**Описание для increment\_dish(request)**

Сценарий описываемой последовательности сообщений между объектами программной системы вызывается при вызове метода increment\_dish файла «views.py» приложения «orders». Вызов данной функции осуществляется по запросу ajax со страницы просмотра корзины заказов, отсылаемому серверу по событию нажатия на кнопку увеличения количества порций блюда.

Сценарий осуществляется только в том случае, если переданный запрос является запросом ajax. Это проверяется с помощью функции is\_ajax() переданного в функцию объекта request. После этого происходит получение значений ключей, переданный в GET запросе с помощью функции get() объекта request.GET типа QueryDict. Также из объекта сессии (session: SessionBase) извлекается значение ключа – полученного из объекта request идентификатора блюда (dish\_id) – которое является количеством порций данного блюда в заказе. С помощью объекта модели класса Dish, соответствующего полученному идентификатору, извлекается цена блюда. Если количество блюд в корзине стало равным нулю, то данный ключ удаляется из объекта сессии с помощью метода pop(dish\_id, None). В противном случае значение ключа, соответствующее количеству порций блюда, уменьшается на величину переданного в запросе значения количества с помощью метода \_\_setitem\_\_. Затем из объекта сессии извлекается цена корзины заказов до нажатия на кнопку уменьшения порций блюд с помощью метода get(‘cart\_price’). Если уменьшение цены корзины на величину цены выбранного блюда приводит к тому, что цена корзины становится равной нулю, то значение, содержащее в себе цену корзины, в объекте сессии удаляется с помощью метода pop(‘cart\_price’, None). В противном случае значение данного ключа уменьшается на величину цены блюда с помощью метода \_\_setitem\_\_ объекта сессии, устанавливающего новое значение для указанного ключа.

**Описание для EmployeePage.** **get\_context\_data(kwargs)**

Сценарий описываемой последовательности сообщений между объектами программной системы вызывается при вызове метода get\_context\_data файла «views.py» приложения «employees». Вызов данной функции осуществляется при переходе приложения по следующему адресу: url(r'(?P<employees\_id>\d+)/$', EmployeePage.as\_view(), name='employees'). При вызове метода get\_context\_data(self, \*\*kwargs) создается объект context, которому передается результат вызова super().get\_context\_data(\*\*kwargs). После этого извлекается идентификатор сотрудника employees\_id с помощью функции self.kwargs.get('employees\_id'). С помощью менеджера по умолчанию объектов Employee извлекается сотрудник заведения, соответствующий полученному идентификатору из строки запроса. В конце с помощью менеджера по умолчанию объектов Order по идентификатору сотрудника создается объект order\_list.

**Описание для auth\_view(request)**

Сценарий описываемой последовательности сообщений между объектами программной системы вызывается при вызове метода auth\_view файла «views.py» приложения «employees». Вызов данной функции осуществляется при переходе приложения по ссылке: url(r'^accounts/after\_login/$', views.auth\_view, name='after\_login'), после нажатия кнопки «Войти» на странице authentication.html. После этого происходит получение значений ключей, переданных в POST запросе с помощью функции \_\_getitem\_\_() объекта request.POST типа QueryDict. Далее объекту user присваивается результат аутентификации по связке «логин-пароль» при помощи auth.authenticate(username=username, password=password). После этого объект user проверяется по двум параметрам: во-первых – есть ли такой пользователь (если нет, то происходит переход по ссылке url(r'^accounts/login/(?P<login\_state>\d+)?', views.login\_view, name='login\_employee') на страницу 'employees/authentication.html' и выводится предупреждение о неверном вводе имени или пароля. Если первая проверка прошла успешно, то пользователь проверяется на предмет наличия блокировки (user.is\_active). Если пользователь не заблокирован, происходит его авторизация auth.login(request, user) и переход по ссылке url(r'(?P<employees\_id>\d+)/$', EmployeePage.as\_view(), name='employees') на страницу employees.html. Если пользователь не прошел проверку происходит переход по ссылке url(r'^accounts/ban/$', views.ban\_view, name='ban\_employee') на страницу ban.html.

**Описание для DishesList.get\_context\_data(kwargs)**

Сценарий описываемой последовательности сообщений между объектами программной системы вызывается при вызове метода get\_context\_data файла «views.py» приложения «dishes». Вызов данной функции осуществляется при переходе приложения по ссылке: url(r'(?P<establishment\_id>\d+)/$', DishesList.as\_view(), name='dishes\_without\_category')или url(r'(?P<establishment\_id>\d+)/(?P<dish\_category>\d+)?', DishesList.as\_view(), name='dishes\_with\_category'). При вызове метода get\_context\_data(self, \*\*kwargs) создается объект context, которому передается результат вызова super().get\_context\_data(\*\*kwargs). После этого извлекается идентификатор dish\_id с помощью функции self.kwargs.get(‘dish\_id’). Далее проверяется: не пуст ли ‘establishment\_id’, если нет, то создается объект типа dish\_objects\_filter типа QueryDict. Потом dish\_objects\_filter проверяется на наличие в нем объекта EstablishmentDish, и, если он есть, он присваивается объекту current\_establishment. В других случаях, первый establishment, получаемый через фильтр EstablishmnetDish, присваивается current\_establishment. После получения current\_establishment происходит получение dish\_category с помощью self.kwargs.get(‘dish\_category’). Далее идет получение choises Dish.TYPE и представление списков Dish с условиями establishment\_id, dish\_category.