Struktur

Projektet innehåller 5 huvud klasser **Inventory**, **Category**, **SellerBuyer**, **SellingBuyingHistory** och **Customer**. **Customer** är **Extending** från **IdentityUser** och innehåller bara information som jag vill att användaren matar in vid registrering som namn och adress m.m. **Inventory** innehåller items information. Det finns 5 viktiga variabler som jag jobbade med främst i den här modellen. **SpecialId** som består av tre bokstäver och sex siffror. Jag har skrivit en funktion som tar hand om att skapa den här id **GetSpecialId** en funktion som finns i **InventoryRepository** och jag anropar funktionen varje gång användaren skapar ett nytt objekt. Jag vill notera att den här specialist id inte är **Pk**. **Decade** eller år är också viktigt för att kunden ville räkna hur mycket objektet kostade för att lägga in. Jag har skapat en funktion för att byta år till årtionde och för varje årtionde lägger jag till 10 kronor på priset. Funktionerna **GetSpecialPrice** och **ChangeYearToDecade** tar hand om det. Det finns också möjlighet att man jobbar vidare med **ChangYearToDecade**, till exempel att man visar årtionde i information för objekt.

FinalPrice ska alltid vara null när objektet skapas och status ska alltid vara **"Auction started"**. Båda ska vara så tills att Admin eller Auktions Ansvarig väljer slutpris och status. Så länge status är Auktion Started kan användaren se objektet och buda. Alla budning sparas i **SellerBuyer** model med både buyerId, sellerId, objektId och priset Admin kommer därefter att välja det bästa priset och markera objektet som såld. När Admin gör status som "**Delivered**" så kommer all budning som finns i Model SellerBuyer att raderas. Email sparas för både köpare och säljare, objekt namn, datum och tid samt priset i klass **SellingBuyingHistory**. Användarna kommer därefter att kunna se vilka objekt de vunnit på auktionen.

Säkerhet

Det var faktiskt inte svårt att jobba med Authenticating och Authorizing. Jag skapade **CustomerController** och samlade alla operationer som användaren behöver. Lägga till nya objekt, visa skapade objekt, visa köpta objekt, radera och även redigera. Med hjälp av Authorize låste jag hela Controller. Jag har även låst budgivningen. Dessutom har jag gömt alla saker som kräver registrering från chtml med hjälp av Razor och **SignInManager**.

Jag tycker att det inte var svårt jämfört med de andra små och viktiga detaljerna enligt mig. Det finns till exempel några detaljer som jag har lagt till i projektet. Endast den som skapat annonsen för försäljning kan redigera och radera den. Den som skapat annonsen har inte rätt att lägga bud på sina varor. Varan som har sålts kan inte ändras eller raderas från arkivet av användaren. Användaren har rätt att ändra det utgångspriset, men budpriset är fast.

Användare

Jag har bara skapat en role "Admin" via Modelbuilder, om man använder InMemoryDatabase så finns två användare User (User@gmail.com , User123!) , Admin (Admin@gmail.com , Admin123!) Men jag har också skrivit en funktion som fungerar bara en gång om man använder Sql istället. Om man använder sql så kommer ett meddelande att dyka upp och be om att sätta en admin och skapar en role "Admin". Därefter försvinner meddelandet och kommer aldrig att dyka upp igen tills rolen raderas.

Från Admin Panel kan Admin lägga till nya kategorier och radera dem och kan även skapa nya Admin eller radera dem. Jag har noterat att kunden var orolig över det här faktiskt. Jag gav Admin möjlighet att radera alla objekt oavsett om Admin skapade de eller inte. Admin kan välja slutpris och markera som såld och levererad och även visa köpare och säljare.