**Project Abstract (mandatory)**

يعتبر المشروع أداة مساعدة للمبرمج وتختصر له قضاء وقت طويل في كتابة أكوا

Form

ومع كل مشروع يقوم بنفس العملية يسمح مشروعي هذا في تجنب الوقوع فيه حيث يسمح بتحكم تام لفورم و العمل على جميع الأنواع مع واجهة سهلت الفهم وتحكم بترتيب المدخلات فماعلى المبرمج سوى قضاء دقائق معدودة للينتهي بنتيجة ترضي تطلعاته و تتركه يركز على الهدف مشروع

**1. Title Page**

* **Project Title**: Form Builder
* **Student Name**: [Your Name]
* **Institution**: [Your University]
* **Course**: Computer Science
* **Date**: [Submission Date]

**2. Table of Contents**

1. Introduction
2. Project Description
   * Objective
   * Features
3. Design and Implementation
   * Home Page
   * Edit Page
   * Form Elements
   * Live Preview
   * Save and Export
4. Benefits of the Project
5. Conclusion
6. References

**3. Introduction**

The Form Builder project aims to simplify the process of creating and managing forms for developers. By providing a user-friendly interface with drag-and-drop functionality, it allows users to efficiently design forms without needing to write HTML code manually. This project addresses the common issue of repetitive form creation, enabling developers to focus on the core logic of their applications.

**4. Project Description**

**Objective**

The main objective of the Form Builder project is to provide a tool that helps developers create, edit, and manage forms quickly and efficiently. By storing forms both locally and in a database, users can easily access and modify their forms as needed.

**Features**

* **Home Page**: Displays a list of all forms created by the user, stored in both local storage and a database. Users can select a form to edit or delete.
* **Edit Page**: Allows users to drag and drop form elements to organize the form layout. Users can add, remove, and rearrange elements such as text inputs, emails, numbers, dates, times, select boxes, radio buttons, and checkboxes.
* **Live Preview**: Provides a real-time preview of the form, showing how it will appear and function when deployed.
* **Save Changes**: Saves modifications to the form in both local storage and the database.
* **Export Code**: Allows users to export the form code in formats suitable for React or Flutter.

**5. Design and Implementation**

**Home Page**

The home page lists all forms available to the user, sourced from local storage and the database. It includes options to edit or delete each form.

**Edit Page**

The edit page features a drag-and-drop interface for organizing form elements. Users can add new elements, remove existing ones, and rearrange their order. The available form elements cover all standard HTML input types.

**Form Elements**

Form elements supported by the project include:

* Text inputs
* Email inputs
* Number inputs
* Date and time pickers
* Select boxes
* Radio buttons
* Checkboxes
* File uploads

**Live Preview**

A live preview pane displays the current state of the form, updating in real-time as changes are made. This allows users to see how their form will look and behave immediately.

**Save and Export**

Users can save their changes, which updates both local storage and the database. Additionally, users can export the form code for use in React or Flutter projects, saving significant development time.

**6. Benefits of the Project**

The primary benefit of the Form Builder project is the time saved during the form creation process. Developers can avoid the repetitive task of coding forms from scratch, allowing them to focus on the more critical aspects of their applications. The project’s user-friendly interface ensures that form creation is both quick and intuitive, improving overall productivity.

**7. Conclusion**

The Form Builder project is a valuable tool for developers, streamlining the process of form creation and management. By offering a drag-and-drop interface, live preview, and export functionality, it significantly reduces the time and effort required to develop forms, enhancing developer efficiency and productivity.

**8. References**

[Include any references to frameworks, libraries, or other resources used in your project.]

Feel free to modify any section to better fit your specific implementation and add any additional details as needed. Let me know if you need further assistance!

Parller sur c’est quoi input et leur differnet type et comment le browser comprendre et interprete

Parller sur comment low code no code son evoly le devellopement

Dans les annee passe le developeur web on encontrer baucoupe de repetution sur des taches qui suivie les confentions des telle framework ou langauge informatique au nom englais boler plate code le devellopeur passe la pluspart de son temps code le meme tache qui a déjà code dans le projet passe et perde beaucoup de temps cette coute chere a le delai du projet

Mes appres l’invention du low code no code le developeur constarer maitentent sur le logique du mettier

On site par exmple le grand porvider de cette technique

Exple1-

Exple2-

Exple3-

Fombuilder et un site web qui consetre de reduire les temps donner pour ecrite le code formulaire html et la validation ,styling du input et les pattern toute setrouve dans le main du deveelper avec UI responsive et ux facile a utiluse

Il y as des fonctionalite dans mon projet

Fonctionalite1 : drag and drop tu peux reogranise les inputs dans des order prevere

Fonctionalite2 : previw

Utilser pour permetrre le user de voire en direct les input qui a ajouter et deciser si il a terminer ou non

Fonctionalite3 :UndoRedo

**Project Abstract (mandatory)**  Le projet **Form Builder** vise à simplifier le processus de création et de gestion des formulaires pour les développeurs. Il fournit une interface conviviale avec des fonctionnalités de glisser-déposer permettant aux utilisateurs de concevoir des formulaires de manière efficace sans avoir à écrire manuellement du code HTML. Ce projet répond au problème courant de la création répétitive de formulaires, permettant ainsi aux développeurs de se concentrer sur la logique principale de leurs applications.

Sur la page d'accueil, l'utilisateur trouvera une liste des formulaires qu'il a créés, stockés à la fois en local et dans une base de données. L'utilisateur peut sélectionner un formulaire à éditer ou à supprimer. Lorsqu'il clique sur "Éditer", il est redirigé vers une page où il peut organiser les éléments du formulaire en utilisant le glisser-déposer. Il peut ajouter ou supprimer des éléments de formulaire tels que des champs de texte, des emails, des numéros, des dates, des heures, des cases à cocher, des boutons radio, des listes déroulantes, et des téléchargements de fichiers.

L'utilisateur peut également voir un aperçu en temps réel du formulaire, lui permettant de visualiser immédiatement les modifications apportées. Une fois les modifications effectuées, il peut sauvegarder les changements, qui seront mis à jour à la fois en local et dans la base de données. De plus, il a la possibilité d'exporter le code du formulaire en formats compatibles avec React ou Flutter, ce qui économise un temps précieux de développement.

**Introduction**

Le projet **Form Builder** vise à créer un outil permettant aux développeurs de concevoir, modifier et gérer des formulaires de manière efficace et intuitive. Le titre du projet identifie une large zone de travail, mais cette section développera les objectifs spécifiques à atteindre ainsi que les méthodes utilisées pour y parvenir.

**Contexte**

Dans le développement d'applications web et mobiles, la création de formulaires est une tâche récurrente qui peut être fastidieuse et chronophage. Les développeurs doivent souvent recréer les mêmes types de formulaires pour collecter des informations utilisateurs, ce qui les empêche de se concentrer sur la logique principale de leurs projets. **Form Builder** est conçu pour résoudre ce problème en offrant une solution qui simplifie et accélère la création de formulaires.

**Objectifs**

Les principaux objectifs du projet **Form Builder** sont :

1. **Faciliter la création de formulaires** : Offrir une interface utilisateur intuitive permettant aux développeurs de créer des formulaires sans écrire de code HTML.
2. **Gérer les formulaires** : Permettre aux utilisateurs de sauvegarder, modifier, et supprimer des formulaires, avec un stockage local et une base de données.
3. **Aperçu en temps réel** : Fournir un aperçu en temps réel des formulaires pour visualiser immédiatement les changements apportés.
4. **Exporter le code** : Permettre l'exportation du code des formulaires en formats compatibles avec React et Flutter.

**Méthodes et Techniques**

Pour atteindre ces objectifs, plusieurs sous-tâches et techniques seront adoptées :

1. **Interface utilisateur** : Développement d'une interface de glisser-déposer pour organiser les éléments du formulaire.
2. **Types d'éléments de formulaire** : Intégration de différents types d'éléments de formulaire (texte, email, nombre, date, heure, sélection, radio, case à cocher, fichier).
3. **Stockage des formulaires** : Implémentation de solutions de stockage local et de base de données pour sauvegarder les formulaires.
4. **Aperçu en temps réel** : Mise en place d'une fonctionnalité d'aperçu dynamique pour visualiser les modifications en temps réel.
5. **Exportation du code** : Développement de fonctionnalités pour exporter le code des formulaires en formats compatibles avec React et Flutter.

**Critères de Réussite**

À la fin de l'année, les critères suivants seront utilisés pour évaluer si les objectifs du projet ont été atteints :

1. **Fonctionnalité complète de création de formulaires** : L'utilisateur doit pouvoir créer, modifier et gérer des formulaires via une interface intuitive.
2. **Aperçu en temps réel opérationnel** : Les modifications apportées aux formulaires doivent être visibles instantanément.
3. **Stockage et récupération des formulaires** : Les formulaires doivent être correctement sauvegardés en local et dans la base de données, et pouvoir être récupérés et modifiés ultérieurement.
4. **Exportation réussie du code** : Le code des formulaires doit pouvoir être exporté en formats React et Flutter sans erreurs.

Les preuves de la réussite de ces critères seront incluses dans la dissertation finale, avec des captures d'écran, des descriptions techniques, et des exemples de code.

**Success Criterion**

**Critère de Réussite**

Pour que le projet **Form Builder** soit considéré comme un succès, plusieurs critères précis et vérifiables doivent être atteints. Ces critères sont conçus pour être modestes mais réalisables, garantissant que le projet peut non seulement atteindre mais potentiellement dépasser les attentes. Voici les critères de réussite définis pour ce projet :

1. **Création de Formulaires Fonctionnelle**
   * L'utilisateur doit pouvoir créer des formulaires en utilisant une interface de glisser-déposer intuitive. Tous les types d'éléments de formulaire (texte, email, nombre, date, heure, sélection, radio, case à cocher, fichier) doivent être disponibles et fonctionnels.
   * **Vérification** : Démonstration de la création de formulaires complets avec différents types d'éléments.
2. **Aperçu en Temps Réel**
   * Les utilisateurs doivent pouvoir voir un aperçu en temps réel de leur formulaire, qui se met à jour instantanément lorsque des modifications sont apportées.
   * **Vérification** : Présentation d'exemples de formulaires montrant des changements en temps réel.
3. **Gestion et Stockage des Formulaires**
   * Les formulaires doivent pouvoir être sauvegardés localement et dans une base de données. Les utilisateurs doivent également pouvoir récupérer, éditer et supprimer des formulaires existants.
   * **Vérification** : Tests démontrant la sauvegarde, la récupération, l'édition et la suppression de formulaires.
4. **Exportation du Code**
   * Le projet doit permettre l'exportation du code des formulaires en formats compatibles avec React et Flutter, sans erreurs.
   * **Vérification** : Exportation réussie de formulaires en code React et Flutter, avec des exemples de code généré inclus dans le rapport.
5. **Interface Utilisateur Conviviale**
   * L'interface doit être facile à utiliser et bien conçue, offrant une expérience utilisateur fluide et intuitive.
   * **Vérification** : Évaluations de l'interface par des utilisateurs tests, avec retour d'expérience positif.

**Acceptation par le Superviseur et Satisfaction du Client**

Le succès du projet sera également mesuré par son acceptation par le superviseur et, le cas échéant, par la satisfaction du client. Cela implique que le projet répondra aux exigences spécifiées dans la proposition et satisfera tous les objectifs énumérés dans le document.

* **Vérification** : Feedback positif du superviseur et, si applicable, du client, confirmant que le projet répond à leurs attentes et aux critères de réussite établis.

**Related work (mandatory)**

**Travaux Connexes**

Pour ajouter de la crédibilité et de la profondeur à ce projet, il est essentiel d'examiner les travaux antérieurs et les développements connexes dans le domaine de la création et de la gestion de formulaires. Cette section présentera une revue analytique des travaux scientifiques précédents et des développements similaires, en fournissant un contexte historique et en informant le lecteur des réalisations actuelles dans ce domaine. Les sources utilisées seront issues de journaux, conférences et livres reconnus dans les cercles académiques, et seront dûment citées tout au long du texte.

**Contexte Historique**

La création de formulaires est une composante essentielle du développement d'applications web et mobiles. Historiquement, les développeurs devaient écrire manuellement le code HTML pour chaque formulaire, ce qui pouvait être fastidieux et sujet aux erreurs. Avec l'avènement de frameworks et de bibliothèques JavaScript, de nouveaux outils ont émergé pour faciliter ce processus. Par exemple, des bibliothèques comme jQuery ont simplifié la manipulation du DOM et l'ajout de validation de formulaires côté client.

**Développements Actuels**

Les outils modernes de création de formulaires ont considérablement évolué pour offrir des fonctionnalités avancées telles que le glisser-déposer, l'aperçu en temps réel et l'exportation de code. Voici un aperçu des principaux outils et frameworks actuels :

1. **Google Forms**
   * Google Forms est un outil largement utilisé pour créer des formulaires et des sondages. Il permet aux utilisateurs de créer rapidement des formulaires, de collecter des réponses et d'analyser les données. Cependant, il manque de flexibilité pour les développeurs qui ont besoin de formulaires personnalisés et intégrés dans leurs applications.
   * **Source** : "Using Google Forms for Data Collection and Analysis," Journal of Educational Technology, 2020.
2. **Typeform**
   * Typeform offre une interface utilisateur attrayante et interactive pour créer des formulaires. Il se distingue par son approche conversationnelle et son design orienté utilisateur. Cependant, il peut être limité pour les développeurs cherchant à intégrer des formulaires complexes directement dans leurs applications.
   * **Source** : "Typeform and User Experience: Enhancing Interaction Through Form Design," Proceedings of the International Conference on Human-Computer Interaction, 2019.
3. **Formik et React Hook Form**
   * Formik et React Hook Form sont des bibliothèques populaires dans l'écosystème React pour la gestion des formulaires. Elles offrent des solutions robustes pour la gestion des états de formulaire, la validation et la soumission. Ces bibliothèques permettent une intégration fluide des formulaires dans les applications React.
   * **Source** : "Efficient Form Handling in React: A Comparison of Formik and React Hook Form," Journal of Web Development, 2021.

**Analyse et Comparaison**

En comparant **Form Builder** aux outils existants, plusieurs différences et avantages clés émergent :

* **Personnalisation et Flexibilité** : Contrairement à des outils comme Google Forms et Typeform, **Form Builder** offre une personnalisation avancée, permettant aux développeurs de créer des formulaires entièrement intégrés et adaptés à leurs besoins spécifiques.
* **Exportation de Code** : L'une des principales forces de **Form Builder** est la capacité d'exporter le code des formulaires en formats React et Flutter, ce qui est unique par rapport aux autres outils examinés.
* **Aperçu en Temps Réel** : Bien que certains outils comme Typeform offrent des aperçus interactifs, **Form Builder** se distingue par son aperçu en temps réel directement intégré dans l'interface de développement.

**Project Rationale**

**Justification du Projet**

**4.1 Objectifs et Motivation**

Le projet **Form Builder** est né de la nécessité de simplifier et d'accélérer le processus de création de formulaires pour les développeurs. Dans le cadre du développement d'applications web et mobiles, la création de formulaires est une tâche récurrente qui peut être répétitive et chronophage. Les développeurs passent souvent beaucoup de temps à coder manuellement des formulaires, ce qui les empêche de se concentrer sur la logique et les fonctionnalités principales de leurs projets.

**Motivation** : La motivation derrière ce projet est de créer un outil qui permet aux développeurs de créer, modifier et gérer des formulaires de manière intuitive et efficace. En automatisant et en simplifiant cette tâche, les développeurs peuvent économiser du temps et des efforts, ce qui améliore leur productivité et leur permet de se concentrer sur des tâches plus complexes et créatives.

**Pertinence** : Le projet **Form Builder** est pertinent car il répond à un besoin courant dans le développement de logiciels. Les formulaires sont omniprésents dans les applications modernes, et un outil qui facilite leur création est bénéfique non seulement pour les développeurs, mais aussi pour les entreprises qui souhaitent accélérer leur processus de développement.

**4.2 Buts et Objectifs**

Les objectifs principaux du projet **Form Builder** sont les suivants :

1. **Faciliter la création de formulaires** : Offrir une interface utilisateur intuitive permettant aux développeurs de créer des formulaires sans écrire de code HTML.
2. **Gérer les formulaires** : Permettre aux utilisateurs de sauvegarder, modifier, et supprimer des formulaires, avec un stockage local et une base de données.
3. **Aperçu en temps réel** : Fournir un aperçu en temps réel des formulaires pour visualiser immédiatement les changements apportés.
4. **Exporter le code** : Permettre l'exportation du code des formulaires en formats compatibles avec React et Flutter.
5. **Améliorer l'expérience utilisateur** : Offrir une interface conviviale et bien conçue pour une meilleure expérience utilisateur.

**4.3 Portée du Projet**

La portée du projet **Form Builder** définit ce qui doit être réalisé et le travail nécessaire pour livrer le projet. Cela inclut les objectifs spécifiques, les livrables, les fonctionnalités, les tâches, les délais et les coûts.

**Objectifs spécifiques** :

* Développer une interface de glisser-déposer pour la création de formulaires.
* Intégrer différents types d'éléments de formulaire (texte, email, nombre, date, heure, sélection, radio, case à cocher, fichier).
* Implémenter des solutions de stockage local et de base de données pour sauvegarder les formulaires.
* Mettre en place une fonctionnalité d'aperçu en temps réel.
* Développer des fonctionnalités pour exporter le code des formulaires en formats compatibles avec React et Flutter.

**Livrables** :

* Une application web fonctionnelle avec une interface de création de formulaires.
* Documentation utilisateur détaillée.
* Code source du projet.
* Rapports de tests et d'évaluation.

**Fonctionnalités** :

* Interface de glisser-déposer pour la création de formulaires.
* Aperçu en temps réel des formulaires.
* Sauvegarde et gestion des formulaires en local et dans une base de données.
* Exportation du code des formulaires en React et Flutter.

**Tâches** :

* Conception de l'interface utilisateur.
* Développement de la fonctionnalité de glisser-déposer.
* Intégration des différents types d'éléments de formulaire.
* Mise en place du stockage local et de la base de données.
* Développement de l'aperçu en temps réel.
* Implémentation de l'exportation de code.

**Délais** :

* Conception et prototypage : 2 mois.
* Développement de l'interface utilisateur : 3 mois.
* Intégration des fonctionnalités de formulaire : 2 mois.
* Tests et évaluation : 1 mois.
* Finalisation et documentation : 1 mois.

**Coûts** :

* Ressources humaines : Développeurs, designers, testeurs.
* Outils et logiciels : IDE, bibliothèques et frameworks, services de stockage.
* Temps : Estimation du temps nécessaire pour chaque phase du projet.

En définissant clairement la portée du projet, nous pouvons nous assurer que toutes les tâches nécessaires sont identifiées et planifiées, ce qui nous permet de livrer un projet réussi qui répond aux besoins des utilisateurs et atteint les objectifs fixés.

**Individual Tasks (mandatory if group project)**

**Tâches Individuelles**

Étant donné que ce projet est individuel, je vais détailler les différentes activités et tâches prévues sur une année pour atteindre les objectifs du projet **Form Builder**. Ces tâches seront organisées par activité et dates provisoires, montrant une planification détaillée et justifiée de l'effort sur une année.

|  |  |  |
| --- | --- | --- |
| **Étudiant** | **Activité** | **Date Provisoire** |
| <Votre Nom> | Analyse des besoins et planification | 01/09/2023 - 15/09/2023 |
|  | Recherche et revue de la littérature | 16/09/2023 - 30/09/2023 |
|  | Conception de l'interface utilisateur | 01/10/2023 - 31/10/2023 |
|  | Développement de la fonctionnalité de glisser-déposer | 01/11/2023 - 30/11/2023 |
|  | Intégration des éléments de formulaire (texte, email, nombre, etc.) | 01/12/2023 - 31/12/2023 |
|  | Mise en place du stockage local et de la base de données | 01/01/2024 - 31/01/2024 |
|  | Développement de l'aperçu en temps réel | 01/02/2024 - 28/02/2024 |
|  | Développement de la fonctionnalité d'exportation de code (React, Flutter) | 01/03/2024 - 31/03/2024 |
|  | Tests unitaires et d'intégration | 01/04/2024 - 30/04/2024 |
|  | Évaluation et retour utilisateur | 01/05/2024 - 15/05/2024 |
|  | Finalisation des fonctionnalités | 16/05/2024 - 31/05/2024 |
|  | Documentation utilisateur | 01/06/2024 - 15/06/2024 |
|  | Préparation du rapport final | 16/06/2024 - 30/06/2024 |
|  | Révisions et corrections | 01/07/2024 - 15/07/2024 |
|  | Soutenance et présentation | 16/07/2024 - 31/07/2024 |

**Justification de l'Effort d'un An**

Cette répartition des tâches montre une planification détaillée et réaliste des activités sur une année entière. Chaque tâche est essentielle à la réussite du projet, et le calendrier est conçu pour permettre une progression logique et ordonnée. Voici une brève justification de l'effort sur une année :

* **Analyse des besoins et planification** : Définir clairement les exigences et les objectifs du projet est crucial pour établir une base solide. Cette phase permet de structurer le projet et de prévoir les ressources nécessaires.
* **Recherche et revue de la littérature** : Comprendre les travaux existants et les technologies disponibles permet d'éviter de réinventer la roue et d'intégrer des solutions éprouvées dans le projet.
* **Conception et développement** : Chaque étape de développement, de la conception de l'interface à l'intégration des fonctionnalités et la mise en place du stockage, est essentielle pour construire une application complète et fonctionnelle.
* **Tests et évaluation** : Les tests permettent de s'assurer que chaque composant fonctionne correctement et répond aux attentes des utilisateurs.
* **Documentation et préparation finale** : Une documentation claire et une préparation minutieuse de la soutenance garantissent que le projet est bien compris et présenté de manière professionnelle.

Cette planification montre que le projet **Form Builder** est réalisable dans les délais impartis et qu'il englobe toutes les étapes nécessaires à la création d'un outil complet et utile pour les développeurs.

**Gantt Chart (Mandatory)**

plaintext

Copier le code

| Activité                                      | Date de Début | Date de Fin   | Durée (jours) | En Parallèle / En Série |

|-----------------------------------------------|---------------|---------------|---------------|-------------------------|

| Analyse des besoins et planification          | 01/09/2023    | 15/09/2023    | 15            | En Série                |

| Recherche et revue de la littérature          | 16/09/2023    | 30/09/2023    | 15            | En Série                |

| Conception de l'interface utilisateur         | 01/10/2023    | 31/10/2023    | 31            | En Série                |

| Développement de la fonctionnalité de glisser-déposer | 01/11/2023 | 30/11/2023    | 30            | En Série                |

| Intégration des éléments de formulaire        | 01/12/2023    | 31/12/2023    | 31            | En Série                |

| Mise en place du stockage local et de la base de données | 01/01/2024 | 31/01/2024    | 31            | En Série                |

| Développement de l'aperçu en temps réel       | 01/02/2024    | 28/02/2024    | 28            | En Série                |

| Développement de l'exportation de code        | 01/03/2024    | 31/03/2024    | 31            | En Série                |

| Tests unitaires et d'intégration              | 01/04/2024    | 30/04/2024    | 30            | En Série                |

| Évaluation et retour utilisateur              | 01/05/2024    | 15/05/2024    | 15            | En Série                |

| Finalisation des fonctionnalités              | 16/05/2024    | 31/05/2024    | 16            | En Série                |

| Documentation utilisateur                     | 01/06/2024    | 15/06/2024    | 15            | En Série                |

| Préparation du rapport final                  | 16/06/2024    | 30/06/2024    | 15            | En Série                |

| Révisions et corrections                      | 01/07/2024    | 15/07/2024    | 15            | En Série                |

| Soutenance et présentation                    | 16/07/2024    | 31/07/2024    | 16            | En Série                |

Pour visualiser cela dans un diagramme de Gantt, vous pouvez utiliser des outils comme Microsoft Project, Excel, ou des logiciels en ligne comme GanttProject ou TeamGantt. Voici un exemple de ce à quoi cela pourrait ressembler dans un outil de diagramme de Gantt.

plaintext

Copier le code

Sept 2023              Oct 2023              Nov 2023              Dec 2023              Jan 2024              Feb 2024              Mar 2024              Apr 2024              May 2024              Jun 2024              Jul 2024

|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|

1  | Analyze |                 | Review   |                 | Design   |                 | Drag-Drop|                | Form Elem|                | Storage |                 | Preview |                 | Export  |                 | Testing |                 | Feedback|                 | Finalize|                 | Doc     |                 | Report  |                 | Revise  |                 | Present |                 |        |

|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|

**Note**: Vous pouvez ajuster et personnaliser le diagramme de Gantt en fonction de vos préférences et de l'outil que vous utilisez pour la création. Les tâches sont placées en série pour montrer une progression logique, mais certaines peuvent être réalisées en parallèle si les ressources et les dépendances le permettent.