

World-Univ-Rank : Analyse mondiale de la performance universitaire et de ses déterminants socio-économiques

I - Contexte

L'enseignement supérieur et la recherche jouent aujourd'hui un rôle central dans le développement des sociétés modernes. Ils constituent à la fois un axe d'innovation, un indicateur de performance nationale, et un facteur clé d'attractivité internationale. Les universités participent activement à la formation du capital humain, à la production scientifique et à la compétitivité économique des pays.

À travers ce projet, nous souhaitons analyser la dynamique mondiale de l'enseignement supérieur en nous appuyant sur le classement international « *Times Higher Education (THE) World University Rankings* », l'un des plus reconnus dans le monde universitaire. Ce classement évalue chaque année les universités selon plusieurs dimensions : enseignement (*Teaching*), environnement de recherche (*Research Environment*), qualité des publications (*Research Quality*), impact industriel (*Industry Impact*), ouverture internationale (*International Outlook*). La méthodologie de ce classement est expliquée via : [*THE methodology*](#).

L'objectif principal est de comprendre comment ces performances évoluent d'une année à l'autre (2016–2025) et d'un pays à l'autre, tout en examinant l'influence des facteurs socio-économiques sur la qualité de l'enseignement supérieur. Pour ce faire, nous avons enrichi les données du classement THE par des indicateurs nationaux issus d'un second jeu de données regroupant des statistiques mondiales par pays. Ces variables permettent d'explorer plusieurs aspects déterminants tels que : le PIB par habitant (niveau de richesse), la migration nette (mobilité internationale), le taux d'alphabétisation (niveau d'éducation de base), la densité de population, le taux d'équipement en télécommunication, etc.

À travers cette analyse, nous chercherons à mettre en évidence les liens possibles entre le développement socio-économique d'un pays et la performance académique de ses universités, ainsi qu'à répondre à certaines questions récurrentes telles que : « Peut-on établir qu'un pays plus riche ou plus alphabétisé produit de meilleures universités ? », « Les sociétés plus ouvertes sur l'international ou plus égalitaires entre les sexes se distinguent-elles davantage dans les classements mondiaux ? », etc.

II - Fichiers à utiliser

Les fichiers à utiliser sont « **Classement_THE_des_universites_mondiales_2016-2025.csv** », « **statistiques_pays_du_monde.csv** » et « **dictionnaire_donnees.txt** » disponibles sur **Ecampus**.

Le premier fichier regroupe les données issues du classement *THE*, couvrant la période de 2016 à 2025. Il contient des informations détaillées sur les universités du monde entier : leur nom, leur pays, leur rang, leur score global, ainsi que plusieurs indicateurs de performance tels que l'enseignement, la qualité et l'environnement de recherche, l'impact industriel, l'ouverture internationale, ou encore la répartition femmes-hommes et la proportion d'étudiants internationaux.

Le second fichier présente pour chaque pays une série d'indicateurs démographiques et socio-économiques permettant de contextualiser les performances des universités : population, PIB par habitant, densité, migration nette, taux d'alphabétisation, part des secteurs industriel et tertiaire, ou encore le nombre de téléphones pour 1000 habitants.

Enfin, le dernier fichier décrit précisément la signification des variables présentes dans les deux jeux de données, afin de faciliter leur compréhension et leur exploitation.

III - Organisation de votre application

Lors de la création de votre application, il est indispensable d'adopter une structure de dossiers claire et organisée. Voici une suggestion de structure de base pour votre projet :

- Dossier racine du projet : Ce dossier contiendra les fichiers et dossiers principaux de votre projet Flask :

- application.py** : Le fichier principal de votre application Flask.

- config.py** : (optionnel) Le fichier de configuration de l'application, contenant au minimum l'URI de la BDD, la clé secrète Flask (**SECRET_KEY**), les configurations **SQLAlchemy** (**SQLALCHEMY_TRACK_MODIFICATIONS**, etc.).

- models** : (Recommandé) Fichiers de modèles SQLAlchemy.

- templates** : Ce dossier contiendra tous les modèles HTML de votre application. Chaque modèle représente une page ou une partie spécifique de votre site web.

- static** : Dans ce dossier, vous stockerez les fichiers statiques tels que les fichiers CSS, JavaScript, images, etc. Créer des répertoires **css** et **js**.

- data** : Si votre application utilise des données statiques (par exemple, des fichiers CSV), vous pouvez les stocker dans ce dossier.

-**tests** : (Optionnel) Vous pouvez créer un dossier séparé pour les tests unitaires de votre application Flask. Cela permet de maintenir une séparation claire entre le code de l'application et les tests.

IV - Étapes du projet

L'objectif de cette SAÉ est de concevoir une application web interactive basée sur *Flask*, combinant les technologies suivantes : *Flask-SQLAlchemy* pour la gestion des données (modélisation et persistance en base *SQLite*), *Flask-Migrate* pour la gestion et le suivi des évolutions du schéma de la BDD (optionnel), *Flask-WTF* pour la création de formulaires de recherche et filtrage, *Bootstrap* pour un design réactif et accessible, et *Chart.js* pour la visualisation dynamique des résultats. Les pages de l'application devront être dynamiques et générées à l'aide du moteur de templates *Jinja2*. Le nettoyage et le traitement des jeux de données seront effectués avec les bibliothèques *Pandas* et *NumPy*.

Aucune autre technologie que celles mentionnées ci-dessus ne sera autorisée.

Voici les étapes à suivre pour réaliser cette SAÉ :

→ Étape N°1 : Analyse des Données et Conception des diagrammes

Avant de commencer le développement de l'application, il est indispensable d'analyser le jeu de données fourni. Cette analyse consiste à explorer la structure des fichiers CSV afin de comprendre les informations qu'ils contiennent et d'identifier les possibilités qu'offrent ces données. Cette étape permet de repérer les entités importantes, leurs attributs et les relations éventuelles.

À l'issue de cette analyse, votre jeu données contiendra les informations suivantes:

```
liste = ['rang', 'nom_univ', 'pays', 'pop_etud',  
'ratio_etud_pers', 'etud_internationaux_pct', 'ratio_fem_hom',  
'score_global', 'indic_enseig', 'indic_env_rech',  
'indic_qualite_rech', 'indic_impact_industrie',  
'indic_rel_intern', 'annee', 'ratio_fem', 'ratio_hom', 'region',  
'population', 'superf_m2', 'pib_hab', 'migration_nette',  
'industrie_part', 'services_part', 'alphabetisation_pct',  
'tel_1000hab']
```

À partir de cette analyse, il faut ensuite concevoir un MCD (Modèle Conceptuel de Données) ou un MLD (Modèle Logique de Données). Ces modèles définissent la structure de la future base de données et permettront de stocker et gérer efficacement les informations sur les universités, les pays, les indicateurs, etc.

En complément, vous devez élaborer un Diagramme de Cas d'Utilisation (DCU) afin d'identifier les fonctionnalités essentielles de l'application et de préciser les

interactions entre les utilisateurs et le système. Enfin, il est nécessaire de créer un diagramme de classes, qui décrit les principales classes, leurs attributs, leurs méthodes ainsi que les relations entre elles. Ces deux diagrammes contribuent à clarifier et organiser le fonctionnement global de l'application « World-Univ-Rank » avant son développement.

Remarques importantes :

1- Pensez à nommer vos descripteurs avec exactement les mêmes noms que ceux de la liste ci-dessus, afin d'uniformiser les conventions et de faciliter le travail du correcteur.

2- Intégrez dans votre fichier de présentation les éléments suivants : le MCD ou MLD, le DCU, le DC (diagramme de classes). Ils font partie intégrante de la phase d'analyse et seront évalués dans votre présentation finale.

→ Étape N°2 : Création de la BDD avec Flask-SQLAlchemy

À cette étape, vous devez mettre en place le modèle de données de votre application en utilisant *Flask-SQLAlchemy*, la bibliothèque recommandée pour gérer les bases de données dans un projet *Flask*. L'objectif est de définir une structure de données robuste qui reflète fidèlement votre MLD conçu lors de l'étape de modélisation.

Vous devrez donc : créer l'ensemble des tables identifiées dans le MLD, définir les clés primaires, clés étrangères et cardinalités, et mettre en place les relations ORM (*one-to-many*, *many-to-many*) grâce aux fonctionnalités offertes par *SQLAlchemy*. La normalisation reste une exigence essentielle : elle garantit la cohérence, l'intégrité, l'absence de doublons et un stockage efficace des informations.

Remarque importante : Dans ce projet, vous ne devez pas écrire de requêtes SQL manuelles. La base de données choisie est de type *SQLite*. Ce choix est motivé par plusieurs raisons : c'est simple et léger, et il fonctionne avec un fichier unique ce qui facilite le partage

→ Étape N°3 : Peuplement de la BDD à partir des fichiers CSV

Pour permettre le fonctionnement de l'application *World-Univ-Rank*, vous devez rédiger un script Python capable d'importer les données provenant des fichiers

CSV, de les nettoyer et les insérer dans votre base de données SQLite. Ainsi, vous devez :

- créer les instances des modèles SQLAlchemy correspondant à chaque ligne
- et insérer les enregistrements dans la base à l'aide de l'ORM,

Une fois le peuplement terminé, vous pouvez interroger la base via l'ORM pour valider la bonne insertion, vérifier la cohérence entre les tables, et tester que les relations ORM fonctionnent (accès aux objets liés, etc.).

Remarque importante : Gestion des erreurs et nettoyage des données

Lors du traitement des fichiers CSV, vous devez impérativement prendre en compte et gérer les erreurs potentielles, notamment : fichiers introuvables, erreurs lors du nettoyage, et celles d'insertion en base. Votre script doit être capable d'informer clairement l'utilisateur en cas d'erreur bloquante.

→ Étape N°4 : Développement de l'interface utilisateur

L'interface utilisateur revêt une importance cruciale au sein de cette application. Les langages de développement comprennent *HTML*, *CSS*, *Bootstrap*, *JavaScript*, et *ChartJS*. Vous avez la liberté de concevoir les modèles HTML nécessaires pour simplifier la navigation et améliorer l'interaction avec les visualisations de données. Voici les fonctionnalités clés minimales de cet outil :

➤ **Page d'accueil (index):**

C'est la porte d'entrée principale de votre application. Elle doit permettre au visiteur d'obtenir une vision synthétique et immédiatement lisible du classement mondial des universités. Elle doit obligatoirement contenir les éléments suivants :

1- Une carte de KPI (indicateurs clés) : vous affichez au minimum 9 KPI, calculés à partir du jeu de données filtré (ou global si pas de filtre). Par exemple : nombre total d'universités, nombre total de pays représentés, score moyen d'enseignement, score moyen de recherche, % moyen d'étudiants internationaux, valeur moyenne du PIB/habitant, valeur moyenne du taux d'alphabétisation, moyen du score global THE et ratio moyen H/F. Les KPI doivent être simples, visuels et compréhensibles.

2- Trois graphiques à barres + storytelling (3-4 lignes maximum) qui explique ce que l'on observe :

→ Graphique 1 – Top 5 pays par score d'enseignement pour répondre à la question suivante : Quels pays possèdent les meilleurs systèmes d'enseignement universitaire ?

→ Graphique 2 – Top 5 pays par score de recherche pour répondre à la question : Les pays leaders en enseignement sont-ils également leaders en recherche ?

3- Un diagramme circulaire (Pie Chart) représentant la répartition des universités par région afin de représenter les régions les plus représentées.

4- Un tableau synthétique qui présente un échantillon réduit des universités, par exemple les 10 premières uniquement. Ce dernier contient ces colonnes : **rang**, **nom_univ**, **pays**, **région**, **indic_enseig**, **indic_qualite_rech** et **score_global** et un bouton « **Détails** ». Le clic sur ce dernier redirige vers une page distincte, accessible via une URL du type : **/universite/<id>** qui correspond à la fiche complète de l'université sélectionnée. Cette dernière contient :

→ tous les indicateurs socio-économique de son pays et ses propres indicateurs.

→ 1 graphique en ligne représentant l'évolution de l'enseignement, de la recherche et du score globale au fil des années considérées. En dessous du graphique, vous devez ajouter un court texte de "storytelling" (3 à 5 lignes) commentant vos observations.

➤ Page des universités :

La page Universités permet d'explorer les universités de façon comparative grâce à des graphiques, puis de rechercher une université selon certains critères et d'accéder à sa fiche détaillée. Elle doit contenir les éléments suivants :

1- **Section graphique – Comparaisons enseignement / recherche** : afin de donner une vue immédiate des meilleures et des moins bonnes universités selon ces deux indicateurs principaux, vous devez afficher 4 graphiques à barres :

→ Top 5 universités en enseignement,

→ Bottom 5 universités en enseignement,

→ Top 5 universités en recherche,

→ Bottom 5 universités en recherche

2- **Formulaire de recherche (Flask-WTF)** : Sous les graphiques, vous devez afficher un formulaire simple pour filtrer les universités selon : Pays (liste déroulante), année (liste déroulante), score minimal d'enseignement (optionnel), score minimal de recherche (optionnel). Une fois le formulaire validé, vous affichez un tableau léger, contenant seulement les universités correspondant aux critères (par exemple, les universités d'un pays donné, ou celles pour une année donnée ou ayant un score min donné). Voici les colonnes à afficher pour chaque université : **rang**, **nom_univ**, **pays**, **région**, **indic_enseig**, **indic_qualite_rech** et **score_global** et un bouton « **Détails** ». Le clic sur ce dernier redirige vers une page distincte, accessible via une URL du type : **/universite/<id>** qui correspond à la fiche complète de l'université sélectionnée. Cette dernière contient :

→ tous les indicateurs socio-économique de son pays et ses propres indicateurs.

→ 1 graphique en ligne représentant l'évolution de l'enseignement, de la recherche et du score globale au fil des années considérées. En dessous du graphique, vous devez ajouter un court texte de "storytelling" (3 à 5 lignes) commentant vos observations.

➤ **Page statistiques :**

Cette page a pour objectif de montrer à partir de quelques graphes simples, comment certains facteurs internes à l'université ou socio-économiques du pays peuvent être associés aux performances universitaires (enseignement, recherche, score global). Vous devez mettre ces graphiques :

1-**Graphique A - Internationalisation et performance en recherche (barChart)** : On étudie le lien entre le pourcentage d'étudiants internationaux et la qualité de la recherche. Pour cela, on regroupe les universités en classes de pourcentage d'étudiants internationaux, par exemple : [0-10 %], [10-20 %], [20-30 %], [30 % et +]. Pour chaque classe, on calcule la moyenne de **indic_qualite_rech** (et éventuellement de **score_global**), et on représente ces valeurs dans un diagramme en barres. Sous le graphique, un court texte commente la tendance observée.

2- **Graphique B - Richesse des pays et qualité de l'enseignement (barChart)** : On s'intéresse au lien entre le PIB par habitant et les performances des universités en enseignement et/ou en recherche. Pour cela, on regroupe les

pays en 3 catégories de PIB/habitant, par exemple : pays à faible revenu (≤ 1135 USD), pays à revenu intermédiaire (entre 1136 USD et 4495 USD) et pays à haut revenu (≥ 4496 USD). [Voir lien](#).

Pour chaque catégorie, on calcule la moyenne de **indic_enseig** et de **indic_qualite_rech** des universités appartenant à ces pays. On affiche ces résultats dans un diagramme en barres groupées (par exemple une barre enseignement et une barre recherche pour chaque catégorie de PIB). Un court texte discute l'éventuel lien entre niveau de richesse et performances universitaires.

3- Graphique C - Niveau d'alphabétisation et score global (pieChart) : On examine le rôle du taux d'alphabétisation des pays sur le score global des universités. Pour cela, vous créez quelques classes d'alphabétisation, par exemple : $< 80 \%$, $[80-90 \%)$ et 90% . Ensuite, vous affichez un diagramme en secteurs montrant la répartition des universités les mieux classées (par exemple top 100 en score global ou en recherche) entre ces classes d'alphabétisation. Vous insérez un commentaire mettant en relation le niveau d'éducation de base et la performance des universités.

4-Graphique D - Ratio femmes / hommes et score d'enseignement (grouped bar chart) : On examine l'influence potentielle de l'équilibre femmes / hommes au sein des universités sur la qualité de l'enseignement et, éventuellement, sur la recherche. Les universités sont regroupées en trois catégories selon le pourcentage de femmes (**ratio_fem**) ou d'hommes (**ratio_hom**): $< 40 \%$, $40-60 \%$ et $> 60 \%$. Pour chaque catégorie, on calcule : la moyenne **indic_enseig** et éventuellement moyenne **indic_qualite_rech**. Les résultats sont représentés dans un diagramme en barres groupées, permettant de comparer facilement les performances des universités selon leur niveau de mixité. Sous le graphique, un court commentaire interprétatif (3 à 5 lignes) doit être rédigé afin de décrire les tendances observées.

(Optionnel) Diagrammes complémentaires : Pour enrichir l'application, vous pouvez ajouter un ou plusieurs diagrammes supplémentaires mettant en lumière l'influence d'autres facteurs socio-économiques sur la qualité de l'enseignement et/ou de la recherche, par exemple : la migration nette, le niveau d'équipement technologique et la part de l'industrie et des services dans l'économie. Chaque graphique devra être accompagné d'un bref texte de commentaire permettant d'interpréter les résultats.

Assurez-vous de gérer les redirections entre les pages et de gérer les erreurs de manière adéquate.

V - Modalités du rendu de projet et de Soutenance orale

Voici les principales modalités à respecter :

1- Fichiers Python contenant les **Docstrings et commentaires pertinents** :

Chaque fonction/classe/méthode doit être documentée avec des **docstrings**. Une chaîne documentaire doit expliquer ce que fait la fonction/classe/méthode, quels sont ses arguments, et ce qu'elle renvoie. Cela aide le correcteur et les autres membres de votre équipe à comprendre et à utiliser vos fonctions. En plus des docstrings, vous devriez ajouter des commentaires pertinents dans le code pour expliquer les parties complexes ou importantes de votre implémentation.

2- Les fichiers PDF de la présentation et du poster.

Votre présentation doit être structurée en trois grandes parties obligatoires :

→ **Partie « Analyse et conception »** : Cette partie doit démontrer le travail de réflexion et de conception effectué en amont du développement. Elle doit contenir :

- x une preuve d'utilisation de Git / GitHub (captures d'écran de l'historique des commits, branches ou contributions des membres de l'équipe, etc.)
- x le Diagramme de Cas d'Utilisation (DCU) ;
- x le Diagramme de classes (DC) et schéma de la base de données ;
- x la répartition des tâches dans le groupe (qui a fait quoi).

→ **Partie « Tests de l'application »** : Cette partie doit montrer le fonctionnement concret de votre application : navigation dans les pages, affichage des graphiques, fonctionnement des formulaires, un scénario de test comme par exemple : ex : recherche d'une université.

→ **Partie « Mise en œuvre et retour d'expérience »** : Chaque membre du groupe devra intervenir oralement pour présenter :

- x La partie du projet qu'il/elle a développée ;
- x Les difficultés rencontrées (techniques, organisationnelles, conceptuelles) ;
- x Les solutions apportées (des extraits de code pour justifier le travail réalisé) ;
- x Les perspectives d'amélioration de l'application

Une attention particulière sera portée à l'implication individuelle de chaque étudiant.

3- Qualité du code :

Vous devez éviter la duplication de code autant que possible. Si plusieurs fonctions ont besoin de faire la même chose, envisager de créer une fonction utilitaire pour éviter de répéter le code. Utiliser des noms de variables/classes/méthodes clairs et explicites pour que le code soit facile à comprendre. Éviter les noms de variables obscurs ou trop courts. Surtout adopter une approche orientée objet.

4- Dépôt sur Ecampus (jusqu'au 18/12 à 18h)

Une archive compressée : **NomV_NomW_NomX_NomY.zip** (V, W, X et Y sont les noms des étudiants) contenant les **fichiers de présentation, de poster, et Python y compris docstrings et commentaires. Un travail non soumis pour évaluation aura 0 comme note finale de projet. Chaque retard engendre des points en moins.**

5- Soutenance orale (09/12) :

Chaque groupe disposera d'environ 1h pour effectuer une présentation orale de son projet. Chacun des étudiants présentera son travail pendant une durée de 5 minutes. Ensuite, chaque étudiant sera interrogé individuellement, en mettant l'accent particulièrement sur la participation individuelle de chaque membre au projet. Il est important de noter que pendant la présentation, l'examineur n'interviendra pas. Les 5 minutes allouées à chaque présentateur sont courtes, un effort important devra être fait pour mettre en avant et clairement, l'implication de chacun dans la réalisation du projet. Après les présentations individuelles, l'examineur disposera de 5 minutes pour poser des questions et évaluer le travail présenté par le groupe.