

| Logo  Description automatically generated |
| --- |
|  |
| Task 2.5 - Student housing Web-application  **Peter Höring**  27/04/2023 |
|  |

Table of Contents

[**1. Description of the project 2**](#_heading=h.3znysh7)

[Introduction 2](#_heading=h.1t3h5sf)

[**2. Solution description 3**](#_heading=h.2ghrf993h38)

[2.1. Actors of the System 3](#_heading=h.zqfe328yo1l)

[[A1] Students 3](#_heading=h.xw7j34s8t4id)

[[A2] Student Service Providers – SSP 3](#_heading=h.yek6j9hx4qas)

[2.2. Architecture 3](#_heading=h.8hukgqmlysdy)

[2.3. User storys 5](#_heading=h.tg869i8or22j)

[[US1] Login and profile completion 5](#_heading=h.dpbt34hn5kin)

[[US2] Searching for an apartment and requesting favoured apartment offers 6](#_heading=h.jwp0saneq6zr)

[[US3] Add accommodation to the request list 7](#_heading=h.6wf6ned3ad59)

[2.4. Functional requirements 9](#_heading=h.jn26ch4b95lr)

[2.5. Out of scope 10](#_heading=h.vfe057wqm8yc)

[2.6. Visual design of the PWA 11](#_heading=h.m7po7b6duui8)

[2.7. Data model 12](#_heading=h.mnw34h3mcmeq)

[2.7.1. Profile information 12](#_heading=h.89fgye4ofwwr)

[Contact information (home address or semester address): 12](#_heading=h.o97zz7i6h7w)

[Bank account: 12](#_heading=h.puc3v77fsinb)

[2.7.2. Residential facilities 13](#_heading=h.43b7wx8jql5e)

[Residential Data 13](#_heading=h.36m6p76gh36u)

[2.8. Infrastructure 13](#_heading=h.tbflm2nc1m54)

[**3. Recommendation report 14**](#_heading=h.mjofynfp0jh)

[3.1. Header Level 2 14](#_heading=h.f9g7umyksmkh)

[3.1.1. Header Level 3 14](#_heading=h.5ed03xmosoas)

[Header Level 4 14](#_heading=h.9kj5q5vkl9b)

[**4. European Digital Student Service Infrastructure L2 15**](#_heading=h.4d34og8)

[4.1. Context 15](#_heading=h.2s8eyo1)

[Core Values 15](#_heading=h.6y48y4qyh0it)

# Description of the project

#### Introduction

Within the grant agreement between all project participants, this task is described as follows:

“A student housing application will be developed to enable public and private student service providers (depending on the legal framework) to manage their housing offer and allow students to apply for them and sign the rental agreements.

The open source code will be made available to student service providers to deploy for their institution(s).

The development will be based on expertise accrued through public student service providers, the national housing management service in France (lokaviz.fr) and research projects like Houserasmus+ and HOME.”

The original project partner for this task was the “Deutsches Studierendenwerk” (DSW). For serious reasons, the DSW had to withdraw from its role in the project. At the beginning of 2022 Studierendenwerk Karlsruhe (STW) stepped into this role instead.

Within the scope of this Task 2.5, a proof of concept in the form of a progressive web app based on Angular 2+, React, Flutter or comparable technologies will be created. This should offer an online selection option of housing offers at a desired place of study and use the real estate inventory managed by student services. For this purpose, filters for essential booking search criteria (booking period, room type, monthly rent, location at the place of study, etc.) should be available. In the event of a successful selection and with agreement to the booking conditions, a rental contract is to be generated in PDF format, which can also be signed online, so that a legally binding contract is concluded with the housing provider.

The app is to provide multilingual user guidance and, accordingly, contract documents that can be generated in multiple languages.

#### 

# Solution description

Within the framework of a proof-of-concept, a housing app will developed to demonstrate the following technical functionalities:

1. By means of a single sign-on based on eduGAIN, myAcademicID or an alternative registration process (social login or registration via e-mail address/password), the log in is to be demonstrated.
2. And by means of the Sunet PDF Sign a rental contract for a selected residential home shall be digitally signed in a trustworthy way.

Between single sign-on and PDF signing there is a simulated and simplified accommodation booking process for the proof-of-concept, in which a residence in a student service organization is to be selected and an accommodation booked. Data of this booking process (accommodation, apartment, rental period and monthly rent) and the personal details (first name, last name, home address, date of birth) should appear in the PDF to be signed, which will be generated based on a standard contract text.

The system will support the following main use cases:

* Users can log in with their HEI access or use an alternative access (social login, portal registration).
* Users will be able to perform a typical booking process for an apartment (selecting the location in the student service organization, the residence, the room type, the rental period)
* Users can generate the rental contract as a PDF after receiving the booking confirmation and sign it digitally.

To manage the corresponding housing offers, an administrator role and a corresponding dashboard would actually be necessary. For reasons of simplification, however, this user role will be missing in the proof-of-concept and replaced by corresponding standard settings. For example, the contract text is not versionable and fixed, only the dynamic details of the tenancy are entered. In addition, the availabilities of the residential facilities are not subject to dynamics, so that the booking status is also fixed. The final contract approval is triggered as an event and does not require a human contract check, which surely would exist in reality.

## Actors of the System

#### [A1] Students

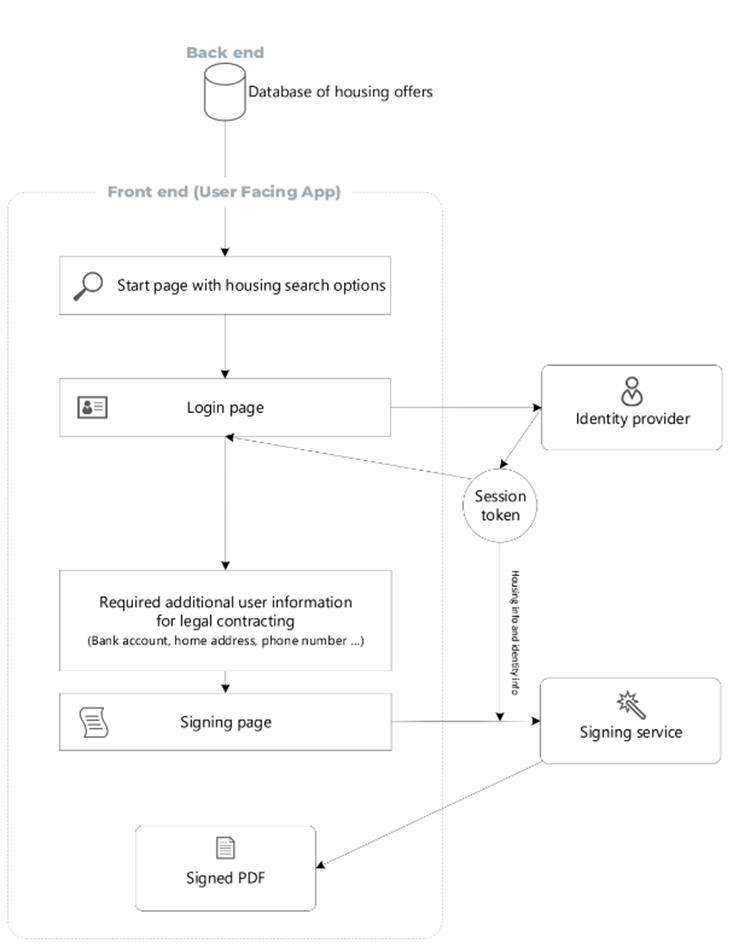
All students who are looking for a flat are able to see and navigate through the accommodation module. If these students already have an eduGAIN account (Higher Education Study and Traineeship Mobility), they can also submit applications, sign and manage contracts.

#### [A2] Student Service Providers – SSP

Student service providers can enter housing offers within this App.

## Architecture

The architecture displayed in the following diagram is not a proposition for a technical solution. It just shows the required functionalities for the well functioning of the entire system, and their interaction with the different actors of the system. The implementation of those functionalities can follow a completely different set and organisation of technical components.



| **Component** | Description |
| --- | --- |
| **Actor** | [A1] Student |

## User storys

These are the most relevant user stories:These are the most relevant user stories:

| **Title** | [US1] Login and profile completion |
| --- | --- |
| **Actor** | [A1] Student |
| **Precondition** |  |
| **Steps** | 1. The start page of the application opens. No registration necessary. 2. If the user is already logged in, it is possible to edit the profile data. 3. If the user is not logged in, Sign-On with myAcademicID is necessary 4. Completion of the required data of the user profile to create the basis for signing the contract. |
|  | |
| **Goal** | Actor is logged in and able to comprehend his personal data. |
| **Relationship to other scenarios** | - |
| **Additional information** | Even if the profile page to be completed is shown after the initial login or after registration, there is no obligation for the user to complete the profile details. Instead, users should be able to search for housing offers even without a completed profile. However, the booking processes always require a completed user profile. Entering further personal data is necessary if there is a real intention to book an accommodation. |

| **Title** | [US2] Searching for an apartment and requesting favoured apartment offers |
| --- | --- |
| **Actor** | [A1] Student |
| **Precondition** |  |
| **Steps** | 1. The start page of the application opens. No registration necessary. 2. A search request can be formulated on the welcome page. The criteria are "City"; "Housing estate"; and "move-in-date". 3. After clicking the "search” button, the results are displayed. |
|  | |
| **Goal** | Actors can search for results according to their needs and get them clearly displayed. |
| **Relationship to other scenarios** |  |
| **Additional information** | Visualization of the hit list along the filters "Sort by price" (lowest rent vs highest rent and vice versa), "Sort by availability" (shortest waiting time vs longest waiting time and vice versa) and sorting by "Distance to university" (shortest distance vs longest distance and vice versa, based on a geo-map service). The filters cannot be combined and determine the order in which the results are displayed. |

| **Title** | [US3] Add accommodation to the request list |
| --- | --- |
| **Actor** | [A1] Student |
| **Precondition** | At least one accommodation is shown in the search results. |
| **Steps** | 1. Accommodation displayed in the search results and selected via the "apply" button. 2. The detail page of the accommodation opens. This contains all the important information for the rental. 3. Clicking on the "add to request list" button:   - if the user is logged in, the offer is added to the request list.  - if the user is not logged in, the login window opens. |
|  | |
| **Goal** |  |
| **Relationship to other scenarios** | [US2] Searching for an apartment and requesting favoured apartment offers |
| **Additional information** |  |

## Functional requirements

The following table lists functions of the app, which have been divided into "Essential" and "Non-Essential". While all functions listed below would be evaluated as necessary in the real-life mode of this housing app, the following overview is intended to represent what is feasible and necessary under the scope of a proof-of-concept and what is not. All functions marked as "non-essential" are considered optional to be integrated, which are not excluded, but whose integration will be decided by budget and available time. All functions marked as "essential" should be integrated in any case in the proof-of-concept.

|  | The housing app is to implement as a PWA in a mobile-first approach, and is thus accessible for mobile devices, but could also operate on a desktop. |
| --- | --- |
|  | The portal should be easy to use and have clear navigation so that I can quickly access the information I need. |
|  | The portal should provide transparent information about room availability, rental costs, the lease agreement, and other associated costs. Additionally, the landlord's contact details and important documents such as the lease agreement should be easily accessible. |
|  | A powerful search function that allows me to search for available rooms based on specific criteria is also important. |
|  | Users can only sign rental contracts for themselves, even though in reality there are rental contracts with shared flats or couples, where more than one contractor appears in the contract. |
|  | Users can select up to three residential offers per booking request, but need to rank them. |
|  | The feedback of the administration to the booking request is made per housing offer. Only one offer can be selected by the student to sign a contract. It is also possible that no offer is made by the administration and all requests are answered negatively. |
|  | Users can access, view and download their signed contracts through the PWA. |
|  | Users receive the signed contract by email in parallel after the signature. |
|  | The app is being developed for multilingual use, but is only available in English in the proof-of-concept. The contract texts are shown in the user language in which the app is set. |
|  | The following events serve as triggers for an additional mail delivery:   1. Successful first registration on the portal 2. Successful booking of a residential object by signing the contract (with attached rental contract)   Cancellation of a contract within the cancellation period (incl. PDF with confirmation of cancellation) |
|  | Before registering, users must agree to the privacy policy and the terms of use. |
|  | Several booking requests for several booking periods can be started in parallel. However, multiple contracts cannot be signed for overlapping booking periods. |
|  | Students can withdraw booking requests that have not yet been signed. |
|  | Students can revoke the contract up to 14 days after signing. The revocation function is hidden after 14 days. There is no provision for early termination of the contract via the PWA. |
|  | Users can delete their own account in Self-Service, but only if a contract has not yet been signed. If a contract has been signed, the account can be deleted at the earliest after the end of the contractually agreed rental period. If a booking request has been started, the account can still be deleted, but this also deletes the booking request in the system. |

## Out of scope

To operate the housing app in reality, an administration dashboard would be necessary.

With this, the student service organisation would define the housing offer (objects, availabilities, prices) and generate the contract texts. Since the proof-of-concept only focuses on the user app, no administration app will be created and there is no administrator role that would be necessary in reality. All information that would typically be defined in the housing app via an administration app (objects, availabilities, prices, contract texts) are „baked“ statically into the PWA, since this is a demo. There is also no user management with the option to have the user profile adjusted by third parties or even to delete it completely.

The following is an overview of functions that cannot be integrated as part of the proof-of-concept:

|  | There will be no admin dashboard that allows configuration of housing objects. Thus, no accommodations can be added or removed or the availabilities and rents can be adjusted. The rental properties are hard-coded into the proof-of-concept with pre-as-signed properties. |
| --- | --- |
|  | The contract text cannot be edited or versioned by the administrator, the contract text is fixed and will only be generated in English. Only the dynamic contract data changes when creating a contract, depending on the tenants profile (name, etc.). |
|  | There is no user management that would be necessary to operate the app in a GDPR-compliant manner and to comply with the disclosure obligations or to initiate data blocking or data deletion actions. |
|  | There is no communication channel via the PWA between student and administration. A communication interface for messages is not provided. |
|  | The user cannot change his core data "First name", "Last name" and "Email address". |

## Visual design of the PWA

Fonts should be hosted locally and not be embedded from fonts.google.com to avoid privacy issues.

All visual design is based on the official project “Brand Guidelines”.

All colors in the PWA are derived from the EDSSI logo.

| Background color | #FFFFFF |  |
| --- | --- | --- |
| Primary color | #14adbc |  |
| Font color | #1f1b35 |  |
| Signal color | #822482 |  |

## Data model

### Profile information

The following information is requested on the app user's profile page. Various data validations need to be applied when the data is entered. Only with a complete user profile the app be used to search for apartments and to sign contracts.

| First name | prefilled after sign up, read only |
| --- | --- |
| Last name | prefilled after sign up, read only |
| Email address | prefilled after sign up, read only |
| Phone number | mandatory: international format (e.g. 001 or +49) |
| Date of birth | mandatory: should not be younger than 18 years |
| gender | mandatory: female, male, divers |
| nationality | mandatory: selection from a country list |

#### Contact information (home address or semester address):

| Street, number | mandatory |
| --- | --- |
| additional address line | optional |
| Zip code, City | mandatory |
| Country | mandatory: selection from a country list |

#### 

#### Bank account:

| Account holder | prefilled, changeable: first name, last name of the user |
| --- | --- |
| IBAN | mandatory: checksum control |
| BIC | mandatory |
| Bank name | mandatory |

### Residential facilities

The following information refers to rental objects, i.e. residential complexes. Residential complexes can be passive or active; if they are labeled as passive, they do not appear in the search. A passive residential complex could, for example, be a property in which no apartment is rented, but which cannot currently be offered on the housing market due to reconstruction etc.

#### Residential Data

| Country | mandatory |
| --- | --- |
| Assigned student service org. | mandatory |
| ZIP Code | mandatory |
| City | mandatory |
| Street, number | mandatory |
| Geo coordinates | mandatory |
| Apartment types | mandatory |
| Active/inactive | mandatory |

## Infrastructure

For the operation of the web application to be developed, it is necessary to set up and operate an appropriate cloud environment.

#### 

# Recommendation report

## Header Level 2

### Header Level 3

#### Header Level 4

#### 

#### 

# European Digital Student Service Infrastructure L2

## Context

The EDSSI L2 project builds on the EDSSI project, which aims to harmonise the operation of the different building blocks of the digital Erasmus+ ecosystem, mainstream the usage of secure authentication and extend the interoperability network to the student service providers. EDSSI created a core infrastructure that provides students with a seamless mobility experience.

The EDSSI L2 project continues and complements the work that has been started in the first EDSSI project. It will

* enhance the authentication services for the infrastructure,
* fine-tune the operation of the Data Exchange Network and
* create a new standardised software component for seamless NFC interoperability of Student eCards.

For 2025, this will provide a student eCard that enables for example eSignature, so students can sign documents with a single click during their mobility administration processes.

The overall project will align with the roll out of the Erasmus+ digitalisation roadmap and the European Student Card Initiative (ESCI).

| Core Values | |
| --- | --- |
| **Mobility empowerment** - We are tearing down the structural barriers to student mobility.  It takes student e-services to the next level and empowers all actors along the mobility chain. It stimulates student mobility by digitising administrative procedures and therefore simplifying the access to e-services. | Inclusive Higher Education - Nobody is left behind.  Whether you are a small or large Higher Education Institution, have an advanced IT infrastructure or a basic one, EDSSI provides you with solutions to connect to Europe’s student e-services. It widens your students’ opportunities and facilitates mobility. |
| **Evolving union** -We are breaking silos in the organisation and provision of student mobility.  It contributes to the digitisation of student mobility processes and encompasses all relevant digital student mobility initiatives under its umbrella. | **Reliability** - Forged on EU standards and regulations, we offer an approved and trusted connector to Europe’s student e-services.  Users can access all European student e-services in a transparent and seamless manner, according to the Once Only Principle. It is made simple for all Higher Education student mobility actors to use a solution based on open standards and clear EU policy regulations (including GDPR). |

| Logo  Description automatically generated  [**www.edssi.eu**](https://edssi.eu/) |
| --- |