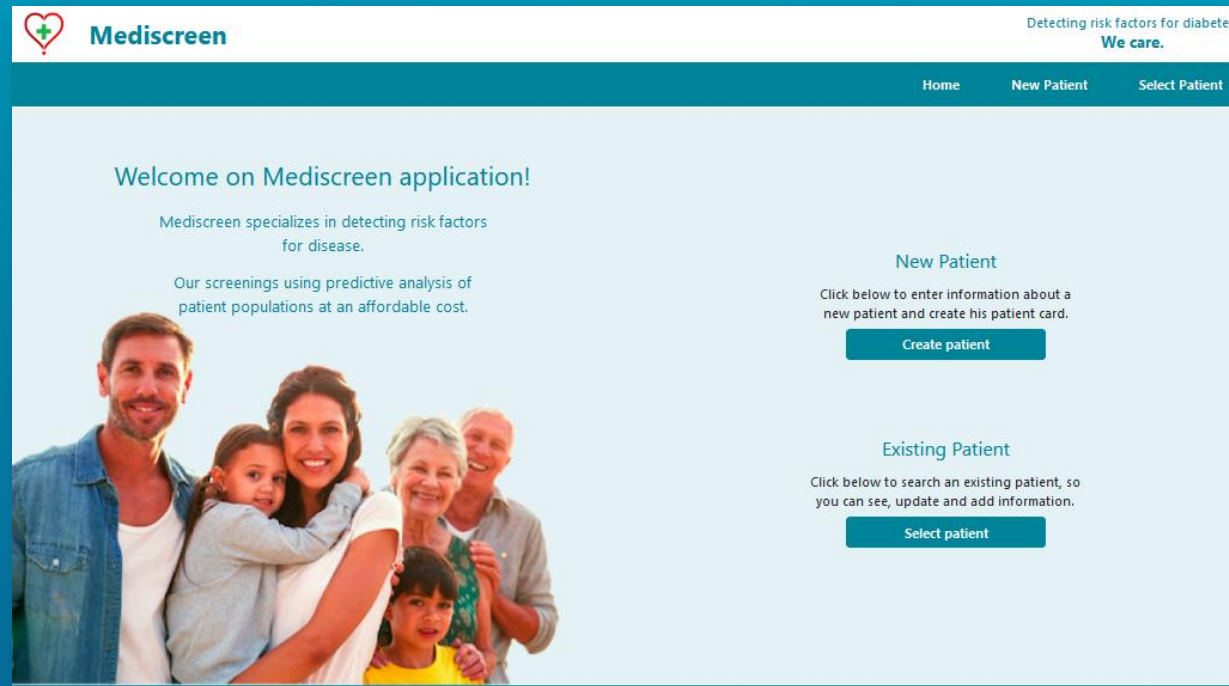


# MEDISCREEN

## Présentation de l'application



Justine CABROL

Parcours: Développeur d'Applications – Java

Projet n°9: Développez une solution en microservices pour votre client

# Sommaire

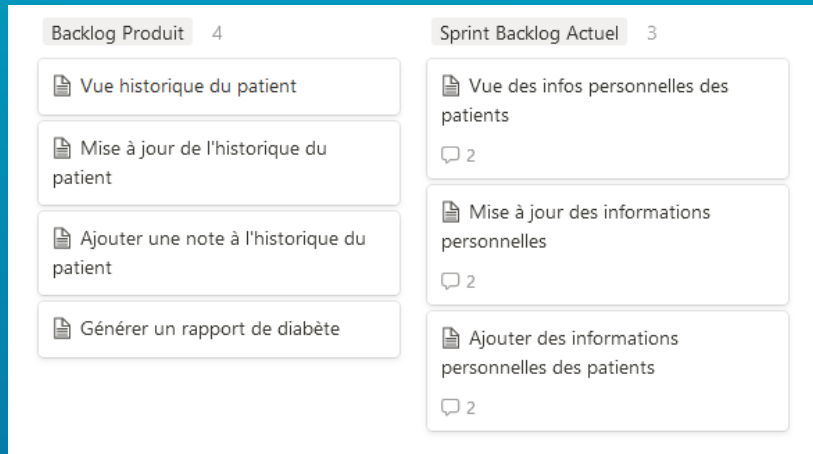
- I- Spécifications techniques
- II- Méthodologie utilisée pour le projet
- III- Architecture générale
- IV- Présentation des microservices
- V- Déploiement Docker
- VI- Tests et documentation

# I- SPÉCIFICATIONS TECHNIQUES

- **Code :**
  - Java 11
  - SpringBoot 2.6.10
  - Maven 4.0.0
- **Gestion des données :**
  - MySQL 8.0
  - SpringDataJPA 2.6.6
  - MongoDB 4.4.2
  - SpringData MongoDB 3.3.6
- **Communication des microservices :**
  - Feign 3.1.0
- **Templates :**
  - Thymeleaf 3.0.15
  - Bootstrap 5.2
- **Tests :**
  - JUnit 5.8.2
  - Mockito 4.0.0
  - JaCoCo 0.8.7
- **Déploiement du code :**
  - Docker 4.11.1
- **Documentation/Suivi :**
  - Swagger 3.0.0
  - Actuators 2.6.10

# II- MÉTHODOLOGIE UTILISÉE

## Méthodologie Agile



### - User Stories et Product Backlog :

Les User Stories classées et ordonnées dans le Product Backlog indiquent les différentes tâches à accomplir pour obtenir le produit final.

### - Sprints :

3 sprints successifs pour déployer toutes les fonctionnalités

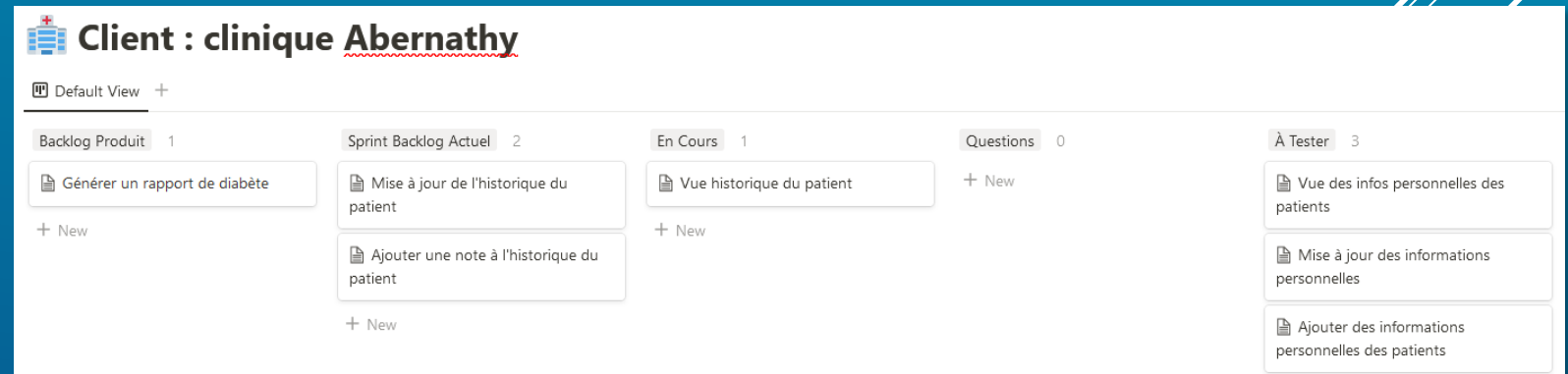
- Sprint 1 : Stockage et opérations CRUD sur les patients.
- Sprint 2 : Stockage et opérations CRUD sur les notes médicales.
- Sprint 3 : Calcul du risque de diabète.

### - Kanban :

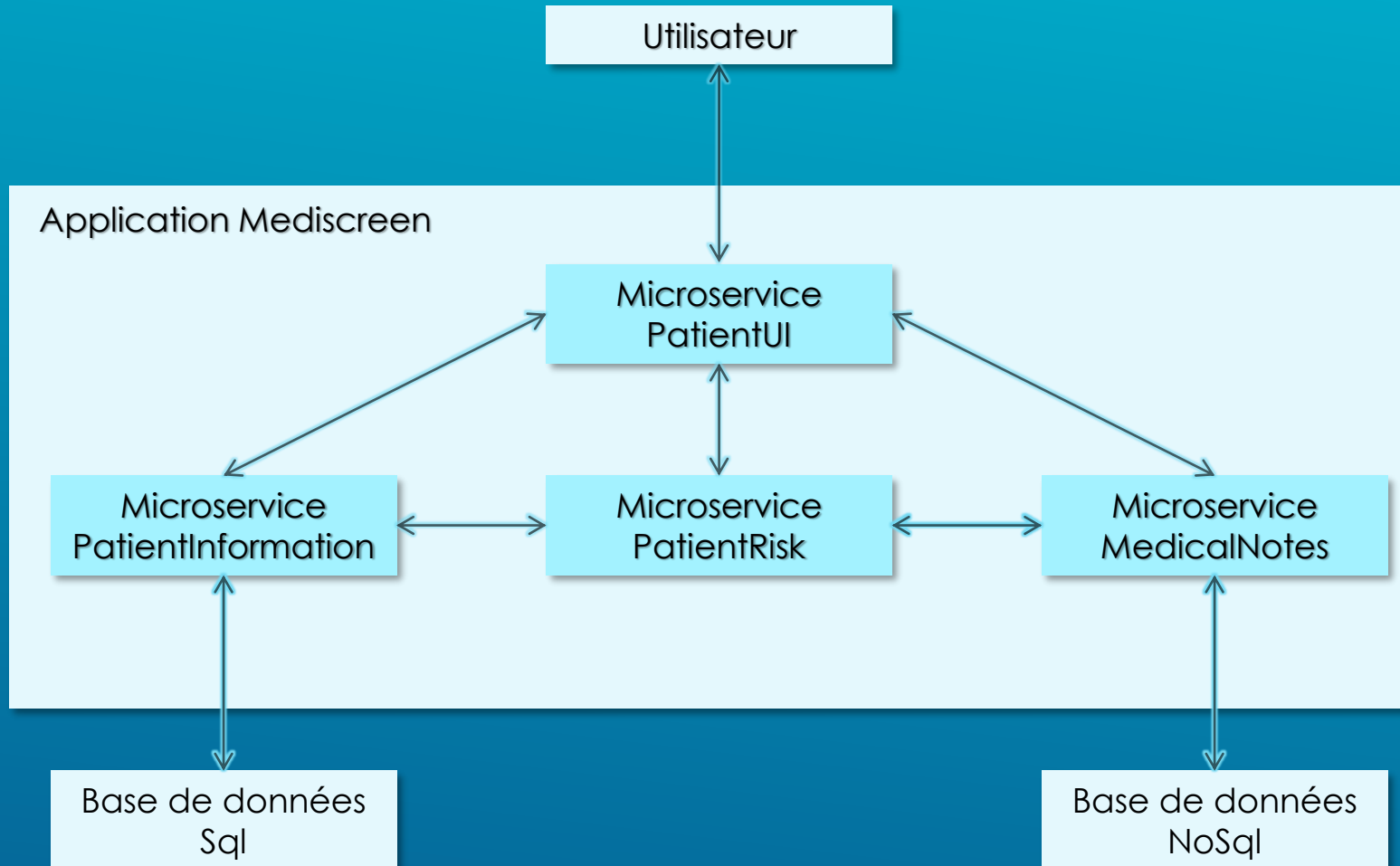
Le Kanban est utilisé pour indiquer l'évolution du projet. Les différentes User Stories sont déplacée au fur et à mesure en fonction de leur avancement.

### - Sprint Review :

A la fin de chaque Sprint, la Sprint Review permet de dresser un bilan du Sprint effectué.



# III- ARCHITECTURE GÉNÉRALE



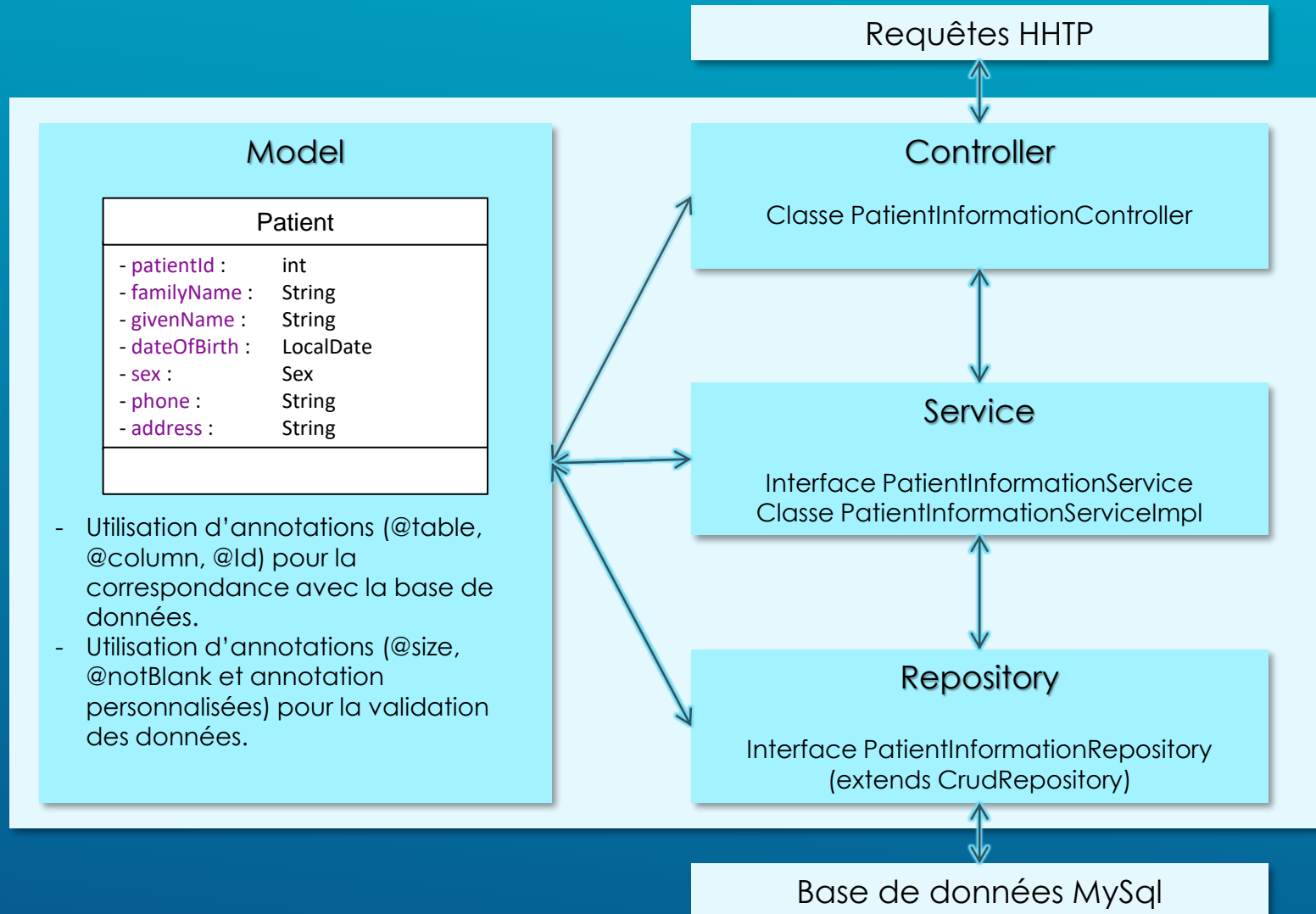
## Architecture en microservices :

4 microservices autonomes qui communiquent entre eux (Feign)

- Plus de flexibilité
- Suivi et maintenance facilités
- Utilisation de technologies différentes

# IV- PRÉSENTATION DES MICROSERVICES

## a) Microservice PatientInformation



Respect des principes SOLID

Modèle MVC et Architecture 3-tiers

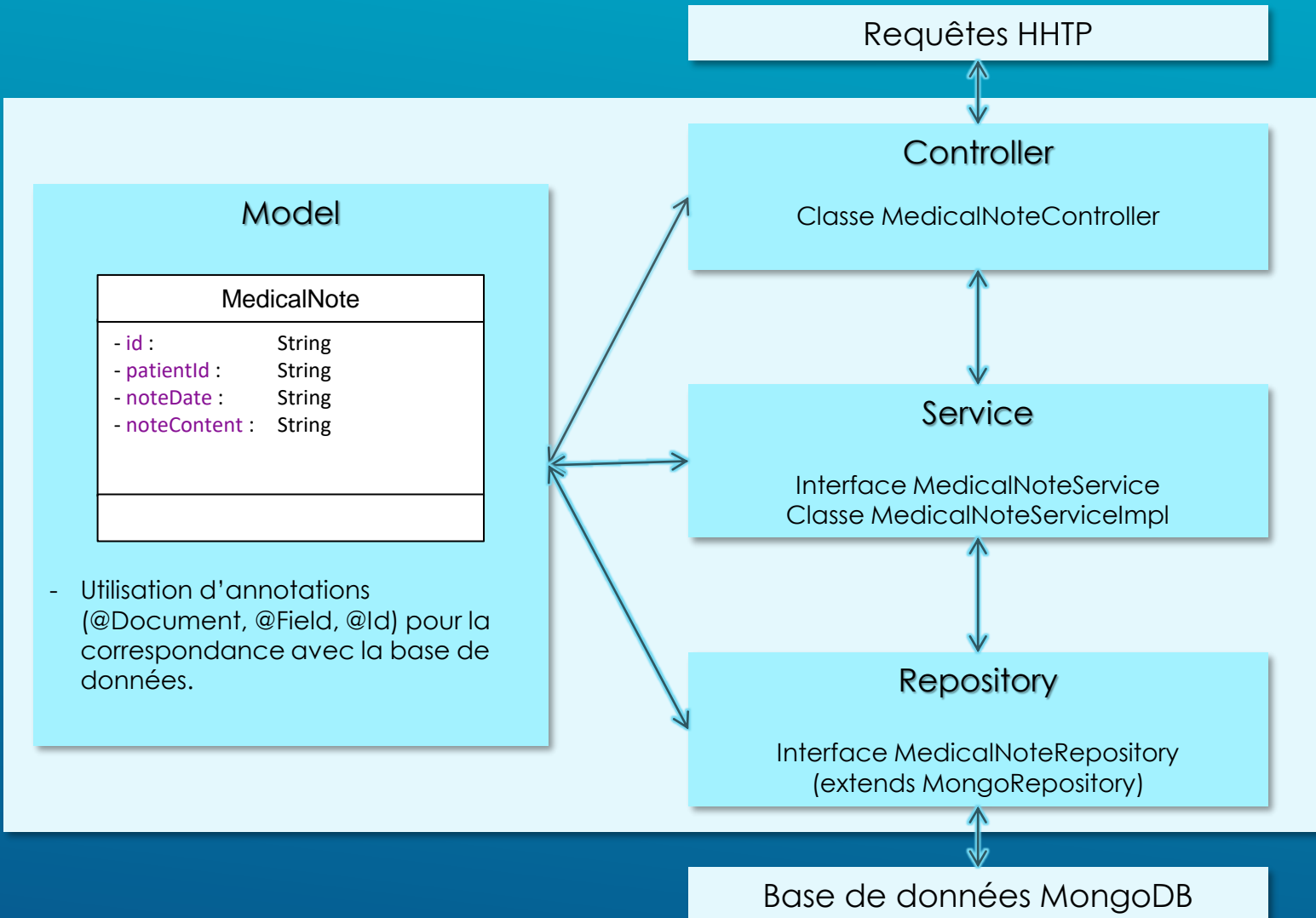
Exposition de 6 endpoints

pour réaliser les opérations CRUD

POST	/patient/add	addNewPatient
DELETE	/patient/delete/{patientId}	deletePatient
GET	/patient/getAll	getAllPatients
GET	/patient/getPatient	getPatientsByName
GET	/patient/getPatient/{patientId}	getPatientById
PUT	/patient/update/{patientId}	updatePatient

# IV- PRÉSENTATION DES MICROSERVICES

## b) Microservice MedicalNotes



Respect des principes SOLID

Modèle MVC et Architecture 3-tiers

Exposition de 6 endpoints

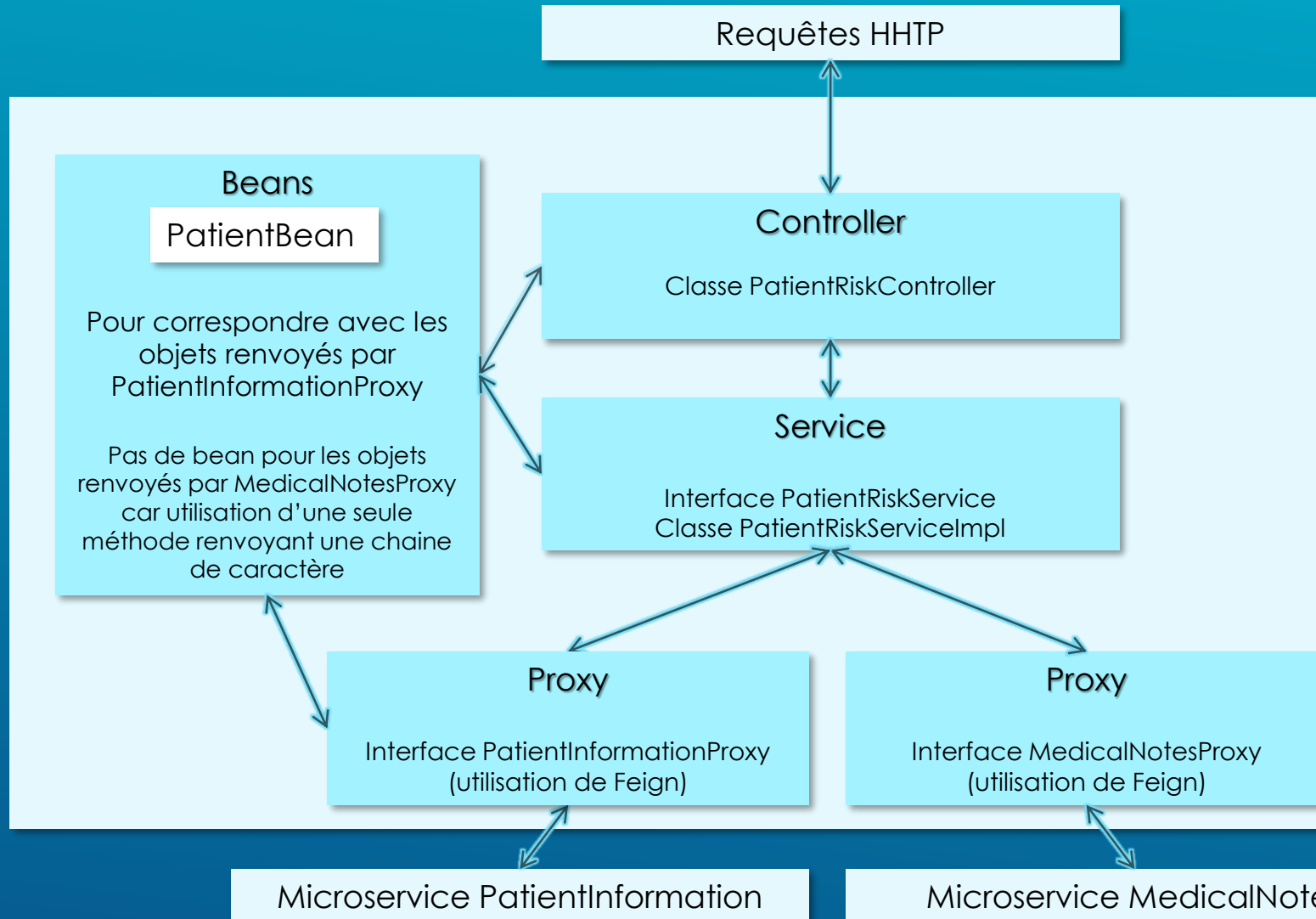
pour réaliser les opérations CRUD

POST	/patHistory/add/{patientId}	addMedicalNote
DELETE	/patHistory/delete/{noteId}	deleteMedicalNote
GET	/patHistory/getAllInformation/{patientId}	getAllInformation
GET	/patHistory/getAllMedicalNotes/{patientId}	getMedicalNotesByPatient
GET	/patHistory/getMedicalNote/{id}	getMedicalNote
PUT	/patHistory/update/{noteId}	updateMedicalNote



# IV- PRÉSENTATION DES MICROSERVICES

## c) Microservice PatientRisk



Respect des principes SOLID

Modèle MVC et Architecture 3-tiers

Exposition de 3 endpoints  
pour obtenir les informations

GET	/assess/familyName	getAssessByName
GET	/assess/id	getAssessById
GET	/diabetesRisk/{id}	getDiabetesRisk

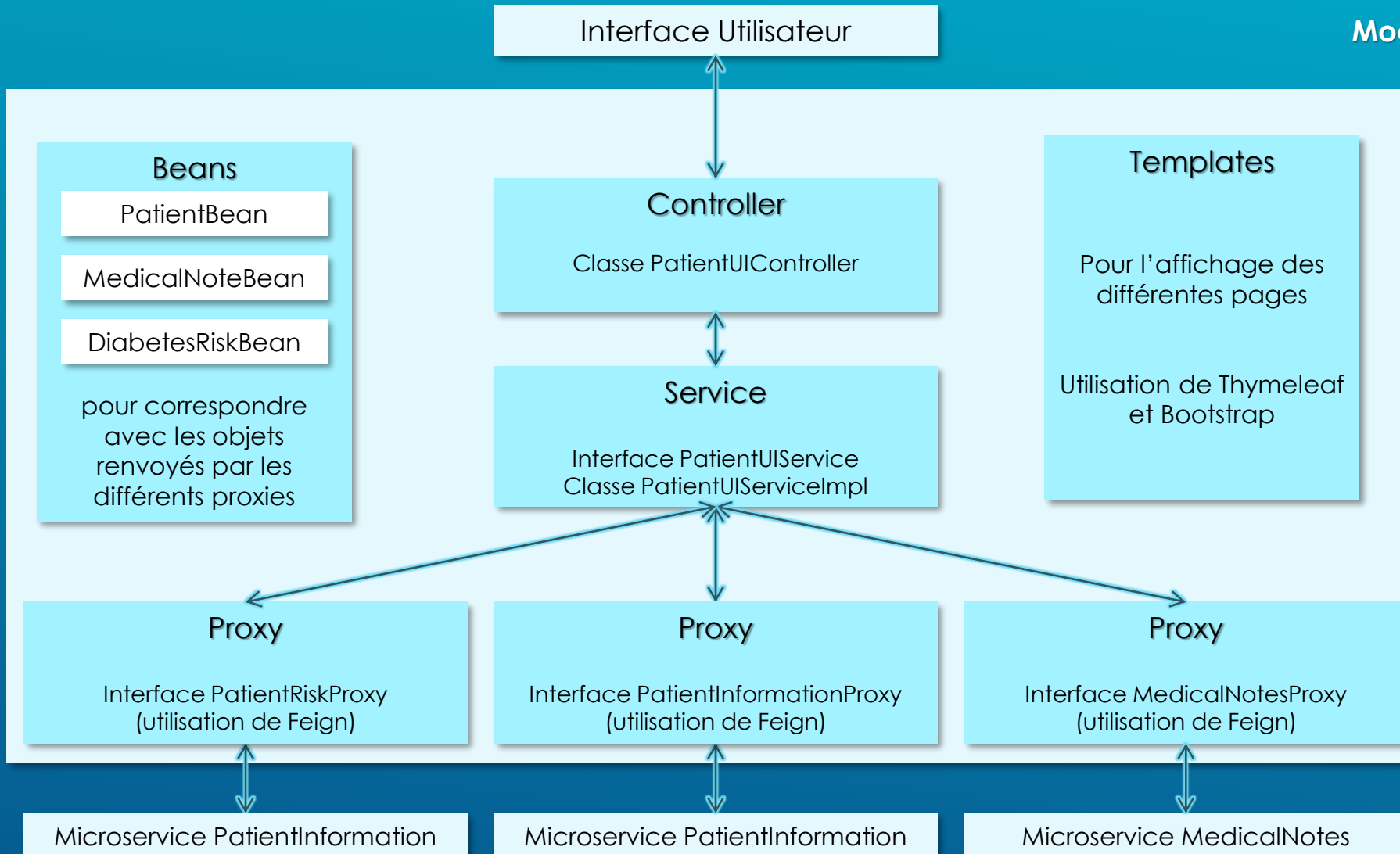


# IV- PRÉSENTATION DES MICROSERVICES

## d) Microservice PatientUI

Respect des principes SOLID

Modèle MVC et Architecture 3-tiers







































# V- DÉPLOIEMENT DOCKER

6 images : 4 microservices +  
2 bases de données

NAME ↑		TAG
medical_notes	IN USE	1.0
mongo	IN USE	latest
mysql	IN USE	8.0
patient_information	IN USE	1.0
patient_risk	IN USE	1.0
patient_ui	IN USE	1.0

Des volumes pour les bases de  
données, afin de persister les données

	NAME ↓	STATUS
<input type="checkbox"/>	mediscreen_mysql_db_data	in use
<input type="checkbox"/>	mediscreen_mongo_db_data	in use
<input type="checkbox"/>	mediscreen_mongo_config	in use

<input type="checkbox"/>		mediscreen 6 containers	-	Running (6/6)	-	   
<input type="checkbox"/>		mediscreen_patientUI_1 12c736f78ca4	patient_ui:1.0	Running	8083	5 minutes ago     
<input type="checkbox"/>		mediscreen_patientRisk_1 28cb491d7c23	patient_risk:1.0	Running	8080	5 minutes ago     
<input type="checkbox"/>		mediscreen_patientInform c081d0a61794	patient_information:1.0	Running	8081	5 minutes ago     
<input type="checkbox"/>		mediscreen_medicalNote: 8480f02b365f	medical_notes:1.0	Running	8082	5 minutes ago     
<input type="checkbox"/>		mediscreen_mysql_db_1 0e5d2b0f1c96	mysql:8.0	Running	3308	5 minutes ago     
<input type="checkbox"/>		mediscreen_mongodb_1 f1e5e05a947e	mongo:latest	Running	27018	5 minutes ago     

Utilisation de Docker-compose  
pour obtenir une application  
multi-conteneurs

# VI- TESTS ET DOCUMENTATION

## a) Rapports de tests et rapports de couverture de code

- Tests unitaires et tests d'intégration
- Utilisation de Junit et Mockito
- Couverture mesurée avec JaCoCo

### patientInformation

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
mediscreen.patientInformation.exception		73%		100%	4	20	12	46	4	18	0	3
mediscreen.patientInformation		37%		n/a	1	2	2	3	1	2	0	1
mediscreen.patientInformation.modele		98%		n/a	1	51	0	54	1	51	0	4
mediscreen.patientInformation.validation		98%		87%	7	34	1	41	0	6	0	3
mediscreen.patientInformation.service		100%		90%	4	31	0	79	0	9	0	1
mediscreen.patientInformation.controller		100%		n/a	0	8	0	17	0	8	0	1
mediscreen.patientInformation.configuration		100%		n/a	0	4	0	8	0	4	0	2
Total	64 of 1,170	94%	11 of 104	89%	17	150	15	248	6	98	0	15

Tests	Errors	Failures	Skipped	Success Rate	Time
115	0	0	0	100%	12.403

### medicalNotes

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
mediscreen.medicalNotes.exception		91%		n/a	2	13	1	20	2	13	0	3
mediscreen.medicalNotes		37%		n/a	1	2	2	3	1	2	0	1
mediscreen.medicalNotes.service		100%		100%	0	9	0	23	0	7	0	1
mediscreen.medicalNotes.controller		100%		n/a	0	8	0	20	0	8	0	1
mediscreen.medicalNotes.model		100%		n/a	0	9	0	6	0	9	0	1
mediscreen.medicalNotes.configuration		100%		n/a	0	4	0	8	0	4	0	2
Total	12 of 331	96%	0 of 4	100%	3	45	3	80	3	43	0	9

Tests	Errors	Failures	Skipped	Success Rate	Time
37	0	0	0	100%	6.741

### patientRisk

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
mediscreen.patientRisk.bean		44%		n/a	7	18	4	11	7	18	0	1
mediscreen.patientRisk.exception		91%		n/a	2	13	1	20	2	13	0	3
mediscreen.patientRisk		37%		n/a	1	2	2	3	1	2	0	1
mediscreen.patientRisk.model		100%		n/a	0	3	0	30	0	3	0	3
mediscreen.patientRisk.service		100%		93%	2	25	0	62	0	8	0	1
mediscreen.patientRisk.controller		100%		n/a	0	4	0	4	0	4	0	1
mediscreen.patientRisk.configuration		100%		n/a	0	4	0	8	0	4	0	2
Total	60 of 727	91%	2 of 32	93%	12	69	7	138	10	52	0	12

Tests	Errors	Failures	Skipped	Success Rate	Time
33	0	0	0	100%	7.245

### patientUI

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
mediscreen.patientUI.modele		68%		n/a	8	23	2	16	8	23	0	3
mediscreen.patientUI.bean		86%		n/a	3	29	1	23	3	29	0	3
mediscreen.patientUI.exception		82%		n/a	3	12	2	22	3	12	0	3
mediscreen.patientUI.patientUIService		96%		89%	2	32	2	99	0	18	0	1
mediscreen.patientUI.validation		95%		74%	12	33	1	41	0	6	0	3
mediscreen.patientUI		37%		n/a	1	2	2	3	1	2	0	1
mediscreen.patientUI.controller		100%		100%	0	16	0	90	0	14	0	1
mediscreen.patientUI.configuration		100%		n/a	0	4	0	8	0	4	0	2
Total	100 of 1,436	93%	17 of 86	80%	29	151	10	302	15	108	0	17

Tests	Errors	Failures	Skipped	Success Rate	Time
65	0	0	0	100%	22.955

# VI- TESTS ET DOCUMENTATION

## b) Documentation et suivi

- Documentation des API:

*Document Mediscreen + Swagger*

- Documentation des méthodes:

*Javadoc*

- Suivi et maintenance:

*Exposition des Actuators health, info, beans, env, metrics et httptrace*