*What is the main goal of the feature?*

The main goal of milestone 5 was creating of new event and creating quote requests. This enables the message and enquiries functionalities into the Artist Dashboard.

*What are the feature inputs?*

* The design mockup of the search page, under the “quote.html” file, in the Mockup repo.
* Acted MariaDB database.
* Updated mockup repository for the Retina screens integration, and recent adjustments.

*What are the feature outputs?*

Added ability for a Client to make quote request to an Artist. When creating a quote request, the Client invites the selected artist to prepare a quote for this event. At this stage we are creating a new offer to the Artist.

1. added new parameters to eventManager.php  
   for sending email message: app/config/services.yml
2. added methods for notifying Artist after created event and offer src/Acted/LegalDocsBundle/Model/EventsManager.php
3. added call methods from service for send email from controller - src/Acted/LegalDocsBundle/Controller/EventsController.php
4. change relation of Offer-> Performance from ManyToOne to ManyToMany
5. add relation Event->RefVenueType and Event->RefEventType ManyToOne
6. Change fields in EventOffer price, currencyId, depositValue, depositType, paymentTerms now can be NULL
7. add template for notify artist after add event and offer. Template is for the email notification. Screenshot available at: <http://joxi.ru/YmENVZKsZVXMG2>

src/Acted/LegalDocsBundle/Resources/views/Email/create\_event\_notify.html.twig

src/Acted/LegalDocsBundle/Resources/views/Email/new\_message\_notify.html.twig  
Change all test fixtures accordingly changes DB

src/Acted/LegalDocsBundle/ [DataFixtures](https://bitbucket.org/actedDev/acted/src/c44e25dafcad/src/Acted/LegalDocsBundle/DataFixtures/?at=master) / [ORM](https://bitbucket.org/actedDev/acted/src/c44e25dafcad/src/Acted/LegalDocsBundle/DataFixtures/ORM/?at=master) / LoadArtistData.php

# [src](https://bitbucket.org/actedDev/acted/src/116f0c0b8c1b/src/?at=check-place-media)/[Acted](https://bitbucket.org/actedDev/acted/src/116f0c0b8c1b/src/Acted/?at=check-place-media)/[LegalDocsBundle](https://bitbucket.org/actedDev/acted/src/116f0c0b8c1b/src/Acted/LegalDocsBundle/?at=check-place-media)/[DataFixtures](https://bitbucket.org/actedDev/acted/src/116f0c0b8c1b/src/Acted/LegalDocsBundle/DataFixtures/?at=check-place-media)/[ORM](https://bitbucket.org/actedDev/acted/src/116f0c0b8c1b/src/Acted/LegalDocsBundle/DataFixtures/ORM/?at=check-place-media)/LoadEventData.php

Fixtures - is test data what add to db, this data include artist, performance, city, country or other information what exist in db (this data we have added to start migration)

1. Update method for create Event, Offer, EventOffer accordingly changes DB

src/Acted/LegalDocsBundle/Model/EventsManager.php

methods:

createEvent()

createEventOffer()

createOffer()

1. add ability for the Artist to reject offer, the corresponding offer/event is marked as rejected.

Now, client not notify after create Offer for artist. We not see in docs describe this feathers

1. add endDate for event according docs

endDate = startDate + 1day

1. add new table for saving chat info, add manager for working with chat

src/Acted/LegalDocsBundle/Model/ChatManager.php

src/Acted/LegalDocsBundle/Entity/ChatRoom.php

src/Acted/LegalDocsBundle/Resources/config/doctrine/ChatRoom.orm.yml

src/Acted/LegalDocsBundle/Resources/config/serializer/Entity.ChatRoom.yml

1. add relation ChatRoom->Event, ChatRoom->User, ChatRoom->Message
2. add ability to add a first message after creation of new event and corresponding offer

After create new EventOffer messages sending only for artist (in docs not info about notify client)

1. add ability for get artist data by slug in the quote form  
   It meants, get info about exist event for curent login artist (if event exist, and artist auth)
2. add check for existing event in form  
   If client chose previously created event we send event id and event data with additional information for offer(performances and comment). This is both backend and front-end feature.
3. add new manager for work with chatroom and messages in a some room

src/Acted/LegalDocsBundle/Model/ChatManager.php  
 createChat(), newMessage(). ChatRoom is place where artist and client can discus with each other.

1. create methods for get chatRooms and messages in a some room
2. add answer after login user(return id and role), change fixtures for user and roles
3. add relation Message->User for receiverUser and senderUser
4. change in Db - add relation ChatRoom->Offer OneToOne
5. change test fixtures for Offer, add fixtures for EventOffer, ChatRoom, Message
6. add ability for get list chat rooms by userId

src/Acted/LegalDocsBundle/Controller/ChatRoomController.php::getChatRoomListAction()

1. Add method for get all message by ChatIdAdd method for get all message by ChatId  
   src/Acted/LegalDocsBundle/Controller/ChatRoomController.php::getAllMessageByFilterAction()
2. Add check exist offer for performances of artist

src/Acted/LegalDocsBundle/Controller/EventController.php::checkExistOfferAction()

1. Add ability for show one message from chat\_room

src/Acted/LegalDocsBundle/Entity/ChatRoom.php::getLastMessage()

1. Add new template - src/Acted/LegalDocsBundle/Resources/views/ChatRoom/message.html.twig

*What are the main business rules implemented?*

Added ability to create events and make quote/offers to artists

*Other informations from previous milestones*

**1: Images cropping and thumbnails creation.**

New mechanism and workflow for uploading media, in particular Photos has been implemented. The goal is to ensure that photos of any size can be uploaded, while storage and file size is optimized depending on the usage of the resulting images.

When user selects image we download it and make canvas from user image. After this user has ability to move, zoom image.   
After user finish cropping and clicks upload image we get part of image that in viewport and make base64 image of that part. Then we send this base64 encoded image to backend where save it in original size.

To reduce weight of pages and improve site speed we implemented thumbnails and save images in three sizes.

0 Thumbnail generate in back-end side after user upload image. For generating we use LipImageBundle

1 small: width:270px;(we use them on search page and as thumbnails in media section)

2 medium: width:481px(we use this pictures on performance blocks);

3 big: width:1080px(this images we use in media section and that places where we need big images)

4 original. In this case we just save images in original size.

**2: Migrations**

For migration we use DoctrineMigrationBundle

1. Was added start migration including all test data from fixtures
2. For now we don’t need use commands like a  
   ***php app/console doctrine:schema:update --force***

***php app/console doctrine:fixtures:load***

We should use only command

***php app/console doctrine:migration:migrate***

**3: User login & Password**

Class FormLoginAuthenticator implements Guard interface.

Guard is the new authentication sub-system. FormLoginAuthenticator extract email/password params from Request and checks user’s credentials.

Class ArtistVoter is Voter. It provides method for checking permissions of authenticated user.

We use the bcrypt for passwords storing. After registration, the system generates random string, sends it inside verification email and sets confirmation\_token field. When user clicks the link, the system checks the token and sets activation flag for this user. We use PHPSESSID cookie to identify authenticated user.

Docs <http://symfony.com/doc/current/cookbook/security/guard-authentication.html>

SSL - is server configuration, but also need add some change to app/security.yml file

*Define a quick “How it works” schema*

Customer clicks on “request quote” button, then quote request form appears. In this form customer can describe his event and make offer to specific artist. Moreover Client can choose already exist event. On the last step if client is not logged in we close quote request form and open registration modal. After registration we will send quote request. If user was already logged in we just send quote request to artist. After creating of quote request we are creating essences “Event, Offer, EventOffer, ChatRoom, Message”.

*Which bundle handles the feature?*

ActedLegalDocsBundle contains all application logic.

*What are the main classes?*

src/Acted/LegalDocsBundle/Controller/EventsController.php - action for create QuoteRequest

src/Acted/LegalDocsBundle/Model/EventsManager.php - service for work with Event, Offer, EventOffer

src/Acted/LegalDocsBundle/Model/ChatManager.php - service for work with chat room

Acted/LegalDocsBundle/Popo/EventOfferData - class for work with Quote Request form

src/Acted/LegalDocsBundle/Controller/ChatRoomController.php - controller for work with chat rooms

*What are the services?*

app.event.manager - service for work with Event, Offer, EventOffer

app.chat.manager - service for work with chat room

*What are the Twig templates?*

ActedLegalDocsBundle:ChatRoom:chat\_room.html.twig

ActedLegalDocsBundle:ChatRoom:list.html.twig

ActedLegalDocsBundle:Email:create\_event\_notify.html.twig

ActedLegalDocsBundle:Email:new\_message\_notify.html.twig

*What are the add-on bundle you installed?*

* nothing to install for M5 specific features
* Installation of extensions are required for the Picture/Photo management (described above). This was made as part of M4

*What are the commands to run to install the feature?*

|  |  |  |
| --- | --- | --- |
| **COMMAND** | **Optional** | **Comment** |
| php app/console do:dat:dr --force |  | Erase database |
| php app/console do:dat:cr |  | Create database |
| php app/console do:mi:mi |  | Start Migration (different than previous milestones) |
| rm app/logs/\*.log |  | Erase earlier LOG if no longer needed |
| php app/console ca:cl --env=dev |  | For Development and Testing only |
| php app/console ca:cl --env=prod |  | For Production only |

*What are the commands to run to test the feature?*

We do not have tests for this feature.

*How to test the feature?  
click on request quote button and check creating quote for artist (login and not login user).*