

KAUNAS INFORMATION TECHNOLOGY SCHOOL

**THE INTERNATIONAL PROFESSIONAL MASTERY
CONTEST**

„WEB <dev/> challenge 2023“

Module B

API Development

Introduction

A hotel company plans to open a hotel in Kaunas.

To start booking rooms, they need a website where visitors can book rooms.

It should be split into an interface and a separate API so that you can later connect your frontend to this API.

Description of project and tasks

The project is split into two phases:

- phase one (Module B, 6 hours) for building the API using a backend programming language/framework and database.
- phase two (Module F, 6 hours) for building the frontend parts using HTML/CSS and a JavaScript framework or library, consuming the API developed in Module B.

When module F starts, an API implementation will be provided so focus can be put on the frontend.

Competitor Information

The API has to be available at `https://<hostname>/module-b/`. For example, the rooms endpoint is then reachable at `https://<hostname>/module-b/api/v1/rooms`.

Database design

The database has to be named hotel.

We are providing data file:

[Module-B-data.csv](#)

This file must be imported to your database.

API

General information:

- The response body contains some static example data. Dynamic data from the database should be used.
- Placeholder parameters in the URL are marked with curly braces (e.g. {slug}).
- The order of properties in objects does not matter, but the order in an array does.

- If not specified otherwise, the Content-Type header of a response is always application/json

GET /api/v1/rooms

Returns a list of all available rooms and all their data.

Rooms are sorted by their name in ascending order.

Responses

Status Code	Data
200	<p>Description: Returned on a successful request.</p> <p>Body</p> <pre>{ "rooms": [{ "id": 1, "number": "120", "capacity": "2 beds", "floor": "2", "room_image": "img/rooms/120.jpg", "price": "200 Eur", "wifi": "true", "parking": "true", "breakfast": "true", "reservations": [{ "id": 1, "checkin": "2021-10-02T20:00:00Z", "checkout": "2021-10-02T23:00:00Z" }] }] }</pre>

GET /api/v1/rooms/{room-id}

Returns a single room.

Request

Path parameters

Parameter	Type	Description
room-id	int	Database ID of the room.

Responses

Status Code	Data
200	<p>Description: Returned on a successful request.</p> <p>Body</p> <pre>{ "room": [{ "id": 1, "number": "120", "capacity": "2 beds", "floor": "2", "room_image": "img/rooms/120.jpg", "price": "200 Eur", "wifi": "true", "parking": "true", "reservations": [{ "id": 1, "checkin": "2021-10-02T20:00:00Z", "checkout": "2021-10-02T23:00:00Z" }] }] }</pre>
404	<p>Description: Returned if a room with the specified ID does not exist.</p> <p>Body</p> <pre>{ "error": "A room with this ID does not exist" }</pre>

GET /api/v1/rooms/availability/checkin/{checkin-date}/checkout/{checkout-date}

Returns rooms availability information for the given checkin and checkout dates.

Request

Path parameters

Parameter	Type	Description
checkin-date	string	User input date
checkout-date	string	User input date

Responses

Status Code	Data
200	<p>Description: Returned on a successful request.</p> <p>Body</p> <pre>{ "rooms": [{ "id": 6, "number": "223", "availability": "true" }] }</pre>
404	<p>Description: Returned if dates are provided in incorrect format.</p> <p>Body</p> <pre>{ "error": "Bad checkin date format" }</pre>

POST /api/v1/rooms/{room-id}/reservation

Reserves room.

Request

Path parameters

Parameter	Type	Description
room-id	int	Database ID of the room.

Header

Content-Type: application/json

Body

```
{
  "name": "John Doe",
  "address": "Bees str. 15",
  "city": "Kaunas",
  "zip": "6010",
  "country": "Lithuania",
  "checkin": "2022-05-10",
  "checkout": "2022-05-12",
}
```

Name, address, city, checkin, checkout, zip and country are all **required**, of type **string**.

Responses

Status Code	Data
201	<p>Description: Returned if the reservation was successful.</p> <p>The reservation code is a 10 character long uppercase alphanumeric string that is generated randomly for each reservation. It will be presented during checkin in the hotel and used to authenticate customer.</p> <p>Body</p>

	<pre>{ "reservations": [{ "id": 1, "code": "QVLJTWK4Y7", "name": "John Doe", "created_at": "2021-09-24T11:02:20Z", "reservation_information": { "id": 1, "checkin": "2021-10-02T20:00:00Z", "checkout": "2021-10-02T23:00:00Z", "room": { "id": 11, "number": "278", } } }] }</pre>
404	<p>Description: Returned if a room with the specified ID does not exist.</p> <p>Body</p> <pre>{ "error": "A room with this ID does not exist" }</pre>
422	<p>Description: Returned if validation failed. Only fields for which the validation failed should be included in the response.</p> <p>The following validation messages are possible:</p> <ul style="list-style-type: none"> • The {field} field is required. • The {field} must be a string. <p>{field} is replaced with the field name.</p> <p>Body</p> <pre>{ "error": "Validation failed", "fields": { "name": "The name field is required.", "city": "The city must be a string.", "zip": "The zip must be a string." } }</pre>

POST /api/v1/reservations

Returns all reservations.

Users can specify their name and one of the reservation codes. If the name is correct , all reservation are returned.

Reservations are sorted by checkin date in ascending order.

Request

Body

```
{
  "code": "QVLJTWK4Y7",
  "name": "John Doe"
}
```

Responses

Status Code	Data
200	<p>Description: Returned on a successful request.</p> <p>Body</p> <pre>{ "reservations": [{ "id": 1, "code": "QVLJTWK4Y7", "name": "John Doe", "created_at": "2021-09-24T11:02:20Z", "reservation_information": { "id": 1, "checkin": "2021-10-02T20:00:00Z", "checkout": "2021-10-02T23:00:00Z", "room": { "id": 11, "number": "278" } } }] }</pre>
401	<p>Description: Returned if either code or name is missing or no reservation matches.</p> <p>Body</p> <pre>{ "error": "Unauthorized" }</pre>

	}
--	---

POST /api/v1/reservations/{reservation-id}/cancel

Deletes the specified reservation. The room assigned in hotel is available again.

All other reservations from the user remain active.

Request

Content-Type: application/json

Body

```
{
  "code": "QVLJTWK4Y7",
  "name": "John Doe"
}
```

Responses

Status Code	Data
204	Description: Returned if the reservation was successfully deleted. Body <pre>{ "message": "success" }</pre>
401	Description: Returned if either code or name is missing or they do not match the reservation. Body <pre>{ "error": "Unauthorized" }</pre>
404	Description: Returned if a reservation with the specified ID does not exist. Body <pre>{ "error": "A reservation with this ID does not exist" }</pre>