|  |  |
| --- | --- |
|  | Task AngularJS Basic |
| EPMR-TCSR |

ИСТОРИЯ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Версия | Описание изменений | Автор | Дата | Утверждено   |  |  | | --- | --- | | Имя | Дата | | |
| <1.0> | Initial version | Михаил Семичев | <7-Aug-2014> |  |  |
| <1.1> | Divided by steps | Анастасия Ефремова | <7-Sep-2014> |  |  |
| <1.2> | Added use mocks | Андрей Егоров | <29-Jan-2015> |  |  |

СВЯЗАННЫЕ ДОКУМЕНТЫ

|  |  |
| --- | --- |
| Номер | Название документа |
|  |  |
| 1 |  |

Практическое задание

Задание 1

Создать SPA (одностраничный сайт с хеш-навигацией) используя AngularJS фреймворк.

Необходимые инструменты: AngularJs, Bootstrap, AngularUI-Bootstrap

Весь бэкенд строится на эмулировании выполнения запросов и ответов внутри angular, с помощью модуля ngMock сервис $httpBackend.

Этапы выполнения:

1. Верстка страниц со статическим контентом.

Страницы:

* Login – страница входа
* Courses – страница список курсов
* Add\Edit Course – страница добавления/редактирования профиля курса

Максимально использовать стили Bootstrap.

1. Настроить роутинг между страницами, используя ngRoute.

Соответствия:

* Login - # /login
* Courses - #/courses
* Add\Edit Course
  + Add - #/courses/new
  + Edit - #/courses/12 (12 – это id курса)

При вводе не существующего пути, если пользователь вошел в систему перенаправлять его на страницу courses, если не вошел, то на страницу login.

1. Реализация страницы Login

Валидация полей по потере фокуса. Поле выделяется красным, и справа выводится надпись «\* введите».

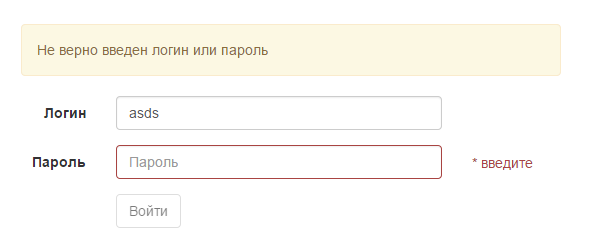


Если хотя бы одно поле не заполнено кнопка «Войти» заблокирована.

Поле «Логин» может содержать, только латинские буквы.

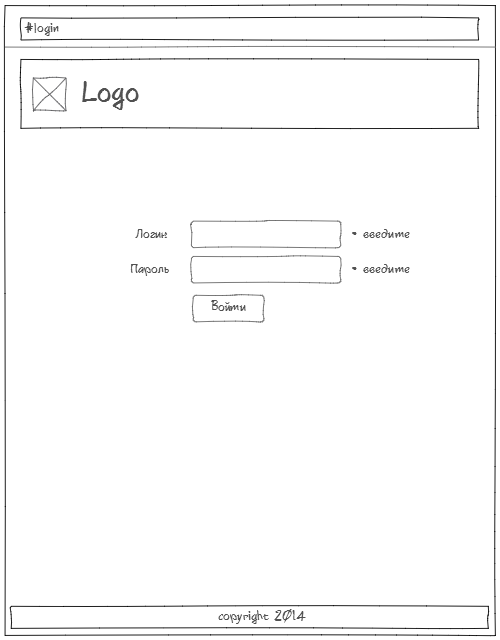
Поле «Пароль» может содержать, латинские буквы и цифры.

Если пользователь ввел неправильный логин или пароль, то вверху выводить сообщение «Не правильный логин или пароль» и обнулять поле «пароль».



После заполнения полей и входа пользователя на сайт, его перенаправляет на страницу courses.

Для авторизации использовать отправку запроса через сервис $http. В качестве обработки этого запроса настроить перехват с помощью сервиса $httpBackend.



1. Реализация страницы Courses

Страница для вывода списка всех курсов. Работа с курсами осуществляется по принципу работы с REST. Завести сервис Courses для получения/создания/сохранения/удаления курсов, использовать сервис $resource, необходимо подключение модуля ngResource. Получение списка курсов или конкретного курса (get), создание курса(post), обновление полей курса (put), удаление курса (delete), эмулировать через сервис $httpBackend.

Курсы имеют поля: название, дата создания, длительность (в минутах), список авторов.

В шапке страницы отображать «логин» пользователя, также ссылку «выход». При нажатии на «выход», пользователь выходит и его автоматически перекидывает на страницу Login.

При вводе в поле «поиска» и нажатии на кнопку «Найти». Список курсов фильтруется по названию (создать для этого фильтр).

Создать фильтр для отображения длительности курса в формате «1 час 30 мин».

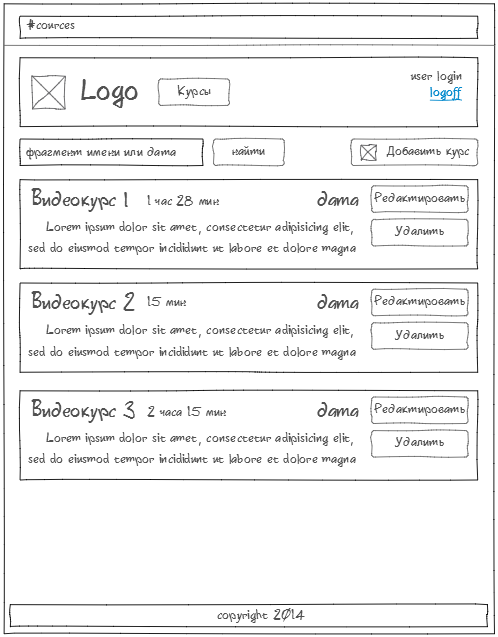
«Дата» должна выводится в формате «дд.мм.гггг». (использовать фильтр date).

При нажатии на кнопку «Добавить курс» перенаправлять пользователя на страницу #/courses/new.

При нажатии на кнопку «Редактировать» перенаправлять на страницу #/courses/id (Id – это id курса).

При нажатии на кнопку «Удалить», вывести окно подтверждения, используя сервис $modal из angular-ui. Удалить курс при подтверждении.

Под описанием курса вывести список авторов, через запятую.



1. Реализация страницы Course

Страница редактирования и создания курса. Если id курса равен new, то это создание курса. Если пришел id отличный от new, то редактирование курса. Для получения и сохранения изменений использовать сервис Courses (созданный на шаге 4).

Всю работу с курсом эмулировать через сервис $httpBackend (расширение работы эмуляции п. 4).

В шапке должен появится breadcrumbs «Курсы->Курс 12». «Курсы» должен быть ссылкой при клике, нас перекидывает на страницу списка курсов.

При изменении названия курса, имя курса также должно меняться в breadcrumbs.

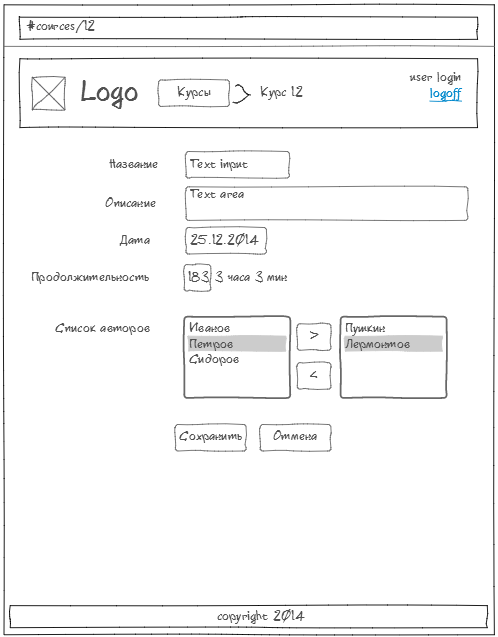
Все поля являются обязательными.

Поле «Дата» сделать директивой допускающей ввод только в формате dd.mm.yyyy. Т.е. Если пользователь кликает в текстовое поле с датой и начинает ввод «22», то текстбокс позволит дальше вводить только “.”, а после точки только цифру (упростим проверку даты до следующего формата \d\d.\d\d.\d\d\d\d).

«Продолжительность» сделать директивой (которая применяется на текстовое поле и на изменение переводит число в минутах – в часы и минуты отображая их справа). Директива должна использовать тот же фильтр преобразования минут в часы минуты, что и на странице курсов.

Если поля заполнены некорректно – то при нажатии “Сохранить” должно выдаваться сообщение об ошибке (использовать angular-ui сервис $modal).

Кнопка «Отмена» переходим на страницу курсов, изменения не сохраняются.



1. Разграничение доступа к страницам. Если пользователь залогинен на сайт ему доступны все страницы. Если нет то перенаправлять его на страницу «Входа»(Login).

Рекомендации:

* Использовать ngView и подменять только центральную часть страниц (header и footer неизменны). Хорошей идеей будет повесить на <body> 1 общий контроллер mainCtrl (который будет управлять шапкой сайта на всех страницах) + для каждой страницы свой собственный контроллер.
* Для хранения логина пользователя можно использовать AngularJS сервис (например, factory с методами login(login,password) сохраняющим эту информацию в localStorage или просто переменную, методом logoff(), и методом getLogin()). Можно посмотреть в сторону angular-local-storage.
* Эмуляция работы бэкенда производить с помощью сервиса $httpBackend.
* jQuery манипуляция с DOM происходит только в директивах (не в контроллерах!).
* Для вывода сообщений использовать сервис $modal (angular-ui).
* Выполнять задание следует строго по этапам.

**Задание 2**

Реализовать jasmine тесты для одного из контроллеров (несколько штук)