SAE S2 – 2022-2023

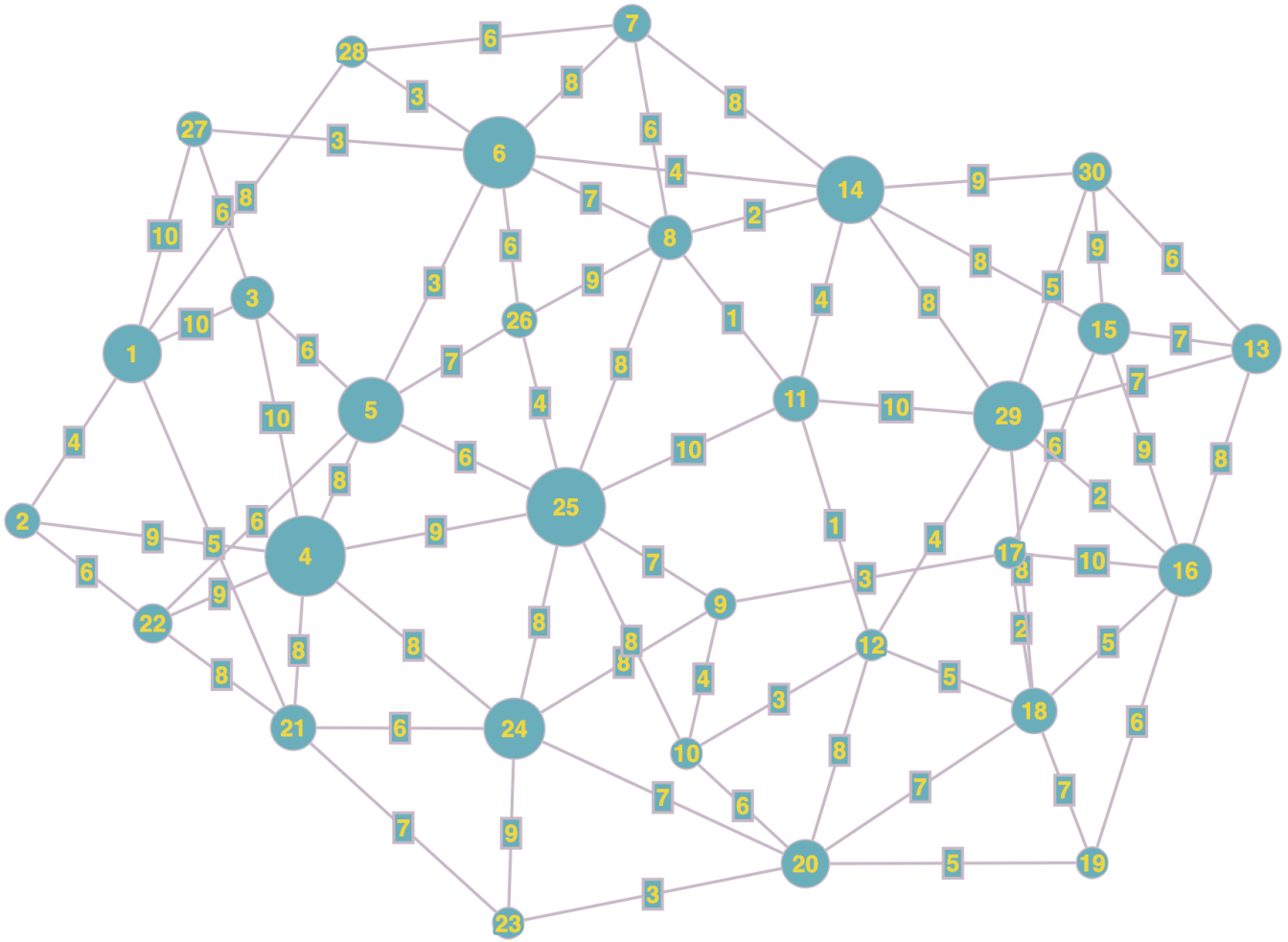
SAE 125 – Aide décisionnelle pour des Centres de Santé

3 SAE en une

- SAE 01 – Développement d’une application
  - Partir d'un besoin exprimé par un client nécessitant une interface graphique, formaliser les besoins, proposer une conception, implémenter et tester son développement.
- SAE 02 – Exploration algorithmique d'un problème
  - Approfondir la réflexion sur l'approche algorithmique des problèmes rencontrés pendant les phases de développement
- SAE 05 – Gestion de projet
  - Analyser les besoins de l'entreprise et rédiger un cahier des charges. Première familiarisation avec la conduite de projet à travers un sujet simple.

SAE de s2 : les modules concernés

But de la SAE : Aide décisionnelle pour la circulation entre des centres de santé



- **Java/UML** : 4h
- **ProgIHM** : 6h + 4h
- **Graphes/Maths** : heures de cours
- **SDD** : 4h
- **Qualité dev** : 2h + 4h
- **Gestion de projet** : heures de cours

24h de PE
(projet encadré)

Opérations sur un graphe

Description

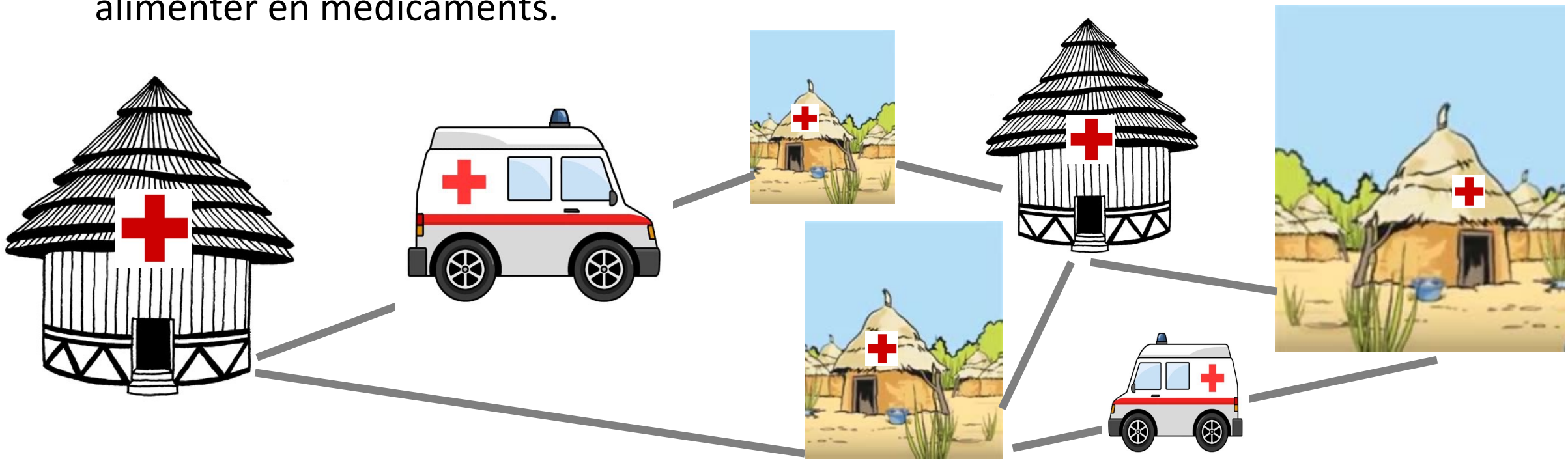
Le responsable des dispensaires de son district (plus de 30) doit acheminer des malades, des médecins/chirurgiens ou des médicaments, d'un hôpital à l'autre de façon régulière.

Dans un pays **sans infrastructure routière** développée, chaque trajet comporte des risques :

- Piste impraticable, pont infranchissable, vols de médicaments sur le chemin, attaques de gangs armés, etc

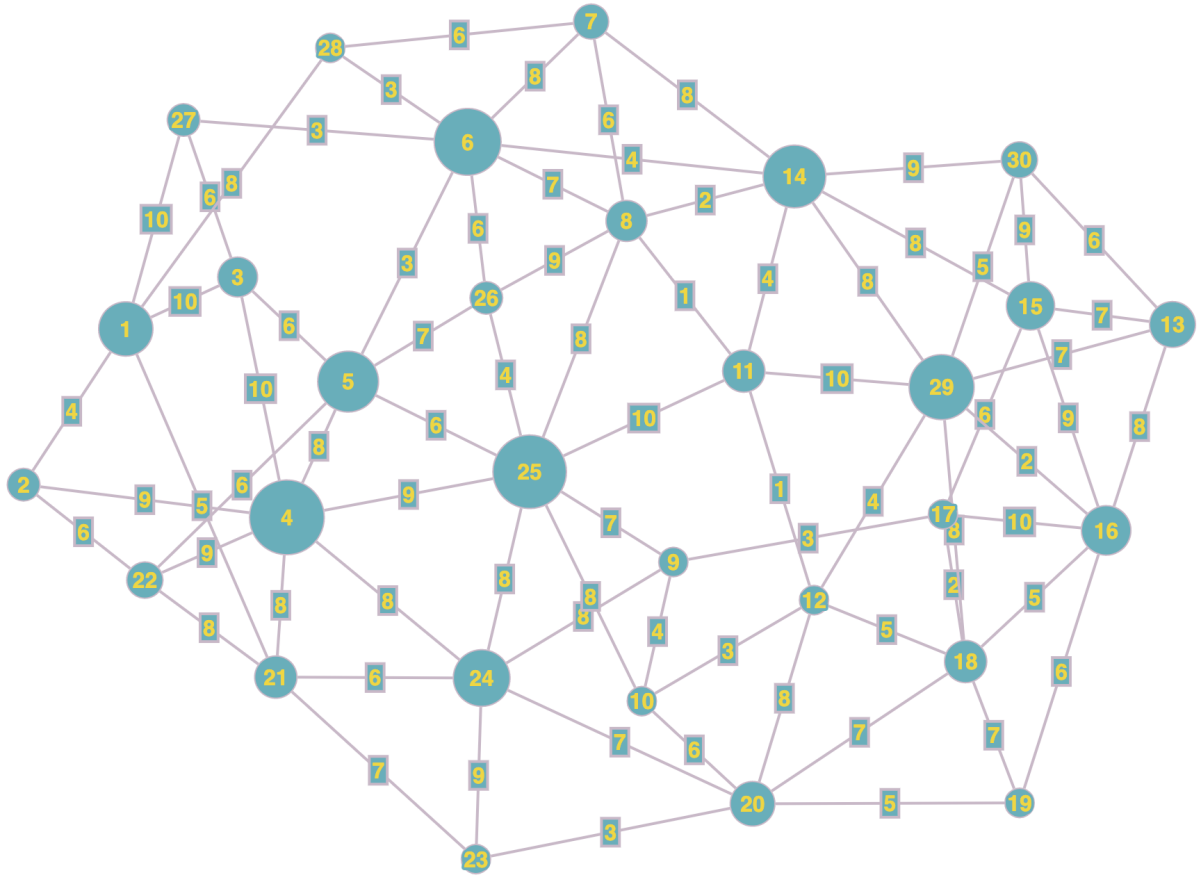
Objectif : Aider à la décision pour un responsable de plus de 30 établissements, via une interface IHM

- Il aimerait disposer **d'un logiciel d'aide à la décision** qui lui permette par exemple de :
  - Connaître les **meilleurs chemins** à emprunter, basés sur la **fiabilité** de chaque trajet qu'il est capable de connaître.
  - Il pourrait aussi vouloir connaître **le chemin le plus court** entre 2 points, en cas d'urgence vitale nécessitant une opération, par exemple.
  - Ou identifier le chemin **le plus court qui traverse tous les centres** pour les alimenter en médicaments.



- On supposera que les **trajets** se pratiquent dans les 2 sens.
- Tous sont effectués avec un **véhicule** (ambulance).
- Les trajets entre 2 dispensaires sont caractérisés par :
  - leur fiabilité (probabilité de bien arriver)
  - une distance en km,
  - une durée moyenne (minutes).

- Un **jeu de test** est fourni :
  - 30 établissements, 75 arêtes
  - Liste d'adjacence avec les valuations :
    - Fiabilité de l'arête
    - Distance
    - Durée



Graphe des dispensaires :

NŒUDS (dispensaire) : bloc opératoire (O), maternité (M), centre de nutrition (N)
LIENS (trajet en ambulance) : fiabilité, distance (km), durée moyenne (minutes)

Travail demandé

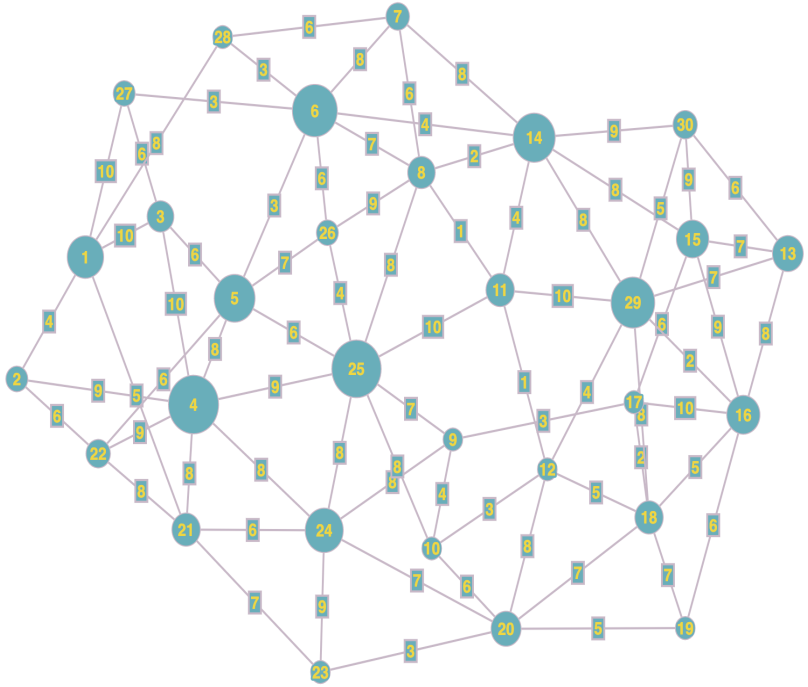
1. Construction du graphe des dispensaires

- En plus du **jeu d'essai fourni**, chaque binôme construit son propre graphe.

- Le graphe est **non-orienté**.

- Il doit être connexe et doit contenir au moins 30 nœuds, avec :

  - Nœuds : 3/5 de maternités, 1/5 de blocs opératoires et 1/5 de centres de nutrition
  - En option (multi-graphe) : possibilité d'avoir plusieurs liens entre 2 sites



2. Chargement du Graphe-MAP en mémoire et structures de données

- Le graphe doit être donné en fichier CSV représentant la **liste d'adjacence** du graphe + la liste des successeurs d'un sommet.
- Le fichier CSV proposé est décrit ainsi :

  // Numéro de sommet (au format Si)
  // Type de sommet (M, O ou N) ensuite répétitions :
  // Probabilité de fiabilité x10 : ex. 4 = proba de 0,4 ou 40%, si autre que zéro : c'est qu'il y a un trajet vers ce sommet
  // Distance en km,
  // Duree moyenne en minutes.

Caractéristiques de l'arête entre S1 et un autre sommet : fiabilité, distance en km, durée en minutes

Exemple :

S1 ; M; 4, 25, 50; 10, 31, 37; 0; 0; 0; 0; etc.
S2 ; M ; 4, 25, 50; 0; 9, 28, 37; 0; 0; 0; etc.

Type de sommet

- Vous pourrez ajouter des **noms** aux dispensaires (option).
- Vous pourrez choisir un autre format pour le fichier CSV de votre graphe.
- Le graphe doit être chargé en mémoire dans une structure de données dynamique (listes).

Travail demandé

3. Modifier le graphe via l'interface IHM :

- Faire évoluer la **fiabilité d'une arête** et la sauvegarder
  - Soit directement sur le graphe
  - Soit par rechargement du graphe (modification dans le fichier)

4. Afficher les éléments du graphe :

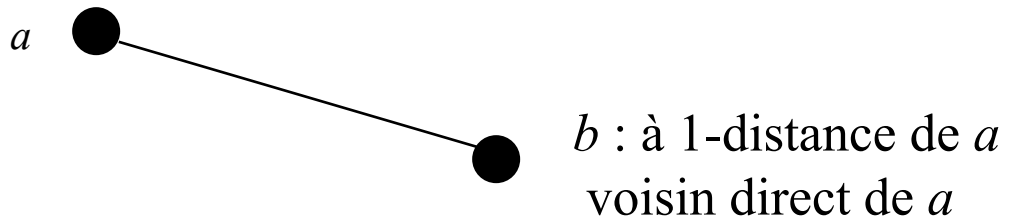
- Les dispensaires d'un type donné (ex. les maternités).
- Le décompte des nœuds du graphe par type (ex. : 12 maternités, 6 centres de nutrition, 12 blocs opératoires)
- Les trajets / arêtes **les plus risquées** du graphe (au-delà d'un seuil saisi auprès de l'utilisateur).

Travail demandé

5. Interroger le graphe-MAP via l'interface IHM pour :

5.1/ Analyser les éléments du graphe (1-distance) :

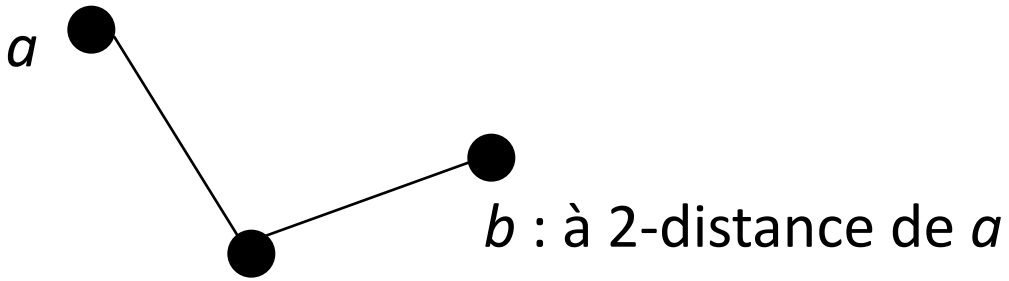
- Pour un nœud donné, possibilité de connaitre les voisins d'un type donné à 1-distance (voisin direct), par exemple :
- Quels sont les *maternités* voisines du dispensaire S (voisins directs) ?
- Quels sont les *centres de nutrition* voisins des centres S1 et S2 (voisins directs) ?
- Pour une arête donnée A1, quels sont les dispensaires reliés ?
- Pour 2 sommets donnés, lister les sommets voisins d'un type donné des centres S1 et S2
  - Par ex. : lister les blocs opératoires en voisins directs de S1 et S2.



Travail demandé

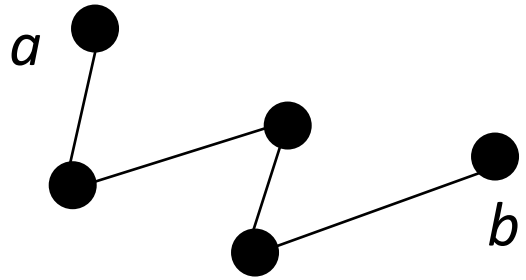
5.2/ **Analyse à 2-distance**

- Dire si deux dispensaires sont **reliés** à 2-distance ;
- **Comparer** 2 dispensaires A et B :
  - Pour chacun, dire lequel est le plus OPERATOIRE que l'autre (plus apte à opérer en cas d'accidents causant un grand nombre de victimes, càd. celui qui possède le plus de dispensaires ayant un Bloc opératoire à 2-distance),
  - Lequel est le plus NUTRITIONNEL (possède le plus de Centres de Nutrition à 2-distance) et lequel a le plus de MATERNITEs (à 2-distance aussi).
- Donner le chemin **le plus fiable** entre 2 sites choisis par l'utilisateur.
- Donner le chemin **le plus court** en kilomètres et en **durée**, entre 2 sites, sous forme de texte.



Des questions bonus sont proposées, exemple :

- Afficher **visuellement** le chemin **le plus court** en distance et en durée.
- Afficher le graphe et saisir les 2 points sur le graphe ; **afficher visuellement le chemin le plus fiable**.
- Pouvoir **déplacer dynamiquement** les nœuds du graphe, etc.
- Analyses à **p-distance**



Quelques questions d'algo/math, exemple :

- Quels sont les *centres de nutrition* voisins des centres S1 et S2 ?
- Proposez un algorithme pour trouver des **cycles hamiltoniens**

Les écrans à développer

- **Ecran accueil** : chargement du graphe
  - Informer si le graphe est bien chargé et mémoire ou pas
  - Visualiser le graphe, modifier des valeurs du graphe
- **Ecran 0** : affichage des éléments (0-distance)
- **Ecran 1** : voisinage direct (1-distance)
- **Ecran 2** : voisinage à 2 sauts (2-distance)
- **Ecran 3** : comparaison de sites à 2 sauts ou plus (≥ 2-distance)
- **Ecran 4** (bonus) : voisinage à p sauts et chemins (p-distance)

- Ces écrans font l'objet d'une **démonstration en fin de module DevIHM**.
- Dates de rendu : semaine du 19/06/2023

Liste des tâches avec les ressources concernées :

- Analyser le sujet (**UML/JAVA**)
- Établir un cahier des charges (**GPI**)
- Comprendre les notions de base sur les graphes (**Maths**)
- Proposer des structures de données pour le Graphe-MAP (**SDD**).
- Chargement du graphe et remplissage des structures de données (**SDD,JAVA**)
- Développement algorithmique (**SDD, JAVA**)
- Développer les différentes interfaces (**IHM, JAVA**)
- Qualité de conception, développement et tests (**Qualité Dev**)

Organisation

- Projet en **binômes**
- **Déroulement** : travailler sur TOUT le projet durant les séances dédiées des modules concernés. L'encadrant peut vous aider pour des questions relatives à sa matière. Une partie s'effectue en totale autonomie.

***En gras** : ceux sur les PC de l'IUT*

Outils utilisables gratuits

- Gestion de projet et *versionning* de code :
  - Trello, **GitLab** (forge Lyon1)
- Outils de maquettage
  - Figma, **Wireframesketcher**, Balsamiq Wireframes, https://cacoo.com
- AGL (Atelier de Génie Logiciel)
  - **PowerAMC**, Astah UML https://astah.net , Visual Paradigm
- IDE (Environnement de Développement)
  - **Netbeans** sera utilisé en Java et IHM Java

Evaluation

- On pourra **individualiser** les notes au sein d'un même binôme
- L'application fera l'objet d'une démonstration et de questions aux examens du R2-02 - Dev d'apps IHM (dernières séances) : *noteDémo* ;
- La conception, la qualité algorithmique et l'implémentation seront évaluées durant les cours et examens des modules concernés (*noteSDD*, *noteJava* et *noteMaths*) ;
- La qualité développement du projet sera évaluée à partir de questions posées aux examens du cours R2-03 : *noteQualité*
- La Gestion de Projet sera évaluée par la qualité du livrable du cours R2-10 - Gestion de projet : *noteGPI*

Une note par SAE :

SAE s2-01 - Réalisation / Qualité logicielle	moyenne pondérée de : *noteDémo* (coeff. 3), *noteJava* (1), *noteQualité* (1)
SAE s2-02 -Optimisation : implémentation algorithmique	moyenne de *noteSDD* et *noteMaths*
SAE s2-05 - Gestion de Projet	*noteGPI*