Full-stack Python Web Applications
Kate Heddeleston
@heddle317
Python is often one part of a much larger system.
What we’re going to talk about...

1. A (semi) typical web application
2. Dev, staging, production, deploy system
3. Helpful python libraries
What we’re not going to talk about...

- Massively scalable web applications.
- Every single installation on the machine.
- Nitty gritty details of anything.
Web Application

Diagram:
- User (Interwebs) interacts with
- Load Balancer (LB)
- Frontend (FE)
- Middleware (MC)
- Application Server (AS)
- Database (DB)
Parts of a Stack

1. Operating System
2. Web Server
3. Database
4. Application Language
Web Server
Apache mod_wsgi

Application Code
Python

Application Code
Python
Postgres Database

Data
Web Server

Nginx/
Gunicorn

Apache/
mod_wsgi
Application Code

Frameworks
Python libraries
Frontend
Static Files (maybe)
Database

- PostgreSQL
- MySQL
- SQLServer
- Redis
- MongoDB
Logging

Log files on the machine
Version Control

- git
- svn

Github

Bitbucket
SMTP Server

PostmarkApp
Sendgrid
Amazon SES
Async Frameworks

- Twisted
- Tornado

Async Tasks

- Python-rq/Redis
- Celery/RabbitMQ
Monitoring

New Relic
Nagios
Pingdom
Web Server

Logging

Application Code

Cache

Async Tasks

Data Store

Database

!Exception Handling!
Exception Handling!

Emails
Sentry
Logging

Log Server
Loggly
Splunk
Version Control

SMTP Server

Monitoring

Logging

!Exception Handling!
Load Balancer

Web Server
Application Code
Cache

Database
Async Tasks
Data Store
Async Tasks
Data Store
System of Systems
Production
(aka everything you just built)

Deploy
Dev

Venv & Venvwrapper
Vagrant & VirtualBox
Deploy

git pull
&
restart
services
Production
(aka everything you just built)

Testing

Dev

Deploy
Testing

Jenkins
CircleCI
TravisCI
Server Config

Chef
Puppet
Ansible
SaltStack
Docker
Deploy

Chef
Puppet
Ansible
SaltStack
Docker
Dev

Venv & Venvwrapper
Vagrant & VirtualBox
Chef Repo for a Web Application

https://github.com/heddle317/django-chef-application
PaaS

Heroku
Elastic Beanstalk
Resources Maps

https://github.com/heddle317/full-stack-resources/tree/master/resources_maps
Talk Maps

https://github.com/heddle317/full-stack-resources/tree/master/talk_maps
Web Server
- Turn your computer into a Server (4/11 - 12:10pm)
- Hello Physical World: A Crash Course on the internet of things (4/11 - 5:10pm)

Application

Cache

Database
- Designing Django's Migrations (4/12 - 11:30am)
- Introduction to SQLAlchemy Core (4/12 - 1:55pm)
- Sane Schema migrations with Alembic and SQLAlchemy (4/12 - 2:35pm)
- Building the App (4/12 - 3:15pm)
- Postgres Performance for Humans (4/13 - 1:10pm)
- PostgreSQL is Web Scale (4/13 - 1:50pm)

Logging

Exception Handling

Asynchronous Tasks
- An Introduction to Twisted (4/11 - 1:55pm)
- Twisted Mixing (4/11 - 2:35pm)
- What is Async, how does it work, and when should I use it? (4/11 - 3:15pm)
- Distributed Computing is Hard, Let’s go shopping (4/11 - 4:30pm)
- Fan-in and Fan-out: The crucial components of concurrency (4/11 - 5:10pm)
- Which messaging layer should you use if you want to build a loosely coupled distributed python app? (4/12 - 5:10pm)

SMTP Server

Monitoring
- Realtime predictive analytics using scikit-learn & RabbitMQ (4/11 - 3:15pm)
- Pushing Python: Building a High throughput low latency system (4/13 - 3:15pm)
Security
- The sorry state of SSL (4/12 - 1:55pm)
- Quick wins for better website security (4/12 - 2:35pm)
- Multi-factor Authentication - Possession Factors (4/12 - 3:15pm)

Production

Testing
- Getting started Testing (4/12 - 1:55pm)
- Unit Testing Makes your code better (4/12 - 2:35pm)
- Advanced techniques for web functional testing (4/12 - 4:30pm)
- Performance Testing and Profiling: A vituous cycle (4/12 - 5:10pm)
- Deliver your software in an envelope (4/13 - 1:10pm)
- Smart Dumpster: Employing Python to report real-time resource fill to operation managers (4/13 - 2:30pm)

Staging

Server Configuration and Deploy
- Ansible - Python-Powered Radically simple IT Automation (4/11 - 1:55pm)
- Puppet Modules: Apps for Ops (4/11 - 2:35pm)
- Getting Started with SaltStack (4/11 - 3:15pm)
- Application Deployment State of the Onion (4/11 - 4:30pm)
- Introduction to Docker (4/12 - 10:50am)

Dev
Education
- The Young Coder: Let’s learn python (or, ‘So you want to run a young coders class) (4/11 - 1:55pm)
- The Python Pipeline: Why you should reach out to local teachers and how to do it (4/11 - 2:35pm)
- Teaching Python: To Infinity and Beyond (4/11 - 3:15pm)
- Technical onboarding, training, and mentoring (4/13 - 1:10pm)
- Software Carpentry: Lessons Learned (4/12 - 4:30pm)
- Outreach Program for Women: Lessons in Collaboration (4/13 - 1:50pm)
- Software Engineering Research for Hackers: Bridging the two solitudes (4/13 - 2:30pm)

Fun
- Castle Anthrax: Dungeon Generation Techniques (4/11 - 4:30pm)
- Blending art, technology, and light, Python for interactive and real time LED installations (4/11 - 4:30pm)
- Hello Physical World: A crash course on the internet of things (4/11 - 5:10pm)
- Discovering Python (4/12 - 3:15pm)
- Cheap Helicopters in my living room (4/12 - 4:30pm)
- Programming an autonomous 20 foot blimp with python (4/12 - 5:10pm)

Gaming
- My big gay adventure. Making, releasing and selling an indie game made in python. (4/13 - 1:10pm)
- 2D/3D graphics with python on mobile platforms (4/13 - 1:20pm)

Open Source
- Free Software, Free People (4/11 - 5:10pm)
- Hitchhiker’s Guide to participating in open source (4/13 - 1:50pm)
- Set your code free: releasing and maintaining an open-source python project (4/13 - 2:30pm)

Personal
- It’s dangerous to go alone: battling the invisible monsters in tech (4/12 - 5:10pm)
- Farewell and welcome home: python in two genders (4/13 - 1:10pm)
3 Takeaways

- What are the basic pieces of a full-stack.
- What do these pieces look like in different environments.
- Resources for learning more and working with these pieces.
Ask your questions now.

Kate Hedleston
@heddle317
https://github.com/heddle317/full-stack-resources
“A complex system that works is invariably found to have evolved from a simple system that works.”

— John Gall, Systemantics (1975)
“A system is never finished being developed until it ceases to be used.”

— attributed to Gerald M. Weinberg
“It is as if perfection be attained not when there is nothing more to add, but when there is nothing more to take away.”

— Antoine de Saint-Exupéry, *Terre des Hommes* (1939)
“There is no such thing as a small change to a large system.”
— systems folklore, source lost in the mists of time
“Everything should be made as simple as possible, but no simpler.”

— commonly attributed to Albert Einstein; it is actually a paraphrase of a comment he made in a 1933 lecture at Oxford