

Prénom, Nom : Équipe :	POT ET Germain - Animax					
Niveau / Qualité de mobilisation:	Pas mobilisé	Découverte	Notions	Application	Maîtrise	Experte
Compétences Mobilisés par le projet (simplifiés / adaptés)*						
Application / Démo						
C02.1	Complexité : Résoudre un problème complexe en mobilisant les concepts, méthodes et outils informatiques et mathématiques adaptés.		Application statique.	API sample sans identification utilisateur.	Appli avec authentification et API protégés.	Appli avec schéma des données avancés et/ fonctionnalités originals (son, cartes, videos...).
C03.2	Schema des Données & BackEnd : Concevoir et modéliser le stockage de données		UML ou schema papier simple.	UML realiste.	Schema implementé et BD instansie.	BD optimisé.
C04.8	FrontEnd : système utile, avec interface cohérent et utilisable, construite avec une approche centrée-utilisateur (conception, évaluation)		Prototype papier ou page Figma simple.	Figma (ou autre prototype) site complete.	Une front basic mais qui fonctionne.	Un front complete, testé par d'utilisateurs.
Développement / Code						
C04.10	Qualité Code : Code stable, et qui répond aux besoins demandés [...]		Code qui se n'execute pas.	Code qui tourne avec quelques bugs.	Code complexe qui tourne sans bugs.	Code avec des bonnes pratiques respectés (structure, generecité, noms et contenue des fonctions, ...), et bien commenté.
C04.7	Test : Tester un logiciel : concevoir, planifier et exécuter un plan de validation logiciel [...]		Code pas testé.	Test partielle manuel et/ou avec des utilisateurs.	Tests unitaires faites.	Code optimisé et possibilité de communication avec des services extérieurs.
Rapport						
C04.3 C06.1	Cahier : Traduire des fonctionnalités attendues en cahiers des charges. [...]. Avoir une réflexion sur le cahier initiale les évolutions. Veille : Tenir en compte de concurrence (veille scientifique / technologique).		Liste des fonctionnalités haute niveau, pas de planification.	Cahier des charges non-exhaustive, planification pas reflechi.	Cahier des charges exhaustive, sans planification ou planification pas réfléchi. Idée de ce qui existe dans le marché (concurrence).	Cahier des charges exhaustive avec planification, connaissance des concurances.
C14.1 C04.4	Réflexion : Développer une pratique réflexive sur son projet. Argumenter pour ces décisions: Bien argumenter sur la pertinence du projet, les choix technologique et fonctionnels, l'organisation du travail. Donner des références/citations des sources utilisés.		Pas de réflexion personnel et/ou pas de reflexion en groupe.	Reflecion superficial sur les choix (technologiques, conception, structure).	Réflexion approfondis sur les choix (techno, conception, solution), sur la demarche (travaille en équipe, gestion, planification), sur le cahier des charges (honeteté). Bien citer les ressources utilisés.	Etre capable de faire une Retour d'Expérience (REX) sur chaque partie du projet (technologie, gestion, front/back, groupe) et être capable d'attirer des leçons dès le prochaine projet.
Communication						
C11.2	Communication: Communiquer et convaincre en s'adaptant aux objectifs et contraintes [...]		Équipe pas préparé.	Avoir fait effort de preparation, mais presentation dehors le limites / contraintes données.	Être capable de présenter le projet dans les contraintes données.	Être capable de convaincre de la maîtrise de son projet (son valuer, les choix faites, etc).
Travail en équipe						
C04.2	Coordination et planification: Mettre en œuvre une méthodologie de projet, planifier votre travail [...]. Communiquer régulièrement la progrès auprès des acteurs (ex professeurs). Communiquer la progression à l'écrit.		Pas present en cours, pas de communication.	Communiquer dans la classe.	Communiquer dans la classe et mini-rapports envoyés.	Communiquer clairement (en verbal + mini-rapport) sur l'avancement et la répartition des tâches à chaque séance. Mettre-en-place de ceremonie de type agile et l'expliquer.
C04.9	DevOps: Gérer le cycle de vie logiciel tout au long des phases de planification, de développement, [...] selon les pratiques DevOps, et mettre en place des architectures orientées services		Absence des outils de gestion (code, taches).	Outils initiés mais peu utilisés.	Outils de gestion code+projet mis-en-place et utilisés régulièrement.	Avoir un git propre (brunches, ...) et connection avec Jira et tâches, mise en place d'un Docker.