So, You Want to Make a Production Django App

A Little Background

- Launching Production Django Apps since '06 and 0.9something beta
- Launched and Managed Django Websites, E-Commerce Sites, and Web Apps for brands such as:
- Rip Curl, UT San Diego, Callaway Golf, DC Shoes, Dunlop, PacSun, and many many more

Goals

- Identify Common Pitfalls in Launching
- Point Out Django Gotchas
- Point Out Django Tools
- Help You Have a Smooth Launch
- Give Some Pointers on Managing and Maintaing a Django App

This Only Partially About Django

and all about launching web apps

Going Production is…

- Putting it in front of consumers
- Promoting It
- Customer data
- 24/7/365
- Someone Else Is Using It
- Your billing relationship changes

But First, Some Quick Tips

Simple Django Things

- DEBUG = False
- Style your 500 / 404.html
- HTTPS Admin, login, and .. anywhere sensitive information is submitted
- Make your Sites record accurate

Simple Non-Django Things

- Favicon
- Run a Site Crawler
- Redirect to one domain - www
- Check your browser console for errors

So What Makes Your App Ready?

The Big Questions

- Is it Stable?
- Did I get it Right?
- Does it Perform?

Stability

- Stability is about predictable performance
- Thorough testing
- A way to track bugs and issues
- A way to monitor what's happening

Bug Tracking

- 500 Emails - Send these to a Gmail Account
- Sentry is cool
- Rotate your logs
- Stomp 500s, signal to noise is very important

Refactoring

- You probably didn't get all the data models and functionality right
- Following launch focus on fixing issues, not solving any misconceptions.

Ok, App Works Good, Until the Newsletter Went Out

Performance part 1

- Memcached + ANONYMOUS_CACHE
- Django Debug Toolbar
- {% cache %}
- But, don't cache session specific stuff!
- Use Production Scale Data

Performance part 2

- We use Nginx + Gunicorn
- CDNs - use them, but use caution
- Expire Tags - speed without effort
- Premature Optimization is the root of all evil, but the night before launch is no longer premature, so save it in a temporary model on a signal damnit

Performance 2.1

- Our Setup
- Static Media from CDN
- Nginx or ELB or other load balancer
- Gunicorn managed by Supervisord
- MySQL or Postgresql
- Memcached, Elasticsearch, RabbitMQ

Performance part 3

- New Relic can tell you a lot
- Process long tasks with Celery
- Varnish can do wonders, if your app is doing the right thing
- But first optimize your queries and indexes
- Yes, you can do group bys and aggregates.. annotate and aggregate are not so intuitive
- DB Templates Are Expensive

Aggregate Digression

```
RateClass.objects.annotate(

      hours = Sum("person__timeentry__hours")).filter(

      person__timeentry__project__name="My Project')

)
```

But Enough About Performance

- Performance is usually something you see as you scale up and add complexity
- Your site should perform quickly, but even more importantly..

Things That Really Matter

Content!

- Is all the content the right content?
- No Lorem Ipsum
- Do all your forms submit? Are there human error messages.
- Is the contact information correct?

Browser Compatibility

- Did you check all your browsers?
- As a rule, CEOs use the oldest possible IE version
- There are over 100 million iPads, what does your site look like on one? CEOs usually have one of these too.

Security

- SSL logins! admins! everything!
- CSRF protection
- Are you assuming anything?

Search Matters

- robots.txt
- canonical tags
- meta tags
- analytics tags
- duplicate content

What Used to Be Searched Also Matters

- Pages with rankings are valuable
- If it was a link on Google, don't make it a 404

Things You Should Have Already Done

Training

- Not everyone groks abstractions, explain how to do things
- Make sure your client knows that access to Users == Super User
- Take a moment to polish your Change List and Change Form Templates

Dev Site

- Have one
- Make sure it doesn't send email to anyone
- Make sure it says "DEV SITE" somewhere on the front end
- Make sure it has robots.txt disallowing all
- If possible, control access
- Keep track of any integration credentials for sandbox vs production

Configuration Management

- Settings Can Get Unruly
- We use cascading imports to structure them
- Beware Stored Passwords!

Testing

- Unit Tests are nice, if you have been doing them all along
- Test it by hand
- Test it like a consumer
- - Bad / Missing Data
- - International Data / Unicode

Migrations

- Use South, or don't but be prepared to tear your hair out either way
- Is there existing data to import? Do a dry run - build this into management commands

Ok We Are Ready!

Launch is in 1 week

- Lower your DNS TTL
- Make sure you have access to all the domain information
- Are all integrations ready for production? With production credentials?
- A Dry Run is worth the effort
- Do you know all the DNS records you are changing? Are the nameservers in your control?
- Test your site with /etc/hosts, make sure it is ready to go
- Write up a plan, and let everyone know the timing

Launch!

- Sometimes, it is better to just get it done.
- But not always
- But sometimes, really, it just has to fly
- Walk through everything
- Run transactions on your credit card, and then cancel them
- Doubt everything

Deployment

- Revision your deploys with Git
- Automate it as much as possible
- But don't put off automating most of it
- Read the 11-Factor App

DNS Gotchas

- Nameservers! Don't change unless you know all the records that are there
- SPF, PTR - make sure your email makes it through the spam filter
- IT departments don't get CNAMEs, so watch your load balancers!

Woo Hoo! It's Up! Now What

Are We Done?

Post Launch

- Keep track of what you agreed to build.
- Good fences make good neighbors
- When You Launch, You are in Maintenance Mode. Make the Mental Shift!

Refactors, Again

- Actually that did need to be a separate model
- See, South was a good idea
- Ouch you should have used South
- Either way, you need to write a manage command

Maintenance Tips and Tricks

- Manage commands are really useful
- So is the shell
- Know what you plan to do on a server before logging in

The Long View

- Don't focus on the bad things, focus on what was achieved
- What simple things can you do to make the next one smoother
- Checklists!

Questions?