1. Layered architecture. Separates concerns, dependency injection of services and repositories – as done in a martin fowler refactoring article. One problem my implementation is my Entities layer. This should really have been put in the business layer, and not used in data layer
2. Stored procedures. Only done for customers
3. Unit of work pattern. Ensures that multiple database operations are completed as a single transaction. Ensures that either all or none of the operations succeed.
4. Strategy + simple factory. Each report performs the same function but is done differently. Simple factory is used to instantiate the correct report type.
5. Rental Façade – Rental interacts with multiple elements, customers, tools, rentals, rentalitems. Simplifies the interaction between the presentation layer and business layer. The presentation layer only needs to interact with the façade, rather than each individual component.
6. Rental builder – doesn’t have an immediate benefit, as the system expands and the introduction of things like insurance or discounts, rentals can be expanded easily.
7. FluentValidation framework on customers.
8. Code quality improvements such as centralised nav to reduce large chunks of duplicate code. Moved db connection string to app.config