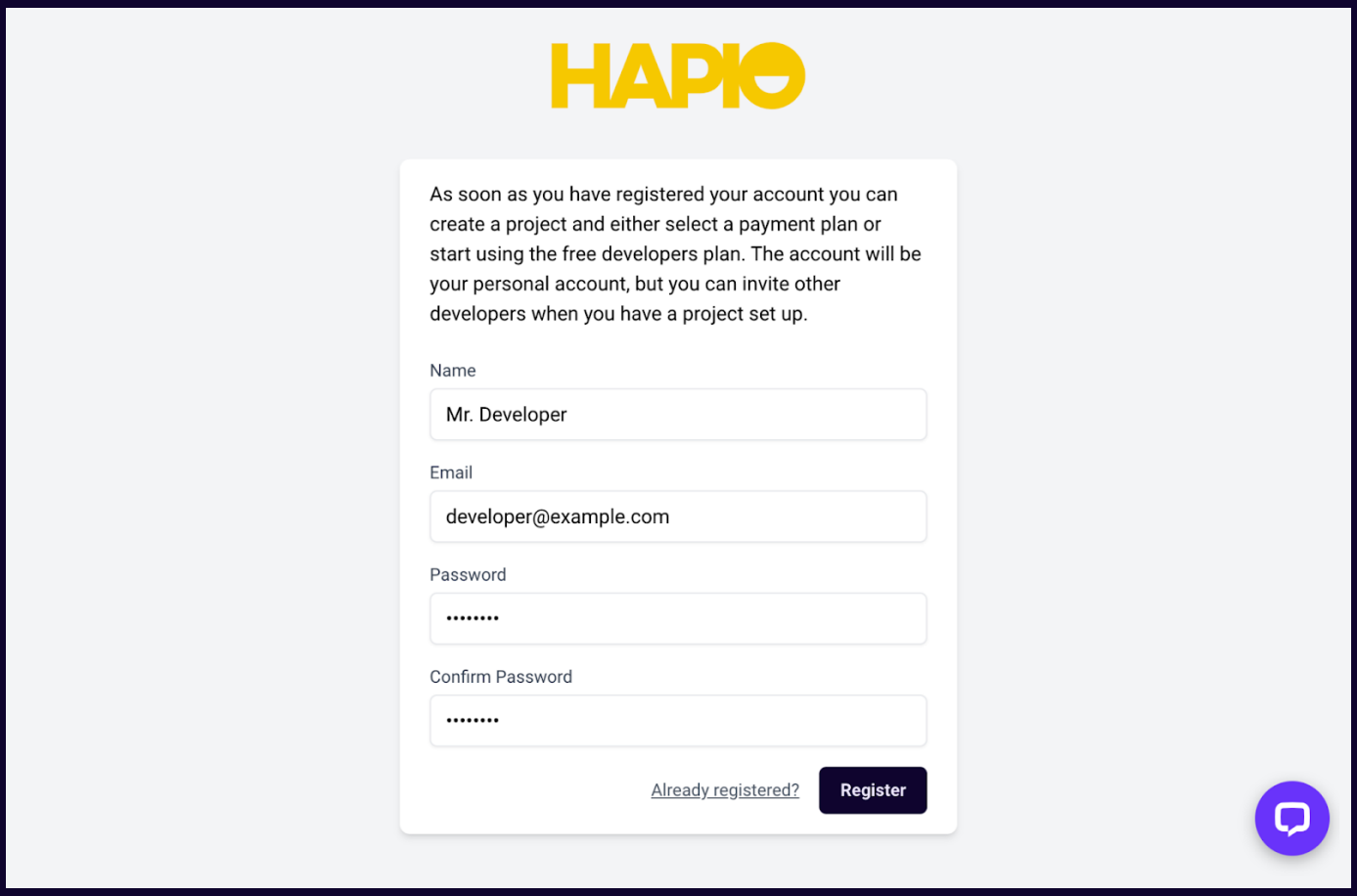
HAPIO

Getting started with Hapio.

This guide will take you through the first steps of getting started with Hapio, and will guide you all the way from registering an account to creating your first booking.

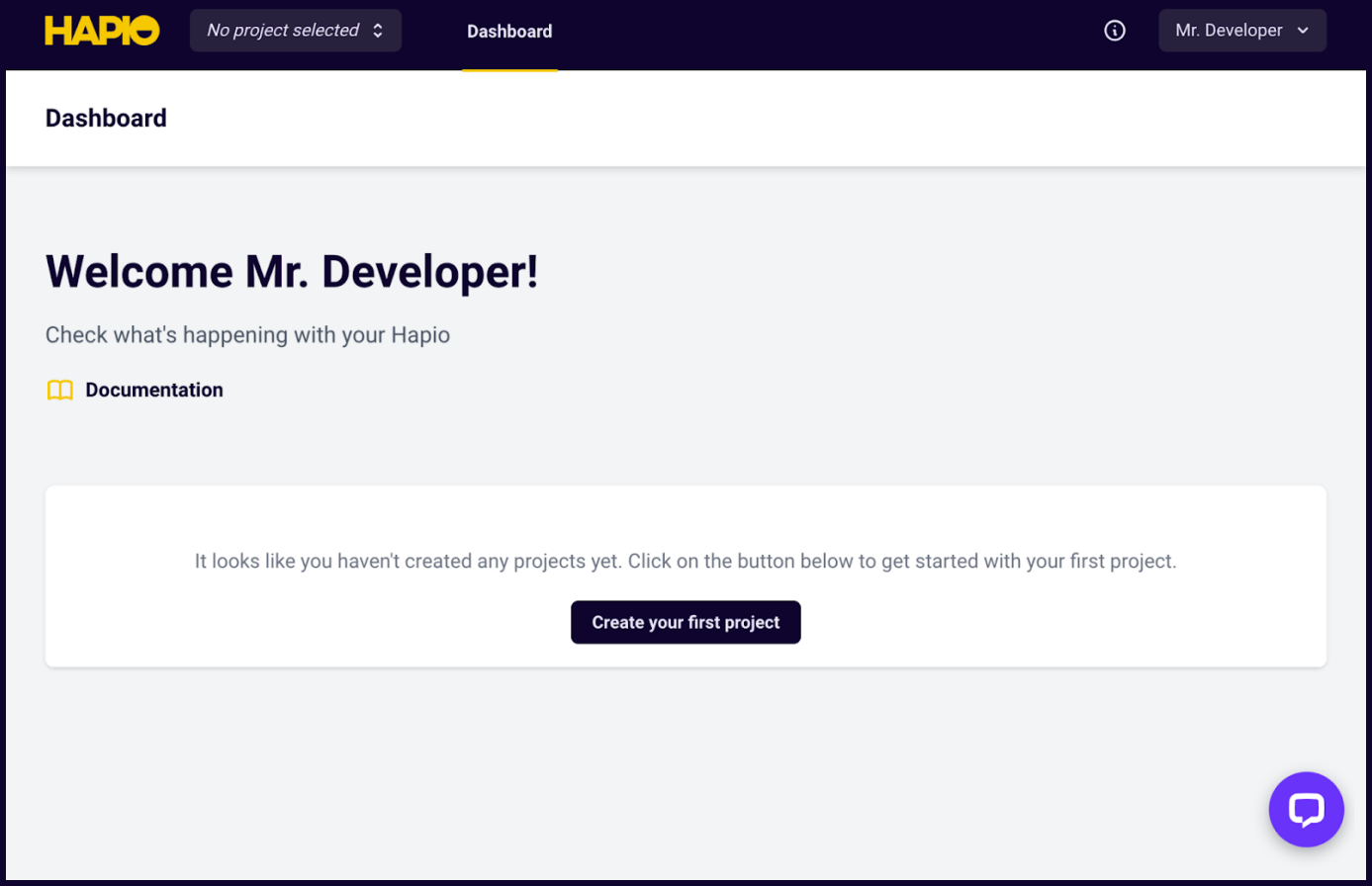
Register

The first step is to register an account in the Hapio Portal. Enter your name, email address, and password, and click on Register. Hapio will then send you an email so that you can verify your account. Remember that multiple people can be invited to join projects, so each person can have their own account in the Hapio Portal.

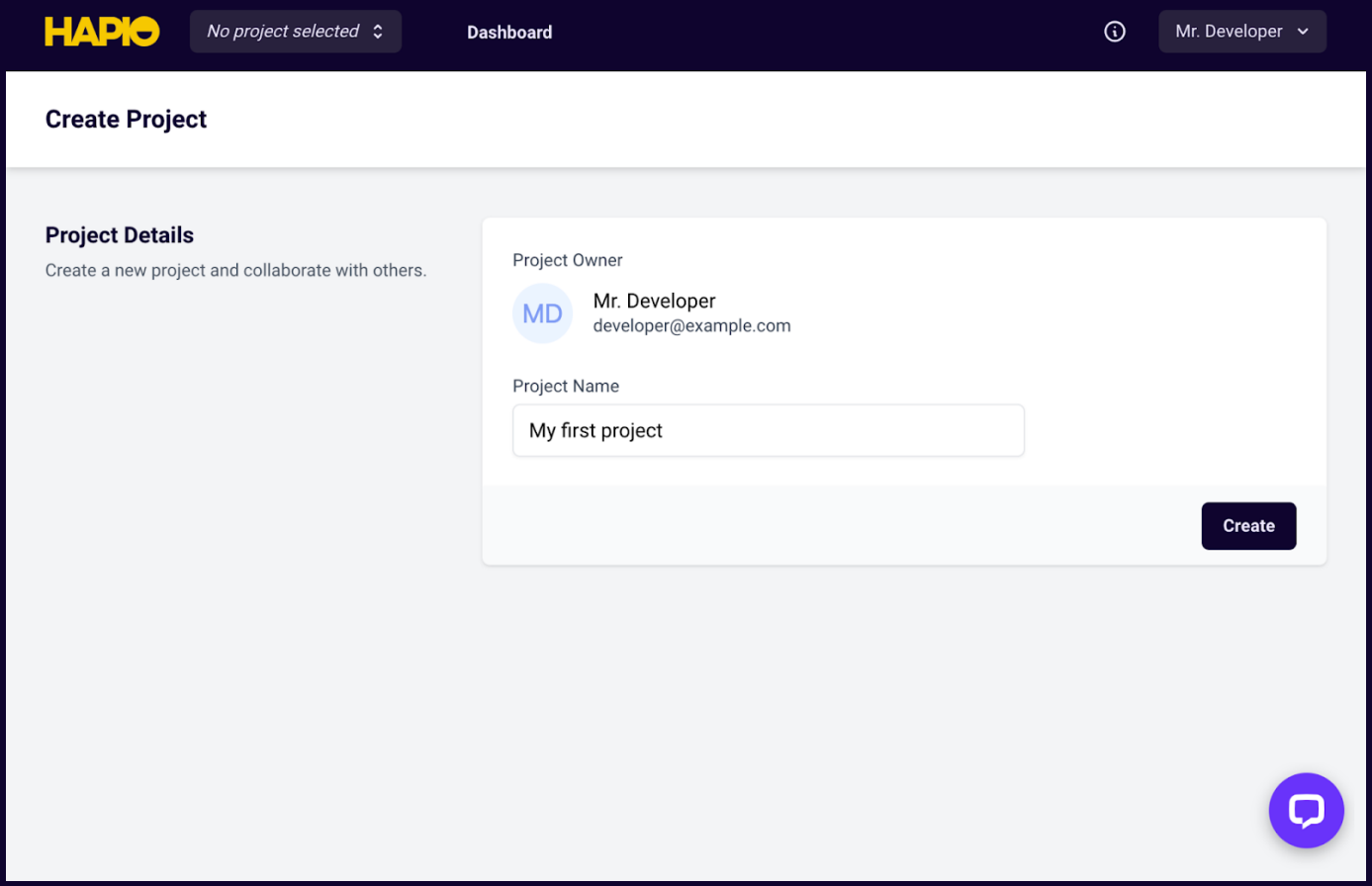


Create your first project

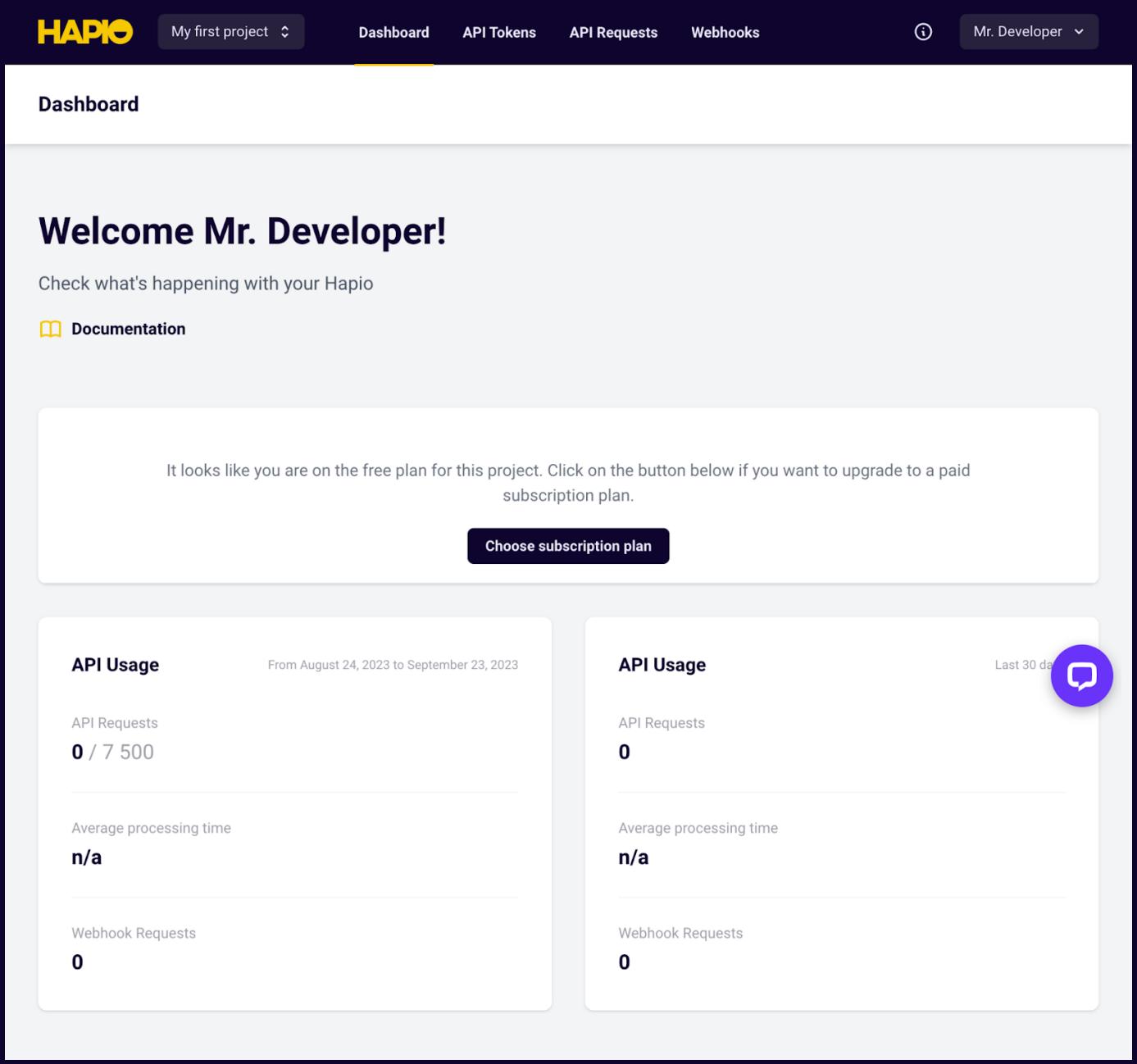
When you have verified your account, you will be logged in to the Hapio Portal. Click on **Create your first project** on the dashboard, or click on the project selector in the top navigation bar, and click on **Create New Project**.



Once you have chosen a name for your project, click on **Create**.

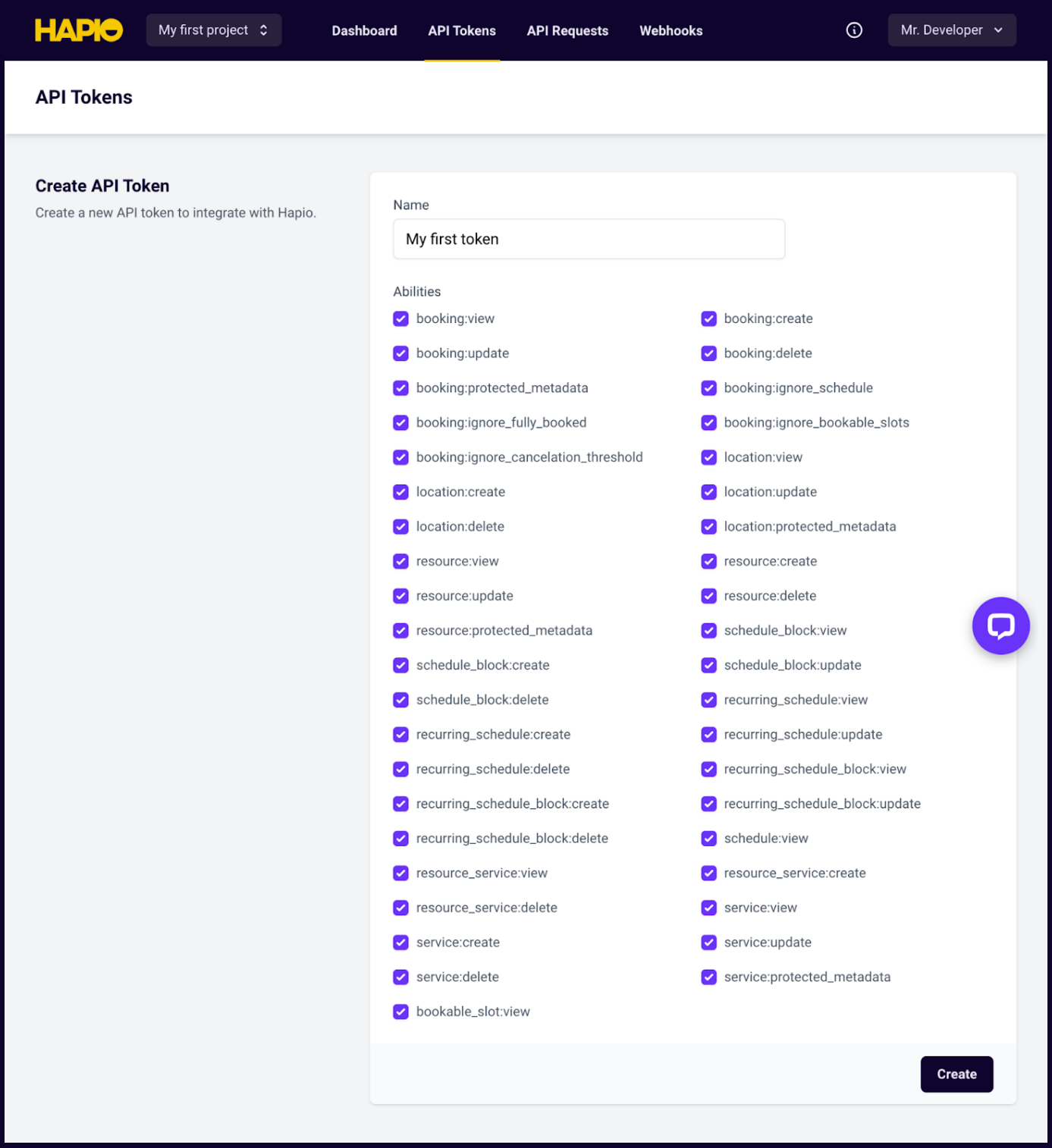


All new projects are on the free plan by default, and this plan is perfect for getting started with Hapio and can be used for development purposes. If you want to increase the limits for your project, you can subscribe to any of the paid plans. Subscription plans are chosen per project, so each project can have a subscription plan that is suitable for that specific project.



Create an API token

If you look in the project selector in the top navigation bar, your newly created project should be selected as your current project. If it is not selected, click on the project selector and select your project. Now, you can click on **API Tokens** in the top navigation bar, to get to the administration page for your API tokens. This page will list all API tokens that have been created for the project, and will also let you create new tokens. Enter a name for your token, select the abilities that your token should have, and click on **Create**. In this case, we will leave all abilities selected for simplicity, but remember that you should tailor each of your tokens to only have the abilities that it actually needs, in order to increase security.



Once you have created your token, the generated token will be shown for you to copy and store in a secure location. It's important to copy your token now, since this is the only time it will be shown, and it cannot be retrieved at a later point. All created API tokens will be listed on the bottom of the page, and from there you can edit the abilities of each token, and also delete any token if needed.

HAPIO My first project Dashboard API Tokens API Requests Webhooks Mr. Developer

API Tokens

Create API Token

Create a new API token to integrate with Hapio.

Name

Abilities

- [x] booking:view
- [x] booking:create
- [x] booking:update
- [x] booking:delete
- [x] booking:protected_metadata
- [x] booking:ignore_schedule
- [x] booking:ignore_fully_booked
- [x] booking:ignore_bookable_slots
- [x] booking:ignore_cancelation_threshold
- [x] location:view
- [x] location:create
- [x] location:update
- [x] location:delete
- [x] location:protected_metadata
- [x] resource:view
- [x] resource:create
- [x] resource:update
- [x] resource:delete
- [x] resource:protected_metadata
- [x] schedule_block:view
- [x] schedule_block:create
- [x] schedule_block:update
- [x] schedule_block:delete
- [x] recurring_schedule:view
- [x] recurring_schedule:create
- [x] recurring_schedule:update
- [x] recurring_schedule:delete
- [x] recurring_schedule_block:view
- [x] recurring_schedule_block:create
- [x] recurring_schedule_block:update
- [x] recurring_schedule_block:delete
- [x] schedule:view
- [x] resource_service:view
- [x] resource_service:create
- [x] resource_service:delete
- [x] service:view
- [x] service:create
- [x] service:update
- [x] service:delete
- [x] service:protected_metadata
- [x] bookable_slot:view

Create

Manage API Tokens

You may delete any of your existing tokens if they are no longer needed.

My first token Abilities Dele

HAPIO

Your first request

Now that you have registered an account, created a project, and also created an API token for your project, you are ready to make your first request to the Hapio API. In order to authenticate your request, you must include your token in the **Authorization** header, using the bearer authorization scheme. For our first request, we will send a **GET** request to the endpoint /v1/project. This endpoint will simply return information about the project that the API token belongs to. This request can look like this:

```http
GET /v1/project HTTP/2
Host: eu-central-1.hapio.net
Accept: */*
Authorization: Bearer GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz
```

```php
use Hapio\Sdk\ApiClient;

$apiClient = new ApiClient('GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz');

$project = $apiClient->projects()->getCurrentProject();
```

```javascript
import axios from 'axios';

const apiClient = axios.create({
    baseURL: 'https://eu-central-1.hapio.net/v1/',
    headers: {'Authorization': 'Bearer
GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz'}
});

apiClient.get('project').then(function (response) {
    const project = response.data;
});
```

It's important to note that in these examples, a dummy token will be included, since we want to show the entire request. You should of course always take the necessary steps to protect your tokens, and never show them to unauthorized people unless they are tokens intended to be public with limited abilities.

Here is the response:

```http
HTTP/2 200
date: Thu, 24 Aug 2023 08:39:02 GMT
content-type: application/json
content-length: 199
cache-control: no-cache, private
x-ratelimit-limit: 100
x-ratelimit-remaining: 99
access-control-allow-origin: *
apigw-requestid: KKCN-gMUFiAEJZw=

{
    "id": "ac590ea9-e8a1-47eb-b301-452531ac962d",
    "name": "My first project",
    "enabled": true,
    "created_at": "2023-08-24T07:51:58+00:00",
    "updated_at": "2023-08-24T07:51:58+00:00"
}
```

Map your entities

You have now made your first request to the Hapio API, so let's get started with something a little more interesting. The first thing we need to do is to map our real-world entities to the entities that are available in Hapio. Let's say that we run a medical clinic, and want to use Hapio to manage appointments with doctors in this clinic. In this case, each doctor can be represented as a resource in Hapio, and each of the services that are offered can be represented as a service in Hapio. The clinic itself can be represented as a location in Hapio. Our mapping then looks like this:

Real world	Hapio
Doctor	Resource
Service	Service
Clinic	Location

This setup will allow us to easily add more doctors, services and clinics to our booking system as needed.

HAPIO

Create a resource

We will begin the process of implementing our real-world scenario in Hapio by creating a resource for one of our doctors, by sending a POST request to the endpoint /v1/resources:

```http
POST /v1/resources HTTP/2
Host: eu-central-1.hapio.net
Accept: */*
Authorization: Bearer GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz
Content-Type: application/json
Content-Length: 84

{
    "name": "Dr. Smith",
    "max_simultaneous_bookings": 1,
    "enabled": true
}
```

HTTP

```php
use Hapio\Sdk\ApiClient;
use Hapio\Sdk\Models\Resource;

$apiClient = new ApiClient('GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz');

$resource = new Resource();
$resource->name = 'Dr. Smith';
$resource->maxSimultaneousBookings = 1;
$resource->enabled = true;

$resource = $apiClient->resources()->store($resource);
```

PHP

HAPIO

```JavaScript
import axios from 'axios';

const apiClient = axios.create({
    baseURL: 'https://eu-central-1.hapio.net/v1/',
    headers: {'Authorization': 'Bearer
GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz'}
});

apiClient.post('resources', {
    name: 'Dr. Smith',
    max_simultaneous_bookings: 1,
    enabled: true
}).then(function (response) {
    const resource = response.data;
});
```

The response for this request will include all information about our created resource:

```HTTP
HTTP/2 201
date: Thu, 24 Aug 2023 08:58:05 GMT
content-type: application/json
content-length: 282
x-ratelimit-limit: 100
x-ratelimit-remaining: 99
access-control-allow-origin: *
cache-control: no-cache, private
apigw-requestid: KKFAihv0liAEJTw=

{
    "id": "c8924e50-af0d-47ae-bf5d-e184b1b59a60",
    "name": "Dr. Smith",
    "max_simultaneous_bookings": 1,
    "metadata": null,
    "protected_metadata": null,
    "enabled": true,
    "updated_at": "2023-08-24T08:58:04+00:00",
    "created_at": "2023-08-24T08:58:04+00:00"
}
```

HAPIO

Create a service

Now, it is time to create our first service by sending a **POST** request to the endpoint /v1/services:

```http
POST /v1/services HTTP/2
Host: eu-central-1.hapio.net
Accept: */*
Authorization: Bearer GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz
Content-Type: application/json
Content-Length: 297

{
    "name": "Physical Exam",
    "price": "149.000",
    "type": "fixed",
    "duration": "PT50M",
    "bookable_interval": "PT1H",
    "buffer_time_after": "PT10M",
    "booking_window_start": "PT2H",
    "booking_window_end": "P14D",
    "cancelation_threshold": "PT12H",
    "enabled": true
}
```

```php
use Hapio\Sdk\ApiClient;
use Hapio\Sdk\Models\Service;

$apiClient = new ApiClient('GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz');

$service = new Service();
$service->name = 'Physical Exam';
$service->price = '149.000';
$service->type = 'fixed';
$service->duration = 'PT50M';
$service->bookableInterval = 'PT1H';
$service->bufferTimeAfter = 'PT10M';
$service->bookingWindowStart = 'PT2H';
$service->bookingWindowEnd = 'P14D';
$service->cancelationThreshold = 'PT12H';
$service->enabled = true;

$service = $apiClient->services()->store($service);
```

```javascript
import axios from 'axios';

const apiClient = axios.create({
    baseURL: 'https://eu-central-1.hapio.net/v1/',
    headers: {'Authorization': 'Bearer
GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz'}
});

apiClient.post('services', {
    name: 'Physical Exam',
    price: '149.000',
    type: 'fixed',
    duration: 'PT50M',
    bookable_interval: 'PT1H',
    buffer_time_after: 'PT10M',
    booking_window_start: 'PT2H',
    booking_window_end: 'P14D',
    cancelation_threshold: 'PT12H',
    enabled: true
}).then(function (response) {
    const service = response.data;
});
```

Again, the response for this request will include all information about our created service:

```http
HTTP/2 201
date: Thu, 24 Aug 2023 09:07:53 GMT
content-type: application/json
content-length: 529
x-ratelimit-limit: 100
x-ratelimit-remaining: 99
access-control-allow-origin: *
cache-control: no-cache, private
apigw-requestid: KKGchgzOliAEPQg=
```

```
{
    "id": "a09c4585-0519-44f6-a5ab-647aeffc6281",
    "name": "Physical Exam",
    "price": "149.000",
    "type": "fixed",
    "duration": "PT50M",
    "bookable_interval": "PT1H",
    "buffer_time_before": "PT0S",
    "buffer_time_after": "PT10M",
    "booking_window_start": "PT2H",
    "booking_window_end": "P14D",
    "cancelation_threshold": "PT12H",
    "metadata": null,
    "protected_metadata": null,
    "enabled": true,
    "updated_at": "2023-08-24T09:07:53+00:00",
    "created_at": "2023-08-24T09:07:53+00:00"
}
```


Create a location

Next up, we will create a location for our clinic by sending a **POST** request to the endpoint /v1/locations:

```
POST /v1/locations HTTP/2                                    HTTP
Host: eu-central-1.hapio.net
Accept: */*
Authorization: Bearer
GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz
Content-Type: application/json
Content-Length: 149

{
    "name": "Perfect Health - Stockholm",
    "time_zone": "Europe/Stockholm",
    "resource_selection_strategy": "equalize",
    "enabled": true
}
```

```php
use Hapio\Sdk\ApiClient;
use Hapio\Sdk\Models\Location;

$apiClient = new ApiClient('GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz');

$location = new Location();
$location->name = 'Perfect Health - Stockholm';
$location->timeZone = 'Europe/Stockholm';
$location->resourceSelectionStrategy = 'equalize';
$location->enabled = true;

$location = $apiClient->locations()->store($location);
```

PHP

```javascript
import axios from 'axios';

const apiClient = axios.create({
    baseURL: 'https://eu-central-1.hapio.net/v1/',
    headers: {'Authorization': 'Bearer
GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz'}
});

apiClient.post('locations', {
    name: 'Perfect Health - Stockholm',
    time_zone: 'Europe/Stockholm',
    resource_selection_strategy: 'equalize',
    enabled: true
}).then(function (response) {
    const location = response.data;
});
```

JavaScript

As with the previous requests, the response will include all information about our created location:

```http
HTTP/2 201
date: Thu, 24 Aug 2023 09:42:13 GMT
content-type: application/json
content-length: 347
cache-control: no-cache, private
```

HTTP

```
x-ratelimit-limit: 100
x-ratelimit-remaining: 99
access-control-allow-origin: *
apigw-requestid: KKLeXiW5FiAEMiA=

{
    "id": "2b113cf6-5d6c-4a31-bd21-88ba7fb440ea",
    "name": "Perfect Health - Stockholm",
    "time_zone": "Europe/Stockholm",
    "resource_selection_strategy": "equalize",
    "metadata": null,
    "protected_metadata": null,
    "enabled": true,
    "updated_at": "2023-08-24T09:42:13+00:00",
    "created_at": "2023-08-24T09:42:13+00:00"
}
```

Connect the resource with the service

Now that both our first resource and service are created, we need to associate them with each other. This tells Hapio that the resource (our doctor) is able to perform the service. We do this by sending a **PUT** request to the endpoint /v1/services/{service ID}/resources/{resource ID}:

```HTTP
PUT /v1/services/a09c4585-0519-44f6-a5ab-647aeffc6281
/resources/c8924e50-af0d-47ae-bf5d-e184b1b59a60 HTTP/2
Host: eu-central-1.hapio.net
Accept: */*
Authorization: Bearer GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz
```

```PHP
use Hapio\Sdk\ApiClient;

$apiClient = new ApiClient('GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz');

$association = $apiClient->services()->associateResource(
    $service->id,
    $resource->id
);
```

```JavaScript
import axios from 'axios';

const apiClient = axios.create({
    baseURL: 'https://eu-central-1.hapio.net/v1/',
    headers: {'Authorization': 'Bearer
GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz'}
});

apiClient.put('services/' + service.id + '/resources/' +
resource.id).then(function (response) {
    const association = response.data;
});
```

The response includes information about this association:

```HTTP
HTTP/2 201
date: Thu, 24 Aug 2023 09:55:13 GMT
content-type: application/json
content-length: 213
x-ratelimit-limit: 100
x-ratelimit-remaining: 99
access-control-allow-origin: *
cache-control: no-cache, private
apigw-requestid: KKNYPiW6FiAEPSw=

{
    "service_id": "a09c4585-0519-44f6-a5ab-647aeffc6281",
    "resource_id": "c8924e50-af0d-47ae-bf5d-e184b1b59a60",
    "created_at": "2023-08-24T09:55:13+00:00",
    "updated_at": "2023-08-24T09:55:13+00:00"
}
```

Set up a schedule

The last thing we need to do is to set up a schedule for our doctor. This will tell Hapio when Dr. Smith (our resource) is available for bookings at our clinic. Let's assume Dr. Smith's working hours are Monday through Friday, 08:00 to 17:00, with lunch between 12:00 and 13:00. For this kind of working hours, a recurring schedule is the best option, since this lets us set up a schedule that repeats every week.

HAPIO

We start by creating a recurring schedule by send a **POST** request to the endpoint /v1/resources/{resource ID}/recurring-schedules:

```http
POST /v1/resources/c8924e50-af0d-47ae-bf5d-e184b1b59a60/recurring-schedules
HTTP/2
Host: eu-central-1.hapio.net
Accept: */*
Authorization: Bearer GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz
Content-Type: application/json
Content-Length: 93

{
    "location_id": "2b113cf6-5d6c-4a31-bd21-88ba7fb440ea",
    "start_date": "2023-08-24"
}
```

```php
use Hapio\Sdk\ApiClient;
use Hapio\Sdk\Models\RecurringSchedule;

$apiClient = new ApiClient('GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz');

$recurringSchedule = new RecurringSchedule();
$recurringSchedule->locationId = $location->id;
$recurringSchedule->start_date = '2023-08-24';

$recurringSchedule = $apiClient->recurringSchedules()->store(
    [$resource->id],
    $recurringSchedule
);
```

```javascript
import axios from 'axios';

const apiClient = axios.create({
    baseURL: 'https://eu-central-1.hapio.net/v1/',
    headers: {'Authorization': 'Bearer GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz'}
});
```

```
apiClient.post('resources/' + resource.id + '/recurring-schedules', {
    location_id: location.id,
    start_date: '2023-08-24'
}).then(function (response) {
    const recurringSchedule = response.data;
});
```

As you can see, the request includes the ID of our location, since every recurring schedule belongs to a specific location. As usual, the response include all information about the recurring schedule that we just created, including the location that it belongs to:

```
HTTP/2 201                                                    HTTP
date: Thu, 24 Aug 2023 10:20:42 GMT
content-type: application/json
content-length: 605
x-ratelimit-limit: 100
x-ratelimit-remaining: 99
access-control-allow-origin: *
cache-control: no-cache, private
apigw-requestid: KKRHFiBLFiAEM8g=

{
    "id": "e518071e-b4d6-4e98-a12f-04b44276e93c",
    "location": {
        "id": "2b113cf6-5d6c-4a31-bd21-88ba7fb440ea",
        "name": "Perfect Health - Stockholm",
        "time_zone": "Europe/Stockholm",
        "resource_selection_strategy": "equalize",
        "metadata": null,
        "protected_metadata": null,
        "enabled": true,
        "created_at": "2023-08-24T09:42:13+00:00",
        "updated_at": "2023-08-24T09:42:13+00:00"
    },
    "start_date": "2023-08-24",
    "end_date": null,
    "updated_at": "2023-08-24T10:20:42+00:00",
    "created_at": "2023-08-24T10:20:42+00:00"
}
```

HAPIO

Now, we can create schedule blocks in this recurring schedule. This is done by sending **POST** requests to the endpoint /v1/resources/{resource ID}/recurring-schedules/{recurring schedule ID}/schedule-blocks:

```http
POST /v1/resources/c8924e50-af0d-47ae-bf5d-e184b1b59a60/recurring-schedules/e518071e-b4d6-4e98-a12f-04b44276e93c/schedule-blocks HTTP/2
Host: eu-central-1.hapio.net
Accept: */*
Authorization: Bearer GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz
Content-Type: application/json
Content-Length: 85

{
    "weekday": "monday",
    "start_time": "08:00:00",
    "end_time": "12:00:00"
}
```

```php
use Hapio\Sdk\ApiClient;
use Hapio\Sdk\Models\RecurringScheduleBlock;

$apiClient = new ApiClient('GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz');

$recurringScheduleBlock = new RecurringScheduleBlock();
$recurringScheduleBlock->weekday = 'monday';
$recurringScheduleBlock->start_time = '08:00:00';
$recurringScheduleBlock->end_time = '12:00:00';

$recurringScheduleBlock = $apiClient->recurringScheduleBlocks()->store(
    [$resource->id, $recurringSchedule->id],
    $recurringScheduleBlock
);
```

```javascript
import axios from 'axios';

const apiClient = axios.create({
    baseURL: 'https://eu-central-1.hapio.net/v1/',
    headers: {'Authorization': 'Bearer GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz'}
});
```

```
apiClient.post('resources/' + resource.id + '/recurring-schedules/' +
recurringSchedule.id + '/schedule-blocks', {
    weekday: 'monday',
    start_time: '08:00:00',
    end_time: '12:00:00'
}).then(function (response) {
    const recurringScheduleBlock = response.data;
});
```

As you can see, our first schedule block in this recurring schedule is for Mondays from 08:00 to 12:00. The schedule block ends at 12:00, since that's when Dr. Smith's lunch starts, and the next schedule block should then start at 13:00. The response will, as usual, include all information about the created recurring schedule block:

```
HTTP/2 201                                                    HTTP
date: Thu, 24 Aug 2023 10:28:22 GMT
content-type: application/json
content-length: 229
access-control-allow-origin: *
cache-control: no-cache, private
x-ratelimit-limit: 100
x-ratelimit-remaining: 99
apigw-requestid: KKSPAj9rFiAEP6A=

{
    "id": "456da208-72d4-4ac0-b11c-324f4b5bcd34",
    "weekday": "monday",
    "start_time": "08:00:00",
    "end_time": "12:00:00",
    "created_at": "2023-08-24T10:28:22+00:00",
    "updated_at": "2023-08-24T10:28:22+00:00"
}
```

By continuing to create similar schedule blocks, we can fill up the entire week so that it reflects Dr. Smith's working hours.

HAPIO

Get bookable slots

Once we have our recurring schedule for the resource set up, we can retrieve bookable slots for the service that we created. We do this by sending a **GET** request to the endpoint /v1/services/{service ID}/bookable-slots:

```http
GET /v1/services/a09c4585-0519-44f6-a5ab-647aeffc6281/bookable-slots?from=2023-08-28T08:00:00%2B02:00&to=2023-08-28T10:00:00%2B02:00&location=2b113cf6-5d6c-4a31-bd21-88ba7fb440ea HTTP/2
Host: eu-central-1.hapio.net
Accept: */*
Authorization: Bearer GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz
```

```php
use Hapio\Sdk\ApiClient;

$apiClient = new ApiClient('GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz');

$page = $apiClient->services()->listBookableSlots(
    $service->id,
    [
        'from' => new DateTime('2023-08-28 08:00:00+02:00'),
        'to' => new DateTime('2023-08-28 10:00:00+02:00'),
        'location' => $location->id,
    ]
);
```

```javascript
import axios from 'axios';

const apiClient = axios.create({
    baseURL: 'https://eu-central-1.hapio.net/v1/',
    headers: {'Authorization': 'Bearer GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz'}
});

apiClient.get('services/' + service.id + '/bookable-slots', {
    params: {
        from: '2023-08-28T08:00:00+02:00',
        to: '2023-08-28T10:00:00+02:00',
        location: location.id
    }
```

```
}).then(function (response) {
    const page = response.data;
});
```

Note that we need to include a time window and a location ID in the query string of the request, since Hapio needs that information to determine when and where to look for bookable slots. It's important to remember to URL encode any special characters in the query string, such as the plus sign between the timestamp and the time zone offset.

The response includes a paginated list of bookable slots during the time window and at the location that was requested. In this example, the time window was from 08:00 to 10:00 on a monday, and since we have defined a schedule that is open during this time on mondays, we get two bookable slots, 08:00 to 08:50, and 09:00 to 09:50.

```
HTTP/2 200                                                                HTTP
date: Thu, 24 Aug 2023 14:19:07 GMT
content-type: application/json
content-length: 2302
cache-control: no-cache, private
x-ratelimit-limit: 100
x-ratelimit-remaining: 99
access-control-allow-origin: *
apigw-requestid: KK0COjkRFiAEMjw=

{
    "data": [
        {
            "buffer_starts_at": "2023-08-28T08:00:00+02:00",
            "starts_at": "2023-08-28T08:00:00+02:00",
            "ends_at": "2023-08-28T08:50:00+02:00",
            "buffer_ends_at": "2023-08-28T09:00:00+02:00",
            "resources": [
                {
                    "id": "c8924e50-af0d-47ae-bf5d-e184b1b59a60",
                    "name": "Dr. Smith",
                    "max_simultaneous_bookings": 1,
                    "metadata": null,
                    "protected_metadata": null,
                    "enabled": true,
                    "created_at": "2023-08-24T08:58:04+00:00",
                    "updated_at": "2023-08-24T08:58:04+00:00"
                }
```

```
            ]
        },
        {
            "buffer_starts_at": "2023-08-28T09:00:00+02:00",
            "starts_at": "2023-08-28T09:00:00+02:00",
            "ends_at": "2023-08-28T09:50:00+02:00",
            "buffer_ends_at": "2023-08-28T10:00:00+02:00",
            "resources": [
                {
                    "id": "c8924e50-af0d-47ae-bf5d-e184b1b59a60",
                    "name": "Dr. Smith",
                    "max_simultaneous_bookings": 1,
                    "metadata": null,
                    "protected_metadata": null,
                    "enabled": true,
                    "created_at": "2023-08-24T08:58:04+00:00",
                    "updated_at": "2023-08-24T08:58:04+00:00"
                }
            ]
        }
    ],
    "links": {
        "first": "https://eu-central-1.hapio.net/v1/services/a09c4585
    -0519-44f6-a5ab-647aeffc6281/bookable-slots?from=2023-08-28T08%3A
    00%3A00%2B02%3A00&location=2b113cf6-5d6c-4a31-bd21-88ba7fb440ea&to
    =2023-08-28T10%3A00%3A00%2B02%3A00&page=1",
        "last": "https://eu-central-1.hapio.net/v1/services/a09c4585-0
    519-44f6-a5ab-647aeffc6281/bookable-slots?from=2023-08-28T08%3A00%
    3A00%2B02%3A00&location=2b113cf6-5d6c-4a31-bd21-88ba7fb440ea&to=20
    23-08-28T10%3A00%3A00%2B02%3A00&page=1",
        "prev": null,
        "next": null
    },
    "meta": {
        "current_page": 1,
        "from": 1,
        "last_page": 1,
        "path": "https://eu-central-1.hapio.net/v1/services/a09c4585-05
        19-44f6-a5ab-647aeffc6281/bookable-slots",
        "per_page": 100,
        "to": 2,
        "total": 2
    }
}
```

HAPIO

Create your first booking

Now that everything is set up, and you have made a request to fetch bookable slots in order to find out when the doctor is available, you are ready to create your first booking. Do this by sending a **POST** request to the endpoint /v1/bookings:

```http
POST /v1/bookings HTTP/2
Host: eu-central-1.hapio.net
Accept: */*
Authorization: Bearer GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz
Content-Type: application/json
Content-Length: 209

{
    "location_id": "2b113cf6-5d6c-4a31-bd21-88ba7fb440ea",
    "service_id": "a09c4585-0519-44f6-a5ab-647aeffc6281",
    "starts_at": "2023-08-28T08:00:00+02:00",
    "ends_at": "2023-08-28T08:50:00+02:00"
}
```

```php
use Hapio\Sdk\ApiClient;
use Hapio\Sdk\Models\Booking;

$apiClient = new ApiClient('GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz');

$booking = new Booking();
$booking->locationId = $location->id;
$booking->serviceId = $service->id;
$booking->startsAt = new DateTime('2023-08-28 08:00:00+02:00');
$booking->endsAt = new DateTime('2023-08-28 08:50:00+02:00');

$booking = $apiClient->bookings()->store($booking);
```

```javascript
import axios from 'axios';

const apiClient = axios.create({
    baseURL: 'https://eu-central-1.hapio.net/v1/',
    headers: {'Authorization': 'Bearer
GsAnhHagTllG9nniTAcwZWd4nUZXnGpAb9Ywyrxz'}
});
```

```
apiClient.post('bookings', {
    location_id: location.id,
    service_id: service.id,
    starts_at: '2023-08-28T08:00:00+02:00',
    ends_at: '2023-08-28T08:50:00+02:00'
}).then(function (response) {
    const booking = response.data;
});
```

In this case, we only include the minimum amount of information needed to create a booking, namely the location, the service, and the start and end timestamps. We let Hapio automatically fill in the rest, including the resource (according to the resource selection strategy of the location).

As usual, the response include all information about the booking that you've just created:

```
HTTP/2 201                                                    HTTP
date: Thu, 24 Aug 2023 14:36:20 GMT
content-type: application/json
content-length: 1900
cache-control: no-cache, private
x-ratelimit-limit: 100
x-ratelimit-remaining: 99
access-control-allow-origin: *
apigw-requestid: KK2jphEuliAEM4A=

{
    "id": "b3f06e22-2e2e-44a4-a9e5-7ed24b0a06a1",
    "resource": {
        "id": "c8924e50-af0d-47ae-bf5d-e184b1b59a60",
        "name": "Dr. Smith",
        "max_simultaneous_bookings": 1,
        "metadata": null,
        "protected_metadata": null,
        "enabled": true,
        "created_at": "2023-08-24T08:58:04+00:00",
        "updated_at": "2023-08-24T08:58:04+00:00"
    },
    "service": {
        "id": "a09c4585-0519-44f6-a5ab-647aeffc6281",
        "name": "Physical Exam",
        "price": "149.000",
```

```
        "type": "fixed",
        "duration": "PT50M",
        "bookable_interval": "PT1H",
        "buffer_time_before": "PT0S",
        "buffer_time_after": "PT10M",
        "booking_window_start": "PT2H",
        "booking_window_end": "P14D",
        "cancelation_threshold": "PT12H",
        "metadata": null,
        "protected_metadata": null,
        "enabled": true,
        "created_at": "2023-08-24T09:07:53+00:00",
        "updated_at": "2023-08-24T09:07:53+00:00"
    },
    "location": {
        "id": "2b113cf6-5d6c-4a31-bd21-88ba7fb440ea",
        "name": "Perfect Health - Stockholm",
        "time_zone": "Europe/Stockholm",
        "resource_selection_strategy": "equalize",
        "metadata": null,
        "protected_metadata": null,
        "enabled": true,
        "created_at": "2023-08-24T09:42:13+00:00",
        "updated_at": "2023-08-24T09:42:13+00:00"
    },
    "price": "149.000",
    "metadata": null,
    "protected_metadata": null,
    "is_temporary": false,
    "is_canceled": false,
    "starts_at": "2023-08-28T08:00:00+02:00",
    "ends_at": "2023-08-28T08:50:00+02:00",
    "buffer_starts_at": "2023-08-28T08:00:00+02:00",
    "buffer_ends_at": "2023-08-28T09:00:00+02:00",
    "created_at": "2023-08-24T14:36:20+00:00",
    "updated_at": "2023-08-24T14:36:20+00:00",
    "finalized_at": "2023-08-24T14:36:20+00:00",
    "canceled_at": null
}
```


That's it! You have now gone through all the necessary steps to create your first booking in Hapio.