# Plateforme tutorat

## TIP 2025

*Anonymisé*

INSA Rouen Normandie

5 juin 2025

# Fonctionnement global

- **Import des utilisateurs** dans la base de données
- **Création d'une campagne** de tutorat (par semestre)
- **Inscription des utilisateurs** à la campagne
- **Saisie des disponibilités** par les participants
- **Génération automatique** des affectations via l'interface
- **Consultation des affectations** sur les espaces personnels
- **Saisie des heures réalisées** par les tutorés
- **Suivi en temps réel** des heures par le responsable
- **Clôture de la campagne** avec toutes les données disponibles

# Structuration du projet : côté client

- Interface web développée avec **Nuxt (Vue.js)**
- Génération de fichiers HTML/CSS/JS optimisés
- Navigation fluide et logique interface séparée du serveur
- Support SSR/SSG (rendu serveur ou statique)
- Utilisation de concepts Vue : composants, props, événements, cycle de vie, stores (Vuex / Pinia)
- Appels structurés à l'API via différentes routes/méthodes

## Utilisation de Vue

Vue est un framework JavaScript qui se repose sur les standards HTML, CSS et JavaScript. Il propose une manière efficace de déclarer des composants pour la construction d'interfaces utilisateur de toute complexité. (vuejs.org)

# Structuration du projet : côté serveur

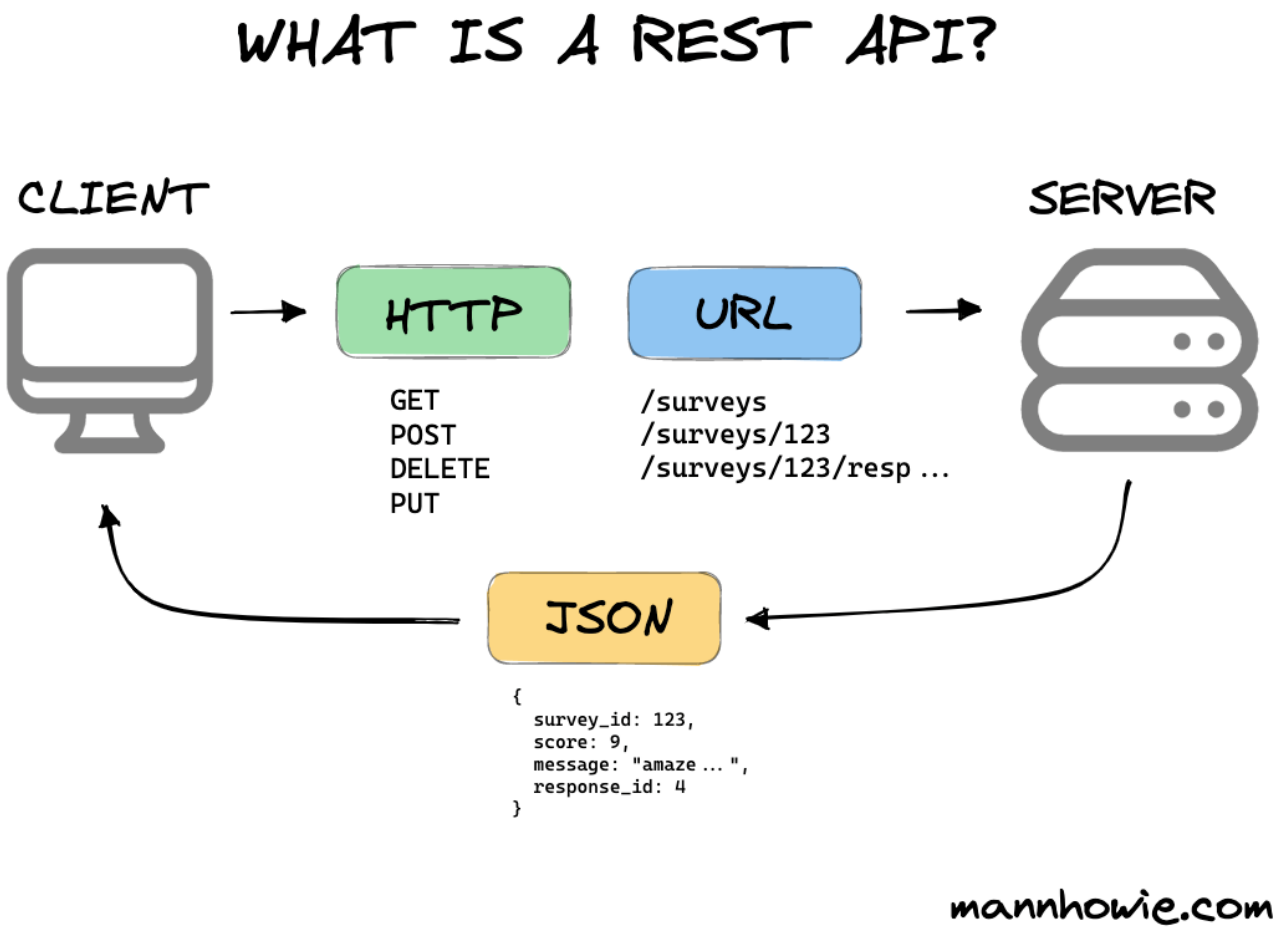



Figure – Qu'est-ce qu'une API RESTful ?

# Structuration du projet : côté serveur

```
POST   /auth/login
POST   /auth/send-link
GET    /auth/self
GET    /auth/logout
GET    /campaign/:campaignId/agenda
GET    /campaign/:campaignId/availabilities
POST   /campaign/:campaignId/availabilities
GET    /campaign/:campaignId/subjects
GET    /campaign/:campaignId/tutee/registrations
POST   /campaign/:campaignId/tutee/registrations
GET    /campaign/:campaignId/tutor/registrations
POST   /campaign/:campaignId/tutor/registrations
...
```

Table – Liste des routes et méthodes disponibles sur l'API REST

# Structuration du projet : côté serveur

- API REST : routes `GET`, `POST`, `PATCH`, `DELETE`
- Gestion des données via l'ORM **Gorm**
- Paramètres dynamiques (`:campaignId`, etc.)
- Authentification par **session via cookies sécurisés**
- Middleware pour contrôle d'accès et gestion des erreurs
- Authentification par e-mail ou intégration CAS (INSA)

## Cookie

Un cookie [...] est une petite quantité de données échangées entre un serveur HTTP et un client HTTP, et qui permet de créer une session avec état lors de la visite d'un site Web. (fr.wikipedia.org)

# Disponibilité des étudiants

- Optimiser la mise en relation entre tuteurs et tutorés selon leurs disponibilités au cours du semestre
- Utilisation des emplois du temps via la plateforme `https://agendas.insa-rouen.fr`
- Développement d'une fonction de score pour représenter la compatibilité entre deux agendas

# Récupération des agendas

- Source principale : `https://agendas.insa-rouen.fr`
- Format unique disponible : flux `RSS` (XML)
- Extraction automatisée via un `RSS parser`

## Extrait RSS :

```
<item>
<guid isPermaLink="false">https://agendas.insa-rouen.fr:443/day.php</guid>
<title>Mar Mai 20 2025: STPI22-M7-TD-3</title>
<ev:startdate>2025-05-20T15:00:00</ev:startdate>
<ev:enddate>2025-05-20T16:30:00</ev:enddate>
<link>https://agendas.insa-rouen.fr:443/day.php?date=20250520</link>
<description>Mar Mai 20 2025 15:00: <br/><br/>STPI22-M7-TD-03<br/>ROUXELIN Nathan<br/>STPI2<br/>(Exporté
↪  le:20/05/2025 21:30)<br/></description>
<ev:location>DU-B-RJ-01</ev:location>
</item>
```

# Saisie des disponibilités

- Interface utilisateur intuitive
- Affichage des créneaux déjà occupés
- Saisie manuelle des disponibilités restantes

**Format JSON stocké :**

```
{"1": [3,13,0,0,13,10,0], "2": [11,15,1,0,14,9,0]} // ...
```

- Clé = jour (1 = lundi)
- Valeur = liste des statuts par créneau

# Représentation des créneaux

**Format JSON stocké :**

```
{"1": [3,13,0,0,13,10,0], "2": [11,15,1,0,14,9,0]} // ...
```

**Valeur d'un créneau $c$ ($\in [-1, +\infty[$) :**

- $c = -1$ : indisponible
- $c = 0$ : disponible
- $c > 0$ : nombre de cours sur le semestre

**But :** Maximiser la compatibilité entre deux agendas

# Première approche : score par fraction

$$V(c) = \begin{cases} \frac{1}{1+c} & \text{si } c \geq 0 \\ 0 & \text{sinon} \end{cases}$$

**Score d'un créneau :** $V(c_1) \cdot V(c_2)$

**Exemples :**

- $c_1 = 0,\ c_2 = 0$ : score $= 1$
- $c_1 = 2,\ c_2 = 1$ : score $= 1/6$

## Problème de l'approche

Les scores sont trop faibles même pour de faibles contraintes

# Amélioration : score exponentiel

$$V(c) = \begin{cases} e^{-c/8} & \text{si } c \geq 0 \\ 0 & \text{sinon} \end{cases}$$

**Score d'un créneau :** $V(c_1) \cdot V(c_2)$

**Exemples :**

- $c_1 = 2,\ c_2 = 4$ : score $\approx 0.48$
- $c_1 = 8,\ c_2 = 16$ : score $\approx 0.06$

**Avantages :**

- Score élevé pour peu de contraintes
- Décroissance rapide avec augmentation de $c$

# Moteur d'affectation : contexte

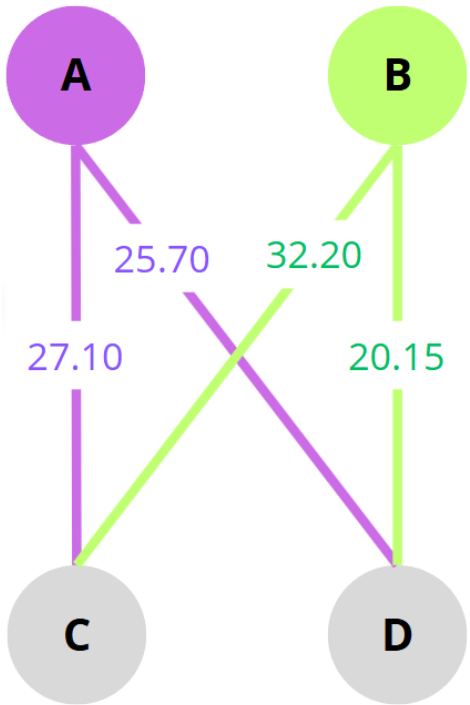
- Les tutorés ont formulé leurs voeux
- Les tuteurs ont saisi leur matières avec un quota
- Les étudiants ont tous saisi leurs disponibilités
- Objectif : maximiser les créneaux communs tuteur-tutoré

**Problème**

Comment réaliser l'affectation des tuteurs et tutorés ? Quel algorithme mettre en place ?

# Première approche : algorithme glouton

- Choix localement optimal pour chaque tuteur
- Ici, choix optimal = le meilleur score de compatibilité
- Résultat parfois sous-optimal à l'échelle globale




**Glouton**

A C
27.10

B D
20.15

Σ = 47.25

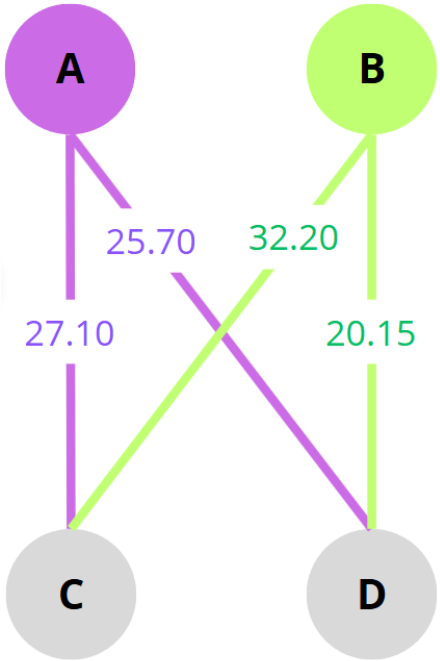
**Idéal**

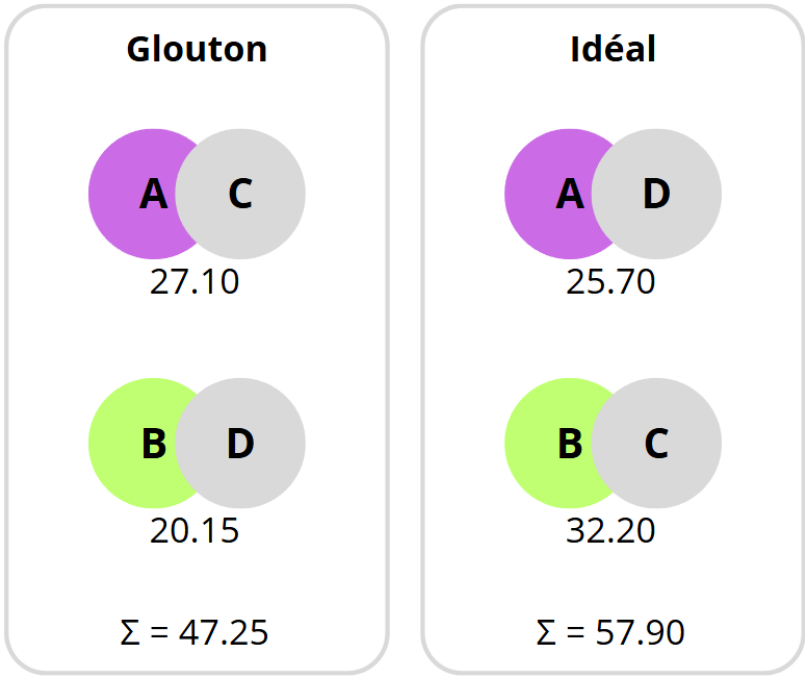
A D
25.70

B C
32.20

Σ = 57.90

# Limite du glouton : exemple




- A → C (score 27.10), B → D (score 20.15)
- Mais B → C aurait été meilleur (32.20)

## Problème

On a effectué une attribution non-optimale dans le sens où le tutoré C aurait préféré être avec le tuteur B. Comment y remédier ?

# Problème des appariements stables

- Problème classique en algorithmique : "mariage stable"
- Deux ensembles de taille égale avec préférences
- Un appariement est **stable** s'il n'existe pas de paire instable.

**Définition**

Une paire instable est un duo (A, B) qui préférerait être ensemble plutôt qu'avec leur partenaire actuel.

- Possède une solution : l'algorithme de Gale et Shapley

# Algorithme de Gale et Shapley

- Algorithme à la recherche d'une solution **stable** (aucun binôme ne souhaite changer de partenaire) mais pas forcément **optimale** (maximisant les scores)
- Utilisé par Parcoursup et le SCEI
- Reprise d'une implémentation existante en Go : github.com/yutohub/matching
- Exécuté pour chaque matière

## Problème (encore)

Cet algorithme cherche à associer de façon bijective les éléments de A avec ceux de B. Dans notre cas, dim $A$ n'est pas forcément égale à dim $B$.

# Problème des quotas

Soit $n_{tuteurs}$ le nombre de places proposés par les tuteurs et $n_{tutores}$ le nombre de demandes pour du tutorat en cette matière.

- $n_{tutores} > n_{tuteurs}$ : **impossible** de réaliser des affectations
- $n_{tutores} = n_{tuteurs}$ : affectation possible, algorithme classique
- $n_{tutores} < n_{tuteurs}$ : possible, algorithme à modifier

# Problème des quotas

Soit $n_{tuteurs}$ le nombre de places proposés par les tuteurs et $n_{tutores}$ le nombre de demandes pour du tutorat en cette matière.

- $n_{tutores} > n_{tuteurs}$ : **impossible** de réaliser des affectations
- $n_{tutores} = n_{tuteurs}$ : affectation possible, algorithme classique
- $n_{tutores} < n_{tuteurs}$ : possible, algorithme à modifier

- Création de tutorés fictifs avec un score de -1
- Tous les cas sont donc explorés

# Problème des quotas

Soit $n_{tuteurs}$ le nombre de places proposés par les tuteurs et $n_{tutores}$ le nombre de demandes pour du tutorat en cette matière.

- $n_{tutores} > n_{tuteurs}$ : **impossible** de réaliser des affectations
- $n_{tutores} = n_{tuteurs}$ : affectation possible, algorithme classique
- $n_{tutores} < n_{tuteurs}$ : possible, algorithme à modifier

- Création de tutorés fictifs avec un score de -1
- Tous les cas sont donc explorés

## Détail technique

Tutorés factices pointent vers `NIL`, les réels vers leur valeur réelle.

# Conclusion

- Côté algorithmique intéressant
- Problèmes auxquels je ne pensais pas être confronté
- Élaboration d'une solution en autonomie
- Côté front-end moins intéressant, assez répétitif
- Rendu final correspondant au cahier des charges
- Estimation horaire respectée (49h prévu, 49h33 réel)
- Fonctionnalités optionnelles non implémentées, dépassement 50h