

# Functional Specifications of Acted Features

---

Development Milestone: **M8**  
Main name: **Administration - Backend**  
Date: 17 August 2016

Current version: **V1.1**  
Current status: Clarification version

Approved by:  
Approval date:

Reviewed by:

## History

Version	Date	By	Comment
V1.0	20 July 2016	Laurent	DRAFT for quotation, with network diagram
V1.1	17 Aug 2016	Laurent/Julien	Clarification of Profile Creation in ADMIN role, and process flow.

## Table of content

1. Profile creation by Acted administrator .....	3
2. Administrator dashboard .....	7
2.1. Administrator user creation .....	7
2.2. Header CSS integration .....	7
2.3. Profile management.....	7
3. Profile edit mode.....	10
3.1. Default picture and performance for NEW profile.....	10
3.2. Picture management.....	10
4. CSS and JS files from design .....	11
4.1. Versioning .....	11
4.2. Documentation and Test.....	11
5. Integrate design for Password recovery .....	11
6. Profile view (and edit) .....	11
7. Update of the design files .....	12
8. Others.....	12
8.1. Bundles names .....	12
8.2. Constraints .....	12
8.3. Code commenting and versioning.....	12
8.4. Database changes or additions .....	13
8.5. Database fixtures & migration for testing purposes .....	13
8.6. Data migration from now on.....	13
8.7. Test materials.....	13
8.8. Documentation .....	14

## List of figures

Figure 1: Deployment diagram.....	9
-----------------------------------	---

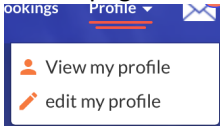
## 1. Profile creation by Acted administrator

The **main** purpose of this feature is to allow Acted employees and site administrators to create and manage profiles on behalf of an Artist or Client.

We will “go-live” with that version and start creating real Artist database, and slowly discard (DELETE) Fake profiles from the database.

Create Artist account on behalf of Artist as well as Client without him being involved in Validation. These profiles are initially created under ACTED management (By an Acted employee that has the ADMIN Role, granted by the Super-ADMIN), until control is passed over to a real user. A real user is primarily an ARTIST, but could also be a CLIENT.

### STEP 1 of the process flow:

- New user creation (ARTIST or CLIENT) shall only be done under ADMIN role. The newly created user will have no Email entered into his profile (empty Email/Password), It will be uniquely identified by the database ID.
    - The reason for which we do not want or need to email address in the Step 1 of the process, is most of the time because Artists may have multiple Email addresses for different purposes, and we would not necessarily know the one he prefers to use.
    - In addition, this will allow us to build the Email change functionality as described later in that document.
  - Prepare backend management table (similar to /profile/list) for selecting / searching any profiles including the ones managed by ACTED Administrators.
    - We can manage the profile by Clicking on “EDIT” button. It shall open a new Window.
    - We can also access ARTIST and CLIENT profiles in **Public View** mode from the Administration page by clicking the “VIEW” button. It shall open a new browser Window.
    - When EDIT or VIEW is clicked, this will open the relevant Profile (In Edit or View mode).
    - Once the Profile is opened in a separate window, the profile menu is available, so that the Administrator user can select View or Edit without returning to the Admin page.
- A screenshot of a user profile menu. The menu is a small white box with a blue border. It contains two items: 'View my profile' with a person icon and 'edit my profile' with a pencil icon. The menu is positioned over a dark background, which is the Admin page mentioned in the text.
- The background in Admin role is **Black**
- Acted employees will create the profile on behalf of the artist using the same process as today (Profile-Edit), and once internally validated will pass it to the Artist (or Client).
  - This implies 2 differences, as a consequence of the above definition. The Email address is no longer a MANDATORY field in the Registration Form (Step2), and the Phone number is also no longer a MANDATORY field.

- Regarding the Phone number: this was not explicitly specified earlier, but was considered obvious. If we do not have the Email address of the artist, we also probably do not have his phone number.

○

## STEP 2 of the process flow:

**The profile is now ready, and the Marketing team of ACTED has decided what email address to use for that particular user. We can now start the handover process and INVITE the user to take over control of his profile.**

- The user email being now known & the profile being ready for handover (validated internally) then the handover process can start.
  - the Email is stored in DB as the user email, and a temporary RANDOM ONE-TIME-USE password is generated. The Email is entered manually, and can be Created and Changed. When the email is SAVED, then it must be verified that another user does not already use this email. An error message must be displayed, and email box content erased.
  - Once the email is entered and validated, the Acted administrator can decide to activate the user account. As per today's logic, the profile becomes immediately Public, and therefore the url for accessing the public profile can be prepared and will be added into the invitation email . (This will be changed in a near future). We will call it **Profile\_URL** here.
  - an INVITATION email is sent to Artist, using a template email, similar to earlier (the email template text is bellow). To send that Email, the Acted administrator needs to click on the SEND INVITATION Button of that user in the Administration panel.
  - If the User account has not been validated, then the Invitation button is Greyed out, and not active.
  - A second url , called **Invitation\_URL** is prepared. This URL contains a token that has a Validation period. We will call it Invitation\_period. The token is also having the RANDON password defined above. The invitation\_period allows to compute the expiration date-time of the invitation, and will be stored into the database for future use. We would like to show a Green INVITATION button as long as the period did not expire, and Red BUTTON when it has expired. This will give the Admin user to clearly and quickly identify users that have not taken any action after the invitation email has been send.
  - As part of the email, the database Email of the user that has been entered into the database will be displayed to user in the email.
  - Using the Profile\_URL, the artist can preview his profile (in Public view as already said earlier)
  - the Artist can login within xx<sup>(\*)</sup> hours (invitation\_period) by clicking the **Invitation\_url** attached in the email.
  - This then opens a page using the Recovery password layout BUT WITH DIFFERENT TITLE.  
The Title shall be "PASSWORD CREATION" instead of "PASSWORD Recovery"
  - Then the first thing he needs to do is to enter his own (new) password, witch is in fact replacing the RANDOM password in the database. Once the user has entered a new password of his choice, we are then in a normal situation where

a new email is send for user account validation. By Normal situation we mean that this is the same as when a user is registering by himself using the Registration process.

- This password confirmation email contains a link with the activation token as per already existing process. We will call it Password\_validation\_URL.
  - In Addition, we also need to create an expiration period for the password\_verification URL, defined by a second parameter.
  - If the expiration period is passed, then the user will have to use the already existing password Recovery procedure for asking for a password change.

We assume the email of the Artist or Client we use is valid.

**Email changing flow must be provisioned, and usual verification is done before validation the field. This is a real email (at least [x@y.z](#) format), and this is not already assigned to an other user**

(\*) Invitation\_period email link validity defined by a parameter in configurations files (in hours)

i.e 48 hours, parameter value = 48

password validity defined by an other parameter in configurations files (in hours)

i.e 24 hours, parameter value = 24

From the Administration dashboard (see bellow), we can re-issue the Email (with new link, and new validity period), as well as re-issue a new password.

#### **Pagination helper:**

As we need to anticipate 1000's of users, the pagination Tab must be user friendly:

- Goto first page, Goto Last page
- At least 10 pages navigation (as we are on big screen)

#### **Auto-Refresh:**

As it was obvious to us, we did not specify earlier, but: Every changes in the DropDown field, Activation check box, Email address saving must generate automatically a Table refresh of the Admin / User list. This is important as the BUTTONS Activation/Visibility/Color will change accordingly.

Email template:

**Dear [ first\_name]**

We are delighted to inform you that our team has created a temporary **Acted** profile on your behalf. Acted is a brand new UK marketplace for entertainment with hundreds of artists already pre-signed up. We are going live soon, and all our first signups will get amazing rewards.

Don't worry; Acted is a free service for you. We are building amazing free tools for you to manage your bookings, payments, contracts, paperwork and much more.

Oh, did I mention our website will bring you many direct leads & bookings? As an entertainment marketplace, our clients are your clients. Event managers, wedding planners, anybody can contact you directly. This is the value we are bringing to you.

Please take a couple of minutes to review your profile by clicking the following [link](#) [Profile\_URL] You can complete your registration and start editing your account and profile by clicking the following link: [Invitation\_URL].

This link is valid until [expiration date/time]. Shall you want a new invitation link if the link has expired, please contact us per email and we will reinitialize the process.

Your login name will be: **[user\_email\_address]**

You will have to enter a new password at the first connection.

In case you do not want to appear on Acted (we would be really, really sad), you can just reply to this email and we will remove your profile permanently.

Any questions about Acted and how it works? Please have a look here: [\[website\\_url/#how-it-works\]](#)

Please do not hesitate to contact us per email (marketing (at) acted.co ) and share your feedback. Always happy to have a chat and understand how we can improve your experience on Acted.

Warm regards,

[JULIEN]

**For other email related specifications, please refer to M5 specification Page 9**

## 2. Administrator dashboard

Continuing from early-stage M7 based administration dashboard, and by re-using already developed method, we need to enhance with additional functionalities

### 2.1. Administrator user creation

This must be done only by a **console app**. Only Linux users can do it. We will restrict the rights of the php file at the Linux level

We need a CRUD model for creating and managing users with Admin role.

- Parameters to be passed shall be: "email", "initial password"

The user will be automatically validated, without the need to send email, and the Password token created and stored into the DB, as any other user

- We may change this in the future, once security assessments will be done

We need to be able to change the email for the same user ID

We need to be able to change password

### 2.2. Header CSS integration

We need to integrate the Header CSS into the Administrator main page

In order to differentiate from other regular users, please use a **Black** background instead of the blue one.

- If design change is required, we will ask the Design team to do a specific CSS header file.
- In the mean-time override the background color

We won't need to integrate other css, UNLESS you think it can speedup development time

This Black BACK Background of the header is for all activities done in ADMIN ROLE:

- Admin Dashboard

- Home Page when user is having Admin role

- Profile Edit / View when initiated by an Admin ROLE based user

### 2.3. Profile management

By keeping a similar Search method from Spotlight and Recommended functionalities from M7 (name or slug):

Add the possibility to search by user ID

Add the possibility to Delete a User and clean-up all related records in linked tables.

- This is a definitive Deletion in the database.
- Therefore Double confirmation is required
- It shall also delete all related Pictures, and thumbnail Cached data for all the sizes.

- If an artist was in Spotlight or Recommended list, it has to be removed

Possibility to mark (and display) an Artist or Client (user in fact) as “FAKE”.

- This requires new column(s) in DB
  - In user table
  - And also in Client or Artist table (to speed up SQL queries)
- You will add new column in other tables whenever it is necessary to simplify coding, and also Query speed.
- If an artist was in Spotlight or Recommended list, it has to be removed when marked as “Fake”

Fake users are the ones generated by the Fixture/Migration, as well as the ones created for testing and validation

We also need to display if a user is Artist or Client. (Admin also) and have a Filter for that

- Migration shall mark user’s created by Fixtures (up to id-300) as fake
- User’s created manually during testing **shall not be marked as Fake** by Migration. This will be done manually

As we want to manage only one MariaDB database at the moment, we can create different front-ends for testing or for staging.

Over time, the FAKE’s shall disappear completely and be replaced by real users. But we need to keep the possibility for internal use only to use the fake users and their related data’s.

For that purpose, a “use\_fake” variable needs to be defined into the configuration file (app/config/parameters.yml), so that we can easily switch the front-end from one to another without changes in the code.

#### 2.4. Deployment plan

We will have multiple front-end servers (Apache + PHP + Symfony application) that can connect to the same MariaDB database. The MariaDB will be a cluster at some point, but we hope this will be transparent to the Symfony application

On EACH application server, when use\_fake is set to TRUE

- All user, profiles, client will be enabled

On EACH application server, when use\_fake is set to FALSE

- Only non Fake users, profiles, and client are in use (valid)

***This does not exclude the fact that we may have copies of the database for other testing purposes.***

Deployment Overview, Short term: the diagram bellow indicates our plan for deployment during the summer.



## Development, Staging, Production overview

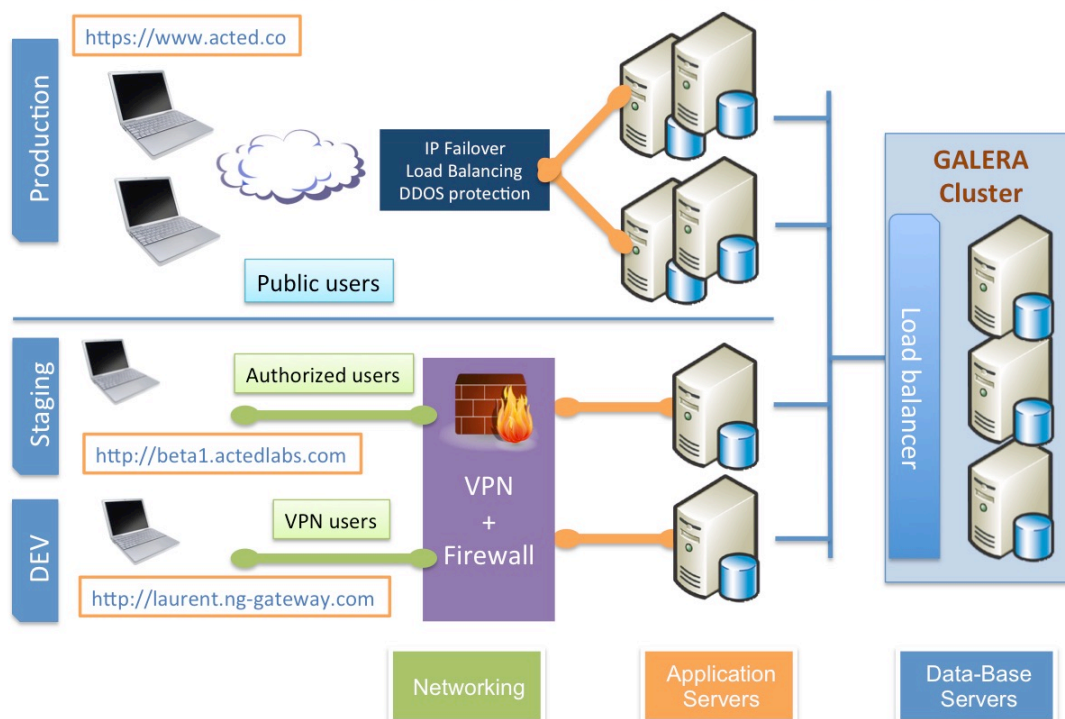


Figure 1: Deployment diagram

**Staging** and **Development** servers will not have public access. We will either use VPN connection or Linux based PEM, or a combination of both.

Development server:

This is the one currently in use (<http://laurent.ng-gateway.com>).

In that server settings we will use the “use\_fake” variable either TRUE or FALSE in order to validate and verify the implementation of the FAKE flags.

Staging server – next 2 months use:

This is a new server to be provisioned during M8 (<http://beta1.actedlabs.com>).

In that server settings we will use the “use\_fake” variable to TRUE. This allows to sign-up Artist.

Every time a new artist is fully validated, we will look at a corresponding FAKE artist with the same sub-category and DELETE this profile.

This will allow to gradually replace Fake profiles by real ones, and keep returning enough number of profiles in the search results.

We do not expected any quote request related issues, as no client other than our testing account shall sign-up in the next 2 months

Production server:

This is a new server to be provisioned (<https://www.acted.co>).

We will enable SSL/TLS

In that server settings we will use the “use\_fake” variable to FALSE. There shall be only a limited amount of Fake profile left.

For any test with “use\_fake” = TRUE, we will use a staging or development server.

The Database may keep some FAKE user, profile, client forever

### 3. Profile edit mode

We need to keep improving the user experience and usability of the profile page, in particular in EDIT mode.

#### 3.1. Default picture and performance for NEW profile

When a NEW ARTIST profile is created, the default **Main profile picture** shall be loaded from the backend, indicating to the user that he needs to load a picture.

- Once a picture is loaded (successfully), the default won't be used anymore.
- The default picture will be delivered separately.

In order to indicate to the user (Artist) what to do next, we need to show a **first performance block**, empty, and marked as DRAFT.

We will also provide a default sample picture in edit mode so the artist knows what to do

- The first sample picture will show a Photo artwork
- The second will Show Photo/Video artwork

#### 3.2. Picture management

##### Optional feature (quote separately)

We want to enhance the user experience of the resize & cropping tool. We want a user to be able to re-use his already existing picture if he wants to change Sizing and Cropping.

- This shall be made possible for ALL PHOTOS (profile, avatar, performance, and photos)

For that we need keep on server the **original** version of the pictures in **high resolution** so that the user can re-edit the currently stored version directly without having to reload them.

- This implies that rendering is always done using thumbnails version.
  - Please re-visit the implementation to ensure that rendering is ALWAYS using the pictures in the cache
- The exception is when a new profile is created.
  - In that case, no image from the user is existing

If the user is selecting a new picture, then this new one will replace the previous one

- This will become the new master

*This also has an advantage on mobile devices where there is no access to file system (IOS devices); then at least, the user can resize and crop the current picture(s).*

## 4. CSS and JS files from design

Finishing automatic integration process of new CSS/JS from mockup repository.

### 4.1. Versioning

You need to integrate with the TAG version 1.3 or later of the mockup repository. During the development of the M8, you may receive newer version until last day with adjustments.

It needs to integrate smoothly with no re-coding of any part of the application if it relates to existing CSS / JS features.

### 4.2. Documentation and Test

Please document any change that needs to be back-ported into the Mockup repository.

**This is very important, and no compromise can be done here.**

## 5. Integrate design for Password recovery

The Mockup includes the password recovery page design. This needs to be integrated.

## 6. Profile view (and edit)

Prices informations: remove “\$” sign or “EUR” text and put “on-request” instead of value

- For profile prices
- For performance block prices

Prices management and generation will be done as part of next Milestone.

## 7. Update of the design files

**As said above, New design files** will be delivered during the development of that milestone, and need to be integrated.

This includes some resizing, position adjustments, new version of logs, images, etc...

This requires continuous integration.

We are aware that this is not covering any HTML related work.

## 8. Others

### 8.1. Bundles names

When applicable, a specific bundle name must be defined, unless it is too much interacting with the rest of the application.

### 8.2. Constraints

No commercial or licensed frameworks or API should be used to achieve this bundle. If any is required, this must first be approved by one of the directors Acted Labs Limited

Only open-source codes or frameworks could be integrated in the final developments.

For Open source components, the terms and conditions of the license must be listed in the final documentation in order to ensure GPL compliancy. The list must include the previously designed and already implemented components from Millstones M1 to M7

### 8.3. Code commenting and versioning

In each source code file (PHP mainly but no limited to these files) a proper header file must be added. The header is made of two distinct parts:

Part 1: In the header it shall mention the Acted Labs Limited copyright

Part 2: Version number in the form of Vx.y.z

Where

X is major (new functionality, incompatibility with previous major version)

Y is minor (new functionalities or enhancement but backward compatibility ensured)

Z is patch (bug fixes)

Part 1 will be the same for all files, and we will create an MD5 signature of that section for our own use. We will provide the master text during the development.

Each major part of the code including every function must be documented. It needs to explain the functionality and any necessary details for comprehension.

Things that are self-explanatory do not need to be documented but this is an exception rather than the rule

#### 8.4. Database changes or additions

Every required change into the database structure, index, data sizing must be discussed with Acted, and approved by Acted. The same is true for any update or upgrade of the DB version or any related component, library, connector, etc...

The main reason is that there are future expected functionalities that will be developed, and we want to start understanding the total sizing of the solution, replication requirements, possible optimisations such as using SSD disks, etc...

We are also happy to discuss any possible improvements when possible scaling issues are foreseen during the development.

#### 8.5. Database fixtures & migration for testing purposes

Sufficient amount of data must be available into the database (via fixtures or migration) in order to ensure efficient testing of that feature.

The additional data entered during testing shall be kept for future use when appropriate, so that it can be re-played after a database DROP.

Any changes to the main database are now made with migrations.

**We may be able to create a new Database from scratch, and replay all the fixtures. Therefore, the fixtures related code must be maintained, and work with the current Schema at any time, if migration alone are not sufficient.**

#### 8.6. Data migration from now on

As previously implemented, we will keep all the data entered during testing, and therefore data migration must be planned accordingly.

#### 8.7. Test materials

Prepare testing scenarios with the 2 values of the FAKE variable

## 8.8. Documentation

A properly formatted Word document will be required to describe all the useful info about the features of this specification, to explain, if it applies:

- What is the main goal of the feature?
- What are the feature inputs?
- What are the feature outputs?
- What are the main business rules implemented?
- Define a detailed “How it works” schema.
- Documents the features that are not implemented here, but part of a “next” milestone
- Document for further reference the “prepared for future use” features and details. In particular (but not limited to) the LOCALE related ones.
- Which bundle handles each of the feature(s)?
- What are the main classes?
- What are the existing services used, and services created in that feature?
- What are the Twig templates?
- What are the add-on bundles you installed, including licensing information that applies?
- What are the commands to run to install the feature?
- What are the commands to run to test the feature?
- How to test the feature?
- What are the deployment server requirements? In particular any specific (minimum version) used during development and validation
- GIT related information’s. In particular, for testing purposes and long term tracking, please indicate key GIT tags (and/or Branches) that relates to fully functional deliverables.
- Database requirement explained. In particular Indexes, foreseen performance issues or challenges