: https://www.w3.org/Style/CSS/specs

[26] D. Flanagan, *JavaScript – The Definite Guide: Master the World's Most-Used Programming Language*. 7th ed., O'Reilly, 2020.

[27] *MySQL: The World's Most Popular Open Source Database* [Online]: https://www.mysql.com

[28] *Microsoft SQL Server* [Online]: https://www.microsoft.com/sql-server

[29] *PostgreSQL: The world's most advanced open source relational database* [Online]: https://www.postgresql.org

[30] *Apache HTTP Server Project* [Online]: https://httpd.apache.org

[31] *WebRTC 1.0: Real-Time Communication Between Browsers* [Online]: https://w3c.github.io/webrtc-pc

[32] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 2021.

[33] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *Proc. NIPS-W*, 2017.

[34] *Nvidia CUDA (Compute Unified Device Architecture)* [Online]: https://developer.nvidia.com/cuda-toolkit

[35] *Introducing JSON (JavaScript Object Notation)* [Online]: https://www.json.org

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Computing Research Repository*, arXiv:1512.03385, 2015.

[37] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *Computing Research Repository*, arXiv:1409.0473, 2014.

[38] M. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *Computing Research Repository*, arXiv:1508.04025, 2015.

[39] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[40] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computing*, vol. 9, no. 8, pp. 1735–1780, 1997.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Computing Research Repository*, arXiv:1412.6980, 2014.