YAAS – Yet Another Auction Site

Course project in Development of Web Applications and Services 2013

Viktor Kåll (34256) – 4.11.2013

1. Implemented requirements

Grade 1

- ✓ UC1 create user account
- ✓ UC3 create new auction
- ✓ TR2.1 automated test for UC3
- ✓ TR1 DB fixture and data generation program

Grade 2

- ✓ UC5 browse and search auctions
- ✓ WS1 browse and search API for web service
- ✓ UC7 ban auction
- ✓ UC4 edit auction

Grade 3

- ✓ UC2 edit user account info
- ✓ UC8 resolve auction
- ✓ UC9 support for multiple languages

Grade 4

- ✓ UC6 bid
- ✓ WS2 bidding API for web service
- ✓ TR2.2 automated test for UC6 bid

Grade 5

- ○ Not implemented

Optional features

- ✓ Soft deadlines
- ✓ Store language preference permanently for registered users

2. Versions used

- Python 2.7.5
- Django 1.5.4
    - Django rest framework 2.3
    - Django cron 0.3
- SQLite3

3. Admin login

The data generation program (TR1) also creates an admin account. The credentials are as follows:

Admin username: admin
Admin password: admin

The generation program also generates 50 normal users. These all have 'userN' as username and password where N is a number from 1-50.

The data generation program can be run by visiting this url:
`^YAAS/populate_database/$`

4. Session management

The main thing that I am using sessions for is the language switching. This is described more in detail in section 10 of this document.

5. Confirmation form

When an auction is created I first check that the auction form is valid. Then the auction form, along with a confirmation form (containing a choice field with "yes" and "no") is sent to a confirmation.html template. Here the auction form is saved in hidden input fields and the user has to answer the confirmation form (yes or no).

If the user selects yes the auction is created and saved in the database. If the user selects no he is then redirected to a message telling him that the auction was not created.

6. Resolving bids

Bids are resolved by.using a Django cron job. The cron job can be run run using the command “python manage.py runcrons” and should be initiated automatically on the server using cron (or scheduled tasks on Windows).

7. Concurrency

Concurrency issues in UC6 (bidding) are handled by comparing the form data with database data. If the “updated” date in the hidden form field is older than the auctions “updated” field in the database this means that the auction has changed since the page was loaded. If this happens the bid is not accepted and the user is redirected to a page with a message informing him what has happened.

8. REST API

- List all active auctions that are not banned:

`^YAAS/api/auctions/$`


```
Viktor-macbook:~ vkall$ curl 127.0.0.1:8000/YAAS/api/auctions/
[{
      "id": 2,
      "title": "Title2",
      "description": "This is a description of item number 2",
      "seller": 47,
      "start_date": "2013-11-04T14:15:31.255Z",
      "updated_date": "2013-11-04T14:15:31.255Z",
      "end_date": "2013-11-07T14:15:31.254Z",
      "minimum_price": "26.5",
      "active": true,
      "banned": false
},
{
      "id": 3,
      "title": "Title3",
      "description": "This is a description of item number 3",
      "seller": 3,
      "start_date": "2013-11-04T14:15:31.290Z",
      "updated_date": "2013-11-04T14:15:31.290Z",
      "end_date": "2013-11-10T14:15:31.290Z",
      "minimum_price": "27.5",
      "active": true,
      "banned": false
}, /*snip* All active auctions are listed here
```

- View auction by id:

`^YAAS/api/auctions/(?P<id>\d+)/$`

```
Viktor-macbook:~ vkall$ curl 127.0.0.1:8000/YAAS/api/auctions/33/
[{
      "id": 33,
      "title": "Title33",
      "description": "This is a description of item number 33",
      "seller": 5, "start_date": "2013-11-04T14:15:32.240Z",
      "updated_date": "2013-11-04T14:15:32.240Z",
      "end_date": "2013-11-08T14:15:32.240Z",
      "minimum_price": "57.5",
      "active": true,
      "banned": false
}]
```

- Bid on auction:

`^YAAS/api/auctions/(?P<id>\d+)/bid/$`

Accepts a bid (2 decimal float) and tries to make a bid. Requires a user to be logged in, auction to be active, bid to be high enough and does not accept bids from the seller or if the user is already winning the auction.

Without login:

```
Viktor-macbook:~ vkall$ curl 127.0.0.1:8000/YAAS/api/auctions/33/bid/
--data "bid=100.50"
{"detail": "Authentication credentials were not provided."}
```

With login:

```
Viktor-macbook:~ vkall$ curl 127.0.0.1:8000/YAAS/api/auctions/33/bid/
--data "bid=100.50" --user user1:user1
{
   "id": 215,
   "auction": 33,
   "bid": "100.50",
   "bidder": 2,
   "timestamp": "2013-11-04T18:17:35.150Z"
}
```

- Search auctions by title and description:

`^YAAS/api/auctions/search/(?P<criteria>(\w\s*)+)/$`

```
Viktor-macbook:~ vkall$ curl
127.0.0.1:8000/YAAS/api/auctions/search/Title33/
[{
      "id": 33,
      "title": "Title33",
      "description": "This is a description of item number 33",
      "seller": 5, "start_date": "2013-11-04T14:15:32.240Z",
      "updated_date": "2013-11-04T14:15:32.240Z",
      "end_date": "2013-11-08T14:15:32.240Z",
      "minimum_price": "57.5",
      "active": true,
      "banned": false
}]
```

9. Testing

For TR2.1 I am testing that a user has to be logged in to be able to create an auction. I am also testing that when an auction is created it is actually saved in the database.

For TR2.2 I am testing that a user has to be authorized in order to make a bid. I am also testing that the seller can't bid on his own auction. Lastly I test that when a bid is made it is connected to the right auction.

10. Language switching

The language switching is implemented with a dropdown menu that has a form and radio buttons for the languages (English, Swedish and Finnish). When the form is posted the choice is saved in the session.

```
request.session['django_language']
```

This activates the language switching in django. If a user is logged in the selected language is saved in a `UserLanguage` model.

If the `change_language()` view is requested with GET instead of POST the users language preference is loaded into the django_language session. If the user isn't logged in the language is set to the default, which is English.