User manual for Postman – the tool for API tests

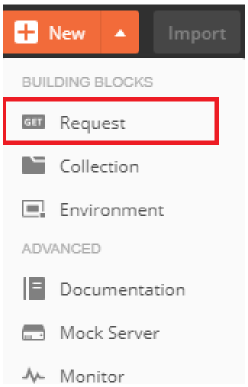
Content

I. Introduction and assumption

1. It is required to set up the Postman application (version 7.1.1) in order to use this manual

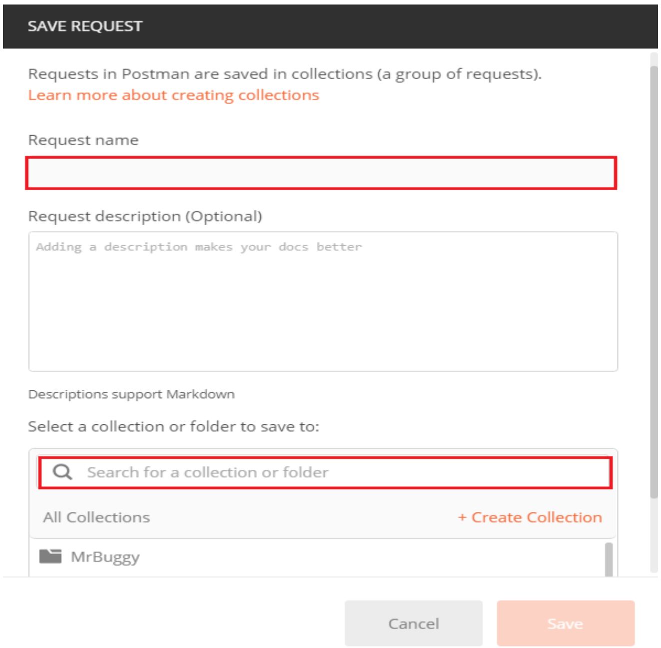
II. Guidelines for new request creation in Postman application

1. Select New (the left corner of an upper bar) -> Request



1. Postman – creating the new request

2. Fill in the 'Request name' field and choose from 'Select a collection or folder to save to' an existing collection or create a new one)



SAVE REQUEST

Requests in Postman are saved in collections (a group of requests).
Learn more about creating collections

Request name

Request description (Optional)

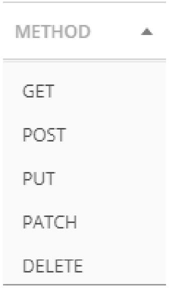
Adding a description makes your docs better

Descriptions support Markdown

Select a collection or folder to save to:

2. Postman – creating a new request form

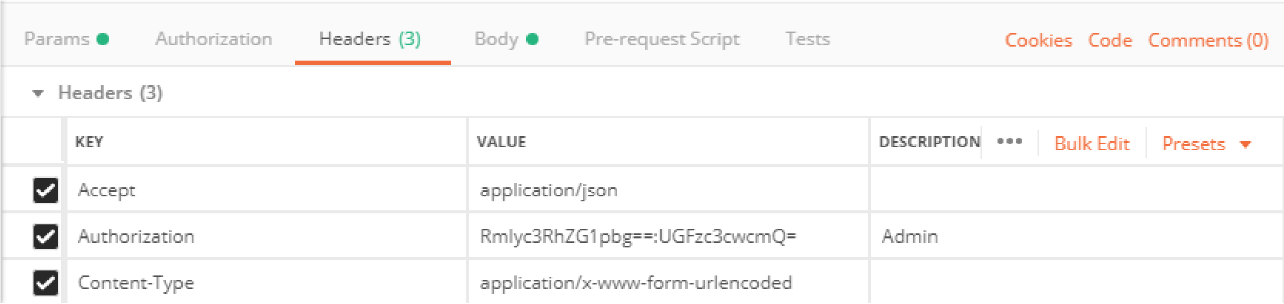
3. Click: Save
4. Choose one of the methods: GET, POST, PUT, DELETE



3. HTTP method list to be chosen from

5. Fill in the URL address with 'http://localhost:8080/', which is the hosts address from API that has to be filled with the appropriate path to the resource.
6. Fill in the Headers tab:
Accept: application/json
Content-Type: application/x-www-form-urlencoded
Authorization: username:password, while 'username' and 'password' have to be replaced with the users' login data created and saved in Base64 🛠 (icon to Encode username and password is available in MrBuggy application). As an example, in the Postman's Headers tab in the KEY column enter 'Authorization', and in the 'VALUE' column enter 'Rmlyc3RhZG1pbg==:UGFzc3cwcmQ=' (this is encoded data for username 'Firstadmin' and password 'Passw0rd'). Remember that encoded username and password must be separated by ':'.

NOTE: In Authorization, the login data, where username and password shall be the same as while logging into the MrBuggy application. For the Authorization for a specific request for example creating a new CR, username and password will be entered by the user with the Employee role, because this role is able to create a new CR.

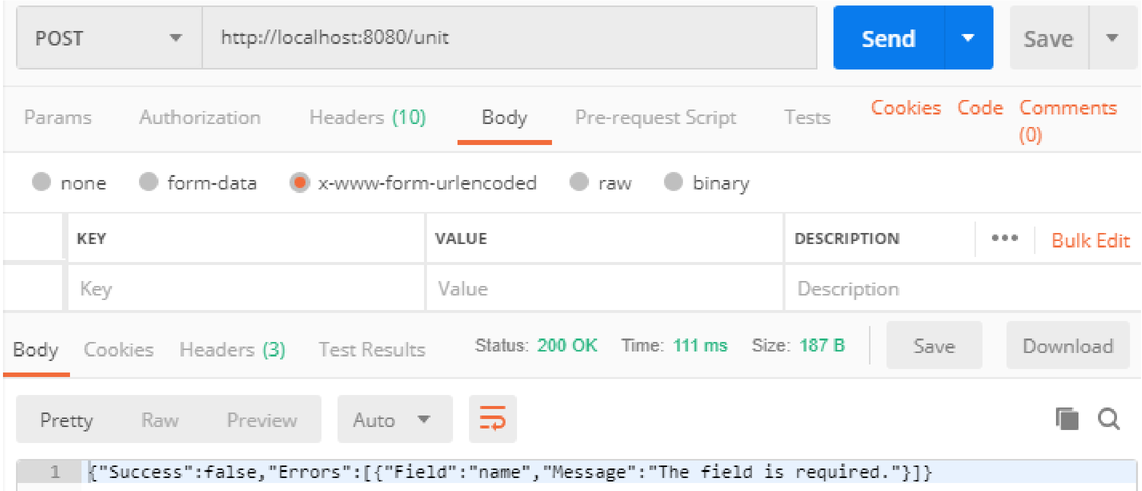


	KEY	VALUE	DESCRIPTION
☑	Accept	application/json	
☑	Authorization	Rmlyc3RhZG1pbg==:UGFzc3cwcmQ=	Admin
☑	Content-Type	application/x-www-form-urlencoded	

4. Postman – the correct complement of the parameters of requests header in 'Headers' tab for 'Firstadmin' user with password 'Passw0rd'

7. Fill in the Body tab by marking radiobutton 'x-www-form-urlencoded' and by entering the proper parameters

NOTE: The parameters for the specific request can be obtained by sending a request with POST or PUT method (not in all requests).
8. Click: Send



POST http://localhost:8080/unit

Body: ● x-www-form-urlencoded

Status: 200 OK Time: 111 ms Size: 187 B

```
{"Success":false,"Errors":[{"Field":"name","Message":"The field is required."}]}
```

5. Postman – an example of the request for creating the new unit with unfilled parameters and the response message

After sending the request the response will show up in the response section below. If the answer "Unauthorized" will appear after sending the request it means that the header in 'Headers' tab has been incorrectly filled in.

III. API Reference

http://localhost:8080

6. Host address and endpoint port

Resources types

1. [USER]
API methods:

GET
/user – users list
/user/ID – returns user data based on the provided number
/user/profile – data of logged in user

POST
/user/TYPE – creates new user of given TYPE (admin, employee, provider). Parameters that needs to be provided in HTTP body request:
- 'first_name',
- 'username',
- 'last_name',
- 'email',
- 'phone_number',
- 'password',
- 'unit_id'/'provider_id' required if new user has one of the 'employee'/'provider' role.

PUT
/user/ID – change user data based on the provided number. Parameters that needs to be provided in HTTP body request:
- 'first_name',
- 'username',

- 'last_name',
- 'email',
- 'phone_number',
- 'password',
- 'unit_id'/'provider_id' (depends on edited user 'employee'/'provider' role);

/user/profile – change data of currently logged in user. Parameters that needs to be provided in HTTP body request:

- 'first_name',
- 'username',
- 'last_name',
- 'email',
- 'phone_number',
- 'password',
- 'unit_id'/'provider_id' (depends on edited user 'employee'/'provider' role);

/user/ID/active – change user status to ACTIVE based on the provided number;
/user/ID/block - change user status to BLOCKED based on the provided number;

DELETE
/user/ID – delete user based on the provided number

2. [UNIT]
API methods:
GET
/unit – units list
/unit/ID – unit data based on the provided number

POST
/unit – new unit. Parameters that needs to be provided in HTTP body request:
- 'name';

PUT
/unit/ID – change data of unit based on the provided number. Parameters that needs to be provided in HTTP body request:
- 'name’;

DELETE
/unit/ID – delete unit based on the provided number.

3. [PROVIDER]
API methods:

GET
/provider – providers list
/provider/ID – provider data based on the provided number

POST
/provider – new provider. Parameters that needs to be provided in HTTP body request:
- 'name',
- 'price';

PUT

/provider/ID – change provider data based on the provided number. Parameters that needs to be provided in HTTP body request:
- 'name',
- price;

DELETE
/provider/ID – delete provider based on the provided number.

4. [CR]
API methods:

GET
/cr – CR list
/cr/ID – CR details based on the provided number
/cr/ID/comments – CR comments based on the provided number
/cr/ID/history – CR history based on the ID number
/cr/ID/related – shows related CRs based on the provided number

POST
/cr - new cr. Parameters that needs to be provided in HTTP body request:
- 'number',
- 'name',
- 'description',
- 'reason',
- 'financial_sources',
- 'provider_id';

/cr/ID/clone – clone CR based on the provided ID number. Parameters that needs to be provided in HTTP body request:
- 'number',
- 'name',
- 'description',
- 'reason',
- 'financial_sources',
- 'provider_id';

/cr/ID/comment – add comment to CR based on the provided ID number. Parameters that needs to be provided in HTTP body request:
- 'content'
- 'private' - if the comment should be private (value: true/false)

PUT
/cr/ID – change data of given CR based on the ID number. Parameters that needs to be provided in HTTP body request:
- 'number',
- 'name',
- 'description',
- 'reason',
- 'financial_sources',
- 'provider_id';

/cr/ID/STATUS – change CR status with the given ID number to STATUS(new, approved, forwarded, to-order, process, finished, archived, received);
/cr/ID/reject – change CR status with the given ID number to STATUS Rejected with comment. Parameters that needs to be provided in HTTP body request:
- 'content';

/cr/ID/to-clarify – change CR status with the given ID number to STATUS ToClarify with comment. Parameters that needs to be provided in HTTP body request:
- 'content';

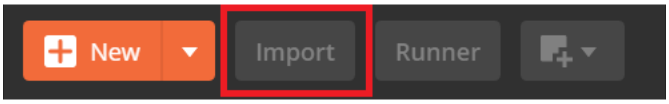
/cr/ID/complete – change CR status with the given ID number to Completed with all amendments and comment. Parameters that needs to be provided in HTTP body request:
- 'number',
- 'name',
- 'description',
- 'reason',
- 'financial_sources',
- 'provider_id'
- 'content';

/cr/ID/valuation – change CR status with the given ID number to Priced with man-days value. Parameters that needs to be provided in HTTP body request:
- 'man_days';

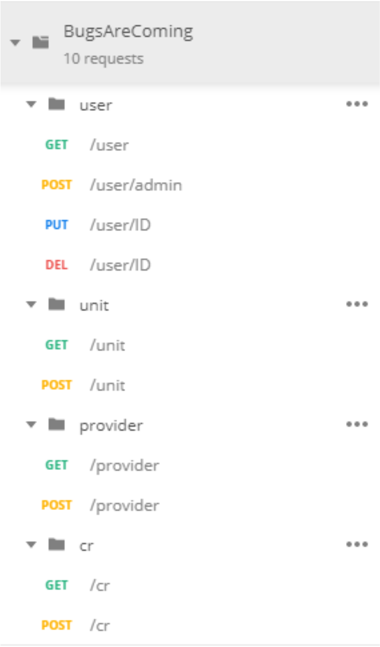
DELETE
/cr/ID – delete CR with given ID number.

IV. Import sample collection and environment

In order to import sample collection of requests You have to click Import icon and then pick the file 'BugsAreComing.postman_collection.json'.



7. Collection import


BugsAreComing
10 requests

user
GET /user
POST /user/admin
PUT /user/ID
DEL /user/ID

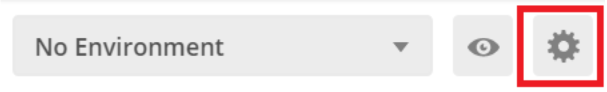
unit
GET /unit
POST /unit

provider
GET /provider
POST /provider

cr
GET /cr
POST /cr

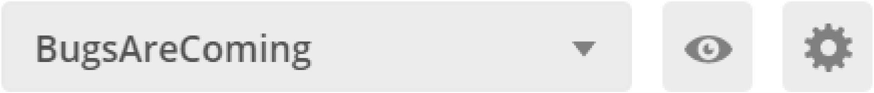
8. Imported collection

The second thing that need to be done is to import environment variables. You have to click 'Manage environments' (settings icon) in the right top of the screen and then click 'Import'. Choose the 'BugsAreComing.postman_environment.json' file.



9. Environment import icon

After successful import both collection 'BugsAreComing' end environment 'BugsAreComing' please ensure that you have chosen newly created environment from the dropdown list in the right top.

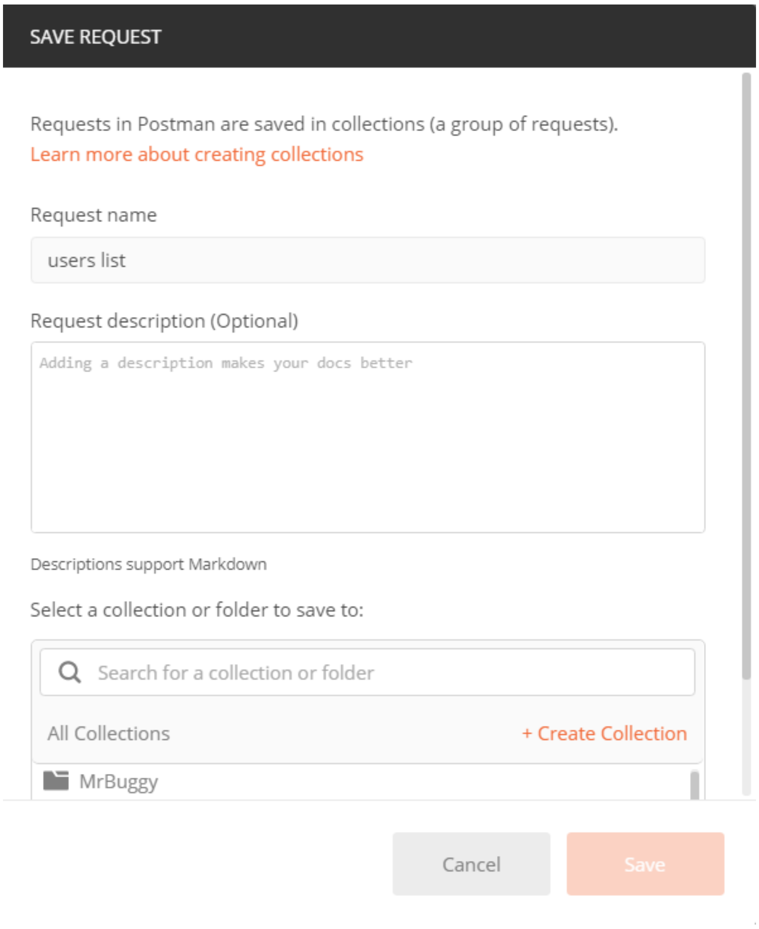


10. Environment imported

ATTENTION: In Authorization (with login data available in file BugsAreComing.postman_environment.json) must be existing user in MrBuggy application or you can change login data of existing user.

V. Create sample requests

1. Request GET /user (get user list)
A. Choose: New -> Request
B. Fill fields: 'Request name': 'users list' and choose from 'Select a collection or folder to save to' an existing collection or create a new one)



Requests in Postman are saved in collections (a group of requests).
Learn more about creating collections

Request name: users list

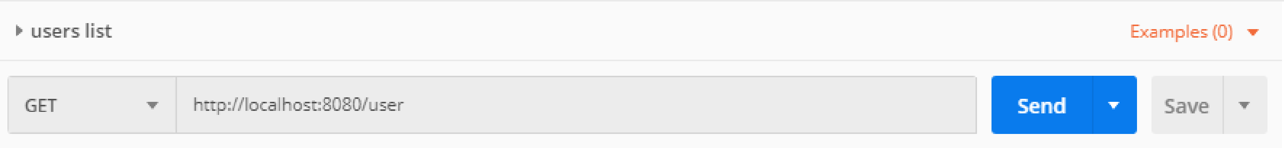
Request description (Optional): Adding a description makes your docs better

Descriptions support Markdown

Select a collection or folder to save to:

11. Request – users list

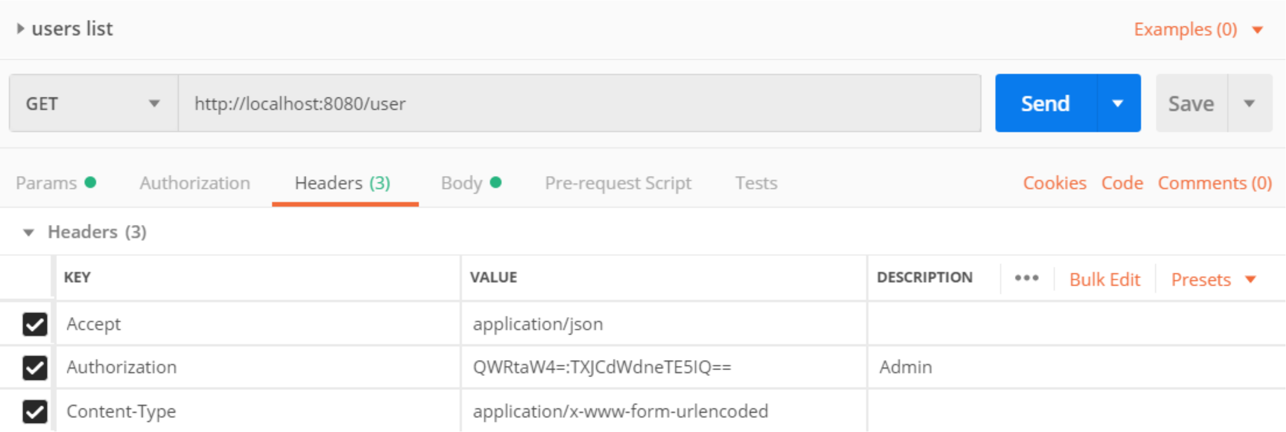
C. Click: Save to MrBuggy
D. Choose method: GET
E. Complete address URL: http://localhost:8080/user



GET http://localhost:8080/user

12. Address URL – users list

F. Complete Headers tab:
Authorization: username:password - 'QWRtaW4=:TXJCdWdneTE5IQ==' (username – Admin, password - MrBuggy19!)
Accept: 'application/json'
Content-Type: 'application/x-www-form-urlencoded'



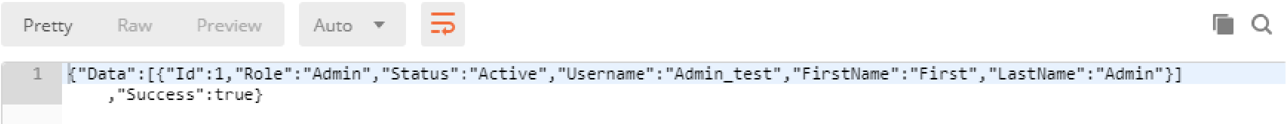
GET http://localhost:8080/user

Headers (3)

	KEY	VALUE	DESCRIPTION
☑	Accept	application/json	
☑	Authorization	QWRtaW4=:TXJCdWdneTE5IQ==	Admin
☑	Content-Type	application/x-www-form-urlencoded	

13. Headers – users list

G. Body tab -> x-www-form-urlencoded – get user list parameters not required
H. Click: Send
The answer request will be visible in response section

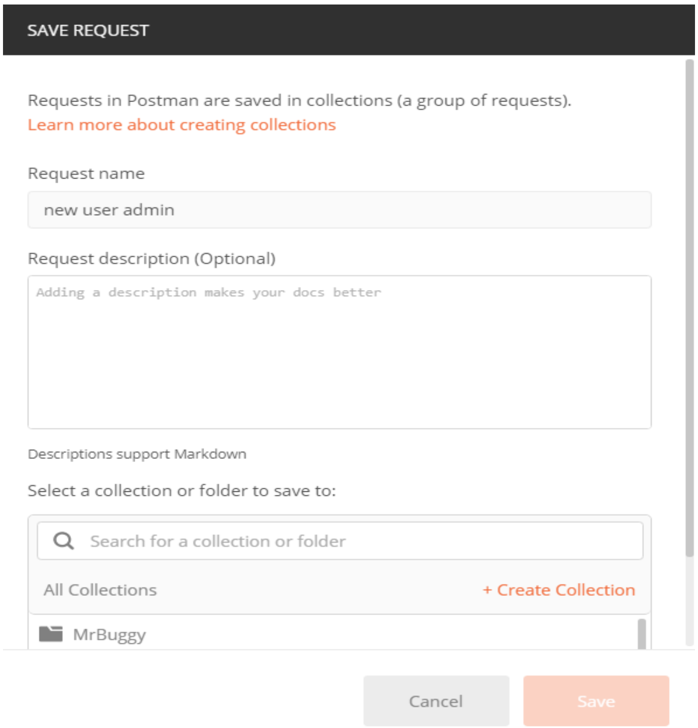


```
{"Data":[{"Id":1,"Role":"Admin","Status":"Active","Username":"Admin_test","FirstName":"First","LastName":"Admin"}]
    ,"Success":true}
```

14. Response section - users list

2. Request POST / user/admin (post new user admin)

A. Select New -> Request
B. Fill fields: 'Request name': 'new user admin' and select with 'Select a collection or folder to save to': 'MrBuggy'



SAVE REQUEST

Requests in Postman are saved in collections (a group of requests).
Learn more about creating collections

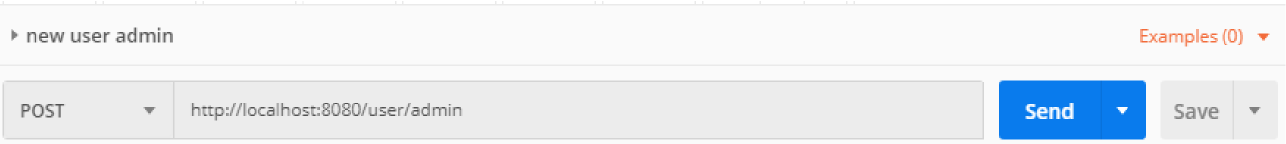
Request name: new user admin

Request description (Optional)

Adding a description makes your docs better
Descriptions support Markdown
Select a collection or folder to save to:
Search for a collection or folder
All Collections
+ Create Collection
MrBuggy
Cancel
Save

15. Request – new user admin

C. Click: Save to MrBuggy
D. Select method: POST
E. Complete address URL: http://localhost:8080/user/admin

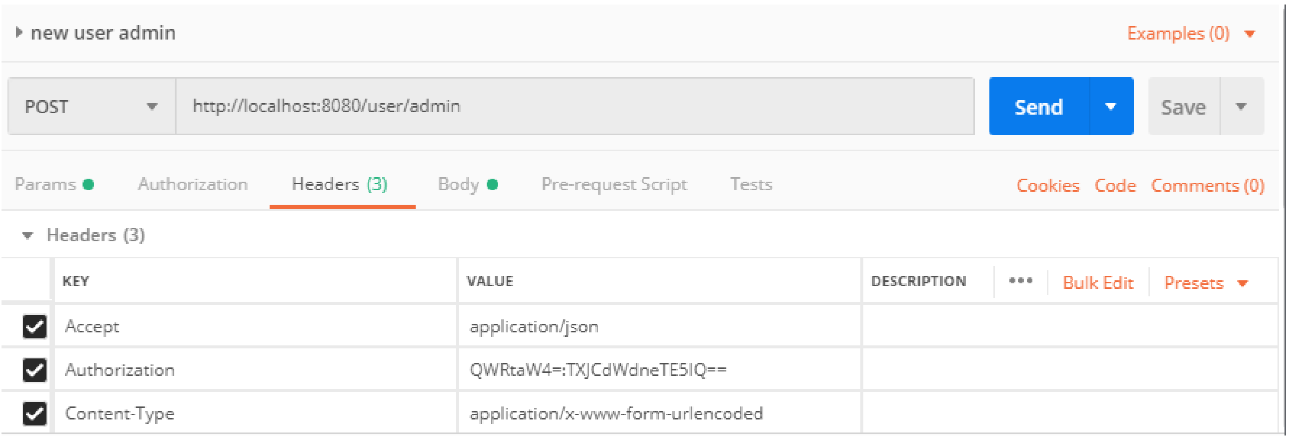


new user admin

POST http://localhost:8080/user/admin

16. Address URL – new user admin

F. Complete Headers tab:
Authorization: username:password - 'QWRtaW4=:TXJCdWdneTE5IQ==' (username - Admin, password - MrBuggy19!)
Accept: 'application/json'
Content-Type: 'application/x-www-form-urlencoded'

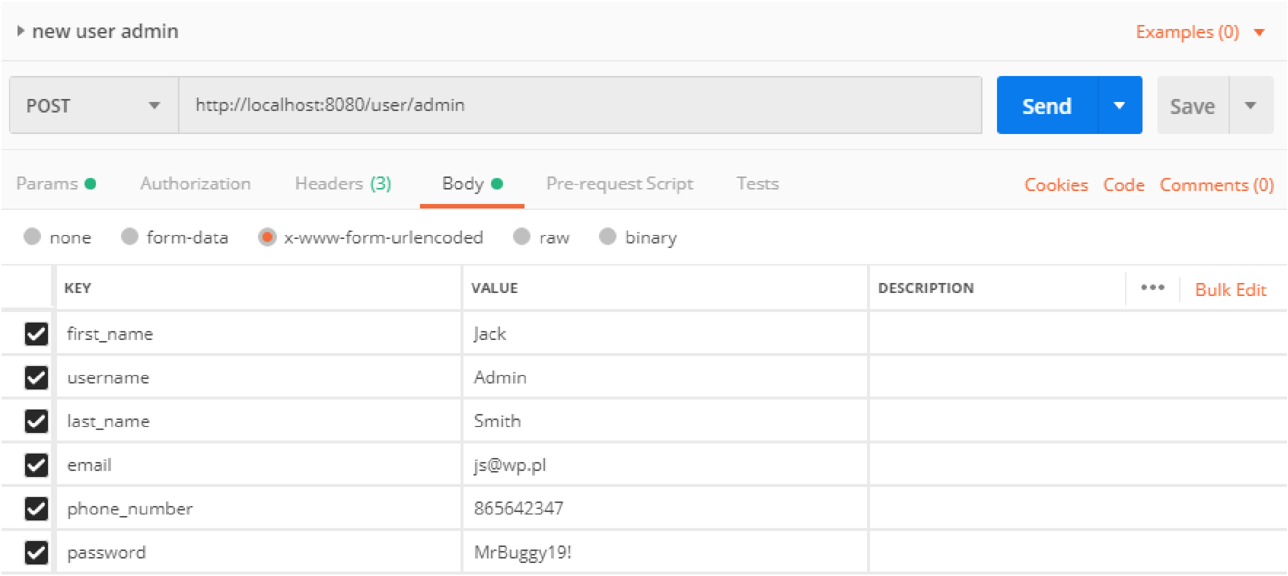


POST http://localhost:8080/user/admin

	KEY	VALUE	DESCRIPTION
☑	Accept	application/json	
☑	Authorization	QWRtaW4=:TXJCdWdneTE5IQ==	
☑	Content-Type	application/x-www-form-urlencoded	

17. Headers – new user admin

G. Complete Body tab -> x-www-form-urlencoded – creation admin user parameters required (parameters presented on the screen below)



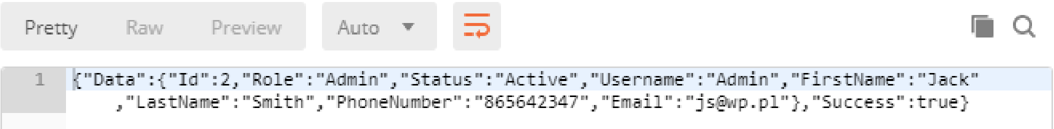
POST http://localhost:8080/user/admin

○ none ○ form-data ● x-www-form-urlencoded ○ raw ○ binary

	KEY	VALUE	DESCRIPTION
☑	first_name	Jack	
☑	username	Admin	
☑	last_name	Smith	
☑	email	js@wp.pl	
☑	phone_number	865642347	
☑	password	MrBuggy19!	

18. Body – new user admin

H. Click: Send
The answer request will be visible in response section



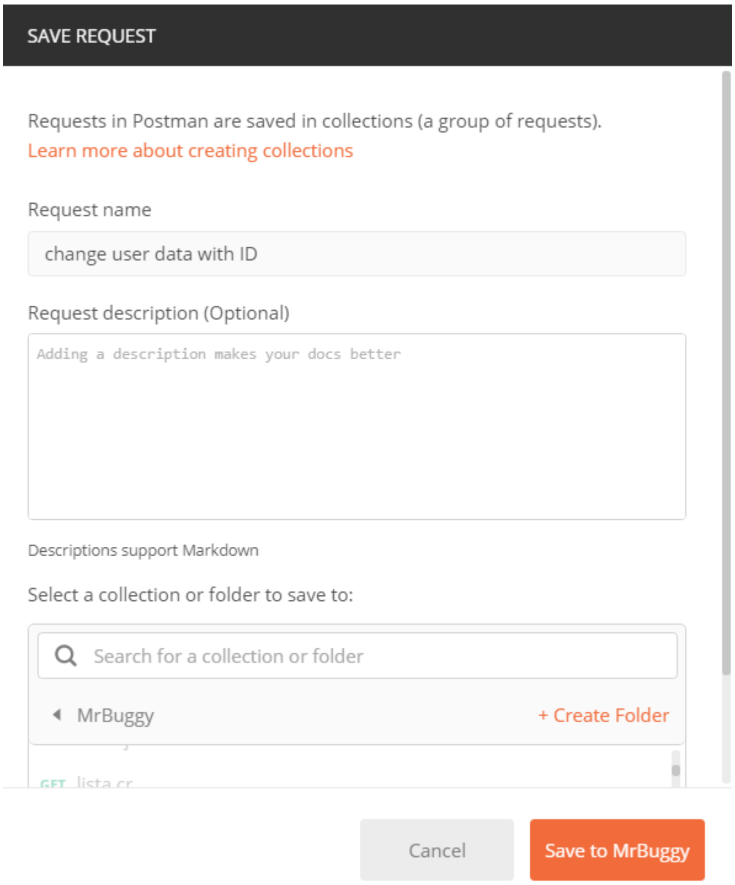
```
{"Data":{"Id":2,"Role":"Admin","Status":"Active","Username":"Admin","FirstName":"Jack"
    ,"LastName":"Smith","PhoneNumber":"865642347","Email":"js@wp.pl"},"Success":true}
```

19. Response section – new user admin

3. Request PUT / user/ID (change user data with ID)

A. Select: New -> Request

B. Fill field: 'Request name': 'change user data with ID' and select with 'Select a collection or folder to save to': 'MrBuggy'



SAVE REQUEST

Requests in Postman are saved in collections (a group of requests).
Learn more about creating collections

Request name: change user data with ID

Request description (Optional): Adding a description makes your docs better

Descriptions support Markdown

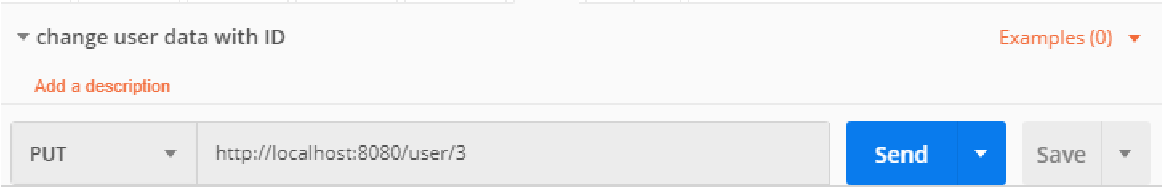
Select a collection or folder to save to:

20. Request – change user data with ID

C. Click: Save to MrBuggy

D. Select method: PUT

E. Complete address URL: http://localhost:8080/user/3



change user data with ID

Add a description

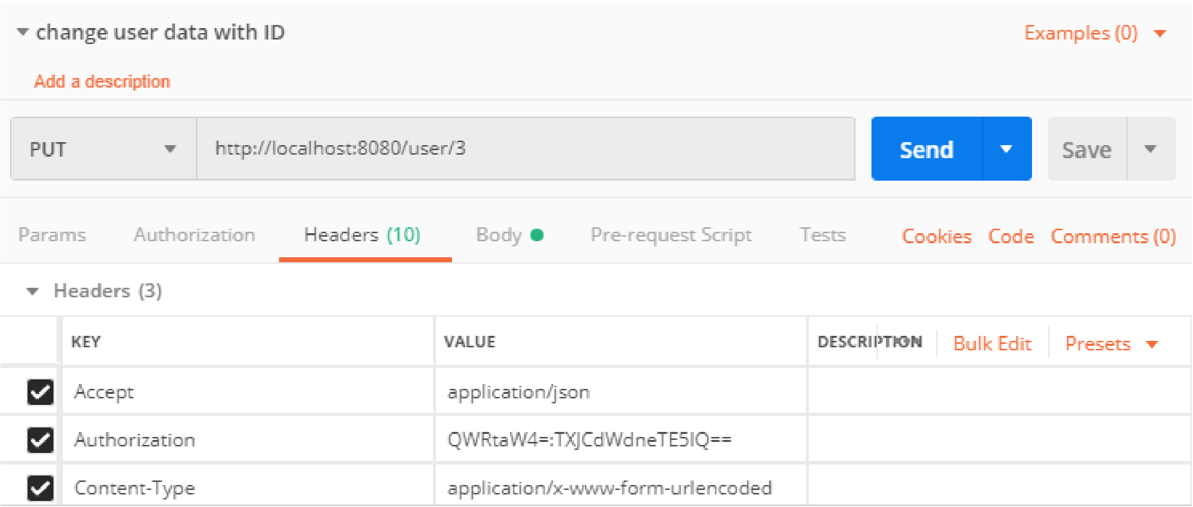
PUT http://localhost:8080/user/3

21. Address URL – change user data with ID

F. Complete Headers tab:

Authorization: username:password - 'QWRtaW4=:TXJCdWdneTE5IQ==' (username - Admin, password - MrBuggy19!)

Accept: 'application/json'
Content-Type: 'application/x-www-form-urlencoded'

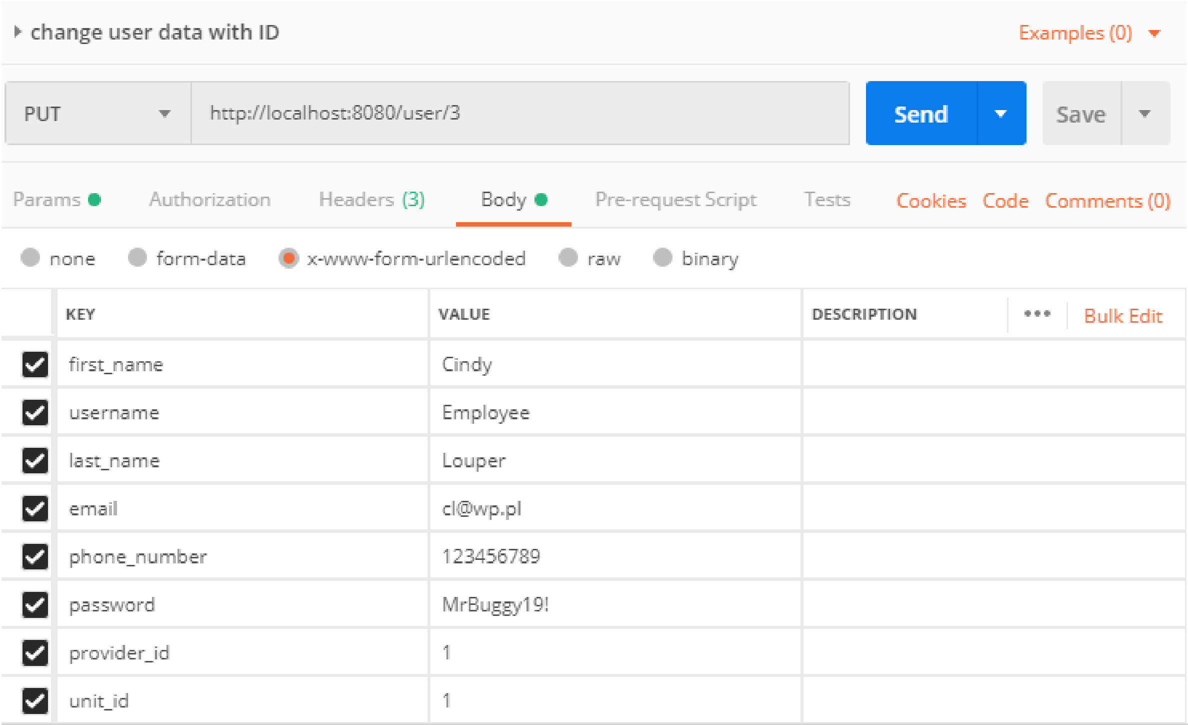


PUT http://localhost:8080/user/3

	KEY	VALUE	DESCRIPTION
☑	Accept	application/json	
☑	Authorization	QWRtaW4=:TXJCdWdneTE5IQ==	
☑	Content-Type	application/x-www-form-urlencoded	

22. Headers – change user data with ID

G. Complete Body tab -> x-www-form-urlencoded – change user data ID parameters required (parameters presented on the screen below)



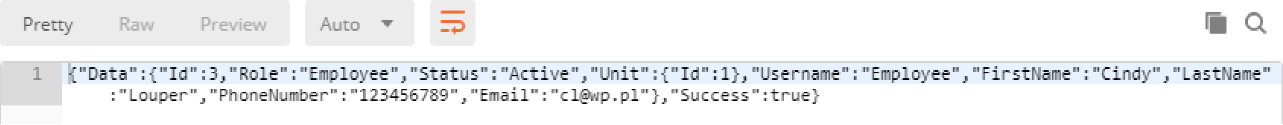
PUT http://localhost:8080/user/3

● x-www-form-urlencoded

	KEY	VALUE	DESCRIPTION
☑	first_name	Cindy	
☑	username	Employee	
☑	last_name	Louper	
☑	email	cl@wp.pl	
☑	phone_number	123456789	
☑	password	MrBuggy19!	
☑	provider_id	1	
☑	unit_id	1	

23. Body – change user data with ID

H. Click: Send
The answer request will be visible in response section



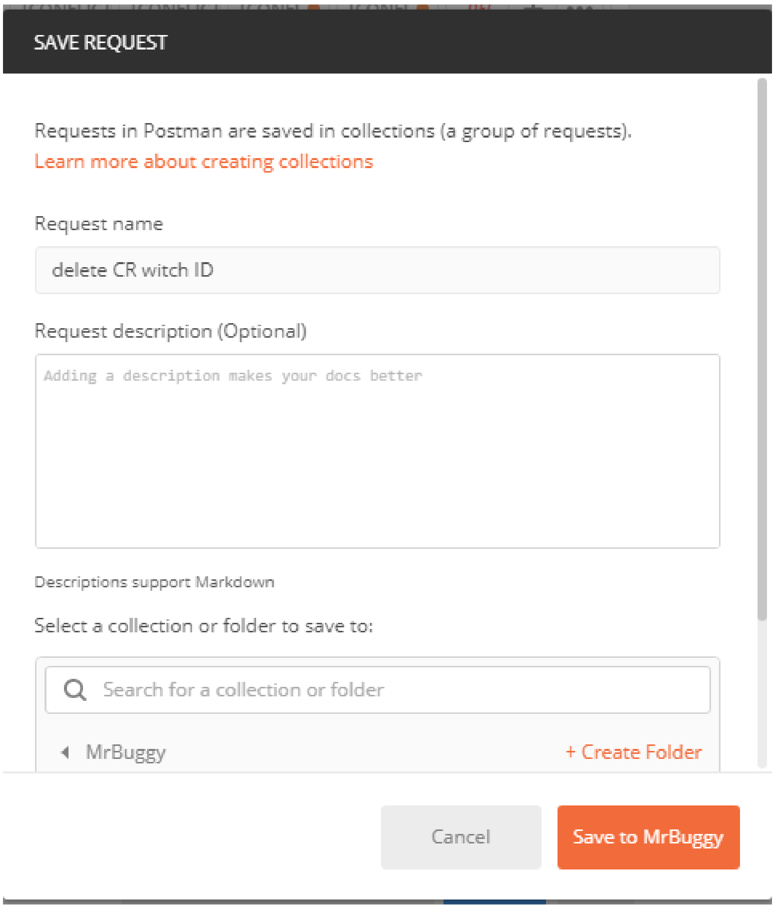
```
{"Data":{"Id":3,"Role":"Employee","Status":"Active","Unit":{"Id":1},"Username":"Employee","FirstName":"Cindy","LastName"
    :"Louper","PhoneNumber":"123456789","Email":"cl@wp.pl"},"Success":true}
```

24. Response section – change user data with ID

4. Request DELETE / cr/ID (delete CR with ID)

A. Select: New -> Request

B. Fill fields: 'Request name': 'delete CR witch ID' and select with 'Select a collection or folder to save to': 'MrBuggy'


SAVE REQUEST

Requests in Postman are saved in collections (a group of requests).
Learn more about creating collections

Request name

delete CR witch ID

Request description (Optional)

Adding a description makes your docs better

Descriptions support Markdown

Select a collection or folder to save to:

Search for a collection or folder

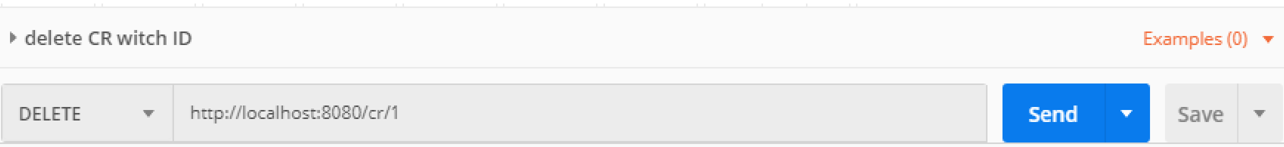
MrBuggy + Create Folder

25. Request – delete CR witch ID

C. Click: Save to MrBuggy

D. Select method: DELETE

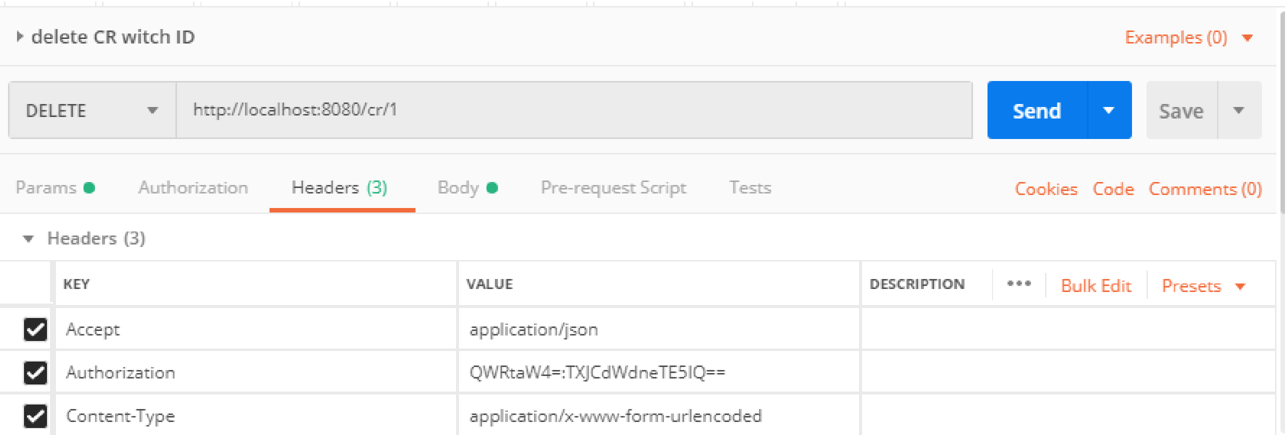
E. Complete address URL: http://localhost:8080/cr/1


delete CR witch ID

DELETE http://localhost:8080/cr/1

26. Adres URL – delete CR witch ID

F. Complete Headers tab:
Authorization: username:password - QWRtaW4=:QnVnZ3kh (username - Admin, password - MrBuggy19!)
Accept: 'application/json'
Content-Type: 'application/x-www-form-urlencoded'

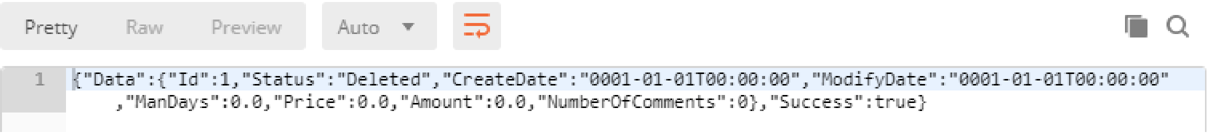


DELETE http://localhost:8080/cr/1

	KEY	VALUE	DESCRIPTION
☑	Accept	application/json	
☑	Authorization	QWRtaW4=:TXJCdWdneTE5IQ==	
☑	Content-Type	application/x-www-form-urlencoded	

27. Headers – delete CR witch ID

G. Complete Body tab -> x-www-form-urlencoded - delete CR with ID not required parameters
H. Click: Send
The answer request will be visible in response section



```
{"Data":{"Id":1,"Status":"Deleted","CreateDate":"0001-01-01T00:00:00","ModifyDate":"0001-01-01T00:00:00"
    ,"ManDays":0.0,"Price":0.0,"Amount":0.0,"NumberOfComments":0},"Success":true}
```

28. Response section – delete CR witch ID