

## Deliverable 2.4

### Integrated Platform (initial)

Dissemination		
Level	Type	Delivery Month
<input type="checkbox"/> Confidential (CO) <input type="checkbox"/> Restricted (RE) <input checked="" type="checkbox"/> Public (PU)	<input type="checkbox"/> Report (R) <input checked="" type="checkbox"/> Prototype (P) <input type="checkbox"/> Other (O)	14

<b>Deliverable</b>	D2.4
<b>Milestone</b>	<i>Not applicable</i>
<b>Work Package Leader</b>	FORTH
<b>Task/Deliverable Leader</b>	FORTH
<b>Deliverable Due Date</b>	31/12/2014
<b>Date of Submission</b>	
<b>Version</b>	1.4
<b>Keywords</b>	
<b>Internal Report Review</b>	Done by management body

<b>Version Control</b>			
<b>Version</b>	<b>Date</b>	<b>Author (Name, Institution)</b>	<b>Comments</b>
<b>1.0</b>	09.12.2014	<i>Stelios Sfakianakis – FORTH-ICS</i> <i>George Christodoulakis – FORTH-ICS</i>	
<b>1.1</b>	19.12.2014	<i>Stelios Sfakianakis – FORTH-ICS</i> <i>George Christodoulakis – FORTH-ICS</i>	
<b>1.2</b>	22.12.2014	<i>Stelios Sfakianakis – FORTH-ICS</i> <i>George Notas – FORTH-ICS</i> <i>Nikolaos Kampanis – FORTH-IACM</i>	
<b>1.3</b>	23.12.2014	<i>Konstantinos Marias – FORTH-ICS</i> <i>George Christodoulakis – FORTH-ICS</i> <i>Alexandros Roniotis – FORTH-ICS</i> <i>Konstantinos Spanakis – FORTH-ICS</i>	
<b>1.4</b>	30.12.2014	<i>Antoine Serrurier – UKA-IMI</i>	<i>Format</i>
<b>2.0</b>			
<b>3.0</b>			

1.X = 1<sup>st</sup> version circulating between the members / 2.X = 2<sup>nd</sup> version following comments of members / 3.X = 3<sup>rd</sup> final version



## Table of Contents

1	LIST OF ACRONYMS .....	6
2	INTRODUCTION .....	7
3	REQUIREMENTS .....	8
3.1	Motivation .....	8
3.2	Use case Scenarios.....	9
3.2.1	Authentication.....	9
3.2.2	Course-ware Loading of User Profile/Performance Metrics.....	9
3.2.3	Course-ware Loading of Training Scenarios/Material.....	9
3.2.4	Course-ware Uploading of User Performance Metrics .....	9
3.2.5	Local and Global Databases Synchronization .....	10
4	LOGICAL ARCHITECTURE.....	10
5	ARCHITECTURE DESIGN.....	11
5.1	Introduction .....	11
5.2	Role Based Access Control (RBAC).....	12
5.2.1	Local Service .....	13
5.2.2	Global Service .....	14
5.3	Components .....	14
5.3.1	Relational Database .....	14
5.3.2	Web Application.....	15
6	IMPLEMENTATION .....	16
6.1	Technological Solutions Used.....	16
6.2	Database Schema .....	17
6.3	Functionality Testing.....	19
7	USER INTERFACE .....	19
7.1	Local Service .....	21
7.1.1	“Home” Menus.....	21
7.1.2	Functionalities.....	24
7.2	Global Service .....	31
7.2.1	“Home” Menus.....	31




---

	7.2.2 Functionalities.....	32
8	APPLICATION PROGRAMMING INTERFACES (APIs).....	33
8.1	Get All Metrics .....	33
8.2	User's Credentials Authentication.....	33
8.3	Real-Time Metrics Insertion.....	34
8.4	Batch Metrics Insertion .....	34
8.5	Training Time.....	35
8.6	User Level .....	35
8.7	User Metrics.....	35
8.8	Local and Global Services Synchronization.....	36
9	CONCLUSIONS .....	36

---

## List of Figures




Figure 1 – Simulator training environment system overview.....	8
Figure 2 – RASimAs Architecture Design of Local and Global Systems Communication .....	11
Figure 3 – Integrated Platform Architecture .....	12
Figure 4 - General Web Application Representation .....	16
Figure 5 - FreeMarker Template Engine.....	17
Figure 6 – Database Schema .....	18
Figure 7 - The "sign-in" page .....	20
Figure 8 - The welcome page (Local Admin) .....	20
Figure 9 - Local Admin “Home” menu.....	22
Figure 10 – Mentor “Home” menu .....	23
Figure 11 – Trainee “Home” menu .....	23
Figure 12 – Mentor Registration .....	24
Figure 13 – Mentor Profile Information .....	25
Figure 14 – Trainee Registration .....	26
Figure 15 – Trainee Information .....	27
Figure 16 – Downloading of Training Material .....	28
Figure 17 – Training History .....	29
Figure 18 – Profile .....	30
Figure 19 – Global Admin “Home” Menu .....	32
Figure 20 – System Files Update Functionality .....	33

## List of Tables

Table 1 – Local System Role Based Access Control.....	13
Table 2 – Local System Profile Functionality Access Rights .....	14
Table 3 - Global System Role Based Access Control.....	14

## 1 List of Acronyms

API	Application Programming Interface
DBMS	Database Management System
ES	Electrical Stimulation
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IP Address	Internet Protocol address
JSON	JavaScript Object Notation
RA	Regional Anaesthesia
RAAs	Regional Anaesthesia Assistant
RASim	Regional Anaesthesia Simulator
RASimAs	Regional Anaesthesia Simulator and Assistant
RBAC	Role Based Access Control
RDBMS	Relational Database Management System
SQL	Structured Query Language
URL	Uniform Resource Locator
US	Ultrasound
UUID	Universally Unique Identifier

	 FP7-ICT-2013-10 - 5.2 - Virtual Physiological Human	
Project No. 610425	<b>Deliverable Report</b> <b>D2.4, 31/12/2014, Revision: Final Version</b>	Page 7 of 37

## 2 Introduction

Regional Anaesthesia (RA) is an increasingly utilized anaesthesia technique that if properly performed can have several beneficial effects on patient clinical outcomes. It has also been suggested that expansion of RA application in Europe could lead to important cost reduction that could reach 100.000€/year and operating theatre. One major obstacle in the expansion of RA application is the lack of physician training programs leading to moderate success rates in this subtle technique. The aim of the RASimAs project is to develop both a Regional Anaesthesia Simulator (RASim) and a Regional Anaesthesia Assistant (RAAs) that will ultimately help in the expansion of RA utilization by providing (i) suitable and quality verified training tools for both novice level and intermediate level anaesthesiologists and (ii) high tech information technology supported systems that will enable increased success rates in RA.

The project is characterized by significant complexity. Deliverable 2.1 (D2.1) presented the *User Requirements* in a number of user stories and need assessment tables for both the RASim and the RAAs. Deliverable 2.2 (D2.2) presented the *Reference Architecture Plan*, which translated those high level descriptions to user cases and component interaction diagrams that describe how the envisaged functionality is implemented. Deliverable 2.3 (D2.3) presented the *Information Storage Service*, a convenient and high capacity system, that in addition to the data storage it also permits the sharing of information among the RASimAs partners and accommodates interoperability with the imaging and modelling tools of RASimAs, in order to support, as a final goal, the integration of the various components of the system. Deliverable 5.1 (D5.1) presented the *RASimAs Simulator Specifications*, which is an extension to the document *Reference Architecture Plan* defined in D2.2. The purpose of that report was to layout the specifications of the simulator and modules used to develop RASim. In addition, this document further defined integration, interfaces, and communication layers between the modules in the RASimAs project. According to these specifications the simulator will need to communicate with a local server for authenticating users, downloading training material, and uploading/downloading users' performance-metrics data, and also with a remote server for downloading system's updates.

Consequently the aim of the present document is the description of the development of two web services, a local and a global one, which will assist in the realization of the RASimAs system by exchanging vital information for its "seamless" integration.

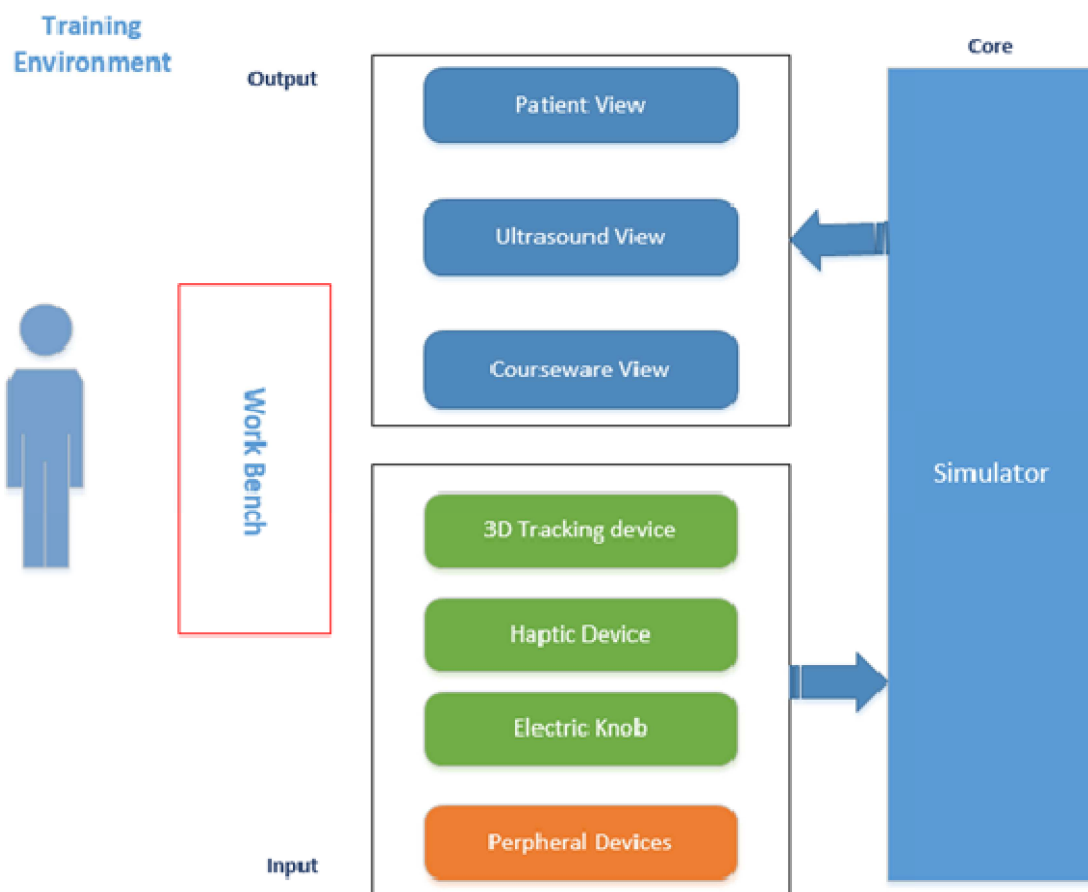
In particular, the current report provides the system's logical and design architecture, the definition of its interfaces, and the selection of the relevant data storage and communication standards based on the managed information. The analysis starts with the system requirements, and following this, the local/global service architecture design is expanded along with their core components. Finally, the implementation of

the overall service is presented along with a detailed description of the various modules that comprise it.

### 3 Requirements




#### 3.1 Motivation

According to the Deliverables 2.1, 2.2, and 5.1 users must have access from anywhere to profile data, performance metrics and training material. In addition, the training systems must be able to work offline and upload the relevant data from a local database. Deliverable 5.1 (*RASimAs Simulator Specifications*) provides additional implementation details for the simulator environment. Based on these specifications, the system consists of three components, namely the input interface,



**Figure 1 – Simulator training environment system overview**

the output interface, and the core simulation software. The user interacts with it through a work bench client computer. Figure 1 shows the RASimAs training environment with the aforementioned components and their sub-components where applicable. As indicated, the input interface primarily connects the input devices, for

	 FP7-ICT-2013-10 - 5.2 - Virtual Physiological Human	
Project No. 610425	<b>Deliverable Report</b> <b>D2.4, 31/12/2014, Revision: Final Version</b>	Page 9 of 37

example the haptic devices, the 3D tracker, the rotating knobs etc., while the output interface is responsible for showing updated views to users.

The output views are divided into three panels. The first one is the patient view, which displays the scene being rendered including the virtual patient and the needle. The second one is the ultrasound screen displaying in real time the ultrasound image while the trainee is moving the ultrasound tracker probe. The third view is the course-ware view, which is the main component that connects the simulator with the server side database and resources. The course-ware view displays the users authentication/profile data, available scenarios, training material, users training history (performance metrics, training time), and performs bi-directional communication with a central server to receive system updates (software, image library, training material) and transmit updated users data (profile/training history).

Thus, the need for the development of a local and a global information service, that will provide the relevant functionalities for the course-ware implementation and its “seamless” integration with the rest of the RASimAs modules, becomes apparent.

### **3.2 Use case Scenarios**

Before proceeding into the architecture design we describe a few indicative use scenarios that highlight the typical use of this component.

#### **3.2.1 Authentication**

Initially the user is required to log into the course-ware by providing its credentials. The course-ware communicates with the local database and proceeds to the relevant authentication.

#### **3.2.2 Course-ware Loading of User Profile/Performance Metrics**




As soon as user is authenticated, course-ware loads from the local database the level, the training hours, and the performance metrics of the trainee in order to provide the available training scenarios.

#### **3.2.3 Course-ware Loading of Training Scenarios/Material**

According to the training procedure described in Deliverables 2.1 and 2.2 the course-ware provides training scenarios and materials (videos and documents) which are loaded from the local database.

#### **3.2.4 Course-ware Uploading of User Performance Metrics**

Either in sequential (i.e. “real-time”, during the training session) or batch update mode the course-ware is requested to connect with the local database and upload the user performance metrics.

	 FP7-ICT-2013-10 - 5.2 - Virtual Physiological Human	
Project No. 610425	<b>Deliverable Report</b> <b>D2.4, 31/12/2014, Revision: Final Version</b>	Page 10 of 37

### 3.2.5 Local and Global Databases Synchronization

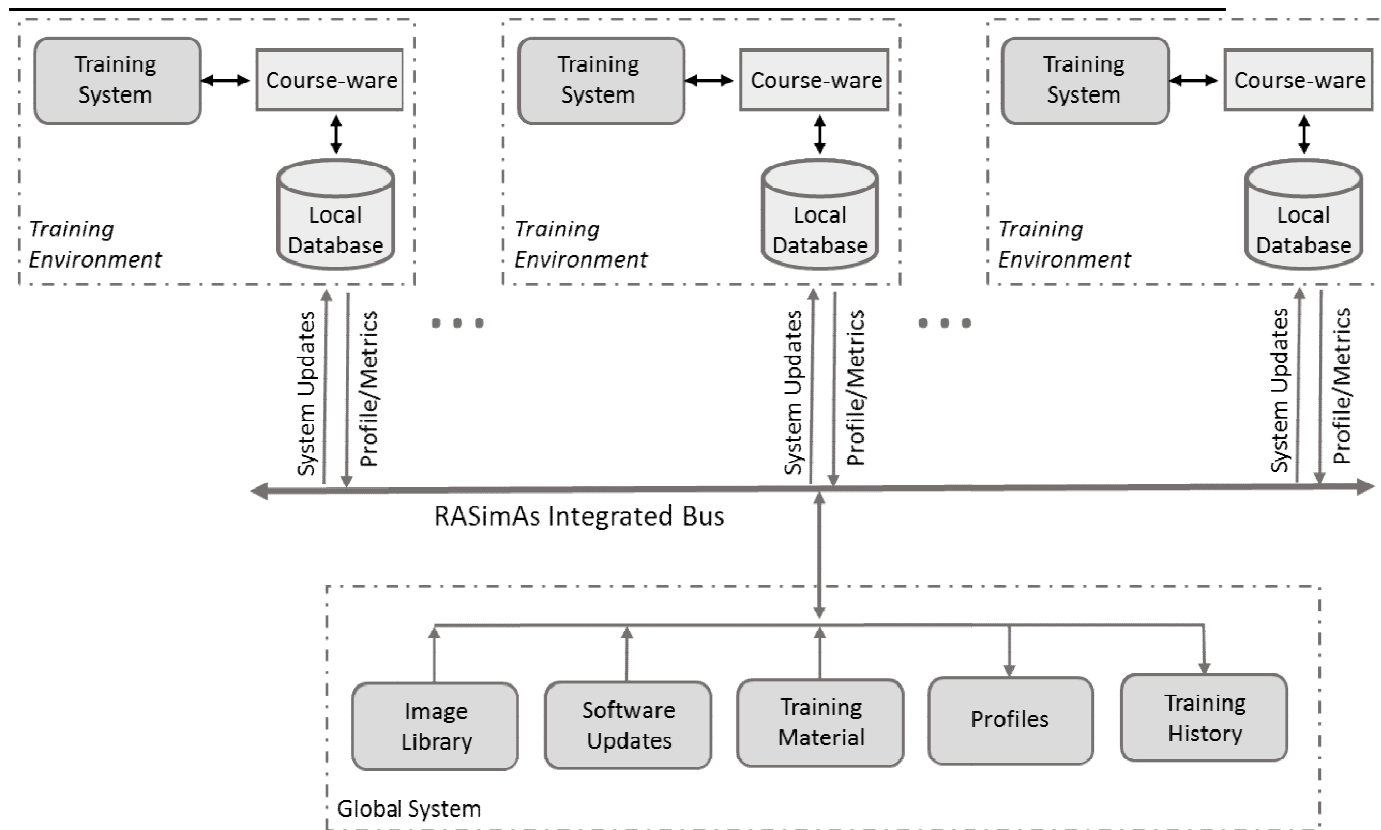
Periodically, the local databases, installed in every training system, will connect to the central database and upload all user related profile and metrics data in order for this information to be accessible outside training premises.

In addition, the local databases will download updates related to the system (software, image library and training material) which will then be made available to the course-ware.

## 4 Logical Architecture

Based on the motivation and the scenarios described in the previous paragraphs, a two level decomposition of the RASimAs architecture has to be considered as shown in Figure 2 below. The “training environment” is located in the participating hospitals’ premises where the RA training takes place. Of course this environment is “replicated” in every hospital that is part of the “RASimAs system”. On the other hand, there are certain components and services that are hosted in the “Central Information Server” (Global system) for the completion of the RASimAs platform. This is essentially a “star network” architecture scheme where the central components are installed once and provide the information “hub” while the distributed “training/assistant environments” are connected to a common central system.

Communication between the central and the local (i.e. hospital’s) components is facilitated through the use of a system-wise “software communication bus”.



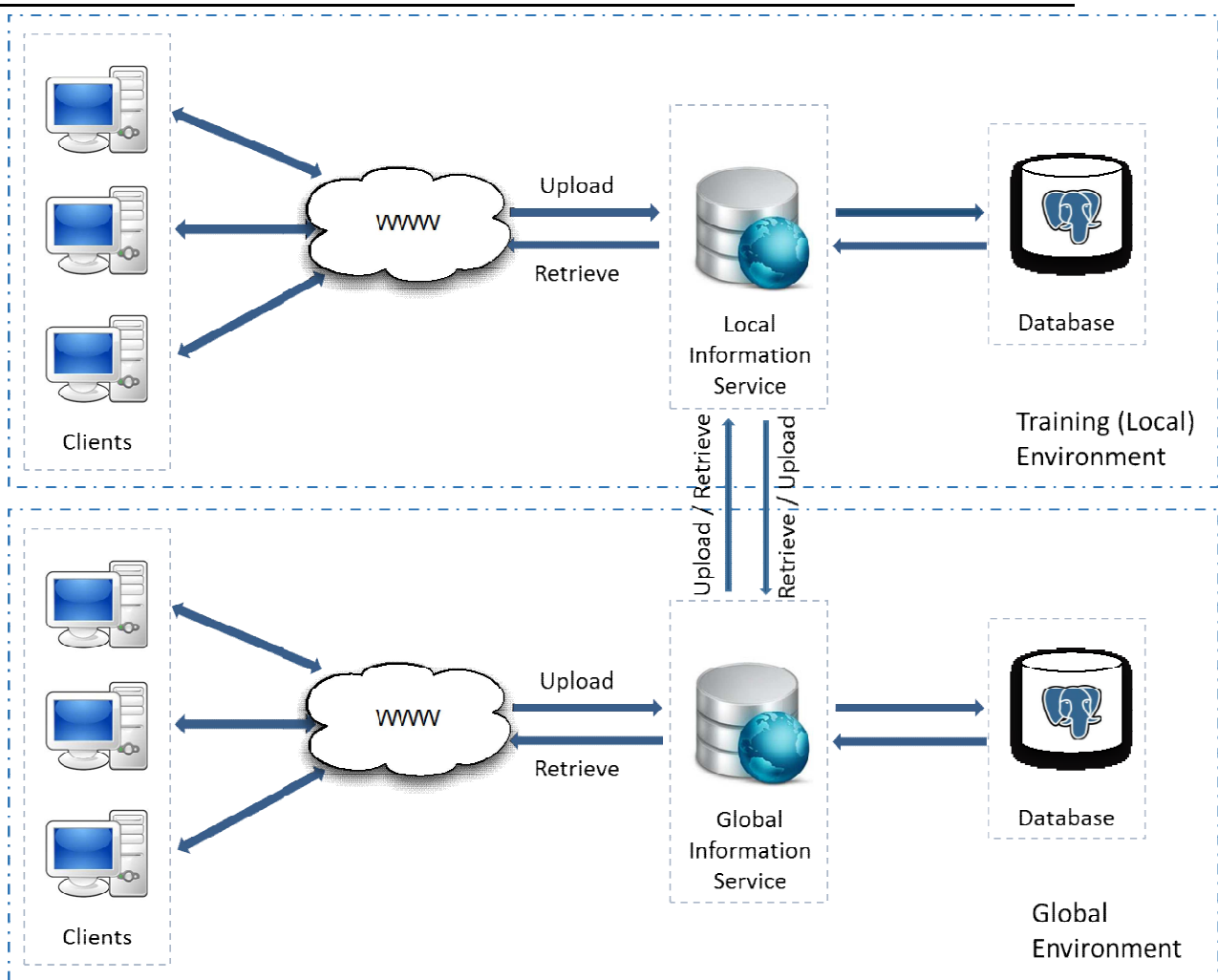
**Figure 2 – RASimAs Architecture Design of Local and Global Systems Communication**

## 5 Architecture Design

### 5.1 Introduction

An important step prior the actual fulfillment of the Integrated Platform is the implementation of the architecture design. The architecture design encloses all the vital components comprising the service, and additionally the way these components interact. This service can be described by the following distinct components (Figure 3):

- I. Relational Database (Profile/Metrics/System Updates)
- II. Web Application (Service access via internet/intranet)



**Figure 3 – Integrated Platform Architecture**

As shown in Figure 3 above, the Integrated Platform Service permits clients to access the database, performing both uploading and retrieval of system and users data.

The communication between local and global environments is restricted to users' data and metrics uploading from local to global, and system updates uploading from global to local.

### **5.2 Role Based Access Control (RBAC)**

In computer systems security, role-based access control is an approach to restricting system access to authorized users. Within an organization, roles are created for various job functions. The permissions to perform certain operations are assigned to specific roles.

In this direction and for this particular service, different roles have been assigned with specific access rights. Therefore, in the RASimAs platform, we have defined the following roles:

- Admin: this is the local or global administrator of the system which generally has full access rights
- Mentor: the role of a user that has increased access rights in a local organization hosting the training environment of RASimAs, since he/she is responsible (“mentoring”) for a number of trainees
- Trainee: this is the role for the users participating in the training sessions of the platform.

In the following sections we provide details about the access rights for each of these roles.

### 5.2.1 Local Service

Table 1 below indicates the Role Based Access Control (RBAC) of the present system in the local version. According to this, *Admin* stands at the top of hierarchy with full (read/write) access rights to both *Mentor* and *Trainee*, while *Mentor* comes second with full access rights to *Trainee* only.

Role	Functionality							
	Mentor Registration	Get All Mentors	Trainee Registration	Get All Trainees	Get Profile	Update Profile	Get Training History	Get Training Material
Local Admin	✓	✓	✓	✓	✓	✓	✓	✓
Mentor			✓	✓	✓	✓	✓	✓
Trainee					✓	✓	✓	✓

**Table 1 – Local System Role Based Access Control**

In the same way, Table 2 specifies the access rights, for each role, related to the *Profile* functionality. In accordance with this, *Admin* has full access rights to his/her profile excluding *Level* field, full access rights to *Mentor* profile excluding *Level* and *Organization* fields, and full access rights to *Trainee* profile excluding *Organization* field too. On the other hand *Mentor* has full access rights to his/her profile excluding *Level* and *Organization* fields, and full access rights to *Trainee* excluding *Organization* field. Finally, *Trainee* has full access rights to his/her profile only, excluding *Email*, *Level*, and *Organization* fields.

Role	Profile Functionality Field						
	Title	First Name	Last Name	Email	Password	Level	Organization
Local Admin	✓	✓	✓	✓	✓	✓*	✓**

Mentor	✓	✓	✓	✓	✓	✓*	
Trainee	✓	✓	✓		✓		

**Table 2 – Local System Profile Functionality Access Rights**

\* Restricted to change only *Trainee Level*

\*\* Restricted to change only his/her *Organization* field. Then automatically all *Mentors* and *Trainees Organization* follows this change.

### 5.2.2 Global Service

In terms of the *Global Service*, Table 3 below indicates the corresponding RBAC. According to this, *Admin* of the global service has read-only access to the rest of the roles, read/write access to his/her profile excluding *Level* field, and the right to upload system updates (software, training material) which will then be synchronized with the local services. In addition, *Local Admin* and *Mentor* have read-only access to *Mentor/Trainee* and *Trainee* respectively that belong to the same organization, while *Trainee* has read-only access to all the information related to his/her account.

Role	Functionality									
	Get All Local Admins	Mentor Registration	Get All Mentors	Trainee Registration	Get All Trainees	Get Profile	Update Profile	Get Training History	Get Training Material	Upload System Updates
Admin	✓		✓		✓	✓	✓*	✓	✓	✓
Local Admin			✓**		✓**	✓**		✓**	✓	
Mentor					✓**	✓**		✓**	✓	
Trainee						✓		✓	✓	

**Table 3 - Global System Role Based Access Control**




\* Restricted to change his/her profile only, excluding *Level* field.

\*\* Restricted to all *Mentors* and/or *Trainees* that belong to the same *Organization*.

## 5.3 Components

### 5.3.1 Relational Database

In the realization of the data storage and retrieval, a database and a relational *database management system* (RDBMS) are required. This relational database is the main storage component of the application and its main functionality is the indexing of the uploaded data.

	 FP7-ICT-2013-10 - 5.2 - Virtual Physiological Human	
Project No. 610425	<b>Deliverable Report</b> <b>D2.4, 31/12/2014, Revision: Final Version</b>	Page 15 of 37

A database is a large collection of data organized especially for rapid search and retrieval (e.g. by a computer), and a database management system is usually a set of libraries, applications, and utilities that relieve an application developer from the burden of worrying about the details of storing and managing data. In addition, an RDBMS provides facilities for searching and updating records that are persisted in the form of “relations” (tables)<sup>1</sup>.

This component is the core storage facility, supporting not only the storage and indexing of the uploaded data, but also the user management (e.g. credentials, access rights, etc.). A notable detail here, with respect to the storage of the uploaded files, is that, due to the possible large size and number of these files, we opt for storing them in the file system (in date-stamped folders) rather than internally in the database. This gives some performance advantages, but the biggest advantage is that the database is “lighter” and arbitrarily large files can be handled. Additionally, it is worthy of noting that, for security reasons, the database does not store the users’ credentials in clear text. Instead, the user passwords are stored in a cryptographically secure, “hashed” and random “salted” form to prevent direct access and dictionary attacks.

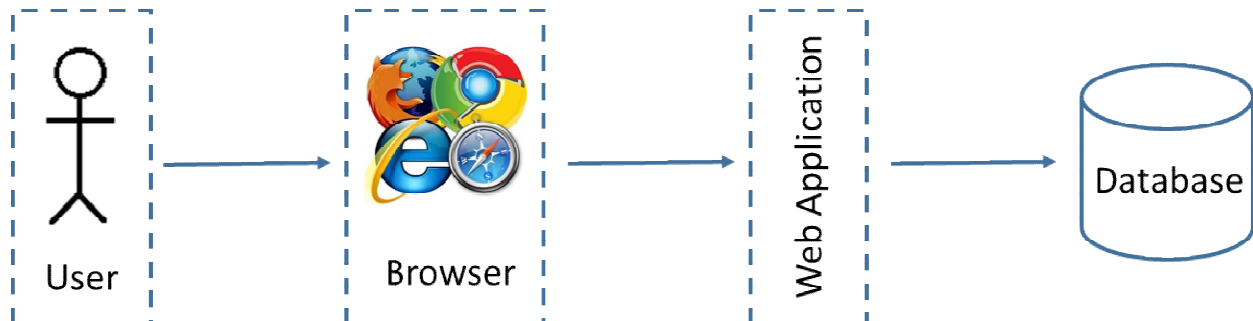
### 5.3.2 Web Application

In order to access the *Integrated Platform* service remotely, a *web application* is needed which is a software component that is contacted by the web browser using the *HTTP/HTTPS* protocols (Figure 4). In the RASimAs *Integrated Platform* system, the web application accepts the client requests, transforms, and forwards them to the database for uploading/downloading data. It subsequently returns the outcome of this interaction to the browser for display.

Web applications commonly use a combination of server-side script (*ASP, PHP, etc.*) and client-side script (*HTML, Javascript, etc.*) in developing the application. The client-side script deals with the presentation of the information, while the server-side script deals with all the demanding processes, including (but not limited to) information storage and retrieval.

---

<sup>1</sup> Neil Matthew and Richard Stones. *Beginning Databases with PostgreSQL: From Novice to Professional*. 2<sup>nd</sup> Ed. New York: Springer New York, 2005



**Figure 4 - General Web Application Representation**

## 6 Implementation

### 6.1 Technological Solutions Used

#### I. RDBMS

The *RDBMS* chosen for the current service is *PostgreSQL*<sup>2</sup>. *PostgreSQL* is a free and open-source *DBMS* and supports the relational model for its databases and the *structured query language* (SQL). It contains just about all the features that a user would find in other commercials or open-source *RDBMS*, and a few additional features that are not provided by other *RDBMS*. *PostgreSQL* can trace its family back to 1977 at the University of California Berkley (UCB). It serves this system by indexing, storing and retrieving information.

#### II. Web Application

For the present service, on the server-side, the *SPARK* “micro” web framework has been chosen<sup>3</sup>, while the client-side was developed with script *FreeMarker*, which is a template engine that combines *html* and *javascript* code with *java* objects (Figure 5<sup>4</sup>). On the client side again, the front-end web development was based on *Bootstrap*<sup>5</sup>. Both *SPARK* and *FreeMarker* are free and open-source.

<sup>2</sup> <http://www.postgresql.org/>

<sup>3</sup> <http://www.sparkjava.com/>

<sup>4</sup> Overview of *FreeMarker* workflow. Available from: <<http://freemarker.org/images/overview.png>>. [14 July 2014].

<sup>5</sup> <http://getbootstrap.com/>



**Figure 5 - FreeMarker Template Engine**

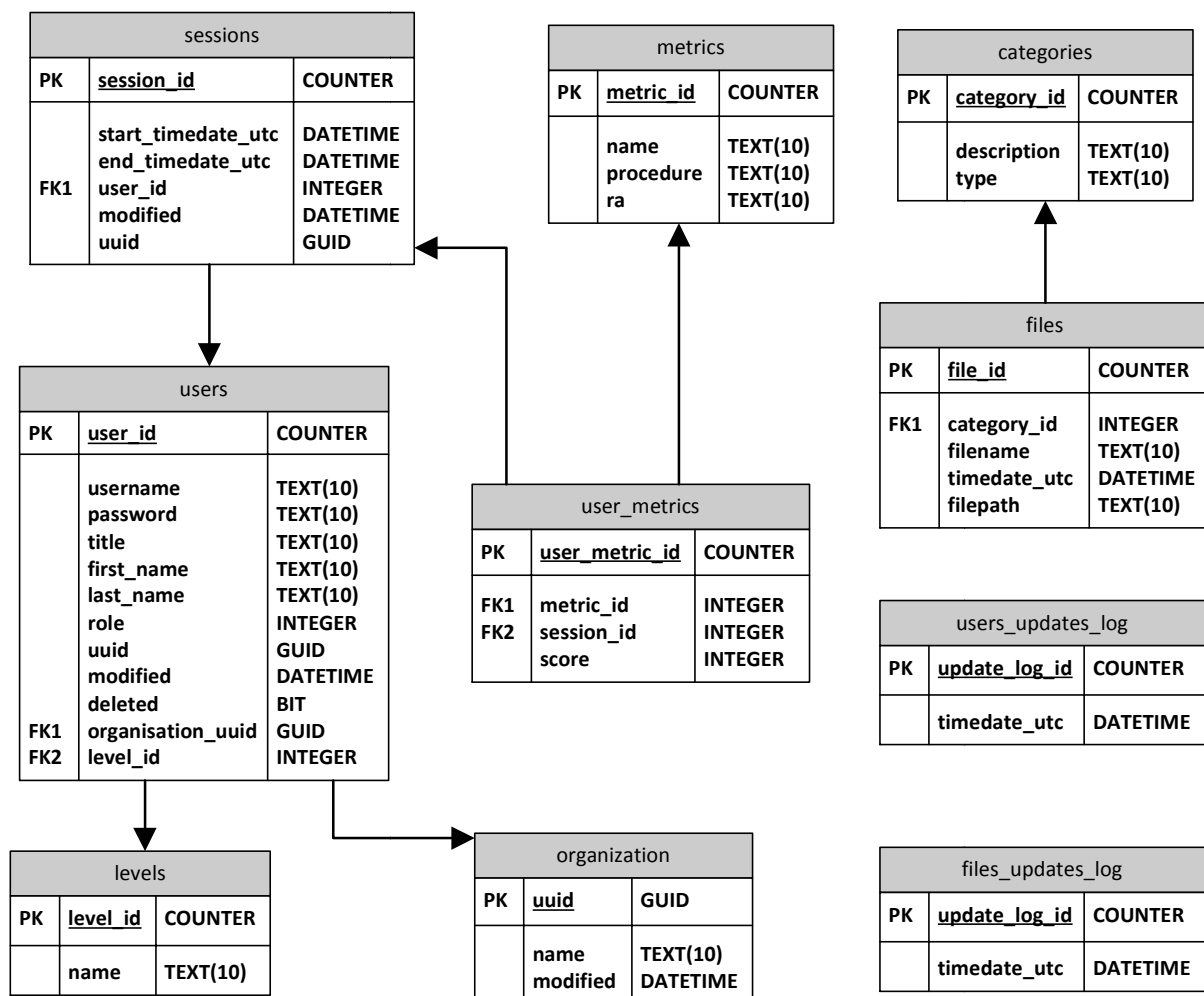
## 6.2 Database Schema

Databases are made up of tables, or in a more formal terminology, relations. A table (a collection of related data) contains *rows* of data (more formally called tuples), and each data row consists of a number of *columns*, or *attributes*. In an *RDBMS*, a schema defines the tables, the attributes in each table, and the relationships between attributes and tables. Figure 6 below describes the schema of the RASimAs *Integrated Platform* database. It consists of ten tables each one containing relative information. These tables are:




- *users* – Containing login details and personal information.
- *organization* – Containing users organization details.
- *levels* – Containing users levels (i.e. admin, mentor, trainee – novice, intermediate, ...)
- *metrics* – Containing all kind of performance metrics information (name, procedure, type (US/ES)) as described in Deliverable 2.1.
- *sessions* – Containing training session related information, like the user it belongs to, and details like starting and ending time.
- *user metrics* – Containing the type of metric, the session it belongs to, and the achieved score.
- *categories* – Containing the types of files stored in the database (system updates, training material, etc.)
- *files* – Containing all relevant information for a stored file, like the category it belongs to, the filename, the time it was uploaded, and the location where it is stored locally.
- *users\_updates\_log* – Containing information about the last successful synchronization of the local with the global service for updating global service with any modified or new entries in the local one.

- *files\_updates\_log* – Containing information about the last successful synchronization of the local with the global service for updating local service with any modified or new file entries in the global one.

In the schema shown below, the underlined attributes denote primary keys (prefix *PK*), those with a prefix *FK* denote foreign keys, while the bolded attributes denote that these fields cannot have empty values (*null*). A primary key is an attribute (or a collection of attributes) that uniquely defines the characteristics of each row and consequently cannot be duplicated, while a foreign key is an attribute (or collection of attributes) in one table that uniquely identifies a row of another table. The arrows in the schema are foreign-key driven and indicate the links/relationships between tables.



**Figure 6 – Database Schema**

	 FP7-ICT-2013-10 - 5.2 - Virtual Physiological Human	
Project No. 610425	<b>Deliverable Report</b> <b>D2.4, 31/12/2014, Revision: Final Version</b>	Page 19 of 37

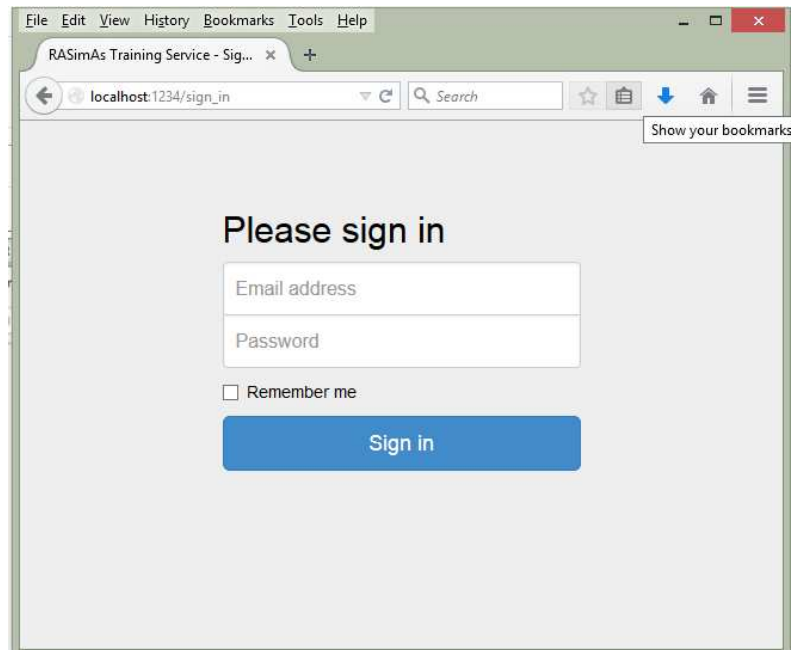
### 6.3 Functionality Testing

A critical step prior launching an application is the software testing. A primary purpose for this is to detect software failures so that defects may be discovered and corrected. Another reason is to assess if the current implementation meets the requirements that guided its design and development. On this goal, the following testing process was employed:

- I. RBAC testing – Functionality of the role based access has been extensively tested with all the restricted fields being disabled (read-only) accordingly in both local and global services. For the global service, where more than one organizations exist, the restriction of accessing roles (where applicable), of the same organization only, was also successfully tested.
- II. Registration testing – Functionality of the registration of new *Mentors* or *Trainees* in the local service was successfully tested.
- III. Personal account information testing – Functionality of the personal account information (*Profile*) was validated, and users were able to update their records successfully. In the case of *Admin* and *Mentor* that have access rights to other than their personal profiles, as discussed in the *Architecture* section, profiles were also updated successfully.
- IV. Metrics insertion testing – Functionality of the API (Application Programming Interface) responsible for the insertion of *Trainee's* performance metrics from the course-ware to the local database system has been tested successfully internally and with one of our partners (URJC).
- V. Synchronization testing – Synchronization between the local and the global services for the users' data and system files updates has been tested successfully.
- VI. Upload and downloading testing – System updates uploading from the side of the global *Admin*, and training material downloading from all users have been tested successfully.

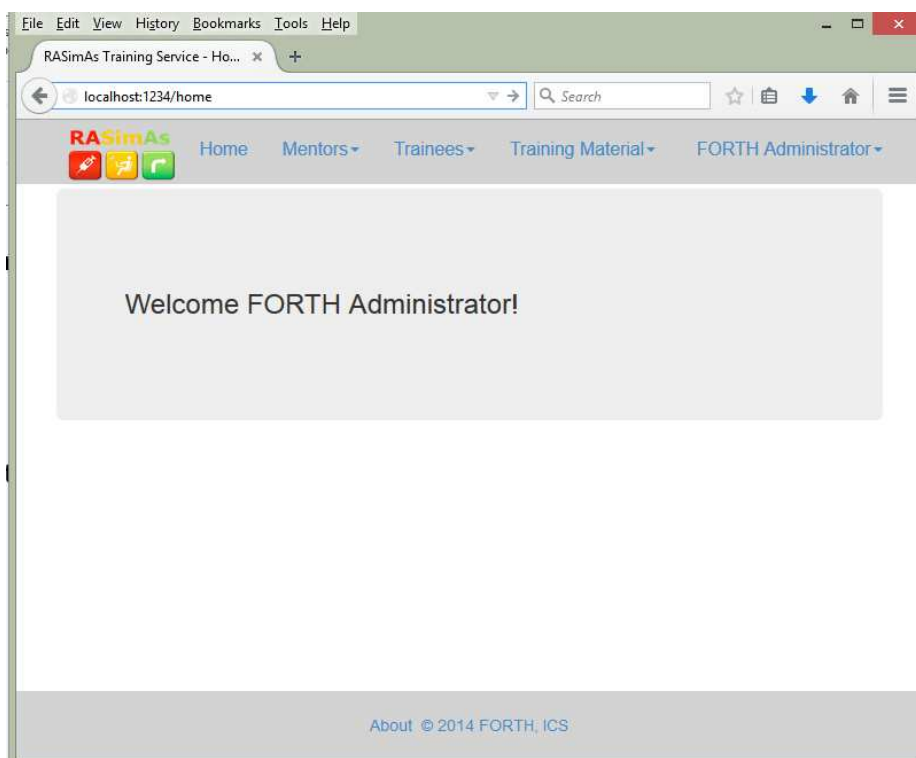
## 7 User Interface

In this section we provide a quick walkthrough over the main functionalities of the *Integrated Platform* component and the corresponding user interface elements. The initial "login" screen of the component when a registered user visits the component's web site is shown in the next figure (Figure 7).



**Figure 7 - The "sign-in" page**

The user is asked to provide his/her username and password and after these are correctly validated by the service the central web page is shown (Figure 8).

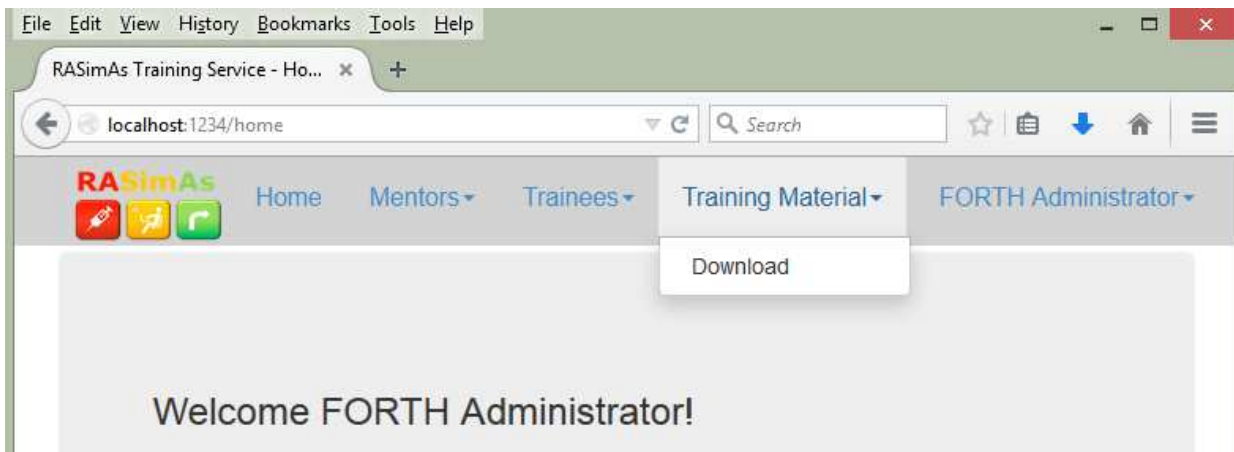
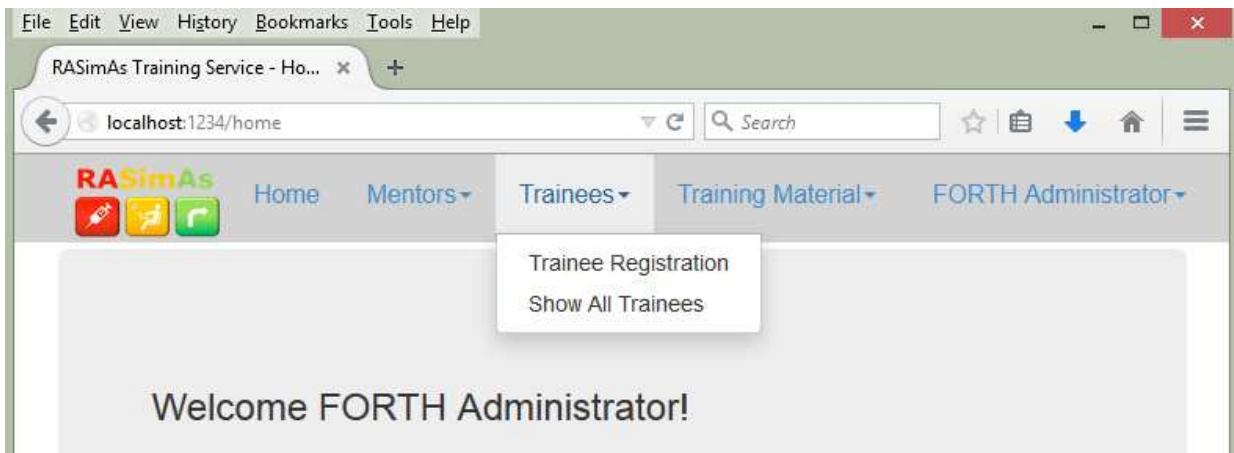
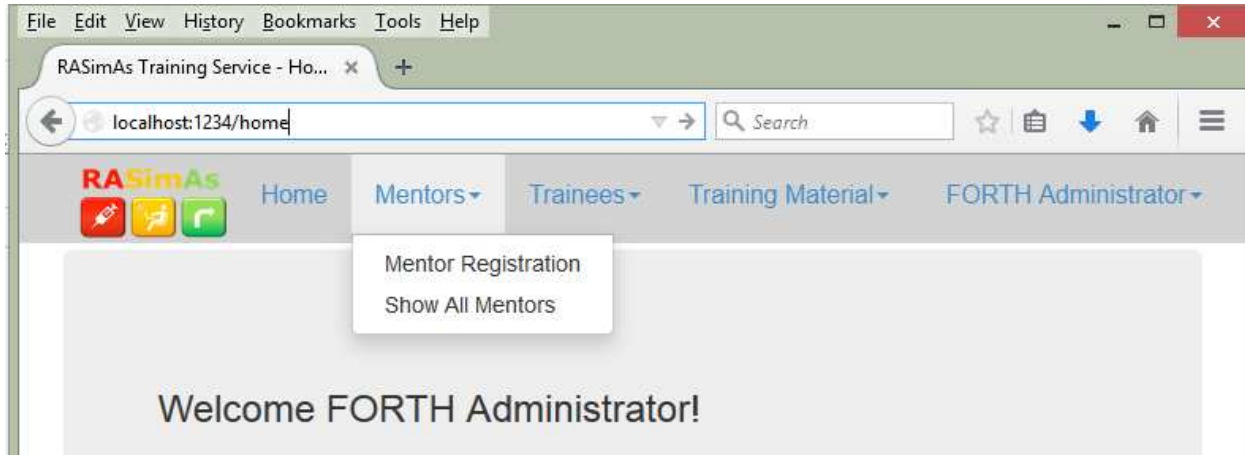


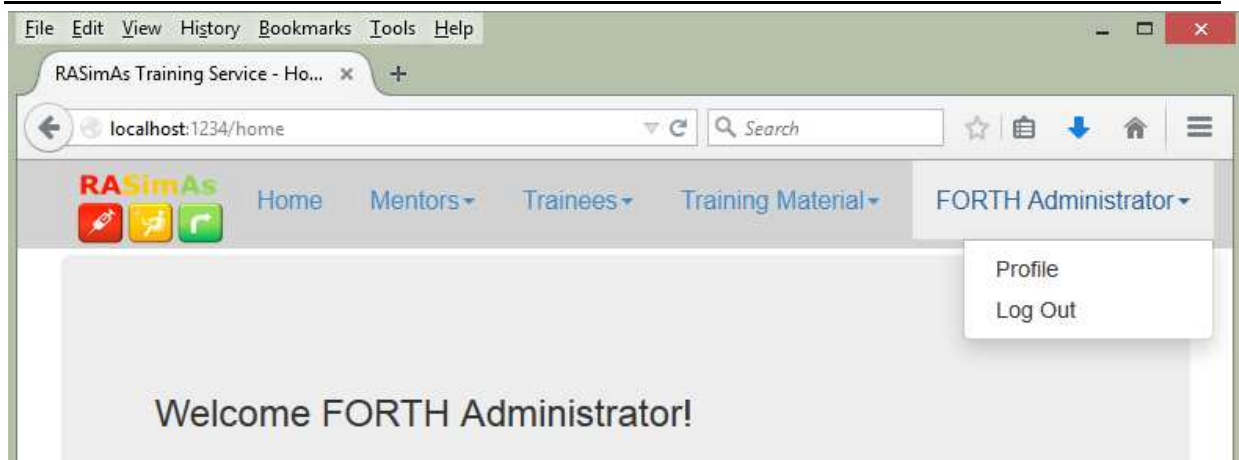
**Figure 8 - The welcome page (Local Admin)**

## 7.1 Local Service

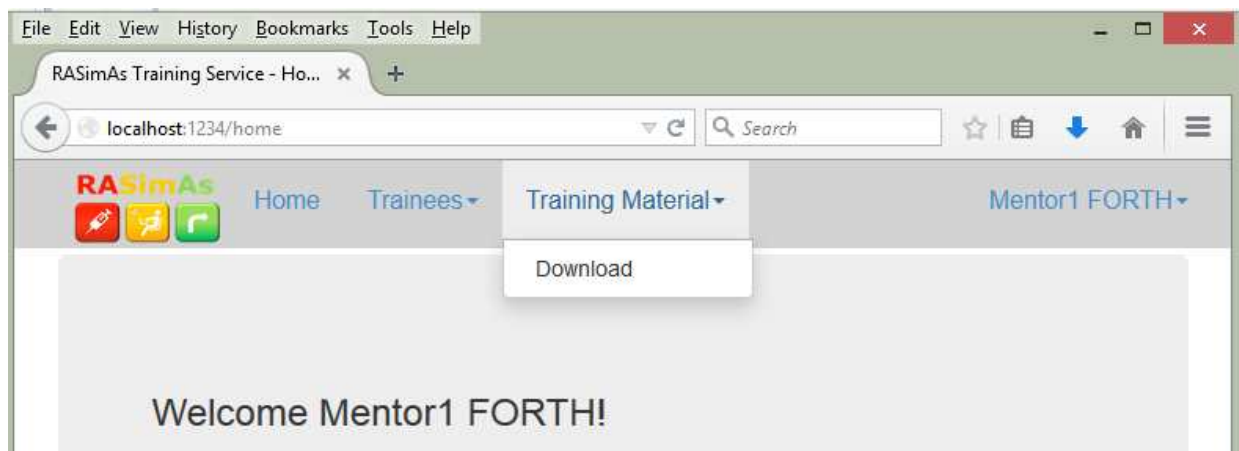
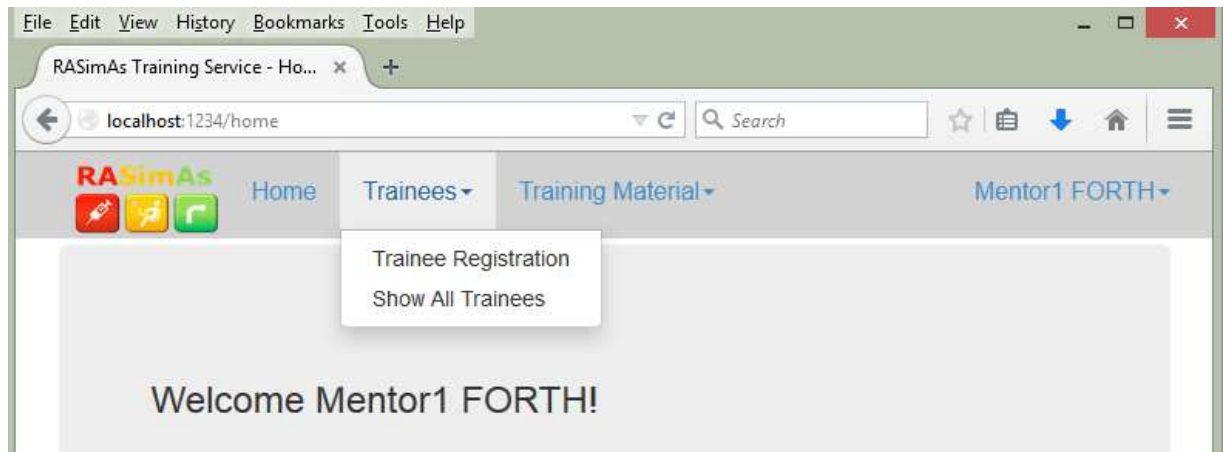
### 7.1.1 “Home” Menus

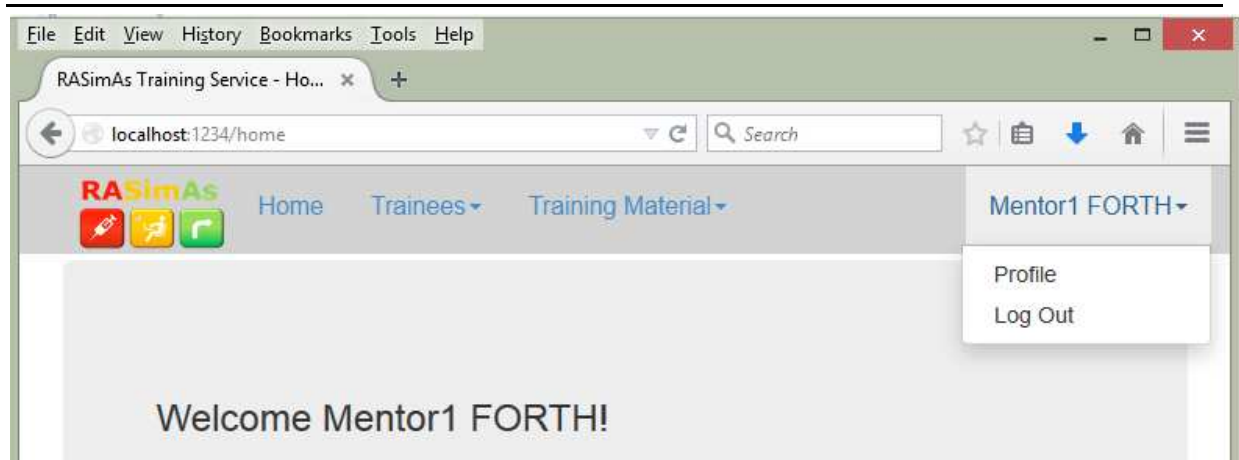
Figure 9, Figure 10, and Figure 11, below, present the “home” menus and their sub-categories of the *Local Admin*, *Mentor*, and *Trainee* respectively. As it is shown, these menus follow the previously defined *RBAC* specifications.



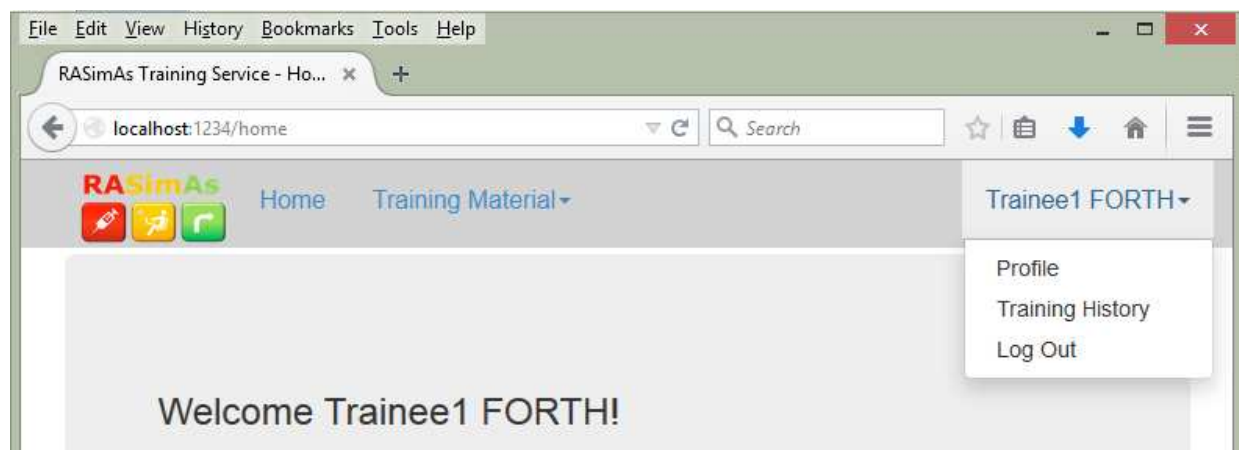
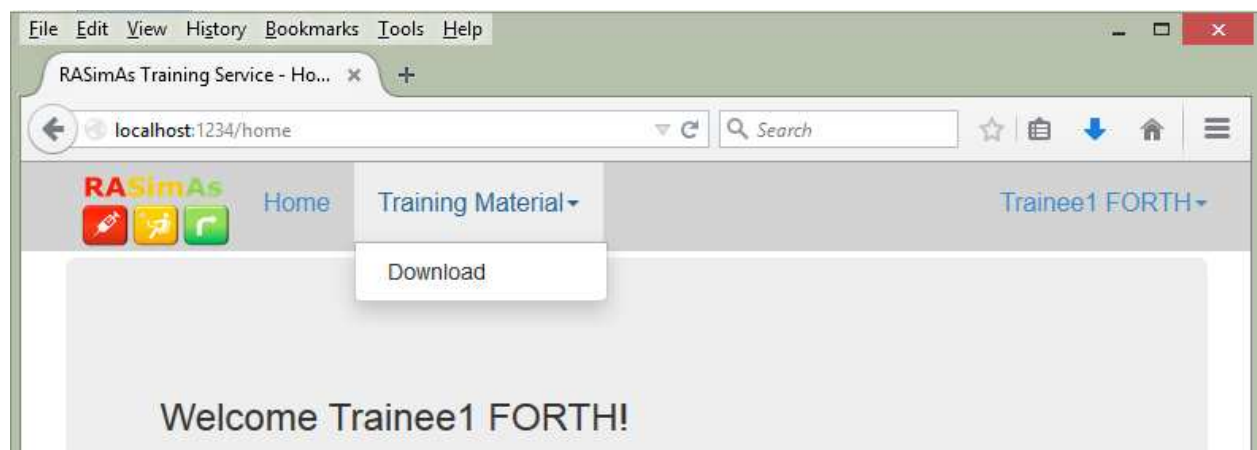


**Figure 9 - Local Admin “Home” menu**





**Figure 10 – Mentor “Home” menu**

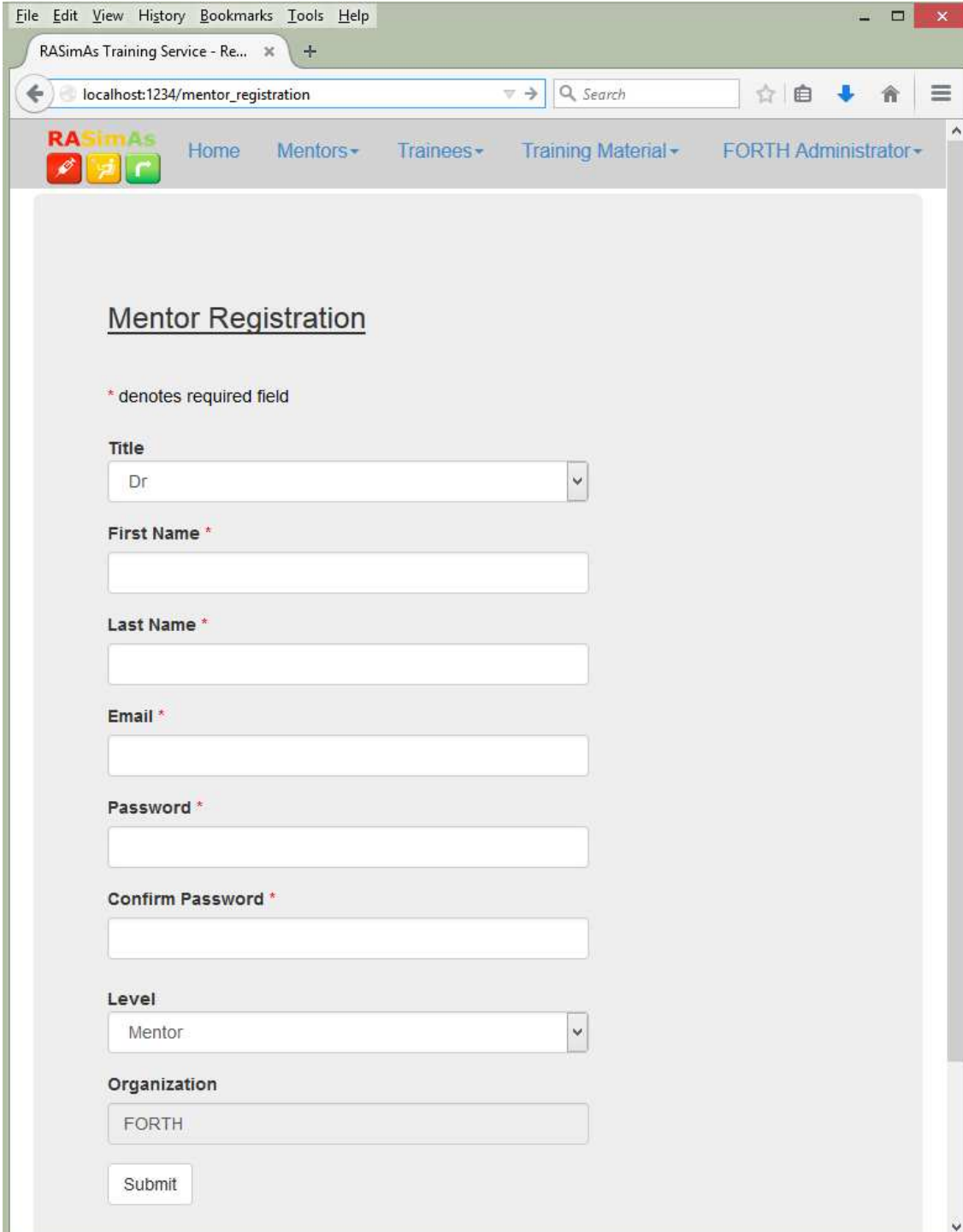


**Figure 11 – Trainee “Home” menu**

## 7.1.2 Functionalities

➤ Mentor Registration/Profile Information

Figure 12 and Figure 13, below, show the *Mentors'* registration and their full list functionalities which are permitted only in *Local Admin* role. A *Local Admin* has full read/write access to a *Mentor's* profile.

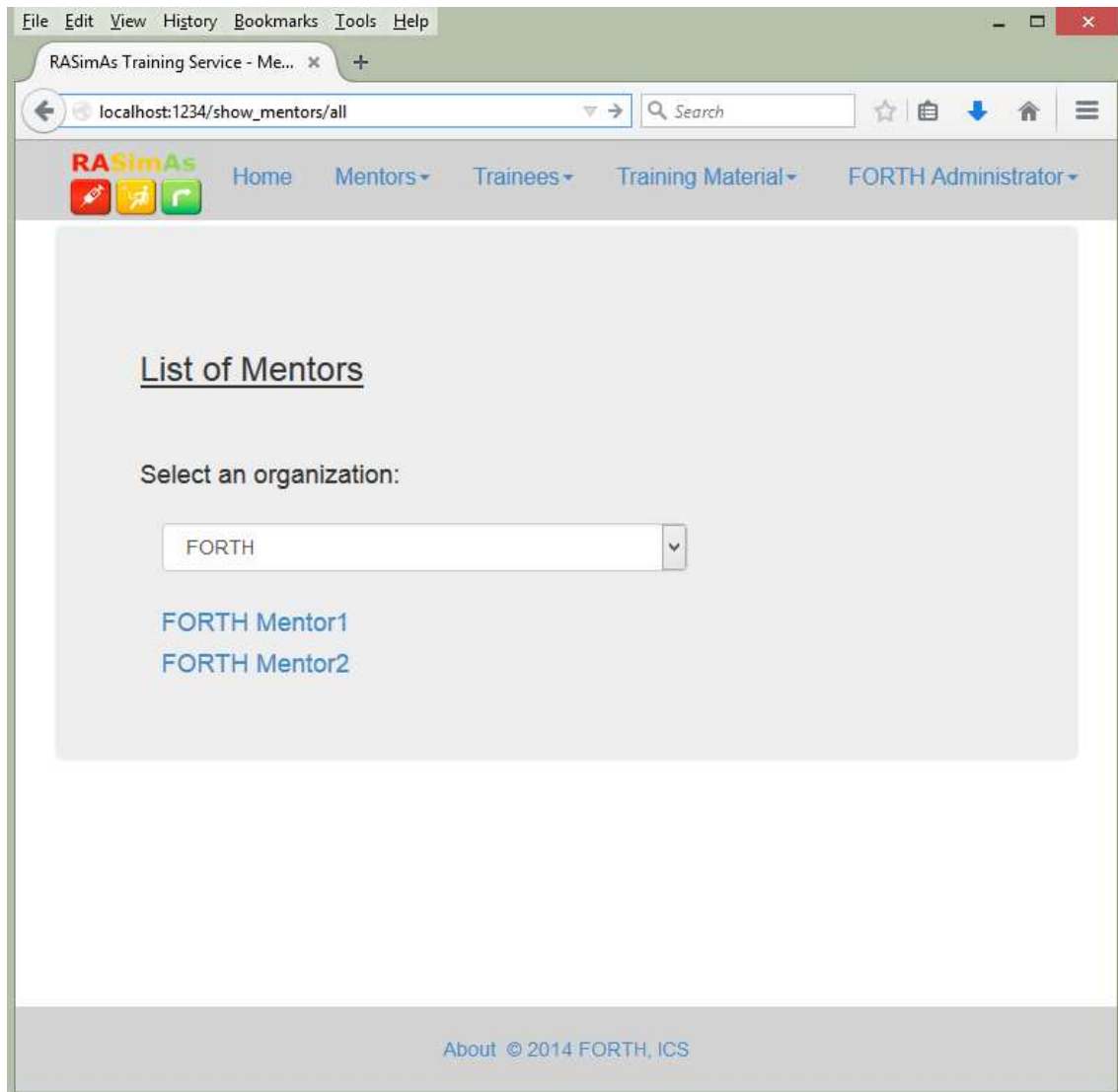


The screenshot shows a web browser window with the URL `localhost:1234/mentor_registration`. The page title is "RASimAs Training Service - Re...". The navigation menu includes "Home", "Mentors", "Trainees", "Training Material", and "FORTH Administrator". The main content area is titled "Mentor Registration" and contains a registration form with the following fields:

- Title**: A dropdown menu with "Dr" selected.
- First Name \***: A text input field.
- Last Name \***: A text input field.
- Email \***: A text input field.
- Password \***: A text input field.
- Confirm Password \***: A text input field.
- Level**: A dropdown menu with "Mentor" selected.
- Organization**: A text input field with "FORTH" entered.

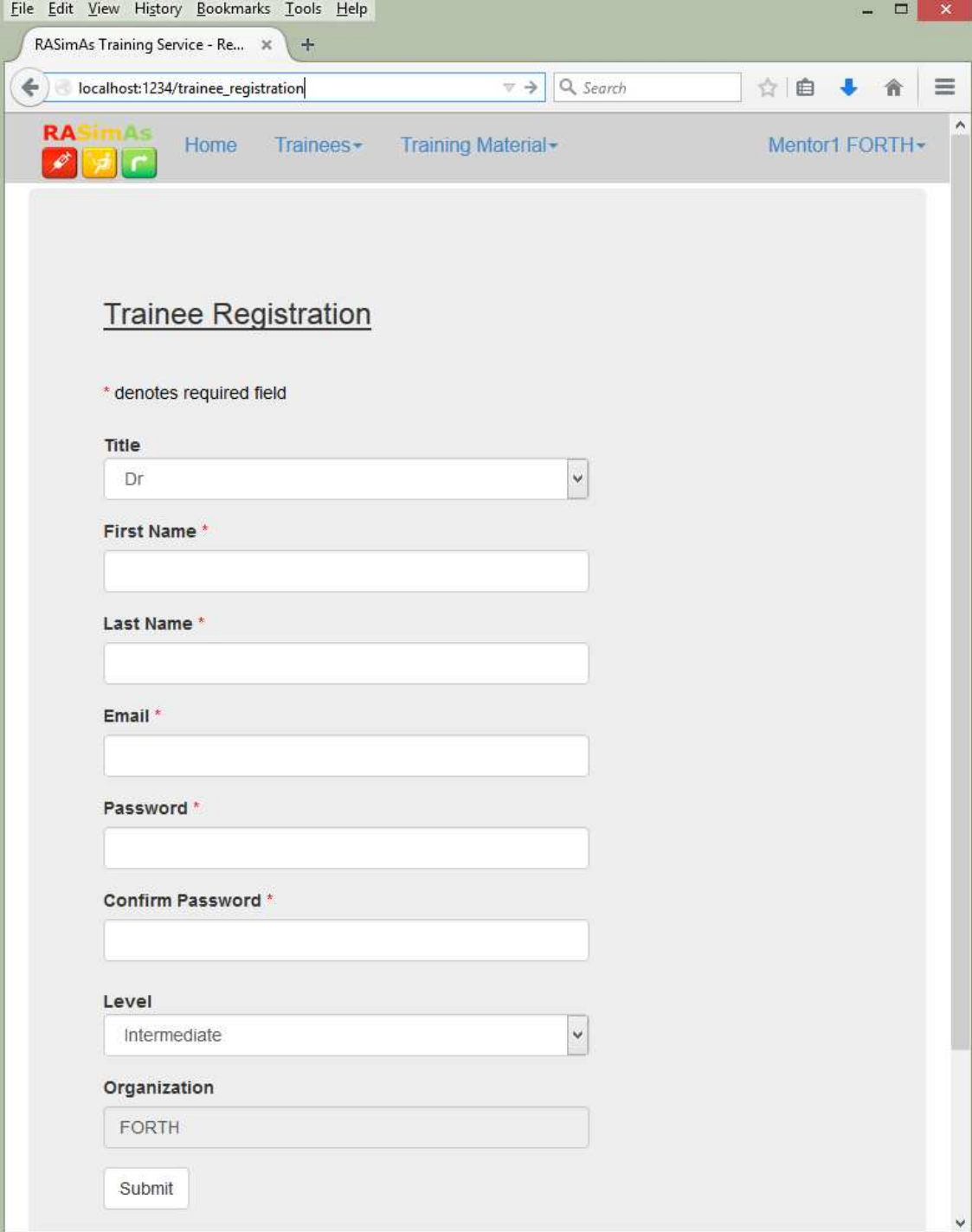
A "Submit" button is located at the bottom of the form. A note above the form states: "\* denotes required field".

**Figure 12 – Mentor Registration**



**Figure 13 – Mentor Profile Information**

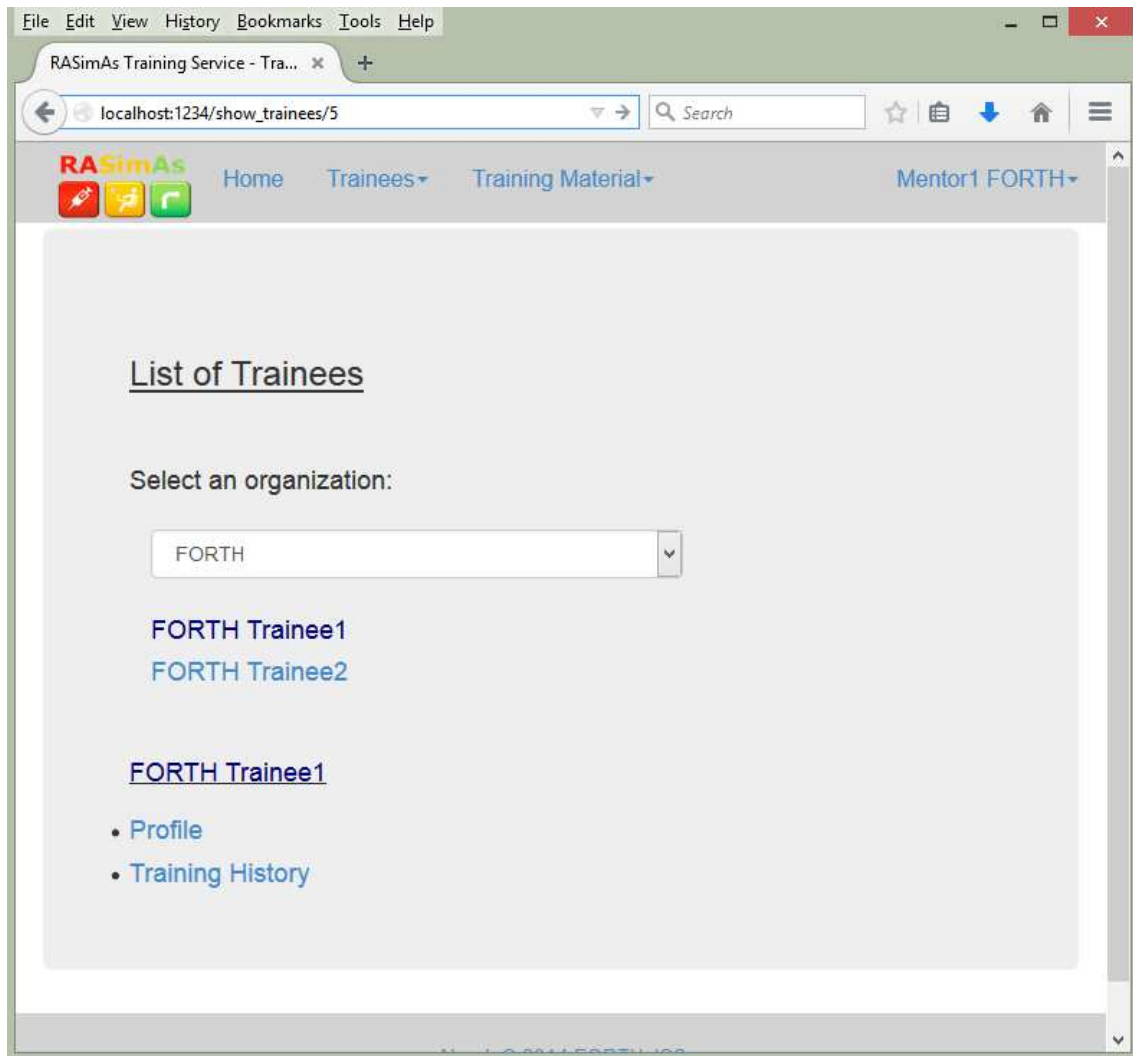
- Trainee Registration/Profile Information/Training History  
Figure 14 and Figure 15, below, show the *Trainees*' registration and their full list profile and training history functionalities, which are permitted in both *Local Admin* and *Mentor* roles. A *Local Admin* and a *Mentor* have full read/write access to all *Trainees*' profiles and read access to their performance metrics too.



The screenshot displays a web browser window with the following elements:

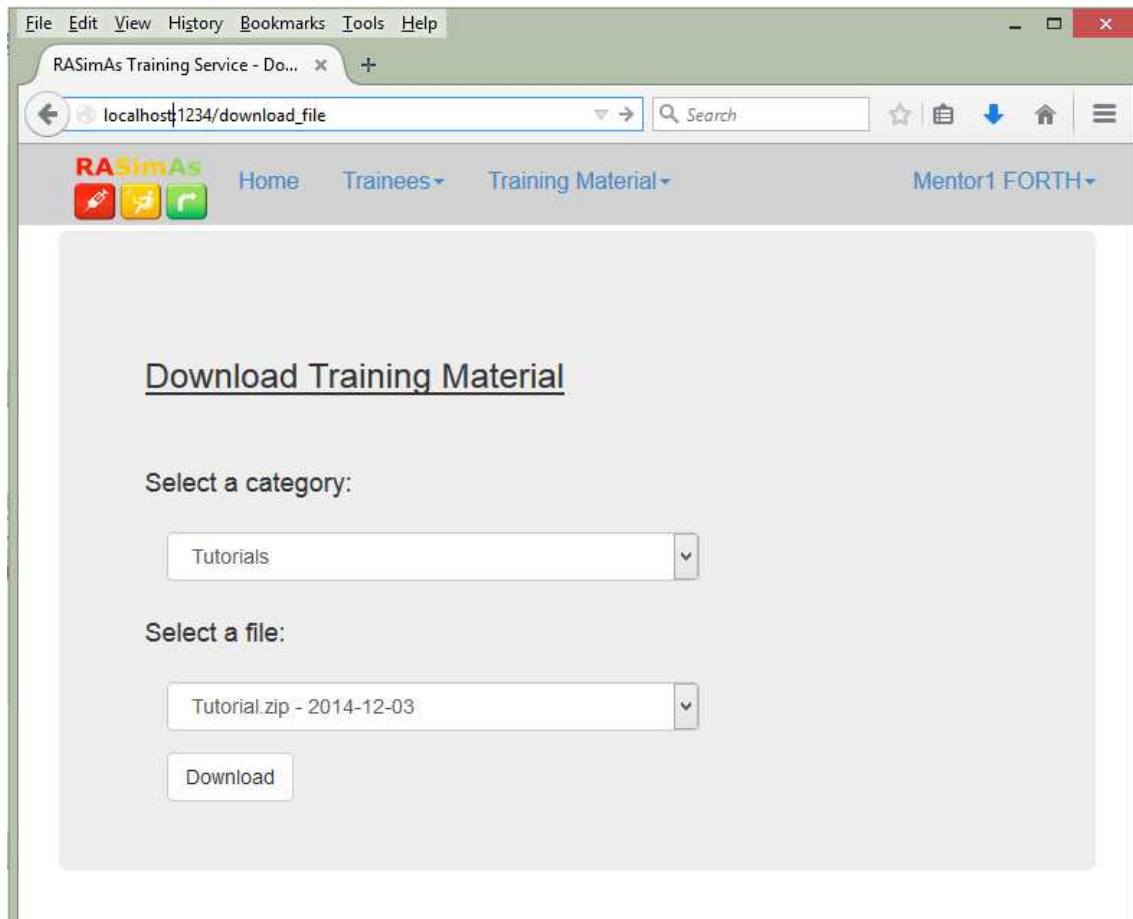
- Browser Tab:** RASimAs Training Service - Re...
- Address Bar:** localhost:1234/trainee\_registration
- Navigation Menu:** Home, Trainees, Training Material, Mentor1 FORTH
- Form Title:** Trainee Registration
- Legend:** \* denotes required field
- Form Fields:**
  - Title:** Dropdown menu with 'Dr' selected.
  - First Name:** Text input field.
  - Last Name:** Text input field.
  - Email:** Text input field.
  - Password:** Text input field.
  - Confirm Password:** Text input field.
  - Level:** Dropdown menu with 'Intermediate' selected.
  - Organization:** Text input field with 'FORTH' entered.
- Submit Button:** A button labeled 'Submit' at the bottom of the form.

**Figure 14 – Trainee Registration**



**Figure 15 – Trainee Information**

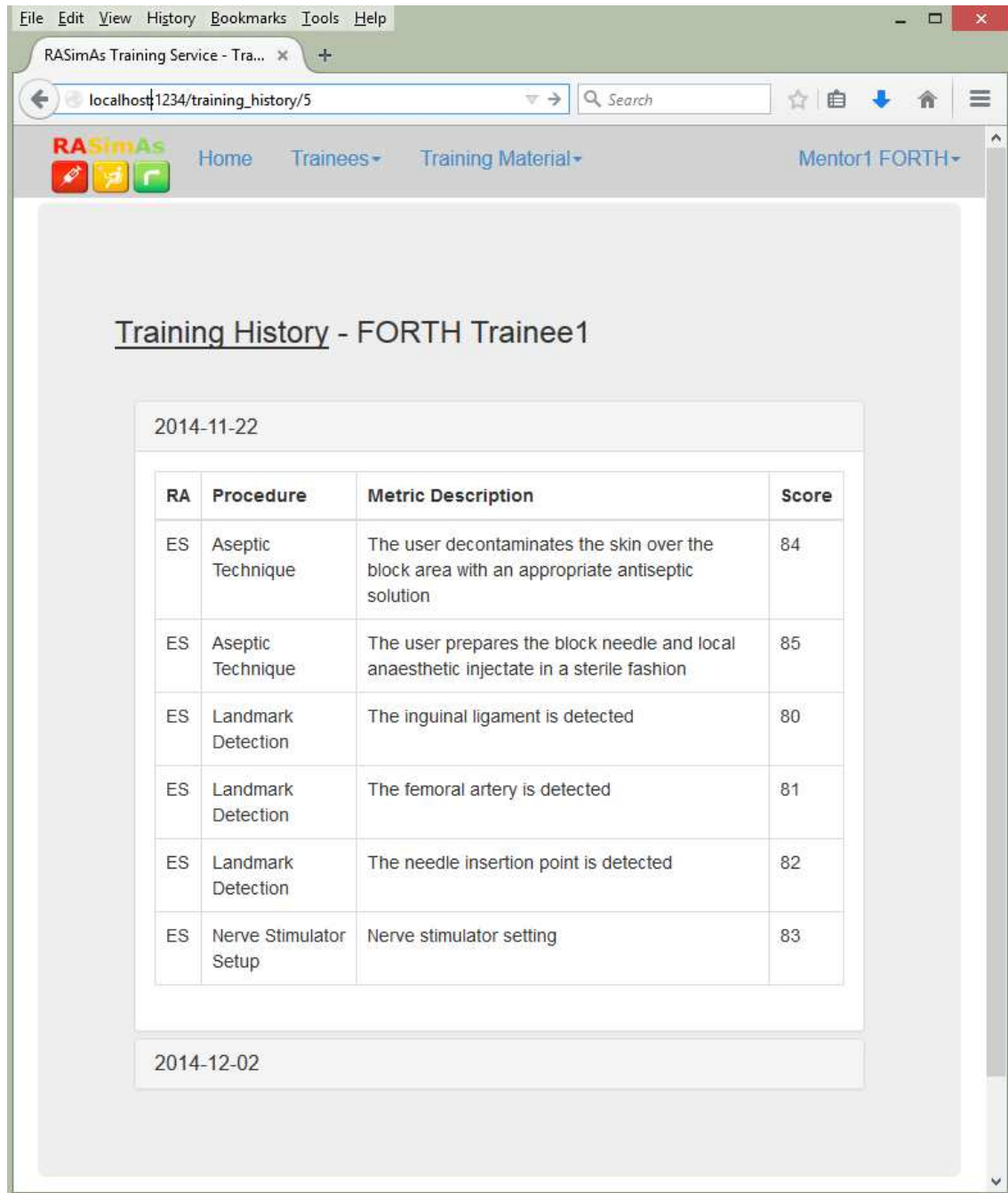
- Downloading of Training Material  
Figure 16 below shows the functionality of downloading training material. This operation is permitted to all users.



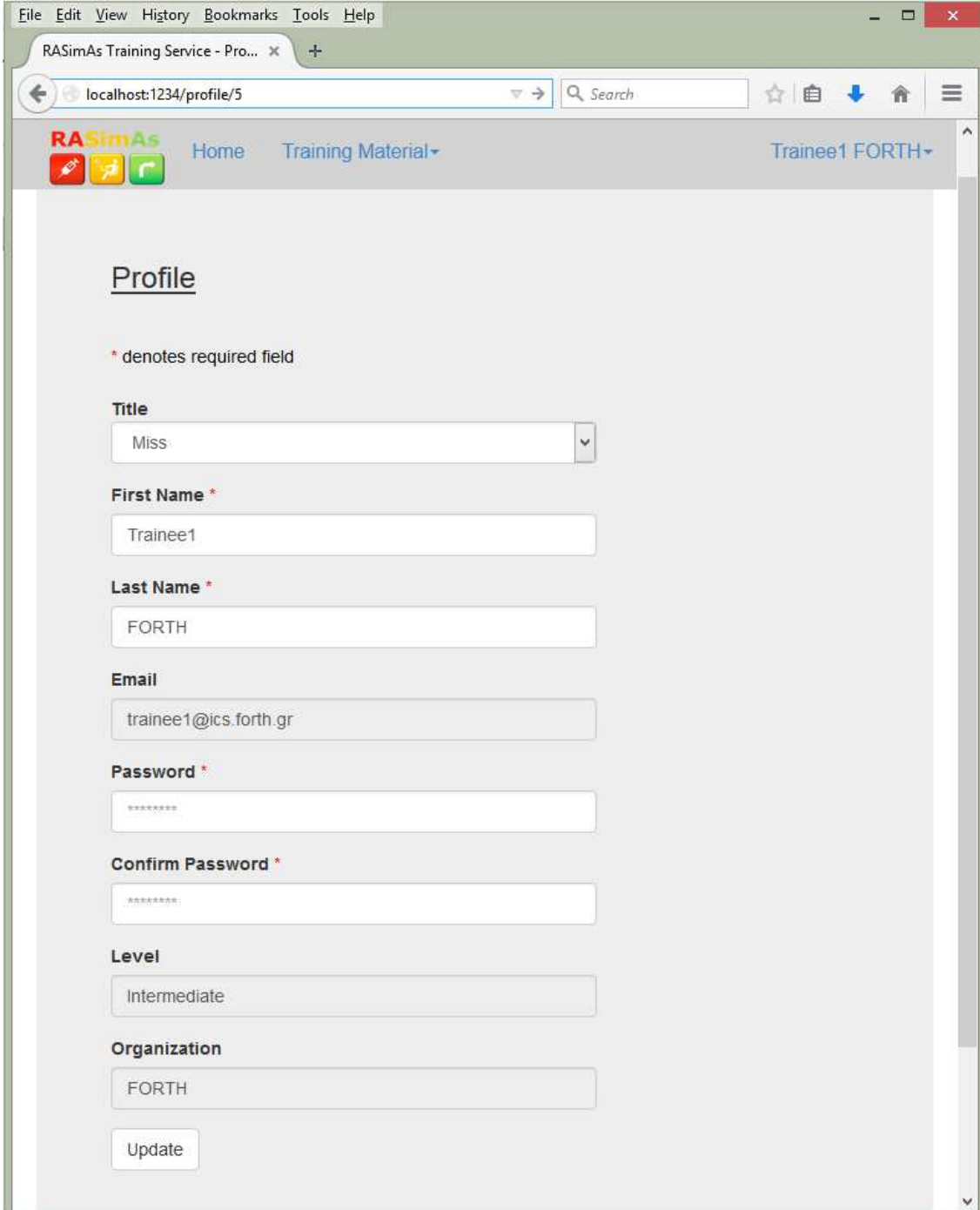
**Figure 16 – Downloading of Training Material**

➤ Training History and Profile

Figure 17 and Figure 18, below, show the training history and profile functionalities which are accessible by all users following the *RBAC* directives. A *Local Admin* may access all users' profile data and *Trainees'* metrics, a *Mentor* can only access *Trainees'* profiles and metrics, while a *Trainee* can access only his/her own profile/metrics. All users may have read/write access to the profile functionality.



**Figure 17 – Training History**



**Profile**

\* denotes required field

**Title**

**First Name \***

**Last Name \***

**Email**

**Password \***

**Confirm Password \***

**Level**

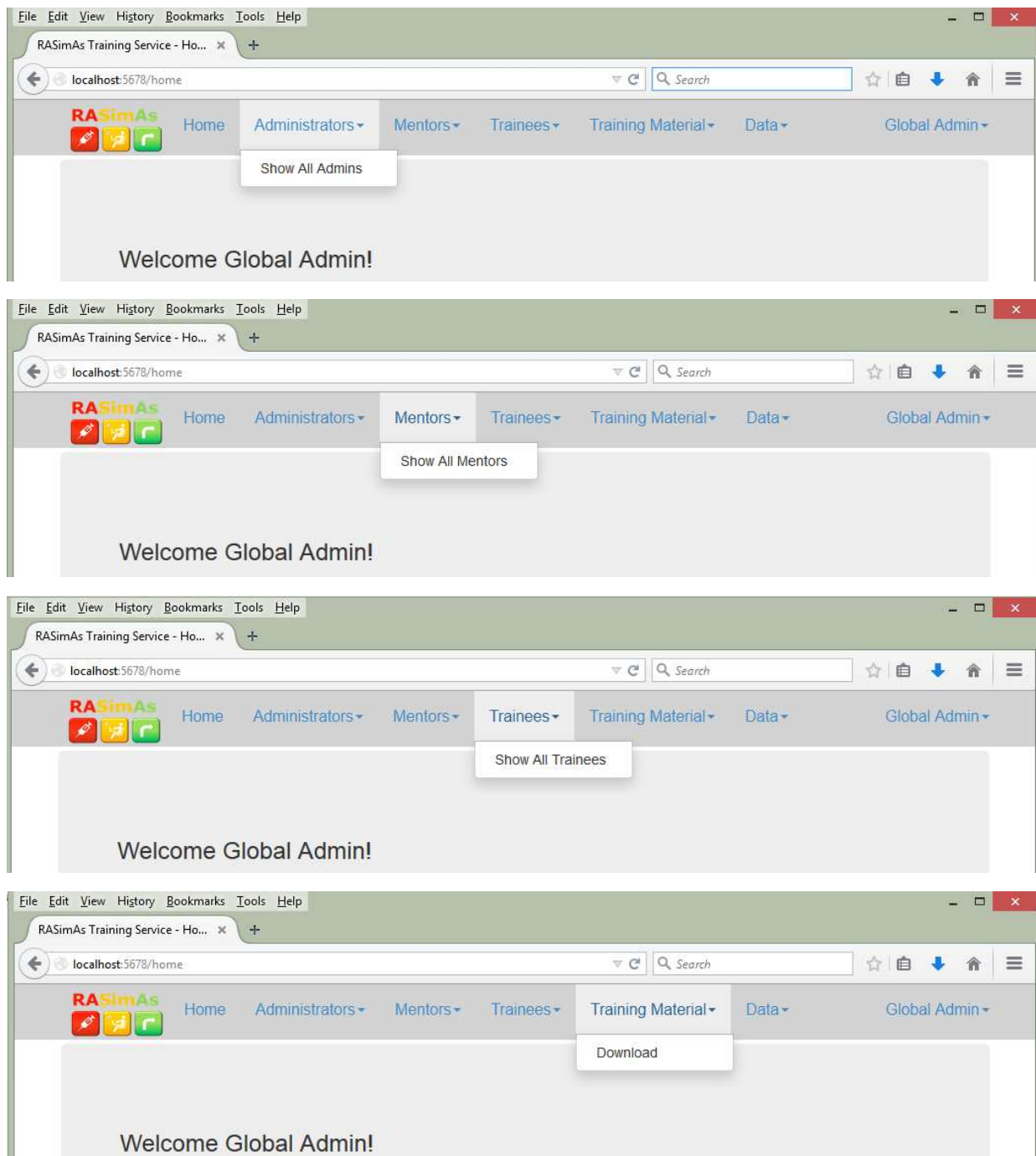
**Organization**



**Figure 18 – Profile**

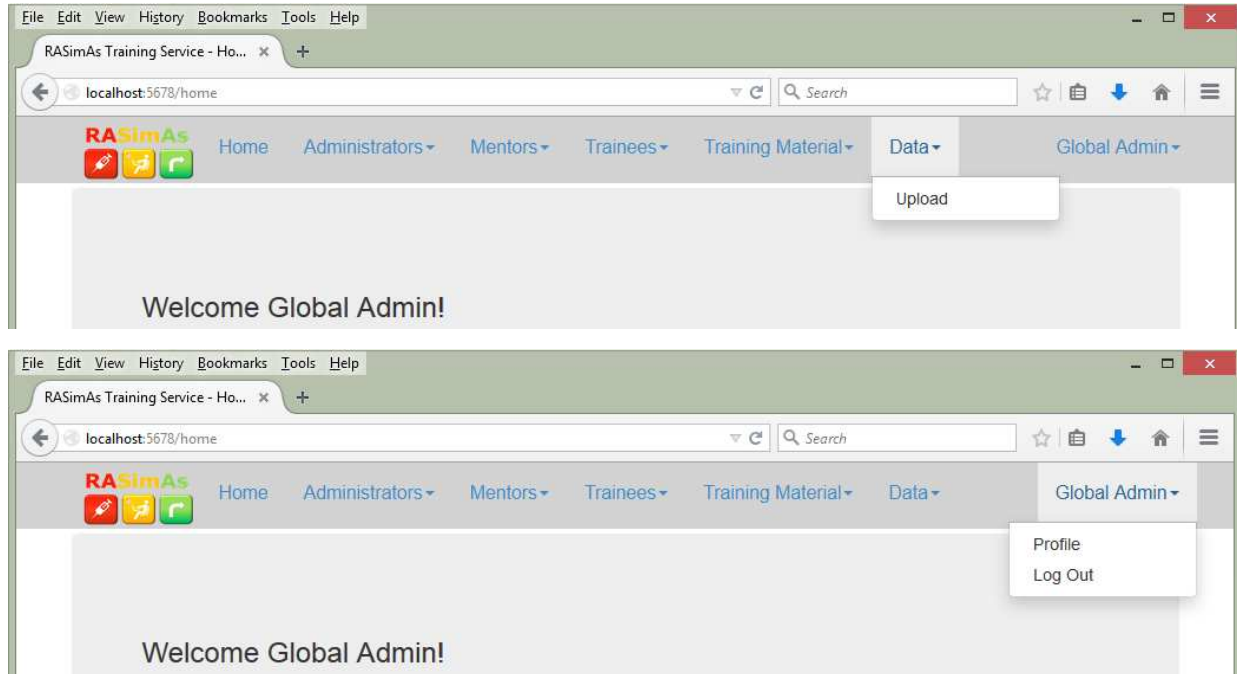
## 7.2 Global Service

### 7.2.1 “Home” Menus

Figure 19, below, shows the “home” menu of the *Global Admin* who has access to all users (*Local Admin*, *Mentor*, and *Trainee*) data. For the rest of the users the “home” menus are the same as for the local service except that registration functionality is not applicable any more. All users, except the *Global Admin*, have read-only access to their profile.





	<h1 style="margin: 0;">RASimAs</h1> <p style="margin: 0;">FP7-ICT-2013-10 - 5.2 - Virtual Physiological Human</p>	
<p>Project No. 610425</p>	<p><b>Deliverable Report</b> <b>D2.4, 31/12/2014, Revision: Final Version</b></p>	<p>Page 32 of 37</p>

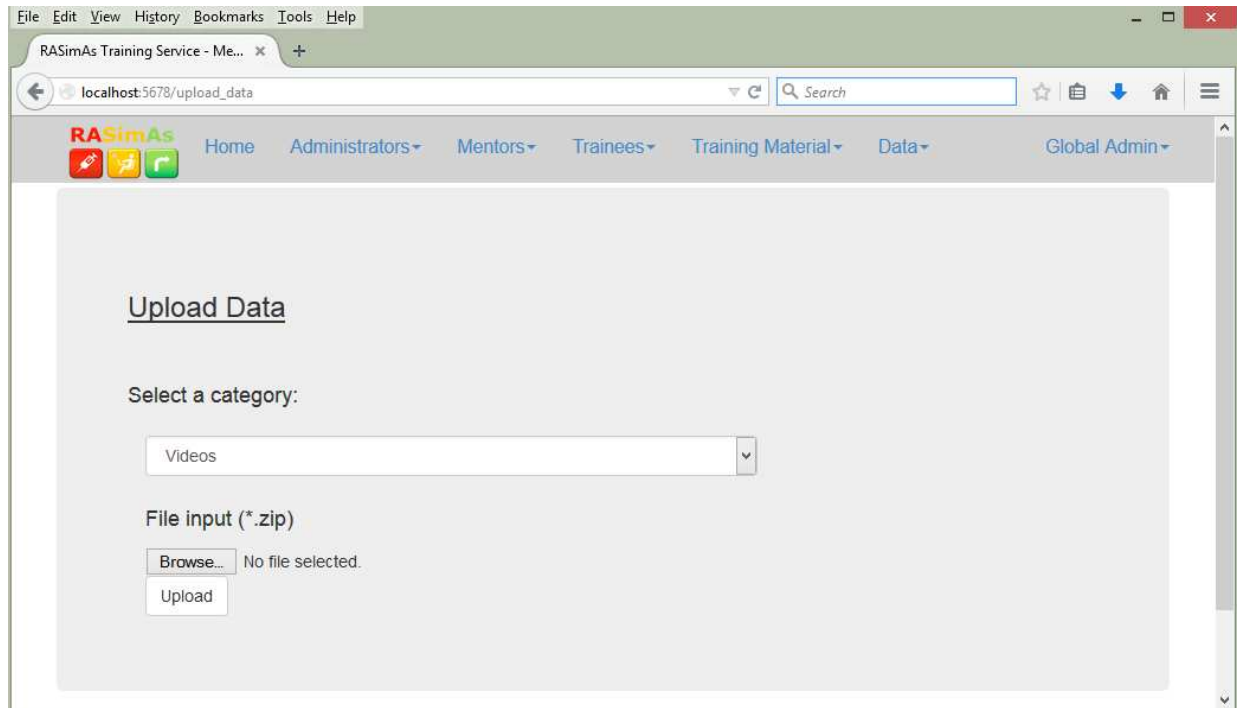


**Figure 19 – Global Admin “Home” Menu**

## 7.2.2 Functionalities

Global service includes all the functionalities of the local service except *Registration*. In addition, in the case of the *Global Admin*, one more functionality is added which is related to the system’s files updates (Figure 20). Each time there is a new or an updated file related to software or training material, the *Global Admin* uploads it to the database by selecting the relevant data category which is then synchronized with the local services. If the uploaded file already exists, then the older one is deleted from the database and is replaced with the new one.

	<h1 style="color: red; margin: 0;">RASimAs</h1> <p style="margin: 0;">FP7-ICT-2013-10 - 5.2 - Virtual Physiological Human</p>	
<p>Project No. 610425</p>	<p><b>Deliverable Report</b> <b>D2.4, 31/12/2014, Revision: Final Version</b></p>	<p>Page 33 of 37</p>



**Figure 20 – System Files Update Functionality**

## 8 Application Programming Interfaces (APIs)




In this section, the functionalities that may provide authorization for an external service to access information of the currently developed platform, are presented. The data retrieved will be mainly used by the course-ware and the synchronization system. In this particular service it is assumed that the *IP* address is *localhost*, the port is *1234*, and the communication protocol is in Javascript Object Notation (*JSON*) format. All presented *APIs* are accessed only through the local services.

### 8.1 Get All Metrics

This current *API* provides all types of metrics used as defined in Deliverable 2.1. Access to these metrics is achieved using the following *URL* [http://localhost:1234/api/all\\_metrics](http://localhost:1234/api/all_metrics).

### 8.2 User's Credentials Authentication

The current *API* checks whether the provided username and password do belong to a registered user. Access to this information is provided with the following *URL* "<http://localhost:1234/api/authentication>" by making a "POST" request with the following *JSON* formatted string: `{"username" : "trainee1@ics.forth.gr",`

	 FP7-ICT-2013-10 - 5.2 - Virtual Physiological Human	
Project No. 610425	<b>Deliverable Report</b> <b>D2.4, 31/12/2014, Revision: Final Version</b>	Page 34 of 37

`"password" : "1234" }` . After the request, the system returns another *JSON* string with the authenticated user's ID `{"user_id" : 5 }` .

### 8.3 Real-Time Metrics Insertion

The current set of *APIs* is dedicated to the real-time metrics insertion from the course-ware into the local database. In comparison to the “Batch Update”, that will follow and inserts all the metrics at once at the end of the session, the real-time one comprises the following *APIs*:

a) Start a new session

The current *API* initiates a new session by receiving the user ID provided by the authentication *API*. Access to this service is provided with the following *URL* “[http://localhost:1234/api/start\\_training\\_session](http://localhost:1234/api/start_training_session)” by making a “POST” request with the following *JSON* string `{"user_id" : 5 }` . After the request, the system returns another *JSON* string with the session's ID: `{"session_id" : 7 }` .

b) Send metrics

The current *API* sends the generated metrics to the current session's ID. Access to this functionality is provided with the following *URL* “<http://localhost:1234/api/metrics>” by making a “POST” request with the following *JSON* string: `{"metric_id" : 15, "session_id" : 7, "score" : 85 }` .

c) End session

The current *API* ends the session as soon as all associated metrics have been sent to the system. Access to this service is provided with the following *URL* “[http://localhost:1234/api/end\\_training\\_session](http://localhost:1234/api/end_training_session)” by making a “POST” request with the following *JSON* string `{"session_id" : 7 }` .

### 8.4 Batch Metrics Insertion

The current *API* is responsible for the insertion of all metrics as soon as a session has ended. Access to this service is provided with the following *URL* “[http://localhost:1234/api/store\\_session](http://localhost:1234/api/store_session)” by making a “POST” request with the following *JSON* string:

```
{
  "session": {
    "start_timedate_utc": "2014-12-1 11:35:33.8",
    "end_timedate_utc": "2014-12-1 12:10:57.1",
    "metrics": [
      {
        "metric_id": 10,
        "score": 72
      },
      {
        "metric_id": 11,
        "score": 73
      },
      {
        "metric_id": 12,
        "score": 74
      }
    ]
  },
  "user_id": 5
}
```

## 8.5 Training Time

The current *API* provides the total training time of a *Trainee*. Access to this service is provided with the following *URL* "[http://localhost:1234/api/training\\_time](http://localhost:1234/api/training_time)" by making a "POST" request with the following *JSON* string `{"user_id": 5}`. After the request, system returns the total training hours and minutes with the following *JSON* string `{"minutes":35, "hours":0}`.

## 8.6 User Level

The current *API* provides the level of the requested user. Access to this service is granted with the following *URL* "[http://localhost:1234/api/user\\_level](http://localhost:1234/api/user_level)" by making a "POST" request with the following *JSON* string `{"user_id": 5}`. After the request, system returns the level's name and ID with the following *JSON* string `{"level_name": "Intermediate", "level_id": 3}`.

## 8.7 User Metrics

The current *API* provides all sessions and associated metrics for the requested user. Access to this service is granted with the following *URL* "[http://localhost:1234/api/user\\_metrics](http://localhost:1234/api/user_metrics)" by making a "POST" request with the following *JSON* string `{"user_id": 5}`. After the request, system returns all sessions and their associated metrics, along with user and the session *UUIDs*, with the following *JSON* string :

```
{
  "sessions": [
    {
      "start_timedate_utc": "2014-12-01 11:35:33.008",
      "end_timedate_utc": "2014-12-01 12:10:57.001",
      "user_id_uuid": "ab80d6f3-5b0f-4e98-95b5-83d2159ed806",
      "modified": "2014-12-18 17:59:28.508",
      "uuid": "af876ffb-5925-4327-b9d5-9ec1c42fdce1",
      "user_metrics": [
        {
          "score": 72,
          "metric_id": 10
        },
        {
          "score": 73,
          "metric_id": 11
        },
        {
          "score": 74,
          "metric_id": 12
        }
      ]
    }
  ]
}
```




## 8.8 Local and Global Services Synchronization

The current *API* is responsible for the synchronization between the central service and all local services. Access to this service, and particularly for the new/updated users' data transfer from the local service to the global one, is provided with the following *URL* "[http://localhost:1234/api/users\\_sync\\_with\\_global\\_service](http://localhost:1234/api/users_sync_with_global_service)". On the other hand, for the new/updated system files transfer from the central service to the local ones, the following *URL* is employed "[http://localhost:1234/api/files\\_sync\\_with\\_global\\_service](http://localhost:1234/api/files_sync_with_global_service)".

## 9 Conclusions

In the current document we have described the software components for the *Integrated Platform* of the RASimAs project. The building blocks for the RASimAs system core are:

- I. A Relational Database for the data storage/retrieval, and user management support (e.g. credentials, access rights, etc.).

	 FP7-ICT-2013-10 - 5.2 - Virtual Physiological Human	
Project No. 610425	<b>Deliverable Report</b> <b>D2.4, 31/12/2014, Revision: Final Version</b>	Page 37 of 37

## II. A Web Application for accessing the service via the World Wide Web.

It comprises two systems, a local one, where the training system is installed, and a global one that acts as a central server that synchronizes periodically with all the local ones. Its primary aim is to complement the course-ware system, developed by SenseGraphics, by providing an *API* for users (credentials, profile, metrics, etc.) and system (software updates, training material, etc.) data storage/retrieval. In the case of synchronization, in one hand, local systems upload users (profile, metrics, etc.) data to the central one for making them accessible outside the training environments, and in the other hand download from that system's updates (software, training material, etc.). It encompasses its own interface for accessing, locally and globally, personal information and training material via a browser after a secure login with their personal credentials. The service has the capacity of uploading and storing large files (more than one gigabyte) and it is fully implemented by free and open-source software.

This is a preliminary version of the Integrated Platform component fulfilling the present needs of the RASimAs platform. It has met all the requirements as stated in Deliverable 5.1 except the provision of an API related to the "Training Scenarios" which will be available as soon as the relevant information will be available by the other partners of the project. It is meant to be continuously updated with new features/functionalities meeting the future needs of the various RASimAs components.