2023-01-05  
Gjort menyer för startsida och miniräknare.   
Gjort ett ER-diagram över hur tabellerna ska se ut i databasen.

2023-01-09  
Tänkt ut hur undermenyerna ska se ut och skrivit in detta som en kommentar.   
Gjort en koppling till databas via Code First och skapat en databas från koden.   
Skapat ett class library för beräkningar. Gjort ett till class library för felhantering. Funderade på om det skulle vara ett och samma class library men tänkte att det blir enklare att hantera dem ifall varje class library bara innehåller en sak. Om jag skulle vilja använda exempelvis felhantering i ett annat projekt så behöver jag inte ha beräkningar med i samma class library.

2023-01-10  
Gjort ett interface IShape för att kunna lägga till fler former, men vet inte hur det ska fungera. Kollat på youtube och gamla anteckningar om interfaces utan framgång, därför inte hänt så mycket med projektet.  
Skapat Sten, sax, påse meny för att välja att spela eller se statistik och undermeny för själva spelet.

2023-01-11  
Kollat mer på youtube om dependency injection.  
Kopierat kod från tidigare sten, sax, påse spel och anpassat så att koden fungerar ok i projektet.  
Lagt till CRUDMenu och createShape. I Shape-menyn ska den form som valts skickas med (genom att använda IShape) och med den skapas olika former utifrån respektive beräkningar.  
Skapat beräkningar av formerna.   
Lagt till klasserna CreateShape och ReadShape som ska gälla för alla tänkbara former som kan skapas så att det bara behövs en klass för själva formens uträkning om man vill redigera programmet och lägga till en ny form.

2023-01-12  
Försökt få till strategy pattern i Calculate för att slippa upprepa kod. Hade visst redan börjat med strategy pattern i shapes så jag fortsatte med det där också. Svårt att få till klasser som ska implementera samma interface men göra olika saker om de ska ta in olika parametrar. Löste ett sådant problem genom att inte använda alla parametrar som skickades in i klassen och metoden.

2023-01-13  
Försökt att få till strategy pattern i miniräknaren för att slippa newa upp klasser fler gånger. I samband med byte av branch glömde jag att spara en ändring i masterbranchen så att det blev en merge-konflikt när jag skulle komma tillbaka till master-branchen. Jag kunde till en början göra en merge men det hände inget med klasserna, de hängde inte med. Jag lyckades inte lösa det så jag kopierade in klasserna i master vilket inte är optimalt eftersom jag kan ha missat något. Jag hade också jobbat väldigt mycket i branchen som skulle mergeas in i master så det blev mycket kod som fick kopieras. Det är väldigt lätt att feature-brancherna blir stora för att jag tappar fokus i vad jag håller på med, till exempel att jag hittar ett fel som jag vill ändra innan jag glömmer bort det och sen minns jag inte vad jag höll på med innan. Försöker bli bättre på att arbeta med en sak i taget och skriva en lista på saker som jag ska göra för att inte tappa bort mig.

Idag har jag lärt mig att jag måste försöka fokusera på en sak i taget för att inte tappa bort vad jag håller på med och att komma ihåg att spara innan jag byter från master-branchen så att det inte blir en merge-konflikt.  
Har lagt till flera interfaces och försökt använda mig av dependency injection för att skicka med information till nästa klass.

2023-10-14  
Igår la jag till Data annotations decimaler för att göra det möjligt att lägga in större tal i databasen – trodde jag. Det gick inte som jag tänkt mig och programmet kraschade med ett felmeddelande att talet jag försökte lägga in i databasen var för stort, därför tog jag bort dem idag.   
Igår la jag till en calculationstrategy som en property i calculation-klassen i efterhand eftersom jag kom på att jag vill visa det i tabellen och för att det blir enkelt att spara ner beräkningsmetoden i beräkningen som ska göras.

2023-01-15  
Gjort enum istället för string som namn på formerna för att minska risken för fel. Ändrat i Datainitialzer så att de fyra formerna seedas från början. Min tanke är att varje form ska förekomma en gång i tabellen och att det är resultaten som ska cruddas och kan förekomma fler gånger.

2023-01-16  
Gjorde om create och update shape om det är en triangel så att man bara ska behöva skriva in basen för triangeln. Om användaren även får skriva in höjden så måste programmet kontrollera så att höjden stämmer och det kan bli fel. Eftersom det går att räkna ut höjden med hjälp av basen på triangeln så är det onödigt att användaren även ska skriva in höjden och riskera att skriva fel.

2023-01-17  
Ändrat så att det inte läggs till en ny shape varje gång ett nytt shape result skapas. Då innehåller tabellen shapes bara en av varje form och kan kopplas till resutltatet genom formens id.

README  
Ett projekt som innehåller tre olika delar: en miniräknare , en del som är ett sten, sax, påse-spel och en del som utifrån användarens input räknar ut area, omkrets och i ett fall höjd på olika former,

EFCore Code First, Strategy Pattern, Dependency Injection och tre olika Class Libraries; ett med olika färger på texten och linjer som ofta används, ett med felhantering och ett där det finns olika typer av beräkningar.