

Nghề Thiết kế trang Web

Module A

Thời gian: 3h

MTSC2017\_WebDesign\_Module\_A \_EN

# INTRODUCTION

New startup company would like to create flight and hotel booking system website. This website should help user to book the flight and hotel. They ask you to create a website with separate architecture, server and client side. They’re planning to create a mobile version later, so they’re asking you to build this website with the web service architecture.

Development phase will be separate into two phases. First phase is creating backend web service and the second phase is to create front end website. The server should serve API only. There should be no any visual interface output. The only interface is the Application Programming Interface (API).

# DESCRIPTION OF PROJECT AND TASKS

The description for this module will be listed below. Your first task is to create a restful web service API that could be used by the front end to communicate the data.

The server takes application/json request. Response should also be in JSON format. All response should follows with a correct status code—Success, Not Found and Bad Request. No authentication is required for all the API calls.

Data Type

* All “date” should return in String
* All “ID” should return in integer
* All fields without specifying the type is string.

The followings are the list of web service that we need:

**Web Service**

1. **Authentication**
   1. Login (*v1/auth/login*)  
      Description: For client to get login token via username and password

Request method: **POST**

Header: header authorization basic

Requested parameter:

* Body:
  + username
  + password

Response result:

* If success,
  + header: response status: 200
  + body:
    - token`: authorization token (to be valid until logout). Token will be generated by the system from logged in username with md5 encryption method
    - Role (ADMIN / USER)
* If username/password not correct or empty,
  + header: response status: 401
  + body: message: invalid login
  1. Logout (*v1/auth/logout?token={AUTHORIZATION\_TOKEN}*)  
     Description: For server to invalid the user’s token

Request method: **GET**

Header: header authorization basic

Response result:

* If success,
  + header: response status: 200
  + body
    - message: logout success
* If unauthorized user access it, data:
  + Message: Unauthorized user
  + Response status: 401

1. **Airline**

The API allows airlines company creation, flight creation, update, delete, flights query and booking of flights.

* 1. Create Airline company (*v1/airline?token={AUTHORIZATION\_TOKEN}*), only admin access this API.

Description: For client to create a new airline company object.

Request method: **POST**

Header: header authorization basic  
Requested parameter:

* + - Body:
      * airline\_name
      * city\_name (which the airline is based at)

Response result:

* + - If success, body:
      * Message: create success
      * id: id of created Airline company
      * Response status: 200
    - If failed, body:
      * Message: Data cannot be processed
      * Response status: 422
    - If unauthorized user access it, body:
      * Message: Unauthorized user
      * Response status: 401
  1. Create airline flight (*v1/flight?token={AUTHORIZATION\_TOKEN}*), only admin access this API.

Description: For client to create a new airline flight object.

Request method: **POST**

Header: header authorization basic  
Requested parameter:

* + - Body:
      * from\_date
      * to\_date
      * flight\_time
      * arrival\_time
      * from\_city\_name (departure city name),
      * to\_city\_name (destination city name)
      * airline\_id
      * price (fare price (interger))

Response result:

* + - If success, body:
      * Message: create success
      * id: id of created flight
      * Response status: 200
    - If failed, body:
      * Message: Data cannot be processed
      * Response status: 422
    - If unauthorized user access it, body:
      * Message: Unauthorized user

Response status: 401

* + body
    - message: logout success
* If unauthorized user access it, data:
  + Message: Unauthorized user
  + Response status: 401
  1. Query/Get airline flights list (*v1/flight/{DEPARTURE\_DATE}/{DEPARTURE\_CITY\_NAME}/{DESTINATION\_CITY\_NAME}?token={AUTHORIZATION\_TOKEN}*).

Description: For client to list of airline flight are available on departure date, departure city name and destination city name.

Request method: **GET**

Header: header authorization basic

Response result:

* + - If success, array of flights. Each flight contains:
      * flight\_id
      * flight\_code
      * departure\_time
      * arrival\_time
      * arrival\_id
      * airline\_name
      * all fields from requested parameters

Response status: 200

* + - If failed, body:
      * Message: Data cannot be processed
      * Response status: 422
    - If unauthorized user access it, body:
      * Message: Unauthorized user
      * Response status: 401
  1. Book a flight (*v1/flight-book?token={AUTHORIZATION\_TOKEN}*)

Description: For client to book a flight.

Request method: **POST**

Header: header authorization basic  
Requested parameter:

* + - Body:
      * flight\_type (One way only or with return flight)
      * from\_date
      * from\_time
      * return\_date (if not one way flight)
      * return\_time (if not one way flight)
      * from\_city\_name
      * to\_city\_name
      * flight\_class (Economy, Business or First class)
      * total\_adults
      * total\_children
      * passengers consist of first\_name, last\_name (array of passengers’s first name and last name)

Response result:

* + - If success, body:
      * flight\_book\_id
      * All fields from Requested parameters
      * Response status: 200
    - If failed, body:
      * Message: Data cannot be processed
      * Response status: 422
    - If unauthorized user access it, body:
      * Message: Unauthorized user
      * Response status: 401
  1. Update airline flight (*v1/flight/{ID}?token={AUTHORIZATION\_TOKEN}*), only admin access this API.

Description: For client to update an existing airline flight object via flight ID.

Request method: **PUT**

Header: header authorization basic  
Requested parameter:

Body:

* + - * [from\_date]
      * [to\_date]
      * [flight\_time]
      * [arrival\_time]
      * [from\_city\_name] (departure city name),
      * [to\_city\_name] (destination city name)
      * [airline\_id]
      * [price] (fare price (interger))

Response result:

* + - If success, body:
      * Message: update success
      * Response status: 200
    - If failed, body:
      * Message: Data cannot be updated
      * Response status: 400
    - If unauthorized user access it, body:
      * Message: Unauthorized user
      * Response status: 401
  1. Delete airline flight (*v1/flight/{ID}?token={AUTHORIZATION\_TOKEN}*), only admin access this API.

Description: A request to delete a airline flight object via flight ID.

Request method: **DELETE**

Header: header authorization basic

Response result:

* + - If success, body:
      * Message: delete success
      * Response status: 200
    - If failed, body:
      * Message: Data cannot be deleted
      * Response status: 400
    - If unauthorized user access it, body:
      * Message: Unauthorized user
      * Response status: 401

1. **Hotel**

The API allows hotel creation, update, delete, hotel query and booking of hotel.

1. Create hotel (*v1/hotel?token={AUTHORIZATION\_TOKEN}*), only admin access this API.

Description: For client to create a new hotel object.

Request method: **POST**

Header: header authorization basic  
Requested parameter:

* + - Body:
      * name
      * city\_name
      * description
      * room\_types consist name, price (array of room types, each type includes a name and a price),
      * images (array of image URLs)
      * capacity

Response result:

* + - If success, body:
      * Message: create success
      * Response status: 200
    - If failed, body:
      * Message: Data cannot be processed
      * Response status: 422
    - If unauthorized user access it, body:
      * Message: Unauthorized user
      * Response status: 401

1. Query/Get hotel list (*v1/hotel/{CITY\_NAME}?token={AUTHORIZATION\_TOKEN}*)

Description: For client to list of hotel are available on city name.

Request method: **GET**

Header: header authorization basic

Response result:

* + - If success, array of hotels. Each hotel contains:
      * hotel\_id
      * hotel\_name
      * city\_name
      * description
      * room\_types consist name, price (array of room types, each type includes a name and a price),
      * images (array of image URLs)
      * room\_capacity

Response status: 200

* + - If failed, body:
      * Message: Data cannot be processed
      * Response status: 422
    - If unauthorized user access it, body:
      * Message: Unauthorized user
      * Response status: 401

1. Book a hotel (*v1/hotel-book?token={AUTHORIZATION\_TOKEN}*)

Description: For client to book a hotel.

Request method: **POST**

Header: header authorization basic  
Requested parameter:

* + - Body:
      * check\_in\_date
      * check\_out\_date
      * hotel\_id
      * total\_guests
      * guests consist first\_name, last\_name (array of guest’s first name and last name)
      * total\_rooms
      * room\_type (name of room type)

Response result:

* + - If success, body:
      * hotel\_book\_id
      * All fields from requested parameters
      * Response status: 200
    - If failed, body:
      * Message: Data cannot be processed
      * Response status: 422
    - If unauthorized user access it, body:
      * Message: Unauthorized user
      * Response status: 401

1. Update hotel (*v1/hotel/{ID}?token={AUTHORIZATION\_TOKEN}*), only admin access this API.

Description: For client to update an existing hotel object via hotel ID.

Request method: **PUT**

Header: header authorization basic  
Requested parameter:

* + - Body:
      * [hotel\_name]
      * [city\_name]
      * [description]
      * [room\_types] consist name, price (array of room types, each type includes a name and a price),
      * [images] (array of image URLs)
      * [capacity]

Response result:

* + - If success, body:
      * Message: update success
      * Response status: 200
    - If failed, body:
      * Message: Data cannot be updated
      * Response status: 400
    - If unauthorized user access it, body:
      * Message: Unauthorized user
      * Response status: 401

1. Delete book (*v1/hotel/{ID}?token={AUTHORIZATION\_TOKEN}*), only admin access this API.

Description: A request to delete a hotel object via hotel ID.

Request method: **DELETE**

Header: header authorization basic

Response result:

* + - If success, body:
      * Message: delete success
      * Response status: 200
    - If failed, body:
      * Message: Data cannot be deleted
      * Response status: 400
    - If unauthorized user access it, body:
      * Message: Unauthorized user
      * Response status: 401

1. **Transaction**

The API allows transaction creation.

1. Create a transaction (*v1/transaction?token={AUTHORIZATION\_TOKEN*

Description: For client to create a new transaction.

Request method: **POST**

Header: header authorization basic  
Requested parameter:

* + - Body:
      * title (Mr, Mrs, Miss, Dr)
      * first\_name (User’s first name)
      * last\_name (User’s last name)
      * email
      * phone
      * payment\_method (transfer, credit card or paypal)
      * card\_name (if pay by credit card)
      * card\_number (if pay by credit card)
      * ccv (if pay by credit card)

Response result:

* + - If success, body:
      * payment\_status (payment status to indicate success or error)
      * transaction\_id
      * Response status: 200
    - If failed, body:
      * Message: Data cannot be processed
      * Response status: 422
    - If unauthorized user access it, body:
      * Message: Unauthorized user
      * Response status: 401

Those are the list that required by the company in the first phase.

# INSTRUCTIONS TO THE COMPETITOR

Save the files of your application in directory on the server called **"XX\_Server\_Side\_A"**, where XX is your code.

You should export the latest SQL schema as a MySQL dump file.

# FILES PROVIDED

|  |  |
| --- | --- |
| **ITEM** | **DESCRIPTION** |
| Laravel PHP Framework incl. short documentation/cheat sheet | Web Design Competitor can choose which one of these frameworks he wants to use. Installation will be done on the familiarization-day (C-2) |
| Yii PHP Framework incl. short documentation/ | Web Design Competitor can choose which one of these frameworks he wants to use. Installation will be done on the familiarization-day (C-2) |