Bidding API Documentation

INTRODUCTION

About the API

- This is an API solution that allows users to post bids regarding a service that they would like to acquire. Additionally the solution allow users to rate the service providers after the completion of the task and service providers can see a list of bids and submit proposal.

Development Tools & Languages

Backend

- PHP

Database

- MySQL

Security

- PHPIDS(PHP Intrusion detection system)

Request Tests

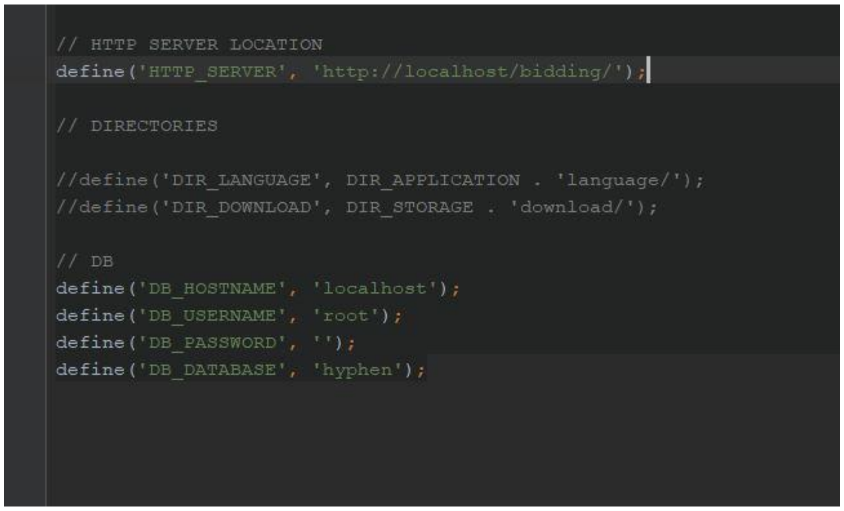
- Postman

Communication Platform

- JSON

Integration (Configuration)

- You should first copy the “bidding” directory that we provided to the place you want to deploy it.
- You should import tables that we provided, as sql file to your database (to try first then you should modify, as you needed.)
- To change the hostname, database name, username and password. You should go to Bidding/Configuration/config.php file.



```
// HTTP SERVER LOCATION
define('HTTP_SERVER', 'http://localhost/bidding/');

// DIRECTORIES

//define('DIR_LANGUAGE', DIR_APPLICATION . 'language/');
//define('DIR_DOWNLOAD', DIR_STORAGE . 'download/');

// DB
define('DB_HOSTNAME', 'localhost');
define('DB_USERNAME', 'root');
define('DB_PASSWORD', '');
define('DB_DATABASE', 'hyphen');
```

- For instance, you should change the “localhost” by your domain name. “root” to your user name. set if you have DB_PASSWORD. Change ‘hyphen’ to the database you named, but, if your database name is ‘hyphen’, then leave it.
- We provided postman’s requests collection as a json file. If you want to check/test the system you can import it to your postman and try the ready-made requests. (It’s just incase you wanted to do further tests)
- If you already have your own security measures and don’t want to be checked by the PHPIDS you can remove it by going to bidding/configuration/header.php file and remove other code in a file except that’s in else{…}.

ALL API SERVICES

- Registration
- Login
- Post Bid
- View Bid
- Edit Bid
- Get Bidding Information
- Approve Bid
- Delete Bid
- Bid
- List My Bids
- Edit Bidding Information
- Delete Bidding Information
- Rate User After Completion of Task

REGISTRATION

Description

- Allows users to register to the system.

Access URL

- /register/
- Example: https://example.com/bidding/register/

Required Fields

- Full name (required)
- E-mail (required)
- Password (required)
- Confirm password (required)

Verification (Server side)

- E-mail (unique, valid)
- Password (min length, valid format)

Database

user_table					
user_id	full_name	e_mail	registration_date	password	profile_pic_url

Request Example

- http://example.com/bidding/register?full_name=abc&email=abc@gmail.com&password=abc123456789&confirm_password=abc123456789

Response

Success

```
{
    "server_response": "registration_success"
}
```

Error

```
{
    "server_response": "registration_error",
    "error": "error_message"
}
```

LOGIN

Description

- Allows users to authenticate and login to the system.

Access URL

- /login/
- Example: https://example.com/login/

Required Fields

- E-mail (required)
- Password (required)

Verification (Basic verification)

- E-mail (unique, valid)
- Password (minimum length, valid)

Database

- Uses user_table

Request Example

- http://example.com/bidding/login.php?email=abc@gmail.com&password=abc12345678

API Response

Success

```
{
    "server_response": "login_success",
    "session_info": {
        "user_id": "18",
        "user_email": "ZW1lYnVAZ21haWwuY29t",
        "user_password": "1a4eeb020e2de6288f66d31c9e56eeea"
    }
}
```

Error

```
{
    "server_response": "login_error",
    "error": "error_message"}
```

POST BID

Description

- Allows users to post bids regarding a service that they would like to acquire.

Access URL

- /post_bid/
- Example: https://example.com/bidding/post_bid/

Pre Requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Required Fields

- Bid Title (required)
- Bid Description (required)
- Bid Picture URL (optional)
- Price (required)
- Expiration Date (optional)

Validation (Server Side)

- Validate whether or not the required fields are empty.

Auto Taken Fields (to be used by API developers, and guide API integrators)

- user_id (foreign key from users_table)
- status = 0
- posted_time = current_timestamp
- rating = null

Database

bids_table				
bid_id	user_id	bid_title	bid_description	bid_pic_url
price	posted_time	expiry_date	status	rating

Request Example

http://example.com/bidding/post_bid.php?bid_title=Buy My Car&bid_description=I want to sell my car you can engage by biding&price=200000&bid_picture_url=https://image-cdn.beforward.jp/large/201911/1550502/BG599364_5b2f7b.JPG &expiration_date_time=2019-11-01 21:39:19

Response

Success

```
{
    "server_response": "bid_post_success"
}
```

Error

```
{
    "server_response": "post_bid_error",
    "error": "error_message"
}
```

VIEW BID

View (List) All Bids

Description

- Allows users to see list of bids.

Access URL

- /view_bids/
- Example: https://example.com/bidding/view_bids/

Pre Requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Database

- Uses bids_table

Rule

- 10 results per each request will be returned. (if limit and offset is not provided)

URL queries (optionals)

- limit = Max result per page
- offset = value which determines the starting point in which database fetching starts.
- bid_status = 1 (to show only open bids)
- bid_status = 2 (to show only closed bids)
- bid_status = 3 (to show only completed bids)
- if bid_status is not stated it show all bids

Request Example

- http://example.com/bidding/view_bids?limit=14&offset=0&bid_status=1

Response

Success

```
{
    "server_response": "view_bid_success",
    "bids": [
        {
            "bid_id": "1",
            "user_id": "10",
            "bid_title": "Software Design",
            "bid_description": "I want programmer",
            "bid_pic_url": "img_url",
            "price": "56",
            "posted_time": "32543654665",
            "expiry_date": "5463547657",
            "status": "1",
            "rating": "5"
        },
        {
            "bid_id": "2",
            "use_id": "10",
            "bid_title": "Back-end development",
            "bid_description": "I want programmer",
            "bid_pic_url": "img_url",
            "price": "56",
            "posted_time": "32543654665",
            "expiry_date": "5463547657",
            "status": "1",
            "rating": "5"
        }
    ]
}
```

Error

```
{
    "server_response": "view_bid_error",
    "error": "error_message"
}
```

View Single Bid

Description

- Allows users to see a specific bid details and list of bidders that submitted proposals to that specific bid.

Access URL

- /view_bids
- Example: https://example.com/view_bids

Pre Requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Database

- Uses bids_table

URL Queries

- bid_id (describes the id of the bid that is going to be fetched) (required)
- limit = maximum proposed bidding results for that specific bidding per page. (optional)
- offset = Start from specific bid (optional)

Process

- Needs to fetch information about the bid.
- Needs to fetch bidders that participated in this specific bid.

Rules

- 10 results per each request will be returned.

Request Example

- `http://example.com/bidding/view_bids?bid_id=5&limit=2&offset=0`

Response

Success

```
{
    "server_response": "view_single_bid_success",
    "bid": {
        "bid_id": "5",
        "user_id": "9",
        "bid_title": "asdfg",
        "bid_description": "Iwant to sell my car you can engage by bidding",
        "bid_pic_url": "qwertyu",
        "price": "200000",
        "posted_time": "2019-11-0120:52:06",
        "expiry_date": "2019-11-0121:39:19",
        "status": "1",
        "rating": ""
    },
    "bidders": [
        {
            "bidding_id": "3",
            "bid_id": "5",
            "user_id": "11",
            "proposed_price": "180000",
            "description": "Heylets agree with this goodprice!"
        },
        {
            "bidding_id": "11",
            "bid_id": "5",
            "user_id": "13",
            "proposed_price": "900",
            "description": "qqqqqqqqqqqqqqqq"
        }
    ]
}
```

Error

```
{
    "server_response": "view_single_bid_error",
    "error": "error_message"
}
```

Edit Bid

Description

- Allows users to edit bids that they posted.

Access URL

- /edit_bid?bid_id=34535
- Example: https://example.com/view_bid?bid_id=34535

Pre-requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Database

- Uses bids_table

Required Fields

- Bid title (required)
- Bid description (required)
- Bid picture URL (required)
- Price (required)
- Expiry date (optional)

Validation (Server side)

- Validate whether required fields are empty or not.

Process

- Check weather or not you are the creator of the bid
- Check bid exists or not and the status is not closed

URL Query

- bid_id (describes the id of the bid that is going to be fetched) (required)

Request Example

- `http://example.com/bidding/edit_bid?bid_id=9&bid_title=My fancy car For sale &bid_description=I want to sell my car you can engage by biding&price=200000&bid_picture_url=qwertyu&expiration_date_time=2019-11-01 21:39:19`

Response

Success

```
{
    "server_response": "edit_bid_success"
}
```

Error

```
{
    "server_response": "edit_bid_error",
    "error": "error_message"
}
```

```
{
   “server_
   “error”
}
```

Delete Bid

Description

- Allows users to delete bids that they posted.

Access URL

- /delete_bid/
- Example: https://example.com/bidding/delete_bid

Pre-requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Database

- Uses bids_table

Process

- Check whether or not you are the creator of the bid.
- Check bid exists or not and the status is not closed.

URL Query

- bid_id (describes the id of the bid that is going to be fetched) (required)

Request Example

- http://example.com/bidding/delete_bid?bid_id=9

Response

Success

```
{
    "server_response": "delete_bid_success"
}
```

Error

```
{
    "server_response": "delete_bid_error",
    "error": "error_message"
}
```

Bid

Description

- Allows users to submit proposals to a specific bid.

Access URL

- /bid/
- Example: https://example.com/bidding/bid/

Pre-requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Database

bidding_table				
bidding_id	bid_id	user_id	proposed_price	description
time	status			

Required Fields

- Amount (required)
- Description (required)

Auto Taken Fields

- bid_id (foreign key from bids_table) (required)
- user_id (foreign key from users_table) (required)

Validation (Server side)

- Validate whether required fields are empty or not.

Process

- Bidder can't bid twice for a single bid.
- Bidder can only bid to open and unexpired bids.

Request Example

- http://example.com/bidding/bid?bid_id=10&proposed_price=8000&bidding_description=I offered this price but I can do that with unbelievable quality!

Response

Success

```
{
    "server_response": "bid_success"
}
```

Error

```
{
    "server_response": "bid_error",
    "error": "error_message"
}
```

List My Bids

Description

- Allows users to see bids that they posted and bids that they participated (submitted proposal to).

Access URL

- /list_my_bids/
- Example: https://example.com/bidding/list_my_bids/

Pre-requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Database

- Uses bids_table, bidding_table

Required Fields

- Amount (required)
- Description (required)

Process

- In order to list the bids the user has to be the creator of the bid or has participated in the bid.

Request Example

- http://example.com/bidding/list_my_bids

Response

Success

```
{
    "server_response": "list_my_bids_success",
    "posted": [
        {
            "bid_id": "9",
            "user_id": "18",
            "bid_title": "ha ha ah a",
            "bid_description": "I want to sell my car you can
engage by biding",
            "bid_pic_url": "qwertyu",
            "price": "200000",
            "posted_time": "2019-11-01 22:06:14",
            "expiry_date": "2019-11-01 21:39:19",
            "status": "1",
            "rating": "5"
        },
        {
            "bid_id": "13",
            "user_id": "18",
            "bid_title": "Buy My Car",
            "bid_description": "I want to sell my car you can
engage by biding",
            "bid_pic_url": "https://image-
cdn.beforward.jp/large/201911/1550502/BG599364_5b2f7b.JPG",
            "price": "200000",
            "posted_time": "2019-11-25 16:20:03",
            "expiry_date": "2019-11-01 21:39:19",
            "status": "1",
            "rating": null
        }
    ],
    "participated": [
        {
            "bidding_id": "12",
            "bid_id": "7",
            "user_id": "18",
            "proposed_price": "900",
            "description": "qqqqqqqqqqqqqqqq"
        },
        {
            "bidding_id": "13",
            "bid_id": "8",
            "user_id": "18",
            "proposed_price": "800",
            "description": "qqqqqqqqqqqqqqqq"
        },
        {
            "bidding_id": "14",
            "bid_id": "9",
            "user_id": "18",
            "proposed_price": "800",
            "description": "qqqqqqqqqqqqqqqq"
        },
        {
            "bidding_id": "15",
            "bid_id": "10",
            "user_id": "18",
            "proposed_price": "800",
            "description": "qqqqqqqqqqqqqqqq"
        },
        {
            "bidding_id": "16",
            "bid_id": "11",
            "user_id": "18",
            "proposed_price": "800",
            "description": "qqqqqqqqqqqqqqqq"
        }
    ]
}
```

Error

```
{
    "server_response": "list_my_bids_error",
    "error": "error_message"
}
```

Edit Bidding Info

Description

- Allows users to edit their proposal that they submitted for a bid.

Access URL

- /edit_bidding/
- Example: https://example.com/bidding/edit_bidding/

Pre-requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Database

- Uses bidding_table

Required Fields

- Amount (optional)
- Description (optional)

Validation (Server side)

- Validate whether required fields are empty or not.

Process

- Check if you have participated (proposed price) in that specific bid.

Request Example

- http://example.com/bidding/edit_bidding?bidding_id=7&proposed_price=899&bidding_description=abuguyuuiiiiida

API Response

Success

```
{
    "server_response": "edit_bidding_success"
}
```

Error

```
{
    "server_response": "edit_bidding_error",
    "error": "error_message"
}
```

Delete Bidding Info

Description

- Allows users to delete proposal/s that they submitted for a bid.

Access URL

- /delete_bidding_info/
- Example: https://example.com/bidding/delete_bidding_info/

Pre-requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Database

- Uses bidding_table

Auto Taken Fields

- bidding_id (foreign key from bidding_table) (required)

Process

- Check if the specified bidding is yours.
- Approved bids can’t be deleted.

Request Example

- http://example.com/bidding/delete_bidding_info?bidding_id=11

API Response

Success

```
{
    "server_response": "delete_bidding_info_success"
}
```

Error

```
{
    "server_response": "delete_bidding_info_error",
    "error": "error_message"
}
```

Get Bidding Info

Description

- Allows users to get the details of a proposal.

Access URL

- /get_bidding_info/
- Example: https://example.com/bidding/get_bidding_info

Pre-requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Database

- Uses bidding_table

Auto Taken Fields

- bidding_id (foreign key from bidding_table) (required)

Process

- Check if bidding item exists with that specific bidding id.
- Get rating of the user from bids-table where the user win the bid and the user completed the task required by the bid poster (doesn't include unrated bids)

Request Example

- http://example.com/bidding/get_bidding_info?bidding_id=8

API Response

Success

```
{
    "server_response": "get_bidding_info_success",
    "bidding_info": {
        "bidding_id": "8",
        "bid_id": "7",
        "user_id": "9",
        "proposed_price": "809",
        "description": "Lets agree with this price",
        "time": "2019-11-03 20:11:37",
        "status": "1"
    }
}
```

Error

```
{
    "server_response": "get_bidding_info_error",
    "error": "error_message"
}
```

Approve Bid

Description

- Allows users to approve proposal for bids that they posted.

Access URL

- /approve_bid/
- Example: https://example.com/bidding/approve_bid/

Pre-requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Database

- Uses bidding_table

Auto Taken Fields

- bidding_id (foreign key from bidding_table) (required)

Process

- Check if the current user is the creator of the bid.
- Check if the bid is not closed and expired.
- Check if the bid is not approved.

Request Example

- http://example.com/bidding/approve_bid?bidding_id=11

API Response

Success

```
{
    "server_response": "approve_bidding_success"
}
```

Error

```
{
    "server_response": "approve_bidding_error",
    "error": "error_message"
}
```

Rate User After Completion of Task (Bid)

Description

- Allows users to rate users (service providers) after the completion of a specific task.

Access URL

- /rate_bid/
- Example: https://example.com/bidding/rate_bid/

Pre-requirements

- Authentication
- user_exists (variable that determines the existence of the user)

Database

- Uses bidding_table

Auto Taken Fields

- bidding_id (foreign key from bidding_table) (required)

Process

- Task that need to be done should have to be completed (bid have to be completed).

Request Example

- http://example.com/bidding/rate_bid?bidding_id=11

API Response

Success

```
{
    "server_response": "rate_success"
}
```

Error

```
{
    "server_response": "rate_error",
    "error": "error_message"
}
```