
Eggplant Documentation

Release 0.1

The Eggplant Open Source Community & Hacklab Team

February 03, 2016

1	Installation	3
2	Contributing	5
2.1	Types of Contributions	5
2.2	Get Started!	6
2.3	Pull Request Guidelines	7
2.4	Tips	7
3	Authors	9
4	History	11
4.1	0.1 (2015-08-15)	11
5	Developer documentation	13
5.1	Component overview	13
5.2	Dependency overview	14
6	Python Reference	15
6.1	eggplant package	15
7	Eggplant	35
8	How to contribute	37
9	Get in touch	39
10	Indices and tables	41
	Python Module Index	43

Contents:

Installation

TODO: Move from README.rst

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

2.1 Types of Contributions

2.1.1 Report Bugs

Report bugs at <https://github.com/kbhff/eggplant/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

2.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” is open to whoever wants to implement it.

2.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

2.1.4 Write Documentation

eggplant could always use more documentation, whether as part of the official eggplant docs, in docstrings, or even on the web in blog posts, articles, and such.

2.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/kbhff/eggplant/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

2.2 Get Started!

Ready to contribute? Here's how to set up *eggplant* for local development.

If you are completely new to source code versioning using git, please search for a video explaining it. And ask for help getting a git tool set up on your machine.

Note: We have decided to use the very conventional [a simple git branching model](#). Read the guide to get a good introduction to Git workflows.

```
$ git clone git@github.com:kbhff/eggplant.git
```

2.2.1 Virtualenv

The project is pretty basic, these are classical just steps. Just make note that it's a Python 3.4 only project. Enter the git project folder.

```
$ pip install virtualenvwrapper
```

To get the mkvirtualenv command you need to:

```
source /usr/local/bin/virtualenvwrapper.sh
```

On debian this file in:

```
/etc/bash_completion.d/virtualenvwrapper
```

start a new bash session to source it.

```
$ mkvirtualenv eggplantenv -p python3.4
$ workon eggplantenv
$ pip install -r requirements/development.txt
$ python manage.py syncdb
$ python manage.py runserver
```

Use “workon eggplantenv” to activate the eggplan virtual environment, and “deactivate” to exit.

This will deploy a local SQLite database and run a local webserver. If you are completely new to Django and Python, notice that you need [pip](<https://pip.pypa.io/en/stable/installing.html>), too.

2.2.2 Vagrant

(Optional) This takes some time (downloading) and some haddisk capacity.

Another way to get started contributing to this project is to download and install git and [Vagrant](<http://vagrantup.com/>), Clone the project (as mentioned above), change directory into the eggplant folder and then run the following commands:

```
$ vagrant up
$ vagrant ssh
$ python manage.py migrate
$ python manage.py test
$ python manage.py createsuperuser
$ python manage.py runserver
```

This will download and bootstrap an ubuntu 14.04 vagrant box, connect to it, start the django development server. The project should now be available at <http://192.168.33.28:8000/>.

2.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.4. https://travis-ci.org/kbhff/eggplant/pull_requests and make sure that the tests pass for all supported Python versions.

2.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_eggplant
```

Authors

Lots of people contributed to this group through the Eggplant Meetup group in Copenhagen.

<http://www.meetup.com/Eggplant/>

History

4.1 0.1 (2015-08-15)

- Adding a history file for the Eggplant application

Developer documentation

5.1 Component overview

The Eggplant application contains the following sub-applications (python packages nested in `eggplant/` containing models and migrations).

Applications inside `eggplant/` are interdependent, except `core` which should not depend on anything else and `permissions` which all other applications are expected to use.

- `accounts` - Administration and user pages related to accounts, being everything that has to do with user's transactions inside the market.
- `core` - an application with models that all other applications may depend on, **dependency of others**
- `dashboard` - Functionality for workflow of members, administrators etc.
- `departments` - Administration logic regarding a department such as shifts.
- `invitations` - Sending of invitation emails to create new user profiles that join accounts, departments etc.
- `membership` - Extends from Django allauth and contains the models of memberships and their organizations.
- `market` - Contains everything related to paying for stuff, invoicing, and creating and managing goods such as grocery bags.
- `permissions` - Models and decorators for permissions, **dependency of others**
- `profiles` - User information

5.1.1 Coupling

Currently, we expect applications to depend on each other's models. But it would get messy if templates, views, and static assets also started being interdependent.

Possible scheme: Only allow models and decorators to be interdependent, all other common elements should live in `eggplant.core`.

5.1.2 Philosophy

To quote Two Scoops of Django (1.8 version) that in turn quotes James Bennett (who in turn quotes Douglas McIlroy):

James Bennett volunteers as both a Django core developer and as its release manager. He taught us everything that we know about good Django app design. We quote him:

“The art of creating and maintaining a good Django app is that it should follow the truncated Unix philosophy according to Douglas McIlroy:

‘Write programs that do one thing and do it well.’

In essence, each app should be tightly focused on its task. If an app can’t be explained in a single sentence of moderate length, or you need to say ‘and’ more than once, it probably means the app is too big and should be broken up.

5.1.3 TODO

The scope of the applications may need to be consolidated.

5.2 Dependency overview

askdasdk

Python Reference

6.1 eggplant package

6.1.1 Subpackages

eggplant.accounts package

Submodules

eggplant.accounts.admin module

```
class eggplant.accounts.admin.AccountAdmin(model, admin_site)
    Bases: django.contrib.admin.options.ModelAdmin
    inlines = [<class 'eggplant.accounts.admin.AccountMembershipInline'>]
    media

class eggplant.accounts.admin.AccountCategoryAdmin(model, admin_site)
    Bases: django.contrib.admin.options.ModelAdmin
    media

class eggplant.accounts.admin.AccountMembershipInline(parent_model, admin_site)
    Bases: django.contrib.admin.options.TabularInline
    media
    model
        alias of Account_user_profiles
```

eggplant.accounts.models module

```
class eggplant.accounts.models.Account(id, name, category, department, start, exit, active)
    Bases: django.db.models.base.Model

exception DoesNotExist
    Bases: django.core.exceptions.ObjectDoesNotExist

exception Account.MultipleObjectsReturned
    Bases: django.core.exceptions.MultipleObjectsReturned
```

```
Account.category
Account.department
Account.get_next_by_start(*moreargs, **morekwargs)
Account.get_previous_by_start(*moreargs, **morekwargs)
Account.name_or_profile_names()
Account.objects = <django.db.models.manager.Manager object>
Account.payment_set
Account.user_profiles
Account.userprofilepermission_set

class eggplant.accounts.models.AccountCategory(id, name)
    Bases: django.db.models.base.Model

    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist

    exception AccountCategory.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned

    AccountCategory.accounts
    AccountCategory.departmentinvitation_set
    AccountCategory.objects = <django.db.models.manager.Manager object>
```

[eggplant.accounts.tests module](#)

[eggplant.accounts.urls module](#)

[eggplant.accounts.views module](#)

[Module contents](#)

[eggplant.core package](#)

[Subpackages](#)

[eggplant.core.templatetags package](#)

[Submodules](#)

[eggplant.core.templatetags.partition_slice module](#)

`eggplant.core.templatetags.partition_slice.partition` (*thelist*, *n*)

Break a list into *n* pieces. The last list may be larger than the rest if the list doesn't break cleanly. That is:

```
>>> l = range(10)

>>> partition(l, 2)
[[0, 1, 2, 3, 4], [5, 6, 7, 8, 9]]
```

```
>>> partition(1, 3)
[[0, 1, 2], [3, 4, 5], [6, 7, 8, 9]]

>>> partition(1, 4)
[[0, 1], [2, 3], [4, 5], [6, 7, 8, 9]]

>>> partition(1, 5)
[[0, 1], [2, 3], [4, 5], [6, 7], [8, 9]]
```

`eggplant.core.templatetags.partition_slice.partition_horizontal` (*thelist, n*)
 Break a list into *n* peices, but “horizontally.” That is, `partition_horizontal(range(10), 3)` gives:

```
[[1, 2, 3],
 [4, 5, 6],
 [7, 8, 9],
 [10]]
```

@see: <https://djangosnippets.org/snippets/6/>

Module contents

Submodules

`eggplant.core.context_processors` module

`eggplant.core.context_processors.coop_vars` (*request*)

`eggplant.core.tests` module

```
class eggplant.core.tests.UtilsTestCase (methodName='runTest')
    Bases: django.test.testcases.TestCase

    test_generate_upload_path()
    test_generate_upload_path_with_dir()
```

`eggplant.core.utils` module

```
eggplant.core.utils.absolute_url_reverse (url_name=None, **kwargs)
eggplant.core.utils.generate_upload_path (instance, filename, dirname=None)
    Generate path with random file name for FileField.
```

`eggplant.core.views` module

```
class eggplant.core.views.LoginRequiredMixin
    Bases: object

    TODO: This mixin should be replaced with django.contrib.auth.mixin.LoginRequiredMixin in django 1.9

    classmethod as_view (**kwargs)
```

eggplant.core.widgets module

```
class eggplant.core.widgets.MoneyWidget (*args, **kwargs)
    Bases: djmoney.forms.widgets.MoneyWidget
    media
    render (name, value, attrs=None)
```

Module contents

eggplant.dashboard package

Submodules

eggplant.dashboard.admin module

eggplant.dashboard.models module

eggplant.dashboard.tests module

```
class eggplant.dashboard.tests.TestDashboard (methodName='runTest')
    Bases: django.test.testcases.TestCase
    test_home ()
```

eggplant.dashboard.urls module

Dashboard urlconf, included by foodnet.urls

Final namespace of these URLs:

eggplant:dashboard:url_name

eggplant.dashboard.views module

```
eggplant.dashboard.views.home (request, *args, **kwargs)
    home page
```

Module contents

eggplant.departments package

Submodules

eggplant.departments.admin module

```
class eggplant.departments.admin.DepartmentAdmin (model, admin_site)
    Bases: django.contrib.admin.options.ModelAdmin
    inlines = [<class 'eggplant.departments.admin.DepartmentAdministratorInline'>]
```

media

```
class eggplant.departments.admin.DepartmentAdministratorInline (parent_model, admin_site)
```

Bases: django.contrib.admin.options.TabularInline

media

model

alias of DepartmentAdministrator

eggplant.departments.models module

```
class eggplant.departments.models.Department (id, name, slug, site)
```

Bases: django.db.models.base.Model

exception DoesNotExist

Bases: django.core.exceptions.ObjectDoesNotExist

exception Department.MultipleObjectsReturned

Bases: django.core.exceptions.MultipleObjectsReturned

Department.**accounts**

Department.**departmentadministrator_set**

Department.**departmentinvitation_set**

Department.**objects** = <django.db.models.manager.Manager object>

Department.**save** (***kwargs*)

Department.**site**

Department.**userprofilepermission_set**

```
class eggplant.departments.models.DepartmentAdministrator (id, department, profile, created)
```

Bases: django.db.models.base.Model

exception DoesNotExist

Bases: django.core.exceptions.ObjectDoesNotExist

exception DepartmentAdministrator.MultipleObjectsReturned

Bases: django.core.exceptions.MultipleObjectsReturned

DepartmentAdministrator.**department**

DepartmentAdministrator.**get_next_by_created** (**moreargs, **morekwargs*)

DepartmentAdministrator.**get_previous_by_created** (**moreargs, **morekwargs*)

DepartmentAdministrator.**objects** = <django.db.models.manager.Manager object>

DepartmentAdministrator.**profile**

`eggplant.departments.tests` module

`eggplant.departments.urls` module

`eggplant.departments.views` module

```
class eggplant.departments.views.DepartmentProfiles (**kwargs)
    Bases: eggplant.core.views.LoginRequiredMixin, django.views.generic.list.ListView

    get_queryset ()

    model
        alias of UserProfile

    paginate_by = 25

    template_name = 'eggplant/departments/profiles.html'

eggplant.departments.views.departments_profiles (request, *args, **kwargs)
```

Module contents

`eggplant.invitations` package

Submodules

`eggplant.invitations.admin` module

```
class eggplant.invitations.admin.DepartmentInvitationAdmin (model, admin_site)
    Bases: django.contrib.admin.options.ModelAdmin

    media
```

`eggplant.invitations.auth_backends` module

```
class eggplant.invitations.auth_backends.InvitationBackend
    Bases: django.contrib.auth.backends.ModelBackend

    authenticate (**credentials)
        Authenticate only invited users with not completed profile.
```

`eggplant.invitations.forms` module

```
class eggplant.invitations.forms.AcceptInvitationForm (data=None, files=None,
    auto_id=u'id_%s', pre-
    fix=None, initial=None,
    error_class=<class
    'django.forms.utils.ErrorList'>,
    label_suffix=None,
    empty_permitted=False)

    Bases: django.forms.forms.Form

    base_fields = OrderedDict([('captcha', <captcha.fields.ReCaptchaField object at 0x7f5652f34210>)])
```



```

    declared_fields = OrderedDict([('captcha', <captcha.fields.ReCaptchaField object at 0x7f5652f34210>)])
    media

class eggplant.invitations.forms.DepartmentInvitationForm (data=None, files=None,
    auto_id=u'id_%s', pre-
    fix=None, initial=None,
    error_class=<class
    'django.forms.utils.ErrorList'>,
    label_suffix=None,
    empty_permitted=False,
    instance=None)

Bases: django.forms.models.ModelForm

class Meta

    fields = ['department', 'account_category', 'email']

    model
        alias of DepartmentInvitation

DepartmentInvitationForm.base_fields = OrderedDict([('department', <django.forms.models.ModelChoiceField object at 0x7f5652f34210>)])
DepartmentInvitationForm.declared_fields = OrderedDict([('email', <django.forms.fields.EmailField object at 0x7f5652f34210>)])
DepartmentInvitationForm.media

```

eggplant.invitations.models module

```

class eggplant.invitations.models.DepartmentInvitation (id, email, accepted, ac-
    cepted_at, verification_key,
    created, invited_by, depart-
    ment, account_category)

Bases: eggplant.invitations.models.InvitationBase

exception DoesNotExist
    Bases: django.core.exceptions.ObjectDoesNotExist

exception DepartmentInvitation.MultipleObjectsReturned
    Bases: django.core.exceptions.MultipleObjectsReturned

DepartmentInvitation.account_category

DepartmentInvitation.department

DepartmentInvitation.get_next_by_created (*moreargs, **kwargs)

DepartmentInvitation.get_previous_by_created (*moreargs, **kwargs)

DepartmentInvitation.invited_by

DepartmentInvitation.objects = <django.db.models.manager.Manager object>

class eggplant.invitations.models.InvitationBase (*args, **kwargs)
    Bases: django.db.models.base.Model

    class Meta

        abstract = False

        permissions = (('can_invite', 'Can send invitation'),)

```

```
InvitationBase.get_next_by_created(*moreargs, **morekwargs)
InvitationBase.get_previous_by_created(*moreargs, **morekwargs)
InvitationBase.invited_by
eggplant.invitations.models.send_email_invitation(sender, instance, created,
                                                    **kwargs)
TODO: Implement an HTML message, commenting out the html_* lines below
```

eggplant.invitations.tests module

eggplant.invitations.urls module

eggplant.invitations.utils module

```
eggplant.invitations.utils.create_verified_user(invitation)
```

eggplant.invitations.views module

```
exception eggplant.invitations.views.AlreadyAcceptedInvitationException
    Bases: exceptions.Exception
eggplant.invitations.views.accept_invitation(request, verification_key)
    Accept invitation.
eggplant.invitations.views.do_accept_invitation(request, invitation)
eggplant.invitations.views.invite(request, *args, **kwargs)
    Invite a new email address.
```

Module contents

eggplant.market package

Subpackages

eggplant.market.models package

Submodules

eggplant.market.models.cart module

```
class eggplant.market.models.cart.Basket(id, user, created, status)
    Bases: django.db.models.base.Model
    CHECKEDOUT = 'checked-out'
    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist
    exception Basket.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned
    Basket.OPEN = 'open'
```

```

Basket.STATUSES = (('open', 'open'), ('checked-out', 'checked-out'))

Basket.add_to_items (product=None, quantity=1, delivery_date=None)

Basket.do_checkout (*args, **kwargs)

Basket.get_items_count ()

Basket.get_next_by_created (*moreargs, **morekwargs)

Basket.get_previous_by_created (*moreargs, **morekwargs)

Basket.get_status_display (*moreargs, **morekwargs)

Basket.get_total_amount ()

Basket.items

Basket.objects = <eggplant.market.models.cart.BasketManager object>

Basket.remove_from_items (product=None, quantity=1, delivery_date=None)

Basket.user
class eggplant.market.models.cart.BasketItem (id, basket, product, quantity, delivery_date)
    Bases: django.db.models.base.Model

    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist

    exception BasketItem.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned

    BasketItem.basket

    BasketItem.objects = <django.db.models.manager.Manager object>

    BasketItem.product

class eggplant.market.models.cart.BasketManager
    Bases: django.db.models.manager.Manager

    open_for_user (user)
        Get open basket for a given user. This is just a wrapper around get_or_create so we can add more default
        kwargs or logic to basket in one place - perhaps a check if user payed some fees(?)...

eggplant.market.models.inventory module
class eggplant.market.models.inventory.Product (id, title, description, category,
                                                price_currency, price, stock, tax, enabled,
                                                image)
    Bases: django.db.models.base.Model

    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist

    exception Product.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned

    Product.basketitem_set

    Product.category

    Product.get_price_currency_display (*moreargs, **morekwargs)

```

```
Product.image
    Just like the FileDescriptor, but for ImageFields. The only difference is assigning the width/height to the
    width_field/height_field, if appropriate.

Product.objects = <djmoney.models.managers.MoneyManager object>

Product.price

Product.tax
class eggplant.market.models.inventory.ProductCategory (id, title, description, enabled)
    Bases: django.db.models.base.Model

    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist

    exception ProductCategory.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned

    ProductCategory.objects = <django.db.models.manager.Manager object>

    ProductCategory.product_set

class eggplant.market.models.inventory.ProductTax (id, title, description, enabled, tax)
    Bases: django.db.models.base.Model

    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist

    exception ProductTax.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned

    ProductTax.objects = <django.db.models.manager.Manager object>

    ProductTax.product_set

eggplant.market.models.inventory.do_upload_product_image (inst, filename)

eggplant.market.models.listeners module Notice: getpaid calls it “order” objects, however since our payments
app does not model orders, we also call this “payment” in eggplant.payments.models

eggplant.market.models.listeners.new_payment_listener (sender, order=None, pay-
                                                         ment=None, **kwargs)

    Log how many and which payments were made.

eggplant.market.models.listeners.new_payment_query_listener (sender, order=None,
                                                                payment=None,
                                                                **kwargs)

    Fills in required payment details.

eggplant.market.models.listeners.order_additional_validation_listener (sender,
                                                                           re-
                                                                           quest=None,
                                                                           or-
                                                                           der=None,
                                                                           back-
                                                                           end=None,
                                                                           **kwargs)

    Custom validation.
```

```
eggplant.market.models.listeners.payment_status_changed_listener(sender,
                                                                    instance,
                                                                    old_status,
                                                                    new_status,
                                                                    **kwargs)
```

Here we will actually do something, when payment is accepted. E.g. lets change an order status based on payment status.

```
eggplant.market.models.listeners.user_data_query_listener(sender, order=None,
                                                           user_data=None,
                                                           **kwargs)
```

Fills in required user details.

eggplant.market.models.payment module

```
class eggplant.market.models.payment.Payment(id, amount_currency, amount, account, created)
```

Bases: `django.db.models.base.Model`

exception DoesNotExist

Bases: `django.core.exceptions.ObjectDoesNotExist`

exception Payment.MultipleObjectsReturned

Bases: `django.core.exceptions.MultipleObjectsReturned`

`Payment.account`

`Payment.amount`

`Payment.get_absolute_url()`

`Payment.get_amount_currency_display(*moreargs, **morekwargs)`

`Payment.get_last_payment_status()`

`Payment.get_next_by_created(*moreargs, **morekwargs)`

`Payment.get_previous_by_created(*moreargs, **morekwargs)`

`Payment.is_ready_for_payment()`

`Payment.objects = <djmoney.models.managers.MoneyManager object>`

`Payment.payments`

Module contents

eggplant.market.templatetags package

Submodules

eggplant.market.templatetags.cart_tags module

```
eggplant.market.templatetags.cart_tags.cart_action(context, action, product_id=None, quantity=None,
                                                       delivery_date=None)
```

Module contents

eggplant.market.views package

Submodules

eggplant.market.views.cart module

```
class eggplant.market.views.cart.AddToCart (**kwargs)
    Bases: eggplant.market.views.cart.BaseCartActionView

    form_valid (form)

class eggplant.market.views.cart.BaseCartActionView (**kwargs)
    Bases: django.views.generic.edit.FormView

    form_class
        alias of BasketItemForm

    form_invalid (form)

    form_valid (form)

    get_template_names ()

    success_url = <django.utils.functional.__proxy__ object>

class eggplant.market.views.cart.RemoveFromCart (**kwargs)
    Bases: eggplant.market.views.cart.BaseCartActionView

    form_valid (form)

eggplant.market.views.cart.add_to_cart (request, *args, **kwargs)

eggplant.market.views.cart.cart_details (request, *args, **kwargs)

eggplant.market.views.cart.checkout (request, *args, **kwargs)

eggplant.market.views.cart.remove_from_cart (request, *args, **kwargs)
```

eggplant.market.views.inventory module

```
eggplant.market.views.inventory.add_product (request, *args, **kwargs)
eggplant.market.views.inventory.market_home (request, *args, **kwargs)
```

eggplant.market.views.payment module

```
class eggplant.market.views.payment.PaymentView (**kwargs)
    Bases: eggplant.core.views.LoginRequiredMixin, django.views.generic.detail.DetailView

    dispatch (func)

    get_context_data (**kwargs)

    get_queryset ()

    model
        alias of Payment

    template_name = 'eggplant/market/payment_detail.html'

eggplant.market.views.payment.payment_accepted (request, pk=None)

eggplant.market.views.payment.payment_detail (request, *args, **kwargs)

eggplant.market.views.payment.payment_info (request, *args, **kwargs)
```

```
eggplant.market.views.payment.payment_list (request, *args, **kwargs)
eggplant.market.views.payment.payment_rejected (request, *args, **kwargs)
```

Module contents

Submodules

eggplant.market.admin module

```
class eggplant.market.admin.ProductAdmin (model, admin_site)
    Bases: django.contrib.admin.options.ModelAdmin

    list_display = ('title', 'category', 'stock', 'price', 'enabled')
    list_filter = ('enabled', 'category__title')
    media

class eggplant.market.admin.ProductCategoryAdmin (model, admin_site)
    Bases: django.contrib.admin.options.ModelAdmin

    list_display = ('title', 'enabled')
    list_filter = ('enabled',)
    media
```

eggplant.market.filters module

```
class eggplant.market.filters.LinksGroupWidget (attrs=None, choices=())
    Bases: django_filters.widgets.LinkWidget

    media
    option_string()
    render (name, value, attrs=None, choices=())
    render_option (name, selected_choices, option_value, option_label)

class eggplant.market.filters.ProductFilter (*args, **kwargs)
    Bases: django_filters.filterset.FilterSet

    class Meta

        fields = ['category']
        model
            alias of Product

    ProductFilter.base_filters = OrderedDict([('category', <django_filters.filters.ModelChoiceFilter object at 0x7f5...
    ProductFilter.declared_filters = OrderedDict([('category', <django_filters.filters.ModelChoiceFilter object at
```

eggplant.market.forms module

```
class eggplant.market.forms.BasketItemForm(data=None, files=None, auto_id=u'id_%s', pre-
                                           fix=None, initial=None, error_class=<class
                                           'django.forms.utils.ErrorList'>, label_
                                           suffix=None, empty_permitted=False)

Bases: django.forms.forms.Form

base_fields = OrderedDict([('product', <django.forms.models.ModelChoiceField object at 0x7f5652e28550>), ('quantity', <django.forms.models.ModelChoiceField object at 0x7f5652e28550>)])
declared_fields = OrderedDict([('product', <django.forms.models.ModelChoiceField object at 0x7f5652e28550>), ('quantity', <django.forms.models.ModelChoiceField object at 0x7f5652e28550>)])
media

class eggplant.market.forms.ProductForm(data=None, files=None, auto_id=u'id_%s', pre-
                                         fix=None, initial=None, error_class=<class
                                         'django.forms.utils.ErrorList'>, label_
                                         suffix=None, empty_permitted=False, instance=None)

Bases: django.forms.models.ModelForm

class Meta

    fields = ['title', 'description', 'price', 'category', 'tax', 'stock']
    model =
        alias of Product
    widgets = {'price': <eggplant.core.widgets.MoneyWidget object at 0x7f5652e28a50>}

ProductForm.base_fields = OrderedDict([('title', <django.forms.fields.CharField object at 0x7f5652e28c50>), ('description', <django.forms.fields.CharField object at 0x7f5652e28c50>), ('category', <django.forms.fields.ModelChoiceField object at 0x7f5652e28c50>), ('tax', <django.forms.fields.DecimalField object at 0x7f5652e28c50>), ('stock', <django.forms.fields.IntegerField object at 0x7f5652e28c50>)])
ProductForm.declared_fields = OrderedDict()
ProductForm.media
```

eggplant.market.tests module

eggplant.market.urls module

Module contents

eggplant.permissions package

Submodules

eggplant.permissions.admin module

eggplant.permissions.models module

Permission philosophy:

- Be explicit! Uses boolean fields for specific tasks
- Be SQL friendly, create permissions that are nice to work with in query set lookups
- Put logic in decorators


```
class eggplant.permissions.models.Permission (*args, **kwargs)
```

```
    Bases: django.db.models.base.Model
```

Permission roles are a set of permissions. Permissions are modeled as booleans in this model.

What can a user do? Examples from discussion of different roles:

A user is a superuser: Don't put it here – THIS IS FOR THE global User.is_superuser field!!

A user can create and manage all departments.: E.g. someone from the central commission can add a new department and close an existing one.

A user is a department manager: Can create and delete accounts and user profiles for everyone in a department.

A user is an "intro vagt": Someone who can create new accounts

A user is a team link: Can manage volunteer shifts

A user owns an account: Can add credit card, can add others to the account

CONCEPT OF THIS MODEL: Create boolean fields for different permissions, create lots of them! We want to be very explicit.

```
exception DoesNotExist
```

```
    Bases: django.core.exceptions.ObjectDoesNotExist
```

```
exception Permission.MultipleObjectsReturned
```

```
    Bases: django.core.exceptions.MultipleObjectsReturned
```

```
Permission.objects = <django.db.models.manager.Manager object>
```

```
Permission.save (*args, **kwargs)
```

```
Permission.userprofile_set
```

```
Permission.userprofilepermission_set
```

```
class eggplant.permissions.models.UserProfilePermission (*args, **kwargs)
```

```
    Bases: django.db.models.base.Model
```

Link between a user profile and a set of permissions (a role).

Example 1:

```
Check if a user has user creation access to a department:
```

```
can_add_users = department.userprofilepermission_set.filter(
    user_profile__user=request.user,
    permission__can_add_user_profiles=True
).exists()
```

```
if can_add_users:
    obama_speech = "YES WE CAN"
    print(obama_speech)
```

Example 2:

```
Check if user can manage an account, like changing the data:
```

```
can_change_account = account.userprofilepermission_set.filter(
    user_profile__user=request.user,
    permission__can_change_accounts=True,
)
```

```
if not can_change_account:
    return HttpNotAllowed("piss off")
```

TODO: Create decorators to manage this easier!

exception DoesNotExist

Bases: `django.core.exceptions.ObjectDoesNotExist`

exception UserProfilePermission.MultipleObjectsReturned

Bases: `django.core.exceptions.MultipleObjectsReturned`

`UserProfilePermission.account`

`UserProfilePermission.department`

`UserProfilePermission.objects = <django.db.models.manager.Manager object>`

`UserProfilePermission.permission`

`UserProfilePermission.user_profile`

eggplant.permissions.tests module

Module contents

eggplant.profiles package

Submodules

eggplant.profiles.admin module

class `eggplant.profiles.admin.UserProfileAdmin` (*model*, *admin_site*)

Bases: `django.contrib.admin.options.ModelAdmin`

`media`

eggplant.profiles.forms module

class `eggplant.profiles.forms.NewUserSetPasswordForm` (*user=None*, **args*, ***kwargs*)

Bases: `allauth.account.forms.SetPasswordForm`

`base_fields = OrderedDict([('password1', <allauth.account.forms.SetPasswordField object at 0x7f5652e89390>), ('password2', <allauth.account.forms.SetPasswordField object at 0x7f5652e89390>)])`

`declared_fields = OrderedDict([('password1', <allauth.account.forms.SetPasswordField object at 0x7f5652e89390>), ('password2', <allauth.account.forms.SetPasswordField object at 0x7f5652e89390>)])`

`media`

`save (*args, **kwargs)`

class `eggplant.profiles.forms.ProfileForm` (*data=None*, *files=None*, *auto_id=u'id_%s'*, *prefix=None*, *initial=None*, *error_class=<class 'django.forms.utils.ErrorList'>*, *label_suffix=None*, *empty_permitted=False*)

Bases: `django.forms.forms.Form`

`base_fields = OrderedDict([('first_name', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('middle_name', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('last_name', <django.forms.fields.CharField object at 0x7f5652e89c90>)])`

`declared_fields = OrderedDict([('first_name', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('middle_name', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('last_name', <django.forms.fields.CharField object at 0x7f5652e89c90>)])`

`media`

```
class eggplant.profiles.forms.SignupForm (data=None,      files=None,      auto_id=u'id_%s',
                                         prefix=None,    initial=None,    error_class=<class
                                         'django.forms.utils.ErrorList'>, label_suffix=None,
                                         empty_permitted=False)
    Bases: eggplant.profiles.forms.ProfileForm
    base_fields = OrderedDict([('first_name', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('middle_name', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('last_name', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('email', <django.forms.fields.EmailField object at 0x7f5652e89c90>), ('password', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('password2', <django.forms.fields.CharField object at 0x7f5652e89c90>)])
    clean_email ()
        Check if user is already registered and if so raise validation error.
        It may be considered a security hole to inform if a user is registered or not but it improves usability.
    declared_fields = OrderedDict([('first_name', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('middle_name', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('last_name', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('email', <django.forms.fields.EmailField object at 0x7f5652e89c90>), ('password', <django.forms.fields.CharField object at 0x7f5652e89c90>), ('password2', <django.forms.fields.CharField object at 0x7f5652e89c90>)])
    media
```

eggplant.profiles.middleware module

```
class eggplant.profiles.middleware.NewUserForceProfileMiddleware
    Bases: object
    process_request (request)
```

eggplant.profiles.models module

```
class eggplant.profiles.models.UserProfile (id, user, middle_name, address, postcode, city, tel, tel2, sex, photo, created, changed)
    Bases: django.db.models.base.Model
    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist
    UserProfile.FEMALE = 'female'
    UserProfile.MALE = 'male'
    exception UserProfile.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned
    UserProfile.OTHER = 'other'
    UserProfile.SEX_CHOICES = ((' ', '—'), ('female', 'female'), ('male', 'male'), ('other', 'other'))
    UserProfile.accounts
    UserProfile.active_accounts ()
        Returns the active accounts.
    UserProfile.administrator_for
    UserProfile.can_be_edited_by (user_profile)
    UserProfile.full_name
        Returns member's full name.
    UserProfile.get_next_by_changed (*moreargs, **morekwargs)
    UserProfile.get_next_by_created (*moreargs, **morekwargs)
    UserProfile.get_previous_by_changed (*moreargs, **morekwargs)
    UserProfile.get_previous_by_created (*moreargs, **morekwargs)
```

```
UserProfile.get_sex_display(*moreargs, **morekwargs)
UserProfile.has_admin_permission(department)
classmethod UserProfile.in_department(department, only_active_accounts=True)
    Returns the user profiles linked to the given department via: UserProfile -> Account -> DepartmentMem-
    bership -> Department
UserProfile.is_complete()
UserProfile.objects = <django.db.models.manager.Manager object>
UserProfile.permissions
UserProfile.photo
    Just like the FileDescriptor, but for ImageFields. The only difference is assigning the width/height to the
    width_field/height_field, if appropriate.
UserProfile.photo_url_or_default()
UserProfile.user
UserProfile.userprofilepermission_set
eggplant.profiles.models.create_user_profile(sender, instance, created, **kwargs)
    Every time a user is created, we automatically create a profile for the user.
```

eggplant.profiles.tests module

eggplant.profiles.urls module

eggplant.profiles.views module

```
class eggplant.profiles.views.NewUserPassword(**kwargs)
    Bases: eggplant.core.views.LoginRequiredMixin, allauth.account.views.PasswordSetView
    Set password only for a new user. Existing users can use password change.
    dispatch(*args, **kwargs)
        Overrides PasswordSetView.dispatch with the View.dispatch
    form_class
        alias of NewUserSetPasswordForm
    form_valid(form)
    get(request, *args, **kwargs)
    get_authenticated_redirect_url(*args, **kwargs)
    get_success_url(*args, **kwargs)
    post(request, *args, **kwargs)
    success_url = <django.utils.functional.__proxy__ object>
class eggplant.profiles.views.Profile(**kwargs)
    Bases: eggplant.core.views.LoginRequiredMixin, django.views.generic.edit.FormView
    Profile form view.
    form_class
        alias of ProfileForm
```

```

    form_valid(form)
    get_context_data(**kwargs)
    get_initial()
    get_object(queryset=None)
    success_url = <django.utils.functional.__proxy__ object>
    template_name = 'eggplant/profiles/profile_detail.html'
eggplant.profiles.views.signup(request)

```

Module contents

eggplant.roles package

Subpackages

eggplant.roles.templatetags package

Submodules

eggplant.roles.templatetags.active_url module

```
eggplant.roles.templatetags.active_url.active(context, pattern_or_urlname)
```

Module contents

Submodules

eggplant.roles.admin module

```

class eggplant.roles.admin.RoleAssignment(model, admin_site)
    Bases: django.contrib.admin.options.ModelAdmin
    media

```

eggplant.roles.models module

```

class eggplant.roles.models.RoleAssignment(id, role, user)
    Bases: django.db.models.base.Model
    ACCOUNTANT = 'accountant'
    CASHIER = 'cashier'
    COMMUNICATOR = 'communicator'
    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist
    exception RoleAssignment.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned

```

```
RoleAssignment.PACKER = 'packer'
RoleAssignment.PURCHASER = 'purchaser'
RoleAssignment.ROLE_CHOICES = (('purchaser', <django.utils.functional.__proxy__ object at 0x7f5654057c10>), ('c
RoleAssignment.get_role_display(*moreargs, **morekwargs)
RoleAssignment.objects = <django.db.models.manager.Manager object>
RoleAssignment.user
```

eggplant.roles.urls module

eggplant.roles.views module

```
eggplant.roles.views.accountant(request)
eggplant.roles.views.cashier(request)
eggplant.roles.views.communicator(request)
eggplant.roles.views.packer(request)
eggplant.roles.views.purchaser(request)
eggplant.roles.views.role(request, role)
```

Module contents

6.1.2 Submodules

6.1.3 eggplant.factories module

6.1.4 eggplant.urls module

6.1.5 Module contents

Eggplant is an open source food coop platform.

It allows buying and selling food products directly between farmers and consumers, cutting out the middle man, transportation, packaging etc.

Eggplant is good for the environment and good for you.

Eggplant

Eggplant is an open source web application that provides simple and flexible infrastructure for organizing food coops and other local community-driven projects.

How to contribute

[Read our documentation](#) to get started reporting bugs, developing code etc. The project description, organisation and goals are on our website [eggplant.dk](#). The list of tickets is available on our [GitHub project](#).

Write code, write tests, have fun.

Get in touch

For AFK stuff, you can join us in Copenhagen on [meetup.com](#).

We use Slack for ad-hoc communication: [Click to recieve an invitation](#). The techical discussion takes place on [Slack#teambblue](#). The design and organisational issues can also be raised on [Slack#teamgreen](#).

Indices and tables

- `genindex`
- `modindex`
- `search`

e

- eggplant, 34
- eggplant.accounts, 16
- eggplant.accounts.admin, 15
- eggplant.accounts.models, 15
- eggplant.accounts.tests, 16
- eggplant.accounts.urls, 16
- eggplant.accounts.views, 16
- eggplant.core, 18
- eggplant.core.context_processors, 17
- eggplant.core.templatetags, 17
- eggplant.core.templatetags.partition_slice, 16
- eggplant.core.tests, 17
- eggplant.core.utils, 17
- eggplant.core.views, 17
- eggplant.core.widgets, 18
- eggplant.dashboard, 18
- eggplant.dashboard.admin, 18
- eggplant.dashboard.models, 18
- eggplant.dashboard.tests, 18
- eggplant.dashboard.urls, 18
- eggplant.dashboard.views, 18
- eggplant.departments, 20
- eggplant.departments.admin, 18
- eggplant.departments.models, 19
- eggplant.departments.urls, 20
- eggplant.departments.views, 20
- eggplant.invitations, 22
- eggplant.invitations.admin, 20
- eggplant.invitations.auth_backends, 20
- eggplant.invitations.forms, 20
- eggplant.invitations.models, 21
- eggplant.invitations.urls, 22
- eggplant.invitations.utils, 22
- eggplant.invitations.views, 22
- eggplant.market, 28
- eggplant.market.admin, 27
- eggplant.market.filters, 27
- eggplant.market.forms, 28
- eggplant.market.models, 25
- eggplant.market.models.cart, 22
- eggplant.market.models.inventory, 23
- eggplant.market.models.listeners, 24
- eggplant.market.models.payment, 25
- eggplant.market.templatetags, 25
- eggplant.market.templatetags.cart_tags, 25
- eggplant.market.urls, 28
- eggplant.market.views, 27
- eggplant.market.views.cart, 26
- eggplant.market.views.inventory, 26
- eggplant.market.views.payment, 26
- eggplant.permissions, 30
- eggplant.permissions.admin, 28
- eggplant.permissions.models, 28
- eggplant.permissions.tests, 30
- eggplant.profiles, 33
- eggplant.profiles.admin, 30
- eggplant.profiles.forms, 30
- eggplant.profiles.middleware, 31
- eggplant.profiles.models, 31
- eggplant.profiles.urls, 32
- eggplant.profiles.views, 32
- eggplant.roles, 34
- eggplant.roles.admin, 33
- eggplant.roles.models, 33
- eggplant.roles.templatetags, 33
- eggplant.roles.templatetags.active_url, 33
- eggplant.roles.urls, 34
- eggplant.roles.views, 34
- eggplant.urls, 34

A

- `absolute_url_reverse()` (in module `eggplant.core.utils`), 17
- `abstract` (`eggplant.invitations.models.InvitationBase.Meta` attribute), 21
- `accept_invitation()` (in module `eggplant.invitations.views`), 22
- `AcceptInvitationForm` (class in `eggplant.invitations.forms`), 20
- `Account` (class in `eggplant.accounts.models`), 15
- `account` (`eggplant.market.models.payment.Payment` attribute), 25
- `account` (`eggplant.permissions.models.UserProfilePermission` attribute), 30
- `Account.DoesNotExist`, 15
- `Account.MultipleObjectsReturned`, 15
- `account_category` (`eggplant.invitations.models.DepartmentInvitation` attribute), 21
- `AccountAdmin` (class in `eggplant.accounts.admin`), 15
- `ACCOUNTANT` (`eggplant.roles.models.RoleAssignment` attribute), 33
- `accountant()` (in module `eggplant.roles.views`), 34
- `AccountCategory` (class in `eggplant.accounts.models`), 16
- `AccountCategory.DoesNotExist`, 16
- `AccountCategory.MultipleObjectsReturned`, 16
- `AccountCategoryAdmin` (class in `eggplant.accounts.admin`), 15
- `AccountMembershipInline` (class in `eggplant.accounts.admin`), 15
- `accounts` (`eggplant.accounts.models.AccountCategory` attribute), 16
- `accounts` (`eggplant.departments.models.Department` attribute), 19
- `accounts` (`eggplant.profiles.models.UserProfile` attribute), 31
- `active()` (in module `eggplant.roles templatetags.active_url`), 33
- `active_accounts()` (`eggplant.profiles.models.UserProfile` method), 31
- `add_product()` (in module `eggplant.market.views.inventory`), 26
- `add_to_cart()` (in module `eggplant.market.views.cart`), 26
- `add_to_items()` (`eggplant.market.models.cart.Basket` method), 23
- `AddToCart` (class in `eggplant.market.views.cart`), 26
- `administrator_for` (`eggplant.profiles.models.UserProfile` attribute), 31
- `AlreadyAcceptedInvitationException`, 22
- `amount` (`eggplant.market.models.payment.Payment` attribute), 25
- `as_view()` (`eggplant.core.views.LoginRequiredMixin` class method), 17
- `authenticate()` (`eggplant.invitations.auth_backends.InvitationBackend` method), 20

B

- `base_fields` (`eggplant.invitations.forms.AcceptInvitationForm` attribute), 20
- `base_fields` (`eggplant.invitations.forms.DepartmentInvitationForm` attribute), 21
- `base_fields` (`eggplant.market.forms.BasketItemForm` attribute), 28
- `base_fields` (`eggplant.market.forms.ProductForm` attribute), 28
- `base_fields` (`eggplant.profiles.forms.NewUserSetPasswordForm` attribute), 30
- `base_fields` (`eggplant.profiles.forms.ProfileForm` attribute), 30
- `base_fields` (`eggplant.profiles.forms.SignupForm` attribute), 31
- `base_filters` (`eggplant.market.filters.ProductFilter` attribute), 27
- `BaseCartActionView` (class in `eggplant.market.views.cart`), 26
- `Basket` (class in `eggplant.market.models.cart`), 22
- `basket` (`eggplant.market.models.cart.BasketItem` attribute), 23
- `Basket.DoesNotExist`, 22
- `Basket.MultipleObjectsReturned`, 22
- `BasketItem` (class in `eggplant.market.models.cart`), 23
- `BasketItem.DoesNotExist`, 23

BasketItem.MultipleObjectsReturned, 23

basketitem_set (eggplant.market.models.inventory.Product attribute), 23

BasketItemForm (class in eggplant.market.forms), 28

BasketManager (class in eggplant.market.models.cart), 23

C

can_be_edited_by() (eggplant.profiles.models.UserProfile method), 31

cart_action() (in module eggplant.market.templatetags.cart_tags), 25

cart_details() (in module eggplant.market.views.cart), 26

CASHIER (eggplant.roles.models.RoleAssignment attribute), 33

cashier() (in module eggplant.roles.views), 34

category (eggplant.accounts.models.Account attribute), 15

category (eggplant.market.models.inventory.Product attribute), 23

CHECKEDOUT (eggplant.market.models.cart.Basket attribute), 22

checkout() (in module eggplant.market.views.cart), 26

clean_email() (eggplant.profiles.forms.SignupForm method), 31

COMMUNICATOR (eggplant.roles.models.RoleAssignment attribute), 33

communicator() (in module eggplant.roles.views), 34

coop_vars() (in module eggplant.core.context_processors), 17

create_user_profile() (in module eggplant.profiles.models), 32

create_verified_user() (in module eggplant.invitations.utils), 22

D

declared_fields (eggplant.invitations.forms.AcceptInvitationForm attribute), 20

declared_fields (eggplant.invitations.forms.DepartmentInvitationForm attribute), 21

declared_fields (eggplant.market.forms.BasketItemForm attribute), 28

declared_fields (eggplant.market.forms.ProductForm attribute), 28

declared_fields (eggplant.profiles.forms.NewUserSetPasswordForm attribute), 30

declared_fields (eggplant.profiles.forms.ProfileForm attribute), 30

declared_fields (eggplant.profiles.forms.SignupForm attribute), 31

declared_filters (eggplant.market.filters.ProductFilter attribute), 27

Department (class in eggplant.departments.models), 19

department (eggplant.accounts.models.Account attribute), 16

department (eggplant.departments.models.DepartmentAdministrator attribute), 19

department (eggplant.invitations.models.DepartmentInvitation attribute), 21

department (eggplant.permissions.models.UserProfilePermission attribute), 30

Department.DoesNotExist, 19

Department.MultipleObjectsReturned, 19

DepartmentAdmin (class in eggplant.departments.admin), 18

DepartmentAdministrator (class in eggplant.departments.models), 19

DepartmentAdministrator.DoesNotExist, 19

DepartmentAdministrator.MultipleObjectsReturned, 19

departmentadministrator_set (eggplant.departments.models.Department attribute), 19

DepartmentAdministratorInline (class in eggplant.departments.admin), 19

DepartmentInvitation (class in eggplant.invitations.models), 21

DepartmentInvitation.DoesNotExist, 21

DepartmentInvitation.MultipleObjectsReturned, 21

departmentinvitation_set (eggplant.accounts.models.AccountCategory attribute), 16

departmentinvitation_set (eggplant.departments.models.Department attribute), 19

DepartmentInvitationAdmin (class in eggplant.invitations.admin), 20

DepartmentInvitationForm (class in eggplant.invitations.forms), 21

DepartmentInvitationForm.Meta (class in eggplant.invitations.forms), 21

DepartmentProfiles (class in eggplant.departments.views), 20

departments_profiles() (in module eggplant.departments.views), 20

dispatch() (eggplant.market.views.payment.PaymentView method), 26

dispatch() (eggplant.profiles.views.NewUserPasswordForm method), 32

do_accept_invitation() (in module eggplant.invitations.views), 22

do_checkout() (eggplant.market.models.cart.Basket method), 23

do_upload_product_image() (in module eggplant.market.models.inventory), 24

E

eggplant (module), 34

eggplant.accounts (module), 16

eggplant.accounts.admin (module), 15

eggplant.accounts.models (module), 15

eggplant.accounts.tests (module), 16

eggplant.accounts.urls (module), 16

eggplant.accounts.views (module), 16

eggplant.core (module), 18

eggplant.core.context_processors (module), 17

eggplant.core.templatetags (module), 17

eggplant.core.templatetags.partition_slice (module), 16

eggplant.core.tests (module), 17

eggplant.core.utils (module), 17

eggplant.core.views (module), 17

eggplant.core.widgets (module), 18

eggplant.dashboard (module), 18

eggplant.dashboard.admin (module), 18

eggplant.dashboard.models (module), 18

eggplant.dashboard.tests (module), 18

eggplant.dashboard.urls (module), 18

eggplant.dashboard.views (module), 18

eggplant.departments (module), 20

eggplant.departments.admin (module), 18

eggplant.departments.models (module), 19

eggplant.departments.urls (module), 20

eggplant.departments.views (module), 20

eggplant.invitations (module), 22

eggplant.invitations.admin (module), 20

eggplant.invitations.auth_backends (module), 20

eggplant.invitations.forms (module), 20

eggplant.invitations.models (module), 21

eggplant.invitations.urls (module), 22

eggplant.invitations.utils (module), 22

eggplant.invitations.views (module), 22

eggplant.market (module), 28

eggplant.market.admin (module), 27

eggplant.market.filters (module), 27

eggplant.market.forms (module), 28

eggplant.market.models (module), 25

eggplant.market.models.cart (module), 22

eggplant.market.models.inventory (module), 23

eggplant.market.models.listeners (module), 24

eggplant.market.models.payment (module), 25

eggplant.market.templatetags (module), 25

eggplant.market.templatetags.cart_tags (module), 25

eggplant.market.urls (module), 28

eggplant.market.views (module), 27

eggplant.market.views.cart (module), 26

eggplant.market.views.inventory (module), 26

eggplant.market.views.payment (module), 26

eggplant.permissions (module), 30

eggplant.permissions.admin (module), 28

eggplant.permissions.models (module), 28

eggplant.permissions.tests (module), 30

eggplant.profiles (module), 33

eggplant.profiles.admin (module), 30

eggplant.profiles.forms (module), 30

eggplant.profiles.middleware (module), 31

eggplant.profiles.models (module), 31

eggplant.profiles.urls (module), 32

eggplant.profiles.views (module), 32

eggplant.roles (module), 34

eggplant.roles.admin (module), 33

eggplant.roles.models (module), 33

eggplant.roles.templatetags (module), 33

eggplant.roles.templatetags.active_url (module), 33

eggplant.roles.urls (module), 34

eggplant.roles.views (module), 34

eggplant.urls (module), 34

F

FEMALE (eggplant.profiles.models.UserProfile attribute), 31

fields (eggplant.invitations.forms.DepartmentInvitationForm.Meta attribute), 21

fields (eggplant.market.filters.ProductFilter.Meta attribute), 27

fields (eggplant.market.forms.ProductForm.Meta attribute), 28

form_class (eggplant.market.views.cart.BaseCartActionView attribute), 26

form_class (eggplant.profiles.views.NewUserPassword attribute), 32

form_invalid() (eggplant.market.views.cart.BaseCartActionView method), 26

form_valid() (eggplant.market.views.cart.AddToCart method), 26

form_valid() (eggplant.market.views.cart.BaseCartActionView method), 26

form_valid() (eggplant.market.views.cart.RemoveFromCart method), 26

form_valid() (eggplant.profiles.views.NewUserPassword method), 32

form_valid() (eggplant.profiles.views.Profile method), 32

full_name (eggplant.profiles.models.UserProfile attribute), 31

G

generate_upload_path() (in module eggplant.core.utils), 17

get() (eggplant.profiles.views.NewUserPassword method), 32

get_absolute_url() (eggplant.market.models.payment.Payment method), 25

<code>get_amount_currency_display()</code> (eggplant.market.models.payment.Payment method), 25	<code>get_previous_by_created()</code> (eggplant.market.models.payment.Payment method), 25
<code>get_authenticated_redirect_url()</code> (eggplant.profiles.views.NewUserPassword method), 32	<code>get_previous_by_created()</code> (eggplant.profiles.models.UserProfile method), 31
<code>get_context_data()</code> (eggplant.market.views.payment.PaymentView method), 26	<code>get_previous_by_start()</code> (eggplant.accounts.models.Account method), 16
<code>get_context_data()</code> (eggplant.profiles.views.Profile method), 33	<code>get_price_currency_display()</code> (eggplant.market.models.inventory.Product method), 23
<code>get_initial()</code> (eggplant.profiles.views.Profile method), 33	<code>get_queryset()</code> (eggplant.departments.views.DepartmentProfiles method), 20
<code>get_items_count()</code> (eggplant.market.models.cart.Basket method), 23	<code>get_queryset()</code> (eggplant.market.views.payment.PaymentView method), 26
<code>get_last_payment_status()</code> (eggplant.market.models.payment.Payment method), 25	<code>get_role_display()</code> (eggplant.roles.models.RoleAssignment method), 34
<code>get_next_by_changed()</code> (eggplant.profiles.models.UserProfile method), 31	<code>get_sex_display()</code> (eggplant.profiles.models.UserProfile method), 31
<code>get_next_by_created()</code> (eggplant.departments.models.DepartmentAdministrator method), 19	<code>get_status_display()</code> (eggplant.market.models.cart.Basket method), 23
<code>get_next_by_created()</code> (eggplant.invitations.models.DepartmentInvitation method), 21	<code>get_success_url()</code> (eggplant.profiles.views.NewUserPassword method), 32
<code>get_next_by_created()</code> (eggplant.invitations.models.InvitationBase method), 21	<code>get_template_names()</code> (eggplant.market.views.cart.BaseCartActionView method), 26
<code>get_next_by_created()</code> (eggplant.market.models.cart.Basket method), 23	<code>get_total_amount()</code> (eggplant.market.models.cart.Basket method), 23
<code>get_next_by_created()</code> (eggplant.market.models.payment.Payment method), 25	
<code>get_next_by_created()</code> (eggplant.profiles.models.UserProfile method), 31	
<code>get_next_by_start()</code> (eggplant.accounts.models.Account method), 16	
<code>get_object()</code> (eggplant.profiles.views.Profile method), 33	
<code>get_previous_by_changed()</code> (eggplant.profiles.models.UserProfile method), 31	
<code>get_previous_by_created()</code> (eggplant.departments.models.DepartmentAdministrator method), 19	
<code>get_previous_by_created()</code> (eggplant.invitations.models.DepartmentInvitation method), 21	
<code>get_previous_by_created()</code> (eggplant.invitations.models.InvitationBase method), 22	
<code>get_previous_by_created()</code> (eggplant.profiles.models.UserProfile method), 31	

H

`has_admin_permission()` (eggplant.profiles.models.UserProfile method), 32

`home()` (in module eggplant.dashboard.views), 18

I

`image` (eggplant.market.models.inventory.Product attribute), 23

`in_department()` (eggplant.profiles.models.UserProfile class method), 32

`inlines` (eggplant.accounts.admin.AccountAdmin attribute), 15

`inlines` (eggplant.departments.admin.DepartmentAdmin attribute), 18

`InvitationBackend` (class in eggplant.invitations.auth_backends), 20

`InvitationBase` (class in eggplant.invitations.models), 21

`InvitationBase.Meta` (class in eggplant.invitations.models), 21

- invite() (in module eggplant.invitations.views), 22
- invited_by (eggplant.invitations.models.DepartmentInvitation attribute), 21
- invited_by (eggplant.invitations.models.InvitationBase attribute), 22
- is_complete() (eggplant.profiles.models.UserProfile method), 32
- is_ready_for_payment() (eggplant.market.models.payment.Payment method), 25
- items (eggplant.market.models.cart.Basket attribute), 23
- ## L
- LinksGroupWidget (class in eggplant.market.filters), 27
- list_display (eggplant.market.admin.ProductAdmin attribute), 27
- list_display (eggplant.market.admin.ProductCategoryAdmin attribute), 27
- list_filter (eggplant.market.admin.ProductAdmin attribute), 27
- list_filter (eggplant.market.admin.ProductCategoryAdmin attribute), 27
- LoginRequiredMixin (class in eggplant.core.views), 17
- ## M
- MALE (eggplant.profiles.models.UserProfile attribute), 31
- market_home() (in module eggplant.market.views.inventory), 26
- media (eggplant.accounts.admin.AccountAdmin attribute), 15
- media (eggplant.accounts.admin.AccountCategoryAdmin attribute), 15
- media (eggplant.accounts.admin.AccountMembershipInline attribute), 15
- media (eggplant.core.widgets.MoneyWidget attribute), 18
- media (eggplant.departments.admin.DepartmentAdmin attribute), 18
- media (eggplant.departments.admin.DepartmentAdministratorInline attribute), 19
- media (eggplant.invitations.admin.DepartmentInvitationAdmin attribute), 20
- media (eggplant.invitations.forms.AcceptInvitationForm attribute), 21
- media (eggplant.invitations.forms.DepartmentInvitationForm attribute), 21
- media (eggplant.market.admin.ProductAdmin attribute), 27
- media (eggplant.market.admin.ProductCategoryAdmin attribute), 27
- media (eggplant.market.filters.LinksGroupWidget attribute), 27
- media (eggplant.market.forms.BasketItemForm attribute), 28
- media (eggplant.market.forms.ProductForm attribute), 28
- media (eggplant.profiles.admin.UserProfileAdmin attribute), 30
- media (eggplant.profiles.forms.NewUserSetPasswordForm attribute), 30
- media (eggplant.profiles.forms.ProfileForm attribute), 30
- media (eggplant.profiles.forms.SignupForm attribute), 31
- media (eggplant.roles.admin.RoleAssignment attribute), 33
- model (eggplant.accounts.admin.AccountMembershipInline attribute), 15
- model (eggplant.departments.admin.DepartmentAdministratorInline attribute), 19
- model (eggplant.departments.views.DepartmentProfiles attribute), 20
- model (eggplant.invitations.forms.DepartmentInvitationForm.Meta attribute), 21
- model (eggplant.market.filters.ProductFilter.Meta attribute), 27
- model (eggplant.market.forms.ProductForm.Meta attribute), 28
- model (eggplant.market.views.payment.PaymentView attribute), 26
- MoneyWidget (class in eggplant.core.widgets), 18
- ## N
- name_or_profile_names() (eggplant.accounts.models.Account method), 16
- new_payment_listener() (in module eggplant.market.models.listeners), 24
- new_payment_query_listener() (in module eggplant.market.models.listeners), 24
- NewUserForceProfileMiddleware (class in eggplant.profiles.middleware), 31
- NewUserPassword (class in eggplant.profiles.views), 32
- NewUserSetPasswordForm (class in eggplant.profiles.forms), 30
- ## O
- objects (eggplant.accounts.models.Account attribute), 16
- objects (eggplant.accounts.models.AccountCategory attribute), 16
- objects (eggplant.departments.models.Department attribute), 19
- objects (eggplant.departments.models.DepartmentAdministrator attribute), 19
- objects (eggplant.invitations.models.DepartmentInvitation attribute), 21
- objects (eggplant.market.models.cart.Basket attribute), 23
- objects (eggplant.market.models.cart.BasketItem attribute), 23

- objects (eggplant.market.models.inventory.Product attribute), 24
 - objects (eggplant.market.models.inventory.ProductCategoryPaymentView (class in eggplant.market.views.payment), attribute), 24
 - objects (eggplant.market.models.inventory.ProductTax attribute), 24
 - objects (eggplant.market.models.payment.Payment attribute), 25
 - objects (eggplant.permissions.models.Permission attribute), 29
 - objects (eggplant.permissions.models.UserProfilePermission attribute), 30
 - objects (eggplant.profiles.models.UserProfile attribute), 32
 - objects (eggplant.roles.models.RoleAssignment attribute), 34
 - OPEN (eggplant.market.models.cart.Basket attribute), 22
 - open_for_user() (eggplant.market.models.cart.BasketManager method), 23
 - option_string() (eggplant.market.filters.LinksGroupWidget method), 27
 - order_additional_validation_listener() (in module eggplant.market.models.listeners), 24
 - OTHER (eggplant.profiles.models.UserProfile attribute), 31
- P**
- PACKER (eggplant.roles.models.RoleAssignment attribute), 33
 - packer() (in module eggplant.roles.views), 34
 - paginate_by (eggplant.departments.views.DepartmentProfiles attribute), 20
 - partition() (in module eggplant.core.templatetags.partition_slice), 16
 - partition_horizontal() (in module eggplant.core.templatetags.partition_slice), 17
 - Payment (class in eggplant.market.models.payment), 25
 - Payment.DoesNotExist, 25
 - Payment.MultipleObjectsReturned, 25
 - payment_accepted() (in module eggplant.market.views.payment), 26
 - payment_detail() (in module eggplant.market.views.payment), 26
 - payment_info() (in module eggplant.market.views.payment), 26
 - payment_list() (in module eggplant.market.views.payment), 26
 - payment_rejected() (in module eggplant.market.views.payment), 27
 - payment_set (eggplant.accounts.models.Account attribute), 16
 - payment_status_changed_listener() (in module eggplant.market.models.listeners), 24
 - payments (eggplant.market.models.payment.Payment attribute), 25
 - PaymentView (class in eggplant.market.views.payment), 26
 - Permission (class in eggplant.permissions.models), 28
 - permission (eggplant.permissions.models.UserProfilePermission attribute), 30
 - Permission.DoesNotExist, 29
 - Permission.MultipleObjectsReturned, 29
 - permissions (eggplant.invitations.models.InvitationBase.Meta attribute), 21
 - permissions (eggplant.profiles.models.UserProfile attribute), 32
 - photo (eggplant.profiles.models.UserProfile attribute), 32
 - photo_url_or_default() (eggplant.profiles.models.UserProfile method), 32
 - post() (eggplant.profiles.views.NewUserPassword method), 32
 - price (eggplant.market.models.inventory.Product attribute), 24
 - process_request() (eggplant.profiles.middleware.NewUserForceProfileMiddleware method), 31
 - Product (class in eggplant.market.models.inventory), 23
 - product (eggplant.market.models.cart.BasketItem attribute), 23
 - Product.DoesNotExist, 23
 - Product.MultipleObjectsReturned, 23
 - product_set (eggplant.market.models.inventory.ProductCategory attribute), 24
 - product_set (eggplant.market.models.inventory.ProductTax attribute), 24
 - ProductAdmin (class in eggplant.market.admin), 27
 - ProductCategory (class in eggplant.market.models.inventory), 24
 - ProductCategory.DoesNotExist, 24
 - ProductCategory.MultipleObjectsReturned, 24
 - ProductCategoryAdmin (class in eggplant.market.admin), 27
 - ProductFilter (class in eggplant.market.filters), 27
 - ProductFilter.Meta (class in eggplant.market.filters), 27
 - ProductForm (class in eggplant.market.forms), 28
 - ProductForm.Meta (class in eggplant.market.forms), 28
 - ProductTax (class in eggplant.market.models.inventory), 24
 - ProductTax.DoesNotExist, 24
 - ProductTax.MultipleObjectsReturned, 24
 - Profile (class in eggplant.profiles.views), 32
 - profile (eggplant.departments.models.DepartmentAdministrator attribute), 19
 - ProfileForm (class in eggplant.profiles.forms), 30
 - PURCHASER (eggplant.roles.models.RoleAssignment attribute), 34

`purchaser()` (in module `eggplant.roles.views`), 34

R

`remove_from_cart()` (in module `eggplant.market.views.cart`), 26

`remove_from_items()` (`eggplant.market.models.cart.Basket` method), 23

`RemoveFromCart` (class in `eggplant.market.views.cart`), 26

`render()` (`eggplant.core.widgets.MoneyWidget` method), 18

`render()` (`eggplant.market.filters.LinksGroupWidget` method), 27

`render_option()` (`eggplant.market.filters.LinksGroupWidget` method), 27

`role()` (in module `eggplant.roles.views`), 34

`ROLE_CHOICES` (`eggplant.roles.models.RoleAssignment` attribute), 34

`RoleAssignment` (class in `eggplant.roles.admin`), 33

`RoleAssignment` (class in `eggplant.roles.models`), 33

`RoleAssignment.DoesNotExist`, 33

`RoleAssignment.MultipleObjectsReturned`, 33

S

`save()` (`eggplant.departments.models.Department` method), 19

`save()` (`eggplant.permissions.models.Permission` method), 29

`save()` (`eggplant.profiles.forms.NewUserSetPasswordForm` method), 30

`send_email_invitation()` (in module `eggplant.invitations.models`), 22

`SEX_CHOICES` (`eggplant.profiles.models.UserProfile` attribute), 31

`signup()` (in module `eggplant.profiles.views`), 33

`SignupForm` (class in `eggplant.profiles.forms`), 30

`site` (`eggplant.departments.models.Department` attribute), 19

`STATUES` (`eggplant.market.models.cart.Basket` attribute), 22

`success_url` (`eggplant.market.views.cart.BaseCartActionView` attribute), 26

`success_url` (`eggplant.profiles.views.NewUserPassword` attribute), 32

`success_url` (`eggplant.profiles.views.Profile` attribute), 33

T

`tax` (`eggplant.market.models.inventory.Product` attribute), 24

`template_name` (`eggplant.departments.views.DepartmentProfiles` attribute), 20

`template_name` (`eggplant.market.views.payment.PaymentView` attribute), 26

`template_name` (`eggplant.profiles.views.Profile` attribute), 33

`test_generate_upload_path()` (`eggplant.core.tests.UutilsTestCase` method), 17

`test_generate_upload_path_with_dir()` (`eggplant.core.tests.UutilsTestCase` method), 17

`test_home()` (`eggplant.dashboard.tests.TestDashboard` method), 18

`TestDashboard` (class in `eggplant.dashboard.tests`), 18

U

`user` (`eggplant.market.models.cart.Basket` attribute), 23

`user` (`eggplant.profiles.models.UserProfile` attribute), 32

`user` (`eggplant.roles.models.RoleAssignment` attribute), 34

`user_data_query_listener()` (in module `eggplant.market.models.listeners`), 25

`user_profile` (`eggplant.permissions.models.UserProfilePermission` attribute), 30

`user_profiles` (`eggplant.accounts.models.Account` attribute), 16

`UserProfile` (class in `eggplant.profiles.models`), 31

`UserProfile.DoesNotExist`, 31

`UserProfile.MultipleObjectsReturned`, 31

`userprofile_set` (`eggplant.permissions.models.Permission` attribute), 29

`UserProfileAdmin` (class in `eggplant.profiles.admin`), 30

`UserProfilePermission` (class in `eggplant.permissions.models`), 29

`UserProfilePermission.DoesNotExist`, 30

`UserProfilePermission.MultipleObjectsReturned`, 30

`userprofilepermission_set` (`eggplant.accounts.models.Account` attribute), 16

`userprofilepermission_set` (`eggplant.departments.models.Department` attribute), 19

`userprofilepermission_set` (`eggplant.permissions.models.Permission` attribute), 29

`userprofilepermission_set` (`eggplant.profiles.models.UserProfile` attribute), 32

`UtilsTestCase` (class in `eggplant.core.tests`), 17

W

`widgets` (`eggplant.market.forms.ProductForm.Meta` attribute), 28