

Project 1 - (User guide) BTP405

By: [Kabeer Harjani]

Course: [System Development and Design]

Professor: [Maziar Sojoudian]

Date due: [2024/03/07, 11:59PM]

User Guide on how to use the PHR (Personal Health Record) System:

Getting started:

- **Register:** To start using the PHR system you must register for an account. To do this send a POST request to the /register endpoint or go to <http://localhost:8000/register> after starting the application. Register with a “username” and “password” field.
- **Log In:** After you have registered for an account then you can login by sending a request to any endpoint after <http://localhost:8000>. In the Authorization header enter your “username” and “password” value. If the values are not correct then you will be returned with an “Unauthorized” response.

Managing Records:

- **Adding a record:**

Set the HTTP method to `POST`.

Enter the request URL as `http://localhost:8000/records`.

Click on `Body`, then select `raw` and `JSON`.

In the text field, enter your record in the following format and adjust the values as you like:

```
{  
  
  "record_id": 2,  
  
  "full_name": "John Doe",  
  
  "dob": "2000-01-01",  
  
  "sex": "M",  
  
  "allergies": "None",  
  
  "medications": "None",  
  
  "diagnosis": "None",  
}
```

```
    "treatment": "None",

    "notes": "None"

}
```

Click on **Send** to make the request.

Status code: 200 -> record successfully created

Status code: 404 -> error

The screenshot shows a REST client interface with the following elements:

- URL Bar:** `http://localhost:8000/records` with a `Save` button and a dropdown arrow.
- Method and URL:** `POST` method selected, with the URL `http://localhost:8000/records` in a text box. A blue **Send** button is to the right.
- Tabs:** `Params`, `Auth` (with a green dot), `Headers (9)`, `Body` (with a green dot and underline), `Pre-req.`, `Tests`, and `Settings`.
- Body Format:** A dropdown menu shows `raw` and `JSON` (selected). A **Beautify** button is on the right.
- Body Content:** A code editor with line numbers 1-12 containing the following JSON:

```
1  {
2    "record_id": 2,
3    "full_name": "John Doe",
4    "dob": "2000-01-01",
5    "sex": "M",
6    "allergies": "None",
7    "medications": "None",
8    "diagnosis": "None",
9    "treatment": "None",
10   "notes": "None"
11 }
12
```
- Status Bar:** At the bottom, it shows `Body` with a dropdown, a globe icon, `200 OK`, `10 ms`, `94 B`, a `Save as example` button, and a menu icon.

- Viewing all records:

Set the HTTP method to `GET`.

Enter the request URL as `http://localhost:8000/`.

Click on `Send` to make the request.

Status code: 200 -> record successfully found

Status code: 404 -> error

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:8000/records`
- Method:** `GET`
- Send Button:** A blue button labeled "Send".
- Response Status:** 200 OK, 6 ms, 518 B.
- Response Body:** A JSON array containing two record objects. The first object has `record_id: 1` and the second has `record_id: 2`. Both records have the same other fields: `user_id: null`, `full_name: "John Doe"`, `dob: "2000-01-01"`, `sex: "M"`, `allergies: "None"`, `medications: "None"`, `diagnosis: "None"`, `treatment: "None"`, and `notes: "None"`.

```
1 [
2   {
3     "user_id": null,
4     "record_id": 1,
5     "full_name": "John Doe",
6     "dob": "2000-01-01",
7     "sex": "M",
8     "allergies": "None",
9     "medications": "None",
10    "diagnosis": "None",
11    "treatment": "None",
12    "notes": "None"
13  },
14  {
15    "user_id": null,
16    "record_id": 2,
17    "full_name": "John Doe",
18    "dob": "2000-01-01",
19    "sex": "M",
20    "allergies": "None",
21    "medications": "None",
22    "diagnosis": "None",
23    "treatment": "None",
24    "notes": "None"
25  }
26 ]
```

- Viewing a specific record

Set the HTTP method to `GET`.

Enter the request URL as `http://localhost:8000/records/:record_id`.
#change `record_id` with the id you want

Click on `Send` to make the request.

Status code: 200 -> record successfully found

Status code: 404 -> error

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:8000/records`
- Method:** `GET`
- URL:** `http://localhost:8000/records/1`
- Send Button:** A blue button labeled "Send".
- Params:** A tab labeled "Params" is selected.
- Auth:** A green dot indicates authentication is enabled.
- Headers (7):** A tab labeled "Headers (7)".
- Body:** A tab labeled "Body" is selected.
- Pre-req. Tests Settings:** Tabs for "Pre-req.", "Tests", and "Settings".
- raw JSON:** A dropdown menu showing "raw" and "JSON".
- Beautify:** A blue button labeled "Beautify".
- Body:** A tab labeled "Body" is selected.
- Status:** `200 OK`, `6 ms`, `322 B`.
- Save as example:** A button labeled "Save as example".
- JSON:** A dropdown menu showing "JSON".
- Pretty Raw Preview Visualize:** Tabs for "Pretty", "Raw", "Preview", and "Visualize".
- JSON:** A dropdown menu showing "JSON".
- Copy Search:** Buttons for "Copy" and "Search".
- Response Body:** A JSON object representing a record with the following fields:

```
1 [
2   {
3     "user_id": null,
4     "record_id": 1,
5     "full_name": "John Doe",
6     "dob": "2000-01-01",
7     "sex": "M",
8     "allergies": "None",
9     "medications": "None",
10    "diagnosis": "None",
11    "treatment": "None",
12    "notes": "None"
13  }
14 ]
```

- Updating a specific record

Set the HTTP method to `PUT`.

Enter the request URL as `http://localhost:8000/records/:record_id`.

Click on `Body`, then select `raw` and `JSON`.

In the text field, enter your record in the following format:

```
{  
  
  "record_id": 3,  
  
  "full_name": "Changed",  
  
  "dob": "2000-01-01",  
  
  "sex": "M",  
  
  "allergies": "None",  
  
  "medications": "None",  
  
  "diagnosis": "None",  
  
  "treatment": "None",  
  
  "notes": "None"  
  
}
```

Click on `Send` to make the request.

Status code: 200 -> record successfully updated

Status code: 404 -> error

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:8000/records`
- Method:** `PUT`
- Path:** `http://localhost:8000/records/3`
- Body:** A JSON object with the following fields:

```
{  "user_id": null,  "record_id": 3,  "full_name": "Changed",  "dob": "2000-01-01",  "sex": "M",  "allergies": "None",  "medications": "None",  "diagnosis": "None",  "treatment": "None",  "notes": "None"}
```
- Response:** `200 OK`, `15 ms`, `94 B`
- Buttons:** `Save`, `Send`, `Beautify`, `Save as example`

- **Deleting a specific record**

Set the HTTP method to `DELETE`.


Enter the request URL as `http://localhost:8000/records/:record_id`.


Where the `:record_id` is the id of the record you are trying to delete


Click on `Send` to make the request.


Status code: `200` -> record successfully deleted


Status code: `404` -> error


 http://localhost:8000/records

 Save











DELETE




http://localhost:8000/records/3

Send



Params

Auth 


Headers (7)

Body


Pre-req.

Tests

Settings



none



This request does not have a body

Body 

 200 OK 29 ms 94 B  Save as example 