IP3 Project Group Report Starter Template

|  |  |
| --- | --- |
| Group Number: |  |
| Project Title: |  |
| Project Style: | Development |
| Group Member Names: |  |
| Date: |  |
|  |  |
|  |  |
|  |  |

[Layout style is your choice]

**Executive Summary**

This report describes the process of creating a prototype software application, undertaken by Pretty-Good-Software Ltd (PGSL) for Vision Ltd[[1]](#footnote-1) in response to the first phase of the company’s tendering process for a new software product. The application is a ‘Health and Safety Documentation System’ to support Vision’s quality assurance team in originating, versioning, storing, distributing and archiving health and safety guidance documents in a range of electronic formats, for use by their staff members. The tender requires software vendors to create a defined subset of the complete system as a prototype demonstrator.

The prototype has been designed and built over a period of ten weeks and meets all the core requirements listed in the tendering specification document. In addition to these requirements being met, PGSL has integrated into the prototype of its ‘Live Reporter’ system that allows a user to easily highlight an inconsistency/problem at a specific location within a document and have this communicated quickly to the quality assurance team.

The software is written as a web application using Microsoft ASP.NET MVC, Vision’s preferred implementation environment. It has been designed to be responsive and has been tested on a range of device form-factors including phones, tablets and larger screen devices.

The report describes the design, system architecture and implementation of the prototype and also describes how the system could be integrated with existing products currently in use at Vision Ltd.

1 Project Introduction 4

1.1 The Client Company – Vision Ltd. 4

1.2 Project Overview 5

2 Project Organisation 6

2.1 Lifecycle Utilised For The Project 6

2.2 Organisation of the group 6

2.3 Allocation of Roles (including definition of roles) 6

3 System Design and Implementation 7

3.1 Architectural Description 7

3.2 Implementation Environment 7

3.3 Design and Implementation 7

3.4 Review of Final Deliverable 7

4 Hardware & Software Requirements 7

5 Conclusion 8

6 Appendix 1 – Product Screenshots 8

7 Appendix 2 – Tender Specification 8

8 Tender Specification Detailed Requirements with Acceptance Tests 9

8.1 Manage Reports 9

8.2 Raise Report 12

8.3 User Management 13

9 References 15

**Tables and Diagrams**

Table 1 System Roles 5

# Project Introduction

[Note: The quality and clarity of the *Introduction* is important. For any document, you should always have a clear introduction that relates the **purpose** of the document (in this case it is a final report that describes the lifecycle of the work) and the **subject** of the document (the product/design that is to be created).]

This report describes the process of creating a prototype software application for Vision Ltd in response to the first phase of the company’s tendering process for a new software product. The application is a ‘Health and Safety Documentation System’ to support Vision’s quality assurance team in originating, versioning, storing, distributing and archiving health and safety guidance documents, in a range of electronic formats for use by their staff members. The introductory section of the report describes the client company and an outline of the business requirements for the new software product that they are commissioning. Subsequent sections cover the design, implementation, test and evaluation of the prototype solution.

## The Client Company – Vision Ltd.

Vision Ltd was founded in 2007 by two partners, Mary Baker and Kurt Crook. Ms. Baker had previous experience of developing quality control processes within previous positions at British Airways and Honest Joe’s Bagel Bakery. Mr Crook had twenty years experience in optical manufacturing within a range of multinational companies such as Olympus and Luxottica as well as being an independent consultant. Vision was set up to manufacture a range of sunglasses that could rival the quality and style of well-known brands but at a more affordable price. The company owns seven patents in key areas of design and manufacture of optical systems [1]

The company set up its headquarters in Reading in 2008 with an initial staff of twelve with manufacturing outsourced to China. <MORE INTERESTING STUFF HERE> Today the company has fifty five staff members and in addition to sunglasses has expanded to manufacture <STUFF>. The company is still headquartered in Reading but now has sales offices in Beijing and New York. Manufacturing is now undertaken in India as well as China. Vision has established a solid and diverse international reseller base that includes premier retailers such as Harrods in London, Macy’s in New York and Lidl in Glasgow; it also sells directly through a strong online presence <ETC>

Due to a revision of its internal processes and policies, Vision requires the creation of a custom document management system to manage its health and safety documentation; the company has issued a tendering document that includes a set of business requirements for the system and have asked interested software vendors to respond to this by creating a prototype demonstrator that covers a defined subset of the complete system. PGSL has responded to the tender and this document describes the prototype that has been created to meet the business requirements.

## Project Overview

[Provide a high-level **overview** of the business objectives for the software product

[The project specification is already provided so please include it as an appendix].

This section provides a high-level overview of the business requirements for the system. A detailed system specification from Vision is shown in Appendix 2 – Tender Specification.

The system should allow staff to access health and safety documentation on a range of computing devices (ranging from phone devices to full-screen computing devices). In order to cover the widest set of devices without having individual software builds for different platforms, it has already been decided that the application should be web-based, utilising responsive design techniques.

The system must allow a range of existing document types to be made available to users and provide an interface that provides the following functionality:

* Importing of documents (with pre-defined document types) into the system
* Versioning of documents
* Reviewing of documents before they are released for use.
* Viewing of documents utilising integral viewer technology
* Viewing lists of current documents categorised in the following ways:
  + By Date
  + By Author
  + By Health and Safety category
  + By Issuing-Department
* Viewing older version of documents
* Deleting documents
  + A ‘trash’ subsystem must be implemented that allows retrieval of deleted documents
* Reporting of issues with documents that need resolved.
* **etc…**

The system will include provision for the *roles* listed in Table 1.

Table 1 System Roles

|  |  |
| --- | --- |
| **Role** | **Responsibilities** |
| Administrator | Creating users and allocating roles  Deleting Users |
| Author | Can originate documents (‘owns’ these documents)  Has read/update/delete rights to ‘owned’ documents |
| Reviewer | Reviews and approves documents before they are released |
| Reader | Has read-only access to released documents |

An administrator can register *users* so that they can access the system (users are named individuals) and allocate one-or-more roles to each user. For example, Mary can be registered and can be both an Author and Reader.

**<etc>**

# Project Organisation

In this section the chosen lifecycle for this project will be explained in relation to the characteristic of the system. The organisation of the group shall be explained along with the roles to be played by each member of the group. Reporting dependencies will also be detailed in this section. Always justify your selection.

## Lifecycle Utilised For The Project

As a team, we looked at several Agile Methodologies a list of these can be seen below.

* Object Oriented Agile (Generic Approach)
* SCRUM
* Kanban
* Spiral

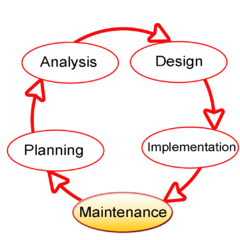
2.1.1 Object Oriented Agile (Generic Approach)

Object Oriented was first used before the mid 1990’s back then there was many approaches labelled as “object oriented”, however in 1994 Grady Booch, James Rumbough and Ivar Jacobson started working on UML (Unified Modelling Language) They successfully merged not just their on methodologies but also those of Philipe Kruchten and Walker Royce some of these old methodologies where known as OMT, OOSE and Booch Method along with insights from many other industry leaders created RUP (Rational Unified Process) which is a comparative incremental system. Since its creation, RUP has been the most recognised and used Object Oriented Methodology (En.wikipedia.org, 2017).

In the Object Oriented Methodology, the software life cycle is split up into parts known as Requirements, Design, Implementation, Verification and Maintenance. During development, there will be a lot of iteration between these steps (Anon, 2017).

The main advantages of this are Development is done faster meaning clients have a finished product faster, previous work tends to be used a lot as to help the iterative nature meaning if something has been coded by the team before don’t code it again reuse and adapt the code to suit, there is an increase in quality as unlike waterfall where they wait till the end before going back to the start to do changes object oriented is an agile methodology that makes use of iteration to perfect the system for end user from day one and object oriented tends to be very good for client server scenarios (Westga.edu, 2017) like the document management system we are producing for ideagen.

However, object oriented also has disadvantages such as: there is a steep learning curve for OOD which some developers might not be able to step up to, Object Oriented programs tend to have a larger size which may be a problem for both end users and developers and Object Oriented programs tend to run slower which may be a problem if the client does not have the suitable hardware to run it (Anon2, 2017).



This diagram clearly shows the return to previous steps done through iteration until the program is ready for launch (It’s a loop that keeps going around till the program is suitable for the client).

The reason our group considered this approach is because it is the approach we have the most experience dealing with seeing as we all came from colleges where we never truly taught how to use any of the other approaches.

2.1.2 SCRUM

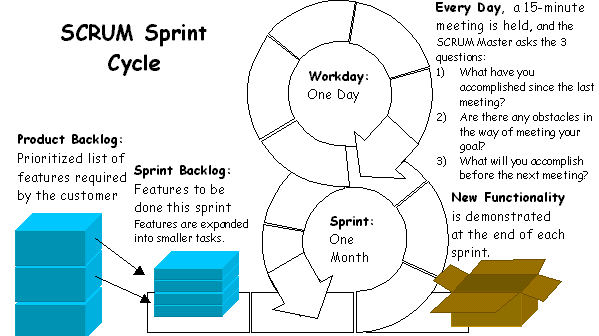
SCRUM was developed in 1986 by Hirotaka Takeuchi and Ikujiro Nonaka and is defined as being a flexible, team based and common goal driven development. The term is also used, in the sport, rugby and is a link between the strong teamwork needed within the game and the teamwork needed when making the software.

SCRUM has many principles and is an in-depth methodology, some of these principles are displayed below.

* One of the main principles of SCRUM is to produce a market ready product as fast as possible. This is achieved through meeting every day and ironing out any problems, sometimes this marketable product does not even need to be fully complete. (ALLIANCE®, 2016)
* It is very flexible meaning it can accommodate the changing ideas of the user between each of the “sprints”. (ALLIANCE®, 2016)
* In this methodology, the most important skill is teamwork. Without it the project will fail. (ALLIANCE®, 2016)

The main phases of SCRUM are Initiate, Plan and Estimate, Implement, Review and Retrospect and Release (also known as Planning, Sprint Cycle, and Closure) (reserved, 2009).

In SCRUM, sprints are where the actual development takes place. At the end of each sprint, the program will be a fully working product. Sprints are breaking down into the following areas: Product Backlog, Sprint Backlog, Every Day, and New Functionality. A diagram of a sprint cycle can be seen below



Mark C. Layton 2015 (SCRUM for Dummies) as seen on Dummies.com (Layton, 2015)

SCRUM, just like any other methodology, has several advantages and disadvantages. Some of these advantages are; SCRUM permits changes to requirements at any point during development, promotes teamwork and working towards common goals, allows for the product to be working before the final iteration as well as providing the client with a functioning product early on. However, it also has several disadvantages as it is easy to become distracted and go off course. SCRUM brings a high work load, requires the entire team to be working well together, any lack of communication will ultimately lead to a project failure, constant testing can become annoying and Scope creep is common.

Our team considered this methodology for a few reasons; We all knew what SCRUM was through touching on it in college and improving our knowledge through ITPM and we all agreed that it seemed like it could work us if performed correctly.

2.1.3 Kanban

Kanban was developed in 1953 by Taiichi Ohno who was an engineer at Toyota it was created to sustain a good level of production (Atlassian, 2016).

Kanban was created as a logistics control system in Kanban production is dictated by the demand of the user (Atlassian, 2016). E.g. when supply time is lengthy the task should be completed fast.

Kanban does exactly what it does for industry teams for software teams by matching the work in progress to the whole team’s capacity (Atlassian, 2016). This gives more flexible planning, faster output, and clearer focus.

The main phases of Kanban are; Visualise and Workflow, Lead using a Team Approach, learn, and Improve and Continuously (Kanban, 2016). Kanban also has several advantages and disadvantages. Some of these advantages are; reduced production cost, enhancement in the quality of the products as well as offering a quick response to customer needs (Posted and Gulve, 2008). However, Kanban also has several disadvantages including; if material from the supplier is delayed the whole production line can be stopped and that the methodology is not exactly meant for software development (Posted and Gulve, 2008).

Kanban was ultimately just suggestion and we never actually planned how we would use it to complete this project the main reason for this was our knowledge of the methodology lacked a lot. However, it was one of the methodologies we considered.

2.1.4 Spiral

The Spiral model was first described in 1986 by Barry Boehm; Boehm describes the spiral model as a “Process model generator” where choices based on the risk of the project create appropriate process models for the project. Therefore, several models including incremental waterfall are special cases of the spiral model that fit the risk pattern of the project.

The spiral model has 4 phases these are known as planning, risk analysis, development, and evaluation.

* **Planning –** This theoretical design of the project. (*SDLC - spiral model*, 2016)
* **Risk Analysis –** Requirements are studied to identify the potential risks, once this is done a risk mitigation strategy is put in place and finally documents to accompany this are drafted (*SDLC - spiral model*, 2016)
* **Development –** Refers to the production of the actual software product through every spiral (*SDLC - spiral model*, 2016)
* **Evaluation –** Customers evaluate the software and provide feedback. (*SDLC - spiral model*, 2016)

The spiral model has several advantages such as: Due to the risk analysis, there is a high avoidance of risk, good for large projects; software is produced early in the life cycle (*SDLC - spiral model*, 2016). However spiral model also has several disadvantages such as it can be costly requires specific expertise and doesn’t work for smaller projects. (*SDLC - spiral model*, 2016)

2.1.5 Decision

Our group ultimately decided to choose SCRUM as our methodology approach for this project, although it was not the approach we had the most knowledge in we truly felt as the requirements of the project became more clear and we reviewed our organization SCRUM just suited both of them for a number reasons such as: daily scrum meetings will also benefit or company a lot as we are not a large company so everyone knowing problems as they arise and being reminded of what they must do on a day to day basis will help with the productivity of the project furthermore the team working enforced by scrum is seen by our company as a great positive. Also by separating Planning, Design, Implementation, Testing, Documentation, and Maintenance up into easier to handle sprints leads to less stress for our team as both workloads and learning curves are lower at each point of project. Also, due to this we can hypothetically present the project to Ideagen in 3 sprints: sprint 1 being a barebones system with users and roles implemented, sprint 2 being a complex system with mostly everything apart from admin user account changes and “bonuses” being ready and sprint 3 being all of the bonus stuff and the admin user account changes being added to the system, by presenting the project to Ideagen in sprints it means we can hypothetically generate profit faster (payed at the end of each sprint). We rejected Kanban and Spiral for the simple reason of not having enough knowledge of using either methodology so although we respect they are professional methodologies we would rather stay clear of them so we don’t mess up the project.

However, by week 4 the use of SCRUM had become lax, daily meetings stopped happening and we ultimately started using a more generic agile approach, the only real SCRUM feature we kept was sprints which can be seen in the appendices.

## Organisation of the group

We have decided to use the “Project” organizational structure. We feel this structure is the most tailored to our organization as the project manager has the highest level of control and the project isn’t large enough to warrant use of the “Matrix” structure. We also feel the project structure is best suited as it clearly shows the reporting structures of the organization meaning project members can see the chain of command and know who to report problems to easier. Although we decided to impose a respected organizational hierarchy on the group we were relaxed on whether the chain of command was followed always, we also decided to give each user two roles so that we didn’t fall into the problem of a member hiding behind the “that isn’t my role though” façade when problems arised.

## Allocation of Roles (including definition of roles)

2.3.1 Project Manager

The Project Manager is ultimately responsible for the project, the employee in this role must ensure that the project deliverables are met, the project finishes within time and hypothetically budget, stakeholders are properly involved when needed. In this projects the project manager will also provide moral support to team members as a demotivated workforce is not good for project productivity.

2.3.2 Senior Developer/Developer

Developers are responsible for the implementation of the system they will also be involved in the design stage with deciding how to best implement the database. This team is beneficial to the project’s success as such they will be led by a senior developer this person (Owen Cannon) has been selected as he is the member of our team with the most knowledge using the chosen implementation environment.

2.3.3 Senior Designer/Designer

Designers are responsible for the look and style of the system the team will be led by \*insert name here\* who will function as the senior designer and will turn the bland completed system into a styled system which fits the Ideagen brand.

2.3.4 Head Tester/Tester

Testers are responsible for rigorously testing the system as it is implemented. this will include both the backend systems and front end design/code. All tests will be documented and supervised by our head tester Gary Barclay.

# System Design and Implementation

## Architectural Description

## Implementation Environment

List the implementation technologies that are used and justify why these have been chosen for your team. If you investigated technology alternatives before finally selecting then you can discuss this here as well. Always justify your selection.

## Design and Implementation

List the deliverables/main activities for each iteration of the project lifecycle

For **each** part of your system do the following:

* Describe the design approach (and alternatives)
* Describe the implementation
* Describe what Software **verification** and **validation** [5] activities you have undertaken

Describe any **evaluation** carried out – this relates to the overall product

Use diagrams to help describe concepts that are difficult to describe verbally. Use appendices for any detailed content.

## Review of Final Deliverable

Provide a review of the final deliverable product and your view of whether it meets the requirements.

Please refer to the completed acceptance test results in Section 8 Tender Specification Detailed Requirements with Acceptance Tests

# Hardware & Software Requirements

Explain the hardware and software used, with three subsections

1. **Development** hardware and software.
2. **Deployment** of the system
3. **End-user** hardware and software requirements

# Conclusion

# Appendix 1 – Product Screenshots

# Appendix 2 – Tender Specification

[Please include a copy of the original specification document]

# Tender Specification Detailed Requirements with Acceptance Tests

## Manage Reports

**[Note: this is just an example. You can provide a similar table based on your specification]**

As the report manager I want to manage my report forms so that I can manage and resolve issues reported within the company.

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Create a report form template | As a report manager I want to be able to create a reporting form template so that users can use it to submit issues identified to be investigated and closed  *Additional Information:*  *The intent is to be able to reproduce paper based forms that they may have had within the business and represent these in an electronic format.* | |  |  | | --- | --- | | * Create and design a form template | Part | | * Form template name | No | | * Add fields to form template | Yes | | * + Free text fields |  | | * + Date fields |  | | * + Person fields |  | | * + [optional] dropdown |  | | * + [optional] images |  | | * + [optional] checkbox group |  | | * + [optional] radio group |  | | * Fields added can be labelled |  | | * Default fields on report template will be: |  | | * + Report owner |  | | * + Report raiser (auto filled with name of logged on user) |  | | * + Submitted date (auto filled when submitted with today’s date) |  | | * + Details (a free text field) |  | | * [optional] Fields can be defined as mandatory (must have a value before it can be submitted) |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| View a report | As a report manager I want to be able to view reports submitted allowing me to easily manage them | |  |  | | --- | --- | | * View filtered and complete list of reports |  | | * [optional] Sort the list of reports |  | | * [optional] Be able to search for a report by name |  | | * View a report record |  | | * Can view who submitted the report |  | | * Can view when date and time when report was submitted |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Edit a report | As a report manager I want to be able to edit reports submitted so that I can investigate and update them | |  |  | | --- | --- | | * Assign a report owner |  | | * Can edit fields on report |  | | * Cannot change report submitted date |  | | * Cannot change name of person who submitted the report |  | | * Report owner can manually change the status to “Investigation” |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Close a report | As a report manager I require to be able to close submitted reports so that I can identify reports that are complete | |  |  | | --- | --- | | * [optional] Notify report raiser when the report is closed |  | | * Status of report is set to “Closed” |  | | * Insert a close date |  | | * Record is set to read only. Report Managers can reopen the report record to edit which will set the status to “Open” |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Delete a report | As a report manager I want to be able to delete reports so that I can remove reports entered in error preventing the system from being cluttered | |  |  | | --- | --- | | * Report is removed from the system |  | | * Confirmation dialog will be presented to the user to ensure that they want to delete |  | | * Ability to delete will be permission controlled |  | | |

## Raise Report

As a reporter I want to be able to create and submit reports of issues identified within the company.

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Create a report | As a reporter I require to record details of the event to the report record before submitting | |  |  | | --- | --- | | * Creates a new report record with the status of “Draft” |  | | * Report raiser name is entered on the record |  | | * Can record text details of issue |  | | * Date issue occurred |  | | * Can record the severity of the issue |  | | * Can add attachments to the report |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Submit a report | As a reporter I want to be able to submit a report to the report managers for investigation | |  |  | | --- | --- | | * [optional] Notify the report manager(s) that the report has been submitted |  | | * Assigns a unique identifier to the report |  | | * Once submitted the report status is set to “Open” |  | | |

## User Management

As the reporting management system administrator I require to be able to create user and manage user accounts and assign them permissions so that I can control who can perform what actions.

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Create user | As a system administrator I require to be able to add users to the system | |  |  | | --- | --- | | * Can create a new user account: |  | | * + Add Forename (Mandatory) |  | | * + Add Surname (Mandatory) |  | | * + Add Email address (Unique, Mandatory) |  | | * + Add to a role(s) |  | | * Username (Unique, Mandatory) |  | | * Password |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Edit user | As a system administrator I require to be able to edit users with the system | |  |  | | --- | --- | | * Can edit details of user account |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Archive user | As a system administrator I require to be able to archive people who are no longer users of the system | |  |  | | --- | --- | | * Can archive user account |  | | * User cannot be archived where they are named on a report that is not “Closed” |  | | * [optional] Where user name appears on reports created the name will be highlighted as archived |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Create role | As a system administrator I want to assign security to a role so that members of this role will inherit the security permissions | |  |  | | --- | --- | | * Can create a role |  | | * + Role name |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Edit role | As a system administrator I want to be able to edit security applied to a role so that I can manage and update security permissions applied to the role | |  |  | | --- | --- | | * Can edit role name |  | | * Can assign/remove permissions to the role: |  | | * + Create report form template |  | | * + Create and submit a report record |  | | * + Delete report record |  | | * + Edit report record details |  | | * + Add attachment |  | | * + Remove attachment |  | | * + Close report |  | | * + Change status of report |  | | * + Reopen report |  | | * Can add users to the role |  | | * Can remove users from the role |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **User Story** | **Acceptance** | **Tested and working (Yes/No/Part)** |
| Delete role | As a system administrator I want to delete a role allowing me to be able to remove roles no longer required | |  |  | | --- | --- | | * Can delete role. Role can only be deleted where no users are assigned to it |  | | |

# References

En.wikipedia.org. (2017). *Object-oriented analysis and design*. [online] Available at: https://en.wikipedia.org/wiki/Object-oriented\_analysis\_and\_design [Accessed 26 Mar. 2017].

Westga.edu. (2017). *Object-Oriented Development*. [online] Available at: http://www.westga.edu/~bquest/1997/object.html [Accessed 26 Mar. 2017].

Anon, (2017). [online] Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.94.1474&rep=rep1&type=pdf [Accessed 26 Mar. 2017].

Anon2, (2017). [online] Available at: http://www.saylor.org/site/wp-content/uploads/2013/02/CS101-2.1.2-AdvantagesDisadvantagesOfOOP-FINAL.pdf [Accessed 26 Mar. 2017].

ALLIANCE®, S. (2016) *Scrum principles*. Available at: https://www.scrumalliance.org/why-scrum/core-scrum-values-roles (Accessed: 27 March 2016).

Layton, M.C. (2015) *The function of the Scrum and sprint within an agile project*. Available at: http://www.dummies.com/how-to/content/the-function-of-the-scrum-and-sprint-within-an-agi.html (Accessed: 26 October 2016).

reserved, A. rights (2009) *Scrum phases overview*. Available at: http://scrummethodology-org.etltools.net/scrum-phases.html (Accessed: 27 October 2016).

Atlassian (2016) *The agile coach*. Available at: https://www.atlassian.com/agile/kanban (Accessed: 31 October 2016).

Kanban, E. (2016) *What is Kanban?* Available at: http://www.everydaykanban.com/what-is-kanban/ (Accessed: 30 October 2016).

Posted and Gulve, De. (2008) *Advantages and disadvantages of Kanban*. Available at: http://functionalguy.blogspot.co.uk/2007/04/advantages-and-disadvantages-of-kanban.html#axzz4Ql0mhFiw (Accessed: 28 October 2016).

*SDLC - spiral model* (2016) Available at: http://www.tutorialspoint.com/sdlc/sdlc\_spiral\_model.htm (Accessed: 31 October 2016).

<MORE APPENDICES>

…

1. Vision Ltd: [www.vision.com](http://www.vision.com) [↑](#footnote-ref-1)