# Individual Retake Project Assignment for the [Web Development Basics @ SoftUni](https://softuni.bg/trainings/1211/Web-Development-Basics-September-2015)

Design and implement a **Conference Scheduler** **using PHP and HTML / CSS / JavaScript**. Your project must meet all the requirements listed below.

## Requirements

* **Use PHP** – the major part of your work should be PHP written
  + You should create custom PHP Framework
  + You should create the project over your framework
  + You should additionallyuse **HTML5, CSS3** to create the content and to stylize your web application
  + You may optionally use **JavaScript, jQuery, Bootstrap**
  + **Use PHP 7** (Windows: <http://windows.php.net/qa/>, Linux: <http://php7.zend.com/>, both require Apache 2.4)
* **User source control system**
  + **Use GitHub** or other source control systemas project collaboration platform
  + Commit your work daily
  + Big series of days without activities might be penalized
* Publish your project **live in Internet** – your project should be public in Internet
  + You may **share your project** to get external feedback
  + Most shared and commented projects will get additional **bonus score**
* **Valid and high-quality PHP, HTML and CSS**
  + Follow the best practices fr PHP development: <http://www.phptherightway.com>.
  + Validate (when possible) your HTML (<http://validator.w3.org>) and CSS code (<http://css-validator.org>)
  + Follow the best practices for **high-quality PHP, HTML and CSS**: good formatting, good code structure, consistent naming etc.
* **Usability, UX and browser support**
  + Your web application should be easy-to-use, with intuitive UI, with good usability
  + Ensure your web application works correctly in the latest HTML5-compatible browsers: Chrome, Firefox, IE, Opera, Safari (latest versions, desktop and mobile versions)
  + You do not need to support old browsers like IE9
* **Framework requirements**
  + **ABSOLUTELY REQUIRED** (without those ones the project is invalidated)
    - **Use PHP7’s strict types (both return types and argument types) and declare strict\_types=1 everywhere in your framework and controller actions**
    - Default routing system
    - Custom routing system
      * Easy way to define custom routes with parameters and parameter types
    - Strongly typed views
    - POST request actions using Binding Models
    - Identity system.
      * Extensible system that abstracts common logic regarding users, their registration, logout, login and roles.
      * Extending the user object (e.g. MyUser extends IdentityUser) should affect the database tables holding the user info
      * The domain should have a config that accepts which is the most concrete class for IdentityUser
    - Http Context holding the Web State, providing access to the request, session, cookies and currently logged in user
      * Request is an object the holds the GET parameters in an array of properties (e.g. Request->params->userId) and the form parameters (POST) in another array of properties (e.g. Request->form->topicBody)
      * Session and Cookies are objects that provide access to the session variables and user cookies. They insert, delete and change values as well as provide access for reading them. Cookie/Session keys are also properties instead of strings (e.g. Session->userId = 56; Session->userId->delete();…)
      * User is an object holding information regarding the current user.
    - Annotations parser
      * A centralized parser for annotations
      * User should be able to create annotations by creating class that extends the base annotation class.
      * If some class/method is annotated with user-defined annotation it should trigger its effect upon calling (before calling the method/instantiating the class)
      * E.g. the developer creates RouteAnnotation extends Annotation and in its constructor accepts a string. One defines a code behind the annotation. Later annotates a method with @Route(“route/here”). Before calling that method, the parser should check if there’s an annotation with that name and trigger the codebehind.
  + **Other**
    - Data annotations
      * @Authorize – the method is not called if the user is not logged in
      * @Admin – the method is not called if the user is not logged and is not administrator
    - Overriding default route config with annotations
      * @POST – the method can be called only on POST
      * @PUT – the method can be called only on PUT
      * @DELETE – the method can be called only on DELETE
      * @GET – can be omitted. If declared or not, the method is called only on GET
      * @Route(“route/here”) – declares the route action is called upon
    - Dependency injection configuration (config file or annotation)
    - User roles and config/annotation for them (method is not called if user is not in the role)
    - Help page that scans all routes (configured, annotated and basic ones) and shows them (e.g. PUT users/{id}). Opening the route shows the serialized Binding Model that accepts if any or primitive parameters if any. (e.g. {name: name, email: email}, or ?id={id}&name={name})

## Forbidden Techniques and Tools

* Using **Laravel**, **Zend Framework**, **CakePHP** or other PHP MVC Framework is forbidden.

## Projects

Please choose one of the projects below.

### Conference Scheduler

**Required** functionalities:

* User registration / login and user profiles.
* User roles (user, site administrator, conference owner, conference administrator)
* Creating a conference. Becoming a conference owner.
  + Manage the conference venue
  + Manage the conference venue’s halls
  + Managing the program – lectures, breaks, etc… They should be time boxed. If it’s led by someone – add the speaker profile.
  + Invite users for speakers. When user is invited as a speaker one receives a notification. By accepting the invitation, automatically the program is edited and the new lecture with its speaker is added.
  + Managing conference administrators.
  + Discard the conference
* View open conferences
* View particular conference
* Mark lectures as “must visit” (register for the lecture). If lecture is marked as “must visit”, each other lecture that collides in the time-box should be blurred and not available for “must visit”.
* “Maximum lectures” functionality where users receive suggestion from the system which lectures to visit in order to visit maximum number of lectures possible (that do not have time collision) for that conference. If there are two (or more) combinations of maximum possible lectures, the user should see them both (or more). User should be able to accept the combination, thus marking as “must visit” each of the lectures in that combination.
* View your own schedule (which lectures you are registered for)
* Halls should have users limit
* Do not let users to exceed the halls limit.
* Site administrators have full access to each conference, each lecture and each user’s profile
* Conference administrators have full access to a particular conference except the venue, halls and discarding the conference.

**Bonus** functionalities

* User groups (e.g. companies, schools, universities, etc…)
* Users initial cash
* Pay model. Users have cash and can pay entrance. The conference administrators and owners should have an option to create free-pass (limited count) for certain users and user groups.
* Conference revenue-share model. E.g. organizer (owner) pays 50% to the venue owners. The other 50% distributes to the other conference administrators or speakers.
* Users from user groups should be able to see if they can receive a free-pass for any conference and eventually get that pass.
* Venue goods. Coffee, tea, sweets, lunch, etc…
* Free-pass access level. Which venue goods the pass can have access to.

## Public Project Defense

Each student will have to deliver a **public defense** of its work in front of the SoftUni team. The students will have **only ~15 minutes** for the following:

* **Demonstrate** the web application (very shortly).
* Show the **source code** and explain how it works.

Be **well prepared** for presenting maximum of your work for minimum time. Open the project assets beforehand to save time.

On the 15-th minute you will be interrupted!

## Assessment Criteria

* **Project Functionality –** **0…50**
* **Framework functionality – 0..70**
* **Overview** (HTML / CSS / Usability / UX) – **0…10**
* **Code quality** (correct naming, code formatting, separation of concerns, etc.) – **0…30**
* **Security** (XSS, SQL Injection, CSRF…) **– 0…25**
* **Bonus** (bonus point are given for implementing optional functionalities according to the type of project you choose) – **0..35**