Output 2 – WP1

MAPPING OF FUNCTIONALITIES OF THE DIGITAL ENTRY POINTS TO EWP

Project Number: 2020-1-TR01-KA203-093849

ERASMUS+PROGRAM – KEY ACTION 2: STRATEGIC PARTNERSHIPS FOR HIGHER EDUCATION 2020

Co-funded by the Erasmus+ Programme of the European Union

Introduction

This study is prepared under the framework of the SUDTE project which seeks to facilitate the digital transformation of HEIs in line with the European Student Card initiative and to contribute to streamlining the gradual shift towards fully digital mobility management. The project will enable exchange of know-how between the participating institutions and ensure extensive support in achieving a smooth and well-coordinated digital transformation process in compliance with ESCI requirements.

In our work where the functionality of the EWP Network is tested, it is necessary to start with defining APIs. API stands for Advanced Programming Interfaces, ie services (mentioned on GitHub) that can be used by network participants to exchange data. A current list of APIs available under the EWP is available at https://developers.erasmuswithoutpaper.eu. To briefly mention how how it works here, EWP Network has a Development environment (DEV) and a Production Environment (PROD). It can be used to test environments, but a number of contracts must be signed.

The list of network-supported services (APIs) known to higher education institutions is available in the "HEI/API Coverage Matrix" on the EWP Registry page. In this study, it is aimed to prepare a guide especially for IRO employees who do not have technical knowledge on complex IT issues. Accordingly, the meaning of the APIs in the HEI/API Coverage Matrix, what functions they have and the deadlines determined by the European Commission are summarized in the table below.

EWP REGISTRY *(HEI/API Coverage Matrix)*			
1. Primary Network APIs (State of the Network connection)			
API	***Definition***	***Functions***	***Deadlines***
a) Discovery	is the first API that the partner developer needs to implement in order to become a basic member of the Erasmus Without Paper Network. Each host which wants to post the supported APIs should implement the Discovery API (manifest file) and send the URL of this file to the Registry.	Discovery manifest files serve to announce which HEIs your system covers, which features (APIs) you have implemented, and which credentials your clients are going to use when fetching the data from the EWP.	In accordance with the EU commission announcement below these API's must be implemented in 2021 to be able to use other functions such as IIA and OLA.
b) Echo	EWP Echo API might seem trivial in itself, but it requires EWP developers to design and test the authentication and security framework which they will use throughout the development of all the other EWP features.	To be able to send the data via the Network, the host should implement at least basic network security protocols (more advanced ones will be needed for the exchange of data with more security demanding partners). It is RECOMMENDED for all developers to implement it (and keep it updated) at least in their development EWP Hosts, to avoid potential problems in the future. It also familiarizes developers with the way EWP APIs are documented (many important parts are documented in XSD files!).	
2. General Purpose APIs (Getting acquainted with each other)			
a) inst.	Institutions API allows sharing of details about the organisation. Institutional API will make possible to access data which is also usually accessible to the anonymous public by other channels. The data includes structure of the organizational units, contact person, address, Erasmus code, web site address, logo and fact sheets with information directed towards incoming students. Lates Realese of Institutions API is version 2.2.0	When Institutinal APIs implemented by the host, it allows external clients to retrieve general information on institutions either covered, or otherwise known, by this host. The data about the institution and the fact sheets for incoming students will be delivered to the partner via EWP Institutions API. It is recommed for server implementers to also provide some basic information on all external institutions which they refer to (by their SCHAC IDs) in other EWP APIs.	This API must implement in 2021
b) ounits	The concept of known organizational Units API is exactly the same as the concept of known institutions. Organizational Units API allows sharing of details about the organization and getting relevant information about other network participants. The data includes faculties, departments, divisions of the organisation.	Organizational Units API can be used to get data about the units. The information about organisational units will be delivered to the partner via EWP Organizational Unit API. When Organizational Units API implemented by the host, it allows external clients to retrieve general information such as units faculties, departments and divisions on selected organizational either covered, or otherwise known by this host. It responds with similar type of information as Institutions API does, but on a lower level.	This API must implement in 2021.

	Lates realese of Institutions API is version 2.2.1.	Each unit is identified by two values such as ID of the institution, and the ID of the organizational unit within the institution.	
c) courses	This document describes the Courses API. This API is not directly related to EWP mobility APIs, but it might help with building user-friendly mobility client software. It allows other HEIs to access information on courses and other learning opportunities conducted in this institution. This API is named "Courses API", but what it really serves are entities called "Learning Opportunities. Server implementers may choose to expose any number of Learning Opportunities via the Courses API.	Thanks to the Courses API, the courses of the student who move between the sending institution and the receiving institution will be matched. The courses which higher education institutions offer to incoming students from the partner institutions can be matched via Courses API on EWP platforms. This process was causing a workload that took quite a bit of time in paper-based operation. Thanks to Courses API, information about students' courses and matching process can be realized in a secure data processing environment. If you are not interested in courses of your partners or do not plan to post your educational offer on the net, you may skip these APIs.	This API must implement in 2021.
d) course replic.	This API can be implemented by any HEI, even it is does not take part in EWP mobility process. Once implemented, it allows the clients to replicate the catalogue of courses conducted on this HEI. This in turn allows the clients to design rich course searching user experience. As it has been explained in the Courses API Course is just one of the types of learning opportunity. This API may expose all types of learning opportunities, not only courses.	Once you implement this API, you effectively allow your requesters to download the entire listing of your programmes and courses, and keep it synchronized later on. This is probably a good idea from a business viewpoint (because your HEIs' course catalogues will be easier discovered in external systems), but it's necessarily so good an idea for your servers. Depending on EWP's popularity and the number of LOS objects in your system, implementing this particular API MAY introduce a significant load on your servers - primarily via the Courses API, not this one (especially if you decide to make these two APIs available anonymously). This API (as well as the Courses API), provides data which is also usually accessible to the anonymous public by other channels. Keep in mind, that this functionality is not essential for EWP's mobility workflow, and you are free to "skip" this API if you're afraid of performance issues. It is also perfectly okay to first make it available, and then change your mind later (if it proves troublesome).	This API must implement in 2021.
3. Interinstitutional Agreements API			
a) Interinstitutional Agreements API	This API allows partners to compare their copies of interinstitutional Erasmus+ mobility agreements with each other, which makes it easier to spot errors. This API is complementary with the Interinstitutional Agreements Approval API where HEIs can approve agreements they exchange via the IIAs API.	The most important feature to understand about this solution, is that HEIs are not required to neither serve nor use this API. This API is not part of the primary mobility flow modeled in EWP. You can still exchange information on Nominations and Learning Agreements without implementing this API. It serves only as a helper API to spot differences between IIAs. If you choose to implement it, then you should probably implement it after all the other APIs. If we expose our agreements to the other partner via an API, then the partner will be able to compare the contents more easily, and possibly find differences in an automated way. In the future, when new agreements are	Academic year 2021/2022 is a transitionary year in which progressively all HEIs join the Erasmus Without Paper infrastructure in order to be able

		forged, it might also enable the partner to copy the agreement's data directly from one computer system to another, without the need of typing it by hand.	to renew IIAs digitally.
	This API allows HEIs to approve agreements sent by their partners in the Interinstitutional Agreements API.	Data on the terms of agreement that needs to be approved by both partners is part of this API. The approval is done in the IIAs Approval API.	
4. Outgoing Mobility			
a) Outgoing Mobilities API (Advanced Programming Interfaces)	This API is implemented by the sending institution. It allows the receiving HEI to read, write and enumerate mobilities stored on the sending HEI's servers.	Negotiations on the list of nominations (sending institutions nominate students for outgoing mobility) are carried out via the EWP network using the Outgoing Mobilities API. The sending institution exchanges a list of nominated students with the receiving institution to get the approval via Outgoing Mobilities API. Note that the number of students and their study fields should be compliant with the cooperation conditions from the Interinstitutional Agreement (IIA). Currently this API describes mobilities of one type only - Student Mobilities for Studies (SMS). This API handles data which is considered private. Server implementers are allowed to forbid less-secure methods of authentication and encryption for this API.	In 2023
b) Outgoing Mobility Learning Agreements API	This API is implemented by the sending institution. It allows the receiving HEI to read and accept Learning Agreements stored on the sending HEI's servers and propose changes to them.	Negotiations on the courses should be carried out via the EWP Network, leading to a list of courses approved by all the parties. If the HEI implements both the Outgoing Mobilities API and the Outgoing Mobility Learning Agreements API, then it must ensure that every learning agreement object will have the same identifier as the corresponding outgoing mobility object served by the Outgoing Mobilities API. In the context of the Erasmus Dashboard, HEIs can also deal with learning agreements via the Online Learning Agreement (OLA) tool. This API is based on the new LA template that will be formally published by the European Commission with the first call of the Erasmus+ program 2021-2027. Also this API handles data which is considered private, so server implementers are allowed to forbid less-secure methods of authentication and encryption for this API	In 2023

c) Outgoing Mobility CNR API	This API is implemented by the receiving institution if it wants to be notified whenever mobilities kept on their partner institutions' servers are changed.	Some APIs have related CNRs (Change Notification Requests). A CNR is used for letting partner institutions know that the Outgoing mobility objects have changed into related system and that the partner (if interested in the changes) should request the changes by calling the corresponding APIs. CNRs do not carry data, they are simple change notifications (like SMSs sent from one system to the other). Requests MUST be made with HTTP POST method. Servers MAY reject all other request methods. This API does not expose any sensitive data, it only notifies the server that it should reload portions of its data. For this reason, it is recommended for server implementers to not be overly strict on security methods they require.	In 2023
d) Outgoing Mobility Learning Agreement CNR API	This API is implemented by the receiving institution. Every modify and/or changes about the students' learning agreement kept on partner institutions' servers creating a notify.	This API supports one function, by which a receiving HEI can communicate a list of identifiers of Outgoing Mobility objects for which Learning Agreement have been recently updated or created. The CNRs allow all partners to wait for a notification and then pull the data.	In 2023
5. Incoming mobilities APIs			
a) Incoming mobilities API	This API is implemented by the receiving institution. It allows the sending HEI to read the receiving HEI's information related to the sending HEIs' outgoing mobilities. (From the receiving HEI's perspective, these are the incoming mobilities.)	The API contains one endpoint that allows the client to retrieve information on specific incoming mobilities from the receiving HEI. Requests must include a list containing the identifiers of the mobilities which the client wants to retrieve information on. Currently, the API describes one type of mobilities only: Student Mobilities for Studies (SMS). It is noted in the Git repository that more types MAY be added in the future. Server implementers choose which security methods they support by declaring them in their Manifest API entry.	2023
b) Incoming Mobility CNR API	This API is implemented by the sending institution if it wants to be notified whenever incoming mobilities kept on their partner institutions' servers are	The API supports one function, by which a receiving HEI can communicate a list of identifiers of Outgoing Mobility objects that have been recently updated on its side.	2023

	changed. CNR stands for “Change Notification Receiver”.	Server implementers choose which security methods they support by declaring them in their Manifest API entry. Since the API does not expose any sensitive data (it only notifies the server that it should reload portions of its data), it is advised that the security methods be not too demanding. It is not guaranteed that all notifications will be delivered promptly, and some notifications may not be delivered at all (e.g. due to implementation errors on the calling institution's server). The sending HEI should periodically verify if its copies are up-to-date.	
6. Incoming Mobility ToRs			
a) Incoming Mobility ToRs API	This API is implemented by the receiving institution. It allows the sending institution to retrieve Transcripts of Records issued by the receiving institution for a given set of mobility IDs.	The API defines two operations: get and index. The get endpoint allows the client (usually the sending HEI) to retrieve Transcripts of Records for specific Incoming Mobilities from the receiving HEI. The index endpoint allows the sending institution to access a list of all mobility IDs for which the receiving institution has already attached corresponding ToRs, and which the caller can read (via the get endpoint). Implementers choose which security methods they support by declaring them in their Manifest API entry. The API handles data which is considered private, and so implementers are allowed to forbid less-secure methods of authentication and encryption.	2023
b) Incoming Mobility ToR CNR API	This API is implemented by the sending institution if it wants to be notified whenever Transcript of Records served by the receiving institution are changed. CNR stands for “Change Notification Receiver”.	The API supports one function, by which a receiving HEI can communicate a list of identifiers of Outgoing Mobility objects for which Transcripts of Records have been recently updated (or created) on the caller's side. Server implementers choose which security methods they support by declaring them in their Manifest API entry. Since the API does not expose any sensitive data (it only notifies the server that it should reload portions of its data), it is advised that the security methods be not too demanding. It is not guaranteed that all notifications will be delivered promptly, and some notifications may not be delivered at all (e.g. due to implementation errors on the calling institution's server). The sending HEI should periodically verify if its copies are up-to-date.	2023

Sources: https://wiki.uni-foundation.eu/download/attachments/1149190/WP%202%20Functionality%20of%20the%20EWP%20Network%20%281%29.pdf?version=1&modificationDate=1598870028909&api=v2