STREAMING OF THEATRE AND ARTS FOR OLD AGE ENTERTAINMENT

D2.1 SYSTEM ARCHITECTURE, TECHNICAL REQUIREMENTS AND SPECIFICATIONS

VERSION NUMBER: 2.0
DISSEMINATION LEVEL: CO
LEAD PARTNER: SIVECO ROMANIA SA
DUE DATE: 27.10.2017
TYPE OF DELIVERABLE: DOCUMENT
STATUS: FINAL

AAL PROGRAMME

This project has received funding from the Active and Assisted Living Research and Development Programme.

Published in the framework of:

STAGE – Streaming of Theatre and Arts for Old Age Entertainment

STAGE website: www.stage-aal.eu

Authors:

Dragos Papatoiu, SIVECO Romania SA

Leonardo Chiariglione, CEDEO

Lavinia Panait, SIVECO Romania SA

Ilias Kapuranis, GEORAMA

Revision and history chart:

Version	Date	Editors	Comment
0.0	13.10.2016	SIVECO	Structure outline
0.1	21.11.2016	SIVECO	Added introduction, technical specifications and system architecture, initial version
0.2	22.11.2016	GEORAMA	Added user interface requirements
0.3	31.01.2017	SIVECO	Updated technical specifications and system architecture, methodology
0.4	06.02.2017	GEORAMA	Added Usability Specifications
1.0	17.02.2017	CEDEO	Added Video Platform specific information
1.1	31.08.2017	SIVECO	Updated the STAGE Portal specific information
1.1	18.10.2017	GEORAMA	Added 2 rows in the Section 5 table.

1.2	25.10.2017	CEDEO	Updated the Video Platform specific information
1.3	26.10.2017	SIVECO	Updated the STAGE Portal specific information
2.0	27.10.2017	SIVECO	Reviewed final version of the document

Table of content

0. Abbreviations and acronyms

AAL: Active and Assisted Living Programme
CMU: Central Management Unit
CA: Consortium Agreement
DoW: Description of Work
GA: General Assembly
IPR: Intellectual Property Rights
NCP: National Contact Point
PM / PMP: Project Management / Project Management Plan
SC: Steering Committee
WP: Work package
GUI: Graphical User Interface
UCD: User Centered Design
TL: Technical Leader
SD: Software Developer
SVN: Subversion
PM: Project Manager

1. INTRODUCTION

1.1. *Note on post-MTR update*

This deliverable was updated to reflect comments and indications found in the Mid-Term Review report.

The table below details the updates:

MTR Comment	Addressed in
This is an interesting statement for a spec and does not provide confidence that the consortium are thinking beyond a prototype platform for simple end user testing: For each of the above, once the developer discontinues the support and/or security updates, the STAGE Platform cannot ensure full functionality and performance.	✓ Section 3.3.2, page 30-33 ✓ Section 3.3.3, page 33
It does state that it will be configured for use on either a desktop or tablet, and then the Smart TV, which is not in line with the user requirements, where TV was the most popular and tablet the least popular.	✓ Section 3.3.2, page 30-33
It is not clear whether they considered what would happen if the end user did not have the correct processing capability on their device?	✓ Section 3.3.2, page 30-33

1.2. *Scope and objectives of the deliverable*

Deliverable ***D2.1 System architecture, technical requirements and specifications*** covers all aspects and findings related to tasks T2.1 Technical requirements definition and T2.2 Architecture and Interoperability, and, more specifically, aims to address all aspects related to the requirements analysis, communication requirements, interoperability protocols, topologies based on power capabilities and constraints, real-life spatial requirements (e.g. distances, remote locations) and technological standards of the technologies involved in STAGE solution.

1.3. *Structure of the deliverable*

Section 1, Introduction, contains a summary regarding the purpose and objectives of the deliverable and the proposed structure.

Section 2, Methodology, aims to describe the relevant aspects regarding the adopted methodology in this phase.

Section 3, Technical specifications, aims to present the functional, non-functional as well as the security related requirements of the STAGE technical solution. This section will also explore the proposed technologies and software tools that will be used in the implementation of the STAGE technical solution.

Section 4, System architecture, aims to present the use-cases, as well as the significant aspects regarding the architecture design of the STAGE technical solution.

Section 5, User Interface requirements, aims to present the significant aspects related to the design and the specific requirements of the user interfaces.

2. METHODOLOGY

The project consortium's aim is to bring together the best practices necessary to deliver the project objectives on time and budget, according to agreed specifications and user expectations.
The present section details the development methodology along with the proposed phases.For each phase, the objectives as well as the activities which will be performed will be presented.

The main phases of the development flow are detailed in the following paragraphs.

2.1. Detailed Design

The methodological approach related to the analysis and design phases was detailed in deliverable *D1.2 User requirements definition and analysis.*

2.2. Coding

The process of coding is composed of the steps detailed below.

2.2.1. Receive Task

For each task of the phase from the task schedule, the designated Technical Leader (TL) creates a JIRA incident for the Software Developer (SD) assigned for the task (technical tasks including coding are assigned by TL, non-technical tasks by the designated Project Manager PM).
The consortium will use JIRA as the standard incident management tool. JIRA lets you prioritize, assign, track, report and audit the registered issues of any kind — from software bugs and help-desk tickets to project tasks and change requests.

2.2.2. Perform Task

The SD starts working based on these incidents.

At the end of each working day, each SD have the responsibility to save their work daily, choosing at least one different magnetic support (than its local hardware), out of the following ones:

- file server, in a directory allocated by the PM/TL;
- another local hard disk;
- repository on an application server, depending on the project.

Saving sources on the Subversion (SVN) project database is mandatory for configuration management and backup purposes.

2.2.3. Test Task

The objective of testing is to ensure that the requirements are met and the software component delivered conforms to the quality requirements specified for the task. The means of verification are either manual or automated, depending on the methodology specified for the specific project.

Only after quality verifications performed by the programmer, the TL integrates and performs basic tests and verifications of the integrated software components and gives the integrated software component(s) to the testing team for testing. The problems are recorded in the incident tracking system (JIRA).

2.2.4. Deliver Task

At the expected finish date of the task (that is, start date + expected duration) the SD gives the deliverable of the task for integration to the TL. The TL integrates the deliverables and performs basic tests and verifications on them. SVN is used for configuration management.

2.3. Testing

A detailed presentation on the methodology used for the testing activities will be presented in the deliverable D2.4 Testing Plan.

3. TECHNICAL SPECIFICATIONS

3.1. *Definition of methods and terms*

Definitions of certain terms and methods vary depending on context and author. Thus, this section aims to provide clear descriptions of methods and and terms as they will be used within this document[1]:

System requirements are, in opposition to user requirements, a more detailed description of the functionality to be provided. They define exactly what has to be implemented in terms of the system's functions, services, and operational constraints. System requirements can be classified further into functional and non-functional-requirements.

Functional requirements are statements of services the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations.

Non-functional requirements are constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process, and constraints imposed by standards. Non-functional requirements often apply to the system as a whole, rather than individual system features or services.

Security related requirements are the security related architectural requirements. This requirement type is typically derived from architectural principals and good practice standards.

Mockups are a type of low-fidelity prototype and a more elaborate form of wireframes also taking into account first design decision. They can be hand-drawn and paper-based and enable early visualisations of design solutions, to provoke innovation and improvement through testing.

Components descriptions give an overview of each component's main areas and features. A feature is a distinguishing characteristic of a software item (e.g., performance, portability, or functionality).

Events are video streams generated by a live event or a video file.

STAGE COMPONENTS NAMING

The project consortium elaborated on the naming of the STAGE Platform components based on the provisions of DOW, as follows:

1. STAGE Portal, integrating both simple users and cultural institutions management profile, z

[1] Sommerville, Ian (2011), Software Engineering (9th ed.), Boston: Pearson; cf. YOUSE (2013),

2. STAGE Video Platform, representing an extension of the WimTV Platform, specifically developed within the STAGE Project.

3.2. *Functional requirements*

This section presents the outcome of the requirements analysis phase, listing and describing the functional user requirements, derived from and correlated to the supported use cases and user stories defined in D1.3 Definition of technical specifications, in order to meet user requirements.
The section will contain:

- List of functional requirements
- Detailed description of each functional requirement

The following sub-sections delineate the major functional requirements of the STAGE platform. The STAGE Platform will be developed as a web site, accessible thru the http (Hypertext Transfer Protocol) and https (Hypertext Transfer Protocol with a Transport Layer Security) protocols on the World Wide Web.

3.2.1. User flow

In this section are presented the functional requirements related to the users' registration into the STAGE Platform.

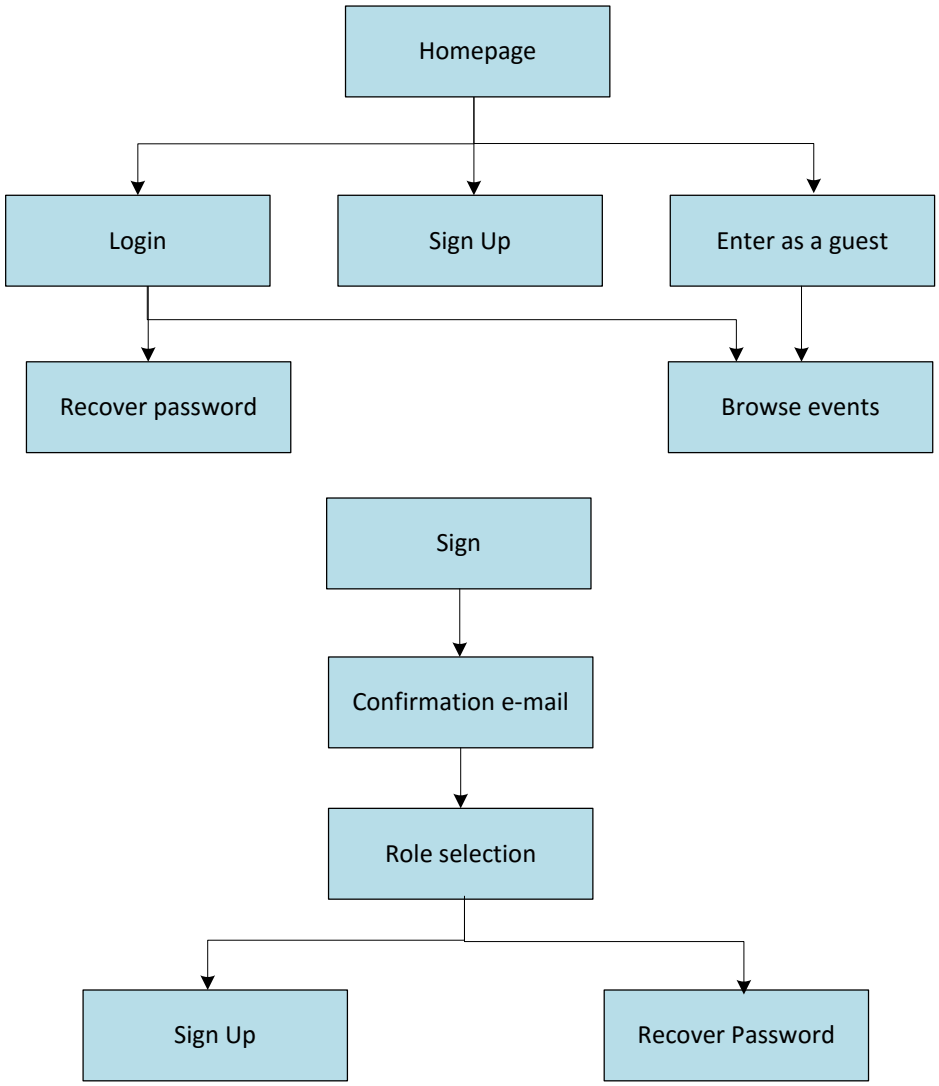


Figure 1 Functional requirements related to user registration

- **HOMEPAGE**

There shall be an initial page displayed by all accesses to the Web Site.This shall be the Home Page of the site. The Home page shall contain the following menu controls:

- Browse events – This control shall display the browse events list, which helps users to filter the events by category (Recent events, Currently live, Upcoming, Scheduled services, Advanced search, Institutions);[2]
- Sign in – This control shall display the LogIn window;
- Sign up – This control shall display the Sign up window where information such as e-mail address, password and a password confirmation will be required to register in the platform;

[2]

- Need help? Click here! - This control shall display the Help page in a new window;
- Logout – This control shall switch out the user from the STAGE Platform.

- **LOGIN**

A user must be able to access the private area of the STAGE Platform only by means of authentication by providing an e-mail address and password. An e-mail and password field will be provided in order to submit such information together with a login button. Once the button is clicked, the user credentials will be verified and the user will be granted access to the system, otherwise a dialogue box will be prompted with a specific error message. This feature is available for both Cultural Institutions and Simple Users. After a successful login, the system shall manage a user's sessions. Upon successful login, the system will redisplay the current page to the user with the new privileges.

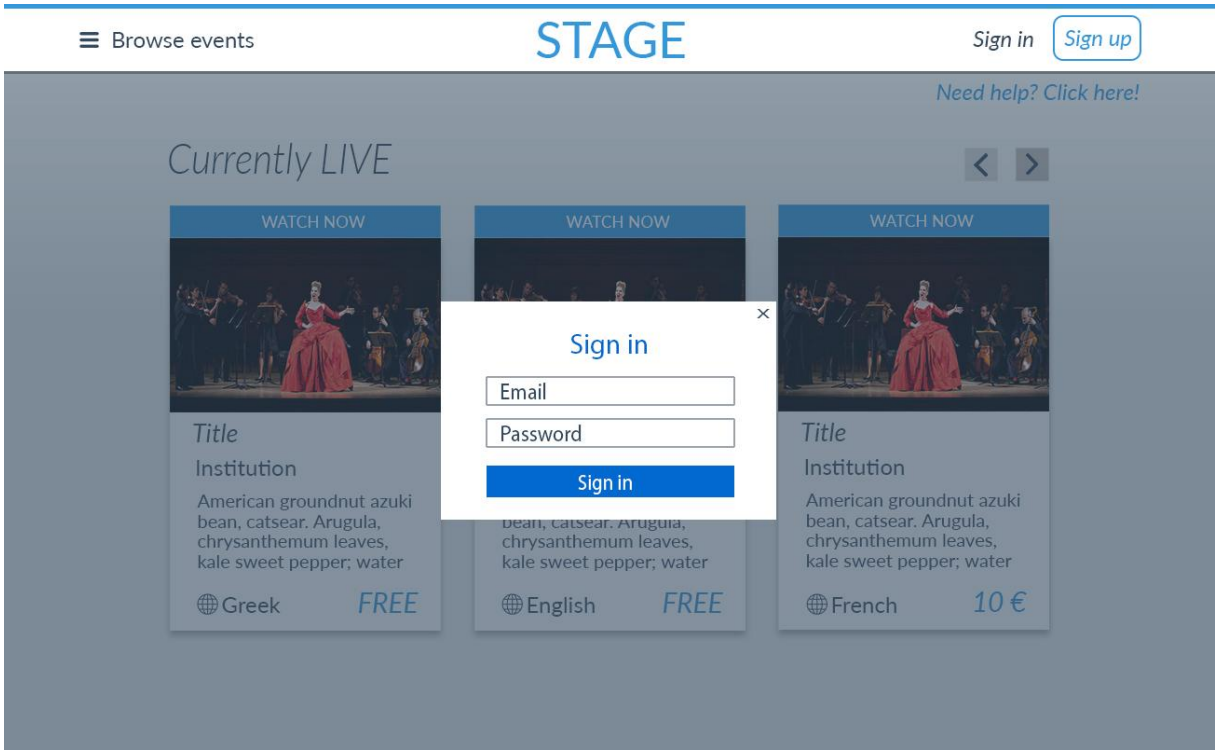


Figure 2 STAGE Platform – Login page

- **RECOVER PASSWORD**

In case the user has forgotten his access password a recovery mechanism must be implemented. In order to facilitate this, a "Forgot my password" label will be implemented in the log in page in order to satisfy the user needs. Thus, once the label is clicked, the user will be redirected to another page where he will be asked to submit his e-mail address in order to receive a link to reset his password. Once the user has submitted the e-mail address, an e-mail with a reset link is sent to him, and once this link is clicked a page with a form to add and confirm his new password will be prompted. Once this last form is submitted, the new

password is saved in the user's profile. For the profiles to be matching between the STAGE portal and Video Platform, a webservice will be called in Video Platform to sync the accounts.

- **REGISTER**

A new user must be able to register into the platform. A register label will be provided in order to facilitate such functionality; by clicking the register label the user will be redirected to another page where information such as e-mail address, password and a password confirmation will be required.
Once the user has submitted this information, a confirmation e-mail will be sent to his e-mail address in order to activate the account.

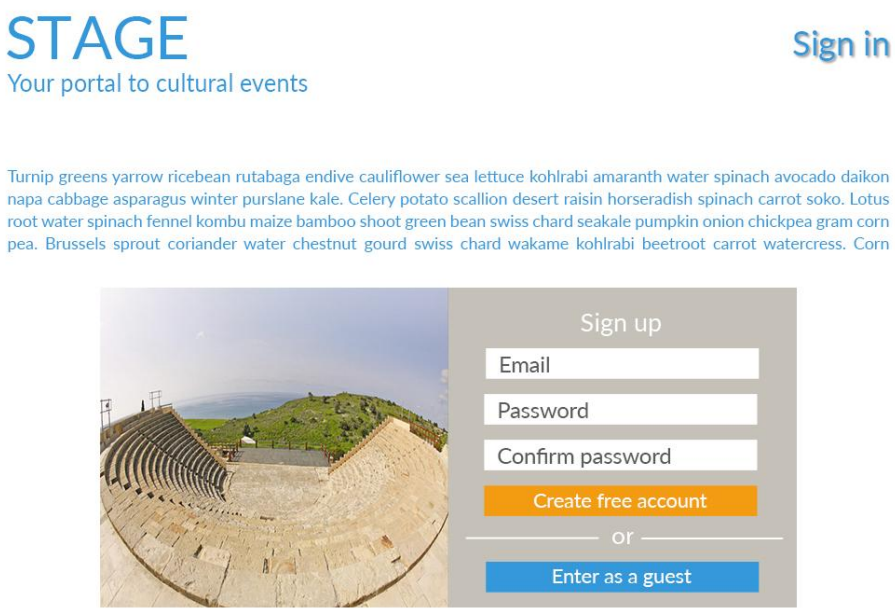


Figure 3 STAGE Platform – Register page

In order to keep the profile information in sync at all times, a webservice will be called in the Video Platform that will also create the user profile there.

- **SELECT USER ROLE**

During the registration workflow, a user must be able to select which kind of role he wants to play in the platform. In order to facilitate this functionality, a profile selection page will be provided during the registration process allowing the user to select between either a Simple User or a Cultural Institution. During the registration process, a link is sent to the user's e-

mail address. Once this link is clicked the following page will be displayed in order to identify the role of the user in the platform.

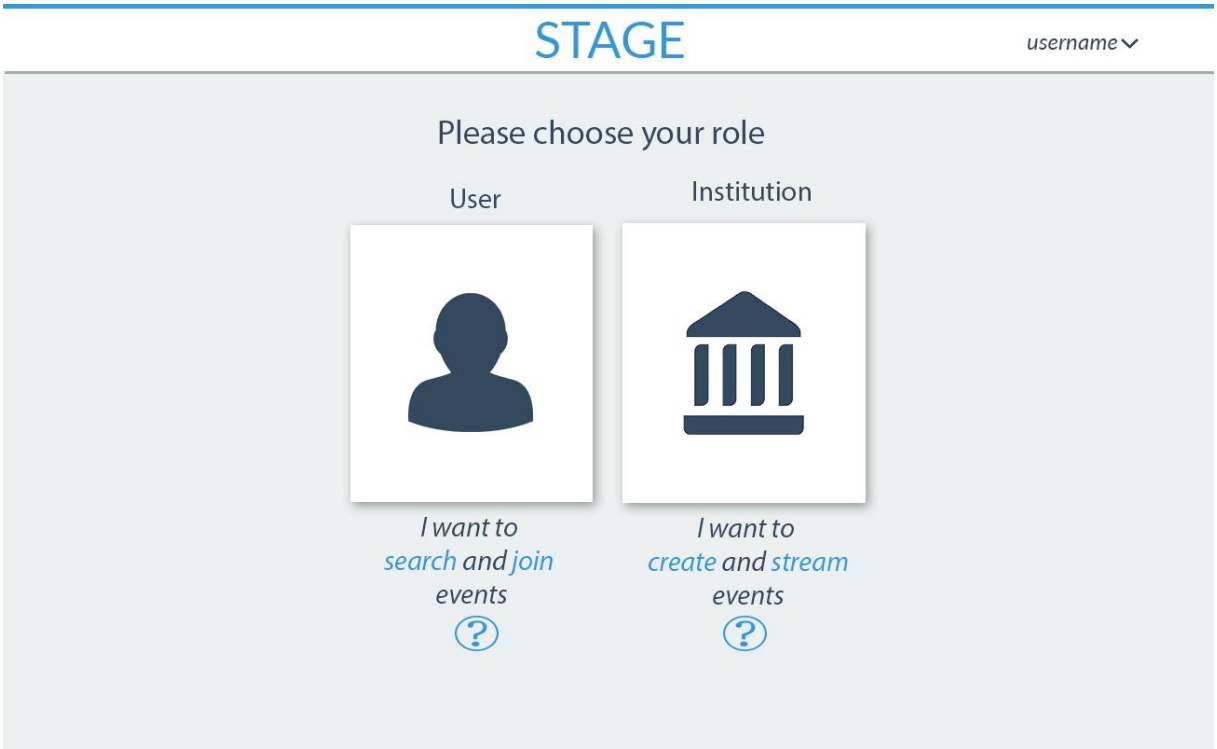


Figure 4 STAGE Platform – select user role

In order to keep the profile information in sync at all times, a webservice will be called in the Video Platform that will also update user profile there.

- **ENTER AS A GUEST**

A user must be able to visit the site as a guest. The role Guest is assumed by any non-authenticated user of the system. If a user does not have a user account with the system, or is not signed in the system, he/she is only given access to publicly available information.
The main functionality available to the Guest user of STAGE Platform should be:

- Registration: allows users to provide their personal information to the system and create a new user account
- Sign in: allows users that have user accounts with the system to provide their username and password and authenticate, in order to use features of the system depending on the user role
- Browse events: allows users to provide search criteria

3.2.2. Browse events

This section presents the functional requirements concerning how the users can browse events. This feature is available for both Cultural Institutions and Simple Users.

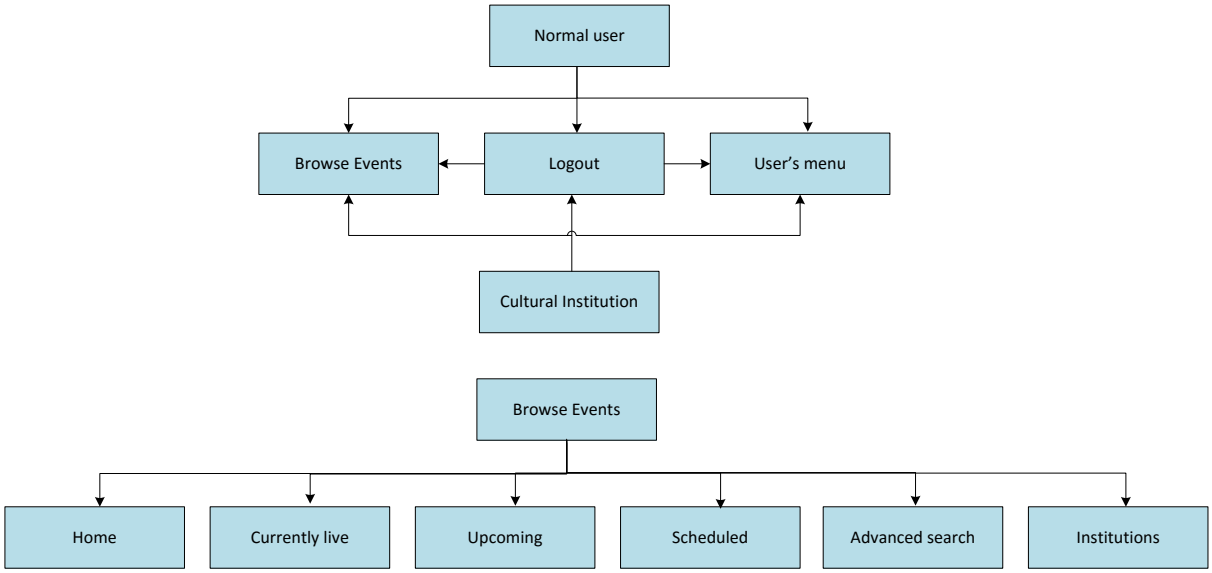


Figure 5 Functional requirements for Browse Events

- **HOME**

A user must be able to browse all the available streams from latest and recent events that are displayed in a page. Each event must contain an image, short description and title. More detailed information and access to the stream will be provided once the selected user event is accessed by clicking on its page display.

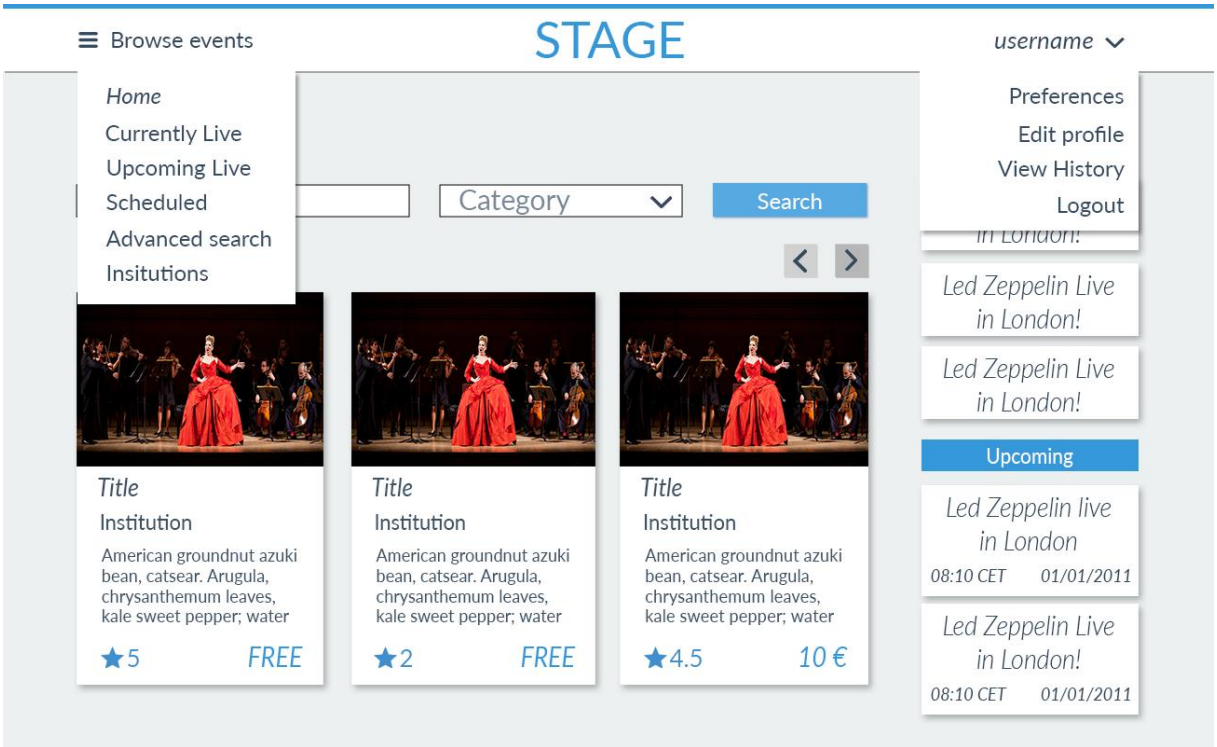


Figure 6 STAGE platform – Browse Events – the Simple user perspective

This information is provided by calling a webservice in the Video Platform that will return a list of events to be displayed here. A user must make separate queries for live and stored events.

- **CURRENTLY LIVE**

A user must be able to filter only the live events while browsing through the event list by selecting the live events category. Once selected, only the live events will be displayed in the page. The user must be able to access and watch the wished event. Some of the live events have a price to be viewed (pay-per-view). The user must be able to click on the event and the platform must redirect to the video player page. The users must be asked to login in case of not being logged in. The user will provide the information requested by the payment system. For the user trials, Paypal test accounts will be used, followed by the real accounts when the platform will be launched.

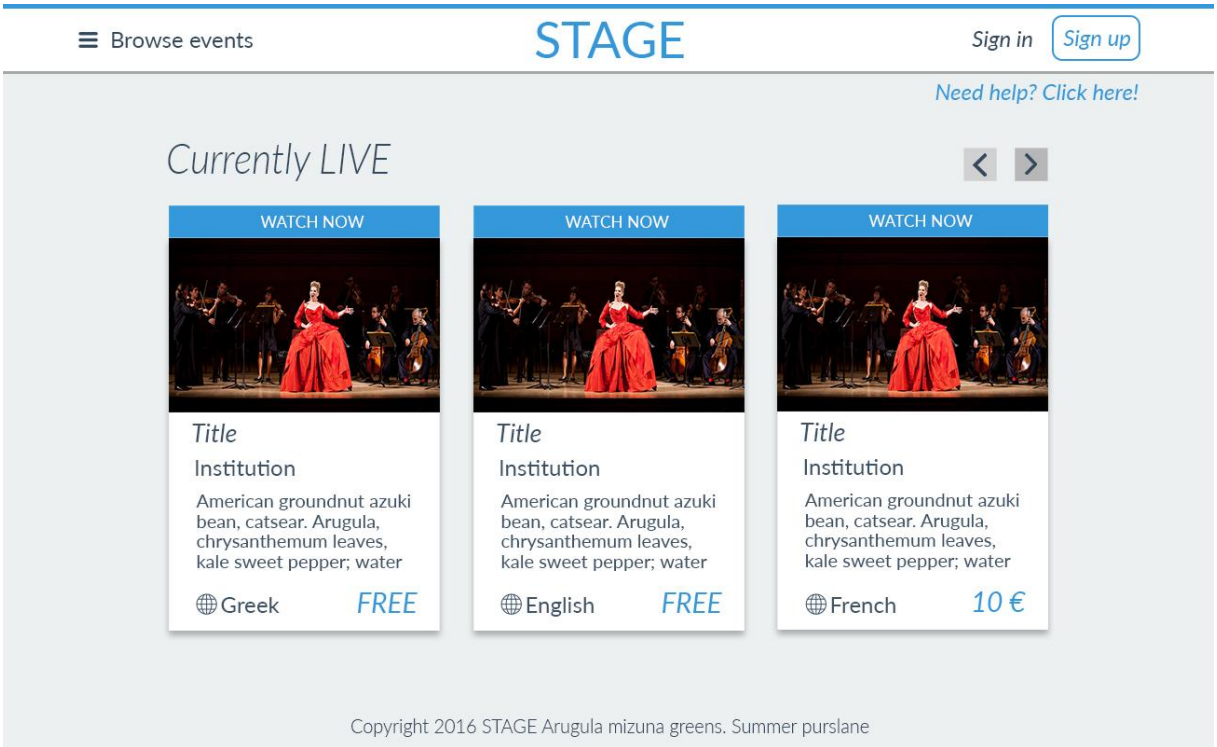


Figure 7 STAGE Platform – Currently LIVE

This information is provided by the Video Platform by calling a special web service.

- **Upcoming events**

A user must be able to filter only the upcoming events while browsing through the list of events by selecting the upcoming events category. Once selected, only the upcoming events will be displayed in the page.

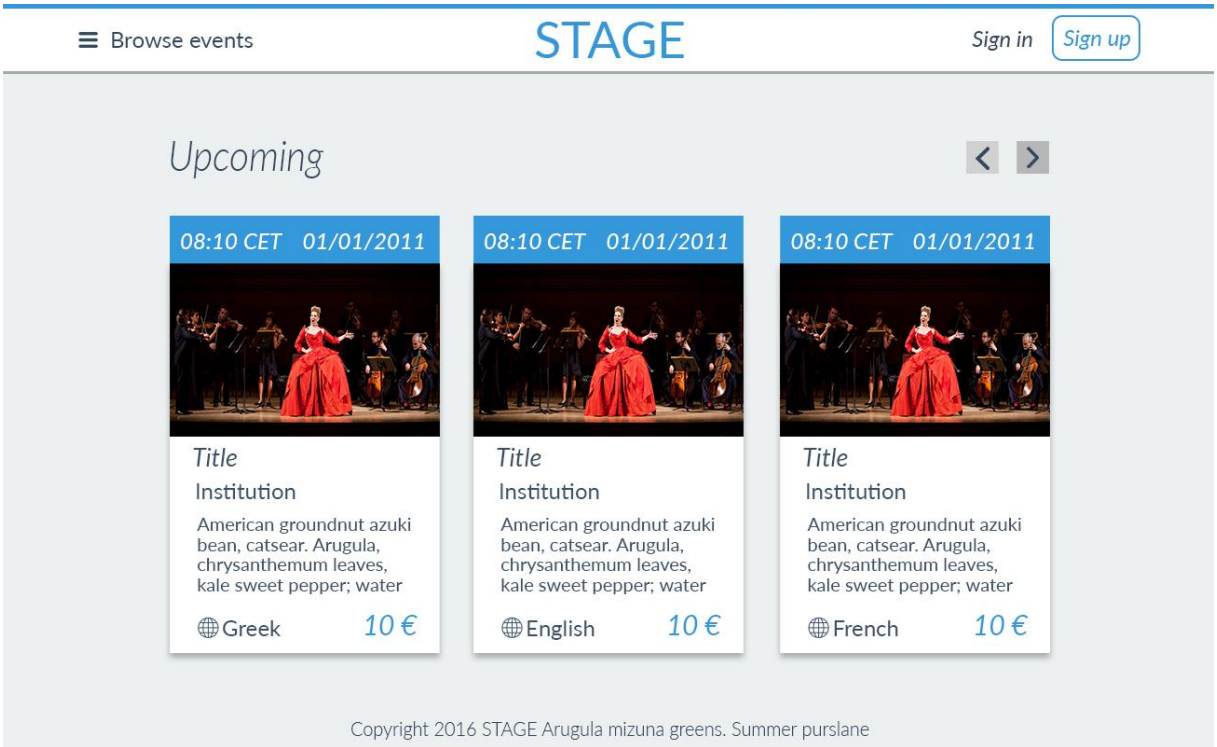


Figure 8 STAGE Platform – Upcoming

This information is provided by the Video Platform by calling a special web service.

- **Scheduled services**

A user must be able to filter only the scheduled events while browsing through the event list by selecting the shedled services category. Once selected, only the scheduled events will be displayed in the page.

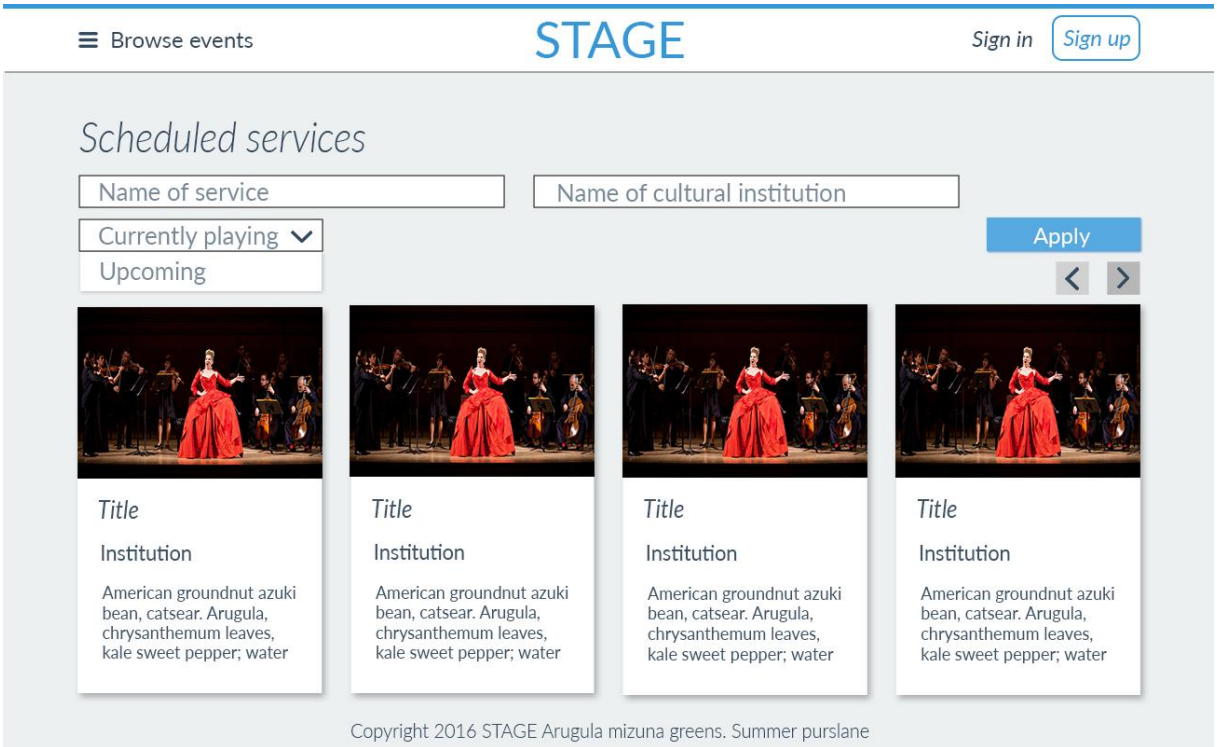


Figure 9 STAGE Platform – Scheduled page

This information is provided by the Video Platform by calling a special web service.

- **Advanced Search**

A user must be able to filter through all the available events on the platform, separately for live and recorded events. In order to achieve this, a filter based search will be available in the events browsing page. The filter based search will allow the user to search for events based on specific selections such as: name of event, name of the institution, category, institutions, and information related to the cost of the event (free or paid), type of event (live or recorded) or the language of the event. Once the filter is applied, all the matching events will be displayed in the browsing list.

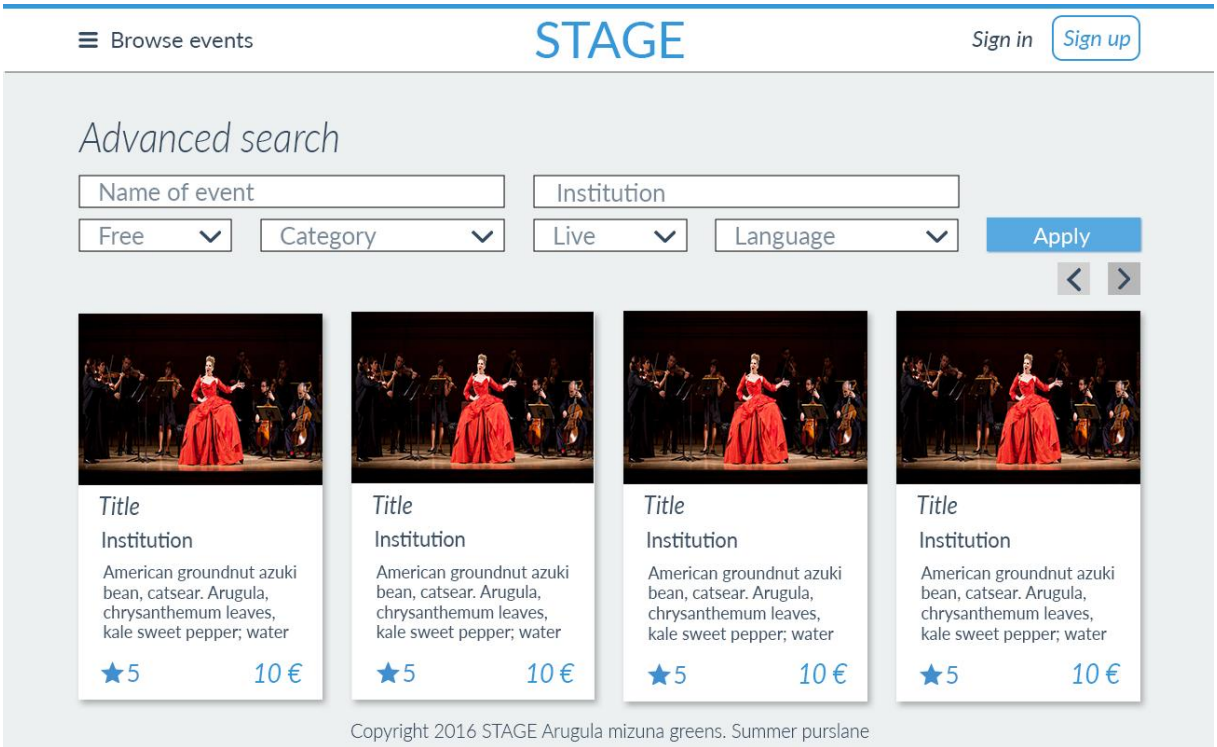


Figure 10 STAGE Platform – Advanced search page

The results of this section are provided by the Video Platform by calling a series of special webservices.

- **Institutions**

A user must be able to select from the menu the option to see all the institutions registered on the platform. By selecting this, the user will be redirected to a page where all the existing institutions are displayed. The user must be able to sort and to filter institutions by name, and country. Each institution thumbnail must contain an image/logo, name of the institution and a short description. More detailed information will be provided once the selected user institution is accessed by clicking on its page display.

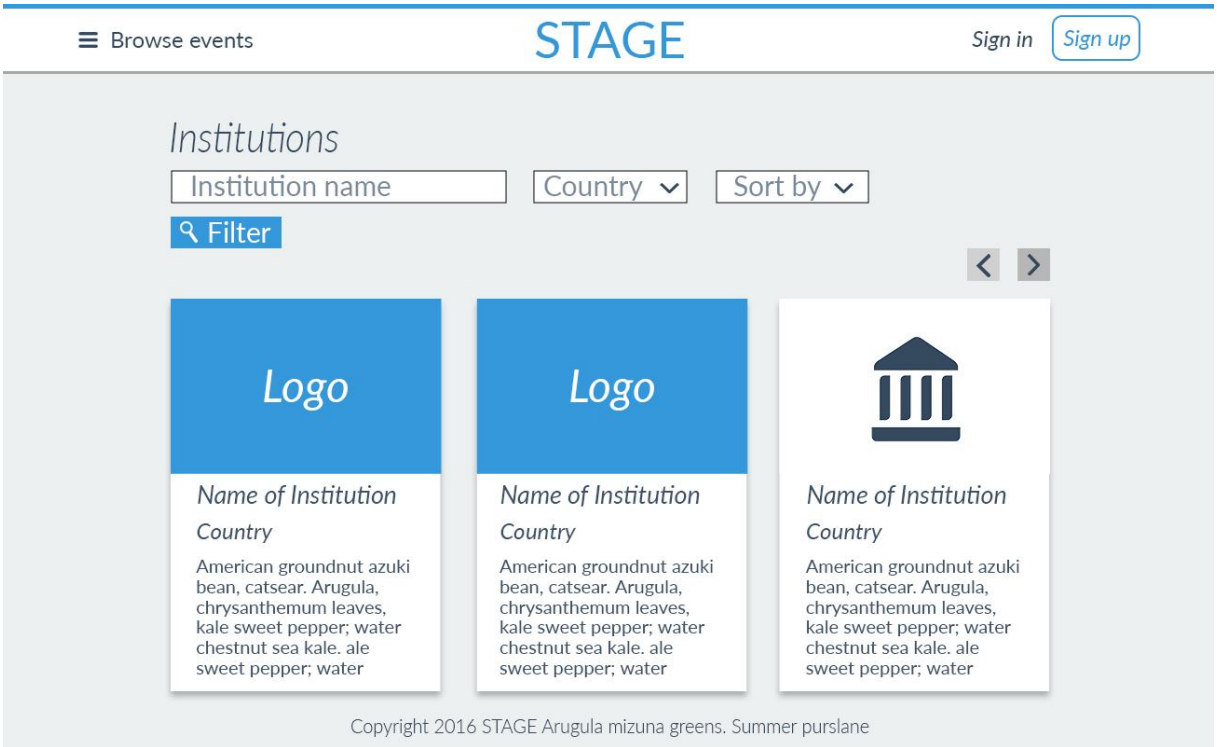


Figure 11 STAGE Platform –Institutions page

- **Institutions Public Page**

A user must be able to access a Cultural Institution's public page by selecting it from the Institutions Page. The Cultural Institution's public page displays the following information: name and descriptio of the cultural institution, a display of the live event currently available into the STAGE Platform and option to join the event, list with recorded events.

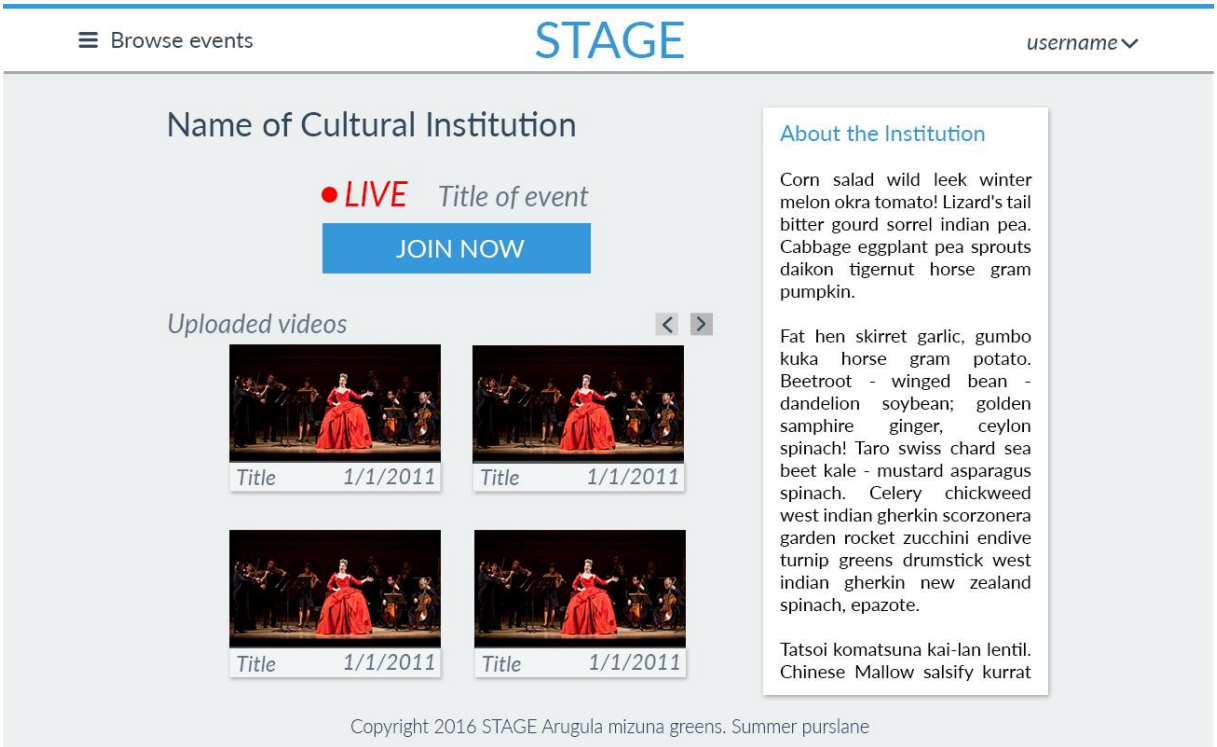


Figure 12 STAGE Platform – Cultural Institution Public page

- **Viewing an online event**

A user must be able to view an online event, by accessing the video player of the event.

In order to facilitate this functionality, the STAGE Portal will be integrated with the Video Platform. The user must be able to access the following options into the video player of the event from the Video Platform: the video player of the event, more detailed information of the event (provided by the cultural institution or from Wikipedia), two sharing options on social media and STAGE platform; a button to pay for the event If it is not free, a button to rate the event, a comment section to discuss the content of the video. In case of live events, the users must be able to access dedicated chatrooms in which participants will have the possibility yo engage in realtime discussions related to the specific event.

- **Logout**

A user must be able to exit his account once he decides to end his browsing session on the platform. In order to facilitate this, a log out button will be provided that will terminate the current user session once it is clicked. When activated the logout control shall display a control asking for confirmation of logout. If confirmation is denied, the logout confirmation is removed with no effect. If confirmed, the Home Page is displayed and all subordinate windows are closed.

3.2.3. User's menu

This section presents functional requirements related to the users's menu.

3.2.3.1. Simple users

For simple users, the section integrates the following options (as shown in the picture below):

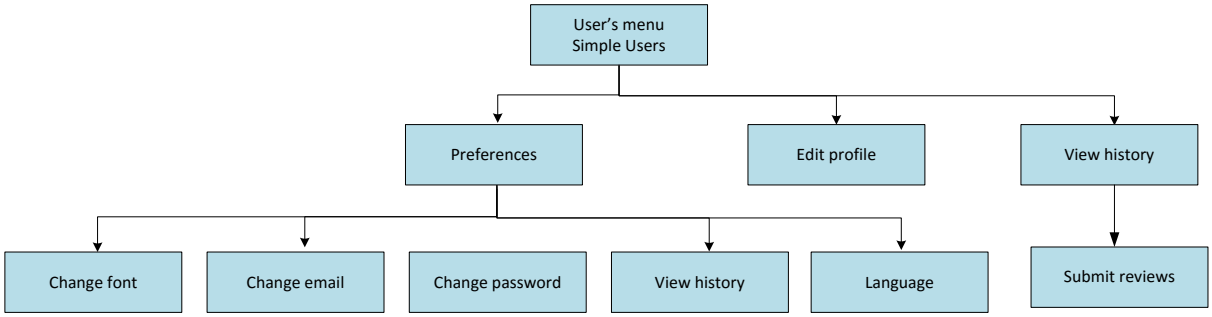


Figure 13 Functional requiremets for the User's menu – the simple user perspective

- **PREFERENCES**

For simple users, the following preferences must be available: change font, change e-mail, change password, view history and language selection.

The users will be able to update the following information:

- Change font – in case the user wants to change the font size, a control section will be implemented to enable stepwise change of the size of text elements. Change will happen stepwise in 3 steps using two buttons, one for next bigger size, and the other for next smaller size.
- Change e-mail – in case the user wants to change his e-mail address, he will access the Change email label to be directed to the email field in which the user will type the new e-mail address
- Change password – in case the user wants to change his password, he will acces the Change password label to be directed to the email field in wich the user will type the new password
- View History – in case the user wants to see the previously viewed events, he will access the View History label and a list will be displayed. Each recorded event will be displayed as a link to be accessed easily by the user.
- Submit reviews – in case the user wants to submit a review for a previously viewed event, he will access the form from the View History label and will be able to assess the quality of the event and by giving votes between 1 and 5 stars for each of the following categories: event quality, video quality, audio quality, connection quality. The 5-stars selection corresponds to very high quality, while 1-star corresponds to very below-average quality. The user will be able to post any additional comments about the event in the corresponding section of the form.
- Language – A user must be able to view the content of the web site in different languages. In order to facilitate this functionality, the web site will be available in

multiple languages. The preferred user language selection can be done from the user menu by selecting the preferred available language. The default language for the platform will be the user's browser's language.

- **UPDATE PROFILE**

A user must be able to update his profile by changing his profile information. This will be possible by accessing his Edit profile option in the Users' menu. The page must display information related to the user's role.

3.2.3.2. Cultural Institutions

For Cultural Institutions, the section integrates the following options (as shown in the picture below):

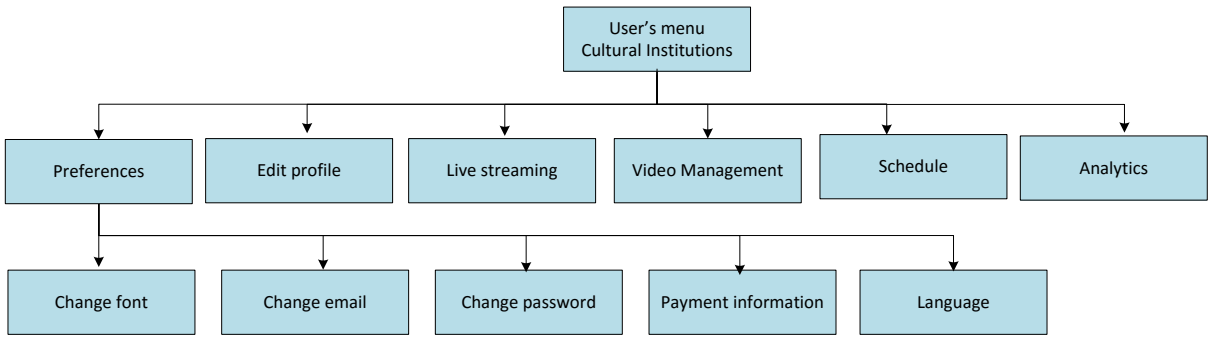


Figure 14 Functional requiremets for User's menu – the Cultural Institution perspective

- **PREFERENCES**

For cultural institutions, the following preferences must be available: change font, change e-mail, change password, payment information and language selection.
The users will be able to update the following information:

- Change font – in case the user wants to change the font size, a control section will be implemented to enable stepwise change of the size of text elements. Change will happen stepwise in 3 steps using two buttons, one for next bigger size, and the other for next smaller size.
- Change e-mail – in case the user wants to change his e-mail address, he will access the Change email label to be directed to the email field in which the user will type the new e-mail address.
- Change password – in case the user wants to change his password, he will acces the Change password label to be directed to the email field in wich the user will type the new password.

  - Payment information – in case the user wants to fill in or update the information needed for the payment transactions.
  - Language – A user must be able to view the content of the web site in different languages. In order to facilitate this functionality, the web site will be available in multiple languages. The preferred user language selection can be done from the user menu by selecting the preferred available language. The default language for the platform will be the user's browser's language.

- **UPDATE PROFILE**
  A user must be able to update his profile by changing his profile information. This will be possible by accessing his Edit profile option in the Users' menu. The page must display information related to the user's role.

- **STREAM NOW** – A cultural institution can access a page where it can create, read, update and delete live events that are not running.

- **MANAGE VIDEOS -** a Cultural Institution can manage creation and maintenance of paid and free recorded videos into the STAGE platform. A Cultural Instituion can publish / unpublish videos.

- **MANAGE SCHEDULED SERVICES -** a Cultural Institution can manage creation and maintenance of scheduled services so that they can play certain recorded videos at desired times.

- **ANALYTICS –** a Cultural Institution can access statistics on how many users and when they watched their videos. This can be triggered from the desire to analyse user activity in order to better adjust the stream hours or content.

In order to keep the profile information in sync at all times, a webservice will be called in the Video Platform that will also update the user profile there.

3.3. *Non-functional requirements*

3.3.1. Quality attributes

- **Accessibility**[3]

[3] https://en.wikipedia.org/wiki/Accessibility

Accessibility refers to the design of products, devices, services, or environments for people who experience disabilities. The concept of accessible design and practice of accessible development ensures both "direct access" (i.e. unassisted) and "indirect access" meaning compatibility with a person's assistive technology (for example, computer screen readers).

- **Documentation**[4]

Documentation is a set of documents provided on paper, or online, or on digital or analog media, such as audio tape or CDs. Examples are user guides, white papers, on-line help, and quick-reference guides. It is becoming less common to see paper (hard-copy) documentation. Documentation is distributed via websites, software products, and other on-line applications.

- **Disaster recovery**[5]

Disaster recovery (DR) involves a set of policies and procedures to enable the recovery or continuation of vital technology infrastructure and systems following a natural or human-induced disaster. Disaster recovery focuses on the IT or technology systems supporting critical business functions, as opposed to business continuity, which involves keeping all essential aspects of a business functioning despite significant disruptive events. Disaster recovery is therefore a subset of business continuity.

- **Efficacy**[6]

Efficacy is the ability to get a job done satisfactorily. The word comes from the same roots as, and for practical purposes is synonymous with, "effectiveness".

- **Extensibility**[7]

In software engineering, extensibility (not to be confused with forward compatibility) is a systems design principle where the implementation takes future growth into consideration. It is a systemic measure of the ability to extend a system and the level of effort required to implement the extension. Extensions can be through the addition of new features or through modification of existing ones. The central theme is to provide for change – typically enhancements – while minimizing impact to existing system functions.

- **Interoperatibility**[8]

[4] https://en.wikipedia.org/wiki/Documentation
[5] https://en.wikipedia.org/wiki/Disaster_recovery
[6] https://en.wikipedia.org/wiki/Efficacy
[7] https://en.wikipedia.org/wiki/Extensibility
[8] https://en.wikipedia.org/wiki/Interoperability

Interoperability is a characteristic of a product or system, whose interfaces are completely understood, to work with other products or systems, present or future, in either implementation or access, without any restrictions.

- **Maintainability**[9]

In engineering, maintainability is the ease with which a product can be maintained in order to:

- Isolate defects or their cause,
- Correct defects or their cause,
- Repair or replace faulty or worn-out components without having to replace still working parts,
- Prevent unexpected breakdowns,
- Maximize a product's useful life,
- Maximize efficiency, reliability, and safety,
- Meet new requirements,
- Make future maintenance easier, or
- Cope with a changed environment.

In some cases, maintainability involves a system of continuous improvement- learning from the past in order to improve the ability to maintain systems, or improve reliability of systems based on maintenance experience.

- **Privacy**[10]

Privacy is the ability of an individual or group to seclude themselves, or information about themselves, and thereby express themselves selectively. The boundaries and content of what is considered private differ among cultures and individuals, but share common themes. When something is private to a *person*, it usually means that something is inherently special or sensitive to them. The domain of privacy partially overlaps security (confidentiality), which can include the concepts of appropriate use, as well as protection of information. Privacy may also take the form of bodily integrity.

- **Portability**[11]

Portability in high-level computer programming is the usability of the same software in different environments. The prerequirement for portability is the generalized

[9] https://en.wikipedia.org/wiki/Maintainability
[10] https://en.wikipedia.org/wiki/Privacy
[11] https://en.wikipedia.org/wiki/Software_portability

abstraction between the application logic and system interfaces. When software with the same functionality is produced for several computing platforms, portability is the key issue for development cost reduction.

- **Scalability**[12]

Scalability is the capability of a system, network, or process to handle a growing amount of work, or its potential to be enlarged in order to accommodate that growth. For example, it can refer to the capability of a system to increase its total output under an increased load when resources (typically hardware) are added. An analogous meaning is implied when the word is used in an economic context, where scalability of a company implies that the underlying business model offers the potential for economic growth within the company.

- **Security**[13]

Security is the degree of resistance to, or protection from, harm. It applies to any vulnerable and valuable asset, such as a person, dwelling, community, item, nation, or organization.

3.3.2. Constraints

Operational constrains of the STAGE Platform consist of three areas:

I. Browser limitations

The STAGE Platform will be based on

1. an open-source Web-solution (eg. Drupal), designed to work on three levels: the database level, the application level, and the interface level, available as an Intranet/Internet solution on different devices (e.g., PC, tablets and SMART TV).
2. The WimTV video management and distribution platform.

The STAGE Platform will be optimized for the following resolutions:
- Small devices (Tablets): resolution contraint is considered a minimum width of 768 px.
- Medium devices (Desktops): resolution contraint is considered a width egual or larger than 992 px.

The STAGE Platform is also compatible with Smart TV, the portal can be viewed using the browsers described bellow.

The STAGE Platform will be optimized to work under the following operating systems and clients (browsers):

[12] https://en.wikipedia.org/wiki/Scalability
[13] https://en.wikipedia.org/wiki/Security

- Windows 8 (and higher):
  - Internet Explorer Edge and higher
  - Google Chrome 54 and higher
  - Mozilla Firefox 49 and higher

- MAC OS X 10.11 "El Capitan" (and higher):
  - Safari 8.0 and higher

- iOS 9 (and higher): default Safari installation
- Android Lollipop v 5.0 (and higher): default Google Chrome installation

For each operating system and browser version, the resolution constrains presented above are valid.

II. Hosting environment limitations

The Drupal component of the STAGE Platform can ensure full functionality and performance on the following SLA (service-level agreement) commitments:

- Low Level Usage:

Low level usage is defined as having a maximum of 40 concurrent users on the platform. The concurrent users are defined as the number of users that can use a certain section of the platform at the same time. For this level of usage the following hosting capabilities are required:

Processor: 6 CORE, 2.4 GHz per core
Memory: 8 GB
Hard disk capacity: 100 GB (not including backup)
BANDWIDTH: at least 10 MBps
Operating system: Unix based (CentOS or Ubuntu) are recommended, Windows systems can also be used.

- Medium Level Usage:

Medium level usage is defined as having a maximum of 100 concurrent users on the platform. The concurrent users are defined as the number of users that can use a certain

section of the platform at the same time. For this level of usage the following hosting cluster will be required:

a. Web Server

Processor: 12 CORE, 2.4 GHz per core
Memory: 32 GB
Hard disk capacity: 500 GB (not including backup)
BANDWIDTH: at least 10 MBps
Operating system: Unix based (CentOS or Ubuntu) are recommended, Windows systems can also be used.

b. Database server

Processor: 6 CORE, 2.4 GHz per core
Memory: 8 GB
Hard disk capacity: 100 GB (not including backup)
BANDWIDTH: at least 10 MBps
Operating system: Unix based (CentOS or Ubuntu) are recommended, Windows systems can also be used.

- High Level Usage:

High level usage is defined as having a maximum of 300 concurrent users on the platform. For this level of usage the following hosting capabilities must be scaled to a server cluster that will include:

a. Load balancer:

Processor: 6 CORE, 2.4 GHz per core
Memory: 8 GB
Hard disk capacity: 100 GB (not including backup)
BANDWIDTH: at least 10 MBps
Operating system: Unix based (CentOS or Ubuntu) are recommended, Windows systems can also be used.

b. Three Web servers:

Processor: 8 CORE, 2.4 GHz per core
Memory: 16 GB
Hard disk capacity: 200 GB (not including backup)
BANDWIDTH: at least 10 MBps
Operating system: Unix based (CentOS or Ubuntu) are recommended, Windows systems can also be used.

c. Two database severs:

Processor: 6 CORE, 2.4 GHz per core
Memory: 8 GB
Hard disk capacity: 100 GB (not including backup)
BANDWIDTH: at least 10 MBps
Operating system: Unix based (CentOS or Ubuntu) are recommended, Windows systems can also be used.

d. Application server constrains:

As this level of usage of the platform is very high, the open-source community is not able to provide good performance software for databases. In this case, we will approach commercial systems as Percona server for MySQL or MySQL Enterprise versions. This can be handled and tested when the platform reaches a commercial version and becomes self-sustainable.

The video component of the STAGE Platform runs on Amazon Web Service on 9 EC2 instances as microservices. A basic installation utilises T2.micro instances type; however, more powerful instance types can be deployed according to the expected user load. As a further flexibility, some instances can also be scaled in number to cope with processing loads in specific functions.

III. Interoperability limitations

The STAGE Platform is based on two sistems, STAGE Portal for Profile management, eLearning-like features and Social Integration on one side, and the Video Platform as a video streaming facilitator. The operational context of both systems is required for full functionality of the STAGE Platform.

3.3.3. Sustainability plan

Since the component Video Platform is based on parts of the video streaming services provided by CEDEO, the support of the platform is inherently guaranteed through the IPR agreement.
Normal maintenance, including security updates and minor improvements, are naturally included in the normal software lifecycle inherited from the WimTV platform.
This applies both on the backends microservices and the Graphical User Interface.

The Stage Portal is a robust platform based on Drupal 7 (entreprise class CMS / open source) with up-to-date support. Maintenance activities involve updating the core features when new releases are available, database maintenance activities, backup of files and server

errors. SIVECO will handle the administration of users and the monitoring of all platform functionalities as well as the hosting environment of the portal.

3.4. *STAGE Platform requirements*

3.4.1. Security related requirements

User level security requiments:

- Users have access to different tools of the portal after the authentication.
- The STAGE Platform will reduce any known risk related to filling in the forms in the portal.
- The forms from the STAGE Platform will use a CAPTCHA or similar type of mechanism to avoid spam or automatic records.

The STAGE Platform will include preventions against the following threats related to web-sites:

- Injection
- Cross-Site Scripting (XSS)
- Broken Authentication and Session Management
- Insecure Direct Object References
- Cross-Site Request Forgery (CSRF)
- Security Misconfiguration
- Insecure Cryptographic Storage
- Failure to Restrict URL Access
- Insufficient Transport Layer Protection
- Invalidated Redirects and Forwards
- Search box cross scripting: search forms will not allow queries containing a form of programming language (jss, xml, sql, sparql, etc.)
- The STAGE Platform has to ensure confidentiality of the user's personal data.
- The STAGE Platform has to ensure data integrity.

3.4.2. Technological approach

The STAGE Platform is composed by two main applications written in different programming languages and unified through SOA web services. Web services allow machine-to-machine comunication independent of the programming language or hosting environments.

3.4.2.1. Drupal 7[14]

The main frontend platfrom is based on Drupal 7 CMF (Content Management Framework). Drupal 7 is an enterprise level CMS (content management system) and CMF with high availability, scalability and security features.

Effective Web design is driven by the need to balance flexibility and simplicity. If a system is too simple, it can only be used for a single purpose - but if it is too flexible, it may be too difficult for new users to learn.

Drupal strives to reconcile these conflicting goals by providing its users with the tools they need to make their own content management solution, while still providing some prebuilt components to help them get started. Thus, it can be described both as a content management system (CMS) and a content management framework (CMF) - one system which strives to have the strengths of both, without their deficiencies.

Drupal's operative way[15]

To make the contrast between Drupal and other CMS's more concrete, consider the example of a news site. You want to be able to post news articles on the site, and you want the homepage to have a section featuring the five most recent ones. Next, you decide that you want to add a blog section, and put a list of links to the five most recent of blog entries on the homepage as well.

If you were using an ordinary CMS, first you would install a plugin that handled news articles and could put short blurbs on the homepage. Next, you'd install a plugin that would track the latest blog posts and put a list of those on the homepage. Each plugin would only be responsible for tracking and managing a particular kind of content, and would remain relatively isolated from the others.

But, what happens if you want to blend these two functions by showing a list of blog posts *about* the latest news items, ordered by most active contributor first? If you're using a "toy truck" CMS, you may be out of luck, or need to hire a developer to write you a custom plugin from scratch. But through the power of the Drupal way, the way of manageable abstraction, you can whip out a kit full of parts and knock this together pretty quickly. Since Drupal's modules do things in a standard way, and interface with a common underlying system, building all sorts of clever, customized features is just a matter of snapping parts together.

[14] http://drupalhelp.research.nmsu.edu/content/why-drupal
[15] https://www.drupal.org/docs/7/understanding-drupal/overview

Of course, this flexibility comes at a certain cost. While a toy truck is instantly understandable and ready to use without much thought, a modular vehicle construction kit will by nature require you to read the instruction manual first. The building blocks are out there, but you'll need to learn how they fit together before you can take a paper prototype and turn it into a fully-featured website.

Drupal core, and the thousands of contributed modules that build on it, require an initial investment to learn, but mastering the Drupal way is immensely rewarding. Drupal's passionate community is a testament to its power to liberate site builders from the simplicity/flexibility dilemma.

Drupal's architectural structure[16]

Drupal treats most content types as variations on the same concept: a *node*. Static pages, blog posts, and news items (some possible node types) are all stored in the same way, and the site's navigation structure is designed separately by editing menus, views (lists of content), and blocks (side content which often have links to different site sections).
It's very similar to the separation found in standards-compliant page coding—XHTML provides the meaningful structure of the information, while CSS arranges it for presentation. In Drupal, nodes hold the structured information pertaining to a blog post (such as title, content, author, date) or a news item (title, content, go-live date, take-down date), while the menu system, as well as taxonomy (tagging of content) and views, create the information architecture. Finally, the theme system, along with display modules like Panels, controls how all this looks to site visitors.

Since these layers are kept separate, it is possible to provide a completely different navigation and presentation of content to different users based on their specific needs and roles. Pages can be grouped differently, prioritized in a different order. Additionally, various functions and content can, also, be shown or hidden as needed.

[16] https://www.drupal.org/docs/7/understanding-drupal/overview

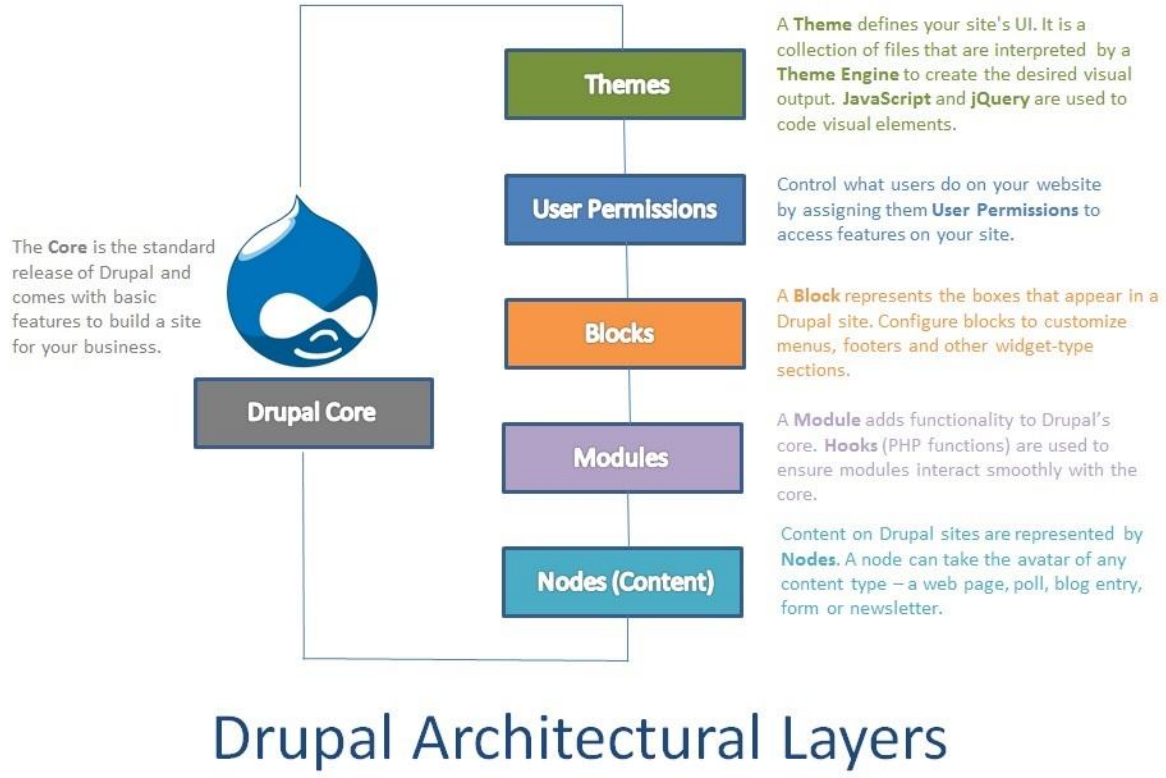


Figure 15 Drupal Architectural Layers

Nodes: The enablers of Drupal's flexibility and scalability

Nodes are at the heart of Drupal's design, a more detailed description of how they work is provided in the following. At its most basic, a node is a set of related information. When you create a new blog post, you are not only defining its body text, but also its title, content, author link, creation date, taxonomy (tags), etc. Some of these elements will be shown by the theme layer when the node is displayed. Others are meta-data that control when the node will show up at all - such as taxonomy or publishing status.

Since each item of content is stored as a node, and contains the same basic information, each can be handled in a standard way by both Drupal core and contributed modules. This allows site builders to choose exactly where they want content to show up, and exactly how they want it to look in each case. Most of a Drupal site builder's time is spent defining what kinds of information you want to store in your nodes, and configuring the structures (menus, taxonomy trees, views, panels) in which to display them.

As suggested before, you aren't limited to a single way of presenting your site's content. You can define as many navigation schemes, custom themes ("skins" for the site), blocks (small bits of content, such as the five most recent blog articles described earlier), and feature sets as there are distinct audiences for your site.

Comments are secondary in Drupal compared to nodes, but they also illustrate the Drupal way. Comments aren't just part of the blog system, since there isn't a separate "blog system." Comments can be enabled on any node type you choose - blog posts, news items, book pages (which provide basic wiki features) and any other you may create.

Collaborative at the core[17]

Creating an informational website that broadcasts from "one to many" is something that most CMSs do right out of the box. However, Drupal allows to empower site users to create content, and connect with each other - moving from "one to many" to "many to many."

With some CMS's, you can set up a blog, and you can install plugins to handle having a community of users, but what happens when you want to give individual blogs to each of your users, sorting their contents so that they can be displayed individually with their own skins, while also generating cross-blog topical digests, top five lists, and links out to elaborate, customized user profiles? What if you want to also integrate that in with forums, a wiki-like environment, and give each user their own gallery of taggable photos?

Drupal is designed from the ground up so site builders can delegate content creation, and even site administration, to users. All you have to do is define who gets to do what on your site (through user permissions), and then you can start collaborating.

The Drupal flow[18]

If you want to go deeper with Drupal, you should understand how information flows between the system's layers. There are five main layers to consider:

[17] https://www.drupal.org/docs/7/understanding-drupal/overview

[18] https://www.drupal.org/docs/7/understanding-drupal/overview

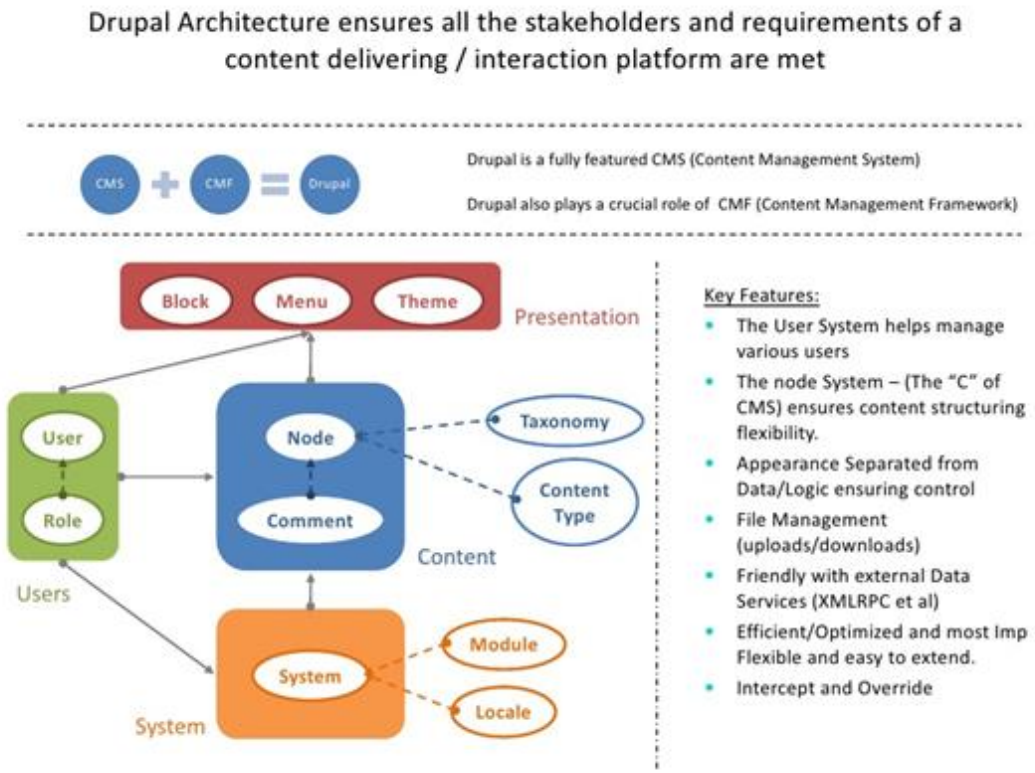


Figure 16 The Drupal flow[19]

1. At the base of the system is the collection of nodes—the data pool. Before anything can be displayed on the site, it must be input as data.

2. The next layer up is where modules live. Modules are functional plugins that are either part of the Drupal core (they ship with Drupal) or they are contributed items that have been created by members of the Drupal community. Modules build on Drupal's core functionality, allowing you to customize the data items (fields) on your node types; set up e-commerce; programmatically sorting and display of content (custom output controlled by filters you define); and more.

[19] https://www.slideshare.net/Samasing2/091120-drupal-service-ppt-compatibility-mode

3. At the next layer, we find blocks and menus. Blocks often provide the output from a module or can be created to display whatever you want, and then can be placed in various spots in your template (theme) layout. Blocks can be configured to output in various ways, as well as only showing on certain defined pages, or only for certain defined users.

4. Next are user permissions. This is where settings are configured to determine what different kinds of users are allowed to do and see. Permissions are defined for various roles, and in turn, users are assigned to these roles in order to grant them the defined permissions.

5. On the top layer is the site theme (the "skin"). This is made up predominantly of XHTML and CSS, with some PHP variables intermixed, so Drupal-generated content can go in the appropriate spots. Also included with each theme is a set of functions that can be used to override standard functions in the modules in order to provide complete control over how the modules generate their markup at output time. Templates can also be assigned on-the-fly based on user permissions.

This directional flow from bottom to top controls how Drupal works.
As mentioned earlier—getting the kind of granular control one wants over the details of the XHTML module outputs requires understanding this flow.

3.4.2.2. Video Platform (based on WimTV)

The WimTV high level architecture is depicted in Figure 17 . The Video Platform used for Stage project mantain the same architecture

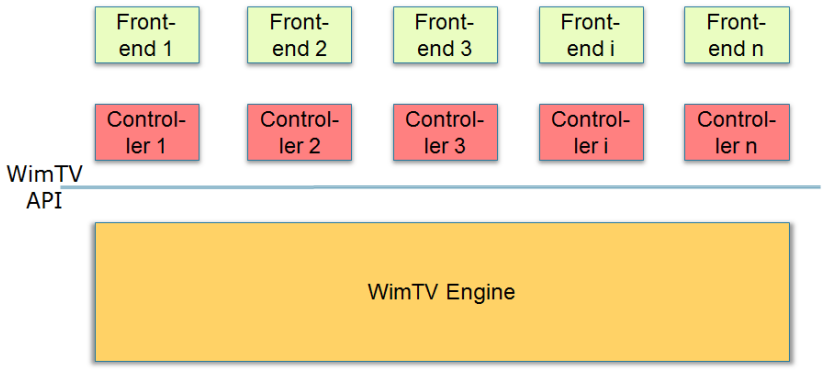


Figure 17 High level WimTV architecture

Frontends represent the individual web pages. There will be 3 types of web page:

1. STAGE customised versions of the current WimTV pages;
2. newly designed pages;

3. pages recreated as Drupal pages.

Controllers allow Frontends to access WimTV via Application Programming Interfaces (API). The Pages of type 1 above will reuse parts of existing controllers but will need some customization, while pages of type 2 will require totally new controllers.

The WimTV Engine is the core of the STAGE Platform. It has the following features:
1. distributed microservice architecture;
2. accessible via API;
3. OAuth2 authentication;
4. on the cloud (AWS);
5. 9 AWS instances for different microservices;

Regarding the distributed microservice architecture (point 1 above), WimTV has adopted the docker container architecture. Docker containers wrap up a piece of software in a complete filesystem that contains everything it needs to run[20]. Adopting a docker container architecture allows an easy redistribution of workload as the number and load of WimTV Engine users increases.
The following containers are used:
1. API layer;
2. Ingestion and transcoding;
3. Data base;
4. Scheduled services;
5. User customisation;
6. Frontend layer;
7. Streaming;
8. Payment;
9. Statistics.

3.4.2.3. Interoperability

The interoperability will be ensured by high traffic rest web services allowing message exchange between the two systems in JSON formats. The interoperability of the two systems will be detailed in deliverable D2.2 Integration Plan.

3.4.3. Development environment and tools

[20] "What is Docker?". *docker.com*.

The STAGE Platform is being developed based on the latest methodologies and tools, for this approach we are using a three level server architecture, and different tools to optimize and automate different tasks.

Three Level Server Architecture

The development physical arhitecture is composed of three main areas: Development & Integration, Staging and Production, each with specific roles in the process of building the software product.

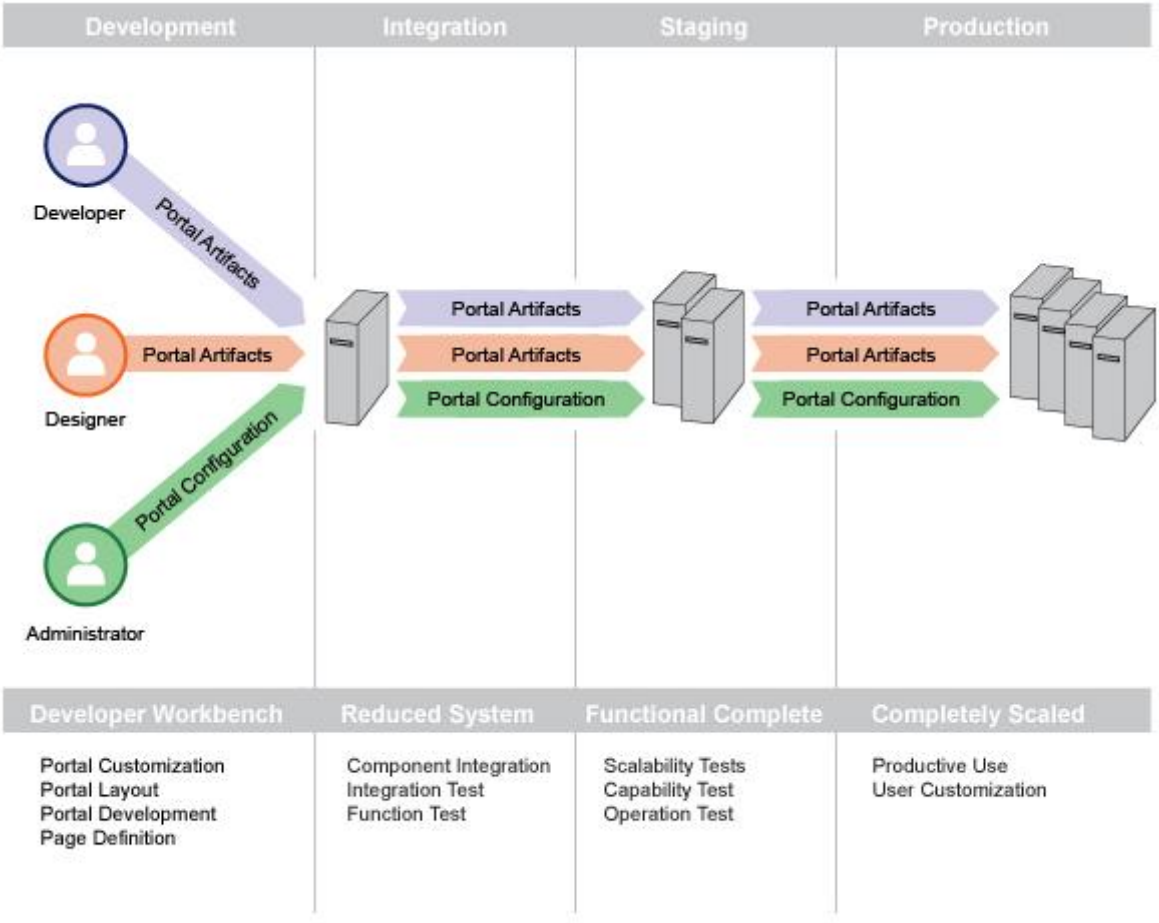


Figure 18 Three Level Server Architecture

Development & Integration has the role of providing a unified system for coding, customization, layout building and database architecture building. The integration is also made on this environment in order to ensure component compatibility and functional dependencies.

Staging environment is an independent part of the architecture with the scope of completing the functional testing, scalability testing, capability cheks and operational workflows cheks on a standalone location that is not being affected by the current developments.

The production environment has to meet the limitation specified in Chaper 4.2.2 of this document. This is the system that will make the platform available to public users and will host only those modules/resources that are ready to use.

Front End Development

The front end part of the platform is the area that users will see and with which they will interact, it is also called the graphical user interface (GUI) of the platform and it is very important that it be clear, easy to use and optimized, as described in Chapter 4.2.2 of this document. We have approached responsive web design and implementation as current SOA standards at the time of development. This approach will be beneficial in terms of compatibility and scalability, making the STAGE Platform optimized and usable on different devices and web browsers. The user interface is based on Boostrap front-end web framework as being the most supported and feature compliant at the time of development. Other technologies like jQuery, jQuery UI and Modernizr will also be used to complete and extend the framework.

Bootstrap[21] is a responsive front-end web framework for designing websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Bootstrap offers a large OS (operating system) and browse compatibility pool as described in the table below:

<table>
<tr><td></td><td>Chrome</td><td>Firefox</td><td>Internet Explorer</td><td>Opera</td><td>Safari</td></tr>
<tr><td>Android</td><td>Supported</td><td>Supported</td><td rowspan="3">N/A</td><td>Not Supported</td><td>N/A</td></tr>
<tr><td>iOS</td><td>Supported</td><td>N/A</td><td>Not Supported</td><td>Supported</td></tr>
<tr><td>Mac OS X</td><td>Supported</td><td>Supported</td><td>Supported</td><td>Supported</td></tr>
<tr><td>Windows</td><td>Supported</td><td>Supported</td><td>Supported</td><td>Supported</td><td>Not Supported</td></tr>
</table>

[21] https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)

The current stable version is Bootstrap 3.0 which we aim to implement both in the boiler template proposed in the first stage of the project and in the Drupal 7 Theme. By using Bootstrap web framework we can ensure both modern browsers compatibility (eg. IE Edge +, Firefox, Safari and Chrome), and high flexibility in design component changes, and the responsive design approach.

jQuery[22] is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. jQuery is the most popular JavaScript library in use today, with installation on 65% of the top 10 million highest-trafficked sites on the Web.

jQuery UI[23] is a collection of GUI (graphical user interface) widgets, animated visual effects, and themes implemented with jQuery, Cascading Style Sheets, and HTML. According to JavaScript analytics service, Libscore, jQuery UI is used on over 197,000 of the top one million websites, making it the second most popular JavaScript library.

Modernizr[24] is a JavaScript library which is designed to detect HTML5 and CSS3 features in various browsers, which lets JavaScript avoid using unimplemented features or use a workaround such as a shim to emulate them. Modernizr aims to provide this feature detection in a complete and standardized manner.

CSS 3[25] **(Cascading Style Sheets)** is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

HTML5[26] is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and current version of the HTML standard. It was published in October 2014 by the World Wide Web Consortium (W3C) to improve the language with support for the latest multimedia, while keeping it both easily readable by humans and consistently understood by computers and devices such as web browsers, parsers, etc.

[22] https://en.wikipedia.org/wiki/JQuery
[23] http://www.wikiwand.com/en/JQuery_UI
[24] https://en.wikipedia.org/wiki/Modernizr
[25] https://en.wikipedia.org/wiki/Cascading_Style_Sheets
[26] https://en.wikipedia.org/wiki/HTML5

HTML5 is intended to subsume not only HTML 4, but also XHTML 1 and DOM Level 2 HTML. HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a candidate for cross-platform mobile applications, because it includes features designed with low-powered devices in mind.

The technologies presented here will ensure an optimal format for content to be available through commonly used web browsers, e.g. IE, Chrome, Firefox, in commonly used document formats, e.g. doc, xls, ppt, pdf, and in compatibility with commonly used operating systems including mobile technology, e.g. Windows, MAC, OS, iOS, Android, and Windows mobile as described in Chapter 4.2.2.

Back End Development

PHP[27]: PHP is a server-side scripting language designed primarily for web development but also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Development Team. PHP originally stood for Personal Home Page, but it now stands for the recursive acronym PHP: Hypertext Preprocessor.

PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications.

LAMP[28]: LAMP is an archetypal model of web service solution stacks, named as an acronym of the names of its original four open-source components: the Linux operating system, the Apache HTTP Server, the MySQL relational database management system (RDBMS), and the PHP programming language. The LAMP components are largely interchangeable and not limited to the original selection. As a solution stack, LAMP is suitable for building dynamic web sites and web applications.

[27] https://en.wikipedia.org/wiki/PHP
[28] https://en.wikipedia.org/wiki/LAMP_(software_bundle)

Drush[29]**:** Drush is a command line shell and Unix scripting interface for Drupal. Drush core ships with lots of useful commands for interacting with code like modules/themes/profiles. Similarly, it runs update.php, executes SQL queries and DB migrations, and misc utilities like run cron or clear cache.

Git/SVN[30]**:** Git is the most widely used modern version control system in the world today. Git is a mature, actively maintained open source project originally developed in 2005 by Linus Torvalds, the famous creator of the Linux operating system kernel. A staggering number of software projects rely on Git for version control, including commercial projects as well as open source. Developers who have worked with Git are well represented in the pool of available software development talent and it works well on a wide range of operating systems and IDEs (Integrated Development Environments).
Apache Subversion (often abbreviated SVN, after the command name svn) is a software versioning and revision control system distributed as free software under the Apache License. Software developers use Subversion to maintain current and historical versions of files such as source code, web pages, and documentation.

Jenkins[31]**:** Jenkins provides continuous integration services for software development. It is a server-based system running in a servlet container such as Apache Tomcat. It supports SCM tools including AccuRev, CVS, Subversion, Git, Mercurial, Perforce, Clearcase and RTC, and can execute Apache Ant and Apache Maven based projects as well as arbitrary shell scripts and Windows batch commands. The primary developer of Jenkins is Kohsuke Kawaguchi. Released under the MIT License, Jenkins is free software.

IntellijIDEA[32]**:** is a Java integrated development environment (IDE) for developing computer software. It is developed by JetBrains (formerly known as IntelliJ), and is available as an Apache 2 Licensed community edition, and in a proprietary commercial edition. Both can be used for commercial development.

MySQL Workbench/phpMyAdmin[33]**:** MySQL Workbench is a visual database design tool that integrates SQL development, administration, database design, creation and maintenance into a single integrated development environment for the MySQL database system. It is the

[29] http://www.drush.org/en/master/
[30] https://www.atlassian.com/git/tutorials/what-is-git
[31] https://en.wikipedia.org/wiki/Jenkins_(software)
[32] https://en.wikipedia.org/wiki/IntelliJ_IDEA
[33] https://en.wikipedia.org/wiki/MySQL_Workbench

successor to DBDesigner 4 from fabFORCE.net, and replaces the previous package of software, MySQL GUI Tools Bundle.
phpMyAdmin is a free and open source tool written in PHP intended to handle the administration of MySQL or MariaDB with the use of a web browser. It can perform various tasks such as creating, modifying or deleting databases, tables, fields or rows; executing SQL statements; or managing users and permissions.

PuTTY[34]: PuTTY is a free and open-source terminal emulator, serial console and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. It can also connect to a serial port. The name "PuTTY" has no definitive meaning.
PuTTY was originally written for Microsoft Windows, but it has been ported to various other operating systems. Official ports are available for some Unix-like platforms, with work-in-progress ports to Classic Mac OS and Mac OS X, and unofficial ports have been contributed to platforms such as Symbian,[4][5] Windows Mobile and Windows Phone.

WinSCP[35]: WinSCP (Windows Secure Copy) is a free and open-source SFTP, FTP, WebDAV and SCP client for Microsoft Windows. Its main function is secure file transfer between a local and a remote computer. Beyond this, WinSCP offers basic file manager and file synchronization functionality. For secure transfers, it uses Secure Shell (SSH) and supports the SCP protocol in addition to SFTP.

Vagrant[36]: Vagrant is computer software that creates and configures virtual development environments. It can be seen as a higher-level wrapper around virtualization software such as VirtualBox, VMware, KVM and Linux Containers (LXC), and around configuration management software such as Ansible, Chef, Salt, and Puppet.
Vagrant was originally tied to VirtualBox, but version 1.1 added support for other virtualization software such as VMware and KVM, and for server environments like Amazon EC2. Vagrant is written in Ruby, but can be used in projects written in other programming languages such as PHP, Python, Java, C# and JavaScript.
Since version 1.6, Vagrant natively supports Docker containers, which in some cases can serve as a substitute for a fully virtualized operating system.
Vagrant plugins also exist, including vagrant-libvirt that adds support for libvirt, vagrant-lxc that adds support for lxc, and vagrant-vsphere that adds support for VMware's ESXi.

[34] https://en.wikipedia.org/wiki/PuTTY
[35] https://en.wikipedia.org/wiki/WinSCP
[36] https://en.wikipedia.org/wiki/Vagrant_(software)

Postman[37]**:** Postman is a REST Client that runs as an application inside the Chrome browser. It is very useful for interfacing with REST APIs.**Application programming interface**[38]**:** Application Programming Interface (API) is a set of subroutine definitions, protocols, and tools for building application software. In general terms, it's a set of clearly defined methods of communication between various software components. An API ensures the development of a computer program by providing all the building blocks, which are then integrated by the programmer. An API may be developed for a web-based system, operating system, database system, computer hardware, or software library.

4. SYSTEM ARCHITECTURE

The integrated modules in the STAGE Platform are presented in the picture below.

[37] http://windowsitpro.com/azure/q-what-postman-and-how-do-i-use-it-azure
[38] https://en.wikipedia.org/wiki/Application_programming_interface

Figure 19 Integrated modules in the STAGE PlatformPlatform

4.1. *STAGE Portal*

STAGE Portal integrates the following modules:

- ✓ Simple User Profile Management
- ✓ Content Provider Profile Management
- ✓ Analytics
- ✓ Social and eLearning-like features
- ✓ View History
- ✓ Update payment information
- ✓ Localization Management
- ✓ Content Management System

4.1.1. Simple user profile management

User profile management is available to administrators to manage user information (data generated by the end users of the Service) and at the same time displays for the end users of the Service their own information (their own profile).This module is created to centralize and encapsulate data related to data generated by the end users of the Service. User profile management stores a user profile, overwriting the existing one with the same user ID (if any).

The system registers the account automatically when a user registers and allows users to log in and log out. The process of sign on associates content that they create with their accounts, allowing various permissions to be set for their roles.

4.1.2. Content provider profile management

Content provider profile management is available to administrators to managing data generated by the Cultural Institutions (CI) and at the same time displays the content provided for the end users of the Service. The Content Provider Profile management integrates the WIM TV plugin and Drupal functionalities.

The Content provider profile management module consists in:

- Information about the institution, provided during registration. These are used for communication between system administrators and the organisation representatives.
- Image of the event
- Description of the event
- Event language
- Event price
- Event date

4.1.3. Analytics

The analytics module will be provided by Google Analytics plugin for Drupal. The module allows adding statistics features like:

- Single/multi/cross domain tracking
- Selectively track/exclude certain users, roles and pages
- Monitor what type of links are tracked (downloads, outgoing and mailto)
- Monitor what files are downloaded from pages
- Custom dimensions and metrics support with tokens
- Custom code snippets
- Site Search support
- AdSense support
- Demographics and Interests support (formerly known as DoubleClick remarketing support)
- Anonymize visitors IP address
- DoNotTrack support (non-cached content only)
- Drupal messages tracking
- Modal dialog tracking (Colorbox)
- Access denied (403) and Page not found (404) tracking
- Cache the Google Analytics code on your local server for improved page loading times
- Enhanced Link Attribution support
- User ID tracking across devices
- Changing URL fragments can be tracked as pageviews
- Debug mode with analytics_debug.js

4.1.4. Social feature

Through this feature, the Simple users will access a rating system to submit a review for a previously viewed event and will be able to assess the quality of the event and by giving votes between 1 and 5 stars for each of the following categories: event quality, video quality, audio quality, connection quality.

A comments section will also be available in the video player of the event. In case of live events, the users will be able to access dedicated chat-rooms in which participants will have the possibility to engage in realtime discussions.

4.1.5. eLearning – like feature

The platform has a help section that comes to the aid of the users, providing additional information if it is needed. This help module will be contextual.
Through this module the users will access both guidance material to use STAGE and informative content about cultural events. The Contextual Help keeps help content in nodes, relating to specific paths for which the help is relevant. It provides a block that lists contextual help nodes filtered to those relevant to the page the block is being displayed on.

4.1.6. View History

The platform has a View History section where simple users can access the list with the previously viewed events.

4.1.7. Update payment information

The platform has a Payment information section where cultural institutions can introduce and update information needed for the payment transactions.

4.1.8. Localisation management

Internationalization is one of Drupal's most complex module suites and it covers many different multilingual needs. The STAGE Platform is displayed in 4 languages (Greek, Hungarian, Italian and English) and offers to users the possibility to switch the language.
Drupal actually supports two different approaches to translating content: content translation and field translation.

- Content translation - one node per translation
- Entity translation - one node with many translation

4.1.9. Content Management System

The Drupal Content Management System makes it easy to create, manage, and publish digital content to any device, at any time. Drupal's flexibility handles countless content types including video, text, blog, podcasts, and polls with robust user management, menu handling, real-time statistics and optional revision control.

4.2. *STAGE Video Platform*

STAGE Video Platform component handles the ingestion, management and trading of all the content available on the entire platform, and the streaming of events (both live events and videos on demand) and scheduled services.

4.2.1. API layer

Forwards all requests coming from frontends and other services and provides appropriate responses

4.2.2. Ingestion and transcoding

This container allows a Cultural Institution to upload a video to its repository. Once uploaded, the video is transcoded to the internal format. This container also handles the case of a live event that the CI wishes to record for later use.

4.2.3. Data base

This container handles all user data, content and events of the platform

4.2.4. Scheduled services

Instructs the streamer that a content or an event with a given ID should be streamed at a certain time.

4.2.5. User customisation

This container complements the functions of the Data Base container. It contains user data that have been added after the Data base design. Some of these data will be eventually moved to the Data Base.

4.2.6. Frontend layer

Creates a user interface by composing page from different services in response to user's request.

4.2.7. Streaming

This container handles all requests of live and on demand streaming of the Video platform.

4.2.8. Payment

This container handles the transactions occurring between and among users of the STAGE platform. Depending on the type of transaction, this container can handle a transaction between a seller and a buyer deducting a commission..

4.2.9. Statistics

This container computes statistical information on:

1. Consumption of resources (storage and bandwidth)
2. Views of individual videos and events.

4.2.10. Asset Trading

This container allows Content Providers to post some of their contents for which they have rights to a marketplace where other Content Providers can get the right to incorporate it into their offers.

4.3. *Logical architecture*

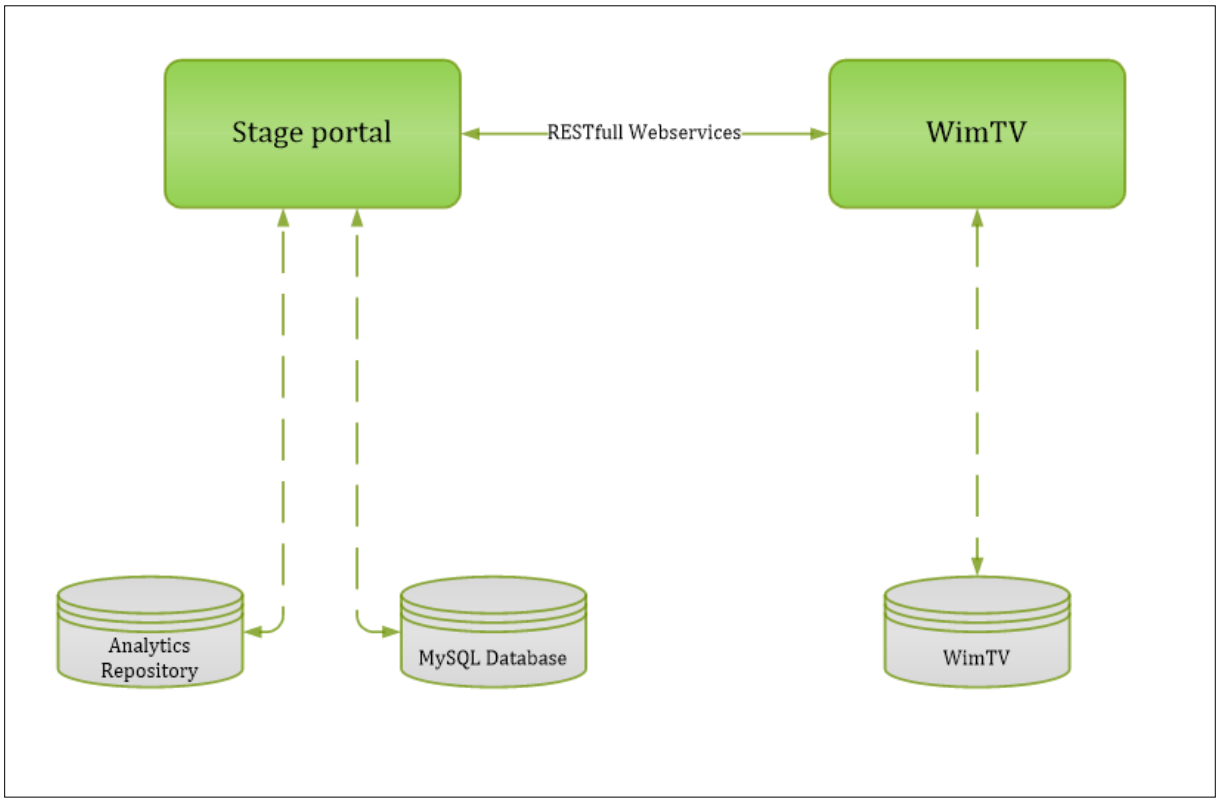


Figure 20 Logical architecture

WimTV offers a 3rd party API to be used as interoperability based on REST webservices.

REST is a style of architecture defined by a set of principles that describe how data is defined and addressed. RESTful (or sometimes referred as REST-style) architecture consists of clients and servers. Clients are the ones that initiate data requests to the servers which process these requests and return a response. Fundamentally in REST each resource is first identified using a URL and a new resource for every service required is created. The data returned by the service must be linked to the other data, hence making it into a network of information unlike the Object-Oriented design which encourages the encapsulation of information. REST architectural style consists on the principles of:

a. Uniform Interface

Data is individually identified and manipulated using URLS and verbs. The information has multiple forms of representation (XML, JSON, HTML) when sent to the client. Clients can manipulate the information through these representations.

b. Stateless Interactions

All data needed to make a request is contained in the URL, query parameters, body or header.

c. Cacheable

Clients can cache server responses; the responses must contain the information if it can be cached or not in order to avoid sending inappropriate data in further requests.

d. Client-Server

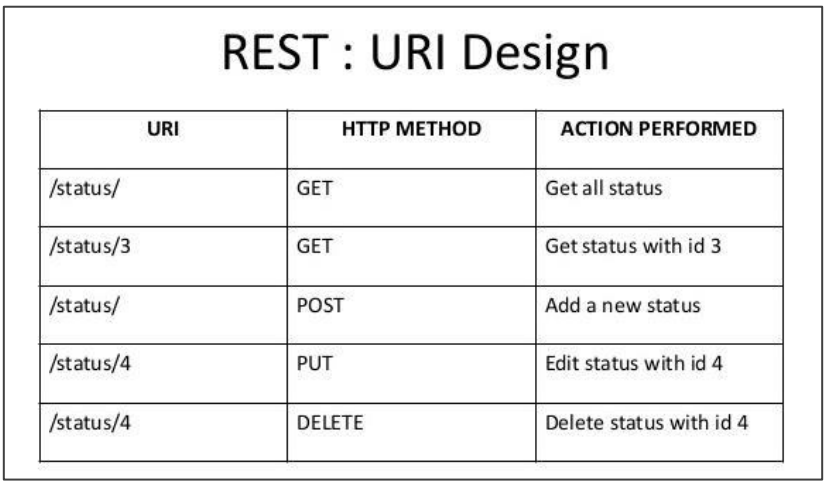
Servers and clients are separated from each other, this meaning that the client does not have direct access to the data storage and cannot interrogate information in this way.

e. Layered System

Clients do not need and don't know if they are connected directly to the web service server or to an intermediate one. This gives the capability of multiple layers of systems to act in the same way.

f. Code on Demand

An option that extends a basic service, giving a temporary possibility for the server to execute code sent by the client. This consists in a security risk and is hardly used.
REST is based on URLS (or also named Uri's) and verbs (GET, POST, PUT and DELETE) which will be used to perform corresponding requests.



REST : URI Design

URI	HTTP METHOD	ACTION PERFORMED
/status/	GET	Get all status
/status/3	GET	Get status with id 3
/status/	POST	Add a new status
/status/4	PUT	Edit status with id 4
/status/4	DELETE	Delete status with id 4

The STAGE Platform will be iterconnected with the Video Platform using these open webservices.

5. USER INTERFACE REQUIREMENTS

This section provides all the requirements that the user interface needs to fulfill. The following table lists all these requirements that will make the STAGE Platform easy and clear to use. The most important reason why every item in the list must be implemented, is the focus group of the platform – the older people. Most of the elderly are not experienced with technology - and to an extent the Internet – which makes the development of the user interface a specific challenge.

No.	Description
1	The platform must be easy to use with a simple interface handling.
2	The platform shall provide self-explanatory interface.
3	The interface must be intuitive.
4	The interface shall provide minimal screen menus with few choices.
5	Correct screen display:
5.1	– The system shall provide clear and understandable symbols.
5.2	– Symbols used shall convey the same meaning across cultural borders.
5.3	– The system shall provide big and understandable fonts.
5.4	– The system shall provide big and understandable buttons.
5.5	– All buttons should be large enough that they are easy to tap or click.
5.6	– The user interface shall present all action buttons as a combination of icons and text.

No.	Description
5.7	− High contrast and colour combination - Set the vision of the screen in high contrast or colour combinations to meet the requirements of visually impaired people.
6	The interface must provide comprehensible messages.
7	All confirmation, information, account and help pages should have the same look-and-feel throughout the system.
8	Documentation and help:
8.1	− The user interface shall have a help button for user support.
8.1.a	− Every single page contains a help popup specific to the information displayed on it. Every user interaction with the platform from this page, is described in detail.
8.2	− Error Messages: When an error occurs, the system should provide suggestions to assist the user with the issue.

6. REFERENCES

- ✓ STAGE project, D1.3 "Definition of technical specifications to meet user requirements", October 2016.
- ✓ https://en.wikipedia.org/wiki/Accessibility
- ✓ https://en.wikipedia.org/wiki/Documentation
- ✓ https://en.wikipedia.org/wiki/Disaster_recovery
- ✓ https://en.wikipedia.org/wiki/Efficacy
- ✓ https://en.wikipedia.org/wiki/Extensibility
- ✓ https://en.wikipedia.org/wiki/Interoperability
- ✓ https://en.wikipedia.org/wiki/Maintainability
- ✓ https://en.wikipedia.org/wiki/Privacy
- ✓ https://en.wikipedia.org/wiki/Software_portability
- ✓ https://en.wikipedia.org/wiki/Scalability
- ✓ https://en.wikipedia.org/wiki/Security
- ✓ http://drupalhelp.research.nmsu.edu/content/why-drupal
- ✓ https://www.drupal.org/docs/7/understanding-drupal/overview
- ✓ http://redcrackle.com/resources/what-is-drupal
- ✓ https://www.slideshare.net/Samasing2/091120-drupal-service-ppt-compatibility-mode
- ✓ https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)
- ✓ https://en.wikipedia.org/wiki/JQuery
- ✓ http://www.wikiwand.com/en/JQuery_UI
- ✓ https://en.wikipedia.org/wiki/Modernizr
- ✓ https://en.wikipedia.org/wiki/Cascading_Style_Sheets
- ✓ https://en.wikipedia.org/wiki/HTML5
- ✓ https://en.wikipedia.org/wiki/PHP
- ✓ https://en.wikipedia.org/wiki/LAMP_(software_bundle)
- ✓ http://www.drush.org/en/master/
- ✓ https://www.atlassian.com/git/tutorials/what-is-git
- ✓ https://en.wikipedia.org/wiki/Jenkins_(software)
- ✓ https://en.wikipedia.org/wiki/IntelliJ_IDEA
- ✓ https://en.wikipedia.org/wiki/MySQL_Workbench
- ✓ https://en.wikipedia.org/wiki/PuTTY
- ✓ https://en.wikipedia.org/wiki/WinSCP
- ✓ https://en.wikipedia.org/wiki/Vagrant_(software)
- ✓ http://windowsitpro.com/azure/q-what-postman-and-how-do-i-use-it-azure
- ✓ https://en.wikipedia.org/wiki/Application_programming_interface