John Ugwu

Anna Carlin

Patrick Doherty

Jack Mahon

# Description

This project was an online gifting system. The aim of this project was to emulate a system based on the Arnott’s online store where members can sign up, create groups with friends, view items and create a wish list for their favourite items. This system can allow for the creation of multiple users. Users will have the option to add and remove for their wish list. To view these products, there is a Category page which has selected categories which displays the product, price, add to wish list option and will have a link to Arnott’s page for user to view in more detail or buy. In the system there will be a home page, profile page, group page and category page. The profile page can allow users to create hobbies and this will allow their group members to see their hobbies so therefore they can see what they are interested in if they are preparing to buy them a gift. This system intends to make life easier for people through the busy periods such as Christmas and will give them more of an idea on what their group members are interested in.

# Functional Requirements

1. User can register as a member – must

A user can enter their details to be stored in a database for future use of the website, so they can login and logout. They will be prompted to enter their first and last name, their email address and to create a new password. They will be asked to confirm their email address and password. They will be asked to enter their address and their telephone number also. All areas marked with an asterisk will be mandatory to fill in to complete their registration. Once a user has registered, they become a member.

1. Member can select hobbies – should

There will be a page displaying a list of hobbies. A member can select as many hobbies as they like. The hobbies selected by a member are stored in a database. These hobbies are displayed as interests when a member visits their profile page.

1. Generate a recommended wish list for the member – would

The system will generate a wish list that would be recommended to a member based on the hobbies that they selected.

1. Member can Login - must

When a member wants to log in to the system. They should be prompted for their username and password which they created when they were a user which was stored in the member database on completion of their registration to the system. A message below the log in will be displayed with a hyper link if the member has forgotten their username and password and this link will direct them to the next step.

1. Member can Logout – should

There will be a button in the corner of the navbar with the option for the member to logout from the system and return to user mode.

1. Member can create a wish list – must

A member must be able to create a wish list of any items they are interested in getting as a gift from other members of their group. These items are stored in an Items table in the database. The aim of the wish list is so that other members in any groups you are in can see what items you want to be gifted so that they know what to get for you.

1. Member can create a group - should

A member must be able to create a group with other members registered on the system in which they can view the wish lists of every other member of the group. The creator gives the group a name and selects members from dropdown boxes and then creates the group. A request is then sent to all of the members chosen for them to accept or reject.

1. Member can add people to a group – should

A member can add several people to a group, but they must be friends with the person on the website in order to send a group request. The person must then accept the group request to be added to the group or else reject the request.

1. Member can browse categories – should

A member can browse categories on the Categories page of the website in order to look through topics that interest you that you may want gifts in. Items in each category can be added to the wish list.

# Non-Functional Requirements

# Appropriate Process Model

An appropriate process model for the team project would have been the Agile model. This is an approach that encapsulates elements of both iterative and incremental development, meaning that increments are small and often and are built upon one after another. The Agile process model is also different from other process models in the sense that it embraces and encourages changing requirements throughout the project.

This would have worked very well for our project due to the fact that we had several instances when we had to add or change a requirement of the system even a number of weeks into development. The incremental and iterative approach to create the product was similar to what we did anyway but it would have been more structured if the Agile approach was implemented.

The Agile process model also promotes collaboration in the forms of meetings known as Scrums, period of work known as Sprints and useful software such as Kanban and websites like JIRA. Scrums are a group meeting in which each member individually tells the others what they have done since the last scrum, what stopped them and what they will be working on next. In this way, each member of the team knows what each other is doing and intending to do thus eliminating the issue that member may be doing the same thing at the same time and so that a member knows who to ask if they need results of some part of the project in order to do their part. A Sprint is a period of time

decided by the team, usually 2 weeks, where an epic (or topic) is decided on for the Sprint and items known as issues are added to the Sprint that are related to that topic. Each member then assigns themselves issues that they will work on over the course of the Sprint and anything not completed is sent to the backlog and can be added to another Sprint. This would have been beneficial to our team as there was some instances of members doing similar work to others at the same time.

# Change Control

Change control is a systematic approach to managing all changes made to a product or system. The purpose is to ensure that no unnecessary changes are made, that all changes are documented, that services are not unnecessarily disrupted and that resources are used efficiently.

**Changing Requirements**

Working in Agile-style two-week sprints helps manage changing requirements because it gains customer and stakeholder feedback on features sooner rather than later, it improves scope control because stakeholders can add new requirements, shift priorities, or rethink requirements on a feature or architectural level, and it gives project teams the room to take risks and innovate based on customer feedback without sacrificing too much time or budget because agile teams can pivot on requirements as needed.

**Changing Timelines**

Another term for changing timelines is scope creep. This is a project risk that must be controlled. Scope creep is really a problem caused by injecting new or unplanned work into the middle of an iteration. the requirements need to be clearly defined before the project starts, and if there is any change during the project then it must go through a change control process which involves the change control board

# Review of Original Approach

**Documentation**

What went well?

The overall documentation of the project went well such as the UML diagrams. Each member of the team was required to complete at least one UML diagram. As we had studied and completed UML diagrams in a previous module, we were all familiar with the layout of each so there were no issues in implementing them for this project. Everyone contributed equally to the writeup of the project which sped up the process and left time for error checking before submission.

What went wrong?

There was only one issue with regard the documentation. One of the team members did not fully understand how to implement their specific UML diagram and needed help from another team member.

What could be improved and how?

As a team we could have improved on our weekly journal write ups. This could have been done by having each member write their own view of how the week went and putting it together rather than leaving it to one person each week.

**Design**

What went well?

The design of the finished website was up to the standard as we had hoped. With only a few missing requirements that were not highest priority the look of the finished product was good. The design was consistent throughout. Whoever created a page would follow the same layout that was decided on by the entire team.

What went wrong?

There was confusion at the beginning of the project as to what the design would be like, but it was quickly resolved and the implementation of it commenced. The Wishlist page was not fully created as hoped.

What could be improved and how?

There was not much to be improved in this aspect as we can agree it was one of our strongest points of the finished project. There were a few requirements we did not get fully implemented such as the Wishlist page and to overcome this we could have planned our timing of each requirement better.

**Code**

What went well?

Each team member had to write code in some way whether it was front end or back end. Every team member was familiar with what each person was doing which made it easier to understand their own part.

What went wrong?

There was some code that did not work and even after asking for another team members help we could not figure out how to get it working so that slightly impacted on the finished project.

What could be improved and how?

In another module this semester we use a tool called Jira. Looking back this would have been very beneficial to use with regards the distribution of tasks and who is implementing what code.

**Requirements Delivered**

What went well?

There were enough functional requirements completed from the list at the beginning that resulted in a finished project that was presentable to the user. The ‘must’ functional requirements were implemented first to allow time for the ‘should’ and ‘would’ priority later.

What went wrong?

There was too much focus on a particular functional requirement at a time by everyone which might have slowed the process of implementing the project.

What could be improved and how?

Being able to complete more than one requirement at a time to leave time for some of the extra requirements, this could have been done by allocating a functional requirement to 2 people instead of all team members focusing on the one at a time.

**Requirements Delivered**

What went well?

Decisions were made quickly within the team. Everyone was properly assigned a task, and nobody was left wondering what to do. Communication worked well within the team for the duration of the project. Issues were discussed and taken care of by other team members if any arose. The main form of communication was meetings, we met twice a week to get up to date on each person's role to keep on the right track. We also used blackboard and GitHub to communicate

What went wrong?

An issue with the team cohesion is one person being assigned too many tasks and their workload was greater than the rest. Another issue is if a team member was not taking advantage of their skills and working on an aspect of the project that they are strong in.

What could be improved and how?

Through this year we have learned more about agile and how to apply that to group projects. Although we had good communication throughout, we could have applied a scrum through our weekly meetings and assigned scrum master for the week. With scrum it enables the team to be more in sync and encourages personal planning for the individual’s interpersonal skills and for a more self-organised team. We also did not use GitHub to its full potential for the project.

Conclusion

This concludes our documentation of our review of the team project implemented last year. There have been new things learned by the team members which will be applied to future projects such as the agile manifesto and Scrum meetings. The project itself turned out as expected with only a few requirements not implemented but the highest priority ones implemented properly. The module gave us a sense of what it is like to work as a part of a team for the delivery of a piece of software which has stood by us in other modules prior.