Test Project

WEB Technologies  
PHP and JavaScript Module for Web Technologies Challenge (Brasilia)

**Submitted by:**

Zoltan Sisak - HU

**Based on pre Test Project for WS2019 submitted by:**

Manuel Schaffner - CH (Lead Expert)

Siew Hwa Kang - SG

Volha Rabychyna - BY

Kevin Chong Shiuh Koong - TW

Ling Zhang - CN

Tsukasa Kamiyama - JP

Tanja Otte - DE

Zoltan Sisak - HU

Dalibor Dvorski - CA

Hassan Ibno El Fata - MA

Jaafar Almoadhen - BH

# Contents

1. [Contents 2](#_Toc12765658)
2. [Introduction 3](#_Toc12765659)
3. [Description of project and tasks 3](#_Toc12765660)
   1. [Phase One – Part A – Administration Site 3](#_Toc12765661)
   2. [Phase One – Part B – RESTful API for the Club Member Site 10](#_Toc12765662)
   3. [Phase Two 13](#_Toc12765663)
4. [Instructions to the Competitor 19](#_Toc12765664)
5. [Equipment, machinery, installations, and materials required 20](#_Toc12765665)
6. [Marking Scheme 20](#_Toc12765666)
7. [Appendix A 21](#_Toc12765667)

# Introduction

ACEF (Alliance for CS Education Foundation) is a non-profit organization foundation in Kazan.The main goal of this NGO to give support and services for CS Education in Republic of Tatarstan, Russia. The target group are the primary and secondary CS teachers. ACEF just announce their new initiative called CS Teacher Club (CST Club). Thanks for the financial support of their partner companies from the industry different IT courses will be delivered for the members of CST Club for free. The project needs a complex web solution including admin pages and Club member area. Admin users of the portal will manage club members and courses. On the club member pages CS teachers can register themselves to the club, club members can sign up for one or more upcoming courses.

# Description of project and tasks

The project is split into two phases:

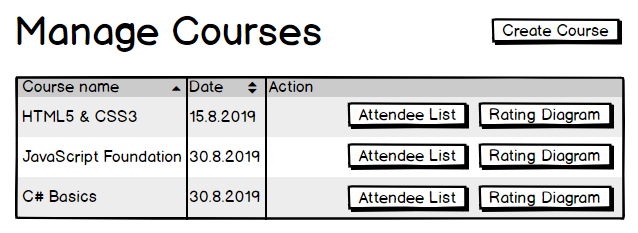
* Phase one (morning, 3hrs) for building the backend parts (admin interface and REST-API) using PHP (framework) and database
* Phase two (afternoon, 3hrs) for building the frontend parts (club member site) using HTML/CSS/JS (framework). (A mockup serever with fully functional REST-API backend will be provided for phase two.)

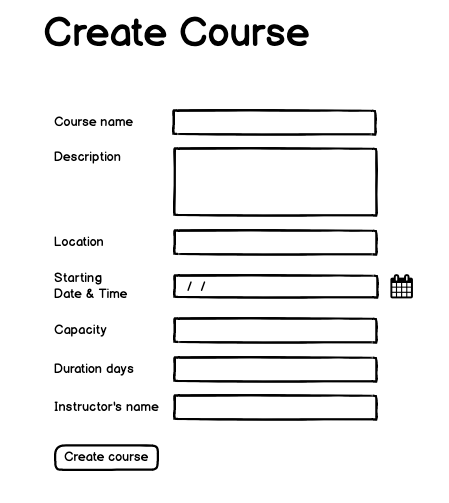
## Phase One – Part A – Administration Site

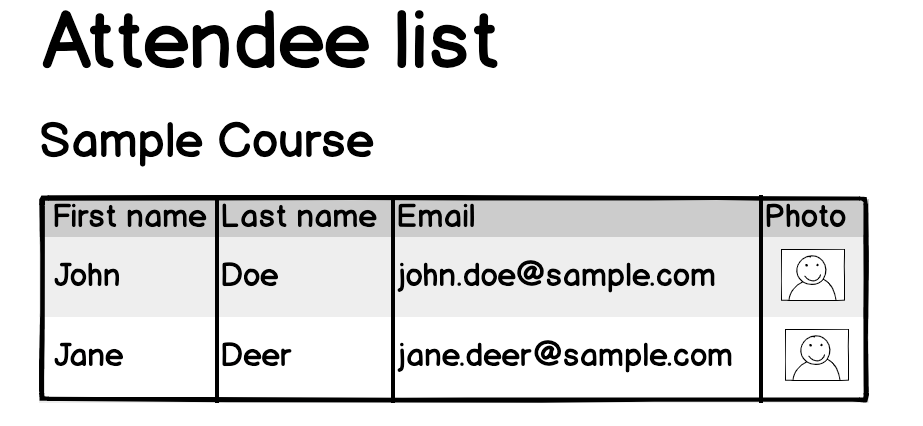
* Use your database (dbYY\_competitor\_day2) and import the schema/dump provided. The dump contains the courses table with a few sample records.
* Extend your database to be able to save
  + Admin users (admin table)
    - id
    - username
    - password
  + Club members’ profile (member table)
    - id
    - firstname
    - lastname
    - username
    - email
    - photo (png image encoded as base64 without format prefix. As an example, see nsevcsenkopng\_base64.txt in the media folder.)
    - teacher\_id (11 digit long integer)
    - is\_activated
    - password (MD5 hash)
  + Registrations of members for courses (registration table)
    - id
    - member\_id
    - course\_id
    - registration\_date
    - rate
  + If you need you may extend the database with additional tables and/or fields.
* Create the following users to login to the system (they will be used during the assessment too):
  + Admin user: {username: 'adminuser1', password: 'adminpass1'}
  + Member: {username: 'member1', firstname: 'Nikolai', lastname: 'Sevcsenko', email: 'nsevcsenko@sk1kazan.ru', photo: <base64 encoded nsevcsenko.png>, teacher\_id: '71862479596', password: <MD5 hash of 'member1pass'>}

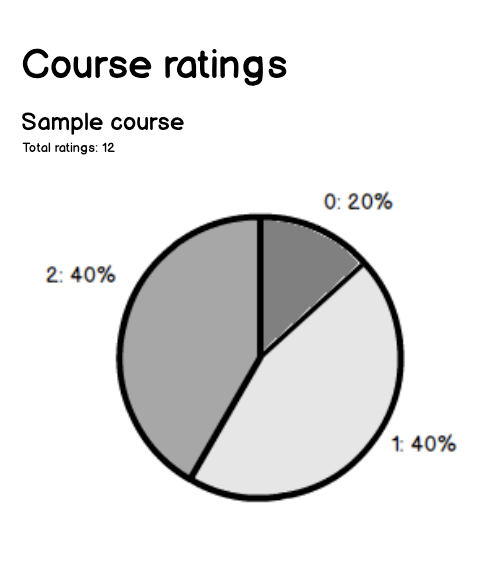
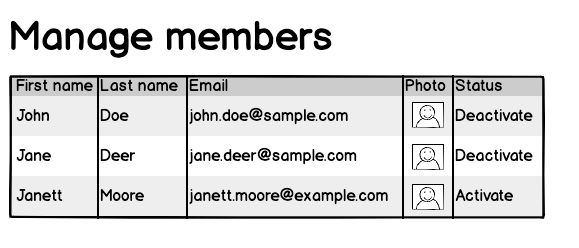
You can find nsevcsenko.png and nsevcsenkopng\_base64.txt files in the media folder.

* Create a server side rendered (SSR) admin interface using the templates provided to
  + Manage courses (create, read, update),
  + Display course attendees,
  + A rating report diagram for a course, showing
    - the total number of ratings
    - percentage of bad/average/excellent ratings









* Additional info for the backend/admin interface specification:

|  |  |
| --- | --- |
| Feature: B1 | Login and Logout as admin: As an admin I want to login / logout to the course admin platform so that I can access the functionality |
| Scenario: B1a | * Admin login ok   + Given I'm on the "login" page   + When I enter a right username and password for admin   + And press "login" button   + Then I'm forwarded to the admin area   + And username and "Logout " button become visible   + And page "Manage courses " becomes visible |
| Scenario: B1b | * Admin login fail   + Given I'm on the "login" page   + When I enter a wrong username and password for admin   + And press "login" button   + Then I stay on the "login" page   + And I see an error message "User or password not correct " |
| Scenario: B1c | * Admin logout   + Given I'm logged in   + When I press the "logout " button   + Then I'm forwarded to the "login" page   + And username and logout button and admin functions disappear |

|  |  |
| --- | --- |
| Feature: B2 | Manage members: As an admin I want to manage existing members |
| Scenario: B2a | * See member list on the “Manage members” page   + *Given* I'm on the "Manage members " page   + *Then* I see a table with all members in alphabetical order with the following elements: first name, last name, email, photo thumbnail, link for activate/deactivate membership |
| Scenario: B2b | * Dectivate a member on the “Manage members” page   + *Given* I'm on the "Manage members " page   + *When* I click on Activate link in a row of an activated member   + *Then* the title of the link changes to “Activate”   + *And* member will be deactivated |
| Scenario: B2c | * Activate a member on the “Manage members” page   + *Given* I'm on the "Manage members " page   + *When* I click on Deactivate link in a row of a deactivated member   + *Then* the title of the link changes to “Deactivate”   + *And* member will be activated |

|  |  |
| --- | --- |
| Feature: B3 | Manage a course: As an admin I want to manage courses and list attendees |
| Scenario: B3a | * See course list on the “Manage Courses” page   + *Given* I'm on the "Manage Courses " page   + *Then* I see     - a table with the all courses in descending order by date (one row for each course), with the following elements: course name, date-time, buttons for attendee list and rating diagram     - “Create course” button to add a new course |
| Scenario: B3b | * See existing course details on the “Course Details” page   + *Given* I'm on the "Manage courses " page   + *When* I click on a course title (in table row)   + *Then* I see     - all course info including course title, starting date and time, location, duration, description, max. seats, seats available and instructors’ name,     - “Edit” button to edit the course details |
| Scenario: B3c | * Show form for a new course   + *Given* I'm logged in   + *And* I'm on the "Manage Courses" page   + *When* I press "Create Course" button   + *Then* a form appears to enter course details |
| Scenario: B3d | * Enter general info for a new course on the “Create Course” page   + *Given* a form to enter course info is shown   + *When* I enter details for the course (all fields required)   + *And* I press "Create Course"   + *Then* the course is saved into the database   + *And* the corresponding “Course Details” page appears with the following success message: "Course successfully created" |
| Scenario: B3e | * Edit existing course details   + *Given* I see course details   + *When* I click on "edit course "   + *Then* a form appears to edit course details   + *And* I edit the info   + *And* I press "Save Course "   + *Then* the course is saved into the database with the changed data   + *And* the corresponding “Course Details” page appears with the following success message: "Course successfully saved" |
| Scenario: B3f | * Show attendee list for a course   + *Given* I'm on the "Manage courses " page   + *When* I press the button "Attendee list " in a course row   + *Then* I see the course name and a list of all attendees for this course in ascending order by registration date with the following elements: first name, last name, email, photo thumbnail |
| Scenario: B3h | * Create a rating report diagram   + *Given* I'm on the "Manage courses " page   + *When* I press the button "Rating diagram" on a course row   + *Then* I see the course name, the total number of ratings for the course and pie chart with the category (0,1,2) and the percentages for 0/1/2-ratings   *For the rating report graphic, you can use one of the provided JS-libraries (Charts.js, D3.js)* |

## Phase One – Part B – RESTful API for the Club Member Site

* Create a RESTful API (http://192.168.10.10Y/api/v1/ …) to
  + Read courses
  + Create a member profile including a profile photo
  + Sign up a currently logged in member for a course
  + Show registered/attended courses for a member
  + Rate a course
  + Login and logout a member.
* API specification:

|  |  |
| --- | --- |
| Feature: A1 | Create new member profile/login: As a developer I want to create a member profile that I can use for login |
| Scenario: A1a | Create a member profile (in case of successful registration new member must be activated)   * Request   + URL: /api/v1/profile   + Method: POST   + Header:   + Body: {firstname, lastname, username, email, password, teacher\_id, photo} (photo: png image encoded as base64 without format prefix ("data:image/png;base64,") As an example, see nsevcsenkopng\_base64.txt in the media folder.) * Response if success   + Header: Response status: 200   + Body: {token} (MD5 encrypted username, valid until logout) * Response if member email or username already registered   + Header: Response status: 422   + Body: {message: "Member email or username already registered"} * Response if teachers\_id is wrong   + Header: Response status: 422   + Body: {message: "Wrong teacher ID"} (See the details of teacher ID checking algorithm below) |

|  |
| --- |
| Teacher ID checking algorithm:  Teacher ID is a 11-digit number starting with number 7. The last digit is a checksum which is created by the first 10 digits:   1. Each digit should be multiplied by its position number 2. Sum the results 3. and calculate the sum mod 11   Example: 71862479596 |

|  |  |
| --- | --- |
| Feature: A2 | Login and Logout as member: As a developer I want to login/logout a member that I can read/save data |
| Scenario: A2a | Member login   * Request   + URL: /api/v1/login   + Method: POST   + Body: {username, password} * Response if success   + Header:   + Response status: 200   + Body: {token} (MD5 encrypted username, valid until logout) * If member login info not correct or member is deactivated   + Header: Response status: 401   + Body: {message: "Invalid login"} |
| Scenario: A2b | Member logout   * Request:   + URL: /api/v1/logout?token={AUTHORIZATION\_TOKEN}   + Method: POST * Response if success   + Header: Response status: 200   + Body: {message: "Logout success"} |

|  |  |
| --- | --- |
| Feature: A3 | Read courses: As a developer I want to read course data that I can use it for display |
| Scenario: A3a | Read course info as logged in user   * Request   + URL: /api/v1/courses?token={AUTHORIZATION\_TOKEN}   + Method: GET * Response if success   + Header: Response status: 200   + Body: [{id, title, description, date\_time, duration\_days, location, instructor\_name, seats},…] * Response if user not logged in   + Header: Response status: 401   + Body: {message: “Unauthorized user”} |

|  |  |
| --- | --- |
| Feature: A4 | Manage registration: As a developer I want to send/read course registration data |
| Scenario: A4a | Register for a course as a logged in member   * Request   + URL: /api/v1/registrations?token={AUTHORIZATION\_TOKEN}   + Method: POST   + Header:   + Body: {course\_id} * Response if success   + Header: Response status: 200   + Body: {message: "Registration success"} * Response if member not logged in   + Header: Response status: 401   + Body: {message: "Unauthorized user"} * Response if course does not exist   + Header: Response status: 404   + Body: {message: "Not found"} * Response if the member already registrered for the course   + Header: Response status: 406   + Body: {message: "Member already registered"} * Response if there is no available seat   + Header: Response status: 406   + Body: {message: "No available seat"} |
| Scenario: A4b | Show registered courses for logged in member   * Request   + URL: /api/v1/registrations?token={AUTHORIZATION\_TOKEN}   + Method: GET * Response If success   + Header: Response status: 200   + Body: [{id, course\_id, user\_id, registration\_type, registration\_date, course\_rating}, …] * If member not logged in   + Header: Response status: 401   + Body: message: {message: "Unauthorized user"} |

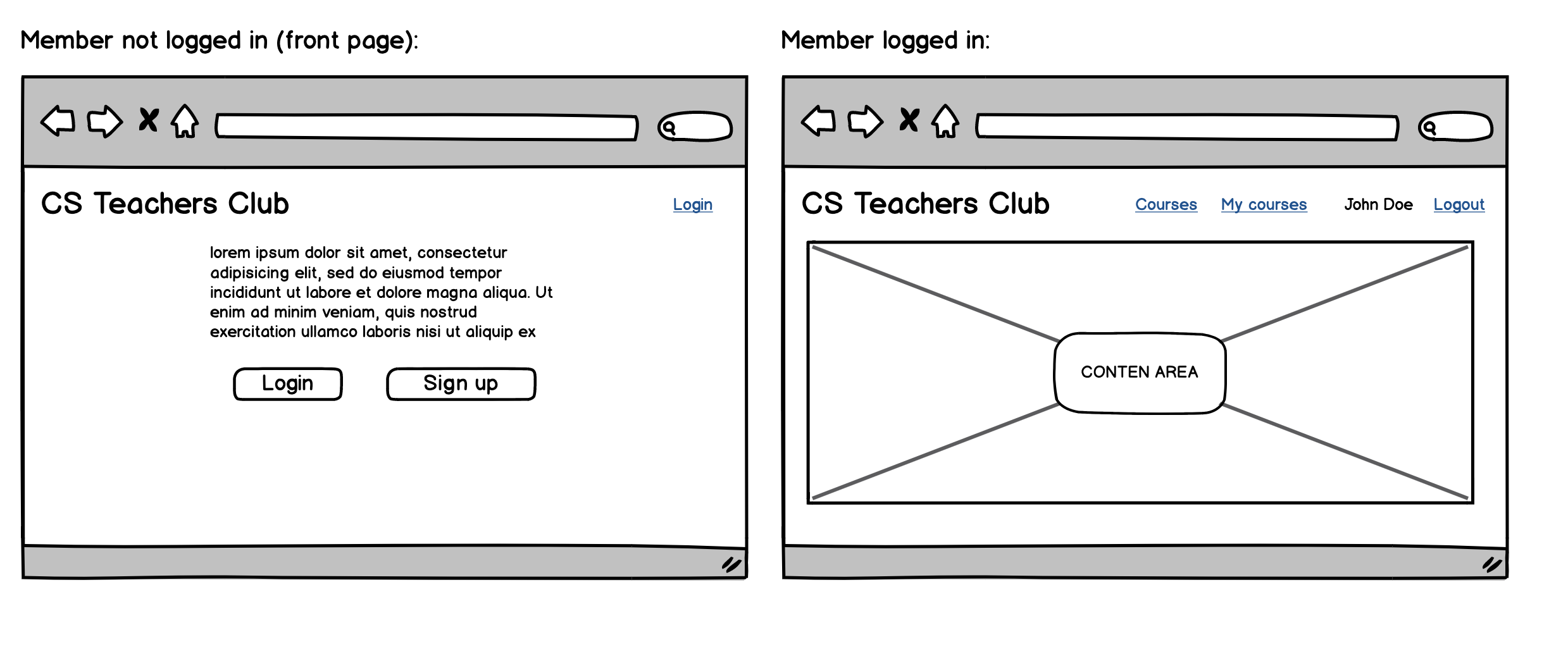
|  |  |
| --- | --- |
| Feature: A5 | Rate registered/attended courses: As a developer I want to send course rating data to the course application so that so that it is persisted |
| Scenario: A5a | Rate a registered/attended course as logged in member   * Request   + URL: /api/v1/registrations/:course\_id?token={AUTHORIZATION\_TOKEN}   + Method: PUT   + Body: {course\_rating} * Response If success   + Header: Response status: 200   + Body: message: {message: " Rating success"} * Response If course\_id does not exist or wrong course\_rating or logged member is not registered for the specific course   + Header: Response status: 400   + Body: message: {message: "Bad request"} * If member not logged in   + Header: Response status: 401   + Body: {message: "Unauthorized user"} |

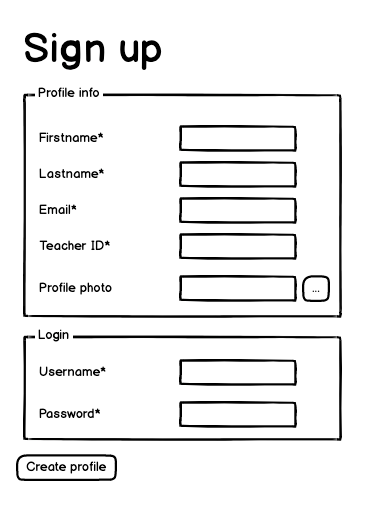
## Phase Two

In this phase you have to create a RESTful single page frontend application (SPA) for club members to browse information about the courses, filter courses by start date, create a member profile, register for a course, download an ical-file (created by the frontend application) for a registered course, rate an attended course. Instead of your backend you created in Phase One you must use a fully functional mockup server provided by the competition organizer on the followingURL: ([http://192.168.10.10Y:8001/api/v1/).](http://192.168.10.10Y:8001/api/v1/).%20)

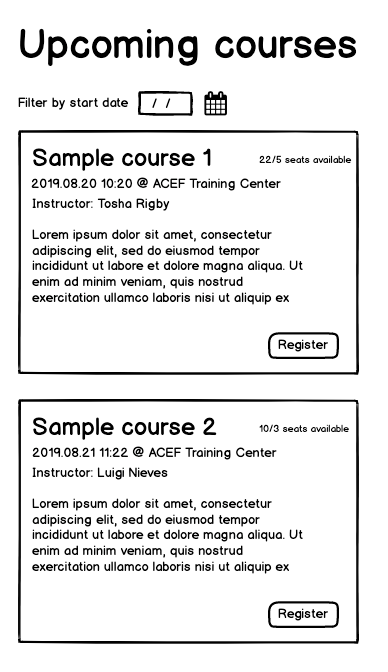
Interaction should show error/feedback messages based on response from API where appropriate.

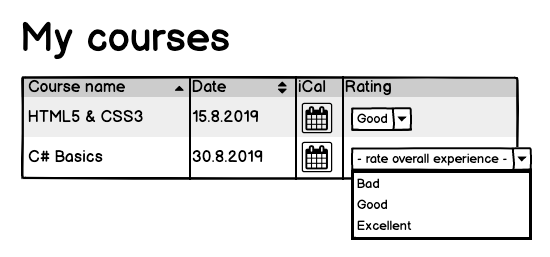
In order to use the member site, CS Teacher have to create a member profile first.











* Additional info for the frontend interface:

|  |  |
| --- | --- |
| Feature: F1 | Registration to the Club - As an CS teacher I want to sign up to the Club |
| Scenario: F1a | * Start the sign up   + *Given* I am on the "front page" or on the “login page”   + *When* I click the “Sign up” button or link   + *Then* „sign up” page appears |
| Scenario: F1b | * Club registration success   + *Given* I am on the "sign up page"   + *When* I fill all the required fields with the appropriate data   + *And* I click on “Create profile” button   + *Then* my membership profile is created (my data is saved in the database)   + *And* I am logged in   + *And* "Courses"*,* “My Courses”, "Logout" link and my first and second name is shown,   + *And* "Login" link disappears   + *And* "courses" page is shown |
| Scenario: F1c | * Club registration fails - one or more required fields are empty   + *Given* I am on the „sign up” page   + *When* I don’t fill all the required fields   + *And* I click on “Create profile” button   + *Then* the data is not saved in the database   + *And* I stay on the „sign up” page   + *And* a “Missing field(s)” warning message appears |
| Scenario: F1d | * Club registration fails – wrong email format   + *Given* I am on the „sign up” page   + *When* I enter my email address in wrong format (not in x@y format)   + *And* I click on “Create profile” button   + *Then* the data is not saved in the database   + *And* I stay on the „sign up” page   + *And* “Wrong email address” warning message appears |
| Scenario: F1e | * Club registration fails – email or username is already registered   + *Given* I am on the „sign up” page   + *When* I fill all the required fields including a previously registered email or username   + *And* I click on “Create profile” button   + *Then* the data is not saved in the database, user is activated   + *And* I stay on the „sign up” page   + *And* “Email or username already registered” warning message appears |
| Scenario: F1f | * Club registration fails – wrong teacher ID   + *Given* I am on the „sign up” page   + *When* I fill all the required fields with the appropriate data but my teacher ID is wrong   + *And* I click on “Create profile” button   + *Then* the data is not saved in the database   + *And* I stay on the „sign up” page   + *And* “Wrong teacher ID” warning message appears |

|  |  |
| --- | --- |
| Feature: F1 | Login and logout as a member - As a club member I want to login that I can see and register for courses |
| Scenario: F1a | * Member login ok   + *Given* I am on the "login" page   + *When* entering correct member username and password   + *Then* I am logged in   + *And* "Courses"*,* “My Courses”, "Logout" button and my first and second name is shown,   + *And* "Login" link disappears   + *And* "courses" page is shown |
| Scenario: F1b | * Member login fail   + *Given* I am on the "login" page   + *When* entering incorrect username and password   + *Then* I stay on the "login" page   + *And* I see an error message "Username or password not correct" |
| Scenario: F1c | * Member logout ok   + *Given* I'm logged in   + *When* I press the "Logout" button   + *Then* I'm forwarded to the "login" page   + *And* username and “Logout” button and course functions disappear |

|  |  |
| --- | --- |
| Feature: F2 | Browse courses - As a member I want to browse courses so that I see upcoming courses |
| Scenario: F2a | * Show course list   + *Given* I'm logged in as a member and I am on the “courses” page   + *Then* I see a list of upcoming courses (a filtered list of courses which start after the actual date) in descending order by date with the following items: course title, start date, location, instructor name, description, available / maximum seats and “Register” button. If I already registered for that course or there is no seat available, “Register” button is disabled |
| Scenario: F2b | * Filter course list by start date   + *Given* I'm logged in as a member   + *When* I select a date in the date filter   + *Then* I see a filtered list of courses wich contains courses which start on or after the selected date but not earlier than the actual day +1 |

|  |  |
| --- | --- |
| Feature: F3 | **Register for a course - As a member I want to register for a course** |
| Scenario: F3a | * Register for a course   + *Given* I'm logged in as a member and I am on the “courses” page   + *And* I click the "Register" button on a course where “Register” button is enabled   + *Then* the registration is saved in the database   + *And “*Register for the course was successful” message appears   + *And* available seats data is updated at that course.   + *And* “Register” button will be disabled |

|  |  |
| --- | --- |
| Feature: F4 | **Manage course registrations - As a member I want to manage my course registrations** |
| Scenario: F4a | * - Show course registrations   + *Given* I'm logged in as a member   + *When* I click on the link "My courses" on the main page   + *Then* a table with the courses I have registered for is shown |
| Scenario: F4b | * Download iCal   + *Given* I'm on the "my courses" page   + *When* I click on an "iCal" icon   + *Then* an ical-file downloads   + *And* the ical-file contains info for the respective course (date, time, course title).   *See iCal file example in the Appendix A.* |
| Scenario: F4c | * Rate course   + *Given* I'm on the "my courses" page   + *When* I select a rating in the dropdown (can be done multiple times for each course)   + *Then* the rating is saved / updates in the database   + *And* “Rating success” message appears |

Notes

* You have to finish your First Phase tasks by the end of the first 3 hour session. If you finish earlier you can start to work on your Second Phase tasks. For the Second Phase you have to use a fully working implementation of the first phase (API) using a specified url ([http://192.168.10.10Y:8001/api/v1/).](http://192.168.10.10Y:8001/api/v1/).%20)
* Competitors must use one of the server-side and client-side frameworks/libraries that are provided.
* The provided template for the admin interface should be used. It can be enhanced to get better functionality for your site. The visual design of the *admin interface* will *not* be assessed.   
  The design of the member interface can be based on the template of the admin interface. The visual design of the *member interface* *will* be assessed: a simple, clean and attractive design is expected
* The specified database tables need to be implemented as provided. More tables may be added if needed. Provide a final SQL-dump and ERD (Entity Relationship Diagram) screen as specified in the “Instructions to the Competitor” session.
* All API should fulfill all requirements as stated in the description. All prefix, RESTful-URL and HTTP-Method from given API link should be implemented correctly and not be changed.
* Changes made in the data need to be propagated between frontend and backend in both directions.

# Instructions to the Competitor

* Files to be collected after the first phase (morning) on the server:
  + Web service (http://<hostname>/XX\_JS-PHP\_A/api/v1/..)
  + Admin area (http://<hostname>/XX\_JS-PHP\_A/) - initially shows login page
  + Folder inside webroot (XX\_JS-PHP\_A/)
  + ERD screen shot named “XX\_ERD.png” in “db-dump” folder inside of XX\_JS-PHP\_A
  + Database dump named “XX\_database.sql” in “db-dump” folder inside of XX\_JS-PHP\_A
* Files to be collected after the second phase (afternoon) on the server:
  + Front-end website (http://<hostname>/XX\_JS-PHP\_B/)
  + Folder inside webroot (XX\_JS-PHP\_B/) - put all development files into XX\_JS-PHP\_B\_dev.zip and leave only files for production-use in this directory.
  + If the front-end root (http://<hostname>/XX\_JS-PHP\_B) does not open the main page directly, have it open an index.html with a link to the main page for marking your front-end.
* XX is your country code
* Assessment will be done using **Google Chrome.**

# Equipment, machinery, installations, and materials required

|  |  |
| --- | --- |
| ITEM | description |
| Media Files | Contains: - DB schema / dump  - template files for the admin-gui  - Charts.js & D3.js incl. Documentation - files for testing REST-API |

# Marking Scheme

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SECTION | CRITERION | JUDGEMENT MARKS | MEASUREMENT MARKS | TOTAL |
| 1 – Work mgmt |  |  | 0.5 | 2 |
| 2 - Communication |  | 1 | 2.25 | 3,25 |
| 3 - Design |  | 2.5 | 1.5 | 4 |
| 4 - Layout |  | 2.5 |  | 2,5 |
| 5 - Frontend |  | 2 | 8.25 | 10,25 |
| 6 - Backend |  | 1.5 | 10 | 11,5 |
| TOTAL |  | 9.5 | 22.5 | 32 |

# Appendix A

iCal example:

BEGIN:VCALENDAR

VERSION:2.0

PRODID:-//ical.marudot.com//iCal Event Maker

**X-WR-CALNAME:Invitation: HTML5 & CSS3 course**

CALSCALE:GREGORIAN

BEGIN:VTIMEZONE

TZID:Europe/Moscow

TZURL:http://tzurl.org/zoneinfo-outlook/Europe/Moscow

X-LIC-LOCATION:Europe/Moscow

BEGIN:STANDARD

TZOFFSETFROM:+0300

TZOFFSETTO:+0300

TZNAME:MSK

DTSTART:19700101T000000

END:STANDARD

END:VTIMEZONE

BEGIN:VEVENT

DTSTAMP:20190628T131353Z

UID:20190628T131353Z-669183810@marudot.com

**DTSTART;VALUE=DATE:20190815**

**DTEND;VALUE=DATE:20190817**

**SUMMARY:HTML5\, CSS3 Basic**

**URL:https%3A%2F%2Fcstclub.ru**

**DESCRIPTION:\n\n\nInstructor: Aleksey Abramov**

**LOCATION:ACEF Training Center**

END:VEVENT

END:VCALENDAR

You can find this iCal example in the media files too.