

Projet d'émargement par QRcode



Membre du projet : Clément GUYOT, Louis PEREDA, Jonathan BALTACI, Mickael POUDROUX.

Présentation Métier de la solution	3
Organisation de l'équipe	4
Identification des tâches à réaliser	4
Planification	5
Gantt	5
RACI	6
Déploiement du Projet	7
Configuration requise	7
Avec la dockerisation	7
Déploiement du front-end	7
Déploiement du back-end	7
Lancer l'application	7
Tableau des Uris	8
Diagrammes des Séquence	17

I. Présentation Métier de la solution

Afin de répondre à la demande d'une mise en place d'un émargement par QRcode nous proposons la solution suivante:

Dans un premier temps, les étudiants et professeurs s'inscrivent sur notre site web.

Les étudiants peuvent flasher des QR code, consulter les fiches ainsi que des options de modification de profil(Changer d'image de profil).

Les professeurs eux ont plus d'options, ils peuvent générer de nouvelle fiche d'émargement, chaque fiche correspond à un cour pour un jour donné. Une fois la fiche générer un QRcode est affiché, les étudiants peuvent le scannée et ainsi être émargés.

L'admin peut gérer la globalité des utilisateurs (Roles / suppression / modifié information).

II. Organisation de l'équipe

A. Identification des tâches à réaliser

Après lecture du sujet nous avons identifié plusieurs tâches répartie entre le back-end et le front-end.

La première tâche à réaliser était de créer l'espace de travail et mettre les technologies que nous allons utiliser, un client angular pour le front et une base de donnée en MongoDB.

Nous avons ensuite réfléchi à comment avancer. Le back à plusieurs parties :

- Le Modèle
- Sécurité
- Les services
- Les Tests

Afin d'établir les objets à créer nous nous sommes posés des Back-log de situation réelle afin d'avoir une idée claire de ce que l'on a besoin.

Ex:

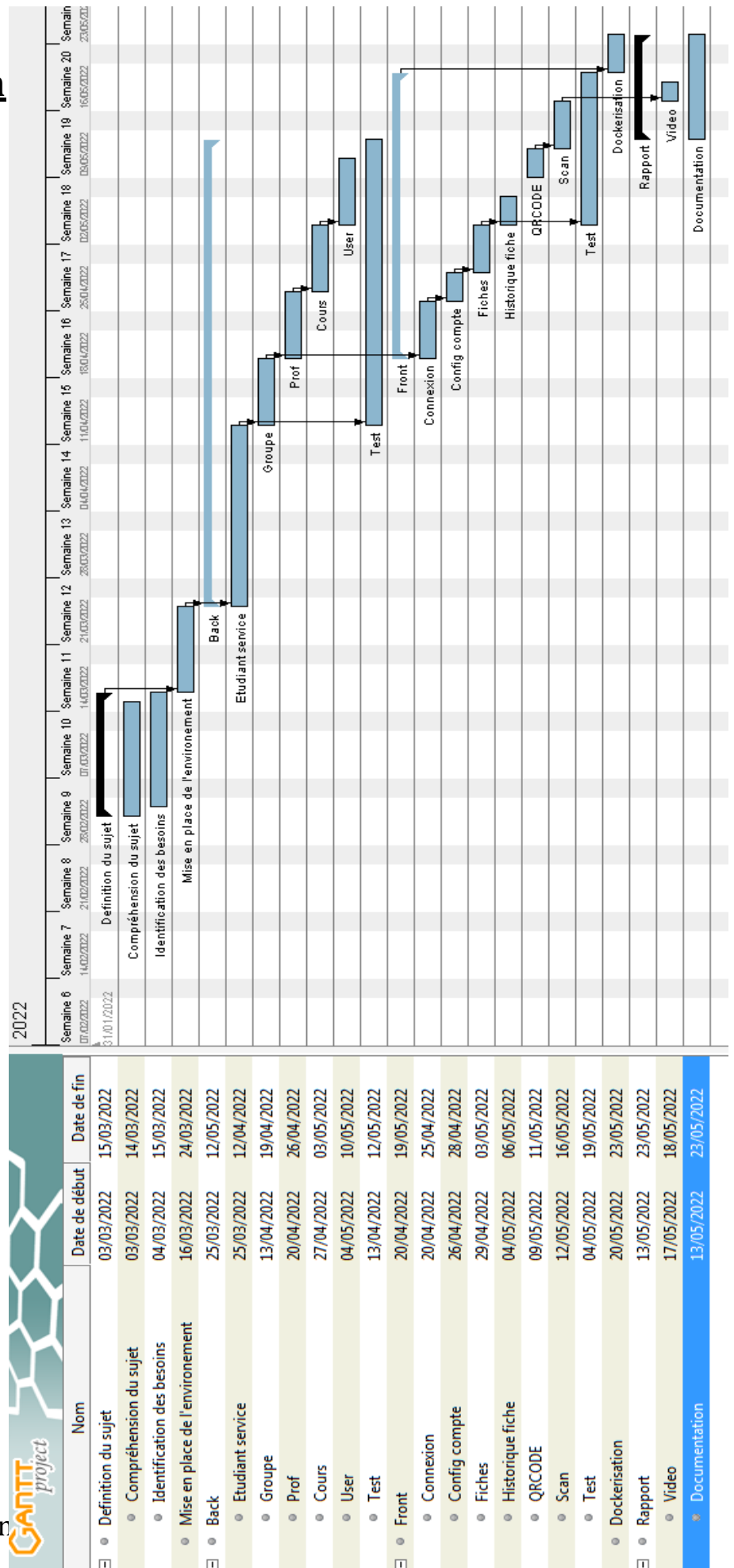
Le professeur affiche le QRcode de son cours.

Données: - Professeur - Cours - QRcode pour différents Cours

B. Planification

a. Gantt

Ar



b. RACI

		Mickaël POUDROUX	Clément Guyot	Jonathan Baltaci	Louis PEREDA
s'organiser en groupe de projet	Faire le point des compétences de chacun	A	A	A	A
	Lister les compétences a priori nécessaires au projet	A	A	A	A
	Mettre en place les différents outils de gestion de projet (espace de travail, outil de planification, outil de reporting...)	A	A	R	A
s'approprier le sujet (Réunion de groupe)	Lire le sujet seul et l'annoter	A	A	A	A
	Mise en commun	A	A	A	A
Mise en place de l'environne- ment	Commit initial avec les Différentes Partie du projet	C	C	R	C
Back	Etudiant	C	A	R	I
	Groupe	C	R	A	I
	Prof	C	A	R	I
	Cours	C	R	A	I
	User	C	A	R	A
	Test	A	R	A	A
Front	Connexion	I	A	R	C
	Config compte	I	A	R	C
	Fiches	I	A	R	C
	Historique fiche	I	A	R	A
	QRCODE	A	A	R	I
	Scan	I	A	R	C
	Test	C	A	A	A
Dockerisation	Compilation du projet en module Docker	I	A	R	I
Rapport	Presentation Metier	R	A	A	C
	GANTT	R	C	I	C
	RACI	R	C	I	C
	Instruction au déploiement du projet	R	A	C	C
	Tableau URIS	R	C	I	I
	Doc Spring REST Docs	C	R	A	I
	Diagramme de séquence	R	C	C	C
	Video	I	I	R	I

C. Déploiement du Projet

a. Configuration requise

Requis : git, node.js, Angular Cli, Docker

Après avoir télécharger le projet, suivre les consignes ci-dessous.

b. Avec la dockerisation

dans une console :

\$ /projet-s2-groupe-11/back/ : docker-compose up --build

c. Déploiement du front-end

Dans le dossier front lancer les commandes :

\$ /projet-s2-groupe-11/front/ : npm install

\$ /projet-s2-groupe-11/front/ : ng serve

d. Déploiement du back-end

maven package pour generer le jar

Lancer le fichier "*docker-compose.yaml*"

Windows : passer par docker desktop

Linux: docker-compose -f docker-compose.yaml up

Pour Ionic : npm install -g ionic

ionic serve

e. Lancer l'application

Sur un navigateur web ouvrir le lien <http://localhost:8100> pour le client IONIC/angular (émargement QR code)

<http://localhost:4200> pour angular simple (Gestion Users sans QR code)

D. Tableau des Uris

Nom methode	URL	Type	Param	Sortie
createCours	"/cours"	POST	cours: Type Cours	201 409 CoursDejaCr eerException 400 NomCourInv alidExceptio n PasHoraireE xception
getAllCours	"/cours"	GET	/	cours: Liste des cours (Type List<>Cours)
getCoursByNom	"/cours/{no mCours}"	GET	nomCours: Type String	202 : data cours 404 CoursInnexis tantExceptio n
updateCours	"/cours/{no mCours}"	PUT	nomCours: Type String	202 404

			cours Type Cours	CoursInnexist antExceptio n
deleteCours ByNom	"/cours/{no mCours}"	DEL	nomCours: Type String	204 400 CoursInnexist antExceptio n
deleteAllCou rs	"/cours"	DEL	/	204
addGroupeT pInCours	"/cours/{no mCours}/gro upe"	PUT	nomCours: Type String numeroGrou pe: Type String	202 404 GroupeInnex istantExcepti on CoursInnexist antExceptio n 409 GroupeTpDej aAjouterExce ption
addProfInCo urs	"/cours/{no mCours}/pr of"	PUT	nomCours: Type String nomProf: Type String	202 404 ProfInnexist antExcepton CoursInnexist antExceptio n 409 ProfDejaAjou terException

createEtudiant	"/etudiants"	POST	Etudiant : etudiant	202 409 NumEtudiantDejaPresent Exception 400 NumEtudiantNonValideE xception 406 InformationIncompletExc eption
getAllEtudiants	"/etudiants"	GET	/	Ok
getEtudiant	"/etudiants/ {numEtudiant}"	GET	String : numEtudiant	204 404 EtudiantInexistentExcep tion
updateEtudiant	"/etudiants/ {numEtudiant}"	PUT	String : numEtudiant Etudiant : etudiant	204 404 EtudiantInexistentExcep tion
deleteEtudiantByNumEtu	"/etudiants/ {numEtudiant}"	DEL	String : numEtudiant	204 404 EtudiantInexistentExcep tion

deleteAllEtudiant	"/etudiants"	DEL	/	204
reinitEmerge	"/etudiants"	PUT	/	Ok
addFiche	"/liste"	POST	FicheEmerge ment : ficheEmerge ment	201
getListeEtudiant	"/liste/{nom Cours}"	GET	String : nomCours	200
getAllListes	"/liste"	GET	/	200
premiereSignatureFiche	"/liste/signer/{idFiche}/ {nomEtu}"	GET	String : idFiche String : nomEtu	200
scanFiche	"/liste/scan/{idFiche}/{u sername}"	GET	String : idFiche String : username	200
deleteById	"/liste/{idFiche}"	DEL	String : idFiche	204
createGroupeTp	"/groupeTp"	POST	String nomGroupe	201 409 GroupeDejaCreerException 400 NomGroupe

				NonValideException
getAllGroupeTp	"/groupetp"	GET	/	200
addEtudiantAuGroupeTp	"/groupetp/{nomGroupe}"	PUT	String nomGroupe String numEtudiant	202 404 GroupeInexistentException EtudiantInexistentException 409 EtudiantDejaDansUnGroupeException
getGroupeTp	"/groupetp/{numeroGroupe}"	GET	String numGroupe	302 404 GroupeInexistentException
deleteGroupeByNumGroupe	"/groupetp/{nomGroupe}"	DEL	String nomGroupe	204 404 GroupeInexistentException
deleteAllGroupe	"/groupetp"	DEL	/	204
deleteAllEtudiantInGroupe	"/groupetp/{numeroGroupe}/etudiant"	DEL	String numeroGroupe	204

createProf	"/prof"	POST	Prof prof	201 400 NomProfInvalideException PrenomProfInvalideException 409 ProfDeja Cree rException
getAllProf	"/prof"	GET	/	200
getProfByNom	"/prof/{nomProf}"	GET	String nomProf	302 404 ProfInnexistentException
deleteAllProf	"/prof"	DEL	/	204
deleteProfByName	"/prof/{nomProf}"	DEL	String nomProf	204 404 ProfInnexistentException
deleteAllCoursInProf	"/prof/{nomProf}/cours"	DEL	String nomProf	204 404 ProfInnexistentException
login	"/login"	POST	User user	200 423 LockedExcep

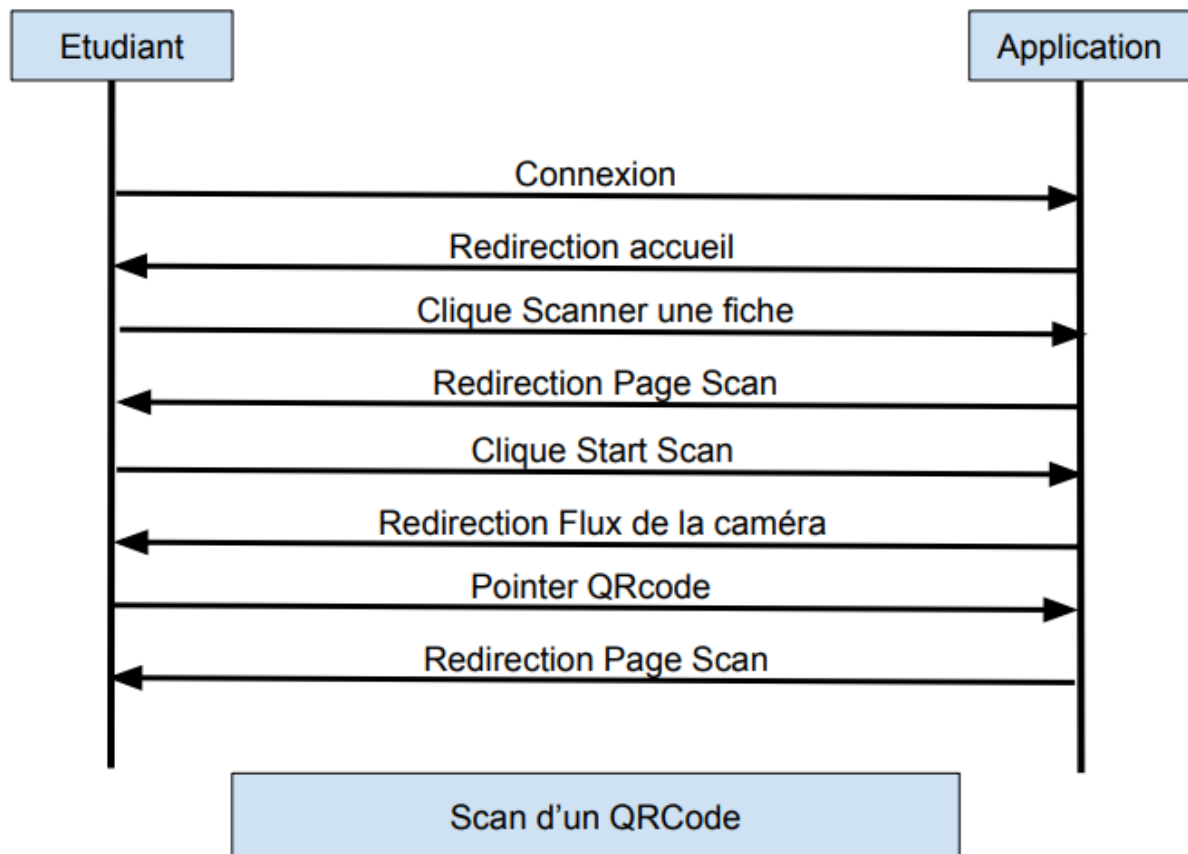
				tion 400 BadCredenti alsException
register	"/register"	POST	User user	200 404 UserNotFoun dException 409 EmailExistEx ception NumEtudian tDejaPresent Exception UsernameExi stException
registerNoGe neratedPass word	"/register/n o-generated- password"	POST	User user	200 404 UserNotFoun dException 409 EmailExistEx ception UsernameExi stException
addNewUser	"/add"	POST	String firstName String lastName	200 404 UserNotFoun dException

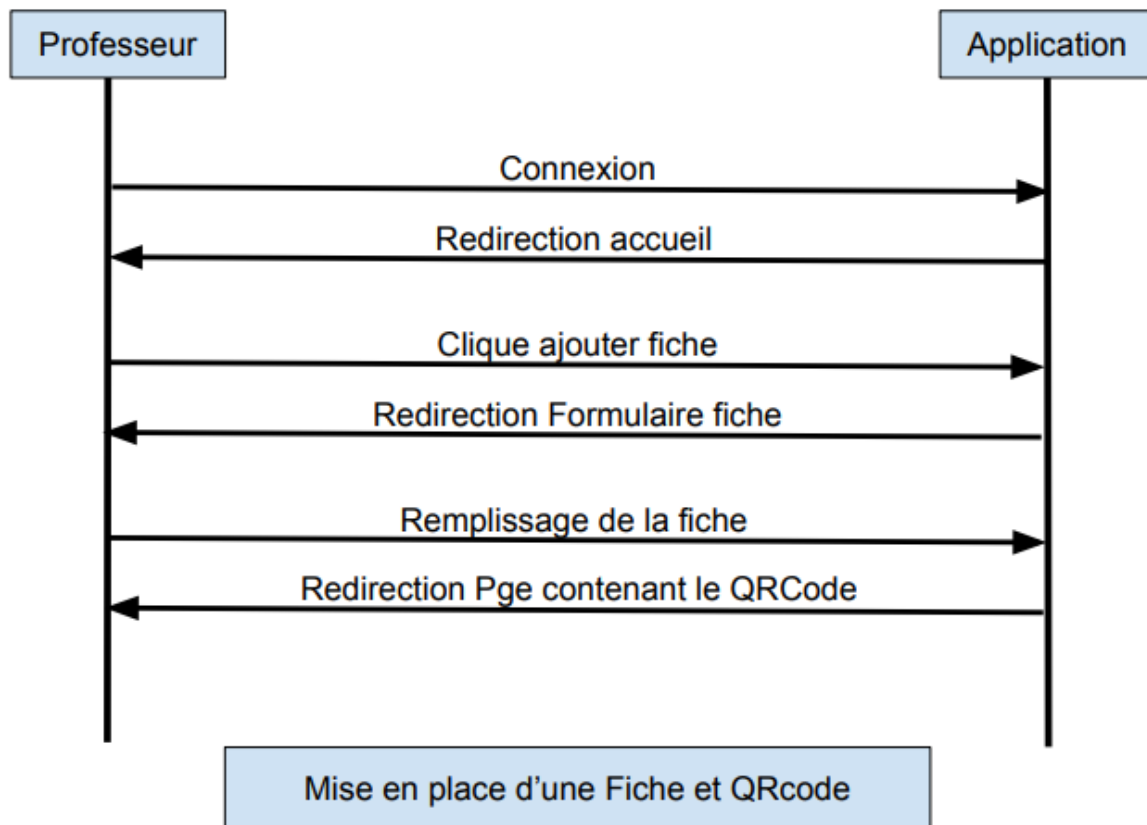
			String username String email String role String isActive String isNonLocked MultipartFile profileImage	409 EmailExistException UsernameExistException 400 IOException NotAnImageFileNotFoundException
updateUser	"/update"	POST	String firstName String lastName String username String email String role String isActive String isNonLocked MultipartFile profileImage	200 404 UserNotFoundException 409 EmailExistException UsernameExistException 400 IOException NotAnImageFileNotFoundException
updateProfileImage	"/updateProfileImage"	POST	String username MultipartFile	200 404 UserNotFoundException

			profileImage	dException 409 EmailExistException UsernameExistException 400 IOException NotAnImage FileNotFoundException
getProfileImage	"/image/{username}/{fileName}"	GET	String username String fileName	//
getTempProfileImage	"/image/profile/{username}"	GET	String username	//
getUser	"/find/{username}"	GET	String username	200
getAllUsers	"/list"	GET	/	200
resetPassword	"/resetpassword/{email}"	GET	String username	200 404 EmailNotFoundException
deleteUser	"/delete/{username}"	DEL	String username	204 404 UserNotFoundException 400

				IOException
--	--	--	--	-------------

E. Diagrammes des Séquence





F. Video Lien

https://www.youtube.com/watch?v=LV7GI2KVU2Q&ab_channel=genzio