**Cronograma de 2 Meses - Lista de Compras com Atualização em Tempo Real**

**Semana 1-2: Configuração Inicial e Estruturação**

**Backend (Você):**

* **Dia 1-3:**
  + **Configurar o projeto backend com Node.js e TypeScript.**
  + **Criar uma API básica com Express.**
  + **Definir as primeiras rotas: criação de usuário, login, e CRUD básico da lista de compras.**
* **Dia 4-6:**
  + **Configurar o banco de dados com Prisma e PostgreSQL.**
  + **Modelar os dados para usuários, listas de compras e itens.**
  + **Definir migrações automáticas.**
* **Dia 7:**
  + **Reunião de integração com o front-end para alinhar o formato de dados e endpoints.**

**Frontend (Colega):**

* **Dia 1-2:**
  + **Configurar o ambiente com React.**
  + **Instalar pacotes essenciais (React Router, Axios, etc.).**
* **Dia 3-5:**
  + **Criar uma estrutura de pastas organizada.**
  + **Definir as páginas principais: Login, Lista de Compras, Detalhes de Item.**
* **Dia 6-7:**
  + **Começar a trabalhar no design da tela de Login com estilização (CSS, Tailwind, ou Material-UI).**

**Semana 3-4: Autenticação e Primeiras Funcionalidades**

**Backend (Você):**

* **Dia 1-4:**
  + **Implementar autenticação JWT com jsonwebtoken.**
  + **Criar rotas para login e registro de usuários.**
  + **Implementar autenticação e middleware de proteção de rotas.**
* **Dia 5-7:**
  + **Adicionar WebSocket para atualização em tempo real das listas de compras.**

**Frontend (Colega):**

* **Dia 1-2:**
  + **Finalizar a tela de Login com integração à API de autenticação (usando Axios).**
* **Dia 3-4:**
  + **Implementar a lógica de login e armazenar o token JWT no localStorage.**
* **Dia 5-6:**
  + **Criar a tela de listagem de compras com dados mock.**
* **Dia 7:**
  + **Testar a integração com a API para login e listar os itens do usuário logado.**

**Semana 5-6: Implementação de Funcionalidades Centrais**

**Backend (Você):**

* **Dia 1-3:**
  + **Refinar o sistema de WebSocket.**
  + **Permitir a criação e atualização de listas de compras em tempo real.**
* **Dia 4-5:**
  + **Implementar Redis para caching das sessões dos usuários.**
* **Dia 6-7:**
  + **Otimizar consultas no Prisma e ajustar migrações conforme necessário.**

**Frontend (Colega):**

* **Dia 1-2:**
  + **Adicionar a funcionalidade de criação e remoção de itens da lista.**
  + **Atualizar a lista em tempo real com o backend.**
* **Dia 3-5:**
  + **Implementar WebSocket no front-end para receber atualizações em tempo real.**
* **Dia 6-7:**
  + **Trabalhar na funcionalidade de marcar itens como comprados/não comprados.**

**Semana 7-8: Finalização de Funcionalidades e Otimizações**

**Backend (Você):**

* **Dia 1-3:**
  + **Refinar a lógica de autenticação e autorizações (garantir que apenas usuários autorizados possam modificar as listas).**
* **Dia 4-5:**
  + **Criar testes unitários e de integração para as principais rotas (usando Jest ou Mocha).**
* **Dia 6-7:**
  + **Preparar o ambiente para CI/CD com GitHub Actions.**

**Frontend (Colega):**

* **Dia 1-3:**
  + **Refinar o design das páginas e melhorar a usabilidade da interface (UX/UI).**
* **Dia 4-5:**
  + **Implementar a lógica para exibir múltiplas listas de compras.**
* **Dia 6-7:**
  + **Criar componentes reutilizáveis para melhorar a organização do código.**

**Semana 9-10: Testes e Otimizações de Performance**

**Backend (Você):**

* **Dia 1-3:**
  + **Otimizar o uso de Redis para caching de dados frequentemente acessados.**
* **Dia 4-5:**
  + **Realizar testes de carga para simular múltiplos usuários em tempo real.**
* **Dia 6-7:**
  + **Refinar e corrigir problemas de performance identificados nos testes.**

**Frontend (Colega):**

* **Dia 1-2:**
  + **Testar todas as funcionalidades, garantindo que o fluxo de dados esteja funcionando corretamente.**
* **Dia 3-4:**
  + **Adicionar tratamentos de erros na interface.**
* **Dia 5-7:**
  + **Revisar o código para melhorar a performance da aplicação e otimizar o consumo de WebSocket.**

**Semana 11-12: Deploy e Preparação Final**

**Backend (Você):**

* **Dia 1-4:**
  + **Finalizar a configuração do CI/CD com GitHub Actions, integrando testes e deploy automático.**
* **Dia 5-7:**
  + **Realizar o deploy da aplicação, utilizando Docker ou Kubernetes para garantir escalabilidade.**

**Frontend (Colega):**

* **Dia 1-3:**
  + **Preparar o build da aplicação React para produção.**
* **Dia 4-5:**
  + **Testar a aplicação completa em ambiente de produção (deploy).**
* **Dia 6-7:**
  + **Reunião final para ajustar qualquer bug e alinhar feedbacks.**

**Tecnologias Chave para o Front-End:**

* **React: Para criar interfaces dinâmicas e rápidas.**
* **Axios: Para consumo de APIs REST.**
* **WebSocket: Para receber atualizações em tempo real.**
* **JWT: Para autenticação segura e persistência do login.**
* **CSS/Tailwind/Material-UI: Para estilização e usabilidade da aplicação.**

**Tecnologias Chave para o Back-End:**

* **Node.js com TypeScript: Para robustez e segurança no desenvolvimento.**
* **Express: Para criação de rotas e middleware.**
* **Prisma: Para gerenciamento eficiente do banco de dados.**
* **WebSocket: Para comunicação em tempo real.**
* **Redis: Para caching e melhoria de performance.**
* **JWT: Para autenticação segura.**
* **CI/CD: Para automação de deploy e testes (via GitHub Actions).**

**Esse cronograma está atualizado com as novas tecnologias que você está estudando e proporciona uma evolução gradual no aprendizado de TypeScript para backend.**