The EasyWay's API is a JSON-based API. All requests are made to endpoints beginning with: '<http://localhost:3000/api>'. Requests can be made with http.

## 

## **1. getServices**

This API endpoint retrieves the list of services that the end-user can avail.

*Request type:* GET

*Output type:* JSON Array

*Sample request:* <http://localhost:3000/api/getServices>

*Sample output:*

| [   {    "id": 1,    "name": "AC Maintanence",    "description": "Any type of AC maintanence such as filter cleaning, part replacement, etc.",    "category": "Electronics",    "image\_name": "air\_conditioning.jpg",    "price": 80   },   {    "id": 2,    "name": "Plumbing",    "description": "Sanitary and household plumbing. No sewage service.",    "category": "Household",    "image\_name": "plumbing.jpg",    "price": 100   },   {    "id": 3,    "name": "Saloon",    "description": "Haricut, massage, nailwork, makeup, etc.",    "category": "Personal Care",    "image\_name": "saloon.jpg",    "price": 25   } ] |
| --- |

*Service DB Schema:*

| **Parameter** | **Type** | **Details** |
| --- | --- | --- |
| id | Integer | Unique identifier for the service. PRIMARY KEY |
| name | String | Name of the service. |
| description | String | Description of the service. |
| category | String | Category of the service. |

## 

## **2. createService**

This API endpoint creates a new service as specified by the end-user. This is a POST request. This is sent with the default "Content-Type" header of "application/x-www-form-urlencoded".

*Request type:* POST

*Input body type:* JSON Object

*Output type:* JSON Object

*Sample request:* <http://localhost:3000/api/createService>

*Sample input:*

| {   "name": "Tutoring",   "description": "Take help in assignments and more.",   "category": "Home tution",   "price": 15 } |
| --- |

*Sample output:*

| {   "id": 6,   "name": "Tutoring",   "description": "Take help in assignments and more.",   "category": "Home tution",   "image\_name": "default.jpg",   "price": 15 } |
| --- |

## 

## **3. register**

This API endpoint registers a new user. This is sent with the default "Content-Type" header of "application/x-www-form-urlencoded"

*Request type:* POST

*Input body type:* JSON Object

*Output type:* JSON Object

*Sample request:* <http://localhost:3000/api/register>

*Sample input:*

| {   "name": "alex",   "email": "alex@ufl.edu",   "gender": "F",   "username": "alex",   "password": "alex1" } |
| --- |

*Sample output:*

| {   "id": 4,   "name": "alex",   "username": "alex",   "password": "alex1",   "email": "alex@ufl.edu",   "gender": "F" } |
| --- |

*User DB Schema*

| **Parameter** | **Type** | **Details** |
| --- | --- | --- |
| id | Integer | Unique identifier for the user. PRIMARY KEY |
| name | String | Name of the user. Must contain First and  Last name separated by space. |
| username | String | Username of the user. |
| email | String | Email ID of the user. Must be of the form  'xxxxx@xxxxxx' |
| gender | String | Gender of the user. Can be either 'M' or 'F'. |
| password | String | Password for the user. Has to be between 7 to  14 characters in length and must contain  atleast one lower case character, one upper  case character and one number. |

## 

## **4. login**

This API endpoint logs in a new user. This request is sent with the default "Content-Type" header of "application/x-www-form-urlencoded". If the user credentials match with that in the database, it returns the user details in JSON format else it returns a *404 Not Found* error.

*Request type:* POST

*Input body type:* JSON Object

*Output type:* JSON Object

*Sample request:* <http://localhost:3000/api/login>

*Sample input:*

| {   "username": "dummy",   "password": "dumdum" } |
| --- |

*Sample output:*

| {   "id": 1,   "name": "Dummy Duck",   "username": "dummy",   "password": "dumdum",   "email": "dummy@ufl.edu",   "gender": "M" } |
| --- |

## 

## **5. getBookings**

This API endpoint retrieves the list of services booked by a user. This request is sent with the parameter *userId* which is the *id* field of the user. Returns a *404 Not Found* error if there are no bookings or if the *userId* is invalid.

*Request type:* GET

*Output type:* JSON Array

*Sample request:* <http://localhost:3000/api/getBookings?userId=1>

*Sample output:*

| [   {    "id": 1,    "user\_id": 1,    "service\_id": 1,    "date": "2022-02-15",    "start\_time": "12:30",    "end\_time": "13:30",    "is\_cancelled": false   },   {    "id": 2,    "user\_id": 1,    "service\_id": 2,    "date": "2022-02-15",    "start\_time": "16:30",    "end\_time": "17:30",    "is\_cancelled": false   } ] |
| --- |

*Bookings DB Schema*

| **Parameter** | **Type** | **Details** |
| --- | --- | --- |
| id | Integer | Unique identifier for the service booking. PRIMARY KEY |
| user\_id | Integer | Unique identifier for the user who booked this particular service. |
| service\_id | Integer | Unique identifier for the service booked by the user. |
| date | String | Date on which the booking was made by the user. Format: "YYYY-MM-DD". |
| start\_time | String | Time at which the service booking commences. Format: "HH:MM". |
| end\_time | String | Time at which the service booking ends Format: "HH:MM". |
| is\_cancelled | Boolean | Cancellation status of the booking. Can be true or false. |

## 

## **6. bookService**

This API endpoint creates a new booking for a service by the end-user. This request is sent with the default "Content-Type" header of "application/x-www-form-urlencoded". It returns a *500 Internal Server Error* status with the message *Time slot unavailable* if it is booked by another user for the same time slot.

*Request type:* POST

*Input body type:* JSON Object

*Output type:* JSON Object

*Sample request:* <http://localhost:3000/api/bookService>

*Sample input:*

| {   "user\_id": 1,   "service\_id": 2,   "date": "2022-02-23",   "start\_time": "11:30",   "end\_time": "12:30" } |
| --- |

*Sample output:*

| {   "id": 4,   "user\_id": 1,   "service\_id": 2,   "date": "2022-02-23",   "start\_time": "11:30",   "end\_time": "12:30",   "is\_cancelled": false } |
| --- |

## 

## **7. cancelBooking**

This API endpoint cancels a particular booking for a service by the end-user. This request is sent with the default "Content-Type" header of "application/x-www-form-urlencoded". The request either sets the *is\_cancelled* field of the booking to *true* and returns a *200 OK* status along with the message *Booking is cancelled* or if the booking has already been cancelled it returns a *Booking already cancelled* message.

*Request type:* GET

*Output type:* String

*Sample request:* <http://localhost:3000/api/cancelBooking?id=1>

*Sample output:*

| "Booking is cancelled" |
| --- |

## 

## **8. getCancelledBookings**

This API endpoint retrieves the list of cancelled services by the end-user. This matches the *user\_id* of the parameter in the request URL and finds Bookings which have *is\_cancelled* field set to *true*. If the *user\_id* field cannot be found, it returns a *404 Not Found* error.

*Request type:* GET

*Output type:* JSON Array

*Sample request:* <http://localhost:3000/api/getCancelledBookings?userId=1>

*Sample output:*

| [   {    "id": 1,    "user\_id": 1,    "service\_id": 1,    "date": "2022-02-15",    "start\_time": "12:30",    "end\_time": "13:30",    "is\_cancelled": true   } ] |
| --- |

## 

## **9. getServiceInfo**

This API endpoint returns information about a service when given its *id*.

*Request type:* GET

*Output type:* JSON Object

*Sample request:* <http://localhost:3000/api/getServiceInfo?serviceId=1>

*Sample output:*

| {   "id": 1,   "name": "AC Maintanence",   "description": "Any type of AC maintanence such as filter cleaning, part replacement, etc.",   "category": "Electronics",   "image\_name": "air\_conditioning.jpg",   "price": 80 } |
| --- |