Documentation Technique Détaillée de

Glimps Gone

L'extravagante galerie d'art des oeuvres invisibles

==========

Introduction

[Slide1] Je vous présente "Glimps Gone" ou l'extravagante galerie d'art des oeuvres invisibles ! La galerie a été conçu pour mon projet ECF 2024, c' est aussi le résultat de ma passion profonde pour l'art ainsi que de mon désir de fusionner ces intérêts avec ma formation.

En tant que créatrice de l'application, j'ai visé à construire un espace numérique proposant une expérience utilisateur à la fois immersive et interactive.

Conception :	Photoshop, Dafont, Coolors
Front-end :	HTML, CSS, JavaScript
Sécurité contre les injections SQL :	Requêtes paramétrées
Back-end :	Node.js et Express.js
Envoi d'e-mails :	SMTPJS
Base de données :	SQLite 3
Vérification du formulaire :	Parsley.js
Partage :	Intégration de l'API Facebook

Conception et Développement

Phase de conception

[Slide2] Outils de Design : Suite à des recherches sur des sites de galerie d'art afin d'inspirer mon design, j'ai commencé par dessiner ceux ci sur Photoshop. Validant mon logo.

En ce qui concerne ma palette couleur, je les validé sur Coolors.com.

Mes polices d'ecriture quand à elle ont été validés, et telechargé dans mon dossier font, sur Dafont.com.

Pour le maquettage de l'interface et l'expérience utilisateur, j'ai utilisé Figma**[Slide3]** , ce qui a permis de visualiser et de partager facilement le design de l'application ainsi que la reflexion sur l'experience utilisateur. J'ai donc pu commener mes user stories suite à la finalité de ma maquette.

Planification et Gestion de Projet : Trello**[Slide4]** qui a servi à organiser et suivre l'avancement du projet, facilitant ainsi la gestion des tâches découlant de mes user stories et en structurant les différentes phases de développement, de la conception à la réalisation.

Cette approche m'a permis de visualiser clairement les exigences du projet et de les organiser de manière logique, tout en offrant une vue d'ensemble du processus. De plus, la numérotation des tâches en fonction des user stories et des commits GitHub facilite la traçabilité pour la relecture d'une tierce personne.

Presentation du projet : J'utilise actuellement Prezi pour vous presenter mon projet de facon interactive.

Développement Front-End

Technologies Utilisées : **[Slide5]** Pour le front-end, j'ai choisi HTML, CSS, et JavaScript pour créer une interface simple et fluide de navigation entre les différentes œuvres d'art. J'ai choisi d'utiliser le framework BootStrap pour rendre mes pages responsive.

Evidement j'ai egalement du utiliser des medias Queries pour rendre le site completement responsive.

Développement Back-End

Serveur et API : **[Slide6]** J'ai utilisé Node.js et Express.js pour interargir simplement avec le serveur et concevoir ma propre API.

Cela m'a permis d'ajuster précisément les fonctionnalités de la galerie, d'améliorer l'efficacité et surtout d'enrichir mes compétences techniques.

J'ai decidé de découpler mon code dans deux fichiers différents, server.js pour le serveur l'API et la logique metier et repository.js pour la logique d'acces aux données, aide à organiser mon projet de façon claire, rend le projet plus facile à modifier, en distinguant bien les fonctions du serveur de celles liées aux données.

Si je decide de changer de systeme de base de données, mySql par exemple, je n'aurai que pas a toucher au server.js !

[Slide7] J'ai opté pour l'utilisation de la synthaxe async/await dans mon code pour simplifier la gestion des opérations asynchrones, rendant ainsi mon code plus lisible et facile à maintenir.

""Await" met en pause l'exécution de la fonction async jusqu'à ce que la promesse soit résolue, puis retourne le résultat de cette promesse. Cela permet de gérer de manière synchrone des opérations asynchrones dans du code JavaScript."

En ce qui concerne mes routes API :

Route GET pour récupérer toutes les œuvres d'art. app.get("/api/oeuvres")

Route POST pour ajouter une nouvelle œuvre. app.post("/api/oeuvres")

Route POST pour augmenter le compteur de "j'aime" d'une œuvre spécifique. app.post("/api/oeuvres/:id/jaime")

Route POST similaire pour les "j'aime pas". app.post("/api/oeuvres/:id/jaimepas")

Interaction avec la Base de Données

Gestion des Données : **[Slide8]** J'ai choisi de coder entièrement ma base de données pour mon projet en utilisant SQLite plutôt que d'utiliser à MySQL Workbench pour plusieurs raisons avantageuses.

Tout d'abord, en codant directement ma base de données, j'ai acquis une compréhension approfondie du fonctionnement et des interactions relationnelles entre les différentes tables, ce qui a renforcé ma connaissance sur la gestion des données.

Deuxiemement, SQLite offre une simplicité d'intégration et de déploiement exceptionnelle, étant donné qu'elle ne nécessite pas de serveur de base de données séparé, ce qui est idéal pour un projet comme celui ci, où la légèreté et l'efficacité sont primordiales.

Enfin, cette décision a encouragé une meilleure maîtrise des compétences en matière de bases de données, en me confrontant directement aux défis de la programmation et de l'optimisation des requêtes.

Sécurité

Prévention des Injections SQL : **[Slide9]** J'ai systématiquement utilisé des requêtes préparées pour toutes les interactions avec la base de données, bloquant ainsi les tentatives d'injection SQL et garantissant la sécurité des données. J'ai choisi cette approche pour renforcer la sécurité de mon application.

Validation des Formulaires : **[Slide10,11,12]** J'ai intégré la librairie Parsley.js pour une validation côté

front, prévenant les soumissions erronées avec un message d'alerte pour le confort utilisateur.

Présentation des pages de la galerie

Page d'accueil : **[Slide13]** Avec une présentation et explication du concept de la galerie d'art.

Page galerie: **[Slide14,15,16]** C'est sur cette page que nous retrouvons toutes les oeuvres d'art publié par les artistes.

Visualisation des œuvres: Les utilisateurs peuvent voir les œuvres d'art avec son titre, son artiste, l'année de sa création et sa description.

Navigation: Les utilisateurs peuvent naviguer entre les différentes œuvres d'art grâce aux boutons "Suivant" et "Précédent".

Interactions: Les utilisateurs peuvent exprimer leur appréciation des œuvres via des boutons "J'aime" et "Je n'aime pas".

Les utilisateurs peuvent partager une oeuvre sur Facebook.

Page de fin de galerie ansi que la page down au cas ou !

Page ajouter:**[Slide17,18]** Nous pouvons trouver un formulaire pour soumettre son oeuvre et l 'a rendre visiognable par les utilisateurs. L'oeuvre ajouté est directement envoyé sur le serveur, stocké en base de donnée et la page de l'oeuvre se creer dans la galerie pour l'aficher instantanément.

Page FAQ: **[Slide19]** page repondant aux questions les plus frequement posées.

Pages infos & contact:**[Slide20/21]** proposant une presentation succinte de la creatrice de la galerie ainsi qu'un formulaire pour une prise de contact.

Contact : les artistes peuvent directement ecrire un message a la galerie d'art envoyé par email grace a un formulaire qui utilise la fonctionnalité SMTP.JS et parsley.js.

Les amélioration à mettre en place par la suite

[Slide22]

En passant de SQLite à MySQL, je pourrais accueillir plus de visiteurs sur ma galerie d'art en ligne et la faire grandir sans souci.

Retravailler les fonctionnalités des compteurs "J'aime" et "Je n'aime pas" pour éviter qu'ils ne soient incrémentés indéfiniment, et pour s'assurer que cliquer sur "J'aime" désactive automatiquement "Je n'aime pas", et vice-versa.

Moderation des oeuvres ajouter par les artistes avec une page administrateur (pageAdmin.html dans le dossier source) avant leurs mises en ligne.

L'ajout d'une fonctionnalité permettant aux artistes de s'inscrire et de disposer de leur propre page dédiée.

Création et ce qui en decoule d'un "S'inscrire" et "Se connecter".

Recherche d'oeuvre par artiste/par nom/année

Soumettre des questions pour l'FAQ...

Conclusion

[Slide23]

Ce projet représente à l'instant T et en 10 jours, ma maitrise en développement web et en design graphique. Tout est resumé sur mon README.md avec les liens cliquable et la techique pour cloner mon projet sur gitHub et l'installer sur son ordinateur. En esperant que cela vous a plus !

Installation

git clone https://github.com/Baldaxx/glimpsGone

npm install

node server/server.js

Vous pouvez accéder à l'interface utilisateur en ouvrant http://localhost:3000 dans votre navigateur.