**Marketing Camps FUP**Сервис состоит из двух частей: сервиса и API для вызова удаленных методов сервиса

## Развертывание сервиса

Требования

* jdk1.7
* Tomcat8
* Cервер RabbitMQ
* Сервер MySQL

### Настройка

Переименовывать marketing-camps-fup.war в marketing-camps-fup.zip

Заполннить файлы ./WEB-INF/classes/META-INF/staging.properties и ./WEB-INF/classes/META-INF/production.properties внутри архива данными для staging и production контуров.

Заполнить параметры:

dbUrl={url до базы данных}

dbUser={аккаунт базы данных с правами создавать таблицы, создавать, читать, редактировать, удалять записи в таблицах}

dbPwd={пароль аккаунта базы данных}

rabbit.hostname={адрес сервера RabbitMQ}

rabbit.username={аккаунт для доступа к серверу RabbitMQ}

rabbit.password={пароль для доступа к серверу RabbitMQ}

ui.username={имя пользователя UI сервиса}

ui.password={пароль пользователя UI сервиса}

Переимновать marketing-camps-fup.zip в в marketing-camps-fup.war

При запуске tomacat использовать флаг -Dspring.profiles.active=staging для staging контура и -Dspring.profiles.active=production для production контура.

Произвести деплой marketing-camps-fup.war в Tomcat, в контекст marketing-camps-fup

Доступна следующая точка входа в UI: /marketing-camps-fup/home , запароленная параметрами ui.username и ui.password

Доступны следующие endpoint'ы для проверки жизнеспособности сервиса:

 /marketing-camps-fup/ok - быстрая проверка того, что сервис отвечает в принципе

 /marketing-camps-fup/okfull - проверка наличия доступа сервиса в базу, предполагается проверка не чаще чем раз в минуту.

## API

Сервис, который захочет получить список текстов созданных маркетинговых рассылок, должен использовать API mcfup-api-0.03.jar и реализовать java-конфигурацию вида:

@Configuration

public class TestConfig {

 @Bean

 public ConnectionFactory rabbitConnectionFactory() {

 CachingConnectionFactory connectionFactory = new CachingConnectionFactory(rabbitHost);

 connectionFactory.setUsername(rabbitUserName);

 connectionFactory.setPassword(rabbitPassword);

 return connectionFactory;

 }

 @Bean

 public AmqpAdmin rabbitAdmin() {

 return new RabbitAdmin(rabbitConnectionFactory());

 }

 @Bean

 public MessageConverter messageConverter() {

 return new Jackson2JsonMessageConverter();

 }

 @Bean

 public AmqpTemplate rabbitTemplate() {

 RabbitTemplate rabbitTemplate = new RabbitTemplate(rabbitConnectionFactory());

 rabbitTemplate.setMessageConverter(messageConverter());

 rabbitTemplate.setReplyTimeout(60\*1000);

 return rabbitTemplate;

 }

 @Bean

 public ApiProvider apiProvider() {

 ApiProvider apiProvider = new ApiProvider();

 apiProvider.setExchange(FupApi.EXCHANGE\_NAME);

 return apiProvider;

 }

 @Bean

 public FupApi fupApi() {

 return apiProvider().getApi(FupApi.class);

 }

}

Вызов удаленных методов :

public class TestConsumer {

 @Autowired private FupApi fupApi;

 public void myMethod() {

 Long userId = 1; int page=0; int size=10;

 List<Campaign> list = fupApi.listCampaigns(userId, page, size);

 long campaignId = 1;

 CampaignData campaign = fupApi.findCampaign(campaignId);

 long campaignStepId = 1;

 StepData step = fupApi.getCampaignStep(campaignStepId);

 }

}