Designen har jag försökt göra så lättläst som möjligt genom att ge kolumner och titlar till de nödvändiga fälten för objekten som visas upp i UI:n. Jag har valt att ha en flik för startsida som har en välkomstfras samt en kort förklaring på navigeringen för att förenkla användarupplevelsen. Utöver den första fliken finns ytterligare tre stycken;

Den första visar en lista av alla användarprofiler samt låter användaren välja vilken som ska användas (ett simulerat inloggningssystem men eftersom det inte fanns med i kravspecifikationen presenteras användarna genom en lista man kan välja från istället för en inloggningsskärm). Man ser även på skärmen vilken användare man är inloggad som och detta gäller alla flikar förutom startsidan.

Den andra tillåter användaren att se och filtrera i listan (efter tidigaste och senaste starttid samt pass typ)  av pass som går att boka samt att boka pass. Filtrering och bokning är båda bundna till varsin knapp som har logik för att reagera på alla olika scenarion som kan uppstå.

Sista fliken visar den aktuella användarens bokade pass och tillåter användaren att avboka pass genom att klicka på ett pass för att markera det och sedan klicka på knappen med texten "Avboka" som även har logik för att hantera om användaren ej har markerat ett pass (mer om detta i code behind delen).

I min code behind har jag skapat fält för alla listor och observable collections som ska visas upp och användaren ska kunna interagera med samt "Selected" fält för att spara ner användarens markerade eller valda användare, pass och filtreringsalternativ dvs starttid för pass och pass typ.

I Public MainWindow() ger jag bara direkta värden till listan över passen, resterande av fälten kan jag använda andra objekts fält, index i en lista eller Linq/lambda metoder. Så om man vill lägga till pass eller användare räcker det att skriva in detta i den angivna listan för aktuell objekttyp så uppdateras resten av gränssnittet korrekt.

Min code behind ärver även från INotifyPropertyChanged för att hantera förändringar av värdet på fält.

Metoderna i min code behind är alla click metoder förutom en som heter BokaPassFlik_ToDefault(). Metoden sätter boka pass fliken till att visa alla pass och ha förvalt alternativ i filtreringen som skulle visa alla tillgängliga pass om man klickar på filtrera. Denna kallar jag på i min MainWindow() så att jag slipper ange värdet för fälten som visas i boka passfliken igen men också vid SelectedAnvändares set metod vilket gör att filtreringen återställs vid byte av användare. Detta anser jag är något som gör att det känns bättre när man byter användare då man inte bör se någon annans filtreringsalternativ vid inloggning. Denna logiken hade även funkat vid ett tillagt inloggningssystem.

Mina click metoder hämtar logik ifrån Bokningshantering Klassen som jag har skapat i en separat fil tillsammans med pass- och användar klasserna. Boka- och avboka pass click metoderna har även logik för avvikelser. Så om en användare försöker boka ett pass som redan är bokat av den användaren eller ett pass som är fullbokat får användaren upp ett meddelande om detta och passet bokas ej. I bokade pass fliken är default att inget pass är markerat då alla användares bokade passlista är tomma. I första hand kommer ett meddelande upp om användaren inte har några bokade pass. Bokar användaren sedan pass och klickar på avboka knappen utan att markera ett pass får användaren upp ett meddelande om detta istället.

I filen för klasser har jag klasser för;

Bokningshantering som innehåller metoder med logik för filtrering, bokning och avbokning som jag kallar på i mina clickmetoder.

Pass som har fält för namn, passtyp, tid och antal platser. Denna klassen ärver också från INotifyPropertyChanged för att antal platser ska uppdateras korrekt när en användare bokar eller avbokar ett pass.

Användare som har fält för namn och bokade pass(en observable collection av pass). Även denna klass ärver ifrån INotifyPropertyChanged för att uppdatera bokade pass korrekt när en användare bokar eller avbokar ett pass.

Detta uppnår i min mening händelsestyrd programmering på ett effektivt sätt med click metoder som kallar på logik som hämtas ifrån en dedikerad klass. Click metoderna hanterar även alla typer av händelser ifall den önskade aktionen inte går att utföra med ett tydligt meddelande till användaren för att förenkla användarupplevelsen.

Skalbarheten gällande pass, användare och nya click metoder eller dylikt uppnås genom att;

Både pass och listor samt alla alternativ relaterade till dessa utgår ifrån två listor. Vid ett tillägg av ett pass eller en användare uppdateras resten av gränssnittet korrekt genom linq/lambda metoder eller hänvisningar av andra slag till dessa listor. Click metoderna är fristående och har vid klickandet av bundna knappar ett tydligt designerat ansvar som endast hanterar det som önskas hanteras. Med andra ord så stör inte logiken i click metoderna något annat som man potentiellt vill lägga till.

Länk till Laboration 3 repo: https://github.com/johhul99/Laboration-3