Tunisian Software Qualification Board

PredictIQ : Une Approche Intelligente pour la Prédiction d'Anomalies

TLILI Mounir (Head of Digital Factory, Maileva – Docaposte)
RIDANE Abdelkarim (Testeur Automaticien, Maileva – Docaposte)

Plan

1. Maileva, qui sommes-nous ?
2. Contexte initial – constat - objectifs
3. Approches IA pour la prédiction d’anomalies
4. Prédiction d’anomalies – Analyse des Tickets JIRA
5. PredictIQ
6. Bilan - Résultats
7. Conclusion et perspectives

Qui sommes-nous ?

MAILEVA
UNE MARQUE DE DOCAPOSTE

Plateforme Courrier pro et solutions de dématérialisations

L'acteur de la transformation numérique des PME et ETI françaises

Notre Mission

Proposer aux entreprises quelque soit leur taille, des solutions de dématérialisation de leur gestion documentaire pour leur permettre d'accélérer leur transformation numérique, en confiance.

Notre Ambition

Œuvrer pour un numérique utile, responsable et choisi pour permettre à tous d'accéder en confiance aux nouveaux services, en respectant la confidentialité des données et les choix de chacun.

Envoi de courrier en ligne

Envoi de courrier recommandé électronique ou papier

Envoi de fiche de paie ou documents RH sur les coffres fort numérique ou papier

Facture électronique client

62,6 millions €
de chiffre d'affaires en 2024

+ de 20 ans
d'expérience

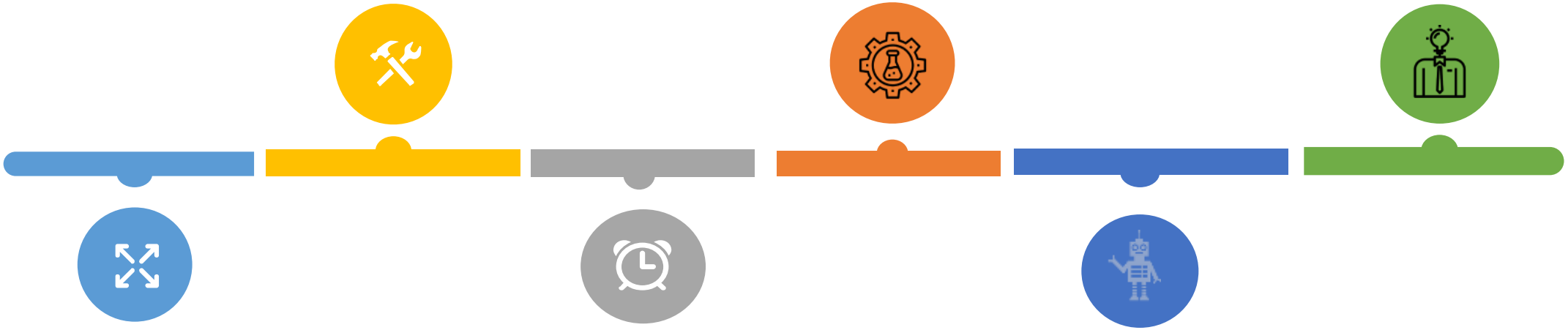
130
Collaborateurs et collaboratrices

Contexte initial – constat - objectifs

Le test logiciel est un composant important pour garantir la qualité de nos produits. Cependant, il reste une activité coûteuse.

Les avancées en intelligence artificielle (IA), notamment en Machine Learning, Deep Learning et NLP, transforment profondément le domaine du test logiciel.

Approches IA pour la détection des anomalies ?
Types de tests concernés ?
Données nécessaires ?



Contexte actuel du développement logiciel : complexité croissante, cycles courts, livraisons rapides (Agile/DevOps/CI/CD)

Besoin de réduire les coûts, le temps et les risques liés aux défauts logiciels.

Utilisation de l'IA pour prédire et prévenir les anomalies avant exécution.

Approches IA pour la prédiction d'anomalies

Approches supervisés classique

- Utilisation des features structurées (priorité, délais, type), parfois couplées à des représentations textuelles simples (TF-IDF).
- **Avantages** : simplicité, bonne interprétabilité, faible besoin de données massives.
- **Limites** : faible prise en compte du contenu sémantique complexe du texte.
- Exemples**: RandomForest, Logistic Regression, XGBoost**

Approches basées sur le NLP avancé

- L'utilisation de modèles pré-entraînés permet d'extraire des représentations contextuelles riches du texte (embeddings),
- **Avantages** : capables d'accomplir plusieurs tâches en langage naturel à partir d'un simple prompt, ce qui les rend particulièrement adaptés à des contextes de faible supervision ou à forte variabilité lexicale.
- **Limites** : Couteux en termes de CPU et nécessitent des ressources en GPU.
- Exemples: **BERT**, **RoBERTa, FLAN-T5**

Approches non supervisées / semi-supervisées

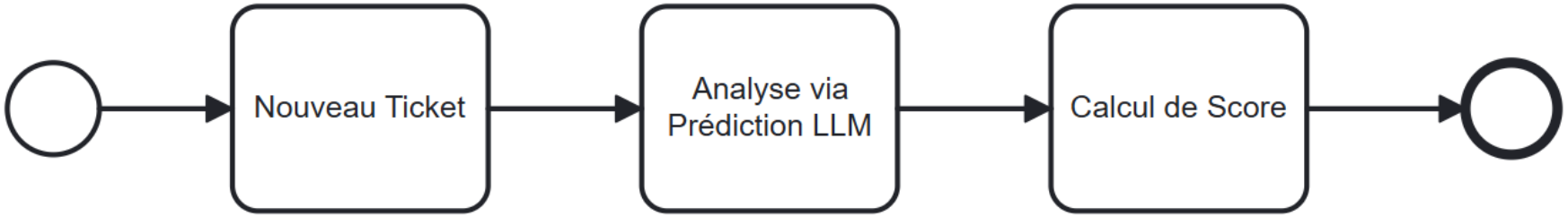
- Sans entraînement supervisé direct : autoencodeurs, Isolation Forest, clustering, ou embeddings + similarité cosinus.
- **Avantages** : pertinent pour des données bruitées ou peu annotées, flexible et moins coûteux en ressources.
- **Limites** : Dépendance à la qualité et à la taille du corpus, sensibilité au seuil de décision, difficulté avec des tickets courts ou mal renseignés.
- Exemples**: all-MiniLM-L6-v2** (embeddings + similarité cosinus), **TextAutoEncoder**, **Isolation Forest**

Prédiction d'anomalies – Analyse des tickets JIRA

- JIRA, en tant qu'outil de gestion des tickets, est désormais largement adopté par les équipes de développement logiciel.
- Suivi des différents types d'éléments : epic, évolutions (stories), bugs, tâches, incidents, etc.
- Chaque ticket regroupe une quantité importante d'informations textuelles : titre, description, critères d'acceptations, commentaires…
- Chaque ticket inclut également de nombreuses métadonnées : priorité, statut, temps de résolution, assignation, dates clés, etc.
- L'ensemble de ces données constitue une source précieuse d'information pour l'analyse et la prédiction d'anomalies
  - ✓ Priorisation automatique des tickets à traiter.
  - ✓ Aide à l'allocation des ressources (assignation, tests approfondis).
  - ✓ Détection précoce de tickets critiques pour intervention rapide.
  - ✓ Monitoring continu.

PredictIQ: overview et workflow

- Un prototype d'analyse automatique des tickets Jira (anomalies / bugs potentiels).
- Permet d'assister les équipes (QA, support, PO) en détectant automatiquement les tickets à risque élevé, dès leur création.
- Basé sur une approche d'intelligence artificielle non supervisée.
- Capable de calculer un score de similarité sémantique entre un ticket nouvellement créé et un corpus de tickets historiques, afin d'estimer le risque d'anomalie sans recourir à un entraînement supervisé lourd.



Business Workflow

PredictIQ: principales étapes

Etape 1

Extraction des données:
Extraction de 2000 tickets avec les données suivantes:

- **Titre**
- **Résumé**
- **Description**
- **Commentaires**
- **Critères d'acceptation**
- **Champs personnalisés** : Sévérité, Composants, Sprint, Étiquettes
- **Liens avec d'autres tickets** (et leur type + résumé)

Etape 2

Préparation & labellisation:
Les tickets Jira sont prétraités pour extraire le texte brut à partir des champs structurés (résumé, description, commentaires, liens).

Etape 3

Chargement du corpus historique:
Lors du démarrage du service :
Tous les tickets historiques (≈2000) sont vectorisés une fois.
Les embeddings sont chargés en mémoire.

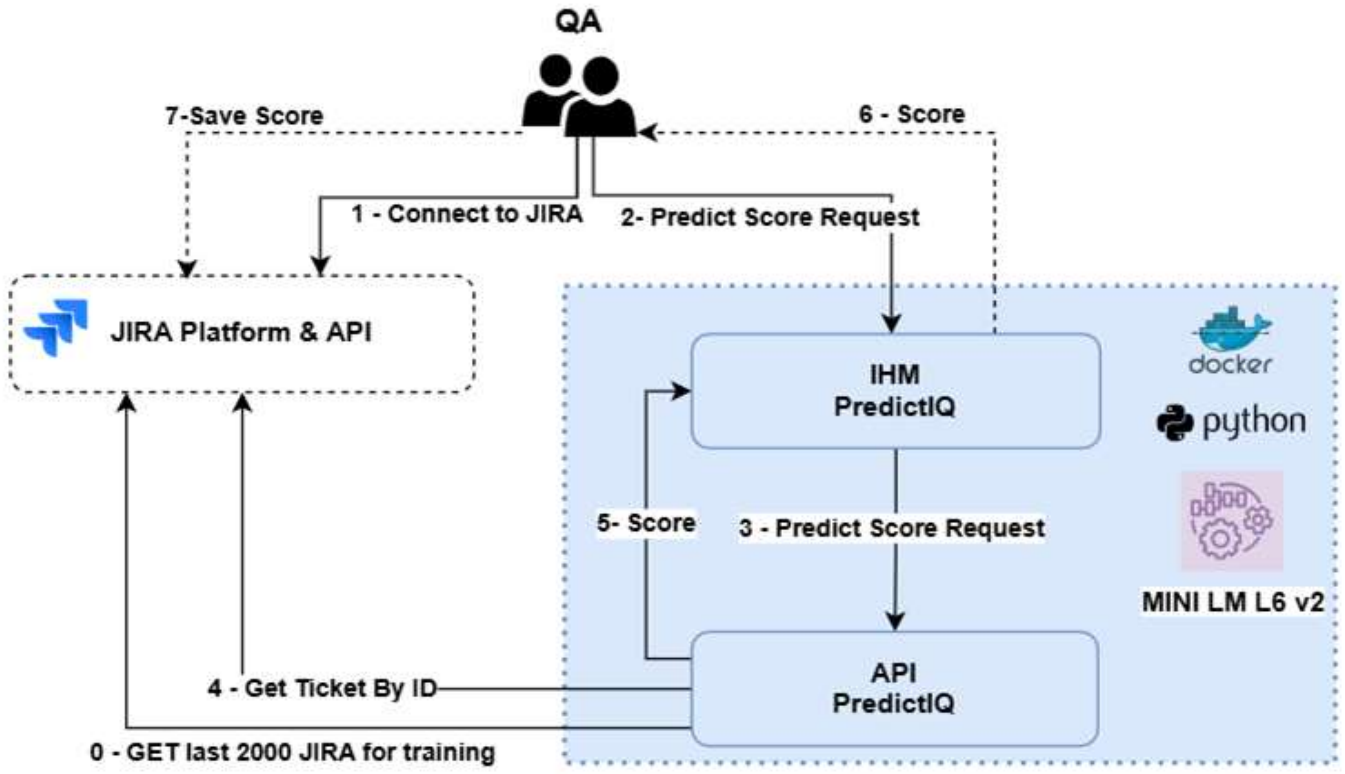
Etape 4

Traitement du nouveau ticket et prédiction
Chaque ticket est transformé en embedding vectoriel à l'aide du modèle SentenceTransformer pré-entraîné (**all-MiniLM-L6-v2**):

- Encoder le texte (représentation sémantique) des vecteurs similaires.
- Mesurer la similarité entre le nouveau ticket et les tickets du corpus pour estimer le score d'anomalie.

PredictIQ

PredictIQ: architecture

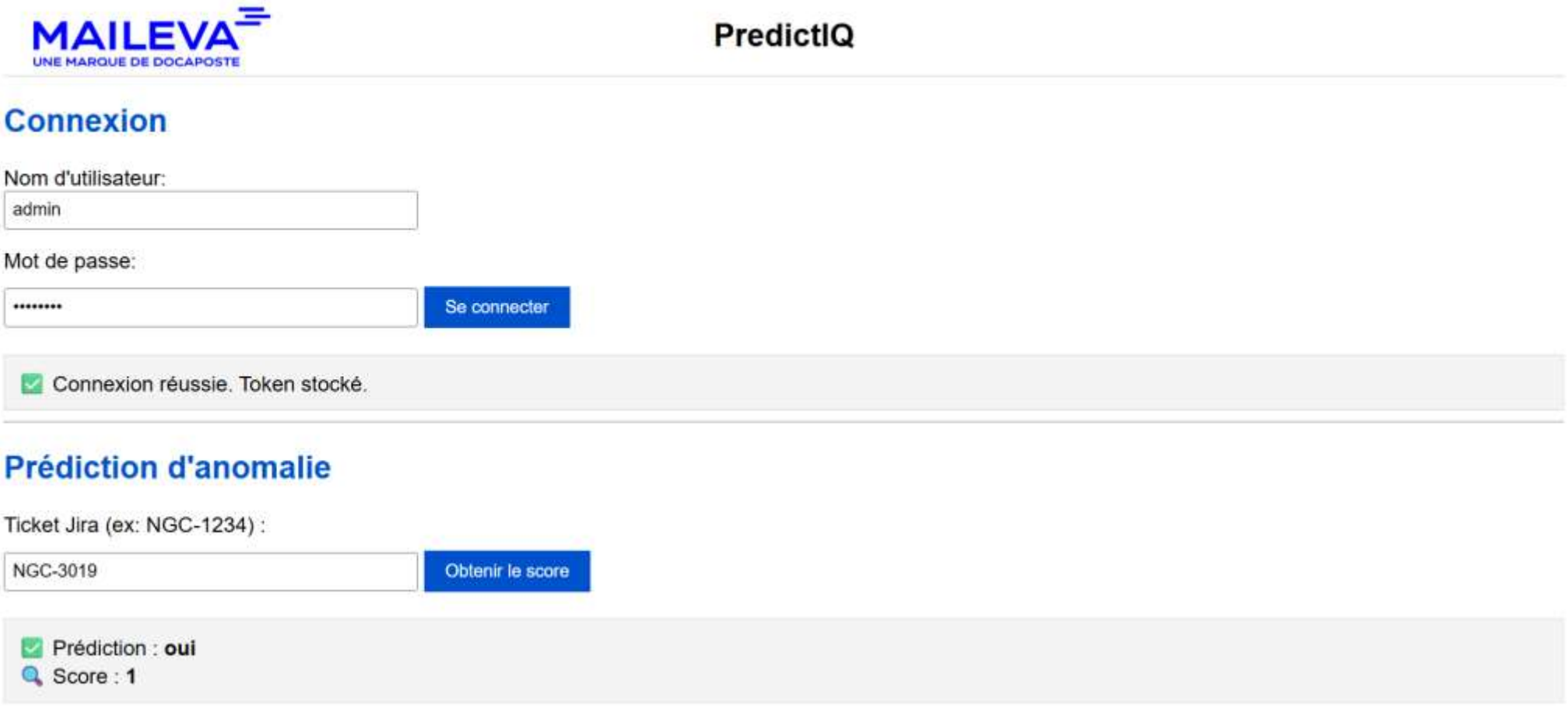


Fonctionnement

1. L'utilisateur s'authentifie et entre un numéro de ticket Jira
2. Le script interroge l'API Jira → récupère toutes les infos du ticket
3. Formatage de l'entrée modèle (texte enrichi avec contexte)
4. Le texte est encodé par **SentenceTransformer**
5. Le vecteur obtenu est comparé au corpus des tickets historiques via similarité cosinus
6. Un score d'anomalie (de 0 à 1) est calculé et une prédiction binaire (oui/non) est retournée.

PredictIQ

PredictIQ: IHM



PredictIQ

Bilan - Résultats

Capacité à détecter les tickets proches d'anomalies connues

Identifier efficacement les tickets présentant des similitudes sémantiques fortes avec des anomalies historiques.

Robustesse en absence d'étiquettes

Cette approche ne nécessite **aucun entraînement supervisé**, ce qui :

- évite la construction manuelle de datasets d'entraînement,
- garantit une mise en place rapide,
- assure une bonne généralisation dans des contextes où les labels sont incomplets, ambigus ou incohérents.

Rapidité et faible coût d'exécution

L'encodage par **all-MiniLM-L6-v2** + similarité cosinus est :

- **léger** (fonctionne sur CPU),
- **rapide** (quelques dizaines de millisecondes),
- compatible avec une intégration dans un pipeline temps réel.

Recommandation pertinente

Les scores d'anomalie permettent :

- la détection des tickets suspects,
- la priorisation proactive pour les équipes QA / Support,

Flexibilité de l'approche

Le système fonctionne quel que soit :

- le projet Jira,
- Il est donc réutilisable pour d'autres équipes ou contextes

Bilan - Limites

Dépendance forte au corpus d'historique

Les résultats reposent entièrement sur la qualité et la diversité :

- des tickets historiques,
- des anomalies déjà observées.

Un corpus incomplet peut induire :

- des faux négatifs (nouveaux types d'erreurs non représentés),
- un biais vers les anomalies les plus fréquentes.

➔ **Solution** : mise à jour régulière des embeddings + versioning du corpus

Absence de fine-tuning sur données internes

Le modèle n'est pas spécifiquement optimisé pour le langage métier Jira (vocabulaire spécifique, formats variés…) acronymes internes.

➔ **Solution** : fine-tuning SentenceTransformer sur le corpus Jira interne.

Qualité inégale des tickets

Les tickets : très courts, incomplets, mal décrits
→ Produisent des embeddings peu informatifs.
➔ **Solution** : enrichir les textes avec contexte (ticket lié, sprint, module, auteur).

Conclusion - Perspectives

Intégration automatisée avec Jira (Webhook)

- Mettre en place un webhook Jira permettant :
- l'analyse automatique **dès le passage en TO TEST d'un ticket**,
- la génération instantanée d'un score d'anomalie,
- l'envoi d'une alerte aux équipes concernées. (mail, notifications teams)

➔ *Permet un fonctionnement 100 % autonome et temps réel.*

Ajout d'un module d'apprentissage semi-supervisé

- Implémenter un mécanisme de feedback utilisateur permettant:
  - validation ou correction des prédictions,
  - mise à jour du corpus, réajustement du seuil.

➔ *Le modèle devient plus pertinent au fil du temps.*

Fine-tuning d'un modèle SentenceTransformer sur le corpus interne

- Adapter le modèle aux spécificités : du vocabulaire métier, des structures textuelles Jira, des projets techniques internes.

➔ *Amélioration significative de la précision et cohérence sémantique.*

Conclusion - Perspectives

Extension du corpus à plusieurs projets Jira

Actuellement centré sur un seul projet. Objectif :

- intégrer plusieurs projets Jira,
- diversifier les types d'anomalies,
- améliorer la généralisation du modèle.

➔ *Réduction des biais et meilleure couverture des cas métiers.*

Optimisation du seuil d'anomalie

Au lieu d'un seuil statique (0.75), on peut implémenter :

- Seuil dynamique en fonction de la distribution des scores

➔ *Des décisions mieux calibrées selon le contexte.*

Merci pour votre attention