Intégration chat externe

? Salut l'agent ! Briefing — Mod Matrix pour Minecraft 1.20.1 Forge

Ce document est un condensé de la conception réalisée avec un autre agent Claude. Bonne lecture et bon courage pour le dev ! □□

? Objectif du projet

Créer un **mod Forge 1.20.1** qui intègre un client **Matrix** directement dans Minecraft :

- Remplace/enrichit le chat vanilla MC avec des salons Matrix
- Interface GUI dédiée (keybind `M`) pour switcher de salon
- Le salon Matrix actif s'affiche dans le chat vanilla dynamiquement
- HUD minimaliste indiquant le salon actif (ex: `[#général]`)

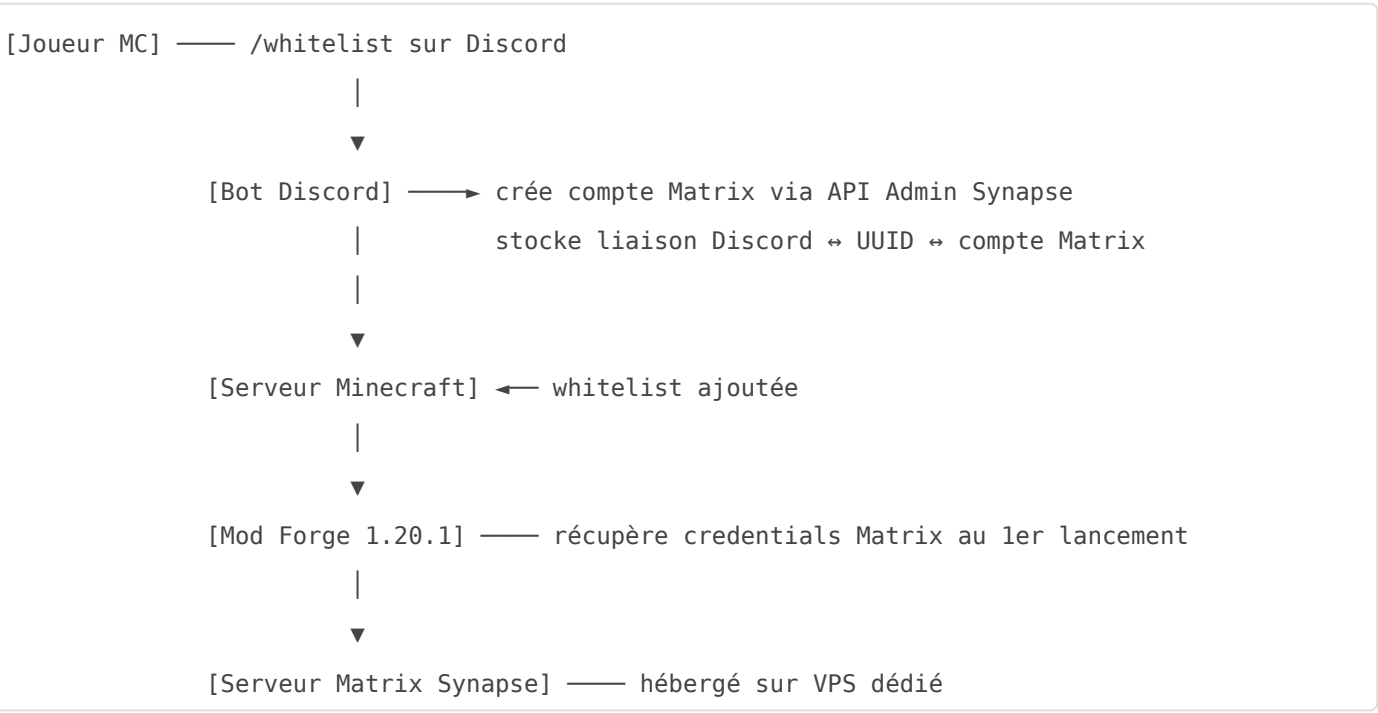
Le mod sera **publié sur CurseForge**, usage sur un serveur Minecraft privé (whitelist).

? Fonctionnalités retenues

Feature	Statut
Authentification Matrix (token)	□ À implémenter
Sync temps réel des rooms	□ À implémenter
Affichage messages dans chat MC	□ À implémenter
Envoi de messages	□ À implémenter
Switch de salon via GUI	□ À implémenter
Filtrage des rooms par Space Matrix	□ À implémenter

Feature	Statut
Notifications in-game (toast/HUD)	□ À implémenter
VoIP / Appels	□ Hors scope
Envoi de fichiers/images	□ Hors scope

?? Architecture globale



Points clés de sécurité

- Le **token d'enregistrement Synapse n'est jamais dans le mod** (mod public sur CurseForge)
- La création de compte Matrix est déléguée au **bot Discord** (point d'entrée unique)
- Le bot Discord détient seul les droits admin Synapse

?? Serveur Matrix Synapse

- **Hébergement** : VPS Ubuntu 22.04/24.04 dédié (4 cœurs, 8 Go RAM, 75 Go NVMe)
- **Domaine** : `matrix.protosrv.fr`
- **Reverse proxy** : Nginx + Let's Encrypt (Certbot)
- **Base de données** : PostgreSQL (pas SQLite, anticipation de la charge)
- **Fédération** : désactivée (serveur privé, port 8448 non exposé)

- **Enregistrement** : `registration_requires_token: true` — tokens gérés par le bot Discord
- **Synapse Admin** : déployé dans Docker, accessible sur `https://matrix.protosrv.fr:8443` (accès restreint par IP)
- **Compte admin** : `@protomehka-admin:matrix.protosrv.fr` — seul compte admin du serveur

Config homeserver.yaml retenue

```
pid_file: "/var/run/matrix-synapse.pid"
server_name: "matrix.mondomaine.fr"

listeners:
  - bind_addresses:
      - "127.0.0.1"
    port: 8008
    type: http
    tls: false
    x_forwarded: true
    resources:
      - names: [client]
        compress: false

database:
  name: psycopg2
  args:
    user: synapse_user
    password: CHANGEME
    dbname: synapse
    host: localhost
    cp_min: 5
    cp_max: 10

log_config: "/etc/matrix-synapse/log.yaml"
signing_key_path: "/etc/matrix-synapse/homeserver.signing.key"
media_store_path: /var/lib/matrix-synapse/media
max_upload_size: 1M

federation_domain_whitelist: []
allow_public_rooms_over_federation: false
allow_public_rooms_without_auth: false
```

```
enable_registration: true
enable_registration_without_verification: true
registration_requires_token: true

trusted_key_servers: []
suppress_key_server_warning: true

event_cache_size: 10K
caches:
  global_factor: 1.0

enable_metrics: false
report_stats: false
user_directory:
  enabled: false
url_preview_enabled: false

rc_message:
  per_second: 0.5
  burst_count: 10
rc_login:
  address:
    per_second: 0.1
    burst_count: 3
  account:
    per_second: 0.1
    burst_count: 3
  failed_attempts:
    per_second: 0.1
    burst_count: 3
```

? Bot Discord — Rôle et responsabilités

Le bot Discord est le **chef d'orchestre** du système :

1. Commande `/whitelist <pseudo_minecraft>` libre (accessible à tous)
2. Vérifie et ajoute le joueur à la whitelist Minecraft
3. Crée automatiquement un compte Matrix via l'API Admin Synapse :

```
POST /_synapse/admin/v2/users/@pseudo:matrix.protosrv.fr
Authorization: Bearer <access_token_admin>
```

4. Stocke la liaison `Discord ID ↔ UUID Minecraft ↔ compte Matrix`
5. Génère/fournit les credentials Matrix au mod au premier lancement

> ℹ Le token d'accès admin s'obtient via :
>
> ```
> curl -X POST "https://matrix.protosrv.fr/_matrix/client/v3/login" \
>   -H "Content-Type: application/json" \
>   -d '{"type": "m.login.password", "user": "protomehka-admin", "password": "***"}'
> ```
>
> La réponse contient le champ `access_token` à stocker dans les variables d'environnement du bot.

> ℹ Le bot est déjà en cours de développement dans une autre session avec ProtoMehka — **à coordonner** pour l'ajout de la partie création de compte Matrix.

? Authentification du joueur dans le mod

Méthode retenue : Option A — Mot de passe envoyé par MP Discord

Le bot génère un mot de passe aléatoire à la création du compte Matrix et l'envoie en MP Discord au joueur. Le mod construit automatiquement l'identifiant Matrix à partir du pseudo Minecraft, le joueur n'a qu'à saisir son mot de passe au premier lancement.

Flow d'authentification :

```
1. Mod détecte qu'aucun token n'est stocké localement
2. Écran de config s'ouvre automatiquement :

   ┌──────────────────────────────────────┐
   │ Bienvenue ProtoMehka !               │
   │                                      │
```

```
   | Mot de passe Matrix :               |
   | [_____________________________]     |
   |                                     |
   |           [Se connecter]            |
   └─────────────────────────────────────┘
```

3. Le mod récupère le pseudo MC automatiquement :
   `String mcUsername = Minecraft.getInstance().getUser().getName();`
   `String matrixUserId = "@" + mcUsername.toLowerCase() + ":matrix.protosrv.fr";`
4. Appel `POST /_matrix/client/v3/login` avec userId + password
5. Token d'accès Matrix stocké localement → plus jamais redemandé

MP Discord envoyé par le bot :

```
 Tu es whitelisté sur ProtoSRV !

Ton mot de passe Matrix : xK9#mP2$nL
Entre-le dans Minecraft au premier lancement du mod.
```

> **i** Le pseudo Minecraft = le pseudo Matrix (sans le @...protosrv.fr). Le bot doit créer le compte Matrix avec exactement le même username que le pseudo Minecraft du joueur.

Le mod est configuré pour utiliser un **Space Matrix dédié** (l'équivalent d'un serveur Discord) plutôt que d'afficher toutes les rooms du compte joueur. Seules les rooms appartenant au Space du serveur Minecraft sont visibles dans le mod.

Structure du Space :

```
Space: "Mon Serveur Minecraft"
├── #général          — chat ouvert à tous
├── #annonces         — read-only pour les joueurs
├── #aide             — support
├── #off-topic        — détente
└── #staff            — privé, réservé aux admins
```

Avantages :

- Les DM perso ou autres rooms Matrix du joueur ne se mélangent pas dans l'interface MC
- Ajouter un nouveau salon dans le Space côté Synapse le fait apparaître automatiquement dans le mod pour tous les joueurs, sans mise à jour du mod
- Rooms publiques et privées gérées nativement par Matrix

Config du mod :

```
{
  "homeserver": "matrix.protosrv.fr",
  "space_id": "!xxxxx:matrix.protosrv.fr"
}
```

> ℹ Le Space et ses rooms sont à précréer sur Synapse avant le premier lancement du mod. Les rooms privées (ex: #staff) ne seront visibles que pour les joueurs qui y ont été invités.

```
Forge 1.20.1
├── OkHttp          — HTTP + WebSocket (connexion Matrix /sync)
├── Gson            — parsing JSON des events Matrix
├── GUI Minecraft   — Screen natif Forge pour l'interface
└── Thread dédié    — sync Matrix hors thread principal MC
```

Dépendances Gradle à ajouter

```
dependencies {
    implementation fg.deobf("...")  // Forge standard

    // OkHttp
    implementation 'com.squareup.okhttp3:okhttp:4.12.0'
    // Gson (déjà présent dans MC, vérifier version)
    implementation 'com.google.code.gson:gson:2.10.1'
}
```

> ⚠ Attention aux conflits de dépendances avec le classpath Forge/MC — shading d'OkHttp probablement nécessaire.

? Architecture des packages suggérée

```
fr.protomehka.matrixmod/
├── MatrixMod.java              — Entry point Forge
├── config/
│   └── MatrixConfig.java       — Homeserver URL, credentials stockés
├── matrix/
│   ├── MatrixClient.java       — Gestion auth + /sync polling
│   ├── MatrixRoom.java         — Modèle d'une room
│   └── MatrixMessage.java      — Modèle d'un message
├── gui/
│   ├── MatrixScreen.java       — GUI principal (liste rooms + messages)
│   └── RoomListWidget.java     — Widget liste des salons
├── chat/
│   ├── ChatInterceptor.java    — Intercepte envoi/réception chat MC
│   └── MatrixChatRenderer.java — Injection messages Matrix dans chat vanilla
├── events/
│   └── KeyBindHandler.java     — Keybind M pour ouvrir le GUI
└── notification/
    └── ToastNotifier.java      — Notifications toast in-game
```

? Flow utilisateur complet

1. Joueur fait /whitelist sur Discord
2. Bot crée compte Matrix (@pseudoMC:matrix.protosrv.fr) + génère mot de passe aléatoire
3. Bot envoie le mot de passe en MP Discord au joueur
4. Bot invite le joueur au Space Matrix
5. Joueur rejoint le serveur Minecraft
6. Mod détecte qu'aucun token n'est stocké → écran de saisie mot de passe
7. Joueur entre son mot de passe → mod fait /login → token stocké localement
8. MatrixClient se connecte, thread de sync démarre
9. Messages du salon actif s'affichent dans le chat MC
10. Joueur tape dans le chat → envoyé dans le salon Matrix actif
11. Appui sur M → GUI s'ouvre → switch de salon possible
12. Notification toast si message reçu dans un autre salon

? Points ouverts à définir

☑ ~~Comment le mod récupère les credentials au premier lancement ?~~ → Mot de passe envoyé par MP Discord, pseudo MC = pseudo Matrix

☐ Chiffrement E2EE ? (complexifie, via libolm — optionnel)

☐ Rooms Matrix précréées (recommandé) ou créées à la volée ?

☐ Le Space ID est-il hardcodé dans le mod ou configurable par fichier ?

☐ Gestion des rooms privées (DM entre joueurs) ?

Briefing généré depuis une session de conception avec Claude — Mars 2026

Revision #2
Created 10 March 2026 09:44:03 by ProtoMehka
Updated 10 March 2026 14:53:24 by ProtoMehka