YaaS interface specification

This file describes the public interface of the YaaS application. The automated tests (testsTDD.py) provided for checking your projects are relying on it.

Revision

Version	**Date**	**Description**
1.0	16.9.2019	First version.
1.1	30.9.2019	● Updated UC1: signup status code from 400 to 200, ● Removed signin specs.

The URLs described below are the same as the ones provided in the urls.py in the skeleton project

UC1 Create User Account

URL: /signup

- GET - get a signup form
  - Return status code 200
- POST - create a user with username and password
  - Sign up with password or email missing return status code 200.
  - Sign up with already taken username, return code 200 and an error message "This username has been taken" is present in response's HTML.
  - Sign up with already taken email, return code 200 and an error message "This email has been taken" is present in response's HTML.
  - Sign up with valid data, return code 302 (redirect).

Example request: students must follow strictly the name fields in their projects

POST /signup

```
{
  "username": "user1",
  "password": "password1",
  "email": "user1@mail.com"
}
```

UC2 Edit account information

URL: /user/ or /user/profile

- GET - get profile of the current user
  - Get profile of unauthenticated user, return code 302 (redirect to signin page).
  - Get profile of a logged in user, return code 200
- POST - edit email or password of the current user
  - Edit email and password of a logged in user, the user should have new email and password. return code 302 (redirect)
  - Edit email to the one that already been taken, email stays the same. return code 200 Not using a form so return code is 302. If extra time left over implement a django form

Example request:

POST /user/profile

```
{
  "email": "newemail@yaas.com",
  "password": "newpassword1"
}
```

UC3 Create a new auction

URL: auction/create

- GET - get returns auction form, return code 200
- POST - create a new auction item
    - Create an auction with unauthenticated user, return code 302.
    - Create an auction with invalid deadline date, an error message is in HTML response "The deadline date should be at least 72 hours from now". Status code 200.
    - Create an auction with invalid deadline date format, an error message is in HTML response "Enter a valid date/time". Status code 200.
    - Create an auction with minimum price lower than 0.01, return an error message "Ensure this value is greater than or equal to 0.01". Status code 200.
    - Create an auction with valid data, return code 200 (after 302) and success message "Auction has been created successfully".
    - When an auction is created successfully, return code 200 (after 302), an email is sent to the user creating it (i.e., an email item is found in the virtual outbox with a message message "Auction has been created successfully".

Example request: students must follow strictly the name fields in their projects

POST /auction/create

```
{
  "title": "item1",
  "description": "something",
  "minimum_price": 10,
  "deadline_date": "30/12/2019 20:00:00"
}
```

UC4 Edit the description of an auction

URL: auction/edit/{id}

- GET - get edit auction form

    - Get to edit an auction of another user, show error message in HTML response “That is not your auction to edit”, status code 200
    - Get to edit an auction, return code 200.
- POST - Edit an auction
    - Edit an auction of another user, show error message in HTML response “That is not your auction to edit”. Status code 200
    - Edit description of an auction, show success message in HTML “Auction has been updated successfully”. Status code 200 (after 302)

Example request: students must follow strictly the name fields in their projects

POST /auction/edit/1

```
{
  “title”: “item1”,
  “description”: “new content”
}
```

UC5 Browse and Search auctions

Browse URL: / or /auction

- GET - get a list of active auctions
    - Return a list of active auctions, status code 200

Search URL: /search/?term={title}

- GET - search auctions by title,
    - Return a list of auctions that match, status code 200

Example request:

GET /search/?term=car

UC6 Bid

URL: /auction/bid/{id}

- POST - bid on an auction
    - Unauthenticated user bids on an auction, return status code 302.
    - Sellers bid on own auctions, show error message in HTML “You cannot bid on your own auctions”. status code 200
    - Bid on inactive auction, show error message in HTML “You can only bid on active auctions”. status code 200

- Bid on auction that has outdated deadline date, show error message in HTML “You can only bid on active auctions”. status code 200
- Bid on auction with invalid bid amount, show error message in HTML “New bid must be greater than the current bid for at least 0.01”. status code 200
- Bid on auction with valid data, return code 200 (after 302) and show success message in HTML “You has bid successfully”.

Example request: students must follow strictly the name fields in their projects

POST /auction/bid/2

```
{
  “new_price”: 15
}
```

UC7 Ban an auction

URL: /auction/ban/{id}

- POST - ban an auction
  - Normal user bans an auction, return code 302 and auction remains unbanned
  - Admin user bans an auction, return code 200 (after 302), auction is banned, and show success message “Ban successfully”.

Example request:

POST /auction/ban/3

UC8 Resolve auction

URL: /auction/resolve/

- GET - resolve auctions
  - Resolve auctions, all the auctions that have a deadline date before the moment should be resolved. Response (**in JSON format**):

HTTP 200 OK

```
{
    “resolved_auctions”: [“auction1”, “auction2”]
}
```

Example request:

GET /auction/resolve

UC9 Support for multiple Languages

URL: /changeLanguage/{lang_code}

- GET - change the language of the system, should only support English and Swedish
    - Change language to Swedish, show success message “Language has been changed to Swedish”. Status code 200
    - Change language to English, show success message “Language has been changed to English”. Status code 200

Example requests:

GET /changeLanguage/en

GET /changeLanguage/sv

UC11 Support for multiple currency exchange

URL: /changeCurrency/{currency_code}

The tests currently supports Optimistic approach.

- GET - change the currency, should only support EUR and USD
    - Change currency to EUR, show success message “Currency has been changed to EUR”. Status code 200
    - Change currency to USD, show success message “Currency has been changed to USD”. Status code 200

Example request:

GET /changeCurrency/usd

WS1 Browse and Search API

/api/v1/browse

- GET: browse for active auctions, return code 200 and a list of active auctions.

/api/v1/search/{title}

- GET: search auctions by title without ‘?term=’, return code 200 and a list of active auctions that match.

/api/v1/search/?term={title}

- GET: search auctions by title with ‘?term=’, return code 200 and a list of active auctions that match the term.

/api/v1/searchid/{id}

- GET: search auction by id, return an auction of that id. Return code 200

Example requests:

/api/v1/browse

/api/v1/search/car

/api/v1/search/?term=bike

Example response:

HTTP 200 OK

```
[
    {
        "title": "old car",
        "description": "very old",
        "minimum_price": 500,
        "deadline_date": "31.12.2019 20:00:00"
    },
    {
        "title": "new car",
        "description": "almost new",
        "minimum_price": 1000,
        "deadline_date": "31.11.2019 20:00:00"
    }
]
```

WS2 Bid API

URL: /api/v1/bid/{id}

- POST - bid auction
    - Unauthenticated user accessing the API, return status code 401 and message in "detail": "Authentication credentials were not provided".
    - Bid on own auction, JSON response has status code 400 and message "Cannot bid on own auction".
    - Bid on banned auction, JSON response has status code 400 and message "Can only bid on active auction".
    - New bid is the same as the current one, JSON response has status code 400 and message "New bid must be greater than the current bid at least 0.01".
    - Bid with invalid type of bid, JSON response has status code 400 and message "Bid must be a number".
    - Bid with valid data successfully, JSON response has status code 200, emails are in outbox and message "Bid successfully".

Example request for user "user1" and password "123"

POST /api/v1/bid/5/

Authorization: Basic dXNlcjE6MTIz

```
{
  “new_price”: 12
}
```

Example responses (in JSON format):

HTTP 200 OK

```
{
  “message”: “Bid successfully”,
  “title”: “pencil”,
  “description”: “a new pencil”,
  “current_price”: 12,
  “deadline_date”: “31.12.2019 20:00:00”
}
```