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 in surveys, expressing their views on critical issues and proposing ideas for future referendums.

SYSTEM PROPERTIES

1. Submission to Citizens
2. Citizen-Proposed Referendums
3. Nation-Level Evaluation
4. Approval Criteria at the National Level
5. Outcome of National Referendums
6. European Referendums
7. Minimum Voting Threshold

Main components

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.

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:

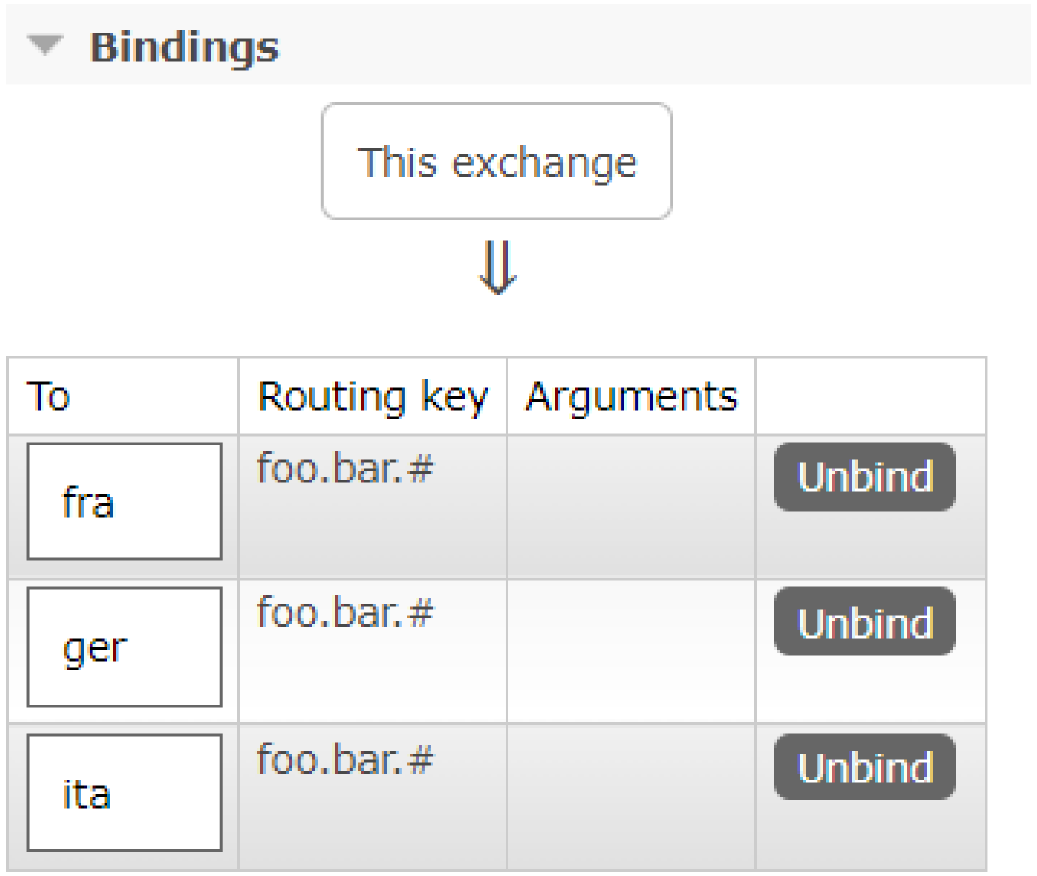
- Web service: Thymeleaf, Spring Boot Starter Web, Spring DevTools, and Spring Boot Starter Test:
- Rest service: Spring Data JPA, Spring Boot Starter Web, Spring DevTools, Spring Boot Starter Test, and PostgreSQL.
- Broadcast service: Spring Boot Starter AMQP, Spring Boot Starter Web, Spring DevTools,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

Docker

- 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 containe 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.



Bindings

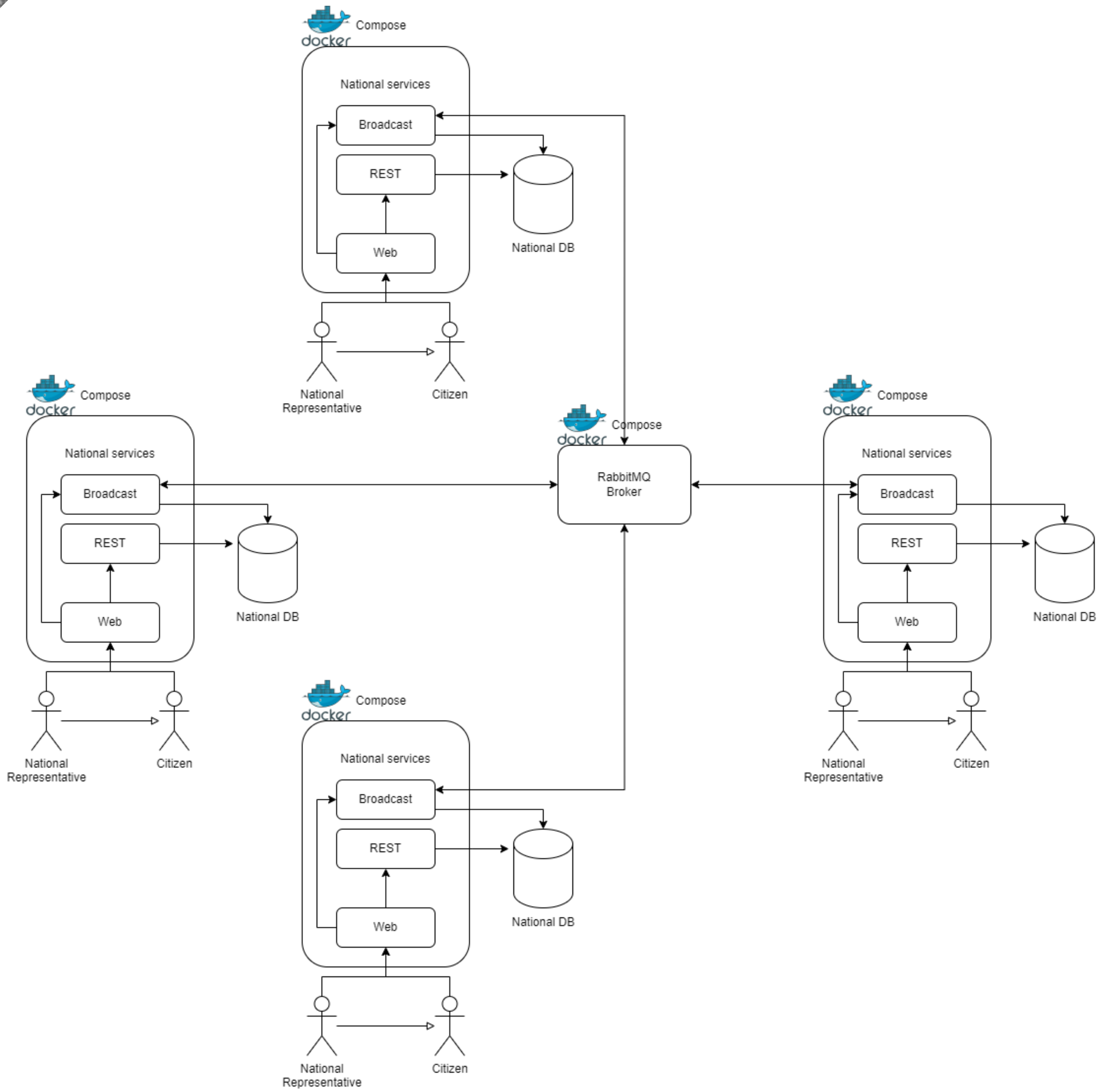
This exchange

⇓

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

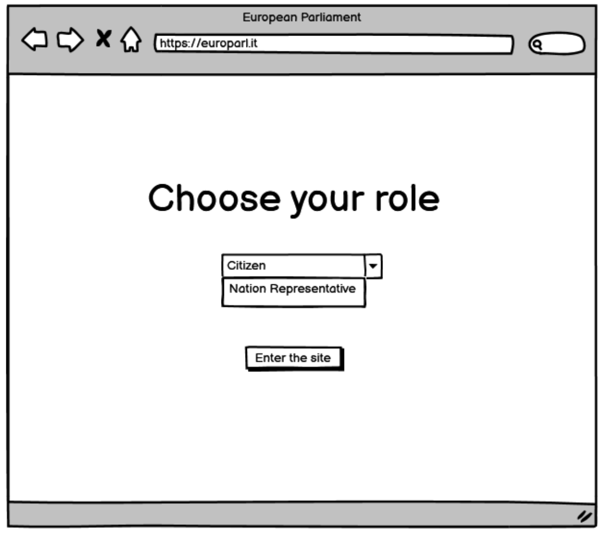
Rabbit MQ

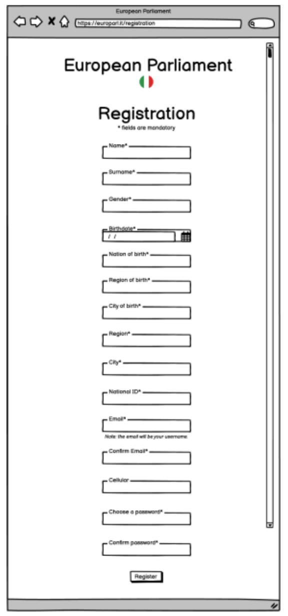
- This node is needed for making asynchronous communication possible between the servers in charge of dispatching the messages between the organization workers.
- Each nation establishes its own queue on this broker and designates a routing key.
- When a nation desires to send a broadcast message, it 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.



docker Compose
National services
Broadcast
REST
Web
National DB
National Representative
Citizen
docker Compose
RabbitMQ Broker

USER STORIES



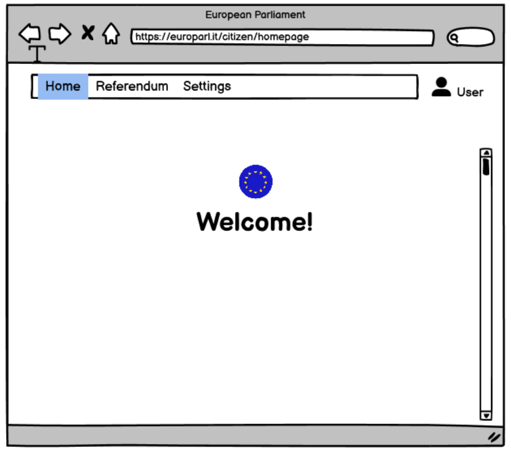


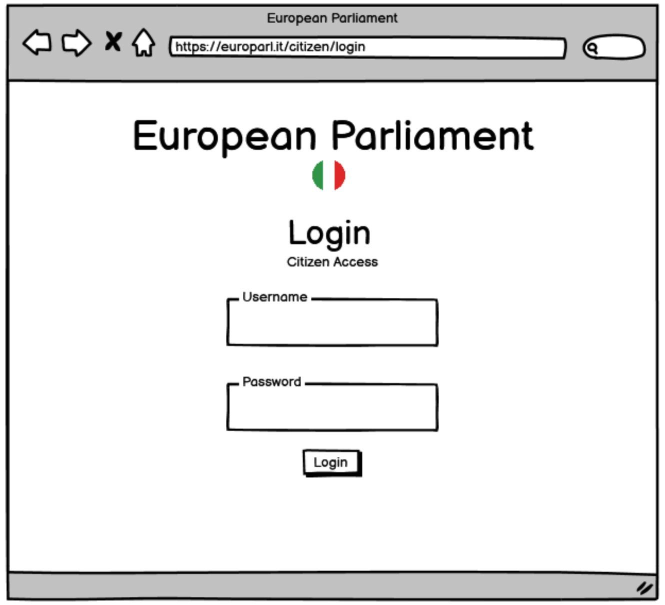
Mockup Balsamiq

- 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

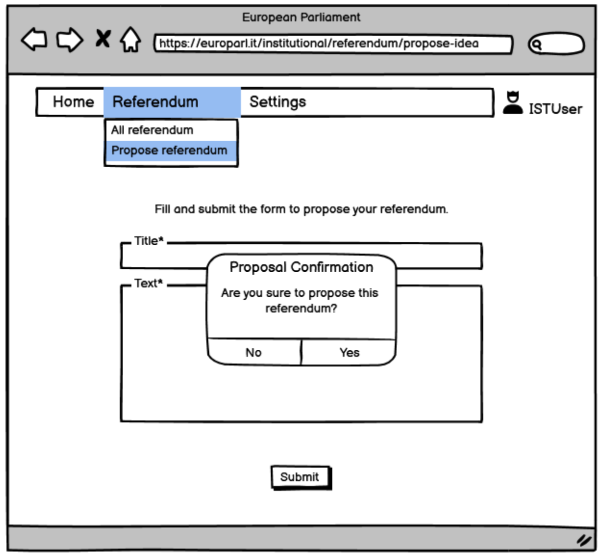
Mockup Balsamiq

- 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)



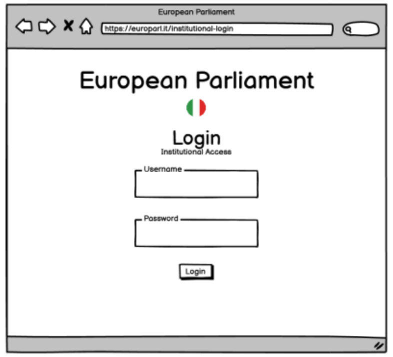


Mockup Balsamiq



Fill and submit the form to propose your referendum.

Are you sure to propose this referendum?

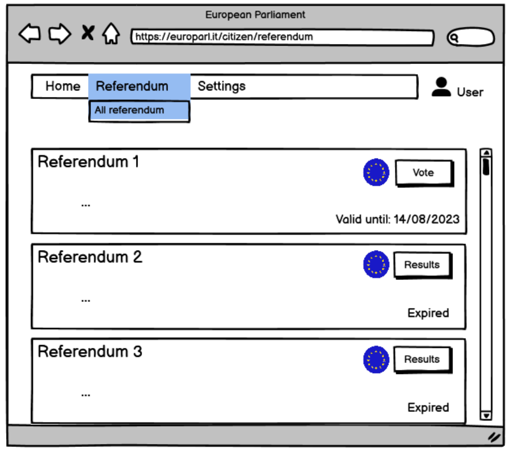


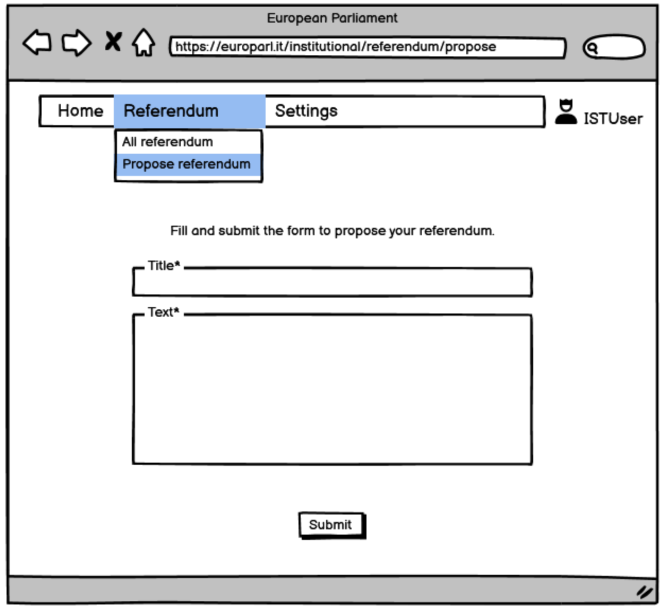
- As a nation representative, I want to propose an idea for an European Referendum to other European nations, so that it can be proposed to all European citizens.
- As a nation representative, I want to authe nticate myself in the context of my nation, so that I can manage the national service.

Mockup Balsamiq

- 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.



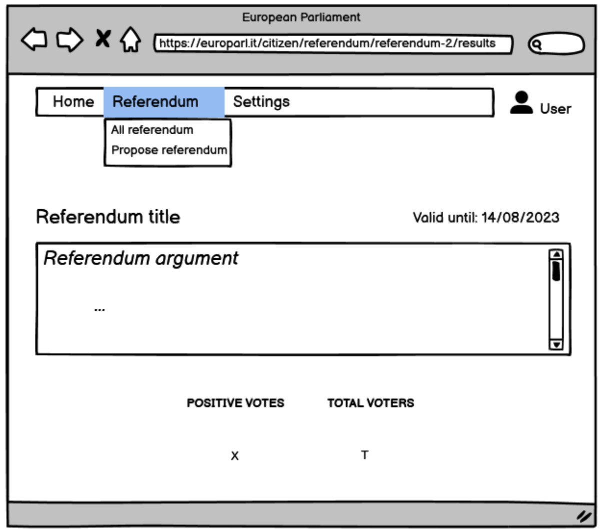


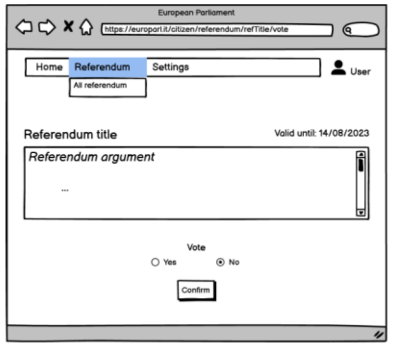
Fill and submit the form to propose your referendum.

Title*

Text*

Mockup Balsamiq





- 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

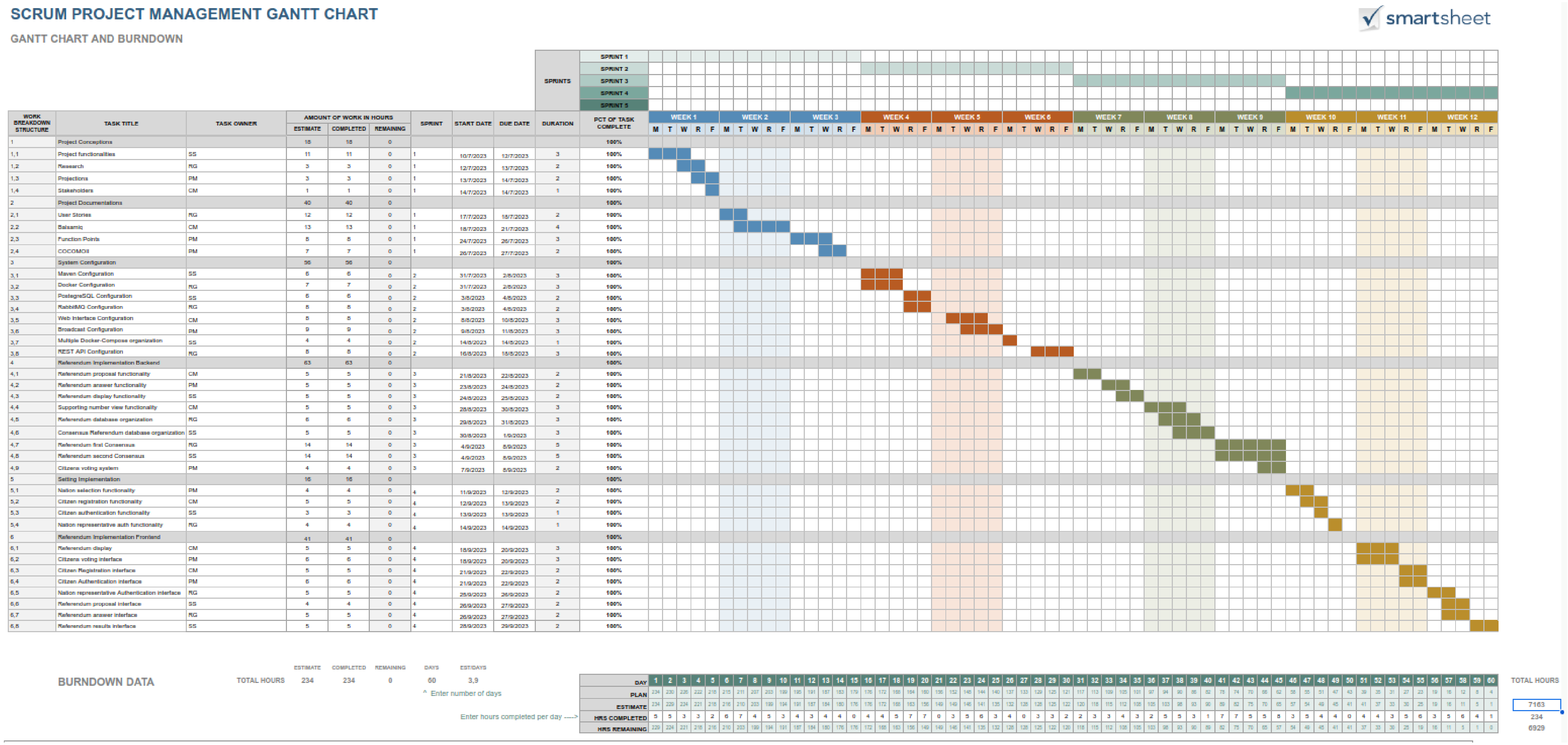
- As a citizen, I want to vote in a referendum declared by my nation, so that I can express my opinion

WORKFLOW ORGANIZATION

Product Backlog
Sprint Planning Meeting
Sprint Goal

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

GANTT CHART AND BURNDOWN

smartsheet

SPRINTS: SPRINT 1, SPRINT 2, SPRINT 3, SPRINT 4, SPRINT 5

WEEK 1, WEEK 2, WEEK 3, WEEK 4, WEEK 5, WEEK 6, WEEK 7, WEEK 8, WEEK 9, WEEK 10, WEEK 11, WEEK 12 (M T W R F)

WORK BREAKDOWN STRUCTURE	TASK TITLE	TASK OWNER	AMOUNT OF WORK IN HOURS			SPRINT	START DATE	DUE DATE	DURATION	PCT OF TASK COMPLETE
			ESTIMATE	COMPLETED	REMAINING					
1	Project Conceptions		18	18	0					100%
1,1	Project functionalities	SS	11	11	0	1	10/7/2023	12/7/2023	3	100%
1,2	Research	RG	3	3	0	1	12/7/2023	13/7/2023	2	100%
1,3	Projections	PM	3	3	0	1	13/7/2023	14/7/2023	2	100%
1,4	Stakeholders	CM	1	1	0	1	14/7/2023	14/7/2023	1	100%
2	Project Documentations		40	40	0					100%
2,1	User Stories	RG	12	12	0	1	17/7/2023	18/7/2023	2	100%
2,2	Balsamiq	CM	13	13	0	1	18/7/2023	21/7/2023	4	100%
2,3	Function Points	PM	8	8	0	1	24/7/2023	26/7/2023	3	100%
2,4	COCOMOII	PM	7	7	0	1	26/7/2023	27/7/2023	2	100%
3	System Configuration		56	56	0					100%
3,1	Maven Configuration	SS	6	6	0	2	31/7/2023	2/8/2023	3	100%
3,2	Docker Configuration	RG	7	7	0	2	31/7/2023	2/8/2023	3	100%
3,3	PostegreSQL Configuration	SS	6	6	0	2	3/8/2023	4/8/2023	2	100%
3,4	RabbitMQ Configuration	RG	8	8	0	2	3/8/2023	4/8/2023	2	100%
3,5	Web Interface Configuration	CM	8	8	0	2	8/8/2023	10/8/2023	3	100%
3,6	Broadcast Configuration	PM	9	9	0	2	9/8/2023	11/8/2023	3	100%
3,7	Multiple Docker-Compose organization	SS	4	4	0	2	14/8/2023	14/8/2023	1	100%
3,8	REST API Configuration	RG	8	8	0	2	16/8/2023	18/8/2023	3	100%
4	Referendum Implementation Backend		63	63	0					100%
4,1	Referendum proposal functionality	CM	5	5	0	3	21/8/2023	22/8/2023	2	100%
4,2	Referendum answer functionality	PM	5	5	0	3	23/8/2023	24/8/2023	2	100%
4,3	Referendum display functionality	SS	5	5	0	3	24/8/2023	25/8/2023	2	100%
4,4	Supporting number view functionality	CM	5	5	0	3	28/8/2023	30/8/2023	3	100%
4,5	Referendum database organization	RG	6	6	0	3	29/8/2023	31/8/2023	3	100%
4,6	Consensus Referendum database organization	SS	5	5	0	3	30/8/2023	1/9/2023	3	100%
4,7	Referendum first Consensus	RG	14	14	0	3	4/9/2023	8/9/2023	5	100%
4,8	Referendum second Consensus	SS	14	14	0	3	4/9/2023	8/9/2023	5	100%
4,9	Citizens voting system	PM	4	4	0	3	7/9/2023	8/9/2023	2	100%
5	Setting Implementation		16	16	0					100%
5,1	Nation selection functionality	PM	4	4	0	4	11/9/2023	12/9/2023	2	100%
5,2	Citizen registration functionality	CM	5	5	0	4	12/9/2023	13/9/2023	2	100%
5,3	Citizen authentication functionality	SS	3	3	0	4	13/9/2023	13/9/2023	1	100%
5,4	Nation representative auth functionality	RG	4	4	0	4	14/9/2023	14/9/2023	1	100%
6	Referendum Implementation Frontend		41	41	0					100%
6,1	Referendum display	CM	5	5	0	4	18/9/2023	20/9/2023	3	100%
6,2	Citizens voting interface	PM	6	6	0	4	18/9/2023	20/9/2023	3	100%
6,3	Citizen Registration interface	CM	5	5	0	4	21/9/2023	22/9/2023	2	100%
6,4	Citizen Authentication interface	PM	6	6	0	4	21/9/2023	22/9/2023	2	100%
6,5	Nation representative Authentication interface	RG	5	5	0	4	25/9/2023	26/9/2023	2	100%
6,6	Referendum proposal interface	SS	4	4	0	4	26/9/2023	27/9/2023	2	100%
6,7	Referendum answer interface	RG	5	5	0	4	26/9/2023	27/9/2023	2	100%
6,8	Referendum results interface	SS	5	5	0	4	28/9/2023	29/9/2023	2	100%

BURNDOWN DATA

	ESTIMATE	COMPLETED	REMAINING	DAYS	EST/DAYS
TOTAL HOURS	234	234	0	60	3,9

^ Enter number of days

Enter hours completed per day ---->

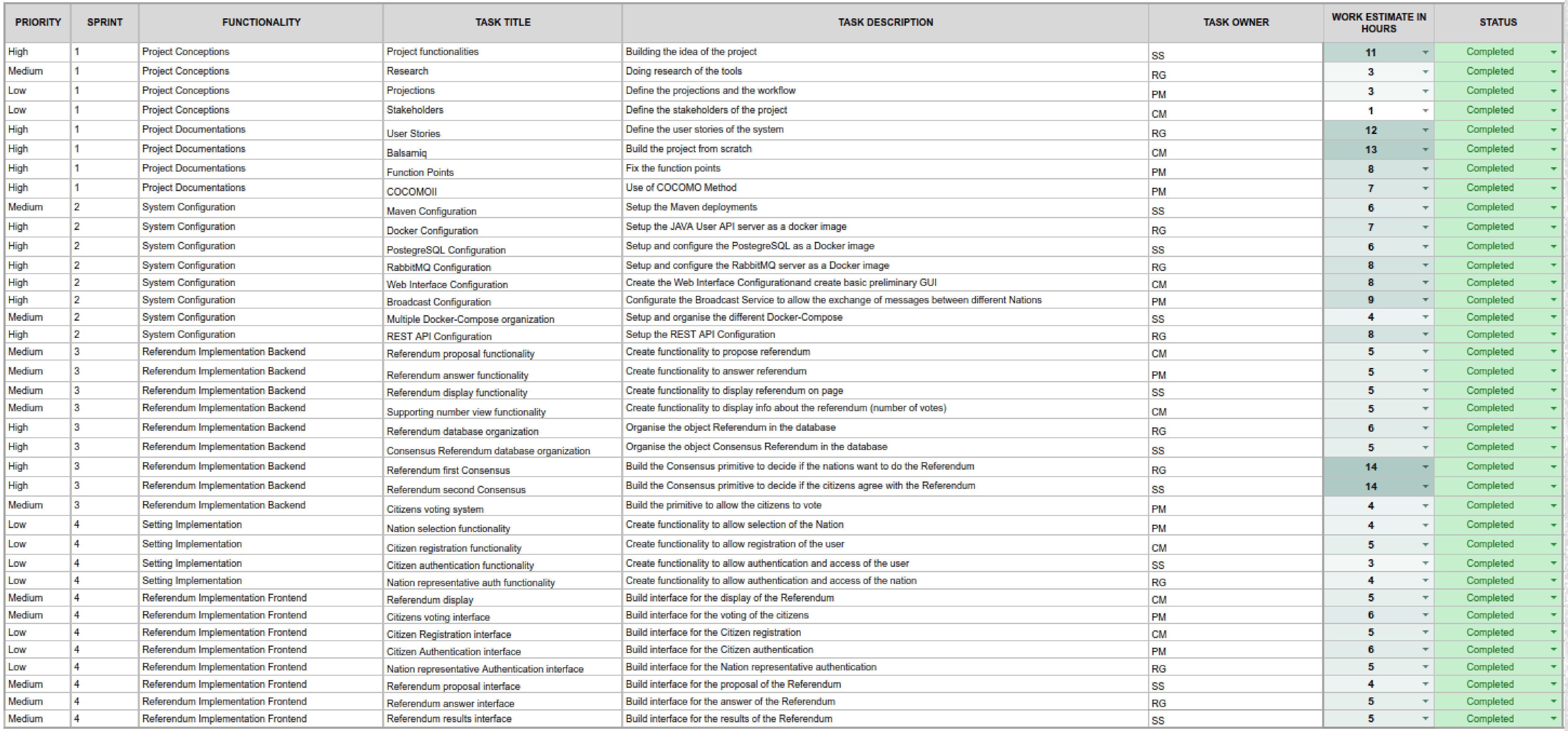
DAY	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
PLAN	234	230	226	222	218	215	211	207	203	199	195	191	187	183	179	176	172	168	164	160
ESTIMATE	234	229	224	221	218	216	210	203	199	194	191	187	184	180	176	176	172	168	163	156
HRS COMPLETED	5	5	3	3	2	6	7	4	5	3	4	3	4	4	0	4	4	5	7	7
HRS REMAINING	229	224	221	218	216	210	203	199	194	191	187	184	180	176	176	172	168	163	156	149

DAY	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
PLAN	156	152	148	144	140	137	133	129	125	121	117	113	109	105	101	97	94	90	86	82
ESTIMATE	149	149	146	141	135	132	128	128	125	122	120	118	115	112	108	105	103	98	93	90
HRS COMPLETED	0	3	5	6	3	4	0	3	3	2	2	3	3	4	3	2	5	5	3	1
HRS REMAINING	149	146	141	135	132	128	128	125	122	120	118	115	112	108	105	103	98	93	90	89

DAY	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
PLAN	78	74	70	66	62	58	55	51	47	43	39	35	31	27	23	19	16	12	8	4
ESTIMATE	89	82	75	70	65	57	54	49	45	41	41	37	33	30	25	19	16	11	5	1
HRS COMPLETED	7	7	5	5	8	3	5	4	4	0	4	4	3	5	6	3	5	6	4	1
HRS REMAINING	82	75	70	65	57	54	49	45	41	41	37	33	30	25	19	16	11	5	1	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	CM	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	Citizen registration functionality	Create functionality to allow registration of the user	CM	5	Completed
Low	4	Setting Implementation	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.

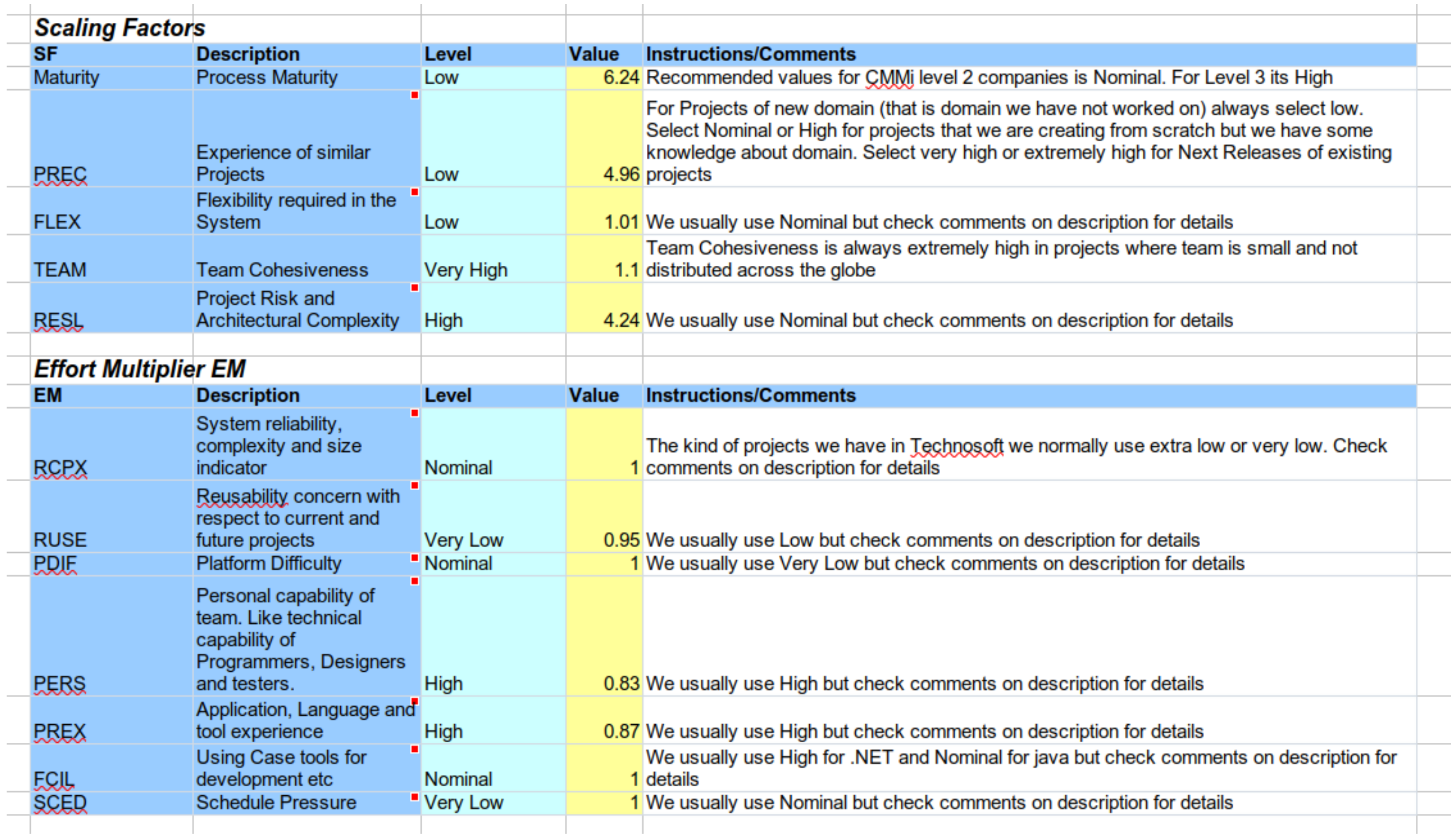
- Cost(DB) = 7 FP

- 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 Factors

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 Multiplier EM

EM	Description	Level	Value	Instructions/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	Reusability concern with respect to current and future projects	Very Low	0.95	We usually use Low but check comments on description for details
PDIF	Platform Difficulty	Nominal	1	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	0.83	We usually use High but check comments on description for details
PREX	Application, Language and tool experience	High	0.87	We usually use High but check comments on description for details
FCIL	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
SCED	Schedule Pressure	Very Low	1	We usually use Nominal but check comments on description for details

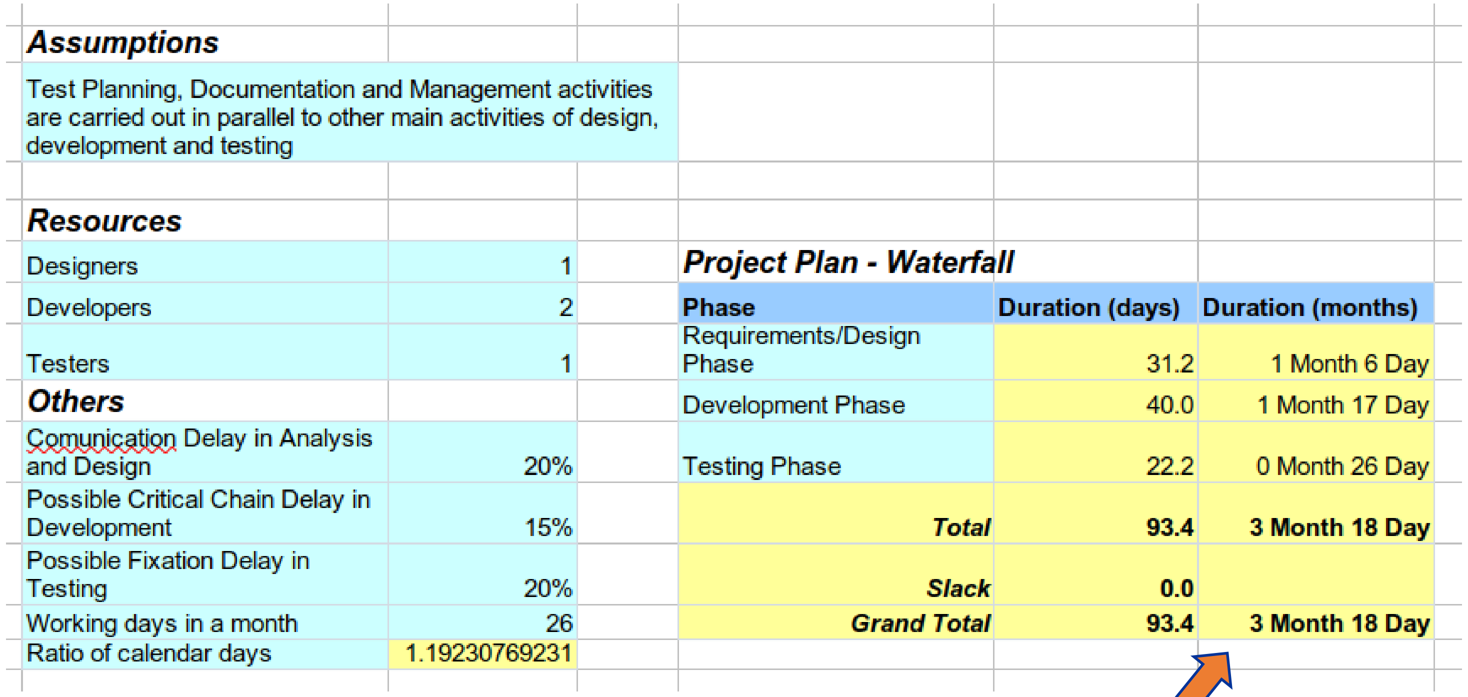
COCOMO II

HTML/JS: 26 FP

1222 SLOC

JAVA: 27 FP

1431 SLOC



Assumptions

Test Planning, Documentation and Management activities are carried out in parallel to other main activities of design, development and testing

Resources

Designers	1
Developers	2
Testers	1
Others	
Comunication Delay in Analysis and Design	20%
Possible Critical Chain Delay in Development	15%
Possible Fixation Delay in Testing	20%
Working days in a month	26
Ratio of calendar days	1.19230769231

Project Plan - Waterfall

Phase	**Duration (days)**	**Duration (months)**
Requirements/Design Phase	31.2	1 Month 6 Day
Development Phase	40.0	1 Month 17 Day
Testing Phase	22.2	0 Month 26 Day
Total	**93.4**	**3 Month 18 Day**
Slack	**0.0**	
Grand Total	**93.4**	**3 Month 18 Day**

DEMO LIVE

https://github.com/RicGobs/EPO-European-Parliament-Online

Building a Distributed System

Processes: The system represents different nations as processes. The parameter *N* is used to indicate the potential for any number of processes, and scalability is considered.

Timing Assumptions: Asynchronous communication is handled by incorporating strategic timeouts, introducing partial synchrony to set bounds on response times and alleviate consensus challenges.

Clock Synchronization: The system relies on External Synchronization using frameworks like Spring Boot and UTC to ensure consistent timekeeping.

Failure Detector: While implementing a failure detector using REST API communication can enhance consensus mechanisms, the assumption is made that all processes are correct.

Building a Distributed System
(continues)

Link Abstractions: Reliable message delivery is ensured using RabbitMQ, which offers perfect point-to-point communication (reliable delivery, no duplication, and no creation).

Flooding Consensus with a single RabbitMQ Broker:

Using 1 Broker for the broadcast communication, Uniform Reliable Broadcast is ensured. URB ensures uniform agree ment on the values decided, and even in the event of crashes, it still only requires n^2 messages (square d) instead of n^3 (cubed). This makes it a more efficient op tion for achieving consensus.

Building a Distributed System (continues)

Flooding Consensus with multiple RabbitMQ Broker:

Using multiple brokers for the broadcast communication, Best Effort Broadcast is ensured. With this property, flooding consensus is not suitable due to the risk of disag reements in case of process crashes. It comes at the disa dvantage of requiring n^3 messages in some cases.

Performance Evaluation (1)

Elapsed time (worst case) $\longrightarrow$

- Time to publish the Referendum: about 0 s;
- Time to conclude the first consensus: t.
- Time to vote by citizens: x.
- Time to conclude the second consensus: t.

Timeout for the last three actions

Total elapsed time $\longrightarrow$ $T = x + 2 \cdot t$

Performance evaluation (2)

Message weight

- Fields: 4 string, 2 boolean and 2 integer:
  - String title (max 50 characters): $50 \cdot 2$ Bytes
  - Integer status: 4 Bytes
  - String nationSourceAnswer: $3 \cdot 2$ Bytes
  - Boolean answer: 2 Bytes
  - String proposals: $4n + (n-1)$ Bytes
  - Integer round: 4 Bytes
  - Boolean isDecision: 2 Bytes
  - String dateStartConsensusProposal (dd/MM/yyyy HH:mm:ss): $19 \cdot 2$ Bytes
  - Overhead added by rabbit: 736 Bytes
- Total $W = 892$ Byte $+ 4n + (n-1)$ Byte.

Number of messages

- z: number of messages based on the consensus primitive:
  - flooding consensus with No failures: $2 \cdot n^2$ messages
  - flooding consensus with $n - 1$ failures: n^3 messages
  - flooding uniform consensus: n^3 messages
- Total messages $M = n + 2 \cdot z$
  - n: number of messages for sharing the Referendum

Performance evaluation (3)

Throughput per Referendum

- (Message weight · Number of messages)/ Total elapsed time
- W · M / T bytes/seconds

Performance evaluation (4)

Database utilization per Referendum

- ConsensusReferendum:
  - Integer status: 2 Byte
  - ConsensusReferendumId id:
    - String title (max 50 characters): 50 · 2 Bytes
    - String dateStartConsensusProposal (dd/MM/yyyy HH:mm:ss): 19 · 2 Bytes
  - String correct: 3n + (n-1) Bytes
  - Boolean decision: 2 Bytes
  - Integer round: 4 Byte String proposals: 4n + (n-1) Bytes
  - String receivedFrom: 3n + (n-1) Bytes
  - So, for a total of 146 Bytes + 10n + 3(n-1) Bytes.

Referendum

- ReferendumId id
  - String title (max 50 characters): 50 · 2 Bytes
  - String dateStartConsensusProposal (dd/MM/yyyy HH:mm:ss): 19 · 2 Bytes
- Integer status: 4 Bytes
- Integer votesTrue: 4 Bytes
- Integer votesFalse: 4 Bytes
- Integer population: 4 Bytes
- String argument: max 1 MegaByte
- String nationCreator: 3 · 2 Bytes
- String dateEndConsensusProposal: 19 · 2 Bytes
- String dateEndResult: 19 · 2 Bytes
- String dateEndConsensusResult: 19 · 2 Bytes
- So, for a total of 274 Bytes + 1 Megabyte.

The total utilization per Referendum is 420 Bytes + 1 Megabyte + 10n + 3(n-1) Bytes

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