**Progetto Portfoglio personale [I.T.A]**

**ANALISI**

**Metodologia:** WatherFall (a cascata).

**Obbiettivo:** Progettare un sito portfolio personale utilizzando **Java & Spring Boot** per il backend e **Angular** per il frontend e MySql Workbench come DataBase, ecco una guida passo passo per iniziare:

**1.** **Progettazione del database:** Ho iniziato progettando il database per il sito portfolio. Ho identificato le entità principali, come le informazioni sulle esperienze accademiche e lavorative, i progetti, una sezione About nella quale ho messo tutte le informazioni su di me e così via. Creare le tabelle del database corrispondenti a queste entità e definire le relazioni tra di loro.

2. **Configurazione del progetto Spring Boot:** Ho creato un nuovo progetto Spring Boot utilizzando spring initializr e Intelij come IDE. Mi sono assicurato di aver incluso le dipendenze necessarie per l'accesso al database, ad esempio Spring Data JPA e un driver per il database che userò.

3. **Definizione delle entità JPA:** Ho creato le classi Java che rappresentano le entità del database. Ho utilizzato le annotazioni di Spring Data JPA per mappare queste classi alle tabelle del database, assicurandomi di includere le relazioni tra le entità, come ad esempio @OneToMany e @ManyToOne o @OneToOne.

4. **Creazione dei repository:** Ho creato delle interfacce di repository utilizzando Spring Data JPA. Questi repository forniranno i metodi per eseguire operazioni CRUD (Create, Read, Update, Delete) sul database.

5. **Creazione dei servizi:** Ho creato i servizi Spring per gestire la mia logica di business. I servizi utilizzeranno i repository per accedere e manipolare i dati del database.

6. **Creazione dei controller REST:** Ho creato i controller Spring per esporre le API REST che saranno utilizzate dal frontend. Questi controller riceveranno le richieste HTTP dal client (frontend), chiameranno i servizi appropriati e restituiranno le risposte al frontend come JSON.

7. **Configurazione della sicurezza:** Per proteggere alcune parti del sito portfolio, potevo configurare la sicurezza utilizzando Spring Security, ma ho optato per la creazione di una classe @Configuration nella quale impedisco alcune azioni.

8. **Creazione del frontend Angular:** Ho creato un nuovo progetto Angular utilizzando Angular CLI. Progettare e creare le pagine e i componenti del sito portfolio, come la pagina principale, la pagina dei progetti, la pagina delle competenze, etc... Poi utilizzo il routing di Angular per navigare tra le diverse pagine.

9. **Chiamata alle API REST:** Utilizzo il modulo HttpClient di Angular per effettuare richieste alle API REST del backend Spring Boot. Ad esempio, chiamo l'API per ottenere i dati salvati in un Database.

10. **Presentazione dei dati:** Ho utilizzato le direttive di Angular per presentare i dati ottenuti dal backend nel frontend. Ad esempio, posso utilizzare \*ngFor per iterare su una lista di progetti e mostrare le informazioni corrispondenti.

11. **Stile e layout:** Utilizzare CSS e/o un framework di stile come Bootstrap per migliorare l'aspetto del sito portfolio.

**Analisi delle Entità**

**Projet de portfolio personnel [F.R]**

**ANALYSES**

**Méthodologie de conception:** WaterFall (cascade).

**Objectif:** Concevoir un site de portfolio personnel en utilisant Java & Spring Boot pour le backend et Angular pour le frontend et MySql Workbench comme DataBase, voici un guide étape par étape pour commencer:

1. **Conception de la base de données:** commencez par concevoir la base de données du site du portfolio. Identifiez les entités clés, telles que les informations sur l'expérience de travail, les projets, les compétences, l'éducation, les contacts, etc. Créez les tables de la base de données correspondant à ces entités et définissez les relations entre elles.

2. **Configuration du projet Spring Boot:** créez un nouveau projet Spring Boot en utilisant Intelij comme IDE. Assurez-vous d'inclure les dépendances nécessaires pour accéder à la base de données, telles que Spring Data JPA et un pilote pour la base de données que j'utiliserai.

3. **Définition d'entité JPA:** créez les classes Java qui représentent les entités de base de données. Utilisez les annotations Spring Data JPA pour mapper ces classes aux tables de base de données. Assurez-vous d'inclure les relations entre les entités, telles que @OneToMany et @ManyToOne.

4. **Création de référentiels:** créez des interfaces de référentiel à l'aide de Spring Data JPA. Ces référentiels fourniront des méthodes pour effectuer des opérations CRUD (créer, lire, mettre à jour, supprimer) sur la base de données.

5. **Création de services:** créez des services Spring pour gérer la logique métier. Les services utiliseront des référentiels pour accéder aux données de la base de données et les manipuler.

6. **Création de contrôleurs REST:** Créez des contrôleurs Spring pour exposer les API REST qui seront utilisées par le frontend. Ces contrôleurs recevront les requêtes HTTP du client (frontend), appelleront les services appropriés et renverront les réponses au frontend au format JSON.

7. **Configuration de la sécurité:** pour protéger certaines parties du site du portfolio, je peux configurer la sécurité à l'aide de Spring Security. Par exemple, je souhaiterais peut-être exiger une authentification pour accéder aux pages d'administration, nous verrons comment l'implémenter plus tard.

8. **Création d'Angular Frontend:** Créez un nouveau projet Angular à l'aide de Angular CLI. Concevoir et créer les pages et les composants du site portfolio, tels que la page maître, la page des projets, la page des compétences, etc. Utilisez le routage angulaire pour naviguer entre différentes pages.

9. **Appel des API REST:** utilisez le module HttpClient d'Angular pour envoyer des requêtes aux API REST backend de Spring Boot. Par exemple, vous pouvez appeler l'API pour obtenir des données de projet ou pour envoyer un message de contact.

10. **Présentation des données:** utilisez les directives angulaires pour présenter les données obtenues à partir du backend dans votre frontend. Par exemple, je peux utiliser \*ngFor pour parcourir une liste de projets et afficher les informations correspondantes.

11. **Style et mise en page:** utilisez CSS et/ou un cadre de style comme Bootstrap pour améliorer l'apparence de votre site de portfolio.

**Analyse d'entité**