



3 December

CONTAINERS TODAY

Stockholm

Powered by



The Service Hotel

How to design a Container platform with Docker

id



**uid=1000(sune-keller),gid=45(denmark),
groups=2013(alm-brand),2018(docker-captains)**

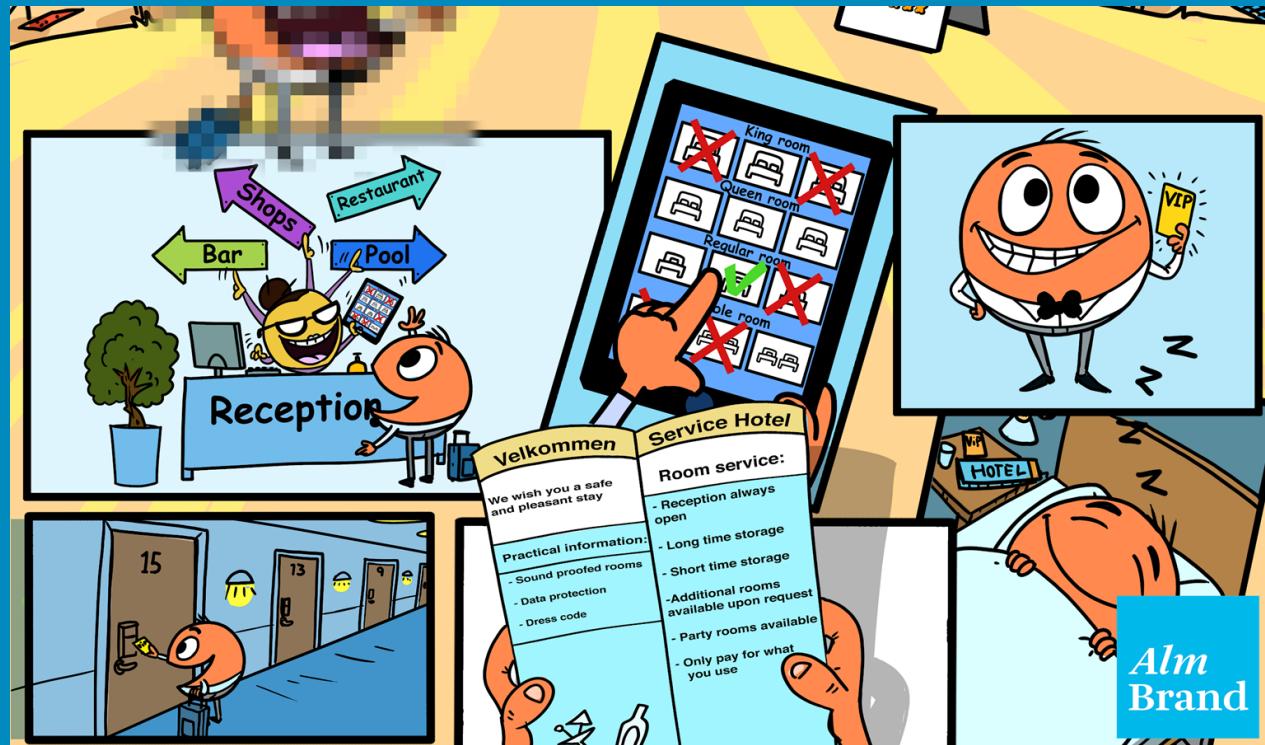
General concept

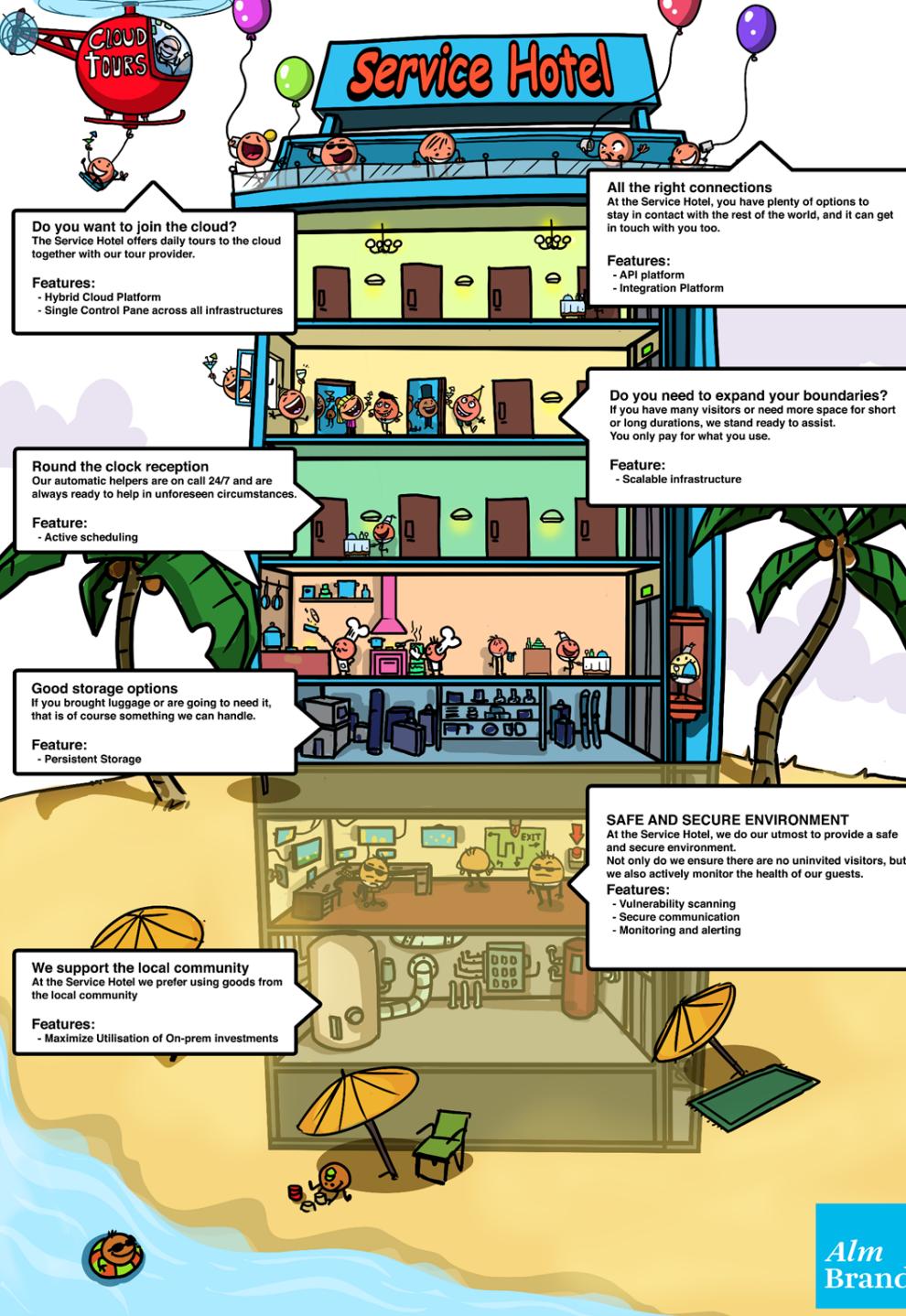
- The aim is to make it as easy as possible to Do The Right Thing™ with as few obstacles as possible in the 80% most common cases
- It serves as the deployment target for a variety of workloads, provided they adhere to the hotel rules
- Languages/stacks are onboarded with assistance from the Platform Team
- The Service Hotel fits into a larger map of "Where to do what?"



The 80% most common cases

- Optimized for throughput
- The easier the happy path is, the more likely users are to choose it
- Leaves more time for handling special cases, further speed up happy path





Feature overview

- Divided into “above the ground” and “below the ground”
- Existing as well as planned features
- Focus on high-level features and concepts rather than implementation specific terms
- Each feature can be loosely associated with a traditional hotel feature

Kitchen and piccolos: Round the clock reception/service



- In order to ensure we keep our services running, Docker Swarm ensures we keep having the requested number of containers
- Scheduling placement constraints allow us to simply indicate a deployment stage (“dev”, “qa”, “prod”), and only nodes in that stage will run the deployed service
- Infrastructure services are deployed in “global” mode to have one instance on every node

Boiler room: Maximizing utilization of existing on-prem investments

- Alm. Brand has its own data centres
- > 99.5% virtualized workloads
- Previously, over-committing implied instability and crashing apps, with difficult troubleshooting
- Containers don't magically reduce the load, but allow isolating services from each other



Book a whole floor: Infrastructure as code allows for horizontal scaling

- Packer, Terraform and Ansible combine well to build, test and deploy your infrastructure the same way as apps and services
- There are many ways of doing this, ours is just one ([link to comparative talk](#))
- Chargeback is something we're looking to add to the platform, to help make investment costs more transparent



Long-term luggage storage: Persistent stateful services

- Using a Docker volume plugin for our storage provider, we support persistent storage in simple setups for Redis, Postgres and others
- If a container or entire node goes down, Swarm will schedule the container on another node in the same stage (based on placement constraints) and the same volume will be mounted
- We have not yet provided high-availability clustered storage options; rather, we expect the consuming services to fail gracefully and recover fast



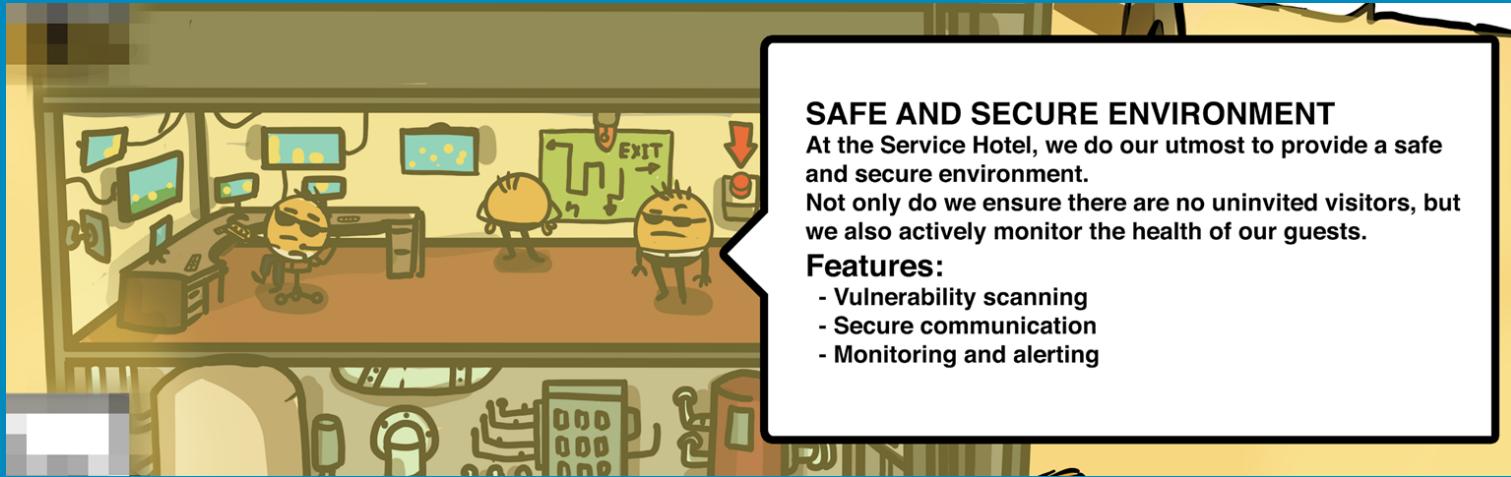
Penthouse: API and integration platform



- With API and integration services being the most senior guests at the hotel, templates and standards are well established
- Creating a new API endpoint and deploying to production takes a matter of minutes; just have to code the logic
- A *very simple* chat bot creates source code repos in GitLab, Docker image repos in Docker Trusted Registry and replies back to the user in seconds; again, paving the happy path for the 80% most common cases

Security crew: Monitoring and alerting for security and health

- Vulnerability scanning of every pushed Docker Image reports back any CVEs to the developer directly in the merge/pull request
- Monitoring is auto-enabled through service discovery (Consul+Prometheus)
- Access to secrets stored in Vault are subject to audit logging and policies
- Automated mutual TLS is a Coming Soon™ feature of the Service Hotel ([Vault+Swarm plugin](#) and policy writing required)
- A fine balance between agility and control; good base level => fast track



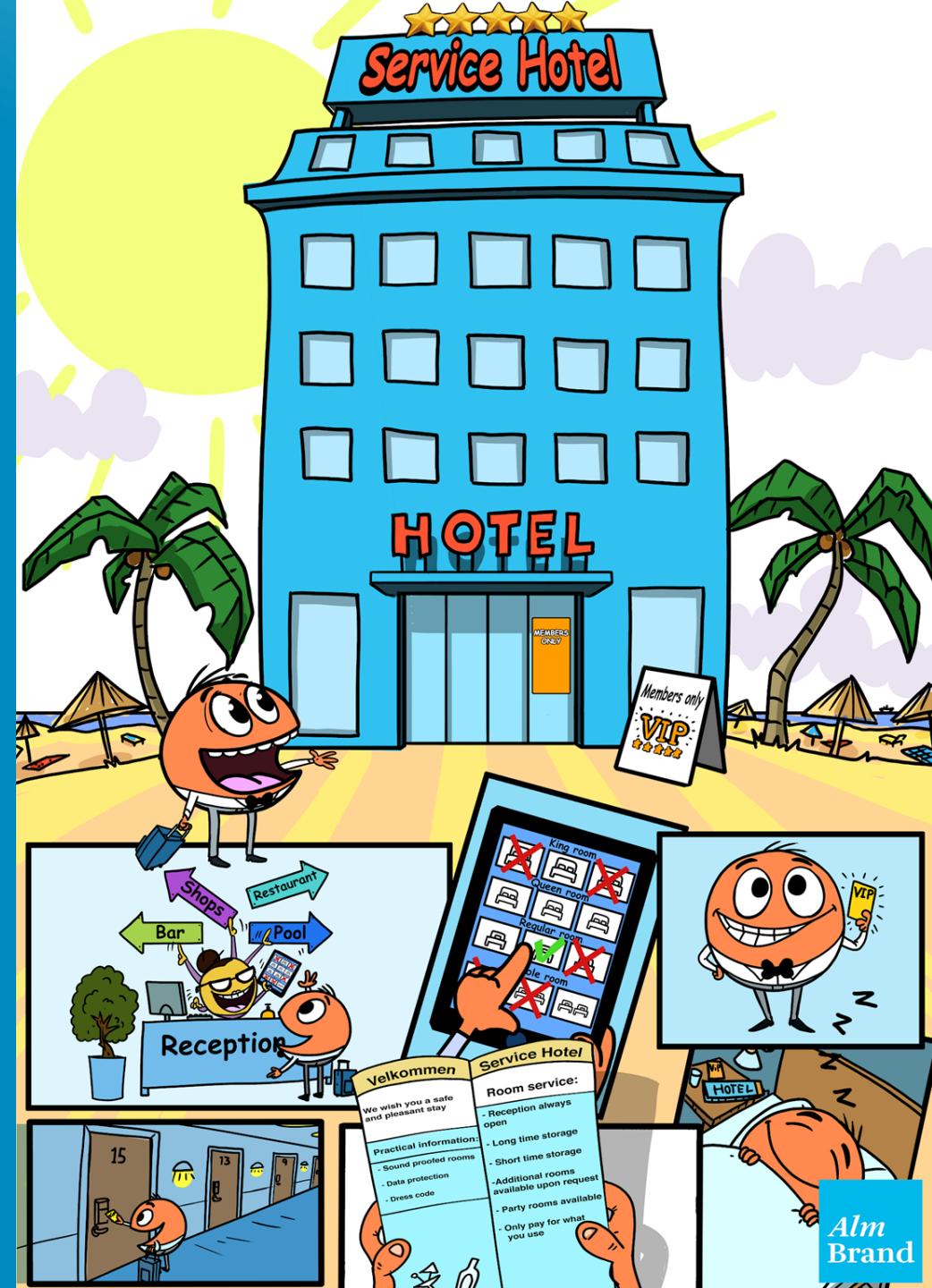
Tour partners: Cloud portability

- Containerized services are portable by design
- We've taken care to design the underlying services to support running on-prem as well as in the cloud
- A centralized control plan has pros and cons
- Federated App Management looks interesting
- We're not there yet



Recap

- The aim is to make it as easy as possible to Do The Right Thing™ with as few obstacles as possible in the 80% most common cases
- It serves as the deployment target for a variety of workloads, provided they adhere to the hotel rules
- Languages/stacks are onboarded with assistance from the Platform Team
- The Service Hotel fits into a larger map of "Where to do what?"



Questions?