



**Widening Access to Virtual Educational Scenarios**

**562463-EPP-1-2015-1-UK-EPPKA2-KA**

## **D2.2. Accessibility and usability enhancements for virtual scenario systems**

Deliverable number D.2.2

Delivery date January, 2018

Status Final

Authors Martin Adler (Instruct), Panagiotis Antoniou (AUTH), Sheetal Kavia (SGUL), Martin Komenda (MU), Andrzej Kononowicz (KI), Daniel Schwarz (MU), Dimistris Spachos (AUTH), Natalia Stathakarou (KI), David Topps (OL), Corey Wirun (OL), Luke Woodham (SGUL)



Funded by the  
Erasmus+ Programme  
of the European Union

## TABLE OF CONTENTS

<b>1. INTRODUCTION.....</b>	<b>4</b>
<b>2. OVERVIEW OF THE RESULTS.....</b>	<b>5</b>
2.1. Division of work.....	5
2.2. Usability and Accessibility Toolkit inventory .....	6
<b>3. DETAILED DESCRIPTION OF THE TASK OUTCOMES .....</b>	<b>8</b>
3.1. A1 Internationalisation support .....	8
3.1.1. Translation tool .....	8
3.1.2. RTL support .....	11
3.2. A2 Web accessibility extensions .....	18
3.2.1. Overview .....	18
3.2.2. Pages evaluated .....	18
3.2.3. Tools and Process .....	18
3.2.4. Results .....	18
3.2.5. Miscellaneous .....	19
3.2.6. Summary and Future .....	19
3.3. A3 Support for mobile learners .....	20
3.3.1. Mobile theme .....	20
3.4. Streamlining authoring workflows .....	25
3.4.1. Heuristic usability evaluation .....	26
3.4.2. Wireframe models.....	28
3.5. Interactions and decision-making training support .....	35
3.5.1. MCQ usability.....	35
3.5.2. Pathway visualisation .....	39
3.6. Just-in-time help system .....	42
3.6.1. Introduction .....	42
3.6.2. Methods and tools .....	43
3.6.3. Guided tour implementation.....	44
3.6.4. Limits of the prototype implementation .....	46
3.6.5. Conclusion .....	46
<b>4. DISCUSSION.....</b>	<b>47</b>
4.1. Impact .....	47
4.2. Limitations.....	48
4.3. Outlook.....	49
<b>5. CONCLUSIONS.....</b>	<b>49</b>

D.2.2. Accessibility and usability enhancements for virtual scenario systems

6.	REFERENCES.....	49
7.	APPENDIX A: .....	51
8.	APPENDIX B: .....	59

## 1. INTRODUCTION

The goal of this deliverable is to report on extensions in accessibility and usability in the exemplar scenario-based learning and authoring tools of the WAVES project: CASUS and OpenLabyrinth (Olab).

The way accessibility is understood in the WAVES project goes beyond the common notion of designing products for people with disabilities. Following the definition by Rubin and Chisnell we regard accessibility as the ability to “have access to the products needed to accomplish a goal” [8]. Our goal is to open the products, in our case tools to enable design and enactment of virtual scenarios, to a wider audience. The trend of opening products to “people with the widest possible range of abilities”, is also called “universal design” [3]. This includes extensions in language support to better enable people from countries where English is not the native language to design and present virtual scenarios. The technology to support this process is usually referred to by the acronym i18n (internationalization). Additionally, we aim to increase the spread of virtual scenarios by enabling them to be used “on the go” on mobile devices such as smartphones. This has application for people wishing to learn while commuting to their jobs and campus-based activities. However, it has also significant impact for people in low income countries where access to desktop computers is often more difficult than to smartphones. Enabling accessibility for mobile devices (mobility) and at the same time retaining the advantages of their use on desktop computers (multimodality) in scenario-based learning has therefore much potential to increase the adoption of this form of learning.

Usability is the ability of the user to use a product to carry out a task successfully [10]. It can be demonstrated by the absence of frustration in using a product [8]. Usability of a product is often developed through an iterative process of small changes in response to end-user feedback. This reflects the user-centered design (UCD) paradigm where the user rather than the software product itself is in the focus of attention [8].

Our goal was not to develop new functionality of the systems, but focus on the existing systems which could be simplified by another view of the data, wording or arrangement of elements in tasks performed with the software. Even though there are opinions that perfect usability supplants the need of manuals, we take a pragmatic stance and reckon that especially in complex systems for novice users a guided tour of the functionality of the system can significantly increase its usability in the long run, without imposing on the user much burden.

We base our work on the project’s needs analysis and development plan outlined in deliverables D1.1, D1.2 and D2.1. According to the conclusions from D1.2 the changes from the **learner’s perspective** should focus on delivering content “through realistic means and media with impactful consequences”. The learning experience should be “free from technical barriers”. In addition, exemplar end user scenarios in D2.1 emphasized the need of access to “scores or reports from the scenarios integrated with any other information [about the learner]” and “the option to revisit completed scenarios and review their content”.

From the **educator's perspective** the end-user specification prioritizes “realistic authentic means of interaction”, smooth “integration with existing systems” to prevent “resource fragmentation” and a responsive user interface that would enable to access the online content on “all kinds of devices”. The WAVES project description mentions “consideration of language issues, making sure that the virtual scenario (VS) systems can be easily adapted to the requirements concerning languages identified in WP1”. The development plan adds the need for mechanism to “re-use and re-personalize the scenario for different uses”. It is important that the educator can easily monitor the usage of the scenarios for instance to “see how many students accessed the scenario”, “how long students spent on each page (...) and what choices were made”.

While planning the project's development actions for this deliverable we had to consider the current status and plans of organizations that maintained the source code of the exemplar scenario-based learning environment which are: Instruct AG and the OpenLabyrinth Development Consortium (OLDC) led by David Topps (University of Calgary). Especially in the case of OpenLabyrinth the changes were significant and involved a major architectural shift. It was announced already in the life time of the project that the existing branch of the OpenLabyrinth system (v3.x) would be discontinued and superseded by a new one based on microservices. While this formed a great opportunity for us to influence change, it also limited our ability to add extensions to source code or rely on existing back-end. For that reason most of our additions are prototypes to be implemented in the new architecture.

## 2. OVERVIEW OF THE RESULTS

### 2.1. Division of work

The workload of the deliverable has been divided into six tasks presented in figure 1. Three of them (A1-3) pertaining to the accessibility (universal design) extensions, and the remaining three (U1-3) to the extensions in usability.

In the accessibility category task A1 focus on widening the potential user group by introducing an internationalization tools for community enabled localization of user interfaces (CASUS translation tool), improve existing translations (e.g. Polish language) and introducing mechanism for new language groups (right to left languages). We optimize the user interface for the use by screen readers enabling people with sight disabilities to use virtual scenarios in task A2. Finally, a new mobile theme for branched virtual patients is introduced in task A3.

We enhance usability by streamlining the layout in authoring workflows using usability inspections and wireframe models in task U1. We improve interaction elements as MCQ questions in virtual scenario and browsing of learning pathways in task U2. Finally, task U3 introduces a technical mechanism for onboarding new users of virtual scenarios through guided tours of the user interface.

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

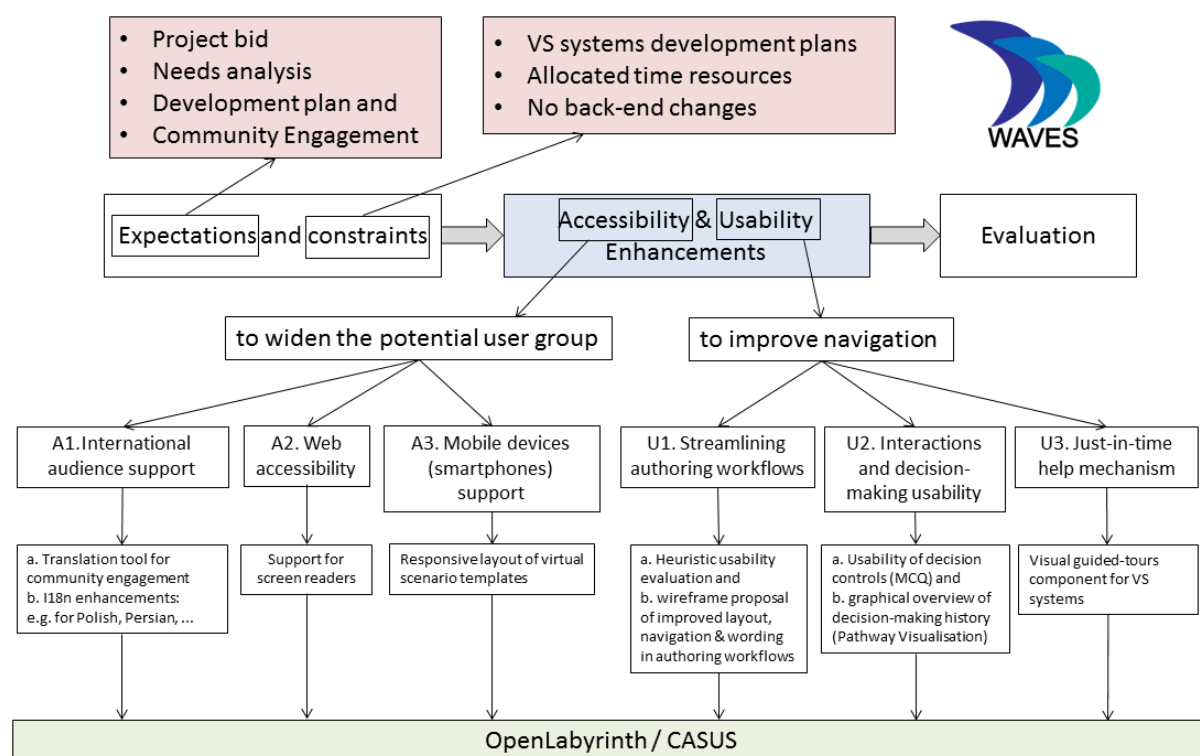


Fig. 1: Division of work in development of accessibility and usability enhancements

## 2.2. Usability and Accessibility Toolkit inventory

The tangible outcome of the project is a set of source code packages of implemented modules, branches to existing source code, resource files (as language, wireframe or demonstration files) and reports with recommendations. We publish the open source code in the GitHub repository or on the project's site depending on the type of content. The table below (table 1) shows a summary of outcomes. A detailed description may be found in the following sections of this deliverable.

Table 1. Inventory of the technical toolkit

Id	Description	Contrib	VS tool	Outcome/Form	Link to resources when open
A1.1	CASUS GUI translation tool	Instruct	CASUS	Java code (JSF web app)	<a href="https://bitbucket.org/gulpipvt/i18nproperties">https://bitbucket.org/gulpipvt/i18nproperties</a>
A1.2	Polish, Persian GUI translation files for CASUS	UJ, MU, Isfahan, Instruct	CASUS	Translation files + Java code improvements (JSF web app)	Language files: Access on request. Polish translation maintained by UJ - for improvements contact: <a href="mailto:support@casus.net">support@casus.net</a>  Tested converters RTL CSS: <a href="http://rtlcsc.com">http://rtlcsc.com</a> CSS-Flip:

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

					<a href="https://github.com/twitter/css-flip">https://github.com/twitter/css-flip</a>
A2.1	Summary of improvements of accessibility (for users with visual impairments)	Instruct	CASUS	Report	This document
A3.1	Working VS player mobile theme for Open Labyrinth	AUTH	Olab	HTML, CSS + JavaScript	<a href="https://github.com/wavesnetwork/mobileplayer">https://github.com/wavesnetwork/mobileplayer</a>
U1.1	Heuristic evaluation of Olab 3 usability	AUTH	Olab	Report	This document
U1.2	Wireframe models of Olab 4 workflows (as initiated by Sam McInerney)	SGUL	Olab	Report (drawings)	This document
U2.1	New MCQ feedback mechanism	Instruct	CASUS	Java code (JSF web app)	Proprietary (closed) code
U2.2	New VS pathway review mechanism for learners	KI	Olab	HTML, CSS + JavaScript	<a href="https://github.com/wavesnetwork/vispath">https://github.com/wavesnetwork/vispath</a>
U3.1	Prototype of guided tour mechanism for Olab 3	MU	Olab	PHP+JavaScript (Intro.js)	<a href="https://github.com/wavesnetwork/guidedtour">https://github.com/wavesnetwork/guidedtour</a>

### 3. DETAILED DESCRIPTION OF THE TASK OUTCOMES

#### 3.1. A1 Internationalisation support

##### 3.1.1. Translation tool

###### 3.1.1.1 Introduction

Translation of user interfaces can be a laborious process for larger system. The CASUS Learning System has at this moment about 2.400 labels and text blocks per language. In the past CASUS focused on

- English and
- German

As part of projects the following languages were added

- Polish
- Spanish
- French
- Portuguese
- Hungarian
- Trial in Persian (see RTL chapter of this document)
- Trial in simplified Chinese
- Czech is currently in work

We classified the languages in the following way

- Core languages, which are maintained by the CASUS team
- Additional languages maintained with support from partners

The language specific resources are JAVA property files, that consist of key and value pairs; the key is a defined label used in the internationalized templates, the value is the actual translation. Direct work on these property files is possible but not really useful for non-technical persons. The objective of this task was to experiment with a tool, providing an easy to use interface to translate such JAVA property files.



### 3.1.1.2 Methods

As first part of this task we researched available translation tools. As a result, we found a few, from which most were written in JAVA, but none of these satisfied us, mainly because they were standalone applications and we were seeking for a web-based solutions fitting into our portfolio.

The second stage was evaluating our current approach and how much additional effort would be required to provide a translation tool on the existing code basis.

We finally decided to enhance the current code and provide a very simple web-based graphical user interface (GUI).

### 3.1.1.3 Results

The tool sources are available in an open bitbucket repository  
<https://bitbucket.org/gulpipvt/i18nproperties>.

Main configuration can be done by setting the directory of the property files, access control can be achieved by using standard JAVA web-app access methods, so can also be integrated into other systems.

The main functionality is a left navigation frame with a sorted tree structure of the available properties, as a delimiter of tree nodes we assume “.” (DOT) within the keys.

On the right side the window shows the property key, a text box for the translated value and the default value in English language. The tree structure shows missing items with slightly different icons.

As an experiment we added a button which allows to let the original/default English text been translated by Google translate. The Google translate service requires having a valid Google API access key installed. The translation proposed by google is pasted in the “Translated Value” field and can be further worked on and modified (Fig 2).

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

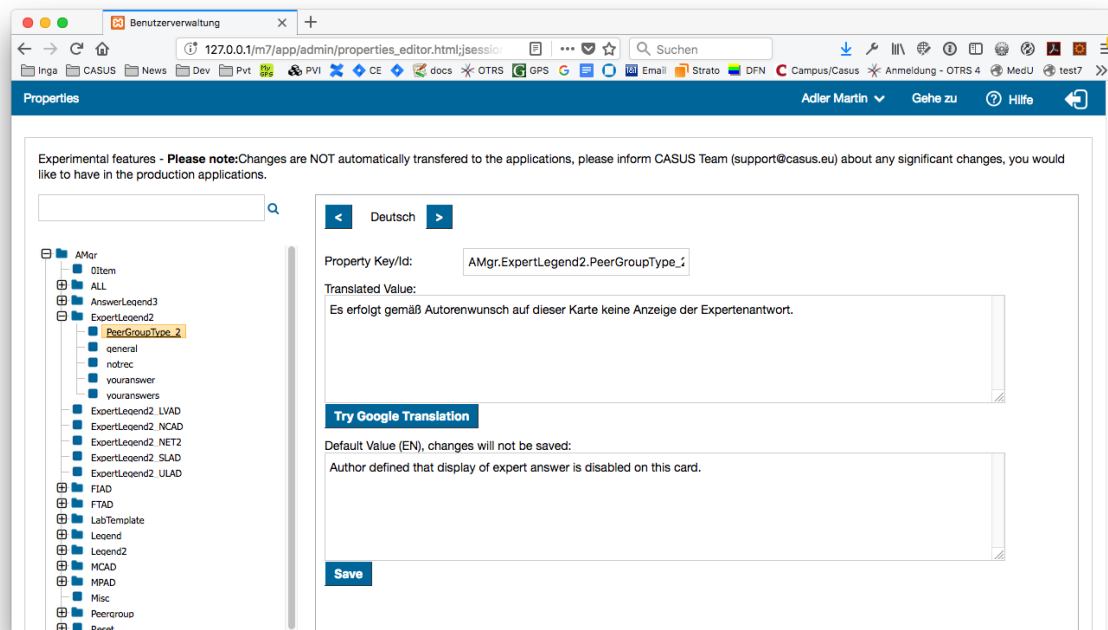


Fig. 2 Translation tool integrated into CASUS

The tool was integrated into CASUS and used for updating the Polish language files and translations into Czech (in progress), Persian (trial), and simplified Chinese (trial).

### 3.1.1.4 User experience from Krakow

The effort for a complete revision of the Polish translation with the new translation tool within the timeframe of the WAVES project was approximately 100 hours work. Due to the developments of the tool in the recent years, approximately 50% of the labels needed to be translated from scratch, but several minor changes were added to the existing translation.

An additional task we had to deal with was the existing mixture of different methods of encoding the Polish language characters in the property files. The legacy iso 8859-1 with HTML entity encoding (e.g. &#321; for capital L stroke: Ł, etc) was unified to UTF-8 encoding. The use of the tool was a pleasant experience for the translator without major usability issues reported.

The automatic Google Translate suggestion turned out to be a helpful feature to get a base for following refinements which saved time.

The features that were most missed in the current version were a reference list of the not yet translated items and a possibility to add short scripts to dynamically adapt the translation to the values reported by the user interface.

An example of the latter could be the way lexical suffixes of plurals are formed in the Polish language. For instance the word “scenario” is translated into Polish as “scenariusz”. If we now would like to display dynamically a list of  $n$ -scenarios, the suffix changes:  $n=1$ . “scenariusz”,  $2 \leq n \leq 4$ .

“scenariuse”, but  $n \geq 5$ . scenariusy”. The rules for other word groups are different - e.g. for the word “patient”: “pacjent” the change would be  $n=1$ . “pacjent” but  $n \geq 2$  “pacjentów”.

### 3.1.1.5 Limitations and Outlook

During research and implementations there were a couple of ideas, realizing some limitations of the present tool:

#### Better update / change support

The tool works pretty good for a bigger initial effort but for a better change and update management additional functionality would be necessary.

The system should detect changes in items of the original and default language (English), and make it visually easy detectable for translators what is new and what changed to be able to track and maintain a translation in an efficient way. To implement it efficiently on the programmers' side it might be necessary to deviate from the standard Java language files and organize the translations in easy to query database tables. A hybrid approach might be possible as well, but overall for a better translation workflow more meta information seems to be necessary. The standard JAVA property files are not designed for that. We realized in our evaluation of existing tools, that some workarounds are used for such additional metadata: a common meta information set is a “comment” about a label, to add context information for the translators. Some tools store such comments in additional files, or with naming conventions within the same file. Both of these approaches might be limiting, so an overall database driven approach is envisioned for the future.

#### Context

A lot of labels are hard to translate properly without any context information, so ideally translation would take place in the real pages of a tool with the full context visible. This would need a very tight integration into tools and can only be achieved in early overall design decisions. We are at this moment experimenting with a reimplementaion of the CASUS frontend in the Angular JS framework, and were surprised that even in such a popular framework the internationalization is not yet optimally solved. Overall, we see i18 support still as a major issue in 2018 regardless of any used framework, so care is necessary in architectural decisions.

### 3.1.1.6 Conclusion

Internationalization is an important part of scenario-based learning systems and should be included into early architectural evaluations and decisions. Supporting the complete workflow would be ideal to facilitate also a more community-based translation approach.

## 3.1.2. RTL support

### 3.1.2.1 Introduction

The current CASUS version has not yet worked with right to left (RTL) languages, i.e., languages which start from the right of the page and continue to the left as e.g. Arabic, Hebrew and, Persian. Translations of the interface were done in: German, English, Spanish, French, Polish, Hungarian and Portuguese.

This document describes the evaluation process of making the platform useful for RTL languages and describing best practices and guidelines for other systems. Additionally, we began to implement a first draft of supporting RTL languages in CASUS.

### 3.1.2.2 Methods

#### Translation of properties

For the project we choose the Persian language as an example because of existing networking opportunities. For an initial trial we selected a set of 50 of over 2.000 strings for translation. We selected three main pages of the CASUS system:

- The intro page (login page)
- The dashboard page for learners
- A screen card displaying a virtual scenario within player

Translation of the selected 50 strings was done in an Excel sheet with screenshots to illustrate the process; the strings were then edited within the CASUS property file editor. Technically the process was not problematic, because CASUS already supports the UTF8 encoding, however the translation of the items is only the first and easy step.

### 3.1.2.3 Design

#### Login Page

As in most web applications CASUS consists mainly of HTML templates. Styles are defined in CSS files. The major structure is completely left to right (LTR) oriented.

For RTL support we evaluated two available tools: RTLCS and css-flip.

#### RTLCS

The first trial was with a npm script rtlcss <http://rtlcss.com/> by Mohammad Younes.

The Installation was running well and we tried first one of the major CASUS CSS files “styles.css” containing a total of over 3500 lines.

```
sudo npm install -g rtlcss
```

CLI run

```
rtlcss <src> <dst>
```

The first attempt failed, because the script complained about a line “noshade;”. Trying the “ignore” declaratives did not solve the problem:

```
/*rtl:begin:ignore*/
```

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

```
noshade;
```

```
/*rtl:end:ignore*/
```

We finally temporarily removed this line, re-ran the script, and put it back afterwards. However, this would not work in an automated deploy process.

We then manually took the result “styles\_rtl.css” and included it in the login page template.

Overall, the result was satisfying, however, there still were some blocks which were not rendered correctly, so, with the development tools of Firefox we added some `dir:rtl` and `align-content: right;` directives. The first result looked as following (Fig 3).

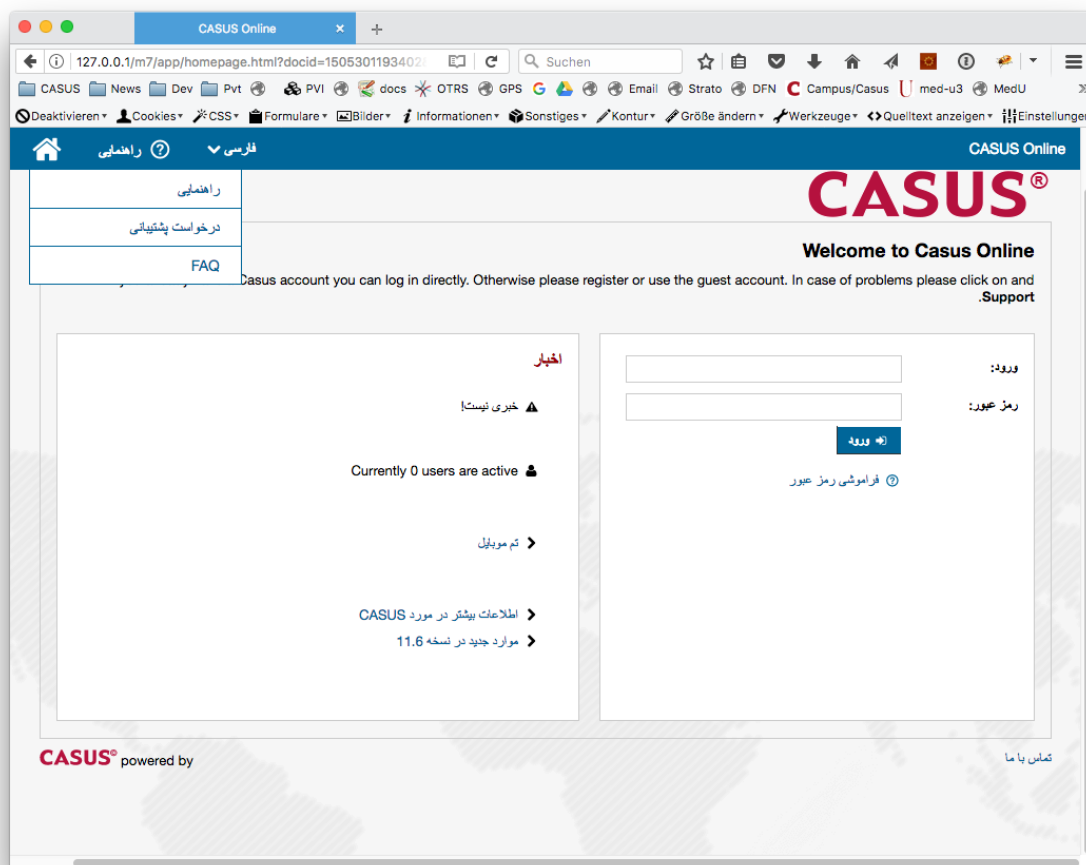


Fig 3: CASUS login page in Persian

Overall, a promising result but we noted the following issues:

- Issues when labels are dynamically populated from language property files, the “:” like for the login and password boxes were directly in the template.

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

- The arrow icons in the left box are design elements (“bullet points”) but are now exactly in the opposite direction. As the icons are drawn by an icon font, exchanging this would be relatively straightforward, BUT the icon might be used differently throughout the application so a global change might not work.
- The CASUS logo direction is not correct.

Please note: the phrases “Welcome to CASUS ...” is not part of the template translation, but a text that can be adapted for each customer group. This leads to an interesting and crucial point: The mixture between GUI interface language and dynamic content language, which both form the “real” content. The navigation language might be different from the content language in any system displaying content separately edited and authored.

**css-flip** (<https://github.com/twitter/css-flip>)

Installation:

```
sudo npm install -g css-flip
```

CLI run

```
css-flip <src> > <dst>
```

Tool complained again about the “noshade” property, manual removing was successful! Result was similar to rtlcss, manual edits were still needed.

### 3.1.2.4 Comparison With-/out CSS Modifications

#### Login Page

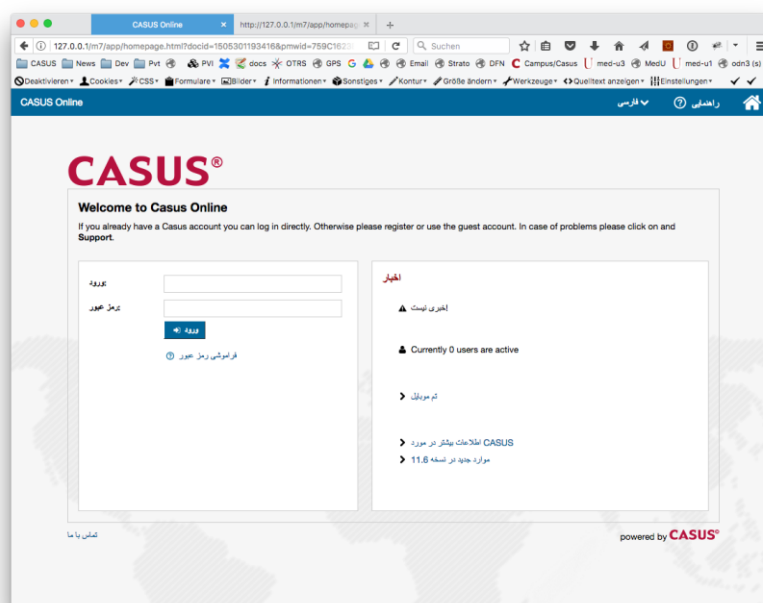


Fig 4: Translation of login page without CSS Modifications

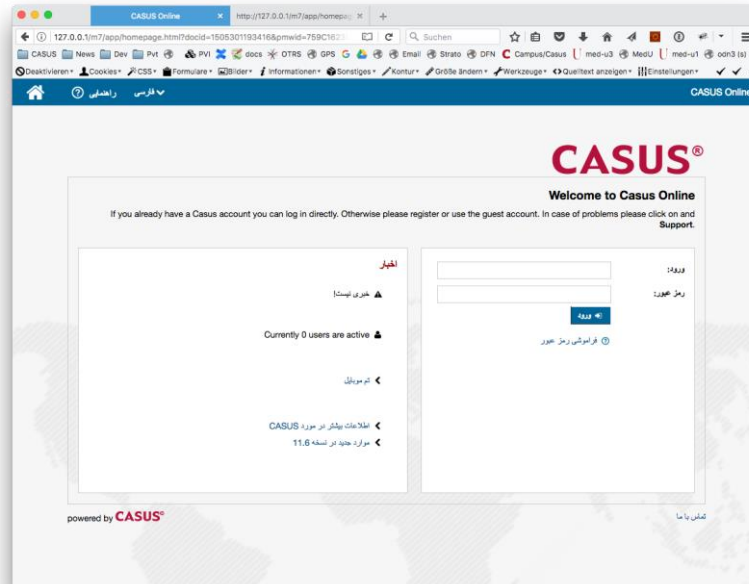


Fig 5: Translation of login page with CSS Modifications

## Dashboard and Course-Dashboard

Some minor changes in the template files were needed, such as removing some element “float:” styles and moving this to css files.

### 3.1.2.5 Miscellaneous

#### Numbered/ ordered lists

At certain places within CASUS, but also in other applications, numbered lists are implemented. In CASUS we use a simple algorithm “number2char”, based on the present indices within the A-Z and a-z characters of charsets. This might not work in persian, as different indices, characters, and digits are used.

Formally, in MCQs or other question numbering systems we use Abjad numerals, which is mainly an Arabic system (Fig 6).

1 or A	ا	alif	ā / ’
2 or B	ب	bā’	b

3...	ج	jīm	ī
4...	د	dāl	d
5...	ه	hā'	h
6	و	wāw	w / ū
7	ز	zayn/zāy	z
8	ح	ḥā'	h
9	ط	ṭā'	ṭ

Fig. 6: Abjad numerals

These challenges have to be evaluated separately, and the method will have to be extended.

### 3.1.2.6 MIXED LTR and RTL content:

Both OpenLabyrinth and CASUS and potentially other virtual scenario systems are Content/Learning Management Systems (CMS, LMS), therefore have:

- A platform language (selected or auto-detected by/for the user)
- A content language (language in which the virtual scenario is written)

It is not uncommon, that a user has selected a platform language that is different from the content (virtual scenario) language. Examples are courses which are mainly compiled of scenarios in the mother language of users and some interesting scenarios, which are not available in this language are selected from another language, most likely English. This can be particularly true for languages where only a limited number of VS are available, so courses are filled up with non-mother language content.

This is different from sites, such as regular web or shopping sites, where a user switches globally to the language desired with a globally present switch; in such sites there is no difference between GUI and the content language. It may be also particularly true for non-commercial use of scenarios. In business context with more available resources both, content and system can be individually tailored to the end user needs including all language aspects.



## D.2.2. Accessibility and usability enhancements for virtual scenario systems

Mixing the GUI language with a different content language is not particularly problematic when platform/content language are with the same orientation category, so e.g. both LTR, but gets somehow unusual with mixed LTR and RTL:

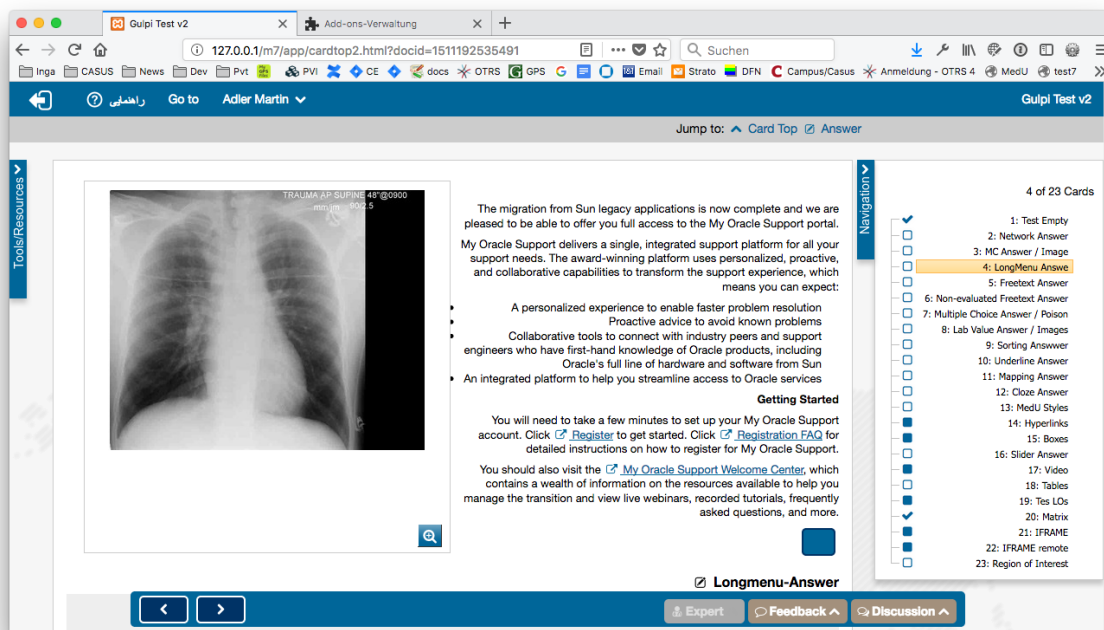


Fig 7: Example of mixed content: GUI is in RTL (only partly translated for a pilot), the VS is in English; the way the English content is now rendered RTL makes it quite hard to read.

Potential solutions for this

- Keep it up to the user and make language switch easier
- Automatically switch platform language (when possible and available) to same language as content
- Switch back to user selected language after VS is completed

Also combinations could be realized.

Even though not a scientific prove we found the following interesting statement of a user in a forum:

*“As an Arabic speaker, when I am on Arab websites that contain mixed language content, if I see Arabic in LTR or English in RTL, I feel that not enough effort has been put into doing the site. Your display of Arabic in RTL and English in LTR showing each language in its correct format shows that effort and thought has been put into this and would increase my trust in your site.”*

(<https://ux.stackexchange.com/questions/67071/mixing-ltr-left-to-right-and-rtl-right-to-left-content-on-the-same-page>.)

### 3.1.2.7 Conclusion for RTL Support

- Translation of the labels is well supported by the system and the translation tool
- Basic support for RTL can be achieved in combination with available tools and fine tuning
- For a full support of RTL languages additional effort is needed especially for the scenario of mixed RTL and LTR content. Whether this concept can be realized depends on the strategies of the scenario-based learning platforms.

## 3.2. A2 Web accessibility extensions

### 3.2.1. Overview

We evaluated the CASUS system for compatibility with the Web Content Accessibility Guidelines (WCAG).

### 3.2.2. Pages evaluated

The following main pages for students were reviewed:

- Self registration + confirmation
- Login page
- Forgot password page
- Dashboard pages
- Main VS card in two variants: basic, including clinical reasoning, including multimedia
- VS session summary page

### 3.2.3. Tools and Process

First, we scanned all pages and checked common guidelines according to WCAG and webaim (webaim.org); We corrected detected issues immediately. Then, all pages listed above were checked with WAVE – the WAVE Web Accessibility Tool (wave.webaim.org) (Fig. 8). All errors were checked and corrected, and all warnings were scanned whether they could result in issues. We also checked the pages with the open source screen reader NVDA ([www.nvaccess.org](http://www.nvaccess.org)) on Windows 7 and with internal screen reader tools available on Mac OS X.

### 3.2.4. Results

After the test all evaluated pages contained no more issues (errors) according to WAVE Web Accessibility Tool (wave.webaim.org), alerts were scanned and fixed when relevant for usability.

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

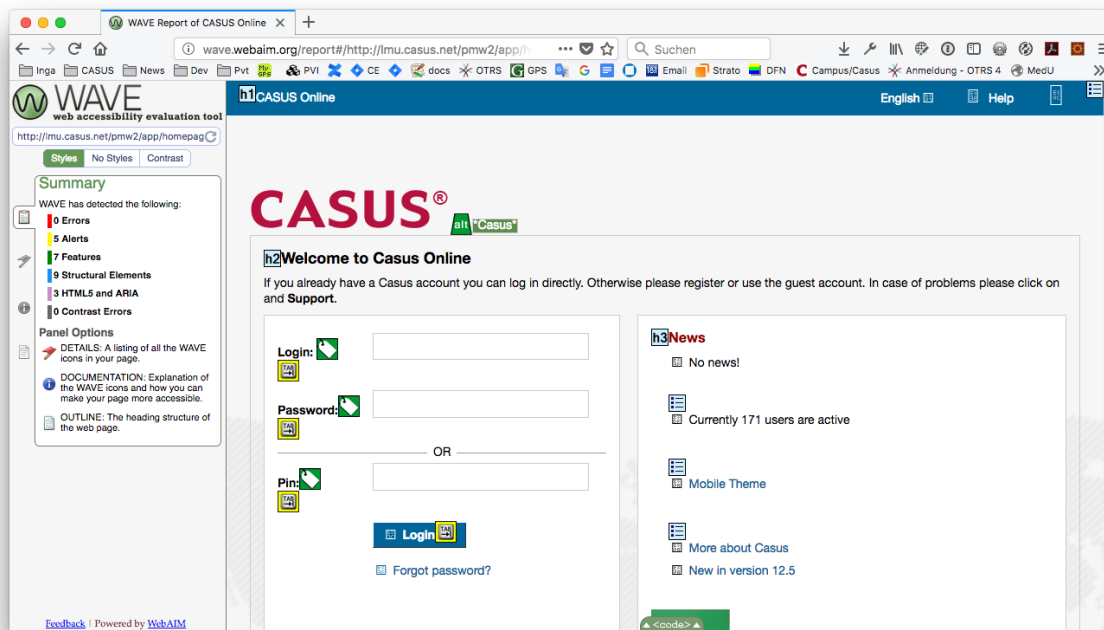


Fig. 8: Sample WAVE check of the login page

### 3.2.5. Miscellaneous

The registration page in CASUS contains a captcha, and captcha by definition creates a common issue with web accessibility. We researched whether the used captcha (Google Recaptcha 2.0) is acceptable for this purpose. While not evaluated as 100% ideal Google ReCaptcha 2.0 seems to be one of the best available products in the market, and with the wide use of Google's ReCaptcha optimization to latest guidelines should be guaranteed.

(<http://simplyaccessible.com/article/googles-no-captcha/> and [http://www.w3.org/WAI/GL/wiki/Captcha\\_Alternatives\\_and\\_thoughts](http://www.w3.org/WAI/GL/wiki/Captcha_Alternatives_and_thoughts)).

### 3.2.6. Summary and Future

The web accessibility review resulted in slightly modified pages which are already public. Even though it cannot always be guaranteed that no new WCAG issues will get into the code base, we will create internal guidelines to enforce that the current state and new developments will be compliant to WCAG guidelines in the best way possible.

### 3.3. A3 Support for mobile learners

#### 3.3.1. Mobile theme

##### 3.3.1.1 Description-Motivation

Although the Open Labyrinth (OL) application is a web application and, thus, can be run using a mobile browser, the user experience (UX) suffers in small screens, as shown in the following figure 9:



Fig. 9: Open Labyrinth scenario played in mobile device.

We can clearly spot hidden areas of the screen, inconsistent buttons sizes etc. Considering that nowadays more than 50% of the users, navigate the web through a mobile device, one of the main project actions was to develop a separate mobile theme layer, to enhance the usability and the user experience.

##### 3.3.1.2 Methods

The mobile theme layer was implemented in its current version as an independent web application but is designed in a way, that makes it easy to integrate it in future with Open Labyrinth using a microservice mechanism or just via the XML (e.g. MedBiquitous Virtual Patient Data) export of the Open Labyrinth scenarios. Implementation of the integration mechanisms is not in the scope of this deliverable.

We followed a bottom-up approach, designing the player not only to fit in small screens, but also to maximise the user experience, when played in a mobile device. The buttons were replaced by a radio button list, and a separate button to continue to the next node. This way we prevent accidental taps on the screen but also make the user interface more consistent, by having the

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

exact same button size in all screens and let the text flow according to each answer size. Using coding techniques, we prevent the user from going forward and backward between nodes: if a user clicks the back button (on an Android device for example), the application routes the action to the main, home screen instead of the previous node. Thus, we can better control user transitions between nodes in the virtual scenario which might prevent dishonest or unexpected behaviours in the system. Modal dialogs for supporting information and pathway, let the user get extra information, without forcing to navigate or scroll. On top of the screen there is a special progress indicator, that lets the user know if he/she reached any given (by the virtual scenario author) milestones.

From a developer's point of view, special functions were added to allow the data exchange between the player and xAPI learning record stores. Excluding milestone top progress indicator, all other components are compatible with the VP MedBiquitous standard.

### 3.3.1.3 Results

Figure 10 presents screenshots of the implemented mobile player from the learner perspective.

## D.2.2. Accessibility and usability enhancements for virtual scenario systems



Fig. 10: Mobile theme for branched virtual scenarios

### Technical aspects

Currently the app code is available under the following GitHub repository:  
<https://github.com/WavesProject/OL-mobile> under the MIT license.

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

An online demo is available at: <http://medresearch2.med.auth.gr/waves-mob>, but for an optimal experience ought to be viewed using a mobile device and not a desktop browser.

A developer can start working on the application, by cloning the repository:

```
git clone https://github.com/wavesproject/OL-mobile
```

Then, in the project folder must install the dependencies:

```
npm install
```

Finally, we launch a development server, and open localhost:4200 in the browser:

```
npm start
```

### Project structure

dist/	compiled version
docs/	project docs and coding guides
e2e/	end-to-end tests
src/	project source code
- app/	app components
- core/	core module (singleton services and single-use components)
- shared/	shared module (common components, directives and pipes)
- app.component.*	app root component (shell)
- app.module.ts	app root module definition
- app-routing.module.ts	app routes
+- ...	additional modules and components
- assets/	app assets (images, fonts, sounds...)
- environments/	values for various build environments
- theme/	app global scss variables and theme
- translations/	translations files
- index.html	html entry point
- main.scss	global style entry point
- main.ts	app entry point
- polyfills.ts	polyfills needed by Angular
+ - test.ts	unit tests entry point
reports/	test and coverage reports
proxy.conf.js	backend proxy configuration

Task automation is based on [NPM scripts](#) presented in table 2.

Table 2. Description of parameters in NPM scripts of the Mobile Theme project

Tasks	Description
npm start	Run development server on http://localhost:4200/
npm run build [-- --env=prod]	Lint code and build app for production in dist/ folder

npm test	Run unit tests via <a href="#">Karma</a> in watch mode
npm run test:ci	Lint code and run unit tests once for continuous integration
npm run e2e	Run e2e tests using <a href="#">Protractor</a>
npm run lint	Lint code
npm run translations:extract	Extract strings from code and templates to src/app/translations/template.json
npm run docs	Display project documentation

When building the application, you can specify the target environment using the additional flag `--env <name>` (do not forget to prepend `--` to pass arguments to npm scripts).

The default build environment is prod.

### Development server

This project was generated with [ngX-Rocket](#) version 1.3.3

Run `npm start` for a dev server. Navigate to `http://localhost:4200/`. The app will automatically reload if you change any of the source files. You should not use `ng serve` directly, as it does not use the backend proxy configuration by default.

### Code scaffolding

Run `npm run generate -- component <name>` to generate a new component. You can also use `npm run generate -- directive|pipe|service|class|module`.

If you have installed [angular-cli](#) globally with `npm install -g @angular/cli`, you can also use the command `ng generate` directly.

### Additional tools

Tasks are mostly based on the angular-cli tool. Use `ng help` to get more help or go check out the [Angular-CLI README](#).

### What's in the box

The app template is based on [HTML5](#), [TypeScript](#) and [Sass](#). The translation files use the common [JSON](#) format.

### Tools

Development, build and quality processes are based on [angular-cli](#) and [NPM scripts](#), which includes:

- Optimized build and bundling process with [Webpack](#)
- [Development server](#) with backend proxy and live reload
- Cross-browser CSS with [autoprefixer](#) and [browserslist](#)



## D.2.2. Accessibility and usability enhancements for virtual scenario systems

- Asset revisioning for [better cache management](#)
- Unit tests using [Jasmine](#) and [Karma](#)
- End-to-end tests using [Protractor](#)
- Static code analysis: [TSLint](#), [Codelyzer](#), [Stylelint](#) and [HTMLHint](#)
- Local knowledge-base server using [Hads](#)

### **Libraries**

Some state-of-the art libraries used to develop the mobile player application are:

- [Angular 2](#)
- [Bootstrap 4](#)
- [Font Awesome](#)
- [RxJS](#)
- [ng-bootstrap](#)
- [ngx-translate](#)
- [Lodash](#)

### **3.3.1.4 Evaluation**

We performed a pilot system usability evaluation of the mobile player with 15 students of the School of Medicine of Aristotle University of Thessaloniki. We used the System Usability Scale (SUS) [Brooke]. The overall score was 81.78%.

### **3.3.1.5 Conclusion for mobile player:**

- We built a state-of-the art mobile player for playing virtual scenarios in smartphones and tablets
- We designed the mobile player to enhance the usability and user experience on small screens
- The mobile player's open architecture makes it easy to embed it on current and future versions of Open Labyrinth

## **3.4. Streamlining authoring workflows**

The process of introducing usability enhancements in OpenLabyrinth was initiated by an in-depth usability inspection. In addition, project partners inexperienced with authoring of virtual scenarios walked through the common workflows in OpenLabyrinth and based on these experiences

wireframe models of a new user interface layout were developed. The results of this process including proposed improvements are described in this section.

### 3.4.1. Heuristic usability evaluation

#### 3.4.1.1 Introduction

This report provides a detailed description of the heuristic evaluation process of OpenLabyrinth version 3.3. The evaluation itself was performed using the heuristic evaluation usability method, based on heuristics provided by Jakob Nielsen [Nielsen05]. This method involves evaluators comparing a pre-defined set of usability principles to an application or website while attempting to complete a system task.

For this project, ten heuristics were used (Appendix A), focusing on the core functionalities of OpenLabyrinth: displaying, creating and playing virtual scenarios (VS). The goal of this evaluation was to identify major usability flaws within the OpenLabyrinth interface through the application of the identified ten heuristics.

#### 3.4.1.2 Summary of findings

Table 5 with the summary of findings can be found at Appendix A. We use the notation from tables 3 and 4 in the same place.

#### 3.4.1.3 Specific problem areas

Overall, by using Nielsen's heuristics, we were able to identify several usability issues in OpenLabyrinth. The evaluation of the system showed that issues exist in almost every screen and, while many of them are easy to fix, there are important remaining which require more effort to be solved. The most important findings are analysed in the sections below.

#### 3.4.1.4 Issue number #1

Help and documentation is missing. Tooltips are missing. Small text areas with hints and tips are missing. While there is an aesthetic and minimalist design, objects, actions and visible options are not easy to recognize. Users must recall instructions (if any) from past training on the system.

#### Recommendation:

- Develop online help and documentation
- Provide a system help mechanism (e.g. guided tour)
- Use tooltips, tips and hints under the labels in forms
- Develop a glossary page with terms related to virtual scenarios

#### 3.4.1.5 Issue number #2

There is an inconsistency between menu labels, path labels, button labels etc. This is relatively easy to fix, but has a strong (negative) effect on the user experience.

Examples:

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

- A. A user clicks on main menu Labyrinths -> My collections
- B. The page “My collections” has the label “Collections”
- C. The user does not know if the list is his/her own collection, system collections or some other user’s collections

The wrong label violates the following (heuristic) principles:

Heuristic #1: Visibility of system status

Heuristic #4: Consistency and standards

Moreover, some labels are written using the first letter capital while some other labels on same type of elements have the first letter small. Forms with action words in their title (e.g. “VUE to OpenLabyrinth upload”) have their action button labeled different (e.g. “submit”). End to end (E2E) testing is a technique that can be followed to prevent this type of errors.

### **Recommendation:**

- Be consistent regarding labels in menus, buttons, title pages etc.
- Be consistent regarding actions in form titles and form button labels
- Develop some E2E tests

### **3.4.1.6 Issue number #3**

The system has several functions available to authors, administrators, and registered learners. While there are lots of forms in the system, we could not find any error prevention strategy. The system allowed us to submit empty forms, without any check and/or error message. Error messages are missing. Moreover, there is no obvious way for a user to undo an action, or go back to a previous state from a current unwanted state. Thus, users use the browser back button most of the times.

This behavior violates the following heuristic principles:

Heuristic #3: User control and freedom

Heuristic #5: Error prevention

Heuristic #9: Help users recognize, diagnose, and recover from errors

### **Recommendation:**

- Develop an error prevention strategy
- Following the above strategy, develop functions to check form fields, display appropriate error messages to users etc.

## Conclusions

- We performed a heuristic evaluation on OpenLabyrinth and, based on the results, we spotted the most important issues across the application
- Most of the issues are relatively easy to fix, while others (e.g. the help system) requires some serious effort
- The findings and recommendations are applicable in the current version of the OpenLabyrinth (3.5), which might be impractical as it is going to be discontinued soon, as well as in the future version of the application

### 3.4.2. Wireframe models

#### 3.4.2.1 Background

The purpose of this task was to initiate a process of iterative refinements to the user interface of VS systems that would streamline the workflows identified in D2.1, and ultimately make the systems more usable. Clearly all VS systems are different, and would require different interfaces in order to reflect their distinct feature sets, but the goal of these wireframe guidelines is to provide a generic set of interface recommendations and principles that can guide VS system developers.

Although the goal was to produce a set of generic recommendations, the wireframes were specifically designed with the OpenLabyrinth 3 system in mind. This is for a number of reasons: the availability of the system and its open source nature means that it is widely accessible; it is the system known to the team creating the wireframes; and the system is being re-architected for a forthcoming v4 release, which means that interface elements will be more readily customisable. As a result, the timing of this deliverable means that it can be fed into the OpenLabyrinth development through the WAVES projects close ties with the OpenLabyrinth Development Consortium.

The wireframe developments tie in closely with the work done on the heuristic evaluation in the previous section, but take a very different approach in identifying usability enhancements. Chiefly, the heuristic evaluation comes from the perspective of a usability expert, who has received training in sound interface design techniques and common expected usability challenges. It measures deviations from a defined list of usability norms. In contrast, the wireframes have been primarily developed and evaluated from the perspective of end users. As a consequence of the different perspectives taken, these tasks allow us to triangulate multiple views on the usability of VS systems, and OpenLabyrinth in particular, to provide a broad and overarching summation of the usability challenges in VS systems.

As a specific VS system, the nature of the usability challenge for OpenLabyrinth partly comes from its open source nature. The technology behind Open Source tools often unfunded and driven by developers own interests on a volunteer basis, developed and expanded according to the needs, development styles, and objectives of a number of different developers or teams. This can lead to a disjointed and overly complex system, that is hard for end users to influence or use. It may contain niche functionality that is of little interest to the wider community, but commands prime space in the user interface due to the primacy of the developer in the process, and the increased likelihood that they are developing for their own use rather than those of the target end users. The

ability of the project owner or coordinator, assisted by version control tools (such as GIT or SVN) and portals (such as GitHub, BitBucket, or SourceForge) is key to ensuring that the open source system is maintainable and streamlined.

OpenLabyrinth is at a key point in its development as it reaches v4; it is being structurally redesigned around a microservices architecture that allows for extensive customisation of the interface, and a focus on the functionality that is of most use to the end user. The wireframes developed and the evaluation of their effectiveness will have great scope to address any challenges identified by the heuristic evaluation, and to influence the design of OpenLabyrinth (and other) scenarios in future.

### 3.4.2.2 Methods

A dual approach was taken to the task, in order to solicit multiple opinions from those with varying levels of expertise and different perspectives. Detailed comments and thoughts were required, so three contributors were considered sufficient to give us a breadth of detailed opinions.

The first contributor was a member of the project team with experience as a technical developer, so had a grounding in the technical limitations of platforms, as well as the key principles of interface design. He is, however, also a practicing medic that has both used VS cases as a learner and created cases as an author, so has extensive experience of the area. This combination of expertise made him uniquely placed to develop a set of wireframes from both an end user and a developer/designer perspective. He then created a set of guidelines which sketched out a view of a proposed user interface that would provide an improved user experience for users of OpenLabyrinth. The intention was that he would use these wireframes as a basis to address the issues that he has found most pressing in his own use of the system.

These interface wireframes would accentuate the functionality that is most important to users of the system and abstract those features which are solely for more advanced users i.e. the user preferences expressed in the WP1 needs analysis task. The end result would be a more readily accessible interface to a still feature-rich system. The wireframes also related to the ideal user workflows identified in D2.1, and supported these.

The produced wireframes were then examined by another member of the project team, and used to document common workflows in OpenLabyrinth as an instructional step-by-step process, also based upon the workflows in D2.1. The instructions (presented in appendix B of this deliverable) were designed to guide the end user to complete two particular tasks; Playing a Virtual Scenario or Authoring a Virtual Scenario in OpenLabyrinth.

These instructions were shared with two other members of the project team; these contributors were new to creating virtual scenarios, and had no prior experience with the OpenLabyrinth tool as either a learner or an author. They were asked to work through the tasks outlined in the instructions, and to provide feedback or any challenges identified while pursuing these tasks. Two openly available (public) scenarios/cases were shared with them to allow them to play the scenarios, and authoring permissions were granted to allow them to create a scenario. Their feedback for each task was recorded as written comments.

Finally, upon completing both tasks, the original wireframe document was also shared with these non-experts, and they were asked to provide detailed feedback about whether and how adopting the interface refinements described in the wireframes would impact the user experience.

### 3.4.2.3 Results

Detailed comments were received from the non-expert participants for both playing and authoring the VS. They also provided feedback for the wireframe document. The participants completed the two sets of tasks successfully, without any significant impediment preventing them from reaching the end.

#### Playing a VS scenario

While ‘Playing a Virtual Scenario’, both the respondents felt the scenario given was easy to follow and have received a score as part of playing through the case. To return to a decision that was incorrect, both of them used ‘Review your pathway’ at the end of the case and independently identified a number of issues with this functionality:

1. ‘Review your pathway’ does not tell user which decisions were incorrect and feedback is not present when a particular pathway is clicked from the list
2. The score received can be changed by going back to a pathway from the list when moving forward.

One proposal was made to resolve the first of these issues, by colour coding the review your pathway in red and green, depending upon whether the response was correct. Additionally, the respondents pointed to a number of spelling, punctuation and typographical errors in the case, which specifically related to the content of the case rather than the system.

#### Authoring a VS scenario

Both respondents thought that the visual editor was intuitive, easy to use, and was effective at providing the possibility of making modifications to the case whilst playing. However they were uncertain about the terminologies used in the system for e.g. text boxes to fill under ‘create step-by-step’ while creating a scenario. One of the users found it difficult to have the desired text size for comments while creating the scenario. As a consequence, both users felt that the “step-by-step” wizard was actually unsuitable for novice users, and found that the manual approach was a better and more understandable option.

One respondent particularly commented that the visual editor was particularly intuitive and easy to use. This respondent self-identified as a very visual person, so felt that this interface was ideal for them.

Both end users were unable to successfully upload an image to the system. This however is a particular security feature of this system instance (for security purposes), rather than the system itself; the instance used prevents uploads from external IP addresses. As a consequence, the error is not elegantly handled by the system, instead leaving the users guessing as to why their file had not uploaded. Although this cannot be considered a usability failure on behalf of the system, it highlights the importance of providing suitable communication to the user in an event of an error.

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

The respondents made the following suggestions for improvements to the system.

1. A description box to specify aspects to consider (e.g. location, player, situation, adding and linking nodes to create pathways etc) when a visual scenario is created first time
2. A demo map with a few nodes for user to see how a scenario is created
3. A journey map available whilst creating a scenario that enables user to go back and view the questions and answers created, in order to identify correct and incorrect pathways
4. Titles in nodes would be truncated without communication if they went over a certain length, and the behaviour of the cursor in the text box was not as expected. Resolving this would benefit the user.
5. Creating a Quick Start option that allowed a case name to be entered that would then take an author directly to the visual editor.
6. The toggle between the Simple/Advanced user interface configurations was hidden and not intuitively placed, being in the Help menu. This would be more usefully labelled under a "View" menu.

### **Wireframes**

The wireframes were created with a focus on Nielsen's 10 Heuristics for good user interface design, and adopted as a general principle a move away from a "toolbar" style implementation, instead focusing key functionality on the centre of the screen. The full Wireframes document is included as an appendix to this deliverable.

The wireframe shown in figure 11 describes a main central screen for the application, in which the key functionality and actionable elements of the screen are shown in blue.

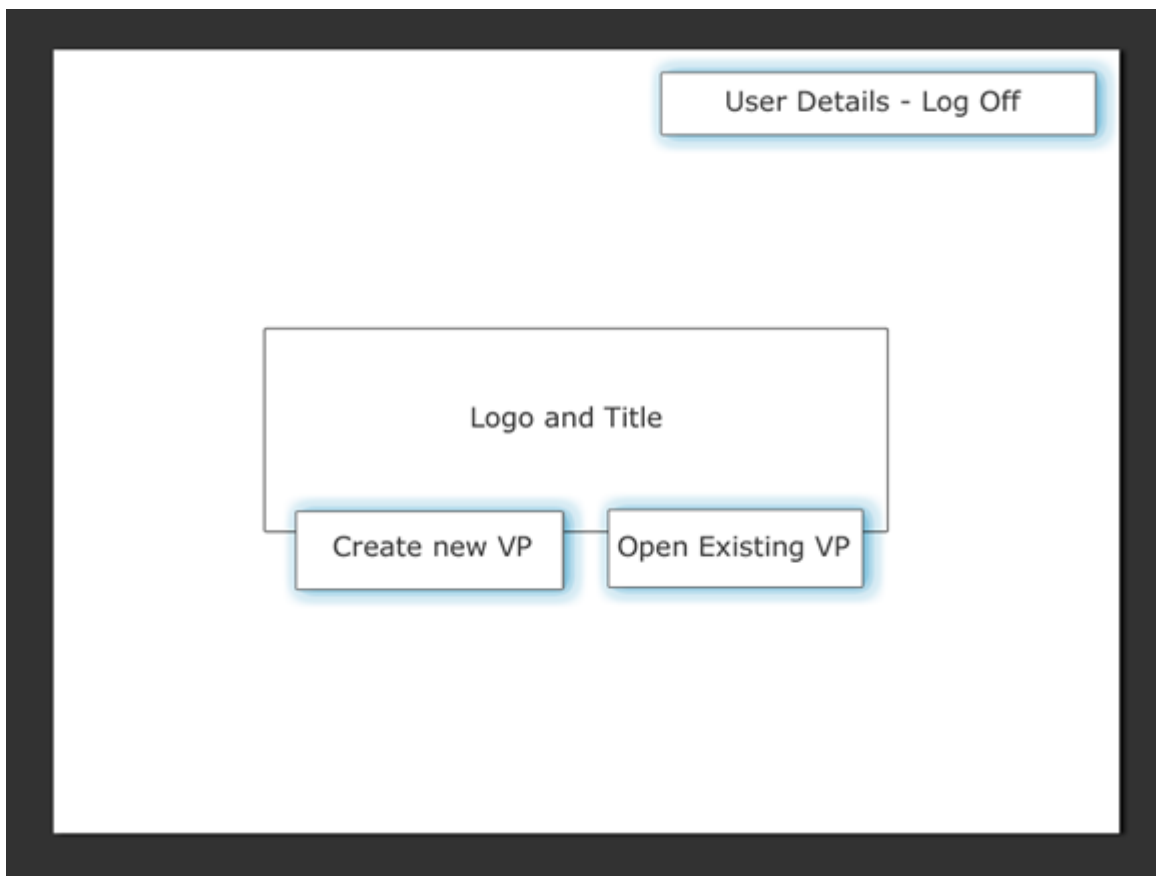


Fig. 11: Wireframe of the main central screen for the next generation of VS systems

A key aspect of this screen is that the interface is contextually aware; if the user is logged in as an author, the open existing VP button will respond differently than if the user is a learner - bringing up a set of cases with author permissions as well as those for playing.

Upon selecting "Open Existing VP", a screen would display that lists the VPs available, along with further information that allows the user to more usefully browse the list. This screen would include when a VP was most recently accessed, immediately providing at-a-glance information that would help sort between archived and current versions of duplicated cases.

Changes are suggested to the visual editor interface as shown in figure 12. This describes a number of changes to the interface, with the biggest change giving authors access to see and edit the detail of the node concurrently with the view of it as part of the wider map, allowing an overview of the general context of the node within the case to inform the editing process more closely.



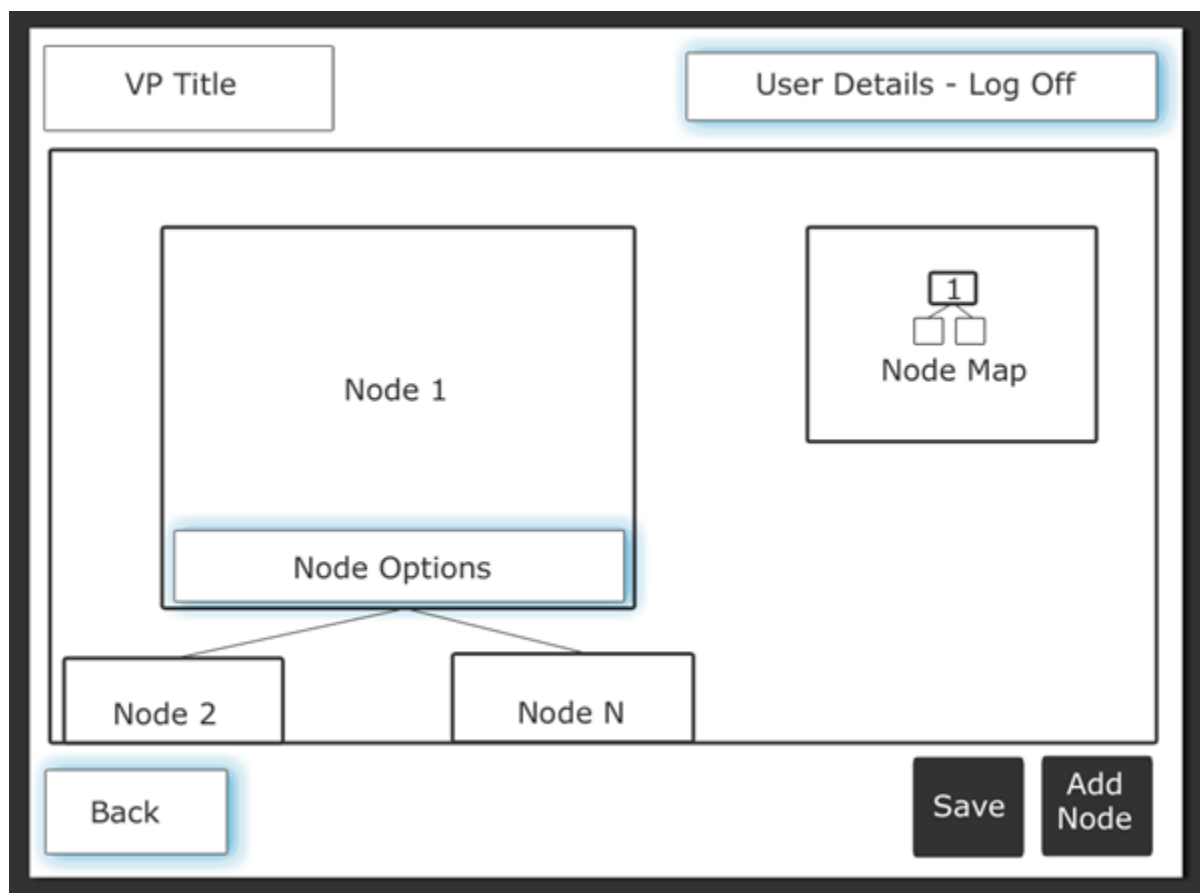


Fig. 12: Wireframe of the visual editor interface for the next generation of VS systems

When considering the “player” aspect of the interface, the general interface is mostly unchanged. However, all extraneous information has been removed, purely leaving the content, the options, and control options to exit the scenario or the system on screen.

However, consideration is given to the use of xAPI statements as analytics that can be generated from the system. These analytics would address usability issues for useful information that would increase the utility of the VS cases for learners. It was this ability to review the patient pathway that drew most comment from the novice users in their review of the system, reinforcing the importance of being able to review the learners pathway easily and effectively at any point, without necessarily influencing the future direction of the case.

Similarly, the comments suggested a context aware component to the opening screen that would detect a users status as a first-time user of the system upon their initial login. These users would be presented with an initial help screen that provides key information, and access to a demonstration scenario as guidance.

### 3.4.2.4 Discussion and Recommendations

The process of developing and testing the wireframes has yielded a number of useful and interesting insights into the challenges that are faced by learners and authors of VS systems alike, and proposed suggestions for how to resolve these. In many instances, these can be related directly to usability challenges that have been identified in the heuristic evaluation, and are coherent with the guidance provided for in Nielsen's heuristics.

One key point noted was the importance of high quality content. Unrelated to the system interface itself, the existence of typographical or spelling errors can be hugely distracting for learners, and indicative of a lack of care that can negatively impact on perceptions of the educational value of the resource. However, these are not entirely free of influence by the system; a poorly designed authoring interface can distract the author from their primary objective of conveying content, allowing mistakes to slip through. Similarly, where appropriate, the inclusion of proofing tools in text editors can be beneficial for some users.

Another noted challenge that all involved commented upon was the importance of a tool for reviewing the learner pathway. This is hugely important for learners when faced with decision-making elements as in VS cases; the benefit of a simulation in a safe-space is that content previously received can be reviewed as a new data-gathering exercise, looking for additional information or context that may have previously been considered irrelevant or superfluous, but which could now be used to greatly inform future decisions. The current system provided, using simple text links, was not sufficiently information-rich to offer the capability of effectively reviewing that information. The proposals put forward by end user participants would help in some contexts, but providing simple colour coded information about successful decisions would not be suitable for VS in some contexts. It will work when a decision is binarily successful (correct, or incorrect) but could not convey the required nuance when a case has multiple options of varying degrees of "correctness"; in teaching paradigms such as Problem-Based Learning, the suitability of choices is often somewhat subjective and the delivery of feedback is delayed until late on in the case.

For these reasons, we would propose that a more information-rich view of the case for reviewing the pathway would be preferable, and that the Pathway Visualisation work described further in this deliverable would offer a potential way of doing this. By showing the pathway taken as emerging nodes on a network diagram, learners will have a greater amount of information to be better able to reflect on their decisions. There would of course be technical and practical challenges to achieving this, but a proposal that this functionality would be beneficial is a key output of this task.

Ultimately, the wireframes produced provide a guideline and increased information for system designers to use as the basis for creating cases. Inevitably, changes are likely required to accommodate specific system functionality, but understanding the importance of these key concerns on behalf of users can inform system developers when creating these interfaces. The forthcoming version of OpenLabyrinth, which introduces far more customisable interface designs that can be used by case authors, as well as an overall rebuild of the main authoring interface, means that these insights are very timely, and will be provided to the OpenLabyrinth Development Consortium as one of the key lessons learned from this deliverable.

### 3.5. Interactions and decision-making training support

#### 3.5.1. MCQ usability

##### 3.5.1.1 Introduction

After some informal feedback from the WAVES project partners we realized small usability glitches on multiple choice question (MCQ) feedback. MCQ is still one of the most important question types used in VS in CASUS.

The display of feedback after answering an MCQ was realized in columns for the user answer and the expert answer. In this approach we received complaints that it is hard to see if a learner made correct or wrong choices. This led to a new approach which emphasized the “correction” of user answers with correct or wrong icons and tooltips and highlighting the expert choices differently. In this solution we received complaints, arguing that it is not easy to understand whether the icons refer to the correction OR the expert choice. Thus, it became obvious that more research is needed to find out how users could better understand the feedback.

##### 3.5.1.2 Methods

First, a small comparative evaluation of other systems was conducted and we found a variety of mechanisms, most of these looked familiar and were implemented in recent versions of CASUS.

Thus, we decided to implement a testing round with students. The test was done with a non-medical question (Fig. 13), consisting of seven items and three correct answers. The task for the student was to answer the question, and interpret the feedback of the system understanding :

- Which answer items were submitted correctly/incorrectly
- Which answer items were the correct solution

Measurements:

- “Think aloud” approach
- Time on task
- After completion of the task participants were asked about their thoughts on potential improvements
- The task was part of a usability test session containing also other tasks not related to the multiple choice feedback.

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

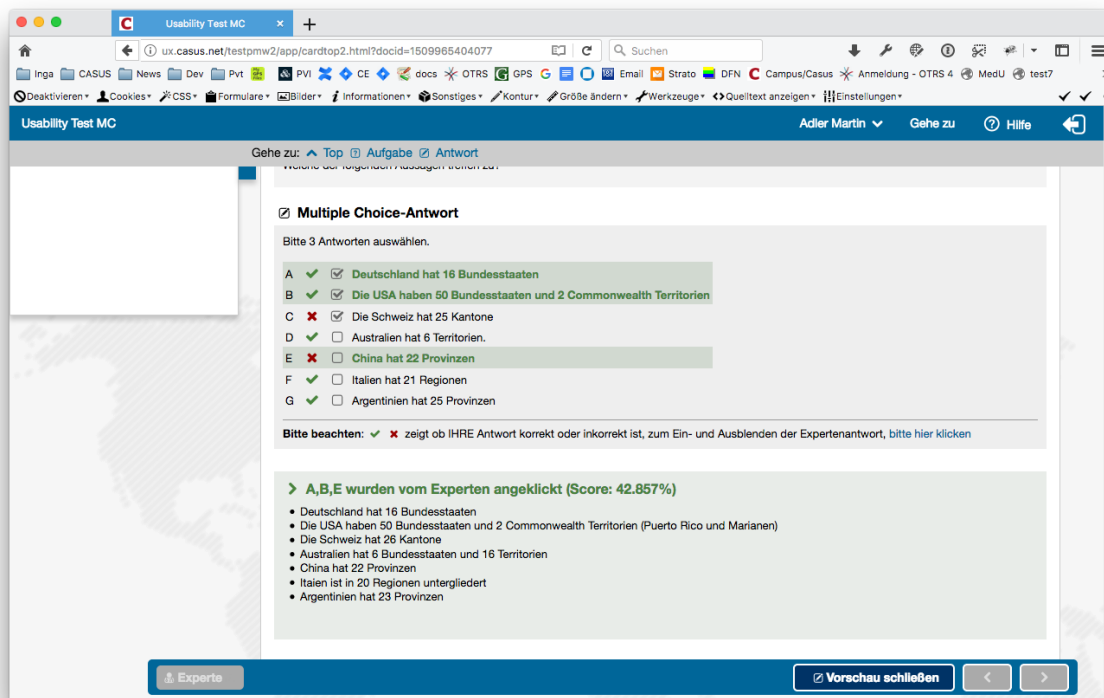


Fig. 13: Exemplary screenshot of the test scenario

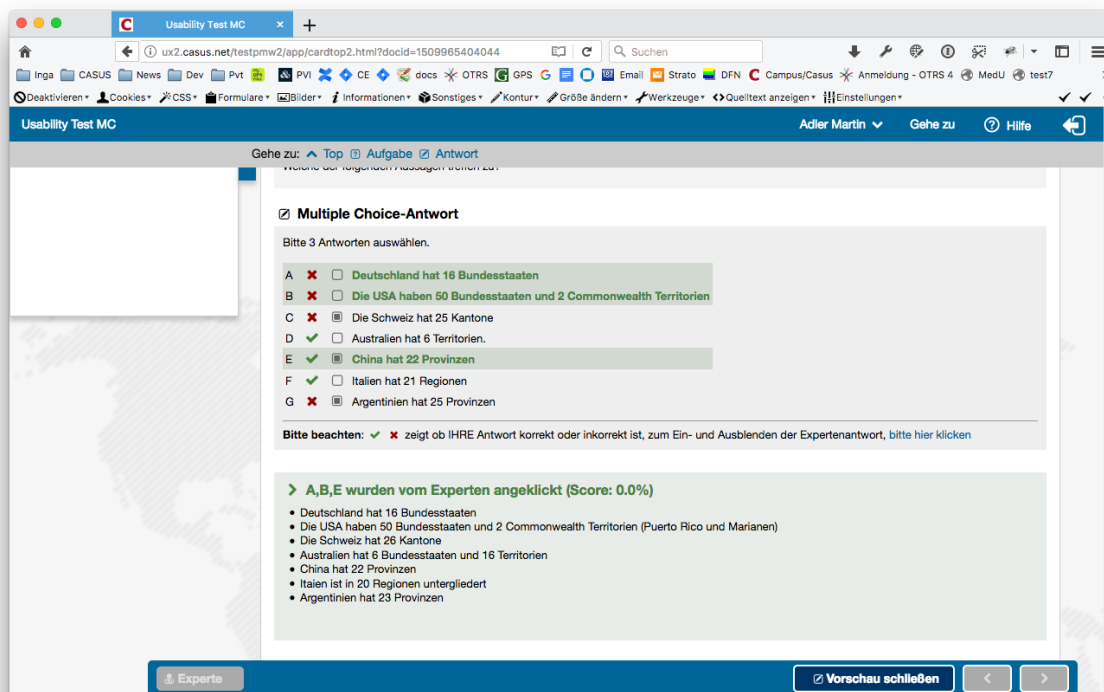


Fig. 14: Alternative display 1 (shown as a screenshot on paper)

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

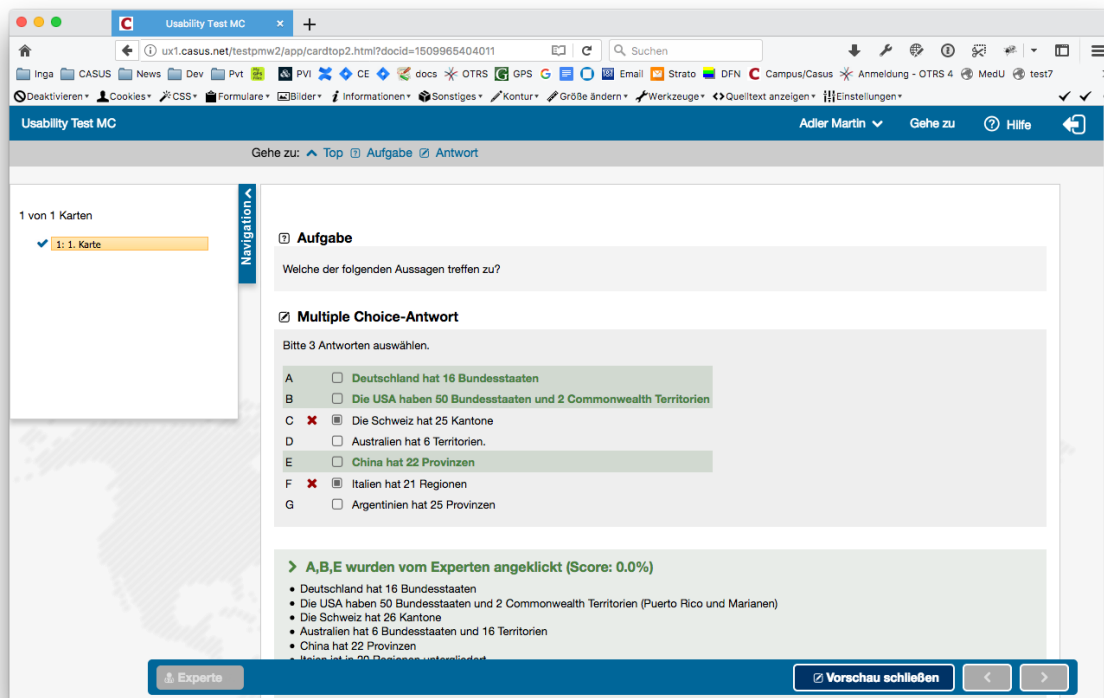


Fig. 15: Alternative display 2 (shown as a screenshot on paper)

### 3.5.1.3 Usability Test Results

#### Session 1

Participant: Medical student (female), preclinical, familiar with CASUS

Task 1: Answer a Multiple Choice Question (number of correct choices was displayed) and interpret feedback

- Question was answered without difficulties (2 out of 3 correct)
- Student mentions that display of how many answers are correct is helpful (points to display **below** answer text)
- Question 1: Which answers are correct/wrong
  - Student a bit confused in the beginning, but in second try answers correctly
- What is the expert solution
  - Expert solution is clear
  - Expert solution in the answer comment is helpful

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

- Overall the student has used CASUS cases (with a lot of MCQs) and thinks the interface of the feedback is intuitive.
- The student has never used the button to hide/show expert solution before (it was not quite clear whether she did not see it or saw no need to use it)
- The student thinks a prompt to click on hide/show expert solution is not necessary, since interface was clear before, but it might be helpful for students who have never used CASUS before.
- The student has no concrete suggestions for improvement
- When shown alternative interface 1 (Fig. 14), the student thinks that this might enhance the clarity of the interface

### Session 2

Participant: Medical student (male), clinical, knows CASUS, has completed a lot of VS

Task 1: Answer a Multiple Choice Question (number of correct choices was displayed) and interpret feedback

- Question was answered without difficulties (2 out of 3 correct)
- The student can quickly describe his correct choices and the expert solution
- The student did never use the hide/show expert link, he never saw the need for this.
- When shown alternative 1 (Fig. 14), the student sees no real difference to the current solution
- The student does not look to the expert solution in the answer comment ("everything else is in the question, so I never look there")
- The student has no idea how it could be changed to be more intuitive, he acknowledges that it is a complex process/feedback, which might be initially complex for new users
- Overall the student thinks CASUS VS are useful, but he only uses them when he has time (in times in which he does not have a full timetable)

### Session 3

Participant: Educator (female), no previous experience with CASUS

Task 1: Answer a Multiple Choice Question (number of correct choices was displayed) and interpret feedback

- Answering a question is not a problem, it was clear how many answers were correct

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

- The educator struggles to see the correct answers, but in second try she described it correctly
- The educator mentions that it is confusing to get green checks for answers that have not been selected previously
- The educator asks "Who is the "expert"? A better name might be "Correct answers"
- The educator thinks it would be less confusing to have the same symbol for the given answers from the beginning
- The educator prefers alternative 2 display (Fig. 15)
- The educator mentions that it would be best if the design of selecting the answers is the same as the feedback, so when using checkboxes with just filled box instead of a checkbox, that should be same before and after submitting an answer.

### 3.5.1.4 Conclusion

The next version of CASUS will have alternative display 1 (Fig 14) as the MCQ feedback, and we will be more careful using check marks, as their meaning depends on the context and precise user thinking model. The usability tests gave us a lot of details, regular usability tests on crucial parts of the system will be conducted and can be strongly recommended for other scenario based learning systems.

## 3.5.2. Pathway visualisation

### 3.5.2.1 Introduction

Feedback is a fundamental part of a learning activity, which allows the learner to reflect on the learning outcomes to be reached and compare it with the current status of the target competencies. For particularly important learning outcomes more than one cycle of provided feedback might be required. Feedback is known to enable self-directed learning and many different feedback models can be found in the literature [Narciss08].

In the context of virtual scenarios and particularly in branched models, feedback may facilitate the learners to elaborate on the decisions made and in case of a wrong one, allow them to return back to the critical task to be performed and take a new decision. Positive feedback can also reinforce right decisions and motivate the learners.

Within the OpenLabyrinth system the learner can navigate within the scenario and select different options. There is also the possibility for the learner to select the "review your pathway" button to display visited screen cards and navigate back to an earlier decision point. However, at the current stage of development of OpenLabyrinth there is no function available for the student to see the whole case including all available decision points and the optimal or correct pathway. Based on the opinion voiced by the learners and educators in the WAVES network (previous section of this deliverable), an overview of the decisions made in a scenario would be welcomed for instance in the debriefing phase of the learning experience.

As a way to achieve learners' awareness of the decisions made in the OpenLabyrinth system we aimed to enable the student when the end of the case is reached to view the whole structure of the scenario in a flowchart diagram that illustrates all nodes of the scenario and the selected options (visited nodes) of the learner. By that the learner will be able to gain a clear overview of the virtual scenario on the whole, reflect on the total available options / required decisions and on the choices made.

We hypothesise that this extension will enhance the usability of the OpenLabyrinth system by reducing the cognitive load in an intensive hypertextual learning environment (so called "Lost in hyperspace" phenomenon) by displaying a scenario map. This will leave more cognitive resources to facilitate the learner to train his/her decision making and critical thinking skills.

### 3.5.2.2 Methods

We explored the literature for different visualization methods that allow interactivity and are dynamic. We identified diverse JavaScript libraries such as D3.js [Meeks15] to allow manipulating documents based on data and sigma.js for graph drawing. We selected the D3.js to extend OpenLabyrinth by a pathway visualization tool. The tool generates a dynamic html 5 graph canvas based on the virtual scenario content in OpenLabyrinth. The scenario data are acquired from the database in the JSON format. This has the additional benefit of being loosely coupled with the current OpenLabyrinth database scheme which is going through major changes while transforming to the fourth generation of the platform. This makes this component also potentially useful for other branched virtual scenario systems which could integrate it as a micro-service.

### 3.5.2.3 Implementation

The pathway visualizer tool (fig. 16) is executed at the end of a scenario, for a single user session, when the final node of the scenario is visited. The graph contains a node for each screen card and the links among them showing the possible transitions between the states in the scenario. The tool highlights all visited nodes from a single session of the user (session\_traces) in a different color. In the visualization in fig. 13 the blue nodes indicate the visited screen cards, and the white nodes not selected options.



### Conclusion

powered by  
OpenLabyrinth

Well done! You have now reached the end of the case. As anticipated, Mrs Violet still has some problems with leakage via the urethra, especially during the night. However, she does not regret her choice of orthotopic neobladder. She still meets her friends for dinner and to play cards. She quit smoking immediately when she found out that smoking increases one's risk for bladder cancer.

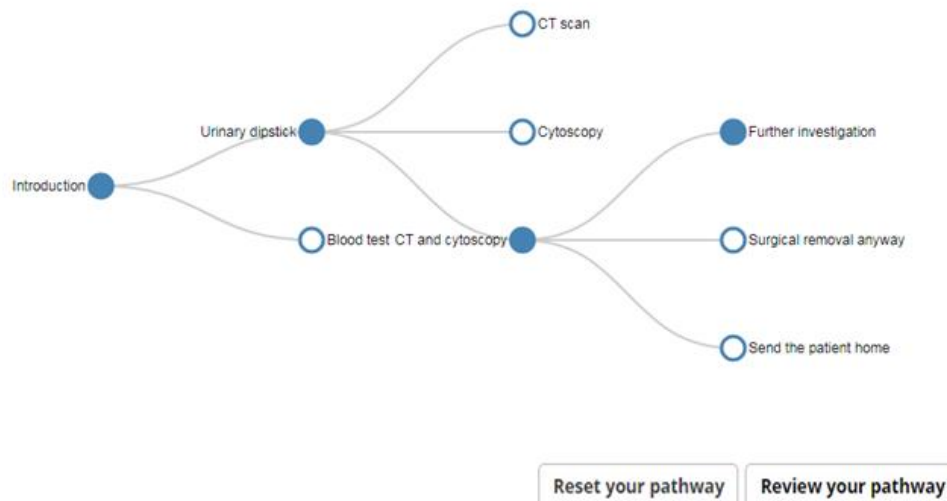


Fig. 16: Layout of the learner pathway visualisation component

### 3.5.2.4 Future work

Future developments will add more functionality to the visualisations by allowing the learner to see the content of individual scenario steps when selecting in the graph a particular node. The display can be divided into two parts: depicting the graph and the node's content in two separate panels.

In the current implementation the color of the node indicates the fact whether the node was visited in the last session by the learner or not. This can be further enhanced by alternative views - for instance showing:

- the pathway recommended by the instructor;
- hand-picked nodes selected by the teacher as representing critical decisions to pay particular attention;
- type of decision node (e.g. information collection, diagnostic/classification task, decision on further course of action);
- popularity of particular opinions among peer students;
- scores collected in the nodes from different peer learners;
- time spent on considering each option, etc.

### 3.6. Just-in-time help system

#### 3.6.1. Introduction

The goal of this task was to enrich the virtual scenario delivery systems with contextualized help/tutorial system which would make navigation for novice users easier. The basic idea was to avoid annoying users with wordy manuals and instead offer them clear help entries for each non-trivial UI element to be used at the moment they may need it.

Tooltips (interactive hints) and guided tours (interactive step-by-step guides) were considered as potentially useful tools to attain the goal. Tooltips are widely used by the developers of all kinds of software and there was found nothing special to recommend besides making them an integral part of user-friendly interfaces for learners and educators.

The guided tours are not so widespread and the unveiled use cases indicated that the step-by-step guides, which optionally appear on each page of an interface, form an ideal way to introduce key features that might be needed by the users working with the interface.

One of the already tested use cases of a guided tour was the “Current survival estimator” tour developed previously by MU (see <http://opencpu.iba.muni.cz:8080/shiny/current-survival/>) for scientists aiming at analysing datasets describing patients suffering from chronic myeloid leukaemia, see figure 17. Although this data analysis tool is not trivial at all, the embedded guided tour made it possible to navigate users through the interface and accomplish the analysis of an own dataset without the need to read any user manual. Thus, it served as an inspiration for our further developments aiming at a prototype of a guided tour embedded into the OpenLabyrinth virtual scenario delivery system.

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

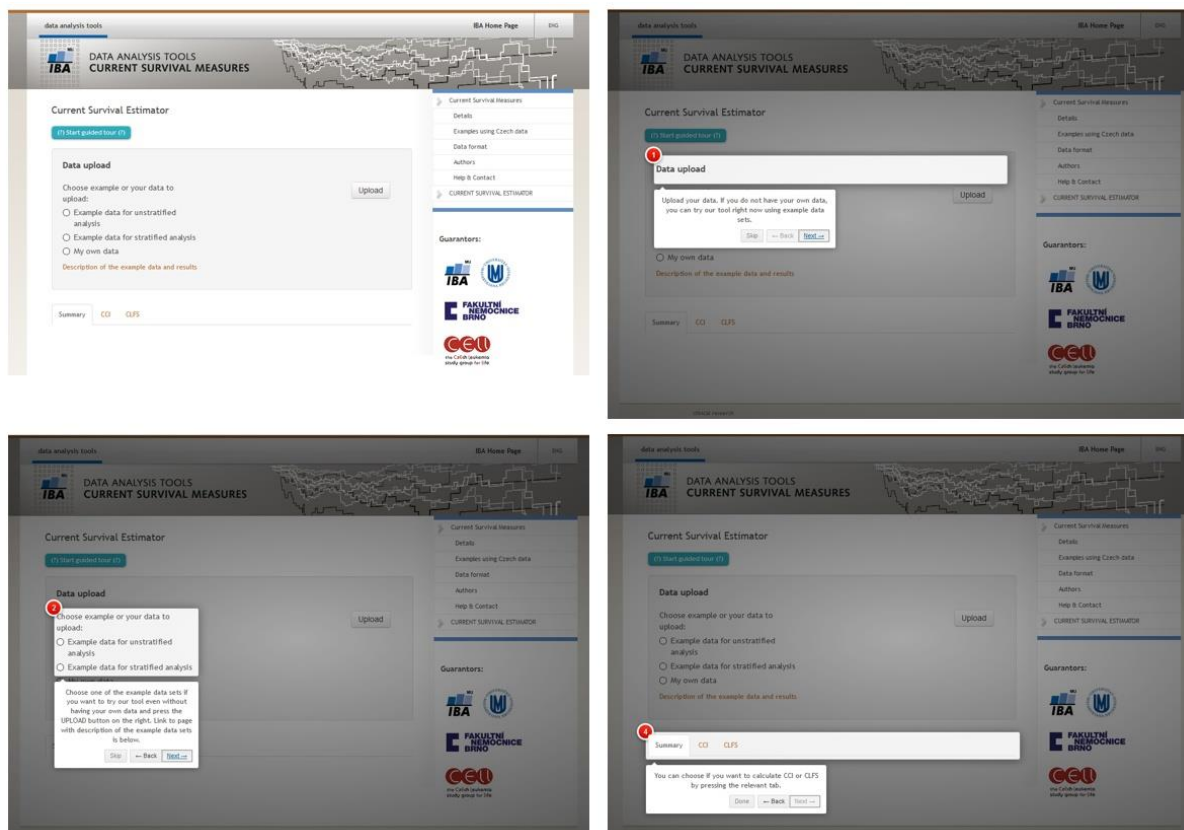


Fig. 17: The Guided tour embedded in the “Current survival estimator” (an online data analysis tool) tour was indicated as a suitable way to introduce key features of user interface and to provide users with a step-by-step guide.

### 3.6.2. Methods and tools

- INTRO.JS – open-source JavaScript library enabling web and mobile developers to create step-by-step guides easily.
- ver. 2.7.0
- <https://introjs.com/>, <https://github.com/usablica/intro.js>
- Commercial license: <http://introjs.com/#commercial>, open-source license: GNU AFFERO GENERAL PUBLIC LICENSE

EDITOR.PHP – open source compact database editor

- ver. 4.3.1
- <https://www.adminder.org/>
- License: <http://www.apache.org/licenses/LICENSE-2.0> Apache License, Version 2.0, license: <http://www.gnu.org/licenses/gpl-2.0.html> GNU General Public License, version 2 (one or other)

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

### OPENLABYRINTH – virtual scenario delivery system

- ver. 3.x
- installation: <http://ol.mefanet.cz> (for the use by medical faculties under the umbrella of the MEFANET project)

### MYSQL – open-source relational database management system

- ver. 5.7.x
- <https://www.mysql.com/>

### 3.6.3. Guided tour implementation

The source codes needed to embed Guided tours into OpenLabyrinth 3.x are provided in the ***guided\_tour\_sources.zip*** (3.8 MB). This package includes:

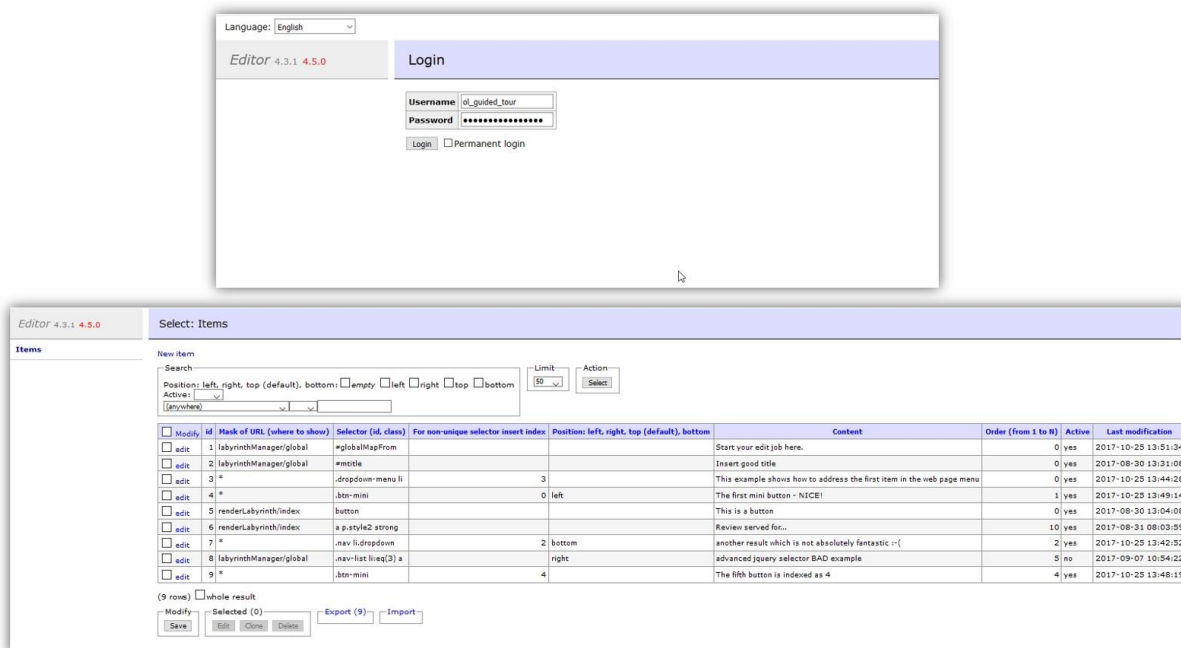
- ***readme.txt*** – step-by-step instructions for developers or OpenLabyrinth administrators.
- ***ol.mefanet.cz.zip*** – files which need to be modified in the OpenLabyrinth package; the files are organized in the same structure (subfolders) as commonly found in OpenLabyrinth 3.x.
- ***database.sql*** – a SQL script which creates ‘items’ table in the database underpinning an OpenLabyrinth installation.
- ***intro.js-2.7.0.zip*** – the above mentioned JavaScript library enabling step-by-step interactive guides.
- ***editor.php*** – the above mentioned compact database editor which was used here as a backend component for administration of the help/tutorial entries, i.e. editing the ‘items’ table created with ***database.sql***.

The figure 18 demonstrates the backend component of the Guided tours with various help/tutorial entries. It represents nothing more than an editor to the ‘items’ table. Each help/tutorial entry presented along the Guided tour is defined by the following attributes:

1. URL mask – to select the page of the particular user interface. The interfaces designed for learners or educators/authors can be distinguished by means of the URL mask.
2. Selector – ID or CLASS of an element defined by the CSS styles applied to the page of the particular user interface.
3. Index – to be used for selectors/elements with multiple appearance on the page (e.g. series of “Play” buttons).
4. Position – left, right, top, bottom position of the hint relative to the position of the element/selector.
5. Content – hint to be shown at the position near the element.

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

6. Order – definition of the step sequence in the tour.
7. Active – status of the item; only the active items are rendered as the tour steps.
8. Last modification – developer's placeholder for a potential future use.



The screenshot shows the Labyrinth Editor 4.3.1 4.5.0 interface. The top section displays a 'Login' form with fields for 'Username' (set to 'ol\_guided\_tour') and 'Password' (masked with asterisks). Below the password field are 'Login' and 'Permanent login' checkboxes. The bottom section, titled 'Select: Items', shows a table editor for the 'Items' table. It includes a 'New item' form with fields for 'Mask of URI', 'Selector', 'Position', and 'Content'. The table below has columns: 'id', 'Mask of URI (where to show)', 'Selector (id, class)', 'For non-unique selector insert index', 'Position: left, right, top (default), bottom', 'Content', 'Order (from 1 to N)', 'Active', and 'Last modification'. The table contains 9 rows of data.

	id	Mask of URI (where to show)	Selector (id, class)	For non-unique selector insert index	Position: left, right, top (default), bottom	Content	Order (from 1 to N)	Active	Last modification	
<input type="checkbox"/> edit	1	labyrinthManager/global	#globalMapFrom			Start your edit job here.	0	yes	2017-10-25 13:51:34	
<input type="checkbox"/> edit	2	labyrinthManager/global	#mtitle			Insert good title	0	yes	2017-08-30 13:31:08	
<input type="checkbox"/> edit	3	*	.dropdown-menu li	3		This example shows how to address the first item in the web page menu	0	yes	2017-10-25 13:44:28	
<input type="checkbox"/> edit	4	*	.btn-mini	0	left	The first mini button - NICE!	1	yes	2017-10-25 13:49:14	
<input type="checkbox"/> edit	5	renderLabyrinth/index	button			This is a button	0	yes	2017-08-30 13:04:08	
<input type="checkbox"/> edit	6	renderLabyrinth/index	a.pstyle2 strong			Review served for...	10	yes	2017-08-31 08:03:59	
<input type="checkbox"/> edit	7	*	.nav li.dropdown		2	bottom	another result which is not absolutely fantastic -{	2	yes	2017-10-25 13:42:52
<input type="checkbox"/> edit	8	labyrinthManager/global	.nav-list li:eq(3) a		right	advanced jquery selector BAD example	5	no	2017-09-07 10:54:22	
<input type="checkbox"/> edit	9	*	.btn-mini	4		The fifth button is indexed as 4	4	yes	2017-10-25 13:48:19	

Fig. 18: Editing of the help/tutorial entries presented during the Guided tour can be accomplished by editing the 'items' table. Here the compact database editor was used.

The steps of the Guided tour defined in the 'items' table are transformed into JSON data echoed from PHP script ***application\views\introjs\_hook.php***. This is the only one PHP script which has to be included - with the PHP function *require\_once()* - into the OpenLabyrinth's code: ***application\views\home.php***.

The figure 19 demonstrates some parts of the resulting guided tours embedded into home page, labyrinth editor and labyrinth player.

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

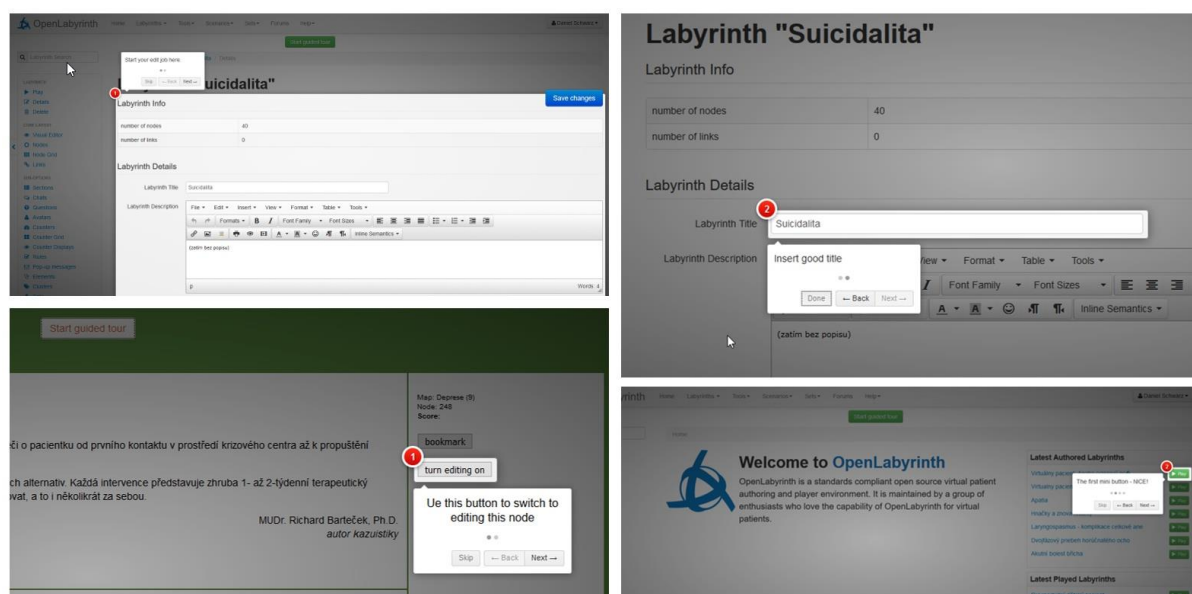


Fig. 19: Some parts of the three guided tours embedded into the home page (bottom right), the virtual scenario editor and player. The hints presented along the tour were defined by means of the 'items' table, which was added to the the database underpinning the OpenLabyrinth installation.

### 3.6.4. Limits of the prototype implementation

There are several limits in our prototype implementation, which makes embedding the guided tour into the OpenLabyrinth 3.x quite challenging (i) The biggest pitfall lies in the impossibility to address all elements of the pages accurately – this regards specially to the elements which had to be addressed by element CSS selectors and class CSS selectors. On the other hand, there were no problems with addressing the elements with ID CSS selectors. (ii) Some problems with readability were detected in horizontal menus defined with <UL></UL> lists and complex CSS styling. (iii) A minor problem was represented by the fact that we were not always able to influence the relative positioning of the Guided tour hints relatively to the elements/selectors. (iv) In case of large elements (e.g. a large text area which requires page scrolling), the Guided tours may even cause severe aggravation of the user interface usability.

### 3.6.5. Conclusion

Guided tours were found to be an ideal way to navigate users through complex interfaces without need to read wordy manuals. Several useful open-source software pieces were selected to deploy a prototype implementation of the Guided tour embedded into OpenLabyrinth 3.x, such as: Intro.JS, Editor.php. A perfect integration of the Guided tours into OpenLabyrinth was not achieved. In case of complete rebuilding of the HTML/CSS code of the pages which form the user interfaces, as is currently the case while moving between the third and fourth generation of OpenLabyrinth, the integration would be, however, possible. Simple amendments to the presented code may provide support for multiple languages.

## 4. DISCUSSION

### 4.1. Impact

The goal of the deliverable was to develop a set of accessibility and usability enhancements for the two exemplar virtual scenario systems: CASUS and OpenLabyrinth. We have operationalized this goal by dividing the work into six tasks: three for accessibility and three for usability enhancements. The project has definitely achieved progress in each of those tasks, at the same time leaving some opportunities for further developments. This is not surprising considering that designing high quality human-computer interfaces is an open ended process that requires longitudinal efforts.

Despite the widespread of the English language worldwide, the needs analysis, as well as our observations show that lack of availability of virtual scenarios software in national languages form a barrier for some groups of learners. At the same time it is unrealistic to address all languages in one project. We approached this challenge by developing a tool that empowers the community to create new language versions by themselves. The translation tool presented above is fully operational and available as open source in the project repository. Its value was demonstrated while doing a complete revision of the Polish language translation of the CASUS user interface. As the system is based on UTF-8 character encoding, we do not see any reasons why such a translation would not be possible in any other European language. By working on JAVA property files - which are a standard i18n mechanism - the tool is open to use in other systems. By employing the powerful Google API for translation suggestions, the time of localization could be reduced. We can now estimate that a complete translation of the environment takes approximately 100 hours. This effort estimation can help in making informed decisions on embarking on localizing the interface to other national languages.

Another type of language related aspect other than translation or character encoding pertains to alignment of interface elements as in Right-to-Left languages. In the WAVES project, we have addressed this on the example of the Persian language, for which a pilot translation of selected parts of the CASUS user interface was prepared. This gave us the opportunity to experiment with different CSS-based transformations, and collect by that experiences on how to build presentation layers open for language extensions.

To assure the user interfaces are compliant with web accessibility guidelines (W3C WCAG) an automated validation with following refinements was performed for the CASUS user interface. This allowed us to achieve compliance with common screen readers opening virtual scenarios also for people with sight disabilities and potentially improving the user experience of people with situational limitations.

Finally, accessibility enhancements were made to fit the needs of mobile learners. As became clear from the needs analysis, more than 50% of the respondents wished to be able to learn with virtual scenarios on smartphones and tablets. A modern mobile theme for branched virtual scenarios (as in OpenLabyrinth) was developed within our project. Attention was paid not only to make it responsive to various resolutions even on small screens, but also to prevent unwanted actions by accidental selection of elements on touchscreens. These types of errors are less common for mouse and keyboard controls and needed special attentions of the developers. The



achieved outcome can have a positive impact for the “on the go” learners, but also those in developing countries with limited access to desktop computers.

The usability enhancements started with a more in-depth heuristic evaluation of usability issues of the OpenLabyrinth than was possible in the needs analysis. This and input of project partners not familiar with authoring of virtual scenarios resulted in elaboration of a new concept of arranging the user interface elements for the next generation of OpenLabyrinth system. The resulting wireframes, but also a set of functional requirements, were presented to the project partners to influence the developments of OpenLabyrinth.

One of the desired elements postulated to be developed after this usability inspection was a graphical view of the user pathway through the scenario to be presented in the debriefing phase of the learning process. A prototype of such module was developed using the D3.js library. The component is designed on an abstract graph model in a way that makes it potentially usable also for other than OpenLabyrinth branched virtual scenario players.

The heuristic evaluation also pointed out several limitations in the availability of help and guidance elements for novice users in the existing version of OpenLabyrinth. This deficiency can be addressed by another task completed within this deliverable which researched the possibilities of introducing to virtual scenario authoring tools a guided tour mechanism. Even though the way how the presentation layer OpenLabyrinth was designed in the third generation precluded a smooth user experience of such tours in the current version of the environment, the lessons learned regarding how to design the presentation layer are fed back and impact the design of OpenLabyrinth v4.

Finally, it was realized that even such common user interface element in educational environments as MCQ items with multiple correct answers (also known as Multiple Response Question), often present in virtual scenarios of various forms, may have some usability glitches. This was analysed on the example of the interactive elements in the CASUS virtual scenario player and resulted in improvements in the design of these elements presented in this deliverable.

## 4.2. Limitations

In spite of the positive outcomes and progress described in the previous section, we also feel that our work has limitations.

One of them is the fact that many of the changes proposed for OpenLabyrinth have not yet been implemented as a branch of the source code available in an open repository. The reason for that was the complete redesign of the system architecture to form the v4 of the environment mentioned already in the introduction to this deliverable. The bulk of the work in redesigning the architecture was done in time of the D2.2 developments. The first alpha version of the fourth generation was released by the end of November 2017 (<https://github.com/olab/OLab4>) which did not leave us enough time to implement the changes directly in the newly developed and still highly experimental source code. At the same time, knowing that OpenLabyrinth v3 architecture will be soon deprecated questioned the sense of introducing those changes as a branch of source code in the old architecture.



We solved this situation by focusing the efforts on developing modules loosely connected with main source code of the OpenLabyrinth (as in the case of the mobile theme or pathways visualisation); in developing wireframe models to illustrate the desired ways of developing the next generation of the source (as with the new layout design); or developing prototypes not yet reaching the expected level of usability (guided tours).

### 4.3. Outlook

As described earlier usability improvements is a longitudinal process requiring several iterations. The implemented enhancements have some degree of face validity and were pilot-tested, but will have to be more rigorously evaluated which is planned for D5.3 of this project. The new insights gained through this process will have to be followed-up by next cycles of improvements going even beyond the life-time of this project.

The loosely coupled modules strategy adopted when extending the functionality of OpenLabyrinth will require linking them with the core architecture through open APIs. This will be addressed as part of the D2.3 deliverable of this project.

It is unclear yet how sustainable the proposed extensions will be. It was our intention that the work initiated in this deliverable is continued by the community resulting for instance in new language versions created by the translation tools developed in this deliverable or tutorials for authoring of virtual scenarios implemented as guided tours.

## 5. CONCLUSIONS

The implemented accessibility and usability enhancements have potential to widen access to virtual scenarios for learners and educators from non-English speaking countries, people with sight disabilities and mobile learners. Furthermore, efforts were invested to improve the layout of user interface elements in the authoring tools, provide better overview of the learning process in branched virtual patients, more intuitive feedback in multiple choice questions and better learnability of the environment by guided tool. In the further stages of the project we will improve integration of the changes in the virtual scenario system and evaluate the extent to which those improvements affect the target audience.

## 6. REFERENCES

Brooke, John. *SUS-A quick and dirty usability scale*. Usability evaluation in industry 189.194 (1996): 4-7.

Intro.js, <http://introjs.com>, (accessed January 15, 2018).

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

Henry, S. L., Abou-Zahra, S., & Brewer, J. (2014, April). *The role of accessibility in a universal web*. In Proceedings of the 11th Web for All Conference (p. 17). ACM.

Meeks, E. (2015). *D3.js in Action*. Manning.

Narciss, S. (2008). Feedback strategies for interactive learning tasks. *Handbook of research on educational communications and technology*, 3, 125-144.

Nielsen, Jakob. *Severity ratings for usability problems*. Papers and Essays 54 (1995): 1-2.

Nielsen, J. (1995). *10 usability heuristics for user interface design*. Retrieved from <http://www.nngroup.com/articles/ten-usability-heuristics>.

Rubin, J., & Chisnell, D. (2008). *Handbook of usability testing: howto plan, design, and conduct effective tests*. John Wiley & Sons.

Tennant, Emily, Dino Anastasia, and Cara D'Amato. "iTunes Heuristic Evaluation Report." (2005).

Tullis, T., & Albert, W. (2013). *Measuring the user experience: Collecting, analyzing, and presenting usability metrics* (2nd ed.). Waltham: Elsevier.

Wang, F., & Hannafin, M.J. (2005). *Design-based research and technology-enhanced learning environments*. Educational technology research and development, 53(4).

## 7. APPENDIX A:

### Nielsen's Usability Heuristics [Nielsen95]

#### Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

#### Match between system and the real world

The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

#### User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

#### Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing.

#### Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

#### Recognition rather than recall

Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

#### Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

#### Aesthetic and minimalist design

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

### Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

### Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

The usability problems found from this evaluation are clustered into eleven areas and are ranked according to their severity and ease with which the problem can be solved.

Table 3. Severity rankings [Nielsen95]

Rating	Definition
1	Violates a heuristic but doesn't seem to be a usability problem.
2	Superficial usability problem: may be easily overcome by user or occurs extremely infrequently. Does not need to be fixed for next release unless extra time is available.
3	Minor usability problem: may occur more frequently or be more difficult to overcome. Fixing this should be given low priority for next release.
4	Major usability problem: occurs frequently and persistently or users may be unable or unaware of how to fix the problem. Important to fix, so should be given high priority.
5	Usability catastrophe: Seriously impairs use of product and cannot be overcome by users. Imperative to fix this before product can be released.

Table 4. Ease of fixing rankings [Tennant05]

Rating	Definition
1	Problem would be extremely easy to fix. Could be completed by one team member before next release.
2	Problem would be easy to fix. Involves specific interface elements and solution is clear.
3	Problem would require some effort to fix. Involves multiple aspects of the interface or would require team of developers to implement changes before next release or solution is not clear.

4	Usability problem would be difficult to fix. Requires concentrated development effort to finish before next release, involves multiple aspects of interface. Solution may not be immediately obvious or may be disputed.
---	--

## Open Labyrinth Heuristic usability evaluation summary of findings

Table 5. Open Labyrinth Heuristic usability evaluation summary of findings

#	Problem	Severity Ranking	Ease of Fixing Ranking	Heuristic Number	Broad Heuristic
1	Help page is totally missing	#4	#3	#10	Help and documentation
2	Buttons have no tooltips	#2	#1	#6	Recognition rather than recall
3	'Duplicate existing" button leads to home page				Consistency and standards
4	'Create step-by-step' leads to a page with a different title	#3	#1	#4	Consistency and standards
5	Collections page should have title 'My collections' (as in menu)	#3	#1	#4	Consistency and standards
6	Import Labyrinths-> VUE: titles are different in menu, page, breadcrumb	#3	#1	#4	Consistency and standards
7	VUE import and MedBiquitous pages: Upload buttons should have the same label, must be in the same position etc.	#3	#2	#4	Consistency and standards
8	Labyrinths-> Create Manually page shows different Path	#3	#1	#4	Consistency and standards
9	Export Labyrinths -> XML leads to home page				Consistency and standards
10	Export Labyrinths -> Medbiquitous VP page shows different path	#3	#1	#4	Consistency and standards

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

11	Export Labyrinths -> Medbiquitous ANSI page shows different Path	#3	#1	#4	Consistency and standards
12	Tools -> Manage Presentations page shows different Path	#3	#1	#4	Consistency and standards
13	Tools-> Manage Remote Services page shows different Path	#3	#1	#4	Consistency and standards
14	Tools->Manage Users and Groups page shows different path	#3	#1	#4	Consistency and standards
15	Tools-> System Settings page shows different path	#3	#1	#4	Consistency and standards
16	Tools-> Today's tips page shows different path	#3	#1	#4	Consistency and standards
17	Tools-> Manage Metadata page shows different path	#3	#1	#4	Consistency and standards
18	Tools-> Manage Scenarios page shows different path	#3	#1	#4	Consistency and standards
19	Scenario Management-> My Scenarios page shows different label path	#3	#1	#4	Consistency and standards
20	Scenario Management-> Manage Scenarios page shows different path	#3	#1	#4	Consistency and standards
21	Forums-> Add new forum button page shows different path	#3	#1	#4	Consistency and standards
22	APIAΔNH page shows different path	#3	#1	#4	Consistency and standards
23	Labyrinths-> Create Step-by-Step screen of step 5 has different title.	#3	#1	#4	Consistency and standards

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

24	Labyrinths-> My Collections has different path.	#3	#1	#4	Consistency and standards
25	Labyrinths-> import VUE submit button returns to home page	#3	#1	#4	Consistency and standards
26	Export Labyrinths -> Medbiquitous VP title is different in menu	#3	#1	#4	Consistency and standards
27	Export Labyrinths -> Medbiquitous ANSI title is different in menu	#3	#1	#4	Consistency and standards
28	Tools -> Manage Presentations titles are different in menu	#3	#1	#4	Consistency and standards
29	Tools-> Manage Remote Services renderLabyrinth instructions unclear.	#3	#1	#4	Consistency and standards
30	Tools-> Manage Remote Services Add a service leads to a page that shows the same path as before	#3	#1	#4	Consistency and standards
31	Tools-> Manage Remote Services Add a service leads to a page with different title	#3	#1	#4	Consistency and standards
32	Tools-> Manage Remote Services submit button returns to the previous screen without an update info	#3	#1	#4	Consistency and standards
33	Tools->Manage Users and Groups Add user button lead to screen with different title	#3	#1	#4	Consistency and standards
34	Tools->Manage Users and Groups title is different in menu.	#3	#1	#4	Consistency and standards
35	Tools->Manage Users and Groups Add Group button lead to screen with different title	#3	#1	#4	Consistency and standards

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

36	Tools-> System Settings page's all update buttons reload without update info	#3	#1	#4	Consistency and standards
37	Tools-> System Settings Support tab button return to the first tab Password Recovery Settings	#3	#1	#4	Consistency and standards
38	Tools-> System Settings OAuth tab button return to the first tab Password Recovery Settings	#3	#1	#4	Consistency and standards
39	Tools-> Today's tips Add a tip button lead to a screen with different title	#3	#1	#4	Consistency and standards
40	Tools-> Today's tips save button of the Add a tip screen lead to Edit tip screen.	#3	#1	#4	Consistency and standards
41	Tools-> Manage Metadata title is different in menu.	#3	#1	#4	Consistency and standards
42	Tools-> Manage Scenarios title is different in menu	#3	#2	#4	Consistency and standards
43	Scenario Management-> My Scenarios title is different in menu	#3	#2	#4	Consistency and standards
44	Scenario Management-> Manage Scenarios title is different in menu.	#3	#2	#4	Consistency and standards
45	Scenario Management-> Manage Create Scenario button page shows different path	#3	#2	#4	Consistency and standards
46	APIAΔNH titles are different in menu.	#3	#1	#4	Consistency and standards



## D.2.2. Accessibility and usability enhancements for virtual scenario systems

47	Labyrinths-> Create Step-by-Step Step button 1 lead to screen with different name.	#3	#1	#4	Consistency and standards
48	Labyrinths-> Create Step-by-Step Step button 2 lead to screen with different name.	#1	#1	#4	Consistency and standards
49	Labyrinths-> Create Step-by-Step Step button 3 lead to screen with different name.	#3	#1	#4	Consistency and standards
50	Labyrinths-> Create Step-by-Step Step button 4 lead to screen with different name.	#3	#1	#4	Consistency and standards
51	Labyrinths-> Create Step-by-Step Step button 5 lead to screen with different name.	#3	#1	#4	Consistency and standards
52	Labyrinths-> Create Step-by-Step step 3 button has false name	#3	#1	#4	Consistency and standards
53	Labyrinths-> import VUE submit button has no update info.	#3	#1	#4	Consistency and standards
54	'Vue to Labyrinth upload' button label should be "Upload" instead of "Submit"Top of FormBottom of Form	#1	#3	#4	Consistency and standards
55	'Vue to Labyrinth upload' when submitting empty form navigates back to the home page	#2	#4	#9	Help users recognize, diagnose, and recover from errors
56	Create scenario -> use existing forum function is broken: displays empty rows in the list of forums	#3	#3	#9	Help users recognize, diagnose, and recover from errors

## D.2.2. Accessibility and usability enhancements for virtual scenario systems

57	Create scenario -> list of users is way too long	#4	#3	#7	Flexibility and efficiency of use
58	Create scenario -> help tip is misleading ( <i>"There are no available webinars right now. You may add a webinar using the add button."</i> ).	#2	#2	#4	Consistency and standards
59	There is no consistency in the way buttons are label. E.g. some buttons have the label "create" and some other "Create"	#2	#3	#4	Consistency and standards
60	The "skin" field in forms has no visual preview of each skin.	#3	#3	#1	Visibility of system status
61	Form labels (everywhere) change the mouse cursor on hover to hand (similar to links)	#3	#3	#4	Consistency and standards
62	Form labels on click focus on some elements (e.g. text fields), while on some have no obviously functionality	#3	#2	#4	Consistency and standards
63	Manage presentations allows users to add totally empty records. Later on, the preview buttons shows nothing	#4	#2	#7	Flexibility and efficiency of use

## 8. APPENDIX B:

Instructions shared with end users to guide them to complete the two tasks namely 'Playing a virtual scenario' and 'Authoring a virtual scenario' (task U1).

### Playing Virtual Scenario in OpenLabyrinth

#### Objective

- To play and complete a virtual scenario in OpenLabyrinth
- Access case at URL.
- Read through the information on the pages thoroughly and choose the most appropriate answer from given options.
- Complete the case.
- Identify the score that you receive, and any feedback received as part of the given case.
- Return to a decision that was made that was incorrect and revisit that decision. Describe how you did this.

### Authoring a virtual scenario in OpenLabyrinth

Please record any comments or usability issues as you encounter them. Please also suggest ways to improve the process.

#### Objective 1

- To author a virtual scenario in OpenLabyrinth
- Access your account in OpenLabyrinth.
- Enter basic information of the scenario/case intended to be authored.
- Make the case private to registered authors and users.
- Access the visual editor screen, create nodes and pathways in a logical manner.
- Add title and text to all created nodes. Add additional nodes as necessary.
- Save all changes as required on node and map.
- Grant co-author permission to Stephen/Lindsay.
- Launch the authored scenario in browser.

### **Objective 2**

- To export and import a case in OpenLabyrinth
- Access your account in OpenLabyrinth.
- Identify the authored case.
- Export it as Medbiquitous VP file to your computer.
- Re-import this Medbiquitous VP as a new case on OpenLabyrinth system.

### **Objective 3**

- To a make a copy of a case in OpenLabyrinth
- Make a copy of your authored case.

### **Objective 4**

- To add an image file to a node in your existing case in OpenLabyrinth
- Identify your authored case.
- Upload and embed image to any node within the case.
- View the image within case in browser.