Game Over Nick Paradeisanos

ITC4214

Purpose:

- The main purpose of the website is for new game developers to be able to submit and advertise their games. The main audience of the website is for game enthusiasts who wish to test games in their beta state. This website is not directed towards games that are in their final state of development but mainly to games that need beta testing for development purposes.

Features:

The main feature of our website reflects on the above purpose. We have a user system where everyone is able to come and upload their games. That includes the name, release date as well as a short description of the game it self. Additionally, we have a search system where every user can search a game they have heard about by its name or generally browse games by searching lets say their desired genre.

Upcoming Features:

- In the upcoming time we plan to develop the home page more so that the interactive media like the carousel reflect on the latest or most hot releases.

Technologies:

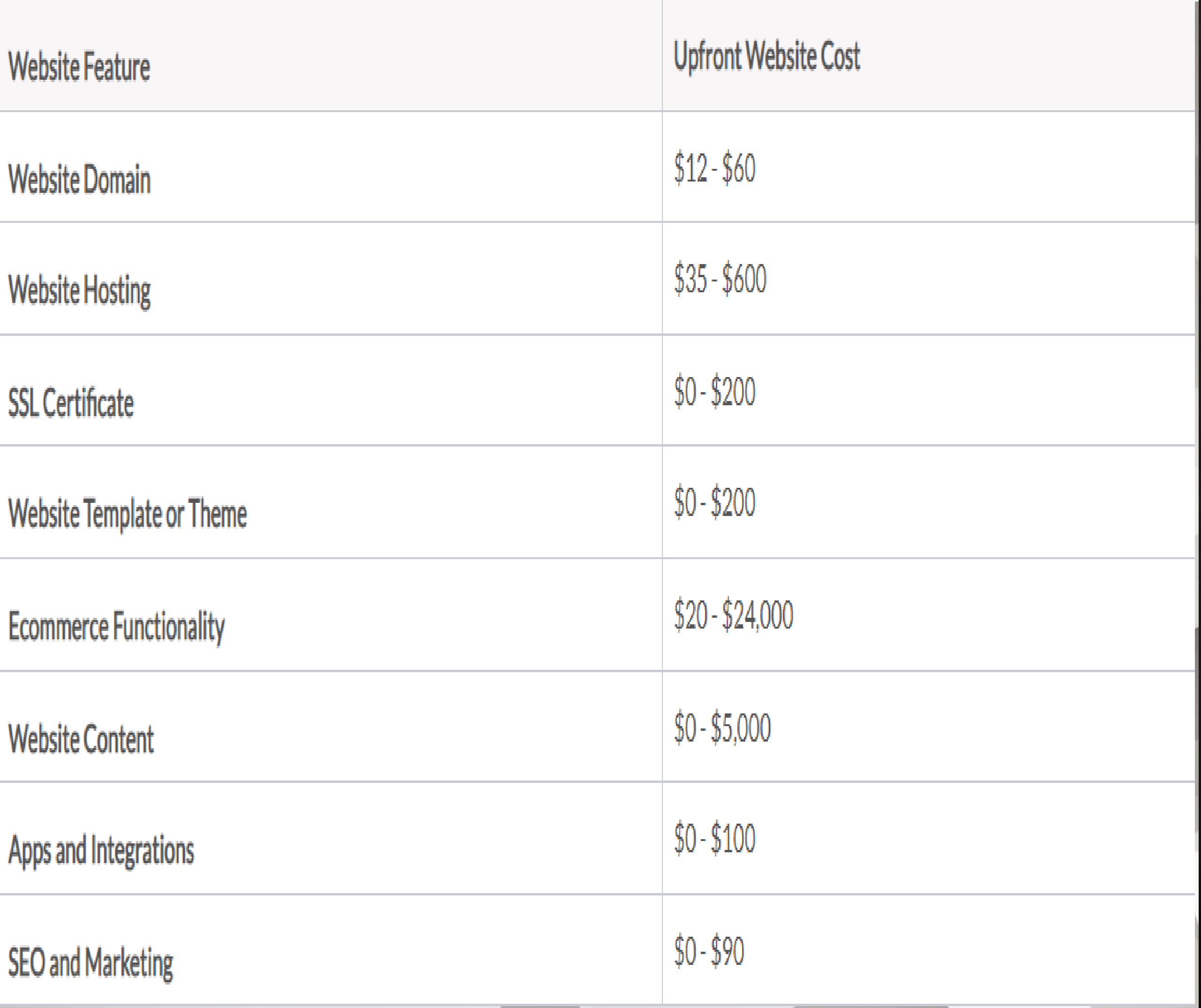
- Technologies used were HTML, Javascript, CSS and Python. In advance, we also used frameworks like Bootstrap and Django.

Search Engines:

- We plan to develop our website to fully to work with the most known search engines like Google, Bing and Yahoo. You can also find our bootcamp campaign in Twitter, Facebook and Instagram.

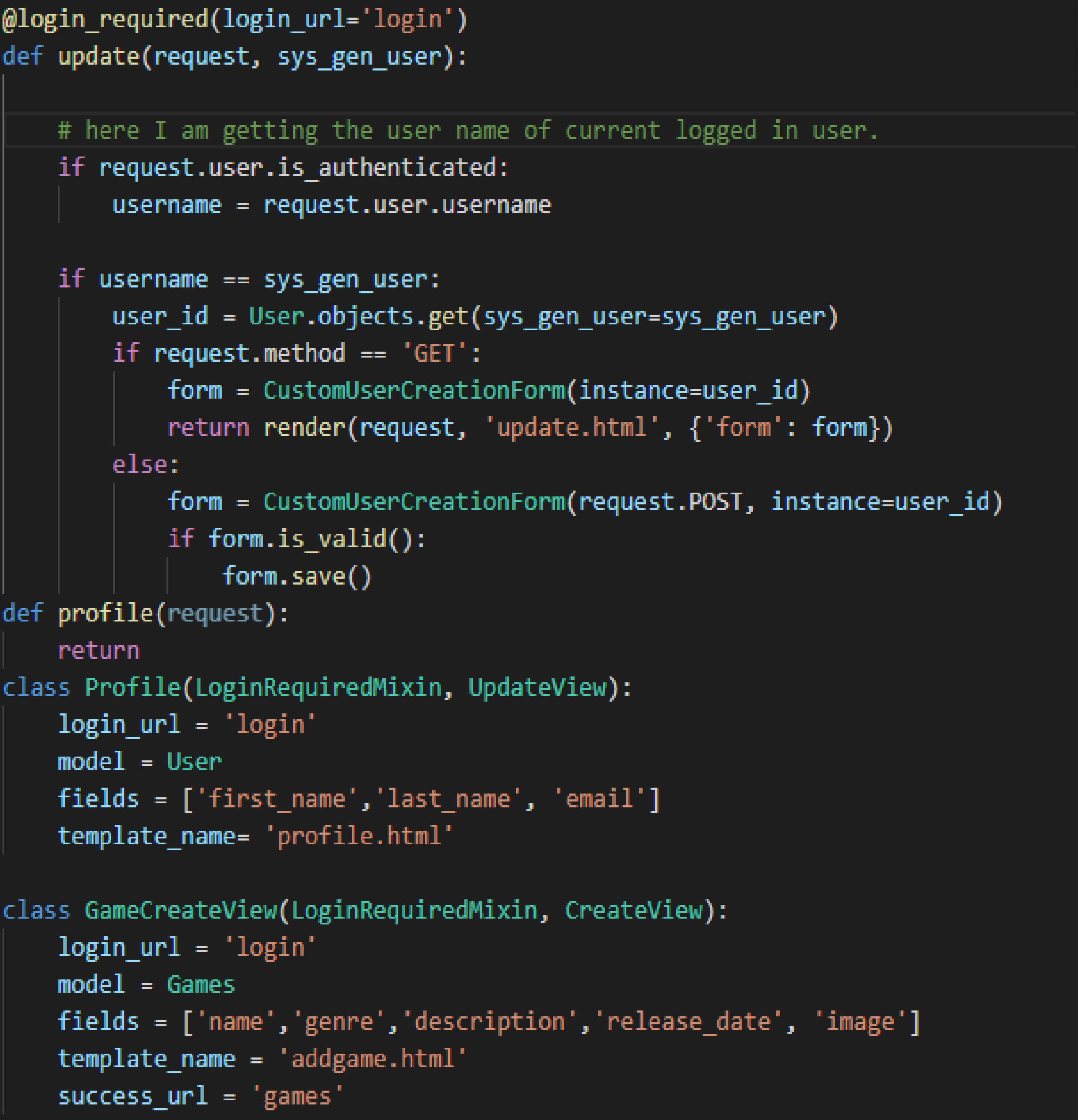
Cost Projection:

- Cost projection varies vastly. An example can be seen below but our costs are still shifting everyday while we upgrade our website and the utilities of it.

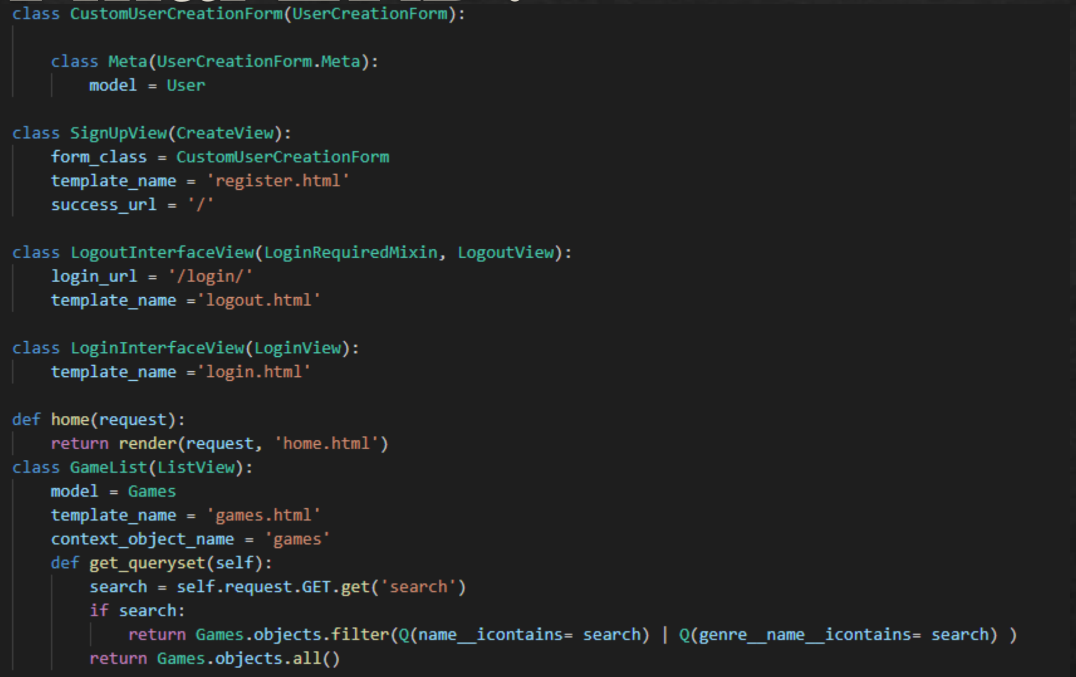


Website Feature	Upfront Website Cost
Website Domain	$12 - $60
Website Hosting	$35 - $600
SSL Certificate	$0 - $200
Website Template or Theme	$0 - $200
Ecommerce Functionality	$20 - $24,000
Website Content	$0 - $5,000
Apps and Integrations	$0 - $100
SEO and Marketing	$0 - $90

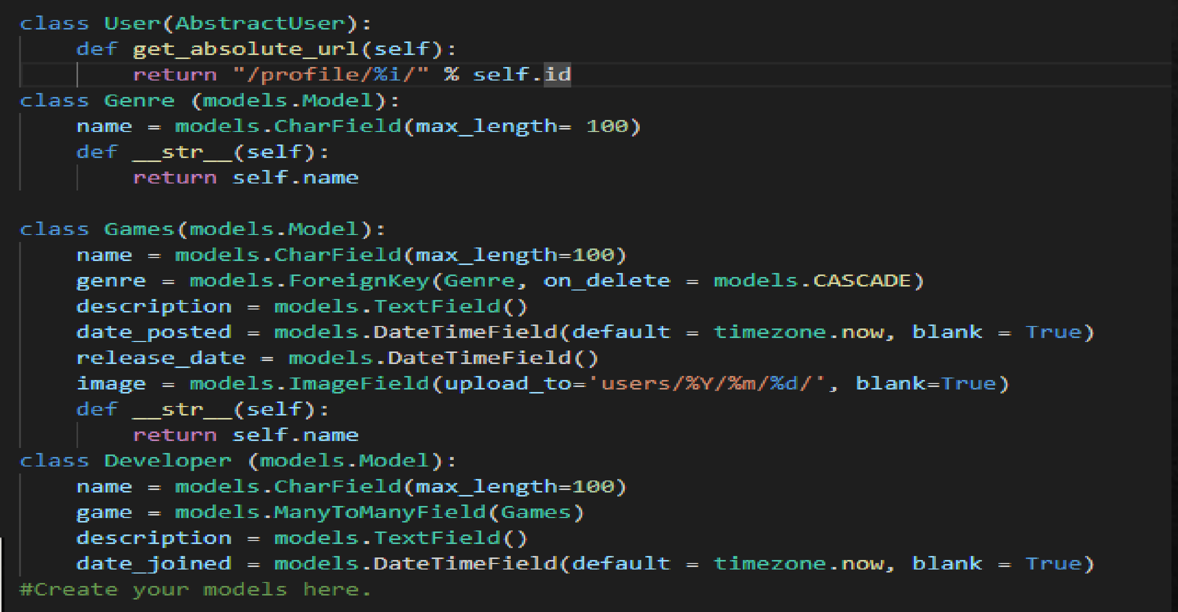
Final ERD:



```python
@login_required(login_url='login')
def update(request, sys_gen_user):

    # here I am getting the user name of current logged in user.
    if request.user.is_authenticated:
        username = request.user.username

    if username == sys_gen_user:
        user_id = User.objects.get(sys_gen_user=sys_gen_user)
        if request.method == 'GET':
            form = CustomUserCreationForm(instance=user_id)
            return render(request, 'update.html', {'form': form})
        else:
            form = CustomUserCreationForm(request.POST, instance=user_id)
            if form.is_valid():
                form.save()
def profile(request):
    return
class Profile(LoginRequiredMixin, UpdateView):
    login_url = 'login'
    model = User
    fields = ['first_name','last_name', 'email']
    template_name= 'profile.html'

class GameCreateView(LoginRequiredMixin, CreateView):
    login_url = 'login'
    model = Games
    fields = ['name','genre','description','release_date', 'image']
    template_name = 'addgame.html'
    success_url = 'games'
```



```python
class CustomUserCreationForm(UserCreationForm):

    class Meta(UserCreationForm.Meta):
        model = User

class SignUpView(CreateView):
    form_class = CustomUserCreationForm
    template_name = 'register.html'
    success_url = '/'

class LogoutInterfaceView(LoginRequiredMixin, LogoutView):
    login_url = '/login/'
    template_name ='logout.html'

class LoginInterfaceView(LoginView):
    template_name ='login.html'

def home(request):
    return render(request, 'home.html')
class GameList(ListView):
    model = Games
    template_name = 'games.html'
    context_object_name = 'games'
    def get_queryset(self):
        search = self.request.GET.get('search')
        if search:
            return Games.objects.filter(Q(name__icontains= search) | Q(genre__name__icontains= search) )
        return Games.objects.all()
```



```python
class User(AbstractUser):
    def get_absolute_url(self):
        return "/profile/%i/" % self.id
class Genre (models.Model):
    name = models.CharField(max_length= 100)
    def __str__(self):
        return self.name

class Games(models.Model):
    name = models.CharField(max_length=100)
    genre = models.ForeignKey(Genre, on_delete = models.CASCADE)
    description = models.TextField()
    date_posted = models.DateTimeField(default = timezone.now, blank = True)
    release_date = models.DateTimeField()
    image = models.ImageField(upload_to='users/%Y/%m/%d/', blank=True)
    def __str__(self):
        return self.name
class Developer (models.Model):
    name = models.CharField(max_length=100)
    game = models.ManyToManyField(Games)
    description = models.TextField()
    date_joined = models.DateTimeField(default = timezone.now, blank = True)
#Create your models here.
```

Access

- Admin access.
  User: admin
- Pass: admin
- Heroku link: https://game-over-cw.herokuapp.com/