

Technologies du projet PSG Académie

1. BACKEND - Framework & Serveur

FastAPI 0.124.4

- Rôle : Framework web Python pour créer des API REST
- Avantages :
 - ⚡ Performance : Plus rapide que Django/Flask (basé sur Starlette + Pydantic)
 - 📄 Documentation auto : Génère OpenAPI/Swagger automatiquement (/docs)
 - ✅ Validation : Vérifie automatiquement les données entrantes avec Pydantic
 - ⚡ Async natif : Support async/await pour hautes performances
 - 💡 Autocomplétion : Typage fort → meilleur support IDE
- Alternatives : Django REST (plus lourd), Flask (moins moderne), Express.js (Node.js)

Uvicorn 0.34.0

- Rôle : Serveur ASGI pour exécuter FastAPI
- Avantages :
 - ⚡ Ultra-rapide (basé sur uvloop)
 - ⚡ Hot reload en développement
 - ⚡ Support async natif
 - 🚀 Production-ready avec workers multiples
- Alternatives : Gunicorn (WSGI, moins performant), Hypercorn

Nginx

- Rôle : Serveur web de production et reverse proxy
- Avantages :
 - 🚀 Reverse proxy : Dirige le trafic vers FastAPI/Uvicorn
 - 🔒 HTTPS facile avec Let's Encrypt (certbot)
 - 📦 Fichiers statiques : Sert React build ultra-rapidement
 - ⚖️ Load balancing : Répartit charge sur plusieurs instances
 - 🛡 Sécurité : Rate limiting, protection DDoS, headers de sécurité

-  Gestion compression gzip/brotli
 - Alternatives : Apache (plus lourd, configuration complexe), Caddy (auto HTTPS mais moins configurable), Traefik
-

2. BACKEND - Base de données

PostgreSQL 15

- Rôle : Base de données relationnelle principale
- Avantages :
 -  Open source leader en fonctionnalités avancées
 -  Relations complexes : Foreign keys, joins performants (Country→Academy→Team→Player)
 -  Types riches : JSON, arrays, dates avancées, géométrie
 -  ACID compliance : Transactions fiables et cohérentes
 -  Excellent pour analytics : Window functions, CTEs, indexation avancée
 -  Extensions puissantes (PostGIS, pgcrypto, etc.)
- Alternatives : MySQL (moins de features avancées), SQLite (fichier simple, mono-utilisateur), MongoDB (NoSQL, pas de relations)

SQLModel 0.0.27

- Rôle : ORM (Object-Relational Mapping) pour PostgreSQL
- Avantages :
 -  Best of both worlds : Combine SQLAlchemy (ORM puissant) + Pydantic (validation)
 -  Moins de code : Une seule classe = modèle DB + schéma API
 -  Type safety : Détection d'erreurs avant exécution
 -  Créé par le même auteur que FastAPI (intégration parfaite)
 -  Migrations faciles avec Alembic
- Alternatives : SQLAlchemy pur (plus verbeux), Django ORM (lié à Django), Tortoise ORM, Peewee

psycopg2-binary 2.9.10

- Rôle : Driver PostgreSQL pour Python
- Avantages :

- 🏆 Standard de facto pour PostgreSQL en Python
- ⚡ Version binary = pré-compilée (pas besoin de compiler libpq)
- 🔒 Très stable et mature
- 🔐 Pool de connexions intégré
- Alternatives : `asyncpg` (async pur, plus complexe), `psycopg3` (nouvelle version)

Redis (optionnel)

- Rôle : Cache en mémoire et stockage de sessions
 - Avantages :
 - ⚡ Ultra-rapide : Stockage RAM (sub-milliseconde)
 - 📊 Cache requêtes fréquentes : Stats, classements, rapports récents
 - 🔒 Session storage pour JWT refresh tokens
 - 📈 Message broker : Peut gérer files d'attente (Celery)
 - 📡 Pub/Sub : Notifications temps réel
 - 💾 Persistence optionnelle sur disque
 - Alternatives : Memcached (moins de features, seulement cache), cache applicatif (moins performant, pas partagé)
-

3. BACKEND - Traitement de données

Pandas 2.3.3

- Rôle : Analyse et manipulation de données tabulaires
- Avantages :
 - 📊 Standard industrie pour data science en Python
 - 🔐 Parsing CSV puissant : Détection auto de types, gestion encodages
 - ⚡ Groupby/Aggregate : Grouper par joueur, calculer moyennes/max
 - 📡 Transformations : Pivot, merge, join de dataframes
 - 📈 Compatible Matplotlib/Seaborn pour visualisations
 - 📁 Export multiple formats (CSV, Excel, JSON, SQL)
- Votre usage : Parser CSV Catapult GPS, grouper par joueur, calculer benchmarks
- Alternatives : Polars (plus rapide mais moins mature), Dask (big data), CSV stdlib (trop basique)

Matplotlib 3.10.8

- **Rôle : Génération de graphiques et visualisations**
- **Avantages :**
 - **Contrôle total : Personnalisation pixel par pixel**
 - **Tous types de graphiques : Gauges semi-circulaires custom, heat maps, etc.**
 - **Export : PNG/PDF/SVG haute qualité**
 - **Bibliothèque la plus utilisée en Python (écosystème énorme)**
 - **Backend Agg pour génération server-side sans display**
- **Votre usage : Rapports professionnels avec header, logo, jauge, tableaux colorés**
- **Alternatives : Plotly (interactif HTML mais lourd), Bokeh (web-based)**

Seaborn 0.13.2

- **Rôle : Visualisations statistiques élégantes**
- **Avantages :**
 - **Thèmes pro par défaut (vs Matplotlib brut)**
 - **Graphiques statistiques en 1 ligne (distributions, corrélations)**
 - **S'intègre parfaitement avec Matplotlib (même backend)**
 - **Palettes de couleurs harmonieuses**
- **Votre usage : Color schemes, esthétique professionnelle des rapports**
- **Alternatives : Matplotlib pur (moins joli), Plotly Express**

Pillow 12.0.0

- **Rôle : Manipulation d'images (fork moderne de PIL)**
- **Avantages :**
 - **Ouvre/redimensionne/affiche images**
 - **Intégration Matplotlib : OffsetImage, AnnotationBbox**
 - **Formats multiples : PNG, JPG, GIF, TIFF, etc.**
 - **Filtres et transformations d'images**
- **Votre usage : Afficher logo Thonon Evian Grand Genève dans header des rapports**
- **Alternatives : OpenCV (overkill pour simple affichage), imageio, scikit-image**

openpyxl

- Rôle : Lire et écrire fichiers Excel (.xlsx)
- Avantages :
 -  Import stats depuis Excel du coach
 -  Intégration Pandas : pd.read_excel(), df.to_excel()
 -  Gestion formules, styles, graphiques Excel
 -  Pas besoin de Microsoft Excel installé
- Votre usage : Import données externes, export rapports en Excel
- Alternatives : xlrd/xlwt (anciens formats .xls), csv (plus simple mais perd formatage)

python-multipart 0.0.21

- Rôle : Parser uploads de fichiers multipart/form-data
 - Avantages :
 -  Requis par FastAPI pour UploadFile
 -  Streaming : Gère gros fichiers sans charger tout en RAM
 -  Supporte fichiers multiples simultanés
 - Votre usage : Upload CSV Catapult via [/catapult/upload](#)
 - Alternatives : Aucune (dépendance obligatoire FastAPI)
-

4. FRONTEND - Framework & Build

React 19.2.0

- Rôle : Framework JavaScript pour interfaces utilisateur
- Avantages :
 -  Composants réutilisables : Découpage modulaire de l'UI
 -  Virtual DOM : Mises à jour ultra-rapides (reconciliation)
 -  Écosystème énorme : Millions de bibliothèques disponibles
 -  Standard industrie : Utilisé par Facebook, Netflix, Airbnb
 -  Hooks : useState, useEffect pour logique propre et moderne
 -  React 19 : Server Components, nouvelles optimisations
- Alternatives : Vue.js (plus simple, moins d'emplois), Angular (plus verbeux, TypeScript obligatoire), Svelte (compile, plus petit)

TypeScript

- Rôle : Superset de JavaScript avec typage statique
- Avantages :
 - 🔎 Détection erreurs avant runtime
 - 💡 Autocomplétion : Meilleur support IDE
 - 📄 Documentation : Types = documentation vivante
 - 👤 Travail d'équipe : Code plus sûr et maintenable
 - 🔐 Compatible 100% JavaScript
- Alternatives : Flow (Facebook, moins adopté), PropTypes (React, moins puissant)

Vite 7.2.4

- Rôle : Build tool et serveur de développement
- Avantages :
 - ⚡ Ultra-rapide : HMR instantané (Hot Module Replacement)
 - 🚀 Dev server démarre <1s (vs 30s webpack Create React App)
 - 🏗️ Build optimisé : Rollup pour production (tree-shaking, code splitting)
 - 🔧 Configuration minimale : Fonctionne out-of-the-box
 - 💡 Support natif : TypeScript, JSX, CSS modules
 - 🎯 ESM natif en développement
- Alternatives : Create React App (obsolète, lent), Webpack (complexe), Next.js (overkill pour SPA), Parcel

ESLint 9.39.1 + Plugins

- Rôle : Linter JavaScript/TypeScript
- Avantages :
 - 🐛 Détecte erreurs avant exécution
 - 📜 Force conventions : Style de code uniforme
 - 🔧 Auto-fix : Corrige automatiquement beaucoup d'erreurs
 - 🌐 Plugins React : react-hooks (vérifie règles des hooks), react-refresh (HMR)
 - 🎯 Configurable par projet

- Alternatives : Prettier seul (formate mais ne vérifie pas logique), TSLint (déprécié), StandardJS
-

5. FRONTEND - UI & Visualisation

Material UI / Chakra UI / Mantine

- Rôle : Bibliothèques de composants React
- Avantages :
 -  Design professionnel : Sans écrire CSS custom
 -  Composants prêts : Tables, forms, buttons, modals, tabs
 -  Accessibilité : ARIA intégré, navigation clavier
 -  Responsive : Mobile-first par défaut
 -  Thèmes : Personnalisation facile (couleurs, spacing)
 -  TypeScript : Types inclus
- Material UI : Google Material Design, très complet
- Chakra UI : Simple, composable, excellente DX
- Mantine : Moderne, hooks puissants, dark mode facile
- Alternatives : Ant Design (très riche mais chinois), Bootstrap React (vieillissant), Tailwind UI (CSS utility)

Recharts / Chart.js

- Rôle : Graphiques interactifs frontend
- Avantages :
 -  React-friendly : Composants React natifs (pas de DOM direct)
 -  Interactifs : Hover, click, zoom, pan
 -  Beaux par défaut : Pas besoin de config complexe
 -  Responsive : S'adapte à la taille d'écran
 -  Personnalisable : Custom tooltips, axes, légendes
- Recharts : Déclaratif React, meilleure intégration
- Chart.js : Plus de types de graphes, très populaire
- Votre usage : Évolution performances joueurs, stats par match
- Alternatives : Plotly.js (3D, plus lourd), D3.js (très custom, courbe d'apprentissage), Victory Charts

6. INFRASTRUCTURE & Production

Docker Compose 3.8

- **Rôle : Orchestration de conteneurs multi-services**
- **Avantages :**
 - **Environnement reproductible** : Même config dev/prod
 - **Un fichier** : [`docker-compose.yml`](#) définit toute l'infra
 - **Commande simple** : `docker-compose up` lance tout
 - **Volumes** : Persistence PostgreSQL, Redis
 - **Networking** : Isolation et communication inter-services
 - **Versions** : Pin versions exactes des images
- **Votre usage** : PostgreSQL isolé, facile à reset, même environnement équipe
- **Alternatives** : Installation locale (pollution système), Kubernetes (overkill petits projets), Podman

Python 3.12

- **Rôle : Version Python du projet**
- **Avantages :**
 - **Performances** : 10-60% plus rapide que 3.10
 - **Type hints améliorés** : PEP 695 (syntaxe type plus simple)
 - **Error messages** : Plus clairs et précis
 - **Debugging** : Meilleur traceback
- **Alternatives** : Python 3.11 (stable mais moins rapide), Python 3.13 (trop récent)

7. AUTHENTIFICATION & Sécurité

JWT (JSON Web Tokens) avec OAuth2

- **Rôle : Authentification et autorisation**
- **Avantages :**
 - **Stateless** : Token contient tout (pas de session serveur)
 - **Multi-rôles** : Coach, admin, joueur, analyste
 - **Refresh tokens** : Renouvellement sécurisé sans re-login

- **Compatible API** : Mobile, SPA, services tiers
- **Signature cryptographique** : Impossible de falsifier
- **Expiration** : Tokens courts pour sécurité
- **Implémentation** : FastAPI OAuth2PasswordBearer
- **Alternatives** : Sessions cookies (stateful, moins scalable), Auth0 (SaaS payant), Keycloak (self-hosted lourd)

HTTPS via Nginx + Let's Encrypt

- **Rôle** : Chiffrement communications
 - **Avantages** :
 - **Chiffrement** : Protège données sensibles (mots de passe, stats)
 - **Certificats gratuits** : Let's Encrypt auto-renouvelé
 - **Certbot** : Configuration automatique Nginx
 - **SEO** : Google favorise HTTPS
 - **Alternatives** : Cloudflare (proxy), Caddy (auto HTTPS intégré)
-

8. CI/CD & Qualité

GitHub Actions

- **Rôle** : CI/CD (Continuous Integration/Deployment)
- **Avantages** :
 - **Tests auto** : Pytest, Jest à chaque commit
 - **Déploiement auto** : Push vers production après merge
 - **Lint + Build** : ESLint, Black, tests avant merge
 - **Gratuit** : 2000 min/mois pour projets publics
 - **Intégration GitHub** : Pull requests, status checks
 - **Docker build** : Cache layers, push registry
- **Workflows** : .github/workflows/ci.yml
- **Alternatives** : GitLab CI (si GitLab), Jenkins (self-hosted, complexe), CircleCI (payant), Travis CI

Pydantic 2.12.5

- **Rôle** : Validation de données et sérialisation

- **Avantages :**
 - **Validation automatique : Types + contraintes (min, max, regex)**
 - **Sérialisation : JSON, dict, ORM**
 - **Pydantic V2 : 5-50x plus rapide (Rust core)**
 - **Custom validators : Logique métier complexe**
 - **Erreurs claires : Messages détaillés**
 - **Votre usage : Validation uploads CSV, requêtes API, modèles SQLModel**
 - **Alternatives : Marshmallow (plus ancien), Cerberus, dataclasses stdlib (pas de validation)**
-

9. IMPORT/EXPORT

wkhtmltopdf / WeasyPrint

- **Rôle : Export PDF de rapports HTML**
 - **Avantages :**
 - **HTML → PDF : Génère PDF depuis templates HTML**
 - **CSS complet : Garde mise en page, couleurs, fonts**
 - **Rapports pros : Calendriers, fiches match, synthèses**
 - **Print-ready : Marges, en-têtes, pieds de page**
 - **wkhtmltopdf : Basé WebKit, très stable**
 - **WeasyPrint : Python pur, plus facile à déployer**
 - **Alternatives : ReportLab (code Python pur, complexe), Puppeteer (Node.js), Playwright PDF**
-

10. MOBILE (Futur)

React PWA (Progressive Web App)

- **Rôle : Application mobile web**
- **Avantages :**
 - **Une seule codebase : Même code React pour web + mobile**
 - **Offline mode : Service Workers cachent données**
 - **Installable : Icône sur écran d'accueil**

-  **Pas d'App Store : Pas de fees Apple/Google**
-  **Push notifications : Via navigateur**
-  **Déploiement instant : Pas de validation stores**
- **Limites : Pas d'accès total hardware (vs React Native)**
- **Alternatives : React Native (code natif, apps séparées), Flutter (Dart, performant), Ionic (Cordova/Capacitor)**