Cahier des charges – DEFIUT

DEFIUT

Auteurs :

Elouan CARDAIRE
Jade COUTANT
Iliès DJERBI
Ayette EL-HOMADI
Camille LE BRECH
Noah NGUYEN

Table des matières :

1. Contexte et objectifs

1.1. Contexte

L'application DEF'IUT que nous souhaitons mettre en place, permet de challenger les étudiants désireux de relever des défis informatiques. L'objectif est d'offrir une expérience éducative complète, qui se veut à la fois ludique et pratique, pour explorer différents domaines comme la programmation, la cybersécurité, l'algorithmique et d'autres concepts fondamentaux. Les différents défis répondent à des problèmes d'actualité et permettent d'acquérir des compétences théoriques et pratiques pour les étudiants de tous niveaux.

1.2. Objectifs

Cette application devra :

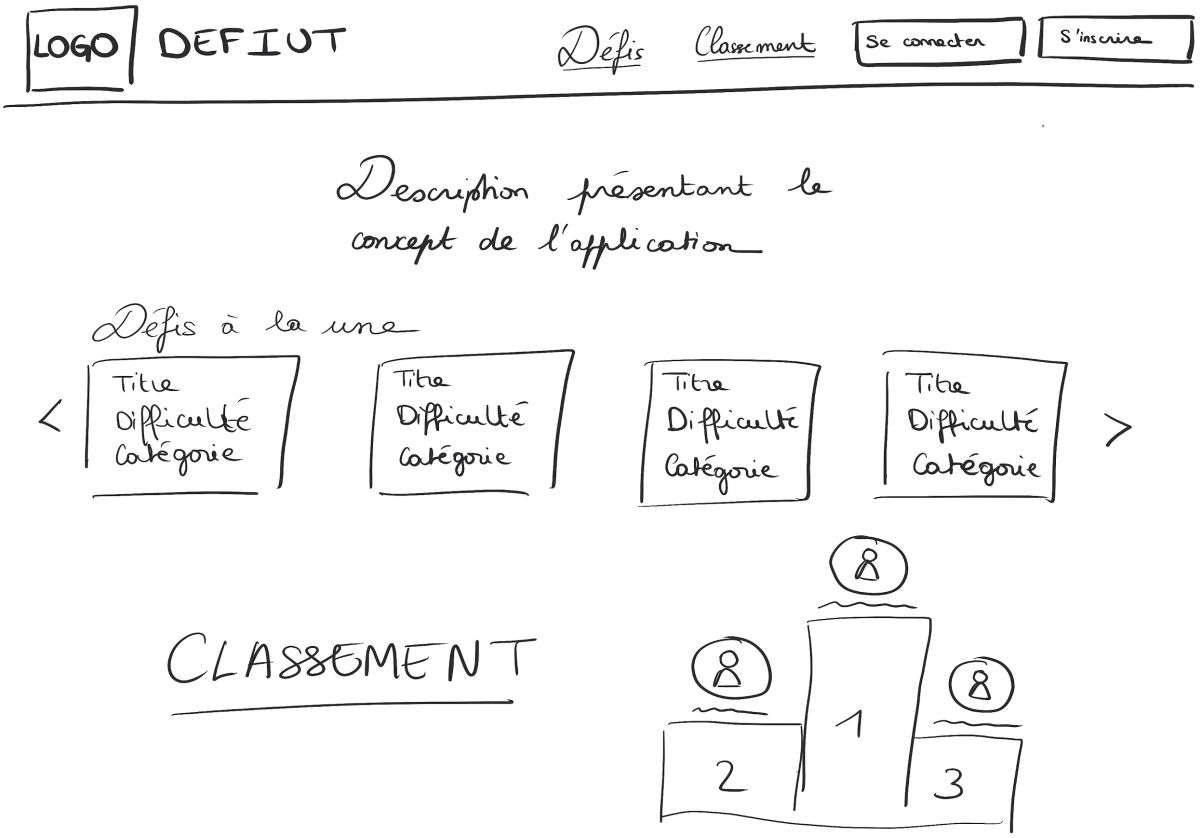
- rendre accessible la résolution de défis afin que chacun puisse obtenir des compétences théoriques et pratiques ;
- assurer un suivi personnalisé pour chaque utilisateur grâce à un système de sauvegarde de résultats (points) ;
- mettre en compétition les différents utilisateurs via un classement des joueurs et un système de récompenses (badge de succès) ;
- être intuitive, ergonomique et attrayante, pour permettre à chaque utilisateur d'avoir une expérience de navigation agréable.

1.3. Public cible

Les principaux utilisateurs concernés sont les étudiants de BUT informatique (1ere/2eme/3eme année) souhaitant améliorer leurs compétences en informatique d'une manière ludique.
Les personnes non initiées à l'informatique doivent également pouvoir trouver des défis adaptés à leurs niveaux, afin de progresser sur des sujets variés.
Les enseignants possédant les droits administrateurs peuvent également proposer des défis et suivre la progression de leurs étudiants.

2. Cahier des charges fonctionnel

Cette section du cahier des charges présente les principales fonctionnalités attendues pour la plateforme. L'objectif est de fournir un cadre structuré pour guider la conception et le développement du site web, garantissant ainsi la couverture des fonctionnalités essentielles, un design harmonieux, et une expérience utilisateur optimale.



2.1. Fonctionnalités principales

Vous trouverez ici, les fonctionnalités que nous souhaitons impérativement trouver dans notre application. Pour chacune, une description détaillée vous permettra de connaître nos attentes avec précision.
Nous sommes évidemment disponibles pour échanger sur le contenu et sa mise en place.

Header : Le bandeau horizontal doit contenir le logo Def'IUT à gauche. A droite, si l'utilisateur n'est pas connecté, Il doit y avoir des boutons pour la connexion et l'inscription, sinon, un bouton sous la forme d'une image de profil, qui redirige vers la page de profil. Le

catalogue de défis ainsi que la page de classement doivent être accessibles pour les utilisateurs connectés et non connectés. En plus, il doit y avoir une icône thème sombre / thème clair qui permet de changer le thème du site. Si l'utilisateur possède un compte, le thème qu'il choisit doit être conservé pour ses prochaines visites. Le thème par défaut doit être sombre.

Page d'accueil : La page d'accueil doit afficher quelques défis à la une (ajoutés récemment ou les plus populaires par exemple) sous la forme d'un carrousel. Un très bon travail serait d'afficher les défis "tendance" du moment, par exemple ceux qui sont le plus trouvé dans les dernières 24h ou autre. Si l'utilisateur est connecté, ce carrousel peut afficher les défis en cours afin de reprendre rapidement une activité récente.
Sous les défis à la une, un podium montre les trois meilleurs joueurs du moment avec leur photo de profil et leur pseudo.

Pages d'authentification : L'inscription et la connexion des utilisateurs doivent se faire via une interface intuitive et simple. Pour l'inscription, il est demandé un email, un pseudo, un mot de passe et la confirmation du mot de passe. Pour la connexion, l'utilisateur peut mettre son email ou son pseudo et mot de passe. L'authentification via Google est demandée et d'autres services comme Github peuvent être ajoutés. L'idéal serait de pouvoir vérifier le compte de l'utilisateur au travers de l'envoi d'un mail sur l'adresse mail renseignée.

Page du catalogue des défis : Cette page permet aux utilisateurs d'accéder à tous les défis de la plateforme. Les défis sont classés par catégorie (WEB, cryptographie, réseau, programmation…). Pour chaque défi, les informations suivantes doivent être affichées :

- son titre ;
- sa difficulté ;
- son nombre de points ;
- une courte description attrayante.

L'utilisateur doit pouvoir trier ces défis selon divers critères, tels que le niveau de difficulté, le nombre de points de récompense, ou encore le nom. Des filtres supplémentaires, comme les tags (par exemple, un langage de programmation), doivent être disponibles pour affiner la recherche. La présentation doit être claire et lisible, même pour un grand nombre de défis. Pour améliorer les performances, les défis doivent être chargés progressivement au fur et à mesure du défilement de la page (infinite scroll).

Page d'un défi : Cette page permet aux utilisateurs de consulter les détails complets d'un défi et d'y soumettre une solution (flag) avec une vérification automatique en place. Un "flag" est une chaîne de caractères que l'utilisateur doit trouver. La page doit afficher un

titre, une difficulté, un nombre de points, un auteur, une liste de tags et une description en format Markdown (La description doit s'afficher correctement, afficher proprement les titres, les textes en gras,...). Si le défi possède des fichiers annexes, ils doivent être disponibles en téléchargement sécurisé. L'utilisateur peut également afficher des indices (si disponibles) pour l'aider à résoudre le défi. Les indices sont "payants" et coûtent un certain nombre de points préalablement accumulés. Nous vous laissons libre sur la mise en place du coût des indices, cela peut être un pourcentage du nombre de points à gagner sur le défi. Si le défi à résoudre permet de gagner 200 points, le premier indice peut coûter 5% de ces 200 points (soit 10 points). Ce pourcentage peut augmenter pour obtenir des indices supplémentaires.
Pour "valider" le défis, l'utilisateur doit saisir son "flag" dans un champ de texte et cliquer sur un bouton "soumettre". La validation du flag est décrite dans la description du défi. Si le défi est validé, un message de succès est affiché, et on sauvegarde le nombre de points qu'il a gagné. Si l'utilisateur se trompe, un message lui indique qu'il s'est trompé, lui suggère de voir un indice (s' il ne les a pas déjà tous révélés) et l'utilisateur peut retenter de valider le défi autant de fois qu'il veut. Pour le moment, seuls les défis de type "flag" sont pris en charge, mais à terme, d'autres types de défis, comme l'introduction de code via une interface de développement intégrée à la page de résolution de code.

Page de profil : La page de profil doit afficher le pseudo de l'utilisateur, son score global, sa dernière connexion et la date d'inscription. Elle doit également lister tous les défis validés par l'utilisateur avec des détails sur chaque défi validé (titre, date de validation, points gagnés) ainsi que la liste des badges obtenus. Si l'utilisateur consulte sa propre page de profil, il doit pouvoir modifier son pseudo et son mot de passe via des formulaires dédiés.
Elle doit également contenir les préférences de l'utilisateur concernant l'affichage du site (thème).

Page de classement : Cette page doit afficher le podium des trois meilleurs joueurs comme sur la page d'accueil mais également le reste du classement. A partir de ce classement, l'utilisateur doit pouvoir cliquer sur le profil d'un autre utilisateur et ainsi voir son profil public avec les défis qu'il a réalisés et les badges qu'il a obtenus.
Voici quelques exemples de badges à mettre en place :

- Badge Alan Turing : résoudre un défi ayant un défi labellisé "crypto"
- Badge Richard Hamming : résoudre un défi après (au moins) un essai raté.
- Badge Ada Lovelace : être le·a premier·e à résoudre un défi
- Badge Margaret Hamilton : résoudre (au moins) deux défis en moins de 24h
- Badge Leslie Lamport : résoudre un défi entre minuit et 6h du matin.
- Badge Linus Torvald : se connecter depuis un OS Linux

Footer : Le footer doit contenir des liens vers les pages Mentions légales, Conditions générales d'utilisation (CGU), FAQ et Contact. Ces pages doivent être en conformité avec les exigences légales et les bonnes pratiques en matière de transparence et de protection des données.

Mentions légales : Cette page doit contenir toutes les informations légales obligatoires.

Conditions générales d'utilisation (CGU) : Les CGU doivent être rédigées de manière claire et détaillée, en couvrant les points suivants : conditions d'inscription et d'utilisation du site, gestion des comptes utilisateurs, responsabilité des utilisateurs et de l'éditeur et protection des données personnelles.

Foire aux questions (FAQ) : La FAQ doit anticiper les principales interrogations des utilisateurs et fournir des réponses claires et pratiques concernant l'inscription, la participation aux défis, le calcul des scores, et la gestion des profils.
En plus de ces 3 liens, un bandeau horizontal affiche simplement l'année en cours et la mention "© Def'IUT - Tous droits réservés.".

Contact : cette page est un formulaire de contact permettant aux utilisateurs de nous envoyer un message. Elle doit contenir les champs suivants : adresse mail de l'utilisateur, l'objet de son message et le message en lui-même.

Encore une fois, nous sommes ouverts aux suggestions et nous encourageons toutes idées ou commentaires qui pourraient améliorer le projet. Vous pouvez poser des questions sur discord (c.f Communication et suivi) concernant des détails ou pour éclaircir certains points s' ils ne sont pas clairs.

2.2. User Story

ID	Titre	Description (User Story)	Critères d'acceptation	Priorité
US01	Création du Header	En tant qu'utilisateur, je veux accéder facilement aux principales sections du site via un bandeau supérieur clair et fonctionnel.	Le header contient le logo, les boutons "Connexion / Inscription" ou "Profil", les liens vers le catalogue et le classement, ainsi qu'un sélecteur de thème.	Haute

US02	Gestion des thèmes (clair/sombre)	En tant qu'utilisateur, je veux pouvoir changer le thème du site (clair ou sombre) et conserver mon choix pour mes prochaines visites.	Le thème sombre est par défaut ; le thème choisi est sauvegardé pour les utilisateurs connectés.	Moyenne
US03	Création de la page d'accueil	En tant qu'utilisateur, je veux visualiser les défis à la une et les meilleurs joueurs dès mon arrivée sur le site.	La page affiche un carrousel de défis récents/tendance et un podium avec les trois meilleurs joueurs.	Haute
US04	Système d'authentification complet	En tant qu'utilisateur, je veux pouvoir m'inscrire, me connecter ou me déconnecter facilement et en toute sécurité.	L'authentification fonctionne par email/pseudo + mot de passe, avec option Google/GitHub et vérification par email.	Haute
US05	Catalogue des défis	En tant qu'utilisateur, je veux parcourir tous les défis du site classés par catégories et filtres pour en choisir un à résoudre.	Le catalogue permet de trier, filtrer et charger progressivement les défis par difficulté, points ou tags.	Haute
US06	Consultation d'un défi	En tant qu'utilisateur, je veux accéder aux détails d'un défi (titre, auteur, description, points, fichiers associés) pour le comprendre avant de le tenter.	La page affiche la description Markdown, les fichiers annexes et les indices disponibles (payants).	Haute
US07	Validation d'un défi	En tant qu'utilisateur, je veux soumettre ma solution (flag) et savoir immédiatement si elle est correcte ou non.	Le système valide le flag, affiche un message de succès ou d'erreur et met à jour les points.	Haute

US08	Gestion du profil utilisateur	En tant qu'utilisateur, je veux consulter et modifier mes informations personnelles, mon score et mes préférences.	Le profil affiche pseudo, score, défis réussis, badges, et permet la modification du mot de passe et du pseudo.	Moyenne
US09	Classement général et badges	En tant qu'utilisateur, je veux voir le classement général et obtenir des badges selon mes performances.	Le classement affiche les 3 meilleurs joueurs et le reste du tableau. Les badges sont attribués automatiquement selon des conditions précises.	Moyenne
US10	Gestion du footer et des pages légales	En tant qu'utilisateur, je veux accéder aux informations légales et conditions d'utilisation du site.	Le footer contient les liens vers Mentions légales, CGU, FAQ, Contact, et la mention © Def'IUT.	Basse
US11	Page de contact	En tant qu'utilisateur, je veux pouvoir contacter les administrateurs du site via un formulaire dédié.	Le formulaire contient les champs email, objet, message, et affiche une confirmation d'envoi.	Moyenne
US12	Sécurité et gestion des sessions	En tant qu'utilisateur, je veux que mes informations soient protégées et que mes sessions soient sécurisées.	Les mots de passe sont hashés, les tokens de session sont valides et la déconnexion fonctionne correctement.	Haute

2.3. Interface utilisateur (UI)

L'interface doit être ergonomique, moderne et responsive (adaptée aux écrans mobiles et ordinateurs). Nous proposons l'utilisation des composants par défauts de Vuetify pour garantir une uniformité dans le design et une intégration rapide des composants UI.
Le site doit être conçu pour une utilisation en thème sombre, avec la possibilité de le passer en thème clair par un choix de l'utilisateur.
Le logo et la palette de couleurs doivent être respectés dans l'interface graphique de la plateforme. Nous vous fournirons les sources du logo en haute qualité, en format svg et png.

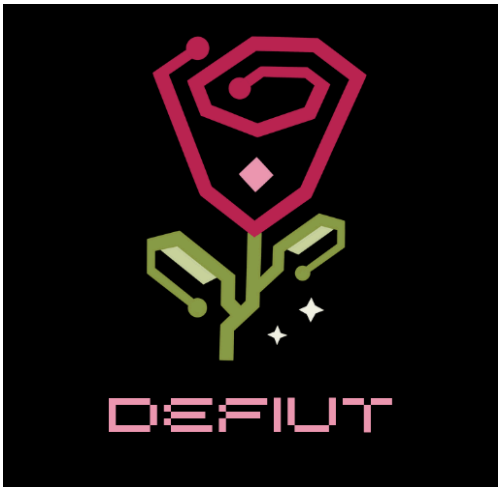
Voici un tableau permettant d'apporter des précisions sur la qualité de la plateforme attendue :

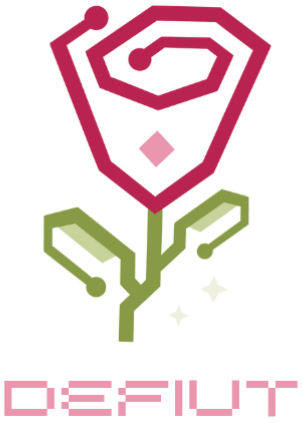
Critère	Description	Mesure / Indicateur
Ergonomie	L'interface doit être claire, intuitive et agréable à utiliser.	Test utilisateur : au moins 80 % de satisfaction (enquête)
Temps de réponse	Les pages et requêtes doivent se charger rapidement.	< 2 secondes pour 90 % des requêtes
Fiabilité	Le système doit fonctionner sans erreurs critiques.	Aucun crash durant les tests finaux
Sécurité	Les connexions et données doivent être protégées.	Authentification sécurisée, données hashées
Compatibilité	Le site doit fonctionner sur plusieurs navigateurs et écrans.	Tests validés sur Chrome, Firefox, Edge + responsive mobile
Évolutivité	L'ajout de nouveaux défis ou utilisateurs ne doit pas nécessiter de refonte.	Ajout de 10 nouveaux défis sans modification du code source
Accessibilité	L'application doit être utilisable par tous, y compris avec lecteur d'écran.	Respect partiel des normes WCAG (niveau A)
Maintenabilité	Le code doit être documenté, versionné et modulaire.	Documentation > 80 % de couverture des modules
Testabilité	Chaque fonctionnalité clé doit être testée automatiquement.	Taux de couverture des tests unitaires ≥ 70 %
Disponibilité	Le système doit être accessible en continu lors de la soutenance.	Aucun crash durant la démo

L’identité visuelle de la plateforme **Def’IUT** repose sur une cohérence graphique moderne, épurée et en lien avec l’univers technologique et compétitif du projet. Elle vise à offrir une expérience utilisateur immersive, claire et homogène sur l’ensemble du site.

https://coolors.co/palette/8a9b46-c8d29d-f1f1e4-eb99b3-ba2653

Le logo officiel de DEF’IUT (disponible en haute qualité sur le serveur Discord du projet) constitue l’élément central de l’identité visuelle.
Il doit être présent dans l’en-tête (header) de toutes les pages, garantissant une reconnaissance immédiate du site et de la marque.





.

Les polices choisies reflètent l’univers futuriste et technologique du projet. Elles sont sélectionnées selon leur lisibilité et leur complémentarité visuelle.
La police d’écriture du logo n’est pas dans cette liste, il s'agit de “Spaceship” de Canva.

- **Titres et éléments marquants :**
  - **Audiowide**

- Orbitron,

Ces typographies au style géométrique et numérique renforcent l'identité moderne du site.

- **Textes techniques / zones de code :**
  - Roboto mono,
  - Space Mono

Polices à chasse fixe garantissant une lecture claire pour les éléments de type code ou flag.

- **Textes courants / interfaces :**
  - Helvetica
  - Arial
  - Calibri
  - Poppins

Polices sobres et universelles assurant une excellente lisibilité sur tous les supports.

2.4. Expérience utilisateur (UX)

L'expérience utilisateur doit être fluide, intuitive et agréable, afin de garantir une navigation claire et cohérente sur l'ensemble du site. Une attention particulière sera portée aux points suivants :

Lisibilité et ergonomie :
Les textes et interfaces doivent être lisibles, avec des polices claires, des tailles adaptées et un contraste suffisant entre les éléments.
L'objectif est d'assurer un confort de lecture optimal, quel que soit le support utilisé.

Navigation intuitive :
Le parcours utilisateur doit être simple et efficace.
Par exemple, la page d'un défi doit être accessible en trois clics maximum à partir de la page d'accueil.
Les menus, liens et boutons doivent être organisés de manière logique et facilement

identifiables.

Retour visuel et interactions :
Chaque action de l'utilisateur doit être accompagnée d'un retour visuel clair : confirmation d'une soumission réussie, signalement d'une erreur, points gagnés, etc.
Des effets d'hover (au survol des boutons ou éléments interactifs) renforceront la perception de réactivité et de dynamisme.

Accès sans authentification :
L'accès aux défis doit être ouvert à tous les utilisateurs, même non connectés.
Cependant, dès qu'un utilisateur soumet une solution pour la première fois, le site lui suggère de se connecter.
La validation du flag et la confirmation du résultat ne seront accessibles qu'aux utilisateurs authentifiés.

Langue et évolutivité :
Le site sera initialement développé en français.
En fonction du temps et des ressources disponibles, une version multilingue (notamment en anglais) pourra être intégrée par la suite.

2.5. Respect du RGPD

La plateforme Def'IUT devra être conforme au Règlement Général sur la Protection des Données (RGPD), afin d'assurer la confidentialité, la sécurité et le bon traitement des informations personnelles des utilisateurs.

Les données collectées (telles que les identifiants, les résultats ou les informations de profil) seront exclusivement utilisées à des fins pédagogiques dans le cadre du fonctionnement de la plateforme.

Lors de l'inscription, un consentement explicite sera demandé à l'utilisateur, via une case à cocher « *J'accepte les CGU* », accompagnée d'un lien vers la page correspondante.

Chaque utilisateur devra également disposer de la possibilité de supprimer à tout moment ses données personnelles via une interface dédiée ou sur simple demande.

Enfin, le stockage des données devra être réalisé de manière sécurisée, conformément aux bonnes pratiques en matière de protection et de chiffrement des données.

3. Cahier des charges techniques

3.1. Technologies recommandées

Pour garantir une compatibilité avec les technologies que nous maîtrisons et faciliter le développement ultérieur, nous proposons les solutions suivantes :

Frontend : Utilisation de Vue.js (api composition) avec Vuetify comme framework de composants pour garantir un design cohérent et responsive.
Nous recommandons l'utilisation des icônes "Material Design Icons", bien que ce ne soit pas obligatoire. L'essentiel est que toutes les icônes utilisées proviennent d'un même pack afin de garantir une cohérence visuelle sur l'ensemble de la plateforme.
Backend : API REST en Node.js ou autre technologie similaire.
Base de données : Nous vous conseillons l'usage de firebase pour le stockage en temps réel des données utilisateurs et la gestion d'authentification. (Firebase permet facilement de gérer les authentifications via email/mot de passe et via des services externes comme Google et Github)

Pour tout usage de technologies différentes que celles recommandées, nous vous demandons un rapport expliquant votre choix et comparant la technologie recommandée avec celle de votre choix.

3.2. Format des défis

La plateforme propose des énigmes et exercices dont l'objectif est pédagogique : retrouver un FLAG (chaîne de caractères) dissimulé dans un code, une image, un binaire, ou obtenu après résolution d'un problème.

Remarque : la plateforme supporte pour l'instant uniquement des défis de type *flag*.
Des types supplémentaires (évaluation de code, machines vulnérables exécutables, etc.) pourront être ajoutés ultérieurement.

Les défis porteront sur une large variété de sujets, représentatifs des compétences techniques et conceptuelles à acquérir :

- Mathématiques / Algorithmique / Logique
- Défis de code (ex. résolution de problèmes en Python, Java, C) - format *flag* pouvant résulter d'un output correct

- Sécurité informatique : trouver un FLAG dans un script, une image (stéganographie), un binaire, etc.
- DevOps : compréhension de Dockerfile, principes CI/CD, conteneurisation (exercices théoriques ou d'analyse)
- IA / Data : tâches d'analyse de données ou d'interprétation de modèles produisant un flag
- Réseau / Web / Cryptographie / Reverse engineering
- Pentest : challenges simulant des machines ou scénarios vulnérables (à adapter selon les contraintes d'hébergement et de sécurité)

Titre	Difficulté	Points suggérés	Durée estimée	Catégorie(s)
Capture Réseau	Moyen (6/10)	150	10 - 25 min	Réseau
JS Algo Fix	Moyen - Difficile (7/10)	200	20 - 45 min	JavaScript / Dev (algos, asynchrone, Unicode)
Jeu de piste Web - À la recherche du flag	Moyen (6/10)	150	10 - 25 min	Web / Recon / OSINT (HTML/CSS/EXIF)
Cryptographie - L'Énigme de l'étudiant disparu	Moyen (6/10)	150	20 - 35 min	Cryptographie

3.3. Les défis

3.3.1. Analyse de capture réseau

Titre : Capture Réseau
Difficulté : Moyen (6/10)
Points suggérés : 150
Durée estimée : 10 - 25 minutes (niveau CTF intermédiaire)

Description :
Un fichier de capture réseau (.pcap) a été mis à disposition. Il contient du trafic réseau varié (ARP, ICMP, DNS, HTTP, TCP, etc.).
Parmi toutes les trames, une seule contient le vrai flag complet, les organisateurs ont aussi dispersé plusieurs leurres (faux flags) pour m'embrouiller.
Ton objectif : analyser la capture et soumettre la chaîne exacte respectant le format du flag.

Format du flag attendu :

```
FLAG-SHARK{...}
```

Conditions :
Le fichier .pcap est fourni sur la page de téléchargement du challenge.
Il n'y a qu'un seul flag valide dans l'ensemble du fichier ; les autres occurrences sont des leurres.

Indices (progressifs) :
Indice 1 : «Certains drapeaux peuvent sembler correct cependant, un seul est véridique. »
Indice 2 : « la relation client-serveur peut apporter un cookie suspect. »
Indice Réponse - « Utilise tshark / Wireshark pour rechercher http.cookie. »

Outils recommandés : Wireshark (GUI), tshark, scapy / Python (pour analyser/automatiser).

Compétences requises : Connaissance basique de Wireshark/tshark ; scripting Python utile mais pas obligatoire.

Solution de test interne :

- Ouvrir ctf_shark_capture.pcap dans Wireshark.
- Filtrer :

```
http.cookie
```

- Identifier le paquet avec Host: **hidden.ctf.local.**
- Lire le champ Cookie → **hidden=FLAG-SHARK{f4k3_tr4ff1c_h1dd3n_fl4g}**
  C'est le flag valide.
  /!\ Tous les autres cookies (FAKE-SHARK{...}) sont des leurres.

3.3.2. JS Algo Fix

Titre : Js Algo Fix
Difficulté : Moyen - Difficile (7/10)
Points suggérés : 200
Durée estimée : 20 - 45 minutes (niveau BUT2 / BUT3)

Description :
Tu disposes d'une page web contenant trois mini-exercices JavaScript (trois éditeurs). Chaque éditeur contient une fonction volontairement boguée (algorithme, manipulation de chaînes Unicode, ou logique asynchrone).
Ton objectif : corriger chaque fonction pour qu'elle passe les tests automatisés intégrés. Les tests s'exécutent dans un sandbox (iframe) et s'affichent sur la page - quand les 3 niveaux sont validés, le flag final apparaît en récompense.

Format du flag attendu :

```
FLAG-ALGOJS-MASTERY
```

Fichiers fournis / Conditions :

- Un paquet statique (HTML / CSS / JS) à ouvrir localement dans un navigateur moderne (Chrome/Firefox).
- Les snippets initiaux sont **sans flag** , le flag n'est jamais codé dans les exercices.
- Les tests sont exécutés côté client dans un iframe sandbox ; le joueur ne doit pas modifier le mécanisme de test pour tricher.
- Plusieurs tests sont exécutés par niveau ; l'interface affiche pour chaque test : l'entrée, la valeur attendue et la valeur obtenue.

Niveaux (résumé) :

1. **Moyenne pondérée** : corriger weightedAverage(grades) (boucles / accumulation / floating).

2. **Palindrome avancé** : corriger isPalindrome(str) pour gérer majuscules, espaces et accents (normalisation Unicode).
3. **Compteur asynchrone** : corriger makeAsyncCounter(step) (closures + async/await / séquencement temporel).

Indices (progressifs) :

Indice 1 : « Vérifie que tu accumules correctement toutes les quantités nécessaires (somme et coefficients). »
Indice 2 : « Normalise les chaînes Unicode (accents → base letters) avant de tester le palindrome. »
Indice 3 : « setTimeout ne suffit pas pour attendre - utilise une promesse / await pour synchroniser l'incrément. »
Indice Réponse : « Ouvre la console devtools si besoin ; les tests s'exécutent dans un iframe et renvoient leurs résultats via postMessage. »

Outils recommandés :

- Un navigateur moderne (Chrome / Firefox) avec devtools.
- Un éditeur de texte pour corriger les snippets (VS Code, etc.).
- Connaissances de base en JavaScript (ES6+), Promises / async-await, et Unicode.

Compétences requises :

- Manipulation de tableaux et d'objets en JavaScript (reduce / for/of).
- Connaissance de la normalisation Unicode (String.prototype.normalize) et des regex.
- Compréhension des closures et de l'asynchronisme en JavaScript.
- Savoir lire la sortie des tests et itérer rapidement sur le code.

3.3.3. Jeu de piste Web

Titre : À la recherche du flag
Difficulté : Moyen (6/10)
Points suggérés : 150
Durée estimée : 10 - 25 minutes (niveau CTF intermédiaire)

Description :

Un simple lien vers un site web intrigue, vous êtes libre d'y donner un contexte ou non.

Il s'agit en apparence d'une page basique, mais plusieurs indices cachés s'y trouvent.
Le but du joueur est de fouiller le site sous toutes ses coutures pour retrouver le flag complet.

L'utilisateur devra explorer différents aspects de la page :

- La console du navigateur (messages console.log())
- Le code source HTML et CSS via l'inspecteur
- Les métadonnées EXIF d'une image présente sur le site
- D'éventuelles ressources externes (scripts, fichiers texte ou commentaires cachés)

Chaque élément découvert permet de reconstituer une partie du flag final.
Le défi vise à initier les joueurs aux réflexes d'analyse et de recherche en web.

N'hésitez pas à guider l'utilisateur dans sa recherche.

Format du flag attendu :

```
FLAG-WEB-PUZZLE{partie1_partie2_partie3}
```

Fichiers fournis / Conditions :

- L'image contenant les métadonnées est fournie, intégrez la dans le site. Vous pouvez en intégrer d'autres, sans données nécessaires à la complétion du défi.
- La page principale du site contient toutes les ressources nécessaires (HTML/CSS/JS/images).
- Le flag n'est pas présent directement dans le texte de la page.
- L'utilisateur doit combiner trois indices distincts pour obtenir le flag complet

Indices (progressifs) :

Indice 1 : "Le développeur a laissé un mot dans la console."
Indice 2 : "Certains secrets se glissent dans les commentaires du code."
Indice 3 : "Une image en dit plus qu'elle ne montre : regarde dans ses métadonnées EXIF." (chercher l'attribut "Description" de l'image avec le logiciel exiftool)
Indice Réponse : "Inspecte le code HTML, ouvre la console, puis vérifie les métadonnées EXIF de l'image."

Outils recommandés :

- Navigateur moderne (Chrome, Firefox, Edge).

- Console développeur (F12 → Console).
- Inspection du code (clic droit → Inspecter).
- Outil d'analyse EXIF (exiftool ou site en ligne dédié).

Compétences requises :

- Connaissance de base en développement web (HTML, CSS, JS).
- Capacité à utiliser les DevTools d'un navigateur.
- Bases de l'analyse d'images et des métadonnées.
- Esprit logique et curiosité technique.

Solution interne (test) :

1. Ouvrir la console → message : "FLAG-WEB-PUZZLE{fouille_".
2. Inspecter le code source → commentaire caché : <!-- bien -->.
3. Charger l'image banner.jpg dans un lecteur EXIF → champ "Description" : _partout}.
4. Flag final : `FLAG-WEB-PUZZLE{fouille_bien_partout}`.

3.3.4. Cryptographie

Titre : L'Énigme de l'étudiant disparu
Difficulté : Moyen (6/10)
Points suggérés : 150
Durée estimée : 20-35 minutes (niveau CTF intermédiaire)

Description :

Un étudiant brillant de BUT Informatique en 3ème année nommé Alexandre Moreau a mystérieusement quitté l'IUT de Vannes il y a trois mois. Avant son départ, connu pour ses compétences exceptionnelles en cryptographie, il a laissé derrière lui une clé USB contenant un message chiffré.
Les enseignants ont retrouvé cette clé USB abandonnée dans la salle B126 .
Sur la clé, un fichier texte contient un message énigmatique d'Alexandre qui défie les étudiants de déchiffrer un mot de passe protégé par plusieurs couches de chiffrement successives.
Le message d'Alexandre indique qu'il a utilisé 4 méthodes de chiffrement et d'encodage classiques pour sécuriser le mot de passe. L'étudiant devra identifier et déchiffrer chaque couche dans le bon ordre pour reconstituer le flag final.

Le défi vise à initier les joueurs aux techniques de cryptanalyse multi-couches et à l'utilisation d'outils de déchiffrement.

Format du flag attendu :

```
FLAG{mot_de_passe_simple_a_decrypter}
```

Fichiers fournis / Conditions :

- **alexandre_message.txt** : Fichier contenant l'histoire, le contexte et le message chiffré
- Le message chiffré est : `TmpNMk5UWmhOV1kzTkRjMU5XWTJOamN4TmprMk9UYzFOV1kyT1RjNU5qTTJOall5TnpVMVpqY3hOV1kzTkRjMU56TTJPRFptTmpZMllUYzFOamc9`
- L'utilisateur doit déchiffrer 4 couches successives : Base64 (x2) - Hexadécimal - Caesar (-3) - ROT13
- Le flag n'est pas présent directement dans le message, il faut déchiffrer pour l'obtenir
- Aucune autre ressource externe n'est nécessaires

Indices (progressifs) :

Indice 1 : "Le message contient uniquement des caractères alphanumériques. Il s'agit probablement d'un encodage Base64. Commence par le décoder."

Indice 2 : "Après le premier décodage Base64, tu obtiens encore du Base64. Décode-le une seconde fois pour obtenir une chaîne hexadécimale (0-9 et a-f uniquement)."

Indice 3 : "L'ordre complet de déchiffrement est : Base64 decode (x2), puis Hexadecimal decode, puis Caesar avec decalage -3, puis ROT13

Indice réponse : Le mot de passe final est : mot_de_passe_simple_a_decrypter. Il faut appliquer 5 étapes de déchiffrement successives. Le dernier resultat avant ROT13 est : `zbg_qr_cnffr_fvzcyr_n_qrpelcgre`

Outils recommandés :
CyberChef : https://gchq.github.io/CyberChef/ (outil tout-en-un pour dechiffrement multi-couches)
Base64 Decode : https://www.base64decode.org/ (decodeur Base64 en ligne)
Hex to ASCII : https://www.rapidtables.com/convert/number/hex-to-ascii.html (convertisseur hexadecimal)
dcode.fr : https://www.dcode.fr/ (identification automatique et dechiffrement)
Python 3 : Pour scripter le déchiffrement complet de maniere automatique
Ligne de commande Linux/Mac (commandes base64, xxd, tr pour ROT13)

Compétences requises :

- Reconnaissance de formats d'encodage (Base64, Hexadécimal)
- Connaissance des chiffrements classiques par substitution (Caesar, ROT13)
- Utilisation d'outils de déchiffrement en ligne ou en ligne de commande
- Méthodologie de cryptanalyse par couches successives
- Capacité à scripter avec Python pour automatiser les opérations (optionnel mais recommandé)
- Esprit logique, persévérance et capacité d'analyse

Solution interne (test)

Etape 1 : Premier décodage Base64

Commande à exécuter dans le terminal :

```
echo "TmpNMk5UWmhOV1kzTkRjMU5XWTJOamN4TmprMk9UYzFOV1kyT1RjNU5qTTJOall5TnpVMVpqY3hOV1kzTkRjMU56TTJPRFptTmpZMllUYzFOamc9" | base64 -d
```

Resultat obtenu :

```
NjM2NTZhNWY3NDc1NWY2NjcxNjk2OTc1NWY2OTc5NjM2NjYyNzU1ZjcxNWY3NDc1NzM2ODZmNjY2YTc1Njg=
```

Etape 2 : Deuxième décodage Base64

Commande à exécuter dans le terminal :

```
echo "NjM2NTZhNWY3NDc1NWY2NjcxNjk2OTc1NWY2OTc5NjM2NjYyNzU1ZjcxNWY3NDc1NzM2ODZmNjY2YTc1Njg=" | base64 -d
```

Résultat obtenu :

```
63656a5f74755f66716969755f6979636662755f715f747573686f666a7568
```

Etape 3 : Décodage Hexadécimal

Commande à exécuter dans le terminal :

```
echo
"63656a5f74755f66716969755f6979636662755f715f747573686f666a7568" |
xxd -r -p
```

Résultat obtenu :

```
cej_tu_fqiiu_iycfbu_q_tushofjuh
```

Etape 4 : Déchiffrement Caesar avec décalage -3

Code Python à exécuter :

```
text = "cej_tu_fqiiu_iycfbu_q_tushofjuh"
 result = ""
 for char in text:
      if 'a' <= char <= 'z':
           result += chr((ord(char) - ord('a') - 3) % 26 +
ord('a'))
      elif 'A' <= char <= 'Z':
           result += chr((ord(char) - ord('A') - 3) % 26 +
ord('A'))
      else:
           result += char
 print(result)
```

Résultat obtenu :

```
zbg_qr_cnffr_fvzcyr_n_qrpelcgre
```

Etape 5 : Déchiffrement ROT13

Commande à exécuter dans le terminal :

```
echo "zbg_qr_cnffr_fvzcyr_n_qrpelcgre" | tr 'A-Za-z'
'N-ZA-Mn-za-m'
```

Ou avec Python :

```
import codecs
```

```
 result = codecs.encode("zbg_qr_cnffr_fvzcyr_n_qrpelcgre",
'rot_13')
 print(result)
```

Résultat final : `mot_de_passe_simple_a_decrypter`

Flag final

```
FLAG{mot_de_passe_simple_a_decrypter}
```

De plus, un fichier python "decrypt_message.py" vous est fourni.

3.4. Performance et sécurité

Performance : Le temps de chargement des pages ne doit pas dépasser 3 secondes pour une expérience utilisateur optimale. La plateforme doit être capable de supporter jusqu'à 200 utilisateurs actifs simultanément.
Sécurité : La plateforme devra respecter les bonnes pratiques de sécurité (protection contre les injections SQL, XSS)

3.5. Maintenabilité

Le code doit être clair et commenté, les méthodes documentées.
L'API doit avoir une documentation au format OpenAPI de préférence

3.6. Tests

Tests unitaires : L'api doit être testée par des scripts de tests automatisés.
Tests de régression visuelle : Pour le frontend, on vous suggère (optionnel) de mettre en place des tests par capture d'écran, avec des outils comme Sélénium ou encore BackstopJS
Tests utilisateurs : Un questionnaire de satisfaction doit être élaboré et recueillir au moins 5 réponses de participants externes au projet. Ce questionnaire doit évaluer la facilité ou la difficulté rencontrée par les utilisateurs lors de l'accomplissement de tâches spécifiques. Il doit également inclure des questions sur leur perception globale de l'expérience utilisateur, afin d'identifier les points forts et les axes d'amélioration du produit.

3.7. Intégration continue & déploiement continue

Le projet Def'IUT doit intégrer une pipeline CI/CD avec les Github Actions pour automatiser les processus de test, de build et de déploiement. Nous vous demandons que la pipeline soit configurée pour :

- Exécuter automatiquement les tests à chaque push sur toutes les branches.
- Construire les images Docker à chaque push sur la/les branche(s) principale(s).
- Pousser automatiquement les images Docker construites vers le registre d'image GitHub Container Registry

4. Licence

Le projet Def'IUT sera publié sous la licence GNU General Public License v3.0 (GPL-3.0). En effet, elle est adaptée pour le contexte éducatif du projet et garantit que les travaux dérivés resteront open sources, et assure que le projet puisse être modifié pour une probable reprise de code.

4.1. Implications de la licence

Tout le code source du projet doit être rendu disponible sous la licence GPL-3.0.

- Les modifications et travaux dérivés doivent également être distribués sous la même licence.
- Les utilisateurs ont le droit d'utiliser, modifier et redistribuer le code, à condition de respecter les termes de la licence

Plus d'information sur la licence peut être trouver sur le site officiel : https://www.gnu.org/licenses/gpl-3.0.fr.html

4.2. Application de la licence

Un fichier LICENCE contenant le tête intégrale de la licence GPL-3.0 doit être inclus à la racine des dépôts GitHub du projet. Chaque fichier source doit inclure une notice de copyright et une référence à la licence GPL-3.0 en en-tête.

5. Livrables

5.1. Livrables attendus

Nous attendons les livrables suivants au format demandé :

- **Code source** : hébergé sur GitHub. Vous devrez créer un dépôt et nous y inviter pour que nous suivions l'avancement.
- **Maquettes UI** : De préférence conçues sur Figma ou au minimum fournies au format PDF.
- **Documentation technique explicative** sur :
  - ➔ les choix technologiques (si différent de ceux suggérés)
  - ➔ l'architecture de l'application
  - ➔ l'installation et le déploiement via Docker (manuel d'installation)
- **Documentation utilisateur** : permettant à l'utilisateur de comprendre les différentes fonctionnalités et comment naviguer dans l'application.
- **Docker** : L'application doit pouvoir être déployée avec 2 dockers image, un pour le frontend et une pour le backend. Un fichier docker-compose doit permettre d'orchestrer le démarrage des conteneurs et faciliter la communication entre les composants de l'application.
- **Tests** : Codes de tests + Résultats d'exécutions sous forme de rapport + Rapport de performances.
- Un **README.md** de page d'accueil du github présentant le projet

5.2. Format des livrables

La documentation devra impérativement contenir les parties suivantes afin d'assurer une homogénéité de cette dernière mais également faciliter sa maintenabilité.

Introduction

Informations générales

Nom du document	Manuel d'installation de l'application Def'IUT
Type de document	Manuel d'installation
Version	v1.0.0
Date de création	15/09/2025
Rédacteur	Jade COUTANT

Relecteur	Ayette EL HOMADI
Status	Terminé

Signalétiques

Icône	Description
	*Note d'information : fournit des précisions ou des détails complémentaires utiles à la compréhension*
	*Note d'avertissement : signale des points critiques ou des précautions à prendre*

Tableau de références

Code	Nom du document	Emplacement du document

Historique des versions

Version	Description des changements	Date
v1.0.0	Version initiale du document	25/09/2025

6. Communication et suivi

- **Suivi de projet** : Le suivi se fera via GitHub avec un tableau de progression (Kanban ou autre). L'équipe prestataire est responsable de la création du dépôt et devra nous y inviter dès le début du projet.

- **Réunions** : Nous prévoyons des points d'étape, que ce soit en présentiel ou en visioconférence, pour assurer le bon déroulement du projet et répondre aux éventuels blocages techniques.
- **Communication continue** : Nous nous chargeons de mettre en place un serveur Discord pour faciliter les échanges en continu. Vous pourrez nous poser tous types de questions et échanger sur l'avancée du projet.
  Lien du serveur discord : https://discord.gg/sVrr4mpg