

PRODYNA AG

Final Presentation – Voting Platform

28.07.2016

Welcome!



> Referent

Sven Lissek, PMP
Senior Consultant

At PRODYNA since: 2007



➤ Agenda:

1. Summary of requirements
2. Solution Overview
3. Architecture in detail
4. Live Demo
5. Next Steps

➤ Summary of requirements

Functional

Voting Application

➤ Summary of requirements

Functional

Voting Application



Browser Based

➤ Summary of requirements

Functional

User

- Login mechanism
- Role concept: Admin, User

Voting Application



Browser Based

➤ Summary of requirements

Functional

Voting Application

User

- Login mechanism
- Role concept: Admin, User



Browser Based

Votes

- Are either voted or not
- Result is visible after vote
- Can only be voted once
- Vote cannot be changed

➤ Summary of requirements

Functional

Voting Application

User

- Login mechanism
- Role concept: Admin, User

REST API

- Must be secured with authentication
- Must be available for mobile clients



Browser Based



Votes

- Are either voted or not
- Result is visible after vote
- Can only be voted once
- Vote cannot be changed

> Summary of requirements

Functional

Voting Application

Requiriements	User 	Admin 
Votes		
View all created votes	Yes	Yes
Create new votes	Yes	Yes
Vote exsisting votes	Yes	Yes
Edit votes	Own	All
Delete votes	Own	All

➤ Summary of requirements

Non-Functional



- Horizontal scaleable
- Highly available
- Real time monitoring



- Developed in Git
- Release unit as zip file
- All deliveries in Git



- DB User with restricted permissions

- Good platform for adding further requirements → Largely extended if successful
- Mixed Team - Devs from Customer and PRODYNA → Take app over for maintenance after release

➤ Agenda:

1. Summary of requirements
2. Solution Overview
3. Architecture in detail
4. Live Demo
5. Next Steps

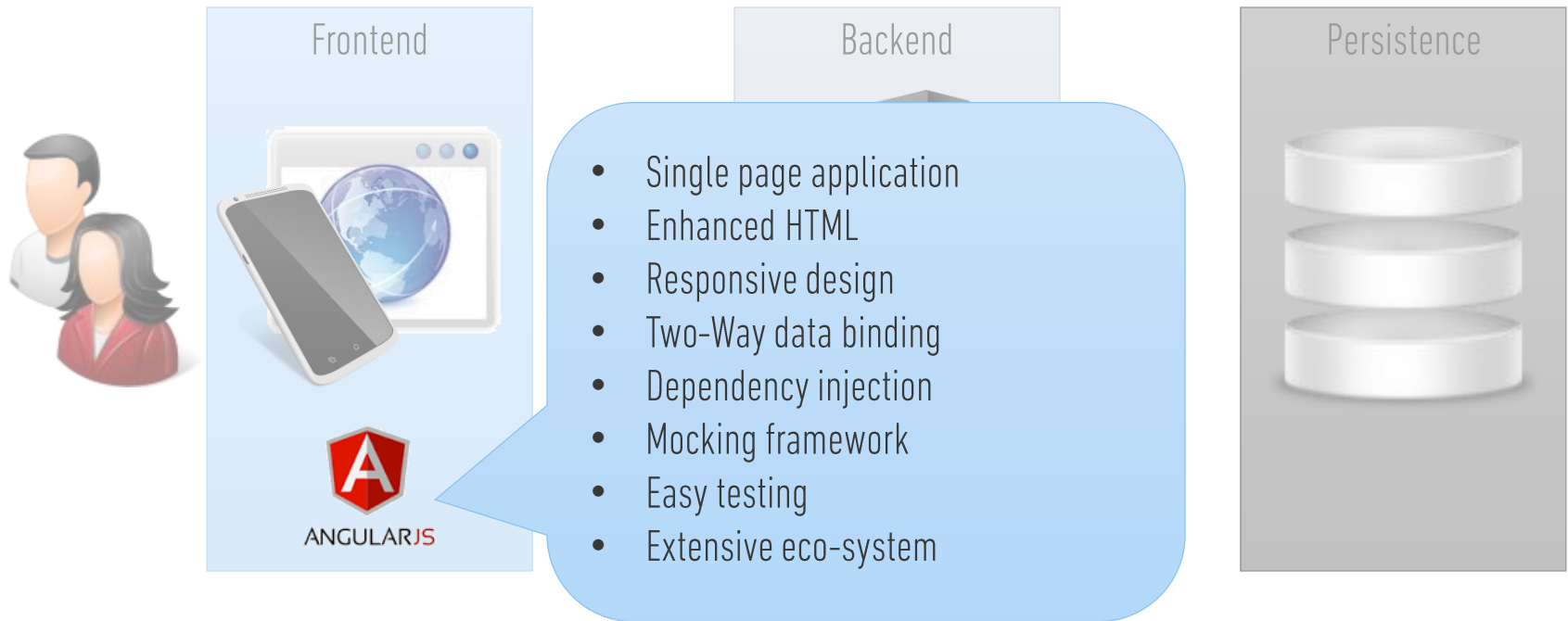
➤ Solution overview

Voting application



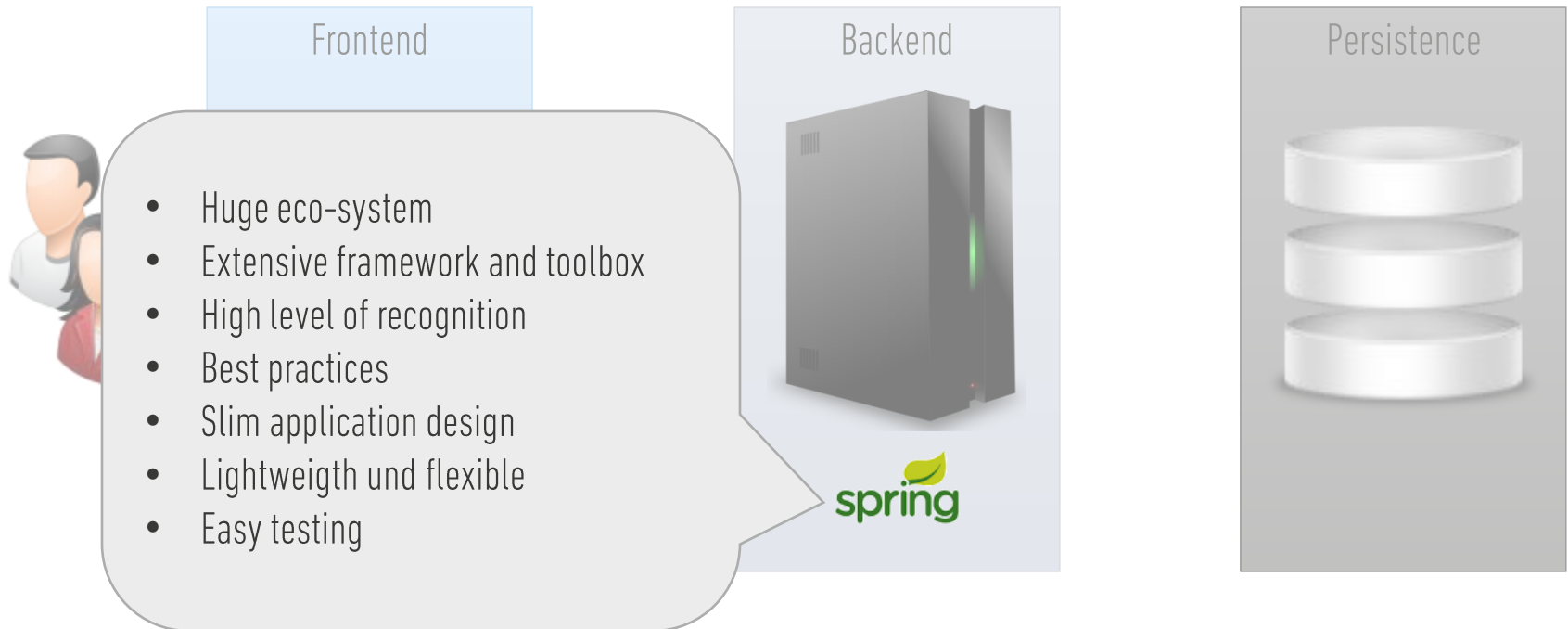
> Solution overview

Voting application



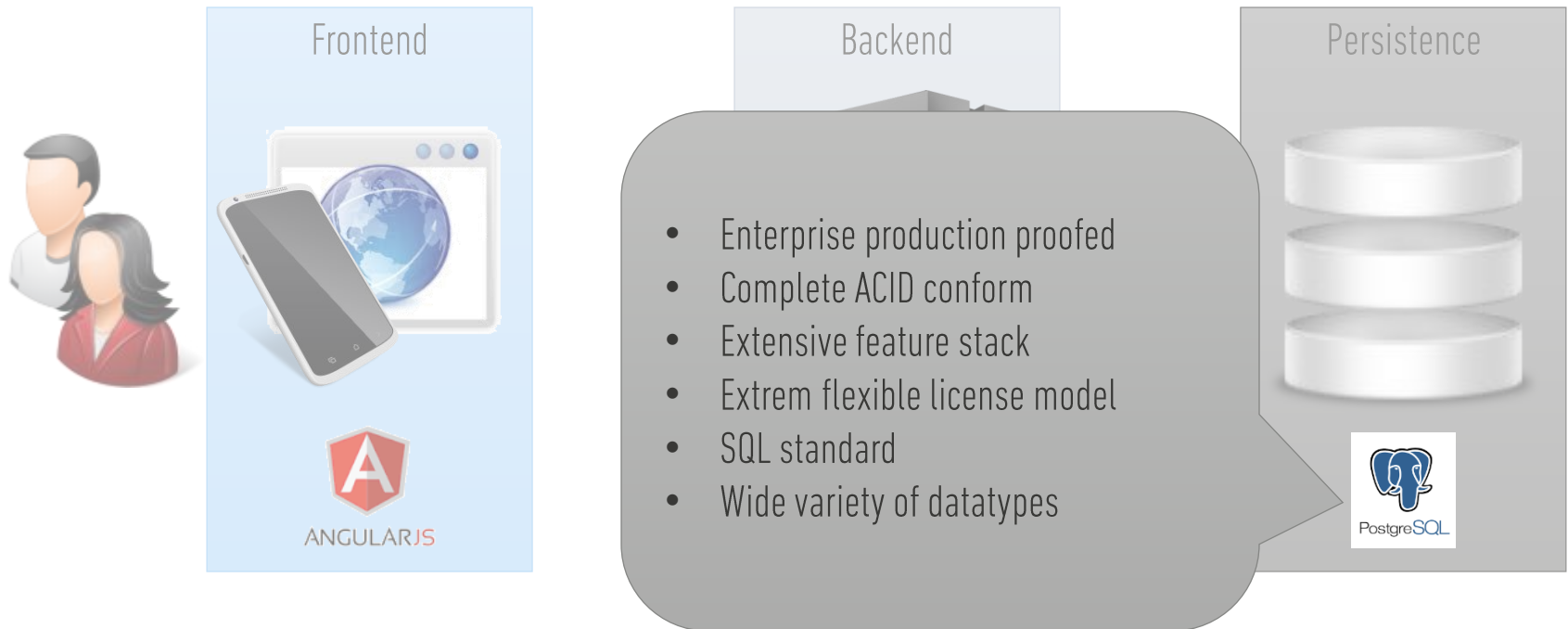
➤ Solution overview

Voting application



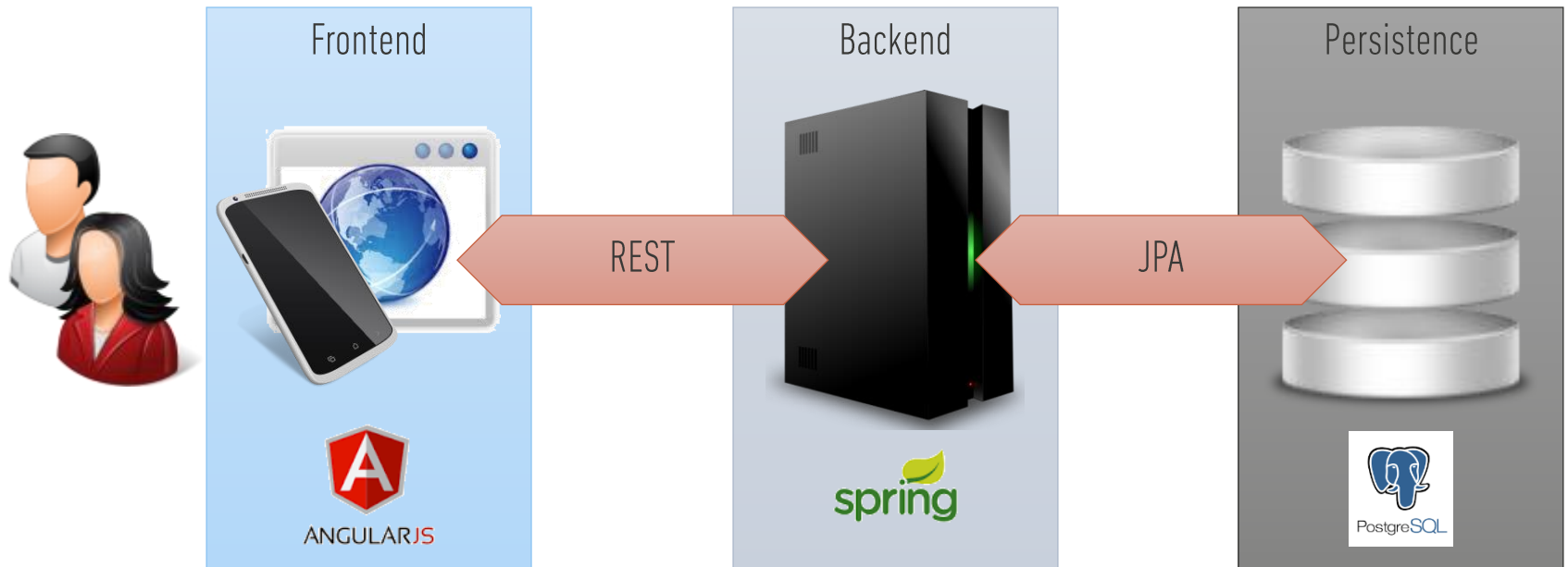
➤ Solution overview

Voting application



➤ Solution overview

Voting application

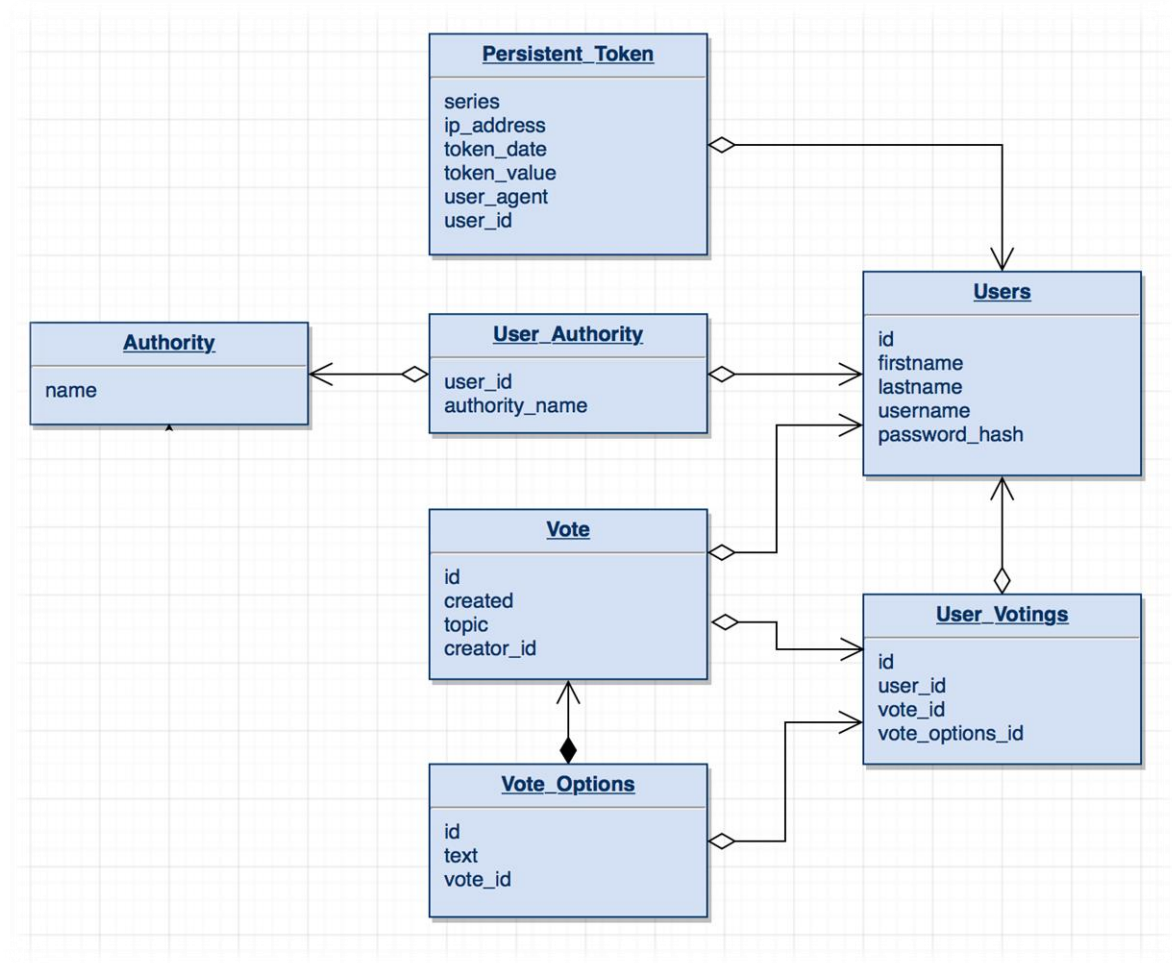


➤ Agenda:

1. Summary of requirements
2. Solution Overview
3. Architecture in detail
4. Live Demo
5. Next Steps

➤ Architecture in detail



Persistence



➤ Architecture in detail

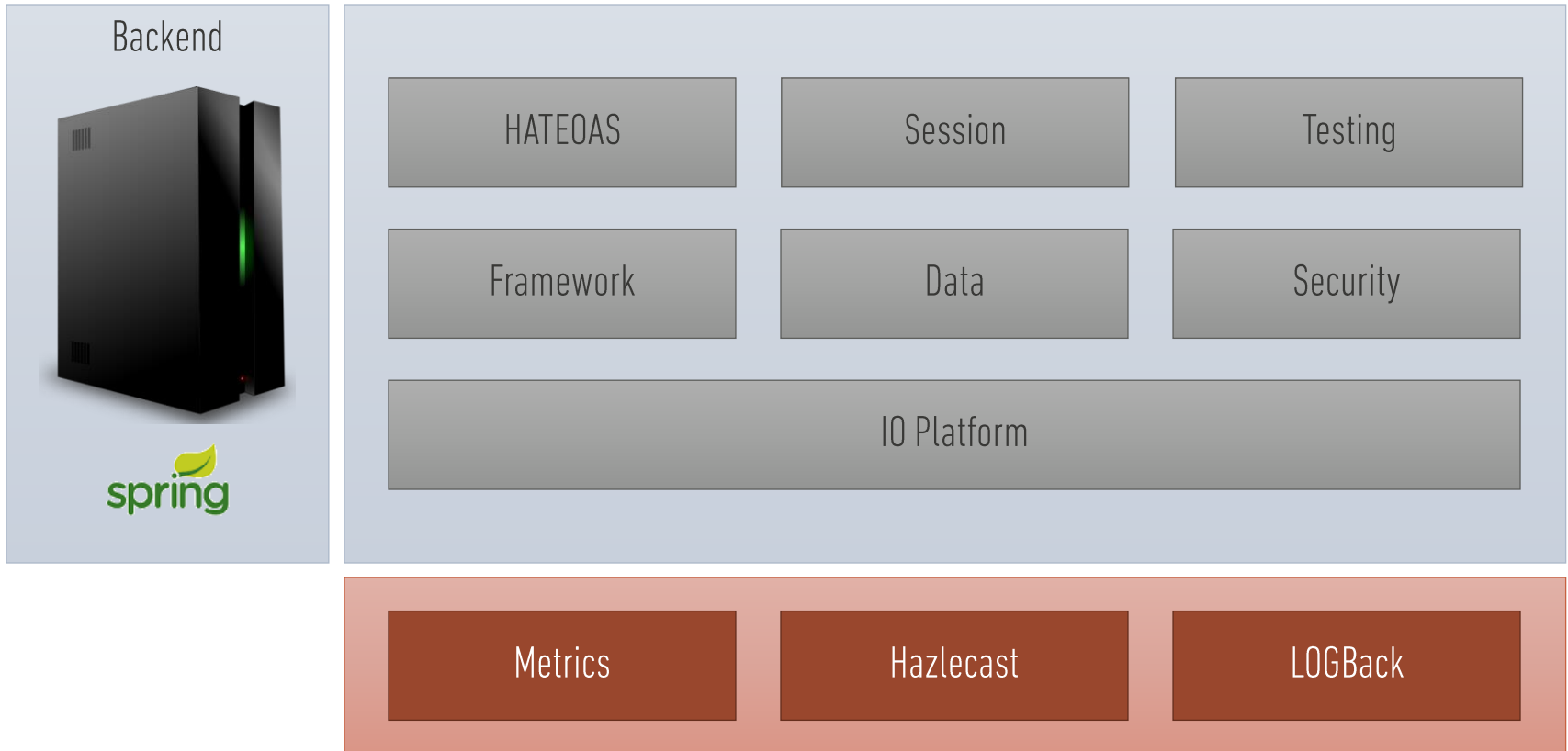
Persistence



Table	Property	User 	Admin 
Authority	Name	ROLE_USER	ROLE_ADMIN
User	Id	2	1
	Firstname	Test	
	Lastname	User	Admin
	Username	Test	Admin
	Password	Test	Admin
User_Authority	User_id	2	1
	Authority_Name	ROLE_USER	ROLE_USER, ROLE_ADMIN

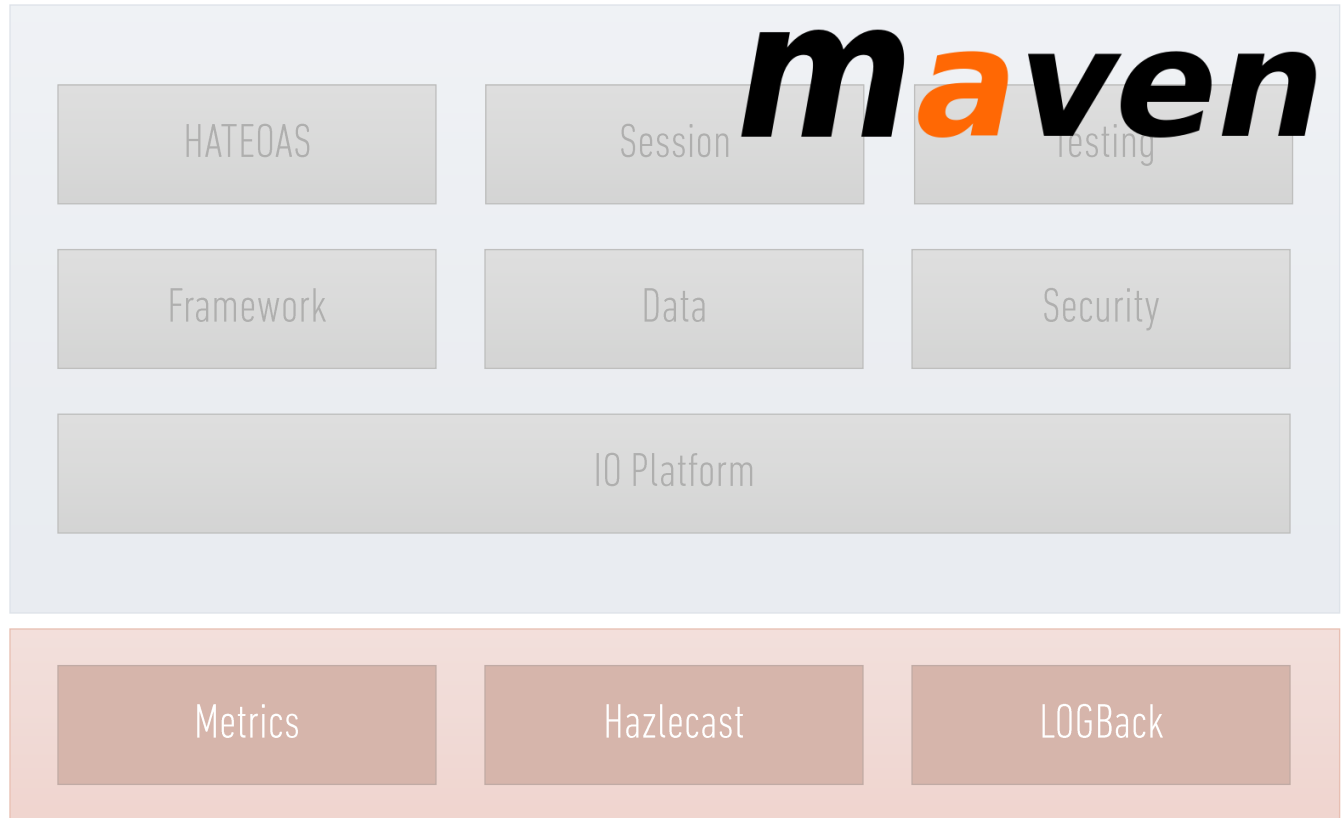
> Architecture in detail

Backend



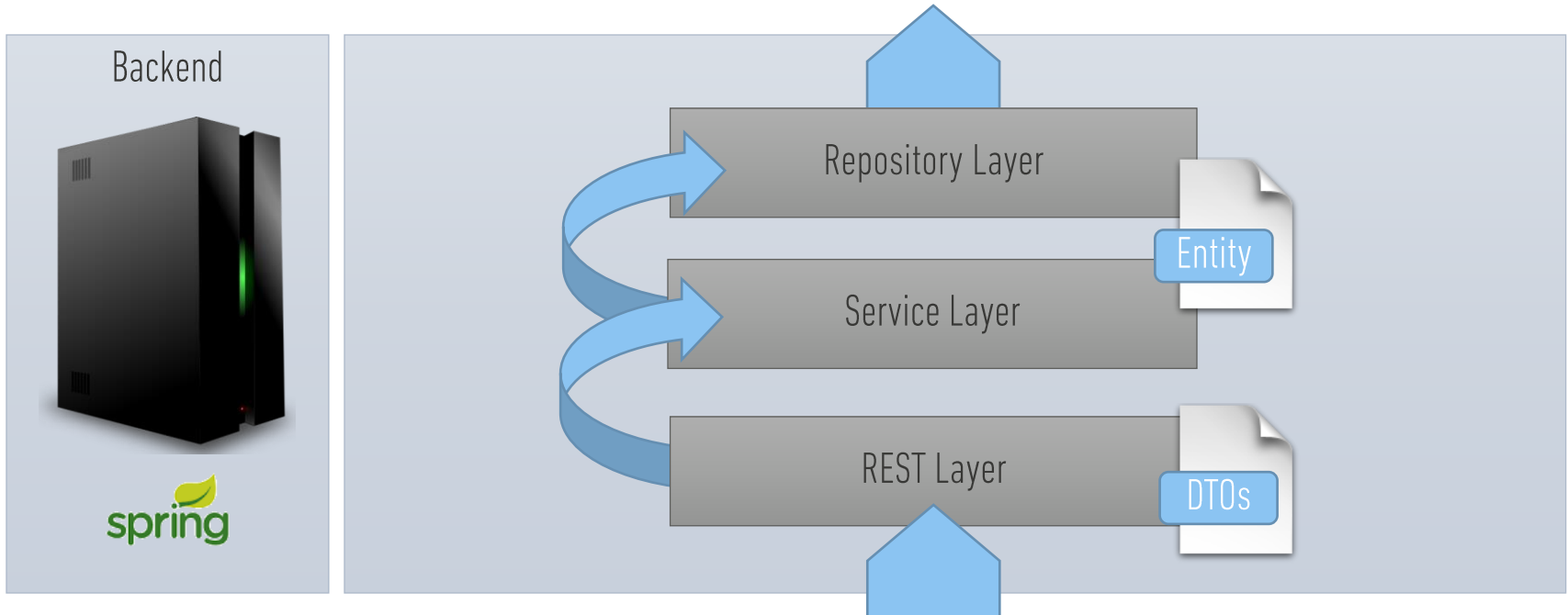
➤ Architecture in detail

Backend



> Architecture in detail

Backend



➤ Architecture in detail

Backend

Backend



Expandable role concept

- Administrator, User, System, Anonymous

Password protection

- BCryptPasswordEncoder for salting and hashing

Authentication

- "classical" Spring security authentication
- Improved remember-me mechanism
- Cookie theft protection
- CSRF protection



➤ Architecture in detail

Frontend



HTML5 Boilerplate

AngularJS

UI Bootstrap

➤ Architecture in detail

Frontend



HTML5 Boilerplate

AngularJS

UI Bootstrap



➤ Architecture in detail

Frontend

Frontend



ANGULARJS

Angular UI Router → state based routing

AngularJS Style Guide → clear update path to version 2.0

➤ Architecture in detail

Application

Application



Horizontal scaling

- Tomcat, Hazlecast, Spring Session

High availability 99,5%

- failover with Cold-Standby strategy for backend and database

Profiling

- Spring actuator enhanced with Dropwizard Metrics and Graphite

➤ Architecture in detail

Application

Application



Code Quality

- JaCoCo plugin (Sonar), FindBugs, Formatter, Templates

REST API

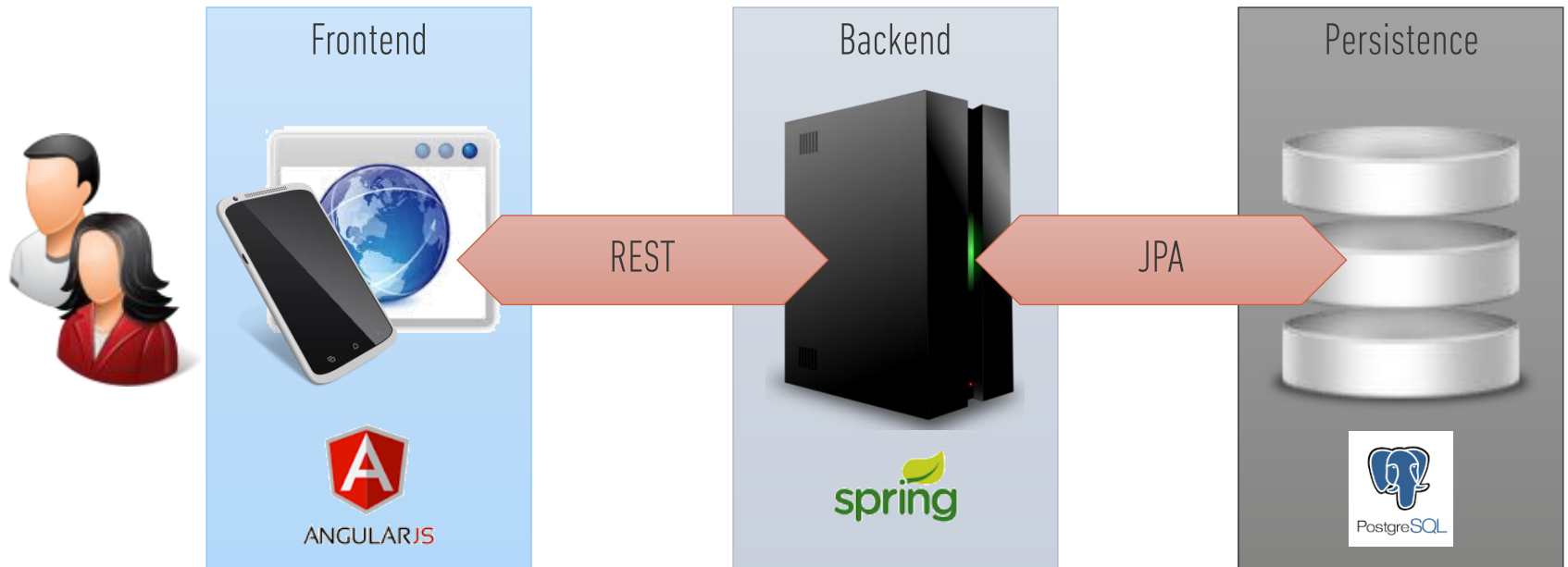
- Swagger-UI, HATEOAS

extendable platform for future requirements

- Spring and Angular ECO-System, flexible RDMS

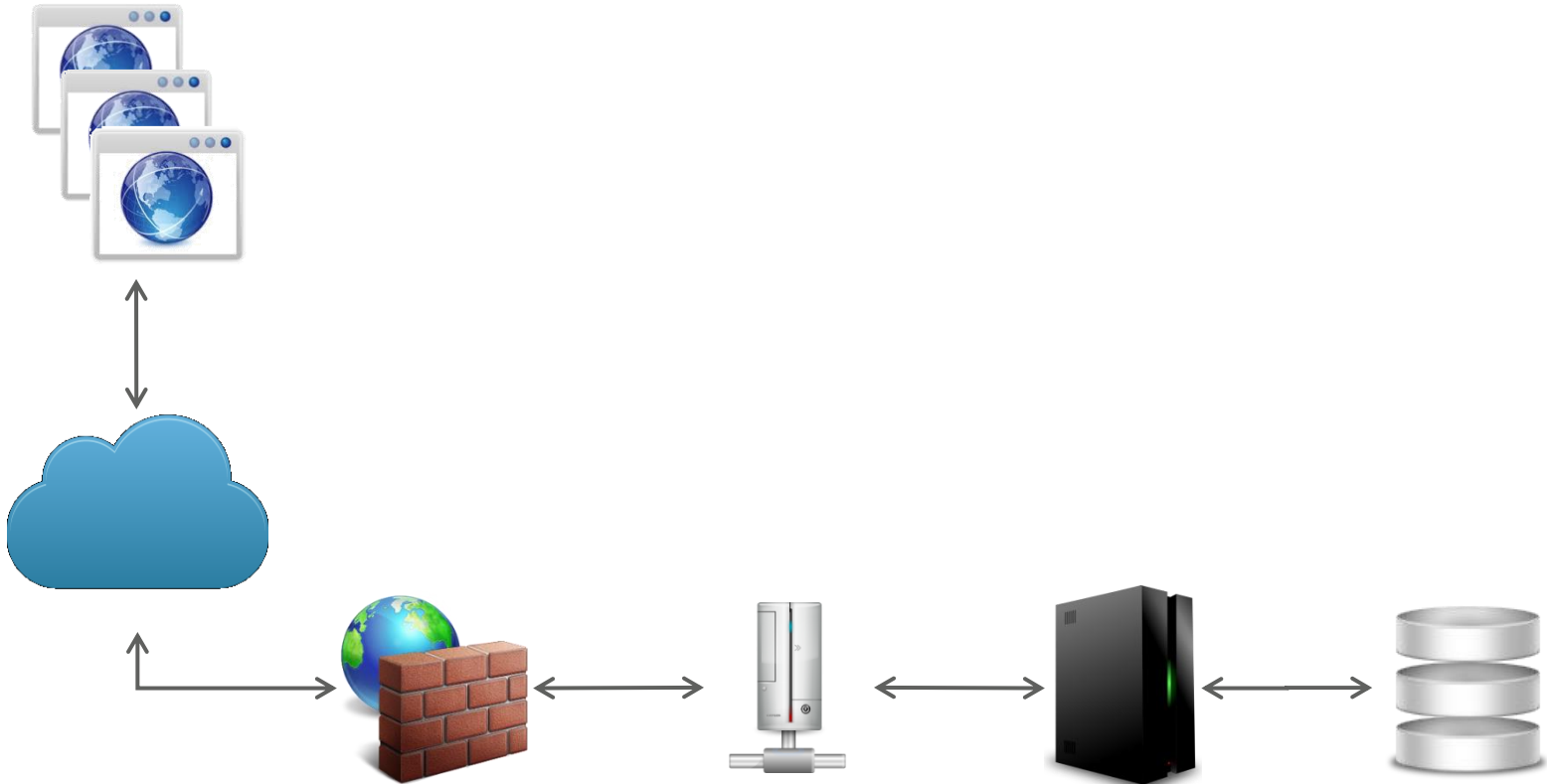
➤ Architecture in detail

Environment



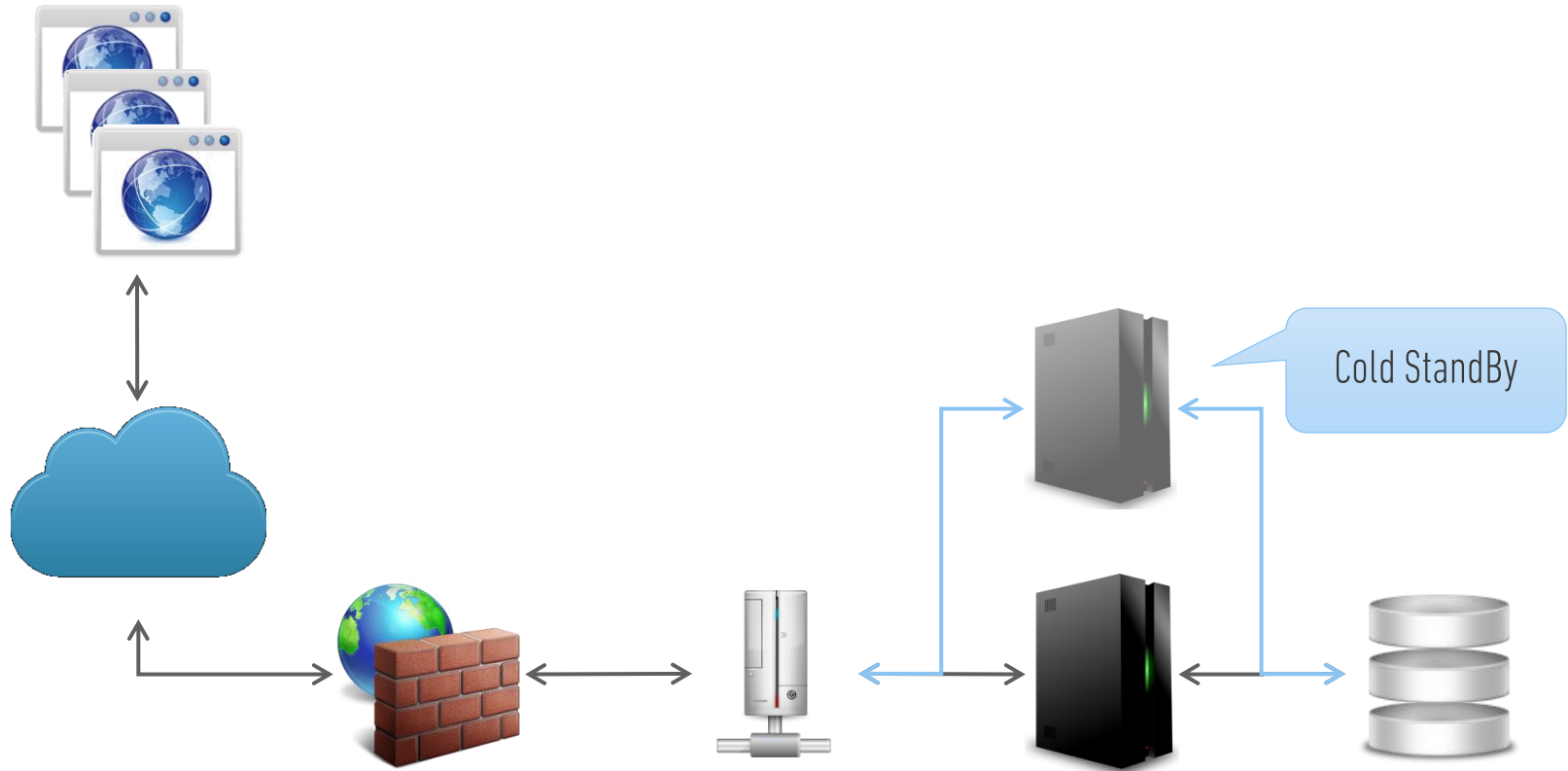
➤ Architecture in detail

Environment Solution One



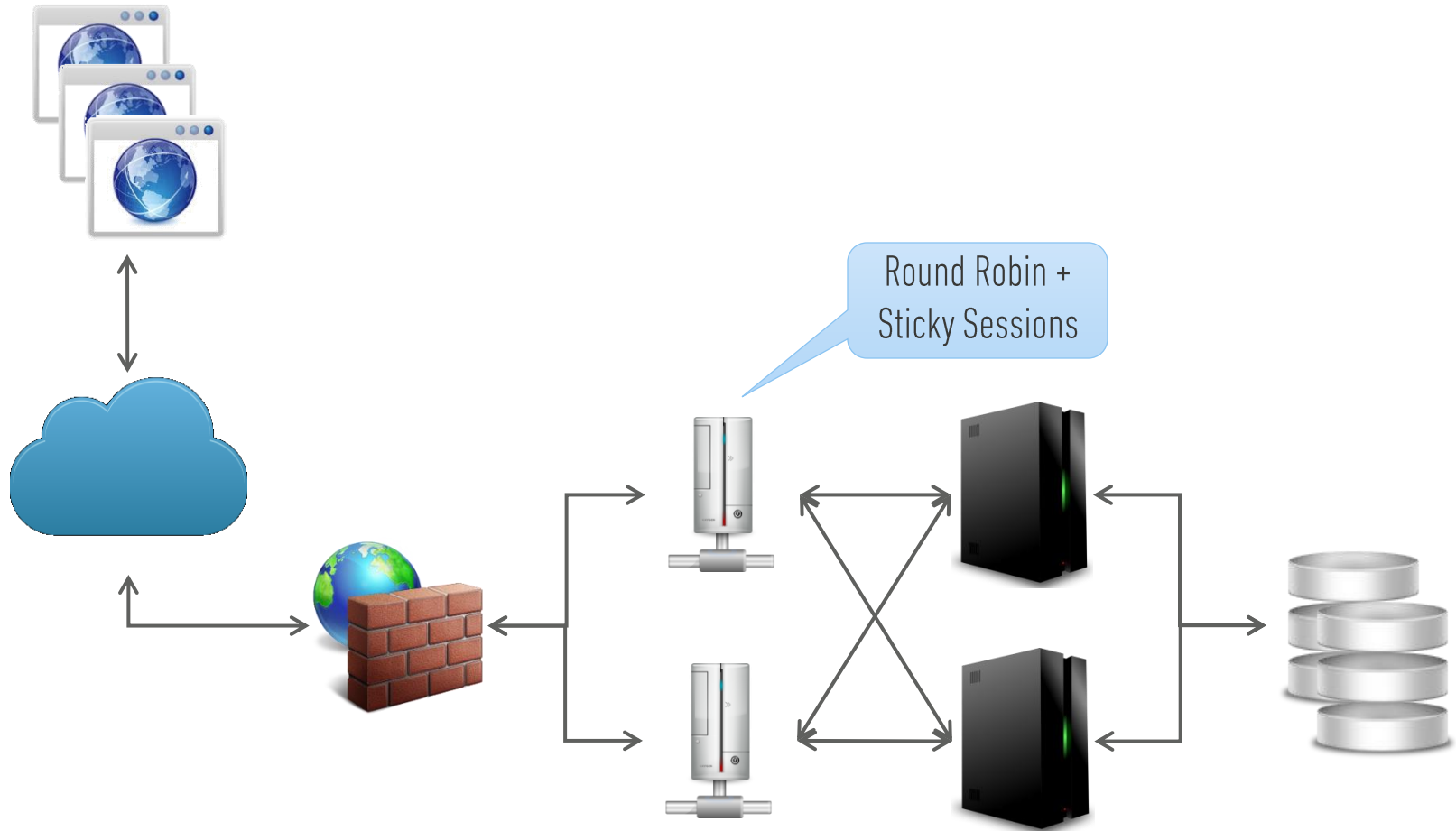
> Architecture in detail

Environment Solution One



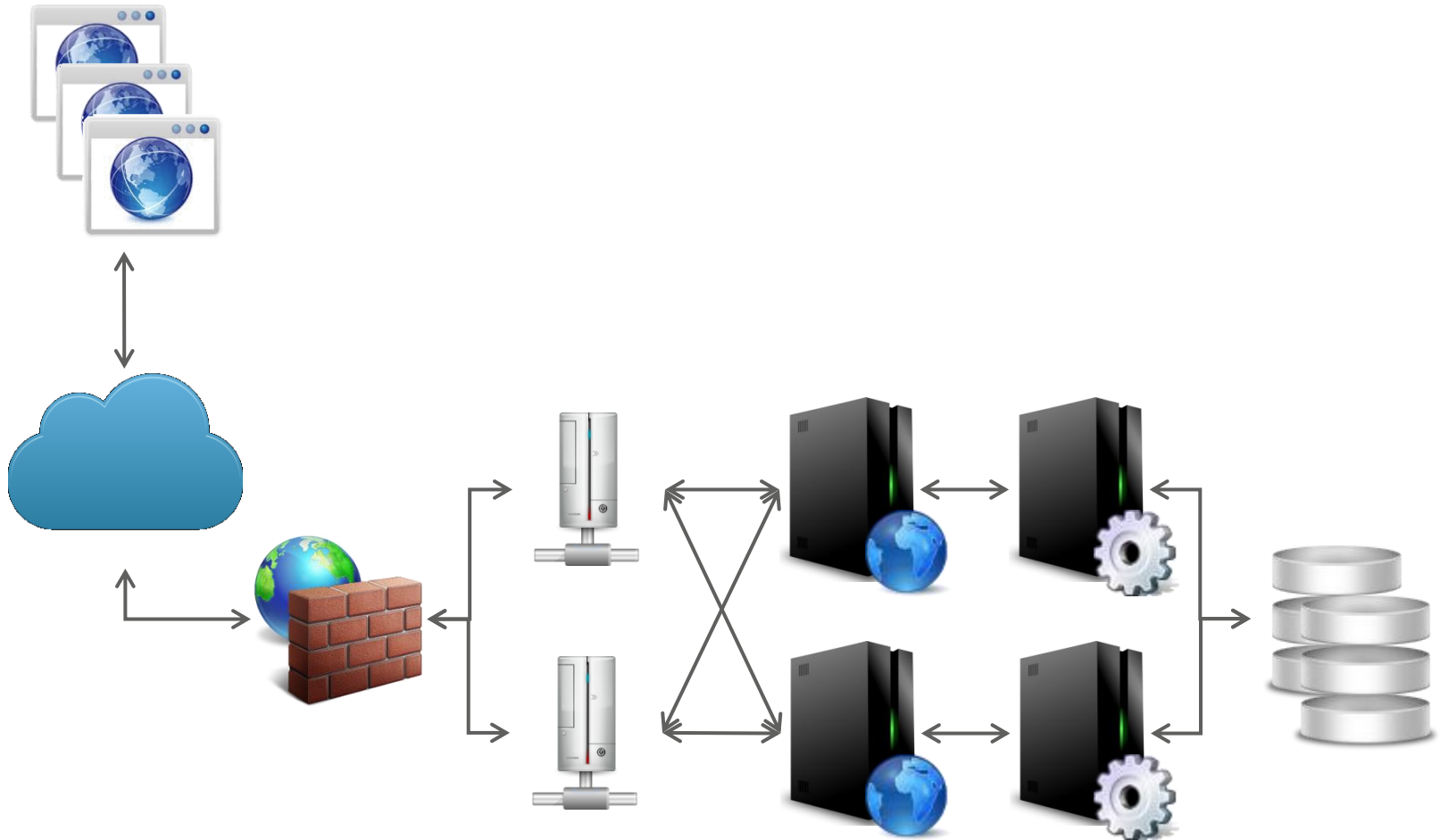
➤ Architecture in detail

Environment Solution Two



> Architecture in detail

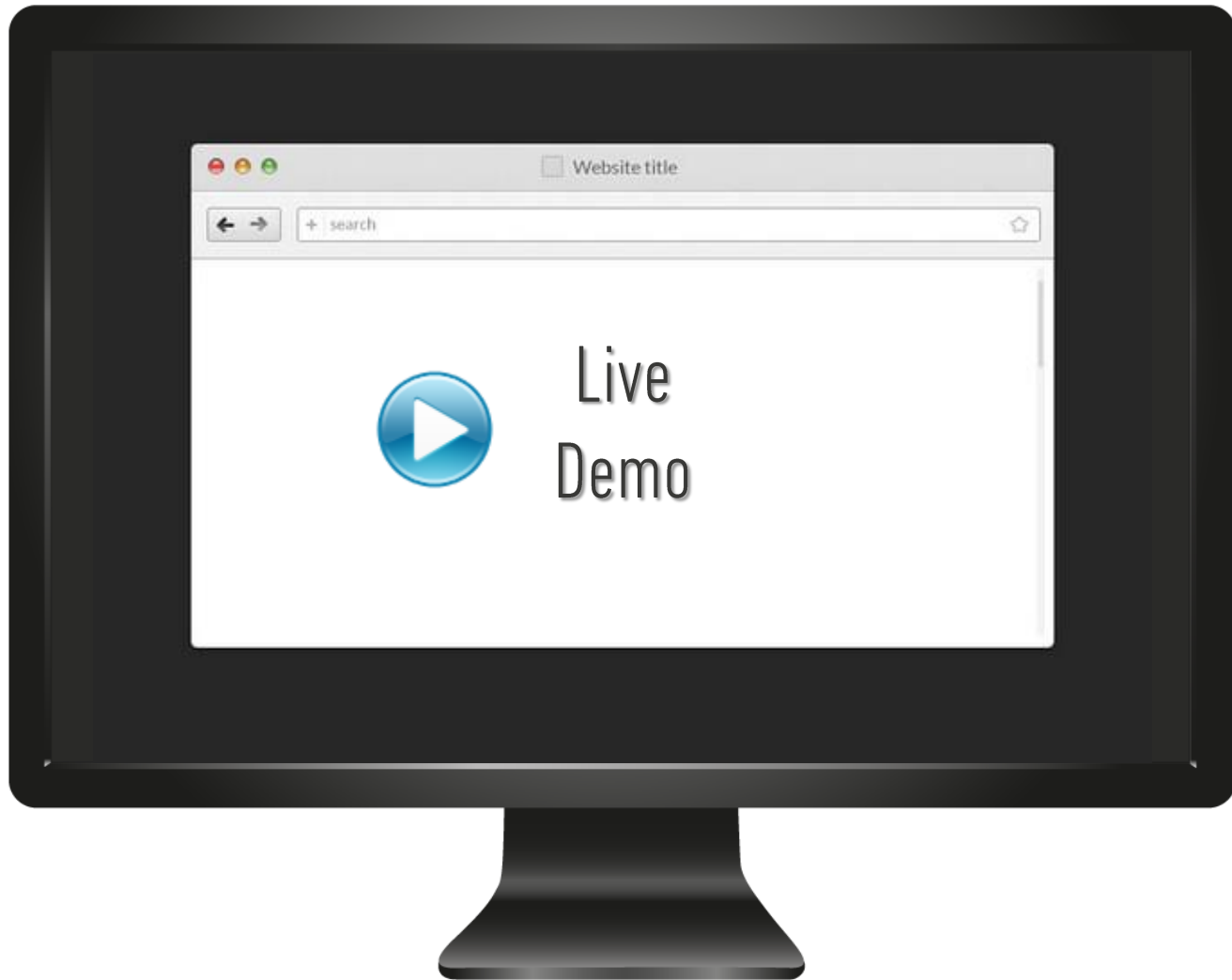
Environment Solution Three



➤ Agenda:

1. Summary of requirements
2. Solution Overview
3. Architecture in detail
4. Live Demo
5. Next Steps

➤ Live Demo



Live Demo

Backend Overview

```
└─ pom.xml
   ├── src/main/java      <- spring boot application code
   ├── src/main/resources <- configuration files
   ├── src/main/test      <- unit tests
   ├── src/main/assembly  <- configuration for the Maven assembly task
   ├── configuration      <- files to setup the (development) environment
   ├── documentation      <- documentation
   └─ target              <- build target
```

```
└─ src/main/java
   └─ com.prodyna.pac.voting
      ├── config          <- configuration files of the application - configured via application.yml
      ├── domain          <- the entities
      ├── exceptions      <- package for custom exceptions
      ├── repository      <- the persistence layer
      ├── security        <- security related handler, services, constants and utilities
      ├── service         <- the business layer
      ├── web             <- the presentation layer
      └─ VotingApplication.java <- the main application
```

Live Demo

Frontend Overview

```
└─ src/main/webapp
   └─ app                                <- the AngularJS application root
      ├── about                         <- the about content
      ├── admin                         <- the administration area
      │   ├── configuration
      │   ├── docs
      │   ├── health
      │   ├── metrics
      │   └── user-management
      ├── components                    <- cross sectional parts
      │   ├── alert
      │   ├── error
      │   ├── login
      │   ├── logout
      │   └── navbar
      ├── config
      ├── home                         <- the application home
      ├── interceptor                  <- app interceptors
      ├── services                     <- services used across the application
      │   ├── auth
      │   ├── profiles
      │   ├── user
      │   └── votes
      ├── votes                        <- the votes area
      └─ bower_components              <- necessary application dependency files - controlled by bower
└─ content                            <- fonts and styling
   ├── css
   └── fonts
└─ swagger-ui                         <- swagger-ui iframe-root
└─ 404.html
└─ index.html                         <- application root html
```

➤ Agenda:

1. Summary of requirements
2. Solution Overview
3. Live Demo
4. Architecture in detail
5. Next Steps

➤ Next Steps

- Javascript Build
- Splitting development artifacts
- UI based tests
- Integrate liquibase
- Complete development environment



Your Personal Contact:

Sven Lissek, PMP

Senior Consultant

sven.lissek@prodyna.com

T +49 69 597724 000 M +49 176 17870 179

PRODYNA AG Ludwig-Erhard-Str. 12-14 65760 Eschborn

prodyna.com