Rainbow table simulation https://en.wikipedia.org/wiki/Rainbow_table	1. Generate rainbow table for n characters (example 5) We need this saved in a database table
	2. Compute MD5/SHA hashes
	The table should look something like this: TABLE rainbow id password md5hash sha1hash sha2hash ...
	3. Implement ReverseHash() function which will have a hash and a type of hash as input and return the password
Database connection from C# code	Use Ado.Net https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/ado-net-code-examples
	1. SqlClient
	1.1. Insert single row multiple times 1.2. Insert multiple rows 1.3. Use bulk insert https://msdn.microsoft.com/en-us/library/system.data.sqlclient.sqlbulkcopy.aspx
	2. Entity Framework
	3. Do your own reasearch here ...
Measure performance	Measure (in seconds) how much every method takes: a. 1.1. Insert single row 1.2. Insert multiple rows ... Extra: Create a reusable component for measuring the performance Hint: read about Stopwatch() b. Measure reading time when searching for a hash

I hope you guys will hit some limitation: OutOfMemoryException ... if not you haven't pushed the system enough	
Transactions	Try to put all code in a transaction and see what will happen 1. Play with Ado.Net transactions 2. Play with EF implicit and explicit transactions
Indexes MSSQL	clustered vs nonclustered index sql Add an index where you guys feel will be needed
Measure performance after adding the index	Expecting some king of report here too
Reading material:	
https://en.wikipedia.org/wiki/Salt_(cryptography)	
https://en.wikipedia.org/wiki/List_of_hash_functions	
http://www.entityframeworktutorial.net/what-is-entityframework.aspx	**!!! Really good content on Entity Framework**
https://app.pluralsight.com/library/courses/entity-framework-6-getting-started/table-of-contents	pluralsight
https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/transactions-and-concurrency	Transactions
https://docs.microsoft.com/en-us/sql/relational-databases/indexes/clustered-and-nonclustered-indexes-described?view=sql-server-2017	