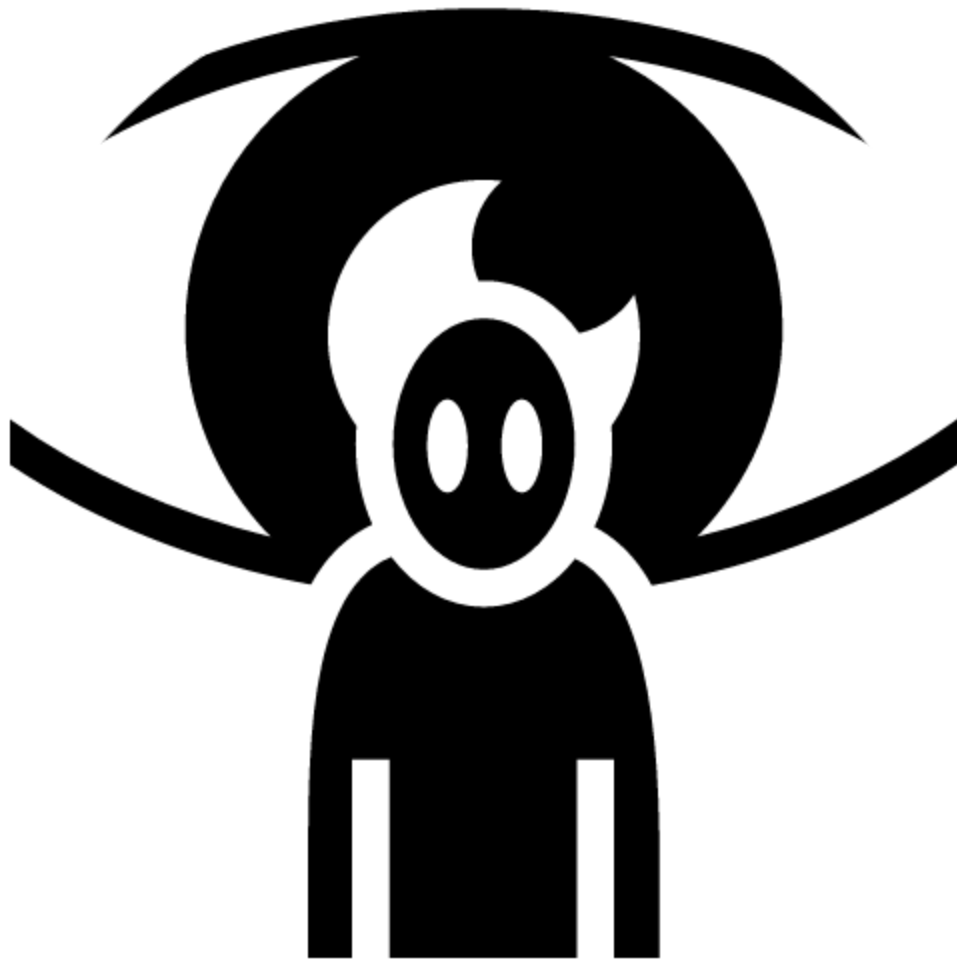

Project Title: eyeRS

08/02/2017 - 20/10/2017



Group number	5
Group name	eyeRS Development Team
Members:	Student number: M4DDK8SQ1 Name: Nathan Surname: Shava
	Student number: XQ9X3WV31 Name: Matthew Surname: Van der Bijl
	Student number: MB2015-0785 Name: Emilde Surname: Arsenio
	Student number: Z46WWQH76 Name: Andrea Surname: Cloete
	Student number: MB2015-0219 Name: Sajjaad Surname: Ishmail
Customer	Name: Ndai Mapaso Company: CTI Education Group Industry: Information Technology

Table of Contents

Methodology	3
1. User Requirements Document	5
Introduction	5
1.1 Purpose	5
1.2 Scope	6
Benefits	8
Goals	8
1.3 Definitions, Acronyms and Abbreviations	8
1.4 Overview	13
2. System Requirements	14
2.1 System Perspective	14
2.2 Functional Requirements	15
2.2.1 Context Diagram (high-level DFD)	17
2.2.2 Entity Relationship Diagram (ERD)	18
2.3 Non-functional Requirements	20
Human-Computer Interactions (User Interfaces)	20
Security Requirements	20
Communications Interfaces	20
Hardware & Software Interfaces	21
Screen resolution and form factor considerations	21
CPU & Memory Characteristics	22
Network Condition Scenarios	23
Battery Usage Considerations	24
Sensor Attributes	25
Media Functionalities	25
Support for various versions of Android OS	26
Interrupts, notifications and multitasking	27
2.4 Technical Requirements	28
2.5 User Characteristics	29
2.6 Operational Environment	30
3. Customer sign-off	31
Bibliography	32



Methodology

The chosen methodology for the system development is the waterfall model as presented by Stair and Reynolds (2008). This model is used to develop a system in a sequential and linear way. The waterfall model allows development to be completed in a steady and downward fashion. A steady and downwards fashion means that each phase of the model is completed before moving to the next phase. The end result of each phase serves as input for the phase that follows. The phases of the waterfall model (Stair & Reynolds, 2008) that will be discussed are:

1. Investigation;
2. Analysis;
3. Design;
4. Construction;
5. Integration and testing and
6. Implementation.

During the investigation phase developers gather all possible information on the requirements that the product requires. The second phase is the analysis phase. During this phase allow developers gain a better understanding of the functions, problems and opportunities of the system being analysed (Rob & Coronel, 2004).

System design is the third phase. During this the design of the system is completed and structure. This includes technical specifications and requirements (Rob & Coronel, 2004).

The fourth phase is the construction phase. During this phase the identified requirements are implemented within the chosen system. The construction phase includes turning abstract ideas into a reality by coding the ideas (Rob & Coronel, 2004).

The fifth last phase is the phase of integration and testing. During this phase the units from the previous phases are integrated (Rob & Coronel 2004). Tests are performed to locate any faults (Rob & Coronel, 2004).

The final phase of the waterfall method is the implementation phase. During this phase the developed system is installed. After the system has been installed it needs to be tested and debugged until the system are ready for the customer (Rob & Coronel 2004).

The diagram below is a depiction of the Waterfall model presented by Stair and Reynolds (2008).

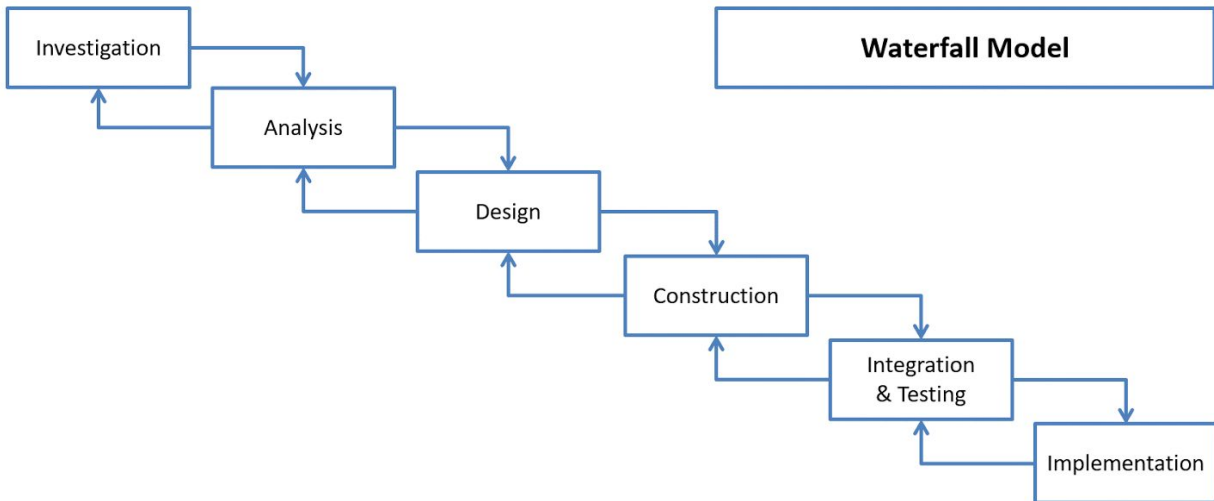


Diagram 1 Depiction of the Waterfall model (Stair & Reynolds 2008)

As can be seen in the diagram above, at the end of each stage a review is conducted to ensure that the requirements are met to a desired standard and quality. According to Sommerville (2011) once a phase has been completed it should be documented and evaluated. The next phase in the model should not be started until the current phase has been evaluated and approved (Sommerville 2011).



1. User Requirements Document

Introduction

This deliverable presents the user requirements document. This will describe the purpose, scope, definitions, acronyms, abbreviations and overview of the project. It also includes the system requirements that need to be focus on. The system requirements will contain the system perspective, requirements, user characteristics and operational environment.

1.1 Purpose

The proposed eyeRS mobile application (eyeRS app) is designed to allow users to catalogue their personal effects and beloved belongings while creating a platform to facilitate the trade of their items in the future. The eyeRS app is a Management Information System (MIS). A MIS is systems that use data to generate a report for the enduser to make routine decisions in response to a daily problem (Cress, 2017). In the case of the eyeRS app a summarised report is created for each item that the user can add to the system.

The purpose of the eyeRS app is to give individuals the power to manage and organise their personal and professional lives with less workforce. The eyeRS app achieves this control by allowing users to freely access and catalogue all of their belongings and items that are uploaded to the app. Upload can be don anywhere, on both an android mobile device. Each belonging is saved by means of photo identification and a short description.

The app allows users to be connected with friends and family. The connection is achieved by allowing users to share and trade items on their catalogues. The app also allows a user to be connected with their personal belongings anywhere.

1.2 Scope

The eyeRS mobile application consists of an inventory for household items. A user is able to upload these items onto the application via the built in camera or a local file source such as the gallery. A bid or buy system is implemented for users who wish to sell any unwanted items. All items that are put up for sale is available to all users that have downloaded the mobile application. Contact information is made available interested parties. Personal inventories can be shared with other users via social media WhatsApp or by means of Bluetooth connections.

Table 1 - Supported features

Supported Feature	Description
Single device support	The app will only allow users to make use of the app on a single device.
Tips & Help	<p>When installing the app, a user will be shown a video tutorial on the basic features of the app.</p> <p>The user will have the option to refer back to this video should they require to do so. Onscreen help is available to assist users with navigating between features.</p>
New category	The app allows user to create personal categories, which may differ from the prebuilt types.
Update existing data	The app allows users to make changes to existing items in their catalogue e.g. name changes.
Image capturing using built-in camera	The app makes use of a camera to capture images of new catalogue items.
Image gallery uploads	The app allows a retrieval of images from a user's personal gallery.
Bluetooth file share	The app allows users to share a QR code with another user for an item in their catalogue. Sharing can occur via a third-party messaging app or via Bluetooth.
File deletion	Users are able to delete existing category or

	<p>catalogue items.</p> <p>There must always be at least one category by default however.</p>
Item trade	The app allows users to trade their catalogue items with other users by sending them a link to the item on offer.
Add new item	The app allows users to add a new catalogue item, specifying its name, its description, and capturing an image of the item using the built-in-camera.
Third-party app interaction	The app can use other third-party apps to carry out certain actions within the bounds of the user-specified intent.

Table 2 - Unsupported features

Unsupported Feature	Description
Multi device support	Due to the project's time constraint, this feature is not be supported in the primary release. It may be available as a future update.
Revert changes	A user will not be able to revert back to their previous settings once changes have been made.
Video recording using built-in camera	Video recording will not be used in the app. The app will only allow the user to capture still images while using the camera feature.
NFC file share	NFC will not be utilised, as a sharing functionality is not loaded onto all mobile devices that support this technology.
File back-ups	The app will not be connected to any online database server, all data will be stored on the user's device.

Group messaging	The app will not allow a user to create a multi-user chat dialog.
-----------------	---

Benefits

1. This is a reliable cataloguing app that will enable its users to manage all their belongings in a visual format;
2. Users will have access to the app anytime of the day;
3. The app can be used while connected to the internet or in offline mode;
4. The app will consume minimal battery life while in offline mode;
5. The app will be available as a free download from the Google Play Store; and
6. Users will be able to share offers with other users for items they want to trade.

Goals

1. The eyeRS team intends developing a user-friendly app that will enable the user to manage their personal belongings in a visual format;
2. The app will allow the user to be able to access the app whether they have Internet access or not;
3. The app will target a wide range of users with different backgrounds regardless of their demographics, culture, location;
4. A prototype for the app will be designed for use on any computer aided software tool. The purpose of the prototype would be to get feedback from customers with regards to the look and feel of the proposed app before any development takes place;
5. The eyeRS development team will deliver a fully functional app that will be deployed on the Google Play Store;
6. The app will be available as a free download from the Google Play Store;
7. The user will be able to send app feedback to the development team in the form of ratings or written reviews; and
8. The user will be able to make suggestions of potential features to upgrade or fix (in case bugs are encountered).

1.3 Definitions, Acronyms and Abbreviations


Table 3 - Definitions used in the document

Definitions	
Term	Definition
Android OS	Is an operating system developed by Google design for mobiles cell phones to easily provide real actions as swiping, tapping and pinching to manipulate objects on screen (Rouse 2017a).
Catalog	A list or record of items systematically arranged and often including descriptive material.
Debugging	A process of detecting and removing errors and defects from a computer program (Hope, 2017).
Deliverable	Something that can be done and provided to a customer, especially something that is a realistic expectation (Martin & Tate, 2002).
Entity	A being or existence, a distinct, independent or self-contained object (Eastman, 1999).
Feedback	The process of returning part of the output of a device to the input, either to oppose the input (negative feedback) or to aid the input (positive feedback) (Coach 2015).
Functionality	A capability in which a device it is able to perform (Bass <i>et al.</i> , 2012).
Intent	With regards to Android development, an

	intent facilitates the performs a late runtime binding between different applications (Android, 2017).
Interface	Computer hardware or software designed to communicate information between hardware devices, between software programs, between devices and programs or between a device and a user (Sonmez, 2010).
Javadoc	A documentation generator created for generating API documentation in HTML format from Java source code.
Mobile Application	A portable computer program used for a particular type of job or purpose(Rouse 2017b).
Mobile Device	A portable, wireless computing device that is small enough to be used while held in the hand/handheld (Viswanathan 2017).
Prototype	The model on which a later product is based or formed (Rouse, 2017).
Waterfall Model	It is a progressive design process used in software development process where the process flows downwards through multiple phases (Rob & Coronel 2004).

Table 4 - Acronyms used in the document

Acronyms and Abbreviations	
Acronyms/Abbreviation	Expanded
3G	Third Generation
ADK	Android Development Kit
ADT	Android Development Tools
API	Application Programming Interface
App	Application
ARM	Advanced RISC (Reduced Instruction Set Computer) Machines
CPU	Central Processing Unit
DDMS	Dalvik Debug Monitor Server
DFD	Data Flow Diagram
DPI	Dots Per Inch
ERD	Entity Relationship Diagram
GPU	Graphics Processing Unit
HCI	Human Computer Interaction
HD	High Definition
HPROF	Heap/CPU Profiling Tool
HTML	Hyper Text Markup Language
IDE	integrated Development Environment



IO	Input/Output
JDK	Java SE (Special Edition) Development Kit
JRE	Java Runtime Environment
JSON	Javascript Object Notation
MIS	Management information systems
NFC	Near Field Communication
OS	Operating System
PIN	Personal Identification Number
RAM	Random Access Memory
SDK	Software Development Kit
UI	User Interface
USB	Universal Serial Bus



1.4 Overview

This section addresses system requirements. Included in system requirements is information users need to know about their software and hardware components in order to run the eyeRS app successfully.

The eyeRS app will be a conventional app when it comes to requirements, in which users will need to have any device that runs an Android OS. There are many apps on the market each one with its identifiers, look and functionality. eyeRS will help users organise their belongings (virtually) into categories, or to sell or trade and share items by using a single app.



2. System Requirements

2.1 System Perspective

eyeRS is relatively unique. There are not many applications that are used for the purpose of making an inventory of household items. Most inventory mobile applications, are mainly used for businesses to track stock, as well as provide sales and purchase information. Onsight is an Android app which is an existing product closest to the eyeRS system's intended functionality.

Another app such as Closet+, involves cataloguing a user's clothing, operating like a mobile closet, in which a user can keep track of their clothing (MY/STATIC/SELF, 2017). Closet+ is a automated style assistant developed for Apple's iPhone that is designed to aid users in organising, cataloging and planning their wardrobes developed by MY/STATIC/SELF (MY/STATIC/SELF, 2017). A user is able to organise clothing under a number of categories.

Onsight is a mobile sales application which corresponds with the eyeRS applications functionality of employing a means for selling items within the catalogue via the application (Onsightapp, 2014). Onsight provides a mobile catalog application designed for manufacturers, wholesalers and distributors to replace paper catalogues (Onsight, 2016). Onsight's website (2016) indicates that they support various mobile platforms, including Android devices, Windows tablets and Apple iPads. Onsight's main purpose is for a salesmen for them to provide potential clients with descriptions and images of products.

The factor that makes our application unique is a combination of allowing the user to catalogue items and have the option of putting them up for sale via the application.

2.2 Functional Requirements

Table 5 - Functional requirements

Identifier	Description	Source	Priority
301	A user must be able to search the app for a specific item in their catalogue.	eyeRS development team	High
302	The user must be able to add, remove or edit an existing catalogue item in a quick and simple fashion.	eyeRS development team	High
303	The app must allow a user to share information about a catalogue item via a third-party app externally (e.g. email, WhatsApp, etc.) as a link or an app generated QR code.	eyeRS development team	High
304	The app must allow a user to capture images of all the items in their catalogue.	eyeRS development team	High
305	A user must be able to browse through their catalogue items in the app.	eyeRS development team	High
306	The app must enable users to delete any existing items from their catalog. Once an item is deleted, changes cannot be reverted.	Client	High
307	The app must allow the user to manage and maintain existing items.	Client	High
308	The user must be able to lock the app with a pin code chosen by the user.	Client	High
201	The user must be able to search their catalogue using various search filters.	Client	Medium
202	A user must be able to receive help and tips on the app's features.	Client	Medium

203	The app must retain all user data and preferences once closed.	eyeRS development team	Medium
204	The app must have the capability to operate on any device that supports the Android OS.	Client	Medium
205	A user must be able to add descriptions for their catalog items.	eyeRS development team	Medium
206	The user must be able to view a history of the traded catalog items in the app.	Client	Medium
101	A user must be able to tailor the app to their requirements (i.e. adding, removing or renaming categories).	eyeRS development team	Low
102	The app must support both portrait and landscape views for all the screens.	eyeRS development team	Low
103	The app must be able to accept null values for certain fields when adding a new item to the catalog.	Client	Low

2.1.1 Context Diagram (high-level DFD)

The diagram below indicates the context diagram (high-level DFD) for the eyeRS mobile application.

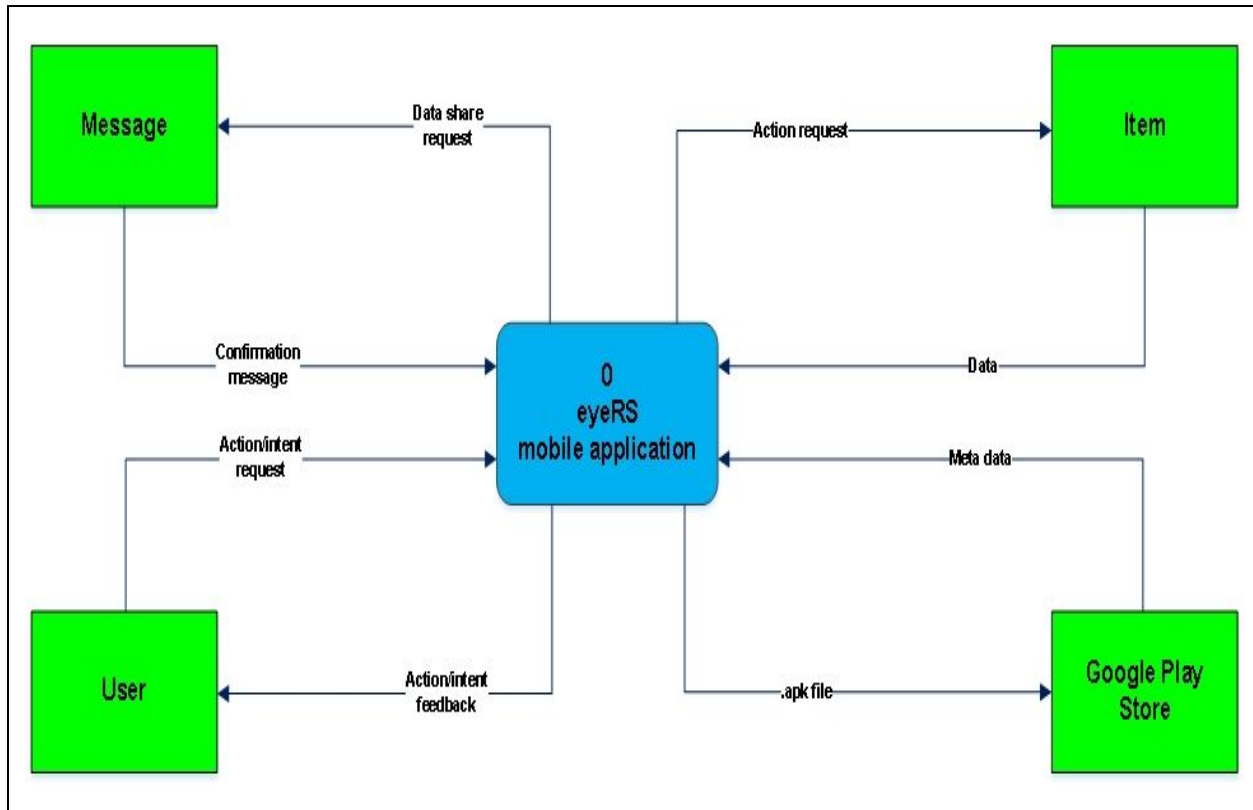


Diagram 2 - Context Diagram

Table 6 - Context DFD information

Entity	Data Flow from Entity	Data Flow to the Entity
User	Action/Intent request	Action/Intent feedback
Message	File share request	Confirmation message
Item	Data	Action request
Google Play Store	Meta-data	.apk file

2.1.2 Entity Relationship Diagram (ERD)

The ERD below does not include attributes and follows Crow's Foot format as presented by (Connolly & Begg 2005).

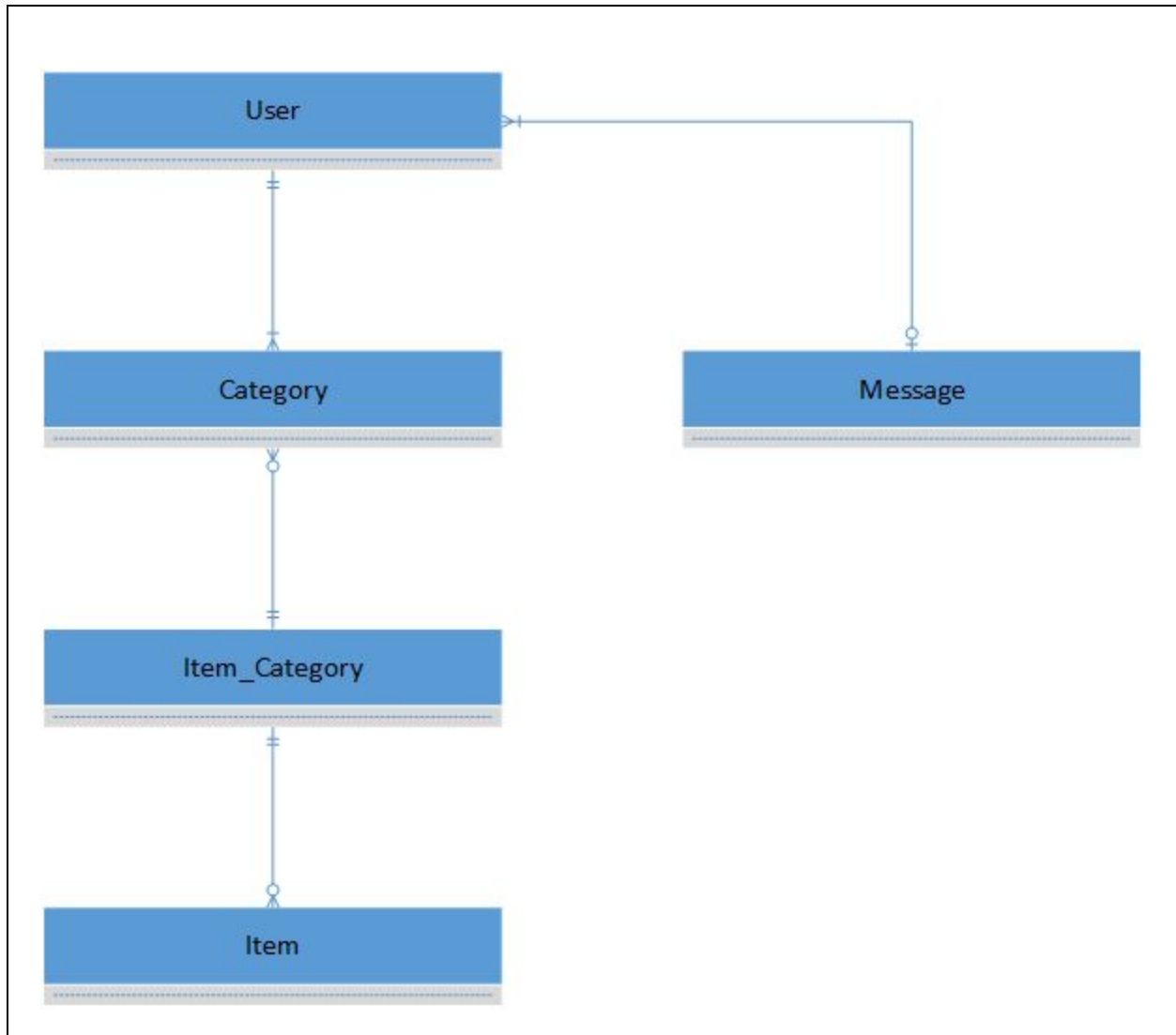


Diagram 3 - An ERD of the system



2.3 Non-functional Requirements

Human-Computer Interactions (User Interfaces)

The application should make use of Nielsen's usability heuristics and incorporate the following non-functional requirements:

1. Users should require no additional training in order to use the app;
2. The app must be accessible to all users;
3. The app should provide the user with productivity gains;
4. The app must not have a high error rate; and
5. The app must have accessible to all users.

Security Requirements

The app will be safe in terms of keeping the integrity of user's data safe. It will be secured by a login password or pin, depending on the user's choice. The storage platform is local which offers more control to the user and availability.

Communications Interfaces

The following non-functional requirements are applicable to the eyeRS mobile app communication interface:

1. The app must provide reliable interaction functions between the user and device.
2. There must be a simplicity factor related to the symbols that will be used in the eyeRS app;
3. The eyeRS app must provide an adaptable communication interface to the user;
4. The user interface must be understandable for the user to communicate to the app; and;
5. The eyeRS app must allow the user to modify the app to ease the communication with the app.



Hardware & Software Interfaces

Screen resolution and form factor considerations

A platform like the Android OS, which can run on many types of mobile and tablet devices, is likely to encounter various screen DPI issues (Android Developers, 2017). It is, however, important to keep a set of target screen resolutions while designing the user interface for the app (Android Developers, 2017). The following kinds of changes will be required for supporting different kinds of screen resolutions:

1. Font adjustment - The font size may be adjusted higher (for large DPI screens) or low (for small DPI screens) in order to ensure text remains readable;
2. Layout tweaks: The layout may need to be adjusted to increase or decrease the spacing between and around labels, and widgets shown on the screen. The ability to tweak will prevent information from becoming cluttered on high DPI screens or spaced apart too much on low DPI screens; and
3. Image changes - Background images or art may have to be provided in two different versions, namely a large size/high resolution version and a small size/low resolution version.

The app will need to utilize the OS and SDK mechanisms that have been provided to be able to cope with various screen size-ranges of devices operating on that specific OS (Android Developers, 2017).

Touch and non-touch screens: It is important to make sure that touch is activated for all the functionalities in it, when the app is running in touchscreen mode (Android Developers, 2017). Generally, API support will be present for the app to establish if the app is running on a touch-enabled screen or not (Android Developers, 2017).

Portrait and Landscape mode: Avoiding to add the support both portrait and landscape modes is necessary if the app will not require both modes to operate (Android Developers, 2017). However, if it turns out to be needed then it is vital to ensure that there is support for both modes on all the user interfaces in the app and not just a few of the interfaces in order to avoid users encountering a usability surprise (Android Developers, 2017).



CPU & Memory Characteristics

Android apps can operate on both a low-end Android phone with minimal amount of RAM available as well as a single core CPU. They can also operate on high-end devices with large amounts of RAM and even making use of a processor specified as an Octa-core (Android Developers, 2017). Should the app, however, be specified to require a specific minimum RAM and CPU power then it will, practically, fail on the low-end devices (Android Developers, 2017).

If the app uses extensive arithmetic and logic functions such as those involved in streaming and decompression of audio and video and in rich animations, or it lets the user view and manipulate large sets of information or images then the minimum CPU and memory requirements for the app will need to be specified in precisely the same way as with similar desktop apps (Android Developers, 2017).

In order to objectively determine the requirements, every feature in the app can be evaluated from the perspective of its respective memory and CPU usage through operating via profiling software tool and arriving at a low tier for the whole app from the assessment (Android Developers, 2017). For the eyeRS app, the following profiling tools can be used to determine the CPU and memory requirements:

ADT comes with the DDMS which can let you view a number of data relating to the operating state of the eyeRS app like heap and thread data, LogCat display and processing data etc (Android Developers, 2017).

Several 3rd-party apps can be found on the Play Store to assist in measuring and analyzing memory, CPU and the battery performance of the app.



Network Condition Scenarios

There are four factors to consider with regards to the network condition of a mobile app:

1. Supporting multiple network protocols - Devices can operate with the network on one or more protocols like WiFi, LTE etc (Date, 2015). Certain features in the app may not operate well (or not operate entirely) on certain protocols. Supporting only the elite protocols like WiFi & LTE will create a risk of excluding the low-range devices which may not function on these protocols (Date, 2015). However, required or recommended protocols need to be specified as failure to do so may lead to users unintentionally trying to use features which require high-bandwidth on lower-tier protocol like GPRS, hence, causing themselves to become frustrated from using the app. For features such as streaming media, it will be advisable for users to make use of high-range protocols like 4G or WiFi (Date, 2015). It is also important to advise the user if a specific feature will become unavailable on a certain channel. Users will also need to know whether any additional usage charges may be incurred by selecting to operate over specific channels such as 3G so as to avoid getting billed;
2. Signal drop/strength reduction - Each network-enabled feature needs to be assessed in the app in the situation where the protocol, over which it is operating, becomes unavailable or its signal strength reduces (Date, 2015). The feature has to be either network-fault tolerant or fails gracefully in such a situation;
3. Network protocol transition - This network condition handles the app's behaviour when the device moves from a particular protocol to another. An example would be where the user leaves the work building and the device moves from WiFi connectivity to 4G (Date 2015). If a process was taking place in the app at this moment in time, *how will the app handle such a scenario?* It is therefore important to assess the app's capability in such scenarios and design it for a seamless transition to the other protocol or for a graceful failure; and
4. Support for multiple protocols - This network condition relates to the app's behaviour when there are multiple network protocols active at a time (Date, 2015). The Android OS today automatically prefers WiFi when available, as to using cellular data (e.g. LTE). It is important to enable listening to a single or multiple network-based events when a network protocol gets active or inactive and also when the strength of the signal for the active protocol is altered (Date, 2015).





Battery-consumption Considerations

On devices, the battery is a rare and important element. There are 3 main causes of battery drain in devices:

1. The device's main processor;
2. The device's screen (including the GPU); and
3. The communications processors.

On a device, the battery is supposed to be wholly available to the phone app which is used for initiating and reception of phone calls. The eyeRS app may then become unpopular or even get uninstalled, if it consumes a lot of battery life during use. The idea of what is meant by 'a lot' may vary, due to the nature of the app and the hardware capabilities of the platform it is running on. A good way to evaluate the battery usage characteristics is to evaluate the app against a 'control' app which is usually a 'well-known app' in its class, in this instance *Closet+*. There are 3 general battery-power tests that should be carried out:

1. Regular use test - Starting with the battery on 100%, use the eyeRS app for six to twelve hours and measuring the battery level after each half or hour mark (Qian, 2015). A dynamic testing tool may be used to perform this test, in order to ensure that the test will run for the specified time interval (Qian, 2015). The test then reveals how fast the eyeRS app consumes battery life while in 'regular' use, and also, ensuring background and foreground features in the app are functioning as they should;
2. Idle-run test - The device should have its screen locked and power saving mode switched off (Qian, 2015). The battery life should be on full and then the app should be kept running on its main-home or dashboard view as specified, measuring the battery life at $\frac{1}{2}$ or 1 hour intervals. This test will measure the battery drain due to such things as intentional or unintentional automatic screen refreshes, and due to the background threads or services running in the app (Qian, 2015); and
3. Screen lock test - Performing an idle-run test again, although this time the device's screen should be locked (Qian, 2015). This enables testing the app to determine if it is using any network and, or, CPU resources (including the battery life) when the app cannot be viewed by the user (Qian, 2015).

In every instance, it may be useful to analyze the battery level against time so as to get a rapid visual depiction of the overall trend.

It is also important to ensure that, while carrying out the tests, no other app is operating on the device and to switch off the phone feature so as to ensure calls will not be received while the test is in progress (Qian, 2015). If possible, turning off apps like chat clients that usually run in the background will also be required.



Sensor Attributes

A lot of devices (including low quality phones) are equipped with an in-built camera, that could be specified as a particular sensor. Mobile devices & tablets usually have multiple types of sensors like GPS-location, gyroscope, ambient light & proximity and accelerometer sensors, for example. They are also capable of connecting to external sensors through Bluetooth or USB connectivity. While the eyeRS app may make use of a sensor to receive information, attention needs to be focused on the following attributes of the sensor:

1. The Maximum Sample rate;
2. The Operating range;
3. Sensitivity; and
4. Accuracy.

Media Functionalities

The device's OS and its hardware that the app operates on, will generally determine which audio and video capabilities the app will support. E.g, which audio/video formats the app supports; whether it will accommodate full HD, and if not, if it can facilitate multi-channel surround sound etc., which are all specified by the OS's capability and traits of the device's hardware. The following needs to be noted before a decision can be made as to which specific audio/video features are to be supported:

1. Details of the OS version that the app will operate on need to be known and whether it has support for any specific media features like surround sound that the app will need to support. The app needs to be targeted at versions only at and higher that version. In a number of cases, it will also be a requirement to provide programmatic access to these capabilities, hence, ensuring that SDK support is available to run the features through the app's code is just as important;
2. If no out-of-the-box support exists while the OS supports the feature via its SDK, then, it is important that there is enough support through 3rd-party libraries. If not then there is the risk of often incurring significant costs of issuing support by creating a unique software library; and
3. It is also important to ensure the mobile hardware targeted for the app has the hardware for running the features required.



Support for various versions of Android OS

Various mobile OSs are designed to allow them to be 'forward compatible'. The OS developers put in a lot of effort to ensure that as long as the app uses the official developer SDK in a prescribed manner by the SDK documentation, the app written for OS version 1.0, for example, will operate on version 2.0 etc.

Up to a certain extent, the forward compatibility stops and the app creator will be required to reconstruct the app for a higher version of the OS. Users generally update the OS on their devices often. It is therefore important to include forward compatibility within the app. If the app no longer operates after the user updates the OS to a better version, the user could decide to uninstall the app simply, and not be worried about getting the higher version of the app, although it may be available. Here are some of the considerations to ensure the app continues to operate on later OS versions:

1. Avoiding extensive use of 3rd party libraries;
2. Avoiding use of API's not recommended by Android;
3. Implementation of best practices while using APIs and avoiding non-standard usage of API methods;
4. Testing the app on all OS versions that are intended to be supported on;
5. If any features are OS specific, it is important to ensure they fail gracefully on prior OSs; and
6. Whenever the app starts, it is important to determine if a better version is available and urge the user to get the version. Usually, the Google Play Store should advise the user through a push-messages when a better version of the app is ready for download. It may be preferable to perform the version check each time the app starts. The user may fall back in the version of the app by what the latest available version is on the Google Play Store , therefore, it could be vital to urge the user to update the app prior to allowing them to use it.



Interrupts, notifications and multitasking

If a phone call, text messages or other types of notifications (e.g. calendar-reminders) occur, the device will probably inform the eyeRS app of the notification. Selecting to respond to the notification, the OS may then background the eyeRS app or, in the event of mobile OSs not capable of multi-task capability, they may just terminate the eyeRS app.

In either situation, the OS will most likely give the eyeRS app a chance to respond to the halt, termination or background event by issuing a handler method that should be implemented. It is necessary for the eyeRS app to handle the interrupt in such a way which will:

1. Not interfere with the OS's processing the user's choice to respond to the interrupt (like accepting a call or reading an SMS); and
2. Not lead to damages to the app's capability to operate as normal after the OS resumes the app after the user has finished dealing with the interrupt or if they decide to ignore it.

Each feature must be evaluated in the app from the perspective of determining how it could, and must operate if the app is pushed to the background by the OS, or made dormant while that feature is in execution-mode, and how it will recover from this interrupt situation after the OS or the user 'foregrounds' the app back after servicing the interrupt.



2.4 Technical Requirements

1. The app functionality must be developed using the Java programming language supporting a minimum JDK of version 7;
2. The app should support a minimum OS version of Android IceCreamSandwich (4.0.3) or higher (Google, 2017);
3. The mobile application should allow for future maintenance updates;
4. The deployment of the mobile application will require an active Google developer account;
5. The app should be developed in Android Studio IDE or other appropriate IDE;
6. The Android Studio SDK version used by the development team to construct the app is to meet the minimum requirement of version 2.2;
7. The app will have API documentation generated using javadoc;
8. Use of the SDK Manager will be required for testing the app on different API levels to target different OSs as well as installing, updating or uninstalling packages;
9. The development team will construct the app using machines that support a minimum JRE of version 7; and
10. To develop a prototype for the proposed end user to approve using Justinmind Prototyper or another equivalent prototyping tool.

2.5 User Characteristics

The users of the eyeRS mobile application should be one that requires an easy to use catalogue for day-to-day life. Just basically keeping track of household items and to improve on your home by upgrading, or getting rid of unwanted items. An application for users to make their house a place they can call home.

The users are not limited to a specific group. Whether they're novices or experts in handling a mobile device, bachelors / bachelorettes living on their own or with roommates or even families the application caters to all. According to (Matz 2013) there are a few preferences to be considered:

Table 9 - Preferences

Characteristic	Preference
Age	3+
Gender	Male/Female
Educational level (Level of experience)	A basic knowledge of operating mobile devices
Language	English
Computer skills	Novice; Technologically literate
Domain-related knowledge and skills	Interior design
Physical environment	Home, office, on the go
Social environment	Family members, friends, business



2.6 Operational Environment

The operational environment will consist of two environments. An environment for development and testing of the application and an environment for the prototype and final application. The eyeRS application will become available for download on android mobile devices.

The application development and testing will take place on a desktop with the Microsoft Windows XP or later versions of the operating system (tutorialspoint.com 2017). The workstations will require the following tools for development:

1. Java Runtime Environment (JRE) 6 or higher and Java Development Kit (JDK) 8 or higher;
2. Android Studio 2.2 or higher;
3. Android Software Development Kit Manager (SDK Manager); and
4. Android Virtual Device Manager (AVD Manager).

Android studio will be the main tool used for the app development and the Java language to will be used to develop eyeRS functionality.

The app will be operating on android mobile devices therefore the mobile devices will need to make use of a supported android version 4.0.3 (IceCreamSandwich) or higher.




3. Customer sign-off

Customer name and surname	Customer signature
Group leader name and surname	Group leader signature

Bibliography

- Android, Intent | Android Developers. Available at:
<https://developer.android.com/reference/android/content/Intent.html> [Accessed April 13, 2017].
- Best Practices for Performances | Android Developers. Available at:
https://developer.android.com/guide/practices/screens_support.html [Accessed April 13, 2017].
- Supporting Multiple Screens | Android Developers. Available at:
https://developer.android.com/guide/practices/screens_support.html [Accessed April 13, 2017].
- Bass, L., Clements, P. & Kazman, R., 2012. 4.2. Functionality | Understanding Quality Attributes in Software Architecture | InformIT. *Informit*. Available at:
<http://www.informit.com/articles/article.aspx?p=1959673&seqNum=2> [Accessed May 3, 2017].
- Coach, B., 2015. What is Feedback? | Definition of feedback in Communication. *Business Communication*. Available at:
<http://bizcommunicationcoach.com/what-is-feedback-definition-of-feedback-in-communication/> [Accessed May 3, 2017].
- Connolly, T.M. & Begg, C.E., 2005. *Database Systems: A Practical Approach to Design, Implementation, and Management*, Pearson Education.
- Cress, K., 2017. 5 Types of Information Systems. *MindMeister*. Available at:
<https://www.mindmeister.com/37310006?title=5-types-of-information-systems> [Accessed April 12, 2017].
- Date, S., 2015. *An Illustrated Guide to Mobile Technology*, CreateSpace.
- Eastman, C.M., 1999. *Building Product Models: Computer Environments, Supporting Design and Construction*, CRC Press.
- Google, 2017. Best Practices. *Android Developers*. Available at:
<https://developer.android.com/guide/practices/index.html> [Accessed February 13, 2017].
- Hope, C., 2017. What is debugging? *Computer Hope*. Available at:
<https://www.computerhope.com/jargon/d/debuggin.htm> [Accessed May 3, 2017].
- Martin, P. & Tate, K., 2002. *Getting Started in Project Management*, John Wiley & Sons.
- Matz, K., 2013. *Designing Usable Apps: An agile approach to User Experience Design*, Winchelsea Press (Winchelsea Systems Ltd.).
- MY/STATIC/SELF, 2017. Closet+ / The Swiss Army Knife of iOS Style Assistants. *Closet+*.

- 
- Available at: <http://closetapp.com> [Accessed April 6, 2017].
- Onsightapp, 2014. Android sales app - Onsight. *android-sales-app*. Available at: <https://www.onsightapp.com/android-sales-app> [Accessed April 6, 2017].
- Qian, H., 2015. *Extending the Battery Life of Mobile Device by Computation Offloading*,
- Rob, P. & Coronel, C., 2004. *Database systems: design, implementation, and management*, Course Technology Ptr.
- Rouse, M., 2017a. What is Android OS? - Definition from WhatIs.com. *SearchEnterpriseLinux*. Available at: <http://searchenterpriselinux.techtarget.com/definition/Android> [Accessed May 3, 2017].
- Rouse, M., 2017b. What is mobile app? - Definition from WhatIs.com. *WhatIs.com*. Available at: <http://whatis.techtarget.com/definition/mobile-app> [Accessed May 3, 2017].
- Rouse, M., 2017c. What is prototype? - Definition from WhatIs.com. *SearchManufacturingERP*. Available at: <http://searchmanufacturingerp.techtarget.com/definition/prototype> [Accessed May 3, 2017].
- Sommerville, I., 2011. *Software Engineering*, Pearson Higher Ed.
- Sonmez, J., 2010. Back to Basics: What is an Interface? *Simple Programmer*. Available at: <https://simpleprogrammer.com/2010/11/02/back-to-basics-what-is-an-interface/> [Accessed May 3, 2017].
- Stair, R.M. & Reynolds, G.W., 2008. *Principles of Information Systems: A Managerial Approach*,
- tutorialspoint.com, 2017. Android Environment Setup. *www.tutorialspoint.com*. Available at: https://www.tutorialspoint.com/android/android_environment_setup.htm [Accessed April 28, 2017].
- Viswanathan, P., 2017. What Is a Mobile Device? *Lifewire*. Available at: <https://www.lifewire.com/what-is-a-mobile-device-2373355> [Accessed May 3, 2017].