Page de garde

Remerciement

Contents

[Description du projet : Isidor’s Quest : Chasing the glow 5](#_Toc155514142)

[Pourquoi ce projet et pas un autre ? 5](#_Toc155514143)

[Phases d’analyses 6](#_Toc155514144)

[Le marché actuel 6](#_Toc155514145)

[Ce qui fait notre singularité 7](#_Toc155514146)

[Choix techniques 7](#_Toc155514147)

[Le rôle de chaque composant 7](#_Toc155514148)

[Pourquoi ces outils plutôt que d’autres ? 8](#_Toc155514149)

[Charte graphique 10](#_Toc155514150)

[Notre organisation 11](#_Toc155514151)

[Planification 11](#_Toc155514152)

[Rôles de chacun(e) 11](#_Toc155514153)

[Répartition des tâches 12](#_Toc155514154)

[Nos deux applications 13](#_Toc155514155)

[Les services proposés (architecture fonctionnel) 13](#_Toc155514156)

[Site 13](#_Toc155514157)

[Jeu 13](#_Toc155514158)

[Nos phases de développement 14](#_Toc155514159)

[Conception des applications (architecture technique) 14](#_Toc155514160)

[Site 14](#_Toc155514161)

[Jeu 14](#_Toc155514162)

[Incorporation du jeu sur le site 14](#_Toc155514163)

[Les tests utilisateurs 14](#_Toc155514164)

[Bilan de montée en compétences 15](#_Toc155514165)

[Les difficultés rencontrées 15](#_Toc155514166)

[Ce qui reste à accomplir 15](#_Toc155514167)

[Perspectives d’amélioration 15](#_Toc155514168)

[List des annexes 16](#_Toc155514169)

[Annexe 1 : Résumé 17](#_Toc155514170)

[Annexe 2 : Abstract 17](#_Toc155514171)

[Annexe 3 : Les sources 18](#_Toc155514172)

[Annexe 4 : Images du digramme de Gantt 20](#_Toc155514173)

[Annexe 5 : Table des illustrations 22](#_Toc155514174)

[Annexe 6 : Cahier des charges 22](#_Toc155514175)

[Annexe 7 : Proposition de projet SAE S5 22](#_Toc155514176)

[Annexe 8 : Poster 22](#_Toc155514177)

Introduction

# Description du projet : Isidor’s Quest : Chasing the glow

Dans le cadre de notre troisième année de BUT informatique, il nous est demandé de concevoir une application qui est alignée avec nos projets professionnels et nos passions personnelles.

Notre jeu (qui sera accessible au travers d’un navigateur) prendra la forme d’une expérience en 2D avec un style pixel art. Au début d’une nouvelle partie, l’utilisateur sera invité à choisir parmi deux classes de héros : le guerrier et l’archer. Le joueur sera ensuite confronté à des monstres et à des énigmes de difficulté croissante, nécessitant des compétences et de la stratégie pour accéder aux niveaux suivants.

# Pourquoi ce projet et pas un autre ?

Après une longue phase de réflexion, notre choix s’est porté sur le domaine du jeu vidéo qui possède à lui tout seul une part du marché non négligeable ! De plus ce domaine suscite un intérêt majeur au sein de notre équipe. Malgré l’immensité des possibilités de création de jeux, nous avons finalement choisi de nous orienter vers un style emblématique, à l’instar de *Mario*, *Kirby* ou encore *Sonic*. Nous avons tous déjà joué à ces jeux plus petits et ce serait un rêve de pouvoir réaliser un tel type de jeu. De plus, la plupart des langages abordé par le projet ne sont pas tous maîtrisés par les membres de l’équipe. C’est donc aussi une occasion idéale d’enrichir notre CV.

Développement

# Phases d’analyses

## Le marché actuel

En 2023, les recettes du secteur jeux vidéo sont estimées à 245.10 milliards de dollars américains et devraient même atteindre 378.08 milliards de dollars américains en 2028. Ce chiffre s’explique par l’augment des plateformes disponibles de nos jours, on peut désormais jouer à des jeux plutôt puissants sur nos smartphones, ce qui n’était pas possible dans les années 80 et 90. De plus, la diversité de type de jeux attire de nombreux consommateurs, en 2022, il existe un peu moins d'1,8 milliard de joueurs dans le monde.

A screenshot of a screen

Description automatically generated

Figure : Bilan du marché Français 2020 dans le secteur du jeu vidéo. [L’essentiel-jeu-vidéo [Fichier PDF]. Page 8. 2021.](https://www.sell.fr/sites/default/files/essentiel-jeu-video/ejv_mars_21_web.pdf)

## Ce qui fait notre singularité

L'intégration d'un jeu sur un site web est courante, comme c'est le cas sur des plateformes comme jeux.fr. Cependant, nous ne sommes pas pareils. Notre jeu sera certes aussi sur un site web, mais il ne sera pas regroupé avec d'autres jeux. Il s'agit plutôt d'un site que nous avons développé nous-mêmes.

Rien que sur le plan du design, notre jeu se démarque déjà sur le marché. Certes, il s'agit d'un jeu de plateforme 2D que l'on retrouve fréquemment, mais le nôtre est également composé de pixel art. De plus, le mécanisme de défense est rare dans ce type de style, sans même parler du « village », une zone où les joueurs peuvent effectuer diverses actions auprès des PNJ (Personnages Non-Joueurs), telles que l'amélioration de leur arbre de compétences ou encore l'achat d'objets avec l'or obtenu dans les différents niveaux.

## Choix techniques

### Le rôle de chaque composant

#### Jeu

Unity Engine : pour le développement du jeu, tant sur le point UI/UX que programmation

Visual Studio : Environnement de développement conseillé par Unity

Piskelapp : Permet de créer des designs en 2D pixelisé

Pinetools : Utiliser pour pixeliser une image

OpenGameArt & CraftPix : Recherches de Sprite (design utilisé pour le jeu)

#### Web

Figma : Maquettage du site

React Native : Framework utilisé pour développer la partie frontend

Node.JS : Plateforme de développement utilisé pour la partie backend

Visual Studio code : Environnement de développement utilisé pour le site

MongoDB : Base de données non relationnelle, utilisée pour stockage des données des utilisateurs

#### Gestion et organisation

Trello : Organisation des taches

Google Drive : Echange de document

GitHub : Gestionnaire de version

Discord : Communication

Notion : Création du diagramme de Gantt

Visual Paradigm : Mise en place des diagrammes UML

### Pourquoi ces outils plutôt que d’autres ?

#### Jeu

Lorsqu'on parle de développement de jeu, on pense tout de suite à Unity Engine et Unreal Engine, mais quelle est la différence et pourquoi avons-nous choisi Unity ? **Unity** possède une plus grande communauté qu'Unreal, il nous est donc plus facile de nous y documenter. De plus, Unity est plus adapté pour les débutants dans le monde du jeu vidéo, contrairement à Unreal qui est beaucoup plus orienté vers des jeux de type AAA (jeux développés et produits par des grandes entreprises de l’industrie du jeu).

Au niveau des designs, nous nous sommes dit qu'il serait mieux de les importer directement d'internet afin de gagner du temps pour le développement. Nous nous sommes alors tournés vers des sites tels que **OpenGameArt** et **CraftPix** qui offrent de nombreux éléments graphiques en libre accès. Il nous arrive de trouver des ressources qui ne sont pas en pixelart, c'est pourquoi nous utilisons un logiciel de pixélisation : **Pinetools**, qui est l'une des seules applications que nous avons testées et qui ne déforme pas l'image originale. Si aucune recherche ne correspond au design souhaité, nous le créons alors de zéro avec **Piskelapp**.

#### Web

Créer des maquettes d’application web en collaborant avec les autres membres, tout en étant gratuit et accessible à tous ? **Figma** est alors l’outil parfait ! Contrairement à ses concurrents, qui sont payants ou qui deviennent payantes après une phase de test. Figma est un outil facile à prendre en main et permet aussi de réaliser des maquettes pour applications mobiles.

Côté frontend, nous avons choisi d'utiliser le Framework **React Native** qui se base sur le langage Javascript. Nous nous sommes orientés dans cette direction, d'une part parce que cet outil est l'un des plus utilisés et demandés sur le marché, d'autre part, contrairement à d'autres Framework, React permet de développer des pages interactives très optimisées grâce aux mises à jour rapides qu'il apporte. De plus, contrairement à ReactJS, React Native permet non seulement de développer une application utilisable sur un navigateur, mais aussi de la compiler en tant qu'application mobile.

Nous avons choisi **Node.js**, accompagné du Framework Express.js (le langage de programmation est aussi le Javascript) pour faciliter la création et la gestion rapide des serveurs recevant les requêtes HTTP, la gestion des sessions utilisateur, ainsi que l'utilisation des cookies. De plus, il s'agit de l'un des backends les plus utilisés en combinaison avec React.

**MongoDB** est la base de données utilisée pour stocker les données utilisateur. Elle présente une particularité : elle est non relationnelle, ce qui signifie qu'elle ne suit pas le schéma tabulaire sous forme de lignes et de colonnes communément rencontré dans la plupart des systèmes de bases de données. Cette approche offre une plus grande flexibilité et évolutivité. De plus, Node.js propose une bibliothèque appelée Mongoose qui simplifie grandement la réalisation d'opérations sur MongoDB.

Nous travaillons avec l'éditeur de code **Visual Studio Code**, principalement pour la raison de sa légèreté et des extensions qu'il propose. Parmi ces extensions, Simple React Snippets se démarque en permettant un développement plus rapide grâce à l'utilisation de mots-clés. Par exemple, pour créer un composant, au lieu de tout saisir manuellement, il suffit de taper "rcs" puis tabulation.

#### Gestion et organisation

**Trello** est le meilleur choix pour la gestion de projets simples. Il suffit de créer des tableaux par thèmes généraux et d'y ajouter des étiquettes. Grâce à cette organisation, il devient aisé de suivre l'avancement de notre projet en naviguant d'un tableau à l'autre.

Qui ne connaît pas Google ? Un des GAFAM ! L’entreprise a développé une application, **Google Drive**, qui permet également de collaborer avec les membres de l’équipe en plus de pouvoir créer, éditer et supprimer différents types de fichiers (documents, tableurs, présentations). Nous utilisons cet outil car c’est celui qui offre le plus d’espaces de stockage sans frais, avec 15 GO.

Les outils de versionning sont nombreux, pourtant il y en a certains qui sont plus pertinents que d’autres, comme la gestion de version décentralisée (chaque développeur possède ses propres dépôts et copie locale), ce qui permet à chacun de travailler à son rythme. Nous nous sommes orientés vers Git, et plus précisément **GitHub**. Offrant la possibilité de créer plusieurs branches locales indépendantes, un tableau de bord personnel pour suivre les issues et les pull requests. De plus, si un problème survient, aucun souci, du fait de la grande communauté qui s’y trouve.

**Discord**, un outil de communication, de visioconférence et d’envoi de fichiers, se distingue par sa capacité à évoluer constamment grâce à la possibilité d’ajouter des bots sur nos serveurs. Étant tous sur la plateforme depuis plus de 2 ans, nous l’avons sélectionnée sans hésitation.

Lorsqu’on évoque **Notion**, l’outil no-code de productivité vient immédiatement à l’esprit. Néanmoins, elle permet aussi de faire des diagrammes de Gantt. Cette fonctionnalité proposée par l’application a été notre dernier recours. En effet, nous avons dans un premier temps porté nos recherches sur d’autres applications, mais elles étaient soit payantes, soit mal conçues (constitution du diagramme avec un assemblage de plusieurs rectangles).

Concernant l’outil pour les diagrammes UML, il y a mieux que **Visual Paradigm**, comme Lucidchart. Cependant, son utilisation est restreinte. Nous pouvons uniquement créer 3 documents et pas plus. Alors que Visual Paradigm permet d’ajouter et de confectionner une infinité de fichiers tant que la limite de stockage n’est pas dépassée (1 GO).

## Charte graphique

Comme pour tous les jeux, notre logo est composé de notre personnage, ou du moins d'une partie de celui-ci. Nous avons uniquement gardé la "partie supérieure" de notre héros, simplement pour des raisons esthétiques et ergonomiques. Notre palette de couleurs a justement été reprise de celui-ci. Étant donné que notre jeu est un jeu 2D de style pixel art, rien de mieux qu’une typographie pixelisé.

A screenshot of a video game

Description automatically generated

Figure : charte graphique utilisée pour l'application

## Notre organisation

### Planification

Avant de commencer le projet, nous avons mis en place un diagramme de Gantt que nous avons intégré sous forme d’un tableau dans le cahier des charges (voir en annexe 6). Bien que celui-ci soit accompagné d’un lien, ce dernier est devenu obsolète, et la lecture du diagramme sur un tableau n’est pas la plus idéale. Nous avons donc ajouté en annexe 4 des images illustrant le diagramme de Gantt. Celui-ci est présent à titre indicatif, mais au cours du projet, nous rajouté des tâches et réalisé des fonctionnalités plus tôt que la date indiquée sur le diagramme.

Lors de la rédaction du cahier des charges, nous avons également consacré un certain temps à la composition du cahier de recette (voir en annexe 6).

### Rôles de chacun(e)

Outre le fait que le projet doit correspondre à nos envies personnelles et professionnelles, c’est avant tout une manière d’apprentissage. Nous avons donc chacun choisi plusieurs rôles, qui nous semblent pertinents (voir en annexe 6 dans le cahier des charges).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | COLLOMBET Nathan | LIN  Oscar | SELVARATNAM Akash | ZHANG Anxian | LIN  Xingtong |
| Web designer | ✓ |  | ✓ | ✓ | ✓ |
| Game designer |  |  |  | ✓ | ✓ |
| Développeur Web | ✓ | ✓ | ✓ | ✓ | ✓ |
| Développeur Logiciel |  | ✓ | ✓ | ✓ | ✓ |
| Testeur |  |  |  | ✓ |  |

### 

### Répartition des tâches

Dans notre organisation il n’y a pas à proprement parler de chef de projet. Nous utilisons une application de gestion des tâches, Trello, qui nous permet d’ajouter des missions et de les labelliser avec un des termes suivants « Terminé », « Pas possible pour le moment », « En cours », « Bloqué » et « À faire ». Cela signifie qu’une fois qu’une mission a été sélectionnée ou commencée par une personne, elle doit se déclarer en tant que membre et indiquer l’état d’avancement sur la carte afin de ne pas refaire des tâches qui ont déjà été réalisées.

A screenshot of a computer

Description automatically generated

Figure : exemple de tâche effectuée et labellisée « Terminer »

# Nos deux applications

## Les services proposés (architecture fonctionnel)

### Site

Nos différentes pages web :

* Accueil : introduction à l’histoire du jeu et aux personnages jouables.
* À propos : présente le projet, l’utilité du site et de la gestion des données de l’utilisateur.
* Connexion : l’utilisateur doit se connecter pour accéder au jeu.
* Inscription : s’il ne possède pas de compte, il peut s’inscrire.
* Oublie de mot passe : permet la réinitialisation du mot de passe.
* Modification des données utilisateur : offre la possibilité de modifier le pseudonyme ainsi que le nom et prénom associés à l'utilisateur.
* Jeu : contient notre jeu Unity au format WebGL.
* Paiement : propose deux différents types de paiement (PayPal ou par carte bancaire).
* Contact : si l’utilisateur a besoin d’une quelconque aide, il peut nous solliciter sur cette page.
* Vérification : Page permettant de vérifier le code entré par l’utilisateur avec celui envoyé sur son mail lors de l’inscription ou lors d’une modification de mot de passe.

### Jeu

Nos différentes scènes :

* Accueil : offre la possibilité de commencer une partie ou de configurer les paramètres du jeu, comme le mode plein écran ou le réglage du volume.
* Sélection de personnages : permet de choisir entre les deux classes disponibles, à savoir **l’archer** et le **guerrier**
* Village : zone non hostile où les joueurs peuvent **améliorer leur arbre de compétences** ou **acheter des objets consommables**, notammentles potions.
* Les niveaux de la première région :
  + Niveau 1 : premier niveau du jeu
  + Niveau 2 : deuxième niveau du jeu

### Base de données

Nos différentes documents :

* User : données de l’utilisateur sur le site web permettant sa connexion ou son inscription par exemple
* Code : sauvegarde des codes de vérification lors de l’inscription d’un utilisateur avec une data d’expiration
* Game : sauvegarde des données d’une partie sur le jeu
* UserGame : sauvegarde des données de l’utilisateur sur le jeu tel que le nombre de pièces ou les niveaux des classes de personnages

## Nos phases de développement

Nous avons commencé par une **recherche et la définition** des objectifs à travers un premier document : « Proposition de projet SAE S5 », qui se trouve en annexe 7. Ensuite, nous avons rédigé le cahier des charges en y intégrant la **planification**. Entre-temps, une personne a mis en place une page Figma pour créer une première représentation du site web. En ce qui concerne le jeu, certaines personnes ont conçu l'UI et l'UX des niveaux, ainsi que les menus directement sur Unity. Enfin, après avoir plus ou moins terminé la conception esthétique des applications, nous sommes passés à la phase de **développement**, puis effectué des **tests unitaires** avec la bibliothèque Jest sur les composants du site.

## Conception des applications (architecture technique)

### Site

Termes techniques (à ajouter plus tard dans le glossaire):

* useState : détecte localement (limité à des composants) si un état a été modifié
* useEffect : indique à React l’exécution d’une tâche à chaque fois qu’un état change, il gère donc les effets de bord
* TextInput : champ d’entrée de texte
* Text : composant permettant d’afficher du texte
* TouchableOpacity : permet de rendre un élément cliquable comme un bouton ou un lien par exemple, avec un effet d’opacité lors du clique.
* useNavigation : permet de naviguer entre les différentes pages du site web
* Regex : Un regex est une chaine de caractères représentant un ensemble de chaine de caractère possible.
* Nodemailer : Nodemailer est un module de Node.js permettant d’envoyer des mails.
* Express : Framework permettant de construire des application web pour Node.js, il s’agit d’un Framework permettant d’exécuter des requêtes HTTP
* Express-session : Librairie proposé par Express permettant de sauvegarder des données directement dans un cookie
* useRef : permet de référencer une valeur (composants, variables…)
* paypal-rest-sdk : Librairie proposé par PayPal permettant de réaliser des paiement PayPal
* Stripe : Librairie proposé par Stripe permettant de réaliser différents type de paiement

Inscription :

Lors de l’inscription, l’utilisateur doit remplir un formulaire comportant différents champs tels que le nom, le prénom, l’adresse mail, le nom d’utilisateur ainsi que le mot de passe. Si l’ensemble de ces champs sont correctement remplie par l’utilisateur alors l’utilisateur sera redirigé vers la page de vérification de code, si le mail entré par l’utilisateur existe dans notre base de données alors l’utilisateur sera invité à se connecter avec son compte plutôt que de s’inscrire, si le pseudo entré par l’utilisateur existe dans notre base de données, alors l’utilisateur devra changer son pseudo car nous avons opté avec l’équipe pour un nom d’utilisateur unique et si le mot de passe entré par l’utilisateur est différent de celui indiqué dans le champs confirmation de mot de passe alors un message d’erreur indiquera à l’utilisateur que les deux mot de passe ne sont pas identique. Cette page a été réalisée avec les composants TextInput pour les six différent champs d’entrée, un composant Text pour le titre « S’inscrire » et un TouchableOpacity pour le bouton. Nous utilisons également les hooks useState pour pouvoir gérer les différents états des variables et useEffect pour pouvoir gérer l’état du bouton ainsi que de la gestion de l’état du mail qui doit comporter un texte avant et après le caractère @ et d’un point avec deux ou trois caractères à la suite, cette vérification du mail est possible avec un regex.

Une image contenant texte, capture d’écran, nombre, conception

Description générée automatiquement

Figure : page d'inscription, exemple avec l'erreur nom d'utilisateur déjà pris

Vérification du code :

Après avoir rempli les champs pour que l’utilisateur puisse s’inscrire, l’utilisateur sera redirigé vers une page permettant de vérifier que le code indiqué par l’utilisateur et bien conforme à celui envoyé sur sa boîte mail. Il s’agit d’une procédure permettant de vérifier que le mail indiqué par l’utilisateur lors du formulaire d’inscription est bien correct. Cette page a été réalisée avec les composants TextInput pour les deux différent champs d’entrée, un Text pour le titre « Vérification du code » et un TouchableOpacity pour le bouton. Nous utilisons également les hooks useState pour pouvoir gérer les différents états des variables et useEffect pour pouvoir gérer l’état du bouton.

A screenshot of a computer screen

Description automatically generated

Figure : page de vérification du code

Le code est envoyé sur la boîte mail de l’utilisateur avec la bibliothèque nodemailer de node.js, il s’agit d’un code d’une taille de six caractères générés aléatoirement. Si le code entré par l’utilisateur est correct, cela créera son compte dans le document User de notre base de données avec les données fourni par l’utilisateur (le mot de passe fournie par l’utilisateur sera crypté dans la base de données) puis l’utilisateur sera redirigé vers la page d’accueil avec un useNavigation, dans le cas contraire un message d’erreur s’affiche indiquant que le code entré est incorrect.

A screenshot of a chat

Description automatically generated

Figure : Exemple de mail envoyé à un utilisateur avec le code à entrer

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Figure : Exemple d'utilisateur dans le document User

Connexion :

L’utilisateur est invité à entrer son nom d’utilisateur ainsi que son mot de passe, si les données sont correct alors l’utilisateur sera redirigé vers la page d’accueil et nous enregistrerons dans les cookies au niveau du back-end, le pseudo de l’utilisateur avec la bibliothèque express-session, cela permet de savoir si l’utilisateur est connecté ou non. En cas d’erreur, le nom d’utilisateur ou le mot de passe est incorrect (différent de ce qui est présent dans la base de données) alors un message d’erreur est affiché sur les deux champs, l’utilisateur devra alors remplir à nouveau le formulaire, ou s’inscrire via le bouton « Créer un nouveau compte », ou se diriger vers la page d’oubli de mot de passe via le lien « Mot de passe oublié ? ». Cette page a été réalisée avec les composants TextInput pour les deux différent champs d’entrée, un Text pour le titre « Connexion » et trois différents TouchableOpacity pour les boutons et le lien menant vers la page d’oubli de mot de passe. Nous utilisons également les hooks useState pour pouvoir gérer les différents états des variables et useEffect pour pouvoir gérer l’état du bouton « Se connecter ».

Une image contenant texte, capture d’écran, Police, conception

Description générée automatiquement

Figure : Page de connexion, exemple de connexion avec un nom d'utilisateur ou un mot de passe incorrect

Oublie de mot passe :

Dans un premier temps, l’utilisateur est invité à saisir son courriel. L’adresse électronique doit être correcte, comportant un arobase et un nom de domaine. Tant que celle-ci est incorrecte, le bouton « Obtenir le code de vérification » restera désactivé. Cette fonctionnalité a été implémentée avec un useEffect, qui va faire une vérification à chaque mis à jour du TextInput par l’utilisateur.

A blue box with white text

Description automatically generated

Figure : page d'oubli de mot de passe, exemple d'un mail non valide

Lorsque le bouton est appuyé, l'utilisateur sera redirigé vers la page de vérification du code qui lui sera envoyé dans sa boîte mail. Une fois la vérification réussie, il pourra alors modifier son mot de passe. Ce changement sera traité au niveau du backend, c'est-à-dire, lorsque le bouton de modification sera pressé, une requête sera envoyée au serveur qui mettra à jour la base de données.

A propos :

La page à propos présente le contexte du projet, l’utilité du site ainsi que la gestion des données de l’utilisateur. Pour pouvoir réaliser cette page, nous avons utilisé différents composants React telles que le composants Text pour pouvoir présenté notre projet ainsi que la création de notre propre composant pour pouvoir réaliser la table de contenus qui nécessite le composant touchableOpacity pour les liens. Nous avons également utilisé le Hook useRef pour pouvoir naviguer vers le bloc correspondant lorsqu’on clique sur la table de contenus.

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Figure : Page à propos

Contact :

L’utilisateur est invité à remplir les différents champs du formulaire de contact avec son mail, son nom ainsi que le message à envoyer à notre service. Cette page utilise différents hooks tels que le useState pour gérer les valeurs des différentes variables ainsi qu'un useEffect pour pouvoir gérer l’état du bouton, nous utilisons le composant Text pour le titre « Formulaire de contact », les composants TextInput pour les différents champs d’entrée et un TouchableOpacity pour le bouton.



Figure : Page de contact

Lorsqu’e l’utilisateur remplie l’ensemble des champs du formulaire et appuie le bouton envoyer cela envoie un mail à notre service avec le message de l’utilisateur à l’aide de la bibliothèque nodemailer et un mail automatique est envoyé au mail indiqué par l’utilisateur sur le formulaire afin de confirmer la bonne réception du mail.



Figure : Mail de confirmation envoyé à l'utilisateur

Paiement :

Pour pouvoir acheter notre jeu, nous avons décidé de deux différentes moyen de paiement par carte bancaire ou par PayPal. Pour pouvoir réaliser cette page, nous avons utilisé deux TouchableOpacity pour pouvoir réaliser nos deux boutons.

Une image contenant texte, capture d’écran, Police, logo

Description générée automatiquement

Figure : Page de paiement

Lorsqu’on appuie sur le bouton « Payer par PayPal, cela nous redirige vers la page de connexion de paiement de PayPal puis après s’être connecte à son compte, l’utilisateur devra acheter le jeu sur PayPal, l’ensemble des opération, la création d’un paiement, la réussite ou l’échec d’un paiement est géré par la bibliothèque paypal-rest-sdk proposé par PayPal.

Lorsqu’on appuie sur le bouton « Payer par carte bancaire », cela nous redirige vers la page de paiement de Stripe, l’utilisateur sera invité à remplir le formulaire en fournissant son mail ainsi que les données de sa carte bancaire. L’ensemble des opération, la configuration et la création d’un paiement, l’échec ou la réussite d’un paiement est géré par la bibliothèque Stripe.

Pour les deux différents moyen de paiement, il s’agit de paiement de test donc les paiements ne sont pas réellement effectuer.

Une image contenant texte, capture d’écran, Page web, Police

Description générée automatiquement

Figure : Paiement par PayPal

Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement

Figure : Paiement par carte bancaire

Si le paiement à bien été effectué alors cela dirigera l’utilisateur vers la page indiquant que le paiement a bien été effectué avec un bouton permettant de télécharger sa facture pour le paiement par carte bancaire, pour le paiement par PayPal, la facture est directement envoyé sur son mail.

En cas d’échec de paiement, l’utilisateur sera dirigé vers la page indiquant que le paiement a été refusé.

Une image contenant texte, capture d’écran, Police, Bleu électrique

Description générée automatiquement

Figure :Paiement par carte bancaire réussie

### Jeu

Écran d’accueil :

Personnages : (Parler de la sélection + actions[attaque déplacement])

Interface utilisateur :

Mettre l’image avec légende

Les ennemis :

Coffres et pièces :

Environnements : (Plateforme et échelle portail de fin de niveau et l’eau)

Inventaire :

Village :

Magasin :

Arbre de compétences :

Portail de sélection de niveau :

## Incorporation du jeu sur le site

## Les tests utilisateurs

Conclusion

# Bilan de montée en compétences

# Les difficultés rencontrées

# Ce qui reste à accomplir

# Perspectives d’amélioration

# List des annexes

[Annexe 1 : Résumé 9](#_Toc154782996)

[Annexe 2 : Abstract 9](#_Toc154782997)

[Annexe 3 : Les sources 9](#_Toc154782998)

[Annexe 4 : Images du digramme de Gantt 9](#_Toc154782999)

[Annexe 5 : Table des illustrations 9](#_Toc154783000)

[Annexe 6 : Cahier des charges 9](#_Toc154783001)

[Annexe 7 : Poster 9](#_Toc154783002)

# Annexe 1 : Résumé

# Annexe 2 : Abstract

# Annexe 3 : Les sources

Collectif. « Le marché du jeu vidéo – Faits et chiffres ». 13 décembre 2023. Statista.

<https://fr.statista.com/themes/9063/le-marche-du-jeu-video/#topicOverview>

Collectif. « Industrie du jeu - Analyse de la taille et des parts – Tendances et prévisions de croissance (2023 - 2028) ». 2018. Mordor Intelligence.

<https://www.mordorintelligence.com/fr/industry-reports/global-gaming-market>

Collectif. « AAA (jeu vidéo) ». 2016. Wikipédia.

<https://fr.wikipedia.org/wiki/AAA_(jeu_vid%C3%A9o)>

Collectif. « Unity ou Unreal engine, quel moteur de jeux choisir ? ». 20XX. Formation facile.

https://www.formation-facile.fr/blog/unity-ou-unreal-engine-quel-moteur-de-jeux-choisir

Pauline Callies. « Les meilleurs outils pour créer votre maquette d’application mobile ». 19 mai 2022. Aventique.

<https://aventique.paris/maquette-application-mobile/#:~:text=Figma%20est%20un%20outil%20de,de%20ses%20fonctionnalit%C3%A9s%20est%20payant>.

Collectif. « 6 alternatives à Google Drive ». 2021. Blogdumoderateur.

<https://www.blogdumoderateur.com/tools/alternatives/google-drive/>

Eloïse Salson. « Suivez vos modifications à la trace avec les 8 meilleurs logiciels de versionning ». 19 janvier 2022. Appvizer.

<https://www.appvizer.fr/magazine/services-informatiques/gestion-versions/outils-versionning>

Alicia Raeburn. « Logiciels et outils de gestion de projet : les meilleurs choix pour 2023 ». 9 octobre 2023. Asana.

<https://asana.com/fr/resources/best-project-management-software>

Collectif. « Top 10 des frameworks frontend les plus populaires à utiliser en 2024 ». 26 septembre 2022. AppMaster.

<https://appmaster.io/fr/blog/frameworks-frontaux-populaires>

Introduction à Express/Node - Apprendre le développement Web | MDN. (s. d.). MDN Web Docs.

[Introduction à Express/Node - Apprendre le développement web | MDN (mozilla.org)](https://developer.mozilla.org/fr/docs/Learn/Server-side/Express_Nodejs/Introduction#introduction_%C3%A0_express)

# Annexe 4 : Images du digramme de Gantt

**A screenshot of a computer

Description automatically generated**

Figure : Digramme de Gantt, première partie

A screenshot of a computer

Description automatically generated

Figure : Diagramme de Gantt deuxième partie

# Annexe 5 : Table des illustrations

# Annexe 6 : Cahier des charges

# Annexe 7 : Proposition de projet SAE S5

# Annexe 8 : Poster