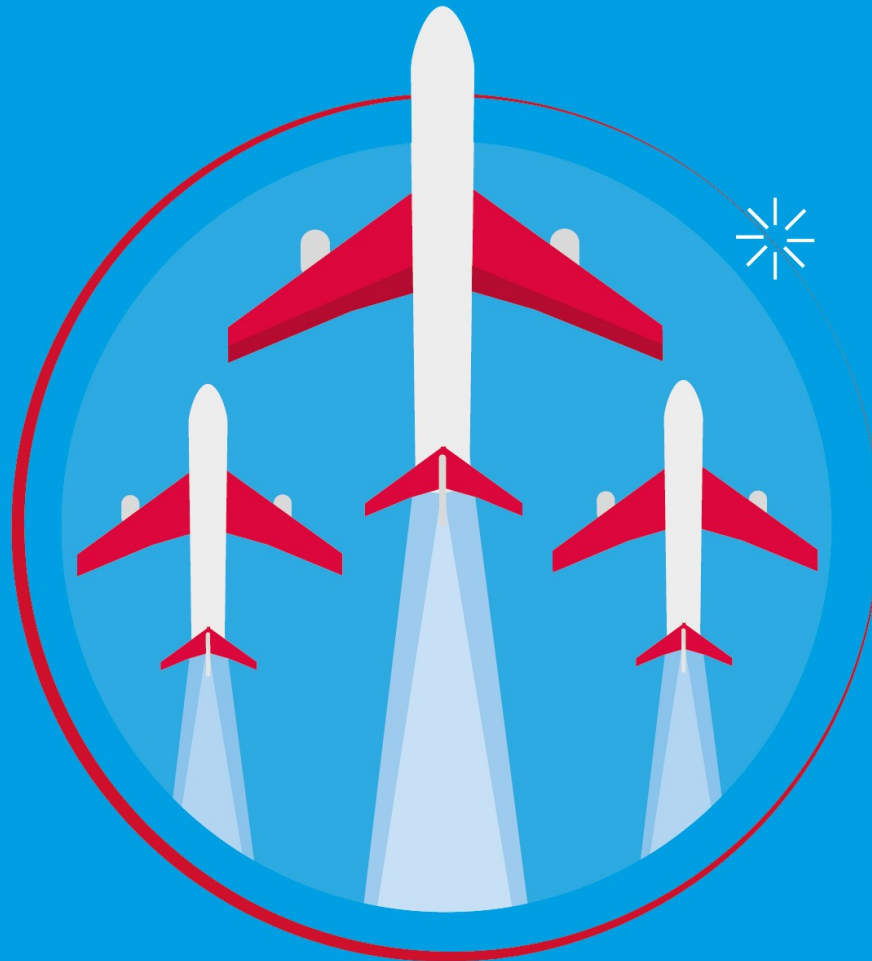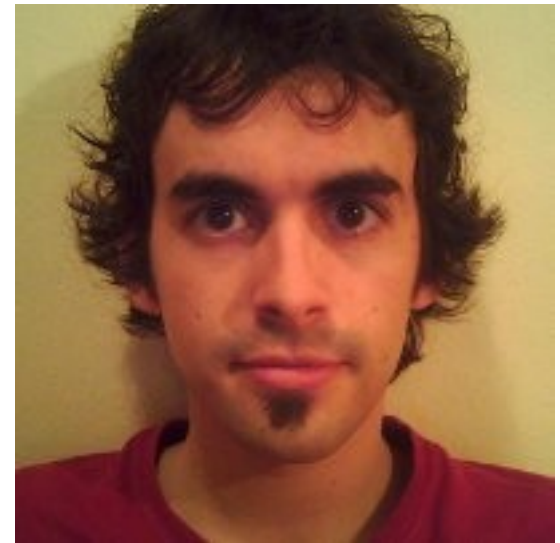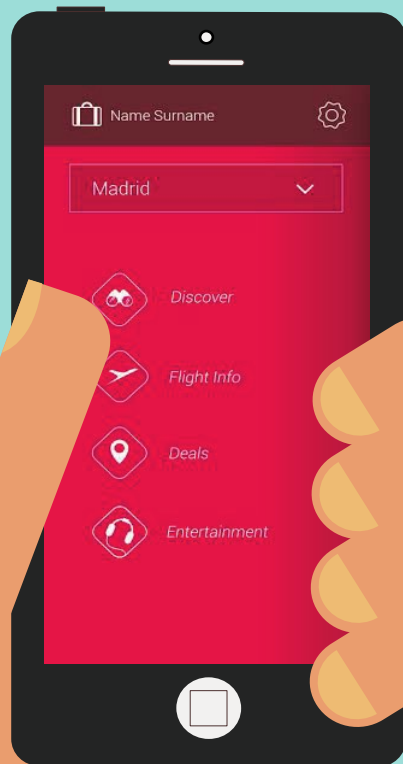# **Python** in the sky

Abstract

- "How we built a *W.I.F.E.** system using Python"
  - ***W**ireless **I**n-**F**light **E**ntertainment

- Main topics:
  - 1) Product requirements
  - 2) Architecture decisions
  - 3) Atypical challenges

Hi!

- I'm David Arcos

- Python/Django developer since 2008

- Co-organizer of **Python Barcelona**

- Lead engineer at **Immfly**
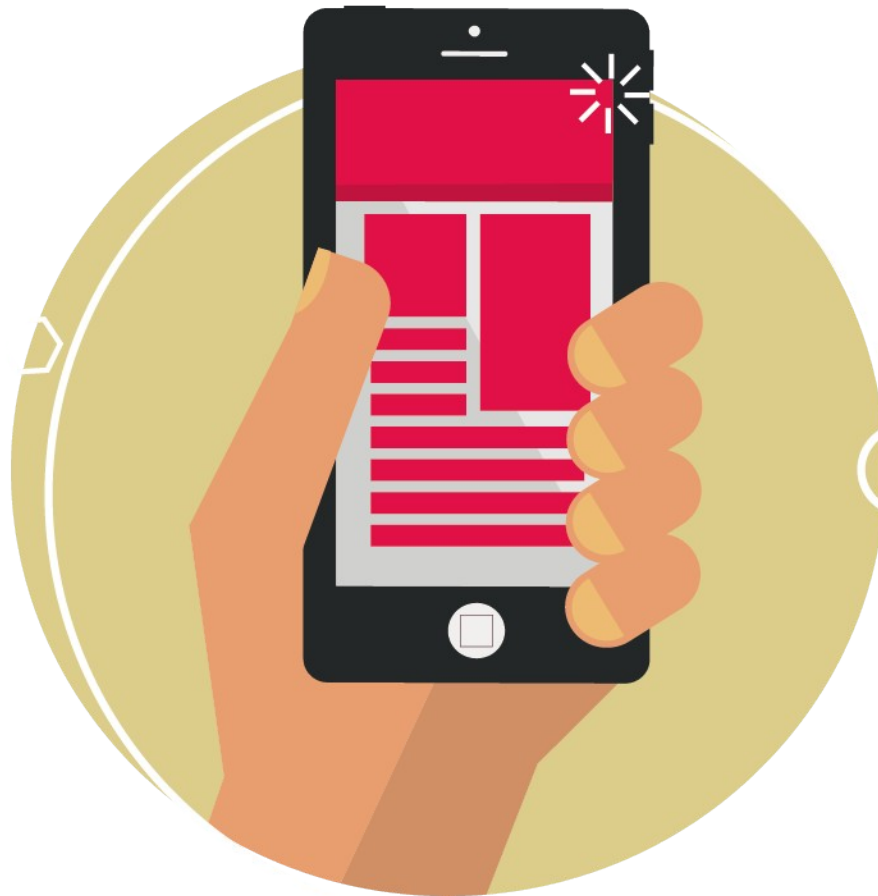
Immfly

Immfly is a new Entertainment, Retail and Communication platform for the in-flight experience.

Focused on the European domestic flights market, Immfly offers wireless content to passengers via their Personal Electronic Devices.

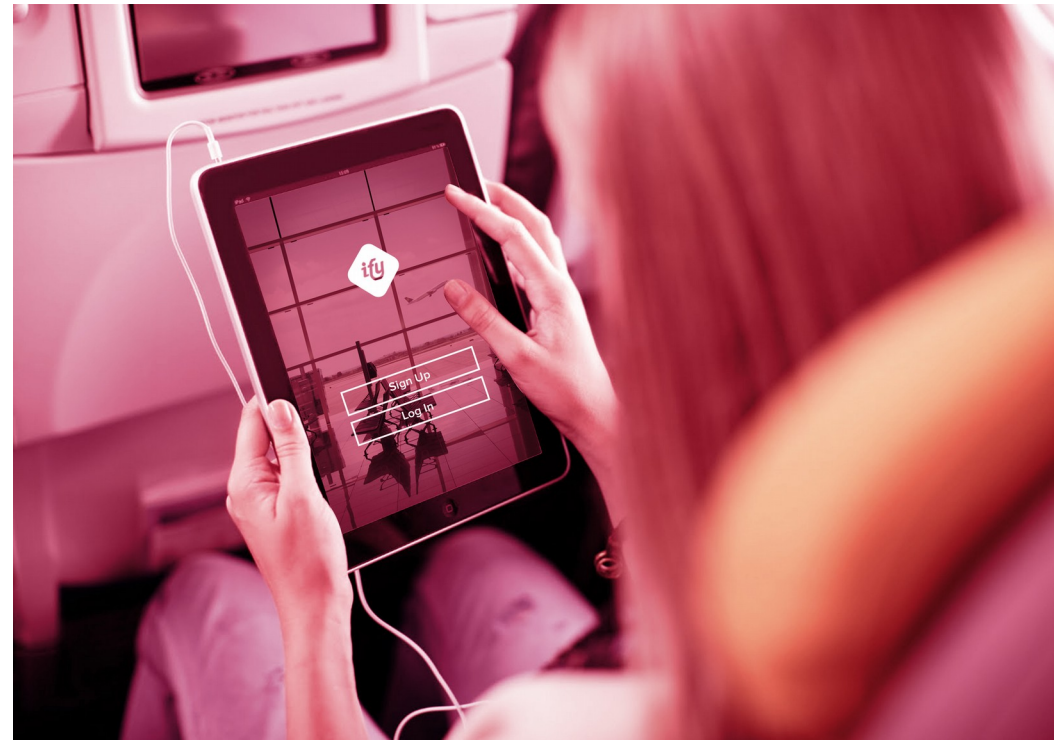# 1) The product requirements

ify

How does it work?

The passenger's device

(smartphone/tablet/laptop)

connects to the wifi

(no internet!)

and opens the app

(web/android/ios)

What kind of services?

What kind of services?

- Flight information

- View TV Shows, videos

- Read newspapers, magazines

- Get guides, offers, deals

- Etc...

But it's off-line
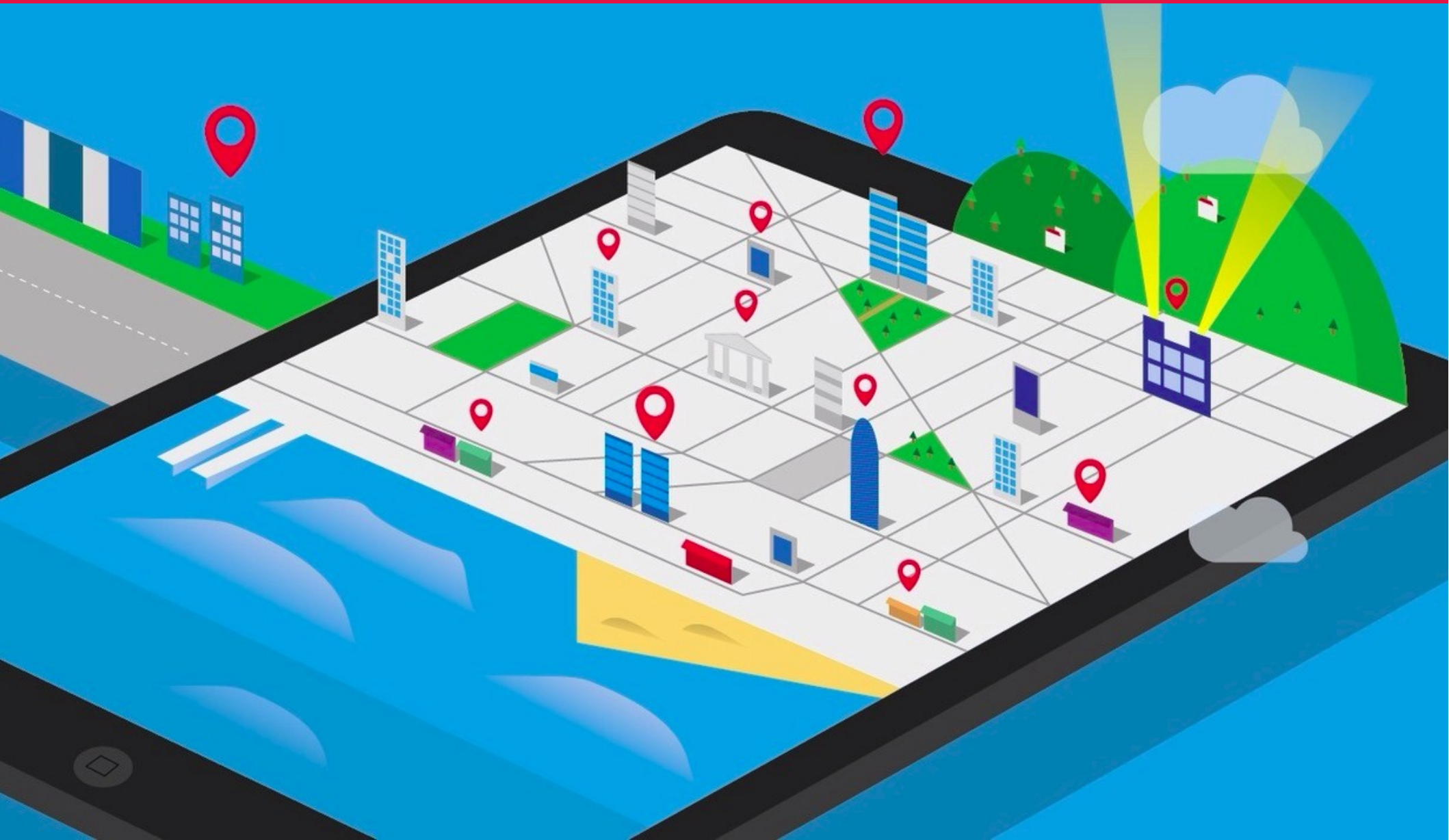
ify

We need eventual connectivity

- Update the contents
  "I don't want yesterday's newspaper!"

- Send "booking" actions to external APIs

- Do payments

- Send emails

Ground mode

ify

Ground mode

- Keep the experience after landing

- Bring your contents with you

- Get email confirmations

ify

2) The Architecture

- In the airplane:
  - Frontend & Backend
  - System services
  - Hardware

- In the datacenter:
  - The "Hangar"
  - Ingests

Frontend apps

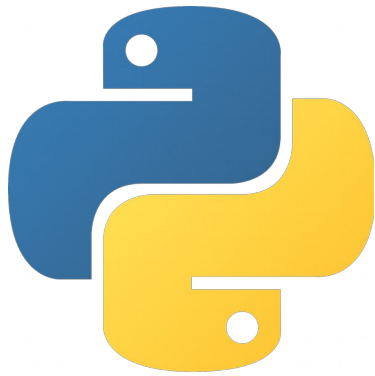- Web app:

- Native apps:

Backend API

- *"A widely used general-purpose, high-level programming language"*

- *"The web framework for perfectionists with deadlines"*

- *"A powerful and flexible toolkit that makes it easy to build Web APIs"*

HTTP server

NGINX

gunicorn

Supervisor hang on to your processes

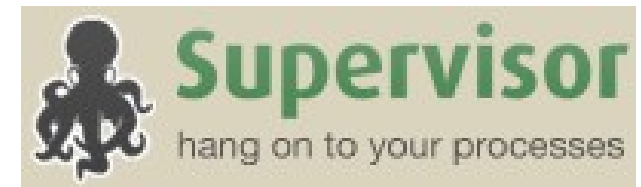- Web Server
- Static contents

- WSGI Server

- Monitor and control:
  - API
  - daemons
  - celery

Databases

- SQL data
  - Django ORM
  - Critical transactions

- NoSQL data
  - Cache, sessions
  - Metrics, expirations

Network

SSH

– Through VPN
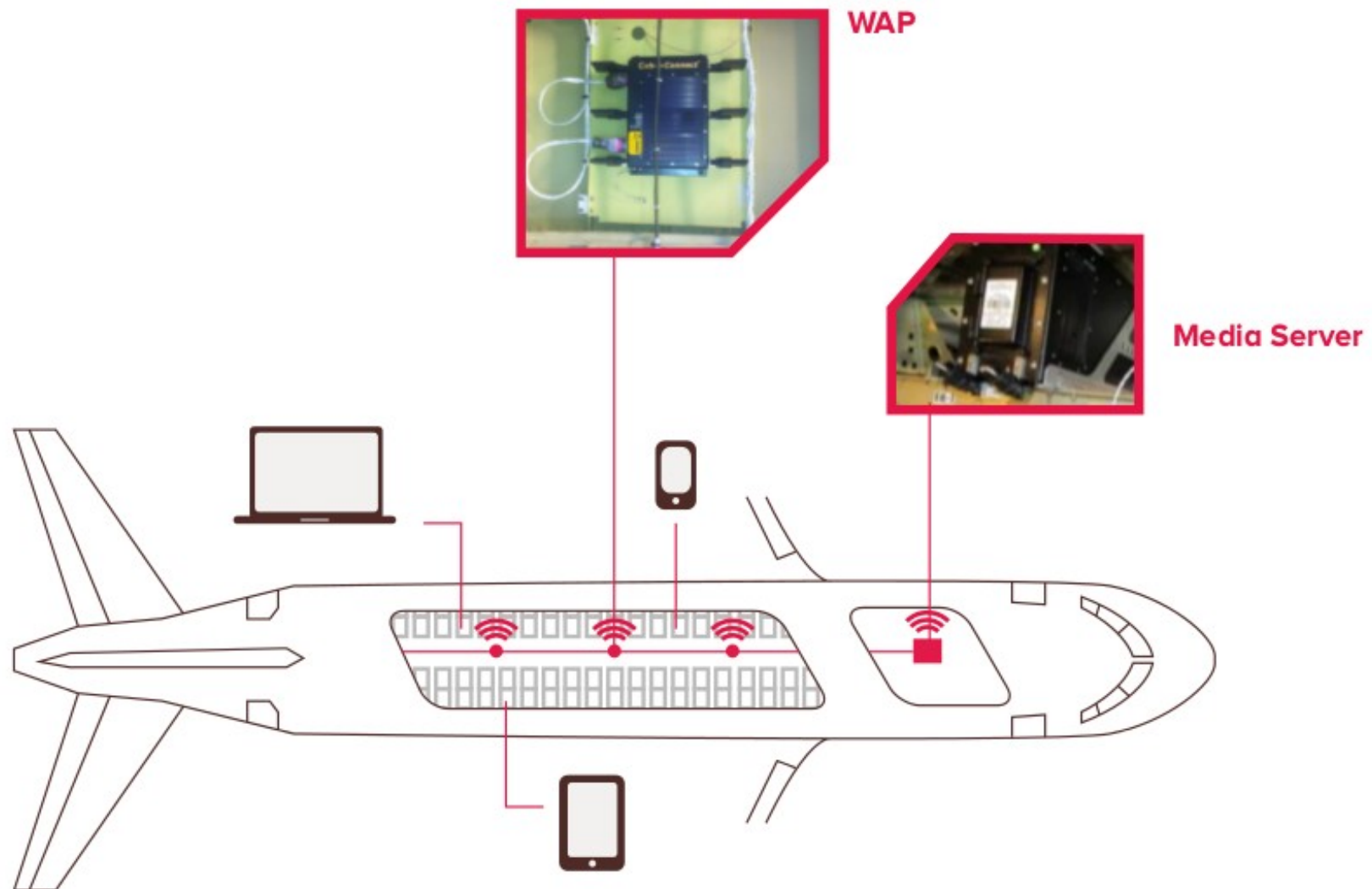
– Hidden port

– PK Auth

Ansible

– Deployment

– Config
Management

– Pull mode

Fabric

– Initializing

– Other tasks

Hardware

Wireless Access Points

- 3 WAPs per plane

- Provide the WiFi
  - Intranet

- Isolated users

Aircraft Server

- Embedded computer:
  - Hardened, certified, military-grade

- Internal 3G data card
  - used when grounded

- Avionics bus
  - read-only!

Avionics data bus

We have access to this data, in real-time:

- altitude
- flight_id
- ground_speed
- heading
- latitude
- longitude

- mach_speed
- outside_temperature
- pitch
- roll
- wind_speed
- yaw

Discrete-time signals

- DCFAILSIG
- ACFAILSIG
- OVERTEMPSIG
- GSM_POWER_STATUS
- ENB2SIG
- ENB1SIG
- ENB0SIG
- GSMSIG_STATUS
- CPLD_REV0
- CPLD_REV1

- SYSENSIG
- ENB3SIG
- ENB4SIG
- ENB5SIG
- ALERT
- CONFIGSIG0
- CONFIGSIG1
- CONFIGSIG2
- INTTEST_OUT
- INTTEST_IN

- ISO_OUT0
- ISO_OUT1
- ISO_OUT2
- ISO_OUT3
- GPIO_DCFAILSIG
- GPIO_ACFAILSIG
- GPIO_OVERTEMPSIG
- GPIO_SYSENSIG

The Hangar

- Hosted in the Internet

- Central point

- Orchestrates the operations

- Update contents, databases, code...
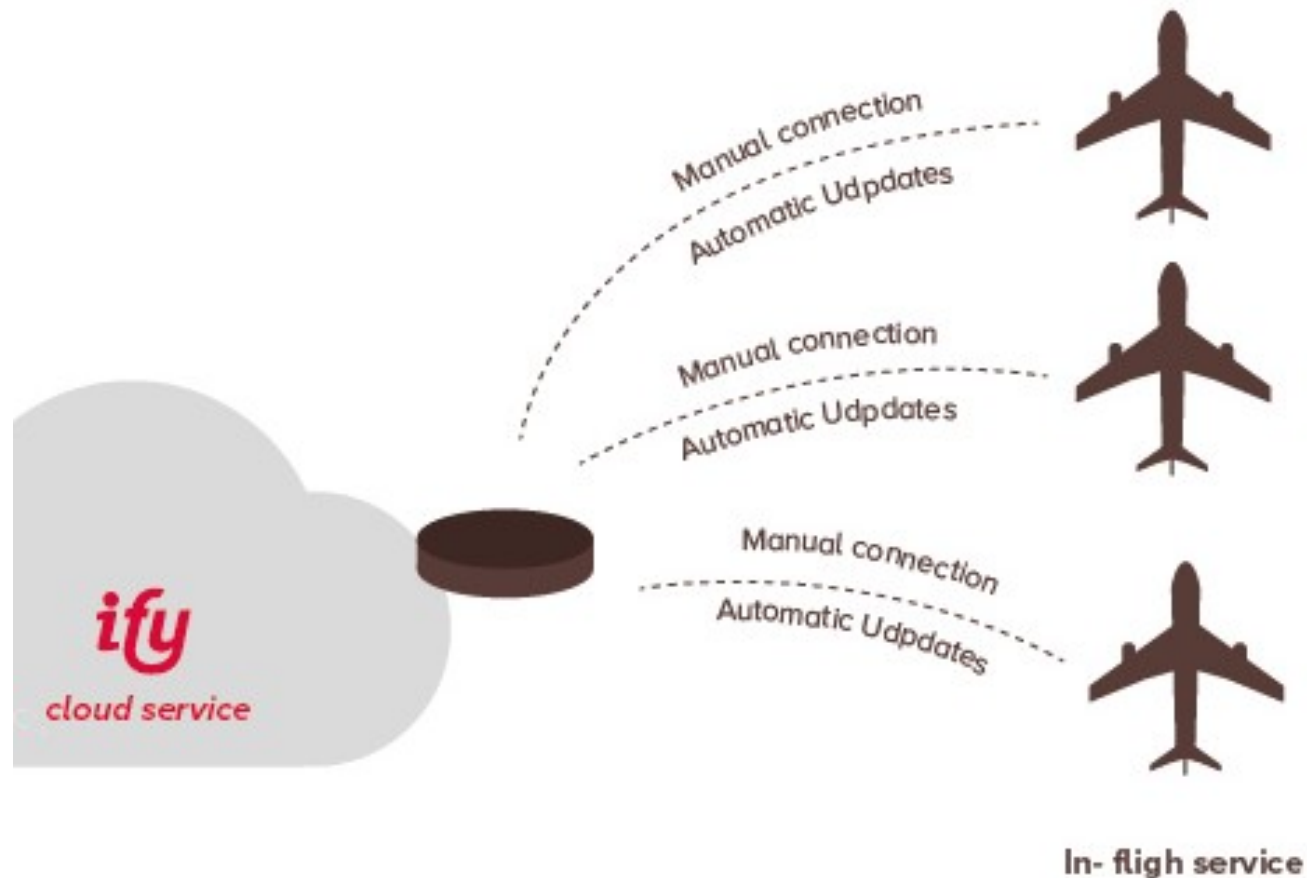
Hangar Operations
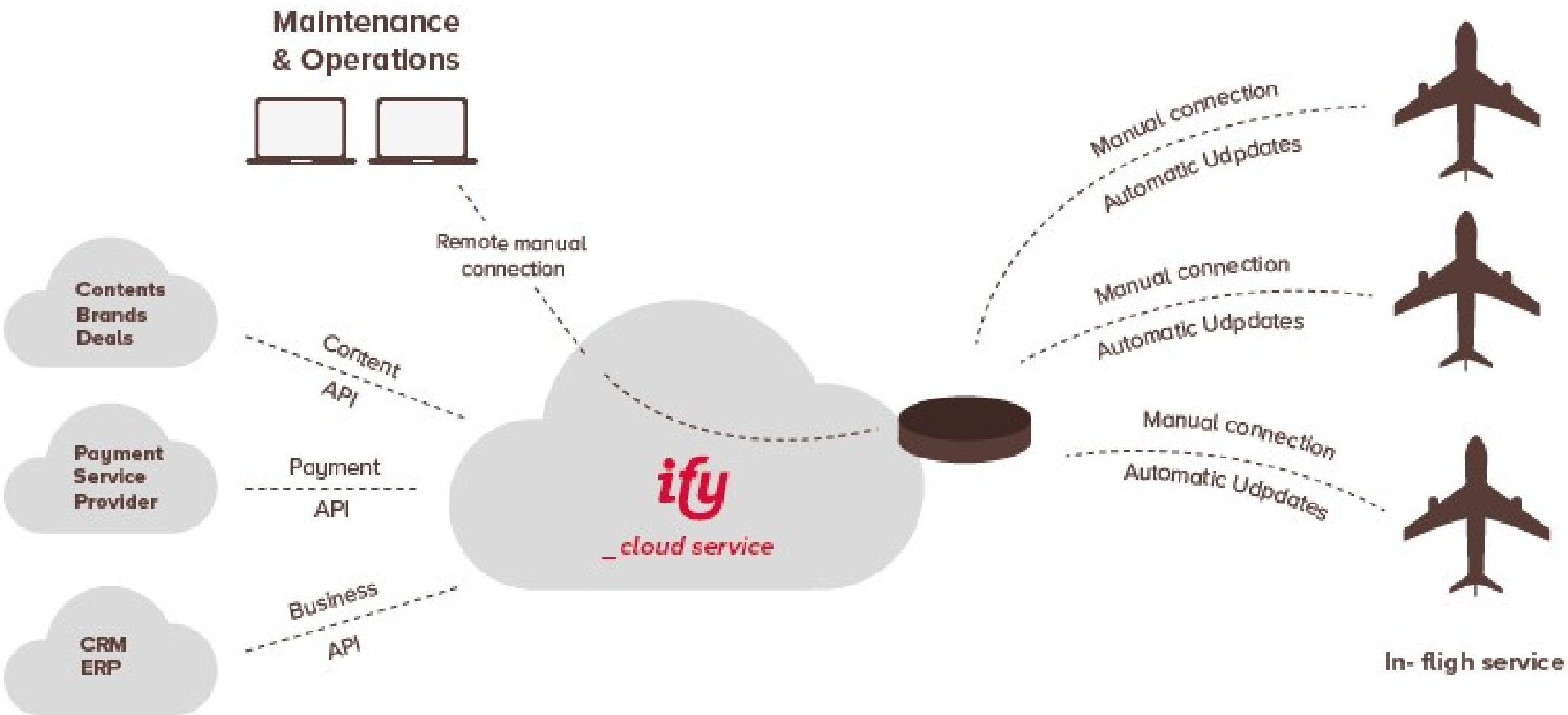
Resource ingest

- Thousands of resources, from ~20 providers
  - Per language
  - Per country
  - Per category: Videos, Readings, Deals...

- Some external APIs are questionable:
  - Missing SSL, Documentation...
  - Manually updates to an ftp...
  - Data in excel files...

Ingesting videos

- Ingest from SFTP / AWS S3

- Send to AWS ElasticTranscoder
  - Convert to HLS (HTTP Live Streaming)

- Sync chunk by chunk

Ingesting readings

- Ingest from SFTP / AWS S3

- Celery task:
  – Reduce size (image resolution)
  – Generate thumbnails

Ground mode

- Similar to the *"Aircraft"* mode

- Minor differences:
    - No Videos (we don't have the permissions)
    - More focus on destination offers
    - Online functionality (like *"forgot my password"*)

- Made to scale

3) Atypical challenges

- Specific perks of working with aircrafts

- Dealing with airplanes has extra challenges

- Problems we weren't expecting beforehand

- Also, *"mistakes where made"*

Regulations & Certifications

End of 2013, EASA allowed use of PEDs

EASA Safety Information Bulletin

SIB No.:   2013-21
Issued: 09 December 2013

Subject: Use of Portable Electronic Devices during Commercial Air
                Transport Aircraft Operation

Certificate everything!

- Had to certificate every step
  - The hardware was already certified
  - The WAPs were already certified
  - Lots of wireless tests

- Even the smallest screw has to be certified!

- Took us 6-9 months!

Certificate everything!

- Had to certificate every step
  - The hardware was already certified
  - The WAPs were already certified
  - Lots of wireless tests

- Even the smallest screw has to be certified!

- Took us 6-9 months!

BORING
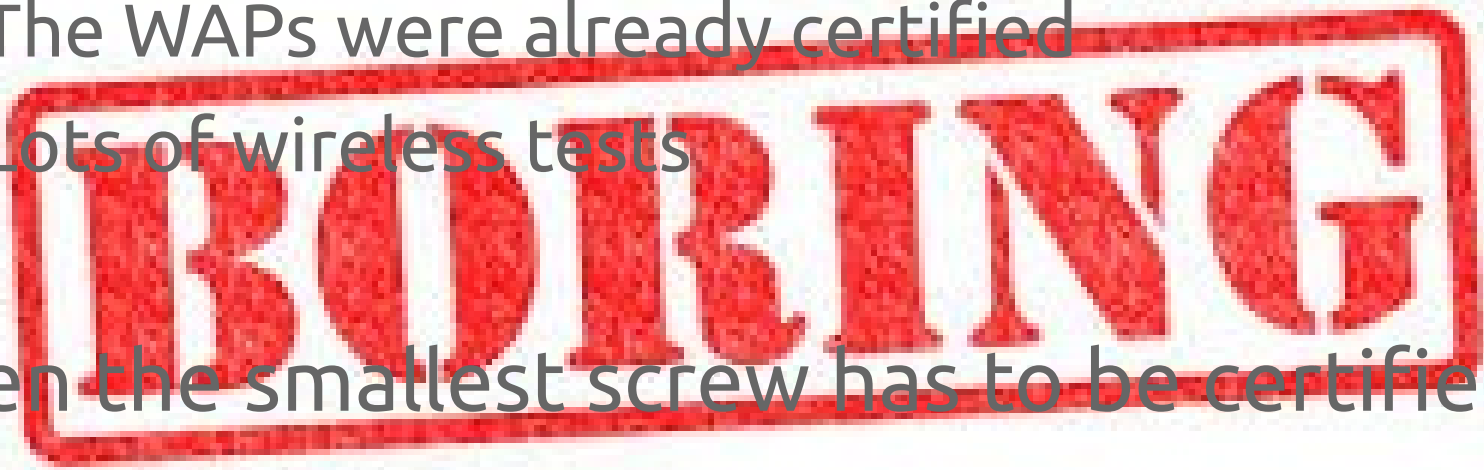
Intermittent connectivity

- We get 30~90m of connectivity per day
  - **10-15** minutes after each flight

- Sometimes, **roaming** applies
  - Limited synchro

- Improve the deployment tools
- Optimize the performance

Hard shutdown

- Electrical power is suddenly removed, and our airplane server is turned off

- Happens often:
  - When changing power from engine to external, after landing
  - If there is a storm, or just if the pilot wishes

- We could only mitigate...

Mistake: trusted the hardware

- File system corruption
  - The HDD write buffer is sometimes lost
  - Corrupted files
  - *fsck* at start
  - *sync* after each deploy
  - Added several consistency checks (for contents)

- Internal clock corruption
  - Logs showed wrong times
  - Added NTP checks (requires connectivity)

The CAP Theorem

"It's impossible for a distributed system to simultaneously provide all three:"

- **C**onsistency

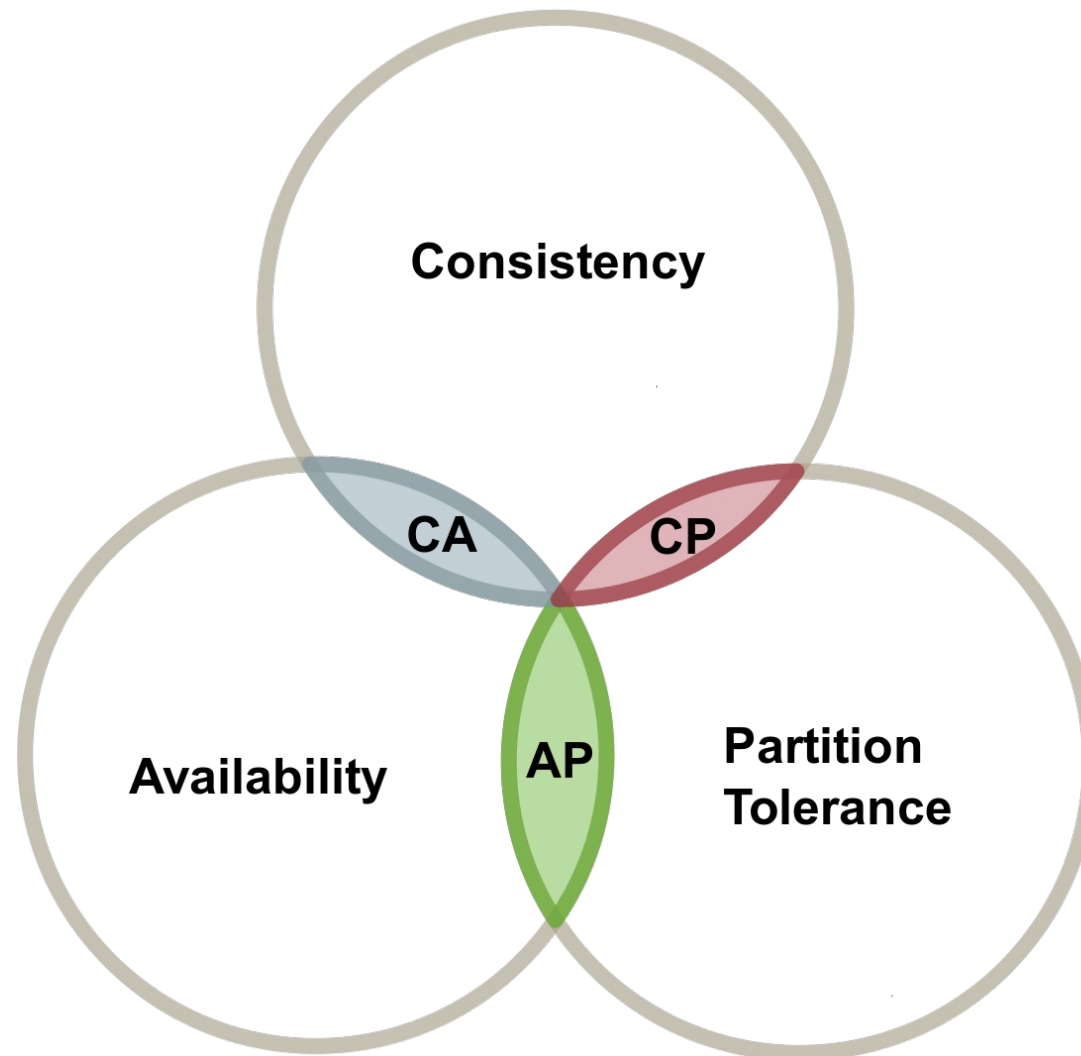All nodes see the same data at the same time

- **A**vailability

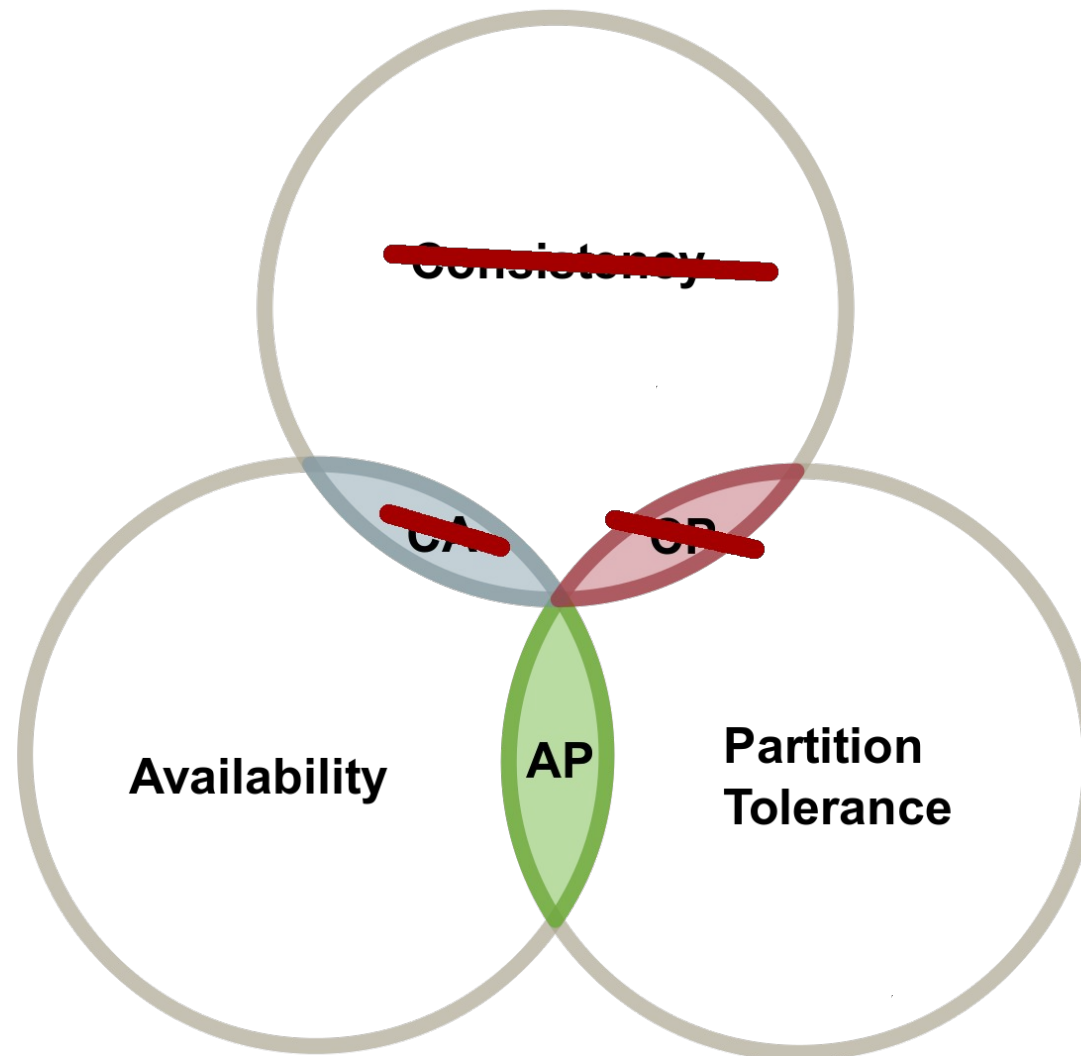Guarantee that every request receives a response

- **P**artition tolerance

Continue to operate despite network partitioning

Big mistake: try to beat the CAP

Settled for "Eventual Consistency"

Recap

- Complex project with lots of features
  - Kept a modular design

- Could implement everything
  - helped by existing Python libraries

- Learned a lot from unexpected challenges

Conclusions

- Python made it possible!

- Very versatile, covers all our use cases

- *"We stand on the shoulders of giants"*

- Developed in a short time

Thanks for attending!

- Get the slides at http://slideshare.net/DZPM

- **We're hiring! http://immfly.com/jobs**

Questions?

ify