Projektplan: Planeringsapp

Gruppnamn: Grupp 6a (Tim, Linus & Kalle)

Syfte: Syftet med detta projekt är att skapa en planeringsapplikation som hjälper användare att strukturera och planera sina dagliga aktiviteter, både privat och professionellt. Applikationen riktar sig främst till privatpersoner som vill hantera sin tid effektivt, men även till yrkespersoner som behöver verktyg för att planera och organisera sitt arbete. Funktionen att skapa egna boards och använda widgets som budgethantering, inköpslistor, anteckningar och att-göra-listor erbjuder användarna flexibilitet att anpassa applikationen efter sina behov.

Förstudie: Några av de största utmaningarna vi identifierar är:

- **Dynamisk dashboard med grids:** Att bygga en interaktiv dashboard där användarna kan dra och släppa widgets kräver att vi implementerar en responsiv grid-layout och tillhandahåller en bra användarupplevelse oavsett plattform (webb och mobil).
- **Användarautentisering:** Integrationen av third-party login med Google, Microsoft, GitHub och eventuellt Discord innebär en viss komplexitet i att hantera OAuth-protokoll och säkerställa att inga känsliga data exponeras i frontend.
- **Backend-struktur:** Eftersom backend byggs i .NET, kräver det att vi säkerställer att all data och funktionalitet hanteras effektivt och säkert, särskilt när det gäller användardata och personliga planer.
- **Cross-platform kompatibilitet:** Applikationen måste fungera sömlöst på både webben och mobila enheter, vilket kräver noggrann planering och implementation i både React och React Native.

Lösningsförslag:

- För att hantera den dynamiska dashboarden kommer vi att använda CSS Grid, vilket ger oss en inbyggd lösning för att skapa en flexibel och responsiv layout. Vi kommer att definiera grid-områden och placera widgets dynamiskt med hjälp av grid-kolumner och -rader. CSS Grid gör det möjligt att enkelt hantera olika skärmstorlekar och skapa responsiva layouts.
- För dynamisk storleksjustering av widgets kan vi använda CSS Grid-funktioner som `grid-template-areas`, `grid-auto-flow`, och media queries för att anpassa layouten till olika skärmar.

User Stories/Features:

1. Användare kan skapa ett konto och logga in med Google, Microsoft, GitHub, eller Discord.
2. Användare kan skapa egna boards.
3. Användare kan lägga till widgets för budget, inköpslistor, anteckningar och att-göra-listor.
4. Användare kan anpassa layouten på sin dashboard genom att dra och släppa widgets.
5. Användare kan spara och återställa sina konfigurationer för olika boards.

6. Användare kan dela sina boards med andra användare.
7. Användare kan sätta upp påminnelser och deadlines för sina att-göra-listor.
8. Användare kan exportera sina planer och listor till PDF eller CSV.
9. Appen ska vara fullt responsiv på både mobil och webbläsare.
10. Autentisering och sessionshantering för att hålla användarens data säker.

Viktigaste User Story/Feature: Vi anser att den viktigaste funktionen är dashboarden och hur användaren kan interagera med och anpassa sina widgets. Detta kommer att utgöra kärnan i användarupplevelsen.

Tasks för dashboard-funktionen (med CSS Grid):

1. Skapa grundlayouten för dashboarden med CSS Grid.
2. Definiera grid-områden där widgets ska placeras, och implementera logik för att dynamiskt justera storlek och placering.
3. Implementera drag-and-drop-funktionalitet med JavaScript eller ett bibliotek som `react-dnd` för att placera widgets på olika grid-positioner.
4. Implementera widget-komponenter för budget, inköpslistor, anteckningar och att-göra-listor.
5. Använd media queries och `grid-template-columns/rows` för att skapa en responsiv layout som fungerar bra på både mobila och stationära enheter.
6. Skapa backend-API
   för att spara och hämta användarens dashboard-konfigurationer.
7. Testa användarinteraktion och justera layoutens flexibilitet.