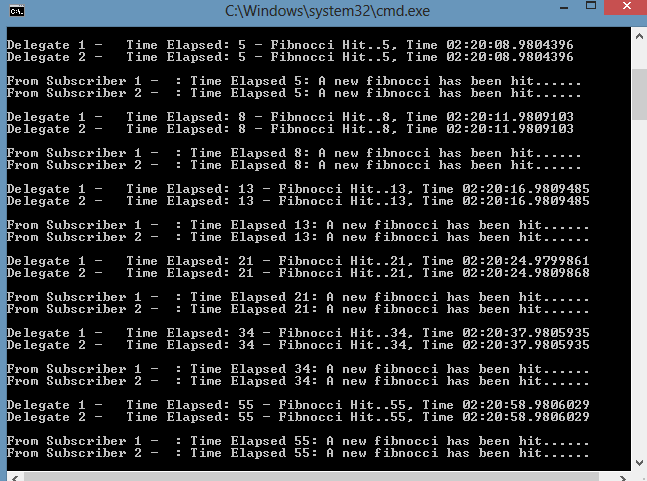
**Fibnocci Numbers with observer pattern**

This solution has been implemented with 3 classes.

1. Generic observer class.
2. The second observer with events and delegates.
3. A class to generate fibnocci numbers.
4. Each observer is subscribed twice.

**See the output from the console application**



**Customer Activity**

|  |
| --- |
|  |

This project is implemented using layered architecture.

It consists of the following layers

1. **Model layer** to implement all the classes required in the solution(**CustomerDataModel**).

* Includes a CRUD class to pass on errors between Business layer and WCF layer.
* Metadata classes to implement Data annotations(Validations) - To avoid overwriting of classes with EF.
* EF Generated classes separated from the Data Model layer for reusability across layers.

1. **DAL layer** to implement data access operations(CRUD-**CustomerDAL**).

* Insert update delete operations are carried out in this layer.
* This layer consists of one class **GenericDataRepository** and one interface **IGenericDataRepository**.

1. **Business Layer** sits between service layer and DAL(**CustomerBusinessLayer**).

* Service layer makes call to the DB through business layer.
* This layer consists of one class **BusinessLayer** and one interface **IBusineessLayer.**

1. **Service Layer** using WCF Services(**CustomerServiceLayer**)

* All the operations are implemented in service layer through business layer.
* This layer can be hooked to any presentation layer(UI).
* Service layer exposes faults through **FaultContract** defined in the ICustomerServiceLayer.
* This layer consist of one class **CustomerServiceLayer** and one interface **ICustomerServicelayer.**

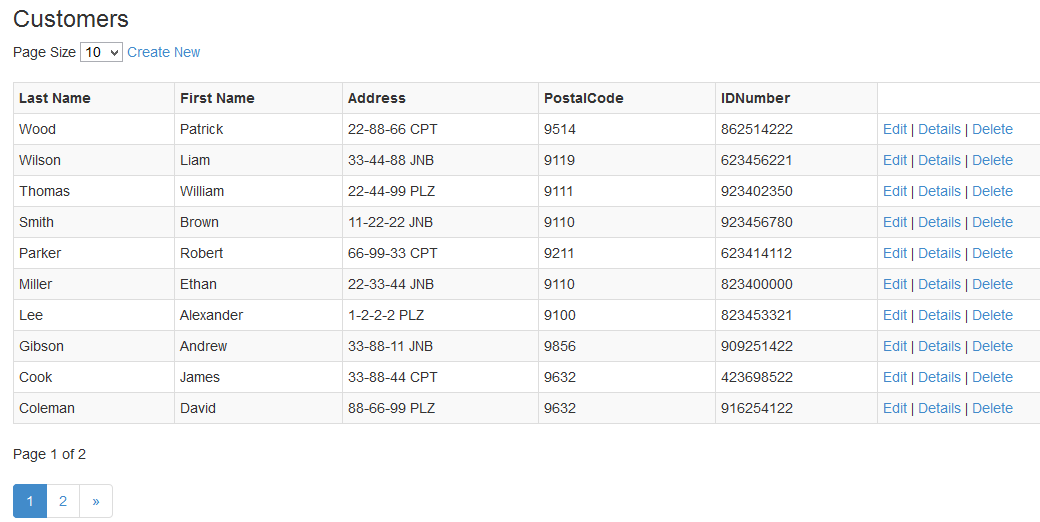
1. **Presentation layer** is implemented with MVC(**CustomerMVC**).

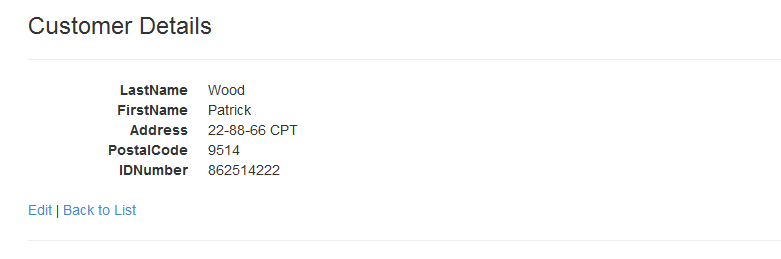
* Service layer uses the **WCF** service proxies to perform operations required.
* This layer consist of Controllers(**CustomerController**) and Views(Index - For customer display, Details -To show a specific customer, Edit -To edit a customer, Delete - To delete a customer.

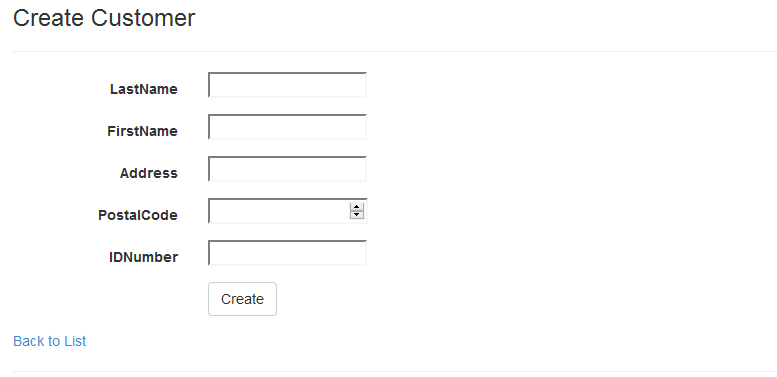
1. **Error Module** to log errors at all layers(**CustomerLoggerNLog**)

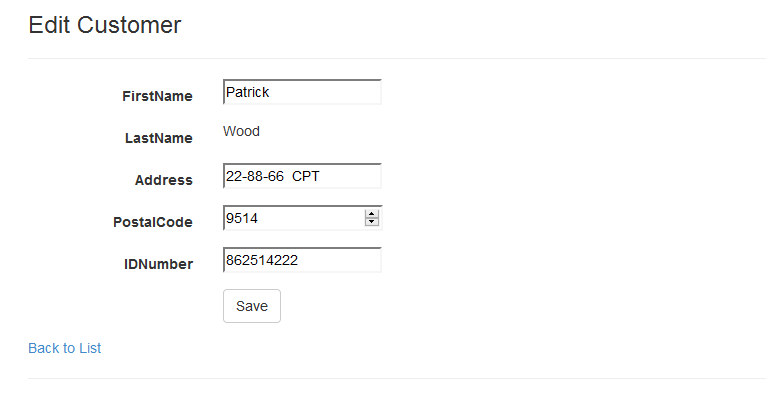
* This module is build on third party **NLog** library.
* NLog can be customized for File and DB Logging.
* Log(s) can be view from Sentinel and Harvester UI.

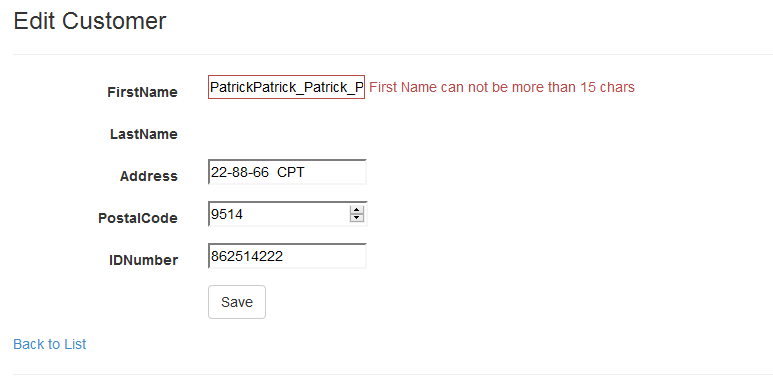
**Customer Screen Shots.**

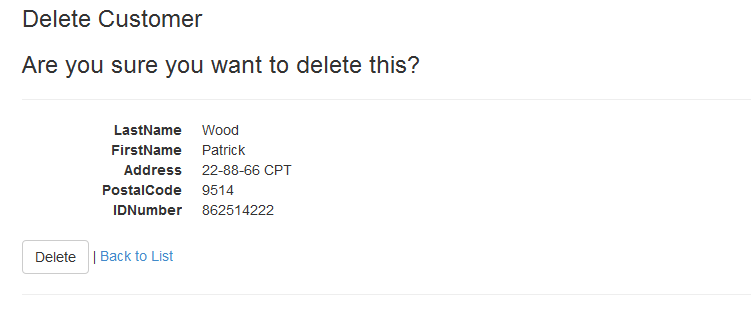
****

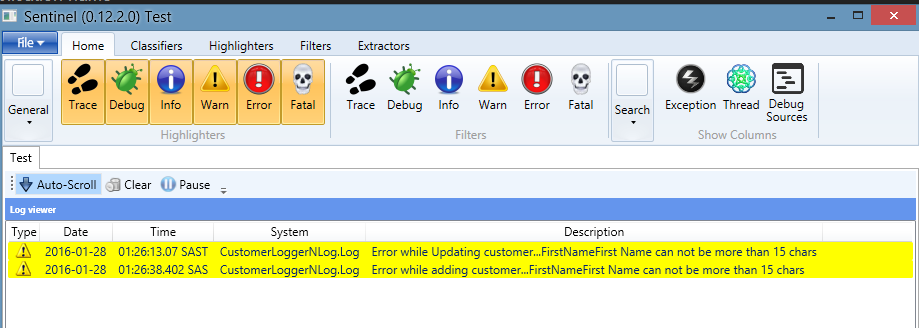
****

****

****

****

****

****