

## European Parliament Online

Riccardo Gobbato - Chiara Maggi Pasquale Mocerino - Simone Scaccia

## Project idea

A system ideally commissioned by the European Union. It empowers citizens from all EU member nations to actively participate and to vote in future referendums.

#### SYSTEM PROPERTIES

- 1. European Referendum proposal
- 2. European-level Consensus
- 3. Submission of a Referendum to Citizens
- 4. National-level Consensus
- 5. National-level Voting
- 6. Result of Referendum



## **Spring Boot**

#### All the services are Spring-Boot project.

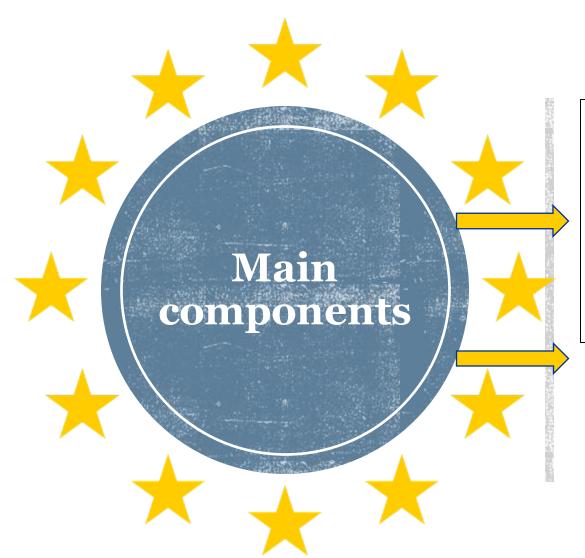
Spring Boot is an open-source framework for building Java applications, designed to simplify development with minimal configuration.

#### Dependecies of the services:

- Web service: Thymeleaf, Spring Boot Starter Web, Spring DevTools, and Spring Boot Starter Test:
- Rest service: Spring Data JPA, Spring Boot Starter Web, Spring DevTools, Spring Boot Starter Test, and PostgreSQL.
- Broadcast service: Spring Boot Starter AMQP, Spring Boot Starter Web, Spring DevTools, Spring Boot Starter Test, Spring Rabbit Test, and Google Gson.



- We have implemented a **docker-compose for each nation**, plus a docker-compose for RabbitMQ.
- The **RabbitMQ Management Container** allows to manage RabbitMQ through a web interface. The container is connected to 3 custom networks, which we have used to facilitate communication between RabbitMQ and other containers in different networks.
- For each service (in every nation) there is a docker file to allow the creation of the jar file and the container file and the container. There are two main parts:
  - Build Stage: runs "mvn clean package" to build the Java application using Maven.
  - Package Stage: sets the entry point for the Docker container to run the Java application using the JAR file ('app-1.0.0.jar'). This
    Dockerfile separates the build process from the final runtime image, resulting in a smaller runtime image that only contains the
    necessary JRE and the application JAR file.



The deployed components are grouped into services for specific nations and those that are available to all nations.

#### **NATIONAL SERVICES**

**Web:** implements the web interface, providing the main functionalities to the users

**Broadcast**: implements the broadcast and the consensus primitives

**Rest:** provide CRUD API to manage resource on the database

**PostgreSQL**: implements the persistence of the nation data

#### STANDALONE SERVICES

**RabbitMQ broker:** provide the exchange of messages between nations by implementing a queue system.

#### Bindings

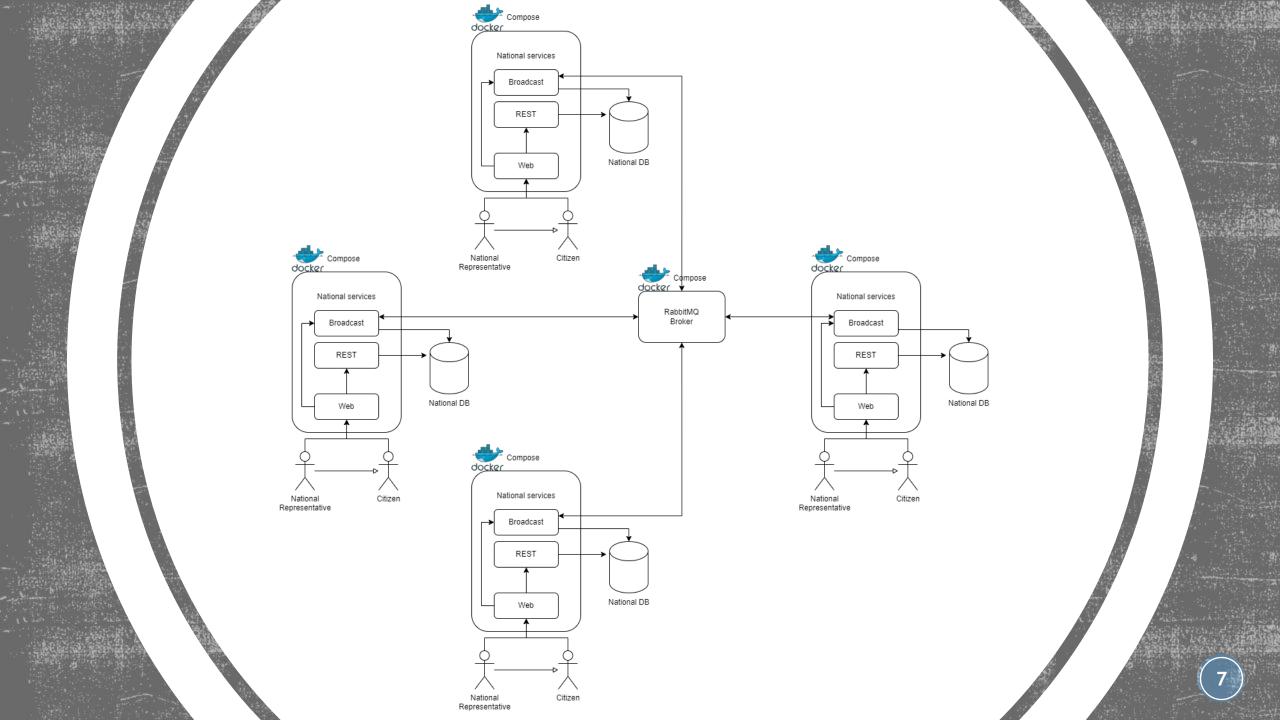
This exchange



То	Routing key	Arguments	
fra	foo.bar.#		Unbind
ger	foo.bar.#		Unbind
ita	foo.bar.#		Unbind

## Rabbit MQ

- This node is needed for making asynchronous communication possible between the servers in charge of dispatching the messages between the organization workers.
- Each nation establishes its own queue on this broker and designates a routing key.
- When a nation desires to send a broadcast message, it selects a routing key that matches all the nations, allowing them to receive the message in their respective queues.
- Once the message has been read, it will be automatically removed from the queue.



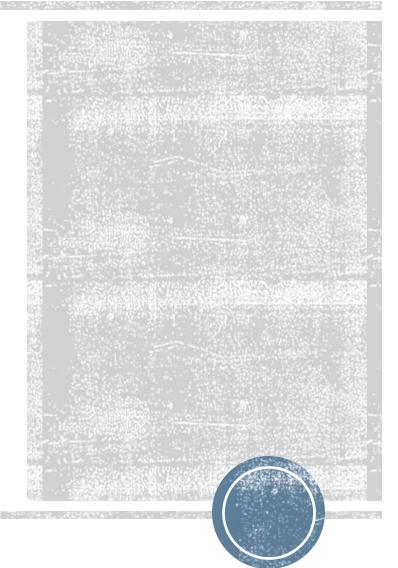
## Consensus Primitive

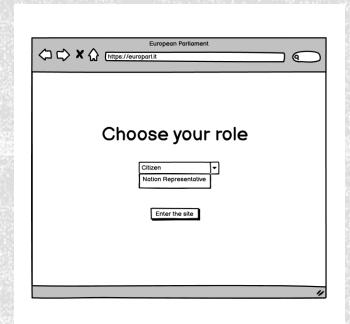
- Consensus A procedure to reach a common agreement in a distributed or decentralized multi-agent platform
- *Timing Assumptions*: Asynchronous communication is handled by incorporating strategic timeouts, introducing partial synchrony to allow consensus implementation.
- Clock Synchronization: External Synchronization using frameworks like Spring Boot and UTC to ensure consistent timekeeping.
- Link Abstractions: Reliable message delivery is ensured using RabbitMQ (perfect point-to-point communication).
- Flooding Consensus with a single RabbitMQ Broker:
  - Using 1 Broker for the broadcast communication, Uniform Reliable Broadcast is ensured.

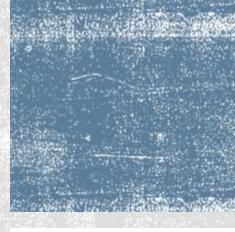
    URB ensures uniform agreement on the values decided, and even in the event of crashes, it still only requires n² messages (squared) instead of n³ (cubed). This makes it a more efficient option for achieving consensus.



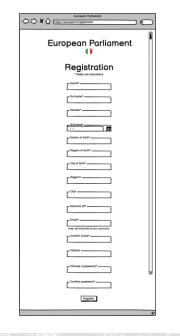
# USER STORIES





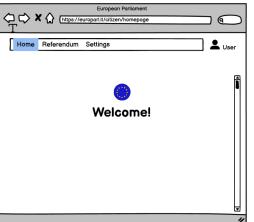


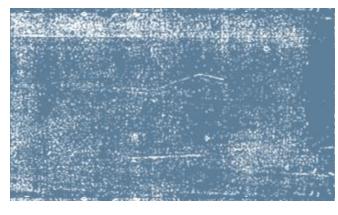




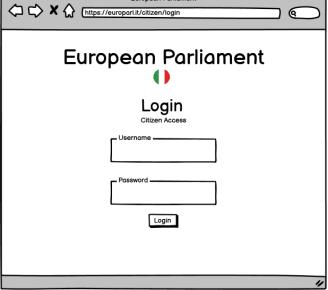
- As a user, I want to select the citizen or the national representative role, so that I can enter and navigate in the page hosted by that nation.
- As a citizen, I want to register myself in the page hosted by the nation related to my legal citizenship, so that I can authenticate myself in the future

- As a citizen, I want to authenticate myself in the the page hosted by the nation related to my legal citizenship, so that I can access the national service
- As a citizen, I want to have a personal area displaying the possible services, so that I can choose a service (referendum and information)

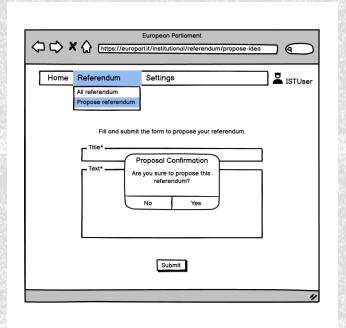


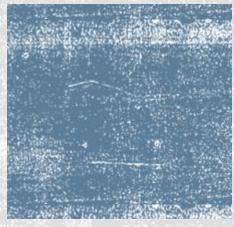










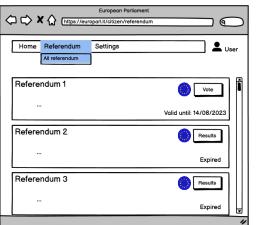


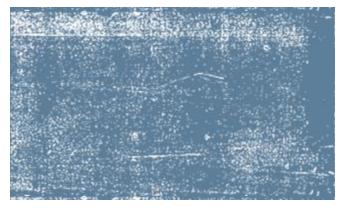




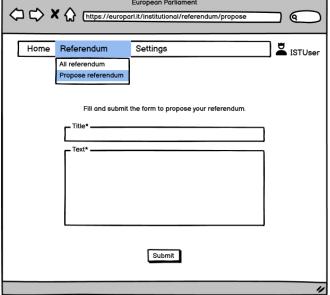
- As a nation representative, I want to propose an idea for an European Referendum to other European nations, so that it can be proposed to all European citizens.
- As a nation representative, I want to authe nticate myself in the context of my nation, so that I can manage the national service.

- As a nation representative, I want to declare a new national referendum, constituted by one question with "Yes" or "No" answer, so that I can obtain votes from citizens of my nation
- As a citizen, I want to see the list of all referendums declared by my nation, so that I can vote.

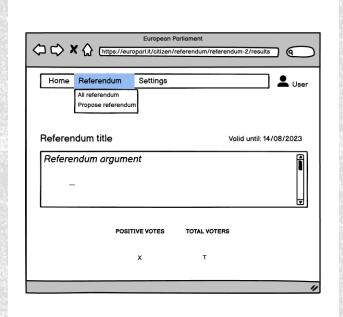


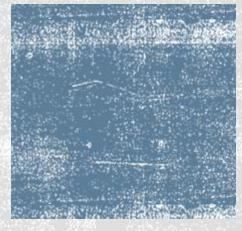


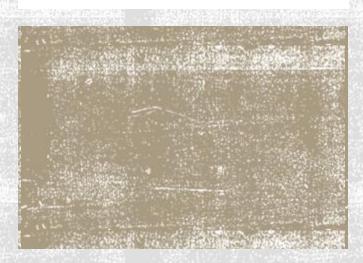


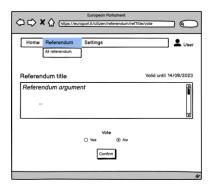










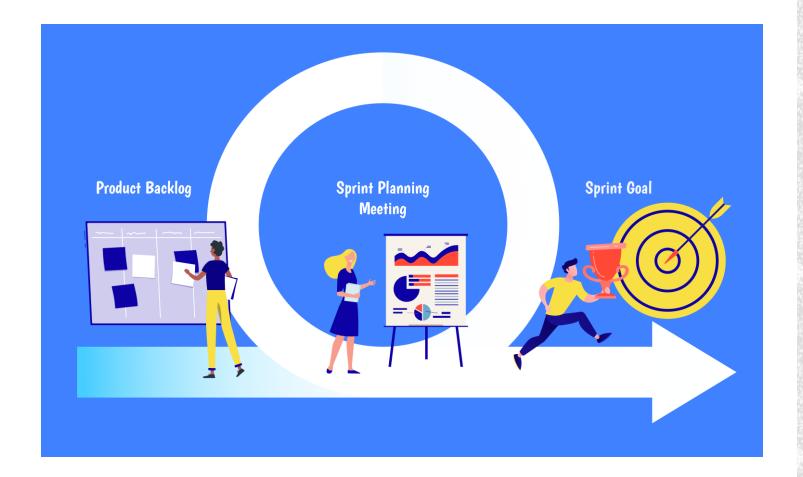


- As a nation representative, I want to see the final result of the voting process related to an European referendum, so that I can apply it if it has been approved by European citizens
- As a citizen, I want to see the results of a referendum declared by my nation, so that I can know the outcome
- As a citizen, I want to vote in a referendum declared by my nation, so that I can express my opinion



## WORKFLOW ORGANIZATION





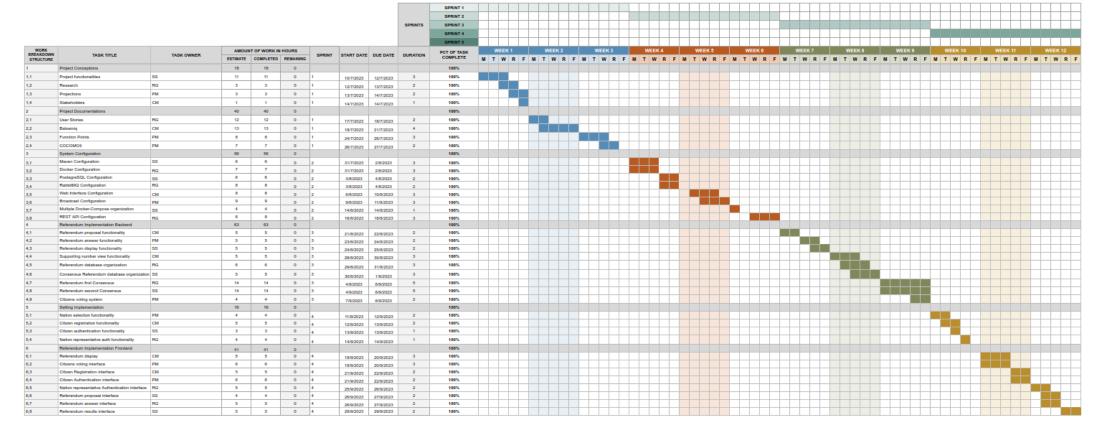
#### SPRINT PLANNING

- Create a product backlog, which included a list of features and requirements
- Prioritize the items in the backlog, based on their value to the end-users and the organization.
- Select a set of items from the product backlog and created a sprint backlog, which included a list of tasks and objectives for the sprint.

#### SCRUM PROJECT MANAGEMENT GANTT CHART

**√** smartsheet

**GANTT CHART AND BURNDOWN** 



**BURNDOWN DATA** 

ESTIMATE COMPLETED REMAINING DAYS TOTAL HOURS 234 234 0 60 3,9 A Enter number of days

Enter hours completed per day ---->

		DAY	1	2	3	4	5	6	7	8	9	10	11	1:	2 1	3 14	1	5 1	6 1	7 1	8 1	9 2	0 2	1 2	2 2	23	24 2	25 2	26	27 2	8 2	19 3	0 3	1 3	2 3	3 3	4 3	5 30	5 3	7 3	3 39	9 4	0 4	1 4	2 4	3 44	4	5 4	5 4	7 4	3 4	9 5	0 5	1 5	2 5	3 54	50	56	57	58	59	6
		PLAN	234	230	226	222	2115	215	211	207	203	199	195	12	1 13	15	3 17	9 11	6 17	2 16	15 11	94 19	10 1	15 1	52 1	145	44 1	40 1	37	133 1	29 1	25 1	21 1	7 1	13 10	9 10	15 10	21 93	9	4 20	0 56	5 5	7	5 7	4 71	0 66	60	2 5	5 5	5 5	4	7 4	3 3	3	3	27	22	19	16	12	8	4
	EST	IMATE	234	229	224	21	218	216	210	203	199	194	191	15	7 15	4 15	17	E 17	6 17	2 16	15 11	53 1	95 1	9 1	49 1	145	41 1	35 1	32	128 1	28 1	25 1	22 1	10 1	15 11	5 11	2 10	15 10	5 10	3 9	90	5 9	0 8	9 5	2 7	70	68	5 5	7 5	4	42	5 4	1 4	1 3	7 3	3	25	19	15	11	5	
٠.>	HRS COMPL	LETED	5	5	3	3	2	6	7	4	5	3	4	3	4	4	0	4	4	5	1	7	7 (	1	3	5	6	3	4	0	3	3	2	1	3 2	4	2	3 2	5	5	3	1	7	7		5		3		4	4		4	4	2	5	6	3	5	6	4	1
	HRS REMA	NINING	229	224	221	115	216	210	203	199	194	191	157	15	4 13	17	5 17	8 11	2 16	5 16	3 1	95 1	0 1	9 1	45 1	141 1	35 1	32 1	28	125 1	25 1	22 1	20 1	8 1	5 11	2 10	15 10	25 10	3 2	5 20	90	0 8	5	2 7	5 7	65	57	7 5	4 4	4	4	1 4	1 3	7 3	3 3	23	19	16	11	5		0

TOTAL HOURS

7163 234 6929

# PRODUCT BACKLOG



PRIORITY	SPRINT	FUNCTIONALITY	TASK TITLE	TASK DESCRIPTION	TASK OWNER	WORK ESTIMATE IN HOURS	STATUS	
High	1	Project Conceptions	Project functionalities	Building the idea of the project	SS	11 🔻	Completed	+
Medium	1	Project Conceptions	Research	Doing research of the tools	RG	3 -	Completed	-
Low	1	Project Conceptions	Projections	Define the projections and the workflow	PM	3 -	Completed	-
Low	1	Project Conceptions	Stakeholders	Define the stakeholders of the project	CM	1 -	Completed	-
High	1	Project Documentations	User Stories	Define the user stories of the system	RG	12 🔻	Completed	-
High	1	Project Documentations	Balsamiq	Build the project from scratch	СМ	13 🔻	Completed	-
High	1	Project Documentations	Function Points	Fix the function points	PM	8 -	Completed	-
High	1	Project Documentations	COCOMOII	Use of COCOMO Method	PM	7 -	Completed	-
Medium	2	System Configuration	Maven Configuration	Setup the Maven deployments	SS	6 -	Completed	-
High	2	System Configuration	Docker Configuration	Setup the JAVA User API server as a docker image	RG	7 -	Completed	-
High	2	System Configuration	PostegreSQL Configuration	Setup and configure the PostegreSQL as a Docker image	SS	6 -	Completed	-
High	2	System Configuration	RabbitMQ Configuration	Setup and configure the RabbitMQ server as a Docker image	RG	8 🔻	Completed	+
High	2	System Configuration	Web Interface Configuration	Create the Web Interface Configurationand create basic preliminary GUI	CM	8 -	Completed	-
High	2	System Configuration	Broadcast Configuration	Configurate the Broadcast Service to allow the exchange of messages between different Nations	PM	9 -	Completed	-
Medium	2	System Configuration	Multiple Docker-Compose organization	Setup and organise the different Docker-Compose	SS	4 -	Completed	-
High	2	System Configuration	REST API Configuration	Setup the REST API Configuration	RG	8 -	Completed	-
Medium	3	Referendum Implementation Backend	Referendum proposal functionality	Create functionality to propose referendum	CM	5 ▼	Completed	-
Medium	3	Referendum Implementation Backend	Referendum answer functionality	Create functionality to answer referendum	PM	5 *	Completed	-
Medium	3	Referendum Implementation Backend	Referendum display functionality	Create functionality to display referendum on page	SS	5 ▼	Completed	-
Medium	3	Referendum Implementation Backend	Supporting number view functionality	Create functionality to display info about the referendum (number of votes)	CM	5 ▼	Completed	-
High	3	Referendum Implementation Backend	Referendum database organization	Organise the object Referendum in the database	RG	6 -	Completed	-
High	3	Referendum Implementation Backend	Consensus Referendum database organization	Organise the object Consensus Referendum in the database	SS	5 -	Completed	-
High	3	Referendum Implementation Backend	Referendum first Consensus	Build the Consensus primitive to decide if the nations want to do the Referendum	RG	14 🔻	Completed	-
High	3	Referendum Implementation Backend	Referendum second Consensus	Build the Consensus primitive to decide if the citizens agree with the Referendum	SS	14 🔻	Completed	-
Medium	3	Referendum Implementation Backend	Citizens voting system	Build the primitive to allow the citizens to vote	PM	4 -	Completed	-
Low	4	Setting Implementation	Nation selection functionality	Create functionality to allow selection of the Nation	PM	4 +	Completed	-
Low	4	Setting Implementation	·	Create functionality to allow registration of the user	CM	5 *	Completed	-
Low	4	Setting Implementation	Citizen registration functionality  Citizen authentication functionality	Create functionality to allow authentication and access of the user	SS	3 *	Completed	-
Low	4	Setting Implementation	Nation representative auth functionality	Create functionality to allow authentication and access of the nation	RG	4 +	Completed	-
Medium	4	Referendum Implementation Frontend	Referendum display	Build interface for the display of the Referendum	CM	5 -	Completed	-
Medium	4	Referendum Implementation Frontend	Citizens voting interface	Build interface for the voting of the citizens	PM	6 -	Completed	-
Low	4	Referendum Implementation Frontend	Citizen Registration interface	Build interface for the Citizen registration	CM	5 ▼	Completed	-
Low	4	Referendum Implementation Frontend	Citizen Authentication interface	Build interface for the Citizen authentication	PM	6 -	Completed	-
Low	4	Referendum Implementation Frontend	Nation representative Authentication interface	Build interface for the Nation representative authentication	RG	5 ▼	Completed	-
Medium	4	Referendum Implementation Frontend	Referendum proposal interface	Build interface for the proposal of the Referendum	SS	4 -	Completed	-
Medium	4	Referendum Implementation Frontend	Referendum answer interface	Build interface for the answer of the Referendum	RG	5 ▼	Completed	-
Medium	4	Referendum Implementation Frontend	Referendum results interface	Build interface for the results of the Referendum	SS	5 ▼	Completed	-

# FUNCTION POINTS



## FUNCTION POINTS INTERACTION WITH THE USER

#### Citizen web interface

- A citizen can login on the interface by submitting username and password, so we have an EI with 1 FTR and 2 DET:
   0-1 FTR and 1-4 DET EI => 3 FP
- A citizen can register by submitting all citizen fields, so we have an EI with 1 FTR and 14 DET:
   0-1 FTR and 5-15 DET EI => 3 FP
- A citizen can get the referendum list, with all fields for each referendum, so we have an EQ with 1 FTR and 13\*(number of referendum) DET EQ:

0-1 FTR and 20+ DET EQ => 4 FP

• A citizen can access to the voting page of a single referendum, getting the details of a single referendum and submitting the vote to that specific referendum:

EQ with 1 FTR and 13 DET. 0-1 FTR and 6-19 DET EQ => 3 FP EI with 1 FTR and 1 DET. 0-1 FTR and 1-4 DET EI => 3 FP

### FUNCTION POINTS INTERACTION WITH THE USER

#### Representative web interface

- A national representative can login on the inteface by submitting username and password, so we have an EI with 1 FTR and 2 DET:
   0-1 FTR and 1-4 DET EI => 3 FP
- A national representative can publish a referendum by submitting title and text, so we have an EI with 1 FTR and 2 DET:
   0-1 FTR and 1-4 DET EI => 3 FP
- Finally, a national representative get the referendum list, with all fields for each referendum, so we have an EQ with 1 FTR and 13\*(number of referendum) DET EQ:

0-1 FTR and 20+ DET EQ => 4 FP

Hence, the cost of the interaction with the user is the following one:

Cost(User) = Cost(Citizen web interface) + Cost(Representative web interface) = (16 FP) + (10 FP) = 26 FP

## FUNCTION POINTS INTERACTION WITH RABBITMQ

- The application receives a referendum proposal from RabbitMQ Broker, so we have an EI with 1 FTR and 13 DET.
   0-1 FTR and 5-15 DET EI => 3 FP
- When proposing a referendum, the application sends a referendum proposal to other nations, so we have an EQ with 1 FTR and 13 DET.
   0-1 FTR and 6-19 DET EQ => 3 FP
- When receiving a referendum proposal, the application sends a response for the first consensus, so we have an EO with 1 FTR and 8 DET.
   0-1 FTR and 6-19 DET EO => 4 FP
- The application receives a response during first consensus execution, so we have an EI with 1 FTR and 8 DET.
   0-1 FTR and 5-15 DET EI => 3 FP
- After receiving a referendum result from the voting process of citizens, the application sends this result to other nations, so we have an EO with 1 FTR and 8 DET.
  - 0-1 FTR and 6-19 DET EO => 4 FP
- The application receives a referendum result from another nation, so we have an EI with 1 FTR and 8 DET.
   0-1 FTR and 5-15 DET EI => 3 FP

Cost(Other applications) = 20 FP

## FUNCTION POINTS TOTAL COST

CitizenUser table has 13 mandatory fields and 1 optional field, InstUser table has 13 mandatory fields (representative ID instead of national ID) and 1 optional field (cellular), Referendum table has 13 fields and ReferendumConsensus table has 8 fields. So, we can consider the DB as a single ILF with 4 RET of 1-19 DET each one, since it can be accessed through the REST API internal component. For this ILF, we have 7 FP. This is the contribution related to the ILF of our application.

Cost(DB) = 7 FP

- Total cost = Cost(DB) + Cost(User) + Cost(Other applications) = (7 FP) + (26 FP) + (20 FP) = 53 FP
- HTML/JS: Cost(User) = 26 FP
- JAVA: Cost(DB) + Cost(Other applications) = 27 FP

## **COCOMO II**

_	Scaling Fac	ctors			
	SF	Description	Level	Value	Instructions/Comments
	Maturity	Process Maturity	Low	6.24	Recommended values for CMMi level 2 companies is Nominal. For Level 3 its High
	PREC	Experience of similar Projects	Low	4.96	For Projects of new domain (that is domain we have not worked on) always select low. Select Nominal or High for projects that we are creating from scratch but we have some knowledge about domain. Select very high or extremely high for Next Releases of existing projects
	FLEX	Flexibility required in the System	Low	1.01	We usually use Nominal but check comments on description for details
	TEAM	Team Cohesiveness	Very High	1.1	Team Cohesiveness is always extremely high in projects where team is small and not distributed across the globe
	RESL	Project Risk and Architectural Complexity	High	4.24	We usually use Nominal but check comments on description for details
_	Effort Multi	inlier EM			
	EM	Description	Level	Value	Instructions/Comments
	CIVI	•	Level	value	ilisti uctions/comments
	RCPX	System reliability, complexity and size indicator	Nominal	1	The kind of projects we have in Technosoft we normally use extra low or very low. Check comments on description for details
	RUSE PDIF	Reusability concern with respect to current and future projects Platform Difficulty	Very Low Nominal		We usually use Low but check comments on description for details We usually use Very Low but check comments on description for details
	PERS	Personal capability of team. Like technical capability of Programmers, Designers and testers.	High		We usually use High but check comments on description for details
	PREX	Application, Language and tool experience	l High	0.87	We usually use High but check comments on description for details
	FOIL	Using Case tools for development etc	Nominal	1	We usually use High for .NET and Nominal for java but check comments on description for details
	FCIL	development etc	Homman		

### **COCOMO II**

HTML/JS: 26 FP



1222 SLOC

JAVA: 27 FP



1431 SLOC

Assumptions				
Test Planning, Documentation an are carried out in parallel to other development and testing				
Resources				
Designers	1	Project Plan - Waterfa	all .	
Developers	2	Phase	Duration (days)	Duration (months)
Testers	1	Requirements/Design Phase	31.2	1 Month 6 Day
Others		Development Phase	40.0	1 Month 17 Day
Comunication Delay in Analysis and Design	20%	Testing Phase	22.2	0 Month 26 Day
Possible Critical Chain Delay in Development	15%	Total	93.4	3 Month 18 Day
Possible Fixation Delay in Testing	20%	Slack	0.0	
Working days in a month	26	Grand Total	93.4	3 Month 18 Day
Ratio of calendar days	1.19230769231			

## Future improvements

So, a more complete list of future improvements is the following one:

- Security measures implementation (verification and validation)
- Removal of no failure assumption with failure detector implementation
- Removal of SPoF with the use of multiple RabbitMQ brokers
- Implementation of Byzantine tolerant protocols

Another important improvement is to design ad-hoc solutions in order to better cope with some undetachable context-related problems:

- Data partition and heterogeneity
- Unpredictable latencies





## DEMO LIVE

https://github.com/RicGobs/EP O-European-Parliament-Online



# THANK YOU