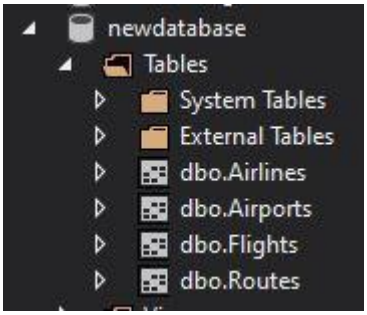
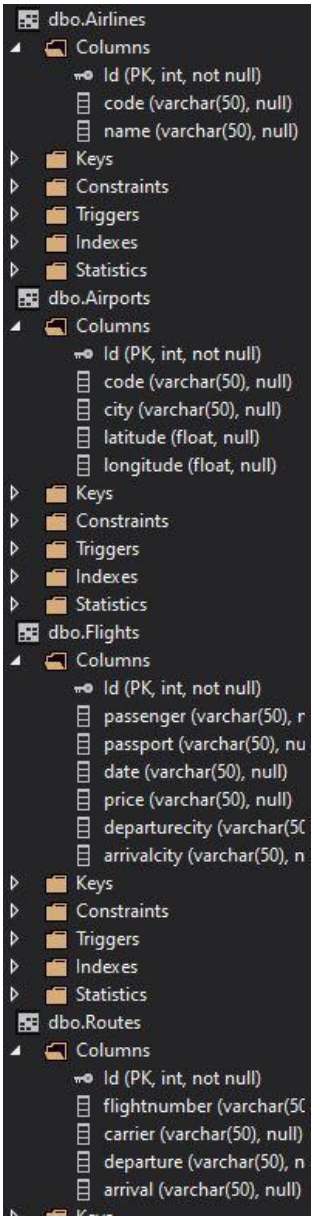
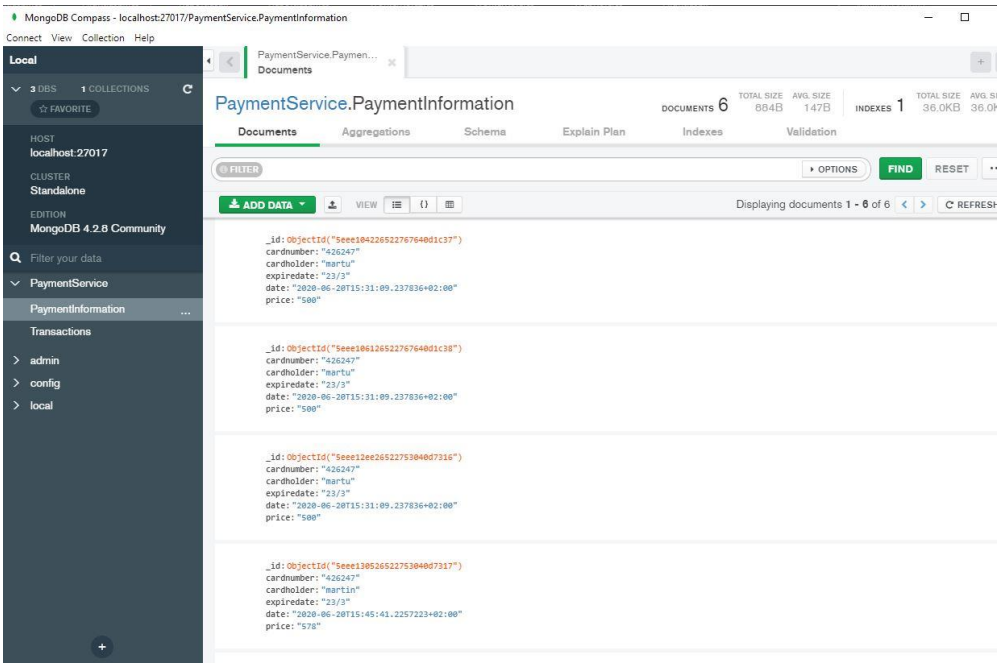
Snippets & documentation

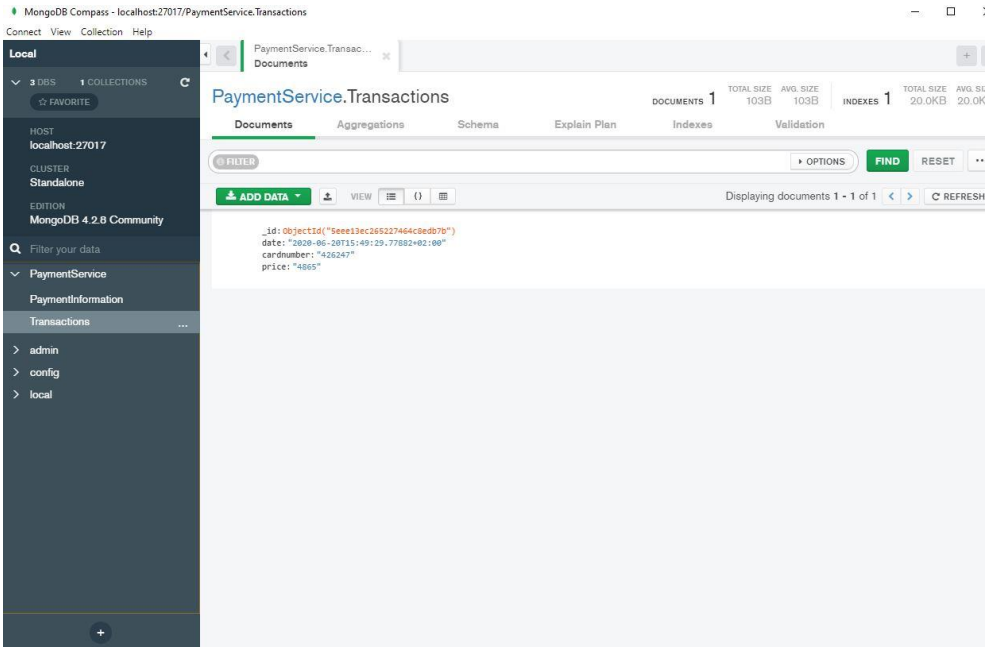
First created the SQL database in the local emulator. Added the tables that were needed and the columns in those tables.





Next was creating the mongoDB for the non SQL, used Bsondocument for storing information in the tables. One for transaction and other for card holders.





Then i set up the SQL connection string and started adding values into the database, as can be shown below. This was done in the front end of the application, client side.

```
{
    //          conn = new SqlConnection("Server=tcp:dawitest.database.windows.net;Database=mynewdb;User ID=dawitmen;Password=DA376c2020;Encryp
    conn = new SqlConnection("Data Source = (localdb)\\MSSQLLocalDB; Initial Catalog = newdatabase; Integrated Security = True; Connect Timeout
}
```

```
//insertion into the SQL database flight info
conn.Open();
cmd = new SqlCommand(@"INSERT INTO dbo.Flights (passenger, passport, date, price)
                       VALUES (@passenger, @passport, @date, @price)", conn);
cmd.Parameters.Add(new SqlParameter("passenger", nameText.Text));
cmd.Parameters.Add(new SqlParameter("passport", passportNr.Text));
cmd.Parameters.Add(new SqlParameter("date", date));
cmd.Parameters.Add(new SqlParameter("price", Session["price"].ToString()));
cmd.ExecuteNonQuery();
conn.Close();
```

After that I had to add mongoDB interface that was supposed to be done in the backend. I added a JSON object that I sent with queues to the server. When I got the Json queue message I parsed it into a BsonDocument that could be stored into the mongoDB.

```
//Create data for database
DateTime date = DateTime.Now;
string sqlFormattedDate = date.ToString("yyyy-MM-dd HH:mm:ss.fff");


JObject jObject = new JObject();

jObject.Add("cardnumber", cardNumber.Text );
jObject.Add("cardholder", nameText.Text);
jObject.Add("expiredate", expires.Text );
jObject.Add("date", date);
jObject.Add("price", totalCost.Text);

JObject transObject = new JObject();

transObject.Add("date", date);
transObject.Add("cardnumber", cardNumber.Text);
transObject.Add("price", totalCost.Text);


try
{

    StorageCredentials creds = new StorageCredentials(accountName, accountKey);     //Acc
    CloudStorageAccount storageAccount = new CloudStorageAccount(creds, useHttps: true);

    CloudQueueClient queueClient = storageAccount.CreateCloudQueueClient(); //Create an i

    // Retrieve a reference to a specific queue
    //Create ques for backend database storage
    CloudQueue queue = queueClient.GetQueueReference("mongodatabase");
    CloudQueue transqueue = queueClient.GetQueueReference("mongotransaction");
    // Create the queue if it doesn't already exist
    queue.CreateIfNotExists();
    transqueue.CreateIfNotExists();
    //remove any existing messages (just in case)
```

```
private string connection = "mongodb://localhost:27017/PaymentService?strict=false";
```

```
if (payMessage != null)
{
    JObject payInfo = JObject.Parse(payMessage.AsString);
    BsonDocument doc = BsonDocument.Parse(payInfo.ToString());

    //MONGO insertion
    MongoClient dbClient = new MongoClient(connection);
    var database = dbClient.GetDatabase("PaymentService");
    var getPaymentInformation = database.GetCollection<BsonDocument>("PaymentInformation");
    //insert
    await getPaymentInformation.InsertOneAsync(doc);

    payqueue.DeleteMessage(payMessage);
```

Now when all was stored into the database the only thing left is to report some values from the databases. To display the SQL values I used a gridview that was connected to the SQL database and just displayed those in tables on a webpage.

```
<asp:GridView ID="GridFlights" runat="server" AutoGenerateColumns="False" DataSourceID="Flights" >
    <Columns>
       <asp:BoundField DataField="Id" HeaderText="flightId" />
        <asp:BoundField DataField="passenger" HeaderText="passenger"  />
        <asp:BoundField DataField="date" HeaderText="Date"  />
        <asp:BoundField DataField="price" HeaderText="Price"  />
        <asp:BoundField DataField="departurecity" HeaderText="DepartureCity"  />
        <asp:BoundField DataField="arrivalCity" HeaderText="ArrivalCity"  />
    </Columns>
</asp:GridView>
<asp:SqlDataSource ID="Flights" runat="server" ConnectionString="Data Source=(localdb)\MSSQLLocalDB;Initial Catalog

  <asp:GridView ID="GridAirlines" runat="server" AutoGenerateColumns="False" DataSourceID="SqlDataSource3" >
    <Columns>
       <asp:BoundField DataField="code" HeaderText="AirlineCode"  />
        <asp:BoundField DataField="name" HeaderText="AirlineName"  />

    </Columns>
</asp:GridView>
<asp:SqlDataSource ID="SqlDataSource3" runat="server" ConnectionString="Data Source=(localdb)\MSSQLLocalDB;Initial


<asp:GridView ID="GridAirports" runat="server" AutoGenerateColumns="False" DataSourceID="SqlDataSource1" >
    <Columns>
       <asp:BoundField DataField="code" HeaderText="Airports"  />
        <asp:BoundField DataField="city" HeaderText="City"  />
        <asp:BoundField DataField="latitude" HeaderText="Laititude"  />
        <asp:BoundField DataField="lonitude" HeaderText="Longitude"  />
    </Columns>
</asp:GridView>
<asp:SqlDataSource ID="SqlDataSource1" runat="server" ConnectionString="Data Source=(localdb)\MSSQLLocalDB;Initial
```

For the mongo report it was a bit harder to actually retrieve all items from the database. However, I was able to do it with AsQueryable method and iterate through all the objects in the database and then display all these in a simple label on the html page.

```
var database = dbClient.GetDatabase("PaymentService");
var getPaymentInformation = database.GetCollection<BsonDocument>("PaymentInformation");

var documents = getPaymentInformation.AsQueryable();

//Create stringbuilder for html display
StringBuilder builder = new StringBuilder();
int i = 0;
foreach (BsonDocument doc in documents)
{
    //Retrieve every object for customer data
    builder.Append("<br/><br/>Iteration: " + i);
    builder.Append("<br/>Card Holder: " + doc.GetElement("cardholder").Value.ToString());
    builder.Append("<br/>Card Number: " + doc.GetElement("cardnumber").Value.ToString());
    builder.Append("<br/>Card Experation Date: " + doc.GetElement("expiredate").Value.ToString());
    builder.Append("<br/>balance: " + "1000");

    i++;
}

IterationText.Text = builder.ToString();


StringBuilder onebuild = new StringBuilder();
```

```
//retrieve one specific object for customer data
foreach (BsonDocument doc in documents)
{
    if (doc.GetElement("cardholder").Value.ToString().Equals("martin")){

        onebuild.Append("<br/><br/>Specific Card Holder");
        onebuild.Append("<br/>Card Holder: " + doc.GetElement("cardholder").Value.ToString());
        onebuild.Append("<br/>Card Number: " + doc.GetElement("cardnumber").Value.ToString());
        onebuild.Append("<br/>Card Experation Date: " + doc.GetElement("expiredate").Value.ToString());
        onebuild.Append("<br/>balance: " + "1000");
    }

}

specificholder.Text = onebuild.ToString();
```