**開発環境手順書 パート2 (13:00 ~ 18:00)**

**Step 1: プロジェクトの整理とスタイリング (13:00 ~ 13:30)**

1. **Viteテンプレートのクリーンアップ**

* **App.tsx と main.tsx の不要なコードを削除**
* **App.css の中身を確認し、必要なスタイルのみ残す**
* **デフォルトのロゴやアイコンを削除**

1. **グローバルスタイルの初期設定**

* **index.css の整理**
* **ベースとなるスタイル（フォント、色、リセットCSS）を設定**

**テキスト

AI によって生成されたコンテンツは間違っている可能性があります。**

* **共通で使用する変数やユーティリティクラスの定義**

1. **アセットの整理**

* **public フォルダ内の不要なファイルを削除**

**Step 2: React Router & 必要パッケージの導入 (13:30 ~ 14:00)**

1. **react-router-dom のインストール**

**npm install react-router-dom @types/react-router-dom**

1. **src フォルダ内に以下の構造で各フォルダを作成**

**src/**

**├── components/**

**│ ├── layout/ // ヘッダー、フッター、ナビゲーションなど**

**│ ├── common/ // 再利用可能なUIコンポーネント**

**│ └── forms/ // フォーム関連のコンポーネント**

**├── pages/ // ページコンポーネント**

**├── assets/ // 画像やアイコンなど**

**├── styles/ // グローバルスタイルやテーマ**

**├── types/ // 型定義ファイル**

**└── utils/ // ユーティリティ関数**

1. **App.tsx の再設定**
   * **main.tsx の更新:**

**上記の画像のようにコードを更新**

**これでページ遷移の機能が使えるようになります**

**テキスト

AI によって生成されたコンテンツは間違っている可能性があります。**

* + **App.tsx に基本的なルート設定を用意する**

**テキスト

AI によって生成されたコンテンツは間違っている可能性があります。**

**Step 3: ベースコンポーネントの作成 (14:00 ~ 15:00)**

**MUIのインストール**

**npm install @mui/material @emotion/react @emotion/styled @mui/icons-material**

1. **ナビゲーションバー (Navbar.tsx)**

**テキスト

AI によって生成されたコンテンツは間違っている可能性があります。** テキスト

AI によって生成されたコンテンツは間違っている可能性があります。

* + **ホームページ（Home.tsx）**

**テキスト

AI によって生成されたコンテンツは間違っている可能性があります。** テキスト

AI によって生成されたコンテンツは間違っている可能性があります。

1. **フッター (Footer.tsx)**

**テキスト

AI によって生成されたコンテンツは間違っている可能性があります。テキスト

AI によって生成されたコンテンツは間違っている可能性があります。**

1. **よく使うMUIコンポーネント**

**<Container> // 中央寄せのラッパー。maxWidthで幅調整**

**<Box> // divのような汎用コンテナ。sxでスタイリング**

**<Typography> // テキスト要素。variant="h1"などで見出しに**

**<Button> // ボタン。variant="contained"で塗りつぶし**

**実際の使用例：**

<Container>

<Box sx={{ padding: '20px' }}>

<Typography variant="h1">会社名</Typography>

<Button variant="contained">お問い合わせ</Button>

</Box>

</Container>

**3つの主要コンポーネントの役割と繋がりを説明します：**

// App.tsx（メインの入れ物）

function App() {

return (

<div>

<Navbar /> {/\* ↓ 上部固定 \*/}

<Home /> {/\* ↓ メインコンテンツ \*/}

<Footer /> {/\* ↓ 下部固定 \*/}

</div>

)

}

* + **ルーティングテスト用に About.tsx や Contact.tsx をオプションで作成する**
  + **構造を図で表すと：**

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

| Navbar | ← ヘッダー（ナビゲーション）

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

| |

| Home content | ← メインコンテンツ

| |

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

| Footer | ← フッター

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

**Step 4: ルーティングの確認 (15:00 ~ 16:00)**

1. **App.tsx のルート設定**

**テキスト

AI によって生成されたコンテンツは間違っている可能性があります。**

1. **各ページコンポーネントの作成：**
   * **会社概要（About.tsx）**

**テキスト

AI によって生成されたコンテンツは間違っている可能性があります。テキスト

AI によって生成されたコンテンツは間違っている可能性があります。**

* + **問い合わせ（Contact.tsx）**

**テキスト

AI によって生成されたコンテンツは間違っている可能性があります。** **テキスト

AI によって生成されたコンテンツは間違っている可能性があります。**

**各リンクが正常に動作するかテストする**

**Step 5: ナビゲーションをレスポンシブ対応にする**

1. **実装内容：**

* **画面サイズの判定を追加（useMediaQuery）**
* **モバイル用のハンバーガーメニューを追加**
* **デスクトップ用のボタンはそのまま維持**

テキスト

AI によって生成されたコンテンツは間違っている可能性があります。テキスト

AI によって生成されたコンテンツは間違っている可能性があります。テキスト

AI によって生成されたコンテンツは間違っている可能性があります。

**実装方法：**

// モバイル画面かどうかを判定

const isMobile = useMediaQuery('(max-width:600px)')

// 画面サイズに応じてメニューを切り替え

{isMobile ? (

// モバイル：ハンバーガーメニュー

<MenuIcon />

) : (

// デスクトップ：通常のボタン

<Buttons />

)}}

**Step 6: 確認 & 追加作業 (17:30 ~ 18:00)**

1. **動作確認 & デバッグ**
   * **全ページのリンクが正しく機能するか確認**
   * **デベロッパーツール（F12）で画面サイズを変え、レスポンシブ表示をチェック**
2. **もし時間に余裕があれば**
   * **ページ遷移時の簡単なアニメーションを追加**
   * **フッターの追加やフォント設定などの微調整**