



REBUILD

ICT-enabled integration facilitator and life rebuilding guidance

Project start date: 01/01/2019 | Duration: 36 months

Deliverable: D3.6 REBUILD recommendation environment

DUE DATE OF THE DELIVERABLE: 28-02-2021

ACTUAL SUBMISSION DATE: 19-02-2021

Project	REBUILD – ICT-enabled integration facilitator and life rebuilding guidance
Call ID	H2020-SC6-MIGRATION-2018-2019-2020 – DT-MIGRATION-06-2018
Work Package	<i>WP3 – Data Analysis and skills matching</i>
Work Package Leader	<i>Universidad Politécnica De Madrid UPM</i>
Deliverable Leader	<i>Universidad Politécnica De Madrid UPM</i>
Deliverable coordinator	Silvia Uribe Mayoral sum@gatv.ssr.upm.es
Deliverable Nature	Report
Dissemination level	Public (PU)
Version	0.8
Revision	Final

DOCUMENT INFO

AUTHORS

Author name	Organization	E-Mail
David Martín Gutierrez	UPM	dmz@gatv.ssr.upm.es
Silvia Uribe Mayoral	UPM	sum@gatv.ssr.upm.es
Panagiotis Stalidis	CERTH	stalidis@iti.gr

DOCUMENT HISTORY

Version #	Author name	Date	Changes
0.1	David Martín	08-01-2021	Starting version
0.2	David Martín	18-01-2021	Initial Contributions
0.3	David Martín	29-01-2021	Contributions in section 3
0.4	Silvia Uribe	04-02-2021	Section 2
0.5	Silvia Uribe	08-02-2021	Section 4
0.6	Panagiotis Stalidis	16-02-2021	Section 4
0.7	Julia Zomignani Giulia Foresti	18-02-2021	Peer-review
0.8	Silvia Uribe	19-02-2021	Final version

DOCUMENT DATA

Keywords	D3.3 REBUILD recommendation environment and follow-up
Editor Address data	Name: David Martín, Silvia Uribe Partner: UPM Address: Av Ramiro de Maeztu 7 Phone: +34 91 464160 ext. 142 Email: [dmz,sum]@gatv.ssr.upm.es
Delivery Date	28-02-2021
Peer Review	Julia Zomignani (VUB), Giulia Foresti and Anna Lauricella (CIDAS)

ACRONYMS

Acronym	Complete	Meaning
API	Application Programming Interface	A technological tool that permits the interconnectivity of multiple systems
ESCO	European classification of Skills, Competences, Occupations and Qualifications	Multilingual classification that is part of the Europe 2020 strategy

EXECUTIVE SUMMARY

This deliverable is reporting the second phase of the activities of REBUILD consortium to design and create a recommendation scenario based on the information of interest for migrants and aiming at guiding them through the resources available. T3.3 is the last task in WP3 (on top of the others), and is the one that provides the information that the users' will receive, therefore it involves all actors in the integration chain.

D3.6 can be considered as an extension of the previous D3.3, where the design of the recommendation engine was presented, and is devoted to provide a detailed explanation of its development, deployment and integration with the other modules of the system.

Together with the matching module that can be analyzed through D3.5, this recommender is able to provide the required information for building the different use cases that have been defined. Moreover, the integration is mainly based on two different parts: the one related to the user's data gathering and item recommendation presentation through the user interface (mainly via REBUILD app) and the one related to the connection with the matching module, by means of a specific API that provides access to the different services.

With the aim of widely describing these ideas, the document is organised as follows: the first section makes an introduction to this document including general information about the project, the second section is devoted to make a final review of the operational design of the module and the third section is intended to describe the development, especially for the different scenarios. Finally, integration process is presented in section four and section five includes the conclusions.



TABLE OF CONTENTS

Document Info	2
Authors	2
Document History	2
Document Data	2
Acronyms	2
Executive Summary	3
Table of Contents	4
Index of Figures	4
1 Introduction	6
2 Recommender system: final definition	7
2.1 General approach	7
2.2 Recommender operation according to each scenario	8
2.2.1 Job seeking recommendation	8
2.2.2 Social mentoring recommendation	9
3 Recommender system development	10
3.1 Job seeking service	10
3.2 Social mentoring service	12
3.3 Educational training service	13
4 Recommender System Integration	15
4.1 Integration within the REBUILD app	15
4.2 Integration with the chatbot	15
4.3 Integration with the dashboard	18
5 Conclusion	20

INDEX OF FIGURES

Fig. 1. General workflow for recommendation in REBUILD	7
Fig. 2. Customization of the recommendation workflow for the job seeking scenario	9
Fig. 3. Customization of the recommendation workflow for the social mentoring scenario.....	10



Fig. 4. Swagger documentation regarding the Job Seeking Service API. More details can be found at <https://gitlab.com/rebuild-eu/job-seeking-matching-service/-/blob/master/docs/static/openapi.json>11

Fig. 5. Swagger documentation regarding the Job Seeking Service API. More details can be found at: <https://gitlab.com/rebuild-eu/social-mentoring-matching-service/-/blob/master/docs/static/openapi.json>12

Fig. 6. Swagger documentation regarding the Educational Training Service API. More details can be found at: <https://gitlab.com/rebuild-eu/educational-courses-matching-service/-/blob/master/docs/static/openapi.json>14

Fig. 7. Example conversation to call the job seeking scenario recommender through the chatbot.17

Fig. 8. Example response of the chatbot with the most relevant job offers18

Fig. 9. First step of the social mentoring recommendation19

Fig. 10. Matching scores for the suggested pairs in the dashboard20

Fig. 11. JSON response from the recommender for mentor-mentee association.....20

1 INTRODUCTION

This project follows a user-centered and participated design approach, aiming at addressing properly real target users' needs, ethical and cross-cultural dimensions, and at monitoring and validating the socio-economic impact of the proposed solution. Both target groups (immigrants/refugees and local public services providers) will be part of a continuous design process; users and stakeholders' engagement is a key success factor addressed both in the Consortium composition and in its capacity to engage relevant stakeholders external to the project. Users will be engaged since the beginning of the project through interviews and focus groups; then will be part of the application design, participating in three Co-Creation Workshops organized in the three main piloting countries: Italy, Spain and Greece, chosen for their being the "access gates" to Europe for main immigration routes. Then again, in the 2nd and 3rd years of the project, users' engagement in Test and Piloting events in the three target countries, will help the Consortium fine-tuning the REBUILD ICT toolbox before the end of the project.

The key technology solutions proposed are:

- GDPR-compliant migrants' integration related background information gathering with user consent and anonymization of personal information;
- AI-based profile analysis to enable both personalized support and policy making on migration-related issues;
- AI-based needs matching tool, to match migrant needs and skills with services provided by local authorities in EU countries and labour market needs at local and regional level;
- a Digital Companion for migrants enabling personalized two-way communication using chatbots to provide them smart support for easy access to local services (training, health, employment, welfare, etc.) and assessment of the level of integration and understanding of the new society, while providing to local authorities' data-driven, easy to use decision supporting tools for enhancing capacities and effectiveness in service provision.

More specifically, this document provides information of the work performed in WP3 for the final design, development and integration of the recommended system, as an extension of the previous D3.3. In this regard, a set of technical details about the implementation of this service, how the REBUILD recommendation engine will interact via API and the technologies are provided. Finally, this overview of the work done is completed with an explanation about how the integration of the recommendation engine is done for each of the main domains defined for the project, according to their main objectives.

2 RECOMMENDER SYSTEM: FINAL DEFINITION

2.1 GENERAL APPROACH

Although the recommendation operation depends on the selected domain, a general overview can be easily defined. Fig. 1 shows the general pipeline that the system performs to retrieve the recommendations. As can be seen, different modules interact by defining the following stages:

1. As the starting point for accessing the different services, REBUILD app first helps the user to select the specific domain (which corresponds to a use case).
2. After different interactions, the system asks for a recommendation. For obtaining it, the recommender calls the matching service via a specific API.
3. The matching engine retrieves the information for both the user and the items to be recommended for starting the operation.
4. The item and user profiling methods are called in order to provide the embeddings to be used in the matching method.
5. The matching module computes this information and generates the outcome that will be sent to the recommender via the API. This outcome includes the identifiers of the items to be recommended together with their associated scores.
6. The items to be recommended are shown in the application according to these scores in an ordered way.

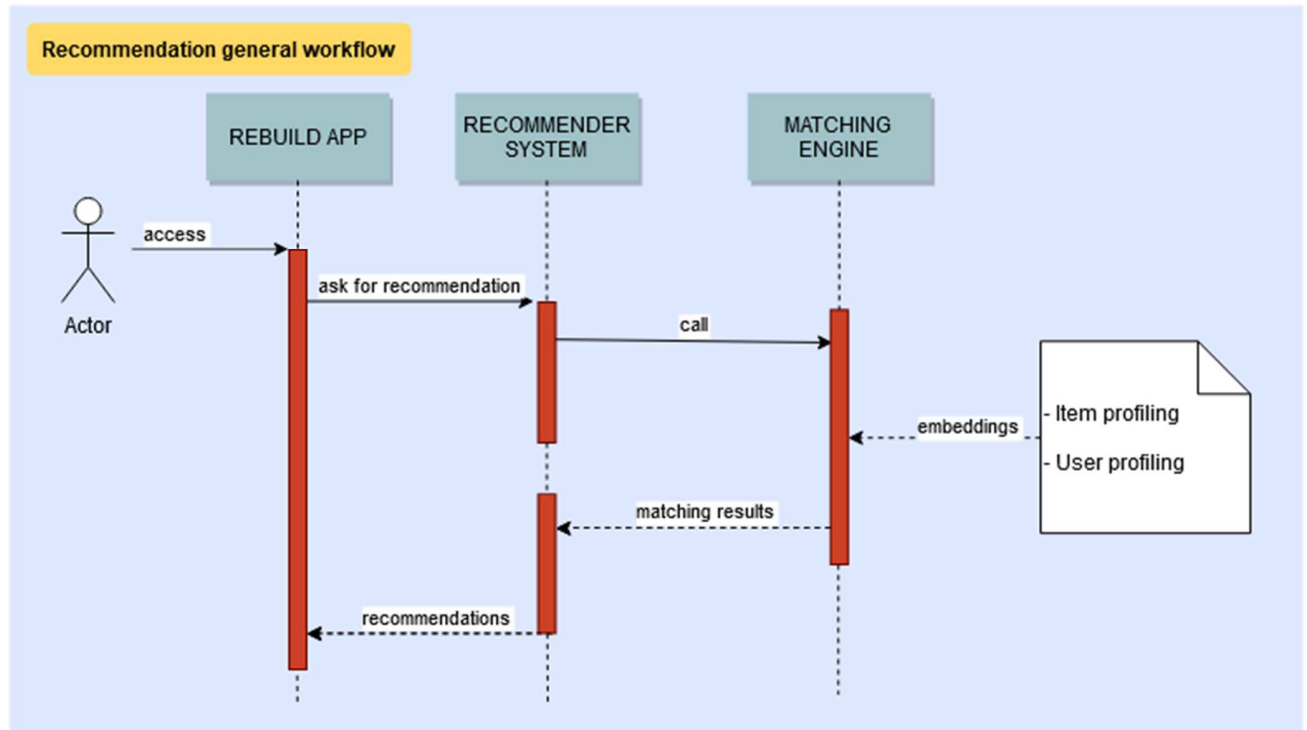


Fig. 1. General workflow for recommendation in REBUILD

Once the general recommendation operation has been explained, next sections show the way the system is customized for each different scenario.

2.2 RECOMMENDER OPERATION ACCORDING TO EACH SCENARIO

As abovementioned, the operation of the entire system with regards to the recommendations depends on the selected scenario. For this reason, next sections are devoted to present these particular workflows, mainly focusing on the job seeking and the social mentoring. The educational scenario can be considered as a similar use case than the former, so it is not going to be described in detail.

2.2.1 JOB SEEKING RECOMMENDATION

In this case, Fig. 2 shows the pipeline for the recommendation flow in the job seeking scenario. The operation is similar to the general one, but there are some aspects that need to be briefly explained:

1. The chatbot can be considered as the specific starting point, since it provides the main interaction tool for the user. In this regard, the chatbot will be in charge of gathering the user's info that the system needs for the recommendation as inputs (different elements such as the work experience, the place of living, if the user has a work permit and the user's skills).
2. After that, the chatbot directly ask for the recommender to provide the outputs.
3. The recommender asks the matching module about the user-jobs matching, according to different parameters (location, skills, etc.).
4. The matching engine asks for the info, computes it and retrieves the results.
5. These recommendations are shown in the application as a typical response of the chatbot.

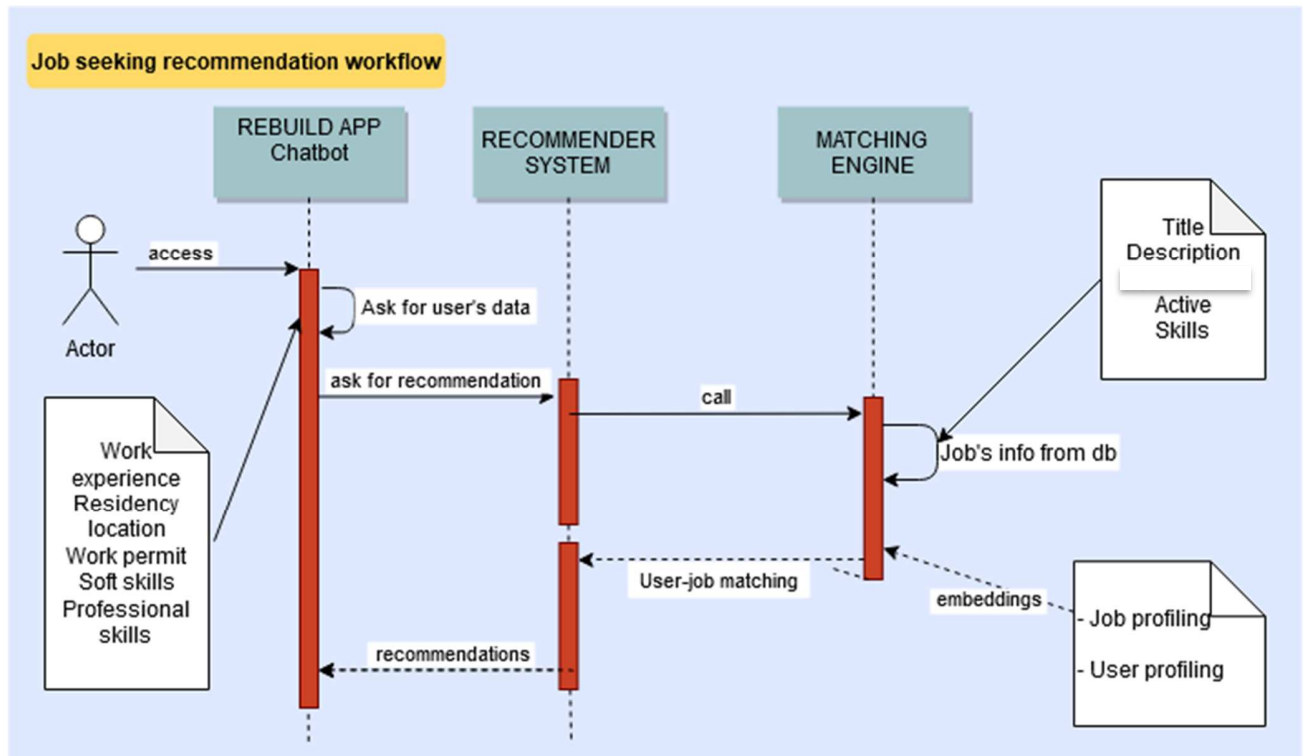


Fig. 2. Customization of the recommendation workflow for the job seeking scenario

The recommendation in the educational scenario will work in a similar way. In fact, the main difference consists in the data that are used for the matching. In this case, the system does not consider the ESCO¹ (European classification of Skills, Competences, Occupations and Qualifications) frame for skill characterization, but directly uses the info about the courses to be retrieved and the capabilities and needs of the users.

2.2.2 SOCIAL MENTORING RECOMMENDATION

Fig. 3 presents the customization of the workflow for this scenario, where some particular modifications can be found. The first one is related to the module that makes the call to the recommender: in this case it is not the chatbot but the dashboard. The second one is the fact that the recommendation about the mentor-mentee matching is not directly provided to the user through the app, but sent to the LSP via the dashboard for its validation.

With regards to the inputs for the matching, they are required through the chatbot via a common conversation, together with the definition of the user as mentor or mentee.

¹ <https://ec.europa.eu/esco/portal?resetLanguage=true&newLanguage=en>

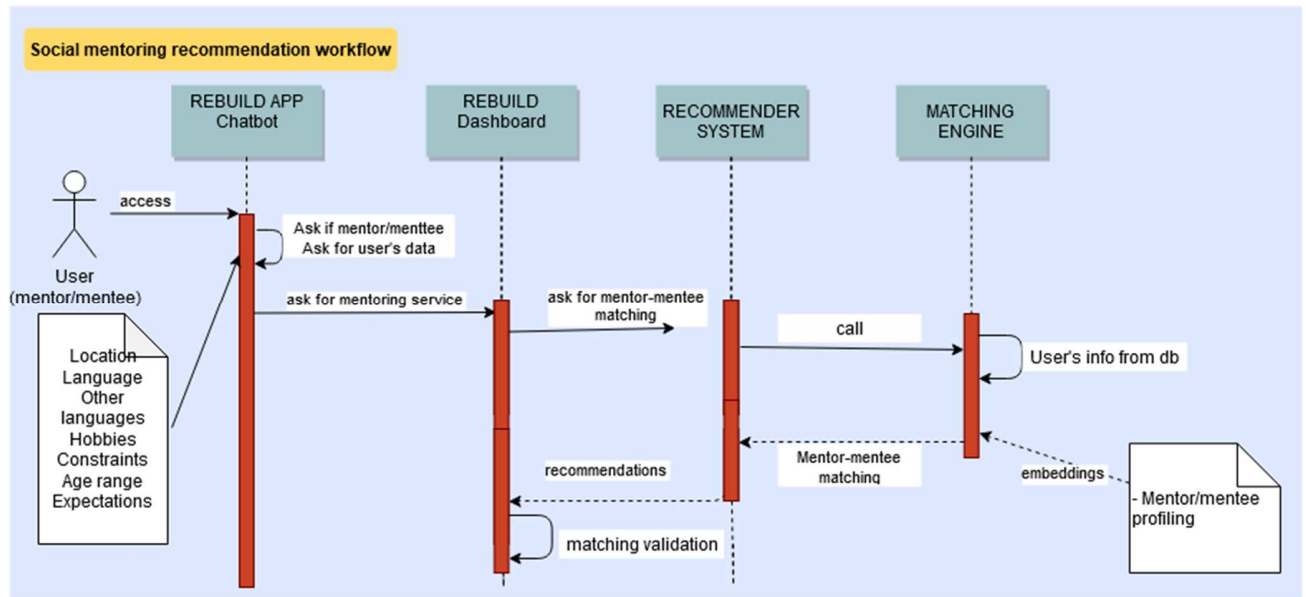


Fig. 3. Customization of the recommendation workflow for the social mentoring scenario.

Once the different operations have been defined and presented, next section is intended to present the recommendation API details that helps the obtaining of the results for each scenario.

3 RECOMMENDER SYSTEM DEVELOPMENT

As explained in D3.5, for each of the available scenarios, a RESTFUL API has been deployed using mainly Python, Flask, Docker. More specifically, Python has been used for programming the matching functionalities whereas Flask has been employed to create the server of the application. Finally, Docker is used to virtualize the application making it easy-to-deploy by automatically creating instances of the application in a virtual environment.

Different endpoints to be called to retrieve the matching results have been defined according to each scenario. In this regard, the recommender engine is the one that generates the requests for retrieving the results that will be presented in the REBUILD app. On the other hand, the matching algorithms are included in all the services as an additional library dependency. In the following sections, the different APIs that have been developed and deployed for the sake of the project will be explained via Swaggers.

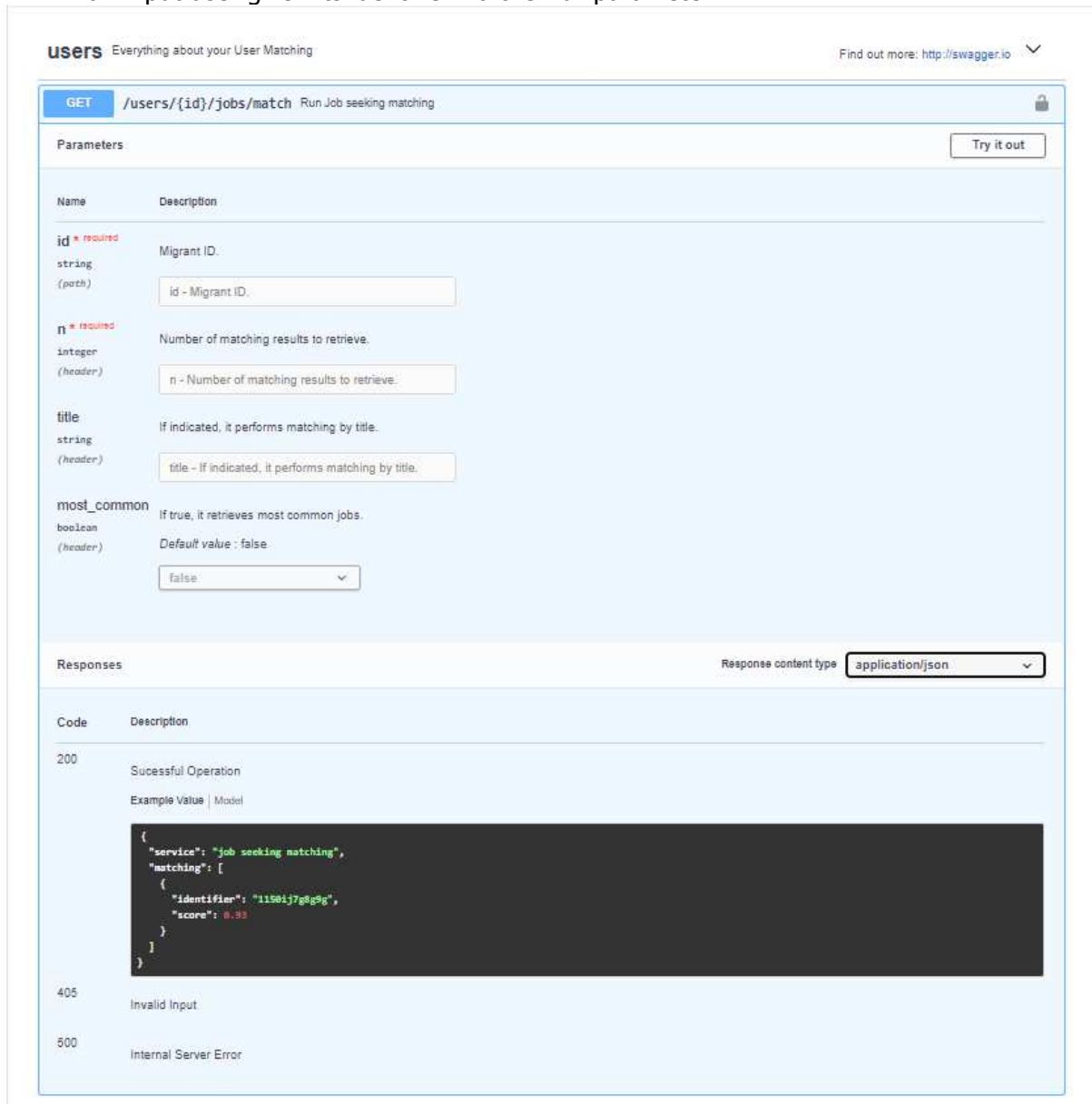
3.1 JOB SEEKING SERVICE

Fig. 4 shows the Swagger with the corresponding endpoints that can be used to call the matching algorithms within the Job Seeking scenario. As can be seen, having as input an identifier of the user (migrant in this case), and the number of results we want to retrieve (n matches), the

matching procedure is performed. Moreover, in case we want to find the most common jobs in the database, the common jobs parameter can be used. On the other hand, if we are interested in finding jobs by title, the API also provides the endpoint with this parameter denoted as title.

The complete development of the service is stored in the Official Gitlab Repository of the REBUILD project at: <https://gitlab.com/rebuild-eu/job-seeking-matching-service>

- **GET /users/{id}/jobs/match** which is employed to generate matchings of jobs for an input user given its identifier via the "id" parameter.



The image shows the Swagger documentation for the endpoint `GET /users/{id}/jobs/match`. The interface includes a "Parameters" section with the following details:

Name	Description
id * required string (path)	Migrant ID. id - Migrant ID.
n * required integer (header)	Number of matching results to retrieve. n - Number of matching results to retrieve.
title string (header)	If indicated, it performs matching by title. title - If indicated, it performs matching by title.
most_common boolean (header)	If true, it retrieves most common jobs. Default value: false. false

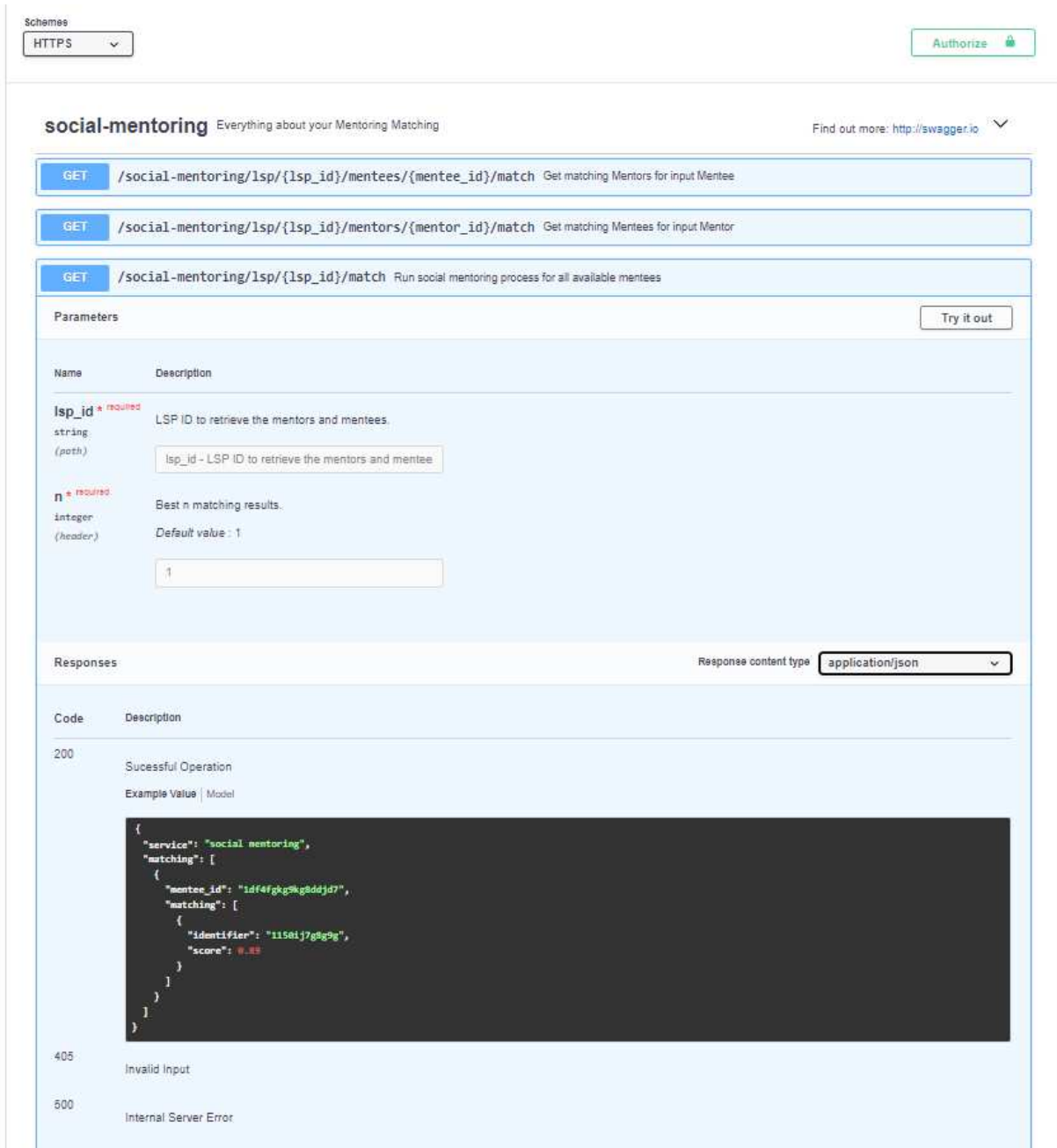
The "Responses" section shows the following details:

Code	Description
200	Successful Operation Example Value Model <pre>{ "service": "job seeking matching", "matching": [{ "identifier": "11581j7g8g9g", "score": 0.93 }] }</pre>
405	Invalid Input
500	Internal Server Error

Fig. 4. Swagger documentation regarding the Job Seeking Service API. More details can be found at <https://gitlab.com/rebuild-eu/job-seeking-matching-service/-/blob/master/docs/static/openapi.json>

3.2 SOCIAL MENTORING SERVICE

In this section, the documentation of the API of the service in terms of endpoints is presented. In this case, the service provides a procedure to generate pairs of mentors-mentees with the information that is currently in the database. The following Swagger remarks the main functionalities that are managed by the API.



The screenshot displays the Swagger documentation for the 'social-mentoring' API. It features a 'Schemes' dropdown set to 'HTTPS' and an 'Authorize' button. The API title is 'social-mentoring' with the tagline 'Everything about your Mentoring Matching'. A link to 'http://swagger.io' is provided for more information.

Three GET endpoints are listed:

- `GET /social-mentoring/lsp/{lsp_id}/mentees/{mentee_id}/match`: Get matching Mentors for input Mentee
- `GET /social-mentoring/lsp/{lsp_id}/mentors/{mentor_id}/match`: Get matching Mentees for input Mentor
- `GET /social-mentoring/lsp/{lsp_id}/match`: Run social mentoring process for all available mentees

The third endpoint is expanded to show its parameters:

Name	Description
lsp_id * <i>required</i> string (path)	LSP ID to retrieve the mentors and mentees. lsp_id - LSP ID to retrieve the mentors and mentee
n * <i>required</i> integer (header)	Best n matching results. Default value : 1

The 'Responses' section shows a dropdown for 'Response content type' set to 'application/json'. The response table includes:

Code	Description
200	Successful Operation Example Value Model <pre>{ "service": "social_mentoring", "matching": [{ "mentee_id": "Id4fgk9kg8ddjd7", "matching": [{ "identifier": "1158ij7g9g9g", "score": 0.88 }] }] }</pre>
405	Invalid Input
500	Internal Server Error

Fig. 5. Swagger documentation regarding the Job Seeking Service API. More details can be found at: <https://gitlab.com/rebuild-eu/social-mentoring-matching-service/-/blob/master/docs/static/openapi.json>

As it is shown in the above Figure, there are three main functionalities:

- **GET /social-mentoring/lsp/{lsp_id}/mentees/{mentee_id}/match** which is employed to generate matchings of mentors for an input mentee via mentee_id.
- **GET /social-mentoring/lsp/{lsp_id}/mentors/{mentor_id}/match** which is employed to generate matching results of mentees for an input mentor using the mentor_id.
- **GET /social-mentoring/lsp/{lsp_id}/match** which is employed to run the matching for all mentors and mentees that are available in the dataset for an input local service provider (LSP).

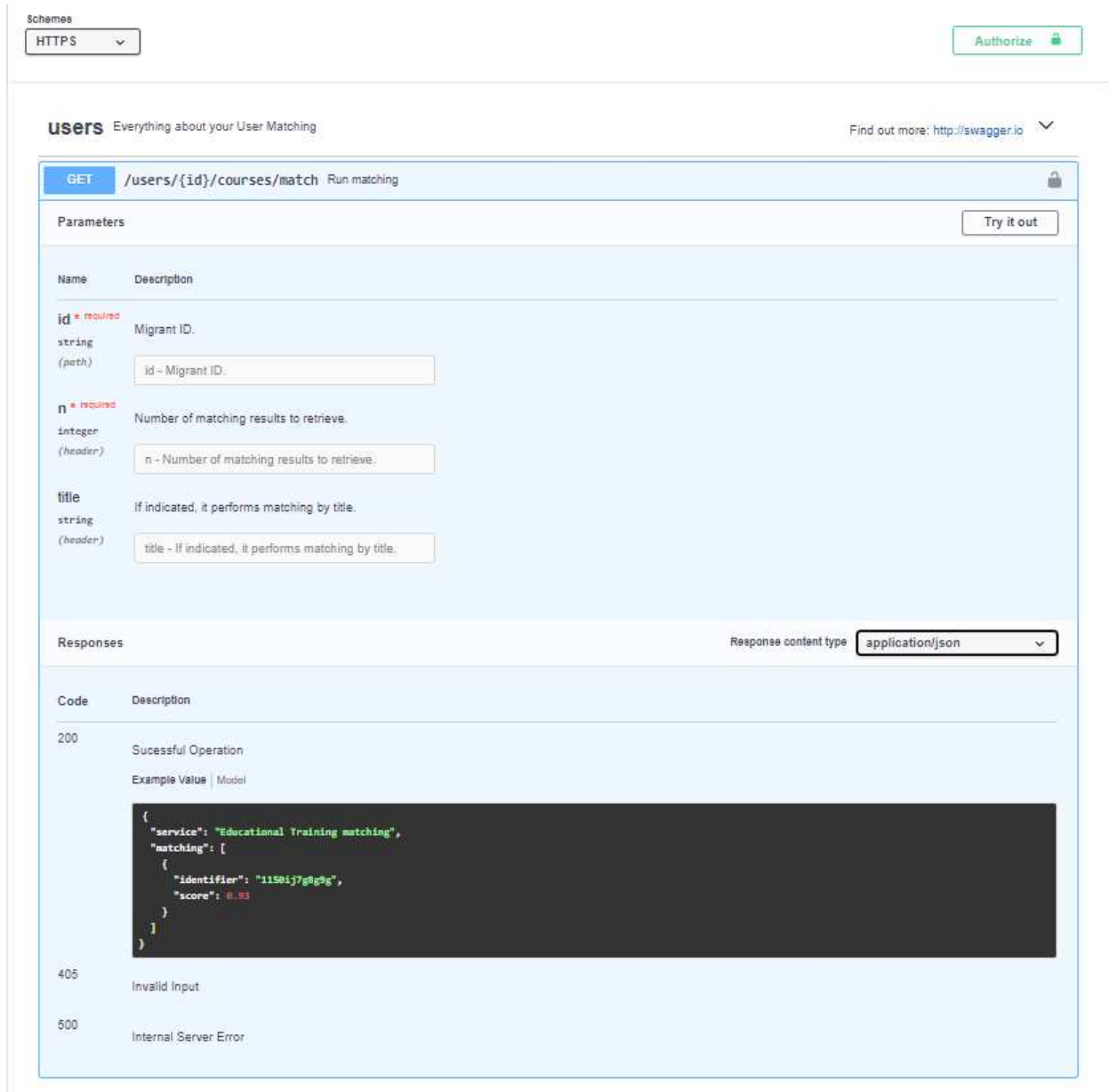
Moreover, the Recommender Engine API calls the last endpoint from time to time in order to produce the whole set of pairs of mentors and mentees that are available at that moment in the premises of the project.

Finally, all the implementation is stored in the Official Gitlab repository of the REBUILD project at: <https://gitlab.com/rebuild-eu/social-mentoring-matching-service>. To access the repository, a permission from its administrator is required.

3.3 EDUCATIONAL TRAINING SERVICE

This API provides an endpoint to retrieve the more adequate Training courses to users according to their needs. It works in a similar way as the Job seeking scenario, since there is a single endpoint that is employed for this precise purpose. The following Figure shows the documentation of the API. More specifically, the following endpoint can be called:

- **GET /users/{id}/courses/match** which is employed to generate matchings of courses for an input user via its identifier. There is also the possibility of filtering courses by title if needed. Moreover, a parameter “n” indicates the number of items it is desired to receive.



Schemes: HTTPS

Authorize

users Everything about your User Matching Find out more: <http://swagger.io>

GET /users/{id}/courses/match Run matching

Parameters Try it out

Name	Description
id * required string (path)	Migrant ID. <input type="text" value="id - Migrant ID."/>
n * required integer (header)	Number of matching results to retrieve. <input type="text" value="n - Number of matching results to retrieve."/>
title string (header)	If indicated, it performs matching by title. <input type="text" value="title - If indicated, it performs matching by title."/>

Responses Response content type: application/json

Code	Description
200	Successful Operation Example Value Model <pre>{ "service": "Educational Training matching", "matching": [{ "identifier": "11501j7g0g0", "score": 0.93 }] }</pre>
405	Invalid Input
500	Internal Server Error

Fig. 6. Swagger documentation regarding the Educational Training Service API. More details can be found at: <https://gitlab.com/rebuild-eu/educational-courses-matching-service/-/blob/master/docs/static/openapi.json>

Finally, all the implementation with regards the Educational Training service is stored in the corresponding repository of the project at: <https://gitlab.com/rebuild-eu/educational-courses-matching-service>

4 RECOMMENDER SYSTEM INTEGRATION

4.1 INTEGRATION WITHIN THE REBUILD APP

According to its own definition, a recommendation engine can be considered as a central point thanks to which the system is able to provide users with the different outcomes from the matching process. For this reason, the integration of this module has a vital importance for the correct performance of the whole chain.

As can be seen in Section 2, there are two main operation scenarios that match to the main domains that have been considered:

- On one hand, the recommendation that can be directly provided to the users since it does not need any kind of previous approval. This is the case of the job seeking and educational scenario, where the matching process can be done thanks to the user's info obtained by the chatbot and the item's info from the database (previously provided by a LSP). We could call this an unsupervised recommendation process.
- On the other, the recommendation that is used for a more complex process such as the one related to the mentor-mentee matching, where the LSP are required to provide a response based on this information. According to this, we can consider this as a supervised method.

Having this in mind, the integration of this module follows two different approaches:

- For the first case, the recommendation engine is directly called from the chatbot and the results are also directly presented to the user through the app as they reach the user's mobile.
- For the second case, there is a connection through the dashboard, which in charge of calling the recommendation engine when needed. Moreover, the obtained recommendation is also previously processed by the LSP before giving an answer to the user.

Next sections are in charge of depicting these two main operational methods.

4.2 INTEGRATION WITH THE CHATBOT

In the REBUILD project the offered services are designed to interact with end users through an automated chatbot interface for the migrants and through the dashboard for the LSPs. From the three services that use the recommender system, the Job Seeking service is used to provide the

migrants with matches between their skills and competences and available jobs in the area that they live, the Education Training service is used to offer migrants matches between their missing skills and the available courses, while in the Social Mentoring service it is used to automatically match candidate mentors with candidate mentees instead of the LSP.

Therefore, the chatbot is integrated with the recommender system only for the first two services. The basic steps for integrating the chatbot with the recommender system are the same for both services:

1. The chatbot gathers the necessary information and updates the user's profile.
2. Then it calls the recommender to perform matchings.
3. Finally, it presents the matches to the end user in a language that can be easily understood.

The first step, of gathering the information, is defined in the relevant conversation tree that is followed by the chatbot for each scenario. The main information that is needed for both scenarios is the previous work experience of the migrant and the additional skills and competences that the user has. Full details about the conversation trees and the information that is gathered for each scenario are defined and presented in deliverable D3.4.

In the second step, the recommender system is called to return the best matching job offers based on the user's skills. The call is performed using the provided API endpoint and giving as parameters the user id and the maximum number of recommendations to be returned. For the first piloting phase, all the calls are performed to return up to 10 recommendations.

Alternatively, a call can be performed to return the 10 best matching job offers to a specific job title, even if the user does not have the skills for this type of occupation. This call is performed by adding a parameter of the job title to be matched. Finally, a third option is offered to call the endpoint and retrieve the 10 most common job titles that are offered. The buttons for the user to select which type of search should be performed, can be seen in the example conversation of Fig. 7. All the calls to the recommender system are constrained to the area that the user currently resides.

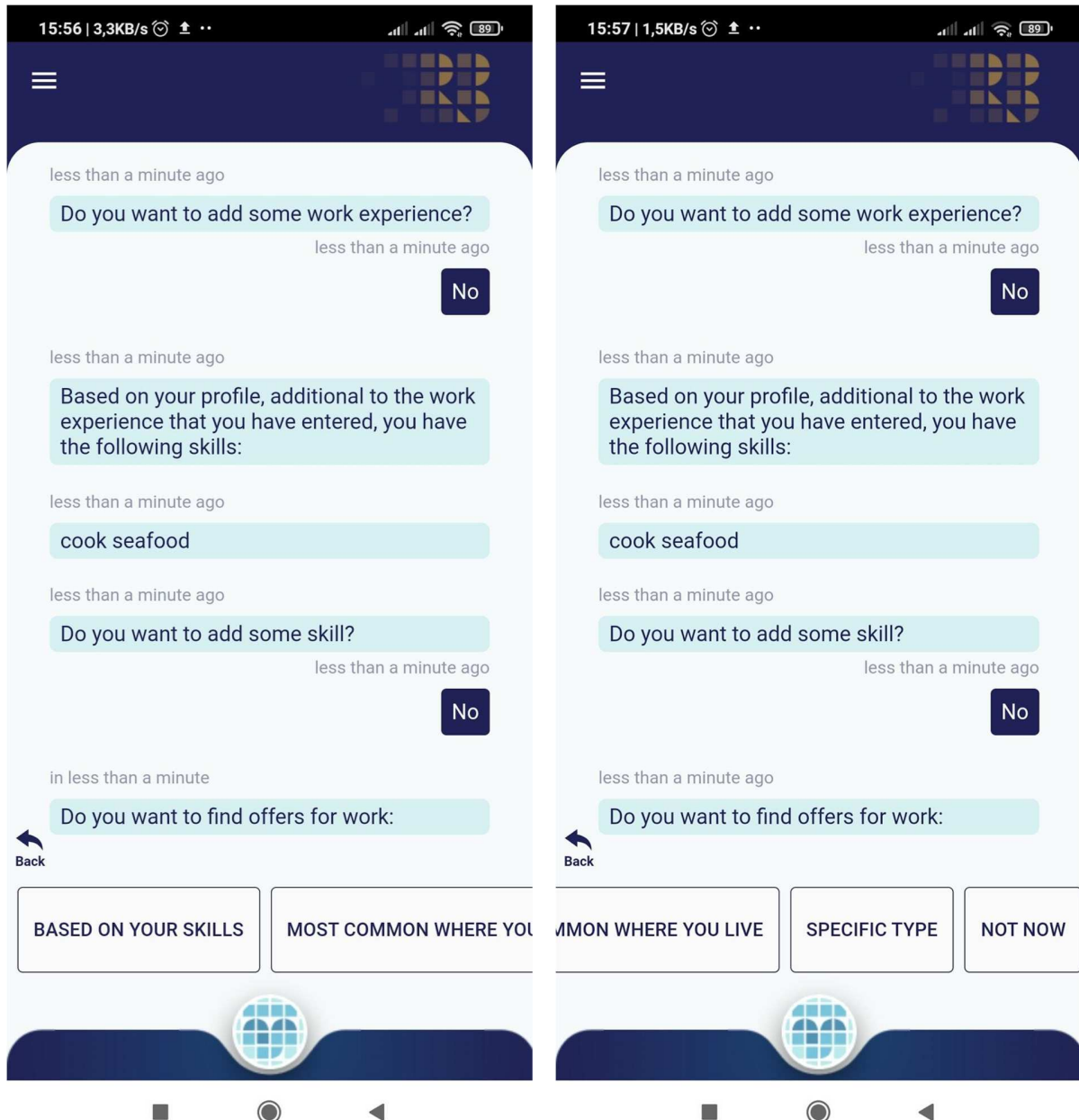


Fig. 7. Example conversation to call the job seeking scenario recommender through the chatbot.

The third step is to present the end user with the retrieved matches. The recommender returns a list of job offer ids and the percentage of matching, that the offer has, to the criteria used. Of course a list of ids is something that is not useful to the user. For each job offer id that is returned by the recommender, a call is made to the job offers' endpoint to retrieve more information about the job offer. The message that is presented to the user includes the name of the company that offers the job, the actual job title and the name of the LSP that should be contacted to get more details about the job that is offered. An example message from the chatbot can be seen in Fig. 8.



Fig. 8. Example response of the chatbot with the most relevant job offers

4.3 INTEGRATION WITH THE DASHBOARD

This second integration is based on the recommender-dashboard direct communication. According to Fig. 9, when a member of the specific LSP involved in this scenario push “Create” button on the dashboard, a call to the recommender is sent.

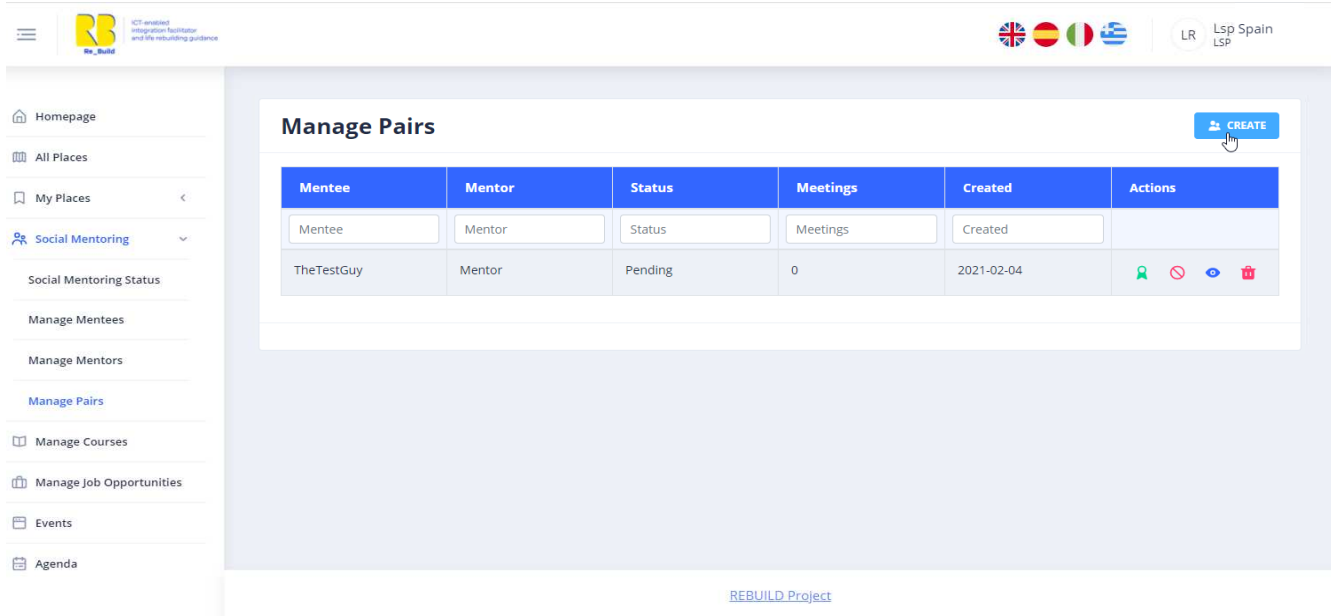


Fig. 9. First step of the social mentoring recommendation

The response of the recommender includes the list of the suggested pairs for the matching, together with each associated score, as can be seen in Fig. 10. Once this is obtained, it is important to say that these pairs need to be reviewed and confirmed by the operator before telling the users through the app.

Finally, Fig. 11 shows the JSON file with the response from the recommender. As it can be seen, it contains the identifier of the mentee together with the identifiers of the mentors and their associated scores for the matching.

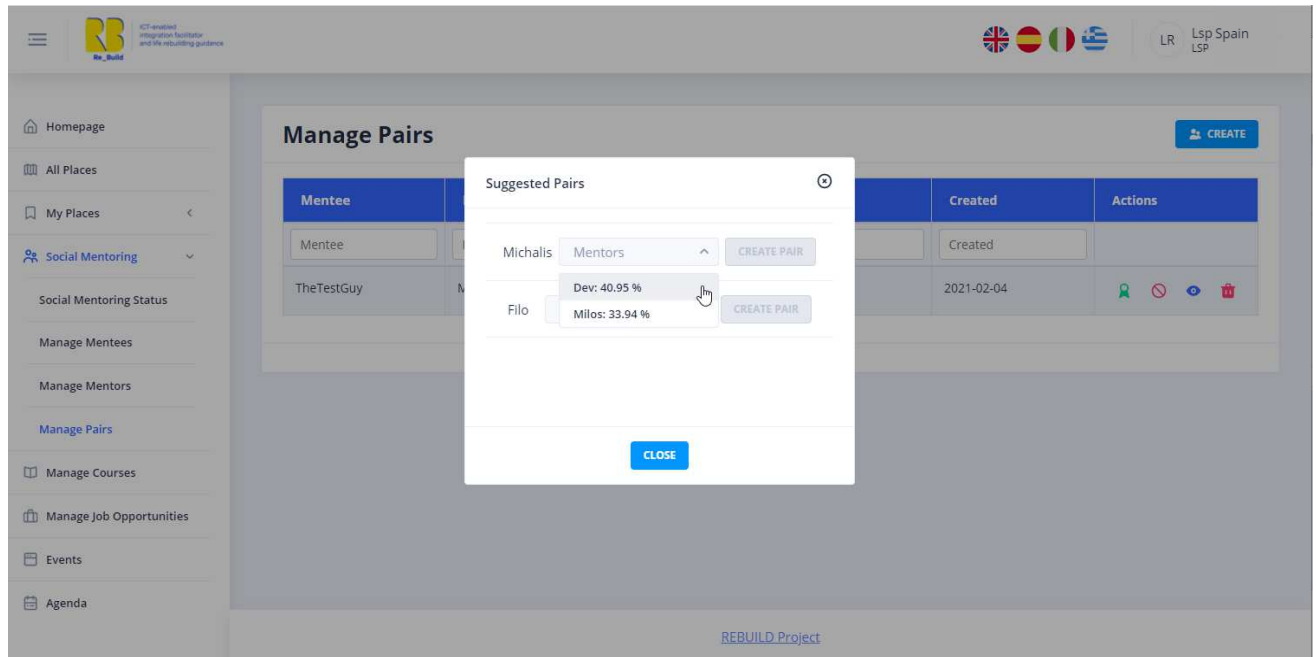


Fig. 10. Matching scores for the suggested pairs in the dashboard

```

{
  "message": "OK",
  "status_code": 200,
  "recommendations": {
    "service": "",
    "matching": [
      {
        "mentee_id": "60129c3da85f1800016619bc",
        "matching": [
          { "identifier": "6012a40ca85f1800016619e3", "score": 0.4248387096774193 },
          { "identifier": "60129da8a85f1800016619bf", "score": 0.40951612903225806 },
          { "identifier": "60129c27a85f1800016619bb", "score": 0.33935483870967739 }
        ]
      },
      ...
    ]
  }
}

```

Fig. 11. JSON response from the recommender for mentor-mentee association

5 CONCLUSION

This deliverable includes a wide description of the recommendation engine implementation, which was the main objective of T3.3 from WP3. In fact, while D3.3 was devoted to present the more relevant techniques to be employed when performing recommendations as a starting point of the task, this document presents the final approach that has been followed.

For doing so, the deliverable describes the API provided by the recommender which was defined for being used both as a standalone application or as a component for the platform as in this



case. Moreover, a complete description of the general and specific workflows has been included. Finally, all the details about the integration within the REBUILD system are also provided, especially for the different domains.



ICT-enabled
integration facilitator
and life rebuilding guidance

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 822215



REBUILD

ICT-enabled integration facilitator and life rebuilding guidance

Deliverable: REBUILD recommendation environment and follow-up



This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 822215.