**Приложение В**

**Листинги**

**Листинг В.1** – **Регистрация**

class RegisterNewUser(CreateView):

form\_class = RegisterUser

template\_name = 'register.html'

extra\_context = {'title': 'Регистрация'}

def form\_valid(self, form):

user = form.save()

login(self.request, user)

return redirect('show\_courses')

**Листинг В.2** – **Просмотр «Мой профиль»**

class MyProfile(DetailView):

model = CustomUser

template\_name = 'my\_profile.html'

context\_object\_name = 'profile'

extra\_context = {'title': 'Мой профиль'}

def get\_object(self, queryset=None):

return self.request.user

def get\_context\_data(self, \*, object\_list=None, \*\*kwargs):

context = super().get\_context\_data(\*\*kwargs)

context['cnt\_sbs'] = len(CustomUser.objects.get(id=self.request.user.id).subscriptions.all())

return context

**Листинг В.3** – **Поиск пользователя**

class SearchUsers(ListView):

model = CustomUser

template\_name = 'search\_users.html'

context\_object\_name = 'users'

extra\_context = {'title': 'Поиск пользователей',

'subtitle': 'По Вашему запросу найдено:'}

def get\_queryset(self):

return CustomUser.objects.filter(username\_\_icontains=

self.request.GET.get('search', ''))

**Листинг В.4** – **Просмотр и сдача заданий в учебной группе**

class ShowPostsGroups(ListView):

model = PostsInStudyGroup

template\_name = 'study\_group.html'

extra\_context = {'title': 'Учебная группа', 'subtitle': 'Группа'}

def get\_context\_data(self, \*, object\_list=None, \*\*kwargs):

context = super().get\_context\_data(\*\*kwargs)

context['posts'] = PostsInStudyGroup.objects.filter(study\_group\_id=self.kwargs['pk'])

context['study\_group'] = StudyGroup.objects.get(id=self.kwargs['pk'])

context['is\_followed'] = False

if StudyGroup.objects.get(id=self.kwargs['pk']) in CustomUser.objects.get(id=self.request.user.id).user\_groups.all():

context['is\_followed'] = True

else:

context['is\_followed'] = False

return context

**Листинг В.5** – **Просмотр и выставление оценок**

class ShowValuations(CreateView):

form\_class = UpdateValuations

template\_name = 'compl\_works.html'

extra\_context = {'title': 'Оценки', 'subtitle': 'Работы и оценки учащихся'}

def form\_valid(self, form):

fs = form.save(commit=False)

fs.post\_id = self.kwargs['post\_id']

fs.user\_id = self.request.user.id

fs.save()

return HttpResponseRedirect(reverse('show\_valuations', args=[self.kwargs['pk'], self.kwargs['post\_id']]))

def get\_context\_data(self, \*, object\_list=None, \*\*kwargs):

context = super().get\_context\_data(\*\*kwargs)

context['compl\_works'] = CompletedTaskInStudyGroup.objects.filter(post\_id=PostsInStudyGroup.objects.get(pk=self.kwargs['post\_id']))

return context

**Листинг В.6** – **Просмотр и комментирование на странице «Форум»**

class ShowCommunication(CreateView):

model = ForumCommunication

template\_name = 'forum.html'

extra\_context = {'title': 'Форум:Обсуждение'}

form\_class = CommForm

def form\_valid(self, form):

fs = form.save(commit=False)

fs.user = self.request.user

fs.subsection\_id = self.kwargs['pk']

fs.save()

return HttpResponseRedirect(reverse('show\_communication', args=[self.kwargs['pk']]))

def get\_context\_data(self, \*, object\_list=None, \*\*kwargs):

context = super().get\_context\_data(\*\*kwargs)

context['subsection'] = ForumSubsection.objects.get(id=self.kwargs['pk'])

context['comments'] = ForumCommunication.objects.filter(subsection\_id=self.kwargs['pk']).order\_by('-id')

return context

**Листинг В.7** – **Показ курсов**

class ShowCourses(ListView):

model = Courses

template\_name = 'courses.html'

context\_object\_name = 'courses'

extra\_context = {'title': 'Курсы', 'subtitle': 'Все курсы'}

def get\_context\_data(self, \*, object\_list=None, \*\*kwargs):

context = super().get\_context\_data(\*\*kwargs)

context['courses'] = Courses.objects.all()

context['categories'] = Category.objects.all()

context['sort'] = self.request.GET.get('sort')

return context

**Листинг В.8** – **Создание курсов**

class CreateCourse(CreateView):

form\_class = CreateOrUpdateCourseForm

template\_name = 'create\_course.html'

success\_url = reverse\_lazy('teaching')

extra\_context = {'title': 'Создание курса'}

def form\_valid(self, form):

fs = form.save(commit=False)

fs.author = self.request.user

fs.save()

return redirect('show\_courses')

**Листинг В.9** – **Отображение оплаты**

class CreateCheckoutSessionView(View):

def get(self, request, \*args, \*\*kwargs):

stripe.api\_key = settings.STRIPE\_SECRET\_KEY

course = Courses.objects.get(pk=self.kwargs['pk'])

checkout\_session = stripe.checkout.Session.create(

payment\_method\_types=['card'],

line\_items=[{

'price\_data': {

'currency': 'byn',

'unit\_amount': course.price\*100,

'product\_data': {

'name': course.title,

'description': f"{course.about\_the\_course} Автор курса: {str(course.author)} ({str(course.author.last\_name)} {str(course.author.first\_name)} {str(course.author.patronymic)})",

},

},

'quantity': 1,

},

],

phone\_number\_collection={

'enabled': True,

},

metadata={

"course\_id": course.id

},

mode='payment',

success\_url=f'http://127.0.0.1:8000/course/payment/success/',

cancel\_url=f'http://127.0.0.1:8000/course/payment/cancel/',

)

return redirect(checkout\_session.url, code=303)