

# Projektplan

## Webb-baserat bokningssystem för flyg

## Innehållsförteckning

Inledning & problembeskrivning.....	1
Systembeskrivning.....	2
Affärsobjekt.....	2
Databasen.....	4
Ajax - jQuery.....	4
Webbtjänst.....	4
Gränssnitt.....	5
Lösningen begränsningar.....	6
Problem & reflektioner.....	6
Referenslitteratur.....	6

## Inledning & problembeskrivning

***Eftersom systemet är tänkt att säljas om det blir bra uppmanas du som läsare att hålla detta dokument konfidentiellt!***

Projektet syftar till att skapa *användbara* gränssnitt med ASP.NET och C#. Dessa kommer att användas för att skapa ett webbaserat bokningssystem för flyg. Webbplatsen ska vara stilren och framförallt enkel med lagom information på varje sida. Menysystemet ska även vara kort & koncis med flera nivåer där det krävs. Alla funktioner bör vara kort förklarade på varje sida med en eventuell hjälpsida eller hjälpsnitt där det kan vara nödvändigt.

I första hand syftar projektet till att skapa tekniska lösningar, design kommer i andra hand. JavaScript (ajax) kommer användas för att utöka funktionalitet men skall inte vara ett krav för användaren för att kunna använda webbplatsen (en del användare har inte javascript installerat / aktiverat). Ett exempel där javascript kan tänkas användas är i formulär som förslag till användaren från databaser.

Systemet syftar till att hjälpa kunder hitta billiga flygningar, själva bokningsprocessen kommer dock i slutändan ske genom flygbolaget. För att systemet (företaget) ska gå med vinst måste kunder registrera sig för att ta del av essentiell bokningsinformation såsom flygbolag och kontaktinformation. Utan registrering kan kunder endast se destinationer och priser, detta kommer förhoppningsvis leda till att de registrerar sig.

## Systembeskrivning

Projektet är utfört med Asp.net 3.5 i Visual Studio 2008, databasen som har använts är Microsoft SQL SMSE. Projektet använder sig av en treskiktad lösning med ett affärsobjekt och en webbtjänst som mellanlager (mellan databasen och gränssnittet). Affärsobjektet är utav god objektorienterad stil.

## Affärsobjekt

Största delen av systemlogiken är implementerad inuti affärsobjektet. Gränssnittet har en mindre del kod som endast påverkar gränssnittet.

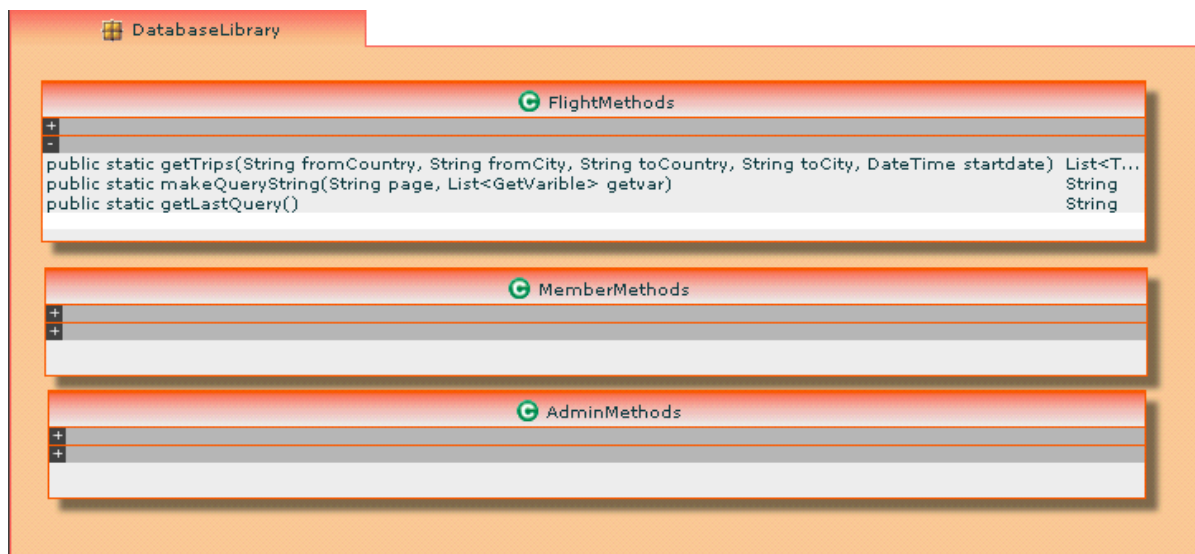


Illustration 1: Affärsobjekt

**FlightMethods** är standardmetoder som används för alla användare, **MemberMethods** är metoder som används för medlemmar och **AdminMethods** är metoder för administratörer.

Metoden `getTrips()` returnerar alla resor från en destination till en annan från ett visst datum. Det går bra att skicka in tomma strängar för allt utom datumet (som inte heller är en sträng).

Metoden `makeQueryString()` används för att skapa en så kallad "query string" med GET-variabler, det är browsers "synliga variabler" när man surfar. Ett exempel:

<http://www.website.com/products.html?id=1296&category=things>

Genom att skicka in en lista av datatypen `GetVariable` (egen datatyp, se illustration 2) och sidan användaren ska dirigeras till får man tillbaka en "query string" enligt exemplet ovan i fetstil.

Metoden `getLastQuery()` returnerar senaste begärda sträng vilket är användbart för tillbakaknappar.

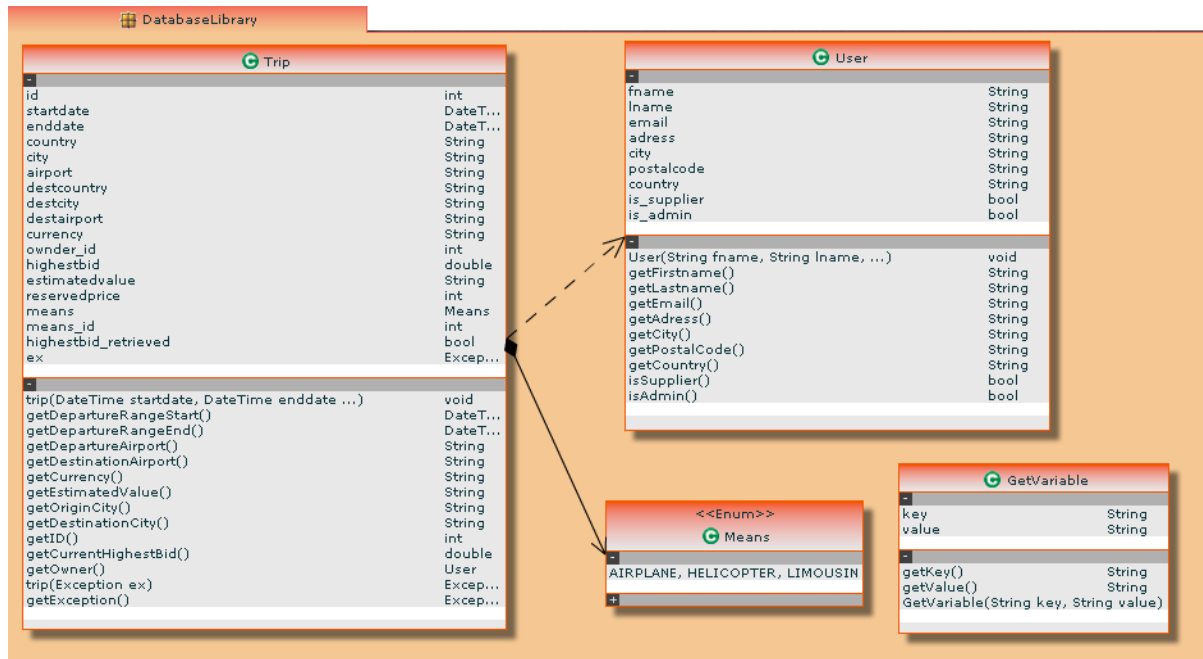


Illustration 2: Instansobjekt

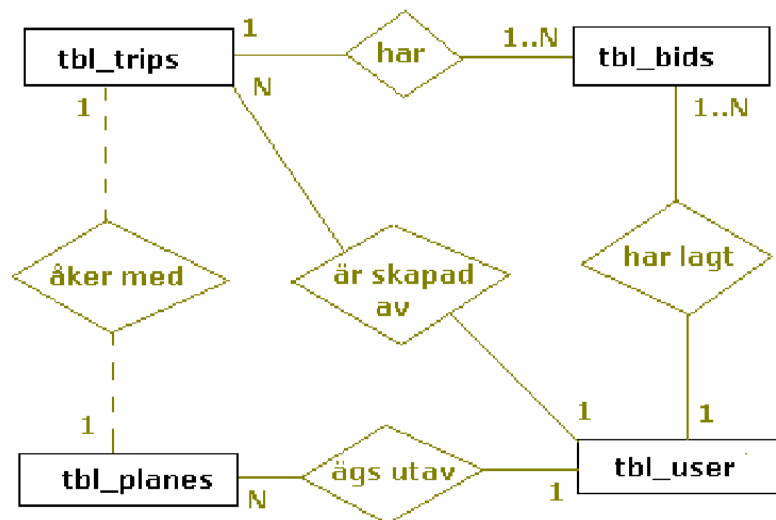
Trip motsvarar en resa, när ett objekt instansieras sparas det högsta budet för omedelbar hämtning eftersom *getTrips()* från *FlightMethods* returnerar en lista med *Trip*'s. För säkerhetsskull går det endast att hämta det högsta budet (*getHighestBid()*) en gång. Om metoden anropas en andra gång kastas därför ett undantag eftersom det högsta budet eventuellt har ökat sedan det hämtades från databasen.

*Currency* är valutan som används för budgivning för att förenkla för flygbolag från olika länder.

Metoden *getException()* returnerar ett undantag om ett sådant har kastats annars returnerar den null (se problem och reflektioner).

*Means* är en Enum som definerar vilken typ av resa det handlar om; flyg, helikopter eller limousin.

## Databasen



Relationen mellan **tbl\_trips** och **tbl\_planes** är streckad eftersom den **inte** måste existera.

## Ajax - jQuery

För att hämta förslag på länder och städer till användaren har Ajax biblioteket jQuery använts. Förslagen hämtas från en webbtjänst som i sin tur anropar databasen och hämtar de länder och städer som det finns resor mellan. En utökning av biblioteket för auto-complete har använts som stöder xml (se referenslitteratur).

## Webbtjänst

Webbtjänsten innehåller följande metoder:

```
public List<string> getLocationCities(string q, int limit)
public List<string> getDestinationCities(string q, int limit)
public List<string> getLocationCountries(string q, int limit)
public List<string> getDestinationCountries(string q, int limit)
```

## Gränssnitt

# FLIGHTACCESS

*This is where the tagline for the company would go.*

<b>Origin</b>	<b>Destination</b>	<b>Members</b>
Country: <input type="text" value="Sweden"/>	Country: <input type="text" value="Norway"/>	E-mail <input type="text"/>
City: <input type="text" value="Stockholm"/>	City: <input type="text" value="Oslo"/>	Password <input type="text"/>
	When: <input type="text" value="2009-06-12"/>	<input type="button" value="Login"/>
	<input type="button" value="Search"/>	<input type="button" value="Register"/>

[How does it work?](#)   [About the company](#)   [FAQ](#)   [Supply us your flights](#)

Illustration 3: Startsidan

# FLIGHTACCESS

*This is where the tagline for the company would go.*

<b>Location</b>	<b>Search results</b>	<b>Members</b>
Country: <input type="text" value="Sweden"/>	<b>You have to <a href="#">register</a> or <a href="#">login</a> to place bids!</b>	E-mail <input type="text"/>
City: <input type="text" value="Stockholm"/>	Results for trips from Stockholm, Sweden to Oslo, Norway.	Password <input type="text"/>
<b>Destination</b>		<input type="button" value="Login"/>
Country: <input type="text" value="Norway"/>		<input type="button" value="Register"/>
City: <input type="text" value="Oslo"/>		
When: <input type="text" value="2009-06-12"/>		
<input type="button" value="Search"/>		

Departure range	Trip	Highest bid	
2009-10-20 09:00:00 - 2009-10-20 18:00:00	Stockholm - Oslo [REGISTER TO SEE AIRLINE]	50 SEK	<a href="#">View Details</a>

[How does it work?](#)   [About the company](#)   [FAQ](#)   [Supply us your flights](#)

Illustration 4: Sidan 2 / Sökresultat

# FLIGHTACCESS

*This is where the tagline for the company would go.*

## Flight Information

**Preferred Origin:** Stockholm, Arlanda (ARL)  
**Preferred Destination:** Oslo, Gardermoen (OSL)  
**Departure Range:** 6/4/09, 9:00 am – 6/4/09, 1:48 pm

## Offer Amount

**Estimated Value:** \$8,296 – \$53,899  
**Offers:** 0  
**Current High Offer:** \$0.00

## Plane Information

**Aircraft:** G-450  
**Class:** Heavy  
**Year Manufactured:** 2005  
**Seats:** 14  
**Lavatory Type:** Enclosed



## Lösningen begränsningar

AdminMethods och MemberMethods är inte implementerade. FlightMethods behöver fler metoder.

Trip.getHighestBid() ska bara användas i direkt samband med hämtning av data (getTrips() t.ex.)

Databasen: En resa måste ha minst 1 bud för att kunna listas, därför ska ett bud (tbl\_bids) på 0 således alltid göras när resor läggs till.

## Problem & reflektioner

Att anropa en lagrad procedur **från** en webbtjänst verkar inte fungera. "SELECT TOP @limit FROM tabell" skickar sql-satsen bokstavigt talat (den ersätter inte @limit) med den parameter du anger). Detta tog mig ganska lång tid att komma underfund med.

Att få auto-complete funktionen att fungera tog mig flera dagar. Webbtjänsten måste ligga i samma projekt eller på en riktig webbserver. Databaser på andra servrar är förbjudna att kontaktas som standard av säkerhetsskäl har jag upptäckt.

## Referenslitteratur

Updated jQuery Autocomplete Plugin for ASMX & XML

<http://www.zachhunter.net/blog/Trackback.aspx?guid=e3e394c1-4a23-4fb5-9682-3121405a7560>