**SoftUni Python WEB Frameworks Django Module**

**Final Project “AUTO CRY”**

**(extended project information)**

**Basic functionality:**

1. Landing page – some CSS animation applied:

* header’s elements – depend on logged user availability
* footer – static text and “contact us”, shortcuts to e-mail form, SoftUni site and sound control button
* main content - depends on logged user availability (“sign in” and “sign up” buttons – unsigned only)

1. Register page – all visitors can register after providing requested information.
2. Login page – available to registered users.
3. Logout page – available to authenticated users.
4. Profile page – available to all authenticated users for viewing, owners can change their profile photo by clicking additional checkbox. All owned items are listed with corresponding rights buttons.
5. Items list page – all items listed with different buttons according to their ownership. Owners and super users have CRUD access. All other users can only see details.
6. Creating a new item - available to authenticated users.
7. Editing existing item – available to owners and superusers (owner is not changed).
8. Deleting existing item – available to owners and superusers.
9. Detailed item view available to authenticated users, who can:

* comment ( if previous comment is referred a “@user\_name” text is placed in the field )
* like ( allowed only once to non-owners, after that disliking is only possible )

1. Project available at <https://github.com/garabedian/AutoCry_Python_WEB_Final_Project>

**Extended features:**

1. Preview of images for items and profiles – achieved through DOM manipulations with additional JS script.
2. Choice offered for image type for an item (WEB or Local).
3. Custom validator for year field of item and profile.
4. Reusable custom .CSS and Django-bootstrap4 used, along with individual styling of elements. Locally downloaded bootstrap files could be used if needed later.
5. Authorization through “groups” of users.

* Common “users” group is assigned upon registration
* “Super user” group can be added through Django admin panel

1. CSS animation and clickable avatar of current user in header.
2. Filter functionality included in “All items” page. Filter seeks in two string fields (“make” and “model” simultaneously).
3. “Password reveal” functionality through DOM manipulations with additional JS script.
4. Custom template tag created for getting dictionary values directly in templates (item\_detail.html).
5. Project deployed at <https://autocry.herokuapp.com/>.
6. Added and tested e-mail functionality, using “sendgrid.com” service. Project reverted to console output of e-mails for development purposes.
7. Added tweaks to user administration (auth\_app/admin.py):

* extended user profile table shows objects as “names” (UserProfileAdmin(admin.ModelAdmin))
* extended user profile object attached to its owner’s edit page (class ProfileInline(admin.StackedInline))
* URL changed to “ …/nimda/ “

1. Experimentally added “background music” with control button in footer. Works perfectly in Firefox incl. stop/resume across all pages, but is buggy in Chrome. Many “www” reports are found about this problem.
2. Two of the FBV (delete\_item, list\_items) refactored to CBV (DeleteItemView, ListItemsView) with the adequate check for permission (decorator and test\_func) and modifying context to include the forms.
3. Pagination which comes from CBV ListItemsView.
4. RESTFul APIs for all of items ( ./item/api/ ) and details ( ./item/api/<item№>/ ).