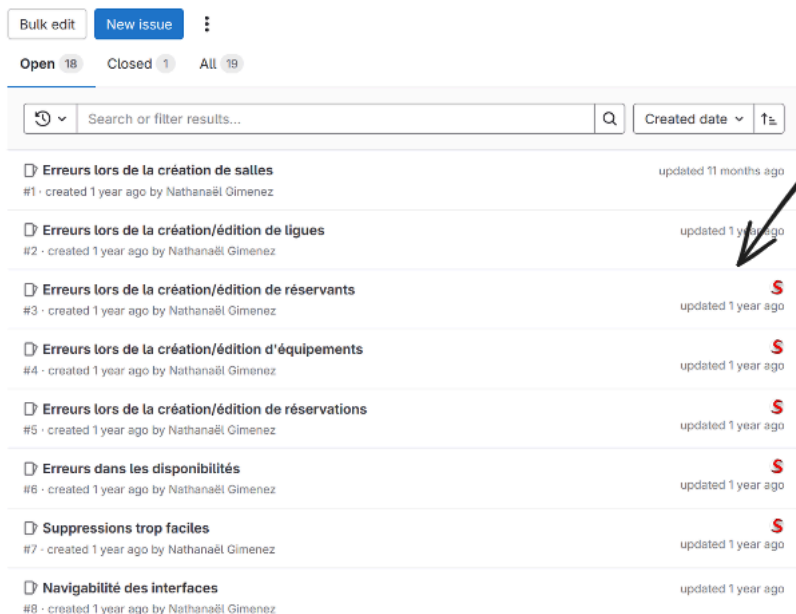


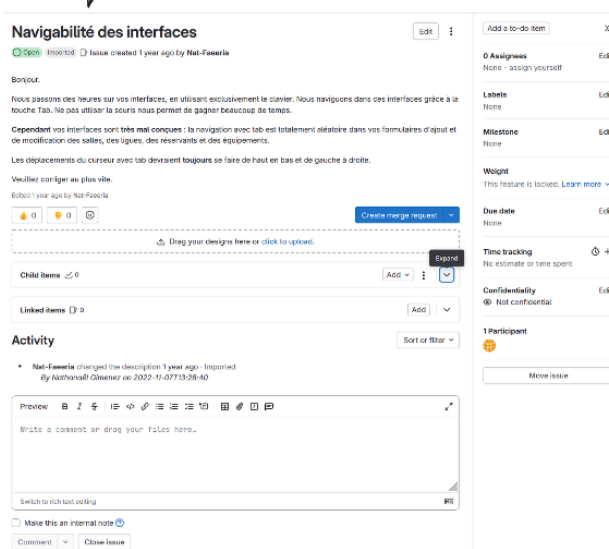
## Ticketing avec Gitlab

### 1. Qu'est-ce qu'un ticket ?



Voici une liste de tickets, parfois appelés Issues

... et voilà un ticket !



Un ticket sert à exprimer :

un nouveau besoin  
(nouvelle fonctionnalité par exemple)

un problème/bug rencontré

les tickets peuvent être créés par

Des développeurs.euses  
qui travaillent sur le projet  
(y compris nous-même !)

Des utilisateurs  
quand ils rencontrent  
un problème ou voudraient  
une nouvelle fonctionnalité

Le client  
(le plus souvent,  
des utilisateurs  
testeurs de l'organisation  
cliente)

Un ticket est un outil de communication !

Il permet de communiquer (à soi-même ou aux autres) :

- ce qui a été fait
- ce qui reste à faire
- ce qu'on voudrait faire en plus
- etc.

## Analysons la composition d'un ticket :

1.

Un titre clair et descriptif

Un statut : ouvert, fermé, en cours, validé, ...

Un contenu qui décrit le problème rencontré (et comment le reproduire) ou la fonctionnalité souhaitée (et ses spécifications)

Un ticket est un outil de communication : il doit au minimum permettre d'écrire des commentaires pour approfondir le besoin ou le problème en posant des questions, ou pour discuter avec d'autres développeurs de l'activité réalisée pour valider ce ticket

Le contenu s'écrit en Markdown, il peut donc être très élaboré (avec tableaux, images, émojis, lien, etc.)

2.

**Navigabilité des interfaces**

Open Imported Issue created 1 year ago by Nat-Faeria

Bonjour.

Nous passons des heures sur vos interfaces, en utilisant exclusivement le clavier. Nous naviguons dans ces interfaces grâce à la touche Tab. Ne pas utiliser la souris nous permet de gagner beaucoup de temps.

Cependant vos interfaces sont très mal conçues : la navigation avec tab est totalement aléatoire dans vos formulaires d'ajout et de modification des salles, des ligues, des réservants et des équipements.

Les déplacements du curseur avec tab devraient toujours se faire de haut en bas et de gauche à droite.

Veuillez corriger au plus vite.

Edited 1 year ago by Nat-Faeria

👍 0 🙏 0 🗨️ 0

Drag your designs here or click to upload.

Child items 0 Add

Linked items 0 Add

Activity

Nat-Faeria changed the description 1 year ago - Imported  
By Nathanaël Gimenez on 2022-11-07 13:28:40

Preview B I S L E P U T Write a comment or drag your files here...

Switch to rich text editing

Make this an internal note

Comment Close issue

Add a to-do item

0 Assignees None - assign yourself Edit

Labels None Edit

Milestone None Edit

Weight This feature is locked. Learn more

Due date None Edit

Time tracking No estimate or time spent +

Confidentiality Not confidential Edit

1 Participant

Move issue

Un ticket peut être assigné à une personne particulière

Un ticket peut aussi avoir des labels, pour différencier son type par exemple (problème, évolution, bugfix, etc.)

Toute cette zone est relative à la gestion de projet : on peut déterminer le temps que va prendre un ticket (ou son poids), la date pour laquelle la correction est prévue, l'étape (milestone) dans laquelle le code résultant de ce ticket sera mis en production, et le temps passé sur le ticket

Parfois, valider un ticket demande beaucoup d'efforts. Il vaut alors souvent mieux le découper en tâches plus petites, pour plus de lisibilité et de clarté

Un ticket peut être relié à d'autres tickets : soit parce qu'ils sont similaires, soit parce qu'ils sont dépendants l'un de l'autre

Un ticket, c'est de la communication : client-développeur.euse, développeur.euse-développeur.euse ... mais aussi avec soi-même ! (pour se souvenir d'où on en est, ou planifier plus efficacement !)

## Processus de gestion des tickets

### Travail à faire :

A partir de vos connaissances et de votre logique, vous devez reconstituer un processus simple de gestion de tickets (dans le cadre d'un support informatique ou d'une équipe de développement)

Pour ce faire, suivez ces étapes :

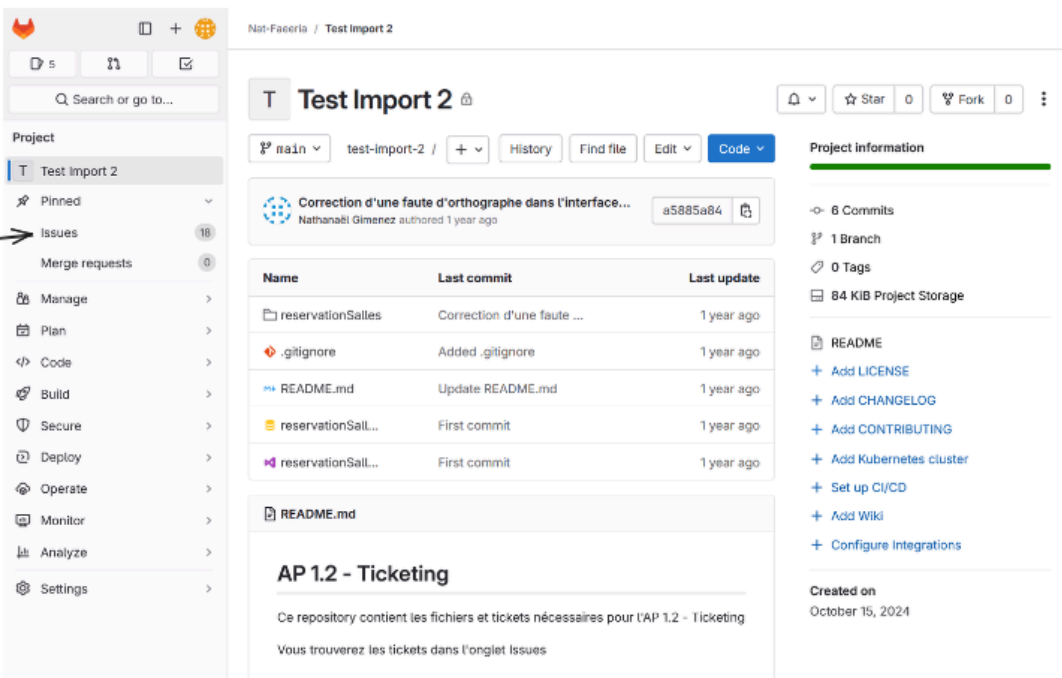
1. Allez sur le site [excalidraw.com](https://excalidraw.com)
2. Ouvrez-y le fichier Modélisation-processus-ticketing.excalidraw qui vous a été fourni
3. Suivez les consignes contenues dans le fichier

## 2. Visualiser des tickets avec Gitlab

Dans Gitlab, les tickets sont appelés **Issues**. Pour les visualiser, il faut :

Cliquer ici ...

... puis là



The screenshot shows the GitLab web interface for a project named 'Test Import 2'. On the left sidebar, the 'Issues' tab is selected, indicated by an arrow and the text 'Cliquer ici ...'. Below it, another arrow points to the 'Issues' tab with the text '... puis là'. The main content area shows a list of issues. The first issue is titled 'Correction d'une faute d'orthographe dans l'interface...' and is highlighted. The right sidebar shows project information, including '6 Commits', '1 Branch', and '0 Tags'.

Name	Last commit	Last update
reservationSalles	Correction d'une faute ...	1 year ago
.gitignore	Added .gitignore	1 year ago
README.md	Update README.md	1 year ago
reservationSall...	First commit	1 year ago
reservationSall...	First commit	1 year ago

**AP 1.2 - Ticketing**

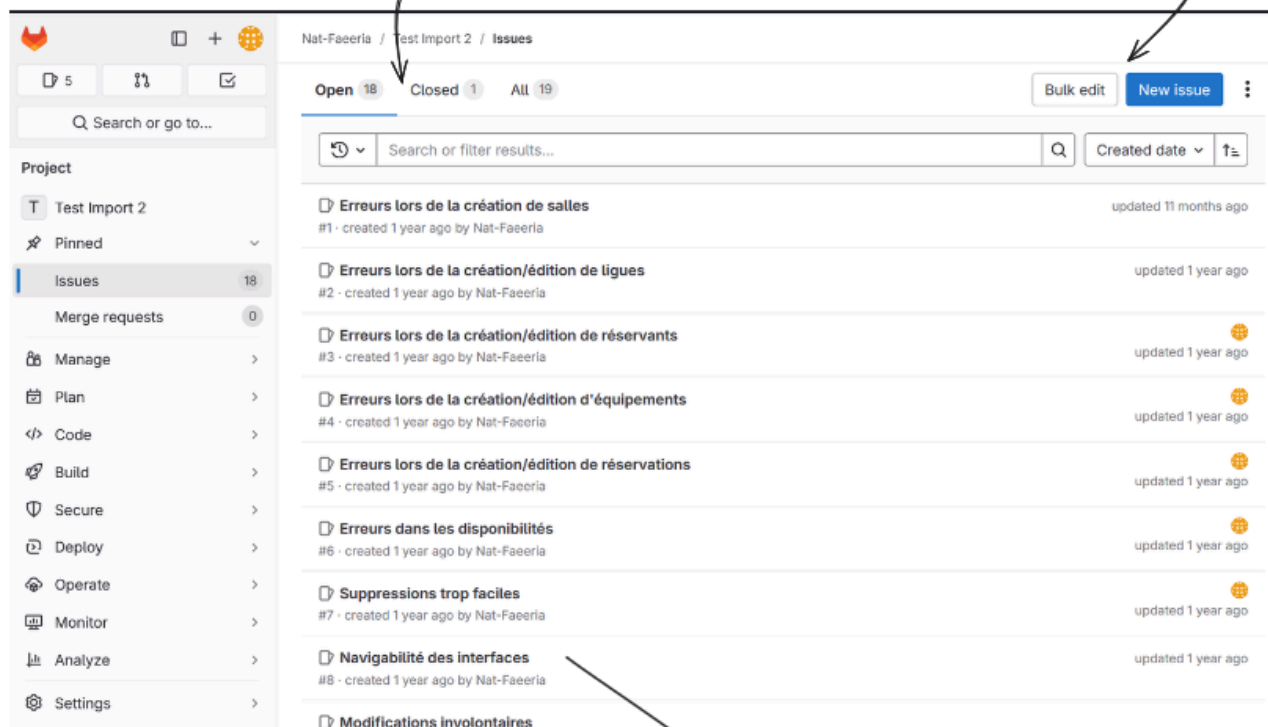
Ce repository contient les fichiers et tickets nécessaires pour l'AP 1.2 - Ticketing

Vous trouverez les tickets dans l'onglet Issues

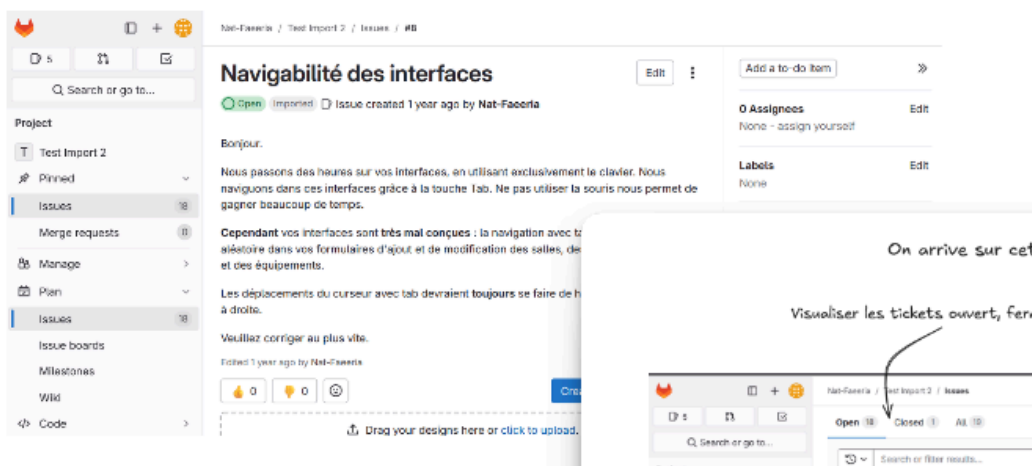
On arrive sur cette page où on peut :

Visualiser les tickets ouvert, fermés (ou tous)

Créer un nouveau ticket



Consulter un ticket en cliquant dessus



### 3. Gérer des tickets avec Gitlab

Pour créer un ticket il faut :

The screenshot shows the 'New Issue' form in GitLab. Handwritten annotations provide guidance on how to fill out the form:

- Title (required):** An arrow points to the title field with the note: "Lui donner un titre clair et explicite".
- Type:** An arrow points to the 'Issue' dropdown menu with the note: "Préciser son type : dans Gitlab on peut utiliser le type Issue pour le tout-venant et Incident pour les rapports d'erreur".
- Description:** An arrow points to the description text area with the note: "Décrire la demande de manière précise et détaillée. Si c'est une demande d'évolution, il faut être clair sur les spécifications attendues".
- Assignee:** An arrow points to the 'Assignee' dropdown menu with the note: "On peut affecter l'Issue à un membre de l'équipe, et même préciser une date limite de traitement".
- Due date:** An arrow points to the 'Due date' field with the note: "Si c'est un rapport d'erreur, il faut expliquer l'erreur en détail ainsi qu'une procédure permettant de la reproduire".

On peut appliquer un Label à l'Issue, pour mieux la qualifier : correction, évolution, critique, à valider, ...

The screenshot shows the 'Select label' dialog box. Handwritten annotations explain the options:

- Select label:** An arrow points to the dropdown menu.
- Create project label:** An arrow points to the 'Create project label' button with the note: "En cliquant ici on peut ajouter de nouveaux labels (et leur affecter une couleur)".
- Manage project labels:** An arrow points to the 'Manage project labels' button with the note: "En cliquant ici on arrive sur une page de gestion des labels (modification, suppression, etc.)".

Quand on accède à un ticket, on peut :

The screenshot shows a GitLab issue page for 'Nat-Faeria / Test Import 2 / Issues / #8'. The issue title is 'Navigabilité des interfaces'. The issue is 'Open' and was created 1 year ago by 'Nat-Faeria'. The description discusses navigation difficulties using the Tab key. The right sidebar contains fields for 'Assignees', 'Labels', 'Milestone', 'Weight', 'Due date', 'Time tracking', 'Confidentiality', and 'Participant'. The bottom section includes a comment area and a 'Close issue' button.

**Annotations:**

- Ajouter des sous-tâches pour mieux organiser son travail** (Add sub-tasks to better organize work) - points to the 'Child items' section.
- L'assigner à quelqu'un** (Assign it to someone) - points to the 'Assignees' field.
- Modifier ses labels** (Modify its labels) - points to the 'Labels' field.
- Modifier sa date limite** (Modify its due date) - points to the 'Due date' field.
- Ajouter un commentaire en l'écrivant puis en cliquant ici** (Add a comment by writing it then clicking here) - points to the comment area.
- Fermer le ticket** (Close the ticket) - points to the 'Close issue' button.

#### 4. Gérer son travail avec Gitlab

Dans l'onglet latéral, on remarque un menu "Planification"

Si on sélectionne "Issue boards"

On arrive sur une interface permettant de gérer les Issues dans des tableaux/listes

On peut faire glisser une Issue ...

... et la déplacer dans une autre liste.

Le label de l'Issue est automatiquement mis à jour

On peut créer une liste par label



Cette interface nous permet de faire de la gestion de projet directement dans Gitlab, sans avoir besoin de passer par un autre outil (comme Trello par exemple)

Quelques avantages de cette approche :

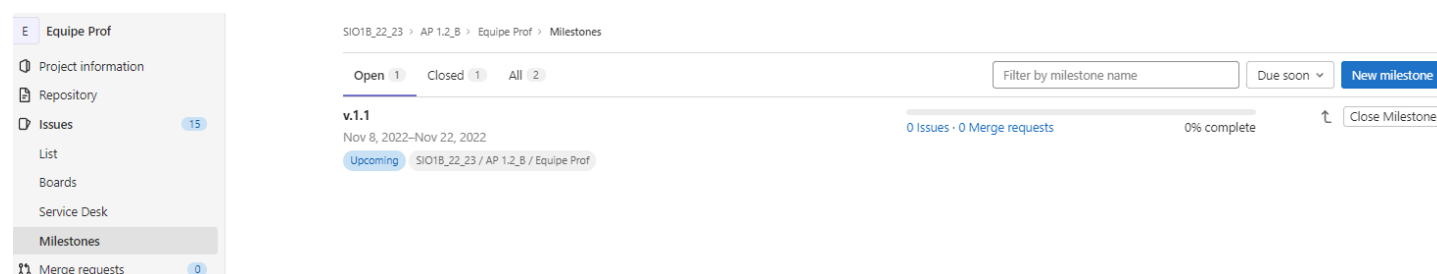
- un seul outil pour tout gérer : le code, les tickets, la gestion de projet
- pas besoin de recopier les tickets à traiter dans un autre outil

Quelques inconvénients de cette approche :

- fonctionnalités limitées (ou obligation de payer)
- les tâches créées dans les Issues ne sont pas prises en compte

## 5. ■ Gérer des milestones avec Gitlab

Pour gérer des milestones avec Gitlab, il suffit d'aller dans le menu de gauche, *Issues* > *Milestones*



Cette interface permet de voir les milestones/étapes existantes/en cours (*Open*), et celles déjà accomplies (*Closed*)

Quand on sélectionne une milestone, on peut visualiser tous les tickets affectés à cette milestone. On peut aussi définir la date à laquelle la milestone doit commencer, et la date à laquelle elle doit se finir/être livrée.

## 6. ■ Pourquoi utiliser ces outils ?

Parce que ces outils vont nous permettre d'avoir une gestion plus professionnelle de notre code :

- Avec le ticketing, nous pouvons **communiquer avec le client et les utilisateurs**. Les développeurs sont prévenus au plus vite des problèmes sur l'application, le client peut plus facilement exprimer ses besoins et les problèmes rencontrés, l'application devient donc de plus en plus fonctionnelle et adaptée aux besoins du client
- Avec l'outil de Boards, un chef de projet ou un client (*selon les paramètres de visibilité affectés*) peut voir **l'avancée du travail** d'un simple coup d'œil. La gestion est très automatisée et permet d'éviter de perdre du temps à recopier des tâches sur Trello à la main, puis de mettre à jour leur statut à la main (en les déplaçant de colonne en colonne)
- Dans la version payante de Gitlab, on peut suivre le **temps passé sur chaque tâche** ou ticket, on peut affecter un poids (*complexité, temps estimé, ...*) à chaque tâche ou ticket et suivre le temps vraiment passé. Ces notions sont essentielles à la gestion de projet !
- Avec l'outil de Milestone, le client et les utilisateurs peuvent voir ce qui est **prévu pour les versions suivantes** de l'application. Ils peuvent prévoir en avance les modifications qu'il faudra faire sur leur SI pour accueillir la nouvelle version de l'application, ou savoir quand leur problème sera résolu.

- Gitlab propose beaucoup d'autres outils permettant d'augmenter la **qualité du code** (*tests automatisés, tests de sécurité, etc.*) ou de simplifier la vie des développeur ou du client : par exemple, Gitlab peut **déployer automatiquement** la nouvelle version d'un projet vers l'environnement du client ou vers un environnement de recette (*tests utilisateurs côté client*), ou alors proposer un lien de téléchargement de l'application au client, etc.

Ces fonctionnalités ne sont pas exclusives à Gitlab : tous les outils modernes de développement proposent maintenant ces fonctionnalités, souvent sous le nom de fonctionnalités **DevOps**. Elles sont parfois entièrement intégrées (*Gitlab, Github, outils Amazon Web Service ou Google Cloud, ...*), ou parfois partiellement intégrées (*Jira, etc.*) Dans d'autres cas, l'organisation qui développe le projet déploiera ses propres outils : Jenkins, Redmine, Mantis, ...