**Validation Documentation**

**Jade Gorton - 7100288111**

Contents

[Client requirements 3](#_Toc434567132)

[Requirements 5](#_Toc434567133)

[Design Review 6](#_Toc434567134)

[Design Validation 6](#_Toc434567135)

[UID Check list 12](#_Toc434567136)

[User interface checklist - 2D Graphical Interface Databases 12](#_Toc434567137)

[Risk Management 16](#_Toc434567138)

[Source Code Review 20](#_Toc434567139)

[Source Code Validation 20](#_Toc434567140)

[Agenda 28](#_Toc434567141)

[Test Plan 29](#_Toc434567142)

[Testing Validation 29](#_Toc434567143)

[Software Test Plan & Technical Documentation 32](#_Toc434567144)

# Client requirements

**Mr Sparkly – Requirements Review**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Client Script | Technical Documentaion.docx |
| 2 | Manage Suppliers | Completed |
| 3 | Manage materials | Completed |
| 4 | Manage purchases | Completed |
| 5 | Manage retailers | Completed |
| 6 | Manage Products | Completed |
| 7 | Manage sales | Completed |
| 8 | Gather user requirements | Completed |
| 9 | Purchase Orders | Completed |
| 10 | Raw Materials Transformation Process | Out of scope |
| 11 | Laundry Products | Completed |
| 12 | Sales Order | Completed |
| 13 | Retailers | Completed |
| 14 | Manage growing # of suppliers | Completed |
| 15 | Manage list of materials | Completed |
| 16 | Manage purchases of raw materials | Completed |
| 17 | Manage growing list of retailers | Completed |
| 18 | Manage different brands of detergents | Completed |
| 19 | Sales by sales agent to retailers | Completed |
| 20 | List of growing employees | Completed |
| 21 | User Security rights | Completed |
| 22 | User Logins | Completed, used for web application & simple user login validation for desktop application |

# Design Review

## Design Validation

**Application - Design**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Timelines and costing | Not included |
| 2 | Development Methodology | Research & Selection of methodology sent via Email. |
| 3 | Programming Standards | Not included, PDF provided by Manager on Programming standards used |
| 4 | System Considerations | Not included |
| 5 | Security Issues and Responsibilities | Not Included |
| 6 | Performance Issues | Not included |
| 7 | Networking Considerations | Not included |
| 8 | Hardware Requirements | Not Included |
| 9 | Operating System Requirements | Not Included |
| 10 | Other Supporting Software Requirements | Not Included |
| 11 | Development Environment | Explained In, Logging for .Net Framework. |
| 12 | Backup and Recovery Strategies and Responsibilities | Not Included |
| 13 | Data Integrity and Validation | Not Included |
| 14 | Managing The System - Roles And Responsibilities | Not Included |
| 15 | Data Flow and Data Dictionary | Not Included |
| 16 | Data Requirements | Not Included |
| 17 | Application Navigation | Site Map |
| 18 | Screen Designs | Interface Design layout and the Final Product |
| 19 | Screen 1 - Main Menu | “   “ |
| 20 | Screen 2 - Add Internet Sites | “   “ |
| 21 | Screen 3 - Edit Internet Sites | “   “ |
| 22 | Screen 4 - Lookup Internet Sites | “   “ |
| 23 | Screen 5 - Search For Sites | “   “ |
| 24 | Report Designs | Not Included |
| 25 | Full Site Listing | Not Included |
| 26 | Current Site Details | Not Included |
| 27 | Current Search Details | Not Included |
| 28 | Onging Support | Not Included |
| 29 | Acceptance Testing | Not Included |
| 30 | Initial Support | Not Included |
| 31 | Extended Service Level Agreement | Not Included |

**Application - Design**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 |  |  |
| 2 | tblCustomers | Completed |
| 3 | tblLogin | Completed |
| 4 | tblOrderLines | Completed |
| 5 | tblOrders | Completed |
| 6 | tblPermissions | Completed |
| 7 | tblProducts | Completed |
| 8 | Screen Prototypes | Completed |
| 9 | Site Map | Completed |
| 10 | Interface Design | Completed |
| 11 | Introduction | Completed |
| 12 | Script | Completed |
| 13 | Requirements | Completed |
| 14 | Site Map | Completed |
| 15 | Version Control | Completed |
| 16 | Final Product Solution | Completed |
| 17 | Design patterns | Completed |
| 18 | Debugging Techniques used | Completed |
| 19 | Logging for .Net framework | Completed |

|  |  |  |
| --- | --- | --- |
| 20 | Test Data and test summary result | Completed |
| 21 | Test Plan | Completed |
| 22 | Interface Design layout and the Final Product | Completed |
| 23 | Sign off | Completed |
| 24 | References | Completed |

**Application - ERD**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Headings from Page 1 of: | 3b\_ERAndTables.doc |
| 2 | **Entities** | First step for creating entities, Completed |
| 3 | **Relationships** | Relationship listed out, Modality wasn’t included, Completed.  3.1, Links Completed  3.2 Cardinality Completed  3.3 Modality, NA  3.4 Labels Completed |
| 4 | **Attributes** | Primary Key’s and Foreign keys included & additional attribute, Completed  4.1 Identify Completed  4.2 Linking Completed  4.3 Descriptive Completed |
| 5 | **Lookup Lists** | Included |

**Application - ERD**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Headings from Page 1 of: | dbSparkly.doc |
| 2 | tblCustomers | Completed |
| 3 | tblLogin | Completed |
| 4 | tblOrderLines | Completed |
| 5 | tblOrders | Completed |
| 6 | tblPermissions | Completed |
| 7 | tblProducts | Completed |

# UID Check list

## User interface checklist - 2D Graphical Interface Databases

|  |  |  |
| --- | --- | --- |
|  | **Application Navigation** |  |
| 1.1 | An optimal number of menu and data entry forms used | Yes |
| 1.2 | Names on various menu form buttons match the titles on the forms & reports | Yes |
| 1.3 | 3 key clicks to move between most areas of the application | Yes |
| 1.4 | Optimal menu-button access for all forms and reports | Yes |
| 1.5 | Closely associated forms or reports are easily accessed from each other | Yes |
| 1.6 | An initial form or menu is presented when the database file is first loaded | Yes |
| 1.7 | Form modality has been applied appropriately | No |
|  |  |  |
|  | **Application Consistency** |  |
| 2.1 | Button & toolbar words & icons presented consistently throughout application | Yes |
| 2.2 | Button and toolbar ordering and relative positioning presented consistently | Yes |
| 2.3 | Drop down menu items and lists worded and positioned consistently | Yes |
| 2.4 | Buttons, toolbars & menus consistent with Microsoft standards where appropriate | Yes |
| 2.5 | Colours used consistently and/or effectively between the various forms | Yes |
|  |  |  |
|  | **Form Layout and Colour** |  |
|  | **Balance of Forms and Reports - Formal for Business applications** |  |
| 3.1 | Graphic elements have appropriate weight and density | Yes |
| 3.2 | Business style forms and reports - static or symmetrical | Yes |
| 3.3 | Page looks stable, components are positioned in an ordered/organised manner | Yes |
| 3.4 | Appropriate use of form buttons, toolbars and drop-down menus | Yes |
|  | **Proportion of Forms and Reports** |  |
| 3.5 | Individual components (fields & groups) are in proportion to the form/report | Yes |
| 3.6 | Individual components (fields & groups) are in proportion to each other | Yes |
| 3.7 | Individual components are appropriately grouped | Yes |
| 3.8 | Elements are presented in an order and position relative to their importance | Yes |
| 3.9 | Breakup of space, graphics, 3Dboxes, borders, lines, buttons, labels & fields | Yes |
|  | **Harmony of Forms and Reports** |  |
| 3.10 | All the elements of the form/report work together to promote one message | Yes |
| 3.11 | The form/report gives the impression of cohesiveness and unity | Yes |
| 3.12 | Elements complement each other and are appropriately positioned | Yes |
| 3.13 | The page flow is appropriate - one direction versus random | Yes |
|  | **Sequence on Forms and Reports** |  |
| 3.14 | Readers can scan the form or report easily in a Z format | Yes |
| 3.15 | The page flows from left to right and top to bottom | Yes |
|  | **Other Visual and Form/Report Considerations** |  |
| 3.16 | Acceptable business colours & textures (Company colours?) | Yes |
| 3.17 | Appropriate use of 3D and/or shadowing effects | NA |
| 3.18 | Forms/Reports display date & time, appropriate heading & company logo/name | Yes |
| 3.19 | Reports presents page numbers & what the report is sorted and/or grouped by | Yes |
| 3.20 | Report indicates the name of the application producing the report | Yes |
| 3.21 | Field and button prompts are clear and appropriate | Yes |
| 3.22 | Use of font styles easy to read and appropriate to business application | Yes |
| 3.23 | Application colours cater for grey-scale laptops and colour blindness | Yes |
| 3.24 | Application forms have been designed for a VGA screen | NA |

|  |  |  |
| --- | --- | --- |
|  | **Error Trapping, Messages & Parameters** |  |
| 4.1 | Inappropriate user input is effectively captured or prevented | Yes |
| 4.2 | Messages and responses to inappropriate user input are appropriate | Yes |
| 4.3 | Forms are designed so as to minimise possible inappropriate user input | Yes |
| 4.4 | Filter or parameter dialog boxes/forms are used effectively | Yes |
| 4.5 | Filter forms are accompanied by clear ‘business-friendly’ instructions and title | Yes |
| 4.6 | Filter forms are appropriately designed (see form layout above) | Yes |
|  |  |  |
|  | **Memory, Default and List Considerations** |  |
| 5.1 | Use of the application requires little or no use of short term memory | Yes |
| 5.2 | Combo and list boxes enhance the application’s ease of use | Yes |
| 5.3 | Regular use of application does not require reference to a manual or paper lists | Yes |
| 5.4 | Defaults values in fields have been used to enhance ease of use | Yes |
|  |  |  |
|  | **Application Help and About** |  |
| 6.1 | On-line help is available in an appropriate format | Offline |
| 6.2 | Help is context sensitive by pressing F1 | No |
| 6.3 | The level of help is appropriate to application users (eg: novice vs expert) | No |
| 6.4 | The language level is appropriate to the application users | Yes |
| 6.5 | The standard help items such as contents, search and index are available | No |
| 6.6 | All contents screen(s) contain 15 topics or less | Yes |
| 6.7 | The levels of sub-contents screens are limited to 3 or less | Yes |
| 6.8 | The contents topic hierarchy is well considered | Yes |
| 6.9 | The search keywords are well considered | Yes |
| 6.10 | The browse sequence of help pages is well considered | Yes |
| 6.11 | Appropriate fonts, sizes and layout considerations have been addressed | Yes |
| 6.12 | Colour use caters for grey-scale laptops and colour blindness | Yes |
| 6.13 | Colour has been applied effectively - enhances harmony | Yes |
| 6.14 | Help windows do not cause eye strain (no bright, faint or camouflaged text) | Yes |
| 6.15 | Help windows have been designed for VGA monitors | NA |
| 6.16 | An ‘About’ form has been included with version, date and author information | No |
|  |  |  |
|  | **Use of Colour** |  |
| 7.1 | Extremes of the spectrum (eg red & blue) are not used together | Yes |
| 7.2 | Forms used for long periods have natural soothing background colour, eg: grey | Yes |
| 7.3 | Strong, intense or complementary colours are used sparingly, mainly for accents | Yes |
| 7.4 | Red has only been used for errors, if at all | Yes |
| 7.5 | Colour has been used to identify similar functions | Yes |
| 7.6 | Related colours / saturations have been used effectively for grouping | Yes |
| 7.7 | Colour has been used effectively to augment (promote) tasks | No |
| 7.8 | Colour has been used with other redundant cues (eg: 3D effects, borders, icons) | No |
| 7.9 | Value (dark, medium, light) differences used to promote sharp edges as required | Yes |
| 7.10 | Cultural considerations (wrt colour) if required, have been taken into account | No |
| 7.11 | There is no confusing use of small items, items far apart, colours close in spectrum | Yes |
| 7.12 | Warm colours (which appear larger) have been effectively used | Yes |
| 7.13 | Colours are user configurable | No |
| 7.14 | Less than 4 colours have been used per screen / form | Yes |

# Risk Management

| **Risk ID** | **Risk Title** | **Risk Description** | **Likelihood** | **Impact** | **Risk Level** | **Consequence** | **Prevention/Mitigation** | **Contingency/Recovery** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| R1 | Misunderstood user requirements |  | 1 | 4 | 4 | Biggest consequence would be not talking to client during entire process clear up misunderstanding and then find out during testing phase that it nothing like client asked for. | Create prototypes, have checklist based on your understanding of client requirements and have Client sign off on your understanding of their user’s requirements. | Offer client required program for fee’s they already been charged for. No additional costs. |
| R2 | Scope Creep |  | 4 | 3 | 12 | Project has less likely chance being completed on time. Introduction of unplanned features that haven’t been authorized. Including tests & plans regarding features that introduce new bugs into system | Have sign off sheet for additional scope changes. Increase complexity of one feature would require changing scope of another, or increase time or cost of the project. | Advise client that any changes to Scope impacts, timeline, Cost & features of final product. |
| 3 | Missing Functionality |  | 2 | 3 | 6 | Delays project completion date with additional costs for development. | Have test plan including all features planned for project.  Once plan been validated based on user requirements & sign off from the user & project manager. | Additional resources would be needed to complete project on time or, within budget. |
| 4 | Poor time estimate |  | 3 | 3 | 9 | Expected due date for project would be pushed back unnecessarily and would result in more work arounds & stress complete features on time for poor time estimate. | Use previous experience with other projects and judge how long estimated time would be. If you not familiar with projects like one working on. Have initial estimate then double it. Client will be bit surprise of length expected estimate but you rather, longer estimate and shorter delivery than shorter estimate and unexpected longer delivery | Talk to client and establish timeline for project regarding bogging of the project. Get tires out of the mud, and estimate how long take get back onto road of successful delivery based on difficulty of downpour of rain. |
| 5 | Data loss & disasters IE, flooding, earthquakes, Fires. |  | 3 | 5 | 15 | Loss of critical development Data & program. Likely hood of disasters impeding development is low, data loss possible still. | Use backups of data & have data recovery & prevention plan. | Use recovery that been implemented. Adjust project delivery based on how much time was lost |
| 6 | Project requirement related changes during production |  | 4 | 3 | 12 | Increase of project scope, cost & timeline | Include scope change management, talk to client about new cost, and timeline requirements if new scope changes to be included. | Talk about scope change management, talk to client about new cost, and timeline requirements of the new scope changes. |
| 7 | Impact of defective product |  | 1 | 5 | 5 | Loss of revenue, burden on production company fix problems caused by faulty product. Loss of money for development team for fixing problems related too defective product. | Have product validation with client and have sign off sheet regarding products prime functionality for meeting client requirement. Any, additional functionality after validation is needed regarding program. It’s meet initial requirements of client thus isn’t faulty regarding increase scope. However if bugs and other issues still found, then its validation teams issue. | When Problem regarding faulty product been found, rectify this by taking time fix problem increasing time of development without any additional cost. Future projects would need more rigorous Validation documentation and project sign off regarding meeting client requirement. |
| 8 | Unclear requirements and scope |  | 2 | 5 | 10 | Complete utter destruction of the project. Additional cost to client regarding fees for developing additional features out of scope of the project, or unneeded costs regarding features that been developed that wasn’t needed. | Checklist, meetings regarding understanding of user requirements, prototypes of program using given requirements. And have sign off sheet for prototype go into further production. | Have meeting with client review requirements & scope.  Once reviewed issue another documentation regarding change in scope and include sign off on the product. |
| 9 | Development environment doesn’t emulate user environment closely |  | 4 | 4 | 16 | Incorrect development environment doesn’t emulate user computer environment. Or even incomplete or completely different platforms.  Program doesn’t work how it’s intended to work. | Have Development member overseeing deployment of the program to environment to fix any issues. Include testing regarding user’s computers and environment during testing phase.  Include acceptable environment deployment issues regarding various different systems. Example users bringing software to their home computer systems. Include list, of what supported and not supported. | Problems arriving during deployment and day one bugs.  Send out team developers to assess the problems and create tickets. Work through weekend to get program meeting requirements.  Accept loss of revenue. |
| 10 | Inadequate staff training regarding use of program |  | 3 | 2 | 6 | Staff would have hard time adapting to new program and project and become frustrated. Users would complain about introducing complicated program to already stressful work life. | Introduce day dedicated to training staff for program on day before deployment. Train staff on basics they required to do for their daily routine.  Have sign off sheet for everyone who completed training and who needs to additional training. | Introduce document or manual and offer training for use of software. |
| 11 | Lack of development staff |  | 1 | 5 | 5 | Increase of production time & increase timeline. Initial timeline given would be inaccurate. | Estimate total time of development based on current staff on hand. Or increase number of developers work on the project to meet scope time requirements at increase development costs. | Increase timeline for project be completed in, or increase costs to speed up production or reduce scope or hire more developers. |
| 12 | Inconsistent source code conventions or lack of one |  | 1 | 5 | 5 | Documentation issues, Unreadable source code. Increase difficulty in maintenance. | Impose consistent source code convention with simple rules that create consistent writing & naming of source code. | Refractor the code into new convention format while documenting the process |
| 13 | New technology for business |  | 2 | 5 | 10 | Introducing new technology can always be hard especially if business is not familiarly with the technology. Thus can cause problems with and even failure of project during deployment. | Include simulated environment for employees to use Technology. Access average employee understanding of technology and adapt training required for new system.  Include in-depth training for users. | Create training course for users understand basics of new system and how use it correctly to be productive. |
| 14 | Inadequate documentation |  | 1 | 5 | 5 | Increase difficulty during development, testing, deployment and maintenance of the project.  When hiring new staff to maintain undocumented system increase time it takes for new person to understand how project works. | Have standards regarding documentation, might include. Internal documentation, self-documentation, External documentation. | Create documentation going through steps create technical documentation for development. Create comments on source code and including possibility refactoring code to create more concise & readable source code. |
| 15 | Poor system scalability |  | 1 | 3 | 3 | During company growth system would poorly handle increase load. Slowing down of the system and reducing worker performance. | Discuss users expected requirements for the size of the company expected in near future. | Replan program with slight adjustments to accommodate requirement for increased scalability. |
| 16 | Changing hardware and uninformed changes or update in hardware during development |  | 4 | 5 | 20 | Potential risks regarding updating operating system. Major risk when changing from 32bit environment to 64bit. Program written might not even execute. | Plan and document any hardware changes regarding network, hardware of user pc or any other major rollouts regarding technology.  Plan and adjust program accordingly and have signoff sheet regarding supported hardware. | Document changes needed and have bug fixes for new hardware environment. |
| 17 | Poor professional relationship between project management and staff |  | 1 | 4 | 4 | Decrease in communication between management and staff. Decrease in developer productivity. | Have open communication between management and staff. Allow discussion regarding issues openly between staff and management. | Have discussion on how conflict can be resolved.  Take team building exercises. |
| 18 | Inadequate project management |  | 2 | 4 | 8 | Potential failure of the project. | Allow for increase discussion everyone involved in development process including stakeholders. | Increase number of hours dedicated to planning for the project. |
| 19 | Shuffling project staff |  | 2 | 5 | 10 | Removing key staff from development would require replacements and learning from new staff be up to date with project requirements, technology being used. Thus increasing time required for the project to finish. | Plan reshuffling of staff not remove key skills from project during development & implementation. If required, have staff train new staff on what is required for the project. | Have staff trained based on requirements of project and roles they occupy. Have good documentation that they can read & improve training of development staff. |
| 20 | Development team Unfamiliarity with project technology requirements |  | 1 | 5 | 5 | Increase risk of project failing. Given resources online & IT projects existing for long time now. Likely hood of this happening is lower. Impact however is just as high. | Learn about technologies relevant to project. Study how this could be applicable to current project. | Use Test Driven development and agile methodology. Release functionality in discreet chunks that meet requirements piece by piece. Once familiarity of technology is achieved, final program can be completed and pieces can be fit together create final product. |

***Note****: Risk Level = Likelihood x Impact, maximum is 25 Lowest is 1*

# Source Code Review

## Source Code Validation

**Application - Class Structure**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Class Breakdown | Having classes broken down into related functionality shared between classes would made work required for classes even less. However most functionality regarding connecting to database and manipulating data has been broken down appropriately. |
| 2 | Inheritance | Inheritance has been used appropriately but with its implementation inheritance wasn’t needed. |
| 3 | Associations | Associations are made clear by using public and private members including creation of additional child classes and functionality between classes & children included with protected functions to define where & when these functions should be used. |
| 4 | Constructors | Constructors allow for multiple ways for instantiating a class. Constructors used where for allowing easy creation and Transparency between a new Instance of class and one that’s been loaded from database. |
| 5 | Polymorphism | Not applicable polymorphism was not needed for implementation. |
| 6 | Interfaces | Interfaces currently wasn’t used. However, if I was do similar project again I would outline key related features like, creating & loading from database then have interface for them to relate key functions & classes together using these interfaces. It community functionality required for these classes. |

**Application - Coupling and Cohesion**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Methods | Methods used, communicate features & connection to database. Most Methods within classes have similar names if, two classes have similar functions like connecting to table, they would have Name of table, as class. And functions like, Update, Delete, Insert.  Similar methods always return same result as all others. Methods could be changed to also include using Interfaces to enforce this. |
| 2 | Parameter Passing | Parameter passing is used connect to tables via the adapter. Classes implementing adapter simple pass table as parameter to connect to table and retrieve its table.  Parameters has been used to limit amount repeated code regarding connecting to database and creating new instances of classes. |
| 3 | Functions | Functions have been used for simple UI methods regarding button events & creation of new forms. |
| 4 | Global Variables | Global Variables have been under-utilize Global variables have been used during development process but then weeded out when refactoring introduced. |
| 5 | Method Naming | Consistent naming convention applied? Each method name reflects the full and complete purpose of its method?  Methods naming followed C# standards from MSDN.  Variables as parameters had p represent they part of parameter and followed by its type example, pStrInput, represents parameter String Input  Methods named using Capital for each new word. |
| 6 | Method Portability | Method portability currently wouldn’t be portable as assumptions where made given input & output. Each method individually would be easily ported besides heavily coupled Data adapter all classes return instances and other Data Tables & rows. However methods themselves would easily be ported without heavily modifying as methods are parameter driven and have little dependencies outside the class.  Each method would have be repurposed for its environment and implementation for it be portable but methods themselves would require very little modification. If generic class was written handle basic connections & tables using parameters, would give up clarity for flexibility however would make method extremely portable as only dependency would be connection to the database. |

**Application - Logic**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Sequence | Appropriately applied? Sequences appropriately applied & follows logical cohesive pattern. |
| 2 | Selection (*If/Case*) | Functions that require a lot nested If’s have been reduce into smaller functions that return Booleans. These Booleans then used in place to control size of the Nested if statements. If statements been used extensively and when needed. Complexity of If statements are simple enough to understand from looking at names & data types within statement. |
| 3 | Iteration (*Loops: Where / For / etc*) | Loops are used for iterating through collection of objects. Loops used mostly for, data tables and data rows. Complexity of loops rather simplistic. |
| 4 | Method and Loop Exits | Methods commonly require exiting execution with return regarding success or failure of method and loops exit once row has been found or, looping isn’t needed anymore given an condition |
| 5 | Complexity of code segments | Complex code been reduce into modules & functions. Each function has comments for its purpose and has single purpose each work towards completing an objective. |

**Application - Naming Conventions**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Classes | Classes Named using Capitals, related objects represented using data would have associated name to properly represent them, and example Cup would be class named Cup. Class handling connection to database would be called, dbConnector.  Naming for modules or external classes within Project where named with lowercase as namespace, was in capital letters. |
| 2 | Objects | Objects used, with lower case & capital for every sequential word following its name. Example, exampleObject. Each object is named for its specific purpose. If object is passed as parameter it has little p represent that it’s of parameter object. |
| 3 | Variables | Variables named using First Three letters of the represent variable type, and Capital letter for each consecutive word. |
| 4 | Methods | Methods & variables followed with capital letter for each word, and self-documenting. Each name for method should describe its purpose with method name & if it returns anything.  Methods where used Considerably to limit number of times reusable code was re-written. |
| 5 | Parameters | Parameters used had names p, Followed by appropriate naming convention for the parameter. |
| 6 | Counters | Counters usually written with num followed by Capital for first letter in each word. Counters are assumed be Integers so no Int is required, however if any other datatype is used, it must follow with, numStrExample.   It’s possible to want store number of examples using string. For example user input & then later convert it into Integer later for use. |

**Application – General Presentation / Functionality**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Code formatting | Formatting, was done manually with also inbuilt functionality. Formatting is used consistently unless short hand single line If statements used. All, indentations are used represent what block of code belongs to where. To allow easier reading of code. |
| 2 | Internal Documentation | Most of the internal documentation is handled by self-documentation only complex algorithms & functions require additional documentation |
| 3 | Complexity of code segments | Any logic that includes multiple steps to complete task, would split up functionality into compartmentalized functions independent from requirements of Higher logic. These functions can be used achieve simple goal, and used elsewhere if needed.  Breaking down high complexity problems into smaller problems means, overall readability of program would increase. |
| 4 | Set and Get Methods | Set & Get methods are used restrict use of private & public variables to be used in specific way. Set and Get gives control to developer for the class to control Type, values & validate input for these get and sets. Default values are used for null, invalid or incorrect parameters. |
| 5 | Use of brackets / curly braces | Each curly braces are placed on separate new line.  Everything within curly braces are tabbed represent where they belong. |

**Data Access Layer (DAL)**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Data Access Layer | Data Access layer has been applied, using class handle connection between user interface & Database. Functions used replace complex sql statements & functionality into reusable function that accepts parameters & data to read, write, update & delete data from a table. |

**Database – Table Structure**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Entity-Relationship structure | ER diagram, entities are correctly identified, Consistent naming for tables, queries. Relationships of tables considered. |
| 2 | Complexity | Tables represented in simple tables & connections. |
| 3 | Table Naming | Naming of tables represent data they represent. |
| 4 | Lookup and Link tables | Naming of lookup List, are named specifically for each lookup list. Using DB access wizard. |
| 5 | Table size | Number of fields for tables are reduce with connections of relevant information that can be reused for other tables. Table size sometimes cannot be reduce and simple split into smaller tables. |

**Database – Fields and Properties**

|  |  |  |
| --- | --- | --- |
| **#** | **Item** | **Status/Comments** |
| 1 | Field Names | Names for fields have data Type within the name. Most fields named appropriately relating to data they represent. |
| 2 | Data Types | Data Types with similar Fields, will have the same data type. Date & time, will both use Date Time. However these two fields could be reduce into Date Time. |
| 3 | Primary Keys | Primary keys use auto Increment number, composite Primary keys only exception. |
| 4 | Foreign Keys | Foreign keys use same name as following field on other table. All, keys work same as Primary keys naming. |
| 5 | Field Properties | Field sizes & types are created to use maximum possible size they expected to be. Without increasing size of database for each entry. Input masks have been used validate when needed. For example mobile numbers, restriction on size of phone numbers been used so largest possible phone number couldn’t be exceeded. |
| 6 | Indexes | Never used. |

# Agenda

**AGENDA**

**Mr Sparkly requirements meeting**

**October 22, 2015**

**9:00AM – 10:00AM**

Meeting called by Mark O’reilly

|  |  |
| --- | --- |
| **Attendees:** | Jade Gorton, Mark O’reilly, Michael Heffernan |
| **Please read:** | NA |
| **Please bring:** | NA |

|  |  |  |
| --- | --- | --- |
| **9:00 – 9:05** | **Introduction**  Continental Breakfast  Introduction of Client M O’reilly | C2.22 |
| **9:05 – 9:10** | **Gather Client requirements**  Discussion of Client requirements Jade Gorton, Michael Heffernan | C2.22 |
| **9:10 – 9:15** | **Design Review**  Client approval of Design Jade Gorton, Michael Heffernan, M O’reilly |  |
| **9:15 – 9:20** | **Overview of Risks**  Risk Management Michael Heffernan,  M O’reilly |  |
| **9:20 – 9:25** | **Source Code Review**  Risk Management Jade Gorton, Michael Heffernan |  |
| **9:25 – 9:30** | **Technical Documentation**  Informing client Documentation Jade Gorton, Michael Heffernan |  |
| **9:25 – 9:30** | **Validating Test Plan**  Discussion if test meet requirement Jade Gorton,  M O’reilly |  |
| **9:30 – 10:00** | **Finalize meeting**  Discussion of development & costs Michael Heffernan | C2.22 |

**Additional Instructions:**

Cblock Level Two Room 22.

# Test Plan

## Testing Validation

**Test Plan**

|  |  |  |
| --- | --- | --- |
| **Product** | **Item** | **Status/Comments** |
| 1 | Headings from: | JadesChocoAsptestPlan.dotx |
| 2 | Overview | Completed |
| 3 | Purpose | Completed |
| 4 | Assumptions | Completed |
| 5 | Scope | Completed |
| 6 | Testing summary | Completed |
| 7 | Test Strategy | Completed |
| 8 | Test Responsibilities | Completed |
| 9 | Test Driven Development | Completed |
| 10 | Environment requirements | Completed |
| 11 | Data requirements | Completed |
| 12 | Resources and Skills requirements | Completed |
| 13 | Testing tools | Completed |
| 14 | Test Cases | Completed |
| 15 | Unit testing | Completed |
| 16 | Test method | Completed |
| 17 | Test class | Completed |
| 18 | Test class – Test method | Completed |
| 19 | Assertions | Completed |
| 20 | Test Summary report | Completed |
| 21 | Sign off | Completed |

**Test Cases**

|  |  |  |
| --- | --- | --- |
| **Product** | **Item** | **Status/Comments** |
| 1 | Headings from: | 5c\_Software Test Plan.dotx – Test Cases Section |
| 2 | Customer webpage Insert Customer | Completed |
| 3 | Customer webpage Edit Customer | Completed |
| 4 | Customer webpage Delete Customer | Completed |
| 5 | Product webpage Insert Product | Completed |
| 6 | Product webpage Edit Product | Completed |
| 7 | Product webpage Delete Product | Completed |
| 8 | Orders webpage Insert Orders | Completed |
| 9 | Orders webpage Edit Product | Completed |
| 10 | Orders webpage Delete Product | Completed |
| 11 | Orders webpage Edit Orders | Completed |
| 12 | Orders webpage Select Order Line | Completed |
| 13 | Orders webpage Delete Orders | Completed |
| 14 | Order Line webpage Select Order Line | Completed |
| 15 | Order Line webpage Insert Order Line Row | Completed |
| 16 | Order Line webpage Edit Order Line Row | Completed |
| 17 | Order Line webpage Delete Order Line Row | Completed |
| 18 | Test Script Crypto Password Invalid | Completed |
| 19 | Test Script Crypto Password Valid | Completed |

# Software Test Plan & Technical Documentation

Provided files, TestCases\_Jade\_Gorton\_7100288111.docx, JadesChocoAspTestPlan.docx, 7100288111\_ICAPRG501A\_Technical\_documentation.docx.